2010

Fast computing of some generalized linear mixed pseudo-models with temporal autocorrelation

Jay M. Ver Hoef
National Marine Mammal Laboratory, Alaska Fisheries Science Center, NOAA National Marine Fisheries Service

Josh M. London
National Marine Mammal Laboratory, Alaska Fisheries Science Center, NOAA National Marine Fisheries Service

Peter L. Boveng
National Marine Mammal Laboratory, Alaska Fisheries Science Center, NOAA National Marine Fisheries Service

Comput Stat (2010) 25:39–55
DOI 10.1007/s00180-009-0160-1

ORIGINAL PAPER

Fast computing of some generalized linear mixed pseudo-models with temporal autocorrelation

Jay M. Ver Hoef · Josh M. London · Peter L. Boveng

Received: 30 July 2008 / Accepted: 3 June 2009 / Published online: 18 June 2009

Abstract This paper considers ways to increase computational speed in generalized linear mixed pseudo-models for the case of many repeated measurements on subjects. We obtain linearly increasing computing time with number of observations, as opposed to O(n^3) increasing computing time using numerical optimization. We also find a surprising result; that incomplete optimization for covariance parameters within the larger parameter estimation algorithm actually decreases time to convergence. After comparing various computing algorithms and choosing the best one, we fit a generalized linear mixed model to a binary time series data set with over 100 fixed effects, 50 random effects, and approximately 1.5×10^5 observations.

Keywords Pseudo-likelihood · Sherman–Morrison–Woodbury · Sparse matrix · Exponential autocorrelation

1 Introduction

Generalized linear mixed models (GLMMs) are emerging as one of the most important classes of statistical models in use today. The combination of random and fixed effects, along with distributions in the exponential family, allow us to model binary, proportional, count, and continuous data. As in linear mixed models, the GLMMs can accommodate correlated error structures such as those found in longitudinal, time

J. M. Ver Hoef · J. M. London · P. L. Boveng
National Marine Mammal Laboratory, Alaska Fisheries Science Center,
NOAA National Marine Fisheries Service, 7600 Sand Point Way NE,
Bldg 4, Seattle, WA 98115-6349, USA

J. M. Ver Hoef (✉)
National Weather Service, P.O. Box 757345, Fairbanks, AK 99775-7345, USA
e-mail: jay.verhoef@noaa.gov

series, and spatial statistics. However, estimation of covariance parameters and fixed effects remains a problem, especially for massive data sets. In this article, we concentrate on the case of many repeated measurements on subjects, which occurs in many fields of science including medicine and biology. Methods such as pseudo-likelihood, like those proposed by Wolfinger and O'Connell (1993), are highly iterative and may require inverting large matrices for spatial or temporal models. This paper considers five ways to increase computational speed in the GLMM pseudo-models in the case of many repeated measures with temporal autocorrelation, including (1) using the Sherman/Morrison/Woodbury matrix identity, (2) using analytical matrix inverses and determinants, (3) using sparse matrix ideas, (4) using block diagonal structures, and (5) using incomplete optimizations within other optimization loops. We develop code using these ideas and compare the results to the glmmPQL function (Venables and Ripley 2002) in R (R Development Core Team 2007). Finally, we show an example by fitting a Bernoulli distribution GLMM to a massive data set of over 150,000 records consisting of long binary time series on 50 harbor seals (*Phoca vitulina*).

This paper is organized as follows. In Sect. 2, we review likelihood estimation for GLMMs. In Sect. 3, we consider the five ways to increase computational speed. We introduce an autocorrelated binary data set in Sect. 4. We use subsets of these data to compare computing speeds among several algorithms, to compare to the glmmPQL function in R, and to model all data (approximately 1.5×10^5 observations). Discussion and conclusions are provided in Sect. 5.

2 Review of likelihood estimation for generalized linear mixed pseudo-models

The mean of a generalized linear mixed model, conditional on the random effects, is given by

$$E(\mathbf{Y}|\boldsymbol{\gamma}) = g^{-1}(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma}) = g^{-1}(\eta) = \boldsymbol{\mu}, \tag{1}$$

where $\boldsymbol{\gamma} \sim N(\mathbf{0}, \mathbf{G}(\boldsymbol{\nu}))$ and var $(\mathbf{Y}|\boldsymbol{\gamma}) = \mathbf{A}^{\frac{1}{2}}\sigma^2\mathbf{R}(\boldsymbol{\rho})\mathbf{A}^{\frac{1}{2}}$ and we assume that $\mathbf{G}(\boldsymbol{\nu})$ is diagonal and positive definite. We show the dependence of the covariance matrix $\mathbf{G}$ and correlation matrix $\mathbf{R}$ on parameter vectors $\boldsymbol{\nu}$ and $\boldsymbol{\rho}$. We use p to denote the rank of $\mathbf{X}$ and k as the number of columns in $\mathbf{Z}$. We assume that the number of covariates and random effects, p and k, are in the hundreds, and are much smaller than the number of observations per subject, which could be thousands. Here, $g(\cdot)$ is a link function with $g^{-1}(\cdot)$ as its inverse, and $\mathbf{A}$ is a diagonal matrix that contains the variance functions $a(\mu)$ of the model, which expresses the variance of a response as a function of the mean. This variance function is often determined by the assumed distribution of the data (e.g., for binomial it is $a(\mu) = \mu(1-\mu)/n$, for Poisson it is $a(\mu) = \mu$, etc.). We will consider pseudo-models of Wolfinger and O'Connell (1993) which are based on a Taylor's series expansions. Expanding $\boldsymbol{\mu}$ around $\tilde{\boldsymbol{\beta}}$ and $\tilde{\boldsymbol{\gamma}}$, the pseudo-model is given by

$$\tilde{\mathbf{Y}} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}, \tag{2}$$

where

$$\tilde{\mathbf{Y}} \equiv \tilde{\boldsymbol{\Delta}}^{-1}(\mathbf{Y} - g^{-1}(\mathbf{X}\tilde{\boldsymbol{\beta}} + \mathbf{Z}\tilde{\boldsymbol{\gamma}})) + \mathbf{X}\tilde{\boldsymbol{\beta}} + \mathbf{Z}\tilde{\boldsymbol{\gamma}}, \tag{3}$$

and

$$\tilde{\boldsymbol{\Delta}} \equiv \frac{\partial g^{-1}(\boldsymbol{\eta})}{\partial \boldsymbol{\eta}}$$

is a diagonal matrix evaluated at $\tilde{\boldsymbol{\beta}}$ and $\tilde{\boldsymbol{\gamma}}$. Based on the covariance matrix specification of $\mathbf{Y}$, we have

$$\mathrm{var}(\boldsymbol{\epsilon}) = \tilde{\boldsymbol{\Delta}}^{-1}\mathbf{A}^{\frac{1}{2}}\sigma^2\mathbf{R}(\boldsymbol{\rho})\mathbf{A}^{\frac{1}{2}}\tilde{\boldsymbol{\Delta}}^{-1}.$$

The matrix $\mathbf{R}(\boldsymbol{\rho})$ is a covariance matrix with off-diagonal elements that might be modeled with spatial, temporal, or other correlation models. Here, we will consider temporal models. The parameters of the temporal autocorrelation model are contained in the vector $\boldsymbol{\rho}$. We consider the marginal variance of the linear mixed pseudo-model as,

$$\mathbf{V}(\boldsymbol{\theta}) = \mathbf{Z}\mathbf{G}(\boldsymbol{\nu})\mathbf{Z}' + \tilde{\boldsymbol{\Delta}}^{-1}\mathbf{A}^{\frac{1}{2}}\sigma^2\mathbf{R}(\boldsymbol{\rho})\mathbf{A}^{\frac{1}{2}}\tilde{\boldsymbol{\Delta}}^{-1}, \tag{4}$$

where $\boldsymbol{\theta} = (\sigma^2, \boldsymbol{\nu}', \boldsymbol{\rho}')'$.

2.1 Pseudo-log-likelihoods

The formulation of pseudo-data in (3) can be viewed as a transformation that creates data that are approximately Gaussian. As for normal mixed models, we can consider maximum likelihood and restricted maximum likelihood (REML, Patterson and Thompson 1971, 1974) to estimate covariance parameters and fixed effect parameters. REML is less biased than full maximum likelihood (Mardia and Marshall 1984), but it is not clear how important this is in the pseudo-models. We present both cases. Define minus two times the pseudo-log-likelihood as

$$l(\boldsymbol{\theta}; \tilde{\mathbf{y}}) = \log|\mathbf{V}(\boldsymbol{\theta})| + \mathbf{r}'\mathbf{V}(\boldsymbol{\theta})^{-1}\mathbf{r} + c, \tag{5}$$

where $V(\boldsymbol{\theta})$ is defined in (4), $\mathbf{r} = \tilde{\mathbf{y}} - \mathbf{X}\hat{\boldsymbol{\beta}}$, $\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{V}(\boldsymbol{\theta})^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{V}(\boldsymbol{\theta})^{-1}\tilde{\mathbf{y}}$, $\mathbf{X}$ is the design matrix, and c is a constant that does not depend on $\boldsymbol{\theta}$ or $\boldsymbol{\beta}$. Equation (5) is the well-known objective function for the maximum likelihood estimator (MLE), and minimizing it for $\boldsymbol{\theta}$ yields the pseudo-MLE $\hat{\boldsymbol{\theta}}$, and then substituting $\hat{\boldsymbol{\theta}}$ into $\hat{\boldsymbol{\beta}}$ results in the maximum likelihood estimator of $\boldsymbol{\beta}$.

Define minus two times the restricted pseudo-log-likelihood as,

$$l(\boldsymbol{\theta}; \tilde{\mathbf{y}}) = \log|\mathbf{V}(\boldsymbol{\theta})| + \mathbf{r}'\mathbf{V}(\boldsymbol{\theta})^{-1}\mathbf{r} + \log\left|\mathbf{X}'\mathbf{V}(\boldsymbol{\theta})^{-1}\mathbf{X}\right| + c. \tag{6}$$

Equation (6) is the well-known objective function for restricted maximum likelihood (REML), and minimizing it for $\boldsymbol{\theta}$ results in the pseudo-REML estimator $\hat{\boldsymbol{\theta}}$. Substituting $\hat{\boldsymbol{\theta}}$ into $\hat{\boldsymbol{\beta}}$ results in a generalized least squares estimator of $\boldsymbol{\beta}$. Restricted maximum likelihood is obtained by integrating over all fixed effects in a multivariate Gaussian likelihood. However, Heyde (1994) and Cressie and Lahiri (1996) have shown that (6) form unbiased estimating equations for covariance parameters, so Gaussian data are not strictly necessary for their use.

2.2 Profiling

We can usually find an analytical solution to an overall variance parameter in (4) for the pseudo-log-likelihoods in (5) and (6). Let us factor out σ^2 so that (4) is

$$\mathbf{V}(\boldsymbol{\theta}) = \sigma^2(\mathbf{Z}\mathbf{S}(\boldsymbol{\nu})\mathbf{Z}' + \tilde{\boldsymbol{\Delta}}^{-1}\mathbf{A}^{\frac{1}{2}}\mathbf{R}(\boldsymbol{\rho})\mathbf{A}^{\frac{1}{2}}\tilde{\boldsymbol{\Delta}}^{-1}) \equiv \sigma^2\mathbf{Q}(\boldsymbol{\psi}), \tag{7}$$

where $\mathbf{S}(\boldsymbol{\nu}) = \mathbf{G}(\boldsymbol{\nu})/\sigma^2$ and $\boldsymbol{\psi} = (\boldsymbol{\nu}', \boldsymbol{\rho}')'$. With the profiling of σ^2, (5) can be minimized for σ^2 analytically,

$$\hat{\sigma}^2 = \mathbf{q}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{q}/n, \tag{8}$$

where $\mathbf{q} = \tilde{\mathbf{y}} - \mathbf{X}(\mathbf{X}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\tilde{\mathbf{y}}$. Substituting (8) into (5) yields the profiled pseudo-log-likelihood,

$$l_q(\boldsymbol{\psi}; \tilde{\mathbf{y}}) = \log|\mathbf{Q}(\boldsymbol{\psi})| + n\log(\mathbf{q}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{q}) + c. \tag{9}$$

Equation (6) can also be minimized for σ^2 analytically,

$$\hat{\sigma}^2 = \mathbf{q}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{q}/(n-p), \tag{10}$$

where p is the rank of $\mathbf{X}$. Substituting (10) into (6) yields the profiled restricted pseudo-log-likelihood,

$$l_q(\boldsymbol{\psi}; \tilde{\mathbf{y}}) = \log|\mathbf{Q}(\boldsymbol{\psi})| + (n-p)\log(\mathbf{q}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{q}) + \log\left|\mathbf{X}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{X}\right| + c. \tag{11}$$

Profiling offers computational gains because (9) and (11) have one less parameter than (5) and (6) for minimization.

2.3 Estimating β and γ

Once we have $\hat{\boldsymbol{\theta}}$, we can use generalized least squares (equal to maximum likelihood for (5)) to estimate $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ with

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{V}(\hat{\boldsymbol{\theta}})^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{V}(\hat{\boldsymbol{\theta}})^{-1}\tilde{\mathbf{y}},$$

which also equals

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{Q}(\hat{\boldsymbol{\psi}})^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{Q}(\hat{\boldsymbol{\psi}})^{-1}\tilde{\mathbf{y}}, \tag{12}$$

when minimizing (9) or (11) to obtain $\hat{\boldsymbol{\psi}}$. Also, $\hat{\boldsymbol{\gamma}} = \mathbf{G}(\hat{\boldsymbol{\nu}})\mathbf{Z}'\mathbf{V}(\hat{\boldsymbol{\theta}})^{-1}\hat{\mathbf{r}}$, which equals

$$\hat{\boldsymbol{\gamma}} = \mathbf{S}(\hat{\boldsymbol{\nu}})\mathbf{Z}'\mathbf{Q}(\hat{\boldsymbol{\psi}})^{-1}\hat{\mathbf{q}}, \tag{13}$$

where recall that $\hat{\mathbf{q}} = \tilde{\mathbf{y}} - \mathbf{X}(\mathbf{X}'\mathbf{Q}(\hat{\boldsymbol{\psi}})^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{Q}(\hat{\boldsymbol{\psi}})^{-1}\tilde{\mathbf{y}}$.

2.4 Parameter estimation algorithm

Suppose that we have initial values $\tilde{\boldsymbol{\beta}}^{[0]}$ and $\tilde{\boldsymbol{\gamma}}^{[0]}$, and we have defined our generalized linear mixed model in (1) with a link function $g(\cdot)$, variance function $a(\mu)$, mixed effects covariance matrix $\mathbf{G}(\boldsymbol{\nu})$, and temporal autocorrelation model $\mathbf{R}(\boldsymbol{\rho})$. Then Wolfinger and O'Connell (1993) suggest the following iterative parameter estimation algorithm.

- Step 1. Form pseudo-data $\tilde{\mathbf{y}}^{[m+1]}$ using (3) with the current estimates $\tilde{\boldsymbol{\beta}}^{[m]}$ and $\tilde{\boldsymbol{\gamma}}^{[m]}$ for the mth iteration.
- Step 2. Estimate $\hat{\boldsymbol{\psi}}^{[m+1]}$ using $\tilde{\mathbf{y}}^{[m+1]}$ in (9) or (11) and minimizing for $\boldsymbol{\psi}$, and then estimate σ^2 using (8) or (10). This requires iterative minimization methods, and we will call this the step 2 optimization.
- Step 3. Estimate $\hat{\boldsymbol{\beta}}$ using (12) and $\hat{\boldsymbol{\gamma}}$ using (13), and then set $\tilde{\boldsymbol{\beta}}^{[m+1]} = \hat{\boldsymbol{\beta}}$ and $\tilde{\boldsymbol{\gamma}}^{[m+1]} = \hat{\boldsymbol{\gamma}}$. This process of updating $\tilde{\boldsymbol{\beta}}^{[m]}$ and $\tilde{\boldsymbol{\gamma}}^{[m]}$ is called the step 3 optimization.
- Step 4. Set $m = m + 1$ and go to Step 1 if convergence criteria are not satisfied. Each iteration through m is called an outer iteration.

The step 3 optimization is closely related to iterative weighted least squares (e.g., McCullagh and Nelder 1989, p. 40) that is used for estimating parameters of generalized linear models when assuming all of the data are independent. When there are large data sets with temporal autocorrelation, the step 2 optimization can be very slow because the minimization of (9) or (11) requires matrix inverses and determinants whose dimensions are equal to the number of observations. Even storing such matrices can be a problem when we have hundreds of thousands of records. In the next section, we use some analytical results on matrix inverses, quick minimization algorithms, and sparse matrix ideas to speed the computation and reduce computer storage requirements.

3 Increasing computation speeds

We consider five main ways to accelerate and stabilize the parameter estimation algorithm of Sect. 2.4, some of which are well-known, but we show all results for completeness: (1) Sherman–Morrison–Woodbury matrix identity, (2) analytical inverse and determinant of an exponential covariance matrix for time series, (3) sparse matrices, (4) block diagonal structures, and (5) incomplete minimization during the step 2 optimization.

3.1 Sherman–Morrison–Woodbury matrix identity

Let us suppress the parameter displays and write $\mathbf{Q}(\boldsymbol{\psi})$ in (7) as $\mathbf{ZSZ} + \boldsymbol{\Sigma}$, where $\boldsymbol{\Sigma} = \tilde{\boldsymbol{\Delta}}^{-1}\mathbf{A}^{\frac{1}{2}}\mathbf{R}(\boldsymbol{\rho})\mathbf{A}^{\frac{1}{2}}\tilde{\boldsymbol{\Delta}}^{-1}$. Let $\mathbf{Z}$ be $n \times k$, let $\mathbf{G}$ be $k \times k$ and nonsingular, and let $\boldsymbol{\Sigma}$ be $n \times n$ and nonsingular. Typically, the number of random effects k is much less than the number of data n for large data sets. Using the Sherman–Morrison–Woodbury formula (Sherman and Morrison 1949; Woodbury 1950; see an excellent review by Henderson and Searle 1981), we have

$$\mathbf{Q}(\boldsymbol{\psi})^{-1} = (\mathbf{ZSZ}' + \boldsymbol{\Sigma})^{-1} = \boldsymbol{\Sigma}^{-1} - \boldsymbol{\Sigma}^{-1}\mathbf{Z}(\mathbf{S}^{-1} + \mathbf{Z}'\boldsymbol{\Sigma}^{-1}\mathbf{Z})^{-1}\mathbf{Z}'\boldsymbol{\Sigma}^{-1}. \quad (14)$$

Typically, $\mathbf{S}$ is diagonal, in which case the inverse is easy; or if it is not diagonal, it is of order k, which can be much less than n and allow a numerical inverse. Equation (14) will provide no computational advantages unless $\boldsymbol{\Sigma}^{-1}$ is easy. That is, if $\boldsymbol{\Sigma}^{-1}$ has an analytical inverse, then the largest numerical inverse that needs computing is the $k \times k$ matrix $(\mathbf{S}^{-1} + \mathbf{Z}'\boldsymbol{\Sigma}^{-1}\mathbf{Z})$, which can be a large computational savings.

Similarly, we have the following determinant identity,

$$|\mathbf{Q}(\boldsymbol{\psi})| = \left|\mathbf{ZSZ}' + \boldsymbol{\Sigma}\right| = |\boldsymbol{\Sigma}|\,|\mathbf{S}|\left|\mathbf{S}^{-1} + \mathbf{Z}'\boldsymbol{\Sigma}^{-1}\mathbf{Z}\right| \quad (15)$$

Again, if $\mathbf{S}$ is diagonal the determinant is easy, and in any case both $\mathbf{S}$ and $(\mathbf{S}^{-1} + \mathbf{Z}'\boldsymbol{\Sigma}^{-1}\mathbf{Z})$ are of order $k \times k$, so if k is not too large, numerical determinants can be used. The main problems in (14) and (15) are $\boldsymbol{\Sigma}^{-1}$ and $|\boldsymbol{\Sigma}|$, which we address next.

3.2 Analytical inverse and determinant of an exponential covariance matrix for time series

It is well-known that an analytical inverse exists for temporal AR(1) covariance matrices (e.g., Hamilton 1994, p. 120). This idea can be extended to irregularly spaced time intervals by using an exponential autocorrelation model. Let the data be ordered by time and indexed on the integers; then, for the AR(1) model, the i, jth element of the covariance matrix is

$$\delta^2\mathbf{R}(\boldsymbol{\rho})[i, j] = \frac{\delta^2}{1-\rho^2}\rho^{|i-j|}, \quad (16)$$

where $\boldsymbol{\rho}$ is a scalar ρ and $0 < \delta$ and $-1 \leq \rho < 1$. Now, let us impose the parameter restriction $0 < \rho < 1$. Then $\rho^{|i-j|} = \exp(|i - j|\log(\rho))$. Next, let's suppose that we denote the time of the ith observation as t_i, and reparameterize by letting $\alpha = 1/\log(\rho)$, then the autocorrelation is $\exp(-\left|t_i - t_j\right|/\alpha)$ where the minus sign occurs because $\log(\rho)$ is negative, and hence we set $0 < \alpha$. We can also reparameterize $\delta^2/(1-\rho^2)$ to obtain

$$\sigma^2\mathbf{R}(\boldsymbol{\rho})[i, j] = \sigma^2 e^{-|t_i - t_j|/\alpha}, \tag{17}$$

where here $\boldsymbol{\rho} = \alpha$. If $t_i = i$ and $t_j = j$ for data ordered by time and indexed on the integers, then (16) and (17) are equivalent for the restricted parameter space, but are different parameterizations. From now on we work with (17) because it is more convenient for irregularly-spaced time intervals.

Let us denote $d_{i,j} = |t_i - t_j|$. If we order the data by time, then the correlation matrix will be $\mathbf{R}(\boldsymbol{\rho}) =$

$$\begin{pmatrix} 1 & e^{-d_{1,2}/\alpha} & e^{\frac{-(d_{1,2}+d_{2,3})}{\alpha}} & \cdots & e^{\Sigma_{i=1}^{n-1}\frac{-d_{i,i+1}}{\alpha}} \\ e^{-d_{1,2}/\alpha} & 1 & e^{-d_{2,3}/\alpha} & \cdots & e^{\Sigma_{i=2}^{n-1}\frac{-d_{i,i+1}}{\alpha}} \\ e^{\frac{-(d_{1,2}+d_{2,3})}{\alpha}} & e^{-d_{2,3}/\alpha} & 1 & \cdots & e^{\Sigma_{i=3}^{n-1}\frac{-d_{i,i+1}}{\alpha}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ e^{\Sigma_{i=1}^{n-1}\frac{-d_{i,i+1}}{\alpha}} & e^{\Sigma_{i=2}^{n-1}\frac{-d_{i,i+1}}{\alpha}} & e^{\Sigma_{i=3}^{n-1}\frac{-d_{i,i+1}}{\alpha}} & \cdots & 1 \end{pmatrix}. \tag{18}$$

Notice that (18) is not Toeplitz, so some traditional inverse techniques do not apply; however there are very simple, patterned row operations that create mostly zeros in $\mathbf{R}(\boldsymbol{\rho})$. For example, if we take the first row minus $e^{-d_{1,2}/\alpha}$ times the second row, all elements in the first row will be zero except for the first column; we can write this row operation as the matrix $\mathbf{O}_1$. Proceeding in this manner, we can create all zeros in the matrix above the diagonal. The product of these row operations, $\mathbf{O}_{m_U} \ldots \mathbf{O}_1 = \mathbf{U}$, where

$$\mathbf{U} = \begin{pmatrix} 1 & -e^{-d_{1,2}/\alpha} & 0 & \cdots & 0 & 0 \\ 0 & 1 & -e^{-d_{2,3}/\alpha} & \cdots & 0 & 0 \\ 0 & 0 & 1 & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -e^{-d_{n-1,n}/\alpha} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}. \tag{19}$$

If we continue with row operations on $\mathbf{UR}(\boldsymbol{\rho})$, we can divide the diagonal elements by their current values to get a 1 for each diagonal element; all row operations so far are represented as $\mathbf{O}_{m_U+m_D} \ldots \mathbf{O}_{m_U+1}\mathbf{O}_{m_U} \ldots \mathbf{O}_1$. Let $\mathbf{D} = \mathbf{O}_{m_U+m_D} \ldots \mathbf{O}_{m_U+1} = \text{diag}[1/(1-\exp(-2d_{i,i+1}/\alpha))]$ for $i < n$ and $\mathbf{D}[n, n] = 1$. Finally, we can proceed to create zeros on the lower half of $\mathbf{DUR}(\boldsymbol{\rho})$ by taking $\mathbf{U}'\mathbf{DUR}(\boldsymbol{\rho})$, which equals the identity matrix $\mathbf{I}$, so $\mathbf{U}'\mathbf{DU}$ is the inverse of $\mathbf{R}(\boldsymbol{\rho})$. Notice that $\mathbf{U}'\mathbf{DU}$ is tridiagonal, just like the inverse of a matrix for an AR(1) model (e.g., Hamilton 1994, p. 120).

If we use $|\mathbf{ABC}| = |\mathbf{A}|\,|\mathbf{B}|\,|\mathbf{C}|$, $\left|\mathbf{A}^{-1}\right| = |\mathbf{A}|^{-1}$, then

$$|\mathbf{R}(\boldsymbol{\rho})| = \prod_{i=1}^{n-1}(1 - e^{-2d_{i,i+1}/\alpha}). \tag{20}$$

3.3 Using sparse matrix ideas

The tridiagonal inverse developed in Sect. 3.2 is sparse. In this section, we show that we can go directly to the products of the sparse matrices, so storage of the inverse is not needed. Let $\mathbf{D}^{1/2}$ be a diagonal matrix with elements that are the square roots of the diagonal elements of $\mathbf{D}$. Note that $\mathbf{D}^{1/2}\mathbf{U}$ is bidiagonal, and so is $\mathbf{L} \equiv \mathbf{D}^{1/2}\mathbf{U}\mathbf{A}^{-\frac{1}{2}}\tilde{\boldsymbol{\Delta}}$. Hence

$$\boldsymbol{\Sigma}^{-1} = \left(\tilde{\boldsymbol{\Delta}}^{-1}\mathbf{A}^{\frac{1}{2}}\mathbf{R}(\boldsymbol{\rho})\mathbf{A}^{\frac{1}{2}}\tilde{\boldsymbol{\Delta}}^{-1}\right)^{-1} = \mathbf{L}'\mathbf{L}. \tag{21}$$

Notice that in (9) and (11), we can write

$$\mathbf{q}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{q} = \tilde{\mathbf{y}}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\tilde{y} - \tilde{\mathbf{y}}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{X}(\mathbf{X}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\tilde{\mathbf{y}}. \tag{22}$$

Substituting (14) for $\mathbf{Q}(\hat{\boldsymbol{\psi}})^{-1}$ in (22), we see that all terms involving $\boldsymbol{\Sigma}^{-1}$ are $\mathbf{X}'\boldsymbol{\Sigma}^{-1}\mathbf{X}$, $\mathbf{X}'\boldsymbol{\Sigma}^{-1}\mathbf{Z}$, $\mathbf{Z}'\boldsymbol{\Sigma}^{-1}\mathbf{Z}$, $\mathbf{X}'\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{y}}$, $\mathbf{Z}'\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{y}}$,and $\tilde{\mathbf{y}}'\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{y}}$. Now, using (21), the six terms above can be computed using the three matrices $\mathbf{LX}$, $\mathbf{LZ}$, and $\mathbf{L}\tilde{\mathbf{y}}$. The important part here is that $\mathbf{L}$ does not need to be computed because it is so sparse; rather, we can compute and store $\mathbf{L}_\mathbf{X} \equiv \mathbf{LX}$, $\mathbf{L}_\mathbf{Z} \equiv \mathbf{LZ}$, and $\mathbf{L}_\mathbf{y} \equiv \mathbf{L}\tilde{\mathbf{y}}$ directly. This obviates the need to even store a sparse matrix. For example, the ith element of $\mathbf{L}\tilde{\mathbf{y}}$ is $\tilde{\boldsymbol{\Delta}}[i,i](\tilde{y}_i - \tilde{y}_{i+1}\exp(-d_{i,i+1}/\alpha))/(\sqrt{1-\exp(-d_{i,i+1}/\alpha)}\mathbf{A}^{\frac{1}{2}}[i,i])$ for $i < n$, and equals $\tilde{\boldsymbol{\Delta}}[i,i]\tilde{y}_n/\mathbf{A}^{\frac{1}{2}}[i,i]$ for $i = n$, and a similar operation is made on each column of $\mathbf{X}$ and $\mathbf{Z}$ for $\mathbf{L}_\mathbf{X}$ and $\mathbf{L}_\mathbf{Z}$, respectively. These computations only require one pass through the data, so, all other things being equal, the computational burden (evaluation of the likelihood) grows linearly with the number of observations, compared to order n^3 for numerical inverses.

In (22) we can compute

$$\tilde{\mathbf{y}}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\tilde{\mathbf{y}} = \mathbf{L_y}'\mathbf{L_y} - \mathbf{L_y}'\mathbf{L_Z}(\mathbf{S}^{-1} + \mathbf{L_Z}'\mathbf{L_Z})^{-1}\mathbf{L_Z}'\mathbf{L_y}, \tag{23}$$

$$\tilde{\mathbf{y}}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{X} = \mathbf{L_y}'\mathbf{L_X} - \mathbf{L_y}'\mathbf{L_Z}(\mathbf{S}^{-1} + \mathbf{L_Z}'\mathbf{L_Z})^{-1}\mathbf{L_Z}'\mathbf{L_X}, \tag{24}$$

$$\mathbf{X}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{X} = \mathbf{L_X}'\mathbf{L_X} - \mathbf{L_X}'\mathbf{L_Z}(\mathbf{S}^{-1} + \mathbf{L_Z}'\mathbf{L_Z})^{-1}\mathbf{L_Z}'\mathbf{L_X}, \tag{25}$$

relatively quickly, where we still require $\mathbf{S}^{-1}$ and $(\mathbf{S}^{-1} + \mathbf{L_Z}'\mathbf{L_Z})^{-1}$, but these are of order $k \times k$, which is often small enough for a numerical inverse, and $\mathbf{S}^{-1}$ is diagonal and has an easy inverse. Also, we assume that we can compute $\mathbf{X}'\mathbf{Q}(\boldsymbol{\psi})^{-1}\mathbf{X}$ relatively quickly (it is of order $p \times p$, along with a numerical determinant of $\left|\mathbf{X}'\mathbf{V}(\boldsymbol{\psi})^{-1}\mathbf{X}\right|$. Thus, we have fast computing methods for everything in (9) and (11) except $|\mathbf{Q}(\boldsymbol{\psi})|$.

For the determinant of $\boldsymbol{\Sigma} = \tilde{\boldsymbol{\Delta}}^{-1}\mathbf{A}^{\frac{1}{2}}\mathbf{R}(\boldsymbol{\rho})\mathbf{A}^{\frac{1}{2}}\tilde{\boldsymbol{\Delta}}^{-1}$, we have

$$|\boldsymbol{\Sigma}| = \left|\tilde{\boldsymbol{\Delta}}^{-1}\right|^2 \left|\mathbf{A}^{\frac{1}{2}}\right|^2 |\mathbf{R}(\boldsymbol{\rho})| \tag{26}$$

which can be computed quickly using (20) and that fact that $\mathbf{A}^{\frac{1}{2}}$ and $\tilde{\boldsymbol{\Delta}}$ are diagonal. Then $|\mathbf{Q}(\boldsymbol{\psi})|$ in (9) and (11) is computed using (26) in (15).

3.4 Block diagonal structure

We may have blocks of samples that we model as independent; e.g., subjects in repeated measures, etc. This creates a block diagonal structure in the covariance matrix. In the case of numerical inverses, this can offer computational advantages. However, in our situation, it confers no advantage. If blocking is used, it is important to get the correct variance structure. Let us denote the hth diagonal block of $\mathbf{R}(\boldsymbol{\rho})$ as $\mathbf{R}_h(\boldsymbol{\rho})$ for H blocks. Then we compute (20) block by block,

$$|\mathbf{R}(\boldsymbol{\rho})| = \prod_{i_h=1}^{H}\prod_{i_h=1}^{n_h-1}\left(1 - \exp\left(\frac{-2d_{i_h,i_h+1}}{\alpha}\right)\right). \tag{27}$$

In a similar block by block manner, we form

$$\mathbf{L}_h \equiv \mathbf{D}_h^{1/2}\mathbf{U}_h\mathbf{A}_h^{-\frac{1}{2}}\tilde{\boldsymbol{\Delta}}_h \tag{28}$$

circa (21), then create $\mathbf{L} = (\mathbf{L}_1'|\mathbf{L}_2'|\ldots|\mathbf{L}_H')'$, and proceed with (22–25). We can see from (27) and (28) that blocking will not speed the computations; they remain on the order of one pass through the data.

3.5 Incomplete step 2 and step 3 optimizations

Both step 2 and step 3 optimizations are moving toward convergence. For each outer iteration m of the parameter estimation algorithm of Sect. 2.4, both optimization steps 2 and 3 can have multiple iterations themselves. For example, step 2 minimizes (9) or (11) with any of a number of iterative minimization algorithms, such as a gradient based method like the Newton-Raphson algorithm (e.g., Khuri 1993, p. 330) or direct methods such as the Nelder-Mead simplex (Nelder and Mead 1965). Additionally, we can iterate within step 3 by iterating among (3), (12), and (13) in an iteratively weighted least squares algorithm while holding $\mathbf{S}(\hat{\boldsymbol{\nu}})$ and $\mathbf{Q}(\hat{\boldsymbol{\psi}})$ fixed. Upon obtaining convergence within the step 2 or step 3 optimization, we move on to the next step. Each step will have to be optimized again when m gets incremented. It seems natural to consider relaxing the within-step convergence criteria for early iterations to speed computations of that step.

For example, the idea here is to relax the convergence criteria of the step 2 optimization at early stages of the outer optimization, as the step 2 optimization will change anyway for each iteration of m. We used a very simple idea. During the step 2 optimization, we used a single cycle of a Gauss–Seidel optimization, also known as cyclic coordinate ascent (see, e.g., Givens and Hoeting 2005). In Gauss–Seidel, optimization occurs for a single parameter at a time, holding all other parameters fixed. The full optimization cycles through all parameters repeatedly until a convergence criteria is met. One dimensional optimizations are relatively easy and can use derivative free methods such as a golden search. We used the "optimize" function in R. Rather than cycle through all of the parameters minimizing each repeatedly during a step 2 optimization until some convergence criteria is satisfied, we only cycled through each parameter once per step 2 optimization. For full step 2 optimization, we used the "optim" function in R with the Broyden–Fletcher–Goldfarb–Shanno (BFGS) method. For incomplete step 3 optimization, we did a single update using (12), and (13). For complete step 3 optimization, we iterated among (3), (12), and (13) until $\hat{\boldsymbol{\beta}}$ converged to some criteria that we describe shortly.

For the rest of this paper, we will call our basic method GLMPSAM for its use of [G]eneralized [L]inear [M]ixed [P]seudo-models using the [S]herman–Morrison–Woodbury matrix identity, [A]nalytical Inverses, and sparse [M]atrix operations. The combination of incomplete and complete step 2 and step 3 optimizations forms 4 methods that we will compare in the next section:

- GLMPSAMI1 where the [I] indicates incomplete step 2 optimization (one cycle of Gauss-Seidel) and the [1] indicates one iteration of the step 3 optimization per outer optimization.
- GLMPSAMIC where the [I] indicates incomplete step 2 optimization (one cycle of Gauss-Seidel) and the [C] indicates iteration to convergence for the step 3 optimization per outer optimization.
- GLMPSAMC1 where the first [C] indicates complete step 2 optimization (BFGS) and the [1] indicates one iteration of the step 3 optimization per outer iteration. Note that if we use (9), this is the pseudo-MLE.
- GLMPSAMCC where the first [C] indicates complete step 2 optimization (BFGS) and the second [C] indicates iteration to convergence for the step 3 optimization per outer iteration.

We set the within-step-3 and outer convergence (tested at step 4 for each iteration of m) for the fixed effects parameters $\boldsymbol{\beta}$, which was our primary interest. The convergence criteria that we used was

$$\max_i \left[\frac{\left|\beta_i^{[m+1]} - \beta_i^{[m]}\right|}{\left|\beta_i^m\right|} \right] < \delta, \tag{29}$$

where m indexes the iteration for the ith parameter of vector $\boldsymbol{\beta}$. We set $\delta = 1 \times 10^{-5}$.

4 Application of GLMPSAM algorithm

Our objective was to model a time series of binary values formed by sequences of haul-out times by harbor seals. We first describe the data, and then compare the various

GLMPSAM algorithms to the glmmPQL function in the MASS library in R, and proceed to use the best GLMPSAM algorithm to model the data in a real application.

4.1 Harbor seal haul-out data

All data came from 50 harbor seals studied in Cook Inlet, Alaska. Satellite data recorders (SDR) (model SPLASH, Wildlife Computers, Redmond, WA) were attached to all age and sex classes of the seals. SDRs were glued to the hair on the mid-dorsal region with 5-min epoxy and were programmed to record an electrical resistance value of the surrounding medium (water or air). For every hour, the tags recorded the portion of time dry. These dry-time percentages were used to determine haul-out status. Data were transmitted by satellite to Argos (CLS America, Inc., Largo, MD) and uploaded for analysis. The response variable was binary with a 1 indicating a seal was mostly hauled out during the hour. Binary series per seal ranged from 478 to 6,336 for a total of 151,269 observations. There was limited storage on the SDR and satellites were not always able to upload data before storage capacity was filled, so there were missing data.

Our interest centered on factors that affect temporal haul-out patterns. We included the following explanatory variables in the model: (1) hour as a categorical variable with 24 levels, (2) minutes from low tide as a categorical variable with 30 levels from 394 min prior to low to 394 min after low tide in 27 min intervals, (3) days from 15 August as a continuous variable, (4) year as a categorical variable for 2003, 2004, and 2005, (5) sex as a categorical variable, and (6) age as a categorical variable with levels pup, yearling, sub-adult, and adult. Each seal was considered a random effect from the population of all harbor seals.

4.2 Software issues

Generalized linear mixed pseudomodels have been implemented in PROC GLIMMIX in SAS (SAS Institute, Inc., Cary, NC, USA), and the function glmmPQL in the MASS package (Venables and Ripley 2002) of R (R Development Core Team 2007). The glmmPQL function uses pseudo-MLE on pseudo-data. PROC GLIMMIX can use either pseudo-MLE or pseudo-REML on the pseudo-data, and the default is pseudo-REML. We wrote our code entirely in R, and tested it by ensuring that it matched both glmmPQL and PROC GLIMMIX for the maximum likelihood case, and that it matched PROC GLIMMIX for the restricted maximum likelihood case. Both PROC GLIMMIX and glmmPQL were written for general purpose use with a wide variety of spatial, temporal, and other covariance structures, so we did not expect them to take advantage of the special structures described in this paper. To test the computing advantages of GLMPSAMI1, GLMPSAMIC, GLMPSAMC1, and GLMPSAMCC algorithms, we compared results and computing speed to the glmmPQL algorithm.

Computing speeds were obtained on an Intel Pentium4 CPU running at 3.8 GHz with 3 GB of RAM. We took the first 10 seals from the harbor seal data set that had at least 1,000 observations each. For computing and testing purposes, we only included the terms date and date^2 as continuous variables, and we created a categorical variable with two classes for haul-out status from midnight to noon (AM) and from noon to midnight (PM); thus there were four fixed effects. We began with 50 observations per seal, and fit a model with the four fixed effects and random effects for the 10 seals, along with an exponential temporal autocorrelation model as described in Sect. 3.2. We increased sample sizes per seal as shown in Fig. 1. In glmmPQL, we used the exponential autocorrelation model. Results from the GLMPSAM algorithms, when they converged, always matched glmmPQL with the exponential autocorrelation model for covariance parameter estimates, fixed effect estimates, and random effect estimates.

Results on computational speed are shown in Fig. 1, with the upper panel focusing on smaller sample sizes, and the lower panel focusing on larger sample sizes. For the smallest samples sizes in Fig. 1, glmmPQL was slightly faster than the GLMPSAM algorithms. However, it is clear that computing speeds for glmmPQL increase exponentially with sample sizes, which is consistent with the use of numeric inverses, while the GLMPSAM algorithms increases linearly, as the theory predicted. In general, the GLMPSAM algorithms are faster for sample sizes beyond 100 per seal. It is interesting, and somewhat surprising, to compare the GLMPSAM algorithms. It appears that incomplete step 2 optimization actually increases both speed and stability. Both

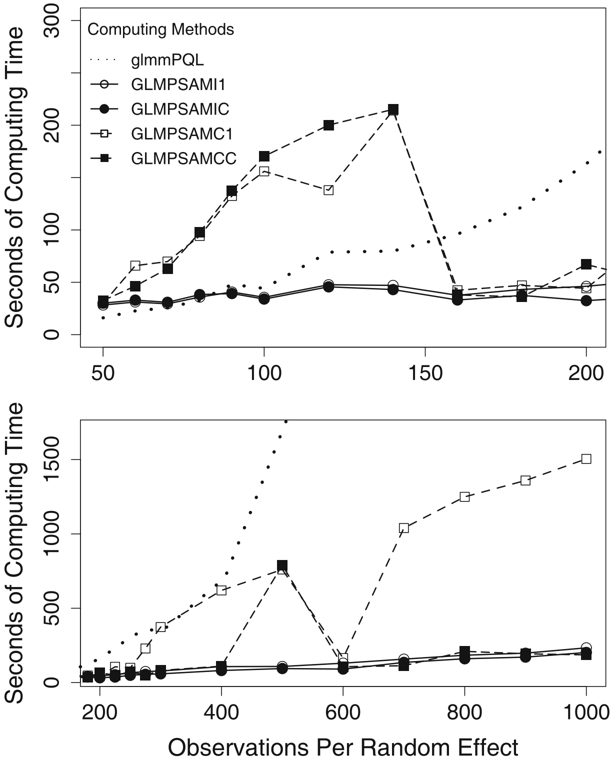

Fig. 1 Comparison of computing speeds for various algorithms proposed in this manuscript. The data used were subsets of the example data on harbor seals. The number of random effects was fixed at 10 (harbor seals), and the x-axis shows the number of repeated measurements per random effect

GLMPSAMC1 and GLMPSAMCC show erratic behavior in computing time, especially for smaller sample sizes. It appears that quite often complete step 2 optimization works to increase the number of iterations of the complete algorithm.

The behavior of the algorithms is investigated more fully in Table 1. In Table 1, the number of outer iterations for each of the GLMPSAM algorithms is shown in parentheses. Prior to the number in parentheses are the number of step 3 iterations per outer iteration. Because GLMPSAMI1 and GLMPSAMC1 had only one iteration for step 3, the number of step 3 iterations is not shown. Only the first 10 numbers of step 3 iterations are shown (if the total number of outer iterations exceeded 10); all further step 3 iterations were two or less. The maximum number of outer iterations was set at 101, so a value of (101) in Table 1 indicated that the algorithm did not converge. The results of Table 1 and Fig. 1 indicate that the number of outer iterations

Table 1 Iteration histories for four different GLMPSAM methods

n	GLMPSAMI1[a]	GLMPSAMIC[b]	GLMPSAMC1[c]	GLMPSAMCC[d]
50	(13)	5544332222...(13)	(23)	5543322222...(25)
60	(12)	5443322221...(12)	(60)	5443321211...(34)
70	(10)	5443322211...(10)	(57)	5433222212...(46)
80	(11)	5443322211...(11)	(67)	5443222222...(69)
90	(11)	5444322211...(10)	(93)	5443222222...(94)
100	(9)	55432211...(8)	(101)	5533333333...(101)
120	(10)	543222211...(9)	(65)	5432211222...(101)
140	(8)	4422211...(7)	(101)	4422221112...(101)
160	(6)	54311...(5)	(12)	5421111111...(11)
180	(6)	43211...(5)	(9)	532111...(6)
200	(6)	4321...(4)	(5)	4321111121...(14)
225	(6)	4321...(4)	(25)	4321211...(7)
250	(7)	43211...(5)	(21)	4321211111...(15)
275	(7)	42211...(5)	(51)	42111...(5)
300	(7)	43211...(5)	(74)	431111...(6)
400	(7)	53321...(5)	(101)	5322111111...(10)
500	(6)	52211...(5)	(101)	5222222222...(101)
600	(6)	4211...(4)	(9)	4211...(4)
700	(6)	42211...(5)	(101)	421111...(6)
800	(6)	52211...(5)	(101)	5211111111...(10)
900	(6)	52211...(5)	(101)	52111111...(8)
1000	(6)	52211...(5)	(101)	52111...(5)

The number of outer iterations is in parentheses. The number of step 3 iterations per outer iteration preceeds the number in parenthesis (up to 10 outer iterations)

n Number of repeated measures per random effect

[a] One step 2 Gauss-Seidel cycle, one step 3 iteration

[b] One step 2 Gauss-Seidel cycle, within-step-3 convergence

[c] Complete step 2 optimization, one step 3 iteration

[d] Complete step 2 optimization, within-step-3 convergence

largely determines computing speed. That is, step 3 iterations were very quick relative to step 2. It is also clear that incomplete optimization in step 2 generally increases stability and decreases the number of outer iterations, which is a somewhat surprising result. However, when using incomplete step 2 optimization, complete optimization of step 3 generally decreases the number of outer iterations (compare GLMPSAMIC to GLMPSAMI1 in Table 1) and hence decreases computing time. The results in Table 1 and Fig. 1 suggest that the GLMPSAMIC is the fastest and most stable algorithm. We also compared the GLMPSAM algorithms using restricted maximum likelihood and obtained very similar results, which are not shown.

4.3 Harbor seal haul-out example

We used restricted maximum likelihood on pseudo-data to fit GLM pseudo-models to all of the haul-out data. The initial model had all main effects and two-way interactions, and then we eliminated all two-way interactions that were not significant at $\alpha = 0.05$. After eliminating two-way interactions, we also eliminated main effects not contained in any two-way interactions. We adjusted the degrees of freedom for those factors that only change among animals (sex, age class, and year) and those that change by time within animal (time of day, tide height, and date) and use a standard F-test as an approximation (in the same way as SAS PROC GLIMMIX) for the null hypothesis that the effect is zero.

We modeled the binary haulout values as $Y_{i,j,k,l,m}(t) \sim \text{Bern}(\mu_{i,j,k,l,m}(t))$, where Bern() is a Bernoulli distribution, and the final GLM pseudo-model obtained was

$$\begin{aligned}\text{logit}(\mu_{i,j,k,l,m}(t)) = \alpha_i + \kappa_j + \eta_k + \tau_l + \beta_1 x + \beta_2 x^2 + (\alpha\beta)_{i,1} x + (\alpha\beta)_{i,2} x^2 \\ + (\kappa\beta)_{j,1} x + (\eta\beta)_{k,1} x + (\eta\beta)_{k,2} x^2 + R_m + Z_m(t), \qquad (30)\end{aligned}$$

where α_i is a fixed effect for the ith age class, κ_j is a fixed effect for the jth sex, η_k is a fixed effect for the kth hour of the day, τ_l is a fixed effect for the lth tide class, x is a continuous variable for date (note that we include a quadratic term because past experience, and exploratory data analysis, shows that haul-outs tend to decrease toward mid-winter before increasing again in the spring), R_m is a random effect for the mth seal [associated with the **G**-matrix in (4)], and $Z_m(t)$ is a temporally autocorrelated random effect for measurements at time t for the mth seal [associated with the **R**-matrix in (4)]. We assume that the temporal autocorrelation follows an exponential model given in (17), and that each seal's temporal haul-out pattern was an independent realization of the temporal process. That, and the logit link function in (30) and the variance/mean relation $a(\mu) = \mu(1 - \mu)$ for the Bernoulli distribution define the pseudo-model in (2). Using the GLMPSAMIC algorithm, computing time for 151,269 observations with 50 random effects and 111 fixed effects on an Intel Pentium4 CPU running at 3.8 GHz with 3 GB of RAM took 2 h and 45 min.

We do not present all parameter estimates, as there are 111 of them. Rather, we show the SAS Type III tests of hypotheses in Table 2. Note that there are some cautions when interpreting the SAS Type III tests (Fox 1997, Sect. 8.2.4).

Table 2 Type III hypothesis table for model given in (30) using the GLMPSAMIC algorithm for 151,269 haulout observations on 50 harbor seals

Effect	Num df	Den df	F value	Prob(F):H0
Date	1	151157	27.9	<0.0001
Date^2	1	151157	43.6	<0.0001
Hour	23	151157	4.3	<0.0001
Tide	29	151157	37.8	<0.0001
Sex	1	38	11.7	0.0015
Age	3	38	3.4	0.0260
Date:hour	23	151157	4.4	<0.0001
Date^2:hour	23	151157	4.3	<0.0001
Date:sex	1	38	13.7	0.0007
Date:age	3	38	4.4	0.0094
Date^2:age	3	38	3.7	0.0209

The table is similar to one provided in SAS PROC GLIMMIX

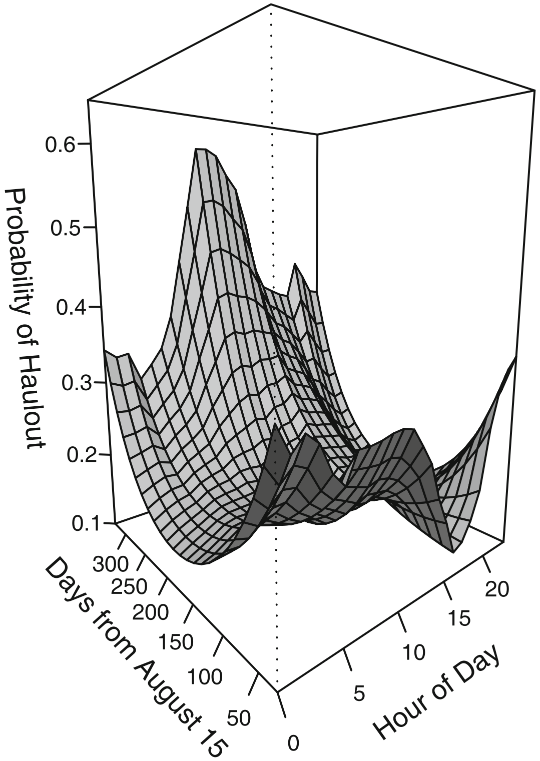

Fig. 2 Fitted haul-out probability model for date and hour-of-day for adult female harbor seals using a temporally autocorrelated logistic regression model with random effects and the GLMPSAMIC algorithm

This is the final model. Notice that there were no significant year effects, but with such large amounts of data, most other effects were significant. We were especially interested in the probability of hauling out throughout the day and year, and the

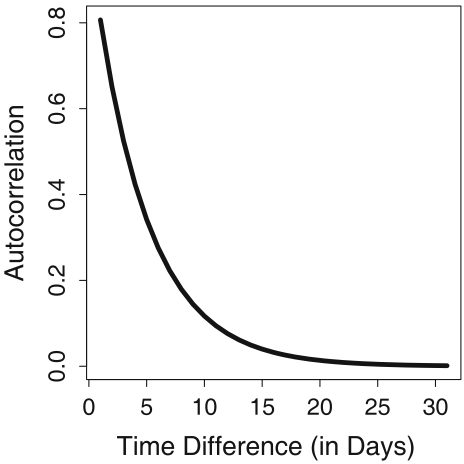

Fig. 3 Estimated temporal autocorrelation function for the harbor seal data set using the GLMPSAMIC algorithm

interaction of day and year. The significance of sex and age implied a separate fit of the date/time-of-day interaction for each sex/age interaction. We held the tide effect constant at low tide, which was the maximum probability of hauling out while holding all other factors constant. The fitted model for date and time of day for adult females is shown in Fig. 2, which shows that haul-out probabilities are highest in the summer and around mid-day, in general, but there are interesting interactions between the two effects. For the final fitted model, the estimated temporal autocorrelation function is shown in Fig. 3.

5 Discussion

In this article, we have developed the GLMPSAM algorithm for a Bernoulli distribution in the exponential family with an exponential autocorrelation model. The GLMPSAM algorithm will obviously work for all GLMM distributions from the exponential family of distributions; in fact, we only need to vary the link and variance functions to extend beyond the exponential family. Also, we can use any covariance matrix that has an analytical inverse; which includes all ARMA models (Zinde-Walsh 1988; Haddad 2004). As long as we can find analytical inverses, the GLMPSAM algorithm will yield linearly increasing computing times as opposed to O(n^3) computing times. Other algorithms, such as the Durbin and Levinson (Golub and Van Loan 1996, pg. 195–196) can offer O(n^2) computing time.

We also investigated variations in the GLMPSAM algorithm with regard to step 2 and step 3 optimization. The parameter estimation algorithm in Sect. 2.4 is somewhat unusual in the sense that there are two optimization steps for each outer iteration. While it would seem that the best approach would be full optimization for both steps 2 and 3, the results given here suggest that full optimization in step 2 actually decreases performance (increases computing times and has problems with convergence). That is, complete optimization in step 2 actually works against convergence in the outer iteration. Due to space limitations, we do not investigate this issue further. In fact, it may be possible that for Gaussian or Poisson distributions the GLMPSAMCC or GLMPSAMC1 may be more stable and faster than GLMPSAMIC. A full investigation of the speed and stability of the algorithms for various variance and link functions is an interesting topic for further research. R code used in this manuscript is available from the first author upon request.

Acknowledgments This project received financial support from the U.S. Minerals Management Service and the National Marine Fisheries Service, NOAA, under inter-agency agreement MMS 0103RU72448. We would like to thank all members of the field crews from the National Marine Mammal Laboratory and the Kachemak Bay Research Reserve. Any references to commercial products do not imply endorsements by the National Marine Fisheries Service.

References

Cressie N, Lahiri SN (1996) Asymptotics for REML estimation of spatial covariance parameters. J Stat Plann Inference 50:327–341

Fox J (1997) Applied regression analysis, linear models, and related methods. Sage, Thousand Oaks

Givens GH, Hoeting JA (2005) Computational statistics. Wiley, New York

Golub GH, Van Loan DF (1996) Matrix computations. Johns Hopkins University Press, Baltimore

Hamilton JD (1994) Time series analysis. Princeton University Press, Princeton

Haddad JN (2004) On the closed form of the covariance matrix and its inverse of the causal ARMA process. J Time Ser Anal 25:443–448

Henderson HV, Searle SR (1981) On deriving the inverse of a sum of matrices. SIAM Rev 23:53–60

Heyde CC (1994) A quasi-likelihood approach to the REML estimating equations. Stat Probab Lett 21: 381–384

Khuri AI (1993) Advanced calculus with applications in statistics. Wiley, New York

Mardia KV, Marshall RJ (1984) Maximum likelihood estimation of models for residual covariance in spatial regression. Biometrika 71:135–146

McCullagh P, Nelder JA (1989) Generalized linear models, 2nd edn. Chapman and Hall, London

Nelder JA, Mead R (1965) A simplex method for function minimization. Comput J 7:308–313

Patterson HD, Thompson R (1971) Recovery of interblock information when block sizes are unequal. Biometrika 58:545–554

Patterson HD, Thompson R (1974) Maximum likelihood estimation of components of variance. In: Proceedings of the eighth international biometric conference. Biometric Society, Washington, DC, pp 197–207

R Development Core Team (2007) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0. http://www.R-project.org

Sherman J, Morrison WJ (1949) Adjustment of an inverse matrix corresponding to changes in the elements of a given column or a given row of the original matrix. Ann Math Stat 20:621

Venables WN, Ripley BD (2002) Modern applied statistics with S, 4th edn. Springer, Berlin

Wolfinger R, O'Connell M (1993) Generalized linear mixed models: a pseudo-likelihood approach. J Stat Comput Simul 4:233–243

Woodbury MA (1950) Inverting modified matrices. Statistical Research Group, Memorandum Report 42. Princeton University, Princeton

Zinde-Walsh V (1988) Some exact formulae for autoregressive moving average processes. Econ Theory 4:384–402