José C. Pinheiro
Douglas M. Bates

Mixed-Effects Models in S and S-PLUS

With 172 Illustrations

Springer

José C. Pinheiro
Department of Biostatistics
Novartis Pharmaceuticals
One Health Plaza
East Hanover, NJ 07936-1080
USA
jose.pinheiro@pharma.novartis.com

Douglas M. Bates
Department of Statistics
University of Wisconsin
Madison, WI 53706-1685
USA
bates@stat.wisc.edu

Library of Congress Cataloging-in-Publication Data
Pinheiro, José C.
Mixed-effects models in S and S-PLUS / José C. Pinheiro, Douglas M. Bates
p. cm. — (Statistics and computing)
Includes bibliographical references and index.
ISBN 0-387-98957-9 (alk. paper)
I. Bates, Douglas M. II. Title. III. Series.
QA76.73.S15P56 2000
005.13'3—dc21 99-053566

Printed on acid-free paper.

Printed in the United States of America. (HAM)

9 8 7 6 5 SPIN 10995662

Springer Verlag is a part of *Springer Science+Business Media*

springeronline.com

5

Extending the Basic Linear Mixed-Effects Model

The linear mixed-effects model formulation used in Chapters 2 and 4 allows considerable flexibility in the specification of the random-effects structure, but restricts the within-group errors to be independent, identically distributed random variables with mean zero and constant variance. As illustrated in Chapters 2 and 4, this *basic* linear mixed-effects model provides an adequate model for many different types of grouped data observed in practice. However, there are many applications involving grouped data for which the within-group errors are *heteroscedastic* (i.e., have unequal variances) or are *correlated* or are both heteroscedastic and correlated.

In this chapter, we extend the basic linear mixed-effects model to allow heteroscedastic, correlated within-group errors. We describe how the `lme` function can be used to fit the extended linear mixed-effects model, illustrating its various capabilities through examples. We also introduce a new modeling function, `gls`—for *generalized least squares*, to fit models with heteroscedastic and correlated within-group errors, but with no random effects.

In §5.1, we show how the estimation and computational methods of Chapter 2 can be adapted to the extended model formulation and describe how the variance–covariance structure of the within-group errors can be decomposed into two independent components: a *variance* structure and a *correlation* structure. Variance function models to represent the variance structure component of the within-group errors are described in §5.2 and their use with `lme` is illustrated through examples. Classes of correlation models to represent the correlation structure of the within-group errors are

presented and have their use illustrated in §5.3. In §5.4, the `gls` function is described and illustrated through examples.

5.1 General Formulation of the Extended Model

As described in §2.1.1, the basic single-level linear mixed-effects model (2.1) assumes that the within-group errors $\boldsymbol{\epsilon}_i$ are independent $\mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{I}\right)$ random vectors. The extended single-level linear mixed-effects model relaxes this assumption by allowing heteroscedastic and correlated within-group errors, being expressed as

$$\begin{aligned} \boldsymbol{y}_i &= \boldsymbol{X}_i \boldsymbol{\beta} + \boldsymbol{Z}_i \boldsymbol{b}_i + \boldsymbol{\epsilon}_i, \quad i = 1, \ldots, M, \\ \boldsymbol{b}_i &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Psi}), \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{\Lambda}_i\right), \quad i = 1, \ldots, M, \end{aligned} \tag{5.1}$$

where the $\boldsymbol{\Lambda}_i$ are positive-definite matrices parametrized by a fixed, generally small, set of parameters $\boldsymbol{\lambda}$. As in the basic linear mixed-effects model of §2.1, the within-group errors $\boldsymbol{\epsilon}_i$ are assumed to be independent for different i and independent of the random effects $\boldsymbol{b}_i$. The σ^2 is factored out of the $\boldsymbol{\Lambda}_i$ for computational reasons (it can then be eliminated from the profiled likelihood function).

Similarly, the extended two-level linear mixed-effects model generalizes the basic two-level model (2.2) described in §2.1.2 by letting

$$\boldsymbol{\epsilon}_{ij} \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{\Lambda}_{ij}\right), \quad i = 1, \ldots, M, \quad j = 1, \ldots, M_i,$$

where the $\boldsymbol{\Lambda}_{ij}$ are positive-definite matrices parametrized by a fixed $\boldsymbol{\lambda}$ vector. This readily generalizes to a multilevel model with Q levels of random effects. For simplicity, we concentrate for the remainder of this section on the extended single-level model (5.1), but the results we obtain are easily generalizable to multilevel models with an arbitrary number of levels of random effects.

5.1.1 *Estimation and Computational Methods*

Because $\boldsymbol{\Lambda}_i$ is positive-definite, it admits an invertible square-root $\boldsymbol{\Lambda}_i^{1/2}$ (Thisted, 1988, §3), with inverse $\boldsymbol{\Lambda}_i^{-1/2}$, such that

$$\boldsymbol{\Lambda}_i = \left(\boldsymbol{\Lambda}_i^{1/2}\right)^T \boldsymbol{\Lambda}_i^{1/2} \quad \text{and} \quad \boldsymbol{\Lambda}_i^{-1} = \boldsymbol{\Lambda}_i^{-1/2} \left(\boldsymbol{\Lambda}_i^{-1/2}\right)^T.$$

Letting

$$\begin{aligned} \boldsymbol{y}_i^* &= \left(\boldsymbol{\Lambda}_i^{-1/2}\right)^T \boldsymbol{y}_i, \qquad \boldsymbol{\epsilon}_i^* = \left(\boldsymbol{\Lambda}_i^{-1/2}\right)^T \boldsymbol{\epsilon}_i, \\ \boldsymbol{X}_i^* &= \left(\boldsymbol{\Lambda}_i^{-1/2}\right)^T \boldsymbol{X}_i, \qquad \boldsymbol{Z}_i^* = \left(\boldsymbol{\Lambda}_i^{-1/2}\right)^T \boldsymbol{Z}_i, \end{aligned} \tag{5.2}$$

and noting that

$$\boldsymbol{\epsilon}_i^* \sim \mathcal{N}\left[\left(\boldsymbol{\Lambda}_i^{-1/2}\right)^T \mathbf{0}, \sigma^2 \left(\boldsymbol{\Lambda}_i^{-1/2}\right)^T \boldsymbol{\Lambda}_i \boldsymbol{\Lambda}_i^{-1/2}\right] = \mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{I}\right),$$

we can rewrite (5.1) as

$$\begin{aligned} \boldsymbol{y}_i^* &= \boldsymbol{X}_i^* \boldsymbol{\beta} + \boldsymbol{Z}_i^* \boldsymbol{b}_i + \boldsymbol{\epsilon}_i^*, \quad i = 1, \ldots, M, \\ \boldsymbol{b}_i &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Psi}), \quad \boldsymbol{\epsilon}_i^* \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{I}\right), \quad i = 1, \ldots, M. \end{aligned} \tag{5.3}$$

That is, $\boldsymbol{y}_i^*$ is described by a basic linear mixed-effects model.

Because the differential of the linear transformation $\boldsymbol{y}_i^* = (\boldsymbol{\Lambda}_i^{-1/2})^T \boldsymbol{y}_i$ is simply $d\boldsymbol{y}_i^* = \left|\boldsymbol{\Lambda}_i^{-1/2}\right| d\boldsymbol{y}_i$, the likelihood function L corresponding to the extended linear mixed-effects model (5.1) is expressed as

$$\begin{aligned} L\left(\boldsymbol{\beta}, \boldsymbol{\theta}, \sigma^2, \boldsymbol{\lambda} | \boldsymbol{y}\right) &= \prod_{i=1}^{M} p\left(\boldsymbol{y}_i | \boldsymbol{\beta}, \boldsymbol{\theta}, \sigma^2, \boldsymbol{\lambda}\right) \\ &= \prod_{i=1}^{M} p\left(\boldsymbol{y}_i^* | \boldsymbol{\beta}, \boldsymbol{\theta}, \sigma^2, \boldsymbol{\lambda}\right) \left|\boldsymbol{\Lambda}_i^{-1/2}\right| = L\left(\boldsymbol{\beta}, \boldsymbol{\theta}, \sigma^2, \boldsymbol{\lambda} | \boldsymbol{y}^*\right) \prod_{i=1}^{M} \left|\boldsymbol{\Lambda}_i^{-1/2}\right|, \end{aligned} \tag{5.4}$$

where $p(\cdot)$ denotes a probability density function.

The likelihood function $L\left(\boldsymbol{\beta}, \boldsymbol{\theta}, \sigma^2, \boldsymbol{\lambda} | \boldsymbol{y}^*\right)$ corresponds to a basic linear mixed-effects model and, therefore, all results presented in §2.2 also apply to it. The compact representation of the profiled log-likelihood based on orthogonal-triangular decompositions, described in §2.2.3, can also be used with the likelihood $L\left(\boldsymbol{\beta}, \boldsymbol{\theta}, \sigma^2, \boldsymbol{\lambda} | \boldsymbol{y}^*\right)$, leading to numerically efficient algorithms for maximum likelihood estimation.

The restricted likelihood corresponding to the extended model (5.1) is defined, as in §2.2.5, by integrating out the fixed effects from the likelihood.

$$L_R\left(\boldsymbol{\theta}, \sigma^2, \boldsymbol{\lambda} | \boldsymbol{y}\right) = \int L\left(\boldsymbol{\beta}, \boldsymbol{\theta}, \sigma^2, \boldsymbol{\lambda} | \boldsymbol{y}\right) d\boldsymbol{\beta} = L_R\left(\boldsymbol{\theta}, \sigma^2, \boldsymbol{\lambda} | \boldsymbol{y}^*\right) \prod_{i=1}^{M} \left|\boldsymbol{\Lambda}_i^{-1/2}\right|.$$

The function $L_R\left(\boldsymbol{\beta}, \boldsymbol{\theta}, \sigma^2, \boldsymbol{\lambda} | \boldsymbol{y}^*\right)$ corresponds to a restricted likelihood function of a basic linear mixed-effects model. Hence, the results in §2.2.5 can be used to obtain a numerically efficient representation of the profiled log-restricted-likelihood.

5.1.2 An Extended Linear Model with No Random Effects

The variance–covariance matrix of the response vector $\boldsymbol{y}_i$ in the extended linear mixed-effects model (5.1),

$$\operatorname{Var}\left(\boldsymbol{y}_i\right) = \boldsymbol{\Sigma}_i = \sigma^2 \left(\boldsymbol{Z}_i \boldsymbol{D} \boldsymbol{Z}_i^T + \boldsymbol{\Lambda}_i\right)$$

has two components that can be used to model heteroscedasticity and correlation: a *random-effects* component, given by $\boldsymbol{Z}_i \boldsymbol{D} \boldsymbol{Z}_i^T$, and a *within-group* component, given by $\boldsymbol{\Lambda}_i$. In practice, these two components may "compete" with each other in the model specification, in the sense that similar $\boldsymbol{\Sigma}_i$ matrices may result from a more complex random-effects component being added to a simpler within-group component (say $\boldsymbol{\Lambda}_i = \boldsymbol{I}$), or a simpler random-effects component (say $\boldsymbol{Z}_i \boldsymbol{D} \boldsymbol{Z}_i^T = \sigma_b^2 \mathbf{1}\mathbf{1}^T$) being added to a more complex within-group component. There will generally be a trade-off between the complexity of the two components of $\boldsymbol{\Sigma}_i$ and some care must be exercised to prevent nonidentifiability, or near nonidentifiability, of the parameters in the model.

In some applications, one may wish to avoid incorporating random effects in the model to account for dependence among observations, choosing to use the within-group component $\boldsymbol{\Lambda}_i$ to directly model variance–covariance structure of the response. This results in the simplified version of the extended linear mixed-effects model (5.1)

$$\boldsymbol{y}_i = \boldsymbol{X}_i \boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{\Lambda}_i\right), \quad i = 1, \ldots, M. \tag{5.5}$$

Estimation under this model has been studied extensively in the linear regression literature (Draper and Smith, 1998; Thisted, 1988), usually assuming that the $\boldsymbol{\Lambda}_i$ are known, being referred to as the *generalized least squares* problem.

Using the same transformations as in (5.2), we can re-express (5.5) as a "classic" linear regression model.

$$\boldsymbol{y}_i^* = \boldsymbol{X}_i^* \boldsymbol{\beta} + \boldsymbol{\epsilon}_i^*, \quad \boldsymbol{\epsilon}_i^* \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{I}\right), \quad i = 1, \ldots, M. \tag{5.6}$$

Hence, for fixed $\boldsymbol{\lambda}$, the maximum likelihood estimators of $\boldsymbol{\beta}$ and σ^2 are obtained by solving an *ordinary least-squares* problem. Letting $\boldsymbol{X}^*$ denote the matrix obtained by stacking up the $\boldsymbol{X}_i^*$ matrices, the conditional MLEs of $\boldsymbol{\beta}$ and σ^2 are

$$\begin{aligned} \widehat{\boldsymbol{\beta}}(\boldsymbol{\lambda}) &= \left[\left(\boldsymbol{X}^*\right)^T \boldsymbol{X}^*\right]^{-1} \left(\boldsymbol{X}^*\right)^T \boldsymbol{y}^*, \\ \widehat{\sigma^2}(\boldsymbol{\lambda}) &= \frac{\left\| \boldsymbol{y}^* - \boldsymbol{X}^* \widehat{\boldsymbol{\beta}}(\boldsymbol{\lambda}) \right\|^2}{N}. \end{aligned} \tag{5.7}$$

The profiled log-likelihood corresponding to (5.5), which is a function of $\boldsymbol{\lambda}$ only, is obtained by replacing $\boldsymbol{\beta}$ and σ^2 in the full log-likelihood by their conditional MLEs (5.7), giving

$$\ell\left(\boldsymbol{\lambda} | \boldsymbol{y}\right) = \text{const} - N \log \left\| \boldsymbol{y}^* - \boldsymbol{X}^* \widehat{\boldsymbol{\beta}}(\boldsymbol{\lambda}) \right\| - \frac{1}{2} \sum_{i=1}^{M} \log \left| \boldsymbol{\Lambda}_i \right|. \tag{5.8}$$

Maximum likelihood estimates are obtained by first optimizing the profiled log-likelihood (5.8) over $\boldsymbol{\lambda}$ and then replacing $\boldsymbol{\lambda}$ in (5.7) with its MLE $\widehat{\boldsymbol{\lambda}}$, to

get the MLEs for $\boldsymbol{\beta}$ and σ^2. Orthogonal-triangular decomposition methods similar to the ones described in §2.2.3 can be used to obtain more compact and numerically efficient representations of the profiled log-likelihood (5.8) and the conditional MLEs (5.7), being left to the reader as an exercise.

The restricted likelihood for the extended linear model (5.5) is defined as in the extended linear mixed-effects model (5.1), by integrating the parameter vector $\boldsymbol{\beta}$ out of the full likelihood. Using the results in Harville (1977), we write the profiled log-restricted-likelihood corresponding to (5.5) as

$$\ell_R(\boldsymbol{\lambda}|\boldsymbol{y}) = \text{const} - (N-p)\log\left\|\boldsymbol{y}^* - \boldsymbol{X}^*\widehat{\boldsymbol{\beta}}(\boldsymbol{\lambda})\right\| - \frac{1}{2}\log\left|(\boldsymbol{X}^*)^T\boldsymbol{X}^*\right| - \frac{1}{2}\sum_{i=1}^{M}\log|\boldsymbol{\Lambda}_i|,$$

with p denoting the dimension of $\boldsymbol{\beta}$. The REML estimator of σ^2 is defined as in (5.7), but with the N in the denominator replaced by $N-p$.

The modeling function `gls` in the nlme library is used to fit the extended linear model (5.5), using either maximum likelihood or restricted maximum likelihood. We describe the use of this function in §5.4, after presenting the capabilities available in the nlme library for modeling heteroscedasticity and correlation.

5.1.3 *Decomposing the Within-Group Variance–Covariance Structure*

The $\boldsymbol{\Lambda}_i$ matrices can always be decomposed into a product of simpler matrices

$$\boldsymbol{\Lambda}_i = \boldsymbol{V}_i\boldsymbol{C}_i\boldsymbol{V}_i, \tag{5.9}$$

where $\boldsymbol{V}_i$ is diagonal and $\boldsymbol{C}_i$ is a correlation matrix, that is, a positive-definite matrix with all diagonal elements equal to one. The matrix $\boldsymbol{V}_i$ in (5.9) is not uniquely defined, as we can multiply any number of its rows by -1 and still get the same decomposition. To ensure uniqueness, we require that all the diagonal elements of $\boldsymbol{V}_i$ be positive.

It is easy to verify that

$$\text{Var}(\epsilon_{ij}) = \sigma^2[\boldsymbol{V}_i]_{jj}^2, \qquad \text{cor}(\epsilon_{ij}, \epsilon_{jk}) = [\boldsymbol{C}_i]_{jk},$$

so that $\boldsymbol{V}_i$ describes the variance and $\boldsymbol{C}_i$ describes the correlation of the within-group errors $\boldsymbol{\epsilon}_i$. This decomposition of $\boldsymbol{\Lambda}_i$ into a *variance structure* component and a *correlation structure* component is convenient both theoretically and computationally. It allows us to model and develop code for the two structures separately and to combine them into a flexible family of models for the within-group variance–covariance. In §5.2 we describe variance function structures to represent the variance component $\boldsymbol{V}_i$ and in §5.3 we present correlation structures for the correlation component $\boldsymbol{C}_i$.

5.2 Variance Functions for Modeling Heteroscedasticity

Variance functions are used to model the variance structure of the within-group errors using covariates. They have been studied in detail in the context of mixed-effects models by Davidian and Giltinan (1995) and in the context of the extended linear model (5.5) by Carroll and Ruppert (1988).

Following Davidian and Giltinan (1995, Ch. 4), we define the general variance function model for the within-group errors in the extended single-level linear mixed-effects model (5.1) as

$$\mathrm{Var}\left(\epsilon_{ij}|\boldsymbol{b}_i\right) = \sigma^2 g^2\left(\mu_{ij}, \boldsymbol{v}_{ij}, \boldsymbol{\delta}\right), \quad i = 1, \ldots, M, \quad j = 1, \ldots, n_i, \tag{5.10}$$

where $\mu_{ij} = \mathrm{E}\left[y_{ij}|\boldsymbol{b}_i\right]$, $\boldsymbol{v}_{ij}$ is a vector of *variance covariates*, $\boldsymbol{\delta}$ is a vector of *variance parameters* and $g(\cdot)$ is the variance function, assumed continuous in $\boldsymbol{\delta}$. For example, if the within-group variability is believed to increase with some power of the absolute value of a covariate v_{ij}, we can write the variance model as

$$\mathrm{Var}\left(\epsilon_{ij}|\boldsymbol{b}_i\right) = \sigma^2 \left|v_{ij}\right|^{2\delta}.$$

The variance function in this case is $g(x, y) = |x|^y$ and the covariate v_{ij} can be the expected value μ_{ij}.

The single-level variance function model (5.10) can be generalized to multilevel models. For example, the variance function model for a two-level model is

$$\mathrm{Var}\left(\epsilon_{ijk}|\boldsymbol{b}_{i,j}, \boldsymbol{b}_{ij}\right) = \sigma^2 g^2\left(\mu_{ijk}, \boldsymbol{v}_{ijk}, \boldsymbol{\delta}\right),$$
$$i = 1, \ldots, M, \quad j = 1, \ldots, M_i, \quad k = 1, \ldots, n_{ij},$$

where $\mu_{ijk} = \mathrm{E}\left[y_{ijk}|\boldsymbol{b}_{i,j}, \boldsymbol{b}_{ij}\right]$. We concentrate, for the remainder of this section, in the single-level model (5.1), but all results presented here easily generalize to multilevel models.

The variance function formulation (5.10) is very flexible and intuitive, because it allows the within-group variance to depend on the fixed effects, $\boldsymbol{\beta}$, and the random effects, $\boldsymbol{b}_i$, through the expected values, μ_{ij}. However, as discussed in Davidian and Giltinan (1995, Ch. 4), it poses some theoretical and computational difficulties, as the within-group errors and the random effects can no longer be assumed to be independent. Under the assumption that $\mathrm{E}\left[\boldsymbol{\epsilon}_i|\boldsymbol{b}_i\right] = \mathbf{0}$, it is easy to verify that $\mathrm{Var}\left(\epsilon_{ij}\right) = \mathrm{E}\left[\mathrm{Var}\left(\epsilon_{ij}|\boldsymbol{b}_i\right)\right]$, so that the dependence on the unobserved random effects can be avoided by integrating them out of the variance model. Because the variance function g is generally nonlinear in $\boldsymbol{b}_i$, integrating the random effects out of the variance model (5.10) does not lead to a computationally feasible optimization

procedure. Instead, we proceed as in Davidian and Giltinan (1995, Ch. 6), and use an approximate variance model in which the expected values μ_{ij} are replaced by their BLUPs $\widehat{\mu}_{ij} = \boldsymbol{x}_{ij}^T\boldsymbol{\beta} + \boldsymbol{z}_{ij}^T\widehat{\boldsymbol{b}}_i$, where $\boldsymbol{x}_{ij}$ and $\boldsymbol{z}_{ij}$ denote, respectively, the jth rows of $\boldsymbol{X}_i$ and $\boldsymbol{Z}_i$,

$$\text{Var}(\epsilon_{ij}) \simeq \sigma^2 g^2(\widehat{\mu}_{ij}, \boldsymbol{v}_{ij}, \boldsymbol{\delta}), \quad i = 1, \ldots, M, \quad j = 1, \ldots, n_i. \tag{5.11}$$

Under this approximation, the within-group errors are assumed independent of the random effects, as in (5.1), and the results in §5.1.1 can still be used. Note that, if the conditional variance model (5.10) does not depend on μ_{ij}, (5.11) gives the exact marginal variance and no approximation is required.

When the conditional variance model (5.10) depends on μ_{ij}, the optimization algorithm follows an "iteratively reweighted" scheme: for given $\boldsymbol{\beta}^{(t)}$, $\boldsymbol{\theta}^{(t)}$, $\boldsymbol{\lambda}^{(t)}$, the corresponding BLUPs $\widehat{\mu}_{ij}^{(t)}$ are obtained and held fixed while the objective function is optimized to produce new estimates $\boldsymbol{\beta}^{(t+1)}$, $\boldsymbol{\theta}^{(t+1)}$, $\boldsymbol{\lambda}^{(t+1)}$ which, in turn, give updated BLUPs $\widehat{\mu}_{ij}^{(t+1)}$, with the process iterating until convergence. The resulting estimates approximate the (restricted) maximum likelihood estimates. When the variance model does not involve μ_{ij}, the (restricted) likelihood can be directly optimized, producing the exact (restricted) maximum likelihood estimates.

Variance functions for the extended linear model (5.5) are similarly defined, but, because no random effects are present, the model for the marginal variance does not involve any approximations, being expressed as

$$\text{Var}(\epsilon_{ij}) = \sigma^2 g^2(\mu_{ij}, \boldsymbol{v}_{ij}, \boldsymbol{\delta}), \quad i = 1, \ldots, M, \quad j = 1, \ldots, n_i, \tag{5.12}$$

where $\mu_{ij} = \text{E}[y_{ij}] = \boldsymbol{x}_{ij}^T\boldsymbol{\beta}$ and $\boldsymbol{v}_{ij}$ and $\boldsymbol{\delta}$ are defined as in (5.10). This coincides with the variance function model proposed in Carroll and Ruppert (1988, §3). When the variance model (5.12) depends on μ_{ij}, the optimization algorithm parallels the linear mixed-effects model iteratively reweighted scheme described before: for given $\boldsymbol{\beta}^{(t)}$ and $\boldsymbol{\lambda}^{(t)}$ estimated $\mu_{ij}^{(t)}$ are obtained and held fixed while the objective function is optimized to produce new estimates $\boldsymbol{\beta}^{(t+1)}$ and $\boldsymbol{\lambda}^{(t+1)}$, which, in turn, give updated $\mu_{ij}^{(t+1)}$, with the process iterating until convergence. This estimation procedure is known in the literature (Carroll and Ruppert, 1988, §3.2) as *pseudo-likelihood* estimation (when the objective function corresponds to a log-likelihood), or *pseudo-restricted-likelihood* estimation (when the objective function corresponds to a log-restricted-likelihood). If the variance model (5.12) does not involve μ_{ij}, the (restricted) likelihood can be directly optimized and the exact (restricted) maximum likelihood estimates obtained.

TABLE 5.1. Standard varFunc classes.

`varFixed`	fixed variance
`varIdent`	different variances per stratum
`varPower`	power of covariate
`varExp`	exponential of covariate
`varConstPower`	constant plus power of covariate
`varComb`	combination of variance functions

5.2.1 *Variance Functions in nlme: The varFunc Classes*

The nlme library provides a set of classes of variance functions, the varFunc classes, that are used to specify within-group variance models in either the extended linear mixed-effects model (5.1), or the extended linear model (5.5). Table 5.1 lists the standard varFunc classes included in the nlme library. The varFunc constructors have the same name as their corresponding classes.

The two main arguments for most of the varFunc constructors are `value` and `form`. The first specifies the values of the variance parameters $\boldsymbol{\delta}$ and the second is a one-sided formula specifying the variance covariate v and, optionally, a stratification variable for the variance parameters—different parameters are used for each level of the stratification variable. For example, to specify `age` as a variance covariate and to have different variance parameters for each level of `Sex`, we would use

```
form = ~ age | Sex
```

The fitted object may be referenced in `form` by the symbol `.`, so, for example,

```
form = ~ fitted(.)
```

specifies the fitted values as the variance covariate.

Several methods are available for each varFunc class, including `initialize`, which initializes the variance covariates and stratification variables, and `varWeights`, which extracts the *variance weights*, defined as the inverse of the variance function values. We now describe and illustrate each of the standard varFunc classes.

varFixed

This class represents a variance function with no parameters and a single variance covariate, being used when the within-group variance is known up to a proportionality constant. The `varFixed` constructor takes a single argument, `value`, which is a one-sided formula that evaluates to the desired variance model. No stratification variables or expressions involving the symbol `.` are allowed in `value`. For example, suppose it is known that

the within-group variance increases linearly with `age`

$$\mathrm{Var}(\epsilon_{ij}) = \sigma^2 \mathrm{age}_{ij},$$

corresponding to a variance function

$$g(\mathrm{age}_{ij}) = \sqrt{\mathrm{age}_{ij}}$$

and being represented as

```
> vf1Fixed <- varFixed( ~ age )
```

The variance covariate is calculated in the `initialize` method, which, besides the varFixed object, also requires a `data` argument naming a data frame in which to evaluate the variance covariate formula. Initialization of a varFunc object is generally done inside the modeling function (e.g., `lme`, `gls`) which uses it.

```
> vf1Fixed <- initialize( vf1Fixed, data = Orthodont )
> varWeights( vf1Fixed )
  [1] 0.35355 0.31623 0.28868 0.26726 0.35355 0.31623 0.28868
. . .
```

The variance weights in this case are given by $1/\sqrt{\mathrm{age}_{ij}}$.

`varIdent`

This class represents a variance model with different variances for each level of a stratification variable s, taking values in the set $\{1, 2, \ldots, S\}$,

$$\mathrm{Var}(\epsilon_{ij}) = \sigma^2 \delta^2_{s_{ij}}, \tag{5.13}$$

corresponding to the variance function

$$g(s_{ij}, \boldsymbol{\delta}) = \delta_{s_{ij}}.$$

The variance model (5.13) uses $S+1$ parameters to represent S variances and, therefore, is not identifiable. To achieve identifiability, we need to impose some restriction on the variance parameters $\boldsymbol{\delta}$. We use $\delta_1 = 1$, so that δ_l, $l = 2, \ldots, S$ represent the ratio between the standard deviations of the lth stratum and the first stratum. By definition, $\delta_l > 0$, $l = 2, \ldots, S$.

The main arguments to the `varIdent` constructor are `value`, a named numeric vector or a named list specifying the values and the strata of the $\boldsymbol{\delta}$ parameters that are allowed to vary in the optimization, and `form`, a one-sided formula of the form `~1|s` specifying the stratification variable `s`. Some, or all, of the variance parameters may be set to fixed values (which do not vary during the optimization) using the argument `fixed`. This may be given either as named numeric vector, or a named list, specifying the values and the strata of the $\boldsymbol{\delta}$ that are to remain fixed during the optimization.

For example, to specify and initialize a variance function with different variances per gender for the `Orthodont` data described in §1.4.1, with the initial value of the δ parameter corresponding to `Female` set to 0.5, we use

```
> vf1Ident <- varIdent( c(Female = 0.5), ~ 1 | Sex )
> vf1Ident <- initialize( vf1Ident, Orthodont )
> varWeights( vf1Ident )
 Male Male Male Male Male Male Male Male Male Male Male Male
    1    1    1    1    1    1    1    1    1    1    1    1
. . .
 Female Female
      2      2
```

If the ratio between the `Female` and `Male` standard deviations, given by δ, is to be kept fixed at 0.5 and not allowed to vary during the optimization, we can use instead

```
> vf2Ident <- varIdent( form =  ~ 1 | Sex, fixed = c(Female = 0.5))
> vf2Ident <- initialize( vf2Ident, Orthodont )
> varWeights( vf2Ident )
 Male Male Male Male Male Male Male Male Male Male Male Male
    1    1    1    1    1    1    1    1    1    1    1    1
. . .
 Female Female
      2      2
```

It is possible to specify several stratification variables simultaneously in `form`, separated by the * operator. The levels of the different stratification variables are pasted together and a different δ is used for each combination of levels. For example, to specify a variance function with different variances for each `age` and `Sex` combination we can use

```
> vf3Ident <- varIdent( form = ~ 1 | Sex * age )
> vf3Ident <- initialize( vf3Ident, Orthodont )
> varWeights( vf3Ident )
 Male*8 Male*10 Male*12 Male*14 Male*8 Male*10 Male*12 Male*14
      1       1       1       1      1       1       1       1
. . .
 Female*12 Female*14
         1         1
```

By default, all variance parameters are initialized to 1, corresponding to equal variance weights of 1 being assigned to all strata.

`varPower`

The variance model represented by this class is

$$\text{Var}(\epsilon_{ij}) = \sigma^2 |v_{ij}|^{2\delta}, \tag{5.14}$$

corresponding to the variance function

$$g(v_{ij}, \delta) = |v_{ij}|^{\delta},$$

which is a power of the absolute value of the variance covariate. The parameter δ is unrestricted (i.e., may take any value in the real line) so (5.14) can model cases where the variance increases or decreases with the absolute value of the variance covariate. Note that, when $v_{ij} = 0$ and $\delta > 0$, the variance function is 0 and the variance weight is undefined. Therefore, this class of variance functions should not be used with variance covariates that may assume the value 0.

The main arguments to the `varPower` constructor are `value` and `form`, which specify, respectively, an initial value for δ, when this is allowed to vary in the optimization, and a one-sided formula with the variance covariate. By default, `value = 0`, corresponding to equal variance weights of 1, and `form = ~fitted(.)`, corresponding to a variance covariate given by the fitted values. For example, to specify a variance model with the δ parameter initially set to 1, and allowed to vary in the optimization, and the fitted values as the variance covariate, we use

```
> vf1Power <- varPower( 1 )
> formula( vf1Power )
~ fitted(.)
```

The `fixed` argument can be used to set δ to a fixed value, which does not change in the optimization. For example,

```
> vf2Power <- varPower( fixed = 0.5 )
```

specifies a model in which the variance increases linearly with the fitted values.

An optional stratification variable, or several stratification variables separated by `*`, may be included in the `form` argument, with a different δ being used for each stratum. This corresponds to the following generalization of (5.14)

$$\mathrm{Var}(\epsilon_{ij}) = \sigma^2 |v_{ij}|^{2\delta_{s_{ij}}}, \quad g(v_{ij}, s_{ij}, \boldsymbol{\delta}) = |v_{ij}|^{\delta_{s_{ij}}}.$$

When a stratification variable is included in `form`, the arguments `value` and `fixed` must be either named vectors, or named lists. For example, to specify a model for the `Orthodont` data in which the variance increases linearly with the fitted values for `Males` and stays constant for `Females`, with both parameters held fixed during the optimization, we use

```
> vf3Power <- varPower( form = ~ fitted(.) | Sex,
+   fixed = list(Male = 0.5, Female = 0) )
```

varExp

The variance model represented by this class is

$$\mathrm{Var}(\epsilon_{ij}) = \sigma^2 \exp(2\delta v_{ij}), \tag{5.15}$$

corresponding to the variance function

$$g(v_{ij}, \delta) = \exp(\delta v_{ij}),$$

which is an exponential function of the variance covariate. The parameter δ is unrestricted, so (5.15) can model cases where the variance increases or decreases with the variance covariate. There are no restrictions on the variance covariate, which, in particular, may take the value 0.

The arguments `value`, `form`, and `fixed` to the `varExp` constructor are defined and declared as in `varPower`, assuming also the same default values. An optional stratification variable, or stratification variables separated by `*`, may be specified in `form`, corresponding to the following generalization of (5.15)

$$\mathrm{Var}(\epsilon_{ij}) = \sigma^2 \exp\left(2\delta_{s_{ij}} v_{ij}\right), \quad g(v_{ij}, s_{ij}, \boldsymbol{\delta}) = \exp\left(\delta_{s_{ij}} v_{ij}\right).$$

As with `varPower`, when a stratification variable is included in `form`, the arguments `value` and `fixed` must be either named vectors, or named lists. For example, a variance model in which the `Male` variance increases exponentially with `age`, but the `Female` variance is held at a constant value is expressed as

```
> vf1Exp <- varExp( form = ~ age | Sex, fixed = c(Female = 0) )
```

`varConstPower`

The variance model represented by this class is

$$\mathrm{Var}(\epsilon_{ij}) = \sigma^2 \left(\delta_1 + |v_{ij}|^{\delta_2}\right)^2, \tag{5.16}$$

corresponding to the variance function

$$g(v_{ij}, \boldsymbol{\delta}) = \delta_1 + |v_{ij}|^{\delta_2},$$

which is a constant plus a power of the absolute value of the variance covariate. δ_1 is restricted to be positive and δ_2 is unrestricted. If $\delta_2 > 0$, which will generally be the case in practice, the varConstPower variance function is approximately constant and equal to δ_1, when the variance covariate is close to 0, and increases with the absolute value of the variance covariate as it gets away from 0. This generally gives a more realistic model than the varPower model, in cases when the variance covariate takes values close or equal to 0.

Initial values for parameters that are allowed to vary during the optimization are specified through the arguments `const` and `power`. The former is used for δ_1 and the latter for δ_2. By default, `const = 1` and `power = 0`, corresponding to constant variance weights equal to 1. The `form` argument is defined as in `varPower` and `varExp`, assuming the same default value. The

argument `fixed` is given as a list with components `const` and `power` and may be used to set either, or both, of the variance parameters to a fixed value. For example, to specify a variance function with δ_1 fixed at the value 1, with δ_2 allowed to vary but initialized to 0.5, and the fitted values as the variance covariate, we use

```
> vf1ConstPower <- varConstPower( power = 0.5,
+       fixed = list(const = 1) )
```

An optional stratification variable, or several stratification variables separated by `*`, may be included in the `form` argument, with different δ_1 and δ_2 being used for each stratum. This corresponds to the following generalization of (5.16)

$$\mathrm{Var}\left(\epsilon_{ij}\right) = \sigma^2 \left(\delta_{1,s_{ij}} + |v_{ij}|^{\delta_{2,s_{ij}}}\right)^2, \quad g\left(v_{ij}, s_{ij}, \boldsymbol{\delta}\right) = \delta_{2,s_{ij}} + |v_{ij}|^{\delta_{2,s_{ij}}}.$$

When a stratification variable is included in `form`, the arguments `const`, `power` and the components of `fixed` must be either named vectors, or named lists.

`varComb`

This class allows the combination of two, or more, variance models, by multiplying together the corresponding variance functions. For example, a variance model in which the variance is proportional to an exponential function of a variance covariate, but in which the proportionality constant varies according to the levels of an stratification variable s can be expressed as a product of a varIdent variance function and a varExp variance function.

$$\begin{aligned} \mathrm{Var}\left(\epsilon_{ij}\right) &= \sigma^2 \delta_{1,s_{ij}}^2 \exp\left(2\delta_2 v_{ij}\right) = \sigma^2 g_1^2\left(s_{ij}, \boldsymbol{\delta}_1\right) g_2^2\left(v_{ij}, \delta_2\right), \\ g_1\left(s_{ij}, \boldsymbol{\delta}_1\right) &= \delta_{1,s_{ij}}, \qquad g_2\left(v_{ij}, \delta_2\right) = \exp\left(\delta_2 v_{ij}\right). \end{aligned} \tag{5.17}$$

The `varComb` constructor can take any number of varFunc objects as arguments. For example, to represent (5.17), with `Sex` as the stratification variable and `age` as the variance covariate, we use

```
> vf1Comb <- varComb( varIdent(c(Female = 0.5), ~ 1 | Sex),
+                     varExp(1, ~ age) )
> vf1Comb <- initialize( vf1Comb, Orthodont )
> varWeights( vf1Comb )
    1   2        3        4     5   6        7        8     9
0.125 0.1 0.083333 0.071429 0.125 0.1 0.083333 0.071429 0.125
. . .
   98      99     100  101 102     103     104  105 106     107
0.2 0.16667 0.14286 0.25 0.2 0.16667 0.14286 0.25 0.2 0.16667
    108
0.14286
```

The varComb variance weights correspond to the product of the individual variance weights of each of its varFunc objects. In the case of `vf1Comb`, these are given by $1/\sqrt{\text{age}_{ij}}$ for `Male` (observations 1–64) and $2/\sqrt{\text{age}_{ij}}$ for `Female` (observations 65–108).

New varFunc classes, representing user-defined variance functions, can be added to the set of standard classes in Table 5.1 and used with the modeling functions in the nlme library. For this, one must specify a constructor function, generally with the same name as the class, and, at a minimum, methods for the functions `coef`, `coef<-`, and `initialize`. The `varPower` constructor and methods can serve as templates for these.

5.2.2 *Using Variance Functions with* `lme`

Variance functions are specified in the `lme` function using the `weights` argument. By default, `weights = NULL`, corresponding to a homoscedastic variance model for the within-group errors. Variance models can be specified in `weights` either as a one-sided formula, in which case it is passed as the single argument to the `varFixed` constructor, or as a varFunc object, created using the standard constructors described in §5.2.1, or a user-defined constructor. In this section, we describe the use of variance models in `lme` through the analysis of two examples of grouped data with heteroscedastic within-group errors.

High-Flux Hemodialyzer Ultrafiltration Rates

Vonesh and Carter (1992) describe and analyze data measured on high-flux hemodialyzers to assess their *in vivo* ultrafiltration characteristics. The ultrafiltration rates (in ml/hr) of 20 high-flux dialyzers were measured at 7 ascending transmembrane pressures (in dmHg). The *in vitro* evaluation of the dialyzers used bovine blood at flow rates of either 200 dl/min or 300 dl/min. These data are also described in Appendix A.6 and are included in the nlme library as the groupedData object `Dialyzer`.

The plots of the ultrafiltration rates versus transmembrane pressure by bovine blood flow rate, displayed in Figure 5.1, reveal that the ultrafiltration rate increases with transmembrane pressure and that higher ultrafiltration rates are attained with the 300 dl/min blood flow dialyzers. These plots also indicate that the variability in the ultrafiltration rates increases with transmembrane pressure.

Vonesh and Carter (1992) use a nonlinear model to represent the relationship between ultrafiltration rate and transmembrane pressure. An alternative analysis is presented in Littell et al. (1996), who compare several linear mixed-effects models and extended linear models to represent the ultrafiltration rate y_{ij} at the jth transmembrane pressure x_{ij} for the ith subject. The best linear mixed-effects model indicated by their analysis

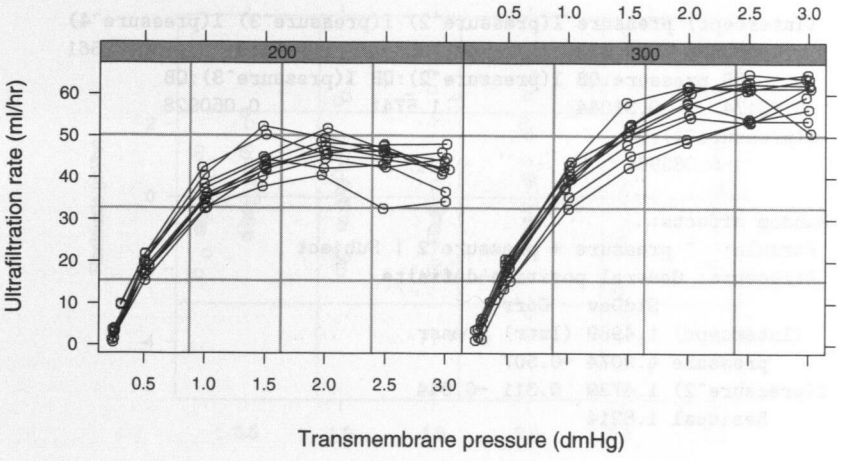

FIGURE 5.1. Hemodialyzer ultrafiltration rates (in ml/hr) measured at 7 different transmembrane pressures (in dmHg) on 20 high-flux dialyzers. *In vitro* evaluation of dialyzers based on bovine blood flow rates of 200 dl/min and 300 dl/min.

is

$$y_{ij} = (\beta_0 + \gamma_0 Q_i + b_{0i}) + (\beta_1 + \gamma_1 Q_i + b_{1i})\, x_{ij} + (\beta_2 + \gamma_2 Q_i + b_{2i})\, x_{ij}^2 + (\beta_3 + \gamma_3 Q_i)\, x_{ij}^3 + (\beta_4 + \gamma_4 Q_i)\, x_{ij}^4 + \epsilon_{ij}, \tag{5.18}$$

$$\boldsymbol{b}_i = \begin{bmatrix} b_{0i} \\ b_{1i} \\ b_{2i} \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Psi}), \quad \epsilon_{ij} \sim \mathcal{N}(0, \sigma^2),$$

where Q_i is a binary variable taking values -1 for 200 dl/min hemodialyzers and 1 for 300 dl/min hemodialyzers; β_0, β_1, β_2, β_3, and β_4 are, respectively, the intercept, linear, quadratic, cubic, and quartic fixed effects averaged over the levels of Q; γ_i is the blood flow effect associated with the fixed effect β_i; $\boldsymbol{b}_i$ is the vector of random effects, assumed independent for different i; and ϵ_{ij} is the within-group error, assumed independent for different i, j and independent of the random effects.

We fit the homoscedastic linear mixed-effects model (5.18) with

```
> fm1Dial.lme <-
+   lme(rate ~(pressure + pressure^2 + pressure^3 + pressure^4)*QB,
+       Dialyzer, ~ pressure + pressure^2)
> fm1Dial.lme
Linear mixed-effects model fit by REML
  Data: Dialyzer
  Log-restricted-likelihood: -326.39
  Fixed: rate ~(pressure + pressure^2 + pressure^3 + pressure^4)*QB
```

```
 (Intercept) pressure I(pressure^2) I(pressure^3) I(pressure^4)
     -16.598   88.673       -42.732        9.2165      -0.77561
       QB pressure:QB I(pressure^2):QB I(pressure^3):QB
 -0.63174     0.31044           1.5741         0.050928
 I(pressure^4):QB
        -0.085967

Random effects:
 Formula:  ~ pressure + pressure^2 | Subject
 Structure: General positive-definite
             StdDev   Corr
  (Intercept) 1.4989 (Intr) pressr
     pressure 4.9074 -0.507
I(pressure^2) 1.4739  0.311 -0.944
     Residual 1.8214

Number of Observations: 140
Number of Groups: 20
```

The primary tool for investigating within-group heteroscedasticity is the plots of residuals against the fitted values and other candidate variance covariates. In the hemodialyzer example, the transmembrane pressure is a natural candidate for the variance covariate. The corresponding residuals plot, obtained with

```
> plot( fm1Dial.lme, resid(.) ~ pressure, abline = 0 ) # Figure 5.2
```

and displayed in Figure 5.2, confirms that the within-group variability increases with transmembrane pressure.

Because of its flexibility, the varPower variance function is a common choice for modeling monotonic heteroscedasticity, when the variance covariate is bounded away from zero (transmembrane pressure varies in the data between 0.235 dmHg and 3.030 dmHg). The corresponding model only differs from (5.18) in that the within-group errors are allowed to be heteroscedastic with variance model

$$\text{Var}(\epsilon_{ij}) = \sigma^2 x_{ij}^{2\delta} \tag{5.19}$$

and we fit it with

```
> fm2Dial.lme <- update( fm1Dial.lme,
+                        weights = varPower(form = ~ pressure) )
> fm2Dial.lme
Linear mixed-effects model fit by REML
  Data: Dialyzer
  Log-restricted-likelihood: -309.51
  Fixed: rate ~(pressure + pressure^2 + pressure^3 + pressure^4)*QB
 (Intercept) pressure I(pressure^2) I(pressure^3) I(pressure^4)
      -17.68   93.711       -49.186        12.245      -1.2426
```

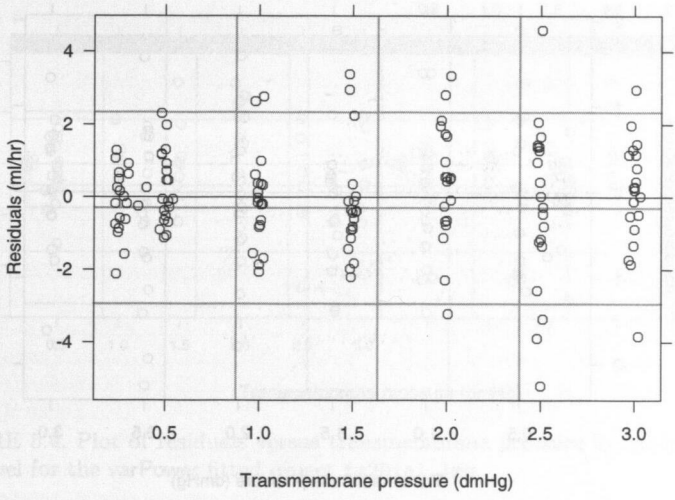

FIGURE 5.2. Plot of residuals versus transmembrane pressure for the homoscedastic fitted object `fm1Dial.lme`.

```
      QB pressure:QB I(pressure^2):QB I(pressure^3):QB
-0.92072     1.3528         0.48071         0.49118
I(pressure^4):QB
     -0.14624

Random effects:
 Formula:  ~ pressure + pressure^2 | Subject
 Structure: General positive-definite
              StdDev   Corr
  (Intercept) 1.8569 (Intr) pressr
     pressure 5.3282 -0.522
I(pressure^2) 1.6483  0.362 -0.954
     Residual 1.2627

Variance function:
 Structure: Power of variance covariate
 Formula:  ~ pressure
 Parameter estimates:
   power
 0.74923
Number of Observations: 140
Number of Groups: 20
```

The `anova` method can be used to test the significance of the heteroscedastic model (5.19).

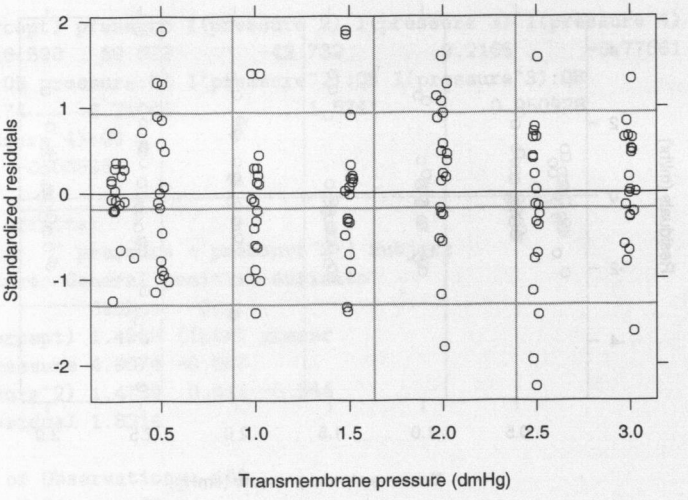

FIGURE 5.3. Plot of standardized residuals versus transmembrane pressure for the varPower fitted object `fm2Dial.lme`.

```
> anova( fm1Dial.lme, fm2Dial.lme )
             Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm1Dial.lme      1 17 686.78 735.53 -326.39
fm2Dial.lme      2 18 655.01 706.63 -309.51 1 vs 2   33.77  <.0001
```

As expected, there is a highly significant increase in the log-likelihood associated with the inclusion of the varPower variance function. The plot of the standardized residuals, defined as $r_{ij} = (y_{ij} - \widehat{y}_{ij}) / (\widehat{\sigma} x_{ij}^{\widehat{\delta}})$ in this case, versus the variance covariate is used to graphically assess the adequacy of the variance model.

```
> plot( fm2Dial.lme, resid(., type = "p") ~ pressure,
+       abline = 0 )                                  # Figure 5.3
```

The resulting plot, displayed in Figure 5.3, reveals a reasonably homogeneous pattern of variability for the standardized residuals, indicating that the varPower model successfully describes the within-group variance.

We assess the variability of the variance parameter estimate $\widehat{\delta}$ with the `intervals` method.

```
> intervals( fm2Dial.lme )
. . .
 Variance function:
       lower    est.  upper
power 0.4079 0.74923 1.0906
. . .
```

The resulting confidence interval is based on a normal approximation for the distribution of the (restricted) maximum likelihood estimators, with

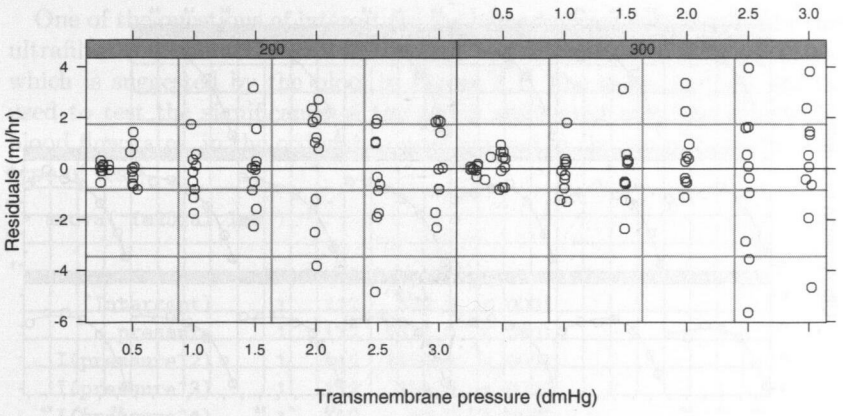

FIGURE 5.4. Plot of residuals versus transmembrane pressure by bovine blood flow level for the varPower fitted object fm2Dial.lme.

an approximate variance–covariance matrix given by the inverse of the observed information matrix evaluated at the converged values.

An interesting question about the hemodialyzer data is whether the within-group variability depends on the bovine blood flow level. We can investigate this graphically with the plots of the raw residuals versus transmembrane pressure by blood flow level, obtained with

```
> plot(fm2Dial.lme, resid(.) ~ pressure|QB, abline = 0)   # Fig. 5.4
```

and displayed in Figure 5.4. Note that the pattern of increasing variability is still present in the raw residuals, as we did not transform the data, but, instead, incorporated the within-group heteroscedasticity in the model. The heteroscedastic patterns seem the same for both blood flow levels. We can test this formally using

```
> fm3Dial.lme <- update(fm2Dial.lme,
+                       weights=varPower(form = ~ pressure | QB))
> fm3Dial.lme
. . .
Variance function:
 Structure: Power of variance covariate, different strata
 Formula: ~ pressure | QB
 Parameter estimates:
     200      300
 0.64775 0.83777
. . .
> anova( fm2Dial.lme, fm3Dial.lme )
           Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm2Dial.lme     1 18 655.01 706.63 -309.51
fm3Dial.lme     2 19 656.30 710.78 -309.15 1 vs 2 0.71091  0.3991
```

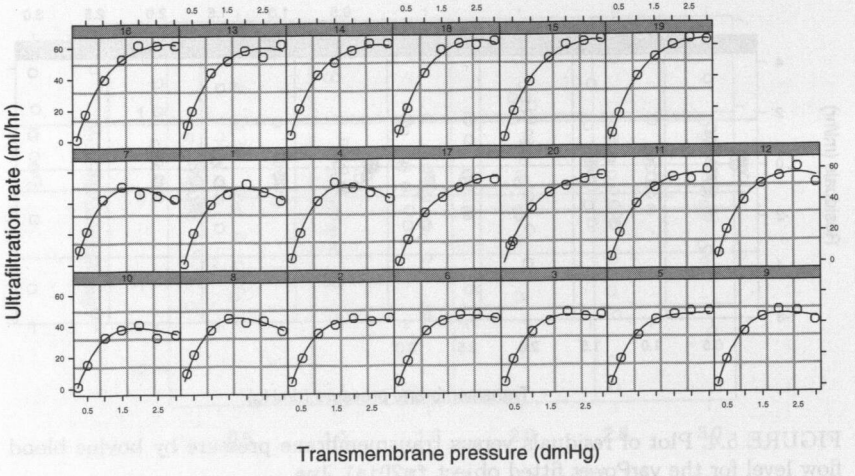

FIGURE 5.5. Plot of predicted ultrafiltration rates versus transmembrane pressure by subject corresponding to the fitted object `fm2Dial.lme`.

As expected, there is no evidence of a significant effect of blood flow in the within-group variability.

Because the transmembrane pressure takes values relatively close to zero, we may wish to investigate if a varConstPower variance model

$$\mathrm{Var}(\epsilon_{ij}) = \sigma^2 \left(\delta_1 + x_{ij}^{\delta_2}\right)^2$$

would give a more appropriate representation of the within-group variance.

```
> fm4Dial.lme <- update(fm2Dial.lme,
+                       weights = varConstPower(form = ~ pressure))
> anova( fm2Dial.lme, fm4Dial.lme )
             Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm2Dial.lme      1 18 655.01 706.63 -309.51
fm4Dial.lme      2 19 656.85 711.34 -309.43 1 vs 2 0.15915  0.6899
```

The nearly identical log-likelihood values indicate that there is no need for the extra parameter associated with varConstPower and that the simpler varPower model should be maintained.

A final assessment of the heteroscedastic version of the linear mixed-effects model (5.18) with within-group variance model (5.19) is provided by the plot of the augmented predictions by subject, obtained with

```
> plot( augPred(fm2Dial.lme), grid = T )                    # Figure 5.5
```

and displayed in Figure 5.5. The predicted values closely match the observed ultrafiltration rates, attesting the adequacy of the model.

One of the questions of interest for the hemodialyzer data is whether the ultrafiltration characteristics differ with the evaluation blood flow rates, which is suggested by the plots in Figure 5.1. The `anova` method can be used to test the significance of the terms associated with the evaluation blood flow rates, in the order they were entered in the model (a sequential type of test).

```
> anova( fm2Dial.lme )
. . .
                  numDF denDF F-value p-value
     (Intercept)      1   112   552.9  <.0001
        pressure      1   112  2328.6  <.0001
   I(pressure^2)      1   112  1174.6  <.0001
   I(pressure^3)      1   112   359.9  <.0001
   I(pressure^4)      1   112    12.5  0.0006
              QB      1    18     4.8  0.0414
     pressure:QB      1   112    80.1  <.0001
I(pressure^2):QB      1   112     1.4  0.2477
I(pressure^3):QB      1   112     2.2  0.1370
I(pressure^4):QB      1   112     0.2  0.6840
. . .
```

The large p-values associated with terms of degree greater than or equal to 2 involving the variable `QB` suggest that they are not needed in the model. We can verify their *joint* significance with

```
> anova( fm2Dial.lme, Terms = 8:10 )
F-test for: I(pressure^2):QB, I(pressure^3):QB, I(pressure^4):QB
  numDF denDF F-value p-value
1     3   112  1.2536  0.2939
```

The large p-value for the F-test confirms that these terms could be eliminated from the model.

Body Weight Growth in Rats

As a second example to illustrate the use of variance functions with `lme`, we revisit the `BodyWeight` data introduced in §3.2.1 and described in Hand and Crowder (1996, Table A.1), on the body weights of rats measured over 64 days. The body weights of the rats (in grams) are measured on day 1 and every seven days thereafter, until day 64, with an extra measurement on day 44. There are three groups of rats, each on a different diet. These data are also described in Appendix A.3 and are included in the `nlme` library as the `groupedData` object `BodyWeight`.

The plots of the body weights versus time by diet, shown in Figure 5.6, indicate strong differences among the three diet groups. There is also evidence of a rat in diet group 2 with an unusually high initial body weight.

The body weights appear to grow linearly with time, possibly with different intercepts and slopes for each diet, and with intercept and slope

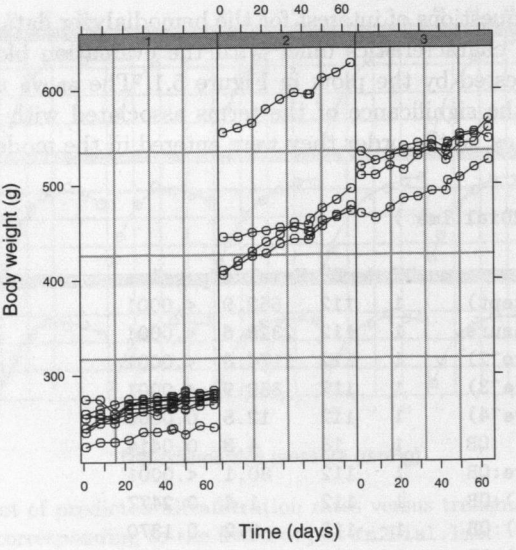

FIGURE 5.6. Body weights of rats measured over a period of 64 days. The rats are divided into three groups on different diets.

random effects to account for rat-to-rat variation. We express this as the linear mixed-effects model

$$y_{ij} = (\beta_0 + \gamma_{02} D_{2i} + \gamma_{03} D_{3i} + b_{0i}) + (\beta_1 + \gamma_{12} D_{2i} + \gamma_{13} D_{3i} + b_{1i})\, t_{ij} + \epsilon_{ij},$$

$$\boldsymbol{b}_i = \begin{bmatrix} b_{0i} \\ b_{1i} \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Psi}), \quad \epsilon_{ij} \sim \mathcal{N}(0, \sigma^2), \tag{5.20}$$

where D_{2i} is a binary variable taking the value 1 if the ith rat receives Diet 2; D_{3i} is a binary indicator variable for Diet 3; β_0 and β_1 are, respectively, the average intercept and the average slope for rats under Diet 1; γ_{0k} is the average difference in intercept between rats under Diet k and rats under Diet 1; γ_{1k} is the average difference in slope between rats under Diet k and rats under Diet 1; $\boldsymbol{b}_i$ is the vector of random effects, assumed independent for different i; and ϵ_{ij} is the within-group error, assumed independent for different i, j and independent of the random effects.

To fit (5.20), we first need to reset the `contrasts` option to use the `contr.treatment` parameterization for factors, as described in §1.2.1,

```
> options( contrasts = c("contr.treatment", "contr.poly") )
```

and then use

```
> fm1BW.lme <- lme( weight ~ Time * Diet, BodyWeight,
+                   random = ~ Time )
```

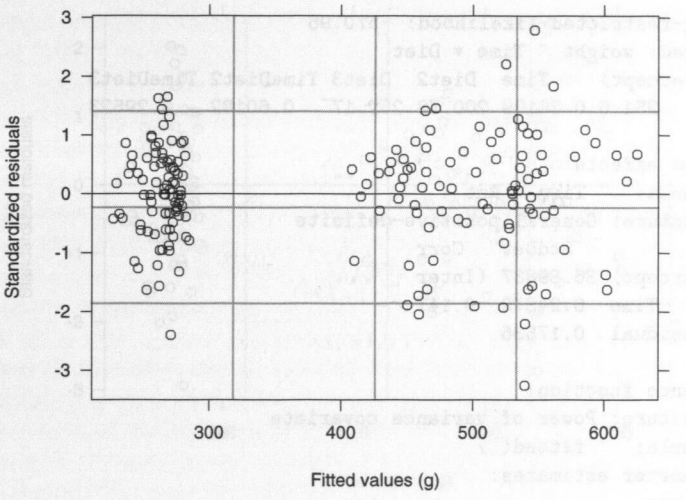

FIGURE 5.7. Plot of standardized residuals versus fitted values for the homoscedastic fitted object `fm1BW.lme`.

```
> fm1BW.lme
Linear mixed-effects model fit by REML
  Data: BodyWeight
  Log-restricted-likelihood: -575.86
  Fixed: weight ~ Time * Diet
 (Intercept)    Time  Diet2  Diet3 TimeDiet2 TimeDiet3
      251.65 0.35964 200.67 252.07   0.60584   0.29834

Random effects:
 Formula:  ~ Time | Rat
 Structure: General positive-definite
             StdDev   Corr
(Intercept) 36.93907 (Inter
       Time  0.24841 -0.149
   Residual  4.44361

Number of Observations: 176
Number of Groups: 16
```

The plot of the standardized residuals versus the fitted values, displayed in Figure 5.7, gives clear indication of within-group heteroscedasticity. Because the fitted values are bounded away from zero, we can use the varPower variance function to model the heteroscedasticity.

```
> fm2BW.lme <- update( fm1BW.lme, weights = varPower() )
> fm2BW.lme
Linear mixed-effects model fit by REML
  Data: BodyWeight
```

```
  Log-restricted-likelihood: -570.96
  Fixed: weight ~ Time * Diet
(Intercept)    Time   Diet2   Diet3 TimeDiet2 TimeDiet3
      251.6 0.36109  200.78  252.17   0.60182   0.29523

Random effects:
 Formula:  ~ Time | Rat
 Structure: General positive-definite
             StdDev   Corr
(Intercept) 36.89887 (Inter
       Time  0.24373 -0.147
   Residual  0.17536

Variance function:
 Structure: Power of variance covariate
 Formula:  ~ fitted(.)
 Parameter estimates:
   power
 0.54266
Number of Observations: 176
Number of Groups: 16
```

Note that the `form` argument did not need to be specified in the call to `varPower`, because its default value, `~fitted(.)`, corresponds to the desired variance covariate.

The plot of the standardized residuals versus fitted values for the heteroscedastic fit corresponding to `fm2BW.lme`, displayed in Figure 5.8, indicates that the varPower variance function adequately represents the within-group heteroscedasticity.

We can test the significance of the variance parameter in the varPower model using the `anova` method, which, as expected, strongly rejects the assumption of homoscedasticity (i.e., $\delta = 0$).

```
> anova( fm1BW.lme, fm2BW.lme )
          Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm1BW.lme     1 10 1171.7 1203.1 -575.86
fm2BW.lme     2 11 1163.9 1198.4 -570.96 1 vs 2  9.7984  0.0017
```

The primary question of interest for the `BodyWeight` data is whether the growth rates differ significantly among diets. Because of the parametrization used in (5.20), the `summary` method only provides tests for differences between Diets 1 and 2 (γ_{12} in (5.20)) and between Diets 1 and 3 (γ_{13} in (5.20)).

```
> summary( fm2BW.lme )
. . .
Fixed effects: weight ~ Time * Diet
             Value Std.Error  DF t-value p-value
(Intercept) 251.60    13.068 157  19.254  <.0001
```

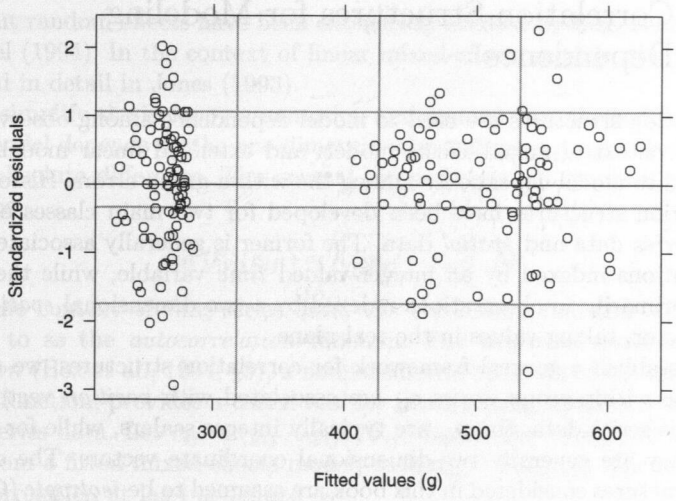

FIGURE 5.8. Plot of standardized residuals versus fitted values for the varPower fitted object `fm2BW.lme`.

```
          Time   0.36     0.088 157   4.084  0.0001
         Diet2 200.78    22.657  13   8.862  <.0001
         Diet3 252.17    22.662  13  11.127  <.0001
    TimeDiet2   0.60     0.155 157   3.871  0.0002
    TimeDiet3   0.30     0.156 157   1.893  0.0602
. . .
```

There appears to be a significant increase in growth rate associated with Diet 2 (`TimeDiet2`) and a borderline significant increase in growth rate for Diet 3 (`TimeDiet3`). We can test the difference in growth rates between Diets 2 and 3 using the `anova` method.

```
> anova( fm2BW.lme, L = c(TimeDiet2 = 1, TimeDiet3 = -1) )
F-test for linear combination(s)
 TimeDiet2 TimeDiet3
         1        -1
  numDF denDF F-value p-value
1     1   157  2.8608  0.0927
```

The argument `L` is used to specify contrasts of coefficients to be tested as equal to zero. The names of the elements in `L` must correspond to coefficients in the model. There does not seem to be a significant difference in growth rate between Diets 2 and 3.

5.3 Correlation Structures for Modeling Dependence

Correlation structures are used to model dependence among observations. In the context of mixed-effects models and extended linear models, they are used to model dependence among the within-group errors. Historically, correlation structures have been developed for two main classes of data: *time-series* data and *spatial* data. The former is generally associated with observations indexed by an integer-valued *time* variable, while the latter refers primarily to observations indexed by a two-dimensional *spatial location* vector, taking values in the real plane.

To establish a general framework for correlation structures, we assume that the within-group errors ϵ_{ij} are associated with *position* vectors $\boldsymbol{p}_{ij}$. For time series data, the $\boldsymbol{p}_{ij}$ are typically integer scalars, while for spatial data they are generally two-dimensional coordinate vectors. The correlation structures considered in this book are assumed to be *isotropic* (Cressie, 1993, §2.3.1); that is, the correlation between two within-group errors $\epsilon_{ij}, \epsilon_{ij'}$ is assumed to depend on the corresponding position vectors $\boldsymbol{p}_{ij}, \boldsymbol{p}_{ij'}$ only through some distance between them, say $d\left(\boldsymbol{p}_{ij}, \boldsymbol{p}_{ij'}\right)$, and not on the particular values they assume. The general within-group correlation structure for single-level grouping is expressed, for $i = 1, \ldots, M$ and $j, j' = 1, \ldots, n_i$, as

$$\operatorname{cor}\left(\epsilon_{ij}, \epsilon_{ij'}\right) = h\left[d(\boldsymbol{p}_{ij}, \boldsymbol{p}_{ij'}),\ \boldsymbol{\rho}\right], \tag{5.21}$$

where $\boldsymbol{\rho}$ is a vector of *correlation parameters* and $h(\cdot)$ is a correlation function taking values between -1 and 1, assumed continuous in $\boldsymbol{\rho}$, and such that $h\left(0, \boldsymbol{\rho}\right) = 1$, that is, if two observations have identical position vectors, they are the same observation and therefore have correlation 1.

The single-level correlation model (5.21) can be easily generalized to multilevel grouping. For example, the correlation model for two nested levels of grouping is

$$\operatorname{cor}\left(\epsilon_{ijk}, \epsilon_{ijk'}\right) = h\left[d(\boldsymbol{p}_{ijk}, \boldsymbol{p}_{ijk'}),\ \boldsymbol{\rho}\right],$$
$$i = 1, \ldots, M, \quad j = 1, \ldots, M_i, \quad k, k' = 1, \ldots, n_{ij}.$$

Note that the correlation model applies to within-group errors within the same innermost level of grouping. We concentrate, for the remainder of this section, in the single-level correlation model (5.21), but all correlation structures presented here can be easily extended to multilevel models.

5.3.1 Serial Correlation Structures

Serial correlation structures are used to model dependence in time-series data, that is, data observed sequentially over time and indexed by a one-dimensional position vector. Serial correlation structures for linear models

without random effects have been extensively studied by Box, Jenkins and Reinsel (1994). In the context of linear mixed-effects models, they are described in detail in Jones (1993).

We simplify the isotropy assumption and assume that the serial correlation model depends on the one-dimensional positions $p_{ij}, p_{ij'}$ only through their absolute difference. The general serial correlation model is then defined as

$$\operatorname{cor}(\epsilon_{ij}, \epsilon_{ij'}) = h(|p_{ij} - p_{ij'}|, \boldsymbol{\rho}).$$

In the context of time-series data, the correlation function $h(\cdot)$ is referred to as the *autocorrelation* function. The *empirical autocorrelation function* (Box et al., 1994, §3), a nonparametric estimate of the autocorrelation function, provides a useful tool for investigating serial correlation in time-series data. Let $r_{ij} = (y_{ij} - \widehat{y}_{ij})/\widehat{\sigma}_{ij}$, denote the standardized residuals from a fitted mixed-effects model, with $\sigma_{ij}^2 = \operatorname{Var}(\epsilon_{ij})$. The empirical autocorrelation at lag l is defined as

$$\widehat{\rho}(l) = \frac{\sum_{i=1}^{M} \sum_{j=1}^{n_i - l} r_{ij} r_{i(j+l)} / N(l)}{\sum_{i=1}^{M} \sum_{j=1}^{n_i} r_{ij}^2 / N(0)}, \tag{5.22}$$

where $N(l)$ is the number of residual pairs used in the summation defining the numerator of $\widehat{\rho}(l)$.

Serial correlation structures typically require that the data be observed at integer time points and do not easily generalize to continuous position vectors. We describe below some of the most common serial correlation structures used in practice, all of which are implemented in the nlme library.

Compound Symmetry

This is the simplest serial correlation structure, which assumes equal correlation among all within-group errors pertaining to the same group. The corresponding correlation model is

$$\operatorname{cor}(\epsilon_{ij}, \epsilon_{ij'}) = \rho, \quad \forall j \neq j', \quad h(k, \rho) = \rho, \quad k = 1, 2, \ldots, \tag{5.23}$$

where the single correlation parameter ρ is generally referred to as the *intraclass* correlation coefficient.

The variance–covariance matrix for the ith response vector in a single-level linear mixed-effects model with independent and identically distributed within-group errors, with variance σ^2, and a single intercept random effect, with variance σ_b^2, is $\sigma^2 \boldsymbol{I} + \sigma_b^2 \mathbf{1}\mathbf{1}^T$, corresponding to the correlation matrix $\sigma^2/(\sigma_b^2 + \sigma^2)\, \boldsymbol{I} + \sigma_b^2/(\sigma_b^2 + \sigma^2)\, \mathbf{1}\mathbf{1}^T$. This is equivalent to a compound symmetry structure with intraclass correlation $\rho = \sigma_b^2/(\sigma_b^2 + \sigma^2)$, indicating that the correlation structure defined by this linear mixed-effects model is a particular case of (5.23). The two correlation models are not equivalent, however, because the intraclass correlation in the linear mixed-effects

model can only take values between 0 and 1, while (5.23) allows ρ to take negative values (to have a positive-definite compound symmetry correlation structure, it is only required that $\rho > -1/[\max_{i \leq M}(n_i) - 1]$).

The compound symmetry correlation model tends to be too simplistic for practical applications involving time-series data, as, in general, it is more realistic to assume a model in which the correlation between two observations decreases, in absolute value, with their distance. It is a useful model for applications involving short time series per group, or when all observations within a group are collected at the same time, as in split-plot experiments.

General

This structure represents the other extreme in complexity to the compound symmetry structure. Each correlation in the data is represented by a different parameter, corresponding to the correlation function

$$h(k, \boldsymbol{\rho}) = \rho_k, \quad k = 1, 2, \ldots . \tag{5.24}$$

Because the number of parameters in (5.24) increases quadratically with the maximum number of observations within a group, this correlation structure will often lead to overparameterized models. When there are relatively few observations per group, the general correlation structure is useful as an exploratory tool to determine a more parsimonious correlation model.

Autoregressive–Moving Average

This family of correlation structures, described in detail in Box et al. (1994), includes different classes of linear stationary models: *autoregressive* models, *moving average* models, and mixture of autoregressive–moving average models. These are also called *Box and Jenkins* models.

Autoregressive–moving average correlation models assume that the data are observed at integer time points and, for simplicity, we use the notation ϵ_t to refer to an observation taken at time t. The distance, or *lag*, between two observations ϵ_t and ϵ_s is given by $|t - s|$. So lag-1 refers to observations one time unit apart and so on.

Autoregressive models express the current observation as a linear function of previous observations plus a homoscedastic noise term, a_t, centered at 0 ($\mathrm{E}[a_t] = 0$) and assumed independent of the previous observations.

$$\epsilon_t = \phi_1 \epsilon_{t-1} + \cdots + \phi_p \epsilon_{t-p} + a_t. \tag{5.25}$$

The number of past observations included in the linear model (5.25), p, is called the *order* of the autoregressive model, which is denoted by $AR(p)$. There are p correlation parameters in an $AR(p)$ model, given by $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_p)$.

The $AR(1)$ model is the simplest (and one of the most useful) autoregressive model. Its correlation function decreases in absolute value exponentially with lag.

$$h(k, \phi) = \phi^k, \quad k = 0, 1, \ldots . \tag{5.26}$$

The single correlation parameter, ϕ, represents the lag-1 correlation and takes values between -1 and 1. The $AR(1)$ model is one of the few serial correlation structures that can be generalized to continuous time measurements. We define the *continuous time* $AR(1)$ correlation function (Jones, 1993, 3.3), denoted $CAR(1)$, as

$$h(s, \phi) = \phi^s, \quad s \geq 0, \quad \phi \geq 0. \tag{5.27}$$

Note that the single correlation parameter ϕ in (5.27) must be non-negative.

For autoregressive models of order greater than 1, the correlation function does not admit a simple representation as in (5.26), being defined recursively through the difference equation (Box et al., 1994, §3.2.2)

$$h(k, \boldsymbol{\phi}) = \phi_1 h(|k-1|, \boldsymbol{\phi}) + \cdots + \phi_p h(|k-p|, \boldsymbol{\phi}), \quad k = 1, 2, \ldots .$$

Moving average correlation models assume that the current observation is a linear function of independent and identically distributed noise terms.

$$\epsilon_t = \theta_1 a_{t-1} + \cdots + \theta_q a_{t-q} + a_t. \tag{5.28}$$

The number of noise terms included in the linear model (5.28), q, is called the order of the moving average model, which is denoted by $MA(q)$. There are q correlation parameters in an $MA(q)$ model, given by $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_q)$.

The correlation function for an $MA(q)$ model is

$$h(k, \boldsymbol{\theta}) = \begin{cases} \frac{\theta_k + \theta_1 \theta_{k-1} + \cdots + \theta_{k-q}\theta_q}{1 + \theta_1^2 + \cdots + \theta_q^2}, & k = 1, \ldots, q, \\ 0, & k = q+1, q+2, \ldots . \end{cases}$$

Observations more than q time units apart are uncorrelated, as they do not share any common noise terms a_t.

Strategies for estimating the order of autoregressive and moving average models in time series applications are discussed in Box et al. (1994, §3).

Mixed autoregressive–moving average models, called *ARMA* models, are obtained by combining together an autoregressive model and a moving average model.

$$\epsilon_t = \sum_{i=1}^{p} \phi_i \epsilon_{t-i} + \sum_{j=1}^{q} \theta_j a_{t-j} + a_t.$$

There are $p + q$ correlation parameters $\boldsymbol{\rho}$ in an $ARMA(p, q)$ model, corresponding to the combination of the p autoregressive parameters $\boldsymbol{\phi} =$

$(\phi_1, \ldots, \phi_p)$ and the q moving average parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_q)$. By convention, $ARMA(p, 0) = AR(p)$ and $ARMA(0, q) = MA(q)$, so that both autoregressive and moving average models are particular examples of the general $ARMA$ model.

The correlation function for an $ARMA(p, q)$ model behaves like the correlation function of an $AR(p)$ model for lags greater than q and like an $AR(p)$ correlation function plus a term related to the moving average part of the model, between lags 1 and q. It is obtained using the recursive relations

$$h(k, \boldsymbol{\rho}) = \begin{cases} \phi_1 h(|k-1|, \boldsymbol{\rho}) + \cdots + \phi_p h(|k-p|, \boldsymbol{\rho}) + \\ \quad \theta_1 \psi(k-1, \boldsymbol{\rho}) + \cdots + \theta_q \psi(k-q, \boldsymbol{\rho}), & k = 1, \ldots, q, \\ \phi_1 h(|k-1|, \boldsymbol{\rho}) + \cdots + \phi_p h(|k-p|, \boldsymbol{\rho}), & k = q+1, q+2, \ldots, \end{cases}$$

where $\psi(k, \boldsymbol{\phi}, \boldsymbol{\theta}) = \mathrm{E}\left[\epsilon_{t-k} a_t\right] / \mathrm{Var}\left(\epsilon_t\right)$. Note that $\psi(k, \boldsymbol{\phi}, \boldsymbol{\theta}) = 0, \quad k = 1, 2, \ldots$, as, in this case, ϵ_{t-k} and a_t are independent and $\mathrm{E}\left[a_t\right] = 0$.

5.3.2 *Spatial Correlation Structures*

Spatial correlation structures were originally proposed to model dependence in data indexed by continuous two-dimensional position vectors, such as *geostatistical data*, *lattice data*, and *point patterns*. Because the isotropic spatial correlation structures we consider are continuous functions of some distance between position vectors, they are easily generalized to any finite number of position dimensions. In particular, they can be used with time series data. The basic reference for spatial correlation structures used with linear models with no random effects is Cressie (1993). Spatial correlation structures in the context of mixed-effects models are described at length in Diggle et al. (1994).

For simplicity of notation, we denote by $\epsilon_{\boldsymbol{x}}$ the observation taken at position $\boldsymbol{x} = (x_1, \ldots, x_r)^T$. Any distance metric may be used with isotropic spatial correlation structures, the most common being the *Euclidean*, or L_2, distance, defined as $d_E\left(\epsilon_{\boldsymbol{x}}, \epsilon_{\boldsymbol{y}}\right) = \sqrt{\sum_{i=1}^{r} \left(x_i - y_i\right)^2}$. Other popular choices are the *Manhattan*, or L_1, distance $d_{\mathrm{Man}}\left(\epsilon_{\boldsymbol{x}}, \epsilon_{\boldsymbol{y}}\right) = \sum_{i=1}^{r} |x_i - y_i|$ and the *maximum* distance $d_{\mathrm{Max}}\left(\epsilon_{\boldsymbol{x}}, \epsilon_{\boldsymbol{y}}\right) = \max_{i=1,\ldots,r} |x_i - y_i|$.

Semivariogram

Spatial correlation structures are generally represented by their *semivariogram*, instead of their correlation function (Cressie, 1993, §2.3.1). The semivariogram of an isotropic spatial correlation structure with a distance function $d(\cdot)$ is defined as

$$\gamma\left[d\left(\epsilon_{\boldsymbol{x}}, \epsilon_{\boldsymbol{y}}\right), \boldsymbol{\lambda}\right] = \tfrac{1}{2}\mathrm{Var}\left(\epsilon_{\boldsymbol{x}} - \epsilon_{\boldsymbol{y}}\right) = \tfrac{1}{2}\mathrm{E}\left[\epsilon_{\boldsymbol{x}} - \epsilon_{\boldsymbol{y}}\right]^2, \tag{5.29}$$

with the last equality following from $\mathrm{E}\left[\epsilon_{\boldsymbol{x}}\right] = \mathrm{E}\left[\epsilon_{\boldsymbol{y}}\right] = 0$. The within-group errors can be standardized to have unit variance, without changing

their correlation structure. So, without loss of generality, we assume that $\mathrm{Var}(\epsilon_{\boldsymbol{x}}) = 1$, $\forall \boldsymbol{x}$. In this case, $\gamma(\cdot)$ will depend only on the correlation parameters $\boldsymbol{\rho}$ and it is easy to verify that

$$\gamma(s, \boldsymbol{\rho}) = 1 - h(s, \boldsymbol{\rho}).$$

It follows from $h(0, \boldsymbol{\rho}) = 1$ that $\gamma(0, \boldsymbol{\rho}) = 0$. To account for abrupt changes at very small distances, it is desirable, in some applications, to allow a discontinuity in $\gamma(\cdot)$ at 0, so that $\gamma(s, \boldsymbol{\rho}) \to c_0$, when $s \downarrow 0$, with $0 < c_0 < 1$. This is called a *nugget effect* in the spatial statistics literature (Cressie, 1993, §2.3.1). In terms of the correlation function, the nugget effect translates into $h(s, \boldsymbol{\rho}) \to 1 - c_0$ as $s \downarrow 0$. It is easy to obtain a correlation function that incorporates a nugget effect from a correlation function that is continuous in s.

$$h_{\text{nugg}}(s, c_0, \boldsymbol{\rho}) = \begin{cases} (1 - c_0)\, h_{\text{cont}}(s, \boldsymbol{\rho}), & s > 0, \\ 1, & s = 0. \end{cases}$$

The standardized residuals $r_{ij} = (y_{ij} - \widehat{y}_{ij}) / \widehat{\sigma}_{ij}$, with $\sigma_{ij}^2 = \mathrm{Var}(\epsilon_{ij})$, are the primary quantities used for estimating the semivariogram. The *classical* estimator of the semivariogram (Matheron, 1962) is

$$\widehat{\gamma}(s) = \frac{1}{2N(s)} \sum_{i=1}^{M} \sum_{d(\boldsymbol{p}_{ij}, \boldsymbol{p}_{ij'}) = s} (r_{ij} - r_{ij'})^2, \tag{5.30}$$

where $N(s)$ denotes the number of residual pairs at a distance s of each other. Because $\widehat{\gamma}(s)$ uses the squared differences between residual pairs, it can be quite sensitive to outliers. Furthermore, because each residual r_{ij} appears in $n_i - 1$ squared differences in (5.30), a single outlier can affect the estimation of the semivariogram at several distances. A robust estimator of the semivariogram, proposed by Cressie and Hawkins (1980), uses the square-root differences to reduce the influence of outliers.

$$\bar{\gamma}(s) = \left(\frac{1}{2N(s)} \sum_{i=1}^{M} \sum_{d(\boldsymbol{p}_{ij}, \boldsymbol{p}_{ij'}) = s} |r_{ij} - r_{ij'}|^{1/2} \right)^4 / \left(0.457 + 0.494/N(s)\right). \tag{5.31}$$

Some Isotropic Variogram Models

Cressie (1993, §2.3.1) describes an extensive collection of isotropic variogram models and give conditions for their validity. The single-parameter models in Table 5.2 are a subset of the collection in Cressie (1993), with the *Linear* variogram model modified so that it is bounded in s. $I(s < \rho)$ denotes a binary variable taking value 1 when $s < \rho$ and 0 otherwise. Most

TABLE 5.2. Some isotropic variogram models for spatial correlation structures.

Exponential	$\gamma(s,\rho) = 1 - \exp(-s/\rho)$
Gaussian	$\gamma(s,\rho) = 1 - \exp\left[-(s/\rho)^2\right]$
Linear	$\gamma(s,\rho) = 1 - (1 - s/\rho)\, I(s < \rho)$
Rational quadratic	$\gamma(s,\rho) = (s/\rho)^2 / \left[1 + (s/\rho)^2\right]$
Spherical	$\gamma(s,\rho) = 1 - \left[1 - 1.5\,(s/\rho) + 0.5\,(s/\rho)^3\right] I(s < \rho)$

of the models in Table 5.2 are also described in Littell et al. (1996, §9.3.1). The correlation parameter ρ is generally referred to as the *range* in the spatial statistics literature (Littell et al., 1996, §9.3).

For one-dimensional position vectors, the exponential spatial correlation structure is equivalent to the $CAR(1)$ structure (5.27). This is easily verified by defining $\phi = \exp(-1/\rho)$ and noting the correlation function associated with the exponential structure is expressed as $h(s,\phi) = \phi^s$. The exponential correlation model can be regarded as a multivariate generalization of the the $CAR(1)$ model.

Correlation functions for the structures in Table 5.2 may be obtained using the relation $h(s,\rho) = 1 - \gamma(s,\rho)$. A nugget effect c_0 may be added to any of the variogram models, using

$$\gamma_{\text{nugg}}(s, c_o, \rho) = \begin{cases} c_0 + (1 - c_0)\,\gamma(s,\rho), & s > 0, \\ 0, & s = 0. \end{cases}$$

Figure 5.9 displays plots of the semivariograms models in Table 5.2, corresponding to a range of $\rho = 1$ and a nugget effect of $c_0 = 0.1$ The semivariograms increase monotonically with distance and vary between 0 and 1, corresponding to non-negative correlation functions that decrease monotonically with distance. All of the spatial correlation models in Table 5.2 are implemented in the nlme library as corStruct classes, described in the next section.

5.3.3 *Correlation Structures in nlme: The* `corStruct` *Classes*

The nlme library provides a set of classes of correlation structures, the corStruct classes, which are used to specify within-group correlation models in either the extended linear mixed-effects model (5.1), or the extended linear model (5.5). Table 5.3 lists the standard corStruct classes in the nlme library. The corStruct constructors have the same name as their corresponding classes.

The two main arguments to most of the corStruct constructors are `value` and `form`. The first specifies the values of the correlation parameters and

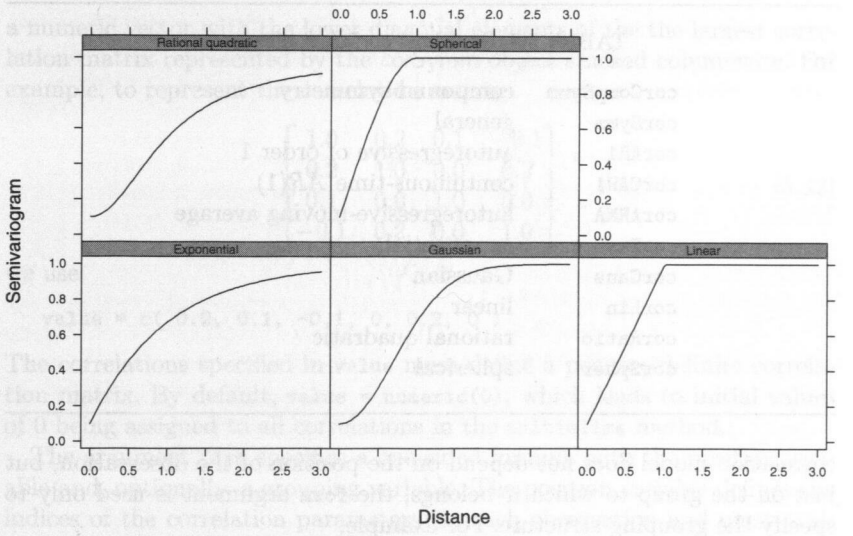

FIGURE 5.9. Plots of semivariogram versus distance for the isotropic spatial correlation models in Table 5.2 with range = 1 and nugget effect = 0.1.

the second is a one-sided formula specifying the position vector and, optionally, a grouping variable for the data—observations in different groups are assumed independent. For example, to specify `age` as a position variable and to have `Subject` defining the grouping of the data, we use

```
form = ~ age | Subject
```

A two-dimensional position vector with coordinates `x` and `y` is specified with

```
form = ~ x + y
```

The argument `fixed`, available in all corStruct constructors, may be used to specify fixed correlation structures, which coefficients are not allowed to change during the numerical optimization in the modeling functions. If `fixed = TRUE`, the coefficients in the structure are fixed. Default is `fixed = FALSE`.

Several methods are available for each corStruct class, including `initialize`, which initializes position vectors and grouping variables, and `corMatrix`, which extracts the within-group correlation matrices. We now describe and illustrate the standard corStruct classes.

corCompSymm

This class implements the compound symmetry correlation structure (5.23. The argument `value` is used to initialize the intraclass correlation coefficient, assuming a default value of 0. Because the compound symmetry

TABLE 5.3. Standard corStruct classes.

`corCompSymm`	compound symmetry
`corSymm`	general
`corAR1`	autoregressive of order 1
`corCAR1`	continuous-time $AR(1)$
`corARMA`	autoregressive-moving average
`corExp`	exponential
`corGaus`	Gaussian
`corLin`	linear
`corRatio`	rational quadratic
`corSpher`	spherical

correlation model does not depend on the position of the observation, but just on the group to which it belongs, the `form` argument is used only to specify the grouping structure. For example,

```
> cs1CompSymm <- corCompSymm( value = 0.3, form = ~ 1 | Subject )
```

specifies a compound symmetry structure with intraclass correlation of 0.3 and grouping defined by `Subject`. The variable used on the left hand side of the `|` operator in `form` is ignored, so

```
> cs2CompSymm <- corCompSymm( value = 0.3, form = ~ age | Subject )
```

gives a `corCompSymm` object identical to `cs1CompSymm`. By default, `form = ~1`, implying that all observations belong to the same group.

The `initialize` method is used to initialize the grouping factor and the correlation matrices per group. It takes an argument `data`, naming a data frame in which to evaluate the variables in `form`.

```
> cs1CompSymm <- initialize( cs1CompSymm, data = Orthodont )
> corMatrix( cs1CompSymm )
$M01:
     [,1] [,2] [,3] [,4]
[1,]  1.0  0.3  0.3  0.3
[2,]  0.3  1.0  0.3  0.3
[3,]  0.3  0.3  1.0  0.3
[4,]  0.3  0.3  0.3  1.0
. . .
```

Typically, `initialize` is only called from within the modeling function using the corStruct object.

corSymm

This class implements the general correlation structure (5.24). The argument `value` is used to initialize the correlation parameters, being given as

a numeric vector with the lower diagonal elements of the the largest correlation matrix represented by the corSymm object stacked columnwise. For example, to represent the correlation matrix

$$\begin{bmatrix} 1.0 & 0.2 & 0.1 & -0.1 \\ 0.2 & 1.0 & 0.0 & 0.2 \\ 0.1 & 0.0 & 1.0 & 0.0 \\ -0.1 & 0.2 & 0.0 & 1.0 \end{bmatrix} \tag{5.32}$$

we use

```
value = c( 0.2, 0.1, -0.1, 0, 0.2, 0 )
```

The correlations specified in `value` must define a positive-definite correlation matrix. By default, `value = numeric(0)`, which leads to initial values of 0 being assigned to all correlations in the `initialize` method.

The argument `form` specifies a one-sided formula with the position variable and, optionally, a grouping variable. The position variable defines the indices of the correlation parameters for each observation and must evaluate to an integer vector, with nonrepeated values per group, such that its unique values, when sorted, form a sequence of consecutive integers. By default, the position variable in `form` is `1`, in which case the order of the observations within the group is used to index the correlation parameters.

For example, to specify a general correlation correlation structure with initial correlation matrix as in (5.32), observation order within the group as the position variable, and grouping variable `Subject`, we use

```
> cs1Symm <- corSymm( value = c(0.2, 0.1, -0.1, 0, 0.2, 0),
+                     form = ~ 1 | Subject )
> cs1Symm <- initialize( cs1Symm, data = Orthodont )
> corMatrix( cs1Symm )
$M01:
     [,1] [,2] [,3] [,4]
[1,]  1.0  0.2  0.1 -0.1
[2,]  0.2  1.0  0.0  0.2
[3,]  0.1  0.0  1.0  0.0
[4,] -0.1  0.2  0.0  1.0
```

`corAR1`

This class implements an autoregressive correlation structure of order 1, for integer position vectors. The argument `value` initializes the single correlation parameter ϕ, which takes values between -1 and 1, and, by default, is set to 0. The argument `form` is a one-sided formula specifying the position variable and, optionally, a grouping variable. The position variable must evaluate to an integer vector, with nonrepeated values per group, but its values are not required to be consecutive, so that missing time points are naturally accommodated. By default, `form = ~1`, implying that the order of the observations within the group be used as the position variable.

For example, to specify an $AR(1)$ correlation structure with $\phi = 0.8$, position variable given by observation order within-group, and grouping variable `Subject`, we use

```
> cs1AR1 <- corAR1( 0.8, form = ~ 1 | Subject )
> cs1AR1 <- initialize( cs1AR1, data = Orthodont )
> corMatrix( cs1AR1 )
$M01:
      [,1] [,2] [,3]  [,4]
[1,] 1.000 0.80 0.64 0.512
[2,] 0.800 1.00 0.80 0.640
[3,] 0.640 0.80 1.00 0.800
[4,] 0.512 0.64 0.80 1.000
```

As described in §5.3.1, the $AR(1)$ model is equivalent to an $ARMA(1,0)$ model, so that the corARMA class can also be used to represent an corAR1 object. However, the corAR1 methods are designed to take advantage of the particular structure of the $AR(1)$ model, and are substantially more efficient than the corresponding corARMA methods.

`corCAR1`

This class implements the continuous time $AR(1)$ structure (5.27). Its arguments are defined as in `corAR1`, but the position variable can be any continuous variable with non-repeated values per group and the correlation parameter ϕ can only take positive values.

`corARMA`

This corStruct class is used to specify autoregressive–moving average models for the within-group errors. The argument `value` specifies the values of the autoregressive and moving average parameters. If both autocorrelation and moving average parameters are present, the former should precede the latter in `value`. By default, all correlation parameters are set to 0, corresponding to uncorrelated within-group errors. The `form` argument is a one-sided formula defining the position variable and, optionally, a grouping variable. The position variable must evaluate to an integer vector, with non-repeated elements per group. Its values do not need to be consecutive, so that missing time points are allowed. By default, `form = ~1`, implying that the order of the observations within the group be used as the position variable.

Two additional arguments, `p` and `q`, are used, respectively, to specify the order of the autoregressive model and the order of the moving average model. Using `p > 0` and `q = 0` specifies an $AR(p)$ model, while `p = 0` and `q > 0` specifies an $MA(q)$ model. By default, `p = 0` and `q = 0`.

For example, to specify an $MA(1)$ model with parameter $\theta = 0.4$, position variable given by the observation order within the group, and groups defined by `Subject`, we use

```
> cs1ARMA <- corARMA( 0.4, form = ~ 1 | Subject, q = 1 )
> cs1ARMA <- initialize( cs1ARMA, data = Orthodont )
> corMatrix( cs1ARMA )
$M01:
        [,1]    [,2]    [,3]    [,4]
[1,] 1.00000 0.34483 0.00000 0.00000
[2,] 0.34483 1.00000 0.34483 0.00000
[3,] 0.00000 0.34483 1.00000 0.34483
[4,] 0.00000 0.00000 0.34483 1.00000
```

An $ARMA(1,1)$ model with parameters $\phi = 0.8$ and $\theta = 0.4$ is specified with

```
> cs2ARMA <- corARMA( c(0.8, 0.4), form = ~ 1 | Subject, p=1, q=1 )
> cs2ARMA <- initialize( cs2ARMA, data = Orthodont )
> corMatrix( cs2ARMA )
$M01:
      [,1]  [,2]  [,3]   [,4]
[1,] 1.0000 0.880 0.704 0.5632
[2,] 0.8800 1.000 0.880 0.7040
[3,] 0.7040 0.880 1.000 0.8800
[4,] 0.5632 0.704 0.880 1.0000
```

Spatial corStruct Classes

The corStruct classes representing spatial correlation structures are corExp, corGaus, corLin, corRatio, and corSpher. As the corresponding constructors all have the same syntax, we will show examples for corExp only.

The argument `value` is used to specify values for the range ρ and the nugget effect c_0, in this order. The range only takes positive values and the nugget effect can only vary between 0 and 1. By default, `value = numeric(0)`, in which case the range is initialized to 90% of the minimum between-pairs distance and the nugget effect is initialized to 0.1.

The argument `form` is a one-sided formula specifying a position vector and, optionally, a grouping variable. The coordinates of the position vector must be numeric variables, but are otherwise unrestricted. By default, `form = ~1`, translating into a one-dimensional position vector given by the observation order within the group.

The argument `nugget` determines whether a nugget effect should be included in the correlation model. If `TRUE`, a nugget effect is included. Its default value is `FALSE`, corresponding to no nugget effect in the model. The argument `metric` is a character string specifying a metric to be used for calculating the between-pairs distances. Possible values include `"euclidean"`, `"maximum"`, and `"manhattan"`, corresponding to the metrics described in §5.3.2.

To illustrate the use of the spatial corStruct classes, we consider an artificial data frame `spatDat`, with columns `x` and `y`.

```
> spatDat
     x    y
1 0.00 0.00
2 0.25 0.25
3 0.50 0.50
4 0.75 0.75
5 1.00 1.00
```

An exponential spatial correlation structure based on the Euclidean distance between `x` and `y`, with range equal to 1, and no nugget effect is constructed and initialized with

```
> cs1Exp <- corExp( 1, form = ~ x + y )
> cs1Exp <- initialize( cs1Exp, spatDat )
> corMatrix( cs1Exp )
        [,1]    [,2]    [,3]    [,4]    [,5]
[1,] 1.00000 0.70219 0.49307 0.34623 0.24312
[2,] 0.70219 1.00000 0.70219 0.49307 0.34623
[3,] 0.49307 0.70219 1.00000 0.70219 0.49307
[4,] 0.34623 0.49307 0.70219 1.00000 0.70219
[5,] 0.24312 0.34623 0.49307 0.70219 1.00000
```

To calculate the distances in the Manhattan (L_1) metric, we use

```
> cs2Exp <- corExp( 1, form = ~ x + y, metric = "man" )
> cs2Exp <- initialize( cs2Exp, spatDat )
> corMatrix( cs2Exp )
        [,1]    [,2]    [,3]    [,4]    [,5]
[1,] 1.00000 0.60653 0.36788 0.22313 0.13534
[2,] 0.60653 1.00000 0.60653 0.36788 0.22313
[3,] 0.36788 0.60653 1.00000 0.60653 0.36788
[4,] 0.22313 0.36788 0.60653 1.00000 0.60653
[5,] 0.13534 0.22313 0.36788 0.60653 1.00000
```

Note that because partial matches are used on the value of the `metric` argument, we only gave the first three characters of `"manhattan"` in the call.

A nugget effect of 0.2 is added to the correlation structure using

```
> cs3Exp <- corExp( c(1, 0.2), form = ~ x + y, nugget = T )
> cs3Exp <- initialize( cs3Exp, spatDat )
> corMatrix( cs3Exp )
        [,1]    [,2]    [,3]    [,4]    [,5]
[1,] 1.00000 0.56175 0.39445 0.27698 0.19449
[2,] 0.56175 1.00000 0.56175 0.39445 0.27698
[3,] 0.39445 0.56175 1.00000 0.56175 0.39445
[4,] 0.27698 0.39445 0.56175 1.00000 0.56175
[5,] 0.19449 0.27698 0.39445 0.56175 1.00000
```

New corStruct classes, representing user-defined correlation structures, can be added to the set of standard classes in Table 5.3 and used with

the modeling functions in the nlme library. For this, one must specify a constructor function, generally with the same name as the class, and, at a minimum, methods for the functions `coef`, `corMatrix`, and `initialize`. The `corAR1` constructor and methods can serve as templates for these.

5.3.4 *Using Correlation Structures with* `lme`

Correlation structures are specified in `lme` through the `correlation` argument. By default, `correlation = NULL`, corresponding to uncorrelated within-group errors. Correlation structures are specified as corStruct objects, created using the standard constructors described in §5.3.3, or a user-defined corStruct constructor. In this section, we describe the use of correlation models in `lme` through the analysis of two examples of grouped data with correlated within-group errors.

When assessing the adequacy of a correlation model, it is often useful to consider diagnostic plots of the *normalized* residuals, defined as $\boldsymbol{r}_i = \widehat{\sigma}^{-1}(\widehat{\boldsymbol{\Lambda}}_i^{-1/2})^T(\boldsymbol{y}_i - \widehat{\boldsymbol{y}}_i)$, where $\widehat{\sigma}^2\widehat{\boldsymbol{\Lambda}}_i$ denotes the estimated variance–covariance matrix for the i within-group errors. If the within-group variance–covariance model is correct, the normalized residuals should be approximately distributed as independent $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ random vectors.

Counts of Ovarian Follicles

Pierson and Ginther (1987) report on a study of the number of ovarian follicles larger than 10 mm in diameter detected in eleven different mares at several times in their estrus cycles. The data were recorded daily from three days before ovulation until three days after the next ovulation. The measurement times for each mare are scaled so that the ovulations for each mare occur at times 0 and 1. These data are also described in Appendix A.18 and are included in the nlme library as the groupedData object `Ovary`.

The plots of the number of follicles versus time per mare, shown in Figure 5.10, suggest a periodic behavior for the number of follicles over time.

Preliminary analyses indicate that the following linear mixed-effects model provides a reasonable representation for the number of follicles y_{ij} for the ith mare at time t_{ij}.

$$\begin{aligned} y_{ij} &= (\beta_0 + b_{0i}) + (\beta_1 + b_{1i})\sin(2\pi t_{ij}) + \beta_2 \cos(2\pi t_{ij}) + \epsilon_{ij}, \\ \boldsymbol{b}_i &= \begin{bmatrix} b_{0i} \\ b_{1i} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \operatorname{diag}\left(\sigma_0^2, \sigma_1^2\right)\right), \quad \epsilon_{ij} \sim \mathcal{N}\left(0, \sigma^2\right), \end{aligned} \tag{5.33}$$

where β_0, β_1, and β_2 are the fixed effects, $\boldsymbol{b}_i$ is the random effects vector, assumed independent for different mares, and ϵ_{ij} is the within-group error, assumed independent for different i, j and independent of the random effects. The random effects b_{0i} and b_{1i} are assumed to be independent with variances σ_0^2 and σ_1^2, respectively.

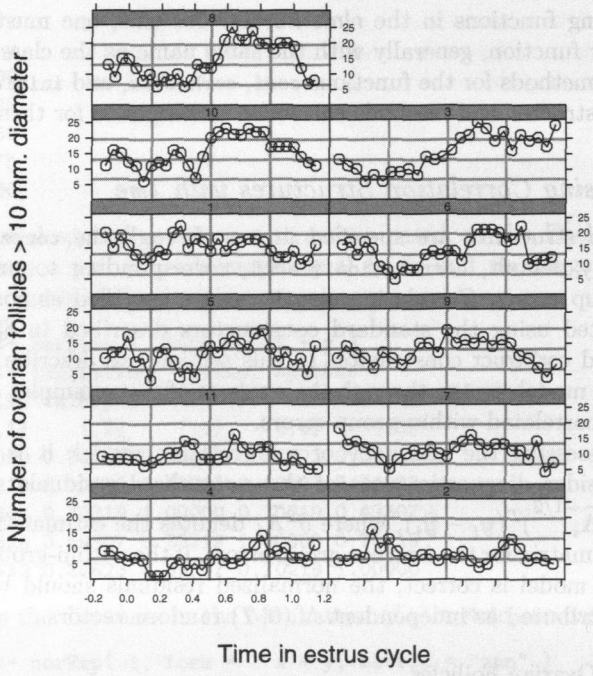

FIGURE 5.10. Number of ovarian follicles greater than 10 mm in diameter detected in mares at various times in their estrus cycles. The times have been scaled so the ovulations occur at times 0 and 1.

We fit the linear mixed-effects model (5.33) with

```
> fm1Ovar.lme <- lme( follicles ~ sin(2*pi*Time) + cos(2*pi*Time),
+                  data = Ovary, random = pdDiag(~sin(2*pi*Time)) )
> fm1Ovar.lme
Linear mixed-effects model fit by REML
  Data: Ovary
  Log-restricted-likelihood: -813.04
  Fixed: follicles ~ sin(2 * pi * Time) + cos(2 * pi * Time)
 (Intercept) sin(2 * pi * Time) cos(2 * pi * Time)
      12.182            -3.2985           -0.86237

Random effects:
 Formula:  ~ sin(2 * pi * Time) | Mare
 Structure: Diagonal
        (Intercept) sin(2 * pi * Time) Residual
StdDev:      3.0521             2.0793   3.1129

Number of Observations: 308
Number of Groups: 11
```

The observations in the Ovary data were collected at equally spaced calendar times. When the calendar time was converted to the ovulation cycle scale, the intervals between observations remained very similar, but no longer identical. Therefore, when considered in the scale of the within-group observation order, the Ovary data provides an example of *time-series* data. We use it here to illustrate the modeling of serial correlation structures in lme.

The ACF method for the lme class obtains the empirical autocorrelation function (5.22) from the residuals of an lme object.

```
> ACF( fm1Ovar.lme )
   lag       ACF
 1   0  1.000000
 2   1  0.379480
 3   2  0.179722
 4   3  0.035693
 5   4  0.059779
 6   5  0.002097
 7   6  0.064327
 8   7  0.071635
 9   8  0.048578
10   9  0.027782
11  10 -0.034276
12  11 -0.077204
13  12 -0.161132
14  13 -0.196030
15  14 -0.289337
```

Empirical autocorrelations at larger lags tend to be less reliable, because they are estimated with fewer residual pairs. We control the number of lags for which to calculate the empirical autocorrelations in ACF with the argument maxLag. A plot of the empirical autocorrelation function, displayed in Figure 5.11, is obtained with

```
> plot(ACF(fm1Ovar.lme,  maxLag = 10), alpha = 0.01)   # Figure 5.11
```

The argument alpha specifies the significance level for approximate two-sided critical bounds for the autocorrelations (Box et al., 1994), given by $\pm z(1-\alpha/2)/\sqrt{N(l)}$, with $z(p)$ denoting the standard normal quantile of order p and $N(l)$ defined as in (5.22).

The empirical autocorrelations in Figure 5.11 are significantly different from 0 at the first two lags, decrease approximately exponentially for the first four lags, and stabilize at nonsignificant levels for larger lags. This suggests that an $AR(1)$ model may be suitable for the within-group correlation and we fit it with

```
> fm2Ovar.lme <- update( fm1Ovar.lme, correlation = corAR1() )
Linear mixed-effects model fit by REML
  Data: Ovary
```

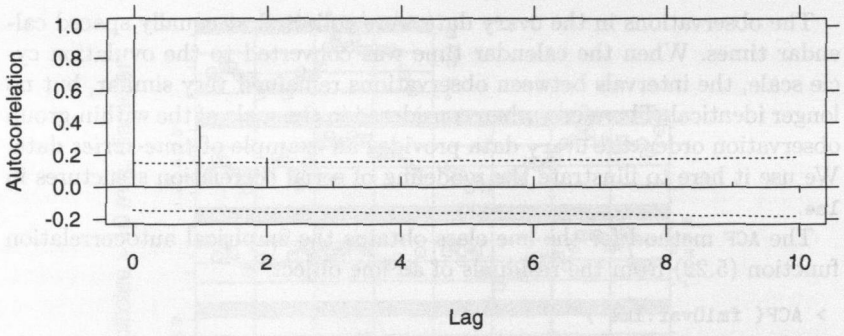

FIGURE 5.11. Empirical autocorrelation function corresponding to the standardized residuals of the `fm1Ovar.lme` object.

```
  Log-restricted-likelihood: -774.72
  Fixed: follicles ~ sin(2 * pi * Time) + cos(2 * pi * Time)
(Intercept) sin(2 * pi * Time) cos(2 * pi * Time)
     12.188            -2.9853            -0.87776

Random effects:
 Formula:  ~ sin(2 * pi * Time) | Mare
 Structure: Diagonal
        (Intercept) sin(2 * pi * Time) Residual
StdDev:      2.8585             1.258   3.5071

Correlation Structure: AR(1)
 Formula: ~ 1 | Mare
 Parameter estimate(s):
    Phi
 0.5722
Number of Observations: 308
Number of Groups: 11
```

Note that no arguments need to be passed to `corAR1` in this case, as its default formula `~1` specifies the position variable as the within-group order of the observations, which is what is desired for the `fm2Ovar.lme` model.

Because the `fm1Ovar.lme` model is nested within the `fm2Ovar.lme` model (corresponding to $\phi = 0$) we can compare them using a likelihood ratio test.

```
> anova( fm1Ovar.lme, fm2Ovar.lme )
            Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm1Ovar.lme     1  6 1638.1 1660.4 -813.04
fm2Ovar.lme     2  7 1563.4 1589.5 -774.72 1 vs 2  76.634  <.0001
```

The very significant p-value for the likelihood ratio test indicates that the $AR(1)$ provides a substantially better fit of the data than the independent errors model (5.33), suggesting that within-group serial correlation is

present in Ovary. We can assess the precision of the correlation parameter estimate in fm2Ovar.lme with the intervals method.

```
> intervals( fm2Ovar.lme )
. . .
 Correlation structure:
      lower   est.   upper
Phi 0.36607 0.5722 0.72481
. . .
```

Consistently with the likelihood ratio test results, the confidence interval on ϕ indicates that it is significantly different from 0.

The autocorrelation pattern in Figure 5.11 is also consistent with that of an $MA(2)$ model, in which only the first two lags have nonzero correlations. We fit this model with

```
> fm3Ovar.lme <- update(fm1Ovar.lme, correlation = corARMA(q = 2))
> fm3Ovar.lme
. . .
Correlation Structure: ARMA(0,2)
 Formula: ~ 1 | Mare
 Parameter estimate(s):
  Theta1  Theta2
 0.47524 0.25701
. . .
```

The $AR(1)$ and $MA(2)$ models are not nested and, therefore, cannot be compared through a likelihood ratio test. They can, however, be compared via their information criterion statistics.

```
> anova( fm2Ovar.lme, fm3Ovar.lme, test = F )
            Model df    AIC    BIC  logLik
fm2Ovar.lme     1  7 1563.4 1589.5 -774.72
fm3Ovar.lme     2  8 1571.2 1601.0 -777.62
```

Even though it has one fewer parameter than the $MA(2)$ model, the $AR(1)$ model is associated with a larger log-restricted-likelihood, which translates into smaller AIC and BIC, making it the preferred model of the two.

Because the fixed- and random-effects models in (5.33) use a continuous time scale, we investigate if a continuous time $AR(1)$ model would provide a better representation of the within-group correlation, using the corCAR1 class.

```
> fm4Ovar.lme <- update( fm1Ovar.lme,
+                        correlation = corCAR1(form = ~Time) )
> anova( fm2Ovar.lme, fm4Ovar.lme, test = F )
            Model df    AIC    BIC  logLik
fm2Ovar.lme     1  7 1563.4 1589.5 -774.72
fm4Ovar.lme     2  7 1565.5 1591.6 -775.77
```

No grouping variable needs to be specified in the call to `corCAR1`, as the innermost grouping variable in the `lme` object is used by default. As indicated by the AIC and the BIC in the `anova` output, the $AR(1)$ model provides a better representation of the within-group correlation than its continuous-time version.

An "intermediate" model between the $AR(1)$ and the $MA(2)$ models is the $ARMA(1,1)$ model, which has an exponentially decaying autocorrelation function for lags ≥ 2, but allows more flexibility in the first autocorrelation. We fit it with

```
> fm5Ovar.lme <- update(fm1Ovar.lme, corr = corARMA(p = 1, q = 1))
> fm5Ovar.lme
. . .
Correlation Structure: ARMA(1,1)
 Formula: ~ 1 | Mare
 Parameter estimate(s):
    Phi1   Theta1
 0.78716 -0.27957
. . .
```

The $AR(1)$ model is nested within the $ARMA(1,1)$ model (corresponding to $\theta_1 = 0$) and we can use `anova` to compare the two fits through a likelihood ratio test.

```
> anova( fm2Ovar.lme, fm5Ovar.lme )
            Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm2Ovar.lme     1  7 1563.4 1589.5 -774.72
fm5Ovar.lme     2  8 1559.9 1589.7 -771.95 1 vs 2  5.5537  0.0184
```

The low p-value for the likelihood ratio test indicates that the $ARMA(1,1)$ model provides a better fit of the data.

We can assess the adequacy of the $ARMA(1,1)$ model using the empirical autocorrelation function of the normalized residuals.

```
> plot( ACF(fm5Ovar.lme,  maxLag = 10, resType = "n"),
+       alpha = 0.01 )                          # Figure 5.12
```

No significant autocorrelations are observed in Figure 5.12, indicating that the normalized residuals behave like uncorrelated noise, as expected under the appropriate correlation model.

Body Weight Growth in Rats

We revisit the `BodyWeight` example of §5.2.2 to illustrate the use of corStruct classes in `lme` in combination with variance functions. As described in §5.2.2, the observations in the `BodyWeight` data are not equally spaced in time, as an extra observation is taken at 44 days. We use the spatial correlation corStruct classes to use fit continuous-time within-group correlation models, which naturally accommodate the imbalance in the data.

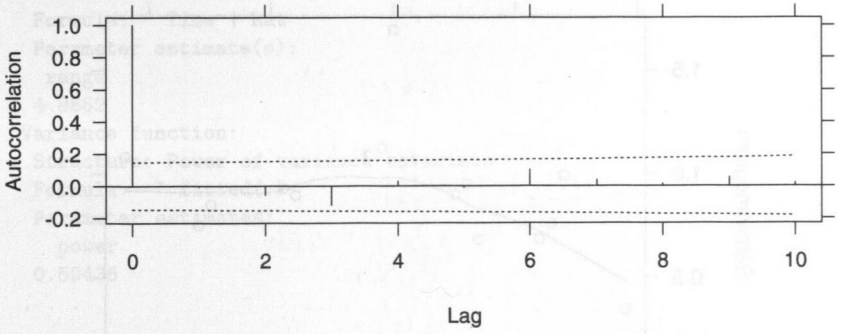

FIGURE 5.12. Empirical autocorrelation function corresponding to the normalized residuals of the `fm5Ovar.lme` object.

The `Variogram` method for the lme class estimates the sample semivariogram from the residuals of the lme object. The arguments `resType` and `robust` control, respectively, what type of residuals should be used (`"pearson"` or `"response"`) and whether the robust algorithm (5.31) or the classical algorithm (5.30) should be used to estimate the semivariogram. The defaults are `resType = "pearson"` and `robust = FALSE`, so that classical estimates of the semivariogram are obtained from the standardized residuals. The argument `form` is a one-sided formula specifying the position vector to be used for the semivariogram calculations.

```
> Variogram( fm2BW.lme, form = ~ Time )
   variog dist n.pairs
1 0.34508    1      16
2 0.99328    6      16
3 0.76201    7     144
4 0.68496    8      16
5 0.68190   13      16
6 0.95118   14     128
7 0.89959   15      16
8 1.69458   20      16
9 1.12512   21     112
10 1.08820  22      16
11 0.89693  28      96
12 0.93230  29      16
13 0.85144  35      80
14 0.75448  36      16
15 1.08220  42      64
16 1.56652  43      16
17 0.64378  49      48
18 0.67350  56      32
19 0.58663  63      16
```

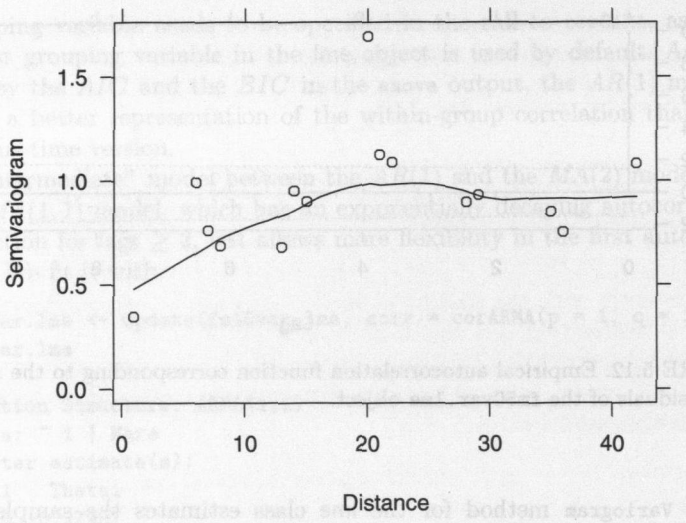

FIGURE 5.13. Sample semivariogram estimates corresponding to the standardized residuals of the `fm2BW.lme` object. A loess smoother is added to the plot to enhance the visualization of patterns in the semivariogram.

The columns in the data frame returned by `Variogram` represent, respectively, the sample semivariogram, the distance, and the number of residual pairs used in the estimation. Because of the imbalance in the time measurements, the number of residual pairs used at each distance varies considerably, making some semivariogram estimates more reliable than others. In general, the number of residual pairs used in the semivariogram estimation decreases with distance, making the values at large distances unreliable. We can control the maximum distance for which semivariogram estimates should be calculated using the argument `maxDist`.

A graphical representation of the sample semivariogram is obtained with the `plot` method for class Variogram.

```
> plot( Variogram(fm2BW.lme, form = ~ Time,
+                 maxDist = 42) )                    # Figure 5.13
```

The resulting plot, shown in Figure 5.13, includes a *loess* smoother (Cleveland et al., 1992) to enhance the visualization of semivariogram patterns. The semivariogram seems to increase with distance up to 20 days and then stabilizes around 1. We initially use an exponential spatial correlation model for the within-group errors, fitting it with

```
> fm3BW.lme <- update( fm2BW.lme, corr = corExp(form = ~ Time) )
> fm3BW.lme
. . .
Correlation Structure: Exponential spatial correlation
```

```
 Formula: ~ Time | Rat
 Parameter estimate(s):
  range
 4.8862
Variance function:
 Structure: Power of variance covariate
 Formula:  ~ fitted(.)
 Parameter estimates:
   power
 0.59436
. . .
```

Note that the model corresponding to `fm3BW.lme` includes both a variance function and a correlation structure. We assess the variability in the spatial correlation parameter estimate with the `intervals` method.

```
> intervals( fm3BW.lme )
. . .
Correlation structure:
      lower   est.  upper
range 1.852 4.8862 12.891
. . .
```

The confidence intervals is bounded away from zero, suggesting that the spatial correlation model produced a significantly better fit. We can also test this using the `anova` method.

```
> anova( fm2BW.lme, fm3BW.lme )
          Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm2BW.lme     1 11 1163.9 1198.4 -570.96
fm3BW.lme     2 12 1145.1 1182.8 -560.57 1 vs 2  20.781  <.0001
```

The likelihood ratio test also indicates that the corExp model fits the data significantly better than the independent errors model corresponding to `fm2BW.lme`.

The semivariogram plot in Figure 5.13 gives some indication that a nugget effect may be present in the data. We can test it with

```
> fm4BW.lme <- update( fm3BW.lme,
+                     corr = corExp(form =  ~ Time, nugget = T) )
> anova( fm3BW.lme, fm4BW.lme )
          Model df    AIC    BIC  logLik   Test    L.Ratio
fm3BW.lme     1 12 1145.1 1182.8 -560.57
fm4BW.lme     2 13 1147.1 1187.9 -560.57 1 vs 2 0.00043111
          p-value
fm3BW.lme
fm4BW.lme  0.9834
```

The nearly identical log-likelihood values indicate that a nugget effect is, in fact, not needed.

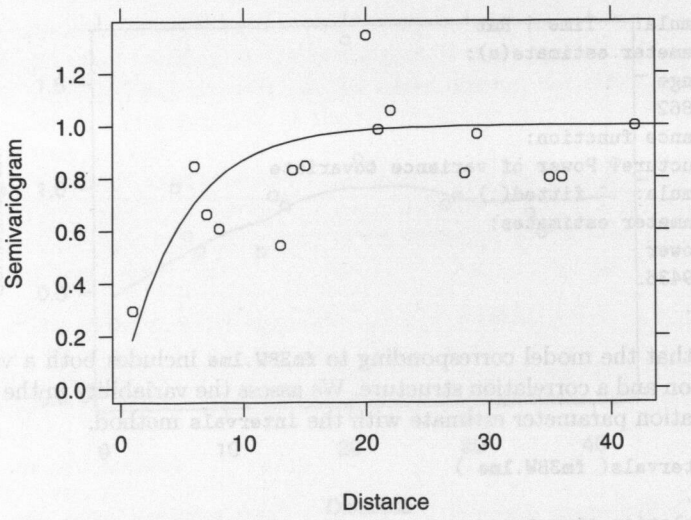

FIGURE 5.14. Sample semivariogram estimates corresponding to the standardized residuals of the `fm2BW.lme` object. The fitted semivariogram corresponding to `fm3BW.lme` is added to plot.

When an lme object includes a spatial corStruct object, we can further assess the adequacy of the correlation model with the `plot` method. In this case, instead of a loess smoother, the fitted semivariogram corresponding to the corStruct object is displayed in the plot, along with the sample variogram estimates.

```
> plot( Variogram(fm3BW.lme, form = ~ Time,
+                 maxDist = 42) )                    # Figure 5.14
```

The fitted semivariogram agrees reasonably well with the sample variogram estimates.

We can also assess the adequacy of the exponential spatial correlation model by investigating the sample semivariogram for the normalized residuals.

```
> plot( Variogram(fm3BW.lme, form = ~ Time, maxDist = 42,
+                 resType = "n", robust = T) )       # Figure 5.15
```

The robust semivariogram estimator is used to reduce the influence of an outlying value at distance 1 on the loess smoother. The sample semivariogram estimates in Figure 5.15 appear to vary randomly around the $y = 1$ line, suggesting that the normalized residuals are approximately uncorrelated and, hence, the corExp model is adequate.

We may compare the corExp model to other spatial correlation models, using the `update` method and the `anova` method. As the models are not

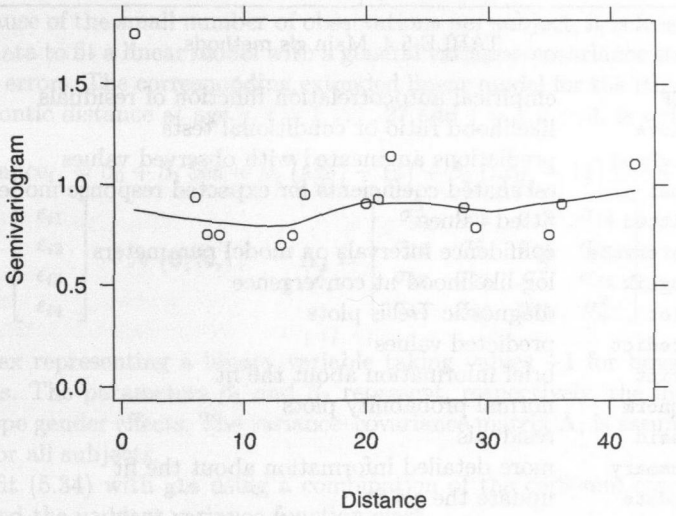

FIGURE 5.15. Sample semivariogram estimates corresponding to the normalized residuals of the `fm3BW.lme` object. A loess smoother is added to the plot to enhance the visualization of patterns in the semivariogram.

nested, they have to be compared based on the information criteria *AIC* and *BIC*.

```
> fm5BW.lme <- update( fm3BW.lme, corr = corRatio(form = ~ Time) )
> fm6BW.lme <- update( fm3BW.lme, corr = corSpher(form = ~ Time) )
> fm7BW.lme <- update( fm3BW.lme, corr = corLin(form = ~ Time) )
> fm8BW.lme <- update( fm3BW.lme, corr = corGaus(form = ~ Time) )
> anova( fm3BW.lme, fm5BW.lme, fm6BW.lme, fm7BW.lme, fm8BW.lme )
           Model df    AIC    BIC  logLik
fm3BW.lme      1 12 1145.1 1182.8 -560.57
fm5BW.lme      2 12 1148.8 1186.4 -562.38
fm6BW.lme      3 12 1150.8 1188.4 -563.39
fm7BW.lme      4 12 1150.8 1188.4 -563.39
fm8BW.lme      5 12 1150.8 1188.4 -563.39
```

The corExp fit has the smallest *AIC* and *BIC* and seems the most adequate within-group correlation model for the `BodyWeight` data, among the spatial correlation models considered.

5.4 Fitting Extended Linear Models with `gls`

The general formulation of the extended linear model, as well as the estimation methods used to fit it, have been described in §5.1.2. In this section,

TABLE 5.4. Main gls methods.

ACF	empirical autocorrelation function of residuals
anova	likelihood ratio or conditional tests
augPred	predictions augmented with observed values
coef	estimated coefficients for expected response model
fitted	fitted values
intervals	confidence intervals on model parameters
logLik	log-likelihood at convergence
plot	diagnostic Trellis plots
predict	predicted values
print	brief information about the fit
qqnorm	normal probability plots
resid	residuals
summary	more detailed information about the fit
update	update the gls fit
Variogram	semivariogram of residuals

we concentrate on the capabilities available in the nlme library for fitting such models.

The gls function is used to fit the extended linear model (5.5), using either maximum likelihood, or restricted maximum likelihood. It can be viewed as an lme function without the argument random. Several arguments are available in gls, but typical calls are of the form

```
gls( model, data, correlation )    # correlated errors
gls( model, data, weights )        # heteroscedastic errors
gls( model, data, correlation, weights )  # both
```

The first argument, model, is a two-sided linear formula specifying the model for the expected value of the response. Correlation and weights are used, as in lme, to define, respectively, the correlation model and the variance function model for the error term. Data specifies a data frame in which the variables named in model, correlation, and weights can be evaluated.

The fitted object returned by gls inherits from class gls, for which several methods are available to display, plot, update, and further explore the estimation results. Table 5.4 lists the most important methods for class gls. The use of the gls function and its associated methods is described and illustrated through the examples in the next sections.

Orthodontic Growth Curve

The Orthodont data were analyzed in §4.2 and §4.3 using a linear mixed-effects model. We describe here an alternative analysis based on the extended linear model (5.5).

Because of the small number of observations per subject, it is feasible for these data to fit a linear model with a general variance–covariance structure for the errors. The corresponding extended linear model for the ith subject orthodontic distance at age j, $i = 1, \ldots, 27$ and $j = 1, \ldots, 4$, is written as

$$\text{distance}_{ij} = \beta_0 + \beta_1 \,\text{Sex} + \beta_2 \left(\text{age}_j - 11\right) + \beta_3 \left(\text{age}_j - 11\right) \text{Sex} + \epsilon_{ij},$$

$$\boldsymbol{\epsilon}_i = \begin{bmatrix} \epsilon_{i1} \\ \epsilon_{i2} \\ \epsilon_{i3} \\ \epsilon_{i4} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Lambda}_i\right), \qquad \boldsymbol{\Lambda}_i = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} & \sigma_{14} \\ \sigma_{12} & \sigma_2^2 & \sigma_{23} & \sigma_{24} \\ \sigma_{13} & \sigma_{23} & \sigma_3^2 & \sigma_{34} \\ \sigma_{14} & \sigma_{24} & \sigma_{34} & \sigma_4^2 \end{bmatrix}, \tag{5.34}$$

with Sex representing a binary variable taking values -1 for boys and 1 for girls. The parameters β_1 and β_3 represent, respectively, the intercept and slope gender effects. The variance–covariance matrix $\boldsymbol{\Lambda}_i$ is assumed the same for all subjects.

We fit (5.34) with `gls` using a combination of the corSymm correlation class and the varIdent variance function class.

```
> fm1Orth.gls <- gls( distance ~ Sex * I(age - 11), Orthodont,
+                    correlation = corSymm(form = ~ 1 | Subject),
+                    weights = varIdent(form = ~ 1 | age) )
```

In this case, because `Orthodont` is a groupedData object with grouping variable `Subject`, the argument `form` could be omitted in the call to `corSymm`.

The `print` method gives some basic information about the fit.

```
> fm1Orth.gls
Generalized least squares fit by REML
  Model: distance ~ Sex * I(age - 11)
  Data: Orthodont
  Log-restricted-likelihood: -213.66

Coefficients:
 (Intercept)     Sex I(age - 11) Sex:I(age - 11)
      23.801 -1.136      0.6516         -0.17524

Correlation Structure: General
 Formula:  ~ 1 | age

 Parameter estimate(s):
 Correlation:
     1     2     3
2 0.568
3 0.659 0.581
4 0.522 0.725 0.740
Variance function:
 Structure: Different standard deviations per stratum
 Formula: ~ 1 | age
```

```
 Parameter estimates:
  8     10      12      14
 1 0.8792 1.0747 0.95872
Degrees of freedom: 108 total; 104 residual
Residual standard error: 2.329
```

The correlation estimates are similar, suggesting that a compound symmetry structure may be a suitable correlation model. We explore this further with the `intervals` method.

```
> intervals( fm1Orth.gls )
Approximate 95% confidence intervals

 Coefficients:
                    lower      est.     upper
    (Intercept)  23.06672  23.80139  24.536055
            Sex  -1.87071  -1.13605  -0.401380
    I(age - 11)   0.52392   0.65160   0.779289
Sex:I(age - 11)  -0.30292  -0.17524  -0.047552

 Correlation structure:
         lower    est.   upper
cor(1,2) 0.098855 0.56841 0.83094
cor(1,3) 0.242122 0.65878 0.87030
cor(1,4) 0.021146 0.52222 0.81361
cor(2,3) 0.114219 0.58063 0.83731
cor(2,4) 0.343127 0.72510 0.90128
cor(3,4) 0.382248 0.73967 0.90457

 Variance function:
    lower    est.  upper
10 0.55728 0.87920 1.3871
12 0.71758 1.07468 1.6095
14 0.61253 0.95872 1.5005

 Residual standard error:
  lower  est.  upper
 1.5985 2.329 3.3933
```

All confidence intervals for the correlation parameters overlap, corroborating the compound symmetry assumption. We can test it formally by updating the fitted object and using the `anova` method.

```
> fm2Orth.gls <-
+    update(fm1Orth.gls, corr = corCompSymm(form = ~ 1 | Subject))
> anova( fm1Orth.gls, fm2Orth.gls )
            Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm1Orth.gls     1 14 455.32 492.34 -213.66
fm2Orth.gls     2  9 452.74 476.54 -217.37 1 vs 2  7.4256  0.1909
```

The large p-value for the likelihood ratio statistics reported in the anova output confirms the compound symmetry model.

The confidence intervals for the variance function parameters corresponding to fm2Orth.gls,

```
> intervals( fm2Orth.gls )
. . .
 Variance function:
     lower    est.  upper
10 0.56377 0.86241 1.3193
12 0.68132 1.03402 1.5693
14 0.60395 0.92045 1.4028
. . .
```

all include the value 1, suggesting that the variability does not change with age. The high p-value for the associated likelihood ratio test confirms this assumption.

```
> fm3Orth.gls <- update( fm2Orth.gls, weights = NULL )
> anova( fm2Orth.gls, fm3Orth.gls )
             Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm2Orth.gls      1  9 452.74 476.54 -217.37
fm3Orth.gls      2  6 448.53 464.40 -218.26 1 vs 2  1.7849  0.6182
```

As with other modeling functions, the plot method is used to assess the assumptions in the model. Its syntax is identical to the plot method for class lme. For example, to examine if the error variance is the same for boys and girls we may examine the plot of the normalized residuals versus age by gender, obtained with

```
> plot( fm3Orth.gls, resid(., type = "n") ~ age | Sex )  # Fig. 5.16
```

and displayed in Figure 5.16. It is clear that there is more variability among boys than girls, which we can represent in the model with the varIdent variance function class.

```
> fm4Orth.gls <- update( fm3Orth.gls,
+                        weights = varIdent(form = ~ 1 | Sex) )
> anova( fm3Orth.gls, fm4Orth.gls )
             Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm3Orth.gls      1  6 448.53 464.40 -218.26
fm4Orth.gls      2  7 438.96 457.47 -212.48 1 vs 2  11.569   7e-04
```

As expected, the likelihood ratio test gives strong evidence in favor of the heteroscedastic model.

The qqnorm method is used to assess the assumption of normality for the errors. Its syntax is identical to the corresponding lme method. The normal plots of the normalized residuals by gender, obtained with

```
> qqnorm( fm4Orth.gls, ~resid(., type = "n") )          # Figure 5.17
```

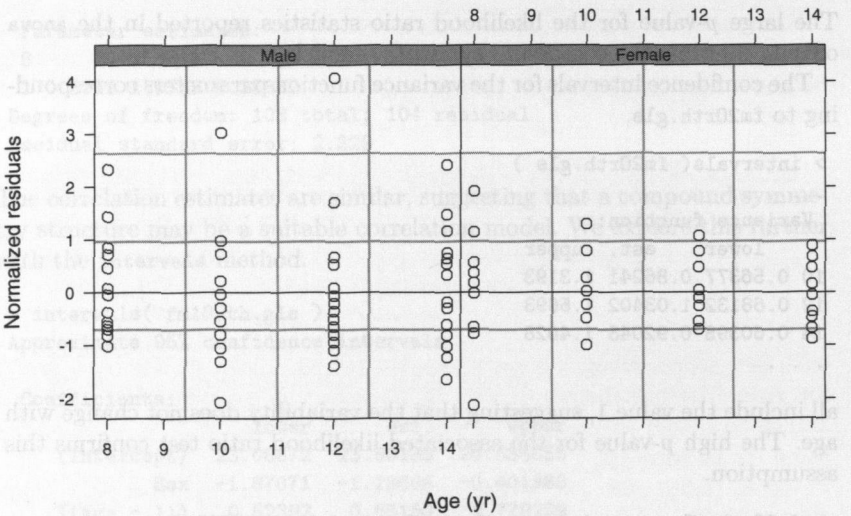

FIGURE 5.16. Scatter plots of normalized residuals versus age by gender for the `fm3Orth.gls` fitted object.

and displayed in Figure 5.17, do not indicate serious departures from normality and confirm that the variance function model included in `fm4Orth.gls` was successful in accommodating the error heteroscedasticity.

It is interesting, at this point, to compare the gls fit corresponding to `fm4Orth.gls` to the lme fit corresponding to `fm3Orth.lme`, obtained in §4.3.1. Because the corresponding models are not nested, a likelihood ratio test is nonsensical. However, the information criteria can be compared, as the fixed effects models are identical for the two fits. The `anova` method can be used to compare gls and lme objects.

```
> anova( fm3Orth.lme, fm4Orth.gls, test = F )
             Model df    AIC    BIC  logLik
fm3Orth.lme      1  9 432.30 456.09 -207.15
fm4Orth.gls      2  7 438.96 457.47 -212.48
```

The lme fit has the smallest AIC and BIC and, therefore, seems to give a better representation of the `Orthodont` data.

The choice between an lme model and a gls model should take into account more than just information criteria and likelihood tests. A mixed-effects model has a hierarchical structure which, in many applications, provides a more intuitive way of accounting for within-group dependency than the direct modeling of the marginal variance–covariance structure of the response in the gls approach. Furthermore, the mixed-effects estimation gives, as a byproduct, estimates for the random effects, which may be of interest in themselves. The gls model focuses on marginal inference and is more appealing when a hierarchical structure for the data is not believed to be

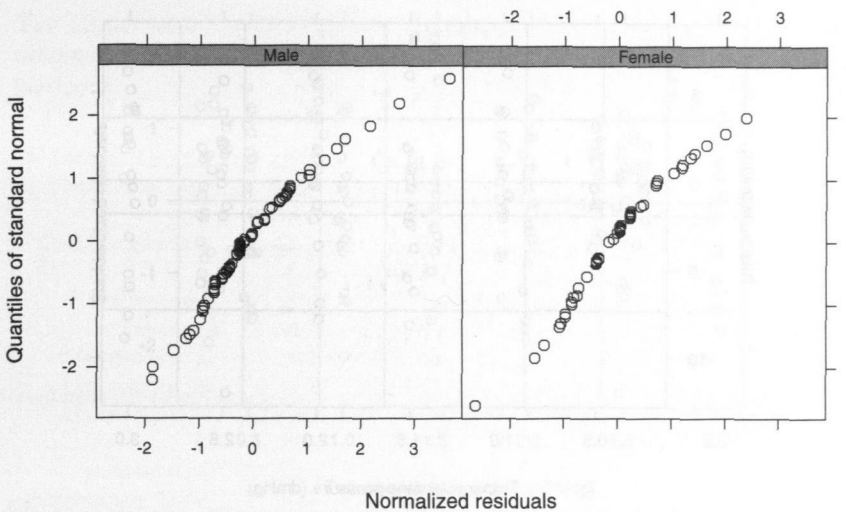

FIGURE 5.17. Normal plots of normalized residuals by gender for the `fm4Orth.gls` fitted object.

present, or is not relevant in the analysis, and one is more interested in parameters associated with the error variance–covariance structure, as in time-series analysis and spatial statistics.

High-Flux Hemodialyzer Ultrafiltration Rates

The hemodialyzer data were analyzed in §5.2.2 to illustrate the use of variance functions with linear mixed-effects models. An alternative analysis, using the `gls` function, is presented here.

We initially consider a linear model with the same expected response as the linear mixed-effects model (5.18) and independent, homoscedastic errors.

$$\begin{aligned} y_{ij} = &(\beta_0 + \gamma_0 Q_i) + (\beta_1 + \gamma_1 Q_i)\, x_{ij} + (\beta_2 + \gamma_2 Q_i +)\, x_{ij}^2 + \\ &(\beta_3 + \gamma_3 Q_i)\, x_{ij}^3 + (\beta_4 + \gamma_4 Q_i)\, x_{ij}^4 + \epsilon_{ij}, \quad \epsilon_{ij} \sim \mathcal{N}\left(0, \sigma^2\right), \end{aligned} \tag{5.35}$$

where Q_i is a binary variable taking values -1 for 200 dl/min hemodialyzers and 1 for 300 dl/min hemodialyzers; β_0, β_1, β_2, β_3, and β_4 are, respectively, the intercept, linear, quadratic, cubic, and quartic coefficients averaged over the levels of Q; γ_i is the blood flow effect associated with the coefficient β_i; and ϵ_{ij} is the error term.

We may fit the linear model (5.35) using the `gls` function

```
> fm1Dial.gls <-
+   gls(rate ~(pressure + pressure^2 + pressure^3 + pressure^4)*QB,
+       Dialyzer)
```

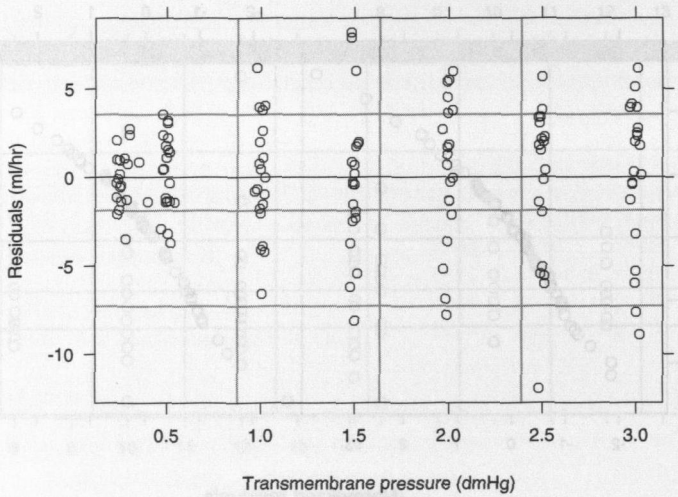

FIGURE 5.18. Plot of residuals versus transmembrane pressure for the homoscedastic fitted object `fm1Dial.gls`.

Because no variance functions and correlation structures are used, the gls fit is equivalent to an lm fit, in this case.

The plot of the residuals versus transmembrane pressure, obtained with

```
> plot( fm1Dial.gls, resid(.) ~ pressure,
+       abline = 0 )                                  # Figure 5.18
```

and shown in Figure 5.18, displays the same pattern of heteroscedasticity observed for the within-group residuals of the lme object `fm1Dial.lme`, presented in Figure 5.2.

As in the `lme` analysis of §5.2.2, we choose the flexible variance function class varPower to model the heteroscedasticity in the response, and test its significance using the `anova` method.

```
> fm2Dial.gls <- update( fm1Dial.gls,
+                        weights = varPower(form = ~ pressure) )
> anova( fm1Dial.gls, fm2Dial.gls)
            Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm1Dial.gls     1 11 768.10 799.65 -373.05
fm2Dial.gls     2 12 738.22 772.63 -357.11 1 vs 2  31.882  <.0001
```

As expected, the likelihood ratio test strongly rejects the assumption of homoscedasticity. The plot of the standard residuals corresponding to `fm2Dial.gls` versus pressure, shown in Figure 5.19, indicates that the power variance function adequately represents the heteroscedasticity in the data.

Because the hemodialyzer ultrafiltration rates are measured sequentially over time on the same subjects, the within-subject observations are likely

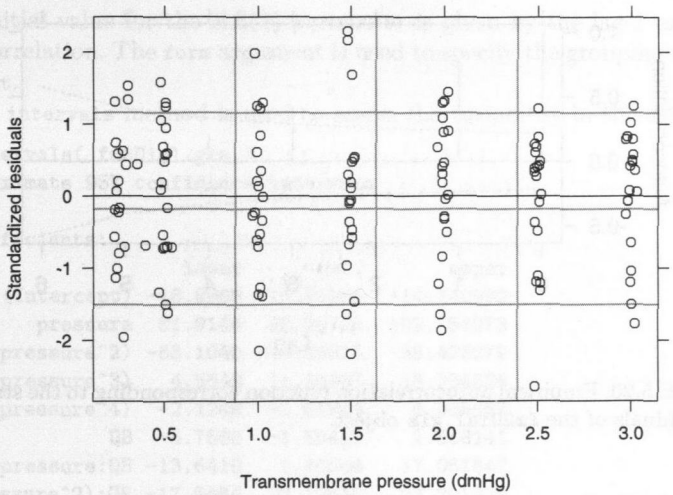

FIGURE 5.19. Plot of standardized residuals versus transmembrane pressure for the heteroscedastic fitted object `fm2Dial.gls`.

to be correlated. In §5.2.2, random effects were used to account for the within-group correlation. We may, alternatively, use a correlation structure to directly model the association among the within-subject errors. Because the measurements in this example are equally spaced in time, the empirical autocorrelation function can be used to investigate within-subject correlation. The `ACF` method for the `gls` class has a syntax similar to the corresponding `lme` method, but includes a `form` argument which allows the specification of a time covariate, to define the lags between observations, and a grouping variable, to specify a partition for the residuals.

```
> ACF( fm2Dial.gls, form = ~ 1 | Subject )
  lag       ACF
1   0 1.000000
2   1 0.770851
3   2 0.632301
4   3 0.408306
5   4 0.200737
6   5 0.073116
7   6 0.077802
```

The empirical ACF values indicate that the within-group observations are correlated, and that the correlation decreases with lag. As usual, it is more informative to look at a plot of the empirical ACF, displayed in Figure 5.20 and obtained with

```
> plot( ACF( fm2Dial.gls, form = ~ 1 | Subject),
+       alpha = 0.01 )                          # Figure 5.20
```

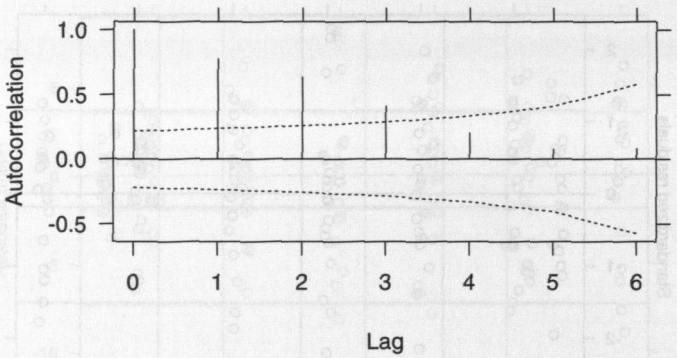

FIGURE 5.20. Empirical autocorrelation function corresponding to the standardized residuals of the `fm2Dial.gls` object.

The ACF pattern observed in Figure 5.20 suggests that an $AR(1)$ model may be appropriate to describe it. The corAR1 class is used to represent it.

```
> fm3Dial.gls <- update( fm2Dial.gls,
+                       corr = corAR1(0.771, form = ~ 1 | Subject) )
> fm3Dial.gls
Generalized least squares fit by REML
  Model: rate ~(pressure + pressure^2 + pressure^3 + pressure^4)*QB
  Data: Dialyzer
  Log-restricted-likelihood: -308.34

Coefficients:
 (Intercept) pressure I(pressure^2) I(pressure^3) I(pressure^4)
     -16.818   92.334       -49.265        11.4       -1.0196
      QB pressure:QB I(pressure^2):QB I(pressure^3):QB
 -1.5942      1.7054           2.1268         0.47972
 I(pressure^4):QB
         -0.22064

Correlation Structure: AR(1)
 Formula: ~ 1 | Subject
 Parameter estimate(s):
     Phi
 0.75261
Variance function:
 Structure: Power of variance covariate
 Formula: ~ pressure
 Parameter estimates:
   power
 0.51824
Degrees of freedom: 140 total; 130 residual
Residual standard error: 3.0463
```

The initial value for the $AR(1)$ parameter is given by the lag-1 empirical autocorrelation. The `form` argument is used to specify the grouping variable `Subject`.

The `intervals` method is used to assess the variability in the estimates.

```
> intervals( fm3Dial.gls )
Approximate 95% confidence intervals

 Coefficients:
                      lower      est.      upper
     (Intercept) -18.8968 -16.81845  -14.740092
        pressure  81.9144  92.33423  102.754073
   I(pressure^2) -63.1040 -49.26515  -35.426279
   I(pressure^3)   4.5648  11.39967   18.234526
   I(pressure^4)  -2.1248  -1.01964    0.085557
              QB  -4.7565  -1.59419    1.568141
     pressure:QB -13.6410   1.70544   17.051847
I(pressure^2):QB -17.9484   2.12678   22.201939
I(pressure^3):QB  -9.3503   0.47972   10.309700
I(pressure^4):QB  -1.8021  -0.22064    1.360829

 Correlation structure:
      lower    est.   upper
Phi 0.56443 0.75261 0.86643

 Variance function:
       lower    est.   upper
power 0.32359 0.51824 0.71289

 Residual standard error:
  lower   est.  upper
 2.3051 3.0463 4.0259
```

The confidence interval on the correlation parameter ϕ is bounded away from zero, indicating that the $AR(1)$ model provides a significantly better fit. We can confirm it with the likelihood ratio test.

```
> anova( fm2Dial.gls, fm3Dial.gls )
             Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm2Dial.gls      1 12 738.22 772.63 -357.11
fm3Dial.gls      2 13 642.67 679.95 -308.34 1 vs 2  97.546  <.0001
```

No significant correlations are observed in the plot of the empirical autocorrelation function for the normalized residuals of `fm3Dial.gls`, displayed in Figure 5.21, indicating that the $AR(1)$ adequately represents the within-subject dependence.

The gls model corresponding to `fm3Dial.gls` may be compared to the best lme model for the `Dialyzer` data in §5.2.2, corresponding to the `fm2Dial.lme` object. As the models are not nested, only the information criterion statistics can be compared.

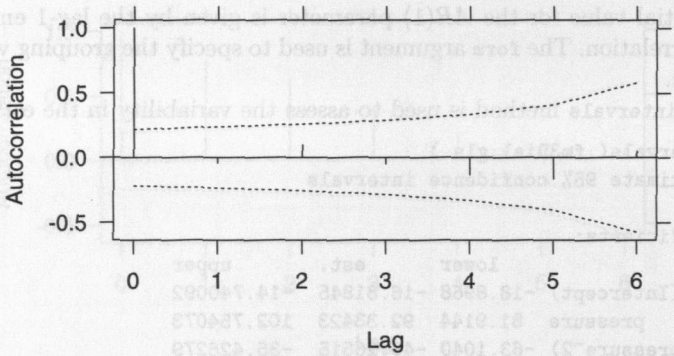

FIGURE 5.21. Empirical autocorrelation function for the normalized residuals of the `fm3Dial.gls` object.

```
> anova( fm3Dial.gls, fm2Dial.lme, test = F )
              Model df    AIC    BIC  logLik
fm3Dial.gls       1 13 642.67 679.95 -308.34
fm2Dial.lme       2 18 655.01 706.63 -309.51
```

The two log-likelihoods are very similar, suggesting that the models give equivalent representations of the data. Because the gls model has five fewer parameters than the lme model, its information criterion statistics take smaller values, suggesting it is a better model. However, as pointed out in the previous example, the choice between a gls and an lme should take other factors in consideration, besides the information criteria.

Wheat Yield Trials

Stroup and Baenziger (1994) describe an agronomic experiment to compare the yield of 56 different varieties of wheat planted in four blocks arranged according to a randomized complete complete block design. All 56 varieties of wheat were used in each block. The latitude and longitude of each experimental unit in the trial were also recorded. These data are described in greater detail in Appendix A.31, being included in the nlme library as the groupedData object `Wheat2`.

The plot of the wheat yields for each variety by block, shown in Figure 5.22, suggests that a *block effect* is present in the data. As pointed out by Littell et al. (1996, §9.6.2), the large number of plots within each block makes the assumption of within-block homogeneity unrealistic. A better representation of the dependence among the experimental units may be obtained via spatial correlation structures that use the information on their latitude and longitude. The corresponding extended linear model for the ith wheat variety yield in the jth block, y_{ij}, for $i = 1, \ldots, 56, \quad j = 1, \ldots, 4,$

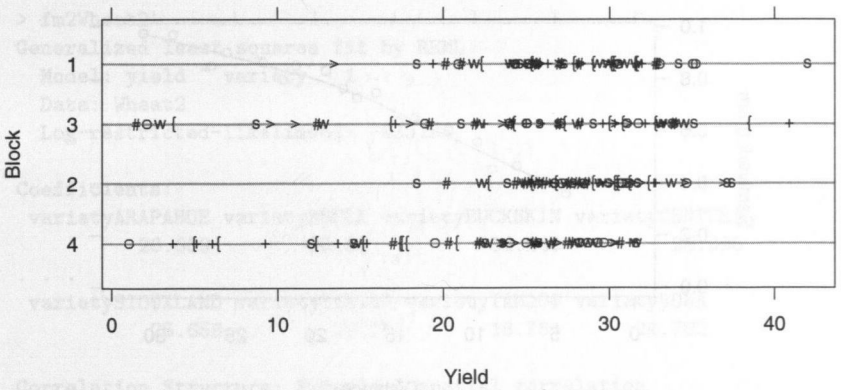

FIGURE 5.22. Yields of 56 different varieties of wheat for each block of a randomized complete block design.

is given by

$$y_{ij} = \tau_i + \epsilon_{ij}, \quad \boldsymbol{\epsilon} = \mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{\Lambda}\right), \tag{5.36}$$

where τ_i denotes the average yield for variety i and ϵ_{ij} denotes the error term, assumed to be normally distributed with mean 0 and with variance–covariance matrix $\sigma^2 \boldsymbol{\Lambda}$.

To explore the structure of $\boldsymbol{\Lambda}$, we initially fit the linear model (5.36) with the errors assumed independent and homoscedastic, i.e., $\boldsymbol{\Lambda} = \boldsymbol{I}$.

```
> fm1Wheat2 <- gls( yield ~ variety - 1, Wheat2 )
```

As described in §5.3.2, the sample semivariogram of the standardized residuals is the primary tool for investigating spatial correlation in the errors. The `variogram` method for class `gls` is used to obtain the sample semivariogram for the residuals of a `gls` object. Its syntax is identical to that of the `Variogram` method for `lme` objects.

```
> Variogram( fm1Wheat2, form = ~ latitude + longitude )
   variog    dist n.pairs
1 0.36308  4.3000    1212
2 0.40696  5.6080    1273
3 0.45366  8.3863    1256
4 0.51639  9.3231    1245
5 0.57271 10.5190    1254
6 0.58427 12.7472    1285
7 0.63854 13.3929    1175
8 0.65123 14.7635    1288
9 0.73590 16.1818    1290
10 0.73797 17.3666   1187
11 0.75081 18.4567   1298
```

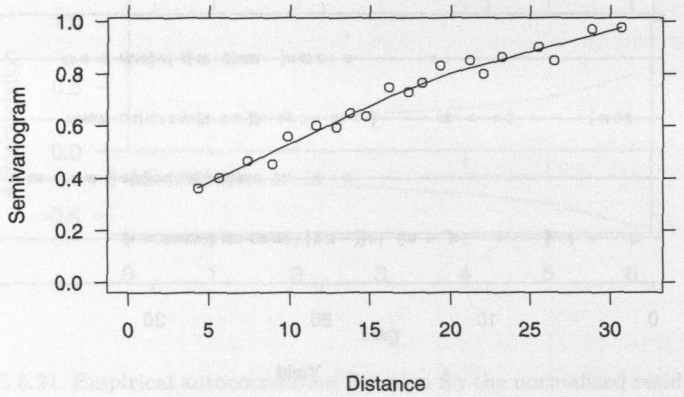

FIGURE 5.23. Sample semivariogram estimates corresponding to the standardized residuals of the `fm1Wheat2` object. A loess smoother is added to the plot to enhance the visualization of patterns in the semivariogram.

```
12 0.88098 20.2428    1226
13 0.81019 21.6335    1281
14 0.86199 22.6736    1181
15 0.86987 24.6221    1272
16 0.85818 26.2427    1223
17 0.97145 28.4542    1263
18 0.98778 30.7877    1228
19 1.09617 34.5879    1263
20 1.34146 39.3641    1234
```

The sample semivariogram increases with distance, indicating, as expected, that the observations are spatially correlated. The graphical representation of the sample semivariogram produced by the `plot` method, displayed in Figure 5.23, allows easier interpretation of the spatial correlation pattern.

```
> plot( Variogram(fm1Wheat2, form = ~ latitude + longitude,
+         maxDist = 32), xlim = c(0,32) )                    # Figure 5.23
```

The distance is "censored" at 32, using the `maxDist` argument, to avoid the less reliable estimates associated with distant plots. A nugget effect of about 0.2 seems to be present and the semivariogram appears to approach 1 around distance 28.

Littell et al. (1996, §9.6.2) use the spherical correlation structure to model the spatial correlation in these data. We can fit it using the corSpher class. Initial values for the range and the nugget effect are obtained from Figure 5.23, noting that, in `corSpher`, the range is the distance at which the semivariogram first equals 1.

```
> fm2Wheat2 <- update( fm1Wheat2, corr = corSpher(c(28, 0.2),
+                     form = ~ latitude + longitude, nugget = T) )
```

```
> fm2Wheat2
Generalized least squares fit by REML
  Model: yield ~ variety - 1
  Data: Wheat2
  Log-restricted-likelihood: -533.93

Coefficients:
 varietyARAPAHOE varietyBRULE varietyBUCKSKIN varietyCENTURA
          26.659        25.85          34.848         25.095
. . .
 varietySIOUXLAND varietyTAM107 varietyTAM200 varietyVONA
           25.656        22.77        18.764      24.782

Correlation Structure: Spherical spatial correlation
 Formula: ~ latitude + longitude

 Parameter estimate(s):
  range  nugget
 27.457 0.20931
Degrees of freedom: 224 total; 168 residual
Residual standard error: 7.4106
```

An alternative model suggested by the slow increase of the semivariogram with distance in Figure 5.23 is the rational quadratic model of §5.3.2. An initial value for the range in this model can also be obtained from the semivariogram plot, noting that, when distance = range the semivariogram in the rational quadratic model is equal to $(1 + \text{nugget})/2$. For a nugget effect of 0.2, this gives an initial estimate for the range of about 12.5 (the approximate distance in Figure 5.23 at which the semivariogram is 0.6). The corRatio class is used to fit the rational quadratic model.

```
> fm3Wheat2 <- update( fm1Wheat2,
+                corr = corRatio(c(12.5, 0.2),
+                  form = ~ latitude + longitude, nugget = T) )
> fm3Wheat2
Generalized least squares fit by REML
  Model: yield ~ variety - 1
  Data: Wheat2
  Log-restricted-likelihood: -532.64

Coefficients:
 varietyARAPAHOE varietyBRULE varietyBUCKSKIN varietyCENTURA
          26.546        26.284          35.037         24.867
. . .
 varietySIOUXLAND varietyTAM107 varietyTAM200 varietyVONA
            25.74        22.476        18.693      25.046

Correlation Structure: Rational quadratic spatial correlation
 Formula: ~ latitude + longitude
```

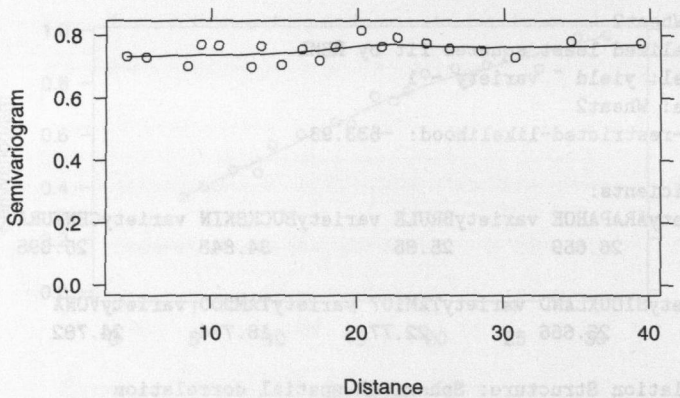

FIGURE 5.24. Sample semivariogram estimates corresponding to the normalized residuals of the `fm3Wheat2` object. A loess smoother is added to the plot to enhance the visualization of patterns in the semivariogram.

```
 Parameter estimate(s):
  range nugget
 13.461 0.1936
Degrees of freedom: 224 total; 168 residual
Residual standard error: 8.8463
> anova( fm2Wheat2, fm3Wheat2 )
           Model df    AIC    BIC  logLik
fm2Wheat2      1 59 1185.9 1370.2 -533.93
fm3Wheat2      2 59 1183.3 1367.6 -532.64
```

The smaller AIC and BIC values for the rational quadratic model indicate that it gives a better representation of the correlation in the data than the spherical model. We can test the significance of the spatial correlation parameters comparing the `fm3Wheat2` fit to the fit with independent errors corresponding to `fm1Wheat2`.

```
> anova( fm1Wheat2, fm3Wheat2 )
           Model df    AIC    BIC  logLik   Test L.Ratio p-value
fm1Wheat2      1 57 1354.7 1532.8 -620.37
fm3Wheat2      2 59 1183.3 1367.6 -532.64 1 vs 2  175.46  <.0001
```

The large value of the likelihood ratio test statistics gives strong evidence against the assumption of independence.

We can verify the adequacy of the corRatio model by examining the plot of the sample semivariogram for the normalized residuals of `fm3Wheat2`.

```
> plot( Variogram(fm3Wheat2, resType = "n") )          # Figure 5.24
```

No patterns are observed in the plot of the sample semivariogram, suggesting that the rational quadratic model is adequate.

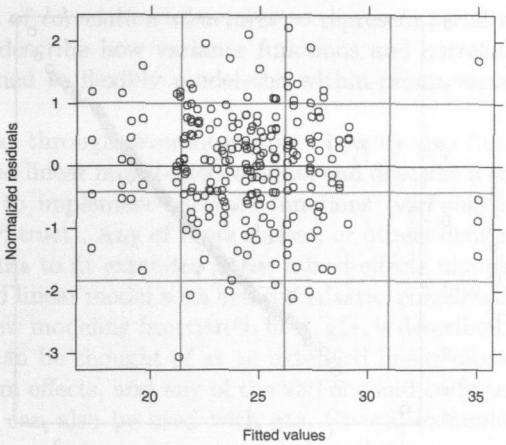

FIGURE 5.25. Scatter plot of normalized residuals versus fitted values for the fitted object fm3Wheat2.

The normalized residuals are also useful for investigating heteroscedasticity and departures from normality. For example, the plot of the normalized residuals versus the fitted values, displayed in Figure 5.25 and obtained with

```
> plot( fm3Wheat2, resid(., type = "n") ~ fitted(.),
+       abline = 0 )                                  # Figure 5.25
```

does not indicate any heteroscedastic patterns. The normal plot of the normalized residuals in Figure 5.26 is obtained with

```
> qqnorm( fm3Wheat2, ~ resid(., type = "n") )        # Figure 5.26
```

No significant departures from the assumption of normality are observed in this plot.

The anova method can be used to assess differences between the wheat varieties. For example, to test if there are any differences between varieties, we first have to reparametrize (5.36) in terms of an intercept plus contrasts between the varieties and then use anova.

```
> fm4Wheat2 <- update( fm3Wheat2, model = yield ~ variety )
> anova( fm4Wheat2 )
Denom. DF: 168
            numDF F-value p-value
(Intercept)     1  30.405  <.0001
    variety    55   1.851  0.0015
```

The small p-value of the F-test for variety indicates that there are significant differences between varieties. We can test specific contrasts using the

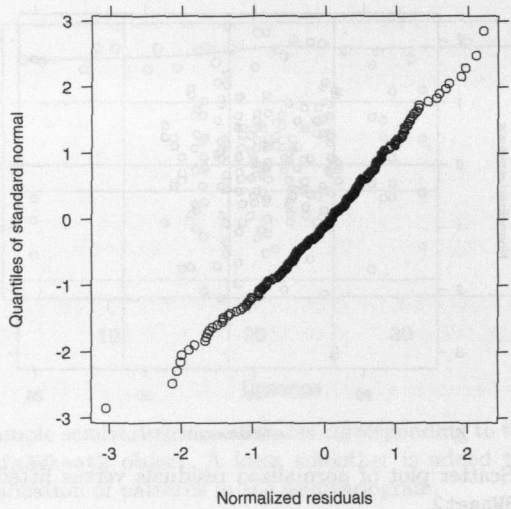

FIGURE 5.26. Normal plot of normalized residuals for the fitted object `fm3Wheat2`.

argument `L` to `anova`, with the original *cell means* parametrization. For example, to test the difference between the first and the third wheat varieties we use

```
> anova( fm3Wheat2, L = c(-1, 0, 1) )
Denom. DF: 168
 F-test for linear combination(s)
 varietyARAPAHOE varietyBUCKSKIN
              -1               1
  numDF F-value p-value
1     1  7.6966  0.0062
```

The small p-value for the F-test, combined with the coefficient estimates displayed previously, indicates that the `BUCKSKIN` variety has significant higher yields than the `ARAPAHOE` variety. Similar analyses can be obtained for other linear contrasts of the model coefficients.

5.5 Chapter Summary

In this chapter the linear mixed-effects model of Chapters 2 and 4 is extended to include heteroscedastic, correlated within-group errors. We show how the estimation and computational methods of Chapter 2 can be extended to this more general linear mixed-effects model. We introduce several classes of *variance functions* to characterize heteroscedasticity and

several classes of *correlation structures* to represent serial and spatial correlation, and describe how variance functions and correlations structures can be combined to flexibly model the within-group variance–covariance structure.

We illustrate, through several examples, how the `lme` function is used to fit the extended linear mixed-effects model and describe a suite of S classes and methods to implement variance functions (varFunc) and correlation structures (corStruct). Any of these classes, or others defined by users, can be used with `lme` to fit extended linear mixed-effects models.

An extended linear model with heteroscedastic, correlated errors is introduced and a new modeling function to fit it, `gls`, is described. This extended linear model can be thought of as an extended linear mixed-effects model with no random effects, and any of the varFunc and corStruct classes available with `lme` can also be used with `gls`. Several examples are used the illustrate the use of `gls` and its associated methods.

Exercises

1. The within-group heteroscedasticity observed for the `fm1BW.lme` fit of the `BodyWeight` data in §5.2.2 was modeled using the power variance function (`varPower`) with the fitted values as the variance covariate. An alternative approach, which is explored in this exercise, is to allow different variances for each `Diet`.

   (a) Plot the residuals of `fm1BW.lme` versus `Diet` (use `plot(fm1BW.lme, resid(.) ~ as.integer(Diet), abline = 0)`). Note that the variability for `Diet`s 1 and 2 are similar, but the residuals for `Diet` 3 have larger variability.

   (b) Update the `fm1BW.lme` fit allowing different variances per `Diet` (use `weights = varIdent(form = ~1|Diet)`). To get a fit that can be compared to `fm1BW.lme`, remember to set the contrasts parameterization to `"contr.helmert"`. Obtain confidence intervals on the variance function coefficients using `intervals`. Do they agree with the conclusions from the plot of the residuals versus `Diet`? Explain.

   (c) Compare the fit with the `varIdent` variance function to `fm1BW.lme` using `anova`. Compare it also to `fm2BW.lme`, the fit with the `varPower` variance function. Which variance function model is preferable? Why?

   (d) Use the `gls` function described in §5.4 to fit a model with a `varPower` variance function in the fitted values and a `corCAR1` correlation structure in `Time`, but with no random effects. Com-

pare the resulting fit to the `fm3BW.lme` fit of §5.3.4 using `anova`. Are the two models nested? Why?

2. A multilevel LME analysis of the `Oats` data was presented in §1.6 and an equivalent single-level analysis using a block-diagonal Ψ matrix (`pdBlocked` class) was discussed in §4.2.2. This exercise illustrates yet another possible approach for analyzing split-plot experiments such as the `Oats` data.

   (a) Fit an LME model to the `Oats` data using the same fixed-effects model as in the `fm4Oats` fit of §1.6, a single random intercept at the `Block` level, and a general correlation structure (`corSymm`) at the `Variety` within `Block` level (use `random = ~1|Block` and `corr = corSymm(form = ~1|Block/Variety)`).

   (b) Obtain the confidence intervals on the within-group correlations associated with the `corSymm` structure. Note that all intervals overlap and the estimates are of similar magnitude. This suggests a compound symmetry correlation structure. Update the previous fit using `corr = corCompSymm(form=~1|Block/Variety)`. Assess the validity of the `corCompSymm` model using `anova`.

   (c) Compare the fit obtained in the previous item to the `fm4Oats` fit. Note that the log-likelihoods are nearly identical and so are the estimated fixed effects and the estimated standard error for `Blocks` ($\widehat{\sigma}_1$). The two models are actually identical: verify that the estimated within-group variance for the `lme` fit with the `corCompSymm` structure is equal to the sum of the estimated `Variety` within `Block` variance ($\widehat{\sigma}_2^2$) and the estimated within-group variance ($\widehat{\sigma}^2$) in the `fm4Oats` fit and that the estimated correlation for the `corCompSymm` structure is equal to $\widehat{\sigma}_2^2/(\widehat{\sigma}_2^2+\widehat{\sigma}^2)$.

   (d) Use `gls` to fit a model with `corCompSymm` correlation structure at the `Variety` within `Block` level, but with no random effects and use it to assess the significance of the "`Block` effect" in the `Oats` experiment.

3. Obtain a `gls` fit of the `Ovary` data using the same fixed-effects model and correlation structure as in the `fm5Ovar.lme` fit of §5.3.4 but with no random effects (use `corr = corARMA(form=~1|Mare, p=1, q=1)` to define the grouping structure). Compare the fit with `fm5Ovar.lme` using `anova`. Is there significant evidence for keeping the random effects in the model? Can you give an explanation for what you found?

4. As mentioned in §5.2, new `varFunc` classes representing user-defined variance functions can be incorporated into the `nlme` library and used with any of its modeling function. This exercise illustrates how a new `varFunc` class can be created.

We use a variance function based on a linear combination of covariates, which we denote `varReg`. Letting $\boldsymbol{v}$ denote a vector of variance covariates and $\boldsymbol{\delta}$ the variance parameters, the `varReg` variance model and variance function are defined as

$$\mathrm{Var}(\epsilon) = \sigma^2 \exp\left(2\boldsymbol{v}^T\boldsymbol{\delta}\right), \qquad g(\boldsymbol{v}, \boldsymbol{\delta}) = \exp\left(\boldsymbol{v}^T\boldsymbol{\delta}\right),$$

where, as in §5.2, ϵ represents the random variable whose variance is being modeled. Note that the variance parameters $\boldsymbol{\delta}$ are unconstrained and, for identifiability, the linear model in the variance function can not have an intercept (it would be confounded with σ). The `varReg` class extends the `varExp` class defined in §5.2 by allowing more than one variance covariate to be used. The `varReg` class is frequently used in the analysis of dispersion effects in robust designs (Wolfinger and Tobias, 1998).

(a) Write a constructor for the `varReg` class. It should contain at least two arguments: `value` with initial values for the variance parameters $\boldsymbol{\delta}$ (set the default to `numeric(0)` to indicate non-initialized structures) and `form` with a linear formula defining the variance covariates. To simplify things, assume that the fitted object can not be used to define a variance covariate for this class, that no stratification of parameters will be available and that no parameters can be made fixed in the estimation. You can use the `varExp` constructor as a template.

(b) Next you need to write an `initialize` method. This takes two required arguments `object`, the `varReg` object, and `data`, a data frame in which to evaluate the variance covariates. For consistency with the generic function, include a ... at the end of the argument list. The initialize method should obtain the model matrix corresponding to `formula(object)`, evaluated on `data`, and save it as an attribute for later calculations. As mentioned before, the model matrix should not have an `(Intercept)` column; check if one is present and remove it if necessary. You should make sure that the parameters are initialized (if no starting values are given in the constructor, initialize them to 0, corresponding to an homoscedastic variance model). Additionally, the `"logLik"` and `"weights"` attributes of the returned object need to be initialized. Note that the weights are simply `exp(-modMat %*% coefs)`, where `modMat` represents the model matrix for the variance covariates and `coefs` the initialized coefficients. You can use the `initialize.varExp` method as a template.

(c) The `coef` method for the `varReg` class is very simple: for consistency with other `varFunc coef` methods, it takes three arguments: `object` (the `varReg` object), `unconstrained`, and `allCoef`. Because

δ is unconstrained and we are not allowing any parameters to be fixed, the method will always return `as.vector(object)`.

(d) The `coef<-` method takes two arguments: `object`, the `varReg` object, and `value`, the new values for the coefficients. This method is also used to update the value of the `"logLik"` and `"weights"` attributes, after the coefficients in `object` have been updated. You can use `coef<-.varExp` as a template.

(e) Write a `summary.varReg` method to provide a description of the `varReg` variance function when it is the `varReg` object is printed (usually as part of the output of some modeling function). Use `summary.varExp` as an example.

(f) Test your class by fitting an LME model to the `BodyWeight` data with a different variances for each `Diet` (you can use `weights = varReg(form = ~Diet)`, because the `initialize` method will remove the `(Intercept)` column of the model matrix).