A highly detailed template model for dynamic optimization of farms

- FARMDYN -

W. Britz, B. Lengers, T. Kuhn and D. Schäfer

Institute for Food and Resource Economics, University Bonn

GHG

- Model Documentation -

Matching SVN version rev495, July 2015

Abstract:

The dynamic single farm model documented in here is the outcome of several research activities. Its first version (named DAIRYDYN) was developed in the context of a research project financed by the German Science Foundation focusing in marginal abatement costs of dairy farms in. It project contributed the overall concept and the highly detailed description of dairy farming and GHG accounting, while it had only a rudimentary module for arable cropping. That version of the model was used by GARBERT (2013) as the starting point to develop a version for pig farms, however with far less detail with regard to feeding options compared to cattle. GARBERT also developed a first phosphate accounting module. Activities in spring 2013 for a scientific paper (REMBLE et al. 2013) contributed a first version with arable crops differentiated by intensity level and tillage type, along with more detailed machinery module which also considered plot size and mechanisation level effect on costs and labour needs. Based on nitrogen response functions, nitrogen loss factors were differentiated for the different intensity and related yield levels. activities, the model was renamed to FARMDYN (farm dynamic). David Schäfer, then a master student, developed in 2014 a bio-gas module for the model which reflects the German renewable energy legislation.

FARMDYN presents a framework which allows, for a wide range of different farms found in Germany, simulating changes in the farm program under different boundary conditions such as prices or policy instruments e.g. relating to GHG abatement such as tradable permits or an emission tax. Given the complex interplay of farm management - such as e.g. adjustments of herd size, milk yield, feeding practise, crop shares and intensity of crop production, manure treatment – FARMDYN is implemented as a fully dynamic bio-economic simulation model template building on Mixed-Integer Programming. It is complemented by a Graphical User Interface to steer simulations and to exploit results.
Introduction

1 Introduction ................................................................................................................. 4

1.1 Methodology ........................................................................................................... 5
1.2 Tool concept ........................................................................................................... 5

2 The template model .................................................................................................. 7

2.1 Herd module ........................................................................................................... 8
  2.1.1 Cattle ............................................................................................................. 8
  2.1.2 Pigs ............................................................................................................... 12

2.2 Feeding module .................................................................................................... 12
  2.2.1 Cattle ............................................................................................................. 12
  2.2.2 Pigs ............................................................................................................... 15

2.3 Cropping, land and land use .................................................................................. 16
  2.3.1 Differentiation by soil, management intensity and tillage type ......................... 16
  2.3.2 Cropping activities in the model ...................................................................... 16
  2.3.3 Optional crop rotational module ....................................................................... 17

2.4 Labour .................................................................................................................... 19
  2.4.1 General concept ............................................................................................. 19
  2.4.2 Labour needs for farm branches ....................................................................... 20
  2.4.3 Working off-farm ............................................................................................ 20
  2.4.4 Labor needs for farm operations, working off-farm and management ............... 21
  2.4.5 Field working days ......................................................................................... 23

2.5 Stables ..................................................................................................................... 23

2.6 Other type of buildings ........................................................................................... 26

2.7 Farm machinery ...................................................................................................... 28
  2.7.1 Farm operations: machinery needs and related costs ......................................... 29
  2.7.2 Endogenous machine inventory ....................................................................... 32

2.8 Investments, their financing and cash flow definition ............................................ 33

2.9 Manure .................................................................................................................... 37
  2.9.1 Manure excretion ............................................................................................ 37
  2.9.2 Manure storage ............................................................................................... 38
  2.9.3 Manure application ......................................................................................... 40

2.10 Synthetic fertilizers ............................................................................................... 41

2.11 N and P2O5 accounting ......................................................................................... 42
  2.11.1 General concept ............................................................................................. 42
  2.11.2 Standard nutrient fate model .......................................................................... 43
  2.11.3 Detailed nutrient fate model by crop, month, soil depth and plot ................. 45
### Introduction

2.12 Biogas module ........................................................................................................... 48
  2.12.1 Biogas economic part ......................................................................................... 48
  2.12.2 Biogas inventory ................................................................................................. 49
  2.12.3 Production technology ....................................................................................... 50
  2.12.4 Restrictions related to the Renewable Energy Act ........................................... 53

3 Dynamic character of FARMDYN ............................................................................... 54
  3.1 Fully dynamic version .............................................................................................. 54
  3.2 Short run and comparative static version ................................................................. 55
  3.3 States of nature (SON) ............................................................................................ 55
  3.4 Objective function .................................................................................................... 56

4 GHG accounting .......................................................................................................... 57
  4.1 prodBased indicator ................................................................................................. 60
  4.2 actBased indicator ................................................................................................... 61
  4.3 genProdBased indicator .......................................................................................... 62
  4.4 NBased indicator ..................................................................................................... 63
  4.5 refInd ...................................................................................................................... 66
  4.6 Source specific accounting of emissions ............................................................... 67

5 Derivation of Marginal Abatement Costs (MAC) ......................................................... 69
  5.1 Normalization of MACs ......................................................................................... 70

6 The coefficient generator ............................................................................................ 72

7 Technical realization .................................................................................................... 74
  7.1 Overview .................................................................................................................. 74
  7.2 MIP solution strategy .............................................................................................. 74
    7.2.1 Fractional investments of machinery ................................................................. 76
    7.2.2 Heuristic reduction of binaries ......................................................................... 76
    7.2.3 Equations which support the MIP solution process ........................................... 77
    7.2.4 Priorities .......................................................................................................... 79
  7.3 Reporting .................................................................................................................. 80
  7.4 Systematic sensitivity analysis based on Design of Experiments .......................... 80

8 Graphical User Interface .............................................................................................. 93
  8.1 Model farm and scenario specifications ................................................................. 93
    8.1.1 Workstep and task selection ............................................................................ 93
    8.1.2 General settings ............................................................................................... 94
## 8.1.3 Farm settings ................................................................. 95
## 8.1.4 Animals........................................................................... 95
## 8.1.5 Cropping .......................................................................... 96
## 8.1.6 Biogas ............................................................................ 96
## 8.1.7 Output prices .................................................................... 96
## 8.1.8 Input prices ....................................................................... 96
## 8.1.9 Prices in experiments.......................................................... 97
## 8.1.10 Environmental impacts .................................................... 97
## 8.1.11 MACs ........................................................................... 97
## 8.1.12 Algorithm ....................................................................... 98
## 8.1.13 Debug and output options ................................................ 98

## 8.2 Visualizing and analysing results ........................................... 99

## 8.3 Using the exploitation tools for meta-modeling ....................... 100

## 9 Restrictions and further work suggested .................................. 104

References: .............................................................................. 105
1 Introduction

The dynamic single farm model documented in here is the outcome of several research activities. Its first version (named DAIRYDYN) was developed in the context of a research project financed by the German Science Foundation (DFG, Nr. HO3780/2-1) focusing on marginal abatement costs of dairy farms in comparison across different indicators for Green House Gases. Relating material and information on the project are available on the project related web-page: [http://www.ilr.uni-bonn.de/agpo/rsrch/dfg-ghgabat/dfgabat_e.htm](http://www.ilr.uni-bonn.de/agpo/rsrch/dfg-ghgabat/dfgabat_e.htm). That project contributed the overall concept and the highly detailed description of dairy farming and GHG accounting, while it had only a rudimentary module for arable cropping. It was – while improvements were going on – used for several peer reviewed papers (LENGERS and BRITZ 2012, LENGERS et al. 2013a, 2013b, LENGERS et al. 2014) and conference contributions (LENGERS and BRITZ 2011, LENGERS et al. 2013c).

That version of the model was used by GARBERT (2013) as the starting point to develop a module for pig farming, however with far less detail with regard to feeding options compared to cattle. GARBERT also developed a first phosphate accounting module.

Activities in spring 2013 for a scientific paper (REMBLE et al. 2013) contributed a first version with arable crops differentiated by intensity level and tillage type. Along with that came a more detailed machinery module which also considered plot size and mechanisation level effect on costs and labour needs. Based on nitrogen response functions, nitrogen loss factors were differentiated for the different intensity and related yield levels.

Activities in summer 2013 then moved to a soil pool approach for nutrient accounts, differentiated by month and soil depth layer while also introducing different soil types and three states of weather. In parallel, further information from farm planning books was integrated (e.g. available field working days depending on soil type and climate zone) and more crops and thus machinery was added. The GUI and reporting parts were also enhanced. As the model now also incorporates beside dairy production also other agricultural production activities, the model was renamed to FARMDYN (farm dynamic).

David Schäfer, then a master student, developed in 2014 a bio-gas module for the model which reflects the German renewable energy legislation.

The original version with its focus on milk and GHGs was developed as milk accounts for about one sixth of agricultural revenues in the EU, being economically the most important single agricultural product. Dairy farms also occupy an important share of the EU’s agricultural area. They are accordingly also important sources for environmental externalities such nutrient surpluses, ammonia and greenhouse gas (GHG) emissions or bio-diversity, but also contribute to the livelihood of rural areas. With regard to GHGs, dairy farming accounts for a great percentage of the worlds GHG emissions of CO$_2$, N$_2$O and CH$_4$ (FAO 2006, 2009), and is hence the most important single farm system with regard to GHG emissions. Given the envisaged rather dramatic reduction of GHGs, postulated by the recent climate agreement, it is therefore highly probable that agriculture, and consequently dairy farms, will be integrated in GHG abatement efforts. Any related policy instruments, be it a standard, a tax or tradable emission right, will require an indicator to define GHG emissions at farm level. Such an indicator sets up an accounting system, similar to tax accounting rules, which defines the amount of GHG emissions from observable attributes of the farm such as the herd size, milk yield, stable system, cropping pattern, soil type or climate. The interplay of the specific GHG accounting system and the
policy instrument will determine how the farm will react to the policy instruments and thus impact its abatement costs, but also the measurement and control costs of society for implementing the policy. The objective of the paper is to describe the core of a tool to support the design of efficient indicator by determining private and social costs of GHG abatement under different GHG indicators. It is based on a highly detailed farm specific model, able to derive abatement and marginal abatement cost curves with relation to different farm characteristics, a highly detailed list of GHG abatement options and for different designed emission indicators.

That documentation is organized as follows. Following the introduction, we will discuss the methodology – the overall concept of the tool, the details of the template model. The third section discusses the dynamic examination of the modelling approach. Afterwards the different GHG accounting schemes are explained to offer their differences in calculation. Section five describes the core of the simulation program, the procedure the marginal abatements costs are calculated with. Also the normalization procedure for the indicator specific MACs is explained to make occurring MAC curves comparable. In the following sections the coefficient generator, the technical implementation and the graphical user interface (GUI) are explained which help the user to define experiments and visualize or analyze the results.

For more information or access to unpublished technical papers of Britz and Lengers please feel free to contact:

Wolfgang Britz, Dr., Institute of Food and Resource Economics, University of Bonn, wolfgang.britz@ilr.uni-bonn.de

1.1 Methodology

The core of the simulation framework consists of a detailed fully dynamic mixed integer optimization (MIP) model. The linear program maximizes an economic objective under constraints which describe (1) the production feasibility set of the farm with detailed bio-physical interactions, (2) maximal willingness to work of the family members for working on and off farms, (3) liquidity constraints, and (4) environmental restrictions.

Using MIP allows depicting the non-divisibility of investment and labour use decisions. Explanations of mixed integer programming models and their theoretical concepts are given by Nemhauser and Wolsey (1999), Pochet and Wolsey (2006) in detail. The fully dynamic character allows finding simultaneously an investment strategy and a future farm plan which maximizes an economic objective over the whole period.

1.2 Tool concept

The aim of FARMDYN is to develop a framework which allows, for a wide range of different farms found in Germany, simulating changes in the farm program under different boundary conditions such as prices or policy instruments e.g. relating to GHG abatement such as tradable permits or an emission tax. Given the complex interplay of farm management - such as e.g. adjustments of herd size, milk yield, feeding practise, crop shares and intensity of crop production, manure treatment - we develop a fully dynamic bio-economic simulation model template building on Mixed-Integer Programming.

In its current version, the model assumes a fully rational and fully informed farmer optimizing the net present value of the farm operation plus earnings from working off farm. A rich set of constraints describe in detail the relations between the farmer’s decision
variables in financial and physical terms and his production possibility set arising e.g. from
the firm’s initial endowment of primary factors. These constraints also cover different
relevant environmental externalities. Its dynamic approach over several years has clearly
advantages for the derivation of MACs, a point also stated by KESICKI and STRACHAN
(2011:p. 1202). But more generally, it gives insides on the impact of sunk costs and other
path dependencies for the development of farms.

The application of a mixed integer programming approach allows considering non-
divisibility of labour use and investment decisions. Neglecting that aspect has at least two
serious dis-advantages. Firstly, economies of scale are typically not correctly depicted as
e.g. fractions of large-scale machinery or stables will be bought in a standard LP. That will
tend to underestimate production costs. Secondly, using fractions increases the production
feasibility set which again will tend to increase profits and decrease costs. And as noted
already above, the combination of MIP and a fully dynamic approach allows capturing as
well implicitly sunk cost and path dependencies.

Conceptually, the model is a microeconomic supply side model for “bottom-up” analysis
based on a programming approach, i.e. constrained optimization. A bottom-up approach
principally connects sub-models or modules of a more complex system to create a total
simulation model, which increases the complexity but hopefully also the realism (DAVIS,
1993). On farm bio-economic processes are described in a highly disaggregated way.
Whereas so-called engineering models also optimise farm level production systems, in that
type of model possible changes in management or e.g. GHG mitigation options are
predefined (different feed rations, defined N intensities...), implemented separately and
ordered concerning their derived single measure mitigation costs to explore the MAC
curve. Contrary to that, the LP-approach of our Supply Side model enables to solve for
optimal adjustments of production processes by continuous variation of decision variables,
such that the optimal combination of mitigation measures is derived. The same argument
holds for analyzing shocks in prices or other type of policy instruments. With regard to
GHGs, a further advantage of the approach relates to interaction effects between measures
with regard to externalities as different gases and emission sources along with their
interactions are depicted (see e.g. VERMONT and DeCARA, 2010). Market prices are
exogenous in supply side models such that market feedback is neglected, contrary to so
called equilibrium models which target regional or sector wide analyses (such as e.g. the
ASMGHG model used by SCHNEIDER and MCCARL 2006).
The template model

2 The template model

An economic template model uses a declarative approach which depicts in generic terms the physical and financial relations in the system to analyze, based on a set of decision variables, exogenous parameters and equations describing their relations. Template models in that sense have a long-standing tradition in economics. In macro-applications, template based computable general equilibrium model such as GTAP (HERTEL 1997) or the IFPRI\(^1\) CGE_template (LOFGREN et al. 2002) models are quite common. For regional and farm type applications, programming model templates are underlying e.g. the regional or farm type model in CAPRI (BRITZ & WITZKE 2008) or the bio-economic typical farm type models in FFSIM (LOUHICHI et al. 2010). The aim of a template model is to differentiate clearly between structural elements which are common to any instance of the system analyzed and attributes of a specific instance. A specific instance of a farm would capture those attributes which are specific e.g. to location, firm and time point or period analyzed, including attributes of the farmer (‘s family) such as his management abilities and preferences.

A template model can be coded and documented independently from a specific instance. It also features clearly defined inputs and outputs so that generic interfaces to other modules can be developed. These modules could e.g. deliver the necessary inputs to generate instances or to use the template model’s results as inputs e.g. for reporting purposes or systematic analysis.

For our purposes, a suitable template must be able to generate instances representing farms characterized by differing initial conditions and further attributes specific to the firm and farmer. Initial conditions are e.g. the existing cow herds, its genetic potential, available family labour, existing stable places and their age, existing machinery and its age, land owned and rented by the farm or its equity. Further attributes could describe the firm’s market environment such as input and output prices, yield potentials, household expenditures etc. and the willingness of the farmer and further family members to work off-farm.

Farming, especially dairy farming is characterized by long lasting, relatively expensive stationary capital stock especially in form of stables and related equipment. High sunk costs related to past investment can lead to sticky farm programs, as key management possibilities such as the reducing the herd size lead to modest saving of variable costs compared to the loss of revenues. The strategies of farms as a response to changes in their market and policy environment such as a GHG emission ceiling are hence path dependent on past investment decisions. Whereas all farms can implement certain short term adjustments regarding to herd-, feed- or fertilizer-management, investment based strategies are not very probable for farms which invested recently in new buildings or expensive machinery. These characteristics mean that both for individual farms but also the industry as a whole, optimal short and long term abatement strategies and in case of GHG related abatement policies or other changes in their policy and market environment might differ considerably.

Accordingly, a framework is needed which covers a longer planning period to capture (re-)investment decisions and their impact on the farm program and externalities such as GHG

\(^1\) International Food Policy Research Institute
The template model

emissions on the long term. The following diagram depicts the basic structure of the template model with different module interactions.

Figure 1. Overview on template model, note: biogas module missing
Own illustration

We use in the following the actual GAMS code of the equations to document the different modules, to avoid a second layer of mnemonics. The following naming conventions are used in the code, and consequently also in the documentation. All decision variables of the farmers start with a “v_“. Technically, they are endogenous to the simultaneous solution of all equations when maximizing the objective function and hence depend on each other. Exogenous parameters start with a “p_“. They can typically be changed in an experiment. Sets, i.e. collection of index elements do not carry a specific prefix.

The model equations are defined in “modelNempl.gms”, declarations of parameters and sets also used outside of the model equations can be found in “modelNempl_decl.gms”.

2.1 Herd module

2.1.1 Cattle

The herd module describes the relation between different cattle types (dairy cows, mother cows, male and female calves, heifers, young bulls) on the farm in a dynamic perspective, with an annual resolution. The heifers process, starting with a female calf raised for one year is available in three intensity levels, leading to different process lengths (12, 21, 27 month) and thus first calving ages (12, 33 and 40 months) for the remonte. Newborn calves can be sold immediately, or after one year, or being raised to a heifer respectively young bull.

We differentiate all herds by age, gender, breeds and production objectives and month in each year, and females optionally by their genetic potential regarding the milk yield in case of milk breeds.
The template model

The model uses two different variables to describe the herd: \( v_{\text{herdStart}} \) describes the animals entering the production process at a certain time, while \( v_{\text{herdSize}} \) describe the number of animals of that type currently on farm.

The number of new calves \( v_{\text{herdStart}} \) of calves, differentiated by gender and breed, in a year \( t \) and specific month \( m \) depend on the herd size of cows of that breed, and the specific calving coefficients:

\[
\begin{align*}
&\text{--- definition of calves born} \\
&\text{newCalves}(\text{calves, breeds, t, m}) \triangleq \text{actHerd}(\text{calves, breeds, t, m}) \ldots \\
&\text{--- new born calves are born from the current herd of cows} \\
&v_{\text{herdStart}}(\text{calves, breeds, t, m}) \triangleq \text{samem}(\text{calves, calves}) \\
&v_{\text{herdStart}}(\text{calvesSold}, \text{breeds, t, m}) \triangleq \text{samem}(\text{calves, calves}) \\
&v_{\text{herdStart}}(\text{calvesRoi}, \text{breeds, t, m}) \triangleq \text{samem}(\text{calves, calves}) \\
&\text{-- sum(actHerd(cows, breeds, t, m))} \\
&v_{\text{herdSize}}(\text{cows, breeds, t, m}) = p_{\text{calbCoeff}}(\text{cows, breeds, calves, n});
\end{align*}
\]

\( t_{\text{Cur}} \) defines the years for which the model instance is set-up, and \( \text{actHerd} \) is a flag set to define which herds might enter the solution for a specific year. The calving coefficients take into account different breed specific parameters (see coeffgen/ini_herds.gms):

\[
\begin{array}{cccc}
\text{breed} & \text{birthPerLact} & \text{livingCalvesPerBirth} & \text{calvLosses} & \text{daysBetweenBirths} \\
RF & 0.95 & 1.04 & 0.08 & 417 \\
Simmental & 0.95 & 1.04 & 0.04 & 395 \\
mc & 0.98 & 1.04 & 0.04 & 325 \\
\end{array}
\]

Note: \( mc \) are mother cows, sales prices for animals are assumed to be equal to the Simmental breed.

\[
\begin{align*}
&\text{--- calving coefficients for females} \\
p_{\text{calbCoeff}}(\text{cows, curBreeds, calves, t, herdm}) \triangleq \text{sum}(t, \text{herd}(\text{cows, curBreeds, t, herdm})) \\
&= p_{\text{livingCalvesPerYear}}(\text{curBreeds}) = 0.495 \times p_{\text{birthDist}}(m); \\
&\text{--- calving coefficients for males} \\
p_{\text{calbCoeff}}(\text{cows, curBreeds, calves, t, herdm}) \triangleq \text{sum}(t, \text{herd}(\text{cows, curBreeds, t, herdm})) \\
&= p_{\text{livingCalvesPerYear}}(\text{curBreeds}) = 0.505 \times p_{\text{birthDist}}(m); \\
\end{align*}
\]

The standing herd of any type of animal \( v_{\text{herdSize}} \) is equal to the number of animals which entered the herd \( v_{\text{herdStart}} \) from the current month backwards to the given production length of the process. The one exemption, as seen below, are cows which can also be slaughtered before reaching the normal number of lactations.
The template model

$$ \sum_{(t,m)} \left\{ \begin{array}{l}
\text{actHerd(herds, breeds, t,m)} \\
\text{v_herdStart(herds, breeds, t,m)}
\end{array} \right\}$$

The parameter \( p_mDist \) describes the difference in months between two time points defined by year \( t, t1 \) and month \( m, m_1 \).

The definition of the number of animals being added to the herd, \( v_{\text{herdStart}} \), is described in the following equation below \( \text{herdBal}_{t, m} \). In the simplest case, where a 1:1 relation between a delivery and a use process exists, the number of new animals entering the use process \( \text{balherds} \) is equal to the number of new animals of the delivery process \( \text{herds} \). That relation between is depicted by the \( \text{herds}_{\text{from_herds}} \) set.

One possible extension is that animals entering the herd can be alternatively bought from the market, defined by the \( \text{bought_to_herds} \) set. The symmetric case is when the raised/fattened animals can be sold, described by the \( \text{sold_from_herds} \) set.

The case where several delivering processes are available, e.g. heifers of a different process length, the \( \text{herds}_{\text{from_herds}} \) set describes a 1:n relation. A similar case exists if one type of animal, say female calves raised, can be used for different processes such that the expression turns into a n:1 expression. That case is captured by second additive expression in the equation.
The template model

In order to allow for an increase of the genetic potential of herd, two mechanisms are available. If the farmer is allowed to buy heifers from the market, the replacement heifers can have a higher milk yield than the replaced cow; prices for heifers depend in their milk yield potential. The other mechanism is to systematically breed towards higher milk yields. However, the breeding process is restricted, which is depicted by the following equation, which restricts the increase to about 200 kg per year:

\[
\text{In comparative-static mode (p\text{\_compStatHerd}) all lags are removed such that a steady-state herd model is described.}
\]

Most equations - such as those relating to stable places needs - abstract from differentiation by genetic potential. Therefore, the individual herds are also aggregated to summary herds:

\[
\text{These equations also provide an easier overview on model results if the model listing is directly used. The following graphic illustrates approximatively the decision points that are simulated by the above described herd module.}
\]

![Figure 2. Herd management decisions (note: males and differentiation of heifers by producton length not yet covered)](Own illustration)
The template model

2.1.2 Pigs

A similar, but simpler module is available for pigs. In opposite to dairy farms for cows, it is assumed that sows leaving the herd are replaced by young sows bought from the market. The farm can sell piglets and use it for fattening (if fattners are allowed). Fattners can be produced from bought piglets or raised on (if sows are allowed). The only additional equations relates to the number of piglets born:

```plaintext
* --- piglets born: herd size of sows times piglets per sow and year
* newPiglets(tCur(t), herd) $ actHerdS("sows","",t,herd) ...
  - v_herdStart("youngPiglets","",t,herd) $= v_herdSize("sows","",t,herd) * p_0Coeff("sows","youngPiglets",t)/card(herdHrd);
```

The other equations are shared with the dairy herd (herdsBal_, herdSize_), by using other sets:

```plaintext
$ifthen1 $pigHrd% == true
  * --- pig herd sizes
  * option kill=v_herdSize.up;
$ifthen1 $fattners "$farmBranchFattners%" == "on"
  actHerdS("fattners","",tCur,m) $= YES;
  actHerdS("pigletsGrown","",tCur,m) $= YES;
  bought_to_herdS("pigletsBought","fattners") $= YES;
  option kill=sold_from_herdS;
  option kill=herds_from_herdS;
$endif $fattners

$ifthen1 $sows "$farmBranchSow%" == "on"
  herds_from_herdS("piglets","youngPiglets") $= YES;
  bought_to_herdS("youngSows","sows") $= YES;
  sold_from_herdS("pigletsSold","piglets") $= YES;
  actHerdS("piglets","",tCur,m) $= YES;
  actHerdS("cows","",tCur,m) $= YES;
  actHerdS("youngSows","",tCur,m) $= YES;
  actHerdS("pigletsSold","",tCur,m) $={ (tCur.pos gt 1) or (m.pos gt 1) } $= YES;
  actHerdS("youngCows","",tCur,m) $= YES;
$endif $sows
```

2.2 Feeding module

2.2.1 Cattle

The feeding module consists of two major elements:

1. **Requirement functions** and related constraints in the model template

2. **Feeding activities**, which ensure that requirements are covered, and link the animal to the cropping sector and the purchases of concentrates

The requirements are defined in “coeffgen\requ.gms”. Requirements for dairy cows are differentiated by annuals milk yield and by lactation period. The model differentiates 5 lactations period with different length (30 – 70 – 100 – 105 – 60 days, where the last 60 days are the dry period). The periods are labelled according to their last day, e.g. LC200 is the period from the 101st day ending after 200 days, LC305 is the period from the 201st to the 305th day and dry denotes the last 60 days of lactation.
**Excursus**

This excursus describes the derivation of the output coefficients for each lactation phase = How much of yearly milk yield is produced by each cow on one day:

![Lactation curves of different yearly milk yield potentials and average milk yield in different lactation phases](image)

Using the above shown lactation functions, the daily fraction of the yearly milk yield in each lactation phase can be derived. On average over the four milk yield potentials, the derived coefficients are shown in the following table:

<table>
<thead>
<tr>
<th></th>
<th>LC30</th>
<th>LC100</th>
<th>LC200</th>
<th>LC305</th>
<th>Dry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daily fraction</td>
<td>0.00356</td>
<td>0.00433</td>
<td>0.00333</td>
<td>0.00233</td>
<td>0</td>
</tr>
</tbody>
</table>

Following these outputs, e.g. on each of the first 30 days of lactation, the cow produces 0.356% of the yearly milk yield (e.g. 28kg per day for a 8000kg cow). These coefficients are now used to calculate the sum of milk output in each lactation phase to be able to calculate feed requirements stemming from the herds in each phase.

**Excursus end**

The daily milk yield in each period is based on the following calculations where the milk yield is defined in t/year and stored on a general output coefficient parameter $p_{OCoeff}$.
The template model

The resulting coefficients are then scaled to match the total yearly milk yield. The model currently differentiates for each herd between energy in NEL, raw protein and maximum dry matter requirements. For heifers and calves, there is currently no differentiation of different feeding phases so that requirements are identical each day.

The distribution of the requirements for cows in specific lactations periods \( p_{\text{reqsPhase}} \) over the months \( m \) depends on the monthly distribution of the births \( p_{\text{birthDist}} \):

\[
\text{p\_milkPerDay(cows,}”\text{LC3Q}"\text{)} = 0.003555556 \times \text{sun(t } (t.\text{pos eq } 1), p.\text{Coeff(cows,”milk”,t)}=1000); \\
\text{p\_milkPerDay(cows,}”\text{LC10}"\text{)} = 0.004393333 \times \text{sun(t } (t.\text{pos eq } 1), p.\text{Coeff(cows,”milk”,t)}=1000); \\
\text{p\_milkPerDay(cows,}”\text{LC20}"\text{)} = 0.005333333 \times \text{sun(t } (t.\text{pos eq } 1), p.\text{Coeff(cows,”milk”,t)}=1000); \\
\text{p\_milkPerDay(cows,}”\text{LC3Q}"\text{)} = 0.006333333 \times \text{sun(t } (t.\text{pos eq } 1), p.\text{Coeff(cows,”milk”,t)}=1000);
\]

In order to test different model configuration and to reduce the number of equations and variables the model, the monthly requirements \( p_{\text{Monthly}} \) are aggregated to intra-annual planning periods \( \text{intrYPer} \) for which a different feed mix can be used for each type of herd:

\[
\text{p\_reqs(cows, phase,}m, \text{reqAll)} = \text{p\_reqsPhase(cows,}”\text{LC3Q}, \text{reqAll)} \times \text{p\_birthDist(m=12/10);}
\]

These requirements per planning period \( p_{\text{reqs}} \) enter the equation structure of the model. The equations are differentiated by herd, year, planning period and state-of-nature, and the related requirements must be covered by a specific endogenous feed mix made out of different feeding stuff (currently grass and maize silage and grass from pasture, which are own produced, and three type of concentrates). A herd consists of cows of the different milk yield potentials, heifers and different type of calves. Depending on the distribution of calving dates in the cow herd, cows of the same milk yield potential can be in different lactation phases during the year which impact total feed requirements on the farm in the different planning intra-yearly periods.
The template model

The requirements and amount of tons fed $v_{feeding}$ are hence differentiated by herd, breed, planning period (lactation phase of cow), state-of-nature and year:

- requirement constraints (per herd, year and SDN)

\[
\sum(t) \{ (t \in \text{pos pose} \geq \text{t pos}) \} p_{\text{reqs}}(\text{passHerds, breeds, phase, intrYPer, reqs} \geq \text{t cur}(t)),
\]

- herd size time requirements per head, minus year and SDN specific reduction in milk yield

\[
\sum(t) \{ (t \in \text{pos pose} \geq \text{t pos}) \} e_{\text{redMlk}}(\text{passHerds, breeds, phase, intrYPer, t, s}) \geq \text{reqs} \text{reqs} \geq \text{t cur}(t),
\]

- energy and protein requirement for specific amount of milk and reduces milk production of the farm accordingly.

The feeding amounts are aggregated to total feed use of a specific product $v_{feeduse}$ for each year, feed and planning period:

\[
\sum(t) \{ (t \in \text{Feeduse}(\text{feeduse}, \text{intrYPer}) \geq \text{FeedsVt, s}) \}
\]

For own produced feed which are not stored and show a variable availability over the year such as grass from pasture, an aggregation to the intra-year periods takes place

\[
\sum(t) \{ (t \in \text{Feeduse}(\text{feeduse}, \text{intrYPer}) \geq \text{FeedsVt, s}) \}
\]

2.2.2 Pigs

The feeding module for the pigs currently works with fixed input requirements for 3 types of concentrates (concFeeds) and cereals:

\[
\sum(\text{conCPigs, intrYPer, t}) \\{ (t \in \text{Feeduse}(\text{feeduse}, \text{intrYPer}) \geq \text{FeedsVt, s}) \}
\]

The related feeding needs are defined in “coeffgen\pigs.gms”:
2.3 Cropping, land and land use

2.3.1 Differentiation by soil, management intensity and tillage type

Crop activities are differentiated by crop *crops*, soil types *soil*, management intensity *intens*, tillage type *till*. The use of different management intensities and tillage types is optional. Management intensities change the yield level:

and thus also crop nutrient needs. Necessary field operations and thus variable costs, machinery and labour needs are adjusted as well, see under machinery.

2.3.2 Cropping activities in the model

Crop activities are defined with a yearly resolution and can be adjusted to the state of nature. The firm is assumed to be able to adjust on a yearly basis its land use to a specific state of nature as long as the labour, machinery and further restrictions allow for it. Land is differentiated between arable and permanent grass land (*landType*), the latter is not suitable for arable cropping. Land use decisions can be restricted by maximal rotational shares for the individual crops. The total land endowment per landtype of the firm is equal to the initial endowment (*p_iniLand*) and land bought (*v_buyLand*) in the past or current year:

The total land by type can be either used for cropping (*v_croppedLand*) or rented out (*v_rentOutLand*), currently on a yearly basis:

Where total cropped land is defined from the land occupied by the different crops (*v_cropha*) as:

```
The \textit{c_s_t_i} set defines the active combination of crops, soil type, tillage type and management intensity.

The maximum rotational shares $p_{\text{maxRotShare}}$ enter the following equation which is only active if no crop rotations are used (see next section):

\begin{verbatim}
* * --- crop rotation constraints: each crop can occupy a max. share on cropped land
* cropRot_\{\text{landType},crops,tCur(t),s\}  $\{ (p_{\text{maxRotShare}}(crops) ne 1) \\
and crops_t\_\text{landType}(crops,landType) \} \\
  u_{\text{cropHa}}(crops,t,s) = u_{\text{croppedLand}}(\text{landType},t,s) \times p_{\text{maxRotShare}}(crops);
\end{verbatim}

Currently, a simple equation ensures that the farm stays under maximum stocking rate ceiling expressed in livestock units per ha:

\begin{verbatim}
* * --- Maximum stocking rate allowed
* * (ensures that a certain amount of land is present)
* l\_\text{land}(tCur(t), .. \\
  \sum(\text{landType}, \ u_{\text{totLand}}(\text{landType},t)-u_{\text{rentOutLand}}(\text{landType},t)) \times p_{\text{maxStockRate}} \geq \\
  \sum(\text{actHerds}(\text{herds},t)), \ u_{\text{herdSize}}(\text{herds},t) \times p_{\text{lu}}(\text{herds});
\end{verbatim}

### 2.3.3 Optional crop rotational module

The model can alternatively to the use of maximal rotational shares also be driven by three year crop rotations. The rotation names (shown below list, see \textit{modelNempl\_decl.gms}, set \textit{rot}, show the order of the crops in the rotations. There are always three which are identical from an agronomic point of view, edited on the same line, differing only by which crop they started originally. That avoids unnecessary rigidities in the model.

\begin{verbatim}
* * --- WC: winter cereals, SC: summer cereals, PO: potatoes
* * SB: sugar beet, IO: idleing land, OT: other
* * set rot "Rotations" / WC,PO,SC,IO,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,WC,PO,SC,IO,
\end{verbatim}

The rotations are linked to group of crops in the first, second and third year of the rotation as follows (only cross-set definitions \textit{rot\_cropTypes} for the first rotation are shown):
The template model

```gams
set rot_croptypes(rot,cropTypes,cropTypes,cropTypes)  "Rotation, first / second / third year crop type"

WC_P_O.winterCere_winterCere_potatoes
WC_P_O.winterCere_potatoes.winterCere
P_O_P_O.winterCere.winterCere

set cropTypes_crops(cropTypes,crops) / winterCere.,winterCere.,summerCere.,maizeCorn.,maizeDCH.,wheatOPS.
other.(winterrape,,sunnerBeans,,summerPeas)
potatoes,potatoes
sugarbeet,sugarbeet
idle,idle

set cropType_0(rot,cropType), cropType_1(rot,cropType), cropType_2(rot,cropType), cropType_3(rot,cropType) $ sum(rot_croptypes(rot,cropTypes,cropTypes,cropTypes),1) = YES;
set cropType_1(rot,cropType), cropType_2(rot,cropType), cropType_3(rot,cropType), cropType_4(rot,cropType) $ sum(rot_croptypes(rot,cropTypes,cropTypes,cropTypes),2) = YES;
set cropType_2(rot,cropType), cropType_3(rot,cropType), cropType_4(rot,cropType), cropType_5(rot,cropType) $ sum(rot_croptypes(rot,cropTypes,cropTypes,cropTypes),3) = YES;
```

The link between individual crops and the crop types used in the rotation definitions is as follows:

In order to use the crop rotations in the model equations, three cross sets are generated which define the crop type in the first, second and third year for each rotation:

```gams
set cropType_0(rot,cropType), cropType_1(rot,cropType), cropType_2(rot,cropType), cropType_3(rot,cropType) $ sum(rot_croptypes(rot,cropTypes,cropTypes,cropTypes),1) = YES;
set cropType_1(rot,cropType), cropType_2(rot,cropType), cropType_3(rot,cropType), cropType_4(rot,cropType) $ sum(rot_croptypes(rot,cropTypes,cropTypes,cropTypes),2) = YES;
set cropType_2(rot,cropType), cropType_3(rot,cropType), cropType_4(rot,cropType), cropType_5(rot,cropType) $ sum(rot_croptypes(rot,cropTypes,cropTypes,cropTypes),3) = YES;
```

The user can choose for each run which crops can be cropped on farm, such that not all rotations might be operational. Accordingly, in "coeffgen'coeffgen.gms", the set of available crop rotations is defined:

```gams
cropType_0(rot,cropType) $ (not sum( cropType_0(rot,cropType),curCrops ) $ cropType_0(rot,cropType) = NO;
cropType_1(rot,cropType) $ (not sum( cropType_1(rot,cropType),curCrops ) $ cropType_1(rot,cropType) = NO;
cropType_2(rot,cropType) $ (not sum( cropType_2(rot,cropType),curCrops ) $ cropType_2(rot,cropType) = NO;
cropType_3(rot,cropType) $ (not sum( cropType_3(rot,cropType),curCrops ) $ cropType_3(rot,cropType) = NO;
cropType_4(rot,cropType) $ (not sum( cropType_4(rot,cropType),curCrops ) $ cropType_4(rot,cropType) = NO;
cropType_5(rot,cropType) $ (not sum( cropType_5(rot,cropType),curCrops ) $ cropType_5(rot,cropType) = NO;
cropType_6(rot,cropType) $ (not sum( cropType_6(rot,cropType),curCrops ) $ cropType_6(rot,cropType) = NO;
cropType_7(rot,cropType) $ (not sum( cropType_7(rot,cropType),curCrops ) $ cropType_7(rot,cropType) = NO;
cropType_8(rot,cropType) $ (not sum( cropType_8(rot,cropType),curCrops ) $ cropType_8(rot,cropType) = NO;
cropType_9(rot,cropType) $ (not sum( cropType_9(rot,cropType),curCrops ) $ cropType_9(rot,cropType) = NO;
cropType_10(rot,cropType) $ (not sum( cropType_10(rot,cropType),curCrops ) $ cropType_10(rot,cropType) = NO;
```

The rotations enter the model via four constraints (see modelNtempl.gms). The RHS always sums up the crop hectares of a certain crop type in the current year, while the LHS exhausts these hectares in the current, next and after next year based on the rotations grown in these years.
Currently, the rotations only restrict the combination of crops and enter the optional soil pool balancing approach, see below.

2.4 Labour

2.4.1 General concept

The template differentiates between

(1) **General management and further activities for the whole farm** \( p_{\text{labManag}}("\text{farm","const"}) \), which are needed as long as the farm is not given up \( (v_{\text{hasFarm}} = 1, \text{binary variable}) \) and not depending on the level of individual farm activities.

(2) **Management activities and further activities depending on the size of farm branches** (arable cropping, dairying, pig fattening, sows). The necessary working hours are broken down into a base need \( (\text{"const"}) \) which is linked to having the farm branch \( (v_{\text{hasBranch}}, \text{integer}) \) and a linear term depending on its size \( (\text{"slope"}) \).

(3) **Labour needs for certain farm operations** (aggregated to \( v_{\text{totLab}} \)).

The sum if these labour needs cannot exceed total yearly available labour. As discussed below, there are further restrictions with regard to monthly labour and available field working days.
The template model

The maximal yearly working hours $p_{\text{yearlyLab}}$ are defined as:

\[
\begin{align*}
* & \quad \text{--- Akh per year: 52 weeks times 40 hours a week} \\
* & \quad p_{\text{yearlyLab}}(t) = 52 \times 40 \times \% \text{Aks};
\end{align*}
\]

which is considerably more than typically assumed for dependent work.

\section*{2.4.2 Labour needs for farm branches}

The size of a farm branch $v_{\text{branchSize}}$ is defined from activity levels mapped to it:

\[
\begin{align*}
\text{branchSize}_{\text{branches,mc}}(t) & = \sum(\text{branches_to_acts}(\text{branches,acts}),t) & 1 \\
v_{\text{branchSize}}(\text{branches},t) & = \sum(\text{branches_to_acts}(\text{branches,crops}),\text{soil,till,Intens},s) \\
& \quad \times \varepsilon_{s,t,i}(\text{crops,soil,till,Intens}), \\
& \quad \times o_{\text{crops}}(\text{crops,soil,till,Intens},t,s) \times p_{\text{prob}}(s)) \\
& \quad \times \text{if the herd} \text{ true} + \sum(\text{branches_to_acts}(\text{branches,herd}),s) \times p_{\text{herd}}(s) \text{,} \\
& \quad \times u_{\text{herdSize}}(\text{herd},s) \text{)} \\
& \quad \text{endif};
\end{align*}
\]

Where the cross-set $\text{branches_to_acts}$ defines which activities count towards certain branches:

\[
\begin{align*}
& \text{set branches_to_acts}(\text{branches,acts}) / \\
& \text{if} \text{ "farmBranchMeat\%"} \text{ "on" cashCrops.(winterCere,summerCere,winterRape,potatoes, idle)} \\
& \text{if} \text{ "farmBranchDairy\%"} \text{ "on" dairy.(set.set_cows)} \\
& \text{if} \text{ "farmBranchSoup\%"} \text{ "on" souPig.sous } \\
& \text{if} \text{ "farmBranchFatmers\%"} \text{ "on" fatPig.fatners \} / ;
\end{align*}
\]

The binary variable $v_{\text{hasBranch}}$ which relates to the general management need for branch is triggered as follows:

\[
\begin{align*}
* & \quad \text{--- trigger for having branches (steers management labour need)} \\
* & \quad \text{hasBranch}_{\text{branches,mc}}(t)) \times \sum(\text{branches_to_acts}(\text{branches,acts}),t) & 1 \\
& \quad \times v_{\text{branchSize}}(\text{branches},t) = 1 \times v_{\text{hasBranch}}(\text{branches},t) < 1000;
\end{align*}
\]

The $\text{hasFarm}$ trigger depends on the trigger for the individual branches:

\[
\begin{align*}
* & \quad \text{--- trigger for having farm (steer general management labour need)} \\
* & \quad \text{hasFarm}_{\text{branches,mc}}(t)) \times (\text{not sameas}(\text{branches,\"Farm\})) & 1 \\
& \quad \times v_{\text{hasFarm}}(\text{branches},t) = 1 \times v_{\text{hasFarm}}(t);
\end{align*}
\]

\section*{2.4.3 Working off-farm}

Farm family members can optionally work half or full time ($v_{\text{workoff}}$) or on an hourly basis off farm $v_{\text{workHourly}}$. Half and full time work thus needs to be realized as integer variables. The normal setting is that wages for working half time per hour exceed those of short time hourly work, and full time work those of working half time. For half and full time work, commuting time can be accounted for:
The template model

```
* --- off farm work (sum of binary variables) in yearly hours
  * offFarmHoursPerYearFixed_(tCur(t)) .
  * v_offFarmWorkFixed(t) =E=
  * --- off farm labour - per month: p_workTime are weekly hours,
  * p_commTime is the commuting time in weekly hours, assumption of
  * 46 weeks work in each year (binary variables)
  * + sum(workDuration(workType), sum(tt $ ((t1.pos le t.pos)
  * $ (t1.pos_p_decPeriLine-1 ge t.pos)),
  * v_workOff(workType,t)1) * (p_workTime(workType) * p_commTime(workType)) * 46);

The workType set lists the possible combinations:

* --- construct a sequence of half, full, half+full, 2 full, 2 full + half, 3 full ...
* p_workTime(workType) = (p_work(“Half”)+p_work(“Full”)+floor(workType.pos/2)) $ (mod(workType.pos,2) eq 1)
  + p_work(“Full”)*floor(workType.pos/2) $ (mod(workType.pos,2) eq 0);

It is assumed that decisions about how much to work flexibly on an hourly basis are taken on a yearly basis (i.e. the same amount of hours are inputted in each month) and can be adjusted to the state of nature.

The total number of hours worked off-farm is defined as:

* --- total off farm work
  * offFarmWorkTot_(tCur(t)) .
  * u_offFarmWorkTot(t) =E=
  * sum(s, v_workHourly(t,s) * p_prob(s)) + card(n)
  + u_offFarmWorkFixed(t);

2.4.4 Labor needs for farm operations, working off-farm and management

The template considers labour needs for each month \( m \) and each SON \( s \). Labour needs are related to certain farm activities on field and in stable. The labour need for work on farm and flexibly off farm is defined by:

* --- labour use for crops and herds and off-farm, per month,
  * without labour used for management of farm and branches
  * totLabSM_(tCur(t),m,s) .
  *
  * --- sum of work in hours in current month
  * u_totlab(t,m,s) =E=
  * --- labour use for crops and herds
  * u_crapLabSM(t,m,s) $if$ %herd% =true + u_herdLabH(t,n)

* --- off farm labour - per month: p_workTime are weekly hours,
  * p_commTime is the commuting time in weekly hours, assumption of
  * 46 weeks work in each year (binary variables)
  * + u_offFarmWork(t)/card(n)
  *
  * --- small scale work on a hourly basis (continuous)
  * + v_workHourly(t,s)
  *
  * --- labour use for biogas plant
  * $if$ %biogas% =true + sum(curn(bnhw,bnhw), curEeg(eeg)),
    * v_useBiogasPlant(bnhw,eeg,t) * p_labBiogas(bnhw,t,n)

The resulting total monthly work is upper bounded by the parameter \( p_{monthlyLabH} \):
The template model

```plaintext
* --- akh per month: much more then yearly sum to allow covering work peaks
* (e.g. at harvest time)
* *
* p_monthlyLabH(t,n) = %Aks% * 8 * p_daysPerMonth(n);
```

The labour need for animals, \( v_{\text{herdLabM}} \), is defined by an animal type specific need \( p_{\text{herdLab}} \) (see equation below, working hours per animal and month) and a time need per stable place, differentiated by stable type. That formulation thus allows depicting labour saving scale effects related to stable size:

```plaintext
* --- labour need of herds, per month
* herdLabH(tCur(t),n) ..
* * --- labour for animal activities, expressed per animal and month
* u_herdLabH(t,n) = \sum\{actHerd(sumHerd(t), u_herdSize(sumHerd(t)) \* p_herdLab(sumHerd(n))
* * --- fixed amount of hours for stables (maintenance, cleansing),
* * captures also labour saving effects of large stables
* *   \sum\{stables, u_stableShareCost(stables,t) \* p_stableLab(stables,n) \};
```

A similar equation exists for crops, however differentiated by state of nature. The \( p_{\text{cropLab}} \) parameter defines the labour hours per hectare and month for the different crops. In addition, the parameters \( p_{\text{manDistLab}} \) and \( p_{\text{syntDistLab}} \) times the N type amount applied to each crop are added to the overall crop labour demand for the application of synthetic and manure N:

```plaintext
* --- croplabSH(tCur(t),s,n) ..
* u_cropLabSH(t,s,n) =
* * --- labour need for crops, expressed per ha of land
* * (will probably change to specific activities later)
* * \sum\{c,s,t,i\{crops,soil,till,intens\},
*   u_croplab(crops,soil,till,intens,t,s) \* p_croplab(crops,till,intens,n)
* * --- labour need for application of N (fertilizer and manure N)
```

```plaintext
$\text{if}(\text{stateN} == \text{true})$
* \sum\{c,s,t,i\{crops,soil,till,intens\},manAppType,manType\}$ ( u_manDist.uP(crops,soil,till,intens,manAppType,manType,t,s,n) ne 0),
  u_manDist(crops,soil,till,intens,manAppType,manType,t,s,n) \* p_manDistLab(manAppType)$
$\text{endif}$

```plaintext
* \sum\{c,s,t,i\{crops,soil,till,intens\},syntFertilizer\},
  u_synthDist(crops,soil,till,intens,syntFertilizer,t,s,n) \* p_synthDistLab(syntFertilizer);
```

The total labour restriction on a yearly basis reflects the labour needs for the management of the farm and the different branches:

```plaintext
* --- yearly labour restriction, includes management
* tutLab(tCur(t),s,n) ..
* sum(n, u_tutLab(t,n,s))
* * --- two hundred hours independent from number of branches or farm size
* * u_hasFarm(t) \* p_labManag("Farm","const")
* + sum\{branches $\sum\{branches_toActs(branches,acts), t\}$,
*   u_hasBranch(branches,t) \* p_labManag(branches,"const")
* + u_branchSize(branches,t) \* p_labManag(branches,"slope")
```

```plaintext
= p_yearlyLabH(t);
```
2.4.5 Field working days

Field working days define the number of days available in a labor periods of half months \(\text{labPeriod}\) where soil conditions allow specific classes of operations \(\text{labReqLevl}\):

\[
\text{FieldWorkHours}\_\text{labPeriod,labReqLevl,tcur(t,s)}
\]

The number of field work hours cannot exceed a restriction which considers the available field working days \(p_{\text{fieldWorkingDays}}\) which depend on the climate zone and the soil type (light, middle, heavy), the distribution of available tractors to the soil type \(v_{\text{tracDist}}\). It is assumed that farm staff will be willing to work up to 12 hours a day (however considering that total work load per month is restricted):

\[
\text{FieldWorkHours}\_\text{labPeriod,labReqLevl,tcur(t,s)} = \text{FieldWorkHours}\_\text{labPeriod,labReqLevl,tcur(t,s)} \times 12
\]

The distribution of tractors is determined endogenously:

\[
\text{TracDistribution}\_\text{labPeriod,tcur(t,s)}
\]

The tractor inventory is upper bounded by the number of farm staff:

\[
u_{\text{tracInv.up}}(t) = \text{ceiling}(\%\text{AKs})
\]

which implicitly assumes that farm family members are willing to spend hours for on farm work even if working off farm, e.g. by taking days off.

2.5 Stables

The template applies a vintage based model for different stable types, other buildings and selected machinery, and a physical used based depreciation for the majority of the machinery park. Under the vintage model, stables, buildings and machinery become unusable after a certain, fixed number of years after construction. If physical depreciation is used, machinery becomes inoperative if its maximum number of operating hours or another measurement of use (e.g. the amount handled) is reached. Investments in stable,
buildings and machinery are implemented as binary variables. In order to keep the possible branching trees at an acceptable size, the re-investment points can be restricted to specific years. For longer planning horizon covering several decades, investment could e.g. only be allowed every fourth or fifth year.

The stable inventory $v_{\text{stableInv}}$ for each type of stable (stables) is hence defined as:

```plaintext
\[
\begin{align*}
\text{v}_{\text{stableInv}}(\text{stables,hor,typ},t) & \Downarrow \Sigma (\text{t}, \text{u}_{\text{buyStables,mp}}(\text{stables,hor,typ},t)) \\
& \text{or} \Sigma (\text{t,old, p}_{\text{inStables}}(\text{stables,hor,old},t))) ..
\end{align*}
\]
```

```plaintext
\[
\begin{align*}
\text{u}_{\text{stableInv}}(\text{stables,hor,t}) & \Downarrow \Sigma (t) \\
& \cdot (p_{\text{year}}(t) \cdot p_{\text{prolongSt}} > p_{\text{year}}(t)) \text{ and } (p_{\text{year}}(t) > \text{min}(\text{hor1,old}) \text{ and } \text{max}(\text{hor1,old}) \cdot p_{\text{inStables}}(\text{stables,hor,old},t))) \\
& \cdot (\text{not } \Sigma (t) \text{ and } p_{\text{prolongSt}}) \\
\text{-- old stables according to building date and lifetime}
\end{align*}
\]
```

```plaintext
\[
\begin{align*}
\text{p}_{\text{lifeTimeS}} & \text{ is the maximal physical life time of the stables and } \text{v}_{\text{buyStables}} \text{ are newly constructed stables.}
\end{align*}
\]
```

For cow stables, a differentiation is introduced between the initial investment into the building, assumed to last for 30 years, and certain equipment for which maintenance investments are necessary after 10 or 15 years, as defined by the investment horizon set $\text{hor}$:

```plaintext
\[
\begin{align*}
\text{u}_{\text{buyStables}}(\text{stables,hor,t}) & \Downarrow \Sigma (t) \\
& \cdot (p_{\text{year}}(t) \cdot p_{\text{lifeTimeS}(\text{stables,hor})} \geq p_{\text{year}}(t)) \\
& \cdot (p_{\text{year}}(t) \cdot \text{lifeTimeS}(\text{stables,hor}) \geq p_{\text{year}}(t)))
\end{align*}
\]
```

A stable can only be used, if the also short and middle term maintenance investment had been undertaken.

The model currently distinguishes between the following stable types for cattle:
The template model

```
set stablesTypes / milk&cows, calves, young&cattle, fattens, sows, piglets /;
set stables(*) /

$if then $dairyModel == true

milk30
milk40
milk50
milk75
milk90
milk100
milk120
milk150
milk175
milk190
milk200
milk250
milk275
young&cattle30
young&cattle45
young&cattle60
young&cattle75
young&cattle90
calves30
calves45
calves60
calves75
calves90

$endif

which differ in capacity, investment cost and labour need per stable place. For pigs, the following size are available:

```
$if then $pigModel == true

fat100
fat150
fat200
fat1000
fat1200
fat1500
fat2000

``` --- sow and piglets

```
sows120
sows150
sows200
sows250
sows300
sows350
sows400
piglets30
piglets50
piglets70
piglets90
piglets100
piglets120
piglets150
piglets190

$endif

The used part of the stable inventory (a fractional variable) must cover the stable place needs for the herds:
The template model

2.6 Other type of buildings

Besides stables, the model currently includes silos, more manure, bunker silos for maize, or grass silage and storages for potatoes.

For each type of manure silo, an inventory equation is present:

- --- manure silo inventory (depreciation over time)
  siloinv(silos,tur(t)) $ (sum(t1, v_buysilos.mp(silos,t1)) or (sum(t0old, p_insilos(silos,t0old))))) ..
  e_siloinv(silos,t)
  =E=

- --- old silo according to building date and lifetime (will drop out of equation if too old)
  sum(t0old) $ ( ((p_year(t0old) * p_lifelineSi(silos)) ge p_year(t))
  $ ( p_year(t) le p_year(t)),
  p_insilos(silos,t0old))

- --- plus (old) investments - de-investments
  + sum(t1 $ (tur(t1) and ((p_year(t1) * p_lifelineSi(silos)) ge p_year(t))
  +( p_year(t1) le p_year(t)),
  v_buysilos(silos,t1));

The manure silos are linked to the manure storage needs which are described below.
A similar inventory equation as for manure silos is found for the other buildings:

```
* --- buildings and structures inventory
* buildinginv(builings(buildings),t,cur(t)) $ (sum(t1, v_buyBuildings_up(buildings,t1))
  or (sum(t0ld, p_inilBuildings(buildings,t0ld)))) ..
  v_buildingsinv(buildings,t)

* --- old silo according to building date and lifetime
* (will drop out of equation if too old)
* sum(t0ld) $ { (p_year(t0ld) + p_lifetimeBuild(buildings)) ge p_year(t))
  $ (p_year(t0ld) < p_year(t)),
  p_inilBuildings(buildings,t0ld)}

* --- plus (old) investments - de-investments
* + sum(t1) $(tcur(t1) and ( (p_year(t1) + p_lifetimeBuild(buildings)) ge p_year(t))
  $ (p_year(t1) < p_year(t)),
  v_buyBuildings(buildings,t1));
```

The buildings covered are:

```
set s_bunkerSilo / bunkerSilo8800 
  bunkerSilo1540 
  bunkerSilo2630 
  bunkerSilo2550 
  bunkerSilo8750 
  bunkerSilo8000 
  bunkerSilo26250 
/;
set buildings / potaStore500t
  set.s_bunkerSilo
/;
```

The attributes of the buildings are defined in "coeffgen\buildings.gms":

```
$context
set buildAttr / invSun "Investment sun"
  capact "Storage capacity in tons"
  capac_m3 "Storage capacity in m3"
  lifetime "Lifetime in years"
  varCost "Variable costs per year"
/;
```

```
<table>
<thead>
<tr>
<th>building</th>
<th>invSun</th>
<th>capact</th>
<th>capac_m3</th>
<th>lifetime</th>
<th>varCost</th>
</tr>
</thead>
<tbody>
<tr>
<td>potaStore500t</td>
<td>178000</td>
<td>590</td>
<td>12</td>
<td>306</td>
<td></td>
</tr>
</tbody>
</table>
```

The inventory of the buildings is linked to building needs of certain activities:

```
```
The template model

2.7 Farm machinery

The model accounts in detail for different farm machineries:

```
set machType / tractor, plough, chiselPlough, corn, directSeedMachine, circHarrow
        springTimeHarrow, fingerHarrow, combine, cuttingMachine
        rotaryHarrow, mulcher, potatoPlanter, potatoLifter, hne
        ridger, hayCut, forkLiftTruck, threeWayLoppingTrailer
        sprayer, single Seeder, beetHarvester, fertilizerSpreader
        chopper, corn header, rower, rice seeder, forklift, trailer
        grass seed, flake, baler, feed mixer
        /
```

Each machinery type is characterized by set of attributes $p_{machAttr}$ (see coeffgen\mach.gms):

```
table p_machAttr(machType,machAttr) "Machinery attribute for default size (APH, 2 ha)"

price         hour         ha         m3         t         varCost_ha         varCost_t         diesel_h         fixCost_h         fixCost_t
Plough
tractor
```

Ficcost are derived by dividing yearly fix costs by total lifetime in hours by $8$ of years

```
trractor 40 000 10 000
```
2.7.1 Farm operations: machinery needs and related costs

Machinery is linked to specific farm operations (see tech.gms):

```plaintext
set operation "Field operators as defined by KTB"
/
soilSample "Bodenprobe"
mansort "Güleiausbringung"
dasFert "P und K Düngung, typischerweise Herbst"
plow "Pflügen"
chiselPlow "Tiefgruberpflügen"
seedBotComb "Saatbettkombination"
herb "Herbizidannahme"
sowMachine "Saemachines"
directSowMachine "Direktsaatmaschine"
circleNarrowSow "Kreislegge u. Drillmaschinen Kombination"
springtimeNarrow "Federzinkenelegge"
weedEvaluation "Unkrautbeurteilung"
weederLight "Strieglern"
weederIntens "Hacken"
plantEvaluation "Bestandsbeurteilung"
Nfertilizer0
Nfertilizer10
combineGrain "Mähdresch, Getreide"
combineRaip "Mähdresch, Raps"
combineMaiz "Mähdresch, Mais"
conYnt Transport "Getreidetransport"
store_n_dry_0
store_n_dry_m
store_n_dry_beans
store_n_dry_rape
store_n_dry_corn
lime_fert "Kalkung"
stubble_shallow "Stoppelbearbeitung flach"
stubble_deep "Stoppelbearbeitung tief"
rotaryNarrow "Kreislegge"
NminTesting "Nmin Probenahme"
mulcher "Mulcher"
chitting "Vorkeimen"
soilManDist "Niststreuer"
seedPotatoTransport "PflanzenKartoffeltransport"
potatoLaying "Legen von Kartoffeln"
rakingHoeing "Hacken, streiegn"
earthingUp "Hüfeln"
knockOffMaize "Kartoffelkraut schlagen"
killingMat "Krautabtötung"
potatoHarvest "Kartoffeln roden"
potatoTransport "Kartoffeln zum Lager transportieren"
potatoStorage "Kartoffeln lagern"
seedlingRaiser "Einzelnkornlegger für Zuckerrüben/Mais"
weederHand "von Hand hacken"
uprootBeets "Zuckerrüben roden"
Diammonium "Diammonphosphat streuen"
grinding "Kornmahlen"
disposal "Ernteget festfahren"
coveringSilo "Silo zudecken"
chopper "Näckseln"
grassReSeeding "Grasnährsäen"
railer "Walzen"
moving "Hähen mit Haulfahreim"
raking "Schwaden"
teeding "Wenden mit Kreiselzettwender"
silageTrailer "Anweibgut bergen mit Ladewagen"
closeSilo "Silo reinigen und mit Folie verschlissen"
/;
```

Labour needs, diesel, variable and fixed machinery costs are linked to these operations:
The template model

\*

\* table op_attr(operation,machVar,plotSize,opAttr)

<table>
<thead>
<tr>
<th>soilSample</th>
<th>.labTime</th>
<th>diesel</th>
<th>fixCost</th>
<th>varCost</th>
</tr>
</thead>
<tbody>
<tr>
<td>60km.2ha</td>
<td>0.2</td>
<td>0.5</td>
<td>1.05</td>
<td>0.30</td>
</tr>
<tr>
<td>manDist</td>
<td>0.67km.2ha</td>
<td>1.7</td>
<td>6.7</td>
<td>20.20</td>
</tr>
<tr>
<td>manFert</td>
<td>0.67km.2ha</td>
<td>0.25</td>
<td>0.9</td>
<td>2.04</td>
</tr>
</tbody>
</table>

\*

\* --- page 153, KTBL 2010/2011

\* plow:

<table>
<thead>
<tr>
<th>60km.2ha</th>
<th>1.09</th>
<th>25.0</th>
<th>20.39</th>
<th>48.75</th>
</tr>
</thead>
<tbody>
<tr>
<td>sow</td>
<td>0.67km.2ha</td>
<td>0.50</td>
<td>6.0</td>
<td>7.98</td>
</tr>
<tr>
<td>sow</td>
<td>0.67km.2ha</td>
<td>0.84</td>
<td>4.0</td>
<td>9.44</td>
</tr>
<tr>
<td>direct sows</td>
<td>0.67km.2ha</td>
<td>1.29</td>
<td>12.9</td>
<td>16.96</td>
</tr>
<tr>
<td>sowing harrow</td>
<td>0.67km.2ha</td>
<td>0.75</td>
<td>7.3</td>
<td>6.56</td>
</tr>
<tr>
<td>weaving</td>
<td>0.67km.2ha</td>
<td>0.16</td>
<td>0.3</td>
<td>1.59</td>
</tr>
<tr>
<td>herb</td>
<td>0.67km.2ha</td>
<td>0.28</td>
<td>1.0</td>
<td>4.37</td>
</tr>
<tr>
<td>weeder light</td>
<td>0.67km.2ha</td>
<td>0.42</td>
<td>2.6</td>
<td>8.90</td>
</tr>
</tbody>
</table>

\*

The models considered the effect of different plot size and the mechanisation level:

\* --- see page 250 KTBL 2010/2011 for winter cereals
\* Describe effect of plot size and mechanisation (= work width) on time, variable and fix
\* machinery costs and diesel.

\* table p_plotSizeEffect(crops,machVar,opAttr,plotSize)

<table>
<thead>
<tr>
<th>winterCere. 60km</th>
<th>labTime</th>
<th>1ha</th>
<th>2ha</th>
<th>5ha</th>
<th>10ha</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4</td>
<td>10.5</td>
<td>9.2</td>
<td>8.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>winterCere. 60km</td>
<td>diesel</td>
<td>90</td>
<td>83</td>
<td>78</td>
<td>73</td>
</tr>
<tr>
<td>winterCere. 10km</td>
<td>varCost</td>
<td>205</td>
<td>180</td>
<td>176</td>
<td>168</td>
</tr>
<tr>
<td>winterCere. 10km</td>
<td>fixCost</td>
<td>258</td>
<td>241</td>
<td>221</td>
<td>211</td>
</tr>
<tr>
<td>winterCere. 100km</td>
<td>labTime</td>
<td>11.1</td>
<td>9.1</td>
<td>7.6</td>
<td>6.8</td>
</tr>
<tr>
<td>winterCere. 100km</td>
<td>diesel</td>
<td>95</td>
<td>86</td>
<td>78</td>
<td>74</td>
</tr>
<tr>
<td>winterCere. 100km</td>
<td>varCost</td>
<td>209</td>
<td>188</td>
<td>172</td>
<td>164</td>
</tr>
<tr>
<td>winterCere. 100km</td>
<td>fixCost</td>
<td>264</td>
<td>262</td>
<td>256</td>
<td>249</td>
</tr>
<tr>
<td>winterCere. 200km</td>
<td>labTime</td>
<td>11.9</td>
<td>9.6</td>
<td>6.3</td>
<td>4.9</td>
</tr>
<tr>
<td>winterCere. 200km</td>
<td>diesel</td>
<td>148</td>
<td>99</td>
<td>84</td>
<td>75</td>
</tr>
<tr>
<td>winterCere. 200km</td>
<td>varCost</td>
<td>248</td>
<td>231</td>
<td>173</td>
<td>157</td>
</tr>
<tr>
<td>winterCere. 200km</td>
<td>fixCost</td>
<td>303</td>
<td>328</td>
<td>292</td>
<td>267</td>
</tr>
</tbody>
</table>

\* use mean of given crops if missing for one of the crops

\* p_plotSizeEffect(crops,machVar,opAttr,plotSize) \$(\text{not p_plotSizeEffect(crops,machVar,opAttr,plotSize)})$
\* \text{sum(crops, p_plotSizeEffect(crops,machVar,opAttr,plotSize))}$
\* \text{sum(crops, p_plotSizeEffect(crops,machVar,opAttr,plotSize),1);}$

The farm operations are linked to cropping activities (below an example for potatoes):
The detailed information on farm operations determines:

1. The **number of necessary field working days** and **monthly labor need** per ha (excluding the time used for fertilizing, which is determined endogenously)
2. The **machinery need** for the different crops
3. Related **variable costs**

The labor needs per month are determined by adding up over all farm operations, considering the labor period, and the effect of plot size and mechanization (coeffgen\labour.gms):

```plaintext

1. The **number of necessary field working days** and **monthly labor need** per ha (excluding the time used for fertilizing, which is determined endogenously)
2. The **machinery need** for the different crops
3. Related **variable costs**

The labor needs per month are determined by adding up over all farm operations, considering the labor period, and the effect of plot size and mechanization (coeffgen\labour.gms):

```
2.7.2 Endogenous machine inventory

The inventory equation for machinery is shown below, where \( v_{\text{machInv}} \) is the available inventory by type (\( \text{machType} \)) in operation hours, \( v_{\text{machNeed}} \) is the machinery need of the farm in operating hours and \( v_{\text{buyMach}} \) are investments in new machines.

```
* --- inventory of mach according to operation hours
* machine_inv(MachType,machLifeUnit,tEnd(t)) $\{ (v_{\text{machInv}(\text{machType},\text{machLifeUnit},tEnd(t)}) \neq 0) 
* \text{pLifeTime}(\text{machType},\text{machLifeUnit}) \times \text{pPriceMach}(\text{machType},t) 
* \text{not sameas}(\text{machLifeUnit},'years') \} ...
* --- inventory end of current year (in operating hours)
* v_{\text{machInv}}(\text{machType},\text{machLifeUnit},t)
  :=
* --- inventory end of last year (in operating hours)
* v_{\text{machInv}}(\text{machType},\text{machLifeUnit},t-1)
* --- new machines, converted in operation time
* + (v_{\text{buyMach}}(\text{machType},t) + v_{\text{buyMachFlex}}(\text{machType},t)) \times \text{pLifeTime}(\text{machType},\text{machLifeUnit})
* --- minus operating hours in current year if in normal planning period
* \times \text{pProb}(s) \times \text{pTCur}(t)
* --- minus operating hours of weighted average over normal planning period
* if beyond the normal planning period
* \times \text{pProb}(s) \times \text{pTCur}(t)
* \times \text{pProb}(s) \times \text{pTCur}(t)
* \times \text{pProb}(s) \times \text{pTCur}(t)
* \text{not TCur}(t) \} \text{ and p_prolongCalc}
```

The last expression is used if the farm program for the simulated period is used to estimate the machinery needs for the years until the stables are fully depreciated.

The machinery need in each year is the maximum of the need in any state-of-nature in that year:

```
machNeedHerds_ (machType,t) ..
  u_{\text{machNeedHerds}}(\text{machType},t) :=
  --- herd sizes times their request for specific machine type
  \times \text{sum(achHerds(sumHerds,t), u_herdsSize(sumHerds,t) \times p_machNeed(sumHerds, machType))};
  --- need of machineries per SUM, the SUM with the highest need
  defines the machinery park investments
  \times \text{machines_ (machType, TCur(t),s) ..}
  u_{\text{machNeedHerds}}(\text{machType},t)
  --- crops times their request for specific machine type
  \times \text{sum( (crops), u_cropHa(crops,t,s) \times p_machNeed(crops, machType))}
  --- must be covered by current stable inventory (not fully depreciated building),
  multiplied with the stable places they offer
  \times u_{\text{machNeed}}(\text{machType},t);
```

A small set of machinery, such as the front loader, dung grab, shear grab or fodder mixing vehicles are not depreciated by use, but by time:
The template model

2.8 Investments, their financing and cash flow definition

The total investment sum \( v_{sumInv} \) in each year is defined by:

\[
\begin{align*}
\text{v_sumInv(t)} & = \text{v_sumInv(t)} + \text{new land bought} \\
& + \text{new land bought} \\
& + \text{buildings and structures} \\
& + \text{new machinery bought} \\
& + \text{new manure silos bought} \\
& + \text{new biogas plant bought} \\
& + \text{biogases bought} \\
\end{align*}
\]

It can be financed either by equity or by credits, and enters according the cash balance \( v_{liquid} \) definition. The cash balance is the cash at the end of the last year plus the net cash
flow \( v_{\text{netCashFlow}} \) from the farm in the current year plus new credits \( v_{\text{credits}} \), minus fixed household expenditures \( p_{\text{hcon}} \) and new investments:

\[
\text{Liquid}(t) = E = v_{\text{liquid}}(t) - E - v_{\text{liquid}}(t-1) + v_{\text{netCashFlow}}(t) - new \text{ credits}
\]

The net cash flow is defined as the sum of the gross margin in each SON (\( v_{\text{objeTS}} \)), interested gained on cash, interested paid on outstanding credits and paying back credits.

The model differentiates credits by repayment periods \( p_{\text{payBackTime}} \) and interest rates. Credits are paid back in equal instalments over the repayment period, so that the annuity drops from year to year. The sum of outstanding credits is defined by the following equation:

\[
v_{\text{sumCredits}}(creditType, t) = E = \sum(t | t \leq t_{\text{year}}(t)) \left( (1 - p_{\text{payBackTime}}(creditType)) \ast (p_{\text{year}}(t) - p_{\text{year}}(t_{\text{year}}(t))) \right) \ast v_{\text{credits}}(creditType, t)
\]
The template model

For the last year, where it is assumed that the firm is liquidated, the following term is added as seen above:

\[
\text{liquidation} = \sum_{t=1}^{T} \text{sum}(\text{creditType}, \text{v_sumCredits}(:\text{creditType}, \text{\"LastYearCalc\"})),
\]

The liquidation is only used if the model runs in fully dynamic mode, and not taking into account in comparative-static and short run mode.

The gross margin for each state-of-nature is defined as revenues from sales (\(v_{\text{salRev}}\)), income from renting out land (\(v_{\text{rentOutLand}}\)) and working off farm, costs of buying intermediate inputs comprised in the equations structure of the model template (\(v_{\text{buyCost}}\)) and other variable costs (\(v_{\text{varCosts}}\)) not explicitly covered. For off-farm work (full and half time, \(v_{\text{workOff}}\)), a weekly workTime in hours is given (\(p_{\text{weekTime}}\)), it is assumed
The template model

that 46 weeks are worked a year, so that income is defined then from multiplied these two terms with hourly wage $p\_wage$.

\[
\text{v\_objTS}(t,s) = \text{income from selling products} \\
\quad \text{v\_salRev(t,s)} \\
\text{single farm premium} \quad \text{v\_sfPrem(t,s)} \\
\text{revenues from selling biogas} \\
\text{if}\%\text{biogas} = \text{true} \quad \text{v\_salRevBiogas(t)} \\
\text{income from renting out land} \\
\text{if}\%\text{landlease} = \text{true} \quad \text{sum( plot, v\_rentOutPlot(tplot, t) \cdot p\_plotSize(plot) \cdot p\_landRent(plot, t))} \\
\text{off farm income, flexible on a hourly basis} \\
\quad \text{v\_workHourly(t,s) \cdot p\_wage(hourly, t,s) \cdot card(m)} \\
\text{half/full time work off the farm, decided upon for several years (p\_decPeriodInYr)} \\
\quad \text{sum(worktype(workType), \sum(t1 \cdot (t1\_pos le t\_pos) \cdot (t1\_pos \cdot p\_decPeriodInYr - 1 \ge t\_pos), \text{v\_workOff(worktype, t1)} \cdot p\_workTime(workType, t1) \cdot 46 \cdot p\_wage(workType, t,s)})} \\
\text{variable cost (not explicitly covered by constraints) minus premiums} \\
\quad \text{- v\_varCost(t,s);}
\]

The sales revenues $v\_salRev$ entering the equation above are defined from net production quantities $v\_prods$ and given prices in each year and state of nature $p\_price$:

\[
\text{v\_saRev(t,s) - E - \sum( cur\_prods(prodsYearly), p\_price(prodsYearly, t,s) \cdot v\_saleQuant(prodsYearly, t,s))};
\]

The sales quantities plus feed use $v\_feedUse$ must exhaust the production quantity $v\_prods$:

\[
\text{v\_saleQuant(prodsYearly, t,s) \cdot p\_price(prodsYearly, t,s)} \\
\quad \text{sum( sameas(prodsYearly, feedY), \sum(t\_feed(t1) \cdot (t1\_pos le t\_pos) \cdot (t1\_pos \cdot p\_decPeriodFeed - 1 \ge t\_pos), \text{v\_feedUse(feedY, t1)}) / \sum(t2 \cdot \cdot (t2\_pos \cdot p\_decPeriodFeed - 1 \ge t2\_pos) \cdot t\_cur(t2), t1))} \\
\text{if}\%\text{biogas} = \text{true} \quad \text{sum( sameas(prodsYearly, crh), \sum( cur\_bhex(bhke), cur\_eeg(eeg), n), u\_feed\_biogas(bhke, eeg, crh, t, n))} \\
\text{endif}
\]

The production quantities are derived from the production quantities not used on farm for feeding and depend on herd sizes respectively cropped hectares:
One specific feature is the variable \( v_{\text{redMlk}} \) which allows the farmer to not fully use the genetic potential of the milk cow by adjusting the herd mix. This could for example be of relevance in the optimization process, when the yield potential of different herds are very high, but when price combinations of in- and output lead to an economic optimal intensity level that is below the maximum milk yield potential. Otherwise, cows would have always to be milked at the maximum. Another relates to the differentiation of calves and young bulls by breed.

### 2.9 Manure

#### 2.9.1 Manure excretion

The calculation for manure amounts in the model template is expressed both in fluid manure quantities in m³ \( v_{\text{manQuant}} \):

\[
\text{manQuant}_\text{(ttwr(t))} = \sum(\text{actHerds}\text{(possHerds,breeds,t),m),}
\text{v\_herdSize(possHerds,breeds,t) = p\_manQuantMonth(herds);}
\]

And in nutrients \( v_{\text{nutManureM}} \):

\[
\text{p\_manQuantMonth(fcalvs)} = \text{p\_lu(fcalvs)} \times 1.5;
\text{p\_manQuantMonth(mcalvs)} = \text{p\_lu(mcalvs)} \times 1.5;
\text{p\_manQuantMonth(fcalvsRais)} = \text{p\_lu(fCalvsRais)} \times 1.5;
\text{p\_manQuantMonth(heifs)} = \text{p\_lu(heifs)} \times 1.5;
\]

Manure quantities excreted per head is either based on fixed coefficients:
Or, for cows, depending on milk yield:

\[
\begin{align*}
\text{p}_{\text{nutQuantMonth}}(\text{cows}, \text{"milK"}, t) &= (12.8 \times \text{smax}(t, \text{p}_{\text{dCoeff}}(\text{cows}, \text{"milK"}, t)) \times 1000 \times 0.0008) \times (1/12);
\end{align*}
\]

Based on typical nutrient content in manure, the nutrient excreted are defined:

\[
\begin{align*}
\text{p}_{\text{nutQuantMonth}}(\text{cows}, \text{"milK"}) &= \text{p}_{\text{nutQuantMonth}}(\text{cows}, \text{"milK"}) \times (3.25 + \text{smax}(t, \text{p}_{\text{dCoeff}}(\text{cows}, \text{"milK"}, t)) \times 1000 \times 0.00087);
\end{align*}
\]

2.9.2 Manure storage

The monthly amount of nutrients in the different storage types is shown by the variable \( v_{\text{NutInStorageType}}(\text{ManStorage}, t, m) \):

\[
\begin{align*}
\text{nutInStorage}_{\text{nut}, \text{tCur}(t, m)} &\quad \{(\text{not } (\text{sameas}(\text{t}, \text{"BlastYear"}) \& \text{ sameas}(\text{m}, \text{"Dec"}))) \} \quad .
\text{v}_{\text{nutInStorage}}(\text{nut}, \text{t}, \text{Dec}) &\quad \{(\text{sameas}(\text{m}, \text{"Jan"}) \& \text{ tCur}(\text{t}, \text{Dec}))\} \\
&\quad \text{v}_{\text{nutInStorage}}(\text{nut}, \text{t}, \text{Jan}) \quad \{(\text{not } \text{ sameas}(\text{m}, \text{"Jan"}))\}
\end{align*}
\]

The nutrient losses during storage depend on the \text{manStorage} type, i.e. whether the silo is covered or not, and how it is covered:

\[
\begin{align*}
\text{NutlossesStorage}_{\text{nut}, \text{tCur}(t, m)} &\quad \text{sameas}(\text{nut}, \text{"N"}) \} \quad .
\text{v}_{\text{NutlossesStorage}}(\text{nut}, \text{t}, \text{m}) &\quad \text{sum}(\text{manStorage}, \text{v}_{\text{nutInStorageType}}(\text{ManStorage}, \text{t}, \text{m})) \\
&\quad \text{p}_{\text{nutLossStorage}}(\text{manStorage}, \text{nut});
\end{align*}
\]

The type of silo cover used for a certain type of silo \( v_{\text{siCovComb}} \) is a binary variable, i.e. one type of silo must be fully covered or not:

\[
\begin{align*}
\text{--- declaration that the model knows what coverage type is on the manure silos}
\text{v}_{\text{siCoverinw}}(\text{silos}, \text{tCur}(t)) &\quad \{(\text{sum}(\text{t1}, \text{v}_{\text{hugsiles}}(\text{silos}, \text{t}))) \or (\text{sum}(\text{t0ld}, \text{p}_{\text{iniSilos}}(\text{silos}, \text{t0ld}))) \} \quad .
\text{v}_{\text{siInlv}}(\text{silos}, t) &\quad \text{sum}(\text{siCover}, \text{v}_{\text{siCovComb}}(\text{silos}, \text{siCover}, t));
\end{align*}
\]

The volume of manure in storage \( v_{\text{volInStorage}} \) is accounted for as follows:
The template model

--- defines manure amount in m³ per month in each storage type

\[
\text{volInStorage}_{t Cur(t), n} ..
\]

\[ u_{\text{volInStorage}}(t, n) := \begin{cases} 
  u_{\text{volInStorage}}(t-1,'Dec') & \text{if sameas(m,'Jan')} \land tCur(t-1) \\
  u_{\text{volInStorage}}(t, n-1) & \text{if not sameas(n,'Jan')} 
\end{cases} \]

--- m³ excreted per year divided by # of month: monthly inflow

\[ * u_{\text{manQuant[t]}}/\text{card(n)} \]

--- m³ taken out of storage type for application

\[ - u_{\text{volManApplied}}(t, n); \]

The volume is distributed to the different storage types based on the following equations:

--- defines how manure amount is distributed to the single storage types

\[
\text{storageDistr}_{t Cur(t), n} ..
\]

\[ u_{\text{volInStorage}}(t, m) = \sum \text{manStorage}, u_{\text{volInStorageType}}(\text{ManStorage}, t, n); \]

The farm has to ensure a certain manure storage capacity in relation to the total manure products. That storage capacity consists of sub-floor capacity \( v_{\text{subManStorCap}} \) of the stables plus the capacity of additional manure silos \( v_{\text{siloManStorCap}} \):

--- total manure storage capacity of overall farm in m³

\[
\text{totalManStorCap}_{t Cur(t)} ..
\]

\[ u_{\text{TotalManStorCap}}(t) -= u_{\text{SubManStorCap}}(t) + u_{\text{SiloManStorCap}}(t); \]

The sub-floor capacity is derived from the stable inventory and stable type specific sub-floor storage capacity \( p_{\text{manStorCap}} \):

--- storage capacity for manure subFloor in stable systems

\[
\text{subManStorCap}_{t Cur(t)} ..
\]

\[ u_{\text{SubManStorCap}}(t) = \sum \text{stables} \{ \text{sum(t1, u_buyStables,up(stables,t1)) or (sum(t0ld, p_initStables(stables,t0ld)), u_StableInv(stables,t1) + p_{\text{ManStorCap}}(stables)); } \]

The capacity in silos for similarly derived, drawing on silo type specific storage capacity \( p_{\text{manStorCapSi}} \):

--- storage capacity for manure in outdoor silo systems

\[
\text{siloManStorCap}_{t Cur(t)} ..
\]

\[ u_{\text{SiloManStorCap}}(t) = \sum \text{silos} \{ \text{sum(t1, u_buySilos,up(silos,t1)) or (sum(t0ld, p_initSilos(silos,t0ld))), u_SiloInv(silos,t1) + p_{\text{ManStorCapSi}}(silos)) } \]

The ability of storing manure in the stable building \( p_{\text{ManStorCap}} \) depends on the stable system. Slurry based systems with a plane floor normally only have small cesspits which demand the addition of manure silo capacities. The manure storage capacity of stables with slatted floor depends on the size of the stable, where a storage capacity for manure of 3 month in a fully occupied stable is assumed here. For storage scarcity, a set of different dimensioned liquid manure reservoirs is implemented into the model as investment

39
opportunities for the creation of new storage capacities ($p_{\text{ManStorCapSi}}$) from 400 to 1400 m³:

* set of different silo manure storage sizes

```plaintext
set silos /silo400,silo600,silo800,silo1000,silo1200,silo1400/;
```

The lifetime ($p_{\text{LivetimeSi}}$) is quantified with 30 years and the investment costs ($p_{\text{priceSilo}}$) depend on the m³ storage capacity ($p_{\text{ManStorCapSi}}$) (35€/m³) following KTBL (2010:p.743) and increase by one percent from year to year:

*--- investment cost for different manure silos nach KTBL 2010 S.743

$$p_{\text{priceSilo}}(silo,t) = 35 \times p_{\text{ManStorCapSi}}(silo) \times (1.01 \times t^{0.01});$$

The necessary silo capacity is given by 50% of the manure quantity in m³ per year $v_{\text{manQuant}}$:

```plaintext
* -- Manure storage capacity must cover at least 50% of annual manure quantity excreted
* following: Verordnungen der Bundesländer über Anforderungen an Anlagen zur
* Lagerung und Abfüllen von Jauche, Gülle, Silagesickersäften, Festmist und Silagen (JRSF-00)

manStorCapNeed(tCur(t)) = 0.5 \times v_{\text{manQuant}}(t);
```

And cannot exceed the given manure storage capacity:

```plaintext
*---- total manure storage capacity has to be greater than required storage capacity ManStorCapNeed(t)

manStorCap(tCur(t)) =

v_{\text{TotalManStorCap}}(t) = g= v_{\text{ManStorCapNeed}}(t);
```

### 2.9.3 Manure application

Different application procedures for manure N are implemented $\text{ManApplicType}$: road spread, drag hose spreader and injection of manure. The different mentioned recovering techniques are combined with different related application costs $p_{\text{manApplicCost}}$, labour requirements as well as affects on GHG flows. These parameters, as well as relevant parameters for the use of synthetic fertilizers are defined in a sub-module “fertilizing”. Investment costs for additional machinery like manure barrel and distribution console is taken from KTBL 2010 (pp.92-93). The lifetimes of the manure application components are defined as a maximal amount of kg N, deviated from by KTBL given total m³ capacity. For the first instance, we assume a 12m³ vacuum tank wagon (lifetime transportation capacity of 120000m³), a 15m drag hose spreader and an injector system with 6m working width.

The distribution of manure has to link nutrient with volumes while accounting for the fact that depending on the herd composition, the nutrient content per m³ might change. Accordingly, different manure types are defined:

```plaintext
set manType(*) "Manure types, characterized by N:P ratio" /
$\text{if}1 \ % \text{dairyHerd} \ % \text{true} \ \text{catt}
$\text{if}1 \ % \text{pigHerd} \ % \text{true} \ \text{pig}

"/;
```
The application of total N ($v_{NTotalApplied}$) (organic from manure or synthetic) is shown on monthly level. The spreading of manure is banned from November till January following the German Nitrate directive. Furthermore the application of manure on maize is not possible by injection technique and also for other crops and grassland manure application is not possible for diverse month in summer (shown by set doNotApplyManure(crops,m)).

Also considering legal guidelines for the German agricultural practice as given by the “Düngeverordnung” (DüV §4, Abs.3) specifications, an application limit per year for manure N ($p_{nManAppLimit(crops)}$) is implemented for crop land (max. 170 kg N/ha from animal origin) and for grassland and pasture (max. 230 kg N/ha from animal origin):

Also considering legal guidelines for the German agricultural practice as given by the “Düngeverordnung” (DüV §4, Abs.3) specifications, an application limit per year for manure N ($p_{nManAppLimit(crops)}$) is implemented for crop land (max. 170 kg N/ha from animal origin) and for grassland and pasture (max. 230 kg N/ha from animal origin):

2.10 Synthetic fertilizers

To strike the N demand of different crops, also the addition of synthetic N fertilizer is allowed ($v_{nSyntDist(crops,syntNFertilizer,t,sAll,m)}$). The synthetic fertilizer has a
The template model

specific price per kg N and furthermore bears application N loss rates as well as requirements for labour \( (p_{\text{syntDistLab}}(\text{syntNFertilizer})) \) and machinery \( (p_{\text{syntDistMachNeed}}(\text{syntNFertilizer}, \text{machType})) \) for tractor and sprayer.

2.11 N and P2O5 accounting

2.11.1 General concept

The template supports two differently detailed ways to account for nutrient accounting:

1. A fixed factor approaches with yearly soil balances per crop
2. A detailed flow model with a monthly resolution by soil depth.

\( p_{\text{nNeed}} \) is derived for each single crop category (see coeffgen\( \text{\textbackslash \text{cropping.gms}} \)):

\[
\begin{align*}
\text{p_nNeed} & \text{(crops, soil, till, intens, nut, t)} = \sum_{\text{prods}} \text{p_0Coeff}\text{(crops, soil, till, intens, prods, t)} \times (\text{p_nContent}(\text{crops, prods, nut}) \times \text{10});
\end{align*}
\]

taking the specific N contents of grain and straw as well as the yield level per ha into account (taken from the German Düngeverordnung, Appendix 1 of §3 Abs.2 Satz1 Nr.1).

The nutrient needs are linked to the different cropping intensities:

```plaintext
set intens / normal "Full N fertilization"
     fert80p "80 % N"
     fert60p "60 % N"
     fert40p "40 % N"
     fert20p "20 % N"
/;
```

based on nitrogen response functions from field trials (see coeffgen\( \text{\textbackslash \text{cropping.gms}} \)).
The output coefficients are used to define the nutrient uptake by the crops $p_{\text{nutNeed}}$ based on the nutrient content defined above:

\[
\begin{align*}
  \text{p\_nutNeed}(\text{crops}, \text{soil}, \text{till, intens}, \text{nut}) & = \text{sum}(\text{soil\_plot(\text{soil, plot, till, intens)}}, \ c\_s\_t\_i(\text{crops, plot, till, intens})) \\
  & \times \text{p\_0Coeff(\text{crops, soil, till, "normal", prod, t})} \times (\text{p\_nutContent(\text{crops, prod, nut})} + 10)
\end{align*}
\]

The curve suggests that with a 53% of the yield, only 20% of the N dose at full yield is necessary. Assuming a minimum nutrient loss factors, that allows defining how much nitrogen the crop takes up from other sources (mineralisation, atmospheric deposition):

\[\text{p\_basNut(\text{crops, soil, till, nut, t}) = \text{sum}(\text{prod, p\_0Coeff(\text{crops, soil, till, "normal", prod, t})} \times \text{sum\_\text{\(W\)}}(\text{\text{\(W\)})}) \times (\text{p\_nutNeed(\text{crops, soil, till, "normal", nut, t})} + 20\%)) \times (\text{p\_nutApplied(\text{crops, soil, till, intens, nut, t})} + 10\%)
\]

The amount of nutrient applied $p_{\text{nutApplied}}$ is estimated as follows, assuming that at least 20% of the default leaching and NH3 losses will occur:

\[\text{p\_nutApplied(\text{crops, soil, till, intens, nut, t}) = \text{sum}(\text{p\_nutSyntAppLossShare(\text{\text{\(W\)})}}) + 20\%)
\]

The nutrient application $p_{\text{nutApplied}}$ together with the basis delivery $p_{\text{basNut}}$ from soil and air allows defining the loss rates for each intensity level $p_{\text{nutSyntAppLossShare}}$ as the difference between the deliveries and the nutrient uptake $p_{\text{nutNeed}}$ by the plants:

\[\text{p\_nutSyntAppLossShare(\text{\text{\(W\)})}} = \text{sum}(\text{p\_nutNeed(\text{crops, soil, till, intens, nut, t})}) - \text{p\_nutApplied(\text{crops, soil, till, intens, nut, t})}
\]

To reflect typical cropping practises, a minimum share of mineral fertilizer can be set, e.g. to reflect quality fertilization:

\[\begin{align*}
  & \text{\(\text{\text{\(W\)}}\)} \text{\text{\(\text{\(W\)}}\))} = \text{sum}(\text{p\_nutNeed(\text{crops, soil, till, intens, nut, t})}) + \text{p\_nutApplied(\text{crops, soil, till, intens, nut, t})}
\end{align*}\]

\[\text{\(\text{\(W\)}}\) - \text{\(\text{\(W\)}}\) = \text{p\_nutSyntAppLossShare(\text{\text{\(W\)})}})
\]

2.11.2 Standard nutrient fate model

The standard nutrient fate model defines the necessary fertilizer applications based on yearly nutrient balances for each crop category ($\text{NutBalCrop}_\text{\(W\)}$). The LHS defines the nutrient need plus planned additionally losses from manure application $v_{\text{nutSurplusField}}$, 

\[
\begin{align*}
  & \text{\(\text{\(W\)}}\) \text{\(\text{\(W\)}}\) = \text{p\_nutNeed(\text{crops, soil, till, intens, nut, t})} + \text{p\_nutApplied(\text{crops, soil, till, intens, nut, t})} + \text{v_{\text{nutSurplusField}}}
\end{align*}\]
the right hand the deliveries from mineral and manure application net of losses plus deliveries from soil and air:

```cpp
The template model

--- definition of N balance for crops and N surplus restriction

\[
\text{NetNut}_{\text{crop,}t_c} = (\text{N}_{\text{crop,}t_c} - \text{N}_{\text{crop,}t_c}) \text{ Nut}_{\text{crop,}t_c} \\
\text{NetNut}_{\text{crop,}t_c} = \text{N}_{\text{crop,}t_c} - \text{N}_{\text{crop,}t_c}
\]

--- nutrient surplus max over per ha restriction

\[
\text{nutSurplusField}_{\text{crop,}t_c} = (\text{N}_{\text{crop,}t_c} - \text{N}_{\text{crop,}t_c}) \text{ Nut}_{\text{crop,}t_c} \\
\text{nutSurplusField}_{\text{crop,}t_c} = \text{N}_{\text{crop,}t_c} - \text{N}_{\text{crop,}t_c}
\]

The “unnecessary” \( v_{\text{nutSurplusField}} \) be restricted for each crop type based on maximal per ha “unnecessary” losses:

--- kg N losses in one month through application

\[
\text{nutLossApp}_{\text{crop,}t_c} = \text{N}_{\text{crop,}t_c} - \text{N}_{\text{crop,}t_c}
\]

--- mineral N application times unity minus specific loss rate for each application technology

\[
\text{nutLossApp}_{\text{crop,}t_c} = \text{N}_{\text{crop,}t_c} - \text{N}_{\text{crop,}t_c}
\]

--- mineral N application times unity minus specific loss rate for mineral fertilizer type

\[
\text{nutLossApp}_{\text{crop,}t_c} = \text{N}_{\text{crop,}t_c} - \text{N}_{\text{crop,}t_c}
\]

In the standard nutrient fate model, reductions in nutrient soil can be achieved:

1. by reducing unnecessary manure applications which decrease \( v_{\text{nutSurplusField}} \)
2. by reducing the cropping intensity, which not only reduces the overall nutrient needs and therefore the losses, but also reduces the loss rates per kg of synthetic fertilizer
3. by switching between mineral and organic fertilization
4. by changing the cropping pattern

44
The reader should note that nutrients applied from manure are net of losses during storage.

2.11.3 Detailed nutrient fate model by crop, month, soil depth and plot

The detailed soil accounting module considers the nutrient flows both from month to month and between different soil layers (top, middle, deep). It replaces the equations used in the standard nutrient fate model shown in the section above. The central equation is the following:

```plaintext
Considered input flows are:

1. Application of organic and mineral fertilizers net of NH3 and other gas losses at application, they are brought to the top layer.
2. Atmospheric deposition (to the top layer)
3. Net mineralisation
4. Nutrient leaching from the layer above

The considered output flows are:

1. Uptake by crops
2. Leaching to the layer below
```
The template model

The difference between the variables updates next month’s stock based on current month’s stock. Monthly leaching to the next deeper soil layer $v_{\text{nutLeaching}}$ is determined as a fraction of plant available nutrients (starting stock plus inflows):

```plaintext
outleaching(crops,soil,till,interc,nt,depth,t0Cur(t),m,s,w) \leq c_s_t_i(crops,soil,till,interc) ..

v_{\text{nutLeaching}}(crops,soil,till,interc,nt,depth,t,m,s,w)
```

```plaintext
\begin{align*}
\text{E} \{ \\
& \left[ (u_{\text{nutInSoilDepthStart}}(crops,soil,till,interc,nt,depth,t,s)) \leq (\text{sameas}(m,"Jan")) \leq t0Cur(t-1)) \\
& \quad \left[ u_{\text{cropHa}}(crops,soil,till,interc,t,s) \leq \text{sameas}(m,"Jan")) \leq (\text{not t0Cur(t-1))) \\
& \quad \left[ u_{\text{nutInSoilDepth}}(crops,soil,till,interc,nt,depth,t,m,s,s) \leq (\text{not sameas}(m,"Jan")) \\
& \quad \left[ \text{atmospheric deposition and mineralization} \\
& \quad \left[ u_{\text{cropHa}}(crops,soil,till,interc,t,s) \leq \{ \\
& \quad \left[ \text{atmospheric deposition by month and per ha} \\
& \quad \left[ p_{\text{atmDry}}(nt,m,w) \leq \text{sameas(depth,"top")} \\
& \quad \left[ \text{mineralization per ha in specific soil strata by month and weather} \\
& \quad \left[ p_{\text{mineral}}(soil,depth,nt,m,w) \\
& \quad \left[ \text{application of manure and synthetic fertilizer} \\
& \left[ \text{sum} \left( \text{Manapplication.mantype.manType,manType,manType,nt,s,m} \right) \right. \\
& \left. \text{sameas(depth,"top")} \right) \\
& \left[ \text{mineral N application times unity minus specific loss rate for mineral fertilizer type of N3} \\
& \left[ \text{sum} \left( \text{N3fertilizer}, \\
& \left. \text{sameas(depth,"top")} \right) \\
& \left[ \text{nutrient leaching from soil level above} \\
& \left[ u_{\text{nutLeaching}}(crops,soil,till,interc,nt,depth,t,m,s,w) \\
& \left[ \text{p_leachFactor}(soil,m,w) \\
\end{align*}
```

The leaching losses below the root zone in combination with ammonia and other gas losses from mineral and organic fertilizer applications define the total nutrient losses at farm level in each month:

```plaintext
\begin{align*}
\text{E} \{ \\
& \left[ \text{nutrient leaching below root zone} \\
& \left[ u_{\text{nutLeaching}}(crops,soil,till,interc,nt,depth,t,m,s,w) \leq \text{sameas(depth,"root")} \\
& \left[ \text{N3 losses from manure} \\
& \left[ \text{sum} \left( \text{Manapplication.mantype.manType,manType,manType,nt,s,m} \right) \right. \\
& \left. \text{sameas(depth,"root")} \right) \\
& \left[ \text{mineral N application times unity minus specific loss rate for mineral fertilizer type of N3} \\
& \left. \text{sameas(depth,"root")} \right) \\
& \left. \text{p_leachFactor}(soil,m,w) \left[ \text{t/365(w)} \right. \\
& \left. \text{p_prob}(w) \right} \\
\end{align*}
```

The approach requires defining the nutrient needs of the crops in different months, which is currently estimated:
Similarly, the update from different soil layer must be set:

```plaintext
set depthCrops(crops) / winterCore,summerCore,winterRape /;

* --- nutrient update need by soil type is a pure guess
* p_uptakeDistByDepth(depthCrops,depth,n) = 0;

p_uptakeDistByDepth(depthCrops,depth,"top","FEB") = 1;
p_uptakeDistByDepth(depthCrops,depth,"top","SEP") = 1;
p_uptakeDistByDepth(depthCrops,depth,"top","OCT") = 1;

p_uptakeDistByDepth(depthCrops,depth,"top","MAR") = 0.9;
p_uptakeDistByDepth(depthCrops,depth,"middle","MAR") = 0.1;

p_uptakeDistByDepth(depthCrops,depth,"top","APR") = 0.88;
p_uptakeDistByDepth(depthCrops,depth,"middle","APR") = 0.15;
p_uptakeDistByDepth(depthCrops,depth,"deep","APR") = 0.05;

p_uptakeDistByDepth(depthCrops,depth,"top","MAY") = 0.78;
p_uptakeDistByDepth(depthCrops,depth,"middle","MAY") = 0.28;
p_uptakeDistByDepth(depthCrops,depth,"deep","MAY") = 0.18;

p_uptakeDistByDepth(depthCrops,depth,"top","JUN") = 0.65;
p_uptakeDistByDepth(depthCrops,depth,"middle","JUN") = 0.25;
p_uptakeDistByDepth(depthCrops,depth,"deep","JUN") = 0.18;

p_uptakeDistByDepth(depthCrops,depth,"top","JUL") = 0.58;
p_uptakeDistByDepth(depthCrops,depth,"middle","JUL") = 0.30;
p_uptakeDistByDepth(depthCrops,depth,"deep","JUL") = 0.18;

p_uptakeDistByDepth(depthCrops,depth,"top","AUG") = 0.60;
p_uptakeDistByDepth(depthCrops,depth,"middle","AUG") = 0.30;
p_uptakeDistByDepth(depthCrops,depth,"deep","AUG") = 0.18;

p_uptakeDistByDepth(depthCrops,depth,n) $ p_uptakeDistByDepth(depthCrops,depth,n) = 1 / sum(depth1, p_uptakeDistByDepth(depthCrops,depth1,n));
```

A weakness of the current approach is the handling of changes in cropping patterns from year to year. It would be favourable to define the transition of nutrient pools from year to year based on a “crop after crop” variable in hectares for each soil type. However, that
leads to quadratic constraints which failed to be solved by the industry QIP solvers (they do not allow for equality conditions where are by definition non-convex). Instead, now the pool is simply redistributed across the crops and a maximum content of 50 kg of nutrient per soil depth layer is fixed.

If the crop rotations are switched on, a further restriction is switched on:

```plaintext
if(cropRotation == true)
    nPoolRot(p, cropTypes, plot, plotType)
        $ tPoolMax(plot, soil, intens, depth, tCur(t), s) \leq nPoolMax{plot, soil, intens, depth, tCur(t), s}$
```

\[ \sum_{c \in \text{crops}} \text{nNutInSoilDepthStart}(crops, soil, intens, depth, nut, t, s) \]

\[ nPoolMax{soil, crops, soil, depths, nut, tCur(t), s} \leq \text{nPoolMax}{crops, soil, intens, depth, nut, t, s} \times 50; \]

2.12 Biogas module

The biogas module defines the economic and technological relations between components of a biogas plant with a monthly resolution, as well as links to the farm. Thereby, it includes the statutory payment structure and their respective restrictions according to the German Renewable Energy Acts (EEGs) from 2004 up to 2014. The biogas module differentiates between three different sizes of biogas plants and accounts for three different life spans of investments connected to the biogas plant. Data for the technological and economic parameters used in the model are derived from KTBL (2014) and FNR (2013). The equations within the template model related to the biogas module are presented in the following section.

2.12.1 Biogas economic part

The economic part describes at the one hand the revenues stemming from the heat and electricity production of the biogas plant, and at the other hand investment and operation costs. The guaranteed feed-in tariff \( p_{\text{priceElec}} \), paid to the electricity producer per kWh, and underlying the revenues, is constructed as a sliding scale price and is exemplary shown in the next equation.
The template model

\[
p_{\text{priceElec}}(\text{bhkw, eeg, tcur}(t)) \cdot \text{(eegg rated(eeg))} = (p_{\text{priceElecBase}}(\text{"150 kWh"}, \text{eeg}) \cdot (150/p_{\text{powRate}}(\text{bhkw, eeg})) \]
\[
+ p_{\text{priceElecDiff}}(\text{bhkw, eeg}) \cdot ([p_{\text{powRate}}(\text{bhkw, eeg}) - 150]/p_{\text{powRate}}(\text{bhkw, eeg})) \]
\]

\[p_{\text{priceElecBase}}\], used to calculate the guaranteed feed-in tariff differentiated by size, includes the base rate and additional bonuses\(^2\) according to the legislative texts of the EEGs. For the EEG 2012, it only contains the base rate. In addition, the guaranteed feed-in tariff is subject to a degressive relative factor \(p_{\text{priceElecDeg}}\) which differs between EEGs and describes price reductions over time. The \(p_{\text{priceElecBase}}\) is then used to calculate the electricity based revenue of the biogas operator by multiplying it with the produced electricity \(v_{\text{prodElec}}\). In order to assure a correct representation of the EEG 2012 payment, the biogas module differentiates the electricity output by input source \(v_{\text{prodElecCrop}}\) and \(v_{\text{prodElecManure}}\) and multiplies it with its respective bonus tariffs \(p_{\text{priceElecInputClass}}\) which are added to the base rate.

In addition to the "traditional" guaranteed-feed in tariff, the biogas module comprises the payment structure for the so-called "direct marketing option" which was implemented in the EEG 2012. The calculation of the revenue with a direct marketing option is defined as the product of the produced electricity \(v_{\text{prodElec}}\) and the sum of the market premium \(p_{\text{dmMP}}\) and the price at the electricity spot exchange EPEX Spot \(p_{\text{dmsellPriceHigh/Low}}\) depending on the amount of electricity sold during high and low stock market prices. Additionally, the flexibility premium \(p_{\text{flexPrem}}\) is accounted for.

Further, the revenue stemming from heat is also accounted for and is included as the product of sold heat \(v_{\text{sellHeat}}\) times the price of heat, which is set to two cents per kWh. The amount of heat sold is set externally and depends on the biogas plant type.

The detailed steps of the construction of prices can be seen in \(\text{coeffgen\_prices\_eeg.gms}\).

2.12.2 Biogas inventory

The biogas plant inventory differentiates biogas plants by size (set \(\text{bhkw}\)), which determines the engine capacity, the investment costs and the labour use. Three size classes are currently depicted. Further, in order to use a biogas plant, different components need to be present which differ by lifetime (investment horizon \(ih\)). For example, in order to use the original plant, the decision maker has to re-invest every seventh year in a new engine, but only every twentieth year in a new fermenter.

\(^2\) For the EEG 2004: NawaRo-Bonus, KWK-Bonus; For the EEG 2009: Nawaro-Bonus, KWK-Bonus or NawaRo-Bonus, KWK-Bonus and Manure-Bonus

49
The template model

The biogas plant and their respective parts can either be bought \( v_{buyBiogasPlant}(Parts) \) or an already existing biogas plant can be used \( p_{iniBioGas} \). Both define the size of the inventory of the biogas plant \( v_{invBioGas}(Parts) \). The model currently limits the number of biogas plants present on farm to unity.

Furthermore, the inventory \( v_{invBioGas} \) determines the EEG under which its plant was original erected, either by externally setting the EEG for an existing biogas plant or the initial EEG is endogenously determined by the year of investment. In addition, the module provides the plant operator the option to switch from the EEG under which its plant was original erected to newer EEGs endogenously, such that the electricity and heat price of the newer legislation determines the revenues of the plant. For this purpose, the variable \( v_{switchBiogas} \) transfers the current EEG from \( v_{invBioGas} \) to the variable \( v_{useBioGasPlant} \). Hence, the \( v_{invBioGas} \) is used to represent the inventory while \( v_{useBioGasPlant} \) is used to determine the actual EEG under which a plant is used, i.e. payment structures and feedstock restrictions.

2.12.3 Production technology

The production technology describes not only the production process, but also defines the limitations set by technological components such as the engine capacity, fermenter volume.
The template model and fermentation process. As heat is only a by-product of the electricity production and therefore the production equations do not differ from those for electricity, the heat production is not explicitly described in the following documentation.

The size of the engine restricts with \( p_{\text{fixElecMonth}} \) the maximal output of electricity in each month. According to the available size classes, the maximal outputs are 150kW, 250kW and 500kW, respectively, at 8,000 operating hours per year, i.e. the biogas plant is not operating for 9% of the available time, e.g. for maintenance.

\[
\text{--- Fixing the maximum electricity produced by the biogas plant differentiated by plant size as well as accounting for transformation losses}
\]
\[
\text{fixkwel}\_\text{(cur}\_\text{bkw}(\text{bkw}),\text{cur}\_\text{eeg}(\text{eeg}),\text{tcur}(\text{t}),\text{m})\ldots
\]
\[
\text{v}_{\text{prodElec}}(\text{bkw},\text{eeg},\text{t},\text{m}) = \text{i\_useBioGasPlant}(\text{bkw},\text{eeg},\text{t}) + p_{\text{fixElecMonth}}(\text{bkw},\text{m});
\]

The production process of electricity \( v_{\text{prodElec}} \) is constructed in a two-stage procedure. First, the biogas \(^3\) \( v_{\text{methCrop/Manure}} \) is produced in the fermenter as the product of the inputs \( v_{\text{usedCrop/Man}} \) and the amount of methane content per ton fresh matter of the respective input. Second, the produced methane is combusted in the engine in which the electricity output \( v_{\text{prodElecCrop/Manure}} \) is calculated by the energy content of methane \( p_{\text{ch4Con}} \) and the conversion efficiency of the respective engine \( p_{\text{bhkwEffic}} \).

\[
\text{--- Stage I}
\]
\[
\text{methCrop}\_\text{(cur}\_\text{bkw}(\text{bkw}),\text{cur}\_\text{eeg}(\text{eeg}),\text{tcur}(\text{t}),\text{m})\ldots
\]
\[
v_{\text{methCrop}}(\text{bkw},\text{eeg},\text{t},\text{m}) = \text{sum}(\text{crm}(\text{biogas}\_\text{seewd}(\text{eeg})), \text{v}_{\text{usedCropBioGas}}(\text{bkw},\text{eeg},\text{crm},\text{t},\text{m}) * p_{\text{crpm}}(\text{crm}));
\]
\[
\text{--- Stage I}
\]
\[
\text{methManure}\_\text{(cur}\_\text{bkw}(\text{bkw}),\text{cur}\_\text{eeg}(\text{eeg}),\text{tcur}(\text{t}),\text{m})\ldots
\]
\[
v_{\text{methManure}}(\text{bkw},\text{eeg},\text{t},\text{m}) = \text{sum}(\text{crm}(\text{manType}(\text{eeg})), \text{v}_{\text{usedManureBioGas}}(\text{bkw},\text{eeg},\text{man},\text{t},\text{m}) * p_{\text{manure}}(\text{man}));
\]

\[
\text{--- Stage II}
\]
\[
\text{kwel}\_\text{(cur}\_\text{bkw}(\text{bkw}),\text{cur}\_\text{eeg}(\text{eeg}),\text{tcur}(\text{t}),\text{m})\ldots
\]
\[
v_{\text{prodElec}}(\text{bkw},\text{eeg},\text{t},\text{m}) = v_{\text{prodElecCrop}}(\text{bkw},\text{eeg},\text{t},\text{m}) + v_{\text{prodElecManure}}(\text{bkw},\text{eeg},\text{t},\text{m});
\]

Recall: The bonus structure of the EEG 2012 requires a differentiation between two input classes. Thus, the production process is separated in methane produced from the Crop input class and the Manure input class.

The production technology imposes a second bound by connecting a specific fermenter volume \( p_{\text{volFermMonthly}} \) to each engine size. The fermenter volume is exogenously given under the assumption of a 90 day hydraulic retention time and an input mix of 70 percent maize silage and 30 percent manure. Hence, the input quantity derived from crops

\(^3\) Biogas is a mixture of methane (\( \text{CH}_4 \)), carbon dioxide (\( \text{CO}_2 \)), water vapor (\( \text{H}_2\text{O} \)) and other minor gases. The gas component containing the energy content of biogas is methane. Thus, the code with respect to production refers to the methane production rather than the production of biogas.
The template model $v_{\text{usedCropBiogas}}$ and manure $v_{\text{usedManBiogas}}$ is bound by the fermenter size $v_{\text{totVolFermMonthly}}$.

The inputs for the fermentation process can be either externally purchased $v_{\text{purchCrop/Manure}}$ or produced on farm $v_{\text{feedBiogas/v_volManBiogas}}$. Further, the module accounts for silage losses for purchased crops, as crops from own production already includes silage losses in the production pattern of the farm. Currently, the model includes only cattle manure, maize silage and grass silage as possible inputs.

The third bound imposed by the production technology is the so called digestion load ("Faulraumbelastung"). The digestion load $p_{d\text{igLoad}}$ restricts the amount of organic dry matter within the fermenter to ensure a healthy bacteria culture. The recommended digestion load of the three different fermenter sizes ranges from $2.5$ to $3 \frac{kg\ oDM}{m^3 \cdot d}$ and is converted into a monthly limit.

---

$4$ oDM = organic dry matter; $m^3$ = cubic meter; $d$ = day

52
The data used for the fermenter technology can be seen in `coeffgen/fermenter_tech.gms`

2.12.4 Restrictions related to the Renewable Energy Act

Within the legislative text of the different Renewable Energy Acts, different restrictions were imposed in order to receive certain bonuses or to receive any payment at all. In the biogas module, most bonuses for the EEG 2004 and EEG 2009 are inherently included such as the KWK-Bonus and NawaRo-Bonus, i.e. the plant is already defined such that these additional subsidies on top of the basic feed-in tariff can be claimed. Additionally, the biogas operator has the option to receive the Manure-Bonus, if he ensures that 30 percent of his input quantity is manure based, as seen in the following code.

Further, the EEG 2012 imposes two requirements which have to be met by the plant operator to receive any statutory payment at all. First, the operator has to ensure that not more than 60 percent of the used fermenter volume $v_{totVolFermMonthly}$ is used for maize. Second, under the assumption that the operator uses 25 percent of the heat emitted by the combustion engine for the fermenter itself, he/she has to sell at least 35 percent of the generated heat externally;
3 Dynamic character of FARMDYN

3.1 Fully dynamic version

As already denoted in earlier sections, the model template will optimize the farm production process over time in a fully dynamic setting. Connecting the different modules over time \((t_1-t_n)\) allows for a reproduction of biologic and economic path dependencies.

Following the illustration above, the dynamic examinations of the template modules are of different character. For example herd management and cropping decisions are annually implemented. Through consideration of a longer time horizon, also time lags have to be incorporated. Naturally given, breeding activities have a time lag of three years, born calves entering the milk producing herd three years after birth. In terms of fodder composition, decision points during the year are allocated every three month, offering the decision maker a more flexible adjustment to feed requirements of the herd (conditional on lactation phase) and actual sources and prices of pasture, silage and concentrates. Furthermore as stated before, the application of manure or synthetic fertilizers, as well as the stored manure amounts on farm are implemented on monthly level.

A peculiar feature of the dynamic approach is that the dynamic calculation of an optimal production plan over time is not simulated from year to year but that all variable values of the planning horizon are optimised at once. This means not only that decision points now impact the effects and profitability of possible development paths in the future, but that also back-loops are implemented, meaning that future decisions impact the degree in optimality of nowadays decisions. Hence, regarding the dynamic optimisation procedure of the model, one can assume a decision maker fully informed over the specific planning horizon. Regarding the results of farm development by the simulation runs, the optimised
farm development paths can be interpreted as best practice solutions within the predefined restrictions.

### 3.2 Short run and comparative static version

The short run version only considers one year and does not comprise a liquidation of the enterprise. The comparative-static version replaces the herd dynamics by a steady state model where e.g. the cows replaced in the current year are equal to the heifers in the current year, which in turn are equal to the calves raised in the current year. In comparative static mode, there is no longer a vintage model for investments in buildings and machinery, instead, the investment costs are related to one year. Still, the binary character is maintained.

### 3.3 States of nature (SON)

The models allows farmers to adjust their cropping pattern, feeding, off-farm labour use on a hourly basis and further farm activities to SONs, whereas decisions with a long-term character (full or half time work off farm, investment decisions, herd size, renting out of land) cannot be adjusted to a specific SON. Due to the long-term character of the model, SONs relate to the farms market environment, specifically the input and output prices, wages and interest rates faced by the firm, and not to short term fluctuations of e.g. yields.

There is hence a deterministic relation between input use and outputs. The differentiation between state-of-nature specific decisions (cropping, feeding) and annual decisions is depicted in the diagram below.

**Figure 5. Systematic view on the model approach**

Own illustration

---

\[ r = \text{endowments at start of year (such as land, stable places, machinery park, liquidity ..)} \]
\[ a = \text{i/o, factor demand /delivery coefficient of decision variables} \]
\[ \varepsilon = \text{cost/revenues related to decision variables} \]
\[ \text{SON}_n = \text{States of nature} \]
3.4 Objective function

We assume that a risk neutral profit maximizing farmer with a given discount rate used to calculate the expected net present values of profits minus household expenditures. The firm is assumed to be liquidated at the end of the planning horizon, i.e. the cow herd, machinery, land are sold and credits paid back. Any remaining equity is discounted to net present value, so that a definition close to flow to equity is used:

\[
\text{OBJE_} = \sum_t \left( (t.\text{pos} \equiv \text{card}(t)) \right),
\]

\[
\begin{align*}
\text{v_\text{obje}} &= \text{v_\text{liquid}(t)} \\
\text{v_\text{liquid}(t)} &= \text{discount factor for full period, and then the mean} \\
&= (1-p_{\text{discountRate}}/100)^{(t.\text{pos})}/\text{card}(t)
\end{align*}
\]

Further on, a special character, due to model construction and choice of fully dynamic mixed integer optimization, is that the decision maker is assumed to be fully informed of future states of nature. Hence, model results always show the optimal farm plan over the chosen planning horizon with given assumed future states of nature (best-practice simulations).
4 GHG accounting

The calculation of the farm specific greenhouse gases is settled on different specified GHG indicators (GHG calculation schemes). The indicators differ in degree of aggregation and feasibility of required production data. The derivation procedures are described separately in the following sections for all indicators and divided concerning gas type.

Short conceptual explanation of the implemented indicator schemes:

The different indicators are mainly based on the IPCC (2006) guidelines which comprise so-called tiers of increasing complexity to calculate GHG emission. Tier 1 provides the simplest approach to account emissions using default parameters e.g. per animal. We use Tier 1 as far as possible to define our simplest indicator termed actBased, where emission factors are linked to herds and crop hectares, only. The exemptions from the IPCC methodology are manure management and fertilization where IPCC links emission factor to organic and synthetic fertilizer amounts. We thus assume average excretion and fertilizer application rates to derive per animal or per ha coefficients.

A somewhat more complex indicator called prodBased links emission factors to production quantities of milk and crop outputs. Generally, at the assumed average yields, the two indicators yield the same overall emissions. Compared to the activity based indicator, farmers have somewhat more flexibility as they might e.g. switch between different grass land management intensities to abate emissions.

A more complex and also presumably very accurate indicator is called NBased. Values for enteric fermentation are calculated from the requirement functions, for energy based on IPCC guidelines, which also drive the feed mix. For manure management, emissions are linked to the amount of manure N in specific storage types in each month. For fertilization, the emission factors are linked to distributed nitrogen differentiated by application technique. The indicator thus gives the farmer the chance to abate nitrogen losses by changing storage types, storage periods or the fertilization application technique, beside changes in herd sizes, herd structure or the cropping pattern.

An intermediate indicator linked to production quantities is called genProdBased. The emission factors are linked mainly to output quantities and as far as possible derived from the NBased one. The differences stem from the calculation of emissions from enteric fermentation and manure management: the Tier 1 default values are either directly replaced by Tier 3 values (enteric fermentation) or Tier 3 values for manure management are used in conjunction of assumed shares for manure storage and application. Specifically, the indicator introduces milk yield dependent emission factors which reflect that higher milk yields reduce per litre emissions by distribution the maintenance and activity need of the cow over a larger milk quantity.

The most complex and accurate indicator scheme is the reference indicator called “refInd”. It is an enhancement of the NBased indicator. The difference is, that real feed intake of different feed compounds is recognized to implement impact of feed digestibility on emissions from enteric fermentation. Further on, the addition of fats and oils to the ration is recognized to heighten digestibility and lower methane from ruminant processes.

The different indicator calculation schemes (except of the NBased and refInd) are listed in a sub-module named “indicators”. In this module, some basic parameters and scale definitions for later emission calculation formulas are explained taken from IPCC (2006)
(chapter 10 and 11), DÄMMGEN (2009), VELTHOF and OENEMA (1997) and the RAINS model definition (for changes in $\text{NO}_x$ and $\text{NH}_3$ fluxes concerning manure application type) (ALCAMO et al., 1990).

```
* parameter and scale definition for calculation of indicators:
  * parameters for CH4 following IPCC 2006 (Tier1 and Tier2)
  * parameter $p_{A2}$: "each content of manure, normally 0.08 for cattle"
    / p_A2 0.08 /
  * parameter $p_{BoVla}$(be)" maximum methane producing capacity for manure produced by livestock in m$^3$CH4/kg"  
    / COVBE 0.24 
    heife 0.18 
    fokirSim 0.18 
    freisvorSim 0.18 
    ncowv 0.18 ;
  * parameter $p_{MCF}$(meanStorage): "methane conversion factors for each manure management system, 0.17 means 17%"  
    / storred 0.17 
    storncov 0.17 
    storestraw 0.10 
    storetoil 0.0 ;
  * parameter $p_{Bu}$(Bact)" methane conversion factor, table 10.12 in IPCC, 2006)"  
    / cow 6.5 
    heife 6.5 
    ovino 6.5 ;
  * parameters for N2O following IPCC 2006 (Tier1 and Tier2)
  * table 10.21
  * parameter $p_{EF}$ (meanStorage) "emission factor for direct N2O emissions from manure management system, kg N2O-N/kgN in manure system"  
    / stornd 0.002 
    storncov 0.000 
    storestraw 0.001 
    storetoil 0.0 ;
  * table 10.22
  * parameter $p_{fman}$ (meanStorage) "fraction of managed manure nitrogen for inventories that volatilizes as $\text{NH}_3$ and $\text{NO}_x$ in the manure management sys"  
    / stornd 0.10 
    storncov 0.15 
    storestraw 0.10 
    storetoil 0.0 ;
  * parameter $p_{fNH4}$ "emission factor for N2O emissions from atmospheric deposition of nitrogen on soils and water"  
    / (CH4) /
  * parameter $p_{VfMan}$ (meanStorage) "increase in N2O emissions for non-surface manure application (lower volatilization losses => more manure"  
    / applvspread 1.00 
    applvstraw 1.10 
    applvtoil 2.05 ;
  * parameter $p_{fRunOffH2O}$ "fraction of managed manure nitrogen losses due to runoff and leaching during storage of manure (i.e.60)-0.01 mean 51)"  
    / (CH4) /
  * parameter $p_{fRunOffN}$ "emission factor for N2O from nitrogen leaching and runoff in kg N2O-N/kgN leaching and runoff"  
    / (CH4) /
  * parameter $p_{EF}$ "emission factor for N2O from N inputs, kg N2O-N/kgNinput"  
    / (CH4) /
  * parameter $p_{fEf}$(Bact)" "emission factor for N2O from urine and dung N deposited on pasture, kgN2O-N/kgNinput":  
    / p_EfEffs(1.01) ;
  * parameter $p_{fRunOff}$ "fraction of synthetic fertilizer N that volatilises as $\text{NH}_3$ and $\text{NO}_x$ and N2O, kgN volatilised per kgNapplied. only relevant for N from sy"  
    / (CH4) /
  * parameter $p_{fRunoffN}$ (meanStorage) "fraction of applied organic N that enters the soil and deposited N by grazing animals that volatilizes as $\text{NH}_3$ and $\text{NO}_x$,  
    / applvSpread 0.20 
    applvstraw 0.15 
    applvtoil 0.04 ;
  * table 10.23
  * parameter $p_{fRunOff}$ "fraction of all N added to/externalised in managed soils in regions where leaching/runoff occurs that is lost through leaching an"  
    / (CH4) /
```
The above shown parameters and scalars are described by the statements within the model code excerpt. These are basic emission factors which are later linked to the indicator schemes representing source specific emission conversion factors or compound rates. Concerning the model run specifications made by the model user, different ways of GHG calculation are activated, depending on the chosen indicator \((ind)\). Hence, the resulting emissions by each simulated farm per year \((v_{\text{GHGEmissions}}(ind,t))\) are either calculated on per head or ha basis, per production quantity or on highly disaggregated way using the NBased or refInd indicator. The equation \(G\text{HG}_{\text{Emissions}}(*,*,*,*,*,*)\) facilitates an indicator, gas and source specific quantification of emissions according to the chosen indicator scheme \((v_{\text{GHGs}}(ind,emitters,gases,t))\). Therefore, \(p_{\text{GHGEmissions}}(*,*,ind,emitters,gases)\) are per head/ha or per production unit specific emission parameters inserted for calculation following actBased, prodBased and genProdBased indicator. Calculation procedures for the NBased and refInd indicator schemes are explained later.

For each indicator scheme different parameters per head, ha or per production quantity are defined in the indicator module (shown later) and will be implemented into the calculation.
formula above following the chosen indicator in the model run. Hence, if the NBased indicator is chosen, only the last summand is activated, leaving the above lines of the equation disregarded because the NBased indicator scheme does not hold any emission parameters per ha, head or per production quantity.

The detailed calculation schemes for the different chosen indicators are illustrated in the following subchapters.

4.1 prodBased indicator

Although it is not the simplest indicator scheme, at first the derivation of the production based (prodBased) indicator is described because other, more simple indicators base some calculations on prodbased calculation formulas, but with only taking average data.

The different GHG emission parameters $p\_GHGEmissions$ per production quantity are implemented in the submodule “indicators”. For idle, only the background emissions per ha are recognized. Emission parameters for all other crops are derived by using equation ... from IPCC (2006), calculating N inputs from yield level and average N content of crops ($p\_nContent$) multiplying this by the above shown standard emission factors for direct and indirect GHG emissions. Further on, also background emissions for arable soils are considered ($p\_backCH4Soil(crops)$, $p\_backN2OSoil(crops)$). To get the per unit of production emission parameter ($p\_GHGEmissions(prods,...)$) the calculated total GHG amount per ha is divided by the output quantity per ha ($p\_OCoeff(crops,prods,t)$).

For dairy cows the GHG emission factor per kg of milk ($p\_GHGEmissions("milk",...)$) is derived by taking default emission factors from IPCC (2006) from enteric fermentation (117 kg CH$_4$) and manure management (21 kg CH$_4$, 1.4 kg N$_2$O) per animal and year and dividing the occurring CO$_2$-equ. by an average milk yearly milk yield per cow of 6000 kg.

For emission from heifers and calves, only per head default emission factors are taken (also from IPCC). For emission parameter calculation of male calves and sold female calves, the residence time on farm is recognized (14 days on average). To calculate the...
default values of N₂O emissions from herds, calculation functions 10.25, 10.26 and 10.30 of IPCC are filled with average weights (e.g. cow: 650kg) and excretion rates of the different herd categories taken from KTBL (2010).

--- prodBased indicators for farms where the exemption of cows are calculated according to the IPCC 2006 Tier 1 approach using default values. Values of average weight 680kg and ex-fer (following KTBL2010):

(1) Method from enteric fermentation and manure storage (~21)
(2) CO₂ from manure storage (~9)

--- prodBased emissions (ch4, perProd, cowProd, mancowProd, emissions, gases) + \{ (10.25) \{ \{ mean(cow(enteric, enterance)) and mancow(gases, "CH4") \} + (10.26) \{ \{ mean(cow(enteric, manureStorage)) and mancow(gases, "N2O") \} \} /680 * (600)

--- default values of N₂O emissions from herds, calculation functions 10.25, 10.26 and 10.30 of IPCC are filled with average weights (e.g. cow: 650kg) and excretion rates of the different herd categories taken from KTBL (2010).

No differentiation in GHG emission rates of the farm are made concerning storage and application techniques of manure and synthetic fertilizers. Further on, storage time of manure as well differences in emissions from applied fertilizer and manure N are not recognized by this indicator.

4.2 actBased indicator

The actBased indicator denotes the simplest emission indicator implemented into the model approach of DAIRYDYN, it just multiplies activity data (ha or heads) with specific default emission factors which are taken from IPCC (2006) on Tier 1 level. E.g. 117 kg CH₄ per cow and year from enteric fermentation and 21 kg CH₄ per cow from manure management. For N₂O emissions per cow and year a default value of 1.4 kg is taken (see section prodBased indicator). For calves and heifers the default emission indicators are the same as for the prodBased indicator. Emission parameters from cropping activities are taken from the derivation scheme of the prodBased indicator taking average yield levels and fertilizer application rates.

--- copy per head coefficients

Also for grassland, (different pasture types) no differentiation concerning emission rates of different intensity levels is made.
4.3 genProdBased indicator

The indicator scheme called genProdBased also accounts emissions of cropping activities based on emission parameters per unit of product, taking the same equations as for the prodBased indicator.

For the calculation of emissions from heifers and calves, the Tier 2 approach from IPCC is used taking average weights (350kg for heifers and 90kg for calves) and occurring gross energy demands following KTBL (2010) and GE demands from Kirchgeuessner (2004) (resulting in average per head and year emissions of 1962 kg CH\(_4\)-equiv. for heifers and 504 kg CH\(_4\)-equiv. for calves). The gross energy demands of cow species are given by requirement functions implemented in the sub-module “requ.gms”, including functions for energy need for maintenance, activity, gross and lactation. GE demands used for that are calculated following equation from IPCC (2006) and taking a default digestibility of feed from IPCC of 60%.

--- list of specific emission factors for cows differentiated by milk yield for the genProdBased indicator

```pl
parameter p_GHEmissionsCowYield(cow, emitter, genus) * emission parameters per kg milk for different milk yield levels divided to source and genus;
```

--- copy prodBased, and then replace heifers and cows

```pl
p_GHEmissions(crop, "prodCowGenus", "genProdBased", emitter, genus) = p_GHEmissions(crop, "prodCowGenus", prodBased, emitter, genus);
```

```pl
p_GHEmissions(crop, "yearBase", "genProdBased", emitter, genus) = p_GHEmissions(crop, "yearBase", prodBased, emitter, genus);
```

--- for heifers using GE-dfemenal from Ericsson et al. (2004) and taking average weights of heifers (350kg) and average storage technique of heifers (can be changed if emission values do not fit any more)

```pl
p_GHEmissionsHeifer(crop, "yearBase", "genProdBased", emitter, genus) = ...
```

--- for calves using GE-dfemenal from Ericsson (2004) and taking average weights of calves (90kg) and average storage technique of calves (can be changed if emission values do not fit any more)

```pl
p_GHEmissionsCalves(crop, "yearBase", "genProdBased", emitter, genus) = ...
```

--- for cows using GE-dfemenal from Ericsson (2004) and taking average weights for cows (400kg), gross energy demands and storage technique of cows (can be changed if emission values do not fit any more)

```pl
p_GHEmissionsCowYield(crop, "yearBase", "genProdBased", emitter, genus) = ...
```

--- for heifers, two weeks on farm (14/365) but multiplied by 2 because one digestibility of feed is 40%, hence multiplying by 3 in to get real GE outputs

```pl
p_GHEmissionsHeiferCalves(crop, "yearBase", "genProdBased", emitter, genus) = ...
```

--- for calves, two weeks on farm (14/365) but multiplied by 2 because cow digestibility of feed is 40%, hence multiplying by 2 in to get real GE outputs

```pl
p_GHEmissionsCalves(crop, "yearBase", "genProdBased", emitter, genus) = ...
```
In case of emission calculations from dairy cows, also the genetic potential is considered. Therefore, the gross energy demand of each different genetic yield potential is considered separately. Also taking formulas from IPCC Tier2 approach lead to different per kg milk GE demand occurring from maintenance, lactation, gross and activity. The calculated emissions per cow are then divided by the milk yield potential, leading to a decline in GHG per kg milk for higher yield levels. For the calculation of emissions by dairy cows, a table with intensity dependent emission factors per kg milk is inserted into the module, displaying the parameter \( p_{\text{GHGEmissionsCowsYield}}(\text{cows}, \text{emitters}, \text{gases}) \). The parameter value of whole emissions per kg milk diminishes from 0.74 for a 4000 kg cow to 0.46 kg CO\text{2}eq/kg milk for a cow with 10000 kg milk output per year. But this decline in emissions per kg of milk with increasing milk yield does not occur on a linear way, illustrated by the following figure.

Figure 6. GHG emissions per cow and per kg of milk depending on milk yield potential

Hence, the contrast to the simple prodBased indicator is, that the emissions from herds are now based on real, output depending energy requirements resulting in different output level dependent emission factors per kg of milk and not a fix emission factor per kg of product disregarding the efficiency effects shown in the figure above.

4.4 NBased indicator

With difference to the other indicator definitions, the calculation formulas of the NBased indicator are not enclosed in the indicator module, but described in the basic template module. This is necessary because as it is a highly detailed and disaggregated indicator scheme, it calls for many model variables which differ between simulation steps. Hence it has to be defined in the basic template module.

As the NBased indicator up to now presents the reference indicator, the emissions which are emitted by each source of the production process (enteric fermentation, soils...) are calculated by this indicator in each simulation step. Hence, the GHG calculation is fragmented concerning the emission origins to credit the emissions to the single emitters.
Summing up the source specific emissions then lead to the overall GHGs $v_{\text{GHGs}}(\text{emitters, gases, t})$.

**CH$_4$ accounting**

The CH$_4$ emissions by enteric fermentation are calculated concerning equation 10.21 of IPCC (2006) guidelines taking the gross energy intake of each livestock category ($p_{\text{grossEnergyPhase}(...)}$) as basic variable to multiply it with the level $v_{\text{herdsize}}$ of each category and a specific methane conversion factor $p_{YM}$. By multiplying with factor 21 CH$_4$ emission amounts are converted to CO$_2$-equivalents.

For the accounting of methane from manure storage, the manure amount, storage period and differences in methane conversion factors ($p_{\text{MCF(manStorage)}}$) between different manure storage types are recognized. This is done following equation 10.23 of IPCC (2006). The monthly manure amount in each storage type $v_{\text{manInStorageType}(...)}$ is multiplied by specific dry matter content ($p_{\text{avDmMan}}$) and then by a methane emission factor (0.21 as a mix for cows and heifers) to get the m$^3$ of CH$_4$ occurring from manure. The parameter 0.67 then converts m$^3$ methane to kg methane. Because manure is quantified on monthly basis and CH$_4$ emissions are also implemented on monthly basis, the methane conversion factors $p_{\text{MCF(manStorage)}}$ have to be divided by 5.66 to not overestimate monthly methane emissions from manure $^5$.

The background emissions of methane from soils are derived by multiplying the crop specific $p_{\text{backCH4soil}}$ (taken from DÄMMGEN, 2009:p.315) emission parameter with the activity level ($v_{\text{cropHa(crops, t,s)}}$).

With the statement $\$(\text{sameas (emitters, "entFerm")})$, the calculated CH$_4$ emissions are credited to the source enteric fermentation. The same is done for all other gases (N$_2$O, CO$_2$) and sources (manStorage, backSoil, manApplic, syntApplic) to be able to analyse source specific emission developments during the simulation runs.

$^5$ Derived by Table 10.17 (continued) from IPCC (2006) guidelines to account for a monthly storage time of manure. Yearly MCF = 17, monthly MCF = 3; 17/3 = 5.66

---

64
For N\textsubscript{2}O emissions from manure storage a differentiation between direct and indirect N losses is made. N amount in specific storage type \(v_{N\text{in}StorageType(manStorage,t,m)}\) is multiplied by an emission factor for direct N\textsubscript{2}O-N flux from storage systems \(p_{EF3s}\) to get direct nitrous oxide emissions from storage systems in each month. Further on also indirect nitrous oxide emissions are considered following IPCC (2006) equations 10.27 and 10.29 to respect outgassing of NO\textsubscript{x} and NH\textsubscript{3} \(p_{FracGasMS(manStorage)}\) and leaching \(p_{FracLeachMS}\).

The next excerpt of the model code represents the detailed N\textsubscript{2}O derivation occurring from manure and synthetic fertilizer application on agricultural soils. Direct nitrous oxide emissions from soils calculated following equation 11.1 and relating auxiliary calculations from IPCC (2006). The nitrous oxide amounts produced by cropland \(N\text{2}O\text{Inputs}\) depends on the applied manure N amounts to the specific crops with different manure application techniques \(v_{nManDist(Crops,ManApplicType,t,s,m)}\) and the N amount fertilized by synthetic N \(v_{nSyntDist(Crops,syntNFertilizer,t,s,m)}\). Applied manure N is multiplied by an application specific increase factor \(p_{n2OIncreaseFact}\) which is higher for non-surface application techniques (taken from RAINS model). The sum of both is then multiplied by the N input dependent conversion factor for croplands, \(p_{EF1}\). The same is done for calculation direct N\textsubscript{2}O emissions from N deposited to pasture \(past33\) using the pasture specific \(p_{EF3prp(\text{"past33"})}\) conversion parameter.

Indirect N\textsubscript{2}O emissions from atmospheric deposition and leaching and runoff are calculating corresponding to equations 11.9 and 11.10 from IPCC (2006) guidelines.
The emission factor for background emissions of N\textsubscript{2}O from soils are however not taken from IPCC (2006) methodology because these are valued to high, stemming from a study basing on peat soils with very high volatilization rates. Because of that, we replaced it by background emission factors per ha of land \((p_{\text{back}\text{N}2\text{Osoil}})\) imposed by Velthof and Oenema (1997:p.351); 0.9 kg N\textsubscript{2}O-N per ha as already shown in the declaration of basic emission parameters before.

The parameters 44/28 and 310 which are multiplied with the calculated N\textsubscript{2}O-N amounts by each source are on the one hand the conversion factor from N\textsubscript{2}O-N to N\textsubscript{2}O (44/28) and the global warming potential (310)

\[
\text{CH}_4\text{emissions from intestine processes are calculated concerning the following formula:}
\]

\[
\text{refInd as well as the NBased indicator, the refInd calculations are not enclosed in the indicator module, but described in the basic template module. This is necessary because as it is the most detailed and disaggregated indicator scheme, it calls for many model variables which differ between simulation steps. Hence it has to be defined in the basic template module.}
\]

The reference indicator is the most disaggregated emission accounting scheme implemented in the model approach. It mainly bases on the calculation mechanisms of the former explained NBased indicator. Enhancements are made concerning the consideration of differences in feed compound digestibility as well as the addition of fat or oils and the occurring impact on methane emissions from ruminant fermentation. The calculation of emissions from soils, manure and fertilizer application and manure storage are equal to the accounting procedure of the NBased indicator.

Only differences in methane calculations occurring from enteric fermentation from the animals exist. The CH\textsubscript{4} emissions from intestine processes are calculated concerning the following formula:
The feed use by each animal \(v_{feeding}(\text{herds, phase...,})\) is multiplied by a specific emission parameter per kg of feed compound \(p_{feedsEmission}(\text{herds, feeds})\) depending on different ingredients of the component and the animal category.

Therefore emission parameters for different feed ingredients are calculated following the below stated routine. Per kg feed emission parameters are derived from IPCC equations basing on GE of specific feedstuff \(p_{feedAttr(feeds, 'GE')}\).

\[
\text{(5) refinement, recognition of digestibility of feed and addition of fats and oils}
\]

\[
\text{--- calculation of emission parameters per kg of feed}
\]

\[
\text{calculated by usage of IPCC equations for Energy intake if animals are fed only with single feeds}
\]

\[
\text{parameter } p_{feedsEmission}(\text{herds, feeds}) \quad \text{“kg methane emissions of methane per kg of feed”}
\]

\[
p_{feedsEmission}(\text{herds, feeds}) = p_{feedAttr(feeds, 'GE')} \times (6.6/100) \times 55.85 \times 1000
\]

Addition of fats and oils is implemented via percentage reduction of whole emissions per kg inserted in ration recognition of maximum addition of fats/oils of 5% of whole ration dry matter.

\[
\text{parameter } p_{oilEmissions} \quad \text{“emission reduction in percent per kg of oil in ration”} /0.02/;
\]

\[
\text{parameter } p_{fatEmissions} \quad \text{“emission reduction in percent per kg of fat in ration”} /0.02/;
\]

### 4.6 Source specific accounting of emissions

To enable the model user to directly explore emission amounts and gas types allocated to the different sources of the production process (enteric fermentation, manure CH\(_4\), manure N\(_2\)O, CH\(_4\) and N\(_2\)O from fertilizer or manure application and background emissions from soils), specific accounting functions in the model by the emission indicators are assigned to the sources and gas types. Hence, a detailed source specific emission analysis occurring form farm management changes is possible to allocate the production areas of the farm, where GHG mitigation efforts are allocated when lowering overall farm emissions.

To date, land use change, afforestation or change of tillage practices is not implemented into the overall model approach. This is the reason why up to now no CO2 accounting is implemented into the indicator schemes. For a higher resolution of land use practices and tillage procedures this has to be expanded due to potentials of carbon sequestration or
release from soils. Additionally, intensity and production depending CO2 emissions from fuel use could be implemented.
Derivation of Marginal Abatement Costs (MAC)

We define marginal abatement costs as the marginal profit loss of a firm due to a marginal reduction of an emission amount (DECARA & JAYET, 2001), in our cases of GHG emissions measured in CO\textsubscript{2} equivalents. For our highly detailed template model, no closed form representation of the abatement costs exist, so the MAC can only be simulated parametrically. Specifically, we derive the MAC by introducing a step-wise reduced constraint on maximal GHG emissions, and relating resulting changes in GHG emissions to the related profit loss (profit loss compared to baseline scenario. Lower emission constraint means new restrictive level which leads to monetary losses). As already mentioned in the introduction, the resulting MAC curve is depending on the indicator used to calculate the emissions, the abatement strategies open to the firm and, clearly, further firm attributes such as its market environment. With regard to the adoption of the farm to new emission constraints, the model template allows to analyse the amount and structure of chosen emission abatement options for each reduction level depending on farm type and chosen emission indicator. This will help for the structural analyses of the level of abatement cost depending on the effective abatement strategies which are biased by the specification of the emission indicator.

Under a given indicator, a stepwise reduction of the emission constraint will lead to a stepwise reduction in farm profits. Relating the change in emissions to the changes in profits allows calculating the total and marginal abatement cost. In the following, \( \varepsilon_{0j} \) are the emissions emitted measured with indicator \( j \) under the profit maximal farm plan without any emission target, where the zero characterizes the reduction level. To derive marginal abatement cost curves, \( n \) reduction steps, each with the same reduction relative to the base \( \varepsilon_{0j} \), will be taken, leading to objective values from \( \pi_{0j} \) to \( \pi_{nj} \). Let \( \alpha_i \) denote the relative reduction in step \( i \) compared to the baseline emissions. The maximal profit under reduction level of step \( i \) and indicator \( j \) is restricted by accounted emissions for the \( k \) decision variables according to:

\[
\sum_k e_{fjk} x_k \leq (1 - \alpha_i) \varepsilon_{0j}
\]

where \( e_{fjk} \) is the emission factor attached to decision variable \( k \) under indicator \( j \), i.e. the CO\textsubscript{2} equivalent emission accounted per unit of variable \( k \). The difference in profits between \( \pi_{0j} \) - the profit without a GHG restriction – and \( \pi_{ij} \) measures the profit foregone due to the specific emission ceiling – the combination of reduction level of step \( i \) and indicator \( j \). Hence the total abatement costs (AC) for the abatement of \( \alpha_i \varepsilon_{0j} \) emissions are defined as:

\[
AC_{ij} = \pi_{0j} - \pi_{ij}
\]

The change in profits from step \( (i - 1) \) to \( i \) is divided by the emission reduction from step to step to derive marginal abatement costs:

\[
MAC_{ij} = \frac{\pi_{i-1,j} - \pi_{ij}}{\varepsilon_{i-1,j} - \varepsilon_{ij}}
\]

\( MAC_{i,j} = \) marginal abatement costs for the reduction step from \( (i - 1) \)to \( i \), using the indicator \( j \)
Derivation of Marginal Abatement Costs (MAC)

\[ \pi_{i,j} = \text{value of objective function in simulation step } i, \text{ using indicator } j \]
\[ \varepsilon_{i,j} = \text{emission amount in simulation step } i, \text{ calculated with indicator } j \]
\[ i = \text{step of simulation} \]
\[ \alpha_i = \text{amount of total percentage reduction of emission in step } i \text{ compared to baseline} \]
\[ \varepsilon_{0j} = \text{baseline emission of the farm without emission restriction} \]
\[ \varepsilon_{fjk} = \text{total amount of GHG emissions related to one unit of activity } k \]

A stepwise reduction of the emission constraint leads to a sequence of changes in the farm program and related profit losses. From there, farmspecific MAC curves can be generated which plot changes in profits against the GHG reduction.

5.1 Normalization of MACs

As shown above, the GHG calculation schemes have significant differences in the detailed way of emission accounting. Hence, the occurring marginal abatement costs of GHG mitigation will be quite different. Furthermore, derived emission amounts for the very same farm production portfolio will not be the same for different indicators. Because of that, also the directly calculated MACs are not comparable between the indicators because the MACs depend on the indicator specific accounting rules.

Because of that, MACs have to be normalized to make them comparable. To do so, the so called reference indicator is taken, relating the mitigation cost induced by the single indicators to the “real” abated emission amounts calculated with the reference indicator. This will lead to results, enabling the model user to make statements concerning e.g. the cost efficiency in GHG abatement of different indicator schemes.

When comparing different emission indicators we face the problem that the MACs of each indicator relate to its specific GHG accounting rules. From a policy perspective, we would clearly need to know how much GHGs are physically released from the farm, in order to correctly assess costs and benefits, and not use the probably rather biased ones which are accounted by a specific indicator. In an ideal world, we would be able to derive the “real” GHG emissions from the farm program. As this is impossible, a so-called reference indicator will be constructed. It will use the best available scientific knowledge to derive from the farm program, i.e. based on all available decision variables, a total GHG emission estimate from the farm. The underlying calculation could be highly non-linear and complex and need not necessarily be integrated in the model template itself. Equally, it does not matter if it could be implemented in reality on a dairy farm given its measurement costs. It simply serves as a yard stick to normalize GHG emissions from different, simpler, but more realistic and applicable indicators. Relating profit losses under different indicators and indicator-specific GHG emission targets to the GHGs abatement under the reference indicator \( r \) at the simulated farm program allows deriving normalized, comparable marginal abatement cost curves:

\[ (2.2) \quad MAC_{i,j}^{\text{norm}} = \frac{\pi_{i-1,j} - \pi_{i,j}}{\varepsilon_{i-1,r} - \varepsilon_{i,r}} \]
\[ \varepsilon_{i,r} = \text{emission amount in simulation step } i, \text{ calculated with reference indicator } r \]
Derivation of Marginal Abatement Costs (MAC)

\[ r = \text{index for reference indicator} \]

\[ j = \text{index for other specific indicator} \]

This will show under which indicator the highest efficiency will be obtained, meaning that “real” abated emissions of the optimized production portfolios of the farms are calculated and related to the abatement costs.

The two calculations of MAC curves (normalized to reference indicator and not normalized) will enable the model user to compare two different impacts of an emission abatement scheme. On the one hand the not normalized calculated abatement cost curves will show the abatement reactions and the associated costs on farm level. This will show the charging of costs that will be induced to the different farm types through a crediting scheme because the not normalized MAC curves are the ones who drive the on farm decisions in abatement strategies. On the other hand, one can evaluate the cost efficiency of different emission indicators by the normalized MACs, because the calculated abatement amounts by a specific indicator can show great divergences to the real abatement efforts of the farm. The second task enables the model to evaluate different emission indicators concerning their real mitigation effect.
6 The coefficient generator

The coefficient generator comprises a number of small modules, realized in GAMS, which define the various exogenous parameters comprised in the template. It is designed such that it can generate from a few central characteristics of the dairy farm (herd size, current milk yield, existing stables and their construction year, labour force and available land) and the realized yields of the crops a plausible set of coefficients for the model template.

It is broken down in the following modules:

- Labour: defines labour needs on a monthly basis for herds and crops, and wages for the off-farm work (coeffgen).
- Manure: quantifies amount of animal excreta depending on livestock category. For cows manure amount is controlled by yearly milk output level. Furthermore coefficients for different manure storage and application types are given by this module.
- Credit: different credit types are defined, varying in interest rate and payback time
- Cropping: different activities for cash-crop production with specific restrictions concerning crop rotation, fertilizer demand and yield potentials.
- Farm constructor: The farm constructor defines relationships between benchmark data of the farms and production specific endowments of e.g. land, stables and machinery in the initial situation.
- Feeds: possible fodder compounds are listed with their specific contents of ingredients (N, C, DM, XP,...)
- Indicators: this module gives a definition of the different GHG indicators and a description of the underlying calculation schemes and parameters. The majority is taken from IPCC methodology and completed by other literature findings.
- Ini_herds: it defines the initial herds
- Machinery: defines the different types of machinery that are available for the farm and quantifies the useful lifetime (defined according to years or on hourly basis) as well as investment or variable cost
- Prices: different default values are defined if prices for variables are not defined by the graphical user interface
The coefficient generator

- Requirements: Definitions of requirement functions for lactating cows in relation to their milk yield, live weight etc, as well as for heifers and calves.

- Stables: different types of stables with relating investment costs and capacities (animal and manure) are defined for cows, calves and heifers in different systems.

- Silos: definition of different types of surface reservoirs for liquid manure. Differentiated concerning capacity and related investment costs. Furthermore, the additional costs of specific coverage types of the surface manure reservoirs are defined for straw coverage and coverage with a foil.

- Cows: Cows, heifers and calves are defined which have different milk yield potentials. Furthermore, a maximum number of lactation is defined which depends on the milk output level of the lactating cows (diminishes with increasing milk output potential).

- Fertilizing: defines coefficients for various application techniques for organic and synthetic fertilizers.
7 Technical realization

7.1 Overview

The model template and the coefficient generator are realized in GAMS (General Algebraic Modelling System), a widely used modelling language for economic simulation models. GAMS is declarative (as seen from the template discussion above), i.e. the structure of the model’s equation is declared once, and from there different model instances can be generated. GAMS supports scripting for data manipulation, which is used many in the coefficient generator and the post-model reporting to draw tables and graphs.

![Diagram of Experiment Designer, Controller, and Exploiter](image)

Figure 7. Overview on tool
Own illustration

Additionally, as an extension of the experiment exploiter, “machine learning” (described by a technical paper of BRITZ 2011) will be usable to derive correlations and dependencies between model results and available model variables.

7.2 MIP solution strategy

In opposite to purely linear problems, Mixed-Integer models (MIPs) are far harder to solve. In order to find the optimum, basically the combinatorial set of all binaries respectively general integer variables would need to be evaluated. Depending on the simulation horizon
Technical realization

of FARMDYN, the number of farm branches considered and the time resolution for investment and labour use decisions, a model instance can comprise between a few dozens to more than a thousand binary variables, with often several ten thousands variables and equations in total.

There are huge differences in the quality of LP and more so MIP solver. Industry solvers such as CPLEX or GUROBI reflect continuous investments into algorithmic improvements over decades. Fortunately, both offer free academic licenses. The code is set-up to work with both solvers, as an insurance should the license conditions change and to allow switching in cases one of the solvers outperforms considerably the other. Current tests seem to show a slight advantage for CPLEX. Both solvers can benefit from parallel processing. Model instances should therefore if possible be solved on a multi-core computing server. The option file are currently defined such that one core is not used by the program.

The relaxed version of the model (where binaries and integers are removed and treated as continuous variables) can typically be solved in a few seconds, and once such a starting point is given, slight modifications to the model take very little time to solve despite the model size. However, despite tremendous algorithmic improvements in solving MIPs, the MIP version could take quite long to solve without some solution tactic.

The model code therefore integrates different strategies to speed up the solution process for the MIP. Some of those are generally applicable to MIP problems, typically offered by GAMS and/or the MIP solvers, others follow tactics proposed to speed up the solution of MIP problems, but require a specific implementation reflecting the model structure. In the following, these strategies are roughly described, starting with the model generic more first.

In order to define a lower bound on the objective which allows the solver to cut-off parts of the tree, the model is first solved in relaxed mode (RMIP) with the farm switched off such that income can only generated by working off-farm (v_hasFarm is fixed to zero). Solving that variant takes less than a second. The solution is used to define the lower cut-off for MIP solver. Next, the model is solved as RMIP with only one state-of-nature, and afterwards, the state contingent variables are copied to all other states-of-natures, before the RMIP is solved again. The main statements (see exp_starter.gms for details) are shown below:

```gams
* --- solve model without farm (= v_hasFarm is zero, solprint=2 -> no model output),
* family will work off farm
* resulting objective function is used to define a lower limit for the objective function
* v_hasFarm.up(tCur) = 0;

solve m_farm using RMIP maximizing v_obj;
* --- next solve with farm allowed, and solprint settings from interface
* v_hasFarm.up(tCur) = 1;

* --- solve farm as RMIP with one state of nature
* solve m_farm using RMIP maximizing v_obj;

* --- copy SDN specific decision variable results
* for the first SDN to all other one
*$include 'model/copy_sdn.gms';

solve m_farm using RMIP maximizing v_obj;
```
The relaxed (RMIP) solution defines the upper cut-off – forcing certain variables to only take on integer values can only reduce the objective function. At the same time, it proves a basis for solving the MIP. It has however in many instances not proven useful to use the solution of RMIP as MIP start starting point, both CPLEX and GUROBI seem to spend considerable time to construct a feasible integer solution from the RMIP solution.

As stated above, solving a MIP problem to its true optimum can be tremendously time consuming. Therefore, typically MIP problems are only solved given an optimality tolerance. The branch-and-cut algorithm used in MIP solvers always provide a safe upper limit for the objective value stemming from a relaxed version of the current tree node. Accordingly, they can quantify the maximal absolute and relative gap to the potentially maximal objective function. Typically, the smaller the desired gap, the larger is number of combination of integer variables the solver needs to test. Forcing the gap to zero basically requires more or less a test of all combination, i.e. ten-thousands of solves of a LP version of the model with binaries and integers fixed. In most production runs, a relative gap of 0.5% has proven as acceptable. The solver will then stop further search for a better solution once a MIP solution has been found which differs by less from the relaxed best node.

The problem with the gap is clearly that differences between two simulations can not only stem from different model inputs (prices, policy etc.), but also simply from the fact that the gap at the best solutions returned by the solver for each run differs.

MIP solvers can also “tune” their options based on one or several given model instance. Tuning is available both with CPLEX and GUROBI, and can be switched on via the interface. That process takes quite long, as the model is repeatedly solved with different solver options. The parameters from the tuning step are stored in an option and can be used by subsequent runs.

### 7.2.1 Fractional investments of machinery

An option to reduce the number of binaries is to treat certain investment decisions as continuous. For machinery, the model allows to replace the binary variable \( v_{\text{buyMach}} \) by a fractional replacement \( v_{\text{buyMachFlex}} \). The replacement depends on a threshold for the depreciation costs per ha or hour, which can be set by the interface. The larger the threshold, the lower is the number of integer variables and the higher the (potential) difference to the solution where more indivisibilities in machine investments are taken into account.

The relevant code section (exp_starter.gms) is shown below:

```plaintext
7.2.2 Heuristic reduction of binaries

On demand, the RMIP solution can be used in combination with some heuristic rules to reduce the set of endogenous variables. As the RMIP solution will e.g. build a fraction of larger stables and thus save costs compared to the MIP solution, the herd size in the MIP
```
solution can be assumed to be upper bounded by the solution of the MIP. Similarly, as investment costs for machinery will be underestimated by the MIP, it can be assumed that machinery not bought in the RMIP solution will not be found in the optimal solution of the MIP.

An example is shown below for investment decision into stables. The program first defines the maximal amount of stable places used in any year. Investments into stables and their usage which are larger than the maximal size or smaller then $2/3$ of the maximal size are removed from the MIP. Equally, investment in stables is set to zero if there was no investment in the RMIP solution.

```
*  * excludes bigger stable investment under binary conditions
*  * parameter p_numberStables(t) with values
*  * p_numberStables(t) = p_numberStables(stables, stableTypes)
*  * v_stable(t, stableTypes) [g_numberStables(stables, stableTypes), v_stable(t, stableTypes, t),
*  * p_numberStables(stables, stableTypes), g_v_stable(t, stableTypes, t)
*  *]
*  * p_numberStables(stables, stableTypes)
*  * v_stable(t, stableTypes) [g_numberStables(stables, stableTypes), g_v_stable(t, stableTypes, t)
*  *]
*  * v_stable(t, stableTypes) [g_numberStables(stables, stableTypes), g_v_stable(t, stableTypes, t)
*  *]
*  * v_stable(t, stableTypes) [g_numberStables(stables, stableTypes), g_v_stable(t, stableTypes, t)
*  *]
*  * v_stable(t, stableTypes) [g_numberStables(stables, stableTypes), g_v_stable(t, stableTypes, t)
*  *]
*  * v_stable(t, stableTypes) [g_numberStables(stables, stableTypes), g_v_stable(t, stableTypes, t)
*  *]
```

Similar statements are available for investments into manure silos, buildings and machinery. These heuristics are defined in “model/reduce_vars_for_mip.gms”. It is generally recommended to use these statements as they can considerably reduce solving time. However, especially after structural changes to the code, checks should be done if the rules do not actually prevent the model from finding the (optimal) MIP solution.

### 7.2.3 Equations which support the MIP solution process

Another tactic to ease the solution of MIPs is to define equations, which decrease the solution space for the integer variables based on the level of fractional variables respectively define logical ordering for the integer decisions. These equations are not necessarily truly restricting the solution space, they only reinforce existing relations between variables. The additional equations often reduce the overall solution time by improving the branching more then by increasing single LP iterations due to the increase in the constraints.

One way to improve the branching order is to link binaries with regard to dynamics. There are currently three ordering equations over time. The first two prescribes that a farm respectively a cow herd in $t+1$ implies a farm respectively a cow herd in the previous year:

```gams
hasFarm(t-1) \imp \hasFarm(t) .
```

The second one implies that working off-farm in an year $t$ implies also working off-farm afterwards:

```gams
workOff(t) \imp \workOff(t-1) .
```
Another tactic followed is to define logical high level binaries which dominate other. These general binaries are partly already shown above: the \texttt{v_hasFarm} and \texttt{v_workOffB} variables. The later one is linked to the individual off-farm working possibilities:

In order to support the solving process, \texttt{w_workOff} is defined as a SOS1 variable, which implies that at most one of the \texttt{workType} options is greater than zero in any year.

The \texttt{v_hasFarm} variables dominates the \texttt{v_hasBranch} variables:

That equation is additionally linked to the logic of the model as \texttt{v_hasFarm} implies working hours for general farm management.

There is also a general binary which controls if a herd is present in any year, \texttt{v_hasAlwaysHerd}. If switch one, it will imply a dairy herd in any one year, which is based on the \texttt{hasAlwaysLast} _ equality together with the order equation \texttt{hasHerdOrder} _ shown above.

The equations which support the MIP solution process by linking fractional variables to binary ones relate to investment decisions. Firstly, investments in machinery are only possible if there is matching machinery need:

Two equations link the dairy herd to investment decisions into stables and manure storage silos:
These supporting restrictions can be switched off from the model via the interface, to check if they unnecessarily restrict the solution domain of the solver. It is generally recommended to use them as they have proven to speed up the solution process.

### 7.2.4 Priorities

Finally, there are options to help the MIP solver to decide which branches to explore first. The variable field .prior in GAMS allows setting priorities which are passed to the MIP solver; lower priorities are interpreted as having precedence. The file “model\def_priors.gms” defines such priorities.

The model is instructed to branch first on the decision to have a herd in any year, next on having a farm and the individual branches:

```plaintext
v_hasAlwaysHerd.prior = $priorOperator$ (p_priorMax=20);

v_hasFarm.prior(t) = $priorOperator$ (p_priorMax=6 + $timeWeight$);

v_hasBranch.prior("branches",t) = $priorOperator$ (p_priorMax=3 + $timeWeight$);

v_hasBranch.prior("dairy",t) = $priorOperator$ (p_priorMax=4 + $timeWeight$);

v_hasBranch.prior("Farm",t) = $priorOperator$ (p_priorMax=5 + $timeWeight$);
```

Generally, early years are given precedence:

```plaintext
$setGlobal TimeWeight (card(t)-ord(t)+1)/card(t) * p_priorMax * 10
```

The $p_{\text{priorMax}}$ is the maximal priorities assigned to stables which is defined by a heuristic rule: large stables are tried before smaller ones, cow stable before young cattle and calves stables, and finally long term investment in the whole building done before maintenance investments:

```plaintext
-- priorities for stable: try large stable first, and give precedence for cow over young over calf stables
parameter p_priorStables(stables);
p_priorStables(stables) = sqrt(1/sum(stableTypes $ p_stableSize(stables,stableTypes), stableTypes.pos)=10)
  * sqrt( sum(stableTypes, p_stableSize(stables,stableTypes)) / sum( stables,stableTypes $ p_stableSize(stables,stableTypes), p_stableSize(stables,stableTypes) ));

scalar p_priorMax;
p_priorMax = max(stables, p_priorStables(stables))*card(hor);
p_priorStables(stables) = p_priorStables(stables)/p_priorMax;
P_priorMax = 1;

scalar p_priorMin;
p_priorMin = min(stables, p_priorStables(stables));
```

Off-farm work decision currently receive a lower priority compared to investments into stables:
For other investment decisions, the investment sum is used for priority ordering:

```
* * *
* for buildings and machinery, use investment price to define priorities
* * *
  parameter p_rank;
  p_rank(buildings) = p_price(building,"FirstYear");
  p_rank(mesh) = p_price(mesh,"FirstYear");
  p_rank(silos) = p_price(silos,"FirstYear");
  set ranks[i]; ranks(buildings) = W; ranks(mesh) = W; ranks(silos) = p_rank;
``` 

The SOS1 variables should have all the same priorities. Therefore, no distinction is introduced for the v_workOff and v_siCovComb equations, with the exemption of the time dimension.

It is generally recommend using these priorities; they have proven to speed up the solution process.

### 7.3 Reporting

As discussed in the following chapter, a GUI allows exploitation of model results, also comparing different model runs. That part requires that all results to be inspected are stored in one multi-dimensional cube. Accordingly, after the model is solved, its variables are copied to a result parameter, as shown in the following example:

```
* * *
* --- financial results
* * *
  p_res[%,2,"liquid","sum",tCur] = v_liquid.I(tCur);
  p_res[%,2,"liquid","sum","mean"] = sum(tCur,v_liquid.I(tCur))/p_cardI(tCur);
  p_res[%,2,"liquid","sum","tCur"] = p_hcon(tCur);
  p_res[%,2,"hcon","sum","mean"] = sum(tCur,p_hcon(tCur))/Card(tCur);
``` 

### 7.4 Systematic sensitivity analysis based on Design of Experiments

As discussed above, solution for one indicator and one GHG emission target might require between a few seconds to several minutes on a powerful multi-core machine. The derivation of the marginal abatement cost curves requires solving repeatedly model instances over a range of GHG emission targets might hence require an hour or more to solve one specific farm configuration.

An application of the model to a larger sample of existing farms is hence computationally impossible. It was therefore envisaged from the beginning to use sensitivity analysis to generate a sufficient number of instances to derive a meta-model, e.g. based on an appropriate regression model, in order to estimate abatement costs for larger population of farms. Meta modeling seems also a suitable tool to learn more about which farm attribute impact abatement costs and to which extent the occurring MACs depend on the GHG calculation procedure of the different indicators.

In order to do so, the following steps need to be taken
1. Setting up of appropriate sensitivity experiments which cover the distribution of farm attributes in an appropriate sample (such as the farm structure survey for North-Rhine-Westphalia). Therefore using an efficient and space filling random sampling design to lower the necessary sample size for the derivation of a meta-model, ensuring that the randomized factor level combinations are smoothly distributed over the range of factor level permutations (This assures also under the restricted number of random values for each factor the components are still represented in a fully stratified manner over the entire range, each variable has the opportunity to show up as important, if it indeed is important (Iman 2008)).

2. Running the single farm model on these experiments and collecting key results

3. Deriving a meta model from these experiments

This section focuses mainly on technical aspects of that process.

The overall strategy consists of combining a Java based package for interface generation and result exploitation, which also comprises a machine learning package, with GAMS code. For the definition of representative sensitivity experiments a sampling routine (lhs_0.10) implemented in R (version 2.15.1) is combined with the GAMS code to generate sample farms under recognition of correlations between factors.

The GAMS code (scen_gen) modifies settings entered via the interface (see next section) to define attributes for each experiment. A single farm run is then executed as a child process with these settings. The user is able to define upper and lower bounds for single factors to define a solution room in which factor levels can vary between scenarios for different production specific attributes of the farm (see next section). The interface also allows to define if correlations between selected variables should be recognized during the sample randomization procedure or not. Further on, depending on the number of draws and the complexity of the assumed correlation matrix a maximum number of sampling repetitions can be selected (this is necessary to restrict sampling time but also guarantee to find a random sample that appropriately implies the correlation structure as proposed by the user (more detailed explanation of this later in this paper)).

Only the factors for which the selected maximum value differs from the minimum value are varied between model runs. Hence, the user is able to fix factor levels for single factors over all experiments by defining the min and max factor level equally. The upper and lower bounds of the variables define the solution room of possible factor level combinations of different factors. If the chosen min and max values are equal, the factor level of the specific attribute is hold fix during the scenario definitions. For the definition of wage rates and prices for concentrates the user is able to select constant differences to the full time wage rate or the concentrate type 1.

With increasing number of factors that can vary between scenarios and increasing possible factor levels per factor, the number of possible scenarios (factor level permutations) will increase exponentially (up to a few thousands). Hence, to create model outputs representative for all

Only the factors for which the selected maximum value differs from the minimum value are varied between model runs. Hence, the user is able to fix factor levels for single factors over all experiments by defining the min and max factor level equally. The upper and lower bounds of the variables define the solution room of possible factor level combinations of different factors. If the chosen min and max values are equal, the factor
level of the specific attribute is hold fix during the scenario definitions. For the definition of wage rates and prices for concentrates the user is able to select constant differences to the full time wage rate or the concentrate type 1.

With increasing number of factors that can vary between scenarios and increasing possible factor levels per factor, the number of possible scenarios (factor level permutations) will increase exponentially (up to a few thousands). Hence, to create model outputs representative for all admissible scenarios, a large number of scenario runs would have to be processed to get reliable outputs for the derivation of a meta-model.

But as this would cause long computing time also on a multi-core processor (several days), we have to restrict the number of scenario runs to a manageable number but still being representative for the real life distribution of farm attributes.

Therefore, the scenario definition is done by Latin Hypercube Sampling (LHS) to create an efficient sample with a small sample size (to lower computing time) but guaranteeing a space filling sample design over the full range of admissible scenarios (McKay et al. 1979, Iman and Conover 1980). This is done, using a bridge from GAMS to the statistical software R. Therefore the LHS package of R has to be installed for being able to create LHS samples for a defined number of draws n and factors k (in our case taking the command “improvedLHS(n,k)”). LHS sampling creates a sample matrix of size n*k incorporating random values between 0 and 1, interpretable as percentages. These are drawn assuming an uniform distribution between 0 and 1. Further on, LHS sampling outputs ensure orthogonality of the output matrix and factor level combinations evenly distributed over the possible permutation area.

The GAMS side of the technical implementation is shown in the following:

```
* Use R to define the DOE
*---------------------------------------------------------------

$setglobal r "%curdir%\..\r-2.15.1\bin\rscript.exe"
$setglobal inputFile  "%curdir%\frame.r"
$setglobal inputFile  "%curdir%\tor.gdx"
$setenv plotFile  "%\results\scenGen\hs_%scen%.pdf"
$setenv outputFile  "%outputFile%"
$setenv inputFile  "%inputFile%"

*** define maximum time for solving of best fitting LHS sample***
$setenv maxRunTime  30
```

The maximal run time for finding a sample can be defined (maxRunTime). If correlations between are known and should be recognized within the sampling procedure, the command useCorr has to be set to “true”. Then the correlation matrix can be defined specifically.
The names of the set of varying factors, the factor names, the scenario name, the desired number of draws and, if activated, also the correlation matrix are send to R. Then the R file "rbridge\lhs.r" is executed.

```
$p_n <- p_n + 1;
$p_cor(factors, factors) = p_cor(factors, factors) + p_cor(factors, factors)/2;
```

The R-bridge is hence activated (R side). Therefore several packages are installed from the R library to be able to do LHS sampling:

```
library(lattice);
library(earth);
library(mvtnorm);
library(plyr);
library(rpart);
library(rpart.plot);
library(reshape);
library(reshape2);
library(gclus);
```

$p_n$ denotes the number of draws defined via the graphical user interface, equivalent to the number of scenarios resulting from the sampling routine. `Sys.getenv(...)` asks for commands or information given by the environment (e.g. if correlations have to be recognized or not).
We decided to use the “improvedLHS” Type for randomization\(^6\) which produces a sample matrix of \(n\) rows and \(k\) columns (\(n\) = number of draws, \(k\) = number of factors). This leads to a quite efficient sample generation in R:

```
out1 <- improvedLHS(n,k);
```

Typically, input variables for sensitivity analysis in computer models are assumed to be independent (Iman et al., 1981a;b). Also LHS sampling was designed to create a sample of factor level combinations for different factors avoiding correlations between factors in random draws to ensure a space filling output. But, for our purposes, it is important to incorporate as much information about the multivariate input distribution as possible to get more realistic sample scenarios and exclude factor combinations that are quite impossible in reality. Hence, following Iman and Conover (1982:p.331-332) correlation structure information among input variables should be recognized within the sampling process, if available. Otherwise “the theoretical properties of the statistics formed from the output may no longer be valid.” (Iman and Conover 1982:p.331)

To also incorporate information about dependencies between interesting variables during the sampling procedure we expanded the sampling method by an approach of Iman and Conover (1982) designing a swapping algorithm which shuffles observations for single factors between the draws to mimic given \(k\times k\) correlation matrix (therefore the R package MC2d including the routine `cornode` is necessary).

```
# --- load correlation matrix from OAMS
c <- rgdX_param(inputfile, "p_co", names=c("f1","f2"), compress="true")
c <- mc2d.corrMat(f1=2, value.var="Value")
c <- mc2d.matrix(c);
```

To increase the possibility to randomize a sample which offers a correlation matrix of factors near the proposed one, the routine allows to repeat the random sampling of demanded \(n\) draws (yielding \(n\) experiment scenarios) for a maximal given computing time (“maxRunTime” e.g. 300sec.). The sample (incorporating \(n\) draws for \(k\) factors) with the smallest mean percentage deviation (meanDev) between given and randomized correlation

\(^6\) another possible routine for LHS sampling is “optimumLHS(***).” But during our test runs it did not lead to more smooth space filling random draws, but increased the runtime of the sampling process. For optimal-LHS see also Park (1994).
matrix is then selected and send back to GAMS as the random sample representing the possible population. Alternatively, the repetition of \(n\) draws \((n \times k\) sampling matrix) will be stopped by a threshold value \((if \ meanDev < 1\) for the deviation between the assumed and the randomized sample correlation matrix.

```
bestFit <- 1;
maxruntime <- as.numeric(System.time()
for (draw <- 1;
   while (runtime < maxruntime ){
      draw <- draw + 1;
      if (LHSType == "optimumLHS"){
         out1 <- optimumLHS(n,k,i,j,0.001);
         print("hit");
      } else {
         out2 <- improvedLHS(n,k);
      }
   }
   # --- use corrand to apply man & conover 1982 to impose correlation
   t <- cor(out1);
   fit <- 0;
   for (i in 1:k)
   for (j in 1:k)
   if (t[i,j] < bestFit)
      fit <- fit + ((1,1)-t[i,j])*(1,1)-t[i,j]);
   if (fit < bestFit ){
      out <- out1;
      bestFit <- fit;
      meanDev <- sqrt(fit/k)**100;
   }
   if (draw %% reportdraws == 0{
      runtime <- system.time()
      print(paste( draw, runtime ,round(runtime)," of ",maxruntime,"seconds, mean sqrt of squared diff between 
      if (meanDev < 1){
         print(paste( draw, runtime ,round(runtime)," of ",maxruntime,"seconds, mean sqrt of squared diff between 
      runTime <- maxruntime;
   }
}
```

For the case that the correlations between factors given by the user lead to an undefined correlation matrix, the program adjusts the correlation matrix to the nearest one possible:

```
# --- find nearest positive definite matrix

t1<-nearPD(t1)

t <- as.matrix(t1mat);
```

The LHS sampling defines random value combinations between all factors in each single draw. Therefore uniform distributed random values between 0 and 1 are drawn. The total set of draws defines one random sample of \(n\) single experiments \((factor level combinations (in this stage of the sampling still between 0 and 1))\). The routine implemented into the LHS-module now tries to find the best fitting sample which corresponds to the demanded correlation matrix most properly. Sampling outputs of the LHS draws show efficiency characteristics, also under recognition of correlations. This means that the mean of drawn random values is still 0.5 (as LHS draws lie between 0 and 1). And if the number of draws is large enough \((greater than 30)\), quantiles as well as the mean of the distribution of LHS random values show that we are still consistent with the assumption of an uniform distribution function of the random draws \((between 0 and 1)\), as necessary for efficient LHS outputs, also under recognition of factor correlations. The best fitting sample with the minimal average percentage deviation of correlations between defined and randomized correlation matrix is then selected and stored by the program and automatically printed as a Pdf-document for visualization, also giving information about average percentage bias of the randomized correlation matrix as well as the number of total draws which define the number of resulting sample experiments:
On the left hand side one can see the scatter plot matrix without any correlations between factors. In contrast, a clear difference in sampled values is visualized by the right hand side matrix. E.g. a correlation between \( nCows \) and \( milkYield \) was assumed to be 0.8. The best fitting matrix lead to the same correlation between these two factors. The correlation coefficients within brackets are the correlations predefined by the operator. The values in front of the brackets are the correlation coefficients fitted by the sampling matrix. The average mean percentage deviation of the randomized correlation matrix and the assumed correlation matrix is quantified by 7.34%, meaning, that on average, the randomized correlations deviate by 7.34% from the predefined ones. The distribution functions in the diagonal show, that the sampled values of each factor still ensure a uniform distribution, also under recognition of correlations.

The random values for the scenarios (still between 0 and 1) are transformed by GAMS to the real factor levels following the distribution functions of single variables. We assume a uniform distribution of factor levels for the relevant variables. These are easily to define by the minimal value \( a \) and the maximum value \( b \). A uniform distribution function can be defined by the following density function (left graph):

\[
(1) \quad f(x) = \frac{1}{b-a}, \quad a \leq x \leq b
\]

Values below \( a \), or above \( b \) have a probability of 0. The antiderivative expresses the cumulative distribution function of the random variable whose values lie within the interval \([0; 1]\) (right graph):

\[
(2) \quad F(x) = \frac{x-a}{b-a}, \quad a \leq x \leq b
\]
Figure 9. Density function and cumulative distribution function of an uniform distributed variable

From the left hand side density function one can easily derive the right hand side cumulative distribution function (also possible to implement other distribution functions of the factor). Though, the \( y \) value of the distribution function equals the integral \( \int_a^x f(x) \) below the density function (cumulative probabilities below \( x \)).

The given random values of the R-routine (\( F(x) \)) enable the allocation of corresponding factor levels (between \( a \) and \( b \)) to this random percentage values from the cumulative distribution function. A random cumulative probability value \( y \) corresponds to the factor level \( x \) which lies within the real value domain of the interesting factor. Hence this random sampling procedure produces random values by transforming uniform distributed random percentages (between 0 and 1) to factor levels conform to the assumed distribution function of the variable. As the model is defined up to this point, we assume uniform distribution functions for the real factor levels (this can be adjusted to other functions if e.g. a known population has to be simulated).

For an assumed uniform distribution function of factor levels this is done following the formula:

\[
(3) \quad F(x) \times (b - a) + a = x
\]

The randomized value \( y \) is transformed to the factor level room concerning the given distribution function of the factor. Hence for each single random draw a value is generated for the interesting variable corresponding to its assumed probability distribution (hence, formula 3 will change if the assumed formal distribution changes).

In GAMS code the formula (3) has to be applied for each factor to calculate the sample values whereat \( p\_doe(draws, "factor") \) is equivalent to \( F(x) \). The random percentage \( p\_doe(*,*) \) has to be multiplied by the difference between the possible max and min value of the factor \( (%factorMax\% - %factorMin\%) \). Afterwards the min value \( (%factorMin\%) \) has to be added to the product to yield the factor level \( x \) for the specific factor and scenario.
Technical realization

p_scenParam(draws,factor) gives the scenario parameter of one factor defined by the random values given by the LHS sampling routine. The combination of factor levels of the different factors for one single draw defines one single sensitivity scenario.

A set defines which settings are (possibly) defined specific for each scenario:

```gams
* --- exclude implausible ones
* (1) large herds and low milk yields
allScen(scen) $( (p_scenParam(scen,"ncows") ge 10) and (p_scenParam(scen,"mlkyield") le 50) ) = no;
allScen(scen) $( (p_scenParam(scen,"ncows") ge 100) and (p_scenParam(scen,"mlkyield") le 50) ) = no;
allScen(scen) $( (p_scenParam(scen,"ncows") ge 10) and (p_scenParam(scen,"mlkyield") le 70) ) = no;
allScen(scen) $( (p_scenParam(scen,"ncows") le 60) and (p_scenParam(scen,"mlkyield") ge 80) ) = no;
allScen(scen) $( (p_scenParam(scen,"ncows") le 10) and (p_scenParam(scen,"mlkyield") ge 70) ) = no;
allScen(scen) $( (p_scenParam(scen,"ncows") le 10) and (p_scenParam(scen,"mlkyield") ge 60) ) = no;
* (3) small herds and high milk yields
allScen(scen) $( (p_scenParam(scen,"ncows") le 60) and (p_scenParam(scen,"mlkyield") ge 80) ) = no;
allScen(scen) $( (p_scenParam(scen,"ncows") le 10) and (p_scenParam(scen,"mlkyield") ge 70) ) = no;
allScen(scen) $( (p_scenParam(scen,"ncows") le 10) and (p_scenParam(scen,"mlkyield") ge 60) ) = no;
* (4) small herds and recent investments in stables
allScen(scen) $( (p_scenParam(scen,"ncows") le 10) and (p_scenParam(scen,"stableyear") eq 2000) ) = no;
* --- delete parameters of deleted scenarios
p_scenParam(allScen,scenItems) $(not allScen(scen)) = 0;
p_scenParam(allScen,"lastyear") = "lastyear";
```

Nevertheless correlations between factors are able to be recognized during the sample generation to avoid factor level combinations within scenarios that conflict with common statistical knowledge; the model code enables the user to specifically exclude factor level combinations that seem to be implausible. For example high labor input per cow and low milk yield levels or high numbers of cows per farm and only very low yielding phenotypes.
These scenario settings must be stored in a GAMS file which then picked up by the child processes. In order to keep the system extendable, first, all settings inputted via the Graphical User Interface are copied over to the specific scenario:

```
* * --- copy content of current scen file into new one
* *     via OS command
* *     execute "type %curDir%\incgen\%scen%.gms > %curDir%\incgen\curScen.gms;"
```

Afterwards, the modifications defining the specific sensitivity experiment, i.e. the scenario, are appended with GAMS file output commands (see scenGen\gen_inc_file.gms):

```
* * --- put statements will append to the new scen file
* *     and overwrite standard setting
* *     put scenfile;
* *     scenfile.ai = 1;
* * * --- send scen specific parameters to include file
* *     loop(scenItems,
* *         put "\$SETGLOBAL \",scenItems.tl," \",p scenParam(scen,scenItems) /;
* *     );
* *     put "\$SETGLOBAL scenDes curScen " /;
* *     putClose scenFile;
```

Finally, we need to copy the content to a specific scenario input file:

```
put Utility batch 'shell' / "type %curDir%\incgen\curScen.gms > %curDir%\incgen\"scen.tl".gms;"
```

The code to build and optimize the single farm model itself is realized in GAMS and uses CPLEX 12.6 in parallel mode as the MIP solver. Automatic tuning is used to let CPLEX use appropriate solver setting on the problem. The model instances are set up such as to avoid any conflicts with I/O operations to allow for parallel execution.

A single instance has a typical load of about 1.8 cores in average. In a multi-core machine, it seems hence promising to execute several such processes in parallel. That is again realized by a GAMS program which starts each model on its own set of input parameters:

```
put Utility batch 'shell' / 'start /NORMAL %GAMS\\bin\gams.exe %GAMS\\bin\exp_starter.gams --scenario=allScen.tl
  --scenario=loop:0:8 --maxProcs=255 --output=allScen.tl.1st',
  '%GAMS\bin\gams -ai=0 --pgmName="allScen.tl" ('\loop:0:8', of '.').card(\allScen.tl.0):0,0,0';
```

The name of the scenario (allScen.tl) is passed as an argument to the process which will lead a specific include file comprises the definition of the scenario.

The GAMS process will use its own commando processors and run asynchronously to the GAMS thread which has started it. The calling mother process has hence to wait until all child processes have terminated. That is achieved by generating a child process specific flag file before starting the child process:

```
* * --- generate a flag file which will be deleted by the GAMS process upon completion,
* *     it steers the further execution of the loop
* *     put Utility batch 'shell' / "echo test > %curDir%\flags\"allScen.tl".flag";
```
Technical realization

That flag file will be deleted by the child process when it finalizes:

```bash
execute 'if exist %curDir%\Flags\%scen%.flag del %curDir%\Flags\%scen%.flag'
```

A simple DOS script waits until all flags are deleted:

```bash
set /a _trys=0
:again
if %Modo% EXIST %FlagFiles% (  
    set /a _tries+=1
    if _tries%>=%MaxTrys% goto errorexit
    sleep.exe %seconds%
    goto again
)
```

Using that set-up would spawn for each scenario a GAMS process which would then execute all in parallel. The mother process would wait until all child processes have deleted their flag files before collecting their results. As several dozen or even hundredth of scenarios might be executed, that might easily block the machine completely, e.g. by exceeding the available memory.

It is hence necessary to expand the set-up by a mechanism which ensures that only a pre-defined number of child processes is active in parallel. That is established by a second simple DOS script which waits until the number of flag files drops below a predefined threshold:

```bash
set /a _trys=0
:again
set _count=1
for %%x in (%FlagFiles%) do set /a _count+=1
if _count% gtr %nfiles% (  
    set /a _tries+=1
    if _tries%>=%MaxTrys% goto errorexit
    sleep.exe %seconds%
    goto again
)
```

Finally, we need to collect and store the results from the individual runs. We use a GAMS facility to define the name of a GDX file to read at run time:

```bash
put_utilities batch 'gdxin' / '.\results\expFarms\res\'scen.tl' until 'p_scenParam(scen,"lastYear")\0:\0\.'
gdx
```

And load from there the results of interest:

We now transformed all MAC estimates which are 0 due to an exit decision of a farm to to be able to select these cases for our meta-modelling estimation (Heckman two-stage selection, described in the next technical documentation: “R routine to estimate Heckman
two stage regression procedure on marginal abatement costs of dairy farms, based on large scale outputs of the model DAIRYDYN” by Britz and Lengers (2012)).

Further on, we store the scenario specific settings as well which can be used as explanatory variables for later regressions:

The results are then stored in a GDX container

The major challenge consists in ensuring that the child processes do not execute write operation on shared files. In the given example, that relates to the GAMS listing, the GDX container with the results and the option files generated by the CPLEX tuning step. For the latter, two options would be available: (1) set up child specific directory, copy the option files into them and use the “optdir” setting in GAMS, or (2) label the option files accordingly. That latter option was chosen which restricts the number of scenarios to 450:
In the case of normal single farm run, the standard option files will be used.
8 Graphical User Interface

The Graphical User Interface (GUI) is based on GGIG (GAMS Graphical Interface Generator, Britz 2014). It serves two main purposes: to steer model runs and to exploit results from one or several runs. The creation of a visual user interface is also described as “visual debugging” (GRIMM, 2002) to allow for an easy adjustment of parameters and quantitative and graphical examinations. With the help of only a few adjustments one can define single or multiple model farm runs for the interesting farm types with their specifications. Here, the former described coefficient generator helps to condense the necessary information for farm run definition by adjusting and calculating all production specific parameters to be consistent with the defined farm type (initial arable land, grass land, initial stables, initial manure storages, initial machine endowment...). After simulating the interesting experiments, the GUI enables the user to systematically analyse the simulated model variables and results.

A separate user handbook for the general use of the GUI is available at:


8.1 Model farm and scenario specifications

In the following, the different tabs of the GUI are shown and shortly described.

8.1.1 Workstep and task selection

In “Single farm runs“ mode, all run specific settings (input and output prices, farm assets etc.) are set by the user in the interface as discussed in the following.

In experiment mode, the user instead define ranges for selected settings which are varied based on stratified random sampling using Design of Experiments for a defined number of experiment. For each experiment, a single farm run is solved. These single farm runs are typically solved in parallel. After they are finalized, their results are combined into one result set.
8.1.2 General settings

The first tab defines the name of the scenario under which the results are stored. Further on, the user chooses general farm characteristics and options for the run: (1) the active farm branches, (2) if different states of nature (with regard to prices) are used, (3) if land can be leased or bought. Furthermore, the threshold to consider machinery investments as binary variables can be set.

The controls under “Time” determine the last simulation year, if information from that simulation period is used to estimate economic returns until stables are depreciated and the time resolution for investment / farm labour and feeding decisions.

The model allows to choose between the following four modes to describe dynamics (or not):

In experiment mode, upper and lower ranges for certain settings can be set, as well as the number of experiments and some further algorithmic detail:
8.1.3 Farm settings

The farm settings panel carries general information about the farm-household and general price increases as shown above.

8.1.4 Animals

The tab “animals” allows setting the initial herd sizes and other attributes related to animal husbandry. The herd size will be used to derive the initial stable and silo inventory, the current stocking rate the land endowment in case of a dairy farm.

In experiment mode, ranges can be defined:
8.1.5 Cropping

The cropping tabs comprises steering options related to land use: the crops considered, if different tillage type, cropping intensities and crop rotations are used. Further on, the average plot size, mechanization level and the climate zone as well as the distribution of the soil type can be chosen.

8.1.6 Biogas

The biogas tab includes possible Renewable Energy Acts (EEGs) for the investment options as well as available biogas plant sizes. Further, it provides the option to select from potential input. Additionally, one can set up an existing biogas plant with the options to choose the size, the valid EEG as well as the construction year. However, in order to function the plant size and EEG has to be activated in the "Investment options" panel. Lastly, some options for scenario premiums are included, which were used in the Master thesis by David Schäfer.

8.1.7 Output prices

That panel allows to set the price of the outputs present in the model.
8.1.8 Input prices

As the aim of the overall constructed model is to compare different designed emission indicator schemes, the GUI sheet “MACs” offers an assortment of the available indicators, usable to force abatement ceilings by the simulated farm to create marginal abatement cost curves. In addition the number of reduction emission reduction steps within the simulation runs and the percentage reduction per step (compared to baseline emissions) has to be defined to define the maximal emission reduction (GHG reduction steps time reduction per step).
8.1.12 Algorithm

The last sheet of the GUI which is shown here defines the chosen solver to optimize the fully dynamic MIP problem and further precision adjustments.

It is normally recommended to use CPLEX as the MIP solvers where more options had been tested.

8.1.13 Debug and output options

That tab allows the user

- to define break points at which model execution stops
- to determine the output level from GAMS models (solution printing)
- to check the marginals for the RMIP solves (resp. feasibility) by forcing the model realize 1 ha of each crop / intensity / tillage combination, respectively a given number of cows or to force a biogas plant into the solution.
Graphical User Interface

8.2 Visualizing and analysing results

After a successful simulation (statement “normal completion” in the simulation window) the user can use the model interface to view results, plot them in tables or in graphs to make further analysis and interpretations. The analysis is based on a set of pre-defined reports, grouped by themes. Currently, the following reports are available:

- Model attributes
- Herd summary, mean
- Land use, mean
- Crops, mean
- Crops, intensities
- Crops, tillage
- Crop costs, mean
- Crop intensities, mean
- Tillage type, mean
- Cows, mean
- Cows, by yield
- Feeding overview, mean
- Production and related revenues, mean
- Feeding cows, mean
- Herd summary, time series
- Land use, time series
- Crops, time series
- Crops, tillage, time series
- Crops, intensities, time series
- Crop intensities, time series
- Tillage type, time series
- Cows, time series
- Cows, by yield, time series
- Feeding overview, time series
- Production and related revenues, time series
- Feeding cows, time series
- Stables overview, mean
- Stables, mean
- Machines, mean
- Stables overview, time series
- Stables, time series
- Machines, time series
- Overview work, mean
- OffFarm, mean
- Overview work, time series
- OffFarm, time series
- Manure, mean
- Manure, time series
- N Crops, total, mean
- P2L5 Crops, total, mean
- N total, mean
- P2O5 total, mean
Graphical User Interface

- Storages, per month, mean
- Storages, total, mean
- N Crops, total, time series
- N Crops, per ha, mean
- N Crops, per ha, time series
- P2O5 crops, total, mean
- P2O5 Crops, per ha, time series
- N in different soil depths
- N in soil, by weather and month
- N in soil, by weather and soil depth
- N Balance, per month, mean
- P2O5 balance, per month, mean
- N balance, per month, per ha
- N balance, per month, time series
- N balance, yearly, time series
- N balance per ha, yearly, time series
- P2O5 balance, per month, time series
- P2O5 balance, yearly, time series
- Overview GHGs
- GHGs by source
- MACs and GHGs
- Revenues, mean
- Revenues per ha, mean
- Costs, mean
- Costs per ha, mean
- Cash balance, mean
- Cash balance per ha, mean
- Revenues minus costs per SON, mean
- Crop costs, mean
- Revenues, time series
- Cash balance, time series
- MACs

8.3 Using the exploitation tools for meta-modeling

In the interface, the “exploit results” button will open a selection dialog to choose results from parallel runs:
Which then show tables with the results:

There are now views necessary to use the machine learning package:

1. A view with the variable to estimate, in our case provide by the table “Meta analysis, MACs”

2. A view with the explanatory attributes, in our chase provided by the table “Meta analysis, explanatory vars”.

In the first view, a click in the table will open a pop-up menu from which “Classify, Classify current view” should be chosen:
Similarly, in the second view, “Classify, Use current view to provide explanatory attributes for numeric classification should be chosen”. The WEKA Explorer can then be used to apply different algorithms from machine learning to the instance. The screen shot below shows the application of the multiple regression model with automatic variable transformation (automatically builds logs, square roots and inverses of the variable values) and variable selection:

Additionally, the user has the possibility to figure histograms or graphs of the interesting output values for graphical visualization. Statistical characteristics like Min, Max or

---

7 The problem here arises, that the simple WEKA regression routines are not prepared for two stage regressions (like e.g. Heckman two stage regression (Heckman, 1979)) like in some cases demanded for our data set as farms which have already exited are in the regression relevant dataset. This may cause a sample selection bias and lead to potentially small explanatory character of the estimated linear regression model. Therefore generated data should be analyzed by a routine, written in R (also designed by Britz and Lengers in 2012 (2012) which is available on enquiry from the responsible authors.
median values are automatically generated as well as mean value of the selected results and the standard deviation.

Nevertheless, the WEKA estimator enables the user to analyze the data as expressed above.
9 Restrictions and further work suggested

- Only adaptable to dairy farm systems with slatted floor conditions. Plane floor and straw based systems are missing and should be integrated into the model approach for future steps of developing the model. This will at the one hand expand the possibilities of farm level decisions also to change the general animal keeping system. At the other hand it will improve the model approach to be usable and applicable to a broader variety of dairy farm systems existing in Germany.

- An improvement of the model is also possible in the area of GHG calculation and mitigation. The possibility of carbon fixation by afforestation, expanding extensive grassland usage and renaturation of peat soils should be included into the approach. This may be an advantage especially with view on up-scaling purposes to regional, sectoral or Germany wide context, then allowing for additional high impact mitigation measures which bear a highly regional dependency related to the heterogeneous natural conditions (e.g. relief, soil conditions).

- The part of farm level technology development and adjustment possibilities is up to this point only implemented very restrictive. Investment possibilities in machinery should be expanded to allow for increases of production efficiency to lower the work time demand for single activities (e.g. by time saving for feeding). The impact of technical development on emission and mitigation aspects is discussed for other sectors by e.g. Clarke et al. (2008) and Gillingham et al. (2008) for general reviews and Baker et al. (2008) for a specific study of the impact of technical change on occurring MACs.

- CO2 accounting has to be implemented, especially to add land use change and change of tillage practices
References:


Britz, W. & B. Lengers (2012): R routine to estimate Heckman two stage regression procedure on marginal abatement costs of dairy farms, based on large scale outputs of the model DAIRYDYN. Technical Paper for modelling documentation, University of Bonn


FAO (2006): livestock’s long shadow – environmental issues and options. Food and Agricultural Organization, Rome, Italy


FAO (2006): livestock’s long shadow, environmental issues and options. Food and Agricultural Organization, Rome Italy

FAO (2009): the state of Food and Agriculture – livestock in the balance. Food and Agricultural Organization, Rome Italy
References:

FNR (2013). Leitfaden Biogas. 6. Auflage, Fachagentur für Nachwachsende Rohstoffe e.V., pp. 1-244, Gülzow

GARBERT, J. 2013. Ökonomische Auswirkung von Politiken zur Umsetzung der Wasserrahmenrichtlinie auf die Schweinehaltung im Münsterland. Dissertation, University of Bonn, Germany.


KTBL (2013): Faustzahlen Biogas. 3. Ausgabe, Kuratorium für Technik und Bauwesen in der Landwirtschaft e.V., Darmstadt

References:


