Developing a Flexible Software Framework for Nutrition Models: A Platform for the NRC Nutrient Requirement Series

M. D. Hanigan, R. R. White

Virginia Tech and the National Animal Nutrition Program (NRSP-9)

Introduction

Currently, NRC Nutrient Requirement models and the supporting software interface are developed independently for each species. The approach taken in developing these software releases is dependent upon the support available at the time the revision is conducted. This results in inefficiencies associated with recreating common elements, and problems associated with multiple pieces of independently developed software. These issues increase the cost of development and maintenance, resulting in less frequent updates and fixes between releases. A common software framework could accommodate models from all of the species, reduce programming time for each release and ease software support between releases, and potentially shorter intervals between releases. This would increase the confidence of stakeholders on the NRC models as it would provide up-to-date systems with fewer programming bugs.

A common interface with common language and acronyms would also allow easy navigation from one species to the next. This would be particularly helpful for teaching purposes. If the interface were clean and simple, more time could be spent on teaching the concepts of ration formulation and less on teaching the intricacies of software navigation. This would be particularly helpful in classes teaching nutritional concepts across species.

A fully functioning ration balancing program controlled and supported by the research community would also allow early deployment of new nutritional knowledge for the purpose of field demonstration and conducting extension work. For example, workers at Virginia Tech have developed a revised phosphorus supply system. This system is not currently resident in a ration balancing program, and thus it cannot be easily demonstrated in the field, nor can extension people and early industry adopters use the new system. If this system were incorporated into a ration balancer , it could be used to demonstrate the value of the new knowledge, to collect field data on use of the system, and to build confidence in the system before full deployment in commercial packages or the next NRC revision. This would help assure adoption of the system. The Virginia Tech team and their collaborators have found this to be a particularly acute need when developing integrated research proposals. A logical extension of nutrition work conducted in an integrated grant is the deployment of the new information on a number of farms. Having a ration balancing platform that could be modified to support that work would be of great benefit in terms of increasing the pace of adoption. It would also ease incorporation into other commercial software packages as the method of incorporation would have been previously defined.

There are a number of individuals in the animal science community with advanced computer programming skills that could serve as a resource to maintain the code over time if the project

were treated as open source. Obviously this could result in lower maintenance costs as the community could assume responsibility for many or perhaps all updates to address operational issues as well as addition of new functionality. This would aid in adoption of new information and equations by commercial software companies as methods of application could be worked out by the research community as new biological and conceptual information are discovered and published in the scientific literature. There may also be opportunities for programming support from companies with proprietary ration balancers in exchange for early access to new concepts and equations.

The objective of this paper is to describe a software framework that would support a generalized platform for NRC Nutrient Requirement models for multiple species. The goal is to identify data driven methods that result in auto configuration of the software for each species at the time of the execution of the program (i.e., runtime). These concepts apply whether the code is maintained as open source or closely held. This effort does not aspire to define all of the intricacies needed to carry out the project, only the broad concepts and some of the species specific information that needs to be considered. It is important to recognize that the goal of this work is not to develop software that will displace commercial efforts. The goal is to develop a tool that can be used for teaching, research, and outreach activities and as a testbed for new ration evaluation and formulation concepts derived from research efforts.

Program Overview

Regardless of species, the information needed to set up a feeding program, to evaluate its nutritional adequacy, and to report results has a similar structure and function. The primary components are: 1) program settings controlling various items such as the units, the analytical basis, level of solution, screen display lists (primarily on the ration specification page), and report formats (**Program Settings**); 2) animal descriptors including breed and breeding, body size and condition, physiological state, production level and composition, and activity level (**Animal Specifications**); 3) climatic conditions that affect nutritional requirements (**Environment Specifications**); 4) a database to store feed ingredients and rations (**Feed and Ration Library**); 5) methods to edit the feed library (**Feed Library Maintenance**), 6) methods to define and edit rations (**Ration Specification**); 7) a mathematical model that accepts feed and nutrient inputs, animal characteristics, and environmental conditions, and calculates nutrient supply and requirements (**Model**); 8) a set of reports that list animal characteristics, nutrient requirements, nutrients supplied by the ration, projected production, and excretion of environmentally important nutrients (**Reports**); and 9) an algorithm to balance the ration to provide adequate nutrients while minimizing the cost of the ration or maximizing the profit (**Optimizer**). Although optional and not included in previous NRC programs, an optimizer is a needed component of teaching programs to help develop capable future industry professionals, for research programs to allow easier development of research trial diets, and for extension programming and application of research results in the field in association with integrated grants.

The framework that will be described to achieve the goal of a common interface is not unique. Several commercial ration balancing programs use these concepts to support multiple species.

Data Driven Program Configuration

To accommodate multiple species, the stand-alone (i.e., compiled) software must be written to configure itself based on a species specification and a set of species specific configuration data that can be read at startup. That is, the program is conceived, written, and compiled to respond to input data and configure the program based on that data. A portion of the configuration involves listing the appropriate items on each screen. However, there are also interactions among the components, and thus a portion of the framework must be devoted to establishing the proper links between components to allow transfer of the correct information. Such links would be predefined for each species.

For example, the Animal Specifications for a lactating dairy cow are currently as listed in Figure 1. Eleven items are required. From a software development point, it is irrelevant what those 11 items are. The code simply needs to display a form with a grid that has 3 columns and 11 rows. That grid will be populated from a list of labels, a list of cross-reference links to the appropriate database fields that hold the default or selected values, and a list of unit labels. Those 3 lists can be predefined for each species and read at startup to populate the form. The number of rows can expand or contract to fit the needs of each species. Additional cross-referencing lists are needed to specify which of the Animal Specifications are to be displayed on other forms, passed to the model, and listed on the reports. So essentially, each form and component in the code becomes a shell that is populated based on species specific lists and cross-linking specifications.

The data to drive the configuration would be held in a data table with species as one of the data identifiers. The species would be identified either at install (assuming the user may have purchased the software for a given species) or at startup (the user is asked which species they want to work with). The input data can be in the form of a file or database or the settings could potentially be stored in the Computer Registry.

The forms, lists, and cross-links will be defined in more detail in the following sections which are devoted to each of the main software components. For purposes of discussion, the Dairy NRC 2001 program has been used as an example. The installation file for that program can be downloaded from www.nanp-nrsp-9.org/nrc-dairy-model/.

Program Settings

Program Settings are primarily user selections and, in general, these setting choices will not vary by species. However, they may vary by ration and customer thus they should be stored with the ration or customer data rather than with species configuration data. In all species, choices of the units (metric or imperial) and the basis for calculation (DM or as-fed) is needed. All data should be stored in metric units and converted to imperial units for display purposes. These common items are listed in Table 1.

In addition to the common items above, it is useful to be able to control the items that are displayed on the Ration Specification screen to aid in setting up and solving the problem. The selections offered would generally be the nutrient concentrations or amounts provided by the entered ration and the corresponding requirement. However, a selection of nutrient ratios are often used, and the user may be interested in predicted model outputs such as the grams of P or N excreted each day. Examples of these include: the percent forage in the ration, the Ca:P ratio, net energy balance, metabolizable protein balance, net energy and metabolizable protein allowable milk or growth, etc. Most of these items would also be displayed on the reports. These items

must be calculated from the ration, and thus are extraneous to the nutrient list. The species selected or defined at startup should filter the nutrient list to only those for the species of interest. A partial listing of the derived values that should be offered for display is provided in Table 2. When implementing an optimizer, the user may also desire to place constraints on many of these derived values.

Customer Information

Although not essential, it is useful to be able to store rations and ingredients by customer as the ingredients used on a particular farm may be unique to that farm. It is also helpful in keeping a historical record. This has value from a teaching standpoint as this is a concept in all commercial packages. Thus, if a database of ingredients and rations is to be established, a customer ID is needed as a record identifier for both data tables.

The information required could be as simple as a customer ID and customer name. One could also collect contact information, but again, this is not essential information.

Animal Specifications

Animal specifications include a mix of items that are specific to a species and those that are common across species. Items that are common include: breed and breeding type, age, sex, body weight, and current and mature body weight. Items that are not necessarily common across all species include: reproductive state, length of gestation, age at first calving, pigs/litter, days in milk, eggs per year, exercise level, body condition score, milk composition, body composition, etc. Line within breed is currently important for swine and poultry but not for other species. Efforts are underway to develop genomic prediction equations for traits of interest such as feed efficiency, but there will be additional nutragenomic traits of interest derived from future research that must also be accommodated. As these are developed, the requirement models will be configured to use that genetic information to nutrient supply and requirement predictions as dictated by the genomic characteristics of the group of animals being modeled. Thus such an event must be considered when planning an input scheme. Eventually, there will be several traits that would be defined by genomic prediction equations.

The structure of the data table for breed and physiology factors is presented in Table 3. Species selection would determine what data fields are displayed on the animal input screen.

The screen presented to the user would contain a data grid for display of the data field labels, the units for each field, and data entry fields for user input associated with each of those data items. The data grid would be oriented in a column format so that the length of the grid, which is determined by the species selection, could automatically extend downward. Selection of a species at startup or install would dictate what data labels and units were populated in the grid and the cross-reference for storage of entered data in the database. If the grid extended beyond the length of the form, a slider bar would allow user negotiation to lower parts of the grid. An example of the form is depicted in Figure 1.

Data entry should be aided by use of the TAB key to move between fields without changing field values and the ENTER key to execute completion of data entry and move to the next field.

Environment Specifications

Some species models make use of information on the animal environment to adjust nutrient requirements. For example, animals that must walk significant distance to obtain feed and water have higher maintenance requirements than those who are more sedentary. Also, heat and cold stress are important components of calculation of maintenance requirements for some species. Additional components in some models include indicators of animal stress such as the floor space per pig in swine operations. Potential future items for consideration may include the level of stress emanating from infectious disease problems in the facility. For example, if the facility typically has an outbreak of salmonella during the production cycle and 20% of the animals are affected, this may be information that could be entered and used by the program to adjust feed intake, nutrient supply, and maintenance requirements for the group. A list of environmental factors that should be considered today in provided in Table 4. This list will clearly expand as more knowledge of environmental factors is developed, and some categories that may not be used today, i.e. heat mitigation system, clearly will be used in the future.

Ingredient Library

Data tables are required to hold a list of ingredients that can be used in rations and a list of rations. The ingredient table or library would contain a comprehensive list of ingredients compiled from the latest species-specific NRC committee. Additionally it would allow user-defined ingredients reflecting regional and on-farm feeds. These user-defined ingredients would be copied from the list provided with the program and edited to reflect the characteristics of the particular feed of interest. As collecting all of the needed information for a given ingredient is typically not possible, copying from an existing ingredient with subsequent editing is the preferred method of entry, and likely should be the only method of entry to ensure the data are complete for the new ingredient.

The ingredient data table would preferably be common across species, although a species specific database could be constructed at the time of compilation or installation. The latter would only seem necessary if database performance is an issue. Use of a common table across species does result in a larger database mostly due to the addition of multiple species specific nutrient fields. For example, the energy systems are generally specific to each species, and thus one would need an energy value for each species for each ingredient resulting in perhaps 6 or more energy fields (NEm, NEg, and NEl for cattle, ME for swine, ME for chickens, and ME for turkeys). The same would likely be the case for metabolizable or absorbable protein and available literature supports unique values for ileal digestible AA for swine and poultry (Lemme et al., 2004; Stein et al., 2007).

A full list of nutrients for use across species will likely represent in excess of 200 items. Of those perhaps 40 will be used for any given species. Thus a table listing which nutrients are to be used for each species would be constructed. This table would be used to populate the viewing and editing data grid on the ingredient maintenance screens, the list of nutrients available for

display on the Ration Specification page, the list of nutrients that can be constrained on the Ration Specification page (if using an optimizer), the list of nutrients to be summed for the final ration, and the list of nutrients to be listed in Reports of ingredient and diet nutrient composition.

A complete feed library representing the combined observations from the Swine, Dairy, Poultry, and Beef NRC publications has been constructed (see http://www.nanp-nrsp-9.org) and represents source data that would be used for this multispecies software. Given the open availability of that database, we have not recreated the list of nutrients in this publication.

Feeds could be selected from the library and used within a ration. Ingredients specific to a farm or customer should be editable without affecting values for other farms. Thus, adding a farm-specific feed to a ration should cause a new feed entry to be created as a copy of an existing feed with the farm or customer name included as an identifier. Once a farm-specific feed is created, individual nutrient values would be editable. To ensure that the original nutrient values are preserved, a copy of the source feed library should be maintained and editing of those ingredients prohibited. The user should be provided with tools to create a regional or local feed library from the source library that could be used across customers and be edited to reflect local ingredient values. If new ingredients are added to the library, those entries should only contain nutrient values for the species of interest. Nutrient fields for other species should not be entered by the user or copied as they will not be presented with a list of those nutrients for editing or addition. Provided the installation on a computer is tied to a single species, deletion of an ingredient could trigger full removal of the ingredient from the library. However, if the user is able to use a single install for multiple species, one cannot allow deletion of an ingredient from the library without checking to see if that ingredient is used in rations across all species. If it is used in another ration for a different species, then it cannot be deleted for both species. However, the nutrient fields for the selected species could be deleted and it could then be marked as not available for that species.

Ration Specification and Library

Rations are generally specified for individual groups of animals on the farm and include a list of ingredients, and the amount of each ingredient to be used. An example of the form is provided in Figure 3. If an optimizer is used, the ration specification screen becomes quite a bit more complicated. One must also specify constraints for ingredients and nutrients in the ration. For example, the dry matter or as-fed intake is normally predicted from the model, and that predicted value is assumed to represent the maximum feed intake, but the user could specify a possible range for intake based on experience or specific knowledge of the farm. Requirements for key nutrients are also generally specified by the animal model and used as minimums, but the user may want to also specify maximum and adjust the minimum based on experience. In ruminants, it is often useful to be able to place bounds on a portion of the feeds to achieve a desired nutritional outcome. The primary example is a constraint on forages. This constraint limits the proportion of the diet made up of forage-type feeds. For lactating dairy cow rations, one would

generally want to constrain this to between 40 and 60% of ration dry matter. If multiple forages are in the ration, this is more difficult to achieve in the absence of the ability to place a constraint on the group. Groups are already intrinsic to some of the NRC feed libraries in the form of ingredient class or type. If group constraints are provided, the solution could make use of the class or type grouping to achieve, for example, a 50:50 mix of forage and grain using an infinite number of combinations of alfalfa hay and corn silage. But given that the farm may often have a limited supply of each, i.e. a fixed ratio of the 2, one may also want to place a constraint on the proportion of the group represented by each forage. If the producer has a 2:1 ratio of corn silage: alfalfa hay stored on farm and does not desire to buy additional feeds, then one may want to solve for a ration with between 40 and 60% forage with the corn silage to alfalfa ratio constrained to 2:1.

In summary, the ration specification screen requires entry grids to contain a list of ingredients used and their specified inclusion rates if the ration is not to be optimized, or starting inclusion rates if it is to be optimized. If an optimizer is used, then there needs to be an extension of the ration ingredient grid to include min and max values for each ingredient and a selection of the basis (as fed or dry matter) for the constraint. An additional grid of constraints should be provided for the nutrients, and it would be useful to allow definition of groups and provision for placing constraints on the group so that the user can tailor the program to their needs.

Model

The model represents the system of equations used to predict nutrient supply to the animal and either animal responses to nutrient supply or animal requirements for each nutrient. In the latter, case, the system calculates how much of each nutrient is required, usually in absorbed units, and how much is supplied by the specified diet. These values are compared to determine whether the diet meets the requirements. These equations are mostly independent of the rest of the program with the exception that they are designed to make use of a defined set of inputs and to generate a defined and static set of outputs. Symbolically, the models are essentially the same across species, and thus they can be treated as a separate piece of code that can be referenced from the interface passing inputs to it, and retrieving outputs. Thus a change of species simply requires a change of code reference to link to the appropriate model, and a change to the list of cross-references to feed data to the model and to retrieve outputs. If no changes to the inputs or outputs are required, i.e. the model is updated to correct an error, one could simply change the dll containing the model, thus requiring no changes to the interface code. One could also set up the code to allow selection of a model. For example, if one wanted to look at the same ration in the newest release of the NRC model vs an older version, it would be possible with this approach to provide a selection mechanism so that the model can be changed at will by the user.

Inputs and outputs can be defined using static lists of ingredient, animal, and environment data links and a static list of outputs used to generate reports and populate program fields. For example, if the dairy model requires inputs of cow size, days in milk, days pregnant, desired milk

yield and composition, dry matter intake, and the fractional proportion of DM present in 10 different dietary nutrients, those items would be listed in the specification table along with the model variable to receive the values, and the location of each input item, i.e. a database field, a screen location, or a variable in memory. This cross-reference table would be used to pass the appropriate values into the corresponding model variables and to retrieve model data and pass to the appropriate variables back to the interface and database. In this manner, the model could be changed based on the species selected and updated models could easily be implemented only requiring an update to the link for the model and the table defining the input and output relationships.

Reports

Most of the reports could have a common format, and any deviations by species could be defined in tables. Thus the reports would have a common format using a data grid approach to place the data.

Optimizer

Inclusion of an optimizer would likely have the largest impact on the scope of the program and have the largest cost of implementation. As noted above, some of the data requirements and screen layouts for the rations depend on whether an optimizer is implemented. Although it would add additional complexity to the development effort, an optimizer is particularly important with the ever increasing complexity of ration formulation problems. Requirement models for ruminants are now non-linear with respect to nutrient inputs and nutrient availability to the animal. More of the systems will become nonlinear in the future. In addition to preventing use of a linear optimizer such as linear programming, such nonlinearity makes it much more difficult to conceptualize the problem and thus more difficult to solve by hand. The addition of output constraints, e.g. maximum phosphorus or nitrogen excretion or maximum greenhouse gas production, or penalties for output, e.g. a carbon tax, will only exacerbate this challenge. Thus a robust nonlinear optimizer should be considered for inclusion in future software releases.

There are several commercially available optimizers, and one should make use of this proven code rather than attempting to develop something. The licensing fees for the code are likely to be much less than hiring a mathematician and a programmer to develop new code. Implementation of the code requires linkage between the optimizer, the program interface used to capture user inputs, and the model. Thus, if an optimizer is to be included at a later date, the planning for this should occur at software inception to support subsequent inclusion. Inputs to the optimizer include the list of ingredients, ingredient cost, and the full list of constraints. The model would be informed of the ingredient composition, and the optimizer would iteratively manipulate the feeding rates of each ingredient within the constraints, and make calls to the model to determine nutrient concentrations and nutrient balance (inputs – requirements). These

outputs would be compared to any nutrient constraints to guide and constrain ingredient inclusion rates.

As for the model, the list of linkages between the optimizer and the chosen model could be defined as a tabular cross-reference list by species. Thus the model could be swapped or updated with minimal effort, only requiring an updating of the cross-reference table.

Inclusion of Stochastic Elements in NRC Models

Current NRC models for swine, poultry, beef, and dairy are all based on deterministic principles, i.e. the model provides a single point estimate for a given set of inputs and does not attempt to represent the uncertainty in that estimate nor in the dietary inputs. Between-animal variation, variance in dietary inputs, and the uncertainty of the parameter estimates are all ignored. Nutrient requirements vary greatly between animals of a given population and each animal follows individual patterns over time (Pomar et al., 2011). For example, the lysine requirement of a lactating sow depends on the animal (e.g. genetics, age, BW, body composition, milk production), environment (e.g. temperature, humidity, pens) and feeding factors (e.g. feed allowance, intake capacity, quality). Some of these factors may be controlled and therefore similar for all animals within a group, but many of them will result in variation between the animals. Model estimates thus generally represent the mean animal in the population fed a diet with precisely known composition.

When mean deterministic recommendations are applied to populations exhibiting large between-animal variation, the requirement of a certain percentage of the population is not met (Pomar et al., 2011, Hauschild et al., 2012) and the mean performance of the group will be lower than expected. At the same time, a proportion of the animals in the population will be overfed. To maximize efficiency, one must balance the inefficiencies associated with overfeeding some animals with those arising from reduced performance for those being underfed. The optimum is not generally the population mean. Variance in feed inputs adds to the complexity of the problem. By representing both the stochastic elements of the requirements and those of the feed inputs, one can construct a feeding program that balances uncertainties and economics to optimize animal efficiency while minimizing nutrient excretion.

So far, stochastic models have only been developed for growing pigs (Schinckel et al., 2003). In traditional models the animal is described by a number of parameters determined for the individual. The basic assumption in this approach is that the true parameter values of the population being simulated are known with complete certainty. Similarly, milk production of cows or sows could be specified by a mean and standard deviation (SD), so that the actual milk production of a particular animal is drawn from a distribution having specific parameters. The assumption is that the specified mean and SD are the true values. In reality parameters can only be estimated with a certain precision and at worst they are based on expert estimates. In contrast to some deterministic nutritional herd simulation models stochastic models take into

consideration the uncertainty of the true parameter values. For a given set of population parameters, a set of animal specific parameters are generated, which reveal the range of nutrient requirements under these specific conditions. The variation in the median (or in any given percentile) nutrient requirement between the simulation runs of different population parameter expresses the uncertainty concerning the true population parameter values. In essence, the between-animal variation can be considerable, and the uncertainty concerning the different population parameter values in the model need be taken into account when applying nutritional management to a population or herd.

Therefore the interface supporting the model and the optimizer should be developed to support the possible addition of stochastic elements to all of the requirement models. This addition would also have minor impacts on reports.

Platform

The past versions of NRC software have been developed both as stand-alone compiled programs and in spreadsheet form. There are advantages and disadvantages for each. Advantages of building the code based on Excel include:

1. The model should run on all computers that run Excel, including tablets, and across operating systems
2. It may be easier to create and maintain Excel version of the model
3. Excel and the associated macro language (required to implement full operation) has become more stable over time

Disadvantages include:

1. Although 3rd party software exist for optimization and stochastic models, the user may have to purchase them separately
2. Implementation of the above generic framework in Excel may be more difficult
3. Unexpected compatibility issues may be encountered that limit the multi-platform support.
4. The user must purchase Excel

Probably the biggest concern is that the Excel overlay adds another layer of complexity to the program which may prove problematic from a maintenance standpoint. One of the authors (Hanigan) implemented a mock-up of a ration balancer in Excel to teach undergraduate students in animal nutrition the concepts of ration balancing. Each year, the macros required to automate the code required updating to address errors that arose in association with the newest release of Excel, the latest release of Windows, and differences between Microsoft and Apple platforms.

After 3 years of continuous problems, the effort was abandoned. If one contrasts that with the stability of the 2001 Dairy NRC program which performed without support for 10 years, it seems that developing a dedicated program would likely require less long term support, although the initial costs may be more. However, if one started with the existing Dairy code and modified it to achieve the above, overall costs would be reduced.

Another option to achieve the goal of a common interface across NRC programs would be to work with one of the existing ration balancing software companies to adapt their platform. This would ensure that adequate support is provided, but ownership of the final product, control and timing of code updates, licensing of the 3rd party software such as the optimizer, and the ability to develop sub versions to test research concepts or use for extension work on integrated projects could be problematic. But this option should be explored before committing to a new coding effort.

Other Considerations

While the interface for the program would generally be worked on by individuals that are skilled in computer programming, it would be useful if the model could be worked on by the broader research community. This would foster model development as it would allow testing of alternative representations. All nutrition researchers and advanced graduate students are skilled in the use of interpreter based program through their use of statistical packages such as SAS and R. If the model code could be constructed using an interpreted language and subsequently compiled to create the dll for use in the software, that would accomplish the goal of fostering a larger and more engaged research community as it obviates the need of advanced computer programming skills.

Use of the program and associated model may be expanded greatly by deployment of a web based version of the software. This would allow easy access by more casual users such as students, producers, and international individuals. The latter would help extend the impact of the effort and promote better nutrition throughout the world thus supporting the goal of improved food sufficiency. A web based version could also be useful to characterize the use of the program and the range of diets used in production systems. From a research and outreach standpoint, such information could have great value. In support of this aspect, a tablet based application to gather the data required for ration balancing may encourage its use. One could enter all the needed information on the tablet and submit the problem for a solution on the web server. This would reduce the computing requirement at the user end allowing the use of more complicated and larger non-linear models. The web server could also be used to distribute database updates for local copies of the software and updates to the software itself.

Summary

A multi-species ration-balancing software package will require an investment in coding time initially, but should decrease expense and resources required to release and support subsequent NRC models, particularly if the code is maintained as open source and community members are willing to donate time to the project. This paper suggests a framework for a multi-species platform which would use species input data to auto-configure at install or runtime. Cross-

reference tables would be used to update input fields specifying animals, environment, feeds, etc. and calculate nutrient requirements based on species selected. Suggestions for incorporating a non-linear ration balancer have been included to ensure that the software can be used as a teaching tool, to demonstrate full application of new concepts, and to aid in extension and research efforts. Ultimately, stochastic elements will be added to the models and thus these elements should be considered for future addition when planning any current efforts.

References

A.Lemme, V. Ravindran, and W.L. Bryden. 2004. Ileal digestibility of animo acids in feed ingredients for broilers. World's Poult. Sci. J. 60:423-438.

H.H. Stein, B. Sève, M.F. Fuller, P.J. Moughan, and C.F.M. de Lange. 2007. Invited review: Amino acid bioavailability and digestibility in pig feed ingredients: Terminology and application. J. Anim. Sci. 85:172-180.

Hauschild, L., P. A. Lovatto, J. Pomar, and C. Pomar. 2012. Development of sustainable precision farming systems for swine: Estimating real-time individual amino acid requirements in growing-finishing pigs. J. Anim. Sci. 90(7):2255-2263.

Pomar, C., L. Hauschild, G. H. Zhang, J. Pomar, and P. A. Lovatto. 2011. Precision feeding can significantly reduce feeding cost and nutrient excretion in growing animals. Pages 327-334 in Modelling nutrient digestion and utilisation in farm animals. D. Sauvant, J. Van Milgen, P. Faverdin, and N. Friggens, ed. Wageningen Academic Publishers.

Schinckel, A. P., N. Li, P. V. Preckel, M. E. Einstein, and D. Miller. 2003. Development of a Stochastic Pig Compositional Growth Model. The Professional Animal Scientist 19(3):255-260.

Figure 1. An example of the Animal Specification Form. The Label and Units column values would be populated based on the species selected.

Animal Specifications: Dairy, Lactation

Label	Value	Units
Breed	Holstein	unitless
RFI[1]	-1.1	lb/d
Age	28	month
BW	1550	lb[2]
Mature BW	1650	lb[2]
BCS	3.0	unitless
Days Pregnant	0	Day
Days in Milk	130	Day
Lactation Number	1	
Age at 1st Calving	24	Month
Milk Production	95	lb[2]
Milk Lactose	4.8	%
Milk Protein (True)	3.0	%
Milk Fat	3.5	%

[1]Line is used to categorically define swine and poultry today. Future consideration of the genomic profile of the animal, such as residual feed intake (RFI) will be needed. This information will be continuous in nature and likely multifactorial. Thus multiple lines of animal genetic values may be required.

[2]The setting for units on the Program Settings page will dictate whether mass are displayed with units of lb or kg. However, all data should be stored in metric form and converted for display if required.

Figure 2. An example of the Environment Specification Form. The Label and Units column values would be populated based on the species selected.

Environment Specifications: Dairy, Lactation

Label	Value	Units[1]
Daytime High	88	F
Nighttime Low	62	F
Previous High	89	F
Dew Point	60	F
Wind Speed	10	mph
Coat Depth	25	inches
Coat Condition	25[2]	% wet
Distance Travelled	4000	Ft
Average Steepness	0	% Incline

[1]The setting for units on the Program Settings page will dictate whether temperature is collected with units of F or C, wind speed is collected in mph or kph, coat depth is collected in inches or millimeters, and distance travelled is collected in feet or meters.

[2]Coat condition likely should be a choice from a drop-down with options exemplified by: dry, 25% wet, 50% wet, 75% wet, 100% wet. The selection would populate both the Value and Units columns.

Figure 3. An example of the Ration Specification Form. The list of Nutrients and the associated Labels would be populated based on the species selected. Units will display based on the metric/imperial DM/AF choices in program settings.

Ration Specifications: Dairy, Lactation

				Model Specified[1]		User Specified		
				Min	Max	Min	Max	Units
		Dry Matter Intake			23.1			
Cost	Ingredient	Inclusion[2]	Units[3]					
$75	Corn Silage	6.9	kg DM			5	8	kg DM
$350	Alfalfa Hay	3.1	kg DM			1	4	kg DM
$300	Corn, Fine Grind	5.23	kg DM				7.5	kg DM
$450	Soybean Meal	2.105	kg DM					
$225	Wheat Midds	2.891	kg DM					
$380	Distillers Grains	1.00	kg DM					
$120	Limestone	1.1	% of DM					
$6,500	Vit/Min	0.990	% of DM					
Group	Group Item	Value	Units					
Forage	Corn Silage	67	% of For			60	70	% of For
Forage	Alfalfa Hay	33	% of For			30	40	% of For
	Nutrient	Value	Units					
	Forage (Group)	45	% of DM	40		45		% of DM
	NEL	2.6	Mcal/kg	60		62		Mcal/d
	Metab. Protein	2.35	kg/d	2.35				kg/d
	ADF	19	% DM	19				% DM
	Vit A	70	kIU/d	70				kIU/d
	Vit E	165	IU/d	165		200		IU/d
	Ca	70	g/d	70				g/d
	Ca:P	3	g/g					
	Met:MP	2.1	% of MP			30		g/d
	Output	Value	Units					
	Manure P	30	g/d				30	g/d
	Manure N	466	g/d					

[1]Specified by the animal requirement model. Entries in the User Specified field over-ride model specified values.

[2]The ingredient inclusion rate would be specified in the absence of optimization or represent a starting value if an optimizer was used.

[3]Units will reflect the users choice of basis and of system (metric or imperial for the program as a default, but a drop down list should be provided allowing selection of expression as lb/d or kg/d and percent of the ration.

Table 1. Program settings data.

General	Calculation Units	Basis for Calculations	Ingredient Groups	Fields to Pass to the Requirement Model	Fields to Display on Reports
Ration Name	Imperial	DM	Forage	Nutrient a	Animal Settings A
Customer Name	Metric	As Fed	Byproducts	Nutrient b	Animal Settings B
Herd Size				Nutrient c	…
Location			Vitamin/Mineral	…	Animal Settings Z
Ration Comments			Supplement	Nutrient z	Environmental Settings A
				Feed Intake	Environmental Settings B
				Forages	…
				Ca:P	Environmental Settings Z
				Lys:Met	Nutrient a
					Nutrient b
					…
					Nutrient z

Table 2. Non-nutrient program display items.

Species Screen	Beef	Dairy	Poultry	Swine	Small Ruminant	Horse
Ration Specification	DMI Predicted ME Allow ADG ME Balance peNDF Balance DMI/Maint DMI Est. Ruminal pH Bact N Bal Peptide Bal Urea Cost Bacteria MP UIP MP Cost/d N Balance MP Allow ADG EAA Allw ADG Most Limit AA RDP/RUP Ruminal N bal.	DMI Predicted MP/MP Req NE Allow Milk MP Allow Milk AA Alow Milk NE Allow Gain MP Allow Gain AA Allow Gain NEL Balance MP Balance RDP Balance BCS Change Rate				

Table 3. Animal setting categories for several production species.

Species	Physiological State	Breed Line	Age	Sex	Weight	Body Condition Score	Mature Weight	Birth Weight	Daily Gain	Gain Composition	Day in Milk	Milk	Milk Compos
Beef	Calf	B	X	X	X	X			X				
	Grow/Finish	B	X	X	X	X			X	X			
	Gestating	B	X	X	X	X							
	Lactating	B	X	X	X	X		X	X		X	X	X
Dairy	Calf	B	X	X	X	X			X				
	Growing	B	X	X	X	X	X		X	X			
	Lactating	B	X	X	X	X	X	X	X		X	X	X
Poultry	Growing	B, L	X	X	X				X	X			
	Laying	B, L	X	X	X								
Swine	Nursery	B, L	X	X	X				X				
	Grow/Finish	B, L	X	X	X				X	X			
	Lactating	B, L	X	X	X				X		X	X	X
Small Ruminant	Growing	B	X	X	X				X	X			
	Lactating	B	X	X	X				X		X	X	X
Horse	Growing	B	X	X	X				X				
	Lactating	B	X	X	X				X		X	X	X
	Working	B	X	X	X	X			X				

Table 4. Environmental settings categories for several production species.

Species	Physiological State	Housing Type	Heat Mitigation System	Floor Space	Current Daytime High Temp	Previous Daytime High Temp	Current Nighttime Low Temp	Dew Point	Wind Speed	Coat Depth	Coat/Skin Condition
Beef	Calf										
	Grow/Finish				X	X	X		X	X	X
	Gestating										
	Lactating										
Dairy	Calf										
	Growing				X	X	X		X	X	X
	Lactating				X						X
Poultry	Growing										
	Laying										
Swine	Nursery										
	Grow/Finish	X		X							
	Lactating										
Small Ruminant	Growing										
	Lactating										
Horse	Growing										
	Lactating										
	Working										