MODELING FOR DUMMIES

Michael Young & John Young

YOUNGS FARM ANALYSIS & FARMING SYSTEMS ANALYSIS SERVICE

7 Aug 2023

Roles/importance of farm economic modelling

Roles:

1. Prioritise research or areas of importance within a project
2. Evaluate new information
3. Aid strategical and tactical on farm decisions e.g. Stocking Rate
4. Test hypothetical scenarios e.g. future climate

The importance of economic modelling lies in its ability to examines how an internal or external change (e.g. new innovation/technology) effects the whole farm system. This is important because the majority of the time, farmers decisions are profit driven. Indirectly this incorporates sustainability because farmers want to maximise long term profitability. Accordingly, the objective is to maximise long term profit which provides motivation to maintain a sustainable farm system (e.g. Limit soil erosion and maintain soil fertility).

Most projects would benefit from some economic modelling. Modelling as part of the initial stages of a project can help identify important areas of a project that should receive a strong focus. For example, a project looking at salt land pasture as a novel feed source for livestock could employ some economic modelling to complete various sensitivity analysis. Such as one on saltbush biomass production and one on saltbush digestibility which could indicate a 30% change in saltbush biomass has a small impact on profit but a 5% change in digestibility has a much bigger profit impact. This information would suggest that any field work being conducted within the trial should focus on obtaining more accurate data on saltbush quality rather than quantity.

Modelling in the final stages of the project can help identify the impact of the new information for farmers. Following on from the previous example, modelling cold be used to identify in what circumstances farmers should establish saltbush, how to best manage it (i.e. when to graze it) and how it could impact profitability.

Note, to get the most out of modelling it is important to include the modeler/s in the initial planning stage of the project so that the project is designed in a way that facilitates useful modelling.

Methods of farm modelling

Farm modelling is a broad topic that encompasses many methods. Simply put, a farm model is anything that reflects or represents part or all of a farm system. Examples include;

a) Gross margins and other high-level analytical tools e.g. EVALUS
b) Simulation models e.g. GrassGro, Ausfarm and APSIM
c) Mathematical programs e.g. MIDAS and AFO

Gross margins are a simple method of reflecting the costs and benefits of various components of the farm system. They are heavily input driven and rely on the skill of the user. A further limitation of gross margins is that they are generally linear, meaning that if the gross margin of 1ha of wheat is $200/ha and the gross margin of pasture is $150/ha, then growing 100ha of wheat rather than pasture will increase profit by $5 000. However, a more detailed analysis (Figure 1) shows that the profit impact is not linear, because of other interactions within the farm system for example soil type, rotation, machinery and labour usage, and grazing of stubble by livestock.

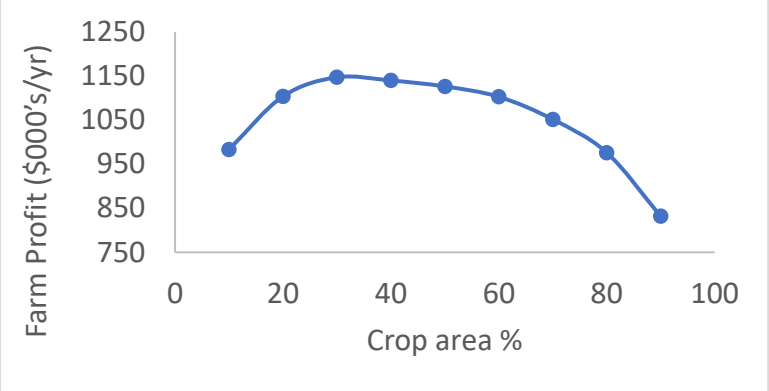

Figure 1: Whole farm profit for different levels of crop area.

Simulation modelling is widely used in Australia particularly by CSIRO. Simulation modelling is a detailed modelling method that aims to replicate the behaviour of a system. It is frequently applied to represent biological systems encompassing the whole farm or a subsection of the farm. However, simulation models do not offer an optimisation function and therefore rely on the skill of the user to test the correct scenarios. Even for skilled users this can be challenging because there are so many changing variables within a farm system. This means simulation modelling would be great for predicting the yield of a crop on a given soil type, but less ideal for determining how the inclusion of this crop into the farm system would impact farm management and economics.

Mathematical programming has also been widely used in Australia because of their detailed economic capacity and their ability to efficiently optimise the farm system. Mathematical programs represent a system using variables, constraints and an objective. Mathematical programs essentially represent a full array of management options in the form of variables and constraints which is passed to a solver, which uses mathematical algorithms to identify the optimal combination of variables. For example, a variable could be the number of 50 kg, 2-year-old female sheep, born as a single, consuming 9 megajoules of energy and gaining 50 g/hd/d. Another variable could be the same type of sheep maintaining weight and only requiring 7.5 megajoule of energy. One of the corresponding constraints is that in order to have one of these animals, it's energy requirements must be met from feed sources on the farm (e.g. pasture, supplement, stubble, etc). When the model solves, it looks at the all the options and decides which combination of variables will yield the highest income whilst satisfying the constraints. The animal above gaining more weight is beneficial because it will cut more wool and have a better mating condition, but to have that animal more feed is required, which means either feeding more or running a lower number of them. The model can simultaneously consider all this information and select the best option.

Often simulation modelling and mathematical programming can be used powerfully together. The detailed biological representation in simulation models makes them an efficient method of generating inputs for optimisation models when experimental data doesn't exist. For example, simulation modelling could be used to generate pasture production data (e.g. pasture growth and quality during the year at different levels of FOO). The role of the optimisation model is then to determine how to best manage the pasture to maximise return.

Differences to simulation modelling

Dynamic simulation (DS) (e.g. APSIM) also referred to bio-physical models aim to replicate the behaviour of a system. It is frequently applied to represent biological systems encompassing the whole farm or a subsection of the farm. Simulation modelling is very good at testing biological understanding and to compare current understanding against actual system behaviour.

However, DS models often lack a whole farm economic focus and don't facilitate efficient optimisation. This means DS models are not well suited to identifying how best to manage a farm because optimal management is highly economic driven and there are so many on-farm decisions that without an optimisation algorithm it is challenging to find the global optimum. For example, if the user is attempting to identify if they should mate yearling ewes, they will get very different answers depending on the liveweight profile used. This is challenging to optimise in a DS model however AFO can easily accommodate this.

In many cases DS and optimisation modelling are complementary because they are suited to different tasks. For example, DS models developed to imitate the biological features of a farm sub-system may generate data for use in whole farm optimisation models. Then the optimised management from the bio-economic model can be used as the management for the DS model.

Linear programming introduction

AFO uses linear programming (LP), a form of mathematical programming. Here we first provide a basic understanding of LP to set the scene for the following chapter. For a more detailed exposition of LP the readers is referred to Pannell (1997) or for even more detail McCarl and Spreen (2007).

LP is a mathematical technique that optimises a linear objective function subject to a set of linear constraints. In other words, it is a method used to find the maximum or minimum value of a particular function, given a set of conditions or constraints. This technique is widely used in various fields, such as economics, engineering, and computer science, to solve a wide range of optimization problems. The linear programming algorithm solves the LP problem which identifies the level of each decision variable that maximises (or minimises) the objective while meeting all the constraints.

A LP model comprises a set of decision variables (also called activities), a linear objective function indicating the contribution of each decision variable to the desired outcome, a set of linear constraints (sometimes called rows) describing the limits on the values of the variables and parameters (also called coefficients) that link the decision variables and the constraints. The parameters are often expressed as a matrix with the decision variables being columns of the matrix whilst the constraints are the rows of the matrix (hence why the constraints are sometimes referred to as rows). The matrix is completed with the inclusion of the objective function above the first constraint and the right hand side coefficients of the constraints are listed after the last decision variable. The "answer" to a linear program is a set of values for the decision variables that result in the best (largest or smallest) value of the objective function consistent with all the constraints.

Objective Function

The objective of a LP model is to maximise or to minimise some numerical value, which is usually farm profit in a wholefarm model. However, other objectives can also be examined, for example, to minimise farm greenhouse gas emissions provided a given level of farm profit is achieved. The objective function indicates how each decision variable contributes to the value to be optimised in solving the problem.

Decision Variables

The decision variables in a linear program are the activities that can be selected on the farm, the quantities of which are determined when the model is solved. Typically, the decision variables represent the amount of a resource to use or the level of an activity to carry out. For example, a variable may represent the number of days of labour utilised or the number of hectares of a particular crop (which can also be thought of as utilising the land resource).

Constraints

Constraints define the possible values that the decision variables of the LP model may take. They can be thought of as equations that control the level of decision variables and the relationships between decision variables.

Constraints can be:

(i) Logical or transfer constraints, e.g. the quantity of an sold must be less than or equal to the quantity produced;
(ii) limitations on available physical resources, e.g. the total area of the land uses must be less than the farm area;
(iii) limits on financial, labour or time resources;
(iv) technological restrictions, e.g. the area planted is the days of seeding multiplied by the machinery work rate;
(v) biological relationships, e.g. the productivity of an animal is linked to the amount and quality of feed consumed.

An example of a transfer constraint is transferring grain produced to grain sold. The grain produced is the area of the crop grown multiplied by the yield. If wheat yields 3.5 t/ha then equation

Wheat transfer (t): 3.5 * Area of wheat (ha) >= Wheat sold (t)

The convention is for constraints to be expressed as less than or equal constraints with all decisions variables on the left side of the inequality and the constant (or 0 if there is no constant) on the righthand side. Therefore, the above equation is rewritten as:

$$Wheat\ transfer\ (t): -3.5\ Wheat\ (ha)\ +\ 1\ Wheat\ sold\ (t)\ <=\ 0$$

This could be made more realistic by including wheat in different rotations and on different land management units, furthermore, the use of the wheat could be expanded to include both sale of the grain or feeding to livestock. E.g.:

$$\text{Wheat transfer (t)}:$$
$$= -3.5\ \text{Wheat after canola (ha)} - 2.5\ \text{Wheat after wheat (ha)}$$
$$+\ 1\ \text{Wheat sold (t)}\ +\ 1\ \text{Wheat fed (t)}\ <=\ 0$$

An example of a resource constraint is the requirement for labour at seeding.

An example of a simple biological constraint is the number of sheep that can be carried on each hectare of pasture. This might be modelled as the number of DSE that can be carried on a hectare of pasture and the number of DSE per head for each class of sheep. If the carrying capacity of the pasture is 12 DSE/ha and ewe are 1.5 DSE/hd then the following equation could be written to describe that the carrying capacity of the pasture (12 * area) has to be greater than the grazing intensity applied by the sheep (1.5 * number of ewes)

$$Carrying\ capacity\ (DSE): 12\ *\ Area\ of\ pasture\ (ha)\ >=\ 1.5\ number\ of\ ewes\ (hd)$$

Converted to:

$$Carrying\ capacity\ (DSE): -12\ pasture\ (ha)\ +\ 1.5\ ewes\ (hd)\ <=\ 0$$

A feature of using the convention that the constraints are '<=' equations is that decision variables that 'provide' into a constraint have a negative coefficient and decision variables that 'require' the resource have a positive coefficient. In the above example, pastures 'provides' carrying capacity and the ewes 'require' carrying capacity.

Note: In the equations above, notice that the units for the constraint and the decision variables have been included. Tracking the units is useful when building and debugging an LP model. The units of the decision variables is usually straight forward, however, sometimes the constraints are less clear. This is when thinking about the constraint as 'utilising a resource' is helpful. What is the resource being utilised (or item being transferred) and what is the unit of that resource?

Parameters

A further advantage of being clear of the units for the decision variables and the constraints is that it makes it simple to determine the units for each parameter in the constraints. The unit for each parameter is the unit of the constraint per unit of the activity. For example, in the equation above the Carrying capacity constraint has units of DSE. The pasture is in units of hectares so the units for the pasture parameter is 12 DSE per hectare. Likewise, the unit of the 'Ewe' is head and therefore the unit of the ewe parameter is 1.5 DSE per head.

In the above example the units of the constraints, the decision variables and the parameters are quite obvious. However, sometimes the constraints are created in less intuitive ways to improve the representation of the biology. This is usually done so the variation in the parameter can be applied on the decision variables that are less intuitive.

Righthand side coefficients

The right-hand side (RHS) coefficient is the amount of resource available (for a ≤ constraint) or the minimum requirement of some criterion (for a ≥ constraint). Typically expressed as a constant for sensitivity analysis.

Often, as in the constraints above, the RHS value is 0. In which case the provide and require components of the constraint are both controlled entirely by the level of the decision variables.

RHS values can be adjusted to calculate:

- Shadow Price: The magnitude of the change in the objective function value for a unit increase in the RHS of a constraint.
- Slack: The difference between the RHS and the left-hand side (LHS) of a ≤ constraint. Typically represents the unused resource.
- Surplus: The difference between the LHS and the RHS of a ≥ constraint. Typically represents the level of over satisfaction of a requirement.

Summary

A functional way to think about an LP is that the decision variables are representing different options that a farmer can choose from when managing their farm. The more options that are represented in the model (with respect to each decision the farmers can make) the greater the choices that are being represented. Including more options and differentiating the outcomes from those choices requires better data, this trade-off is part of the decision about the level of detail to represent in the model. The other component to the detail to represent is the importance of the decision, the

computing capacity required to the solve the model and the skills required to design the model and interpret the output.

Representing non-linear relationships

Although the word linear features in the term linear programming, it is possible to approximate non-linear relationships within an LP framework. This occurs by applying what is known as piecewise representation, where non-linear relationships are closely approximated by linear segments. See Figure 2 for an example of linearising a curve with 5 segments created with 5 decision variables. An efficient matrix for representing a non-linear production function follows. In addition to the constraint that transfers the resource ('Resources') it requires an extra constraint, 'Piecewise limit'. The limit constraint ensures that if 5 units of the resource are applied that it is done with 1 unit of the x5 decision variable rather than 5 units of the x1 decision variable. Note: 5 units of the x1 decision variable would produce more output than 1 unit of the x5 decision variable.

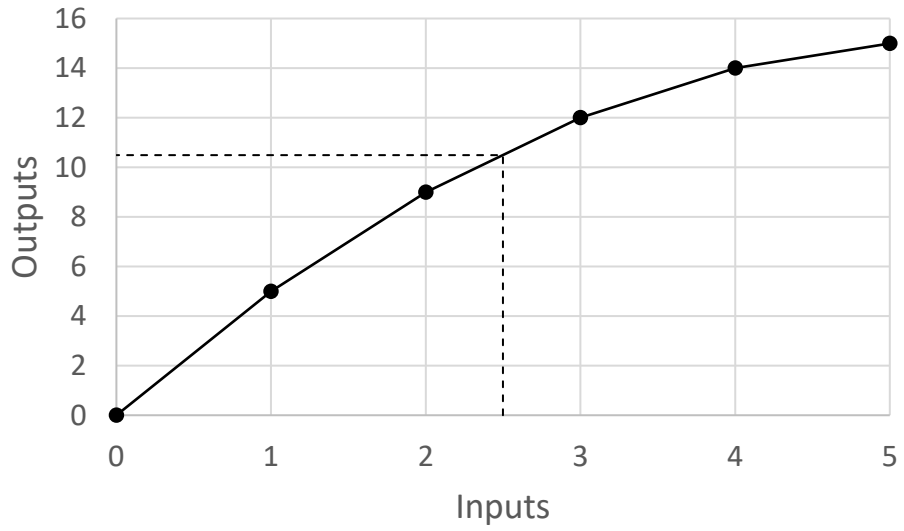

Figure 2: linearising a convex function showing the level of output for a maximisation problem if resource use is limited to 2.5 units

Variables are:

1. Resources applied (kg)
2. Area of crop (x1, x2, x3, x4, x5) with different levels of the resource (ha)

Constraints are:

1. $Resources\ (kg): -1\ Resource\ applied\ +\ 1\,x1\ +\ 2\,x2\ +\ 3\,x3\ +\ 4\,x4\ +\ 5\,x5\ <=\ 0$
2. $Piecewise\ limit\ (ha): x1\ +\ x2\ +\ x3\ +\ x4\ +\ x5\ <=\ 1$

Objective: $max(5\,x1\ +\ 9\,x2\ +\ 12\,x3\ +\ 14\,x4\ +\ 15\,x5 - cost\ *\ Resources\ applied)$

Where 5, 9, 12, 14 and 15 are the output from the given level of inputs as per Figure 2. Making the assumption that the product value is $1/unit.

Linear segmentation of non-linear relationships is a feature of most LP models of farming system because of the biology being represented. It is most often done to represent decision variables that are non-linear with respect to the production function or objective function. An example of this is the response of wheat to inputs of nitrogen, this can be represented as several wheat decision variables each with a different quantity of nitrogen applied and different yield. A similar but less

obvious example is the response of animals to nutrition which can be represented as multiple decision variables with different energy intake. The nitrogen nutrition of a crop and the nutrition of an animal are the same concept, however, the representation for animals is more complicated because there are more interactions between periods for the animal than the crop. For example, nutrition in the current period affects the animal liveweight in subsequent periods which affects maintenance requirement and intake capacity in subsequent periods.

There is an important proviso when linearising no-linear functions that is related to the slope of the non-linear function. For a maximisation problem the convex function in Figure 2 will be represented as graphed by linear segments joining adjacent points and the optimal solution if constrained to 2.5 units will be the output of 10.5 being the average of 2 units and 3 units. Whereas the concave function in Figure 3 will not be represented correctly with the matrix design above. With that matrix the optimal solution if constrained to 2.5 units will not be 4.5 (the average of 2 units and 3 units of input), instead it will be 7.5 being the average of 0 and 5 units i.e. half a unit of x5 will be selected to provide 7.5 units of output while requiring 2.5 units of input. This is presented graphically by the heavy dashed line in Figure 3.

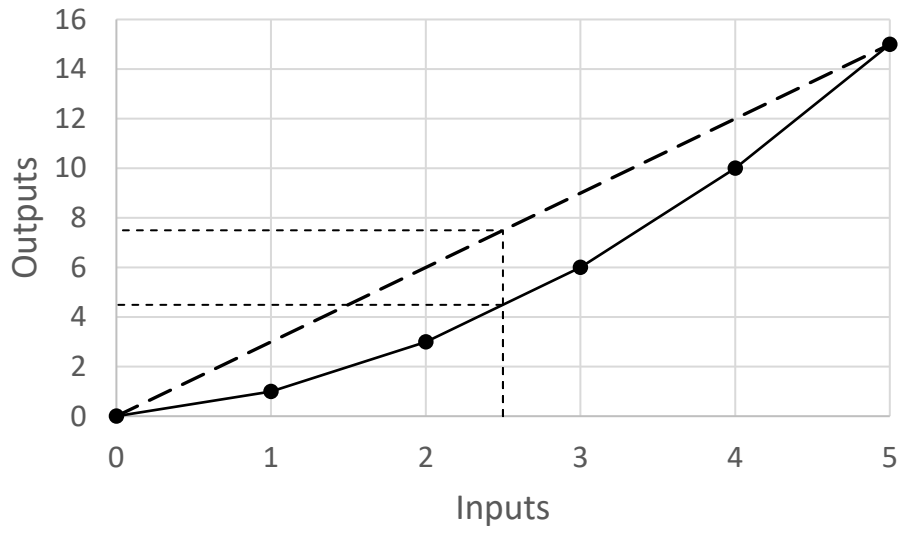

Figure 3: linearising a concave function showing the level of output for a maximisation problem if resource use is limited to 2.5 units.

The above examples are of non-linearity with respect of the objective function or the production function. The non-linearity can also be associated with the constraints or the quality of the resources, which is difficult to represent graphically, however, it is handled in a similar manner, but in this case by replicating the constraints rather than the decision variables. The common terminology is resource 'pools', each constraint represents a discrete resource pool and the supply of resources and the requirements for resources are allocated to different pools.

An example of non-linearity in the resource quality is farm labour. The owner manager is usually the most skilled labour, with permanent labour also being skilled whereas casual labour is less skilled. Therefore, there are some jobs eg. Business management decisions, that can't be done by casual labour. This can be represented by having several labour pools (manager, permanent and casual) with the jobs and the labour units allocated to different pools.

The nutritive value of feed is another example. Animals produce more and gain more weight when fed higher quality feed. It is further complicated by the different scarcity (cost) of feed by quality

with high quality feed being more expensive. Again, this can be represented by including different nutritive value pools that constrain the quantities of feed with different quality.

Modelling options

When developing MP models there are often multiple ways to represent the same concept and it is the model developers role to select the representation, but it necessary for the model user to understand the implied assumptions (such as in the above example that hiring labour was not an option available). This has multiple levels including

1. Selecting the boundary of the model. For farm modelling this is usually the resources under the individual farmers control i.e. a single farm, however, this could be expanded to a catchment or a whole industry. The different boundaries might affect, for example, including the impact of changing production levels on the price received for products based on the quantity produced.
2. The breadth of the decision that will be optimised. There is almost no limit to the decisions that could be represented because farmers can and do make a myriad of decisions. There is a question for the model developer, what decision options will be included in the standard model. There is also a question for the analyst using the model, what questions are best addressed using the model and which are best addressed with a simpler framework such as a gross margin or a partial budget.
3. The level of biological reality that will be represented in the model. This is usually linked to the number of decision variables and constraints included in the model (presuming that the model is designed efficiently).
4. Matrix structure, an aggregated or disaggregated matrix.

The LP algorithm and how the model is stored

There are 2 alternatives about how the model is presented in a text file for reading, which align with how the model interfaces with the LP algorithm. Both require defining the decision variables and the constraints. The difference is in how the parameters/coefficients are stored.

1. MPS format – used in MIDAS. In this format the constraints are referred to as Rows and DVs as Columns. The format is activity centric with the coefficients linked to the decision variables. The MPS format for the example in 'Non-linear relationships' is:

```
NAME   NonLinear
ROWS
 N  Obj
 L  Resources
 L  PiecewiseLimit
COLUMNS
   Resources Applied   Obj            -cost
   Resources Applied   Resources      -1.0
   X1                  Obj            +5.0
   X1                  Resources      +1.0
   X1                  PiecewiseLimit +1.0
   X2                  Obj            +9.0
   X2                  Resources      +2.0
   X2                  PiecewiseLimit +1.0
   X3                  Obj            +12.0
```

```
    X3              Resources      +3.0
    X3              PiecewiseLimit  +1.0
    X4              Obj           +14.0
    X4              Resources      +4.0
    X4              PiecewiseLimit  +1.0
    X5              Obj           +15.0
    X5              Resources      +5.0
    X5              PiecewiseLimit  +1.0
RHS
    FirstRHS          PiecewiseLimit  +1.0
ENDATA
```

When viewing a MPS file the thought process is relating to examining which resources an activity requires (+ve coefficient) and which resources it provides (-ve coefficients). This is straightforward when it is a simple resource constraint but more difficult when it is a transfer constraint. For a transfer constraint in a large MPS file it is necessary to locate the other activities that provide or require that item to be sure that the transfer is operating correctly.

2. Equation format – used in AFO. This format is constraint centric as in the format shown in the original 'Non-linear relationships' example. This format appears to be the industry standard now and is the format used in pyomo and AFO. This is because it is easier to mathematically represent multi-dimensional variables using this format.

Farm example

A farmer wishes to maximise their profit from either growing canola or having pasture and running ewes.

Canola yields 2.0 tonnes per hectare, sells for $800 a tonne and the chemicals, fertiliser, seeding operation and harvest costs are $500/ha. Net income $1100/ha.

Ewes are purchased for $100/hd and are sold at the end of the year for $80/hd, having each produced a lamb for sale at $150/hd and produced $40/hd of wool. The costs of husbandry, mating, shearing and supplementary feed are $45/hd. Ewes are 1.5 DSE/hd. Net income $125/ewe.

Pasture fertiliser is $50/ha and 12 DSE/ha can be carried per hectare of pasture.

The total farm area is 1000 hectares.

Canola can only be grown after a 2 year break for disease management (i.e. for every hectare of canola there must be two hectares of pasture).

There is one labour unit available on the farm during seeding and they work a 12 hour day, the seeding period is 10 days and the sowing capacity of the seeding rig is 4 ha per hour moving with a 75% field efficiency. During the seeding period the ewes require feeding twice per week and this takes 2 hrs for a mob of 1000 ewes.

The decision variables are:

1. Canola area (ha)
2. Pasture area (ha)
3. Ewes (hd)

The constraints are:

1. $Total\ area\ (ha)$: $1\ Canola\ +\ 1\ Pasture\ \leq\ 1000$
2. $Canola\ break\ (ha.yrs)$: $2\ Canola\ -\ 1\ Pasture <=\ 0$
3. $Carrying\ capacity\ (DSE)$: $-8\ Pasture\ +\ 1.5\ Ewes\ \leq 0$
4. $Labour\ (hrs)$: $0.333\ Canola + 0.006\ Ewes \leq 120$
5. $Non-negativity$: $Canola, Pasture, Ewes\ >=\ 0$

The objective is: $Max(1100\ Canola -\ 50\ Pasture + 125\ Ewes)$

This problem is simple enough to solve by eye. The gross margin of canola ($1100/ha) is greater than for pasture and ewes ($950/ha), so canola will be grown in preference to pasture with ewes. Constraints 1 & 2 together mean the maximum area of canola is 333.3 ha with 666.7 ha of pasture, which would be 5333 ewes. Constraint 4 would allow 360ha of canola if there were no ewes however, with 5333 ewes (which requires 32 hours of labour), only 264 ha of canola can be grown from the available labour. The alternatives are therefore:

1. Reduce the area of canola to allow the extra ewes to be carried
2. Grow 333ha of Canola and fewer ewes (3333hd) so that the labour constraint is met. In LP jargon this would be called having 'slack' on the carrying capacity constraint. This solution may be counter-intuitive because it common to think that the limitation on the number of sheep carried is the quantity of feed available. In fact, this is an implied assumption in most gross margin analyses, however, as in the above example labour can be a limiting resource or if incorporating methane emissions in an analysis the level of emissions may be limiting rather than carrying capacity.
3. A practical solution, that is not part of the above LP representation, is to hire labour. It is often useful to consider the binding constraints and consider if there are alternative ways of providing the limiting resource that are not represented in the LP model.

In the problem as formulated the optimum solution is to reduce the area of canola to about 220ha and have 780ha of pasture carrying 6240 ewes. However, if the gross margin of pasture with ewes was less than $666/ha (rather than $950/ha) then the optimal solution would be to have slack on the carrying capacity row i.e. the maximum area of canola would be grown (based on the disease break constraint) and the number of sheep would be limited by labour rather than feed availability.

The above problem is simplistic. Further detail could be added such as including other land uses, linking crop yields, costs and pasture carrying capacity to the land use history, representing a self-replacing flock and including interactions between the enterprises such as the benefit of grazing crop stubbles. These additions would make the representation of the farming system more realistic but would require far more complex matrices.

AFO

Full documentation can be found here: https://australian-farm-optimising-model.readthedocs.io/en/latest/index.html

Australian **F**arm **O**ptimisation Model (AFO) is a whole farm bio-economical optimisation model based around an Australian farm system. AFO is the successor of the well-known MIDAS model and represents the biology and economics of an Australian farm system in a linear programming framework. AFO has been built with flexibility in mind which allows the user to customise the level of detail represented (e.g. number of crops and soil types) and allows AFOs to easily facilitate future development.

AFO has been written in Python and therefore requires some Python programming skills, an editor and the Python interpreter installed on your computer. There are several options for editors and the AFO development team are currently using the PyCharm editor. For instructions to instal PyCharm and Python check this. There are several courses on the internet that introduce Python and these are a good way to get started. The official introductory page on the python.org site is available here to get started in interpreting the structure of a Python program and the basic syntax, however, other sites are easier to follow (see here for a long list). The data for AFO is stored in spreadsheets so some basic Excel skills are also required to interact with AFO.

The AFO code includes modules for rotations, crops, pastures, livestock, stubble, supplementary feeding, machinery, labour and finance. Additionally, it includes land heterogeneity by considering enterprise rotations on a range of soil classes. Furthermore, AFO includes a detailed representation of uncertainty, including price weather variation and farmer risk attitude. AFO includes a large array of short-term management tactics that can be used to respond to unfolding opportunities or threats to generate additional income or to avoid losses. The model is built and calibrated such that it represents current farm management technology insofar as the types of machinery complements, herbicides used and rates applied. Tasks contracted and crop and livestock options considered are all consistent with those used by farmers in the modelled region.

The core units of the model are:

i. Inputs.

ii. Precalcs.

iii. Pyomo and solver.

iv. Reporting

and the general flow of data in the model is: Inputs → precalcs (calculations using the inputs) → pyomo (compiles linear model) → solve → report.

Inputs

The model inputs are stored in three excel spreadsheets.

i. **Property.xlsx** which contains inputs that are likely to change for different regions or properties such as farm area
ii. **Universal.xlsx** contains inputs that are “universal” and likely to be consistent for different regions or properties such as prices.
iii. **Structural.xlsx**, the structural inputs such as the number of liveweight options represented in the model, which are typically fixed and only changed by experienced users.

Precalcs

The precalcs are the calculations applied to the input data to generate the data for the Pyomo parameters (in other terms, the conversion of the inputs to the parameters for the LP matrix). The precalcs for each individual trial (trial is the name for a single model solution) can be controlled by the user with the ‘experiment’ spreadsheet which allows inputs from the three input spreadsheets to be temporarily adjusted , or the intermediate calculations in the precalcs to be temporarily adjusted.

Pyomo and solver

This is the LP component of the model. It defines all the decision variables, the objective function, the constraints and parameters then utilises them to construct the model equations (i.e. the constraints). Components of the LP model can also be temporarily adjusted by the user via the 'experiment' spreadsheet. Pyomo formulates all the equations into a linear program format and passes the file to a solver. AFO has multiple compatible solver options. Most frequently used are CPLEX (Cplex, 2009) and GLPK (Makhorin, 2014). When tested both solvers resulted in the same answer. CPLEX has some advanced features unavailable in GLPK. However, GLPK is open source whereas CPLEX is costly proprietary software.

Reports

There are 2 levels of reports that can be generated from AFO.

1. Full output generated from the solver. This includes all the levels of all the decision variables and all the constraints. It can also include shadow costs and shadow prices. This output is very useful when debugging the model but is not usually used when doing a routine analysis.
2. Custom designed reports that can combine the level of decision variables with values generated in the pre-calcs. For example, reporting the total FOO on the farm by summing the pasture area decision variables multiplied by the FOO per hectare for each variable. These reports are defined in the ReportControl.py module.

Simple walk through example – Labour cost for seeding

This example is a somewhat simplified example but will hopefully provide an understanding of some of the underlying concepts in AFO.

Inputs

Labour supply:

- Cost per hour
- Cost of super
- Cost of workers compensation
- Hours worked per weekday
- Hours worked per weekend
- Hours worked during seeding
- Hours worked during harvest

Labour	
manager_cost ($/yr)	80000
permanent_cost ($/yr)	80000
permanent_super	9%
permanent_workers_comp	4%
permanent_ls_leave	0.023
casual_cost ($/hr)	28
casual_super	9%
casual_workers_comp	4%

Daily hours and efficiency

	Manager	Permanent	Casual
weekdays	9	8	8
weekends	4.5	0	0
seeding	10	9	8
harvest	9	9	8

Supervision required

	Permanent	Casual
normal	0.07	0.15
seedingharv	0.02	0.1

Labour required for seeding:

- Seeding preparation
- Seeding hours per day
- Helper time required

Seeding preparation		
date	days	hours
63	30	40
94	30	40
166	14	5

Machine hours	
daily_seed_hours	10

Helper	
seeding	0.5

Precalcs

Precalcs for this example are relatively simple. The inputs are manipulated to calculate the hours of labour supplied by each unit of staff in the seeding period and the labour required to seed 1 ha of each rotation on each soil type (seeding can be slower on heavier soils).

A simple way to search for the precalc code that is relevant to the spreadsheet inputs is to "search all" in the code editor (Shft-Ctrl-F in PyCharm) for the range name of the input from Excel.

Pyomo

Variables – parts of the farm system that can be changed/optimised:

- Number of staff
- Number of hectares of cropping rotations

Constraints – constraints within the farm system that must be satisfied

- Each hectare of cropping rotation requires a hectare of seeding which requires labour

```
def f_con_labour_phase_anyone(model):
    '''
    Tallies labour used in the crop enterprise that can be completed by anyone (casual/permanent/manager) and ensures
    that there is sufficient labour available to carry out the jobs.
    '''
    def labour_crop_anyone(model,q,s,p,w,z):
        if pe.value(model.p_wyear_inc_qs[q, s]):
            return -model.v_phase_labour_casual[q,s,p,w,z] - model.v_phase_labour_permanent[q,s,p,w,z] - model.v_phase_labour_manager[
                q,s,p,w,z] + lphspy.f_mach_labour_anyone(model,q,s,p,z) <= 0
        else:
            return pe.Constraint.Skip
    model.con_labour_crop_anyone = pe.Constraint(model.s_sequence_year, model.s_sequence, model.s_labperiods['any'],model.s_season_types,
                                                 rule=labour_crop_anyone,
                                                 doc='link between labour supply and requirement by crop jobs for all labour sources')
```

Figure 4: CorePyomo constraint that links labour supply with labour requirement for seeding.

```
def f_mach_labour_anyone(model,q,s,p,z):
    '''
    Calculate the total labour required by anyone for seeding.

    Used in global constraint (con_labour_anyone). See CorePyomo

    '''
    seed_labour = sum(model.v_seeding_machdays[q,s,z,p,k,l] for k in model.s_landuses for l in model.s_lmus)        \
    * model.p_daily_seed_hours *(1 + model.p_seeding_helper)
    return seed_labour
```

Figure 5: Function that calculates the labour required for one day of seeding. This gets utilised in the constraint above.

Overall

Basically, the model optimises the number of hectares to sow, but in order to sow there must be sufficient labour to complete the activity. Manager and permanent labour is limited and casual labour costs money but the crop will grow and provide cashflow in the future. The model compares other alternative pathways for example, using the land for pasture and grazing it with stock which provide income in the form of wool and meat.

Essentially, AFO uses inputs to generate many different options/pathways within the farm system and then processes all the options simultaneously in an external solver and identifies the optimal farm strategy and the associated profit.

Improvements of AFO over MIDAS

Usability

- Open source and free
- Built in python
- Version control via GitHub
- Online and automatically updating documentation
- Multiprocessing capacity
- Easy integration with cloud computing
- Multi solver options
- Easy integration with simulation models (e.g. APSIM, AusFarm)

Model features

- Scalable between regions/properties
- Improved livestock representation
    - Stock liveweight / nutrition optimisation based on feed variation periods
    - Sale time optimisation
    - Differential feeding of dams
- Improved feed budgeting
    - Multiple nutritive value pools
    - Confinement feeding
    - Novel feed sources e.g. crop grazing
    - Multiple pasture types (e.g. annual pasture, perennial pasture)
- Flexible rotation system based on rotation phases
- Farmer risk attitude
- Price variation
- Weather variation

The modelling mindset

- What do the decision makers want to know. Who is your analysis aimed at (farmers, policy makers or researchers). How is your analysis going to answer their questions.
    - Eg. If farmers are considering adopting new technology they will need to know what other aspects of the farm need to be changed to capitalise. Eg. If adopting summer active perennial species then this will be best utilised in a meat rather than wool system.
- What sensitivity analysis is required to test the robustness of the model and the robustness of the solution.
- Is the model representing enough options to allow the optimisation to actually optimise, or is the potential to optimise being limited by a minimal range of options included in the model.

When carrying out an analysis, prior to starting create a picture of the likely results. Think through the above questions to ensure that the model is sufficient for the analysis to be carried out. This idea of the results that you expect also helps identify model and analysis errors if there is discrepancy. Sometimes the discrepancy is due to errors in the model or aspects that are not well represented or it could mean that there is a behaviour of the farm system that you didn't fully understand, so it is a learning opportunity.

These expectations about the analysis results are best gained if you can adopt an LP mindset. Considering the resources that are limiting in the base case, do you expect the limiting (or binding) resources to change and how the resources will be allocated in the new solution. The optimal allocation of resources is such that the marginal value of the resource is equal in all the areas where it is allocated. Think MV of feed to different classes of stock in different periods of the year.

Interpreting results

One of the most difficult task of whole farm modelling is interpreting the results of an analysis and relating them back to the management of the farm system being analysed. The complexity is related to the fact that the model, however good it is, is only an abstraction of the real farm system. Furthermore, the representation of the management question has to be within the bounds of the model design. When interpreting the results both of these abstractions need be considered to determine the implications of the model result for farm management. This is difficult because it requires a good understanding of the farm system and a good understanding of the farm model.

The usual approach to interpreting an analysis is a preliminary interpretation of the model results assuming that the model and the analysis are a good representation of the problem to be solved. Then asking the question, what are the results suggesting that the farmer (or policy maker) could change to improve farm profitability? In many situations this will highlight extra questions that need to be answered before there is confidence in the interpretation of the results.

After this preliminary interpretation it is necessary to then consider the model assumptions and the analysis assumptions that are the 'abstraction'. This can be facilitated by considering what production relationships, what management constraints and what farmer objectives are not represented in the model (or not well represented). Then consider, how would including those factors change the preliminary interpretation.

Creating expectations about analysis results (as discussed in the previous section) and then testing those expectations against the model results is an effective way to build understanding about the

system. It is this understanding that is necessary to quantify the implications of the modelling and analysis assumptions discussed in the previous paragraph.