Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *http://www.jstatsoft.org/*

Inverse Estimation with Linear Mixed-Effects Models in R

Brandon M. Greenwell
Aptima, Inc

Christine M. Schubert Kabban
Air Force Institute of Technology

Abstract

Inverse estimation, more commonly known as calibration, is a classical and well-known problem in regression. In simple terms, it involves the use of an observed value of the response (or specified value of the mean response) to make inference on the corresponding unknown value of the explanatory variable. In this paper, we describe how to calculate approximate calibration confidence intervals for the unknown value of the explanatory variable in random coefficient models using a well-known data set.

Keywords: calibration, random coefficients, **investr**, **lme4**, **nlme**, R.

1. Introduction

Consider an ordinary regression model $\mathcal{Y}_i = f(x_i; \boldsymbol{\beta}) + \epsilon_i$ $(i = 1, \ldots, n)$, where f is a known expectation function (called a *calibration curve*) that is monotonic over the range of interest and $\epsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$. A common problem in regression is to predict a future response $\mathcal{Y}_0$ (or estimate the mean response f_0) for a known value of the explanatory variable x_0. Often, however, there is a need to do the reverse; that is, given an observed value of the response $\mathcal{Y} = y_0$ (or a specified value of the mean response), estimate the unknown value of the explanatory variable x_0. This is known as the *calibration problem*, though it is often refered to more generally as inverse estimation. A thorough overview of the calibration problem is given in Osborne (1991). Oman (1998) considers the case of a random intercept and slope model — the goal of this paper is to show how easily this can be accomplished using a few packages from the R programming language.

2. Linear Mixed-Effects Models

Linear regression models with random coefficients can be represented in many different (but equivalent) forms. One of the most common forms, attributed to Laird and Ware (1982), is

$$\boldsymbol{\mathcal{Y}}_i = \boldsymbol{X}_i\boldsymbol{\beta} + \boldsymbol{Z}_i\boldsymbol{b}_i + \boldsymbol{\epsilon}_i, \quad i = 1, \ldots, m, \tag{1}$$

where

- $\boldsymbol{\mathcal{Y}}_i$ is an $n_i \times 1$ response vector for the i-th subject/cluster/group;
- $\boldsymbol{X}_i$ is an $n_i \times p$ design matrix for the fixed effects;
- $\boldsymbol{Z}_i$ is an $n_i \times q$ design matrix for the random effects;
- $\boldsymbol{\beta}$ is a $p \times 1$ vector of fixed effects coefficients;
- $\boldsymbol{b}_i$ is a $q \times 1$ vector of random effects coefficients with mean zero and variance-covariance matrix $\boldsymbol{D}$;
- $\boldsymbol{D}$ is a $q \times q$ variance-covariance matrix for the random effects;
- $\boldsymbol{\epsilon}_i$ is an $n_i \times 1$ vector of random errors with mean zero and variance-covariance matrix $\sigma^2\boldsymbol{I}$.

Equation (1) is known as a linear mixed-effects model (LMM). The random effects and errors are often assumed to follow a normal distribution. By stacking the data, the (normal) LMM can be written concisely as

$$\boldsymbol{\mathcal{Y}} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{b} + \boldsymbol{\epsilon}, \quad \begin{bmatrix}\boldsymbol{b}\\ \boldsymbol{\epsilon}\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix}, \begin{bmatrix}\boldsymbol{D} & \boldsymbol{0}\\ \boldsymbol{0} & \sigma^2\boldsymbol{I}\end{bmatrix}\right), \tag{2}$$

where $\boldsymbol{\mathcal{Y}} = \operatorname{col}\{\boldsymbol{\mathcal{Y}}_i\}$, $\boldsymbol{X} = \operatorname{col}\{\boldsymbol{X}_i\}$, $\boldsymbol{Z} = \operatorname{diag}\{\boldsymbol{Z}_i\}$, $\boldsymbol{b} = \operatorname{col}\{\boldsymbol{b}_i\}$, and $\boldsymbol{\epsilon} = \operatorname{col}\{\boldsymbol{\epsilon}_i\}$ for $i = 1, \ldots, m$. Since $\mathsf{COV}[\boldsymbol{b}, \boldsymbol{\epsilon}] = \boldsymbol{0}$, it is assumed that the random vectors $\{\boldsymbol{b}_i, \boldsymbol{\epsilon}_i\}_{i=1}^m$ are mutually independent.

The additional term $\boldsymbol{Z}\boldsymbol{b}$ in the model imposes a specific variance-covariance structure on the response vector $\boldsymbol{\mathcal{Y}}$:

$$\boldsymbol{\mathcal{Y}} \sim \mathcal{N}(\boldsymbol{X}\boldsymbol{\beta}, \boldsymbol{V}), \quad \boldsymbol{V} = \boldsymbol{Z}\boldsymbol{D}\boldsymbol{Z}^\top + \sigma^2\boldsymbol{I}.$$

Thus, the fixed effects determine the mean of $\boldsymbol{\mathcal{Y}}$, while the random effects govern the variance-covariance structure of $\boldsymbol{\mathcal{Y}}$. Different random effects structures impose different variance-covariance structures on the response resulting in a highly flexible framework for modelling *grouped data*.

The random effects variance-covariance matrix $\boldsymbol{D}$ has at most $q(q+1)/2$ unique elements which we represent by the vector $\boldsymbol{\theta}$. There are a number of methods available for estimating $(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta})$; see, for example, McCulloch, Searle, and Neuhaus (2008, chap 6.) and Demidenko (2013, chap. 2). Most commonly, the fixed-effects $\boldsymbol{\beta}$ are estimated via the method of maximum likelihood (ML), while the variance components $(\sigma^2, \boldsymbol{\theta})$ are estimated via restricted maximum likelihood (REML). The ML estimator of $\boldsymbol{\beta}$, given by

$$\widehat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^\top\widehat{\boldsymbol{V}}^{-1}\boldsymbol{X}\right)\boldsymbol{X}^\top\boldsymbol{\mathcal{Y}},$$

depends on the estimated variance components through $\widehat{\boldsymbol{V}}$ which makes it difficult to capture the variability of $\widehat{\boldsymbol{\beta}}$ in small sample sizes (see McCulloch *et al.* (2008, pp. 165-167)). The usual practice is to ignore the variability of the estimated variance components when making inference about the fixed effects; that is, treat $\widehat{\boldsymbol{V}}$ as the true (fixed) value of $\boldsymbol{V}$. Modern computational procedures such as the parametric bootstrap and Markov chain Monte Carlo (MCMC) methods are two ways account for the variability of the estimated variance components.

2.1. Bladder Volume Data

For illustration, let us consider the bladder volume data which can be found in Brown (1993, pg. 7) and Oman (1998). In Brown's words:

> "A series of 23 women patients attending a urodynamic clinic were recruited for the study. After successful voiding of the bladder, sterile water was introduced in additions of 1, 1.5, and then 2.5 cl increments up to a final cumulative total of 17.5 cl. At each volume a measure of height (`H`) in mm and depth (`D`) in mm of largest ultrasound bladder images were taken. The product `H` $\times$ `D` was taken as a measure of liquid volume."

We took Brown's suggestion and transformed the data so that the relationship is approximately linear. Spaghettiplots of both the original and transformed data are displayed in Figure 1.

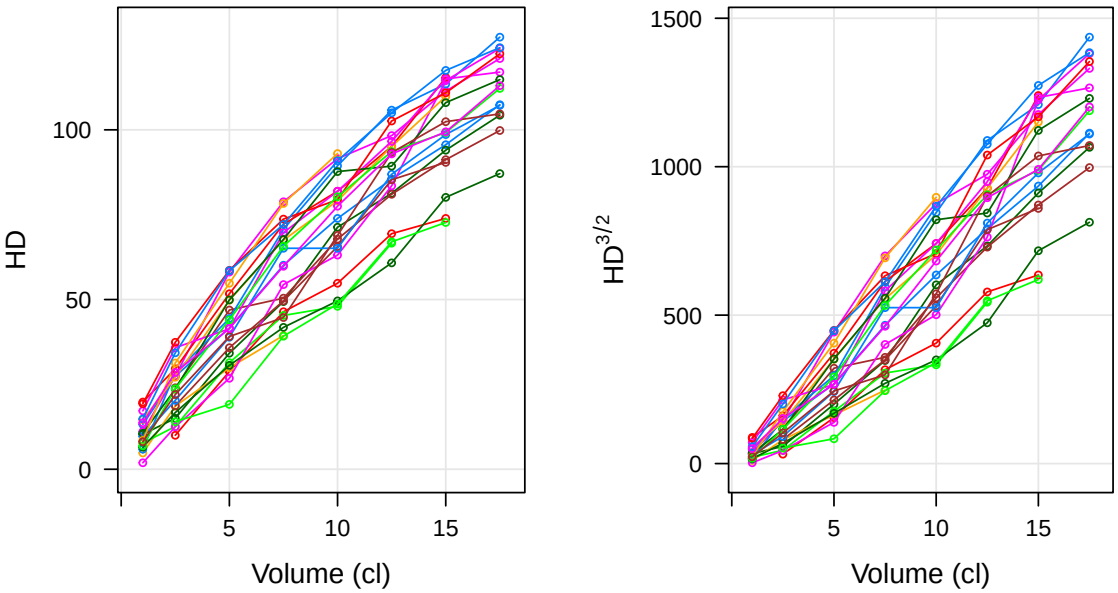

Figure 1: Spaghettiplot of bladder volume data (lines connect measurements belonging to the same subject). *Left*: Original data. *Right*: Transformed data.

A random intercept and slope model with uncorrelated random effects fits the transformed data well:

$$\begin{aligned} \texttt{HD}_{ij}^{3/2} &= (\beta_0 + b_{0i}) + (\beta_1 + b_{1i})\,\texttt{volume}_{ij} + \epsilon_{ij} \\ b_{ki} &\sim \mathcal{N}\left(0, \theta_k^2\right), \quad k = 0, 1, \\ \epsilon_{ij} &\sim \mathcal{N}\left(0, \sigma^2\right), \end{aligned}$$

where $\mathsf{COV}\left[b_{0i}, b_{1i}\right] = 0$. To fit such a model in R, we can use the recommended **nlme** package (Pinheiro, Bates, DebRoy, Sarkar, and R Core Team 2013) as follows:

```
library(nlme)  # nlme should already be installed
(fit.nlme <- lme(HD^(3/2) ~ volume, random = list(subject = pdDiag(~volume)),
                 data = bladder))

## Linear mixed-effects model fit by REML
##   Data: bladder
##   Log-restricted-likelihood: -943
##   Fixed: HD^(3/2) ~ volume
## (Intercept)      volume
##       -53.8        69.1
##
## Random effects:
##  Formula: ~volume | subject
##  Structure: Diagonal
##         (Intercept) volume Residual
## StdDev:        39.6   14.3     53.7
##
## Number of Observations: 166
## Number of Groups: 23
```

The `pdDiag` function forces a diagonal variance-covariance structure on the random effects; hence, a covariance of zero. The same model, but with an additional quadratic term, provides a useful fit to the original (untransformed) data:

$$\begin{aligned} \texttt{HD}_{ij} &= (\beta_0 + b_{0i}) + (\beta_1 + b_{1i})\,\texttt{volume}_{ij} + \beta_2 \texttt{volume}_{ij}^2 + \epsilon_{ij} \\ b_{ki} &\sim \mathcal{N}\left(0, \theta_k^2\right), \quad k = 0, 1, \\ \epsilon_{ij} &\sim \mathcal{N}\left(0, \sigma^2\right). \end{aligned}$$

For an in-depth treatment on fitting LMMs using the **nlme** software, see Pinheiro and Bates (2000).

3. Extending the Classical Methods of Calibration

The standard methods of calibration, (i.e., the Wald-based and inversion confidence intervals) are easily extended to the case of random coefficients. For convenience, let us write the linear random coefficient model as

$$\mathcal{Y}_{ij} = f\left(x_{ij}; \boldsymbol{\beta}\right) + R\left(x_{ij}; \boldsymbol{b}_i\right) + \epsilon_{ij},$$

where $f(\cdot)$ and $R(\cdot)$ are linear in $\boldsymbol{\beta}$ and $\boldsymbol{b}_i$, respectively. For instance, the model for the transformed bladder data has $f\left(\texttt{volume}_{ij}; \boldsymbol{\beta}\right) = \beta_0 + \beta_1 \texttt{volume}_{ij}$ and $R\left(\texttt{volume}_{ij}; \boldsymbol{b}_i\right) = b_{0i} + b_{1i}\texttt{volume}_{ij}$ with $\mathsf{VAR}\left[R\left(\texttt{volume}_{ij}; \boldsymbol{b}_i\right)\right] = \theta_0^2 + \texttt{volume}_{ij}^2\theta_1^2$.

3.1. Point Estimation

Assume that, after the data are collected and a model is fitted, we obtain a new observation, denoted $\mathcal{Y}_0$, from the same population under study for which the value of the explanatory variable x_0 is unknown. We assume that the new observation belongs to a group not included in our analysis. Estimating x_0 is rather straightforward. By assumption, the new observation $\mathcal{Y}_0$ is distributed as a $\mathcal{N}\left\{f\left(x_0; \boldsymbol{\beta}\right), \sigma_0^2\right\}$ random variable with $\sigma_0^2 = \mathsf{VAR}\left[R\left(x_0; \boldsymbol{b}_0\right)\right] + \sigma^2$. A natural estimator for x_0 is then

$$\widehat{x}_0 = f^{-1}\left(\mathcal{Y}_0; \widehat{\boldsymbol{\beta}}\right), \tag{3}$$

where $\widehat{\boldsymbol{\beta}}$ is the ML estimator of $\boldsymbol{\beta}$. We shall refer to Equation (3) as the classical estimator. Note that the point estimate $\widehat{x}_0$ does not involve any of the random effects; the random effects only contribute to the variance-covariance structure of the response.

3.2. An Asymptotic Wald Interval

An approximate $100(1-\alpha)\%$ Wald-type confidence interval for x_0 has the simple form

$$CI_{wald} = \left(\widehat{x}_0 - \mathrm{SE}\left[\widehat{x}_0\right]\Phi\left(\alpha/2\right), \widehat{x}_0 - \mathrm{SE}\left[\widehat{x}_0\right]\Phi\left(1-\alpha/2\right)\right). \tag{4}$$

There is no "textbook" formula for the standard error of $\widehat{x}_0$, instead, an estimate of this standard error is obtained using a first-order Taylor series approximation, or better yet, a bootstrap approximation. For the Taylor series approximation, we need the variance-covariance matrix of $\left(\mathcal{Y}_0, \widehat{\boldsymbol{\beta}}\right)$,

$$\Sigma = \begin{bmatrix} \mathsf{VAR}\left[\mathcal{Y}_0\right] & \mathbf{0} \\ \mathbf{0} & \mathsf{VAR}\left[\widehat{\boldsymbol{\beta}}\right] \end{bmatrix} = \begin{bmatrix} \sigma_0^2 & \mathbf{0} \\ \mathbf{0} & \left(\boldsymbol{X}^\top \boldsymbol{V}^{-1} \boldsymbol{X}\right)^{-1} \end{bmatrix}.$$

Since $\mathcal{Y}_0$ is independent of $\boldsymbol{\mathcal{Y}}$, it is also independent of $\widehat{\boldsymbol{\beta}}$, hence the diagonal structure of Σ. Recall that our point estimate has the form $x = f^{-1}\left(y; \boldsymbol{\beta}\right)$. Let $f_1^{-1}\left(y; \boldsymbol{\beta}\right)$ and $f_2^{-1}\left(y; \boldsymbol{\beta}\right)$ denote the partial derivatives of f^{-1} with respect to the parameters y and $\boldsymbol{\beta}$, respectively. Our point estimator is given by $f^{-1}\left(\mathcal{Y}_0; \widehat{\boldsymbol{\beta}}\right)$, where $\mathcal{Y}_0$ is a new observation and $\widehat{\boldsymbol{\beta}}$ is the ML estimator of $\boldsymbol{\beta}$. A first-order Taylor-series approximation for the variance of $\widehat{x}_0$ is given by

$$\begin{aligned} \mathsf{VAR}\left[\widehat{x}_0\right] = &\left[f_1^{-1}\left(\mathcal{Y}_0; \widehat{\boldsymbol{\beta}}\right)\right]^2 \sigma_0^2 \\ &+ \left[f_2^{-1}\left(\mathcal{Y}_0; \widehat{\boldsymbol{\beta}}\right)\right]^\top \left(\boldsymbol{X}^\top \boldsymbol{V}^{-1} \boldsymbol{X}\right)^{-1} \left[f_2^{-1}\left(\mathcal{Y}_0; \widehat{\boldsymbol{\beta}}\right)\right]. \end{aligned} \tag{5}$$

To obtain $\mathrm{SE}\left[\widehat{x}_0\right] = \left\{\widehat{\mathsf{VAR}}\left[\widehat{x}_0\right]\right\}^{1/2}$, we simply replace σ_0^2 and $\boldsymbol{V}$ with their respective estimates $\widehat{\sigma}_0^2$ and $\widehat{\boldsymbol{V}}$.

The Wald-based interval is simple to compute as long as we have an estimate for the standard error. As we will discuss in Section 5, the R package **investr** (Greenwell 2013) can be used

to obtain the Wald-based interval (4) using a Taylor series approximation of the standard error. If a closed-form formula is available for $\widehat{x}_0$, then the `deltaMethod` function from the **car** package (Fox and Weisberg 2011) can also be used to obtain the Taylor series approximation of the standard error. Alternatively, one can use a parametric bootstrap estimate of the standard error instead of relying on a Taylor series approximation — see Section 4.1. The bootstrap estimate may be more accurate in smaller sample sizes because, unlike the Taylor approximation estimate, it takes into account the variability of the estimated variance components. The new `bootMer` function in the **lme4** package (Bates, Maechler, Bolker, and Walker 2014) can be used for model-based parametric bootstrapping in mixed models.

3.3. An Asymptotic Inversion Interval

In the case of the simple linear regression model with constant variance, an exact $100(1-\alpha)\%$ confidence interval for x_0 can be derived (Graybill 1976). This can be generalized to an approximate method in the case of polynomial or nonlinear regression models with independent observations and constant variance (see Seber and Wild (2003) and Huet (2004)). In a similar fashion, we can generalize the same results to an approximate method for random coefficient models. Let $\widehat{f}_0 = f\left(x_0; \widehat{\boldsymbol{\beta}}\right)$ be the predicted mean at $x = x_0$. A prediction interval for $\mathcal{Y}_0$ at x_0 with asymptotic coverage probability $100(1-\alpha)\%$ is

$$\mathcal{I}_{\infty}\left(x_0\right) = \widehat{f}_0 \pm z_{1-\alpha/2}\left\{\widehat{\mathsf{VAR}}\left[\mathcal{Y}_0 - \widehat{f}_0\right]\right\}^{1/2}. \tag{6}$$

If instead, $\mathcal{Y}_0$ is observed to be y_0 and x_0 is unknown, then an asymptotic $100(1-\alpha)\%$ confidence interval for the unknown x_0 can be obtained by inverting (6):

$$CI_{inv} = \left\{ x : z_{\alpha/2} \leq \frac{\mathcal{Y}_0 - f\left(x; \widehat{\boldsymbol{\beta}}\right)}{\left\{\widehat{\mathsf{VAR}}\left[\mathcal{Y}_0 - f\left(x; \widehat{\boldsymbol{\beta}}\right)\right]\right\}^{1/2}} \leq z_{1-\alpha/2} \right\}. \tag{7}$$

This is known as the *inversion interval* and typically cannot be written in closed-form; therefore, numerical techniques are required to find the lower and upper bounds. Further, note that CI_{inv} is not symmetric about $\widehat{x}_0$ and will not necessarily result in a single finite interval. Fortunately, the inversion interval (7) can be computed automatically using the **investr** package. However, like the Wald-based interval (4), the inversion interval ignores the variability of the estimated variance components and will likely perform poorly in small sample sizes. An alternative approach involving the parametric bootstrap will be discussed in Section 4.3. Finally, the inversion interval uses a normal approximation. While it is likely that a t distribution with some finite degrees of freedom may be more accurate, it is difficult to find the appropriate degrees of freedom. Oman (1998), suggests a t distribution with $N-1$ degrees of freedom (N being the total sample size).

4. A Parametric Bootstrap Algorithm

The bootstrap (Efron 1979) is a general-purpose computer-based method for assessing accuracy of estimators and forming confidence intervals for parameters. Jones and Rocke (1999) proposed a nonparametric bootstrap algorithm for controlled calibration with independent

observations. However, since our application involves random coefficients (i.e., dependent observations), the nonparametric bootstrap does not easily apply, and instead, we adopt a "fully parametric" approach. In a parametric bootstrap, bootstrap samples are generated from a fitted parametric model rather than sampling with replacement directly from the data. Fortunately, parametric bootstrap confidence intervals are usually more accurate than nonparametric ones, however, by sampling from a fitted parametric family, we are implicitly assuming that we have the "correct model".

Let $\widehat{\sigma}_0^2$ be an estimate of the variance of the new observation $\mathcal{Y}_0$. An algorithm for bootstrapping $\widehat{x}_0$ in an LMM is given in Figure 2. Note that step 5. is crucial for calibration problems because we need to treat y_0 as a random quantity in the bootstrap simulation, otherwise the variability of $\widehat{x}_0$ will be underestimated; see, for example, Jones and Rocke (1999) and Greenwell and Kabban (2014).

1. Fit a mixed model (2) to the data and obtain estimates $\widehat{\boldsymbol{\beta}}$, $\widehat{\boldsymbol{D}}$, and $\widehat{\sigma}^2$.
2. Define $\boldsymbol{y}^\star = \boldsymbol{X}\widehat{\boldsymbol{\beta}} + \boldsymbol{Z}\boldsymbol{b}^\star + \boldsymbol{\epsilon}^\star$, where $\boldsymbol{b}^\star \sim \mathcal{N}_q\left(\boldsymbol{0}, \widehat{\boldsymbol{D}}\right)$ and $\boldsymbol{\epsilon}^\star \sim \mathcal{N}_N\left(\boldsymbol{0}, \widehat{\sigma}_\epsilon^2 \boldsymbol{I}\right)$;
3. Update the original model using $\boldsymbol{y}^\star$ as the response vector to obtain $\widehat{\boldsymbol{\beta}}^\star$ and $\widehat{\sigma}_0^{2\star}$;
4. Generate $y_0^\star \sim \mathcal{N}\left(y_0, \widehat{\sigma}_0^{2\star}\right)$;
5. Define $\widehat{x}_0^\star = f^{-1}\left(y_0^\star; \widehat{\boldsymbol{\beta}}^\star\right)$;
6. Repeat steps (2)-(5) R times.

Figure 2: Parametric bootstrap algorithm for linear calibration with random coefficients.

There are three main bootstrap confidence interval procedures: The percentile methods introduced in Efron (1979), the studentized bootstrap t method introduced in Efron (1982), and the double bootstrap method (Hall 1986). In this paper, we focus on the simple percentile interval and the studentized method. For a good overview of all these confidence interval procedures, see Davison and Hinkley (1997, chap. 5) and Boos and Stefanski (2013, chap. 11).

4.1. The percentile interval

The percentile method is the simplest and is given by the sample $\alpha/2$ and $1-\alpha/2$ quantiles of the bootstrap sample. Let $\widehat{x}_{1*}, \ldots, \widehat{x}_{R*}$ be a bootstrap sample obtained from the algorithm in Figure 2. If $\widehat{F}_R$ is the empirical distribution function of the bootstrap sample, then an approximate $100(1-\alpha)\%$ confidence interval for x_0 is given by

$$\left(\widehat{F}_R^{-1}(\alpha/2), \widehat{F}_R^{-1}(1-\alpha/2)\right).$$

The percentile interval is transformation respecting; thus, if we want a confidence interval for any one-to-one transformation $g(\widehat{x}_0)$, then we can just apply the transformation to the

endpoints of the percentile interval for $\widehat{x}_0$. The drawback is that the percentile interval is only *first-order accurate* (see Boos and Stefanski (2013, pp. 429-430)). Efron (1987) offered an improvement over the percentile interval called the *bias-corrected and accelerated* (BC_a) interval that often obtains second-order accuracy and is transformation respecting. However, the recommended R package **boot** (Canty and Ripley 2013) we will be relying on currently does not allow for BC_a confidence intervals to be constructed from a parametric bootstrap.

4.2. The bootstrap t interval

A bootstrap t interval for x_0 is essentially a bootstrap adjusted Wald-type interval. The Wald interval for x_0 (4) assumes that

$$Q_W = \frac{\widehat{x}_0 - x_0}{\text{SE}\left[\widehat{x}_0\right]} \sim \mathcal{N}(0,1).$$

Rather than assuming that Q_W is normal, the bootstrap t method uses the bootstrap distribution of $Q_W^\star = \left(\widehat{x}_0^\star - \widehat{x}_0\right) / \text{SE}\left[\widehat{x}_0^\star\right]$ to estimate the true distribution of Q_W. If $\widehat{F}_{Q_W}$ is the empirical distribution function for a sample of R bootstrap replicates of Q_W, then the bootstrap t interval for x_0 is given by

$$CI_{wald}^\star = \left(\widehat{x}_0 - \text{SE}\left[\widehat{x}_0\right] \widehat{F}_{Q_W}\left(\alpha/2\right), \widehat{x}_0 - \text{SE}\left[\widehat{x}_0\right] \widehat{F}_{Q_W}\left(1-\alpha/2\right)\right).$$

In order to implement this method, we need to calculate the standard error of each bootstrap replicate. To do this, we can either use an additional (nested) bootstrap to estimate the standard error, or use a Taylor series approximation as discussed in Section 3.2. If time is not a factor, then the nested bootstrap is preferred since bootstrap standard errors are usually more accurate than those based on a first-order Taylor series approximation (Casella and Berger 2002, pp. 478-480). The benefits of using this interval over (4) are that (a) it does not assume normality (hence, likely to be more accurate in smaller sample sizes) and (b) it is not symmetric about $\widehat{x}_0$; thus, more realistic when the response is nonlinear in x (e.g., polynomials, etc.).

4.3. Bootstrap adjusted inversion interval

Huet (2004) suggests a bootstrap modification of the usual inversion interval in nonlinear regression models with dependent data. In a similar fashion, we could use the parametric bootstrap to adjust the approximate inversion interval given in Equation (7). The inversion interval assumes that the *predictive pivot*

$$Q_I = \frac{\mathcal{Y}_0 - f\left(x;\widehat{\boldsymbol{\beta}}\right)}{\left\{\widehat{\text{VAR}}\left[\mathcal{Y}_0 - f\left(x;\widehat{\boldsymbol{\beta}}\right)\right]\right\}^{1/2}} \sim \mathcal{N}(0,1).$$

A bootstrap modified inversion interval would then use the bootstrap distribution of

$$Q_I^\star = \frac{\mathcal{Y}_0^\star - f\left(\widehat{x}_0;\widehat{\boldsymbol{\beta}}^\star\right)}{\left\{\widehat{\text{VAR}}\left[\mathcal{Y}_0 - f\left(\widehat{x}_0;\widehat{\boldsymbol{\beta}}^\star\right)\right]\right\}^{1/2}},$$

to estimate the true distribution of Q_I. If $\widehat{F}_{Q_I}$ is the empirical distribution function for a sample of R bootstrap replicates of Q_I, then the modified inversion interval for x_0 is given by

$$CI^{\star}_{inv} = \left\{ x : \widehat{F}_{Q_I}(\alpha/2) \leq Q_I \leq \widehat{F}_{Q_I}(1-\alpha/2) \right\}.$$

5. Implementation in R

Here, we discuss how to implement the previous procedures in the R programming languages. We discuss two main packages: **investr** and **lme4**. The **investr** package can be used for obtaining the Wald-based and inversion intervals. This functionality is demonstrated over the next two sections. The **lme4** package is a popular package for fitting linear, generalized linear, and nonlinear mixed models. Recently, however, the **lme4** package creators have added functionality for model-based parametric bootstrapping. In Section 5.3, we demonstrate the potential of this new functionality by applying our parametric bootstrap algorithm to the bladder volume data discussed earlier.

5.1. The investr package

The R package **investr** facilitates calibration/inverse estimation with linear and nonlinear regression models. The main function, `invest`, can be used for inverse estimation of x_0 given an observed response y_0. More recently, the package has been updated to also handle objects of class `lme` from the **nlme** package. Current functionality includes both the Wald-based and inversion methods outlined in Sections 3.2-3.3. The main arguments for this function (as it applies to `lme` objects) are noted in Table 1 below. The code for the package is hosted on GitHub at https://github.com/w108bmg/investr, but the latest stable release can be found on CRAN at http://CRAN.R-project.org/package=investr.

Argument	Description
`object`	An R object that inherits from class `lm`, `nls`, or `lme`.
`y0`	The value of the observed response.
`interval`	The type of interval required. Currently, only `"none"`, `"Wald"`, and `"inversion"` are supported.
`level`	A numeric scalar between 0 and 1 giving the confidence level for the interval to be calculated. The default is `0.95`.
`lower`	The lower endpoint of the interval to be searched.
`upper`	The upper endpoint of the interval to be searched.
`q1`, `q2`	Optional quantiles to be used for constructing confidence intervals.
`tol`	The desired accuracy to be passed on to `uniroot`.
`maxiter`	The maximum number of iterations to be passed on to `uniroot`.

Table 1: Main arguments for the `invest` function.

Returning to the bladder volume example, suppose we obtained an ultrasound measurement from a new patient for which $\texttt{HD}^{3/2} = 500$ (that's roughly 63 on the original scale). What is

the true volume of fluid (x_0) in the patients bladder? We can estimate the true volume and form an approximate 95% confidence interval using the methods discussed previously. The point estimate is simply given by

$$\widehat{x}_0 = \frac{500 + 53.83164}{69.09491} = 8.0155 \text{ (cl)}.$$

This estimate can be obtained in R as follows:

```
library(investr)
(x0.est <- invest(fit.nlme, y0 = 500, interval = "none"))

## [1] 8.02
```

The code used by `invest` to obtain this point estimate is basically

```
fun <- function(x) {
  predict(fit.nlme, newdata = list("volume" = x), level = 0) - 500
}
uniroot(fun, lower = 1, upper = 17.5, tol = 1e-10, maxiter = 1000)$root

## [1] 8.02
```

In other words, `invest` relies on the function `uniroot` from the **stats** package to solve the equation $f\left(x; \widehat{\boldsymbol{\beta}}\right) - y_0 = 0$ numerically for x. If the solution does not lie in the range of predictor values, then an error message will be displayed, as in

```
invest(fit.nlme, y0 = 1500)

## Error:  Point estimate not found in the search interval (1, 17.5).  Try tweaking
the values of lower and upper.
```

The values for `lower`, `upper`, `tol`, and `maxiter` are controlled via the arguments of the same name listed in Table 1.

When `interval = "Wald"`, an asymptotic $100(1-\alpha)\%$ confidence interval (where α is equal to `1 - level`) for x_0 is calculated according to Equation (4):

```
invest(fit.nlme, y0 = 500, interval = "Wald")

## estimate    lower    upper       se
##     8.02     4.18    11.85     1.95
```

The standard error is computed using a First-order Taylor series approximation. Similar to the code snippet shown below, `invest` calls the **stats** function `numericDeriv` to numerically evaluate the gradient of $\widehat{x}_0$ as a function of y_0 and $\widehat{\boldsymbol{\beta}}$.

```
dmFun <- function(params) { # function of parameters whose gradient is required
  fun <- function(x) {
    X <- model.matrix(eval(fit.nlme$call$fixed)[-2],
                      data = data.frame("volume" = x))
    X %*% params[-length(params)] - params[length(params)]
  }
  uniroot(fun, lower = 1, upper = 17.5, tol = 1e-10,
          maxiter = 1000)$root
}
params <- c(fixef(fit.nlme), 500)
covmat <- diag(3)  # set up variance-covariance matrix
covmat[1:2, 1:2] <- vcov(fit.nlme)  # fixed effects var/cov matrix
covmat[3, 3] <- 17572.35  # VAR[Y_0]
gv <- attr(numericDeriv(quote(dmFun(params)), "params"), "gradient")
(se <- as.numeric(sqrt(gv %*% covmat %*% t(gv))))

## [1] 1.95
```

Alternatively, one can use the very useful `deltaMethod` function from the **car** package to obtain `se`:

```
library(car)  # assuming package car is already installed
params <- c(fixef(fit.nlme), 500)
covmat <- diag(3)  # set up var/cov matrix
covmat[1:2, 1:2] <- vcov(fit.nlme)  # fixed effects var/cov matrix
covmat[3, 3] <-  17572.35  # VAR[Y_0]
names(params) <- c("b0", "b1", "y0")
(se <- deltaMethod(params, g = "(y0 - b0)/b1",  vcov. = covmat)$SE)

## [1] 1.95
```

The only drawback here is that `deltaMethod` relies on the **stats** package symbolic differentiation function `D`; hence, $\widehat{x}_0 = f^{-1}\left(y_0; \widehat{\boldsymbol{\beta}}\right)$ has to be obtainable in closed-form.

To obtain the approximate inversion interval (7), we specify `interval = "inversion"` (the default) as in the following:

```
invest(fit.nlme, y0 = 500, interval = "inversion")

## estimate    lower    upper
##     8.02     4.23    11.92
```

Essentially, `invest` finds the lower and upper inversion confidence limits (7) by solving the equations

$$Q_I - z_{\alpha/2} = 0 \quad \text{and} \quad Q_I - z_{1-\alpha/2} = 0$$

numerically for x using the R function `uniroot`. To use the quantiles from a t distribution instead (see Section 3.3), we can supply them via the arguments `q1` and `q2`:

```
N <- nrow(bladder)  # total sample size
tvals <- qt(c(0.025, 0.975), df = N-1)  # quantiles from t distribution
invest(fit.nlme, y0 = 500, q1 = tvals[1], q2 = tvals[2])

## estimate    lower    upper
##     8.02     4.20    11.95
```

Being able to specify the arguments `q1` and `q2` will also be useful when implementing the bootstrap adjusted inversion interval described in Section 4

5.2. Monte Carlo Study

To assess the empirical performance of these confidence intervals, we carried out a small Monte Carlo study. The simulations described below were conducted in R using packages **plyr** (Wickham 2011), **nlme**, and **lme4** (Bates *et al.* 2014). The results are reported in Table 2 and indicate that the Wald-based confidence interval (4) and inversion confidence interval (7) have asymptotic coverage probability close to $100(1-\alpha)\%$. This experiment also highlighted the fact that it is the number of subject m, not the sample size per subject n, that is more important for good asymptotic coverage. The code used for the simulation is available upon request.

We consider the values 5, 10, 30, 50, and 100 for both the number of subjects m and and the number of observations per subject n. For each combination of sample sizes, we generated 1,000 data sets from a random intercept and slope model with parameters given by those listed in `summary(fit.nlme)`. In other words, the fixed effects were $\boldsymbol{\beta} = (-53.83164, 69.09491)^\top$, the standard deviations for the (uncorrelated) random intercept and slope were 39.62499, and 14.28841, respectively. The residual standard deviation was $\sigma = 53.71511$. We chose $f(x_0; \boldsymbol{\beta}) = 500$ so that the true unknown is $x_0 = 8.0155$. The standard deviation of the coverage estimates is approximately $\sqrt{0.95(1-0.95)/1000} = 0.001$. A trellis plot of the results is given in Figure 3. The coverage estimates are plotted against the number of subjects m and paneled by number of observations per subject n.

m	Method	$n=5$	$n=10$	$n=30$	$n=50$	$n=100$
5	Wald	0.89	0.89	0.89	0.87	0.89
	Inversion	0.89	0.90	0.89	0.88	0.90
10	Wald	0.92	0.92	0.94	0.93	0.93
	Inversion	0.92	0.92	0.94	0.94	0.93
30	Wald	0.95	0.95	0.95	0.94	0.95
	Inversion	0.94	0.94	0.94	0.94	0.95
50	Wald	0.95	0.96	0.95	0.94	0.94
	Inversion	0.94	0.95	0.95	0.94	0.94
100	Wald	0.95	0.95	0.95	0.94	0.94
	Inversion	0.95	0.95	0.95	0.94	0.95

Table 2: Coverage probability of 95% confidence intervals for simulated bladder data.

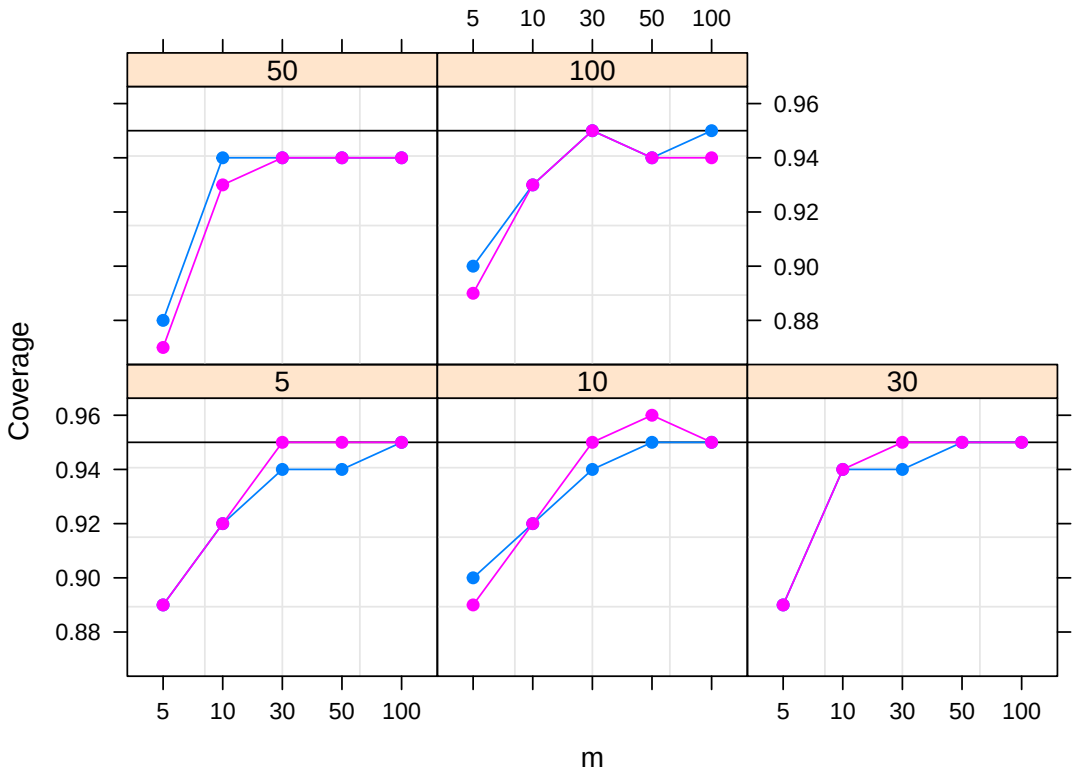

Figure 3: Coverage probability of 95% confidence intervals for simulated bladder data. The coverage estimates based on the inversion method are colored blue.

5.3. Using the lme4 package

Implementation of the parametric bootstrap algorithm in Figure 2 is relatively straight forward using the new `bootMer` function from the well-known R package **lme4** (Bates *et al.* 2014) in conjunction with the **boot** package.

Since we will be using the **lme4** package, we need to refit the model using the `lmer` function:

```
library(lme4)  # assuming lme4 is already installed
fit.lme4 <- lmer(HD2 ~ volume + (0+1|subject) + (0+volume|subject),
                 data = bladder)
```

Theoretically, the parameter estimates from this model should be the same as those from `fit.nlme`; however, there are likely to be small numerical differences between the two. For this reason, let us re-estimate $\widehat{x}_0$ using `fit.lme4`. Since `invest` does not work on objects of class `lmer`, we have to do things manually:

```
fe <- unname(fixef(fit.lme4))  # fixed effects without dimnames attribute
(x0.est <- (500 - fe[1]) / fe[2])
```

```
## [1] 8.02
```

Also, for convenience, we define the following function which estimates $\mathsf{VAR}[\mathcal{Y}|x] = \sigma_0^2 + x^2\sigma_1^2 + \sigma^2$ for a given value of x:

```
var.y <- function(object, x) {
  vc <- as.data.frame(lme4::VarCorr(object))$vcov
  vc[1] + vc[2]*x^2 + vc[3]
}
```

For example, to estimate $\sigma_0^2 = \widehat{\mathsf{VAR}}[\mathcal{Y}_0]$, we have `var.y(fit.lme4, x = x0.est)`, which gives 1.757×10^4, the same value used in the previous section.

Although we could easily compute all the bootstrap intervals previously discussed in one call to `bootMer` and `boot.ci`, we will discuss and compute each interval separately.

The following snippet of code generates $R = 9999$ bootstrap replicates of $\widehat{x}_0$, Q_W, and Q_I according to the algorithm in Figure 2:

```
boot.fun <- function(.) {  # bootstrap function

  ## Point estimate
  var.y0.boot <- var.y(., x = x0.est)  # VAR[Y0]
  fe.boot <- unname(fixef(.))  # fixed effects
  if (all(getME(., "y") == bladder$HD2)) {
    y0.boot <- 500
  } else {
    y0.boot <- rnorm(1, 500, sqrt(var.y0.boot))
  }
  x0.boot <- (y0.boot - fe.boot[1])/fe.boot[2]

  ## Approximate variance
  covmat <- diag(3)
  covmat[1:2, 1:2] <- as.matrix(vcov(.))
  covmat[3, 3] <-  var.y0.boot
  params <- c("b0" = fe.boot[1], "b1" = fe.boot[2], "y0" = y0.boot)
  dm <- deltaMethod(params, g = "(y0 - b0)/b1",  vcov. = covmat)
  var.x0.boot <- dm$SE^2

  ## Approximate predictive pivot
  mu0.boot <-  as.numeric(crossprod(fe.boot, c(1, x0.est)))
  var.mu0.boot <- t(c(1, x0.est)) %*% as.matrix(vcov(.)) %*% c(1, x0.est)
  QI.boot <- (y0.boot - mu0.boot)/sqrt(var.y0.boot + var.mu0.boot)

  c(x0.boot, var.x0.boot, QI.boot)

}
pb <- bootMer(fit.lme4, boot.fun, nsim = 9999, seed = 105)  # run simulation
```

The `bootMer` function returns an object of class `boot` which can then be processed via the **boot** package to obtain the various bootstrap confidence intervals discussed earlier. A basic summary of `pb` is given by

```
library(boot)  # load boot package
summary(pb)

##      R original bootBias bootSE bootMed
## 1 9999     8.02 -0.00219   1.97  8.0186
## 2 9999     3.82  0.05243   1.02  3.7702
## 3 9999     0.00 -0.00810   1.00  0.0016
```

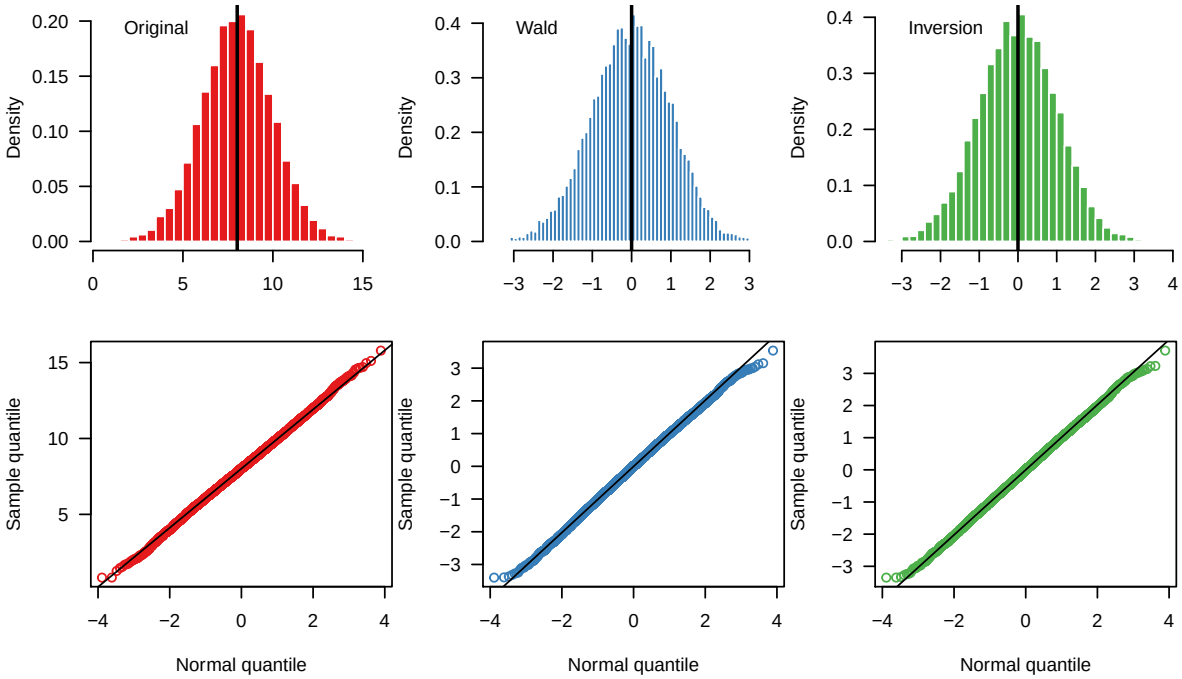

Figure 4: Graphical summary of bootstrap replicates. $R = 9,999$ bootstrap replicates of $\widehat{x}_0$ (left), Q_W (middle), and Q_I (right).

To obtain the percentile and studentized t intervals (see Sections 4.1-4.2), we can use the **boot** package function `boot.ci`:

```
boot.ci(pb, type = c("norm", "perc", "stud"))

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 9999 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = pb, type = c("norm", "perc", "stud"))
##
```

```
## Intervals :
## Level      Normal            Studentized          Percentile
## 95%   ( 4.15, 11.89 )   ( 4.30, 11.99 )   ( 4.05, 11.91 )
## Calculations and Intervals on Original Scale
```

For comparison, we also included the option to compute a bootstrap normal-approximation confidence interval. This interval has the form

$$(\widehat{x}_0 - \texttt{bootBias}) \pm z_{\alpha/2}\texttt{bootSE}$$

where `bootBias` and `bootSE` can be found in the first row of `summary(pb)`. In other words, it is just a Wald-type interval that uses a bias-corrected estimate of x_0, along with a bootstrap estimate of the standard error of $\widehat{x}_0$. While this may be more accurate than the ordinary Wald-based interval (4), it may still not perform well in small sample sizes because of the strict normality assumption. In this example, however, normality does not appear to be an issue.

The bootstrap adjusted inversion interval can be computed as easily as the ordinary inversion interval, except we need to supply `invest` with the estimated quantiles $\widehat{F}_{Q_I}(0.025)$ and $\widehat{F}_{Q_I}(0.975)$:

```
QI.boot <- pb$t[, 3]  # bootsrap replicates of Q_I
qvals <- quantile(QI.boot, c(0.025, 0.975))  # sample quantiles
invest(fit.nlme, y0 = 500, q1 = qvals[1], q2 = qvals[2])

## estimate    lower    upper
##     8.02     4.28    12.02
```

All of the approximate 95% confidence intervals we computed for the true volume of fluid are summarized in Table 3 below.

Method	Estimate	SE	95% Bounds	Length
CI_{wald}	8.016	1.954	(4.185, 11.846)	7.66
CI_{inv}	8.016	—	(4.228, 11.919)	7.691
$CI^{\star}_{percentile}$	8.016	1.974	(4.05, 11.913)	7.863
$CI^{\star}_{wald}$	8.016	1.974	(4.297, 11.99)	7.693
$CI^{\star}_{inv}$	8.016	1.974	(4.278, 12.019)	7.741

Table 3: Summary of results for the bladder volume example. A $\star$ symbol indicates a parametric bootstrap-based confidence interval.

Suppose instead that we observed a new value `HD = 60` and we wish to estimate the true volume of liquid in the patients bladder using the original data. The point estimate is easily obtained using the quadratic formula: $\widehat{x}_0 = 11.105$. Recall, from Figure 1 that each patient has a slightly nonlinear trajectory; thus, there is no reason to expect the sampling distribution of $\widehat{x}_0$ to be normal, or even symmetric in this case. To see that this is indeed the case, we applied the parametric bootstrap. The results are summarized in Figure 5. Clearly, the Wald-based confidence interval (4) will not be accurate in this case. However, the approximate

predictive pivot used in the inversion interval (7) appears reasonably normal. Thus, our recommendation is that, if the number of subjects m is reasonably large (say $m \geq 25$) and the bootstrap replicates are approximately normal (see Figure 4), then the Wald-based and inversion methods are useful. Otherwise, it is probably best to stick with the parametric bootstrap confidence intervals or, if prior information is available, adopt a fully Bayesian approach.

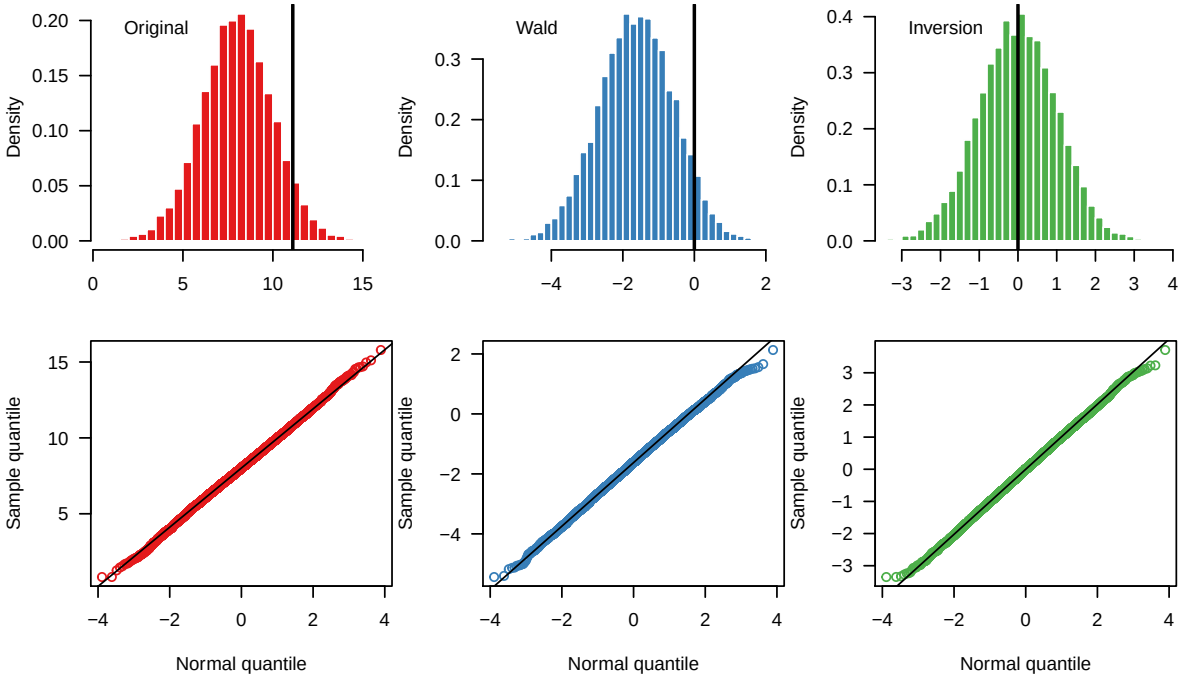

Figure 5: Graphical summary of bootstrap replicates. $R = 9,999$ bootstrap replicates of $\widehat{x}_0$ (left), Q_W (middle), and Q_I (right).

6. Conclusion

We have discussed a number of confidence interval procedures for statistical calibration in linear models with random coefficients with a single level of grouping. We have described two R packages for implementing these procedures: **investr** and **lme4**. The **investr** package can be used for obtaining the asymptotic confidence intervals (i.e., the Wald-based and inversion confidence intervals). We also showed how the **lme4** package can be used to obtain calibration intervals based on a parametric bootstrap using the recently added `bootMer` function. Future work will likely extend the methods discussed in this paper to more complicated cases such as nonlinear mixed-effects models and multi-level hierarchical models (i.e., more than one grouping variable).

References

Bates D, Maechler M, Bolker B, Walker S (2014). *lme4: Linear mixed-effects models using Eigen and S4*. R package version 1.1-6, URL http://CRAN.R-project.org/package=lme4.

Boos D, Stefanski L (2013). *Essential Statistical Inference: Theory and Methods*. Springer Texts in Statistics. Springer London, Limited.

Brown P (1993). *Measurement, Regression, and Calibration*. Oxford science publications. Clarendon Press.

Canty A, Ripley B (2013). *boot: Bootstrap R (S-Plus) Functions*. R package version 1.3-9, URL http://cran.r-project.org/web/packages/boot/index.html.

Casella G, Berger R (2002). *Statistical inference*. Duxbury advanced series in statistics and decision sciences. Thomson Learning.

Davison AC, Hinkley DV (1997). *Bootstrap Methods and their Applications*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge.

Demidenko E (2013). *Mixed Models: Theory and Applications with R*. Wiley.

Efron B (1979). "Bootstrap Methods: Another Look at the Jackknife." *The Annals of Statistics*, **7**(1), 1–26.

Efron B (1982). *The Jackknife, the Bootstrap, and Other Resampling Plans*. Society for Industrial and Applied Mathematics.

Efron B (1987). "Better Bootstrap Confidence Intervals." *Journal of the American Statistical Association*, **82**(397), 171–185.

Fox J, Weisberg S (2011). *An R Companion to Applied Regression*. Second edition. Sage, Thousand Oaks CA. URL http://socserv.socsci.mcmaster.ca/jfox/Books/Companion.

Graybill FA (1976). *Theory and Application of the Linear Model*. Duxbury classic series. Duxbury, Pacific Grove, CA.

Greenwell BM (2013). *investr: Functions for Inverse Estimation with Linear and Nonlinear Regression Models in R*. R package version 1.1.0, URL http://CRAN.R-project.org/package=investr.

Greenwell BM, Kabban CMS (2014). "investr: An R Package for Inverse Estimation." *The R Journal*, **6**(1), ??–??

Hall P (1986). "On the Bootstrap and Confidence Intervals." *The Annals of Statistics*, **14**(4), 1431–1452.

Huet S (2004). *Statistical Tools for Nonlinear Regression: A Practical Guide with S-PLUS and R Examples*. Springer Series in Statistics. Springer.

Jones G, Rocke DM (1999). "Bootstrapping in Controlled Calibration Experiments." *Technometrics*, **41**(3), 224–233.

Laird NM, Ware JH (1982). "Random-Effects Models for Longitudinal Data." *Biometrics*, **38**(4), 963–974.

McCulloch CE, Searle SR, Neuhaus JM (2008). *Generalized, Linear, and Mixed Models*. Wiley.

Oman SD (1998). "Calibration with Random Slopes." *Biometrika*, **85**(2), 439–449.

Osborne C (1991). "Calibration: A Review." *International Statistical Review*, **59**(3), 309–336.

Pinheiro J, Bates D (2000). *Mixed-Effects Models in S and S-PLUS*. Statistics and Computing. Springer.

Pinheiro J, Bates D, DebRoy S, Sarkar D, R Core Team (2013). *nlme: Linear and Nonlinear Mixed Effects Models*. R package version 3.1-110.

Seber G, Wild C (2003). *Nonlinear Regression*. Wiley Series in Probability and Statistics. Wiley.

Wickham H (2011). "The Split-Apply-Combine Strategy for Data Analysis." *Journal of Statistical Software*, **40**(1), 1–29. R package version 1.8.1, URL http://www.jstatsoft.org/v40/i01/.

Affiliation:

Brandon M. Greenwell
Associate Research Engineer
Aptima, Inc.
3100 Presidential Drive, Suite 220
Fairborn, OH 45324
E-mail: greenwell.brandon@gmail.com
URL: https://github.com/w108bmg/investr