Conference on Applied Statistics in Agriculture | 2006 - 18th Annual Conference Proceedings

EVALUATING NONLINEAR CROSSED RANDOM EFFECTS MODELS FOR COMPARING TEMPERATURE OF FEEDING PIGS UNDER DIFFERENT THERMAL ENVIRONMENTS

M. Zhou

A. M. Parkhurst

R. A. Eigenberg

J. A. Nienaber

G. L. Hahn

Recommended Citation

Zhou, M.; Parkhurst, A. M.; Eigenberg, R. A.; Nienaber, J. A.; and Hahn, G. L. (2006). "EVALUATING NONLINEAR CROSSED RANDOM EFFECTS MODELS FOR COMPARING TEMPERATURE OF FEEDING PIGS UNDER DIFFERENT THERMAL ENVIRONMENTS," *Conference on Applied Statistics in Agriculture*. https://doi.org/10.4148/2475-7772.1126

EVALUATING NONLINEAR CROSSED RANDOM EFFECTS MODELS FOR COMPARING TEMPERATURE OF FEEDING PIGS UNDER DIFFERENT THERMAL ENVIRONMENTS

M. Zhou[1], A.M. Parkhurst[1], R.A. Eigenberg[2], J.A. Nienaber[2], G.L. Hahn[2]

1. Department of Statistics, University of Nebraska at Lincoln
2. U.S. Meat Animal Research Center

ABSTRACT

The thermal environment plays a large role in an animal's ability to convert feed into weight gain. A better understanding of a pig's metabolism will help swine producers select environmental specifications for optimizing feed conversion. The objectives of this study are to 1) characterize the thermoregulatory responses of pigs during a feeding event 2) compare those responses for three thermal environmental treatments applied in a Latin Square design 3) investigate different procedures for fitting nonlinear mixed-effect models with crossed random effects (NLME function in R, %NLINMIX macro in SAS, random effects modeling in AD Model Builder: ADMB-RE). We found that the three-parameter first-order compartment model provides a reasonable representation of the tympanic temperatures of feeding pigs during feeding events. The thermal environmental treatments (28ºC + High air speed) and (18ºC + Low air speed) are significantly different from the reference treatment (28ºC + Low air speed), at the 5% level. Both NLME and ADMB-RE successfully fit the nonlinear mixed-effects model and produce similar results. The %NLINMIX macro did not converge unless restrictions were placed on the model. The estimates of fixed and random effects from the restricted model using %NLINMIX macro were generally different from those from NLME and ADMB-RE.

1. INTRODUCTION

The well-being of a meat producing animal is considered to be linked to its ability to convert feed to weight gain. The thermal environment of the animal is of interest to animal producers and to researchers because it plays a large role in the animal's feeding efficiency. Previous work by Hahn et al. (1990) suggests tympanic temperature provides valuable insight into an animal's response to the thermal environment. By using tympanic temperature time series data we can estimate an animal's dynamic overall heat transfer coefficients, such as the temperature growth rate constant and the temperature decay rate constant, and help producers define an optimum range for the thermal environment so that they can adjust their production facilities to the environment best suited to enhance an animal's well being and feed efficiency.

There are three objectives for this study. First, we fit a three-parameter first-order compartment model to characterize the thermoregulatory responses such as the initial tympanic temperature, the temperature growth rate constant, and the temperature decay rate constant of pigs during an feeding event. Second, we compare those responses for

three thermal environmental treatments (28°C air temperature and low air speed, 28°C air temperature and high air speed, and 18°C air temperature and low air speed) applied in a Latin Square design. Finally, we investigate three procedures for fitting nonlinear mixed-effect models with crossed random effects. They are: NLME function in R, %NLINMIX macro in SAS, and random effects modeling in AD Model Builder (ADMB-RE). Currently, fitting nonlinear mixed-effects models with crossed random effects is still a difficult statistical problem. Although a number of software packages have been developed to fit nonlinear mixed models and generalized linear mixed models, most of them such as the SAS NLMIXED procedure (SAS, 1999), NONMEM (Beal and Sheiner, 1992) and the MIXOR family programs (Hedeker and Gibbons, 1996) allow **ONLY ONE** random statement, which limits them to single-level nonlinear mixed models and/or generalized linear mixed models without nested and crossed random effects. There are drawbacks to the existing statistical software packages which can handle multilevel nonlinear mixed models. We consider the above three procedures for different reasons: 1) Both R and SAS are widely used statistical software packages. The NLME function in R and %NLINMIX macro in SAS use the first-order Taylor series expansion to solve nonlinear mixed-effects models, 2) ADMB-RE is a newer, less widely used statistical package based on the second-order Taylor series expansion. Therefore, it is expected to produce more accurate estimates. This paper will provide ways to use these procedures to fit nonlinear mixed-effects models with crossed random effects and discuss the advantages and disadvantages of each procedure.

2. MATERIALS AND METHODS

2.a Data

Eigenberg (1994) conducted an experiment to study the tympanic temperature of feeding pigs in response to three predefined thermal conditions. The experiment was designed as a Latin Square with three treatments, three pigs, and three treatment periods that are about three days in length. The treatments consisted of three combinations of temperature and air speed. For the reference environment, treatment 1, the ambient temperature was set to 28 C and air speed was set to low (20 cm/s). Pigs housed in this environment are expected to be at rest for much of the time, and thus, generate a relatively stable temperature record. For treatment 2, the air temperature was set to 28 C and air speed was set to high (90 cm/s). Treatment 3 completes the treatment group with air temperature set to 18 C and air speed set to low (20 cm/s). Both treatments 2 and 3 would be expected to produce higher thermal loads on the pig than treatment 1. Treatment 2 has higher convective loss and treatment 3 has higher loss due to lower temperature. Six pigs were randomly selected from eleven litters and they were split into two weight ranges: three heavy animals (29.5±1.8 kg) and three light animals (22.5±1.0 kg). The heavier animals were exposed to the treatments first, then the lighter animals. Each weight group was repeated once producing a total of four Latin Squares (two with heavy animals and two with light animals). During the experiment, each pig had the opportunity to eat approximately three meals every day for three days and each of the meals had the potential to produce one set of thermal index values such as the initial tympanic temperature, the temperature growth rate constant, and the temperature decay

rate constant. The access to feed was controlled by solenoid latches on the feeding system. The pigs had access to feed only three times per day for a one-hour period. The meal times were: 3:00 AM, 8:30 AM and 3:00 PM. The tympanic temperature and feed intake of each pig were recorded every 48 seconds. An example showing changes in tympanic temperature and feed intake is presented in Figure 1. In this example, pig #27 (a member of the heavy group) was observed during the first experimental period. The treatment was #2: 28ºC + High air speed. During this period, there were six feeding events and each feeding event produced a tympanic temperature spike.

This study will focus on tympanic temperature data for one Latin Square: first run of heavy group (Table 1). Only first feeding events of the largest meal on the second and third days of each period were included in the study. For each feeding event, the temperature record is analyzed for a record length of 2 hours. In total, there are 18 feeding events in the study (Figure 2). Two events for each combination of pig and period: one for the second day and one for the third day of each period.

2.b Statistical Model

Compartment models are nonlinear models in which the response is described by a linear system of ordinary differential equations. Compartment models have been widely used in the literature. For examples see: Bates and Watts (1988, Ch. 5), Davidian and Giltinan (1995, Ch. 9), Lindsey (1999, Ch. 6) and Pinheiro and Bates (2000, Ch. 6 and Ch. 8). We used a three-parameter first-order compartment model to fit the temperature of pigs during feeding events. The model is given by

$$Y = Y0 + \frac{KG}{KG - KD}(e^{-KD \cdot X} - e^{-KG \cdot X}) + \varepsilon, \quad \varepsilon \sim \text{iid}N(0, \sigma^2), \qquad \text{Eq. 1}$$

where the response variable, Y, is the tympanic (inner ear) temperature (ºC), and the independent variable, X, is the time (fraction of day). There are three parameters in the model: Y0 is the initial tympanic temperature (ºC), KG is the temperature growth rate constant (day^{-1}), and KD is the temperature decay rate constant (day^{-1}). The temperature growth rate constant KG is a measure of the rate of increase in the body temperature proportional to the temperature produced by the feeding event; while the temperature decay rate constant KD is the rate of decrease in the body temperature proportional to the body temperature. The larger KG, the faster the body temperature approaches its maximum; the larger KD, the faster the body temperature goes back to its initial value.

For the nonlinear mixed-effects model with crossed random effects, three treatment levels and three random effects were incorporated in the three-parameter first-order compartment model (Eq. 1) for each of the three parameters:

$$Y0 = \beta_{11} + \beta_{12} \cdot C_1 + \beta_{13} \cdot C_2 + b_{PIGi1} + b_{PERj1} + b_{EVTk1},$$
$$KG = \beta_{21} + \beta_{22} \cdot C_1 + \beta_{23} \cdot C_2 + b_{PIGi2} + b_{PERj2} + b_{EVTk2},$$
$$KD = \beta_{31} + \beta_{32} \cdot C_1 + \beta_{33} \cdot C_2 + b_{PIGi3} + b_{PERj3} + b_{EVTk3},$$
$$b_{PIGi} = \begin{bmatrix} b_{PIGi1} \\ b_{PIGi2} \\ b_{PIGi3} \end{bmatrix} \sim iidN(0, D_{PIG}), where D_{PIG} = \begin{bmatrix} \sigma^2_{PIG1} & 0 & 0 \\ & \sigma^2_{PIG2} & 0 \\ & & \sigma^2_{PIG3} \end{bmatrix}, i = 1,...,3,$$
$$b_{PERj} = \begin{bmatrix} b_{PERj1} \\ b_{PERj2} \\ b_{PERj3} \end{bmatrix} \sim iidN(0, D_{PER}), where D_{PER} = \begin{bmatrix} \sigma^2_{PER1} & 0 & 0 \\ & \sigma^2_{PER2} & 0 \\ & & \sigma^2_{PER3} \end{bmatrix}, j = 1,...,3,$$
$$b_{EVTk} = \begin{bmatrix} b_{EVTk1} \\ b_{EVTk2} \\ b_{EVTk3} \end{bmatrix} \sim iidN(0, D_{EVT}), where D_{EVT} = \begin{bmatrix} \sigma^2_{EVT1} & \sigma_{EVT12} & \sigma_{EVT13} \\ & \sigma^2_{EVT2} & \sigma_{EVT23} \\ & & \sigma^2_{EVT3} \end{bmatrix}, k = 1,...,18,$$
$$G = \begin{bmatrix} D_{PIG} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & D_{PIG} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & D_{PIG} & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & D_{PER} & 0 & 0 & 0 & 0 & 0 \\ & & & & D_{PER} & 0 & 0 & 0 & 0 \\ & & & & & D_{PER} & 0 & 0 & 0 \\ & & & & & & D_{EVT} & 0 & 0 \\ & & & & & & & \ddots & 0 \\ & & & & & & & & D_{EVT} \end{bmatrix}. \quad \text{Eq. 2}$$

where C_1 and C_2 are two dummy variables defined to incorporate the treatment effects. For the first treatment, the reference level, we let both C_1 and C_2 be zero. To assess the difference between treatments 2 and 3 and the reference level, we let C_1 be 1 and C_2 be 0 , for the second treatment; and for the third treatment, we let C_1 be 0 and C_2 be 1. Under this setting, the meaning of those fixed-effect coefficients is as follows: β_{11}, β_{21}, and β_{31} are the means of the first treatment for Y0, KG, and KD respectively; β_{12}, β_{22}, and β_{32} are the differences of the means between the second treatment and first treatment for Y0, KG, and KD respectively; and β_{13}, β_{23}, and β_{33} are the differences of the means between the third and first treatments for Y0, KG, and KD respectively. The random effects b_{PIGi1}, b_{PIGi2}, and b_{PIGi3} represent the deviation from the population mean associated with the i^{th} pig. The random effects b_{PERj1}, b_{PERj2}, and b_{PERj3} represent the deviations associated with the j^{th} period. The random effects b_{EVTk1}, b_{EVTk2}, and b_{EVTk3} represent the deviations associated with the k^{th} feeding event. D_{PIG}, D_{PER}, and D_{EVT} are the variance-covariance matrices for the random effects. We assume D_{PIG} and D_{PER} are diagonal matrices. Since we only have three pigs and three periods, we do not have enough information to estimate all the elements in the two matrices. We can assume D_{EVT} is an unstructured matrix since

we have 18 feeding events. We further assume that b_{PIGi}, b_{PERj}, b_{EVTk}, and ε are independent of each other. The diagonal matrix G is the variance-covariance matrices for all the random effects associated with the pig, the period, and the feeding event.

2.c Crossed Random Effects

In our study, there are three pigs, three periods, and eighteen feeding events with two feeding events in each pig-period combination (Table 2). First, let us look at the row for a specific pig, say pig 85. It contains six feeding events: Event 1, Event 2, Event 3, Event 4, Event 5, and Event 6. These six feeding events are from all three periods. Now let us look at different feeding events from any column, say the column for Period 1: Event 1, Event 2, Event 7, Event 8, Event 13, and Event 14. These six feeding events are from all three pigs. Thus, Pig and Period are crossed with each other and Feeding Event is nested within the combination of Pig and Period. The random effects associated with Pig and Period are called crossed random effects.

2.d Procedures for Fitting Nonlinear Mixed Models with Crossed Random Effects

We consider three procedures for fitting a nonlinear mixed-effect model with crossed random effects in the study. They are: the NLME function in R, %NLINMIX macro in SAS, and random effects modeling in AD Model Builder (ADMB-RE).

2.d.i Random Effects Modeling in R: NLME

R was initially written by Robert Gentleman and Ross Ihaka of the Statistics Department of the University of Auckland. It provides a suite of software facilities using programming principles of S, a language for manipulating objects, and a platform for new algorithms which can call functions written in R, C, C++ and Fortran. The NLME package fits nonlinear mixed-effects models for Gaussian outcome variables using first-order Taylor series expansion approximation. It alternates between two steps: 1) penalized nonlinear least square and 2) linear-mixed-effects, until the process converges. To use the package, users need to know the basic syntax structure of NLME. It helps to study examples from documentation on the Web or textbooks such as the one written by Pinheiro and Bates (2000). NLME in R is very powerful for fitting nonlinear mixed-effects models with NESTED random effects, but it does not fit nonlinear mixed-effects models with CROSSED random effects. Goldstein (1999) shows how to fit a Linear Mixed Model with crossed random effects as a purely hierarchical formulation of nested random effects. We developed a method to enable NLME in R to fit a Nonlinear Mixed-Effects Model with crossed random effects based on Goldstein's idea for linear mixed-effects model (Goldstein, 1999).

Goldstein's Method

For illustration, we assume a linear mixed-effects model with two crossed grouping factors: A and B, where A has five levels and B has three levels. For efficiency we choose one grouping factor, A, the one with the most levels, as a standard hierarchical level 1 grouping factor. For the other grouping factor, B, we declare a hierarchical level 2 grouping factor with one level that spans the entire data set. Then, we define a dummy (0, 1) variable for each level of B, which is one if the observation belongs to that level of B and zero if not. Finally, we specify that each of these dummy variables has a random

coefficient at hierarchical level 2 and constrain the resulting set of hierarchical level 2 variances to be equal. Then level 1 and level 2 are variances for A and B, respectively. If we have a third grouping factor at level 1, that is, A, B, and C crossed at level 1, we can obtain the third variance by defining a similar set of dummy variables with coefficients varying at level 3 and variances constrained to be equal. This method can be generalized to linear mixed-effects models with arbitrary p-way crossed random effects.

To apply Goldstein's method for fitting nonlinear mixed-effects models with crossed random effects, we need to modify the method to make it suitable for the NLME function. First, no matter how many crossed grouping factors are in the model, only one extra hierarchical grouping factor needs to be declared. Second, the grouping factor is always defined as the highest hierarchical level with one level that spans the entire data set. Third, it is not necessary to define dummy variables for any levels of crossed grouping factors. However, only one correlation for the crossed random effects can be estimated.

The NLME code for fitting the nonlinear mixed-effects model is as follows:

```
# Feeding Event is specified as the lowest hierarchical level grouping factor
  # (hierarchical level 1 grouping factor)

# Specify Pig as a standard hierarchical level 2 grouping factor

# Create a new grouping factor as the highest hierarchical level
  # (hierarchical level 3 grouping factor) with one level spanning the entire data set
newGF <-  factor(rep(1,length(heat$Y)))

# Specify that each level of Period has a coefficient random at newGF
# " -1" indicates that the specific term factor(Period ) does not have an intercept
# Cannot estimate the correlation associated with Period

full.nlme <-
   nlme(model=Y~Y0+KG*(exp(-KD*X)-exp(-KG*X))/(KG-KD),
      fixed=Y0+KG+KD~factor(Trt),
      random=list(newGF=pdIdent(Y0~factor(Period)-1),
                  newGF=pdIdent(KG~factor(Period)-1),
                  newGF=pdIdent(KD~factor(Period)-1),
                  Pig=pdDiag(Y0+KG+KD~1),
                  Event=(Y0+KG+KD~1)),
      start=c(39,0,0,60,0,0,20,0,0), data=heat)
```

2.d.ii %NLINMIX Macro in SAS

The %NLINMIX macro was written by Russell D. Wolfinger (1993) and it fits nonlinear mixed-effects models with both nested and crossed random effects using PROC NLIN and PROC MIXED (SAS, 1999). It is based on linearization methods of estimation. The basic idea behind the linearization method is: 1) Take a first-order Taylor series of the model around some values of fixed effects and random effects; 2) This yields an approximate model that is of the linear mixed model form; 3) Fit this model with a linear mixed model package; 4) Update the expansion loci and repeat the process until a convergence criterion is met.

The %NLINMIX macro fails to converge for the nonlinear mixed-effects model when the variance-covariance matrix for feeding events (D_{EVT}) in Eq. 2 is assumed to be unstructured but converges when we further assume that D_{EVT} is diagonal and change the default expansion locus of the random effect of the macro from its current empirical best linear unbiased predictor (EBLUP) to zero (the expected value of the random effect)..

The %NLINMIX macro code for fitting the nonlinear mixed-effects model with diagonal D_{EVT} in Eq. 2 is shown below:

```
%nlinmix(data=a,
  procopt=convg=1e-7,
  model=%str(
    Y0  = beta11 + beta12*C1 + beta13*C2 + b11 + b12 + b13;
    KG = beta21 + beta22*C1 + beta23*C2 + b21 + b22 + b23;
    KD = beta31 + beta32*C1 + beta33*C2 + b31 + b32 + b33;
    predv = (Y0) + (KG)*(exp(-(KD)*x)-exp(-(KG)*x))/((KG)-(KD));
  ),
  parms=%str(beta11=39 beta12=0 beta13=0  beta21=65 beta22=-3 beta23=-11
             beta31=18 beta32=18  beta33=25),
  stmts=%str(
    class Pig Period Event;
    model pseudo_y = d_beta11 d_beta12 d_beta13 d_beta21 d_beta22 d_beta23
          d_beta31 d_beta32 d_beta33 / noint notest solution cl ddfm=residual;
    random d_b11 d_b21 d_b31 / type=vc subject=Pig      solution cl;
    random d_b12 d_b22 d_b32 / type=vc subject=Period solution cl;
    * Assume diagonal D_EVT in Eq. 2 to make the macro converge;
    random d_b13 d_b23 d_b33 / type=vc subject=Event  solution cl;
  ),
  expand=zero, * change the default expansion locus of the random effect
  converge=1e-6,
  maxit=30,
  options=skipnlin
)
```

2.d.iii Random Effects Modeling in AD Model Builder: ADMB-RE

ADMB-RE (2005) was developed by Otter Research Ltd., Canada. It handles nonlinear mixed-effects models with both nested and crossed random effects. It is based on Laplace approximation. The letters AD represent automatic differentiation, which refers to a collection of techniques that exploit the chain rule of calculus to automatically evaluate derivatives of functions defined in computer programs. To use ADMB-RE, we need to formulate the likelihood function in a template file using a C++ like language and then turn the template file into an executable program using a C++ compiler.

Because the ADMB-RE code for fitting the nonlinear mixed-effects model is lengthy (more than 200 lines), we include it in Appendix A.

3. RESULTS AND DISCUSSION

Model Diagnostics

The standardized residuals, shown on the vertical axis in Figure 3, are the raw residuals, $e = y - f(\hat{\beta}, \hat{b})$, divided by the estimated standard deviation, $\hat{\sigma}$ of the ε. The plot of the standardized residuals versus the fitted values corresponding to the nonlinear mixed-effects model (Figure 3), does not indicate departure from the model assumptions. The assumption of normality for the within-group errors appears reasonable although it suggests the error distribution has lighter tails than expected from normally distributed errors (Figure 4). A final assessment of the adequacy of the nonlinear mixed-effects model is given by the plot of the augmented predictions in Figure 5. From the plot, we can see that the predicted temperatures are close to the observed values. Therefore, we conclude that the nonlinear mixed-effects model provides a reasonable representation of the tympanic temperatures during feeding events.

Evaluation of the Three Procedures for Fitting the Nonlinear Crossed Random Effects Model

Both NLME and ADMB-RE successfully fit the nonlinear mixed-effects model. However, the %NLINMIX macro fails to converge and always stays on the first PROC MIXED call for fitting the nonlinear mixed-effects model. The possible reasons are: 1) G matrix becomes non-positive definite during iterations; and 2) Size of G matrix is large – more than 1300*1300. The %NLINMIX macro converges for fitting the nonlinear mixed-effects model when we further assume that D_{EVT} is diagonal. The results of fitting the nonlinear mixed-effects model for both NLME and ADMB-RE, as well as the nonlinear mixed-effects model with diagonal D_{EVT} for %NLINMIX macro, are shown in Table 3 and Table 4. Estimates of both fixed and random effects from NLME and ADMB-RE are very close to each other. However, estimates of most fixed and random effects from %NLINMIX macro are different from those of NLME and ADMB-RE, especially for fixed effects of β_{21}, β_{22}, and β_{23}, and random effect σ_{PIG3}. A possible reason is that in linearizing a nonlinear mixed model, we need to choose an expansion locus for the fixed effects and the random effects. For the fixed effects, the estimates from the previous iteration are used. However, there are different ways of choosing the expansion locus for the random effects. The default expansion of the random effect in

%NLINMIX macro is about its current empirical best linear unbiased predictor (EBLUP). The estimates produced by this expansion are the same as those produced by NLME in R although it uses a different algorithm. The details can be found in Wolfinger (1993). However to make our nonlinear mixed model converge in %NLINMIX, we needed to change the default. We directed the macro to do the expansion about the expected value of the random effect. (Note: the expected value of any random effect is zero and the expansion about zero might produce inaccurate estimates (Littell et al. 1996).)

Both NLME and %NLINMIX Macro are based on the first-order Taylor series expansion while ADMB-RE is based on the Laplace approximation (ADMB-RE 2005), which uses the second-order Taylor series expansion. Therefore, the estimation from both NLME and %NLINMIX macro is less accurate compared with that from ADMB-RE.

The time to converge of both NLME and %NLINMIX macro is much faster than that of ADMB-RE because of the simpler computation (Pinheiro 1995). For fitting the nonlinear mixed-effects model, the time to converge of both NLME and %NLINMIX macro is less than 3 minutes while the time to converge of ADMB-RE is about 20 minutes.

Writing code in NLME and %NLINMIX macro is straightforward, while writing code in ADMB-RE is very challenging. We need to deal with details of computation to make ADMB-RE code work efficiently. The length of code in both NLME and %NLINMIX macro is much shorter than that in ADMB-RE. For fitting the nonlinear mixed-effects model, the code is about 10 lines in NLME, 20 lines in %NLINMIX macro, and more than 200 lines in ADMB-RE.

Comparison of the Three Thermal Environmental Treatments

We compare the three thermal environmental treatments based on the results from NLME in R (Table 3). We find that both treatment 2 (28ºC + High air speed) and treatment 3 (18ºC + Low air speed) are significantly different from the reference treatment 1 (28ºC + Low air speed). Treatment 2 is significantly different from treatment 1 for the temperature decay rate constant (KD), but not for the initial tympanic temperature (Y0) and temperature growth rate constant (KG), while treatment 3 is significantly different from treatment 1 for both Y0 and KD, but not for KG. From the parameter estimates, we find that both increasing the air speed and decreasing the environmental temperature can help pigs dissipate heat effectively. In comparison with the reference treatment 1, increasing the air speed (treatment 2) increases the temperature decay rate constant by 18.2 day^{-1} while decreasing environmental temperature (treatment 3) increases the temperature decay rate constant by 25.3 day^{-1} and also decreases the initial tympanic temperature of pigs by 0.2°C.

Future work on Model Building for the Complete Dataset

In this study we used a subset of the data to address the problem of crossed random effects in a nonlinear model. Future work will be done on the complete set of Latin Squares, the inclusion of covariates providing information about the amount of feed consumed and duration of feeding event, correlations over time and the possibility of a more parsimonious set of parameters.

4. SUMMARY

This study provides a nonlinear mixed-effects model to describe the thermoregulatory responses of pigs during a feeding event and to compare those responses for three thermal environmental treatments applied in a Latin Square design. It also investigates three different procedures for fitting nonlinear mixed-effect models with crossed random effects: NLME function in R, %NLINMIX macro in SAS, and random effects modeling in AD Model Builder (ADMB-RE). As expected, based on the estimation methods, the ADMB-RE produces more accurate results. However, it is simpler to fit nonlinear mixed effects models with crossed random effects in NLME and %NLINMIX macro, although, the %NLINMIX macro did not converge for the nonlinear mixed-effects model when the variance-covariance matrix for feeding events was assumed to be unstructured .

5. REFERENCES

ADMB-RE. 2005. ADMB-RE User Guide. Otter Research Ltd., Canada.

Bates, D.M. and Watts, D.G. 1988. Nonlinear Regression Analysis and Its Applications. New York: John Wiley and Sons.

Beal, S. L. and Sheiner, L. B. 1992. NONMEM User's Guides. NONMEM Project Group, University of California, San Francisco.

Davidian, M. and Giltinan, D. M. 1995. Nonlinear Models for Repeated Measurement Data. London: Chapman & Hall.

Eigenberg, R.A. 1994. Tympanic Temperature Transient Response as an Index of Heat Dissipation in Swine. Ph.D. thesis, University of Nebraska - Lincoln.

Goldstein, H. 1999. Multilevel Statistical Models. New York: Halstead Press.

Hahn, G.L., Y.R. Chen, J.A. Nienaber, A.M. Parkhurst and R.A. Eigenberg. 1990. Characterizing livestock stress "by the numbers". Journal of Thermal Biology 17(2): 115-120.

Hedeker, D. and Gibbons, R. D. 1996. MIXOR: A Computer Program for Mixed-Effects Ordinal Regression Analysis. *Biometrics* 50: 933-944.

Lindsey, J.K. 1999. Models for Repeated Measurements. Oxford: Oxford University Press.

Littell, R.C., G.A. Milliken., W.W. Stroup and R.D. Wolfinger. 1996. SAS System for Mixed Models. SAS Institute Inc., Cary, NC, USA.

Pinheiro, J.C. and D.M. Bates. 1995. Approximations to the Loglikelihood Function in the Nonlinear Mixed-Effects Model. Journal of Computational and Graphical Statistics 4(1): 12-35.

Pinheiro, J.C. and D.M. Bates. 2000. Mixed-Effects Models in S and S-PLUS. New York: Springer-Verlag.

SAS. 1999. SAS/STAT User' s Guide. Version 8. SAS Institute Inc., Cary, NC.

Wolfinger, R.D. 1993. Laplace's Approximation for Nonlinear Mixed Models. Biometrika 80: 791-795.

Table 1. Latin Square of the First Run for Heavy Group with Three Treatments, Three Pigs, and Three Treatment Periods.

Pig No.	Period 1	Period 2	Period 3
85	Treatment 2	Treatment 1	Treatment 3
27	Treatment 3	Treatment 2	Treatment 1
59	Treatment 1	Treatment 3	Treatment 2

Table 2. Illustration of Crossed Random Effects.

Pig No.	Period 1	Period 2	Period 3
85	Event 1, Event 2	Event 3, Event 4	Event 5, Event 6
27	Event 7, Event 8	Event 9, Event 10	Event 11, Event 12
59	Event 13, Event 14	Event 15, Event 16	Event 17, Event 18

3. Estimates of Fixed Effects Coefficients for Compartmental Model from the Three Procedures.

Fixed Effects		NLME			%NLINMIX Macro		ADMB-RE	
		Estimate	Std. Err.	P-value	Estimate	Std. Err.	Estimate	Std. Err.
Y0	β_{11}	39.060	0.050	<0.001	39.142	0.054	39.063	0.117
	β_{12}	-0.040	0.070	.571	-0.108	0.057	-0.042	0.069
	β_{13}	-0.199	0.070	.005	-0.254	0.057	-0.201	0.069
KG	β_{21}	65.323	19.700	.001	38.868	22.560	63.929	18.970
	β_{22}	-3.296	9.524	.729	16.222	12.635	-1.252	10.192
	β_{23}	-11.099	9.441	.240	6.789	12.596	-9.489	10.184
KD	β_{31}	18.862	4.766	<.001	28.865	7.753	19.174	4.721
	β_{32}	18.160	6.623	.006	9.748	7.070	18.121	6.525
	β_{33}	25.262	6.704	<.001	19.659	7.176	25.373	6.626

4. Estimates of Random Effects (Standard Deviation and Correlation) for Compartmental Model from the Three Procedures.

Random Effects		NLME		%NLINMIX Macro		ADMB-RE	
		Estimate	Std. Err.	Estimate	Std. Err.	Estimate	Std. Err.
PIG	σ_{PIG1}	< 0.001	NA	0.061	NA	0.021	0.063
	σ_{PIG2}	28.700	NA	32.439	NA	28.365	12.466
	σ_{PIG3}	2.164	NA	9.773	NA	2.163	4.724
PERIOD	σ_{PER1}	< 0.001	NA	0	NA	< 0.001	< 0.001
	σ_{PER2}	13.672	NA	15.374	NA	10.195	6.788
	σ_{PER3}	0.003	NA	3.421	NA	0.025	1.928
FEEDING EVENT	σ_{EVT1}	0.118	NA	0.096	NA	0.116	0.023
	σ_{EVT2}	14.860	NA	21.569	NA	16.242	3.758
	σ_{EVT3}	10.966	NA	11.764	NA	10.903	2.251
	$\rho_{EVT1EVT2}$	-0.472	NA	NA	NA	-0.415	NA
	$\rho_{EVT1EVT3}$	0.379	NA	NA	NA	0.393	NA
	$\rho_{EVT2EVT3}$	-0.274	NA	NA	NA	-0.115	NA
Residual	σ	0.061	NA	0.060	NA	0.060	0.001

Figure 1. Example of Changes in Tympanic Temperature (ºC) and Feed Intake (kg) of Pigs over Julian calendar time for pig 27 (a member of the heavy group) during first experimental period under treatment 2 (28ºC and High air speed).

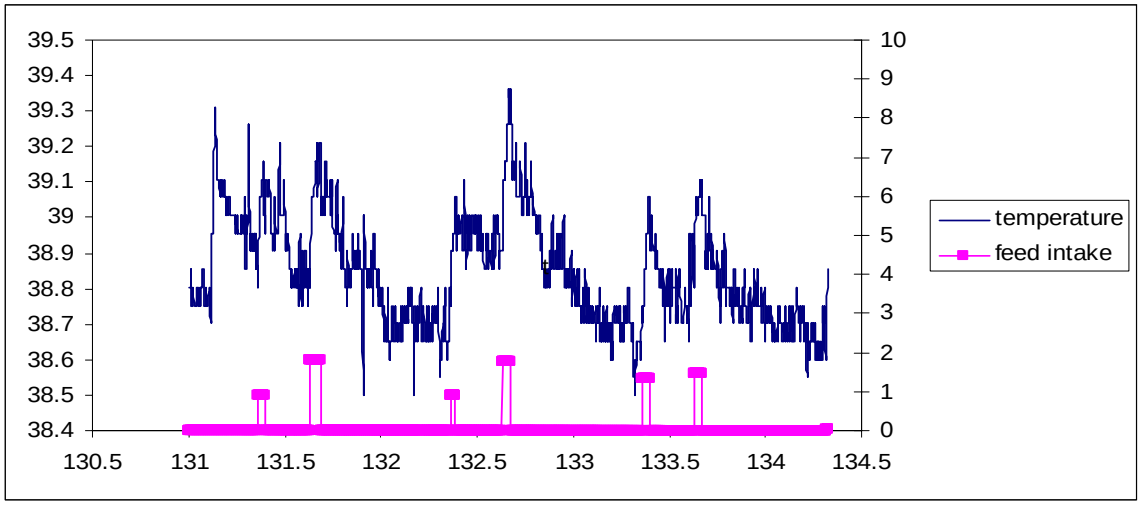


Figure 2. Eighteen Feeding Events of Observed Tympanic Temperature (ºC) versus Time (fraction of day) for Three Pigs and Three Periods.

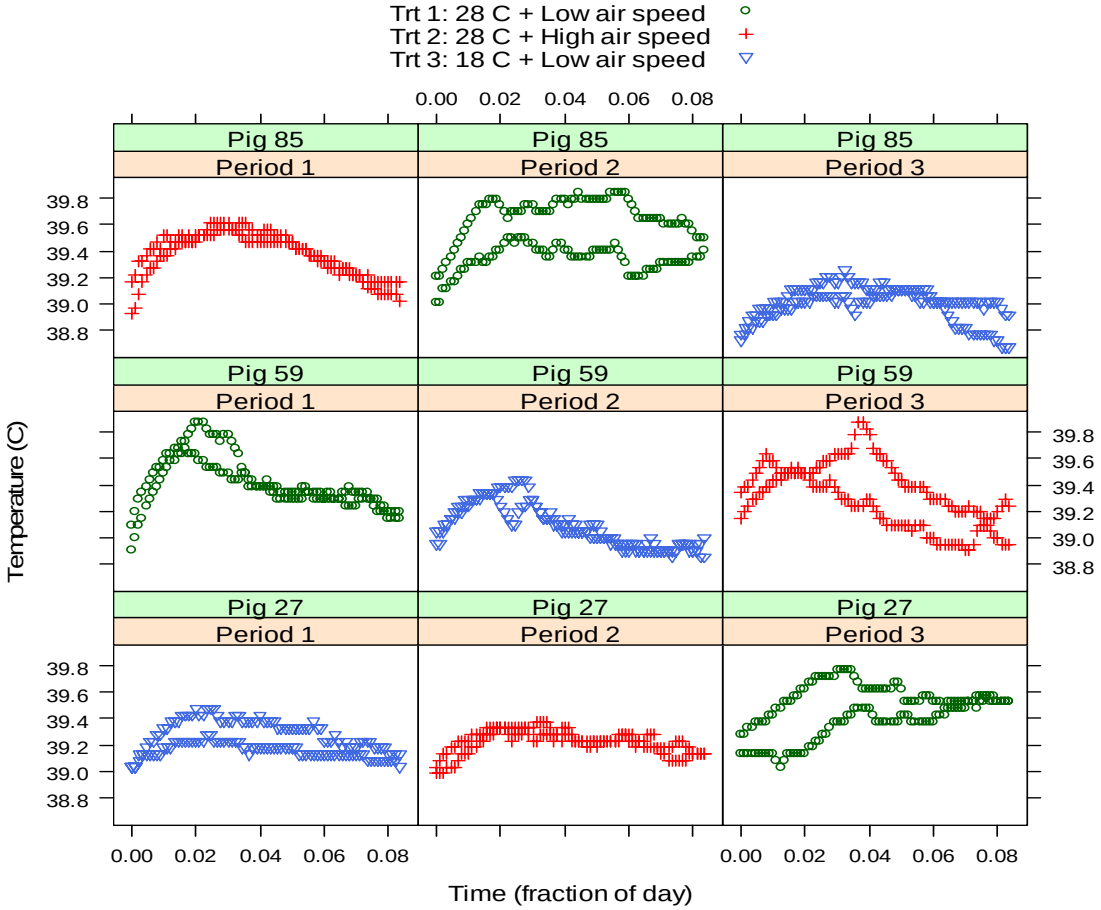


Figure 3. Scatter plot of standardized residuals versus fitted values for the nonlinear mixed-effects model fit.

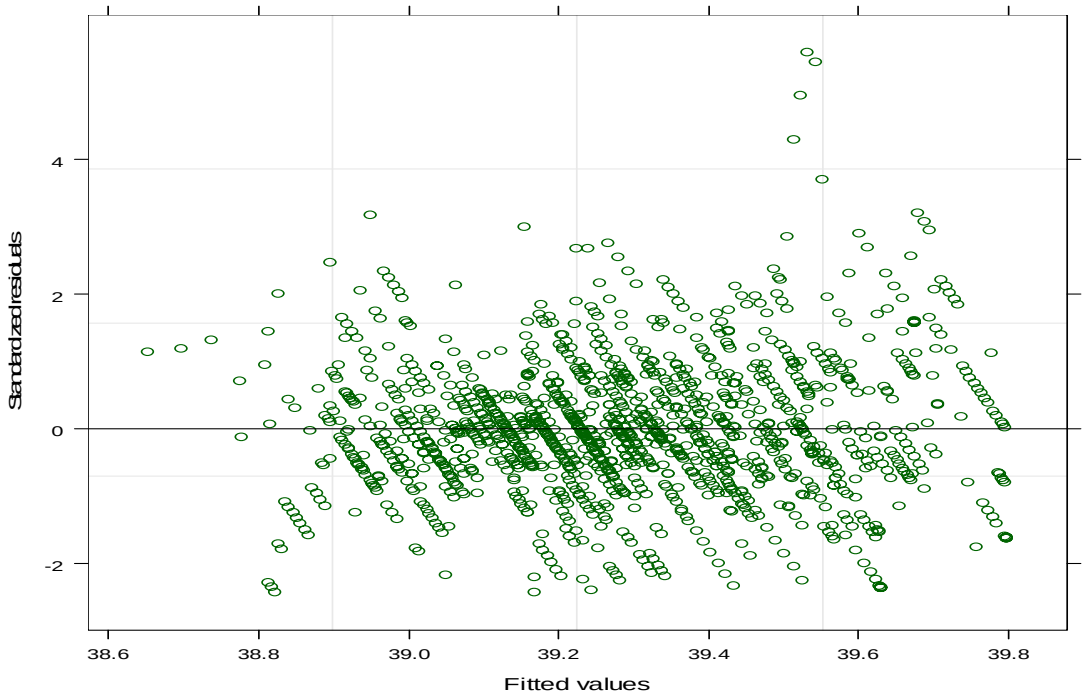


Figure 4. Normal plot of standardized residuals for the nonlinear mixed-effects model fit.

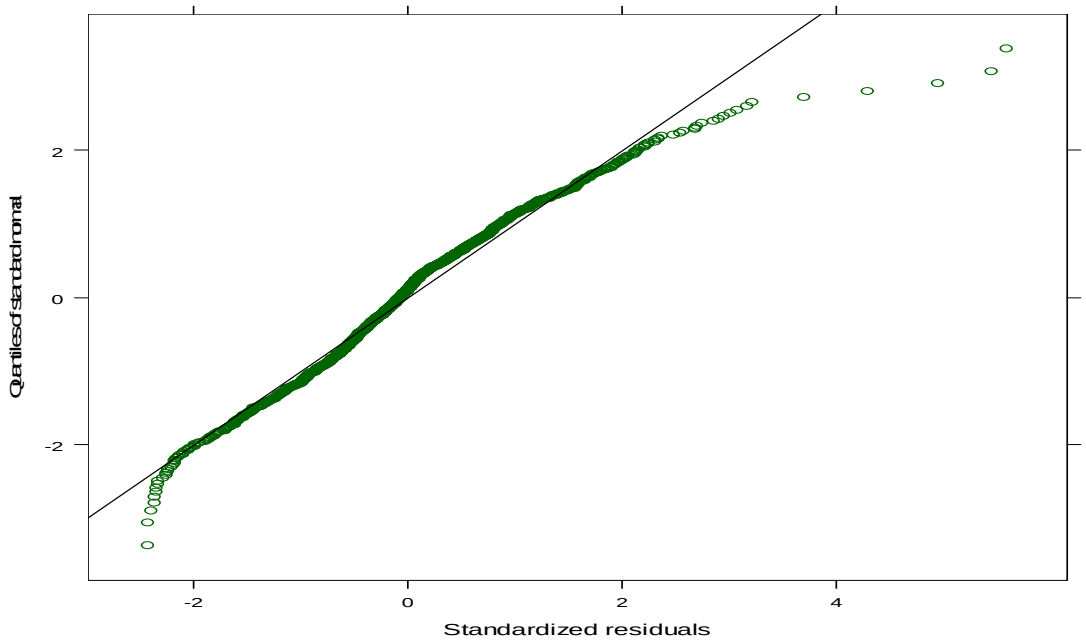
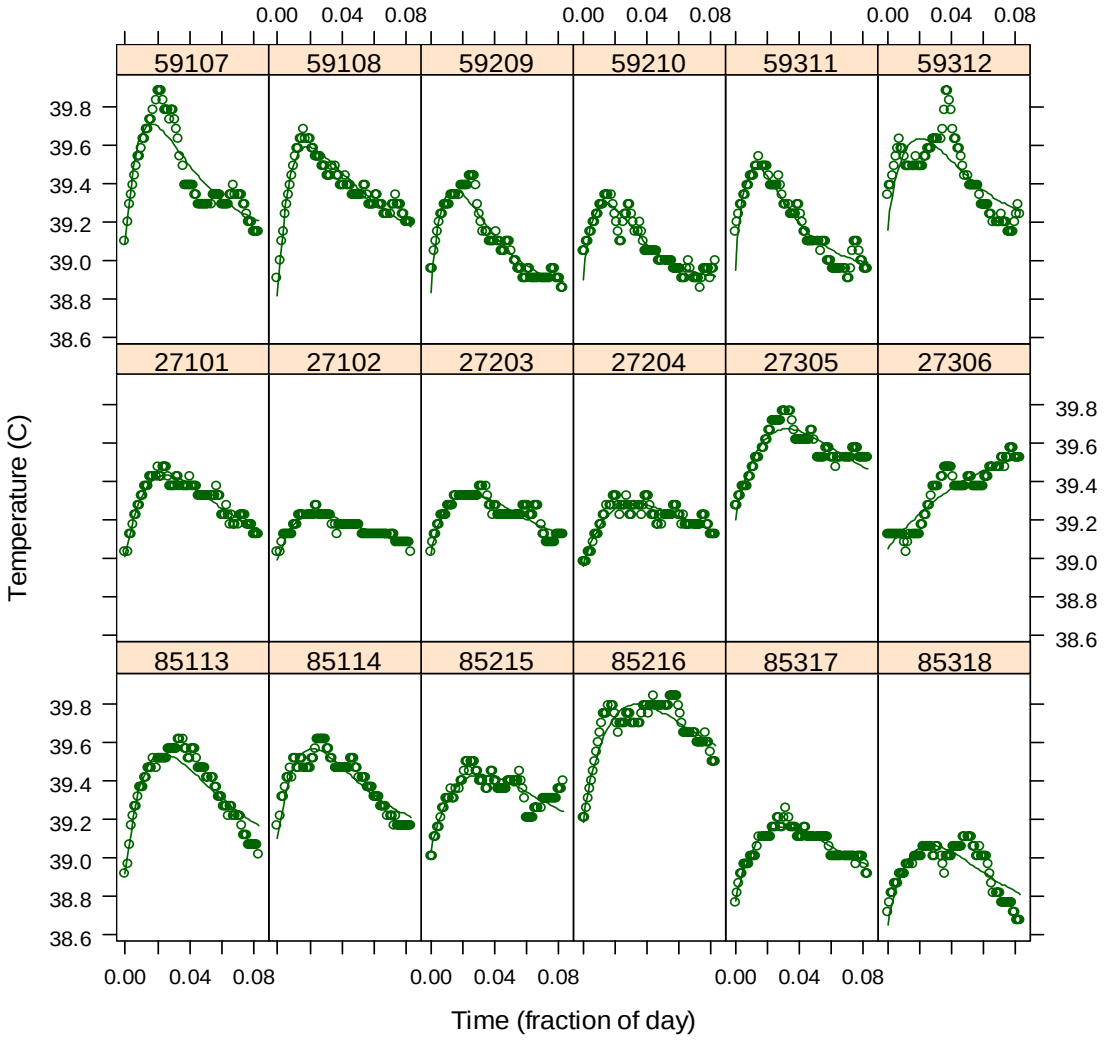


Figure 5. Observed (0) and predicted (--) tympanic temperatures (C) over time (fraction of day) for eighteen pig-period feeding events.



Appendix A. ADMB-RE code for fitting the nonlinear mixed-effects model.

```
DATA_SECTION

 init_int n                    // Number of data points
 init_vector y(1,n)        // Response vector
 init_vector t(1,n)        // Primary covariate
// scale the time to be near 1.0
// !! t/=max(t);
 init_vector Z1(1,n)       // Dummy variable #1
 init_vector Z2(1,n)       // Dummy variable #2
 init_int M1                   // Number of pigs
 init_int M2                   // Number of periods
 init_int M3                   // Number of events
 init_int m                    // Number of parameters in nonlinear regression model

PARAMETER_SECTION

 init_bounded_vector beta(2,9,-100,100,1)                  // Fixed effects parameters
 sdreport_number sigma              // log(residual variance)
 sdreport_number sigma_u1_1;
 sdreport_number sigma_u1_2;
 sdreport_number sigma_u1_3;

 sdreport_number sigma_u2_1;
 sdreport_number sigma_u2_2;
 sdreport_number sigma_u2_3;

 sdreport_vector sigma_u3(1,3);
 sdreport_vector a(1,9);

 init_bounded_number log_sigma_u1_1(-7.0,7.0,2)       // 0.5*log(variance component)
 init_bounded_number log_sigma_u1_2(-7.0,7.0,2)       // 0.5*log(variance component)
 init_bounded_number log_sigma_u1_3(-7.0,7.0,2)       // 0.5*log(variance component)
 init_bounded_number log_sigma_u2_1(-7.0,7.0,3)       // 0.5*log(variance component)
 init_bounded_number log_sigma_u2_2(-7.0,7.0,3)       // 0.5*log(variance component)
 init_bounded_number log_sigma_u2_3(-7.0,7.0,3)       // 0.5*log(variance component)
 init_bounded_vector log_sigma_u3(1,3,-7.0,7.0,4)     // 0.5*log(variance component)
 init_bounded_number alpha(0.5,1.5)
 init_bounded_vector u3_corr(1,3,-10.0,10.0,5)

 random_effects_vector u1_1(1,M1,2)          // Unscaled random effects
 random_effects_vector u1_2(1,M1,2)          // Unscaled random effects
 random_effects_vector u1_3(1,M1,2)          // Unscaled random effects
 random_effects_vector u2_1(1,M2,3)          // Unscaled random effects
 random_effects_vector u2_2(1,M2,3)          // Unscaled random effects
```

```
  random_effects_vector u2_3(1,M2,3)          // Unscaled random effects
  random_effects_matrix u3(1,3,1,M3,4)        // Unscaled random effects

  objective_function_value g

PRELIMINARY_CALCS_SECTION
  cout << setprecision(4);

GLOBALS_SECTION

  #include <df1b2fun.h>
  //#include <fvar.hpp>

PROCEDURE_SECTION

  const double lstp=0.91893853320467274177;
  int i,ii,j,k,l;

  dvariable tmp, f;

  //a(1) = 39.0+beta(1);
  a(2) = 0.0+beta(2);
  a(3) = 0.0+beta(3);
  a(4) = 65.0+beta(4);
  a(5) = -3.0+beta(5);
  a(6) = -11.0+beta(6);
  a(7) = 18.0+beta(7);
  a(8) = 18.0+beta(8);
  a(9) = 25.0+beta(9);

  g = 0.0;

  ii = 0;

  sigma_u1_1=mfexp(log_sigma_u1_1);
  sigma_u1_2=mfexp(log_sigma_u1_2);
  sigma_u1_3=mfexp(log_sigma_u1_3);

  sigma_u2_1=mfexp(log_sigma_u2_1);
  sigma_u2_2=mfexp(log_sigma_u2_2);
  sigma_u2_3=mfexp(log_sigma_u2_3);

  sigma_u3=mfexp(log_sigma_u3);

  dvar_vector su1_1=sigma_u1_1*u1_1;
  dvar_vector su1_2=sigma_u1_2*u1_2;
```

```
dvar_vector su1_3=sigma_u1_3*u1_3;

dvar_vector su2_1=sigma_u2_1*u2_1;
dvar_vector su2_2=sigma_u2_2*u2_2;
dvar_vector su2_3=sigma_u2_3*u2_3;

dvar_matrix CHD(1,3,1,3);
CHD.initialize();
for (i=1;i<=3;i++)
{
 CHD(i,i)=1;
}
CHD(2)(1,1)=u3_corr(1);
CHD(2)/=norm(CHD(2));

CHD(3)(1,2)=u3_corr(2,3).shift(1);
CHD(3)/=norm(CHD(3));
for (i=1;i<=3;i++)
{
 CHD(i)*=sigma_u3(i);
}

dvar_matrix su3=CHD*u3;

dvar_vector su3_1=su3(1);
dvar_vector su3_2=su3(2);
dvar_vector su3_3=su3(3);

dvariable r2=0.0;

dvar_vector pred0(1,n);
for(k=1;k<=M3;k++)
{
 for(l=1;l<=(n/M3);l++)
 {
  i = (k-1)/6+1;
  j = (k-1)%3+1;
  ii++;

  // get rid of a(1)
  dvariable A=a(2)*Z1(ii)+a(3)*Z2(ii)+su1_1(i)+su2_1(j)+su3_1(k);
  dvariable B=a(4)+a(5)*Z1(ii)+a(6)*Z2(ii)+su1_2(i)+su2_2(j)+su3_2(k);
  dvariable C=a(7)+a(8)*Z1(ii)+a(9)*Z2(ii)+su1_3(i)+su2_3(j)+su3_3(k);

  pred0(ii) = A+B*((mfexp(-C*t(ii))-mfexp(-B*t(ii)))/(B-C));
```

```
  }
}
dvariable mp0=mean(pred0);
double my=mean(y);
// this is the maximum likelihood estimate for a(1)
// which can be solved for explicitly like this so it can be removed
// from the optimization
a(1) = my-mp0;
// so the sum square residulas look like
// norm2(y-my-pred0+mp0);

r2=norm2(y-my-pred0+mp0);
if (ii != n)
{
  cerr << " bad " << endl;
  ad_exit(1);
}
// this is the maximum likelihood estimate for sigma
// which can be solved for explicitly like this so it can be removed
// from the optimization as well
//sigma=sqrt(r2/double(ii));
sigma=sqrt(alpha*r2/double(ii));
// when sigma is equal to its MLE the log-liklihood
// becomes
g+=double(ii)*log(sigma)+0.5*double(ii)/alpha;
g+=double(ii)*lstp;

// a very small penalty so that components with
// estimated 0 variance do not cause the hessian estimate
// to become singular
double eps=1.e-5;

g+=eps*square(log_sigma_u1_1);
g+=eps*square(log_sigma_u1_2);
g+=eps*square(log_sigma_u1_3);

g+=eps*square(log_sigma_u2_1);
g+=eps*square(log_sigma_u2_2);
g+=eps*square(log_sigma_u2_3);

g+=eps*norm2(log_sigma_u3);

// Random effects contribution from u1
g += 0.5*norm2(u1_1);
g += 0.5*norm2(u1_2);
g += 0.5*norm2(u1_3);
```

```
  g += 0.5*norm2(u2_1);
  g += 0.5*norm2(u2_2);
  g += 0.5*norm2(u2_3);

  g += 0.5*norm2(u3);

  g+=3*(M1+M2+M3)*lstp;

  double wght=0.0;
  switch(current_phase())
  {
  case 1:
    wght=10.0;
    break;
  case 2:
    wght=1.0;
    break;
  case 3:
    wght=1.0;
    break;
  default:
    wght=0.0;
    break;
  }
  g+=wght*norm2(beta);

REPORT_SECTION

  //report << beta0+beta << endl;
  report << sigma << endl;
  report << exp(log_sigma_u1_1) << endl;
  report << exp(log_sigma_u1_2) << endl;
  report << exp(log_sigma_u1_3) << endl;
  report << exp(log_sigma_u2_1) << endl;
  report << exp(log_sigma_u2_2) << endl;
  report << exp(log_sigma_u2_3) << endl;
  report << exp(log_sigma_u3) << endl;


TOP_OF_MAIN_SECTION
  arrmblsize = 40000000L;
  gradient_structure::set_GRADSTACK_BUFFER_SIZE(3000000);
  gradient_structure::set_CMPDIF_BUFFER_SIZE(200000);
  gradient_structure::set_MAX_NVAR_OFFSET(10000);
```