DISSERTATION

HEIFER PREGNANCY GENETIC PREDICTION AND SIMULATION MODELING TECHNIQUES

Submitted by
Carlton R. Comstock
Department of Animal Sciences

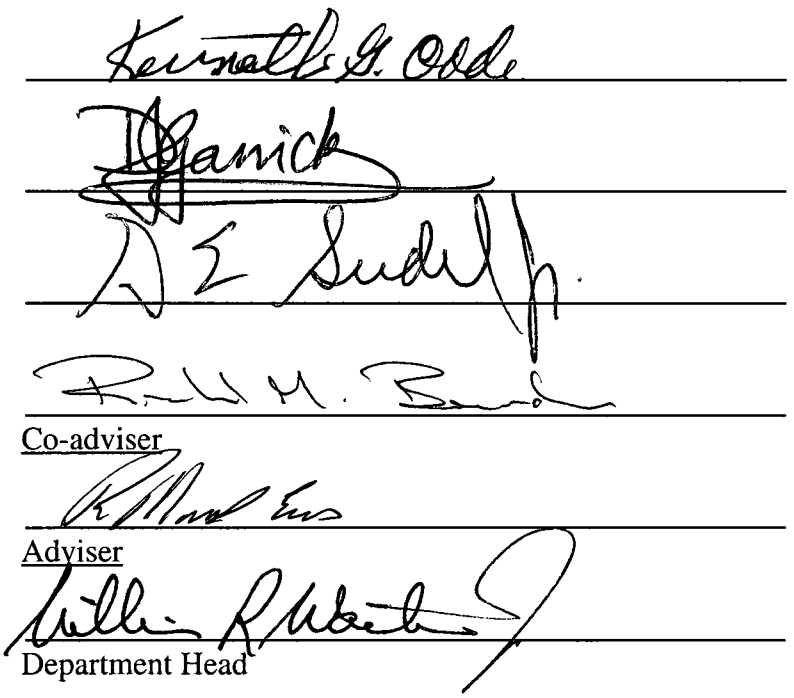
In partial fulfillment of the requirements
for the Degree of Doctor of Philosophy
Colorado State University
Fort Collins, Colorado
Spring, 2009

COLORADO STATE UNIVERSITY

June 12, 2003

WE HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER OUR SUPERVISION BY CARLTON R. COMSTOCK ENTITLED HEIFER PREGNANCY GENETIC PREDICTION AND SIMULATION MODELING TECHNIQUES BE ACCEPTED AS FULFILLING IN PART REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY.

Committee on Graduate Work



Co-adviser

Adviser

Department Head

ABSTRACT OF DISSERTATION

HEIFER PREGNANCY GENETIC PREDICTION AND

SIMULATION MODELING TECHNIQUES

The Colorado Beef Cattle Production Model (CBCPM) was rewritten with an object oriented design and used to simulate heifer pregnancy data with varying levels of age at puberty (AAP), probability of conception (PCON), and length of breeding season. Five percent of the heifers were simulated infertile due to non-genetic causes. Simulated data were used to estimate heritability of heifer pregnancy and to obtain EBV using threshold models. The EBV were tested for accuracy of prediction of the simulated genetic fertility traits.

Object oriented methods used illustrated the ability of these techniques and tools, such as Unified Modeling Language, at organizing complex processes in ways to reduce errors and code maintenance effort, and to facilitate collaboration among developers. Adoption of these tools will be critical to the advancement of systems models.

Heritability of heifer pregnancy from 20,000 heifers with very early puberty in the first 25 d of the breeding season was .139, .107, and .143 for mean PCON of 60, 70, and 80 %, respectively, close to the .10 input heritability of PCON. The higher estimates may be due to a few heifers having two opportunities to breed. With very late puberty and 80 % mean PCON in a 25 d breeding season the heritability estimate of AAP was .337, lower than the

simulated .40 heritability. The estimate was lower because there was not 100 % conception, some heifers were infertile, and puberty was observed as a threshold trait.

Heritability estimates of heifer pregnancy generally decreased as breeding season length increased, likely due to an increasing percent of open infertile heifers. The ability of the variance component estimation software to converge on an estimate decreased as the number of open, fertile heifers decreased. The most difficulty was with high PCON, early puberty, and long breeding seasons; only one out of 100 estimates converged at 340 d AAP, 80 % PCON, and 120 d breeding season.

Calculated accuracy for heifer pregnancy EBV for the sires of the heifers using prediction error variances from a linear model, with the binary pregnancy observations treated as continuous data, overestimated accuracy of the EBV with respect to the simulated traits in all cases. Calculated accuracy was insensitive to changes in frequency of heifer pregnancy observations.

Accuracy calculated as the simple correlation of the EBV with each simulated fertility trait for the sires of the heifers was highest in most cases at the shortest breeding season. The correlation with AAP was essentially zero for early puberty, and strongest (-.775) at late puberty. As breeding season length increased the AAP correlation declined toward zero. The accuracy for PCON was less sensitive to changes in AAP and breeding season length, ranging from .146 to .753; the strongest correlations were with early puberty and low PCON.

Carlton R. Comstock
Department of Animal Sciences
Colorado State University
Fort Collins, CO 80523
Spring, 2009

ACKNOWLEDGMENTS

I likely would not have pursued this doctorate without the encouragement and the job given to me by Bruce Golden. The training I received in data analysis and analytical thinking have been invaluable; in addition, I was literally shown the world. My heart-felt thanks for all the years and opportunities.

I greatly enjoyed studying under Rick Bourdon and appreciate the lending of an ear on topics ranging from animal breeding to raising kids. My thanks to Mark Enns for stepping up and seeing me out the door as advisor and for the years of friendship, and thanks also to Ken Odde and George Seidel for their help as committee members on this degree.

Particular thanks to Warren Snelling and Jim Nagel for their advice and patience with my questions on this dissertation. Given my prolonged stay at CSU there are a large number of graduate students who I had the pleasure of getting to know. Special thanks to Lauren Hyde, Gary Mathiews, Patrick Doyle, John Evans, and Larry Kuehn for all we experienced together and for the depth of your friendship.

I owe the largest debt to my family for enduring these long years. The tired refrain of "Dad has to work on his dissertation" will never darken our doorstep again. My most sincere thanks, Laurie, for sticking with me and not giving up hope I would some day finish.

TABLE OF CONTENTS

Chapter		Page
1	INTRODUCTION	1
2	DESIGN OF REPRODUCTION SIMULATION MODEL	5
	Introduction	5
	Literature review	6
	The TAMU model and its descendants	6
	The Colorado Beef Cattle Production Model	10
	Mechanistic genetics with potentials	11
	Modeling probability potentials	13
	Simulation of fertility in CBCPM	15
	Other systems models	17
	Kentucky BEEF model	17
	Johnson and Notter genetics of reproduction model	17
	Nebraska reproduction management model	18
	North Carolina deterministic systems model	19
	Taylor and Naazie efficiency models	20
	Hirooka breeding objective model	21
	Montana beef systems model	23
	Rao sheep reproduction genetics model	24
	Non-systems models of reproduction	24
	Model deficiencies	25
	Materials and Methods	26
	Non-reproduction changes to CBCPM	28
	Multiple pastures and locations	28
	Pedigree	28
	Simulation of continuous traits	29
	Simulation of categorical traits	33
	Reproduction sub-module organization	36
	Modeling puberty	36
	Hormonal changes near puberty	36
	Nonpubertal estrus	38
	Genetic effects on puberty	39
	Age at puberty	39
	Weight at puberty	40
	Environmental effects on puberty	41
	Season of birth	41
	Social cues	42

Puberty simulation 43
Modeling conception 45
Threshold trait phenotype simulation 45
Estrous cycle simulation 46
Heifer conception simulation 47
Pubertal estrous 48
Permanent infertility 49
Nutritional anestrous 49
Cow conception simulation 50
Dystocia 51
Modeling gestation length 51
Modeling postpartum infertility 51
Modeling male effects on conception 55
Validation 56
Results and Discussion 57
Verification of simulation of genetics 57
Discussion 63
Puberty 64
Rebreeding 65
Summary 67

3 RATIONALE AND DESIGN FOR AN OBJECT ORIENTED SIMULATION MODEL 68
Introduction 68
Early systems modeling tools 69
Hardware, operating systems, and networks 69
Software development tools 70
Improvement of development tools 72
Current systems modeling tools 72
Hardware, operating systems, and networks 72
Software development tools 74
Dynamic memory allocation and data structures 75
State of the art 78
The need for change 78
Advantages of object oriented design 80
Why programs are difficult and expensive to develop 81
Programming tools to reduce development and maintenance time . . 82
Object oriented design 83
OO: What are objects? 84
OO: Classes 84
OO: Encapsulation 85
J-CBCPM's Object Oriented Design 87
Unified Modeling Language 88

The design process 88
J-CBCPM's implementation 90
UML: Class diagrams 90
J-CBCPM: Herds, pastures, and ranch 91
OO: Inheritance 92
UML: sequence diagrams 96
J-CBCPM: Driver 99
OO: Java interfaces 100
J-CBCPM: Management groups 101
OO: Generalization 101
J-CBCPM: Fertility 104
OO: Polymorphism 104
UML: State diagrams 104
OO: Polymorphism 108
OO: Java interfaces revisited 115
OO: Java packages 117
OO: Dynamic binding 118
OO: Refactoring 118
Summary 120

4 ANALYSIS OF SIMULATED HEIFER PREGNANCY DATA 121
Introduction 121
Literature review 123
Threshold traits theory 123
Linear models 123
Non-linear models 125
Variance component estimation 126
Traits used to select on puberty 127
Scrotal circumference 127
Reproductive tract score 128
Heifer conception 128
Materials and Methods 132
Overview 132
Simulation model 132
Non-reproduction parameters 133
Reproduction parameters 137
Data generation 140
Analysis of simulated data 141
Results and Discussion 144
Data generated 144
Analysis of simulated data 149
Variance component estimation 149
Correlation of breeding values and the EBV accuracy 157

Summary 165

LITERATURE CITED 167

LIST OF TABLES

Table		Page
2.1	Traits simulated as potentials in the Java CBCPM and their classification with respect to simulation method and use in simulation of fertility.	30
2.2	Effect of calf presence and milk removal from the cow on the cow's postpartum interval length (Lamb et al., 1999).	54
2.3	Input and observed means and variances for the components of AAP and PCON from 40,000 simulated heifers.	58
2.4	Input and observed means and variances for the components of AAP and PCON from 1,000 simulated sires.	59
2.5	Observed simple correlations among simulated components of AAP and PCON from 40,000 heifers (above the diagonal) and 1,000 sires (below the diagonal).	59
2.6	Cumulative pregnancy rate of heifers that reached puberty at least two estrous cycles prior to the start of the breeding season, with simulated 60 % mean conception rate.	62
3.1	Fortran CBCPM driver subroutine steps (Bourdon, 1992).	100
4.1	Data description and heritability estimates of pregnancy for two Hereford populations obtained with linear and nonlinear models.	130
4.2	Genetic parameters of simulation model	138
4.3	Percent of heifers pregnant at different levels of simulated age at puberty (AAP) and probability of conception (PCON) by breeding season length.	146
4.4	Number and percent of heifers, within each of the nine simulated data sets of 20,000 heifers, that were simulated to be sterile.	149
4.5	Number of heifers, within each of the nine simulated data sets of 20,000 heifers, with a non-zero inbreeding coefficient (F) and average heifer inbreeding coefficient including non-inbred animals.	150

4.6 Heritability estimates of heifer pregnancy using 20,000 heifer pregnancy observations from data with simulated mean 340 d AAP at three levels of PCON. 152

4.7 Heritability estimates of heifer pregnancy using 20,000 heifer pregnancy observations from data with simulated mean 390 d AAP at three levels of PCON. 153

4.8 Heritability estimates of heifer pregnancy using 20,000 heifer pregnancy observations from data with simulated mean 440 d AAP at three levels of PCON. 154

LIST OF FIGURES

Figure		Page
2.1	Distribution of simulated probability of conception (PCON) phenotypic potentials (BV + Et + Ep) output at the end of a simulation run.	61
2.2	Distribution of simulated probability of conception (PCON) phenotypic potentials minus temporary environment effect (BV + Ep) output at the end of a simulation run.	61
3.1	Unified modeling language class diagrams of the relationship between herd and pasture in the Fortran and Java models, respectively.	91
3.2	Unified modeling language class diagram of how cattle on a simulated ranch are grouped in herds and located in pastures.	93
3.3	Supporting classes added to the J-CBCPM design for input parameters, simulation time, and pedigree management.	96
3.4	Unified modeling language sequence diagram overview of J-CBCPM.	98
3.5	Unified modeling language class diagram to illustrate the use of generalization in the J-CBCPM design	103
3.6	Cattle management characteristics managed in a separate class from biological characteristics.	105
3.7	Unified modeling language state diagram of J-CBCPM reproduction states.	107
3.8	Alternative state diagram for reproduction with super states for anestrous and estrous.	109
3.9	State diagram of division of reproduction states into four classes, Prepuberty, OpenHeifer, PregnantCow, and OpenCow, as indicated by the bold horizontal lines.	110
3.10	Class diagram of reproduction subclasses to illustrate polymorphism.	111

3.11 Unified modeling language class diagram of relationships between Bovine class and the reproduction and growth models. 116

4.1 Cumulative number of heifers reaching puberty in the 340 d AAP and 60 % PCON data, and cumulative number of heifers conceiving within the breeding season in the 60, 70, and 80 % PCON data sets with 340 d AAP 147

4.2 Cumulative number of heifers reaching puberty in the 390 d AAP and 60 % PCON data, and cumulative number of heifers conceiving within the breeding season in the 60, 70, and 80 % PCON data sets with 390 d AAP 147

4.3 Cumulative number of heifers reaching puberty in the 440 d AAP and 60 % PCON data, and cumulative number of heifers conceiving within the breeding season in the 60, 70, and 80 % PCON data sets with 440 d AAP 148

4.4 Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 340 d AAP and 60 % PCON 158

4.5 Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 340 d AAP and 70 % PCON 158

4.6 Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 340 d AAP and 80 % PCON 159

4.7 Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 390 d AAP and 60 % PCON 159

4.8 Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 390 d AAP and 70 % PCON 160

4.9 Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 390 d AAP and 80 % PCON 160

4.10 Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 440 d AAP and 60 % PCON 161

4.11 Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 440 d AAP and 70 % PCON 161

4.12 Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 440 d AAP and 80 % PCON 162

Chapter 1

Introduction

Beef cattle production depends on the ability of animals to reproduce. As the first step in the production process, reproduction rate sets the limits for production and for further management decisions. This dependency has lead researchers (Wilham, 1973; Melton, 1995) to conclude that within typical production systems, the relative economic importance of reproduction is several times greater than that of production.

Current beef cattle production practices require a heifer to first calve at 2 y of age and to calve at the same time each subsequent year. This requires females to breed within a limited amount of time at a fixed time of year. They are able to meet these requirements under ideal conditions, but ideal conditions for reproduction are typically not the most profitable. Ideal conditions consume management and feed resources, both of which come at a cost. In addition to low feed costs, the producer has other objectives which may conflict with ideal conditions for reproduction, such as maximizing sale weight. The ability to genetically improve reproductive performance potentially allows the producer to lower costs while maintaining the same level of output.

Until recently, few genetic predictions for reproductive traits have been available to producers. Lacking a genetic prediction, they have primarily used culling of open females as a way to control genetic potential for reproduction. However, selection on phenotype can have very low accuracy, and pregnancy is sex-limited, making accuracy of bull selection even lower. An exception is scrotal circumference of yearling bulls, which has been shown to be

genetically correlated to age at puberty of heifers and moderately heritable (Martin et al., 1992; Morris et al., 2000).

In an effort to increase accuracy of selection, several female traits have been studied as indicators of fertility (e.g. age at puberty, days to calving). In general, these traits have been considered to be lowly heritable. This is in part because reproduction is influenced by many additional management and environmental factors, and because reproductive potential is expressed as a binary observation (pregnant or not pregnant). Low heritability estimates also result from traits that are analyzed as a composite of several underlying factors and because linear statistical models have been used for traits expressed categorically. In other cases, the required data are unavailable (e.g., a list of all heifers exposed to breeding). Recently more detailed record keeping by breed associations and development of threshold models have allowed development of genetic predictions for traits such as Stayability and Heifer Pregnancy.

In addition to predicting genetic potential for reproduction, cattle producers must be aware of the effects of management decisions on reproduction. These effects are difficult to observe and quantify, especially their long-term implications on breeding decisions. One way to help researchers and producers determine optimal genetic and management decisions is to develop computer simulation models of beef production. A comprehensive systems model is ideal because there are so many factors that interact with reproduction.

Several models of beef cattle production systems exist. However, current models have limited abilities to model beef cattle reproduction, and models with greater detail on reproduction are generally not systems models. I found only one systems model using a 21-d

estrous cycle (Azzam et. al, 1990), as opposed to daily probability of conception models, in the literature. Most models work empirically, and many use an average animal instead of individual animals with variability. The ability to model individual animal variation is helpful and provides a better picture of the risk involved in genetic and management decisions.

In addition, current models of beef cattle systems are primitive with respect to the software engineering techniques used to implement them. All beef cattle simulation models I have found in the literature were written in Fortran. As best I can determine, only one of the models (Tess and Kolstad, 2000a) used data types more advanced than the intrinsic Fortran 77 data types. None of the models was implemented using object oriented techniques or languages.

The current state of beef cattle simulation modeling is not a reflection of available computer power (e.g. processing speed or storage space), and it is likely not limited by knowledge of animal science. Rather, I believe it has been limited by the ability of the individuals developing models to manage systems of this complexity. To date the complexity has been handled by having a few people dedicated to developing and maintaining a particular model, but with the result that the model becomes so complex that other researchers can not easily make modifications for their own purposes. Instead, they start from scratch and develop a model they understand and trust based on their familiarity with it. This limits the availability of modeling as a research tool for those without the resources to immerse themselves in the code. It also limits the complexity of the models because the developers seldom have expertise at all levels.

Given the potential of a systems model to be used as a research tool for improving

reproduction, my first objective was to develop a mechanistic, individual based model of female beef cattle reproduction. The model was designed to be implemented within a comprehensive beef cattle production simulation model. My intent was to develop this model in a manner that would illustrate the advantages of modern programming techniques and languages.

My second objective was to determine the effect of varying the levels of inherent fertility and varying management (by altering the length of the breeding season) on genetic predictions for heifer pregnancy. In particular, I wished to 1) determine if important genotype by environment interactions exist for genetic predictions of heifer pregnancy with respect to inherent fertility, simulated as varying levels of age at puberty and probability of conception, and 2) characterize the accuracy of heifer pregnancy genetic predictions with respect to the frequency of pregnancies as controlled by breeding season length and influenced by inherent fertility.

Both Chapters 2 and 3 discuss the development of the reproduction model, but from different perspectives. Chapter 2 details the biological components considered for the model, and is intended for users of the model. Chapter 3 explains the use of object oriented design and programming to implement the model, and is intended for use by model developers. Chapter 4 follows the two development chapters with application of the model to generating and analyzing data for study of genetic predictions of heifer pregnancy. I hope Chapter 3 will be helpful for modelers not necessarily interested in reproduction. Chapter 2 is more useful for interpreting the results of Chapter 4.

Chapter 2

Design of The Reproduction Simulation Model

Introduction

Existing models of beef cattle production span the spectrum of systems complexity. They range from models of one component of a single animal's physiology to models that include hundreds of individual animals in a herd over many years. While it is possible to gain insight from a model of a specific phase of beef cattle production or a specific biological system, it is more revealing to model the whole production system. By modeling the animal biology, management, and other environmental factors, it is possible to observe interactions that might not be expressed when modeling just a few components of the system.

Animal scientists have been using computers to model beef cattle production systems since at least the mid 1970's, and as computer abilities have increased, so has the complexity of the models. Beef cattle modeling has reflected limitations in computer capability; tracking individuals in large herds across multiple years consumes computer memory and requires high speed processors.

My objective was to develop a reproduction module to be used within a comprehensive beef cattle systems model. This module was intended to be as mechanistic as possible, with individual based stochastic and genetic components, and to have management flexibility.

Literature Review

The TAMU Model and its Descendants. One of the more widely used systems model was developed at Texas A&M University (TAMU), and is known as the TAMU or Texas model. Described by Sanders and Cartwright (1979a,b), the model was designed to allow components, such as growth, body condition, and fertility, to interact and to use equations that were biologically interpretable. Like nearly any model, the TAMU model was influenced by previous models. A major difference in design was that previous models took performance level as an input to the simulation, while the TAMU model simulated performance levels based on biological inputs in the context of environment inputs (Sanders and Cartwright, 1979a).

The TAMU model was designed to simulate classes of animals, with animals being grouped by age and with cows further separated by month of lactation and (or) pregnancy. The number of animals in each class was determined by user input and by setting management rules which affected each class size dynamically as the simulation ran. For example, a management rule to cull all cows at 10 y of age could cause an increase in the number of younger breeding animals necessary to maintain an input target breeding herd size. Simulating classes of animals as opposed to individual animals was necessary due to the limited amount of computing power available at the time.

The TAMU model worked on a 30 d time step, as opposed to simulating each day; this was another method of reducing computing requirements. At each time step a class was modeled as a single average animal, and then the results were multiplied by the number of animals in the class to determine the performance of the whole class. Within a time step a

simulated animal consumed nutrients and expended them for growth as lean and fat gain (loss) resulting in weight gain or loss, and for milk production if lactating.

The model had several genetic components, including size, milk production, maturation rate, and lactational persistency. Sanders and Cartwright (1979a,b) used input variables they called genetic potentials to parameterize biological types within the model. They defined genetic potential as the maximum performance of an animal in an unlimiting environment. In other words, given ideal conditions the animal had the potential to produce at that level. However, ideal conditions do not occur, and this was modeled by modifying genetic potentials with a number of correction factors.

The model was not designed to study breeding and selection within and between genotypes; all classes of animals within a given simulation run had the same genotype. Conception of calves did not include mid-parent mean (parents were all the same), Mendelian sampling, inbreeding depression, or heterosis. Sanders and Cartwright (1979a) clearly intended the potentials in their model to reflect the animals' genetics. However, since the inheritance of individual animal's potentials were not mechanistically simulated it was not necessary to go beyond their level of description. Since then, Bourdon (unpublished) has expanded and formalized the concept of genetic potentials. To avoid confusion with Bourdon's genetic potentials I am going to refer to the TAMU genetic potentials simply as potentials.

Sanders and Cartwright (1979a) mentioned how fertility of heifers and cows was in part a function of their reproduction potential. However, their description of how reproduction was modeled does not specifically identify any parameter as being a

reproduction potential. Smith (1979) noted that any of the reproduction parameters could be used to specify breed differences, which might be interpreted as being a potential. Referring back to the description of potentials being a maximum level of performance in an unlimited environment, it is likely Sanders and Cartwright (1979a,b) intended the genetic components of fertility to be the maximum probability of beginning to cycle and the maximum probability of conception. With this interpretation, puberty was modeled as a non-genetic modification to the maximum probability of beginning to cycle.

To reach puberty in the TAMU model, a heifer class had to first reach a certain degree of maturity, determined by a heifer body weight at puberty that was input by the user. This weight was adjusted for condition, daily weight gain, and maturity. Puberty was also controlled by degree of maturity; puberty occurred when a heifer's weight adjusted for body condition was between 40% and 60% of mature weight. Heifers not yet reaching 40% of mature weight had delayed puberty, while animals reaching 60% of mature weight were assumed to have reached puberty. This range could be adjusted by changing coefficients within the model.

The fraction of animals in a class in estrous was calculated each time step for each class of females greater than 9 mo old. The fraction of animals that exhibited estrus was the sum of the fraction of animals cycling the previous time step that were still cycling and the fraction of animals that (re)started cycling that time step. Previously cycling animals could stop cycling due to factors based on condition and daily weight gain. Animals previously not cycling could start cycling due to factors based on condition, daily weight gain, maturity, and time since calving for animals that had calved within 3 mo. Smith (1979) indicated that the

TAMU model included lactation as an additional factor that could delay the start of estrous cycling.

The fraction of cycling animals in a class that conceived was a function of body condition, daily weight gain and time since calving for animals that had calved within 3 mo. Animals only conceived during the breeding season. There was no variation in gestation length and no dystocia was explicitly simulated, although the effects of dystocia were assumed to be captured in the existing coefficients. The literature contains no discussion of breeding season lengths that were not multiples of 30 d.

The TAMU model had been applied to several studies by various researchers by the time it was described in the 1979 papers. Notter modified the TAMU model (Notter, 1977), and used the model to study effects of different levels of milk production on biological and economic efficiency (Notter et al., 1979). They found that by varying milk production potentials and cow size the model generated interactions with fertility, even though the model was not explicitly programmed to do that. The ability of interactions that are not explicitly programmed to arise is a strength of mechanistic systems models, and is one of the main reasons for constructing this type of model.

A series of papers (Kahn and Speeding, 1983, 1984; Kahn and Lehrer, 1984) describe a model based on the TAMU model, developed by Kahn (1982). The TAMU model was modified to get more realistic results while studying small herds. Stochastic events were introduced for conception, mortality, and calf sex. This model simulated individual animals, as opposed to the classes of animals in the original TAMU model. A variable time step was implemented, ranging from 1 to 30 d in length. Although they determined the 30 d was

suitable for their study (Kahn and Speeding, 1983), one of their arguments for using a 1 d time step was that most research is reported in terms of changes per day. Presumably they felt the units used for both parameterization and results would be more intuitive on a 1 d time step.

The additions of randomness and individual animals allowed more realistic simulation of instability in small herds by allowing risk to be modeled. They were able to model individual animals primarily because they were simulating very small herd sizes (e.g. 30 animals). At that time, larger herds would have been very time consuming with respect to computer power.

They made several changes to the TAMU model's reproduction simulation (Kahn and Lehrer, 1984) because their validation of the original TAMU equations against experimental data showed some discrepancies. To address this, they added factors to account for suckling, sterility, and advanced age. They also decreased the effect of weight loss on conception in a postpartum cow based on validation against experimental data.

The Colorado Beef Cattle Production Model. Another model, the Colorado Beef Cattle Production Model (CBCPM), also arose from modifications to the TAMU model (Bourdon, 1983; Bourdon and Brinks, 1987a,b,c). The CBCPM model included some of the modifications of Notter (1977), but not the modifications of Kahn; it initially was completely deterministic and simulated classes of animals (Bourdon and Brinks, 1987a). Several modifications were made with respect to fertility. Calving difficulty was simulated as a function of calf birth weight and dam size in heifer dams, and calf weight in cows. Herd size was limited by a linear program based on predicted forage availability. Genetic traits were

added, including potentials for starting estrous cycles, probability of conception given cycling, and age at puberty.

Bourdon's CBCPM was further modified in subsequent studies (Bourdon, 1992; Baker et al., 1992; Schafer, 1991; Enns, 1995; Doyle, 2000). The CBCPM became individual based, and individual animal variation was added, along with variable time steps, additional stochastic processes, a forage model (Baker et al., 1992), and the FLIPSIM economic model. A climate simulator was also added to provide input to both the forage model and the animal portion of CBCPM.

Mechanistic Genetics with Potentials. The concept of simulating an animal's potential performance for a trait, from the TAMU model, was extended into a more formal genetic model. The version of CBCPM used as the starting point for my study was the final version from Enns (1995), in which an animal has a potential for each of 18 traits.

Potentials were defined as the maximum level of performance attainable in a non-limiting environment (Bourdon, 1992). For a given trait in CBCPM the potential has two genetic components - a breeding potential (BP) and a non-additive potential (NAV). The BP is similar to a breeding value in that it represents the additive direct genetic effects for the trait in question, but it also contains a mean (μ_{BP}). The NAV models any heterosis effects from breed combinations and also the variance from non-additive gene combinations.

In CBCPM a trait's potential (PO) is calculated as

$$PO_{female} = BP + NAV + E_p + (E_t), \text{ and}$$

$$PO_{male} = (BP + NAV) * SexAdjustment + E_t,$$

where E_p is the permanent environment effects, E_t is the temporary environment effects, and

the SexAdjustment parameter for males multiplicatively scales the PO for differences due to sex. The components of PO are distributed as

$$BP \sim MVN(\mu_{BP}, \Sigma_{BP}),$$

$$NAV \sim MVN(\mu_{NAV}, \Sigma_{GCP}),$$

$$E_t \sim MVN(0, \Sigma_{Et}), \text{ and}$$

$$E_p \sim MVN(0, \Sigma_{Ep}),$$

where Σ_{BP} is the additive direct genetic (co)variance and Σ_{GCP} is the non-additive gene combination potential (GCP) (co)variance among the eighteen simulated genetic traits. The terms μ_{BP} and μ_{NAV} are means for the breeding potential and nonadditive potential, respectively, used to parameterize breed differences. Error (co)variance matrices for E_t and E_p are Σ_{Et} and Σ_{Ep}, respectively. These four matrices are symmetrical, and any covariance may be zero but all variances must be nonzero to allow a Cholesky decomposition to be calculated. It is important to understand that E_t and E_p do not account for all of the non genetic variation in a trait in CBCPM, and that the equation above for PO is only the parameterized input for a given trait. There are additional simulated effects in the model (e.g. age of dam) that may add variation.

Bourdon has further developed the concept of potentials in an unpublished paper entitled Genetic Potentials: Theory and Development. In it he defines genetic potentials and their relationship to conventional genetic values (e.g. breeding value) in a more general treatment but has references to CBCPM throughout. In CBCPM a potential is the maximum performance possible in a non-limiting environment, while Bourdon's genetic potential is maximum performance possible given the animal's genotype. The key difference is that a

CBCPM potential includes environmental deviations, while Bourdon's genetic potential does not. The net result is environmental deviations for a genetic potential as defined by Bourdon can not further increase performance in the trait, and therefore they require special, non-normal distributions.

Modeling Probability Potentials. Several traits in CBCPM are modeled as probabilities, including the probabilities of cycling, conception given cycling, and dystocia. Potentials for traits expressed as a probability are calculated as

$$PO = BP + (NAV + (E_p) + E_t) \times (1 - BP)$$

in an attempt to bound the potential between 0 and 1, and to make it difficult to approach either zero or one. I say "attempts" because it fails, resulting in a normal distribution with both tails extending beyond the probability parameter space. Potentials for probability traits out of the parameter space are modified by truncation, changing negative values to zero and values greater than 1.0 to 1.0. Care must be taken when parameterizing the model with a mean BP near one or the other end of the distribution as the formula above will result in simulated PO variances much lower than would be expected by summing the input variance parameters.

Data files are used to allow the user to parameterize CBCPM prior to a simulation. At run time the model reads these files and builds foundation herds and initializes the environment (pasture) before starting the daily simulation. A foundation herd is specified by describing the cows in one of the input files. Since the foundation cows can be of multiple breed types, each breed type is described as a foundation group. In other words, a foundation herd is composed of one or more foundation groups, with a foundation group for each breed

type. CBCPM allows the user to define 10 breed types in terms of mean genetic potentials. A foundation group allows the user to define either "purebred" or cross-bred foundation cows as combinations of the initial 10 breed types by specifying the foundation cow's sire and dam breed types. A foundation group also allows the user to parameterize the age distribution in terms of number of foundation cows to generate per age category. All foundation cows are assumed to be pregnant, so it is necessary to define the sire of the fetus.

Potentials for foundation animals in CBCPM are simulated from input additive direct genetic, temporary environment, and permanent environment (co)variances, and non additive genetic (co)variances. An animal's potential is the sum of these effects (with a sex adjustment for males on some traits). A foundation cow's breeding potential is the input mean breeding potential plus an appropriate random multivariate normal deviation to represent Mendelian sampling from an unknown sire and dam.

The CBCPM does not rely on equilibrium age distributions of cows (Bourdon, 1992), unlike earlier versions of TAMU (Notter, 1979). Although the user inputs an initial age distribution for the foundation cows, the age distribution of cows during a simulation run interacts with other parameters in the model, such as culling policies and fertility rates. This is an advantage of modeling individual animals, in that it allows the effects of management changes to be observed over the time it takes to reach a new equilibrium. Earlier models that used classes of average animals were more suited to studying animals in equilibrium both with respect to biological (e.g. breed composition) and management parameters of the model. Equilibrium in a herd of beef cattle is a theoretical concept useful for simplifying the world. There are many cases where changes in a simulation held at "equilibrium" would yield

different results if applied to a simulation that did not assume or force equilibrium. This is particularly important with respect to cow age distributions as it affects the composition of the product sold and the number of replacements required to maintain a target herd size. Often the time period from a change in management or biotype until equilibrium is approached can be of more interest and importance from a risk standpoint than the performance once equilibrium has been reached.

Simulation of Fertility in CBCPM. The CBCPM has potentials for three traits directly related to a female's ability to conceive. These are age at puberty, probability of beginning cycling, and probability of conception given cycling. Additional traits with potentials related to reproduction are gestation length and maternal and direct dystocia. Puberty is reached when a heifer's effective age, based on a maturity index, is greater or equal to her potential age at puberty. The maturity index is based on her actual weight relative to her individual growth curve weight at its inflection point, which is assumed to be the point she reaches puberty. Once she reaches puberty she is considered a virgin mature cow for reproductive purposes. However, she does not automatically begin cycling. The model allows an animal's actual weight to deviate from its growth curve weight based on prior nutrition, so it is possible for a heifer to reach her puberty weight prior to, at, or after the inflection point of her parameterized growth curve.

A mature cow has a potential for the probability that she will begin cycling. This potential is modified by correction factors for her body condition and daily weight gain. For cows that have previously calved it is also modified for time since calving, her current lactation status, and whether she experienced dystocia or not. To determine if she actually

begins cycling her adjusted potential is compared to a value drawn from a random uniform distribution ranging from zero to one. Only if the value is less than the value of her adjusted potential will she begin cycling. Cows that are cycling can stop cycling, even if they just began. A probability for this is calculated from their body condition and daily weight gain, and is compared to a random number obtained as above.

Conception is possible for mature, cycling cows on days within a breeding season. There is a potential for conception given cycling which is modified for body condition, daily weight gain, and time since calving. Virgin heifers' conception is also decreased; heifers are considered to be virgin until they have their first calf. There is no simulated abortion, so the probability of conception given cycling could also be called the probability of calving given cycling. In the case of artificial insemination there are adjustments to simulate the skill of the inseminator. This adjusted probability of conception is compared to a random number obtained as above.

The USDA ARS developed a model (Jenkins and Williams, 1998) for use as a decision support aid by commercial beef cattle producers . Previous systems models had been developed by researchers primarily as a research tool, and relied on the user modifying both data files and program source code to model different scenarios, requiring a high degree of understanding of both beef cattle production and computer programming. The ARS researchers started with CBCPM, added a Microsoft Windows™ graphical user interface, and modified the growth equations. This new model was named the Decision Evaluator for the Cattle Industry (DECI). The DECI model has been parameterized and used in many environments and for many genotypes. While its interface is a huge step beyond the text file

interface of CBCPM, the DECI model has serious limitations. On one hand it is still too complicated for most producers to use properly, and on the other it has been simplified to the point that it is not possible to model many types of systems from the graphical interface.

To summarize, the DECI/CBCPM models are the most comprehensive and flexible systems models of beef production at this time. The CBCPM is highly flexible and modifiable. However, the user interface for CBCPM is archaic, and the DECI interface limits access to a subset of the CBCPM abilities.

Other Systems Models.

Kentucky BEEF Model. The Kentucky BEEF model (Loewer et al., 1978) combined crops with pastured livestock while tracking economic variables. It was a deterministic model with a 1 d time step. Animals were modeled in one of twelve classes based on age, sex, and reproductive status. Interaction between grazing intensity and nutrients available to the animal was modeled. Reproduction was vaguely described; puberty required reaching a minimum age and weight, and postpartum interval was described as a time lag for lactating females between calving and being available for breeding. Female conception was a function of bull serving capacity and the female's plane of nutrition, with females losing weight having lower conception rates. No dystocia or calf survival was modeled, and there was no simulation of genetics.

Johnson and Notter Genetics of Reproduction Model. Johnson and Notter (1987a,b,c) built a model of beef production, based on a model by Willham and Thompson (1970), to study the genetics of reproduction. This model included stochastic elements and was individual animal based. It assumed a 21-d estrous cycle, but did not use a daily timestep; it

appears to have been an event-based model rather than a time step based model, but this is not clear from the paper. They simulated additive genetic, temporary environment, and permanent environment parameters for single service conception rate on an underlying, continuous scale, as opposed to the phenotypic binomial scale. The binomial phenotypes were determined by truncating the underlying distribution based on the mean frequency of single service conception desired in the simulated population. In their simulation they truncated at $-.53\sigma$ and considered the animals above that point in the distribution to have conceived, which corresponds to a 70% single service conception rate.

They also simulated a postpartum interval with additive genetic and permanent environment effects with normal distributions, but temporary environment with a gamma distribution to right-skew the phenotypic distribution. This raises a question of whether this skewed distribution would be necessary in a systems model where other simulated biological components might interact with the postpartum interval to produced a skewed phenotypic distribution. In addition, they simulated randomly infertile cows due to non-genetic causes by selecting the lowest 5% of the permanent environmental distribution to be barren.

Nebraska Reproduction Management Model. Azzam et al. (1990) modeled reproduction with the objectives of evaluating economic efficiency of different lengths and times of breeding season and retaining open cows for the next breeding season. Females were simulated in one of three classes; heifers, first parity cows, or mature cows. Within a class animals were simulated individually with a one day time step for the length of the breeding season. The simulation was essentially a series of single year simulations because individual animals were not tracked from year to year. However, calving date distribution within age

class was used to calculate calving date distribution for the next older age class the following year. Age structure of the breeding herd was determined prior to the simulation using Monte Carlo simulation (Azzam et al., 1990).

Heifers were assumed to have reached puberty prior to the start of the breeding season, except for 3% assumed to be non fertile due to failure to cycle. A heifer's birth date was drawn from a normal distribution of its theoretical dam's gestation length for a female calf, based off the start date of that dam's breeding season. Age at puberty was drawn from a normal distribution with 365 d mean and 25 d sd, which was added to Julian birth date while ignoring years. Estrous cycle length was drawn from a normal distribution with 21 d mean and 1 d sd. The date of first estrus within the breeding season was calculated by adding as many estrous cycle lengths to the puberty date as necessary.

For the older two age classes, postpartum interval was modeled with a 65 d mean and 18 d sd, with an additional 10 d following a first parity, and 14 d following dystocia. The date of first estrus within the breeding season was calculated by adding as many estrous cycle lengths as necessary to the date of the first cycle following calving. First service conception was decreased by 30 % for cows that were less than 60 d postpartum.

North Carolina Deterministic Systems Model. Lamb et al. (1992) used a deterministic model of cow-calf production to study breed combinations. Genetic effects were simulated as breed means and heterosis for F1 crossbreds, and cows were simulated by age class, as opposed to individuals. Animal growth was determined using a Brody (1945) curve,

$$W_t = A\,(1 - be^{-kt}),$$

where **t** is time in days, $\mathbf{W_t}$ is weight at day of age **t**, **A** predicts mature weight, **b** is indirectly

a y-axis intercept (birth weight) parameter, and **k** is a curve shape (daily growth rate) parameter. Lamb et al. (1992a) parameterized for Angus with the values 542, 0.946, and 0.0023 for parameters A, b and k, respectively. The primary reproductive trait was pregnancy rate, modeled by breed and age class, with parameters from the U.S. Meat Animal Research Center Germ Plasm Evaluation Program. Pregnancy rate ranged from a low of 68.2 % for Charolais heifers to a high of 97.1 % for mature Limousin cows. Pregnancy rate was reduced by 15 % for cows that experienced dystocia. There was no discussion of either puberty or postpartum anestrous. In their model pregnancy rate, calving difficulty, and milk yield described 76.6 % of biological efficiency, and pregnancy rate and average calf weaning weight described 82 % of economic efficiency.

Taylor and Naazie Efficiency Models. Naazie et al. (1997, 1999) used a deterministic model based on the model of Taylor et al. (1985) to study feed conversion efficiency of beef production. Animals were simulated as classes, as opposed to individuals, with class based on sex and breeding status of animals within the class. Reproduction was simulated as a potential number of calvings per dam and a reproductive rate that was the proportion of calves surviving to weaning per cow exposed to breeding. The lower limit of potential number of calvings was dependant on the reproductive rate in order to maintain sufficient numbers of replacement females. Taylor et al. (1985) referred to genetic differences and variables in the model, but these appear to have been simulated as input means, with no mechanistic simulation of genetics. Taylor et al. (1985) varied reproductive rate from 0.75 to 1.0, and also varied sex ratio, while Naazie et al. (1999) held reproductive rate at .8 while varying average age at culling, and then varied reproductive rate from 0.5 to 1.0.

Hirooka Breeding Objective Model. Hirooka et al. (1998a) developed a bio-economic deterministic simulation model for development of breeding objectives based on beef cattle production in Japan. It operated on a 1 d time step, modeling classes of animals. Inputs were described as being one of either animal traits, or nutritional, management, or economic variables. Animal traits were assumed to be partly controlled by genetics, although this was an issue of interpretation as opposed to model implementation as genetics were not explicitly simulated.

Females were classed by their reproductive parity. Puberty was addressed only as a management variable, influencing the time from heifer selection until breeding. Postpartum interval was also a management variable; this could be partly justified by considering management of prepartum nutrition. Conception was a function of an animal trait for single service conception rate and a management variable for an estrus detection rate (artificial insemination was assumed for all breeding). The single service conception rate was held quite low (0.5), in part to simulate artificial insemination, but also suggesting it accounted for many effects not modeled (e.g. extended postpartum interval, and interactions with lactation and nutrition).

Dystocia was determined per Bourdon and Brinks (1987a), and animals experiencing dystocia had a .15 lower probability of conceiving in the subsequent breeding season. While dystocia was not listed as an animal trait, it modified the single service conception rate. The .15 reduced probability of conception was applied proportionately. They also modeled gestation length and calf survival as animal traits.

They mentioned that other researchers had indicated reproduction should be modeled

stochastically but that the same researchers questioned the underlying biological reasons for reproduction traits having certain (non-normal) distributions. Observed reproduction traits will have non-normal distributions due to the typical management imposed on cattle. For example, a single calving season of limited length is a typical management goal. Heifers that reach puberty early and cows with a short anestrous period will not be bred until the breeding season begins, resulting in a skewed distribution. I suspect in some instances the need for a non-normal distribution for the underlying biology can be caused by not modeling all components and interactions of a system.

Hirooka et al. (1998b) also cited the complexity and cost of execution of the model as reasons for developing a deterministic model, and suggested that adding stochastic elements may not always improve benefits from modeling. While they may be correct that some stochastic processes could be modeled deterministically for clarity without loss of predictive ability, I feel they missed one of the main advantages of using individual based models. Stochastic processes used in these models allow the modeler to address risk. Individual based models are seldom constructed to predict the fate of a specific individual in an environment, but rather to model the interactions of individuals within a population. This is of importance in limiting environments, because the most fit individuals may not be intuitively obvious prior to the simulation.

I also have trouble accepting that cost of execution was a limitation issue for the types of questions Hirooka et al. (1998b) were using their model to answer. However, they would have a valid point when considering the use of simulation models as the objective function of a search for optimal decisions based on a techniques such as genetic algorithms. For example,

Meszaros (1999) used genetic algorithms to find optimal indexes with a non-linear profit model. A single search requires evaluating the model thousands of times, so cost of execution could become an issue with a complex mechanistic model.

Montana Beef Systems Model. Tess and Kolstad (2000a,b) developed an individual animal beef cow simulation model to model response of diverse genetics in diverse environments while capturing economic data. A simulated herd remains in equilibrium with respect to size and age structure over the simulated time, and the simulation has a discrete 1 d time step. All animals in a group have the same genetic make-up but are individually simulated and tracked through their life span. They also called their genetic input parameters genetic potentials. However, they defined a genetic potential as the maximum level of performance obtainable when sufficient nutrients were consumed. This is similar to the TAMU and CBCPM potentials, but is limited to nutrition effects, as opposed to all other environmental and genetic effects.

Their model development focused on animal growth, as suggested by their definition of genetic potential, and the genetic parameters were primarily to simulate breed differences. Since there is no individual genetic variation, it is not possible to study the unintended consequences of management decisions such as correlated response to selection. It also can not model changes in herd structure that result from management decisions.

Age at puberty in their model is a function of weight and age with respect to the animal's growth potential. This was done to correspond with field data that shows younger ages at puberty correspond to heavier weights. The model simulates a 21-d estrous cycle with no variation in cycle length. Conception is determined from a binomial distribution with an

input mean probability of conception. Pubertal estrus has 21% lower probability of conception, and cows that had calving difficulty have their probability of conception reduced by 10%. Both these reductions are applied additively, as opposed to multiplicatively.

A heifer's estrous cycle can stop due to low body condition (fatness less than 10 % of empty body weight), and can restart once fat is above 12 % empty body weight. Postpartum interval is a genetic effect, adjusted for body condition and rate of weight gain. It is modeled as 4 d longer for 2-yr-old dams and 1 d longer for older dams with dystocia. Dystocia is a random, non-genetic variable, but it incorporates dam lean weight with calf birth weight, which do have genetic components.

Rao Sheep Reproduction Genetics Model. Rao (1997) developed a model to study genetics of sheep reproduction. While not a beef cattle simulation model, this model is of particular interest because reproductive traits expressed categorically were simulated as having an underlying normal distribution. Litter size phenotypes, for example, were simulated by generating an underlying breeding value and adding underlying random normal deviations for permanent and temporary environment. The underlying "phenotype" was the sum of these effects, and was converted to observed categories by applying thresholds to the distribution.

Non-Systems Models of Reproduction. Pleasants (1997) built a stochastic model of reproduction to search for optimal calving date distributions as related to profit from selling a weaned calf at a fixed date. The premise was that calving prior to some date carried an increased cost, while calving later than that date was associated with reduced profits due to selling younger (lighter) calves. It was modeled with normal probability distributions of prior calving date, postpartum interval, length of estrous cycle, conception, and gestation length.

This question seems ideal for a systems model, as there are many factors one has to assume Pleasants appropriately accounted for in the increased cost of an earlier calving date. Factors such as the cow regaining body condition following weaning and prior to calving to reduce postpartum interval, feed costs, and management costs are all lumped into one number without explanation or variation. Pleasants suggested this sort of simulation is faster and easier to show the importance of how different variables operate. He later shows one of the down sides of this approach when mentioning parameters of one simulation were chosen to avoid complex regression equations. It is possible to maintain simplicity in a systems model by only varying the parameters of interest. If nothing else, it makes the assumptions chosen more clear. As discussed previously, the question of speed is largely a non-issue with systems models and current computer hardware speed.

Model Deficiencies. While there are likely many simplistic probability-based models of estrus synchronization, this review did not discover any systems models that incorporated estrus synchronization. I have been told of stand-alone hormonal models of reproduction, but none of that work was found in the literature. A mechanistic hormonal model of reproduction implemented as a module of a comprehensive systems model would be very useful for studying estrous induction and synchronization protocols.

Nearly all models in the literature focused on female reproduction and ignored male reproduction. The effects of bull serving capacity, behavior, and fertility were, I assume, absorbed into the female's probability of conception. The ability to simulate effects of different cow to bull ratios could be useful for both economic and genetic studies. Separating male and female fertility is potentially useful for studying artificial insemination.

Few models seriously addressed the genetics of reproduction. Tess and Kolstad (2000a) modeled breed mean parameters, but had no individual variation, making it impossible to use their model to estimate variance components for a fertility trait. The CBCPM models the genetics of age at puberty, probability of conception, gestation length, and dystocia, but it does not simulate an estrous cycle or postpartum interval mechanistically.

In summary, early models were based on empirical, deterministic equations, often with little resemblance to the underlying biology. Over time the models have evolved to include stochastic components, including genetic components, and researchers have refined components, making them more mechanistic. Early models typically simulated an average animal within some defined age and production category, primarily due to computation constraints. Beef cattle simulations have evolved to be individual based models, with each animal modeled throughout its life, and the time step of the models has decreased from seasons to months to days as computing power has increased.

Materials and Methods

I wrote a reproduction module that worked within the larger CBCPM model, making use of its simulation of biological, environmental, and management effects to provide input to the reproduction module. As described previously CBCPM had an existing reproduction module with genetic parameters for age at puberty, probability of cycling, probability of conception given cycling, gestation length, and dystocia, and many additional environmental effects.

The CBCPM is written in Fortran, and while it was possible to write my reproduction

module in Fortran, I chose Java instead. A professional programmer had developed an object oriented design for CBCPM (Nagel, mimeograph), with a few fragments of Java code written to illustrate the design. I implemented much of his object oriented design to provide a framework for the reproduction module to work within.

This decision to start a new version of CBCPM had its trade-offs. My Java version of CBCPM was only a partial implementation, due to time constraints. The most notable omission was lack of simulation of realistic growth and nutrition; animal growth was determined from a Brody (1945) curve with the same parameters for every animal. There was no simulation of climatic effects, no grazing, and no culling or selection simulated.

On the other hand, the Fortran version has many design elements aimed at optimization of execution, not optimization of development effort, dating back to its development on much slower computer processors. As mentioned, efforts to convert it to an object model were under way, and any reproduction module written in Fortran would soon have been abandoned.

My primary effort was to construct a model which would allow conducting genetic analyses of heifer pregnancy data (Chapter 4). The Java version of CBCPM is able to create a foundation herd from input files with appropriate simulation of the genetic traits, multiple sire groups, breeding groups, and mating groups, and can do so with multiple herds in one simulation run. I built the model anticipating that I and others will continue to develop and refine it beyond the capabilities of the current Fortran CBCPM. Both the following text and the model include references to things not yet implemented, but which should be included in a more complete version. These design elements had no effect on the data generated for

Chapter 4, but will be important to a more complete model.

Non-Reproduction Changes to CBCPM.

Multiple Pastures and Locations. The Java CBCPM is designed similar to the Fortran version with respect to what is simulated and the high level view of how the simulation functions. From the high level view it is still a timestep model, the sequence of events within a timestep is the same, it simulates essentially the same genetic traits in the same mechanistic way, and input parameter files have the same format. Changes to this part of the design were primarily additions or restructuring for easier maintenance. In the Java CBCPM a simulation occurs for a given ranch over a number of years. The ranch can have one or more herds, and a herd can be in one or more pastures. Pastures can have different climate, forage, and soils to allow for different locations. In contrast, the Fortran version allowed a single pasture at one location.

Pedigree. I added a pedigree manager to allow output from the simulation to be used in genetic analyses. When a Java method requests a new animal from the pedigree manager it supplies the identification numbers for the sire and dam if they exist, or a code for unknown parent in the case of a foundation animal. The method returns a new, unique animal identification number and adds that animal, with its sire and dam, to the pedigree. At the end of the simulation, the pedigree, consisting of each individual's unique identification number and that of their sire and dam, is written to a file. The pedigree records written to file represent unknown sire and/or dam, in the case of foundation animals, with a period for a placeholder.

Both CBCPM versions are time-step models, although the Java CBCPM was designed

with thought toward migrating to an event-based model or some hybrid of the two. The Java CBCPM has a fixed, one day time-step. Input parameter files are essentially the same, and I managed them with an ASCII text file editor. The model's output is written to ASCII text files for use in subsequent analyses external to CBCPM.

Simulation of Continuous Traits. The Java CBCPM simulates potentials for eighteen genetic traits (Table 2.1). The potentials contain a genetic component as described in the literature review of CBCPM. The reproduction traits are different than the Fortran CBCPM, but all other traits are the same. The input parameters for potentials are an additive direct genetic covariance matrix, temporary and permanent environment matrices, and breed mean potentials. Each of these three types of matrices has eighteen rows and columns, corresponding to the eighteen simulated traits. Each is symmetric, with variances on the diagonal and covariances on the off diagonals. The input values are stored in ASCII data files, and are read into the matrices at the start of a simulation. These variances and covariances do not change during a simulation.

Different breeds, or biotypes, of animals are simulated by setting input means of the potentials, as in the Fortran CBCPM. Each breed definition has a vector of eighteen means which correspond to the eighteen genetic traits simulated.

The Java CBCPM models phenotypic potentials the same as the CBCPM potentials (PO), defined previously. However, the methods to calculate the NAV were not implemented, so NAV was set to zero for all animals' traits. The simulations for this dissertation did not involve cross breeding, so this was not a limitation. The Java methods

Table 2.1. Traits simulated as potentials in the Java CBCPM and their classification with respect to simulation method and use in simulation of fertility.

Trait	Probability	Repeated	Fertility Trait
Birth weight	N	N	
Yearling weight	N	N	
Mature weight	N	N	
Milk	N	Y	
Age at puberty	N	N	Y
Postpartum interval	N	Y	Y
Probability of pregnancy given cycling	Y	Y	Y
Dystocia, direct	N	N	Y
Dystocia, maternal	N	Y	Y
Gestation length	N	N	Y
Mature empty body fat	N	N	
Appetite	N	N	
Unsoundness	N	N	
Probability of survival	Y	N	
Maintenance	N	N	
Intramuscular fat	N	N	
Fat-free composition	N	N	
Yield grade	N	N	

were designed to include the NAV, and later implementation will require minor modifications and construction of the input data files.

The BP_i is calculated as

$$BP_i = .5 \times (BP_s + BP_d) + \phi_i,$$

where BP_s (or BP_d) is the additive direct genetic effect of the sire (or dam) of animal *i*, and ϕ_i is Mendelian sampling for animal direct genetic effects. Unlike breeding values, which have an arbitrary base, breeding potentials include a trait mean and a breed mean. For simplicity, these two means are parameterized as a single input (their sum) in both versions of CBCPM. For a foundation animal its BP_s (or BP_d) equals the input breed mean for the corresponding sire group (or foundation dam group) plus ϕ. All other animals' BP are calculated as above, so they also contain some function of the input means. Mendelian sampling is simulated using a normal distribution with mean zero and variance equal to genetic variance unexplained by parents' genetic effects. Inbreeding is not accounted for in this model, although the method developed by Meuwissen and Luo (1992) as described by Lee and Pollak (1997) was considered. Although the simulated effect of inbreeding was not needed for the models tested in this dissertation, it can often be useful, and should be added to the Java model.

Mendelian sampling is calculated as

$$\phi_t = L_{18x18} \times r_{18x1}$$

where ϕ_t is an 18 x 1 vector of Mendelian sampling deviations corresponding to the eighteen simulated genetic traits, L is the lower triangular matrix from the Cholesky decomposition of a genetic covariance matrix (i.e. LL′ equals an appropriate genetic covariance matrix) and r is a random number vector from a standard normal distribution. The L matrix is obtained from the input genetic covariance matrix for each of the simulated genetic traits using a Java Cholesky decomposition method obtained from the Internet (Verrill, 1996). The vector R is

obtained using Java's ***Random*** class, which supplies the method ***synchronized public double nextGaussian()*** to obtain pseudo random deviates from a standard normal distribution. Each simulation is started with an integer parameter to use as a seed to the constructor method for a new Java ***Random*** object. One ***Random*** object is used to generate all random numbers within a simulation. A different input seed for each simulation can be obtained from published tables of random numbers.

Environmental deviations are calculated in a similar way; a vector of temporary environment deviations for animal *i* (E_{ti}) is calculated as

$$E_{ti} = K_{18x18} \times r_{18x1},$$

where K is the lower triangular matrix from the Cholesky decomposition of the input temporary environment covariance matrix (i.e. KK′ equals the temporary environment covariance matrix), and r is a vector obtained as described above, with random values different from those used to calculate ϕ_t. A vector of permanent environment deviations is calculated as

$$E_{pi} = J_{18x18} \times r_{18x1},$$

where J is the lower triangular matrix from the Cholesky decomposition of the input permanent environment covariance matrix (i.e. JJ′ equals the permanent environment covariance matrix) and r is as described above, with random values different from those used to calculate ϕ_t and E_{ti}. Permanent environmental variance must be non-zero for all traits in the input matrix so the Cholesky decomposition can be performed, but the method which calculates PO for a given trait only uses the permanent environment deviation if the trait is a repeated trait. Each of the Cholesky decompositions (i.e. L, K, and J) are obtained once

per simulation run, whereas the r vectors are resampled for each new deviate vector (i.e. ϕ_t, E_{ti}, or E_{pi}) to be calculated. In other words, when a new animal is created it requires three different r vectors for calculating the three deviate vectors, ϕ_t, E_{ti}, and E_{pi}, and the subsequent animal created will use three new, different r vectors.

Foundation animals are simulated first, with the full input genetic variance since their unknown parents can account for no reduction in Mendelian sampling. The parental breeding potential for foundation animals is the input breeding potential for the appropriate sire group and foundation group. All other animals' Mendelian sampling are calculated using half the additive direct genetic (co)variance, as their parents' breeding potentials (BP_s and BP_d) account for the other half of the genetic variance.

The Java CBCPM has a parameter for the number of times to run the simulation with the same set of parameters. The model reruns using the initial foundation sires, but all dams are regenerated as new, unique animals. Using the same foundation sires allows the multiple runs to be connected through the pedigree when analyzing the model outputs. This was desirable for increasing the number of offspring per sire. Multiple runs may be necessary due to memory constraints on the computer used to simulate the data. The random seed is set once, before starting the first run, and each rerun uses the next available random number following the last random number of the previous run.

Simulation of Categorical Traits. The Fortran CBCPM simulates potentials expressed as a probability with an approximation, as described in the literature review. This method allows some phenotypes to be out of the parameter space of a probability, requiring the approximated distribution to be truncated at zero and one. It also allows additive

environmental deviations to yield results out of the parameter space, also requiring truncation of the distribution.

I changed the model so that potentials for all traits were simulated as having a normal distribution of phenotypes. Probability traits were assumed to have a normal distribution on an underlying scale. Producing a normally distributed value to represent a probability is desirable because we can simulate its genetic and environmental deviations the same as all other traits. This is analogous to the assumption that threshold traits have an underlying normal distribution (Falconer, 1989), so I will refer to the normally distributed value as an underlying value on a corresponding underlying x-axis. When the simulation requires the use of a potential in probability units, the underlying value can be converted to a probability by calculating the area under the normal distribution between negative infinity and the underlying value. I obtained a Java class, Statistics.java (Liew, 1996), from a search on the Internet for Java classes that did numerical integration. The class Statistics.java contains the method ***public static double standardNormal (int N, double x)***, which returns the probability associated with the area under the standard normal curve from negative infinity up to the input value **x** (i.e. P(X<**x**)). The method also requires the number of trapezoids to use in the numerical integration, **N**, which I set to 20 for all calls to the method. I discovered that the method ***public static double standardNormal (int N, double x)*** only works in the range within 4 sd of the mean. I modified the method to convert any input less than -4 sd to -4, and any greater than 4 to 4. Therefore, all probabilities derived from this method have a slight bias to them.

The genetic and environmental variances for traits expressed as probabilities were

parameterized so they summed to one. Since the simulated mean of all genetic and environmental deviations equaled zero, the sum of these deviations had a standard normal distribution. The mean breeding value of foundation animals was parameterized in units of probability, as opposed to an underlying value, to make parameterization easier for the user to relate to. The model converted the input mean from a probability to its underlying value, and added the deviations, resulting in phenotypes distributed N~(input mean,1). Animals simulated as progeny of existing animals were simulated as parental mean breeding potential plus genetic and environmental deviations calculated as above.

To convert the input mean probability to an underlying value I wrote a Java method ***double findUnderlying(double probability)*** that did twenty iterations of a binary search for an x-axis value, **x**, of a standard normal distribution such that the probability of a random value from the standard normal being less than or equal to the value **x** (i.e. P(X<=**x**)) equaled the input probability. The probability was calculated from the value **x** using the Java method for numerical integration of the standard normal described above.

In addition, I wrote methods which allowed environmental deviations, in units of probability, to be added to or multiplied with the genetic potential. For example, Tess and Kolstad (2000a) decreased the probability of conception on the pubertal estrus by 21 %, which they did by subtraction from the input probability. I wrote a Java method which converted the underlying genetic potential to a probability by numerical integration, added the probability deviation (e.g. -.21), and converted the result back to an underlying value as described above. I wrote a similar Java method to allow a multiplicative deviation; the underlying genetic potential was converted to a probability, and the product of that

probability and the deviation was converted back to an underlying value. A multiplicative deviation can be used in cases where the size of the effect is proportional to the magnitude of the value being adjusted.

Reproduction Sub-Module Organization. I developed the reproduction module as four submodels to describe the different phases a beef female passes and then cycles through. These are birth through puberty followed by heifer estrous cycles. Then females are either anestrous (including pregnancy and postpartum interval) or in cow estrous cycles. This design allows a level of abstraction in the main time step routines. For instance, the main driver methods can call a Java method named **is_cycling()** without being concerned if the animal is prepubertal, postpartum anestrous, estrous, or gestating. This is discussed in greater detail in Chapter 3.

Management was separated from biology throughout the object oriented design of the Java CBCPM. A given animal is described in two different Java classes; one class describes characteristics of how it is managed, such as its service sire group, while another contains all biological data. This part of the design is also covered in greater detail in the next chapter.

Modeling Puberty.

Hormonal Changes Near Puberty. Puberty can be defined as the first time at which a heifer can conceive by natural service (i.e. exhibits estrus) and maintain a full-term pregnancy. The ability of a heifer to display estrus, conceive, and maintain a full-term pregnancy is a complex interaction of the central nervous system, hypothalmus, pituitary, ovaries, and uterus. The individual components of the reproduction system each are capable of functioning prior to puberty given (exogenous) stimulation, but do not initially function as

a whole due to interactions of the components (Kinder et al., 1995). In particular, the limiting step appears to be the regulation of LH pulse frequency. Prior to puberty the LH pulse rate is 1-4 per d, while at puberty it is 1 per h. Detailed, well-written descriptions of the reproduction axis prior to and at puberty can be found in the review papers of Schillo et al. (1992) and Kinder et al. (1995).

To summarize Kinder et al. (1995), the prepuberty low LH pulse frequency is due to inhibition of GnRH release from the tonic center of the hypothalmus controlled by a negative feedback response to estradiol. As puberty approaches, the negative feedback effect of estradiol diminishes. Decreased negative feedback on GnRH release allows more GnRH to reach the anterior pituitary, triggering more frequent LH pulses. The increased frequency of LH release results in more estradiol by increasing follicular development. These effects cascade, and eventually ovarian follicles release enough estradiol to trigger behaviourial estrus and a preovulatory surge of gonadotrophins, and puberty has been reached. The cause of the decreased sensitivity of the negative feedback of estradiol is not well understood. Associations have been found with several variables, but the physiological steps that lead to this critical step toward puberty are still not clear (Kinder et al., 1995, Hall et al., 1995).

Given that the onset of puberty is not well understood at the hormonal level, it is necessary to find other measures of puberty to base a mechanistic model on. Hall et al. (1995) hypothesized that the onset of puberty could be predicted as some function of body composition, metabolites, and (or) metabolic hormones. They defined puberty as estrus followed by formation of a CL and serum progesterone exceeding 1 ng/mL, and used an androgenized steer to assist their twice-daily estrus detection. Their study used two different

growth rate heifer biotypes which were fed two diet levels, resulting in a two by two factorial design. Live animal composition and metabolic status were assessed every 56 d from 7 mo age until puberty. At puberty 32 heifers were slaughtered and physical and chemical composition analyses of their whole empty bodies were done. In their various analyses they measured blood urea nitrogen, IGF-I, glucose, insulin, body weight, hip height, heart girth, body condition score, longissimus muscle area, and backfat thickness.

They did not detect a biotype effect on age at puberty, but diet affected the age at puberty, with those on a higher plane of nutrition reaching puberty 43 d before the other heifers. Plane of nutrition also altered the body composition in the slaughtered heifers. They cite work of Frisch (1976) which proposed a critical body composition that must be reached prior to puberty. However, they concluded the observed wide range of percentage of body fat did not support the hypothesis of a critical body fat level. In addition, they concluded that rate of fat deposition does not necessarily reduce age at puberty, and that "body composition has little direct physiological or biological relationship to the onset of estrous cycles." They also looked for metabolic signals, but found nothing to support a metabolic signal hypothesis, and found metabolic status did not predict puberty. They rejected the hypothesis of similar body composition, and also the hypothesis of similar concentrations of metabolic hormones and metabolites. They did not rule out metabolic and hormone status, but concluded the signals have not yet been identified.

Nonpubertal Estrus. This leaves mechanistic modelers of puberty with little to build on, which is why most simulations of puberty simply have heifer age as the primary component in determining when puberty was reached. Even here, further caution needs to be

added. Most research data on age at puberty has been obtained by visual observation of puberty. However, observing estrus alone is not accurate due in part to the occurrence of nonpubertal estrus. Nonpubertal estrus (NPE) is when a heifer exhibits social behavior consistent with estrus, but is unable to either conceive or to maintain pregnancy due to a lack of maturity. The heifers studied by Nelsen et al. (1985) that exhibited NPE averaged 89 d between first NPE and puberty. The incidence of NPE has been reported at 16.6, 25.5, and 62.8 percent (Nelsen et al., 1985; Byerley et al., 1987; Rutter and Randal, 1986, respectively), leading Bellows and Short (1994) to suggest it is a common occurrence.

In addition to problems with NPE, accurate detection of the pubertal estrus by observation can be difficult. Use of aids to assist detecting behavior associated with estrus can reduce the number of heifers in estrus that the researchers do not observe, and protocol for research data collection such as requiring presence of a CL and a minimum level of serum progesterone will limit the number of false estrus that could occur by using additional aids.

Genetic Effects on Puberty.

Age at Puberty. Genotype is one of the better predictors of when puberty will occur. Puberty genetics are typically discussed with respect to age of the heifer because it is a convenient scale for observation of puberty and because it is the scale that allows assessments of management implications on puberty (e.g. age at start of breeding season), so the trait is often referred to as age at puberty (AAP). Heifers of dairy breeds tend to have earlier AAP than beef heifers, and beef heifers with higher milk production tend to have earlier AAP than beef heifers with lower milk production (Martin et al., 1992). Beef heifers with earlier AAP had higher milk production within-breed (Laster et al., 1979).

There have been several studies of the heritability of AAP; the review paper by Martin et al. (1992) cited nine studies, with an average heritability of .40. The review papers by Koots et al. (1994) and Rust and Groeneveld (2001) report age at first calving, which is likely a different trait as it incorporates additional variance from ability to conceive and gestation length. There is also the possibility of fixed length breeding seasons effectively truncating the data space. Splan et al. (1998) reported heritability of AAP from USDA-MARC data to be 0.46. Arije and Wiltbank (1971) reported AAP phenotypic sd in two different populations of Hereford heifers as 26.7 and 35.8 d.

The bulk of research data is reported in terms of age at puberty, but it has been found age only describes part of the variation. There is likely an underlying scale of reproductive maturity, with puberty occurring prior to full maturity (Byerley et al., 1987).

Weight at Puberty. A heifer's weight is also useful for predicting AAP. Some studies suggest there may be a minimum weight for puberty to occur (Arije and Wiltbank, 1971; Short and Bellows, 1971). Weight at puberty accounts for prepuberty nutrition with respect to gain, but does not necessarily account for the balance of nutrients and other factors which might effect the timing of puberty (Greer et al., 1983). The timing of weight gain has been suggested as being important but with conflicting results. Clanton et al. (1983) concluded the timing of growth between weaning and breeding was not important, as long as an adequate amount of growth occurred. Greer et al. (1983) concluded that weight does not cause puberty; they determined weight at puberty to be the result of nutrition. Martin et al. (1992) noted that breeds with larger mature size are apt to be older and heavier at puberty.

I found nothing in the literature that suggested a heifer could sense or determine its

own weight. It is more likely that weight and onset of puberty are correlated responses to maturation, as opposed to weight causing puberty.

Environmental Effects on Puberty.

Season of Birth. Season of birth has been shown to have an influence on AAP (Arije and Wiltbank, 1971; Schillo et al., 1983). Heifers born in the fall and exposed to spring conditions after 6 months of age reach puberty younger than heifers born in the spring (Schillo et al., 1992). The season effect in some studies may be confounded with increasing plane of nutrition due to spring grass, but Tortonese and Inskeep (1992) showed treatment of prepubertal heifers with melatonin resulted in decreased age at puberty independent of differences in nutrition and growth. The way season affects puberty is not well understood, but is thought to influence the LH pulse frequency. In terms of natural selection there may be an advantage for being born in the spring or summer (Reksen et al., 1999), and the ability for fall-born calves to reach puberty younger may be a correcting mechanism as it would allow them to breed and therefore calve earlier.

Modeling effects which decrease age at puberty require recognition of the difference in age at puberty potential from most other genetic traits in CBCPM. The definition of a CBCPM potential is maximum performance in a non-limiting environment, which corresponds to a numerical maximum for most traits. With puberty, maximum performance is very early puberty, and earlier AAP is a numerical minimum. The definition says an individual's puberty cannot occur prior to the genetic potential, so the effect of limiting conditions are modeled by adding days to puberty. For example, a heifer born in the spring would have an adjustment added to its potential AAP to reflect the environmental effect of spring-born heifers being

older at puberty. This is the same as fall-born heifers being younger at puberty, but one cannot model the potential properly by reducing the AAP.

There is likely a decrease in any seasonal effect for animals raised closer to the equator. Certainly the effect of season is dependant on the hemisphere, and all the research cited above was done in the northern hemisphere. A model of seasonal effects may well require modeling the impact of season with respect to latitude. With all that said, I did not implement a seasonal effect on AAP, although I believe it should eventually be added to the model.

Social cues. Social cues may have an effect on AAP, but experimental results have been inconsistent, possibly due to interactions with other effects (Kinder et al., 1994), and possibly also due to error in observing puberty, such as NPE. In an experiment repeated over 4 years prepubertal heifers were either exposed or not exposed to bulls (Kinder et al., 1994). Bull exposure started when heifers were 350 d of age, and puberty was assessed by concentration of progesterone in serum samples obtained twice weekly from 12 to 16 mo age. In three of the four years more heifers exposed to the bull reached puberty at 13, 14, 15, and 16 mo age ($P<.05$). In a second experiment they tested the hypothesis that bull exposure's effect on age at puberty interacted with plane of nutrition. Heifers were fed for 1.6 or 1.3 lb/d gain and were either exposed or not exposed to a mature bull. Actual growth rates were higher than the targeted rates. In results pooled from 2 yr, heifers fed for higher growth and exposed to a bull reached puberty 73 d earlier than heifers fed for moderate growth and not exposed to a bull. Heifers fed for moderate growth and not exposed to a bull reached puberty 23 d later than heifers fed for moderate growth and exposed to a bull, and 23 d later than

heifers fed for high growth and not exposed to a bull. The reason for this design was to test if the lack of a bull exposure effect in the work by Roberson et al. (1987) was due to the relatively high growth rate observed in that study's heifers. While they detected an interaction of growth rate and bull exposure with age at puberty, the experiment by Kinder et al. (1994) resulted in heifers with high plane of nutrition and bull exposure having the greatest effect on age at puberty. I did not implement bull exposure as an effect on age at puberty.

Puberty Simulation. I modeled puberty as a genetic trait using potentials, the same as in the Fortran CBCPM, based on the trait AAP. It was modeled as a non-repeated trait, so the AAP potential was the sum of the breeding potential, non-additive value, and temporary environment deviations. I set the AAP genetic and environmental variances to 360 and 540 d squared, respectively, to model a 30 d phenotypic sd and a .40 heritability.

The AAP potential is one of the cases where maximum performance in an optimum environment corresponds to a minimum value, so it should not be possible to reach puberty prior to the AAP potential. This has implications for how environmental deviations are simulated; the environmental effects should not decrease AAP below the genetic AAP potential.

Simulated heifers as old or older than their AAP potential are next checked for adequate body condition; heifers are required to have at least 13 % empty body fat (approximately BCS 3.7) to start cycling. The Java CBCPM model currently does not simulate changes in body fat, and weight is strictly a function of age, but the code was included for use when the growth model is updated. Because nutrient utilization was not simulated, percent empty body fat (EBF) is held constant at 16 % (approximately body

condition score 5.0), which is assumed to be non-limiting for all conditions.

Heifers as old or older than their potential for AAP and in adequate body condition are also required to be more than a minimum weight. This weight is determined from the Brody growth curve parameters and simulated AAP, as Tess and Kolstad (2000a) did. I did this in a Java method named **heavyEnough()**, as follows,

```
private boolean heavyEnough()
{
        // Decrease puberty weight once past puberty age
        // Allows younger&heavier puberty weights
  double aapPotential =
thisCow.GetBreedingPotential(TraitList.ageAtPuberty);

  pubertyWeightAdjust =  ( thisCow.getAge() - aapPotential ) *
                        ( 1.4 * thisCow.growthModel.getAWT() / 525. ) ;

  if( thisCow.getWeight() >= pubertywt - pubertyWeightAdjust )
      return true ;
  else
      return false ;
}
```

where ***aapPotential*** is the heifer's potential AAP, and ***thisCow.getAge()*** returns the heifer's age in days. The variable ***pubertyWeight***, initialized in another method, is taken from the Brody growth curve at the time of her ***aapPotential***; it is the weight she would be at her potential for AAP given nutrition that had not limited her growth prior to that. The net result is she reaches puberty when she is AAP days old if she is at or above her growth curve weight and not less than 13 % empty body fat. If her growth was restricted prior to the AAP date puberty, she reaches puberty later (older) than AAP, when her actual weight exceeds the minimum puberty weight and she has at least 13 % empty body fat. The minimum puberty weight is a decreasing limit over time, modeling the observation of older heifers reaching

puberty lighter than similar heifers. The method getAWT() returns the asymptotic mature weight parameter for the Brody curve (A, or AWT). Including A as a term puts this equation into units of degree of maturity as measured by age and weight. The constant 525 represents the A value used by Tess and Kolstad (personal communication) when modeling Herefords, and will need to be refined when the growth model is updated.

This does not differentiate between preweaning and post weaning gain, while some studies have shown each to be related to age at puberty. The requirement of 13 % body fat does place some limit on diet, as does her weight. Although they were not implemented in the model, the places to include effects of season, rate of gain, and bull presence were marked with comments in the code.

Modeling Conception.

Threshold Trait Phenotype Simulation. The Fortran CBCPM simulates the phenotype of threshold traits, such as a female's probability of conception, by converting the phenotypic potential for the trait into an observed categorical trait (e.g. pregnant or not) by sampling from a uniform distribution between zero and one. If the sampled value is less than or equal to the phenotypic potential for probability of conception then the animal is categorized as having conceived. If not, she remains open until bred again, when a new random sample is drawn.

As described previously, the Java CBCPM models potentials for threshold traits on the underlying scale. To convert the values from the underling scale to the observed scale it is necessary to place a threshold on the underlying scale. I placed the binomial threshold at zero; in the case of conception, all animals with a phenotypic potential greater than zero will

conceive. Because conception is modeled as a repeated genetic trait (Table 2.1), the temporary environment deviation of the potential is resampled each time the animal is tested for conception. The model is structured so that animals are only tested for conception if they are within a breeding season and if they are in estrus. Conception is modeled as a probability. There is a zero probability of conceiving on any day other than a day of estrus within a breeding season. Also, heifers and cows that conceive are pregnant, as neither embryonic mortality nor abortion are simulated. The simulated trait is therefore the probability of pregnancy given estrus within a breeding season. The model is structured so that embryonic mortality and abortion can be added without modifying any other logic. If this is done, the trait will become the probability of conception given estrus within a breeding season, and the input probability parameters will likely need to be increased to offset the decrease in pregnancies following conception.

I also structured the model to allow a female to be bred by AI even if she were not in estrus, but she would have a zero probability of conception. This was done to account for mass mating following a synchronization protocol; it is primarily a book keeping feature to track semen usage.

Estrous Cycle Simulation. The Fortran CBCPM has a variable time step and includes code to modify the probabilities for the length of the timestep. Rather than simulating a 21 d estrous cycle, the Fortran CBCPM probability of conception given cycling is weighted for the length of the time step using the equation

$$PCON_L = 1 - (1 - PCON)^{step/30},$$

where *step* is the length of the time step in days, $PCON_L$ is the probability of conception in

a time step of length *step*, and *PCON* is the probability of conception in a 30 d time step. The probability of beginning to cycle is adjusted for the length of the time step in the same way.

In the Java CBCPM I modeled a 21 d estrous cycle with no variance to cycle length. When a female starts cycling the first day is estrus (d 21 of her simulated estrous cycle). This is different than Tess and Kolstad (2000a), who set the day of the estrous cycle to a random day drawn from a uniform distribution of the 21 days of the cycle. The result was they added, on average, 10.5 d to their AAP input parameter since it took heifers that many additional days on average to reach their first estrus.

In the Java CBCPM, females that do not conceive have their day-of-estrous counter incremented each simulated day (time step) unless the animal stops cycling (e.g. pregnant or anestrous). This is done each day until d 21 of her estrous cycle passes and the counter is reset to day one. Again, d 21 is the day of estrus, and since the model works on a full day time step, estrus essentially takes place the entire day (24 h). In Chapter 3 I discuss event-based models as an alternative to the daily time step; an event-based model would allow both the onset and the length of estrus to be simulated at any time interval, but such event-based modeling was not implemented in this study.

Heifer Conception Simulation. I modeled heifer estrus in a separate Java class than estrus of primi- and multiparous cows. Heifer estrus and conception are unaffected by dystocia, lactation, suckling, and the presence of their calf. On the other hand, heifers have been shown to be less than fully reproductively mature at puberty, with greater conception on subsequent estrous cycles (Byerley et al., 1987). These differences were enough to warrant separate Java classes for heifer and cow conception.

Pubertal Estrus. Byerley et al. (1987) showed that heifers in their study had pregnancy rates of 57 and 78 % at puberty and third estrus, respectively. Animals determined to exhibit NPE were removed from the study, and the above percentages are of the remaining animals. Based on this, Tess and Kolstad (2000a) adjusted conception rate of heifers at pubertal estrus by lowering the probability of conception by .21.

The Fortran CBCPM does not modify pubertal estrus probability of conception, but does decrease the probability of conception potentials for all heifers. In the Java CBCPM, I lowered probability of conception for the pubertal and second estrus using multiplicative adjustment factors. In contrast, Tess and Kolstad (2000a) applied their adjustment with subtraction. Their model did not include individual animal variation for conception rate, but the Java CBCPM does, and I felt the adjustment should scale with the magnitude of conception rate.

It was necessary to use a multiplicative factor of .268, as opposed to .21, to model the lower fertility on pubertal estrus from the 78 % conception observed on third estrus by Byerley et al. (1987). In other words, when mature conception rate (CR) is 78 %, my multiplicative adjustment, CR * (1 - .268), is equivalent to the additive adjustment for first estrous cycle, CR - .21, as used by Tess and Kolstad (2000a). Similarly, I decreased the probability of conception in the second estrous cycle by a factor of 0.128. This factor applied to Byerley et al.'s (1987) observed mature conception rate (78 %) models second estrus conception rate at 68 %, corresponding to an additive adjustment of 10 % decrease in conception.

In the Java CBCPM the probability of conception is stored as an unobserved

underlying value, as described previously in this chapter. An appropriate deviation for the underlying scale is obtained by converting the underlying probability of conception to the observed probability scale, multiplying it by 1.0 - .268 in the case of pubertal estrus, and converting this value back to the underlying scale. The underlying deviation can be obtained by subtracting the underlying probability of conception from this value.

Permanent Infertility. I modeled permanent infertility by placing a threshold on the permanent environment potential's distribution, as did Notter (1977). Females with a value to the left of that threshold can never conceive, regardless of their phenotypic potential.

Nutritional Anestrous. Once a heifer starts cycling, it is possible for her to stop if her body condition decreases too much (Bossis et al., 1999; Rhodes et al., 1996). Bossis et al. (1999) placed cycling beef heifers on a restricted diet. The heifers stopped ovulating 32 ± 3 wk from the start of the restricted feeding, as determined by transrectal ultrasonography. The heifers lost .38 kg/d and 22% of their initial body weight during the period of restricted diet, and had a 3.8 BCS at the time ovulation stopped. In another study beef heifers on a restricted diet stopped ovulating after 23 wk of restricted feeding and had lost 19% of their body weight (Rhodes et al., 1996). Using the approximation where BCS = (EBF - .02)/.03 (Bourdon, 1992), a 3.8 BCS is about 13.4 % EBF.

I simulated heifers becoming anestrous once their EBF went below 12 %, and estrous cycles resuming once it increased to at least 14 %. This is intermediate to the levels used by Tess and Kolstad (2000a); they stopped estrous at EBF less than 10 % and resumed estrous at greater than 13 % EBF. Short et al. (1990) noted that BCS 4 was generally sufficient to support estrous cycles. It is worth repeating the caveat by Tess and Kolstad (2000a) that body

condition is likely not the cause of anestrous, but is the best indicator currently available.

Cow Conception Simulation. I did not model the remaining components of fertility as thoroughly as puberty because they were not necessary for my study of puberty in Chapter 4. I have structured the Java code for simple insertion of some effects not implemented, and have commented the code where I would place the modifications.

Cow conception was modeled similarly to heifer conception. Both cow and heifer conception are based on the one genetic trait, PCON. I simulated cow PCON in a separate Java object due to the modifying effects on PCON different from the effects that modify heifer PCON. I distinguished a cow from a heifer based at least one pregnancy; since abortions are not simulated this is the same as requiring at least one calving to be considered a cow.

Following calving the cow has a period of anestrous, described below in the "Modeling Postpartum Anestrous" section. Many studies on the modifiers of fertility do not separate the postpartum interval of anestrous (PPI) from conception rate, so it is difficult to interpret the effects with respect to either PPI or conception.

Cow conception has been shown to be modified by time since calving, dystocia, age (or parity), weight gain after calving, and BCS at calving. Like heifers, once cows begin cycling it is possible for them to become anestrous again due to nutritional stress. In one trial, beef cows on a restricted diet stopped ovulating after 26 wk of the diet and lost 24% of their body weight, with a 36% decrease in BCS (Rhodes et al., 1996). I used the same BCS levels to trigger anestrous and resumption of estrous as with heifer conception. I also decreased probability of conception on the first two estrous cycles, in part to account for the occurrence of short, infertile estrous cycles, as discussed below, but did not simulate shorter cycle

lengths. I used the same multiplicative factors as for heifers.

Dystocia. I did not implement the simulation of dystocia in the Java CBCPM because it was not essential to my study of heifer pregnancy. It is a genetic trait in the Fortran CBCPM and should eventually be included as one in this version. I included direct and maternal dystocia in the covariance matrices used to simulate phenotypic potentials, but I did not parameterize them and did not model dystocia effects on other traits.

Modeling Gestation Length. Gestation length has been shown to be a heritable trait (Wheat and Riggs, 1958; Burris and Blunn, 1952; Andersen and Plum, 1965; Burfening et al., 1978; Azzam and Neilsen, 1987). I modeled gestation length as a genetic trait because the variation in its length could be important in some environments and management systems. I modeled it as a trait of the calf, as opposed to a repeated trait of the dam. Gestation length data collected from Holsteins by Jafar et al. (1950) had a 4.8 d sd, while Burris and Blunn (1952) reported gestation lengths for Angus, Hereford, and Shorthorn as having 7.1, 5.6, and 6.0 d sd, respectively. Azzam et al. (1990) modeled a 6 d sd with a 284.4 mean for a mature cow carrying a male calf. They adjusted the mean for parity and sex of the calf. The review paper by Andersen and Plum (1965) pooled estimates of gestation length for Angus and Hereford were 279.5 and 286.2 d, respectively.

Modeling Postpartum Infertility. Postpartum infertility can be defined as the time from calving until behavioral estrus is exhibited and a pregnancy can be maintained. It is useful to distinguish postpartum anestrus from infertility, because short, infertile estrous cycles can occur before the time at which pregnancy can be maintained (Short et al., 1990). The method used to observe postpartum anestrus will impact how the results are interpreted

when parameterizing the model. Methods which use a combination of observation of behavior and blood assay of progesterone (Zalesky, 1984; Custer, 1990) may be the minimum to detect the resumption of estrous cycles accurately. Additional measures include serial transrectal palpation (or ultrasound) of follicles on the ovary, and use of actual calving dates. The definition of the trait PPI may vary from study to study depending on how the end of the interval is measured.

I did not explicitly model short estrous cycles, but it is something to consider changing. Although the cow will not become pregnant she will potentially consume a portion of the bull's attention or semen from AI. I did decrease the conception on the first two estrous periods following the PPI, which could be interpreted as a portion of the cows experiencing short cycles.

The PPI begins at calving, with the uterus undergoing a phase of involution, taking from 20 d (Short et al., 1990) to 33 d (Custer et al., 1990). During this time it is not possible for sperm to be moved past the uterus, so even if estrus and ovulation were to occur the cow would be infertile (Short et al., 1990). The time of uterine involution has been shown to be insensitive to effects of bull exposure and sex of suckling calf. While it is the first phase of postpartum infertility, it is not considered to be the limiting phase (Short et al., 1990).

Once sperm can navigate through the uterus and the uterus can support a pregnancy there can still be infertility due to short estrous cycles and anestrous. During anestrous the limiting factor appears to be low LH pulse frequencies, but the signals which result in resumption of estrus are not well understood. There are waves of follicular growth on the ovaries, but the follicles regress before ovulation (Jolly et al., 1995). The factors which cause

the regression of the dominant follicle may change from conditions of slight under- nutrition as compared to conditions of more severe under-nutrition, but the net result is still follicle regression (Jolly et al., 1995). Periods of short estrous cycles are caused by the uterus prematurely initiating regression of the CL (Short et al., 1990).

In their review, Short et al. (1990) listed suckling and nutrition as main effects on PPI and season, breed, age or parity, dystocia, and bull presence as minor effects on PPI. Short et al. (1972) observed PPI in suckled, nonsuckled, and nonsuckled mastectomized cows to be 65, 25, and 21 d, respectively. Lamb et al. (1999) determined the suckling effect in Angus X Hereford cows to be due to the calf's presence, as opposed to the removal of milk (Table 2.2). In a subsequent study they showed no difference between machine milking 2X versus 5X per day, but a significant decrease of 11.6 d in PPI. Wetterman et al. (1978) showed longer PPI for dams suckled by two calves than dams suckled by one calf. Custer et al. (1990) observed a 16.7 and 16.4 d reduction in PPI without suckling in two trials. They also reported a 15 d longer PPI in primiparous dams suckled by female calves, but cited conflicting studies.

The Fortran CBCPM simulates postpartum anestrous using a genetic potential for the probability of beginning to cycle. I modeled postpartum anestrous as a genetic trait with a potential which is the minimum amount of time before estrous can resume. It is a repeated trait, so E_t is resampled at each calving. Because the PPI potential is a case where maximum performance corresponds to a minimum numerical value I decided to add, rather than subtract, time to the PPI to adjust for environmental effects.

Table 2.2. Effect of calf presence and milk removal from the cow on the cow's postpartum interval length (Lamb et al., 1999).

		Cow milked by		
Treatment	Calf Presence	Calf	Machine	Time to first estrus (d)
1	No	No	No	14.1 ± 3.1
2	No	No	2X daily	13.0 ± 3.1
3	Yes	No	No	14.2 ± 3.1
4	Yes	No	2X daily	17.2 ± 3.1
5	Yes	2X daily	No	33.9 ± 3.3
6	Yes	Yes	No	34.7 ± 3.1

I used an additive adjustment for BCS (ADJ_{BCS}),

$$ADJ_{BCS} = -20.8 + 1.8631 / EBF^{1.5},$$

which is essentially the same as Tess and Kolstad (2000a). They did not include a suckling effect, but the Fortran CBCPM did. I included a suckling effect to better model PPI in cows that lose a calf, and to potentially allow management techniques such as 48-h calf removal to be simulated. Cows that were suckled had 20 d added to their PPI. I also added 20 d to a cow's PPI if there was no bull exposure. The biology behind these effects is certainly more complex than these simple additive factors. I suspect they should be modified for time since calving, BCS, and plane of nutrition since calving.

Tess and Kolstad (2000a) allowed PPI to be reduced up to 7 d if the cow's 4-wk rolling average weight gain was positive. I modified their equation to allow cows with negative gains to have longer PPI (Randel, 1990; Short et al., 1990). Cows losing 0.454 kg

or more per day rolling average have 19.2 d added to their PPI. The length of time added to the PPI is decreased linearly as the average weight gain increases, reaching zero additional days at approximately 0.26 kg daily gain, and kept at zero for greater gains. At zero gain there are 7 d added to the PPI. Again, additive non-negative adjustments avoid decreasing the PPI below the phenotypic potential.

Finally, all cows have at least a 20 d PPI. While it may be possible for a few cows to resume cycling prior to that, it is unlikely they are fertile, due to the period of uterine involution. This truncation point is arbitrary, but in most scenarios this will not be the limiting factor. Dystocia has been shown to have a significant effect on PPI, but since I did not model dystocia I did not modify PPI for its effect.

When parameterizing the mean PPI it is necessary to remember that a cow with BCS 6, gaining no weight, suckling a calf, and without a bull present will have an additional 47 d of PPI from the adjustments described above. The maximum adjustment is 83 d for a cow BCS 4, losing weight, suckling a calf, without a bull present, and the minimum adjustment is 0 d for a cow BCS 6, gaining weight, not suckling a calf, with a bull present.

Modeling Male Effects on Conception. The Fortran CBCPM does not simulate bull fertility, although inspection of the source code shows an AI technician effect on fertility was introduced in later versions. All bulls could breed any number of cows on a given day and were 100 % fertile. I designed the Java CBCPM to allow for simulation of male fertility. I designed the mating to allow for bull competition and to account for the number of cows a bull could breed in a single day. Within a breeding pasture the number of cows in estrus are counted. Then the bulls are assigned a percentage of the matings. In the future this percent

can reflect bull social interactions. Next the bulls are assigned to mate specific individual cows. Before mating the bull's fertility for the day can be adjusted to reflect recent breeding use and the number of cows he is allocated to this day. Once his fertility is set the cow is essentially given a dose of semen with a specified fertility. The semen fertility is multiplied by the cow's probability of conception to arrive at the final probability of conception.

Validation. Meaningful validation of this model against the literature was difficult because the model was not fully functional. The lack of a growth model with individual animal variability reduced the output of the reproduction model to a function of the input reproduction parameters and the management imposed. While this model did not allow realistic simulations, it did provide the data necessary to study simulated genetic reproduction parameters in different management environments independent of interactions with other components in the system. Since essentially no biological or environmental modifications to the potentials were simulated, the primary validation was for proper simulation of the genetic traits. I wrote the components of potentials for the simulated genetic traits AAP and PCON (i.e. BP, E_t, and E_p) for each animal in a simulation run to a file.

The final validation data came from the model as described in Chapter 4. Of note, this includes modifications to allow foundation animals to have the same parents and the ability to run consecutive simulations that reuse sires. The ability to reuse sires allowed data sets for consecutive simulation runs to have genetic connections. This was necessary to increase the number of heifer pregnancy observations, as available computer memory was limited.

I calculated the mean and variance of each component and the simple correlation of each component with the other components for both traits, with the assumption that since

they were simulated independently the correlation should be zero.

Results and Discussion

The code was tested as it was written, but not with a formal method. Object oriented design techniques, discussed in Chapter 3, helped avoid and identify errors in the code. Runtime errors were treated as fatal, as the model currently is not intended for interactive use. Toward the end of this study I learned of programming techniques called extreme programming. I discuss this in more detail in Chapter 3, but one of the intriguing aspects is the reliance on writing test code as the model is developed, even before the modeling code is written. I believe there is great potential in this practice. In particular, it would mesh well with suggestions from Bourdon (personal communication) to develop a database of test cases for validation of any change to the model. Given that a goal is an open source model to be improved by researchers not necessarily intimate with the entire model, testing will be extremely important. I believe this holds great potential and can be pursued in the framework of extreme programming.

Verification of Simulation of Genetics. The observed mean components of the two genetic traits of interest, AAP and PCON, were considered close enough to the input parameters, with the exception of PCON E_t (Table 2.3). The PCON E_t was simulated to be zero, but the observed value of 0.504 suggested bias. Analysis of the sires, simulated with the same input parameters, do not show the bias (Table 2.4). The bias in the heifers was a byproduct of the way the model resamples E_t until the female conceives. It is more likely she will conceive with a large, positive E_t than with a negative value, particularly when her BP

and E_p have a relatively low value. This is supported by the non zero negative correlation between heifers' PCON BP and E_t (Table 2.5). In addition, the mean E_t for the 1,000 sires in the same simulation was -0.004, and the correlation between sires' PCON components BP and E_t was 0.003 and both were considered reasonably close to zero. Sire E_t is never resampled, so the near zero correlation supports the conclusion that the BP components were simulated independently, as intended.

Table 2.3. Input and observed means and variances for the components of AAP and PCON from 40,000 simulated heifers.

Trait	Component	Input Mean	Observed Mean	Input Variance	Observed Variance
AAP	BP	390	390.2	360	335.6
	Ep	0	0.000	0.0001	0.0001
	Et	0	0.013	540	538.3
PCON	BP	0.253[1]	0.261	.15	0.153
	Ep	0	-0.001	.10	0.1001
	Et	0	0.504	.75	0.462

[1]corresponds to a 60 % conception rate.

Table 2.4. Input and observed means and variances for the components of AAP and PCON from 1,000 simulated sires.

Trait	Component	Input Mean	Observed Mean	Input Variance	Observed Variance
AAP	BP	390	389.9	360	323.4
	Ep	0	0.0001	0.0001	0.0001
	Et	0	-1.492	540	535.5
PCON	BP	0.253[1]	0.251	.15	0.152
	Ep	0	0.006	.10	0.103
	Et	0	-0.004	.75	0.749

[1]corresponds to a 60 % conception rate.

Table 2.5. Observed simple correlations among simulated components of AAP and PCON from 40,000 heifers (above the diagonal) and 1,000 sires (below the diagonal).

		Age at Puberty (AAP)			Probability of Conception (PCON)		
Trait	Component	BP	E_p	E_t	BP	E_p	E_t
AAP	BP		-0.005	0.002	-0.009		
	E_p	0.016		-0.004			
	E_t	-0.002	0.0748				
PCON	BP	-0.038				-0.003	-0.212
	E_p				-0.000		-0.055
	E_t				0.003	0.018	

I plotted the distribution of phenotypic potential for the heifer PCON (Fig. 2.1), expecting a slight skew due to the resampling of E_t. If resampling of E_t had not occurred the figure should appear normal with mean at 0.253, which corresponds to 60 % conception given that the threshold was modeled at zero. However, the figure shows a dramatic shift of

heifers whose phenotype were initially less than zero to a positive phenotype. Again, a heifer with a positive phenotype would become pregnant during the breeding season and then E_t would not be resampled. The heifers remaining with a negative phenotype are primarily ones modeled as infertile. Since infertility was based on E_p, which was simulated independent of E_t, they appear to be normally distributed. The idea that this is due to resampling E_t is further supported by Figure 2.2, where the distribution of heifers' phenotype minus their E_t effect (i.e. BP + E_p) are plotted. This curve appears normal with mean close to the desired 0.253.

The observed variances were also reasonably close to the input variances. The input value for AAP E_p was non zero to allow the Cholesky decomposition to be performed. The observed variance for PCON E_t is likely reduced due to the same reason the mean is non zero. The variance of E_t of the 1,000 sires from the same simulation was 0.749, supporting the conclusion that E_t was simulated correctly and the heifer E_t variance was reduced due to the resampling.

The correlations between the components were close to zero (Table 2.5) as expected, with the exception of the heifers' PCON E_t correlations described above. Heifers in the simulation run reached puberty at 390.8 d on average, with a phenotypic variance of 875.3. There were 1,982 (4.96 %) heifers simulated as sterile, close to the threshold, which was set at 5 %. There were an additional 1,129 (2.82 %) heifers that did not conceive during the 120-d breeding season.

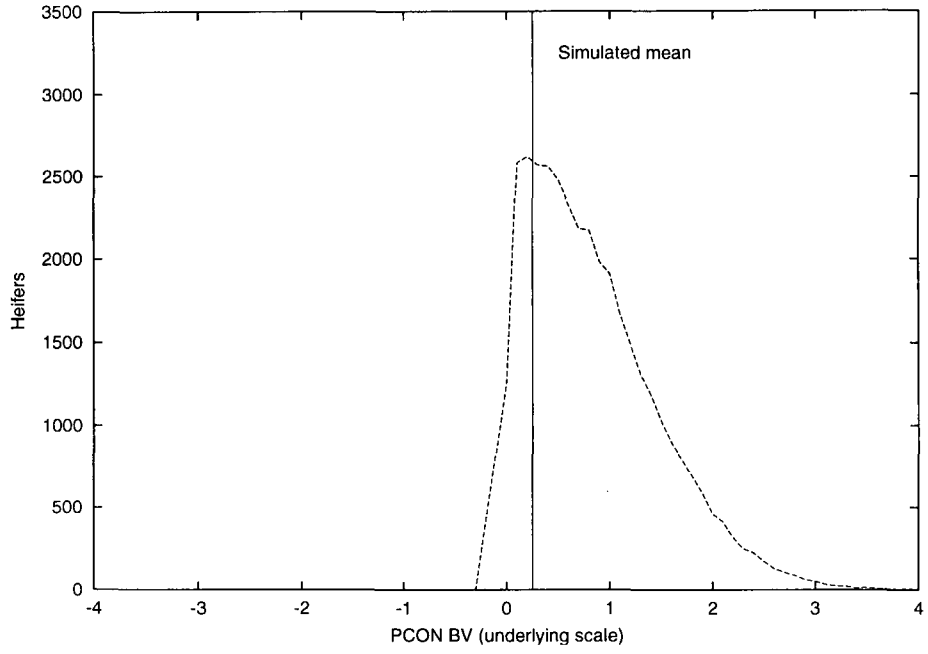


Figure 2.1 Distribution of simulated probability of conception (PCON) phenotypic potentials (BV + Et + Ep) output at the end of a simulation run.

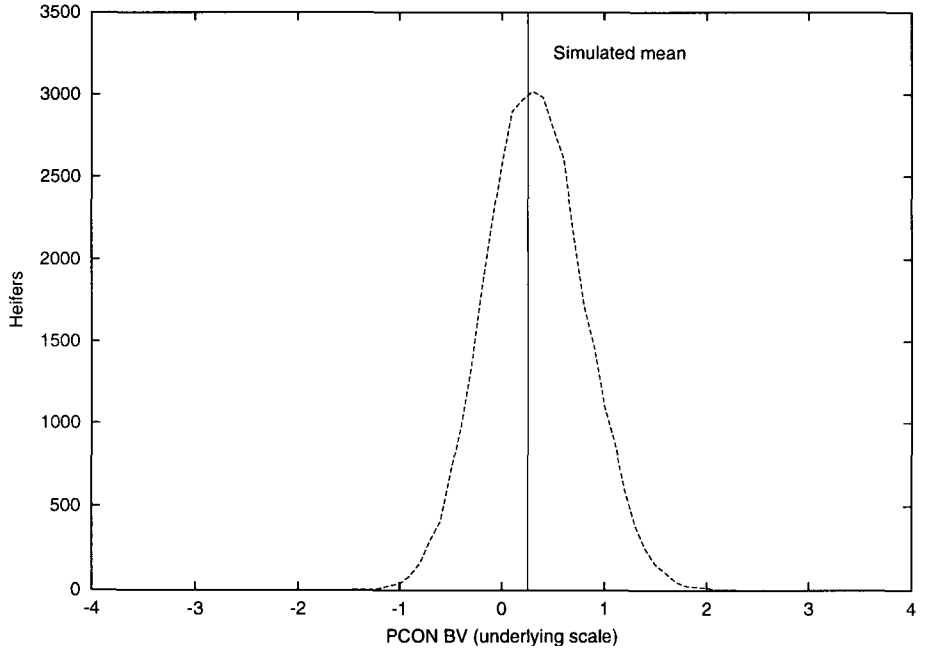


Figure 2.2 Distribution of simulated probability of conception (PCON) phenotypic potentials minus temporary environment effect (BV + Ep) output at the end of a simulation run.

To test the simulation of pregnancy I counted only heifers who reached puberty more than 62 d prior to the start of the breeding season to avoid the effects of reduced PCON on the pubertal and second estrus. There were 14,632 heifers that reached puberty more than 62 d prior to the start of the breeding season. Of these, 13,922 were fertile, while the remainder were simulated to be infertile. If infertility had not been simulated, the expected pregnancy rate would be the simulated mean PCON, which was 60 %. The observed pregnancy rate of all heifers that met the above criteria and conceived on their first estrus in the breeding season was 59.9 % (Table 2.6). Of the heifers that met the above criteria and were also fertile, 63.0 % became pregnant on their first estrus in the breeding season. Because infertility was based on the lowest 5 % of E_p potentials, there would be a tendency for infertile heifers to have a low phenotypic PCON potential. Mean PCON phenotypic potentials for fertile and infertile heifers were 0.48 and -0.43, respectively. Fertile heifers were more likely to conceive than the input parameter for conception might suggest, and their observed 63.0 % conception in the first estrus supports this.

Table 2.6. Cumulative pregnancy rate of heifers that reached puberty at least two estrous cycles prior to the start of the breeding season, with a simulated 60 % mean conception rate.

Estrous cycle in breeding season	Pregnant heifers (n)	Percent of heifers that were pregnant	Percent of fertile heifers that were pregnant	Expected percent pregnant if all were fertile
First	8,767	59.9	63.0	60.0
Second	11,481	78.5	82.5	84.0
Third	12,590	86.0	90.4	93.6

If all heifers in the simulation were fertile then the third and last column of Table 2.6 should be the same. However, the observed pregnancy rate (third column) for second and third estrous cycles was less than the expected pregnancy rate (last column). This was because as time progressed, measured by estrous cycle, an increasing percent of the heifers remaining open were infertile. It is also less because only E_t is being resampled, and E_t only accounts for 75 % of the variation in the PCON phenotypic potential. This is more clearly shown in the percent of fertile heifers that became pregnant (Table 2.6), again compared to the expected pregnancy rate in the last column. It is one difference between simulating using a threshold as opposed to the method of the Fortran CBCPM.

The fertile heifers that met the above criteria yet remained open after three opportunities to breed had an average PCON phenotypic potential of -0.88 on the underlying scale. These heifers could eventually conceive if given enough opportunities to have a favorable E_t.

I also tested the reduction in conception for the pubertal and second estrus. With an input mean PCON of 60 % and the reductions described previously, the expected conception for the pubertal and second estrus were 43.9 and 52.3 %, respectively. There were 4,189 heifers that reached puberty in the first 21 d of the breeding season, and 1,866 (44.5 %) conceived. Similarly, 8,691 heifers had their second estrus in the first 21 d of the breeding season and 4,594 (52.9 %) conceived.

Discussion. Evolution has led to a reproductive system that protects the female from jeopardizing her ability to replace herself. Not only does she need a properly functioning reproductive system, she also needs to be able to deliver and nourish the calf while

maintaining reproductive viability herself. It takes a minimum of one female born to replace her, which implies raising at least two calves to maturity assuming equal sex ratios and no death loss. Obviously the average number of necessary calves would be greater than two in the wild. In managed populations, where profitability is the measure of fitness, a line's survival requires up to five calves (Snelling, 1994).

Puberty. A heifer's survival would be jeopardized by allowing conception when she was structurally too small to deliver or too young to eat enough to maintain herself, feed a calf, and rebreed. It appears her survival is safeguarded by ensuring the reproduction system is functionally capable prior to its earliest use, but is disabled until the supporting systems signal that they are ready. Small wonder then, that the control of the onset of puberty is complex.

Weight and age only explain a portion of the variation in AAP, and it is likely not a cause and effect relationship. Care should be taken when using this simulation of AAP with an alternative growth model. It should be checked against field data for reasonableness. In particular, the key equation in the **heavyEnough()** method should be checked against field data when a different growth model and nutrient partitioning and prioritization are implemented.

There are advantages to early AAP, but disadvantages when AAP is too early (e.g. bred preweaning), suggesting there is an optimal AAP. There are several factors for a producer to consider when trying to determine optimal AAP. The main constraints are the desired age at calving and the use of a restricted length-breeding season. A review paper by Morris (1980) concluded that calving as a two-year-old has been repeatedly shown to

increase lifetime productivity as compared to calving at older ages. Many heifers in the U.S. are managed to calve as two-year-olds. Short et al. (1994) suggest that the cost to calve at two years age for *Bos indicus* breeds under some management and feed costs can increase beyond the return from greater production.

Rebreeding. Postpartum anestrous likely protects the cow from getting pregnant too early. The suckling effect protects the cow, as the calf will likely survive regardless of its dam's pregnancy status. While it may be modified by other factors, such as BCS, loss of a calf appears to be a major signal to resume estrous cycles. Likewise, the arrival of a bull signals the cow to resume estrous cycles.

Johnson and Notter (1987a) simulated PPI as a genetic trait, using normal distributions to simulate the additive direct genetic and permanent environment deviations. They used a Pearson III gamma distribution to extend the right tail of the temporary environment deviations. This was done in part to simulate the first 25 to 30 d postpartum as infertile. They refer to work by Short et al. (1972) as the cow being unable to initiate estrous during this period. More recent work (Short et al., 1990) suggests the cow is infertile due to uterine involution but that involution does not cause anestrous. Once a suitable growth model and nutrient partitioning are added to the model it will be interesting to compare the simulated phenotypic PPI distribution to the non-normal distribution Johnson and Notter (1987a) expected. I expect that as the model accounts for the major environmental effects, by simulating them mechanistically, the remaining temporary environmental deviations can be simulated with a normal distribution.

A restricted-length breeding season is a management tool to lower cost, increase

management intensity, and reduce effort. Year-round breeding results in having cows and calves in all stages of production at any one point in time. Breeding seasons as short as 82 d will result in calving extending into the breeding season (Short et al., 1990). While longer seasons can spread peak workloads, they can make management more difficult. It becomes more difficult to monitor production with a long or an undefined breeding season, and it may require some operations, such as castration, to be performed more than once a year, resulting in decreased efficiency. One of the stronger arguments for restricted length breeding seasons is meeting the nutritional requirements of the lactating cow with the least expensive feed. This implies matching lactation to the growing season; traditionally this has done by matching peak lactation stress to peak forage availability.

I feel a mechanistic model of reproduction at the hormone level should be the ultimate goal. Such a model will be valuable for exploring alternative estrous induction and synchronization protocols, as well as normal reproductive function. It might also be useful for designing research trials to increase the understanding of signals which trigger puberty and the resumption of cycling following calving. Prototypes of such a model might be explored using agent based modeling methods. However, many details are not well understood, and such a model incorporated into a systems model would add considerably to the difficulty of parameterization and validation.

The use of probabilities is convenient and intuitive, particularly for outcome traits such as heifer pregnancy and stayability. However, probabilities as a model input are not revealing. The use of probabilities in the model input suggest a lack of understanding of the biology by the modeler and potentially the literature. There are cases where knowing and modeling the

biological details would have little to no impact on the ability to use the model to answer questions, where the model would suffer little from the use of a probability. For example, modeling the process of conception given ovulation and the presence of fertile sperm can be left to probability for the majority of production scenarios.

Summary

A new version of the CBCPM model was written in Java. This model was designed with the long-term goal of being the start of a comprehensive, mechanistic systems model of beef production, and it was implemented with the short-term goal of generating data to use to better understand the relationship between AAP and PCON with heifer pregnancy EPD. As a result, many components necessary for realistic simulation of a beef system are not yet implemented, and even some of the reproduction components are "roughed-in", but are not fully developed (e.g. postpartum interval and dystocia). Additional work will be required to have a fully functioning reproduction model. However, this model does simulate AAP, PCON, and the management of breeding seasons.

Heifer puberty data collection is expensive. Studies of management effects on puberty may gain power by using related animals and analyzing the results with a mixed model that removes variation due to additive direct genetic effects while fitting the effects of interest. We know age at puberty is heritable and therefor has genetic variance.

Chapter 3

Rationale and Design for an Objected Oriented Simulation Model

Introduction

Systems modeling requires translating biological and management research results into computer instructions that simulate the real conditions. The modeler interprets the literature, designs and builds the model, designs comparative experiments, parameterizes the model, and analyzes the results. Useful models are difficult to design and build because interesting questions are often the result of interactions within and between biological systems and the management placed upon them. Building a model of a component of the system is comparatively easy, but does not allow the modeler to discover interactions, during development or execution, with other components of the system.

It has been more than 20 years since the TAMU model papers were published, and beef cattle systems models have progressed in many ways since then. The CBCPM, for example, models the interaction of forage growth with grazing, animals are individually modeled, and eighteen genetic traits are mechanistically modeled. Despite this, I feel simulation models of beef cattle systems are not living up to their potential; I see evidence of this in the things that beef systems models are not used for.

Simulation models could be useful educational tools in that they are a fast, cheap way to explore management alternatives. Both students and producers could benefit from the experience of parameterizing a model and then reasoning through the biology to understand the results. Producers could use systems models for more than education; the models could

be used to explore the impact and risk of changes in the existing production system. Researchers could also use models to test experimental designs.

Part of the reason for the limited use of existing models is the difficulty in parameterizing them. The CBCPM obtains its parameter from ASCII text files, and does not provide a tool for editing them, although the DECI model partially overcame this. Modeling some types of scenarios with CBCPM requires modifying code and recompiling. Parameterization can be made easier through use of different user interfaces depending upon the task. Databases of default parameters for typical production systems and interfaces to these databases, such as geographical information systems, could make the process easier, particularly for educational purposes.

I believe the real bottleneck in developing more complex models is in managing complexity, particularly in implementing and maintaining the design in software. Modelers use a variety of tools to implement their models, including computers, operating systems, programming languages, compilers, debuggers, and editors. Initially the computer hardware was the primary limitation, but the tools are evolving faster than modelers are adapting, and faster than modelers are making use of them.

Early Systems Modeling Tools.

Hardware, Operating Systems, and Networks. In the era before personal computers and high speed networks, a large percent of the developer's time was spent waiting for results following a modification of the code. Errors as simple as improper indentation of code could consume a large amount of time, especially if the job was batch submitted. The development team was often no more than a few collaborators; access to computer resources limited the

number of developers as much as anything else. For instance, the latest version of a model was likely stored on punch cards in a box in someone's office. Computers were not multiuser and were largely not interactive. Changes were made to the code by modifying individual punch cards, and then the cards were loaded into the computer for execution (often involving a walk across campus to a shared card reader).

Software Development Tools. Programming languages and compilers originally were developed with thought toward efficient use of the computer resources during runtime. Fortran was one of the first third-generation languages and was the language of choice for scientific computing. Fortran 77, the language most beef cattle models were written in, has many limitations compared to more modern languages. For example, variable names were limited to six characters in length, most likely to make efficient use of punch cards and random access memory (RAM) as the program was compiled, but this resulted in code that was difficult to read. Educators recommended programming practices which might be viewed as risky today, such as using a particular common block for multiple purposes within a program to reduce RAM requirements (Ageloff and Mojena, 1981). The design of the most current Fortran version of CBCPM still reflects the era of vector computing in its heavy reliance on arrays.

Programmers were faced with limited resources that often required creative solutions for sake of speed and access to RAM, but which resulted in code that was difficult to debug and maintain. As early as 1975 it was observed that up to half the cost of data processing went toward program maintenance and modification (Yourdon, 1975). As managers and educators became aware of this, more emphasis was placed on improving the structure of

programs to control the maintenance costs.

The programmer was given tools in Fortran 77 to help create better code, most notably the ability to write code in modules. Modules can improve a programmer's efficiency by allowing code to be reused, both within one program and also in different programs. This is efficient with respect to initial development time but also with respect to debugging time. A well designed module can be debugged and then (re)used with little further debugging effort and with increasing confidence. The more of these proven modules there are in a program the fewer places there are left to check for bugs. Modules also provide the advantage of breaking the program into discrete chunks, which ideally are specialized to a single type of task. These modules could be stored in separate files or card decks, making larger applications more manageable.

Still, programs were built largely with the limitations of the tools in mind, and the design process reflected the procedural languages used to implement them. Top-down design and flow charts emphasized the sequence of events, and the real world was modeled within that context. The hardware limitations also had a strong effect on how systems were modeled (e.g. average animals). Compilers were able to detect and report syntax errors, but there was little direct help within the languages for the programmer to avoid logic errors. In fact, the C language took things backward in this respect, as its compilers gave the programmer nearly unlimited direct access to memory, and the programmer was assumed to understand the consequences and side effects of each statement.

Documentation of applications was encouraged but typically neglected. Flow charts were drilled into students, but I never observed anyone using them to the degree we were

taught. Code was supposed to be "self documenting", but usually it was obscure, often even to the programmer who wrote it, given a little time.

Improvement of Development Tools. Development tools have continually improved. Personal computers, while initially too slow for simulations, were used as terminals, increasing the availability to mainframes. As networking on campuses began it was possible to modify code from the comfort of one's own building. Punch cards were replaced by electronic storage, and on-line editors became available. A model could be stored in a central on-line site with more than one modeler having access to it at a time. The speed of mainframe computers increased, which allowed more complicated models and more testing runs. Still, there was time to analyze data while waiting for results from the latest modification or simulation, and the computer hardware was still a limiting factor in the advancement of models.

At some point in the late 1980's or early 1990's this balance began to change. Computers became faster and the tools to modify the programs became much better. Computers became more interactive, on line storage capacity increased, cost of computer cycles decreased, and terminals made them more accessible. With better networks it was possible for teams to work on a single program, each modifying and testing. These advances eventually had the combined effect of removing computer hardware as the limiting factor for most beef cattle systems models.

Current Systems Modeling Tools.

Hardware, Operating Systems, and Networks. Tools available to modelers now have seen huge improvements since the TAMU model papers were first published, and new tools

have been added. Computer power has increased rapidly while at the same time the computers have become smaller, cheaper, and pervasively accessible. Moore (1965) observed that the amount of information which could be stored in a given area of integrated circuit doubled every year. The rate of progress has slowed to a doubling every 18 months since about 1980, where it holds steady even now. This rate has become known as Moore's Law. Modern desktop computer processors can do in a morning what would have taken a 1980s era computer a month or more. The size of RAM has increased, as have long-term storage devices such as hard disks. Early models were limited to using an average animal; Kahn and Spedding (1983) were able to use an individual based model only because they simulated 40-head herds. In contrast, the study done in Chapter 4 used thousands of animals in each simulation run, and the computers used for that analysis have since been replaced with much faster computers that also have more disk and RAM storage.

Today's operating systems are interactive, multi-user, multi-tasking, and allow the programmer to address huge amounts of storage. Networks have become an integral part of a computer system and the distinction of access to multiple computers on a network versus access to a single computer that happens to have networked components is blurring. Networks allow disk drives to be shared, and some computers' primary function is to manage storage area. Central processing units have been connected in clusters or grids on high speed networks with the help of languages such as LAM/MPI and PVM, and extension of these techniques across the Internet is happening. Few animal science modelers are taking advantage of these huge leaps in computer power, although optimization techniques which use a simulation model as the search engine (Meszaros, 1999; Bourdon, 1998) clearly will be

able to exploit these systems.

Software Development Tools. Programming languages and software development environments have also improved greatly, making it easier for the programmer to handle larger projects and easier for the programmer to write good code. As noted previously, simulation models of beef cattle production have historically been written in Fortran, due in part to it being the primary programming language when modeling started. It appears that those early researchers in turn trained their graduate students in Fortran, most likely because there was a base of existing code to work with, but also because the researchers were familiar and comfortable with it. Even as recently as 2000, PhD students were investing substantial programming efforts into models written in Fortran 77 (Doyle, 2000).

This is not to say Fortran is not up to the task of building complex models, but most existing models are based in early versions of Fortran which lack modern programming tools such as data structures, dynamic memory allocation, and object classes. Analysis of source code for the model by Tess and Kolstad (personal communication) showed their use of Fortran 90 was a timid step; while they did use data structures, they implemented the model with only five subroutines and a few functions, and did not make use of dynamic memory allocation. I have not found any beef system simulation models written in C or Pascal.

Integrated development environments (IDE) have become available on most platforms, allowing the programmer to code, compile, run, and debug from within the same environment. Artificial limitations, such as six character variable names on instructions beginning after a certain number of blank spaces are less prevalent. Point-and-click interfaces and editors which provide completion of names as they are typed encourage the programmer

to use longer, more descriptive names. Editors make use of color to help the programmer identify statements and variables. While debuggers have been available for quite some time, the IDE environments make using them a seamless tool for analyzing the code during execution and for tracking errors.

System design has come a long way from flowcharts, and programming languages have gone through a major change. Rather than separate, non-integrated steps from design to coding to debugging, the use of object oriented design and languages is becoming more common. The strengths of object oriented design is the focus of the rest of this chapter, but first I'll digress to ensure Fortran modelers are on the same footing.

Dynamic Memory Allocation and Data Structures. Modelers using the earlier versions of Fortran did not have use of two powerful programming tools - programmer defined data structures and dynamic memory allocation. It is a bit surreal to me to feel it is necessary to describe data structures and dynamic memory allocation to modelers in 2003 since I first used them in PASCAL programs in 1984. However, it is apparent from the literature that most beef simulation modelers have not made use of these tools and may not appreciate what they offer to modeling.

Data structures allow the programmer to create new data types from aggregations of variables using the intrinsic data types (e.g. REAL, CHAR, FLOAT) and using other data types defined by the programmer through the use of data structures. As a new data type it is then possible to define a variable to describe an instance of this data type or even an array whose elements are each an instance of this new data type. This is primarily an organizational construct. It reduces the programmer's work and is a higher level of conceptualization. For

example, a programmer-defined data structure called COW might be an aggregation of variables for weight, birth date, sex, sire, and dam. The programmer could create an array of COW elements, with each COW element in the array having its own set of the variables. While an array of new programmer-defined data types is intriguing, their real power comes from the understanding and use of dynamic memory allocation.

Dynamic memory allocation is a programming tool which allows the amount of RAM needed for execution of the program to be determined at the time of program execution rather than time of compilation. For example, CBCPM lacks dynamic memory allocation and requires the programmer to determine maximum herd size prior to compiling the program. To increase herd size beyond that limit, the code must be recompiled. This is similar to buying a word processor that can write up to 100 pages, and to get 101 pages requires the user to obtain a different version of the word processor from the developer. While the developer and user of a simulation model are often the same person, lack of dynamic memory allocation creates an artificial constraint during the design process and possibly later when modeling. Versions of Fortran 90 are capable of using dynamic memory allocation. Tess and Kolstad (2000a) used Fortran 90 but did not make use of dynamic memory allocation capabilities (personal communication).

To illustrate the power of defining new data types combined with the use of dynamic memory allocation I'll return to the example of a COW data type which includes variables for weight, birth date, sex, sire, and dam. When defined as an array, a given animal's sire (dam) variable would contain an index to the sire's (dam's) COW element within the array. If the array were implemented in the C language and named **animals**, then animal i's sire's birth

date might be accessed as

animals[animals[i].sire].birth_date.

Alternatively, with dynamic memory allocation, the COW elements could be organized in an ancestors tree. The variable for sire (dam) could be defined to "point" to the location of the sire's (dam's) element in RAM, allowing navigation from a given animal to all its immediate ancestors without use of array indexes. Again in C, the birth date of a sire of an animal I'll access with a variable called **animal** is

animal.sire->birth_date,

which is simpler and more intuitive, and the paternal grandsire's birth date would be

animal.sire->sire->birth_date.

Writing that in one step with an array would result in code very difficult to debug and maintain. Subgroups of COW elements can be organized essentially the same as with CBCPM's control vectors. By allowing one user defined data element to "point" to the location of another data element in RAM complex organizational structures can be created.

In addition to allowing access to COW elements based on their location in RAM with pointers is also possible to dynamically create (and destroy) COW elements. A Fortran 77 array implementation requires knowing the maximum herd size at compile time, but an implementation using dynamic memory allocation would not have that artificial limitation. Instead, COW elements would be created as needed and destroyed when no longer necessary. Once an element is destroyed its RAM is released for reuse, perhaps by a later generation of animals in the simulation.

Dynamic memory allocation comes at the price of increasing the potential for

programming errors. In the C language a typical programming error is to over-index an array. That is, to access a memory location past the end of the memory dynamically allocated for the array. The results are unpredictable and often cause side effects which are difficult to trace back to the root cause.

State of the Art. The CBCPM was the most complex systems based model I found in the literature. While DECI was a step forward in usability with its graphical user interface, the user had access to only a subset of CBCPM's capabilities. It was one of the few individual based models I found, and it was the only model that attempted to mechanistically model genetics for all traits of importance to cattle production.

However, it was evident that the core of the CBCPM model aged as efforts were shifted from the system design to refinement of components. Modifications were made by commenting out previous programmers' implementations and adding new ones. Keeping the original code was helpful, but it added to the amount of information the programmer had to sift through. It is likely some of the saved code would no longer function as intentioned because assumptions made when it was written potentially changed over time. The code remained as Fortran 77, and although excellent use of modules was made, the limitations of the language remained. In short, CBCPM was the most complete systems model but was becoming unmanageable due to its complexity, largely due to the limitations of Fortran 77.

The Need for Change. It is expensive to retool a complex simulation model, and a researcher cannot afford to spend much time chasing the latest fads in software development. Time spent on keeping up to date with advances in software engineering is time not spent on research and the money it generates. At some point, however, the cost of stopping the model

refinements and doing a complete rewrite outweigh the costs of continuing with the status quo.

More complicated models were being envisioned years ago, such as sire selection by simulation (Bourdon, 1998). None of the existing beef simulation models allow for selection on EPD, let alone on EPD calculated from the ongoing simulation. This is a tractable problem and would be interesting to use to investigate correlated response to selection. While many forms of correlated response may not be modeled and therefore would not appear, some less obvious forms may arise from the model.

Bourdon (personal communication) has suggested that productivity could be increased by creating a model core which could be reused by researchers rather than the current practice of creating models of subsystems. If a researcher's work resulted in an improved module of the model, it could replace the existing module. The potential is for experts in the various fields to modify and upgrade the model while using it for their own research. The individual cost is low and the group's benefit is high.

Graduate students are one of the tools available to an academic modeler, and they, too, are changing. My computer-phobic wife wrote Fortran as an animal science undergraduate, but I would challenge you today to find animal science undergraduate students with Fortran experience. Today's students come trained on point and click environments, and are likely more intimidated by UNIX environments, editors, makefiles, and cryptic compiler messages than were students who had experienced command-line DOS.

As a final observation on how far behind beef systems modelers are, none make use of the Internet. Even DECI, which had a brief web page description, was not available

through the Internet. Advances in networking and in programming languages now allow computers on a network to function as a large, multi-processor computer, all executing parts of the same application. These computers, known as clusters or grids, provide huge, untapped resources, making computer speed a non-issue for most simulation applications. In addition to the points I have argued above, I believe the main reason there has been stagnation in beef modeling efforts is due to the complexity of the models. Management of the complexity of the systems has become the limiting factor; models have reached the point where it takes immersion in a development effort to understand the system and what side effects to expect from any changes to the code.

As computers become faster the complexity of what we ask of them increases. Programmers (and modelers) have a finite capacity for complexity, but advances in software development tools have allowed the problems of greater complexity to be addressed. This can be seen in changes from machine language to assembler, to third generation languages (e.g. Fortran), and as I will argue, to object oriented techniques. Computers have become faster and runtime has become cheaper than development and maintenance time. Computer languages have been able to make use of the cheaper runtime to make it easier for the programmer to write better code. Fast, cheap computers have allowed the emphasis on design to shift from fast, memory efficient programs to programs that are more reliable and cheaper to maintain.

Advantages of Object Oriented Design

I believe a major step in advancing beef simulation modeling is to make the transition to object oriented design and programming. Object oriented techniques were devised to

increase the probability that the software developed will do what the user intends, have fewer bugs, gracefully handle inconsistencies, and be less expensive to develop and maintain. It is not necessary to use object oriented design to produce a robust application, and object oriented design is not a silver bullet or guarantee of success. It is simply a set of tools to develop more robust applications. However, these tools are withstanding the test of time, and deserve a second look by beef cattle simulation modelers.

Why Programs are Difficult and Expensive to Develop. A computer programmer's job is to write instructions that will allow a computer to simulate some real-world system at some acceptable level of detail and accuracy. While simple to describe, this process is rife with potential for failure. First, the programmer is often not an expert in the domain of the real-world system being modeled. Those who are expert often have difficulties explaining the system, so the programmer's conceptual model may not agree with the expert's conceptual model (let alone with the real system), but the difference may be difficult to recognize or articulate for either the programmer or domain expert. Instructions given to the computer will likely produce results, but not necessarily correct results based on the programmer's conceptual model, the expert's conceptual model, or the real world.

A domain expert often refines the conceptual model, requiring changes to be made to the set of computer instructions. Unless the original programmer is still available to make these changes, another programmer has to learn the domain expert's conceptual model, the previous programmer's conceptual model, and how that conceptual model was implemented. As models become more complex the number of domains, domain experts, and programmers involved in the project increase, increasing the potential for misinterpretations and errors.

Programming Tools to Reduce Development and Maintenance Time. In modern software development the major cost of program development is spent on its construction and maintenance of code, and not on the actual running of it. With that in mind, trade-offs have been proposed which allow for cheaper development and maintenance at a cost to runtime performance. The increasing speed of processors has allowed the runtime cost to be increased, and technologies such as optimizing compilers and late linking have made the increased cost more bearable.

As discussed previously, modularity is a software design technique which can be used to improve a program. Ageloff and Mojena (1981), referring to Fortran 77, wrote that "subprogramming capability is a sophisticated refinement of the language that ... promotes programs that are easier to code, debug, and maintain." Fortran supports modularity with subroutines and functions. Although CBCPM makes use of functions to act as containers for different tasks, their use is limited. Functions allow details of the code to be hidden and can make expressions more readable. Functions allow repetitive tasks to be written once in a generic fashion, reducing programming errors by allowing for one place to debug.

Modules allow code to be organized and compartmentalized by its purpose. They allow a level of abstraction, which can be used to make the code more readable; the details are available but are not in the way of the larger picture. Modules also allow code to be reused within a program, which saves programming time, primarily by reducing the need to debug a time-proven module.

A proven module can be used to decrease programming errors and decrease development time through reuse in any number of subsequent programs. This was a large part

of the success of the Animal Breeder's Tool Kit (ABTK) (Golden et al., 1992). Not only were the modules useful in the initial development of the toolkit, they were also available to rapidly develop future tools, such as tkblup, ds6, and dscat. As an aside, the ABTK illustrates an interesting point, which is that object oriented techniques do not require use of an object oriented programming language for implementation. A corollary is that an object oriented language can be used to produce a working program that does not take advantage of the language's object capabilities.

Advances in computer networks, especially since the early 1990's, have made collaboration among researchers at different physical locations easier. It is helpful for each programmer to have access to the code and make changes, while maintaining source control. Programmers can "check out" modules from a central library, modify them and then replace the original with the modified version. Initially this occurred by checking the modified module back into the central library, recompiling all dependant code and linking the objects into an executable program. With the advent of high speed networks it has become possible to delay the linking until run time.

Object Oriented Design. Object oriented design and programming has evolved as a way to capitalize on the individual various strategies that have evolved to increase programmer productivity and decrease bugs. Object oriented design goes beyond simply gathering the individual strategies together; rather it approaches software engineering from a very different perspective, resulting in a model greater than the sum of its parts.

Gosling and McGilton (1996) describe object oriented (OO) techniques as "a collection of analysis, design, and programming methodologies that focuses design on

modelling the characteristics and behavior of objects in the real world." The emphasis on modeling is their own, and it is interesting how closely their description fits the goals of those modeling beef cattle systems. I suspect this description is more foreign to a programmer accustomed to developing business applications than it is to a systems modeler.

What are objects? Objects in the real world of beef production include things like cattle, pastures, feedlots, and feedstuffs. Each object in the real world has its own state and behavior, and the computer objects we create to model them also each have their own state and behavior. A particular cow's state might include its weight, body condition, pregnancy status, and genotype, while examples of its behavior include feed consumption, growth rate, and ability to conceive.

In the following sections I explain the organization of the new Java CBCPM and introduce some key object oriented design and programming concepts. In addition, I illustrate the use of the Unified Modelling Language as a communication tool. These topics are interwoven. I have prefaced section headings for the new model, object oriented techniques, and the Unified Modeling Language, with J-CBCPM, OO, or UML, respectively, to indicate the main emphasis of the section

OO: Classes. The state of an object is represented in instance variables, and the behavior of an object is modeled by instructions, called methods. Methods can describe changes in behavior by changing instance variables and therefore the state of the object. Objects from the real world are modeled in software by writing a class. A class is not an object, it is a blueprint of an object, and is used each time a new object is created. For example, an individual cow could be modeled by creating a cow object from a cow class, and

her sister could be modeled by creating a new, different cow object from the same cow class. Each cow object is an instance of the cow class, which is why the state of a particular object is said to be represented in its instance variables.

While an object may appear to be a fancy name for a Fortran subroutine with local variables (or C function and data structures) it really is quite different. Picture each cow object with its own instance variables and methods. At the highest level we are modeling a cow, and in keeping with that we have designed a cow object to describe the state and behavior of a cow. In Fortran we might create a subroutine to describe the behavior of a cow and then create arrays of variables to describe some number of cows, but we do not mentally model each Fortran cow as having its own copy of the code and just the data that describes a single, specific cow. In Fortran the design emphasis is on the code, with the data serving a supporting role, while in object oriented design, by articulating the difference between state and behavior, both the methods and instance variables are there to support the object.

OO: Encapsulation. Similarly, an object is like a data structure but is much more than a data structure. Like a data structure, a specific cow object contains the data that describes its state. However, the object typically contains methods which control the access to the data. It is possible, and usually advantageous, to not allow direct access to an object's instance variables from outside the object. Instead, the instance variables are queried and changed through methods. Gosling and McGilton (1996) describe the methods as a protective layer around the data. As a programmer trained in the world of limited computer resources (RAM, in particular), this appeared quite wasteful to me. However, the point is that we are no longer in the world of limiting computer resources, and the purpose of this protective layer is to do

everything possible to ensure data integrity.

Let's go back to the cow object and assume it has a weight instance variable. By restricting access to the variable through methods it is possible to allow a cow object to be responsible for the range of values the variable can store. A setWeight() method might limit allowable weights to fall between zero and one thousand kg. Contrast this to a weight variable in the Fortran CBCPM which is likely to be globally visible to most subroutines. A programmer could write code which calculated the daily weight gain for a cow and add that gain to the weight variable. In that example it is the programmer's responsibility to test and ensure the weight gain results in a valid value before adding it to the current weight. In an object it becomes the cow object's responsibility to ensure its weight is valid. Someone testing or debugging the Fortran code would need to examine the value being placed in the variable at every location in the code where it is potentially accessed or changed. With an object that enforces access through methods there is only one place to check, which is right in the object itself.

Clearly the cow object is created by the programmer, so it again falls back on the programmer to ensure the data is tested. However, OOD allows us to think in terms of it being the cow object's responsibility. It is a mental model which makes the responsibility clear and increases the probability of better code.

A related key point of OOD is that the cow class may well have been written by someone not part of local development team; the class may have been obtained through a network such as the Internet. Accessibility of objects through networks is a key part of some implementations of OOD, most notably the Java language. Again, the mental model of it being

the object's responsibility to ensure the integrity of its state is quite helpful.

Getting back to the role of methods in an object, they perform another important function by hiding an object's internal workings from the outside world. For example, an object that tracks a cow's estrous cycles might have initially been developed to use a fixed 21 d cycle. However, a modeler could later modify it to be a variable length cycle with genetic and environment effects. By requiring the cycling status to be accessed through a method it greatly reduces the likelihood that the change will affect code elsewhere in the model.

To summarize, a key feature of OOD is encapsulation (Gosling and McGilton, 1996), which is the ability and encouragement to hide the internal workings of the object from the outside. The integrity of an object's state can be maintained by placing that responsibility on methods within the object. Encapsulation also allows an object to have defined behavior while hiding the implementation of that behavior, which allows it to be modified without causing an impact on client objects.

To continue with Gosling and McGilton's (1996) characteristics of an object oriented language, it should also support inheritance and polymorphism. These are best understood with an example, which I will do while explaining the design and implementation of the Java version of CBCPM.

J-CBCPM's Object Oriented Design

Rick Bourdon recognized the need to restructure the Fortran version of CBCPM, and subsequently hired a professional programmer to develop an object oriented design of CBCPM while keeping the capabilities of the existing CBCPM, with an eye toward future uses. The programmer's design objectives were: 1) centrally located model that could be

accessed and run over the Internet to allow source control, access to specific model versions developed by users, and to provide usage statistics; 2) ability to change modules without changing and recompiling source code; and 3) design in an object oriented language for more intuitive programming (Nagel, mimeograph). He designed it with the Java language specifically in mind because it fills these requirements and because Java is platform independent, it comes with a large array of user interface classes, and several IDE are available.

Unified Modeling Language. I am going to begin the design description of the Java CBCPM using the Unified Modeling Language (UML) (Booch et al., 1999) because it is independent of the programming language and appears to have become the standard language for communicating object oriented designs. Nagel's design did not use UML, and I first became aware of UML halfway through my implementation of the design. A search for object oriented modeling languages on the Internet lead me to UML, and then I purchased UML Distilled (Fowler and Scott, 2000). I highly recommend this book as a starting place - within a week of reading it my code writing productivity increased substantially, largely from having built a class model of Nagel's design.

The Design Process. Design does not start at the keyboard or even with a specific programming language in mind. Formal design processes, such as the Rational Unified Process (Jacobsen et al., 1999), have been created to avoid common mistakes made at this phase of development. A design typically begins with the designer interviewing the domain experts. The designer must model the system in a way that allows him to be certain he grasps what is being asked of him, and to identify issues the domain experts are not raising. In the

case of beef cattle simulation models this step may be blurred because the domain experts are often the ones doing the design, although it was not the case with the Java CBCPM.

The UML is not a single tool, but many, and each tool can be used with varying levels of supporting detail and from different viewpoints. Its purpose is for communication and the level of detail should allow the intent of a design to be clear. The balance is between ambiguity from too little detail and lack of clarity from too much detail. I doubt there is a single best way to use UML to design an application and my own experience is too limited to draw heavily on. Fowler and Scott (2000) address the design process at length, and describe how they tend to use the various UML tools as their designs progress.

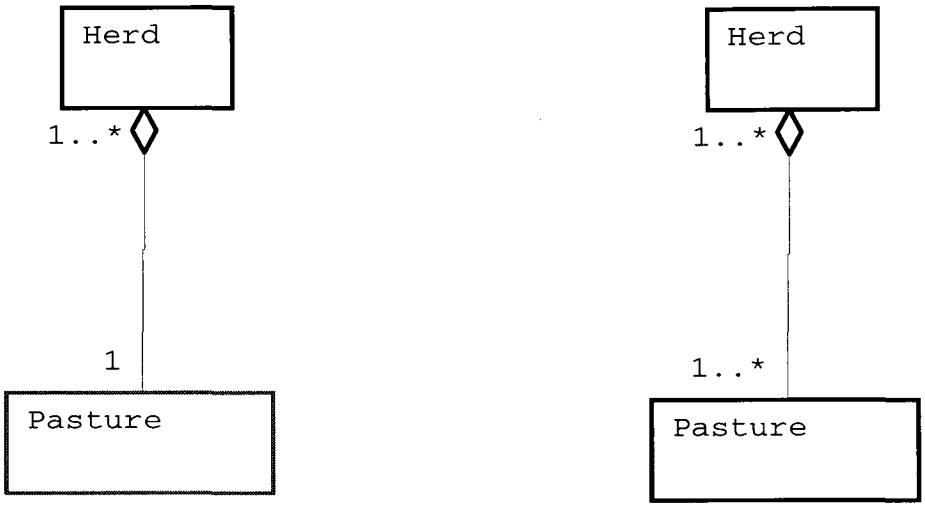
I approached the Java coding with the intent of converting the minimal amount of the Fortran CBCPM necessary to support the reproduction simulation for Chapter 4 while remaining true to Nagel's design. I began by building the skeletons of the classes, followed by the methods to read the parameter files, working from the design and from Fortran CBCPM code and parameter files. My testing and debugging relied primarily on writing data to ASCII files and examining them with a text editor. Somewhere in the midst of creating foundation cows from the foundation herd parameters I learned of UML. At that point I stopped coding and built a UML class diagram from Nagel's design and then corrected some of my classes to conform to what he intended before continuing with new code.

I approached error handling by viewing the model as a non-interactive process, and decided to make most errors be fatal with an accompanying message to aid debugging. This was likely a short sighted approach because the classes may eventually be used in a more interactive way. For example, Bourdon (personal communication) has suggested a

parameterization interface which has background threads testing the validity of the parameters when applied to the model. Java is quite strong in this area, providing the programmer tools for detecting an error or inconsistency and allowing the user options in recovering and continuing onward.

J-CBCPM Implementation.

UML: Class Diagrams. The original CBCPM was designed with one large pasture, large enough that pasture feed was not a constraint on herd size, and cattle were grouped in herds, with all herds in the same pasture (Fig. 3.1a). The rectangles in Figure 3.1 represent objects, and the line between them describes their relationship. The diagram at this point is primarily a communication tool between the domain expert and the programmer/designer, and does not need detailed implementation notations. Therefore, these are conceptual objects, as opposed to classes or instance objects. In Figure 1a, the relationship between Herds and Pastures in the Fortran implementation is enumerated to show that one or more herds (signified by 1..*) can be in one pasture, but that a given herd is in only one pasture. Figure 3.1b illustrates that in the Java implementation one herd can be in one or more pastures and one pasture can contain one or more herds by the symbol 1..* on both sides of the relationship.



Figures 3.1a and 3.1b. Unified modeling language class diagrams of the relationship between herd and pasture in the Fortran and Java models, respectively.

J-CBCPM: Herds, Pastures, and Ranch. Figure 3.2 shows a more in-depth view of the object oriented design of CBCPM. While both Herd and Pasture are in the design there are several new features. This model says a simulation has a Ranch, which in turn has one or more Herds, and each Herd has many Cows, which are in Pastures. Both a Ranch and a Pasture are described in part by their Location, which includes the Climate, Soil, and Forage resources.

The Ranch object is new to the Java CBCPM. It was added to allow multiple Pastures in addition to multiple Herds. The Ranch is in part described by its soils, climate, and growing forage resources. Most pastures on a ranch are likely to have the same climate, but the forages and(or) soils may vary. Climate was separated out of the Location class so it

would be possible to simulate pastures in different climates (e.g. a western ranch on the plains with high mountain summer pastures).

Rather than describe the location within the Pasture and within the Ranch class, the Location is a separate class. Since the Ranch and Pasture classes each describe a Location, it makes sense to make Location a class so it can be used by both Ranch and Pasture. A change to what describes a Location is made in only one place (within the Location class), minimizing the amount of code that needs to be maintained and reducing code fragmentation and the potential for errors. Before examining the relationship of Location with Pasture and Ranch I'll explain the Location class.

The Location class in Figure 3.2 has Climate, Soil, and Forage classes connected to it. The connection between Location and Climate has a diamond on the Location end of the line, which implies that a Location object is composed, in part, of a Climate object. Since the diamond is filled, it means a specific Climate instance is associated with a specific Location instance.

OO: Inheritance. Returning to the relationship of Location with Pasture and Ranch in Figure 3.2, Pasture is connected to Location with a relationship line which has an open triangular arrow on the Location end of the line. This is the UML way to designate inheritance, one of Gosling and McGilton's (1996) required features of an object oriented language. In this case, a Pasture is said to inherit the behavior and state of Location. A Pasture object does not contain a Location object; the Location becomes an integral part of Pasture, as if Pasture was described with all the Location methods and instance variables within the Pasture class.

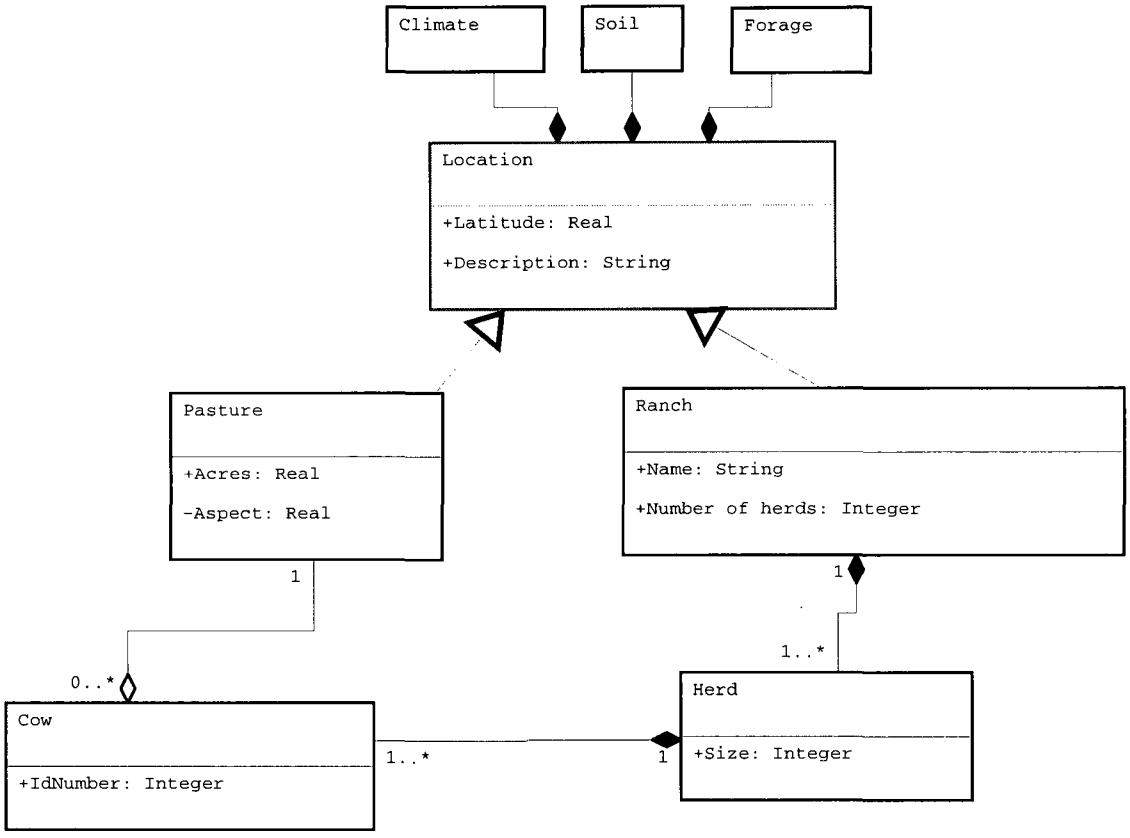


Figure 3.2. Unified modeling language class diagram of how cattle on a simulated ranch are grouped in herds and located in pastures.

If we created an instance of the Pasture class and called that object northPasture, it's latitude (in Java) would be referred to as northPasture.latitude. By designing the Pasture to inherit the state and behavior of Location all the characteristics of Location become an integral part of Pasture. Contrast this to the relationship between Location and Soil. Location would contain a variable called soils although by convention it is not included in the UML class for Location because it is implied. If the Soil class contained a variable for pH, a Location object called plains might address the pH with the statement plains.soil.pH. In this

case the contents of the Soil class, including pH, is not an integral part of Location. Location does not inherit Soil, but rather Location is comprised, in part, of a distinct Soil object.

In addition to Pasture, Ranch also inherits the Location class, illustrating why Location was separated from Pasture. Inheritance allows both Pasture and Ranch to use location parameters as if they were an integral part of the class but the Location is in a module (class) of its own which allows changes to location to be made in a single place in the source code. One of the mental shifts from traditional procedural programming to object oriented design is to be on the lookout for opportunities to generalize classes.

To continue with the explanation of the Java CBCPM design, the Ranch class is potentially associated with more than one Herd. The filled diamond on the relationship line between Ranch and Herd says that a given Herd belongs to one specific Ranch and to delete the Ranch implies the Herd is also deleted.

A Herd has one or more Cows, and an individual Cow belongs to one specific herd. Again, deleting the Herd implies deleting the Cows in the Herd. A Cow is in one Pasture, and a Pasture can have many Cows or no Cows, but also note a Pasture is not an attribute of any specific Cow, as indicated by the open diamond on the relationship line, so a Pasture object can be shared by any number of Cow objects. The text accompanying the design I started with indicated that each Pasture was to be initially created with a copy of the Ranch's Location state, but that these could then be modified with values to describe a Pasture in a different Location

The CBCPM obtains its input parameters from ASCII text files at the start of each simulation. There are parameter files for defining herds, foundation cows within a herd, sires

of foundation cows and sires of future generations, and breeding rules. There are also parameter files to describe the simulated genetic traits, including genetic, temporary environment, and permanent environment covariance matrices.

The design I started with did not address how the starting parameters of the simulation would be handled. I decided to create a Resources class accessible from the Ranch class (Fig. 3.3). The Resources class stores the parameters which are read in from ASCII files at the start of a simulation. It is composed of a number of supporting classes for the various types of starting parameters.

The design also did not indicate how time would be tracked and managed within the simulation, although it did include the definition of how dates would be stored. I created a class which acted as the simulation clock (Fig. 3.3). It is possible to set a calendar date (month, day, and year) as the start date of the simulation. Dates in the simulation input parameter files can then be an offset number of days from the start, the Julian day of the year, or in month, day, and year format. I implemented the model with the intention that the time step was a single day. However, it will be possible to further refine the time by changing to a mixture of time step and event-based model in the future.

I added a Pedigree class (Fig. 3.3) to the design. The pedigree class maintains a unique identifier for each animal in the simulation, along with that animal's sire and dam identifiers. I needed the pedigree to conduct genetic evaluation of the output. It also will make it easier to develop breeding schemes based on relationships (e.g. inbreeding minimization in a closed herd).

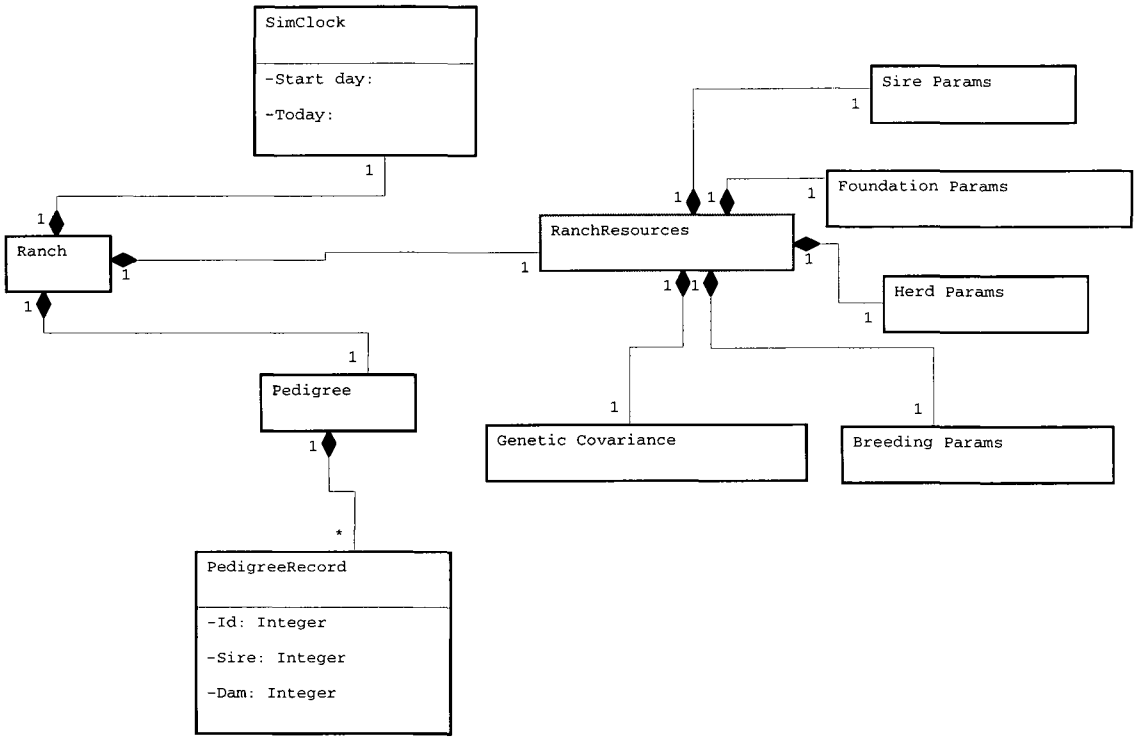


Figure 3.3. Supporting classes added to the J-CBCPM design for input parameters, simulation time, and pedigree management.

UML: Sequence Diagrams. I decided to design the model so that subsequent simulations could be run with the exact same individual sires but with different individual cows. This was a modification of the design to increase the number of offspring per sire to increase the accuracy of predictions for the Chapter 4 analyses. The design is illustrated in a UML sequence diagram in Figure 3.4. To start the simulation a new Simulation object is created. This object creates a new Ranch object, which in turn creates a Pedigree object. At that point the control returns to the Simulation object, but the dashed lines, called lifelines,

below the Ranch and Pedigree objects (Fig. 3.4) indicate they still exist in memory. The Simulation object survives and is active, analogous to being on an instruction stack, the whole simulation, as indicated by the vertical box lifeline beneath it. The Ranch object survives the whole simulation, but at times is inactive, as indicated by the vertical dashed lifeline in place of the box.

Initialization of the ranch begins, with control returning to the Ranch object. Based on a parameter file the appropriate number of new individual Sire objects are created.

In Figure 3.4 the idea that the creation of a sire can be repeated is indicated by the asterisk on the line between the Ranch lifeline box and the Sire object. I have not described the reading of the parameters to avoid cluttering the sequence diagram. The creation of the sires is followed by reading parameter files into the RanchResources object.

Once the parameters have been read, control returns to the Simulation object, which begins iterations (again signified by an asterisk) of simulations. In each iteration the Ranch object creates new Herds and each Herd creates its Foundation Herd as described in the parameters from the RanchResources. A single multi-year simulation is run and the results are written to file. Next, each Herd destroys its associated FoundationHerd, indicated by the **X** and termination of the FoundationHerd lifeline. The Ranch object then destroys that Herd (and therefore all the individual cows and their offspring created in that simulation) in order to reuse the RAM it occupied. These steps are repeated for all Herds. Once the Herds are destroyed control returns to the Simulation object and the next iteration of multi-year simulations is begun. At the end of the iterations the pedigree is written to a file and the application terminates.

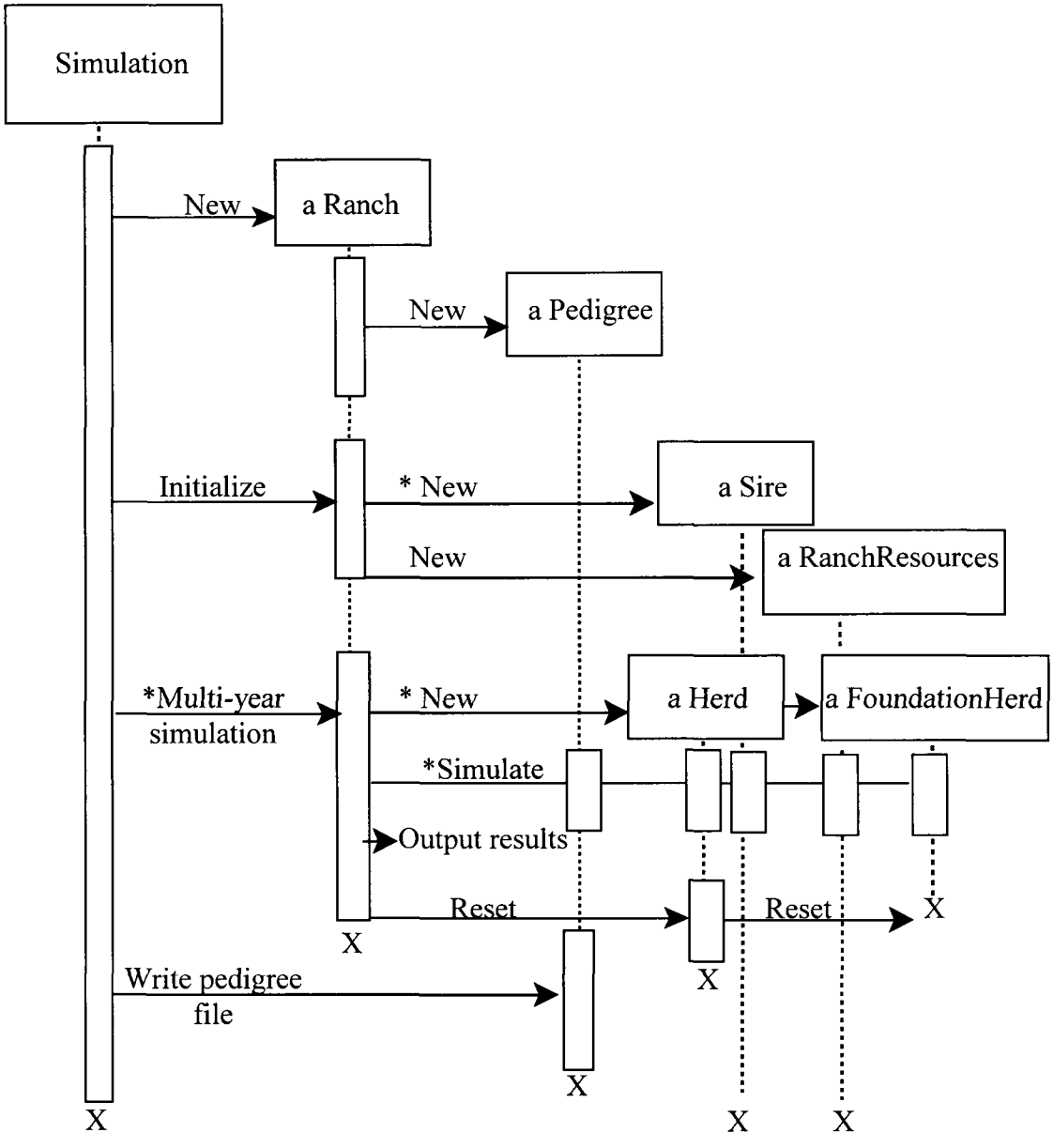


Figure 3.4. Unified modeling language sequence diagram overview of J-CBCPM.

Note that the Ranch, RanchResources, Pedigree, and all the Sire objects are not destroyed at the end of an iteration. This sequence diagram allows the idea that they are used by each multi-year simulation to be illustrated, and that new foundation herds and herds are created for each multi-year simulation.

A UML sequence diagram illustrates a complex process. You will find many well written applications have numerous, relatively small classes, and that tracing the program flow from class to class soon becomes overwhelming. A sequence diagram is a useful UML tool for coping with this complexity.

J-CBCPM: Driver. In the Fortran CBCPM, the subroutine named driver contained the sequence of events which occurred each time step (Table 3.1). The Java CBCPM functions in much the same way but with some notable exceptions. The Fortran CBCPM simulated "super" bulls, capable of breeding any number of cows day after day. The Java reproduction design was changed to allow male fertility to be simulated. I designed it so the number of cows in estrus can be determined before beginning to assign bulls to cows (in a multi-sire pasture case), and before determining conception. Since the design allows cows from different herds to be in the same pasture it is necessary to count cows in estrus in each herd in the case of a natural service bull in a pasture with other bulls. Therefore it was not possible to simulate a whole days' events for one herd at a time. Instead, I made lists of all cows in each pasture so that all animals in a given pasture were processed for breeding at once, regardless of which herd they were in.

A similar case occurred from having multiple pastures and allowing them to contain animals from different herds, complicating the forage utilization. If animals were simulated

each day in sequence of the herd they were in, animals in herds simulated first could potentially consume all the available forage within a pasture, leaving none for other animals in the same pasture simply because their herd was processed later.

Table 3.1 Fortran CBCPM driver subroutine steps (Bourdon, 1992).

	*Read input data
	*Generate foundation herds
	Timestep loop
	Group loop (cows before calves)
	Nutrition loop
	Determine nutrient requirements by individual animal
	Determine intake limits
	Determine ration, including forage
	Divide nutrients among physiological functions
	Remove and grow forage (SPUR)
*	Breed
*	Determine fertility
*	Conceive fetuses
	Determine death loss
*	Grow animals
*	Calve
	Graft calves
	Castrate
	Cull from the breeding herd
*	Determine replacements
*	Dispose of non-breeders
	Import animals
*	Summarize results
*	Output results
*	Update variables
	Clean out rows for animals that are gone
	Stop

An * indicates I implemented the subroutine in the Java CBCPM

OO: Java Interfaces. I used the Java interface Enumeration to construct these lists of all animals within a pasture. A Java interface is essentially a template for a class, specifying how the class should be constructed with respect to instance variables and methods. It is not

possible to create an object directly from an interface. Instead, a class is created by implementing one or more interfaces. Interfaces are used to enforce consistency among all classes which implement a given interface.

For example, the Java Enumeration interface says a class which implements it must have a method defined as **boolean hasMoreElements();** and another defined as **Object nextElement() ;**. Any class that implements this interface must have these two methods, and the return value from these methods must match the definition in the interface. An Enumeration is used to allow access to a series of elements. As a minimum, it is possible to test for elements remaining in the series and to obtain the next element if one does exist. Since all objects in Java inherit from the Object class, the method **nextElement()** can return any type of object. How a series is constructed and accessed is not important and therefore hidden from the method which uses a class that implements Enumeration. In my example, I simply wanted all Cows in a pasture, but do not care what order they are accessed and do not care if they came from a linked list, stack, Vector, etc.

One example of where the Java CBCPM design could be strengthened by defining an interface is the growth model. The interface would say, in effect, that future developers must at least provide a certain subset of methods. This would allow other objects to expect those methods, regardless of which growth model used. It would enhance the consistency among different versions of the model by defining a consistent naming convention.

J-CBCPM: Management Groups.

OO: Generalization. The Java CBCPM design relies heavily on generalization. In particular, the management groups (i.e. Foundation, Sire, Import, and Transfer groups) from

the Fortran CBCPM were found to have common components. In addition, the groups were described by similar information used to describe an individual animal. For example, the parameters used to create a group of foundation cows include variables such as day of gestation, last calving date, breed composition, and genetic potentials. The class Bovine (Fig. 3.5) was created to describe both a type of animal and to describe an individual animal. The Bovine class inherits from the Sex class, which contains methods used in multiple parts of the design. The management groups each require a group size and parental breed combinations, so the Bovine Group class holds these state variables. A composition connection is shown from Breed Composition class to both the Bovine Group and Bovine class, since the parental breed compositions are needed to describe the group while the individual's breed composition exists in Bovine. The Bovine Group class inherits Bovine (and therefore Sex). Each of the different types of management groups inherit from the Bovine Group class.

The Java CBCPM was intentionally designed to separate management of the herd from the biology of the animal. Part of the rationale was to construct classes describing the biology of the cattle which could be incorporated in other models. It would also be possible to construct a model from these classes which focused more on the management than the biology. An example of this would be a version to eat grass for SPUR simulations with much of the bovine biology removed for sake of simplicity and speed.

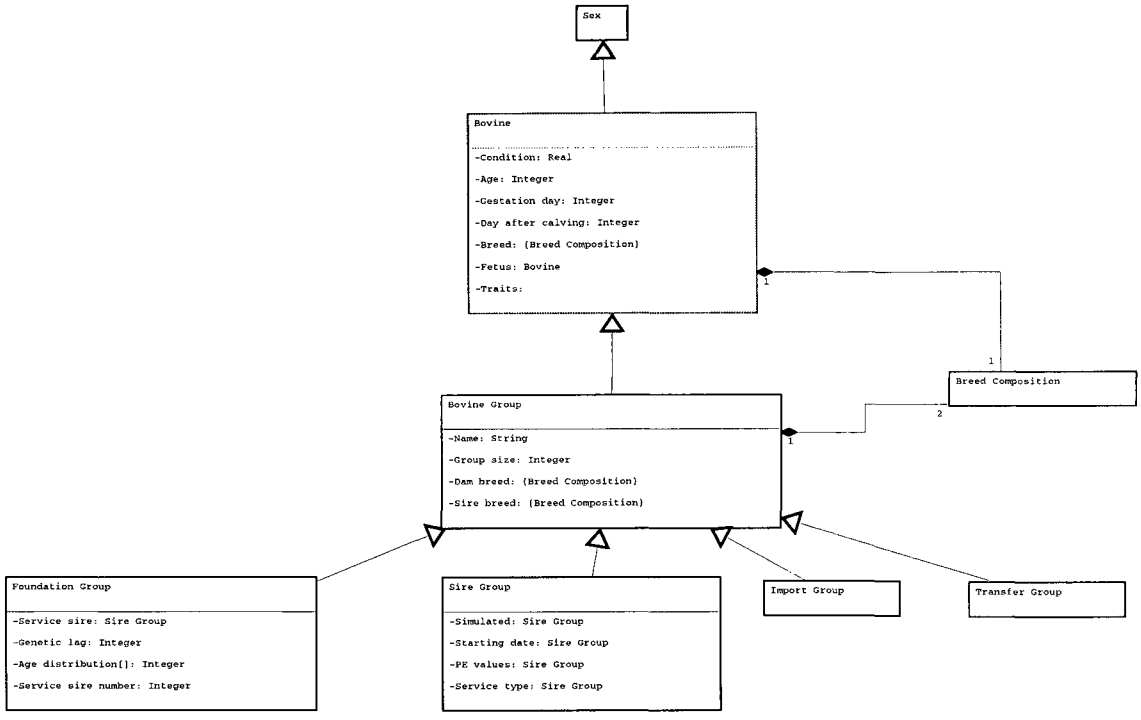


Figure 3.5. Unified modeling language class diagram to illustrate the use of generalization in the J-CBCPM design.

A separate class, CowManager (Fig. 3.6), was created to manage information describing the management of a particular animal. It includes the pasture the animal is in, its breeding season and mating group, its identification number, and variables used within the simulation to signal change of status, such as becoming a replacement, being culled, dying, sold, etc. This is in contrast to the Bovine class, which describes and manages all the biology of the animal, including feed consumption, growth, and reproduction. In Figure 3.6, a Herd is made up of many individual animals, each of which is described by its own instance of both a CowManager and a Bovine class.

J-CBCPM: Fertility.

OO: Polymorphism. The core of the Java CBCPM reproduction simulation is a method which updates the cow's estrous state as shown in the code below. In this code fragment the object's state can change in one of three ways: the cow begins to cycle; the cow stops cycling; or the cow's 21-day cycle is advanced a day.

```
if( isAnestrous() ) {
        if( cyclingStarts() )
                setCycling() ;
}
else
        if( isCycling() )
                if( cyclingStops() )
                        setAnestrous() ;
                else
                        advanceCycle() ;
```

Later in the same time step, a cow in a breeding season is checked for estrus and conception, and her estrous state can potentially change accordingly. To properly model changes in a cow's reproductive state the methods must reflect the cow's current state, maturity, and nutritional state. A UML tool to allow a designer and the domain experts to communicate complex changes in state such as these is the state diagram.

UML: State Diagram. In Figure 3.7, a UML state diagram, the model for the reproduction class begins the state procession with a new-born heifer, starting at the top with the black filled dot. A heifer's reproduction state is first a time of maturing, waiting for puberty. Once puberty is reached she begins to cycle, as indicated by the line from the maturing state to the cycling state. Lines between states are called transitions (Fowler and

Scott, 2000), and can be labeled with a guard statement. The conditions of a guard statement (e.g.. [puberty reached]) must be met before the transition may occur. Processes that occur on a class are either actions or activities. Actions are quick, uninterrupted processes during the state transition, while activities may be longer, interruptible processes. If she is bred (an action) and conceived (another guard statement) her state becomes pregnant and the fetus is created and begins to grow (an activity). However, if she is bred and does not conceive her state remains cycling.

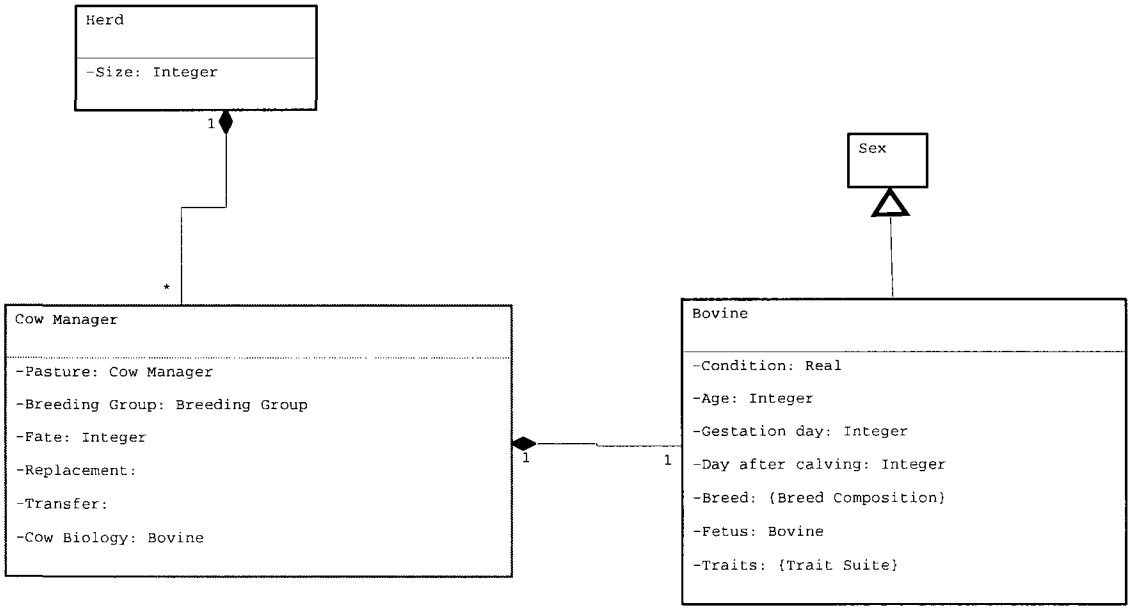


Figure 3.6. Cattle management characteristics managed in a separate class from biological characteristics.

Alternatively, if her body condition changes and drops below a BCS of 4.0, then her state changes to not-cycling. The only state change allowed from not-cycling is back to cycling, and then only once the guard statement is satisfied with her nutritional status.

When a cow object's reproduction state changes from cycling to gestating it stays there until the calf's gestation length is reached and then she moves to recovering. The recovering state in this diagram is the entire postpartum interval. In the Java CBCPM there is a genetic trait for postpartum interval which is an absolute minimum length of days in the recovery state. The actual number of days can exceed the genetic interval due to temporary environment, dystocia, suckling, and nutritional stress. Once fully recovered from calving, the cow's state returns to cycling. I have not shown the states for animals as they are selected, not selected, culled, and sold because I did not view them as being reproductive states, but those states could be added to this model to show the exit point(s).

An alternative state diagram is shown in Figure 3.8. Its primary advantage is to highlight the anestrous and estrous states through use of superstates, but I've also separated the heifer states from those of a cow that has calved at least once. I did this because my primary interest in implementing the Java CBCPM was to model reproduction, and I felt including all the state changes described above in a single class would create a lengthy, confusing collection of methods. Instead, I created four subclasses of the reproduction class, grouping the reproduction states as shown by the solid bold lines in Figure 3.9. The subclasses for these groups of states are named Prepuberty, Open Heifer, Pregnant Cow, and Open Cow, as illustrated in Figure 3.7.

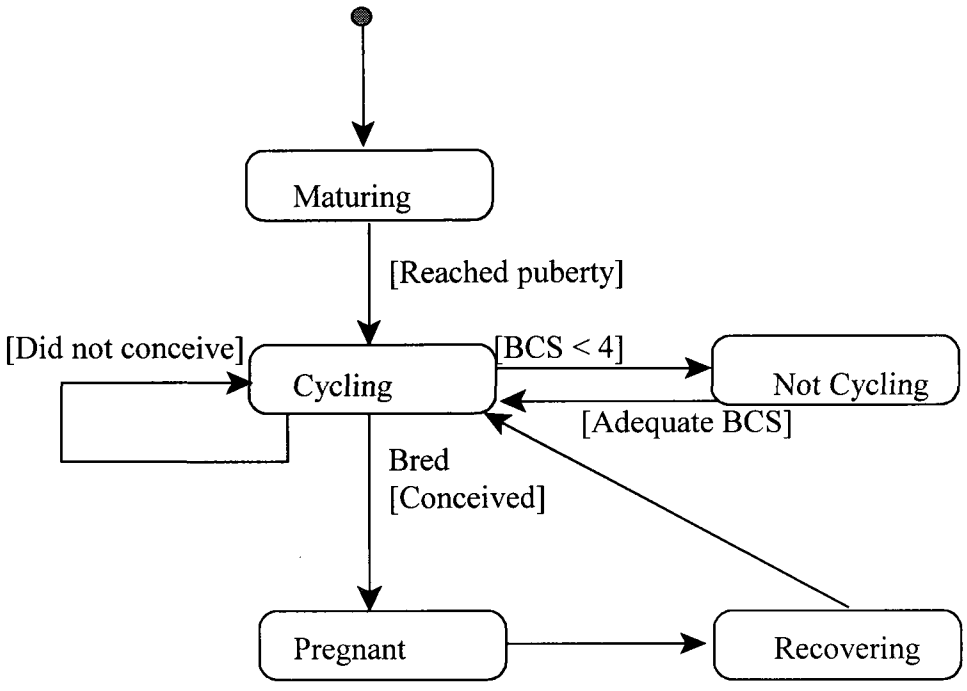


Figure 3.7. Unified modeling language state diagram J-CBCPM reproduction states.

OO: Polymorphism. Figure 3.10, a class diagram, represents the subclasses as generalizations of the reproduction class. Object oriented languages allow subclasses to be used to allow the superclass (i.e. Repro) to assume different forms, and is therefore referred to as polymorphism, another of Gosling and McGilton's required features of an object oriented language. Justifying the advantage of polymorphism requires some explanation and a little mind-bending for a traditional procedural programmer.

Working from Figure 3.10, start with a Bovine object representing a prepuberty heifer. This Bovine object will have a variable of type Repro, called reproduction, to represent the relationship between the Bovine and Repro classes (Fig. 3.6).

This heifer Bovine object's reproduction variable can be initialized by a Java statement like

```
reproduction = new Prepuberty() ;,
```

which has the effect of associating the variable with an object created from the Prepuberty subclass. This means a Java Bovine object named cow could execute the code in the oldEnough() method with a statement like

```
if( cow.reproduction.oldEnough() )
```

to test for puberty. However, a runtime error would be generated for

```
cow.reproduction.underlyingConception();
```

because the Prepuberty subclass does not contain the underlyingConception() method. This illustrates a limitation of subclasses in Figure 3.10; now to illustrate the strength.

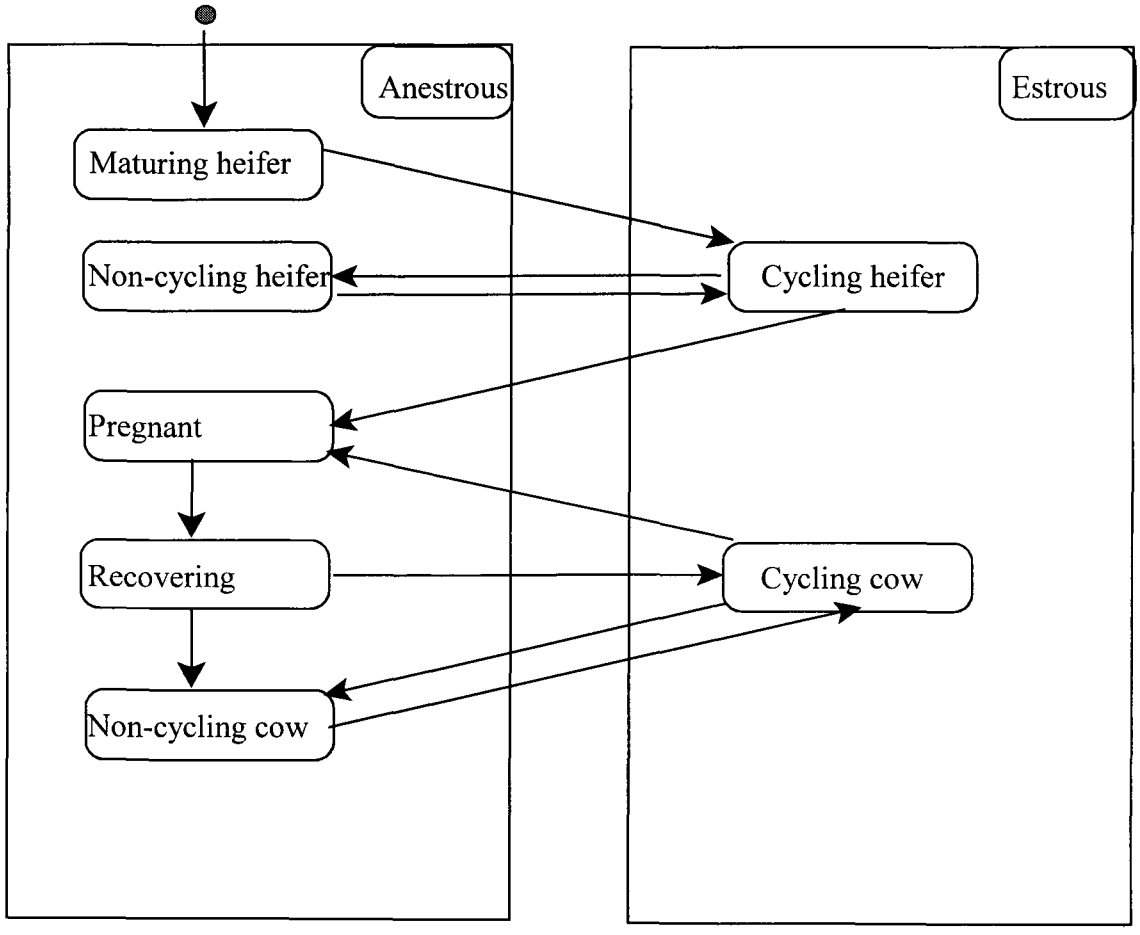


Figure 3.8. Alternative state diagram for reproduction with super states for anestrous and estrous.

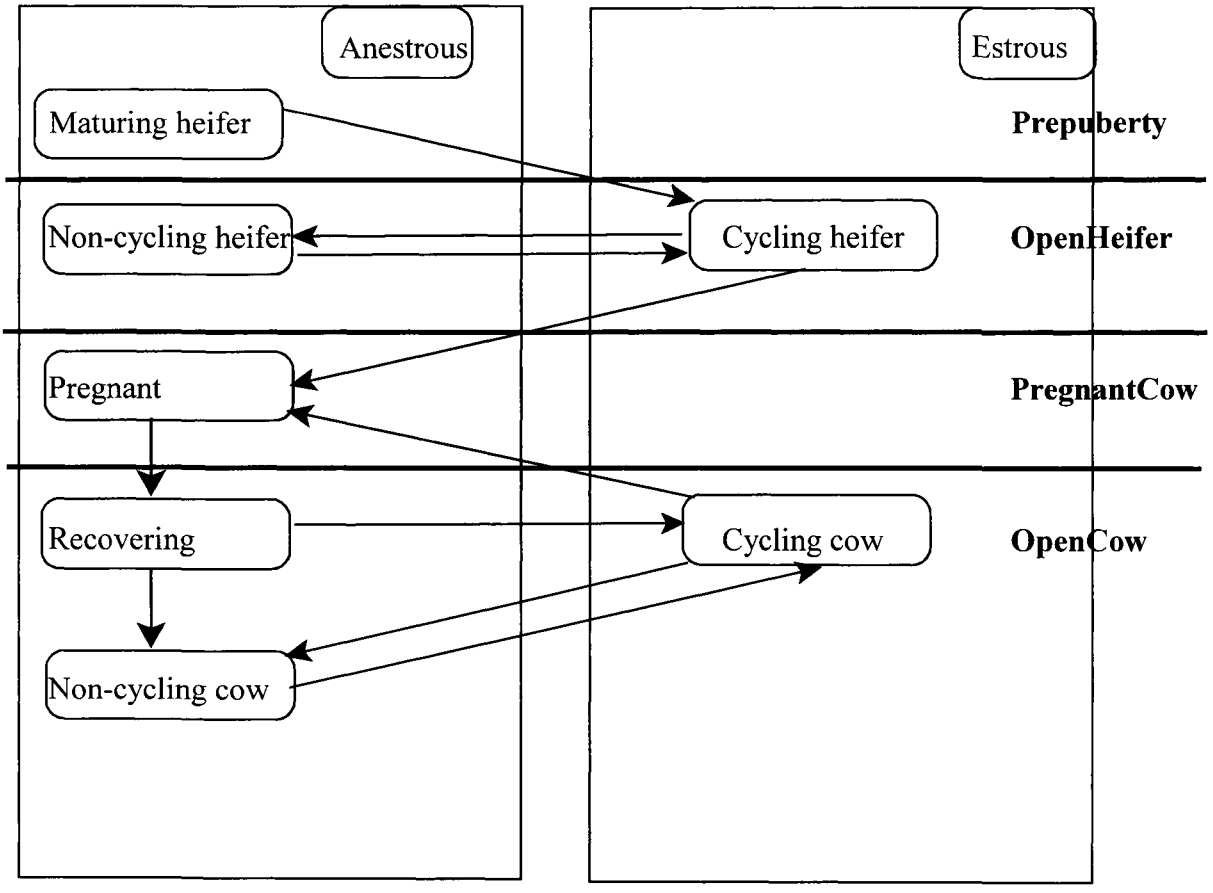


Figure 3.9. State diagram of division of reproduction states into four classes, Prepuberty, OpenHeifer, PregnantCow, and OpenCow, as indicated by the bold horizontal lines.

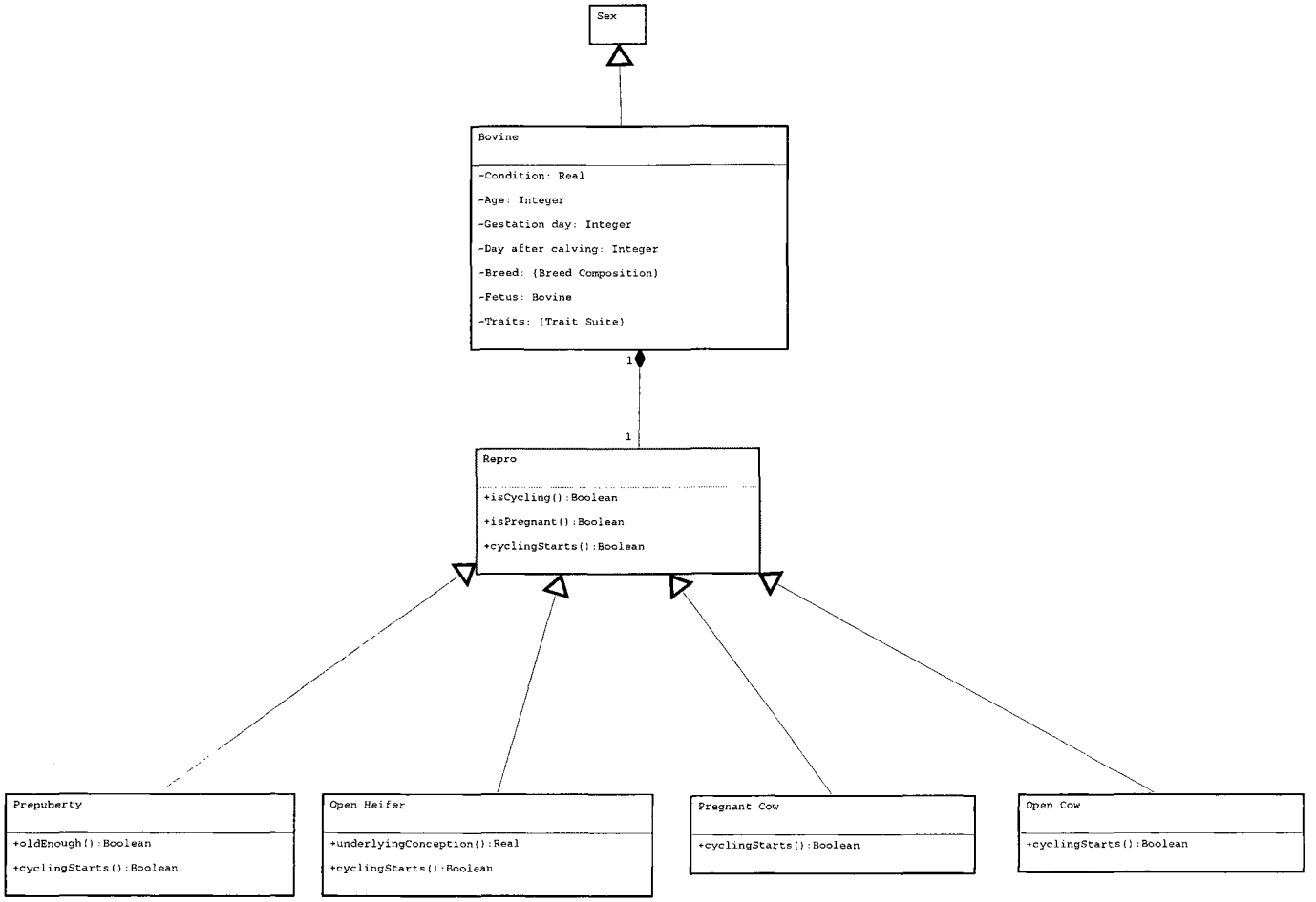


Figure 3.10. Class diagram of reproduction subclasses to illustrate polymorphism.

The super class Repro contains a method cyclingStarts(), as does each of the subclasses. The statement

```
cow.reproduction.cyclingStarts() ;
```

in the example of the pre-pubertal heifer will execute the version of the cyclingStarts() method described within the Prepuberty subclass. However, once the following statement is issued, then the subsequent reference to cyclingStarts() executes the method described in the Open Heifer subclass.

```
Cow.reproduction = new OpenHeifer() ;
cow.reproduction.cyclingStarts() ;
```

In the Java CBCPM the Prepuberty version of cyclingStarts() tests for puberty, while the Open Heifer version of cyclingStarts only tests for adequate nutritional status. This allows the generic reproduction code fragment (shown previously) to be used regardless of which subclass is being used to model the animal's state. The cyclingStarts() method in PregnantCow currently returns a boolean false (although it could be modified to simulate an abortion), and in OpenCow the cyclingStarts() method models postpartum interval and adequate nutritional status.

Note that the superclass Repro also has a cyclingStarts() method, but that it is not executed by the previous statements. The copy of the method in the subclass is said to have *overridden* the method in the superclass, making the subclass version the default method. Different forms of this method exist (hence the name polymorphism), and the form which is accessed depends on which subclass is currently being used.

It is possible to define the version of cyclingStarts() in the superclass such that any

subclass that inherits Repro as a superclass must also implement a version of cyclingStarts(). This is helpful when another developer will be using the superclass with their own implementation of a subclass. For instance, a developer upgrading CBCPM could be required to implement the cyclingStarts() method in any classes that were subclasses of Repro.

Not all methods in a superclass need be implemented in a subclass. The generic code fragment above is a good example. It is in the updateReproState() method of the Repro class. This method is available to all subclasses, as I described earlier when explaining inheritance.

To summarize polymorphism, it allows generic code to be written which accesses different forms of a method depending on the subclass. The use of a superclass allows the designer the option to force subclasses to implement some methods and it allows some methods which are not polymorphic to be written just once. This is all also true for subclass and superclass instance variables. This ability to require implementation of methods is similar to a Java interface.

My use of subclasses to model changing states (Fig. 3.9 and 3.10) of a female's reproductive status adds some complexity to the code. As a Bovine object's state changes from a maturing heifer to a cycling heifer the reproduction variable in the Bovine object has to be changed from using a Prepuberty object to using an OpenHeifer object. This is implemented by a method in each subclass called getNewReproductionState(), called immediately after updateReproState(). Each subclass's version of the method contains the guard statements (Fig. 3.8) to test the updated state to determine if a different subclass is now appropriate. If a new subclass is needed the method returns an object of the needed subclass, and if not, it returns itself.

An additional complication can be seen from Figure 3.10; the navigation between Bovine and Repro is bi-directional. When the Prepuberty subclass determines that puberty has been reached and returns an OpenHeifer object, that OpenHeifer object needs a state variable set to allow it to navigate back to the Bovine object it is associated with.

Despite this, I feel the trade-off in increased clarity of the subclass methods is worthwhile. A modeler wishing to examine or modify the logic behind a heifer reaching puberty needs only to look at the Prepuberty class. Only code related to puberty is in that class, whereas if the design used a single reproduction class it would also include postpartum interval (e.g. dystocia, suckling, presence of a bull, etc.).

I chose to keep all state variables associated with reproduction in the Bovine class, not in the Repro class. This made it easier to change between the reproduction subclasses because it was not necessary to copy the state data from the old subclass to the new, but there are additional reasons for this design, including visibility and ease of later significant modifications to reproduction and growth modules, or submodels, of the CBCPM model.

With respect to visibility, reproduction could have been modeled with a collection of methods within the Bovine class, eliminating any visibility concerns, as all state variables within the class would have automatically been visible. However, this would have created a large, complex class, so I split the reproduction methods into a set of subclasses. Modeling nutrient uptake and use will likewise require a collection of method, and I anticipate them also in a class separate from Bovine for the same reasons. My reproduction simulation requires information about body condition and plane of nutrition, and nutrient uptake is in part a function of reproductive status (e.g. pregnant cows have increased dry matter intake).

Accessing state variables in a "Growth Model" class from a reproduction subclass (Fig. 3.11) involves navigating back through the Bovine class and out to the Growth Model class, likely through use of a method stored in the Bovine class. Storing the reproduction state variables in Bovine makes for easier access from the nutrition class, and storing nutrition and body composition state variables in Bovine will ease their access from the reproduction methods.

With respect to the ease of updating the submodels of CBCPM, particularly reproduction and growth, I felt storing the state variables in Bovine made it more clear to developers of alternative submodels what minimal subset of state variables they needed to maintain to ensure compatibility. In hindsight, I believe the use of Java interfaces will be of greater value in conveying this message.

OO: Java Interfaces Revisited. An interface allows a layer of communication between developers of the Java code. For example, its definition can be provided by one group of developers to another to describe how the first group's classes will be expecting to be able to communicate with the second group's classes. Implementation details beyond what are specified in an interface can be left up to each group of developers.

This would allow the core design of CBCPM to specify a growth model interface. How growth was modeled would be immaterial to the rest of CBCPM as long as the requirements of the interface were met. In theory, an alternative growth model which implemented the interface could simply replace the existing one without any other changes to the code. In practice it rarely is that simple, because the requirements in the simulation are often two-way, and the interface only defines one side of the relationship. For example, my implementation of a reproduction module requires access to the weight gain. A reproduction

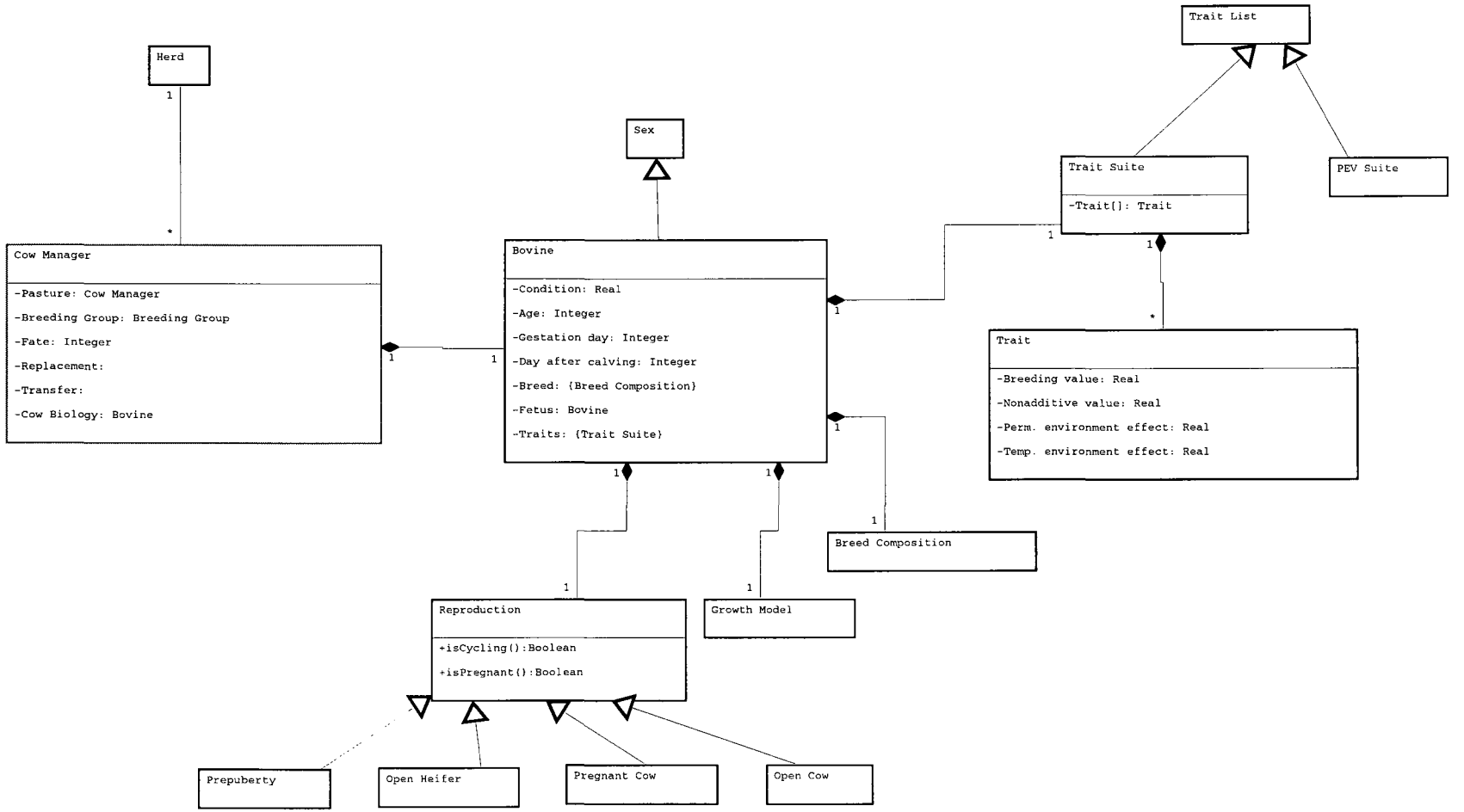


Figure 3.11. Unified modeling language class diagram of relationship between the Bovine class and the reproduction and growth models.

interface would describe what information the reproduction model would make available to the rest of CBCPM, but would not describe what the rest of CBCPM was responsible for supplying to the reproduction model.

The relationship between classes where one classes relies on another to supply a specific service is called a dependency. There are benefits to reducing dependencies. Changes to a class without dependencies can be done with confidence that there will not be side effects elsewhere in the application. Java allows instance variables and methods within a class to be declared as *public*, *private*, or *protected*. A private member of a class can only be accessed from within the class, so it would only have dependencies within the class. A public member can be accessed from any other class, and a protected member can only be accessed from classes in the same package, as described below.

OO: Packages. One way to manage dependencies is to group classes which are related in some way together into *packages*. A package is simply a collection of classes, and an application can be created using classes from any number of packages. The use of packages allows clearer articulation of the dependencies between packages. In Java, for example, a class which resides in one class must explicitly *import* a class which resides in another package. By importing only necessary classes it becomes easier to check for potential side effects, as the number of places to check has been reduced. In addition, only public members of a package can be accessed from another package, further reducing and articulating where dependencies may exist. The protected members can be accessed by classes within the package, but not from classes in other packages.

It is possible to indicate dependencies on a class diagram using a dashed line as the

connector between classes. It is also possible to use the class diagram concept, using packages instead of classes, to communicate dependencies between packages.

OO: Dynamic Binding. I have made reference several times to the possibility of segments of the model being replaced by either updated versions or different versions. Java is different from traditional languages like Fortran and C or C++ in that it is interpreted at runtime, as opposed to precompiled. Memories of the speed difference between interpreted and compiled BASIC came to mind when I learned this. However, improvements in interpreters, and more importantly increases in processor speeds, have made runtime speed nearly as fast as compiled code. For most applications the difference would not even be noticed.

Being interpreted allows Java to use the most recent classes at runtime without having to go through a compile and link process. This is a benefit with distributed users, as it is not necessary to continually provide updated versions of the executable program.

Another Java advantage is that the classes do not need to all reside at the same physical location, as they can be accessed at runtime through the Internet. Developers can easily share classes, and applications given to the users can be updated without having to redistribute the code.

OO: Refactoring. As changes are made to code there is a tendency for the organization and clarity of the code to degrade. Refactoring is a relatively new process used to rewrite the internal structure of the code. The process is to make small, incremental changes to the code without changing its functionality (Fowler and Scott, 2000). The key is to have a method of testing to ensure the functionality has not changed, both in the class(es)

being refactored or through side effects in dependent classes. A change is made, and then all the tests are run to ensure the system has not changed. The ease of comprehensive testing is essential to successful refactoring, and I believe to successful applications.

Examples of places the J-CBCPM could benefit from refactoring are adding interfaces, as suggested above, and renaming classes (e.g. CowManager is also used for males) and variables. Refactoring can also include changes like adding a layer of generalization or splitting a large class into smaller classes. I have noticed some IDE now offer support for refactoring, but I have no experience using them.

I am intrigued by some of the ideas articulated in the process called extreme programming (Beck, 1999) for their potential to minimize programming bugs and maintain simplicity in design and code. Extreme programming involves all stages of development, including the planning, design, coding, and testing of an application. I find the idea of intensive automated testing of all components of the application most interesting. The basic idea is to create the testing code while, or before, creating the production code. Every new piece of production code and all modifications must pass all tests to be accepted. Tools such as JUnit have been developed to automate the testing. One benefit of the extensive testing is there is less fear of refactoring code as needed - if adequate tests were written as the initial code was written then bugs introduced by refactoring should be detected.

Extreme programming was developed for corporate application development; it is meant to increase communication among a large team of programmers developing complex applications for a group of clients. Some aspects of extreme programming may be over-kill for a simulation model, like code being written by pairs of programmers at a single terminal.

Summary

Mechanistic systems models allow the study of interactions among components of the system and allow ideas to be tested in a range of environments. However, as these models become more detailed they become increasingly complex and difficult to manage. The field of computer science has provided tools for managing complexity, including object oriented design and languages. There are also languages like UML for communicating among developers and with clients. There are more UML tools than I have described here, most notably "use case" models. These tools (i.e. computer hardware, OO design, UML, and programming languages) have evolved rapidly, and many of the assumptions with respect to computers and software engineering that animal scientists were trained in are now obsolete.

Systems models in animal science have not lived up to their potential. In some fields of study models are being used to predict phenomenon, and research is then carried out to validate the prediction. Modeling can highlight where knowledge of a subject area is lacking. One test of the progress of beef modeling may be the time when research funding is contingent on suggestive modeling results.

Chapter 4

Analysis of simulated heifer pregnancy data

Introduction

It is important to a beef producer for economic and management reasons to have heifers that conceive at 15 to 18 mo age. Heifers that are managed to calve before the mature cow herd can be observed more closely for calving difficulty and they will have a longer postpartum period to recover from calving. Dystocia occurs most frequently in heifers, making the longer postpartum period desirable. The additional time is also beneficial because heifers are still growing while raising a calf. The heifer will have more difficulty returning to adequate body condition at the start of the breeding season than an older cow would. Lesmeister et al. (1973) demonstrated that heifers that calved early had a greater average annual production than heifers that conceived later in the breeding season. In addition, a producer can compensate for the likelihood of a heifer weaning a lighter calf by allowing for the calf to be older at weaning, resulting in a more uniform calf crop.

However, there will be heifers that do not conceive during the breeding season. These heifers are usually culled as a way to select for fertility and to avoid her maintenance cost while waiting for next year's breeding season. Knowing some heifers will likely not conceive, producers select and breed more heifers than they need as replacements. While some producers are able to sell bred heifers, an extra heifer is more often a liability due to the cost of development; the cost of developing a heifer can approach twice the annual cow cost.

Another consideration is that matings to produce replacement heifers reduces the number of cows available to mate in terminal systems.

In herds where heifer conception is very high, it may be optimal for reasons of economics and/or management to manage for lower heifer conception rates. Single service conception rarely, if ever, reaches 100 %, with rates near 70 % more typical. Allowing heifers repeated chances at breeding increases the overall pregnancy rate but also extends the breeding and calving season. An extended calving season results in greater diversity in cow and calf management and calf weight at sale time. There are a diminishing number of additional animals that become pregnant as the breeding season extends. At some point, the cost of additional management needed for cows and calves at different stages of production cancels out the return from getting additional animals bred by further extending the breeding season. Having all heifers conceive in a breeding season is likely an indicator of too long a breeding season or too many resources being spent on heifer development.

A heifer's ability to become pregnant is influenced by many factors including the ability to start cycling before or during the breeding season, conceive, and maintain the pregnancy. As discussed in the Chapter 2, puberty is influenced by age and weight, with genetic differences between and within breeds. A number of genetic predictions for fertility traits have been produced. Recently, threshold model techniques have been applied to observations of pregnancy to produce an EPD for heifer pregnancy. The ability to become pregnant is likely the outcome of several underlying traits, particularly puberty and ability to conceive. These traits are difficult and expensive to measure accurately. Simulation is a relatively easy first step in exploring the effect of different levels of the underlying traits on

the heifer pregnancy EPD.

My objective was to determine the effect of genetic potential for fertility traits and management-constrained length of the breeding season on genetic predictions for HP. Specifically, I wanted to determine if important genotype by environment interactions (management differences) might influence genetic predictions of HP with respect to fertility, modeled as age at puberty and probability of conception. Second, I wanted to determine the influence of pregnancy rate, as controlled by breeding season length and influenced by fertility, on accuracy of subsequent genetic evaluation.

Literature Review

Threshold Trait Theory. Traits such as heifer conception or pregnancy are called categorical traits, as the phenotypes are observed in categories (e.g. success or failure). One genetic theory to explain ordered categorical traits is that while we observe a trait as having discrete categories, there is an underlying, unobserved continuum of effects, both genetic and environmental, that cause the phenotype observed (Falconer, 1989). On that underlying continuum there is a threshold between each category, and the observed category indicates where the individual animal's phenotype is with respect to the threshold(s). A binary trait, such as heifer pregnancy, has one threshold; animals whose sum of underlying effects exceeds this threshold are able to become pregnant, while the others remain open.

Linear Models. Prior to the availability of alternative statistical models, categorical traits were analyzed using linear models by assigning a numerical value to each category and analyzing the recoded data as if it were continuous data. In the example of heifer pregnancy,

it is possible to assign a value of one to pregnant heifers and a value of zero to open heifers, and analyze that data with a linear model. A heritability estimate obtained from analysis of categorical data with a linear model is referred to as a heritability on the observed scale ($\mathbf{h^2_o}$).

Threshold models make use of the assumption that the underlying effects have a normal distribution, and since this distribution is unobserved, the unit of measure used is the standard deviation. Lush et al., (1948), showed that the heritability on the underlying scale ($\mathbf{h^2_u}$) for a binary trait can be obtained from h^2_o by

$$h_u^2 = \frac{h_o^2 \, p(1-p)}{z^2}$$

where **p** is the probability of observation in the first category (e.g. not pregnant) and **z** is the height of the standard normal density function at the threshold corresponding to **p**. As the observed mean genotype approaches either end of the probability scale the environmental variance becomes nearly linearly associated with the population mean (Dempster and Lerner, 1950). Therefore, this transformation is necessary to properly compare $\mathbf{h^2_o}$ estimates obtained from populations with different probabilities of observation in the first category (Koots et al., 1994).

A number of alternative methods to recode categories into numerical values have been studied (Kaiser, 1996). Still, while it is possible to obtain variance component estimates and calculate EPD for categorical traits using linear models, many disadvantages remain. Gianola (1980) listed seven problems: 1) scores are arbitrarily assigned to response categories; 2) mixed model solutions do not incorporate the restrictions in the estimation space that the sum

of response probabilities must total one across categories; 3) the variance in the observed scale is not constant and depends on the genotypic value of the candidates for selection; 4) the additive genetic variance in the observed scale depends on the mean incidence of the character in the subpopulations considered in the model; 5) nonadditive genetic variation is present in the observed scale; 6) linear relationships fail outside a restricted range of data; and 7) ranking optimality of best linear predictors is lacking when the conditional expectation of the predictand given the data is not linear. Studies suggest that categorical trait analyses using linear models are at best the same as using nonlinear models, but that nonlinear models can provide better results than linear ones (Snelling, 1994).

Non-Linear Models. Procedures for *maximum a posteriori* (MAP) probit threshold models were developed independently by Foulley and Gianola (1984) and Harville and Mee (1984). Start with the model

$$y = X\beta + Zu + e$$

where

$$\text{var}\begin{bmatrix} u \\ e \end{bmatrix} = \begin{bmatrix} A\sigma_a^2 & 0 \\ 0 & I \end{bmatrix}$$

and **y** is a vector of subclass observations on the underlying scale. **X** and **Z** are known incidence matrices relating subclasses to fixed and random effects. **β** and **u** are vectors of fixed and random effects, respectively, of length **p** and **q**, respectively. **A** is the matrix of additive relationships among animals in the analysis, **I** is an identity matrix, and σ^2_a is the

known additive genetic variance on the underlying scale. The residual variance on the underlying scale is one. Estimates of β and u can be obtained with nonlinear mixed model equations using Newton-Raphson (Foulley and Gianola, 1984) or Fischer scoring (Kaiser, 1996) iterative methods.

Variance component estimation. Genetic variance can be estimated with marginal maximum likelihood (MML) (Harville and Mee, 1984; Hoeschele et al., 1987), and was used to analyze stayability (Snelling, 1994).

Reverter et al. (1994) developed a method of variance component estimation which they called Method R. It is based on the covariance between more accurate and less accurate predictions equaling the variance of the less accurate predictions, so in an unbiased analysis the regression of more accurate predictions on less accurate predictions equals one. They made use of this property by using the whole data set to be analyzed to represent the more accurate predictions, and a subsample of the whole data set to represent less accurate predictions. In a single trait analysis, the regression

$$\frac{u_w' A^{-1} u_p}{u'_p A^{-1} u_p}$$

of more accurate (u'_w) predictions on less accurate predictions (u'_p) is used to search for a heritability which yields a regression of one, indicating an unbiased analysis. Repeated estimations of heritability from the whole data set using different subsamples can describe the distribution of the variance components (Mallinkrodt, 1996).

Snelling et al. (1995) used Method-R with predictions on the underlying scale from MAP analyses to estimate heritability of stayability; applying Method R to MAP predictions has been called MAP-R. They estimated heritability multiple times from the same whole data set using different subsamples and reported the average heritability estimate.

Reverter et al. (1994) extended Method-R to multiple trait, multiple-component linear models. Kaiser (1996) further extended it and MAP-R to provide predictions from multiple trait, multiple-component models where one or more of the traits is analyzed with a nonlinear (MAP) model. He implemented this in software starting from the ABTK tool ds6, developed by Golden (unpublished), which was initially capable of multiple-component , multiple trait Method-R. Kaiser's addition to the ABTK was called dscat.

Traits used to select on puberty.

Scrotal circumference. As discussed in Chapter 2, timing of puberty is an important reproduction trait but is difficult to measure. As a result, most efforts have been in the area of finding correlated traits which can be measured easily and inexpensively. Scrotal circumference has been shown to be genetically correlated to puberty (Coulter and Foote, 1979; Bourdon and Brinks, 1986; Lunstra, 1988; Morris et al., 2000). It has the advantage of being easy and inexpensive to measure. It has the disadvantage of being sex limited and requires measurement of intact bulls at approximately a year of age. Since only a few bulls are needed as replacements and because the USDA meat grading system discourages production of bulls for slaughter, most males are castrated at or before weaning. Males left intact have not been randomly selected, so it is difficult to obtain measurements from all males, and bulls kept as replacements are not randomly selected.

There is additional evidence to question the use of scrotal circumference EPD. Evans et al. (1999) analyzed scrotal circumference and heifer pregnancy. Calculations of genetic correlation were close to zero. Analysis using genetic group techniques showed a non-linear relationship between heifer pregnancy and scrotal circumference.

Reproductive Tract Score. Another heifer fertility trait which has been investigated is reproductive tract score (RTS) (Andersen et al., 1990). Yearling heifers' ovaries and uterus are palpated trans-rectally and a score from 1 to 5 is assigned to rank development of the reproductive tract. While not strictly a measure of timing of puberty, RTS has been shown to be associated with the ability of a heifer to conceive and has been shown to be heritable (Andersen et al., 1990). However, a 1994 National Animal Health Monitoring System study showed use of RTS by 1.2 % of producers surveyed; they did not include a RTS question in more recent surveys.

Heifer Conception. Heifer conception can be reported as a single-service conception rate or conception rate following a certain length breeding season. Heritability estimates of conception rate are often based on palpation or pregnancy data, and are more similar to the heifer pregnancy EPD than they are to the PCON trait.

In addition, the statistical model used to analyze conception data can also effect the results. Early estimates came from linear models, often paternal half sib models. When animal models became more widespread they were used, but typically still as a linear model. The resulting heritability was at times transformed to the theoretical underlying scale, but the categorical data were analyzed on the observed scale. The use of the linear models likely contributed to the low heritability estimates (Evans et al., 1999; Doyle et al., 2000). Recent

estimates using models developed for analyzing categorical data have yielded higher heritability estimates.

There are few reports of the heritability of conception rate because it is difficult to measure. Legates (1954) concluded services per conception in dairy cattle to be largely non genetic. Dearborn (1971) reported conception heritabilities of 0.06, 0.085, and 0.04 for Guernseys, Holsteins, and mixed dairy cows, respectively. Koots et al. (1994) reported conception rate direct genetic effect of cows and heifers as 0.28 and 0.05, respectively. These were weighted means, with 19 cow studies and 7 heifer studies. Also, this was conception rate at the end of the breeding season expressed as percent calved, as opposed to conception rate at a given point in time, independent of breeding season length.

The easiest and least expensive way to measure a heifer's ability to conceive is by visually observing the results - the production of a calf. This is crude, at best, since we can only categorize the ability to produce a calf as yes or no - contrast this to our ability to rank calves by their weaning weight across a continuum. In fact, a given heifer's successful pregnancy by itself tells essentially nothing about her genetic potential. However, by using numerous observations of the performance of sires' daughters' reproduction we can begin to separate the sires based on their genetic potential.

Snelling et al. (1996) estimated heritability for heifer pregnancy for Line 1 Herefords and for a population of linecross Hereford cattle that were randomly mated and selected from 1976 until 1988. Yearling and two-year-old pregnancies were analyzed as different traits. In addition, they analyzed the data using both a linear model and a nonlinear marginal maximum likelihood (Hoeschele et al., 1987) procedure. Their data and results (Table 4.1) show the

linear model method yielded estimates much lower than the nonlinear model method. Attempting a selection program based on the linear heritability estimates would be discouraging, while the nonlinear heritability estimates suggest substantial progress could be made.

Evans (1996) analyzed 986 heifer pregnancy records from Herefords bred to calve as two yr olds. Heifer pregnancy was observed by rectal palpation 120 d after the start of the breeding season. Breeding season length was targeted to 63 d but varied from year to year. Only heifers given the opportunity to breed were given an observation; observations were binary coded for either success or failure. The data were from 11 y on a single ranch. The average pregnancy rate across years was 78 %.

Table 4.1. Data description and heritability estimates of pregnancy for two Hereford populations obtained with linear and nonlinear models (Snelling et al., 1996).

	Linecross Herefords		Line 1 Herefords	
	Records	% pregnant	Records	% pregnant
Yearling pregnancy	679	86.3	765	79.9
2-year-old pregnancy	504	79.8	600	77.3
		Heritability estimates		
	Nonlinear	Linear	Nonlinear	Linear
Yearling pregnancy	.21	<.001	.30	.02
2-year-old pregnancy	.17	<.001	.49	.006

Heritability of heifer pregnancy in these data was reported as .14 (Evans et al., 1999). Heifers from 2 y old dams were 10 % less likely to conceive in the breeding season. Heifers

born earlier were more likely to conceive; every 20 d increase in age resulted in a 10 % increase in conception.

Doyle (2000) analyzed heifer pregnancy in Angus, using 1,299 records of heifer pregnancy collected across nine consecutive years. Heifers were synchronized and bred by AI, and were then exposed to a clean-up bull, although the breeding season length was not indicated. Pregnancy was observed at approximately 120 d following the start of the breeding season by rectal palpation, with an average 89.2 % pregnant. Heritability estimates were obtained using MAP-R. Fixed effects included age of dam, a contemporary group that included birth year and cleanup sire, and a covariate of age. They reported average and median h^2 estimates of .21 and .20, respectively. The fixed effect for age of dam was significant, with heifers from younger dams less likely to become pregnant than heifers from mature dams. The age covariate was not significant; this may be in part due to early puberty. They reported that reproductive tract scores taken one month prior to breeding indicated that most heifers were or would be cycling by the start of the breeding season, which suggests that puberty was not a limiting factor in conception.

Golden et al. (2000) developed a heifer pregnancy EPD for the Red Angus Association of America. Heifer pregnancy was observed for all heifers bred to calve as two-year-olds with a breeding season less than 90 d in length. They used 10,310 records of pregnancy status, with an 83 % overall pregnancy rate, to estimate heritability and obtain heifer pregnancy EPD. From these data they estimated heritability to be 0.27. The genetic trend from the analysis showed a 3 % decrease in heifer pregnancy from 1989 through 1998, with the decrease happening at a relatively even pace over this time. They hypothesized that

the decrease was due to unfavorable correlated response to selection for increased yearling weight.

Materials and Methods

Overview. The simulation model described in Chapter 2 was used to generate heifer pregnancy data for three levels of age at puberty and 3 levels of probability of conception, yielding nine different data sets. The simulation model was parameterized for a 120 d breeding season. The model output results of the breeding season in terms of heifer identification number and the Julian day of the year if she conceived, or heifer identification number and a zero if she remained open at the end of the breeding season. Each of the nine data sets were further divided in retrospect by truncating the breeding season length at 25, 45, 60, and 90 d, resulting in five breeding seasons (including the original 120 d season), for a total of 45 data sets. These data sets were individually analyzed to obtain heifer pregnancy variance components, EPD, and the EPD accuracies.

Simulation Model. The data were generated using the object oriented version of CBCPM, with the changes as described in Chapter 2. The conversion of CBCPM from Fortran to Java was not complete, with components of the model not available for use in this simulation. Most notably, the model lacked any variation in nutrition and growth; each animal's weight was a deterministic function of its age through use of a Brody (1949) growth curve.

The scenarios simulated were not realistic in that the reproduction components lacked input from environmental effects. This was an advantage, in a way, in that it was

possible to construct theoretical scenarios lacking noise and complexity that would be present in the full model. Although the model may seem over simplified, had many of these effects been simulated I would have subsequently attempted to remove their effect in the variance component and EPD analyses.

Non-Reproduction Simulation Parameters. Parameters of interest but not directly related to fertility were growth rate, age of heifer and age of dam, and the structure of the foundation herd. Daily growth was determined with a Brody (1949) curve, with parameters **A**, **b**, and **k** fixed at 454.0, 0.93, and 0.003, respectively, for all animals. There was no variation in parameters, and growth was strictly a function of age. Similarly, there was no variation in body condition, with all animals assumed to be in good condition (6.0 BCS).

Heifer age was a significant fixed effect in data analyzed by Evans et al. (1999), with a 10 % increase in probability of conception for every 20 d increase in age. Heifer age at the start of the breeding season in a beef cattle herd is a function of many factors including the current breeding season's start date and the start date of the breeding season the heifers were conceived in. In addition, it is possible that some heifers had heifer dams who were bred earlier than older dams. The distribution of calving dates is also a function of gestation length, dam and sire fertility, and breeding season length.

There are several ways to account for these effects. Azzam et al. (1990) used Monte Carlo simulation to find an appropriate dam age (parity) structure before starting a simulation. An alternative is to run the simulation model additional years and then discard

results of the initial "burn-in" years. This allows the distributions to form based on the input biological and management parameters rather than assumed distributions.

I chose to have all heifers be born on the same day largely for the sake of clarity. If I had simulated an age distribution I would have then tried to remove its effect by fitting age in the variance component and BLUP analyses, as have previous analyses of heifer pregnancy data (Evans et al., 1999; Snelling et al., 1996; Golden et al., 2000). There is the potential for confounding age with fertility, as more fertile dams will conceive earlier and therefore have older heifers. This potential confounding is unavoidable in field data, but easily avoided in simulation, and may provide better understanding of the degree of bias in field data.

A secondary reason for not fitting an age distribution was selecting the appropriate breeding season length for the season these heifers were conceived. This was a problem because the simulated data were used to form heifer pregnancy observations for five different breeding season lengths (described below). Basing a heifer birth date distribution on any one of the five breeding season lengths would have been arbitrary. It would have been possible to run individual simulations for each breeding season length, but one of the objectives was related to truncating field data to the ideal breeding season length.

The heifers used to obtain heifer pregnancy observations were simulated as fetuses of foundation cows on day one (January 1st) of the simulation. To obtain heifers that were all the same age as described above, all foundation cows were assumed to have conceived on the same day and all foundation cows had a 285 d gestation. Their fetuses were 204 d old on day one of the simulation, so the heifers were born on day 82.

Starting date of the breeding season was day 140, and it was assumed fixed across years, including the year these heifers were conceived. Given the above assumptions of fetus age, this implies the foundation cows conceived these heifers on the 22 day of the breeding season. The date the heifers were conceived effects their age at the start of their breeding season. The heifers were bred the following year to calve at 2 y age, so all heifers were 423 d old at the start of their breeding season.

Foundation cows in the Fortran version of CBCPM were the cow herd that existed before simulating the first day. These cows were unrelated, with unknown sire and dam, and their breeding potentials were simulated with full genetic variation accordingly. All foundation cows carry a fetus; presumably open cows were previously culled. Preliminary variance component analyses indicated lack of power (heritability estimates did not converage), likely due to some form of confounding of the data. In an attempt to increase the power of the data, the foundation cows were simulated with known sire and dam. The sire was drawn at random from the sire group described above. The dam of a foundation cow was simulated as a cow with unknown sire and dam, as above, but she was assumed dead, and was not simulated on a daily basis once the herd was built. She was added to the pedigree, but no other information about her was stored once she had been used to create foundation cows. These dams of foundation cows were used to each create three half-sib foundation cows (foundation cows with common dam but potentially different sires). Using dams of foundation cows allowed genetic ties between heifers. One trade-off of allowing foundation cows to be related was some heifers were potentially inbred.

Foundation cows were all simulated as 2 y old. Snelling et al. (1996) and Evans et al. (1999) reported age of dam fixed effects on heifer pregnancy to be significant, while Doyle et al. (2000) reported inconsistent age of dam fixed effect solutions. In this study no age of dam effects were simulated either directly or indirectly due to the limited capabilities of the model. All foundation cows carried a heifer fetus to make the simulation more efficient at producing heifer pregnancy observations.

Foundation sires were initially simulated as unrelated, with unknown sire and dam, and their breeding potentials were simulated with full additive direct genetic variation accordingly. However, I was still experiencing difficulty in getting the parameter estimation models to converge, so I modified the foundation sires in J-CBCPM similarly to the foundation cows. A set of 500 foundation sires was created by first creating one sire of a foundation sire and then five foundation sires were created as his offspring, and then repeating with a new sire of a foundation sire until all 500 were created. The sires of foundation sires were added to the pedigree, but their genetic potentials were not saved. This allowed foundation sires to be related, increasing the power of the pedigree. Sires of the foundation sires were created using a separate but identical sire group in the parameter file.

In each simulation run there was one herd comprised of two sire groups (sires of foundation sires, and foundation sires) and one foundation cow group. My Java version of CBCPM allowed consecutive simulations using the same sires but different cows. This was necessary to simulate large data sets of related animals. Simulations with a foundation herd of more than 3,000 animals would run out of available memory on the computer used

before running the necessary two years to obtain the heifer pregnancy observations. The code was modified to rerun simulations by removing all animals except sires, call Java's ***System.gc()*** method to force release of memory, and then rebuild the foundation cow herd as described above. This allowed any number of heifers to be simulated, and keeping the same sires allowed the heifers to be related across reruns. There were no climatic parameters or other parameters that were variable across reruns, so all heifers could be treated as contemporaries in the subsequent analyses.

Heifer observations were only used from the first generation of heifers to avoid effects of natural selection caused by low fertility cows not producing a heifer for subsequent generations. All heifers from the foundation cows were selected for breeding. Heifer pregnancy observations were parameterized/simulated as heifer calvings.

The number of sires was varied in preliminary runs to obtain good convergence of variance components. The data used for this study had 500 sires of heifers, and 100 sires of sires and dams. The dams of heifers were mated by randomly drawing one of the 500 sires of heifers, using a uniform distribution. On average, each sire had 40 heifers in the analysis.

Reproduction Parameters. Two fertility traits, AAP and PCON, were simulated as heritable traits, with input parameters for additive direct genetic variance, permanent environment variance, temporary environment variance, and foundation sire and dam group mean breeding potential for each trait. Within a simulation run the sire and the dam input mean breeding potentials were the same. The traits were simulated with zero covariance between traits in the additive direct genetic, permanent environment, and

temporary environment covariance matrices. The variances were held constant across all simulations, in keeping with the theory of genetic potentials (Bourdon, unpublished).

Arije and Wiltbank (1971) estimated AAP phenotypic variation in Angus and Hereford heifers to range from 31 to 50 d sd. Literature estimates of AAP heritability range from .4 to .8 (Martin et al., 1992). I used a phenotypic 30 d sd and a 0.40 heritability to arrive at the variance components in Table 4.2. Permanent environment variance was essentially zero, but it had to be none-zero to allow the Cholesky decomposition to be calculated

Table 4.2. Genetic parameters of simulation model.

	Age at puberty	Probability of conception given cycling
Additive direct genetic variance	360 d^2	.10
Permanent environment variance	0.00001 d^2	.15
Temporary environment variance	540 d^2	.75
Mean	340, 390, and 440 d	.60, .70, and .80

The population mean AAP was set at three different levels (Table 4.2) to simulate early, medium and late age at puberty with respect to the start of the breeding season, which corresponded with the heifers being 423 d old. I selected the mean for early age at puberty, 340 d age, so that most heifers would have cycled at least twice before the start of the breeding season. Recall from Chapter 2 that the simulation decreases conception on the first two estrous cycles following puberty. Due to the limited environmental effects simulated these heifers could be representative of heifers with even earlier age at puberty. They could be thought of as British breeds, such as Angus.

Many of the medium age at puberty animals had started cycling but were effected by reduced conception at the start of the breeding season. They were perhaps representative of continental breeds. Fewer than half the late age at puberty animals had started cycling at the start of the breeding season. These animals could be thought of as *Bos indicus* breeds.

Literature estimates for probability of conception given cycling parameters were more difficult to find. Few experiments have directly measured this trait; conception studies are generally for multiple estrous cycles and puberty status may not be well known. Larsen et al. (1990) concluded "There is little evidence to suggest that a cow of one genotype, returning to estrus before the onset of the breeding season, with normally involuted uterus, bred by a fertile bull on a subsequent estrus will have a higher probability of conceiving than a cow of a different genotype given the same conditions."

The review paper by Koots et al. (1994) calculated a weighted mean heritability of direct conception rate of cows to be .28 based on 19 studies, and heritability of direct conception rate of heifers to be 0.05 based on 7 studies. This was conception rate for the entire breeding season, not just a single estrous cycle, converted from h^2_o to h^2_u. These studies suggest there are heritable effects on the ability to conceive. Heritability estimates of the trait Stayability further support this. The trait conception rate is difficult to use to parameterize the model since it may be confounded with the length of the breeding season.

I implemented general, permanent infertility uncorrelated to genetic effects using a threshold level for permanent environment such that the lowest 5 % of the distribution were infertile (Johnson and Notter, 1987a). Subsequently, I had to increase the PCON

mean breeding potentials to reflect the decrease that the model was accounting for. This was in line with the theory of genetic potentials (Bourdon, 1998); as the simulation accounts (mechanistically) for additional effects, the breeding potential increases.

Data Generation. I simulated data for 20,000 heifers for a single breeding season by rerunning the simulation 10 times with the same sires as described above. Each simulation output a data file of animal identification number, breeding potential, and temporary and permanent environmental variance for each of the two traits. In addition, the Julian day of year conceived was output for pregnant heifers, while open heifers had a zero in that field as a placeholder. Actual age at first estrous was also output in case it was different from the AAP potential due to low body weight. The pedigree, consisting of identification numbers for each animal, its sire, and dam, was written to a separate file, with unknown sire and(or) dam represented with a period as a placeholder.

I simulated heifers for each of the nine combinations of the mean AAP and PCON parameters (Table 4.2) with a 120 d breeding season. Then I created four additional breeding season data sets for each of the nine combinations by truncation of the data at 25, 45, 60, and 90 d from the start of the breeding season. Heifers that conceived within a (truncated) breeding season were considered successes for that breeding season, while heifers not pregnant at the end of a breeding season were considered failures for that breeding season, even if they later conceived before the end of the 120 d season. The heifer pregnancy data was given a binary coding, with zero for open and one for pregnant. This resulted in 45 data sets to be analyzed.

Analysis of Simulated Data. I estimated heifer pregnancy variance components for each of the 45 data sets and then calculated EPD and their accuracies for each data set using the variance components from its corresponding variance component analysis. Variance components were obtained by applying a Method R(Reverter et al., 1994) approach using a maximum *a posteriori* probit (MAP) threshold model (Foulley and Gianola, 1984; Harville and Mee, 1984; Snelling et al., 1995), using the dscat software developed by Kaiser (1996). This software performs Method R by taking random 50 % sub-sample of the data. The MAP model fit for all heifer pregnancy variance component analyses was

$$\mathbf{Y = X\beta + Zu + e,}$$

where

$$\text{var}\begin{bmatrix} u \\ e \end{bmatrix} = \begin{bmatrix} A\sigma_a^2 & 0 \\ 0 & I\sigma_e^2 \end{bmatrix}$$

The vector **Y** are subclass "psuedo-observations" on the underlying scale. **X** and **Z** are known incidence matrices relating subclasses to fixed and random effects. Vector **β** are fixed effect levels, vector **u** are random effect levels, and vector **e** are residual error. The additive direct genetic variance, σ^2_a, was dispersed by Wright's numerator relationship matrix, **A**. The residual error variance, σ^2_e, was dispersed by an identity matrix, **I**. The software constrained the residual error variance to be 1 (Kaiser, 1996), in accordance with a probit model.

No fixed effects had been simulated but the dscat software required at least one to be fit. I created a fixed effect column in the data file and assigned each heifer one of three possible levels; the levels were assigned in repeating sequence down the list of heifers. The heifers had been created and written to the file in random order, except with respect to their maternal grand dam, so the assignment of the fixed effect level was essentially random.

I obtained 50 Method R estimates of heritability for each of the 45 data sets, and selected the median of estimates within the parameter space. Kaiser (1996) reported a mode as being a better estimator than the mean, but later work by Mahibir (2003) suggests the mode is a better statistic. A different random seed, obtained sequentially from a table of random numbers (Steel and Torrie, 1980), was used to start each set of 50 estimates. The dscat software used the seed to set a random number generator which was then used to obtain the 50 % subsamples.

There were a number of convergence criteria to consider when using Method R and a threshold model to obtain a point-estimate of heritability. Iteration within a given Fisher Scoring round was considered converged when change was less than 1×10^{-9}. Sequential Fisher scoring rounds were considered converged when the changed between rounds was less than 1×10^{-4} and at least 25 rounds of scoring were done. These were all hard-coded in dscat and were not changed for this study.

An input parameter to the dscat software was the R statistic convergence, which I set to 1×10^{-6}. The R statistic for a given Method R round had to be within this distance from 1.0 for that heritability estimate to be considered converged.

Finally, the number of heritability estimates necessary for a reliable point estimate can be selected by increasing the number of estimates until a suitable standard error is reached. Preliminary trials indicated that 50 samples yielded a standard error less than 0.005, so I targeted 50 samples as my convergence criteria for the heritability point estimate.

The variance components obtained were used to produce BLUP breeding values for the data set they were estimated from, using the same statistical model and software. The dscat software (Kaiser, 1996) performed BLUP using the input variance components on the whole data set, and it output EBV on the underlying scale.

One of the objectives was to characterize the accuracy of heifer pregnancy EBV for predicting the simulated component traits of fertility. Since the simulated AAP and PCON BP were true values, I calculated a simple correlation of heifer pregnancy EBV with AAP and with PCON as measures of accuracy. I also calculated accuracies of the heifer pregnancy EBV by creating a reduced animal model coefficient matrix and then estimating the prediction error variance using the *tinv* tool from the ABTK (Golden et al., 1992). This tool calculates exact prediction error variances from a coefficient matrix. Prediction error variance ($\mathbf{PEV}_i$) for animal ***i*** was converted to accuracy ($\mathbf{r}_i$) for that animal by

$$r_i = \sqrt{1 - \frac{PEV_i}{\sigma_a^2}},$$

where $\boldsymbol{\sigma}^2_a$ is the additive direct genetic variance. Inbreeding was minimal in these data sets and was not accounted for in calculation of accuracy from PEV.

Results and Discussion

Data generated. The percent of heifers pregnant in each of the 45 data sets (Table 4.3) is a function of the simulated mean PCON and AAP, and the breeding season length. As discussed in Chapter 2, it is also affected by the sterile heifers (Table 4.4), the resampling of temporary environmental variance in the simulation each estrus, and decreased fertility on the pubertal and second estrus. Most heifers in the 340 d AAP data sets reached puberty two estrous cycles before the start of the breeding season, as 62, 71, and 80 % of the heifers with simulated PCON of 60, 70, and 80 %, respectively, became pregnant within the 25 d breeding season. Half the heifers reached puberty at least 70 d before the start of the breeding season, and approximately 88 % reached puberty 42 d before the start of the breeding season (Fig. 4.1). The curves for cumulative count of heifers reaching puberty in Figures 4.1, 4.2, and 4.3 are from the data sets for 340, 390, and 440 d AAP, respectively, at 60 % PCON. I initially plotted the cumulative puberty data from data sets for 70 and 80 % PCON and found essentially no difference from 60 % PCON, as expected, so for simplification I did not include them in the figures. Figures 4.1, 4.2, and 4.3 do each include curves for cumulative pregnancy at 60, 70, and 80 % PCON because they are predictably different.

The percent of heifers pregnant in simulations with puberty later than 340 d showed a decrease in conception in the 25 d breeding season, below the simulated PCON level (Table 4.3). Figure 4.2 shows that while most heifers in the 390 d AAP had begun cycling by the start of the breeding season, fewer than half were in their third estrus. Since

the pubertal and second estrus were modeled with reduced conception, the reduced conception in Table 4.3 is expected. The reduced pregnancy rate at 25 d was even greater in the 440 d AAP datasets (Table 4.3). By definition, fewer than half of these heifers had reached puberty at the start of the breeding season because it started when they were 420 d old.

Table 4.3. Percent of heifers pregnant at different levels of simulated age at puberty (AAP) and probability of conception (PCON) by breeding season length.

Breeding season length (d)	Simulated mean 60 % PCON Simulated mean age at puberty (AAP) 340 d	390 d	440 d
	----------Heifers pregnant (%)----------		
25	62.0	57.7	30.1
45	78.8	76.5	52.1
60	84.5	83.1	65.1
90	90.1	89.4	81.7
120	92.4	92.1	88.7

Breeding season length (d)	Simulated mean 70 % PCON Simulated mean age at puberty (AAP) 340 d	390 d	440 d
	----------Heifers pregnant (%)----------		
25	71.4	66.0	36.7
45	85.5	83.0	59.8
60	89.6	88.1	73.0
90	92.8	92.5	87.4
120	93.9	93.9	92.2

Breeding season length (d)	Simulated mean 80 % PCON Simulated mean age at puberty (AAP) 340 d	390 d	440 d
	----------Heifers pregnant (%)----------		
25	80.0	76.3	43.0
45	90.9	89.0	66.4
60	93.0	92.2	78.9
90	94.7	94.4	91.1
120	95.1	94.9	94.0

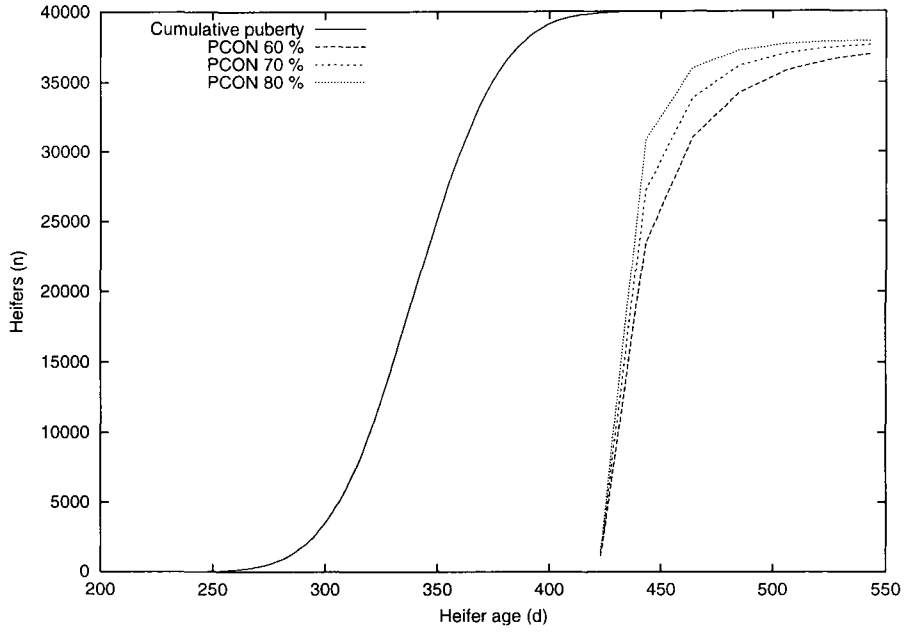


Figure 4.1.Cumulative number of heifers reaching puberty in the 340 d AAP and 60 % PCON data, and cumulative number of heifers conceiving within the breeding season in the 60, 70, and 80 % PCON data sets with 340 d AAP.

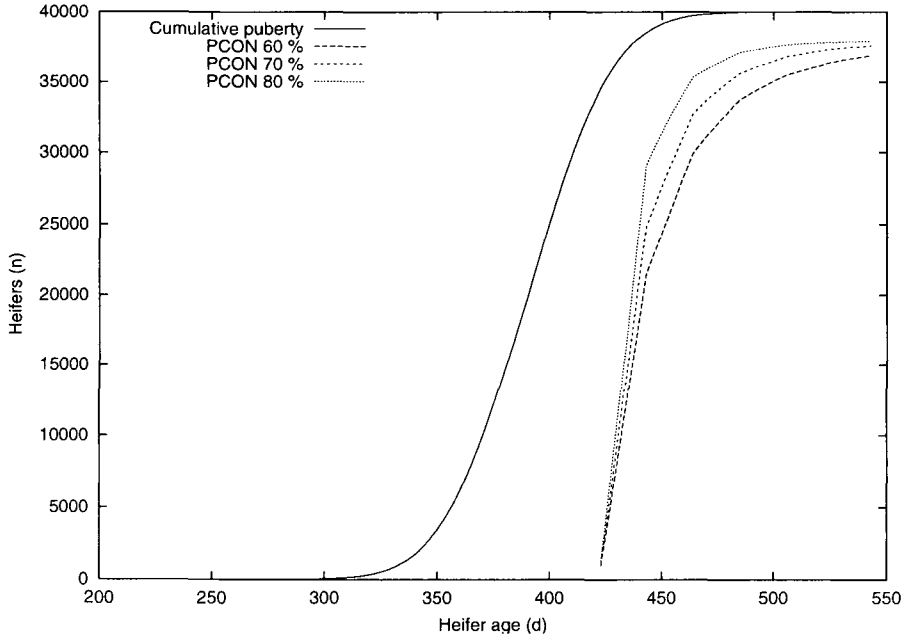


Figure 4.2.Cumulative number of heifers reaching puberty in the 390 d AAP and 60 % PCON data, and cumulative number of heifers conceiving within the breeding season in the 60, 70, and 80 % PCON data sets with 390 d AAP.

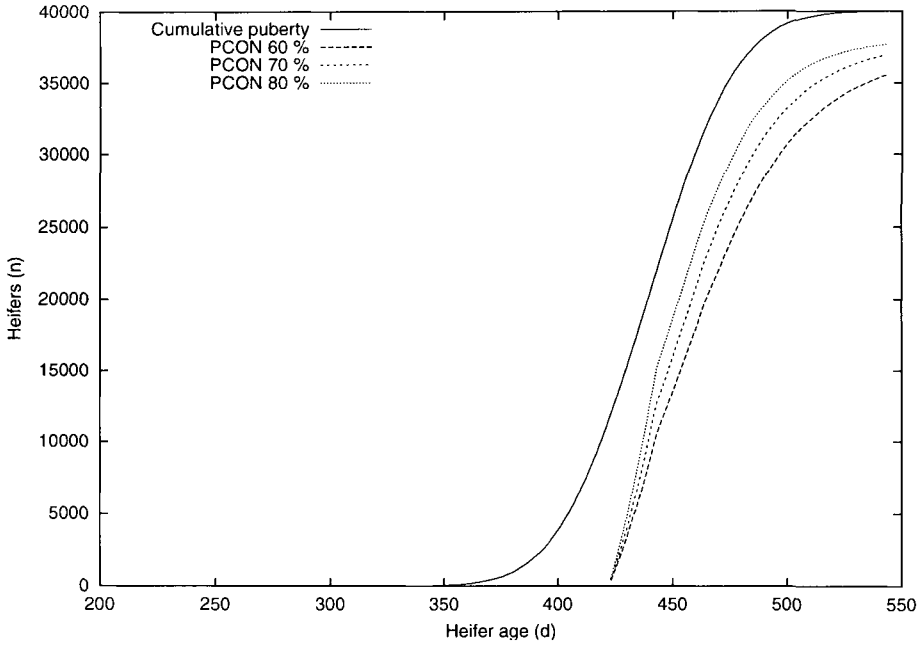


Figure 4.3. Cumulative number of heifers reaching puberty in the 440 d AAP and 60 % PCON data, and cumulative number of heifers conceiving within the breeding season in the 60, 70, and 80 % PCON data sets with 440 d AAP.

Table 4.4 Number and percent of heifers, within each of the nine simulated data sets of 20,000 heifers, that were simulated to be sterile.

	PCON (%)					
	60		70		80	
AAP (d)	n	%	n	%	n	%
340	985	4.9	1019	5.1	928	4.6
390	1008	5.0	1003	5.0	966	4.8
440	1019	5.1	1010	5.1	985	4.9

pubertal and second estrus have reduced conception the reduced conception in Table 4.3 is expected. The reduced pregnancy rate at 25 d was even greater in the 440 d AAP datasets (Table 4.3). By definition, fewer than half of these heifers had reached puberty at the start of the breeding season because it started when they were 420 d old.

As the breeding seasons increase in length, the cumulative pregnancy curves (Figs. 4.1, 4.2, 4.3) increase at a decreasing rate because there were fewer remaining heifers and the remaining heifers were less fertile. This decreased rate was more pronounced at earlier AAP (Fig 4.1) because with later AAP (Fig 4.3) the AAP delayed conception more than PCON. However, by 120 d all cumulative pregnancy curves are converging to the asymptote. This is less than 100 % pregnancy, due to the simulated infertility.

Analysis of Simulated Data.

Variance Component Estimation. I had difficulties with getting dscat to estimate heritability, particularly in cases where a high percentage of heifers were pregnant. In Tables 4.6, 4.7, and 4.8 the second column indicates the number of converged samples out

of the total number of samples attempted. Preliminary attempts had even fewer converged estimates within the parameter space, which lead to the modifications allowing foundation cows and sires to be related. This modification to the model did result in more samples converging, but not all. An alternative would have been to run the simulation for a number of years to build more generations to the pedigree and possibly increase the overall degree of relatedness among the heifers with observations.

Table 4.5. Number of heifers, within each of the nine simulated data sets of 20,000 heifers, with a non-zero inbreeding coefficient (F) and average heifer inbreeding coefficient including non-inbred animals.

	PCON (%)					
	60		70		80	
AAP (d)	Non-zero F (n)	Average F	Non-zero F (n)	Average F	Non-zero F (n)	Average F
340	175	0.0008	207	0.0010	198	0.0010
390	202	0.0009	208	0.0010	198	0.0011
440	191	0.0009	212	0.0011	207	0.0011

However, that would have added the complexity of natural selection to the interpretation of the results, as only heifers fertile enough to breed would have had daughters with observations. It also would have added heifer age as a variable.

An alternative explanation is that the convergence method within dscat is not capable of handling samples with very little variation. Kaiser (1996) added an acceleration term to speed convergence which may contribute to oscillation or divergence.

At one extreme, with AAP, PCON, and breeding season length at 340 d, 80 %, and 120 d, respectively, only one estimate out of 100 attempts produced an answer within

the parameter space (Table 4.6). In one of the remaining 99 cases the software was unable to converge on a solution, and in the other 98 cases the estimate of genetic variance was negative. Doyle et al. (2000) analyzed Angus field data for fertility and concluded estimates of heifer pregnancy outside the parameter space indicated the model was inappropriate for the data in those cases. In addition, they were hesitant to conclude that rebreeding was heritable because half their samples yielded estimates outside the parameter space.

In this study's most extreme case (described above) only 57 (0.285 %) of the fertile heifers remained open, in addition to the 928 heifers open due to simulated infertility. In this case it is probably more appropriate to conclude these data have too little power to estimate the heritability, as opposed to saying the model is inappropriate. Since I know how the data were simulated, particularly the lack of other simulated effects, I feel it is appropriate to interpret the heritability estimates despite the number of samples that either went out of the parameter space or did not converge.

The frequencies of observations should not affect the ability of the threshold model to estimate heritability. However, it is possible that too many sires have no variation among their daughters' performance.

Table 4.6. Heritability estimates of heifer pregnancy using 20,000 heifer pregnancy observations from data with simulated mean 340 d AAP at three levels of PCON.

		Heritability Estimates				
Breeding season length (d)	Converged samples (total samples) (n)	Mode	Mean	Variance	Minimum	Maximum
			Simulated 60 % PCON			
25	49 (50)	0.139	0.139	0.0003	0.086	0.184
45	49 (49)	0.172	0.180	0.0003	0.125	0.250
60	50 (50)	0.156	0.159	0.0007	0.108	0.219
90	18 (50)	0.127	0.140	0.0001	0.106	0.240
120	9 (50)	0.060	0.070	0.0024	0.053	0.151
			Simulated 70 % PCON			
25	24 (50)	0.107	0.106	0.0003	0.078	0.145
45	44 (50)	0.125	0.126	0.0002	0.094	0.182
60	25 (50)	0.125	0.113	0.0005	0.063	0.156
90	8 (50)	0.049	0.050	0.0002	0.031	0.067
120	13 (50)	0.063	0.070	0.0024	0.011	0.159
			Simulated 80 % PCON			
25	25 (50)	0.143	0.147	0.0008	0.105	0.212
45	13 (50)	0.118	0.132	0.0017	0.078	0.226
60	9 (100)	0.086	0.084	0.0010	0.037	0.123
90	7 (100)	0.083	0.088	0.0023	0.038	0.155
120	1 (100)	0.106	na	na	na	na

Table 4.7. Heritability estimates of heifer pregnancy using 20,000 heifer pregnancy observations from data with simulated mean 390 d AAP at three levels of PCON.

		Heritability				
Breeding season length (d)	Converged samples (total samples) (n)	Mode	Mean	Variance	Minimum	Maximum
		Simulated 60 % pregnancy				
25	21 (50)	.169	0.175	0.0023	0.117	0.292
45	39 (50)	.195	0.222	0.0016	0.147	0.322
60	50 (50)	.164	0.170	0.0007	0.125	0.239
90	22 (50)	0.119	0.120	0.0010	0.068	0.200
120	13 (50)	0.091	0.096	0.0015	0.061	0.205
		Simulated 70 % pregnancy				
25	47 (50)	0.124	0.124	0.0005	0.070	0.188
45	29 (50)	0.113	0.116	0.0006	0.076	0.168
60	24 (49)	0.088	0.086	0.0003	0.047	0.119
90	3 (50)	0.042	0.040	0.0004	0.020	0.058
120	4 (100)	0.047	0.071	0.0003	0.026	0.153
		Simulated 80 % pregnancy				
25	44 (50)	0.125	0.129	0.0003	0.092	0.171
45	39 (50)	0.125	0.121	0.0008	0.054	0.211
60	20 (50)	0.076	0.079	0.0004	0.050	0.138
90	6 (50)	0.049	0.052	0.0001	0.041	0.064
120	4 (50)	0.046	0.045	0.0009	0.004	0.074

Table 4.8. Heritability estimates of heifer pregnancy using 20,000 heifer pregnancy observations from data with simulated mean 440 d AAP at three levels of PCON.

		Heritability				
Breeding season length (d)	Converged samples (total samples) (n)	Mode	Mean	Variance	Minimum	Maximum
		Simulated 60 % pregnancy				
25	50 (50)	0.277	0.276	0.0016	0.180	0.391
45	50 (50)	0.266	0.265	0.0013	0.188	0.375
60	50 (50)	0.250	0.244	0.0011	0.180	0.328
90	50 (50)	0.188	0.188	0.0009	0.127	0.279
120	50 (50)	0.165	0.167	0.0008	0.102	0.250
		Simulated 70 % pregnancy				
25	42 (50)	0.284	0.298	0.0020	0.238	0.416
45	34 (50)	0.343	0.342	0.0019	0.260	0.424
60	42 (50)	0.262	0.270	0.0031	0.149	0.399
90	13 (50)	0.167	0.175	0.0016	0.126	0.260
120	10 (50)	0.059	0.100	0.0040	0.043	0.236
		Simulated 80 % pregnancy				
25	32 (50)	0.337	0.333	0.0022	0.229	0.459
45	50 (50)	0.330	0.329	0.0017	0.252	0.431
60	49 (50)	0.242	0.247	0.0019	0.172	0.357
90	23 (61)	0.095	0.104	0.0005	0.076	0.146
120	19 (50)	0.053	0.054	0.0006	0.010	0.103

In theory it is possible to calculate the expectation of the heritability of AAP from these simulated data at a given day in the breeding season. In practice it is complicated because of the simulated infertility and because as the season progresses there are repeated opportunities to breed. I'll start with a simplified case with no infertility and only one opportunity to breed (i.e. a 21 d breeding season). The probability of heifer conception (**P(c)**) is a function of the probability a heifer reached puberty at a given age (**P(p)**) and her probability of conception given puberty(**P(c|p)**). Using Bayes' theorem,

$$P(c|p) = \frac{P(p|c) * P(c)}{P(p)},$$

where P(p|c) is the probability of puberty given conception. Since a heifer must, by definition, reach puberty in order to conceive, that term is set to 1.0, and the equation simplifies to

$$P(c) = P(p) * P(c|p).$$

A heifer's **P(c)** at a given time is equivalent to pregnancy in this model, as no embryonic loss or abortions were simulated. The probability density function for both P(p) and P(c|p) were simulated with a normal distribution and variance of 1.0, and both traits were simulated as genetic potentials, with no simulated correlations between the traits.

When thinking about the distribution of heifer pregnancy being a joint distribution of AAP and PCON it may be helpful to first consider the extremes of the two distributions. If all animals have reached puberty before the breeding season starts then the distribution of heifer pregnancy in the first 21 d of the breeding season should be identical

to the simulated distribution of PCON and estimated heritability of heifer pregnancy should equal input heritability of PCON. If none of the heifers have reached puberty by the end of the first 21 d of the breeding season then the distribution will be a point with a corresponding zero probability of pregnancy. Similarly, if PCON is one, then heifer pregnancy at a given time becomes similar to a measure of AAP, and estimated heritability of heifer pregnancy should be close to input heritability of AAP. It is slightly different because the observation of heifer pregnancy has converted the (underlying) continuous AAP trait to a binary threshold trait. Finally, when PCON equals zero or when no heifers have reached puberty it is less informative, as there is no heifer pregnancy variance to analyze. To summarize, when there is variation in heifer pregnancy the estimated heritability is expected to be AAP heritability when PCON equals one and PCON heritability when all animals have reached puberty by the start of the breeding season.

Determining the expectation of the heritability of heifer pregnancy for values of PCON and AAP between these extremes is more complex. The joint distribution can be written as

$$\phi_{preg}(t) = \int \phi_{preg|puberty}(t) * \phi_{puberty}(a)\, d_a$$

where $\phi_{preg}(t)$ is the probability of pregnancy at time t, $\phi_{puberty}(a)$ is the probability of puberty at or before time a, and $\phi_{preg|puberty}(t)$ is the probability of pregnancy at time t given puberty has already been reached. The expected variance of pregnancy observations, $E[\sigma^2_{preg}]$, depends on the expectation of the probability of pregnancy at time t, μ_{preg}, as seen in

$$E[\sigma^2_{preg}] = \int (t-\mu_{preg})^2 * \phi_{preg|puberty}(t) * \phi_{puberty}(a)\, d_a d_t.$$

The μ_{preg} is calculated as

$$\mu_{preg} = E[\phi_{preg}(t)] = \int t^{*}\phi_{preg|puberty}(t) * \phi_{puberty}(a)\, d_a d_t.$$

Returning to the analysis of my simulated data, the case when AAP and breeding season length were 340 and 25 d, respectively, was as close to one extreme as I simulated. If all animals were cycling and fertile and had one chance to conceive then the heritability would be expected to be 0.10 given the input parameters. My estimates were 0.139, 0.107, and 0.143 for 60, 70, and 80 % PCON, respectively. Two of these are higher than expected. It is not clear what role the repeated opportunity to breed for some animals and the infertility played.

I did not simulate the other extreme, with 100 % PCON and late AAP; the expected heritability would be 0.40 in that case. The closest to this extreme was the 440 d AAP and 80 % PCON data set. It yielded the second highest heritability estimate (0.337). The expectation supports the result of increasing heritability with later AAP given the input parameters, but it is not possible to do a conclusive test with these data.

Correlation of Breeding Values and the EBV Accuracy. To study the relationship between the heifer pregnancy EBV and the two simulated fertility traits I calculated the correlation between the EBV and each trait for all data sets, using the 500 sires of heifers. I plotted these correlations and also plotted the average accuracy (r_i) of the EBV for these 500 sires (Figs. 4.4 through 4.12). Since the simulated BP are true values, the correlations are essentially the accuracy of prediction of AAP and PCON, while the additional accuracy, r_i, is the accuracy of the heifer pregnancy EBV.

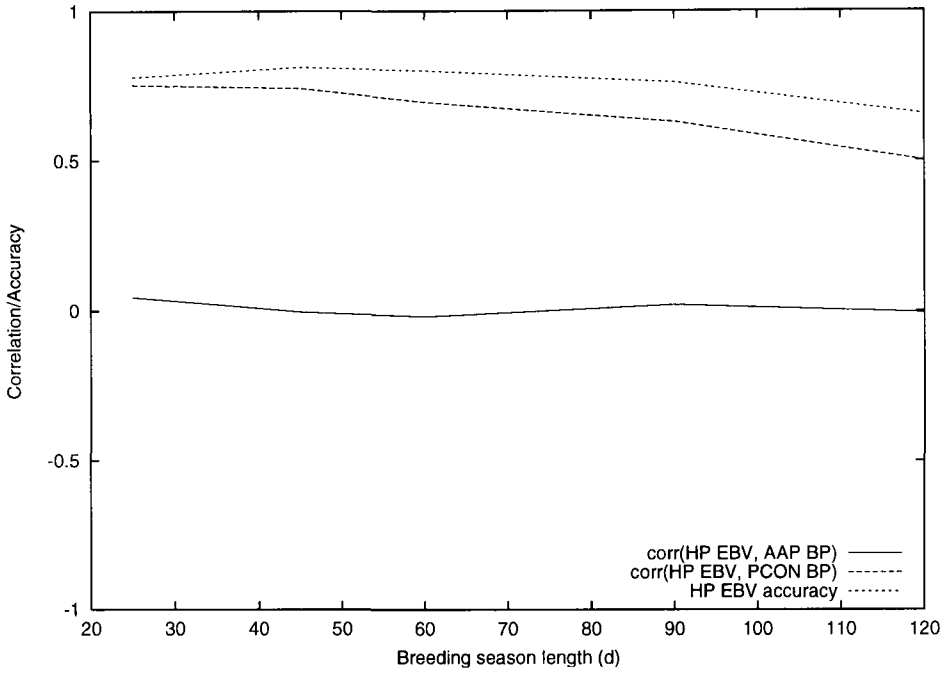


Figure 4.4. Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 340 d AAP and 60 % PCON.

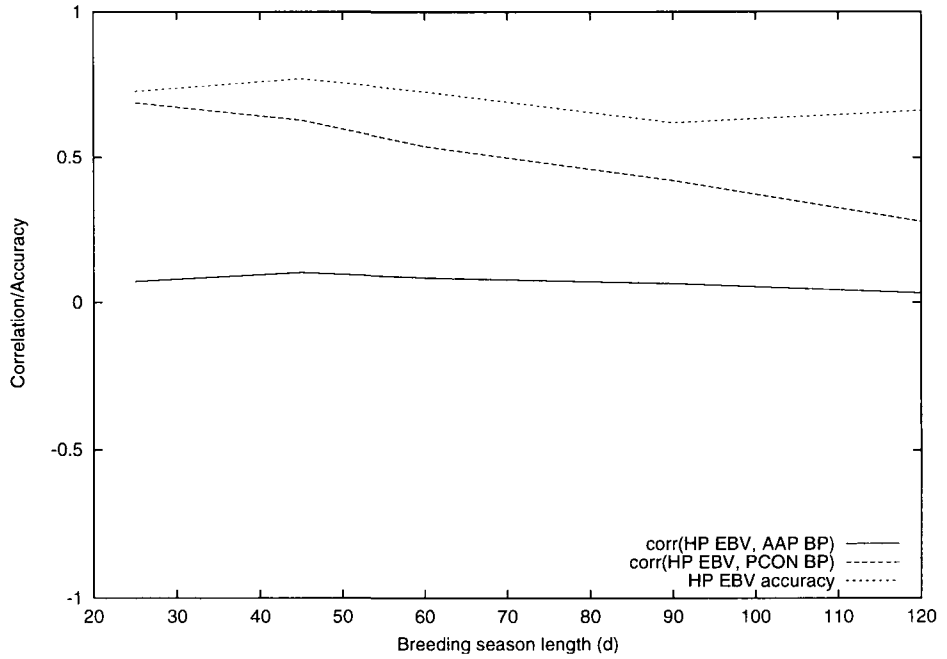


Figure 4.5. Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 340 d AAP and 70 % PCON.

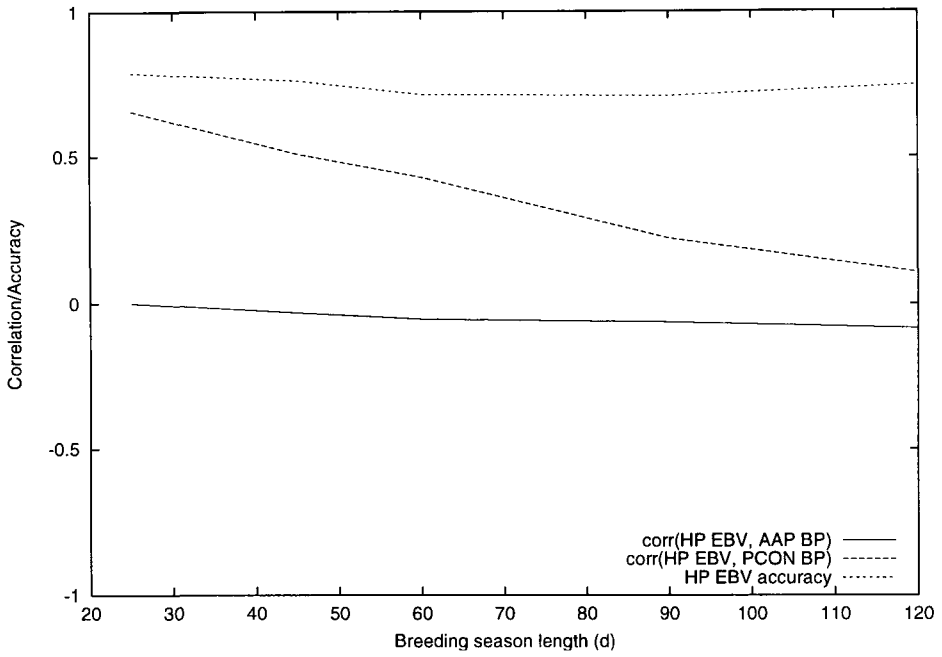


Figure 4.6. Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 340 d AAP and 80 % PCON.

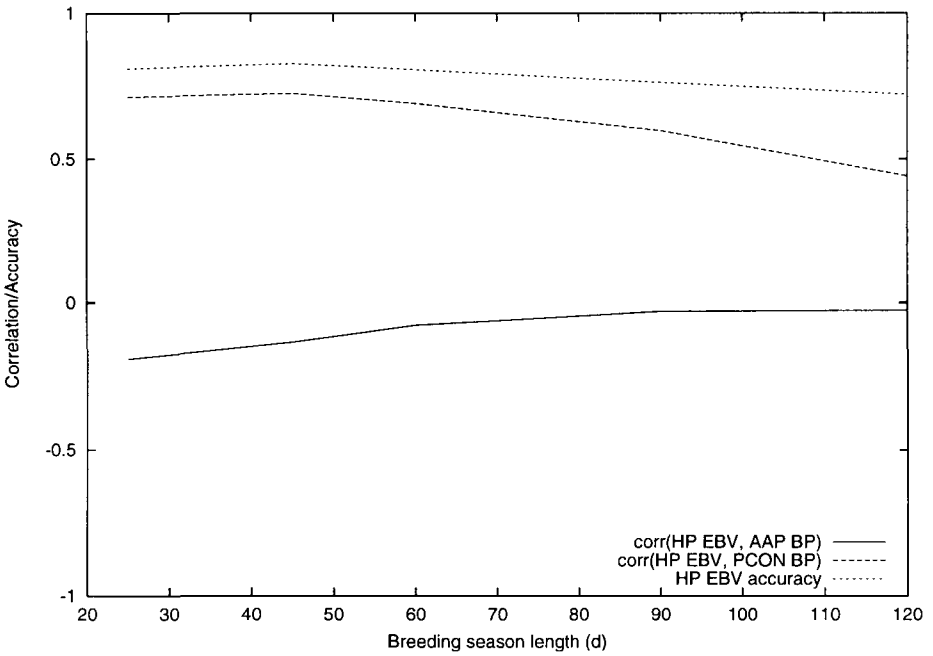


Figure 4.7. Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 390 d AAP and 60 % PCON.

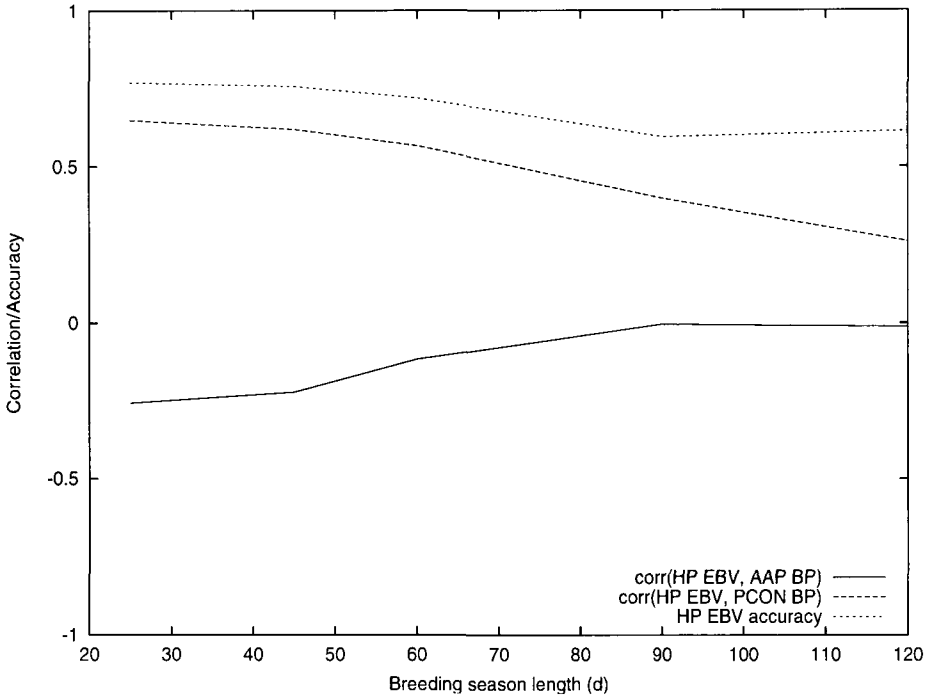


Figure 4.8. Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 390 d AAP and 70 % PCON.

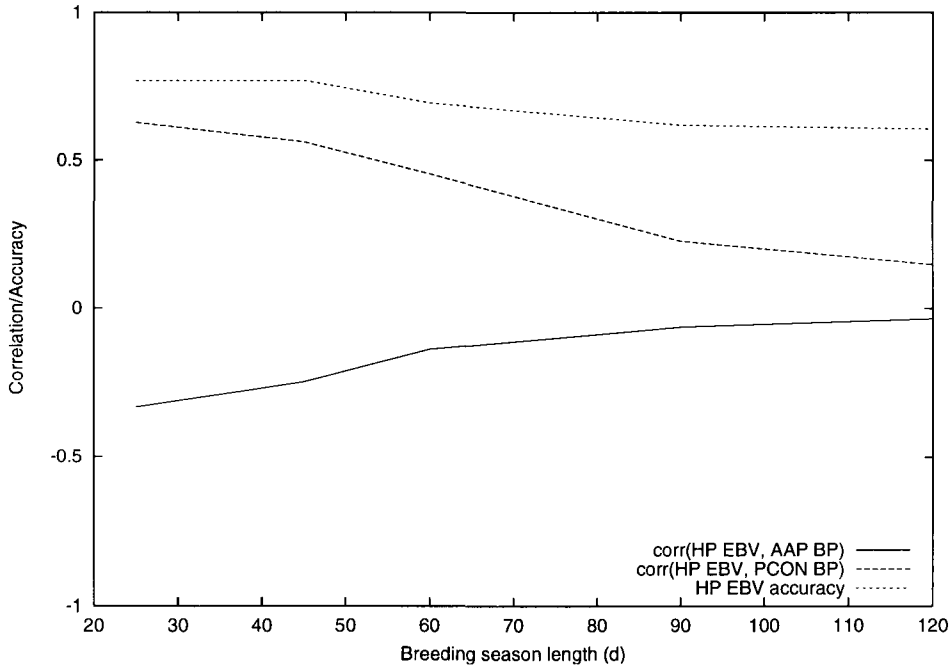


Figure 4.9. Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 390 d AAP and 80 % PCON.

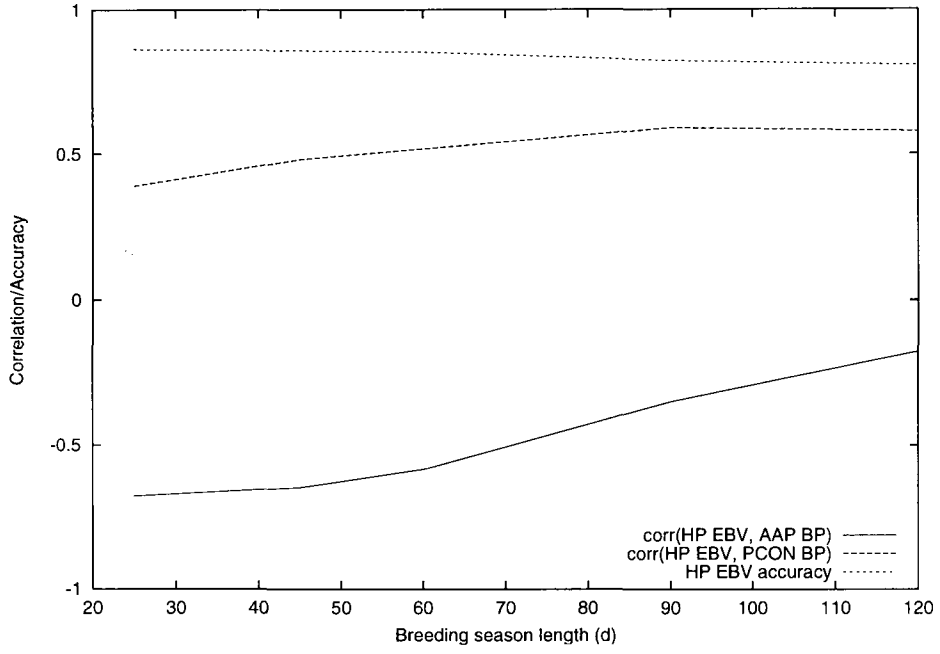


Figure 4.10. Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 440 d AAP and 60 % PCON.

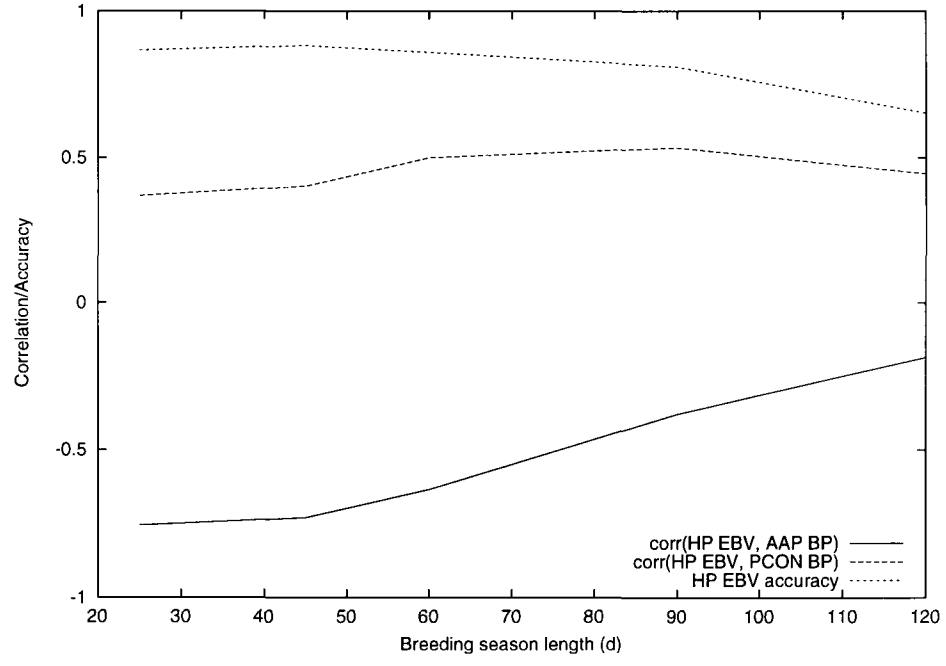


Figure 4.11. Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 440 d AAP and 70 % PCON.

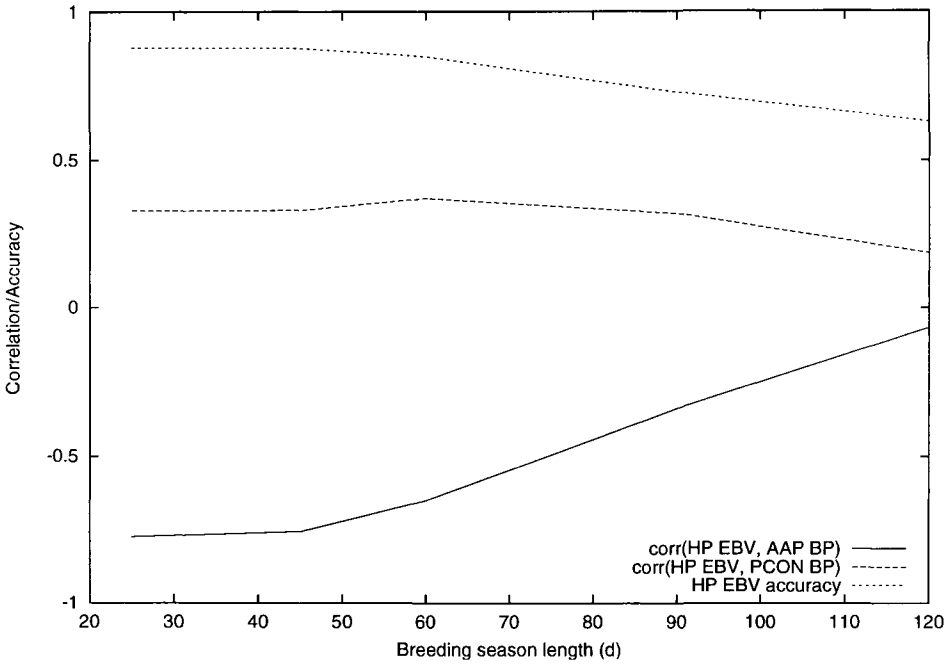


Figure 4.12. Accuracy of heifer pregnancy (HP) EBV, and correlation of HP EBV with age at puberty breeding potential (AAP BP) and with probability of conception breeding potential (PCON BP) for simulated 440 d AAP and 80 % PCON.

In the data sets with 340 d mean AAP (Figs 4.4, 4.5, and 4.6) there was essentially no correlation between the heifer pregnancy EBV and the AAP BP. Since essentially all heifers had reached puberty prior to the start of the breeding season there was no way for the EBV to detect differences in sire's AAP BP. The correlation with PCON, however, showed a clear, consistent trend in these data. The shortest breeding season provided the best measure of PCON BP, and the relationship deteriorated steadily as the breeding season length increased. It remained the strongest with low PCON (Fig 4.4), and became essentially zero with high PCON and 120 d breeding season (Fig 4.6). In Figure 4.4 it appears that a heifer pregnancy EBV from these data would still provide reasonable ability to select for PCON BP with a 120 d breeding season.

The calculated accuracy of the EBV, on the other hand, was quite insensitive to these changes. This was likely in part because it was calculated with a linear model, not a threshold model. This method has been standard procedure at Colorado State University Center for Genetic Evaluation of Livestock (CSU-CGEL) for threshold traits. This method was not able to account for the changes in frequency of observations in a binary threshold trait in these data. In theory, the calculated accuracy of the EBV would decline as the frequency of one class of observations (e.g. pregnant) moved from 50 % toward 100 %. However, the linear model simply accounted for the amount of available observations, which remained the same, and also the heritability, which changed slightly in these data. This would provided a false sense of the correlation between the EBV and the true value for the trait; although I do not have a true value for heifer pregnancy, I know from the simplicity of the simulation that there are not likely to be any additional,

unforeseen effects. The best illustration of the problem is Fig. 4.6 at 120 d into the breeding season, where the EBV accuracy remained unchanged but the EBV correlations with the simulated values was essentially zero.

Continuing on to the correlations in 390 d mean AAP, the trend in PCON remained the same but there was also a trend in AAP (Figs 4.7, 4.8, and 4.9). The EBV from short breeding seasons had a negative correlation with AAP BP, and as the season length increased the correlation moved toward zero. The negative correlation implies that animals with higher heifer pregnancy EBV had earlier AAP BP. The EBV in these data would allow favorable selection simultaneously in AAP and PCON in short breeding seasons. As breeding season length increased the power of this relationship decreased. The 120 d breeding seasons would allow essentially no selection pressure on AAP.

In the last set of Figures (4.10, 4.11, and 4.12), for 440 d AAP, the trend for AAP was similar and more pronounced than with 390 d AAP. The EBV from short breeding seasons provided an accurate measure of AAP BP, as indicated by the relative strength of the correlation. Again, as the breeding season lengthened, this relationship deteriorated steadily toward zero. The correlation of the EBV with PCON BP had a reverse relationship with low PCON (Fig 4.10), although it was not a large difference from the start of the breeding season to the end. It appears that AAP was initially the dominating factor in determining which heifers became pregnant, and as the season lengthened it was less so. The PCON BP was a moderate factor at 60 and 70 % PCON, and less so at 80 % PCON.

In these data I would recommend using EBV from the 25 d truncated breeding season, except in the 440 d AAP, where I would use the 45 d truncated breeding season. The 440 d AAP data have less than 50 % pregnancy in the 25 d breeding season and so would likely benefit by increasing the length to 45 d. Real life recommendations, however, are not quite so clear cut. Heifers will be different ages, have different age dams, have variation in gestation length, and experience a number of other genetic and environmental effects that add to the problem. Still, I am inclined to believe that using data from shorter breeding seasons in field data would add to the true accuracy of the EBV.

Summary

Indications of GxE, GxGxE, or heterogeneity in these simulated data would suggest to look for it in field data, while absence of it here would suggest if it is observed in field data it is either an artifact of the data structure or it is due to other causes. Clearly in this study the heritability changed significantly depending on the simulated levels of AAP and PCON. It also decreased as breeding season length increased, although the reliability of estimates where few samples converged in the parameter space was questionable.

In addition, the strength of the correlation between the simulated traits and the HP EBV tended to decrease toward zero as breeding season length increased. In these data a HP EBV based on a 120 d breeding season would provide little value in improving heifer pregnancy, while one from a 25 d breeding season would provide the most.

In practice, the effect of selection using a HP EPD will depend largely on the genetic potential for puberty, and on the genetic potential for probability of conception. It will also depend on the environment dictated by management, particularly the length of the breeding season. There are a number of additional effects which will likely need to be accounted for in a heifer pregnancy analysis but which were not simulated in this study.

Calculating EBV accuracy for a threshold trait using a linear model coefficient matrix was a poor measure of true accuracy, as evidenced in Figures 4.4 through 4.12 in long breeding seasons.

LITERATURE CITED

Ageloff, R., and R. Mojena. 1981. Applied FORTRAN 77 featuring structured programming. Wadsworth Publishing Co. Belmont, CA.

Andersen, K.J., D.G. LeFever, J.S. Brinks, and K.G.Odde. 1990. Reproductive tract scores in beef heifers. Agri-Practice, II:6.

Anderson, H., and M. Plum. 1965. Gestation length and birth weight in cattle and buffaloes: a rieview. J. Dairy Sci. 48:1224.

Arije, G.F., and J.N. Wiltbank. 1971. Age and weight at puberty in Hereford heifers. J. Anim. Sci. 33:401.

Azzam, S. M., and M. K. Nielsen. 1987. Genetic parameters for gestation length, birth date, and first breeding date in beef cattle. J. Anim. Sci. 64:348-356.

Azzam, S.M., J.E. Kinder, and M.K.Nielsen. 1990. Modelling reproductive management systems for beef cattle. Agric. Sys. 34:103.

Baker, B. B., R. M. Bourdon, and J. D. Hansen. 1992. FORAGE: A simulation model of grazing behaviour for beef cattle. Ecol. Mod. 60, 257.

Beck, K. 1999. Extreme programming explained: embrace change. Addison-Wesley.

Bellows, R. A., and R. E. Short. 1994. Reproductive losses in the beef industry. Page 109 in Factors Affecting Calf Crop. M. J. Fields and R. S. Sand, ed. CRC Press, Boca Raton, FL.

Booch, G., J. Rumbaugh, and I. Jackobson. 1999. The unified modeling languange user guide. Addison-Wesley.

Bossis, I., R.P. Wettermann, S.D. Welty, J.A. Vizcarra, L.J. Spicer, and M.G. Diskin. 1999. Nutritionally induced anovulation in beef heifers: ovarian and endocrine function preceding cessation of ovulation. J. Anim. Sci. 77:1536.

Bourdon, R. M. 1983. Simulated effects of genotype and management on beef production efficiency. Ph.D. dissertation. Colorado State University, Fort Collins.

Bourdon, R.M., and J.S. Brinks. 1986. Scrotal circumfrence in yearling Hereford bulls: adjustment factors, heritabilities, and genetic, environmental, and phenotypic

relationships with growth traits. J. Anim. Sci. 62:958.

Bourdon, R.M., and J.S. Brinks. 1987a. Simulated efficiency of range beef production. I. Growth and milk production. J. Anim. Sci. 65:943.

Bourdon, R.M., and J.S. Brinks. 1987b. Simulated efficiency of range beef production. II. Fertility traits. J. Anim. Sci. 65:956.

Bourdon, R.M., and J.S. Brinks. 1987c. Simulated efficiency of range beef production. III. Culling strategies and nontraditional management systems. J. Anim. Sci. 65:963.

Bourdon, R. M. 1992. Course notes for AN681: Computer simulation of beef cattle production. Colorado State University, Fort Collins (Mimeo).

Bourdon, R. M. 1998. Shortcomings of current genetic evaluation systems. J. Anim. Sci. 76:2308.

Brody, S. 1945. Bioenergetics and growth. Reinhold Publishing Corp., New York.

Burris, M.J. and C.T. Blunn. 1952. Some factors affecting gestation length and birth weight of beef cattle. J. Anim. Sci.11:34-41.

Byerley, D.J., R.B. Staigmiller, J.G. Berardinelli and R.E. Short. 1987. Pregnancy rates of beef heifers bred either on puberal or third estrus. J. Anim. Sci. 65:645-650.

Clanton, D.C., L.E. Jones, and M.E. England. 1983. Effect of rate and time of gain after weaning on the development of replacement beef heifers. J. Anim. Sci. 56:280.

Coulter, G.H., and R.H. Foote. 1979. Bovine testicular measurements as indicators of reproductive performance and their relationships to productive traits in cattle: A review. Theriogenology. 11:297-311.

Custer, E.E., J.G. Berardinelli, R.E. Short, M. Wehrman, and R. Adair. 1990. Postpartum interval to estrous and patterns of LH and progesterone in first-calf suckled beef cows exposed to mature bulls. J. Anim. Sci. 68:1370-1377.

Dearborn, D.D. 1971. An analysis of reproductive traits in beef cattle. Ph.D. Dissertation. University of Nebraska, Lincoln.

Dempster, E.R. and I.M. Lerner. 1950. Heritability of threshold characters. Genetics. 35:212.

Doyle, S.P. 2000. Impacts of sexed semen utilization on commercial beef production.

Ph.D. dissertation. Colorado State University, Fort Collins.

Doyle, S.P., B.L. Golden, R.D. Green, and J.S. Brinks. 2000. Additive genetic parameter estimates for heifer pregnancy and subsequent reproduction in Angus females. J. Anim. Sci. 78:2091.

Enns, R. M. 1995. Simulation of across-breed comparisons. Ph.D. dissertation. Colorado State University, Fort Collins.

Evans, J.L. 1996. Genetic parameter estimates of yearling heifer pregnancy and yearling bull scrotal circumference in Hereford cattle. M.S. thesis. Colorado State University, Fort Collins.

Evans, J.L., B.L. Golden, R.M. Bourdon, and K.L. Long. 1999. Additive genetic relationships between heifer pregnancy and scrotal circumference in Hereford cattle. J. Anim. Sci. 77:2621.

Falconer, D.S. 1989. Introduction to Quantitative Genetics. John Wiley & Sons, Inc., New York.

Foulley, J.L., and D. Gianola. 1984. Estimation of genetic merit from bivariate "all or none" responses. Genet. Sel. Evol. 16:285.

Fowler, M., and K. Scott. 2000. UML sistilled: a brief guide to the standard object modeling language. 2nd edition. Addison-Wesley.

Frisch, R.E. 1976. The physiological basis of reproductive efficiency. In: D. Lister, D.N. Rhodes, V.R. Fowler, and M.F. Fuller (Ed.). Meat Animals, Growth and Productivity. P 327. Plenum Press, New York.

Gianola, D. 1980. A method of sire evaluation for dichotomies. J. Anim. Sci. 51:1266.

Golden, B.L., W.M. Snelling, and C.H. Mallinckrodt. 1992. Animal breeder's tool kit user's guide and reference manual. Agric. Exp. Stn. Tech. Bull. LTB92-2. Colorado State Univiversity, Fort Collins.

Golden, B.L., L.S. Gould, R.L. Hough, and B.R. Schutte. 2000. Performance update: heifer pregnancy EPD - an economically relevant trait for improving heifer fertility. Amer. Red Angus. May/June.

Gosling, J., and H. McGilton. 1996. The Java languange environment. Sun Microsystems. Mountain View, CA.

Greer, R.C., R.W. Whitman, R.B. Staigmiller, and D.C. Anderson. 1983. Estimating the impact of management decisions on the occurance of puberty in beef heifers. J. Anim. Sci. 56:30.

Hall, J.B., R.B. Staigmiller, R.A. Bellows, R.E. Short, W.M. Moseley, and S.E. Bellows. 1995. Body composition and metabolic profiles associated with puberty in beef heifers. J. Anim. Sci. 73:3409.

Harville, D. A., and R. W. Mee. 1984. A mixed-model procedure for analyzing ordered categorical data. Biometrics. 40:393.

Hirooka, H., A.F. Groen, and J. Hillers. 1998a. Developing breeding objectives for beef cattle production 1. A bio-economic simulation model. Anim. Sci. 66:607-621.

Hirooka, H., A.F. Groen, and J. Hillers. 1998b. Developing breeding objectives for beef cattle production 2. Sci. 66:623-633.

Hoeschele, I., D. Gianola, and J.L. Foulley. 1987. Estimation of variance components with quasi-continuous data using Bayesian methods. J. Anim. Breeding Genet. 104:334.

Jacobsen, I., G. Booch, and J. Rumbaugh. 1999. The unified software development process. Addisson-Wesley.

Jafar, S.M., A.B. Chapman, and L.E. Casida. 1950. Causes of variation in length of gestation in dairy cattle. 9:593.

Jenkins, T.G., and C.B. Williams. 1998. DECI- Decision evaluator for the cattle industry. In: Proc. 6th World Cong. Genet. Appl. Livest. Prod., Armidale, Australia. 27:461-462.

Johnson, M.H., and D.R. Notter. 1987a. Simulation of genetic control of reproduction in beef cows. I. Simulation model. J. Anim. Sci. 65:68.

Johnson, M.H., and D.R. Notter. 1987b. Simulation of genetic control of reproduction in beef cows. II. Derived genetic parameters. J. Anim. Sci. 65:76.

Johnson, M.H., and D.R. Notter. 1987c. Simulation of genetic control of reproduction in beef cows. III. Within-herd breeding value estimation with known breeding dates. J. Anim. Sci. 65:88.

Jolly, P.D., S. McDougall, L.A. Fitzpatrick, K.L. Macmillan, and K.W. Entwistle. 1995. Physiological effects of undernutrition on postpartum anoestrus in cows. J. of Repro. And Fert. Suppl. 49:477-492.

Kahn, H.E. 1982. The development of a simulation model and its use in the evaluation of cattle production systems. PhD Thesis. University of Reading.

Kahn, H.E., and C.R.W. Speeding. 1983. A dynamic model for the simulation of cattle herd production systems: Part 1-General description and the effects of simulation techniques on model results. Agric. Syst. 12:101-111.

Kahn, H.E., and C.R.W. Speeding. 1984. A dynamic model for the simulation of cattle herd production systems: Part 2- An investigation of various factors influencing the voluntary intake of dry matter and the use of the model in their validation. Agric. Syst. 13:63-82.

Kahn, H. E., and A. R. Lehrer. 1984. A dynamic model for the simulation of cattle herd production systems: Part 3 - reproductive performance of beef cows. Agric. Sys. 13:143.

Kaiser, C.J. 1996. Incorporating birth weight information into a calving ease threshold model analysis. Ph.D. dissertation. Colorado State Univ., Fort Collins.

Kinder, J. E., M. S. Roberson, M. W. Wolfe, and T. T. Strumpf. 1994. Management factors affecting puberty in the heifer. Page 69 in Factors Affecting Calf Crop. M. J. Fields and R. S. Sand, ed. CRC Press, Boca Raton, FL.

Kinder, J.E., E.G.M. Bergfeld, M.E. Wehrman, K.E. Peters and F.N. Kojima. 1995. Endocrine basis for puberty in heifers and ewes. J. Repro. And Fertility Suppl. 49:393-407.

Koots, K.R., J.P. Gibson, C. Smith, and J.W. Wilton. 1994. Analyses of published genetic parameter estimates for beef production traits. 1. Heritability. Anim. Breed. Abstr. 62:309-338.

Lamb, M.A., M.W. Tess, and O.W. Robison. 1992. Evaluation of mating systems involving five breeds for integrated beef production systems. I. Cow-calf segment. J. Anim. Sci. 70:689-699.

Lamb, G.C., B.L. Miller, J.M. Lynch, K.E. Thompson, J.S. Heldt, C.A. Loest, D.M. Grieger, and J.S. Stevenson. 1999. Suckling Twice daily suckling but not milking with calf presence prolongs postpartum anovulation. J. Anim. Sci. 77:2207-2218.

Larsen, R. E., S. C. Denham, and J. Boucher. 1990. Breeding season length versus calving percentage in beef cattle herds. 39th Annual FL. Beef Cattle Short Course Proc. University of FL., Gainsville.

Laster, D.B., G.M. Smith, L.V. Cundiff and K.E. Gregory. 1979. Characterization of biological types of cattle (Cycle II). II. Postweaning growth and puberty of heifers. J. Anim. Sci. 48:50.

Lee, C., and E.J. Pollack. 1997. Influence of partitioning data by sex on genetic variance and covariance components for weaning weight in beef cattle. J. Anim. Sci. 75:61-67.

Legates, J. E. 1954. Genetic variation in services per conception and calving interval in dairy cattle. J. Anim. Sci. 13:81.

Lesmeister, J.L., P.J. Burfening and R.L. Blackwell. 1973. Date of first calving in beef cows and subsequent calf production. J. Anim. Sci. 36:1.

Liew, S.D. 1996. Statistics.java class from Internet. Accessed August 20, 2001.

Loewer, O.J., E.M. Smith, G. Benock, T.C. Bridges, N. Gay, L. Wells, S. Burgess, L. Springate, and D. Debertin. 1978. A simulation model for assessing alternative strategies of beef production with land, energy and economic constraints. ASAE paper num. 78-5025, ASAE summer meeting, Logan Utah.

Lunstra, D. D. 1988. Heritability estimates and adjustment factors for the effects of bull age and age of damon yearling testicular size in breeds of bulls. Theriogenology. 30:127.

Lush, J.L., W.F. Lamoreux, and L.N. Hazel.1948. The heritability of resistance to death in fowl. Poultry Sci. 27:375-388.

Mahabir, S. 2003. Evaluation of the method R procedure for one-way random effects models. Ph. D. Diss., Colorado State Univ., Fort Collins.

Mallinckrodt, C.H., B.L. Golden, and A. Reverter. 1997. Approximate confidence intervals for heritability from Method R estimates. J. Anim. Sci. 75:2041-2046.

Martin, L.C., J.S. Brinks, R.M. Bourdon, and L.V. Cundiff. 1992. Genetic effects on beef heifer puberty and subsequent reproduction. J. Anim. Sci. 70:4006.

Melton, B. E., 1995. Conception to consumption: The economics of genetic improvement. Proc. Beef Improvement Federation 27th Research Symposium and Annual Meeting.

Meszaros, S.A. 1999. Optimizing the objective and design of breeding programs with the use of genetic algorithms.

Meuwissen, T. H. E., and Z. Luo. 1992. Computing inbreeding coefficients in large populations. Genet. Sel. Evol. 24:305-313.

Moore, G.E. 1965. Cramming more components onto integrated circuits. Electronics, 38:8.

Morris, C. A. 1980. A review of relationships between aspects of reproduction in beef heifers and their lifetime production. 1. Associations with fertility in the first joining season and with age at first joining. Anim. Breeding Abstr. 48:655.

Morris, C.A., G.L. Bennett, N.G. Cullen, S.M. Hickey, and J.C. Hunter. 2000. Genetic parameters for growth, puberty, and beef cow reproductive traits in a puberty selection experiment. N. Z. J. Agric. Research. 43:83-91.

Naazie, A., M. Makarechian, and R.J. Hudson. 1997. Efficiency of beef production systems: Description and preliminary evaluation of a model. Agric. Sys. 54:357.

Nazzie, A.M., M. Makarechian, and R.J. Hudson. 1999. Evaluation of life-cycle herd efficiency in cow-calf systems of beef production. J. Anim. Sci. 77:1.

Nelsen, T. C., R. E. Short, D. A. Phelps, and R. B. Staigmiller. 1985. Nonpubertal estrus and mature cow influences on growth and puberty in heifers. J. Anim. Sci. 618:470.

Notter, D.R. 1977. Simulated efficiency of beef production for a cow-calf feedlot management system. Ph. D. dissertation, University of Nebraska, Lincoln.

Notter, D.R., J.O. Sanders, G.E. Dickenson, G.M. Smith, and T.C. Cartwright. 1979. Simulated efficiency of beef production for a midwestern cow-calf-feedlot management system. I. Milk production. J. Anim. Sci. 49:70.

Pleasants, A. B. 1997. Use of a stochastic model of a calving distribution for beef cows for formulating optimal natural mating strategies. Anim. Sci. 64:413.

Randel, R.D. 1990. Nutrition and postpartum rebreeding in cattle. J. Anim. Sci. 68:853.

Rao, S. 1997. Genetic analysis of sheep discrete reproductive traits using simulation and field data. Ph.D. dissertation. Virginia Polytechnic Institute and State University, Blacksburg.

Reksen, O., A. Tverdal, K. Landsverk, E. Kommisrud, K.E. Boe, and E. Ropstad. 1999. Effects of photointensity and photoperiod on milk yield and reproduction performance of Norwegian Red cattle. J. Dairy Sci. 82:810.

Reverter, A., B.L. Golden, R.M. Bourdon, and J.S. Brinks. 1994. Method R variance components procedure: Application on the simple breeding value model. J. Anim. Sci. 72:2247.

Rhodes, F. M., K.W. Entwistle, and J.E. Kinder. 1996. Changes in ovarian function and gonadotrophin secretion preceding the onset of nutritionally induced anestrous in *Bos Indicus* heifers. Biol. Reprod. 55:1437.

Roberson, M.S., R.P. Ansotegui, J.G. Berardinelli, R.W. Whitman and M.J. McInerney. 1987. Influence of biostimulation by mature bulls on occurance of puberty in beef heifers. J. Anim. Sci. 64:1601.

Rust, T., and E. Groeneveld. 2001. Variance component estimation on female fertility traits in beef cattle. So. African J. Anim. Sci. 31:131.

Rutter, L. M., and R. D. Randal. 1986. Nonpubertal estrus in beef heifers. J. Anim. Sci. 63:1049.

Sanders, J.O., and T.C. Cartwright. 1979a. A general cattle production systems model. I. Structure of the model. Agric. Sys. 4:217.

Sanders, J.O., and T.C. Cartwright. 1979b. A general cattle production systems model. Part 2 - Proceedures used for simulating animal performance. Agric. Sys. 4:289.

Schafer, D. W. 1991. Within-herd predicted breeding value comparisons and genetic trend in Red Angus cattle. Ph.D. Diss. Colorado State University, Fort Collins.

Schillo, K.K., P.J. Hansen, L.A. Kamwanja, D.J. Dierschke, and E.R. Hauser. 1983. Influence of season on sexual development in heifers: age at puberty as related to growth and serum concentrations of gonadotropins, prolactin, thyroxine, and progesterone. Biol. Reprod. 28:329.

Schillo, K.K., J.B. Hall, and S.M. Hileman. 1992. Effects of nutrition and season on the onset of puberty in the beef heifer. J. Anim. Sci. 70:3994-4005.

Short, R.E., R.A. Bellows. 1971. Relationships among weight gains, age at puberty and reproductive performance in heifers. J. Anim. Sci. 32:127.

Short, R.E., R.A. Bellows, E.L. Moody, and B.E. Howland. 1972. Effects of suckling and mastectomy on bovine postpartum reproduction. J. Anim. Sci. 34:70.

Short, R.E., R.A. Bellows, R.B. Staigmiller, J.G. Berardinelli, and E.E. Custer. 1990. Physiological mechanisms controlling anestrus and infertility in postpartum beef

cattle. J. Anim. Sci. 68:799.

Short, R. E., R. B. Staigmiller, R. A. Bellows, and R. C. Greer. 1994. Breeding heifers at one year of age: Biological and economic considerations. Page 55 in Factors Affecting Calf Crop. M. J. Fields and R. S. Sand, ed. CRC Press, Boca Raton, FL.

Smith, 1979. The TAMU beef cattle production model. Texas Agri. Exp. Sta. Project H2101.

Snelling, W. M. 1994. Genetic analyses of binary stayability measures of beef female. Ph.D. Dissertation. Colorado State Univ., Fort Collins.

Snelling, W.M., B.L. Golden, and R.M. Bourdon. 1995. Within-herd genetic analyses of stayability of beef females. J. Anim. Sci. 73:993.

Snelling, W.M., M.D. MacNeil, and B.L. Golden. 1996. Application of continuous and binary trait methods to reproductive measure of Hereford cattle. J. Anim. Sci. (Abstract) 74 suppl. 1:115.

Splan, R. K., L. V. Cundiff and L. D. Van Vleck. 1998. Genetic correlations between male carcass and female growth and reproductive traits in beef cattle. 6th World Cong. Genet. Appl. Livest. Prod., Armidale, Australia 23:274.

Steel, R.G.D., and J.H. Torrie. 1980. Principles and Proceedures of statistics. McGraw Hill Publishing Company, New York.

Taylor, St. C. S., A. J. Moore, R. B. Thiessen, and C. M. Bailey. 1985. Efficiency of food utilization in traditional and sex-controlled systems of beef production. Anim. Prod. 40:401.

Tess, M.W., and B.W. Kolstad. 2000a. Simulation of cow-calf production systems in range environment: I. Model development. J. Anim. Sci. 78:1159.

Tess, M.W., and B.W. Kolstad. 2000b. Simulation of cow-calf production systems in range environment: II. Model evaluation. J. Anim. Sci. 78:1170.

Tortonese, D.J., and E.K. Inskeep. 1992. Effects of melatonin treatment on the attainment of puberty in heifers. J. Anim. Sci. 70:2822-2827.

Verrill, S. 1996. Cholesky.java. Available: http://www1.fpl.fs.fed.us/Cholesky.html. Accessed May 21, 2001.

Wetterman, R.P., E.J. Turman, R.D. Wyatt, and R. Totusek. 1978. Influence of suckling

intensity on reproductive performance of range cows. J. Anim. Sci. 47:342.

Wheat, J.D., and J.K. Riggs. 1958. Heritability and repeatibility of gestation length in beef cattle. Texas Ag Exp Station paper TA-2536.

Willham, R.L. 1973. Beef breeding programs. Beef Cattle Sci. Handbook 10:193-199.

Willham, R.L., and G. Thompson. 1970. Instructions for use of beef cattle simulation program. Iowa State Univ., Ames, IA (Mimeo).

Williams, G.L. 1990. Suckling as a regulator of postpartum rebreeding in cattle: A review. J. Anim. Sci. 68:831.

Yourdon, E. 1975. Techniques of Program Structure and Design, Prentice-Hall, Englewood Cliffs, New Jersey.

Zalesky, D.D., M.L. Day, M. Garcia-Winder, K. Imakawa, R.J. Kittok, M.J. D'Occhio, and J.E. Kinder. 1984. Influence of exposure to bulls on resumption of estrous cycles following parturition in beef cows. J. Anim. Sci. 59:1135.