Journal of Statistical Software

August 2018, Volume 86, Issue 9. doi: 10.18637/jss.v086.i09

logbin: An R Package for Relative Risk Regression Using the Log-Binomial Model

Mark W. Donoghoe
Stats Central

Ian C. Marschner
Macquarie University
NHMRC Clinical Trials Centre

Abstract

Relative risk regression using a log-link binomial generalized linear model (GLM) is an important tool for the analysis of binary outcomes. However, Fisher scoring, which is the standard method for fitting GLMs in statistical software, may have difficulties in converging to the maximum likelihood estimate due to implicit parameter constraints. **logbin** is an R package that implements several algorithms for fitting relative risk regression models, allowing stable maximum likelihood estimation while ensuring the required parameter constraints are obeyed. We describe the **logbin** package and examine its stability and speed for different computational algorithms. We also describe how the package may be used to include flexible semi-parametric terms in relative risk regression models.

Keywords: relative risk regression, log-binomial model, EM algorithm, R.

1. Introduction

Logistic regression is commonly used for the analysis of binary outcome data, and has good computational properties deriving from the fact that it uses the canonical link function for a binomial generalized linear model (GLM). The resulting effect measure is the adjusted odds ratio, which has attracted criticism because it is difficult to interpret and communicate accurately (Holcomb Jr, Chaiworapongsa, Luke, and Burgdorf 2001). Studies by Forrow, Taylor, and Arnold (1992) and Lacy *et al.* (2001), in the context of medical statistics, have shown that clinical decision-making can be heavily influenced by the choice of effect measure.

The odds ratio is the only estimable measure of effect in case-control designs, and approximates the relative risk (or prevalence ratio) if the event probability is low. However, the magnitude of the odds ratio always exceeds that of the relative risk, and misinterpretation can result in the exaggeration of effect sizes in prospective or cross-sectional studies of common events (Davies, Crombie, and Tavakoli 1998).

For these reasons, many authors have advocated the use of relative risk regression instead of logistic regression (e.g., Sackett, Deeks, and Altman 1996; Grimes and Schulz 2008). But computation of the maximum likelihood estimate (MLE) using standard software can be difficult, leading to the development of several methods that produce alternative estimates of adjusted relative risks (e.g., Schouten *et al.* 1993; Deddens, Petersen, and Lei 2003; Zou 2004). Although these methods generally have good properties, in a comparative study Marschner (2015) concluded that the MLE has some efficiency advantages, provided that it can be computed.

Over recent years there has been discussion and development of various approaches for maximum likelihood estimation which avoid the need to use approximate methods when the standard methods fail (Lumley, Kronmal, and Ma 2006; de Andrade and Carabin 2011; Marschner and Gillett 2012). The R (R Core Team 2018) package **logbin** (Donoghoe 2018) presented in this paper implements a number of different methods for computation of the MLE and is designed to address the calls of Lumley *et al.* (2006) and Marschner (2015) that such methods be made readily available in standard software.

In Section 2, we introduce the log-binomial model that is used for relative risk regression. In Section 3, we describe the **logbin** package and briefly outline the algorithms that underlie its main function, `logbin`. In Sections 4 and 5 we discuss the stability and speed of each approach, demonstrating each with simulated data. In Section 6 we highlight the potential effect of the choice of parameter space that is considered by different methods, and in Section 7 we describe and demonstrate an extension that allows the inclusion of semi-parametric components in the regression function through the `logbin.smooth` function.

2. The log-binomial model

The log-binomial regression model is a generalized linear model that uses a binomial outcome distribution and a log link function. That is, given n independent binomial outcomes $Y_i \sim \text{Bin}(N_i, p_i)$, we relate the event probability p_i to a linear combination of the covariate vector $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{iJ})$ and the parameter vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_J)$ via

$$\log p_i = \log p(\boldsymbol{x}_i; \boldsymbol{\theta}) = \sum_{j=1}^{J} \theta_j x_{ij}. \tag{1}$$

If $p(\boldsymbol{x}; \boldsymbol{\theta})$ is interpreted as the risk of an event for an individual with covariate vector $\boldsymbol{x}$, then the relative risk associated with covariate vector $\boldsymbol{x}_1$ compared to $\boldsymbol{x}_2$ is

$$RR(\boldsymbol{x}_1, \boldsymbol{x}_2; \boldsymbol{\theta}) = \frac{p(\boldsymbol{x}_1; \boldsymbol{\theta})}{p(\boldsymbol{x}_2; \boldsymbol{\theta})} = \exp\left(\sum_{j=1}^{J} \theta_j (x_{1j} - x_{2j})\right).$$

In particular, if we consider the case in which all of the components of $\boldsymbol{x}_1$ are $\boldsymbol{x}_2$ are identical except that $x_{1j} = x_{2j} + 1$ for some j, the exponentiated parameter $\exp(\theta_j)$ represents the relative risk associated with a one-unit increase in the jth covariate, keeping all others equal. Thus the log-binomial model is often referred to as a relative risk regression model.

Marschner (2015) has provided a thorough overview of estimation issues in relative risk regression. In particular, the standard method for maximum likelihood estimation, Fisher scoring, may encounter two distinct numerical problems. The first is associated with the fact that the

log link is not the canonical link function for the binomial GLM, meaning that the maximum likelihood estimate may be a repelling fixed point of the iterative algorithm. This issue can be addressed by a simple line search modification to the Fisher scoring algorithm, which has been implemented in R by the **glm2** package (Marschner 2011), and is used by both Stata's and SPSS's default GLM-fitting algorithms (StataCorp. 2015; IBM Corporation 2016).

The second problem arises because parameter constraints must be imposed so that the fitted event probabilities do not exceed 1. This does not have a simple solution in the Fisher scoring framework: although step-halving can be used to prevent intermediate estimates from moving outside the parameter space, Lumley *et al.* (2006) point out that the MLE may not be reached if the boundary of the parameter space is almost perpendicular to the gradient of the log-likelihood.

3. The logbin package

The **logbin** package provides an R interface to perform maximum likelihood estimation for log-binomial regression models, using different algorithms. The package is available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=logbin`. In Section 3.1 we describe the syntax for the main function `logbin`, giving an example for a particular model. In Section 3.2 we outline the various computational methods that are available in the package, and in Section 3.3 describe the object returned by the `logbin` function.

3.1. Input arguments and example

The main function in the **logbin** package is `logbin`, which has the following usage:

```
logbin(formula, mono = NULL, data, subset, na.action, start = NULL,
   offset, control = list(...), model = TRUE,
   method = c("cem", "em", "glm", "glm2", "ab"),
   accelerate = c("em", "squarem", "pem", "qn"),
   control.method = list(), warn = TRUE, ...)
```

The arguments are mostly identical to those used by the `glm` function, except that `family` does not need to be specified by the user. As with `glm` for a binomial model, `logbin` allows the response part of the `formula` to be either a binary vector or a two-column matrix with the columns giving the number of events and non-events.

The `method` argument selects the computational approach that will be used to find the maximum likelihood estimate. These are each described in Section 3.2. Argument `mono` allows the user to specify covariates that should be constrained to have non-negative coefficients, a feature that is restricted to the `"cem"` and `"em"` methods. Argument `accelerate` is also relevant only for `"cem"` and `"em"`, and potentially provides additional speed by using features of the **turboEM** package. More details are provided in Section 5.1.

Example data

The ASSENT-2 study (ASSENT-2 Investigators 1999) was a randomized clinical trial conducted in 16,949 patients who had experienced a recent heart attack. The primary outcome

of the study was 30-day mortality, and the comparison between the randomized treatments (tenecteplase and alteplase) showed a difference that was within the prespecified criteria for equivalence on both absolute and relative scales.

Four baseline characteristics of each patient were recorded, each in three categories: age (<60, 60–75, >75 years), heart attack severity (mild, moderate, severe), time from heart attack to treatment (<2, 2–4, >4 hours) and geographical region (Western countries, Latin America, Eastern Europe). The cross-tabulation of this data with the outcome is included in the **glm2** package as dataset `heart`.

We will demonstrate the use of the `logbin` function by estimating the adjusted relative risk of death associated with the four baseline characteristics. The general code we will use to fit the model is

```
R> model.heart <- logbin(cbind(Deaths, Patients - Deaths) ~
+    factor(AgeGroup) + factor(Severity) + factor(Delay) + factor(Region),
+    data = heart, start = rep(-0.2, 9))
```

Note that because we did not specify `method` in this call to `logbin`, the combinatorial EM algorithm (`method = "cem"`) will be employed by default.

3.2. Computational methods

The main purpose of the **logbin** package is to provide an implementation of the combinatorial EM (CEM) algorithm described by Marschner and Gillett (2012) for stable maximum likelihood estimation in log-binomial regression. However, the package also provides an interface to use modified Fisher scoring (via the function `glm`, or package **glm2**) and adaptive barrier (via `constrOptim`) algorithms to fit the same model. Full details of these algorithms are provided elsewhere, but we give a brief overview of each approach below.

EM-type algorithms

The default method, selected by specifying `method = "cem"` in the call to `logbin`, implements the combinatorial EM algorithm described by Marschner and Gillett (2012). This approach is based on the fact that, because of the product probability structure of the log-binomial model, each observed Y_i can be viewed as being derived from a collection of unobserved latent binary outcomes. An EM algorithm (Dempster, Laird, and Rubin 1977) based on this latent variable model can be used to maximize the log-likelihood $L(\boldsymbol{\theta})$.

However, due to constraints imposed by the latent variable model, the maximization is performed only over a subspace of the parameter space Θ. This restriction can be overcome by considering a family τ of reparameterizations, each $t \in \tau$ of which can be used to define an EM algorithm that maximizes over a subspace $\Theta(t) \subseteq \Theta$ such that

$$\bigcup_{t \in \tau} \Theta(t) = \Theta. \tag{2}$$

By implementing each EM algorithm, we obtain a collection of constrained MLEs $T = \{\hat{\boldsymbol{\theta}}(t); t \in \tau\}$, where

$$\hat{\boldsymbol{\theta}}(t) = \arg\max_{\boldsymbol{\theta} \in \Theta(t)} L(\boldsymbol{\theta}).$$

Due to the covering property in Equation 2, one of these must be the global MLE. So the combinatorial EM algorithm (Marschner 2014) consists of applying the family of EM algorithms, followed by a discrete optimization step to choose the constrained MLE that gives the greatest likelihood:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta} \in T} L(\boldsymbol{\theta}).$$

The number of parameter subspaces, and hence the number of EM algorithms that must be applied, is determined by the number of parameters in the model. Specifically, if there are A categorical covariates, where the ath has c_a levels, and B continuous covariates, the number of parameter subspaces defined by the CEM algorithm is

$$\prod_{a=1}^{A} c_a \times 2^B.$$

Because the number of parameter subspaces grows exponentially with the number of parameters, an exhaustive search of all of the parameter subspaces may be prohibitive for large models. Marschner (2014) and Donoghoe and Marschner (2016) have described some strategies that may be used to improve the computational efficiency of the CEM algorithm.

One strategy makes use of the fact that the log-likelihood of the log-binomial model is concave in $\boldsymbol{\theta}$, so that any stationary point must be a global maximum. `logbin` with `method = "cem"` implements this approach by stopping the search if the constrained maximum $\hat{\boldsymbol{\theta}}(t)$ is in the interior of its parameter subspace $\Theta(t)$. The interior is defined such that none of the parameters are within δ of the boundary, where δ is the argument `bound.tol` supplied to `logbin.control`. Care must be taken not to make δ so large that true interior points are thought to be on the boundary, or so small that true boundary points are numerically considered to be in the interior. The `start` argument can be used to specify the starting values of the parameter vector, which determines the parameter subspace that will be examined first.

Another strategy, implemented using `method = "em"`, is that of parameter expansion. This was outlined by Donoghoe and Marschner (2016), and amounts to adding an auxiliary parameter that allows each of the component latent variable models that make up the CEM algorithm to be nested within a single overparameterized latent variable model. In contrast to the CEM algorithm, this only requires a single application of an EM algorithm, which maximizes the observed-data log-likelihood over the expanded parameter space. Appropriate identifiability constraints can then be applied to recover the MLE in the original parameter space.

Because of the natural constraints imposed by the underlying EM algorithm, it is straightforward to impose non-negativity constraints on any parameters using these EM-based approaches. The `mono` argument to `logbin` can be used to identify these parameters by either name or index. For categorical covariates, `mono` will ensure that the level-specific parameters are non-decreasing in the order determined by the factor levels.

Modified Fisher scoring

Fisher scoring, accessed by using `method = "glm"`, is a general algorithm for maximum likelihood estimation. It is a Newton-type approach that uses the expected information matrix in approximating the gradient of the score function. Used for fitting GLMs, it can be implemented with an iteratively reweighted least squares (IRLS) algorithm (McCullagh and

Nelder 1989), which is the approach used by R's `glm.fit` function, SPSS's `GENLIN` command and SAS's `GENMOD` procedure (SAS Institute Inc. 2013). The default option in Stata's `glm` command implements the Fisher scoring algorithm directly, but an IRLS algorithm can be selected by specifying the `irls` option.

When the canonical link function is used in a GLM, Fisher scoring coincides exactly with Newton's algorithm, which has guaranteed quadratic local convergence (Lange 2013, p. 293). With non-canonical links, as in the log-binomial model, the fact that the expected information is always positive definite provides some stability by ensuring that the algorithm will always head in an ascent direction, but local convergence is not guaranteed without step-size modification (Lange 2013, p. 296).

The default GLM-fitting algorithms in SAS, SPSS and Stata each allow the option to perform Fisher scoring for a chosen number of iterations before switching to Newton's algorithm until convergence. Nevertheless, either algorithm can overshoot the MLE, and SPSS and Stata also implement step-halving if the likelihood decreases between iterations. This is the strategy used by the `glm.fit2` method provided by the **glm2** package in R (Marschner 2011, 2018), which can be accessed for the log-binomial model by specifying `method = "glm2"` in the call to `logbin`.

The log-binomial model also requires that fitted event probabilities do not exceed 1, but Newton-type algorithms do not naturally impose any parameter constraints. In R, the `glm` and `glm2` functions both employ step-halving to bring estimates back into the parameter space if the deviance becomes infinite or any fitted values are outside $(0, 1)$ at any iteration. Although undocumented, the `GENMOD` procedure in SAS addresses the issue in the same way.

The `glm` command in Stata does not perform any check of the validity of parameter estimates, and it is possible that it returns estimates that are outside the parameter space (Williamson, Eliasziw, and Fick 2013). By contrast, the `binreg` command implements the method of Wacholder (1986): If fitted values go outside $(0, 1)$ at any iteration, they are replaced by a value inside the range, rather than changing the parameter estimates (Hardin and Cleves 1999). Thus this approach can also return parameter estimates that produce invalid fitted probabilities.

Adaptive barrier

The adaptive barrier approach (Lange 1994), chosen by specifying `method = "ab"`, is a general method for convex optimization with constraints. The constrained problem is converted into an unconstrained problem by adding a logarithmic barrier term, which is infinite when the constraints are active, and is updated at each iteration. Within each iteration, the optimization can be performed using any method that allows infinite values for the objective function.

This approach is well-suited to maximum likelihood estimation of the log-binomial model because the negative log-likelihood is convex, as are the required parameter constraints:

$$\sum_{j=1}^{J} \theta_j x_{ij} \leq 0 \quad \text{for all } \boldsymbol{x}_i.$$

Neither SPSS or Stata currently include an inbuilt optimization routine that allows inequality constraints. The `NPL` procedure in SAS provides a wide range of constrained optimization

	cem	em	glm	glm2	ab
(Intercept)	−4.027	−4.027	−4.000	−4.027	−4.027
factor(AgeGroup)2	1.104	1.104	1.106	1.104	1.104
factor(AgeGroup)3	1.927	1.927	1.900	1.927	1.927
factor(Severity)2	0.703	0.703	0.711	0.703	0.703
factor(Severity)3	1.377	1.377	1.260	1.377	1.377
factor(Delay)2	0.059	0.059	0.065	0.059	0.059
factor(Delay)3	0.172	0.172	0.131	0.172	0.172
factor(Region)2	0.076	0.076	0.071	0.076	0.076
factor(Region)3	0.483	0.483	0.270	0.483	0.483
Deviance	149.32	149.32	162.99	149.32	149.32
Iterations	446272	6526	10000	14	214
Converged	TRUE	TRUE	FALSE	TRUE	TRUE
Boundary	FALSE	FALSE	FALSE	FALSE	FALSE

Table 1: Parameter estimates, deviance and convergence status of the fitted model to the heart attack data, using the various computational methods available with `logbin`.

methods, and has been used by Yu and Wang (2008) to fit a log-binomial regression model. In R, `logbin` calls upon the `constrOptim` function, an adaptive barrier algorithm that allows affine inequality constraints.

Because the gradient of the log-likelihood is known, the default `optim` method used for the inner iterations is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) quasi-Newton algorithm. An alternative algorithm can be chosen, or any of the other optional arguments to `constrOptim` (which are passed to `optim`) can be set, by using the `control.method` argument of `logbin`.

3.3. Output and reporting tools

Having already used the default CEM approach at the end of Section 3.1, we can fit the model to the heart attack data with each of the other computational approaches using

```
R> model.heart.em <- update(model.heart, method = "em")
R> model.heart.glm <- update(model.heart, method = "glm")
R> model.heart.glm2 <- update(model.heart, method = "glm2")
R> model.heart.ab <- update(model.heart, method = "ab")
```

The MLE for this model is in the interior of the parameter space, and Table 1 shows the parameter estimates, deviance and convergence status of the fitted model using each approach. Both EM-type algorithms, the modified Fisher scoring of `glm2` and the adaptive barrier approach all successfully converged in this case. As noted by Marschner and Gillett (2012), without the implementation of step-halving to ensure that the sequence of estimates has non-decreasing likelihood, the Fisher scoring algorithm used by `glm` enters a cycle of period 8 and never converges.

The object returned by `logbin` has class `c("logbin", "glm", "lm")`. The **logbin** package includes S3 methods so that the usual methods for '`glm`' objects are also available for those returned by `logbin`. In particular, `summary` and `print.summary` return the usual output as well as reporting the the small-sample corrected AIC_c (Burnham and Anderson 2002), and,

for `method = "cem"`, the number of EM iterations performed in the parameter subspace that contained the MLE.

When the MLE is on the boundary of the parameter space, the assumptions that underlie the usual method of estimating the covariance matrix by the inverse of the information matrix may not be valid. Unlike the `summary` method for '`glm`' objects, in such a situation the `summary` method for '`logbin`' objects returns a matrix of `NA`s along with an appropriate warning. This also affects the results from `confint` and `vcov` applied to a '`logbin`' object.

The `anova`, `confint`, `predict` and `vcov` functions have all been modified so that they work well with '`logbin`' objects. Other useful methods for '`glm`' objects, such as `extractAIC`, `residuals` and `simulate`, will also perform as expected when used with a '`logbin`' object.

4. Stability

In the previous section we saw an example of convergence problems that may occur when Fisher scoring is used for relative risk regression. In this section we consider this issue in more detail. The issue of numerical instability in relative risk regression has been discussed in detail by Marschner (2015). Many of the potential problems associated with Fisher scoring for non-canonical links can be avoided by using step-halving to force the deviance to decrease at each iteration, as implemented by **glm2**. However, this is not a guaranteed solution to the problem, particularly if the MLE is on or near the boundary of the parameter space. We now consider a numerical illustration of the problem within the Fisher scoring framework, and then investigate how other methods, available within **logbin**, can be used to overcome it.

4.1. Numerical illustration

As illustrated in the previous section, one type of convergence problem is a non-convergent iterative sequence. Another type of problem, which we illustrate in this section, is convergence to a suboptimal value. We illustrate the problem using a simple simulated dataset with 100 binary observations Y_i and a single continuous covariate $x_i \in [0, 1]$, $i = 1, \ldots, 100$, fitting a two-parameter log-binomial model $\log p(x_i; \boldsymbol{\theta}) = \theta_1 + \theta_2 x_i$. The MLE in this data is $\hat{\boldsymbol{\theta}} = (-0.00, -1.31)$, which is on the boundary of the parameter space since $p(0; \hat{\boldsymbol{\theta}}) = 1$.

Figure 1(a) shows the path taken by `glm` from a starting estimate of $(-1, -1)$. The lighter lines with red crosses show the first few occasions on which the Fisher scoring algorithm produced estimates outside the parameter space, and step-halving was invoked to find a valid estimate along the direction of the step. This occurred at every iteration, and although the Fisher scoring step approaches a direction that is parallel to the boundary, this means that eventually the steps along the boundary become so small that convergence is declared at a suboptimal estimate. In this case, the failure to reach the MLE is not solved by `glm2`, which follows the same path, and occurs from a large number of different starting estimates, and even if the convergence criterion is made stricter.

Figures 1(b) and (c) show the path taken by the EM-type algorithms from the same starting estimate. Both methods converge stably to the MLE. They appear to have similar local convergence, but the parameter-expanded version converged faster overall because it followed a more direct route in the first few iterations. The main advantage of the parameter-expanded version is not evident here: Had we used a starting estimate with positive θ_2, our combinatorial EM algorithm would have converged to the constrained MLE with $\hat{\theta}_2 \geq 0$, and we would need

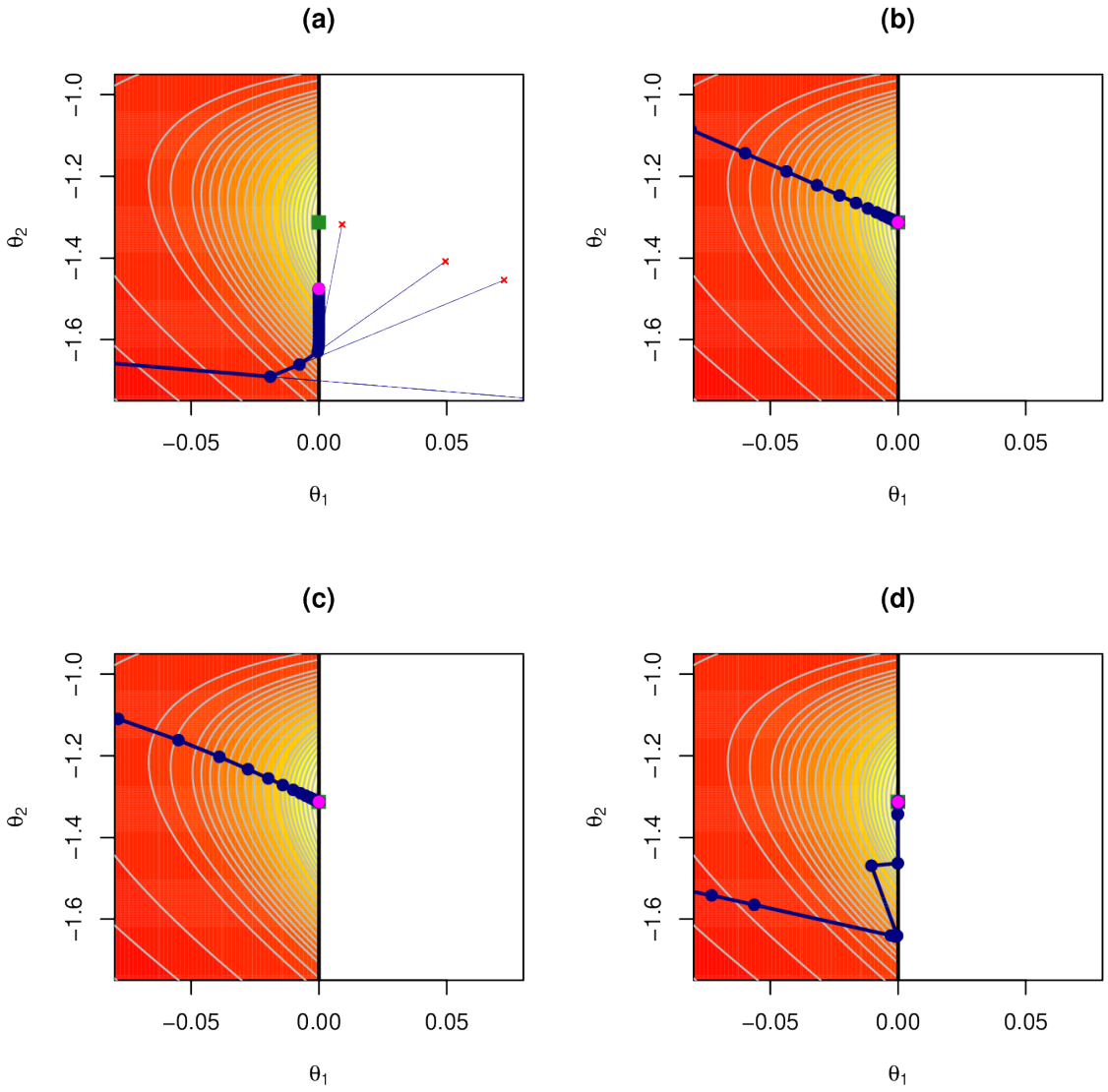

Figure 1: The path taken by (a) modified Fisher scoring, (b) combinatorial EM, (c) parameter-expanded EM, and (d) adaptive barrier algorithms in the simulated data. The MLE is indicated by a green square, with the final estimate from each method marked in magenta.

to use the algorithm again in the other parameter subspace to find the global MLE. The parameter-expanded version, on the other hand, requires just a single implementation from any starting estimate.

Finally, Figure 1(d) shows the trace of the adaptive barrier algorithm using the default options provided by the `logbin` function. This algorithm follows a similar path as Fisher scoring in approaching the boundary, but instead of seeking to move outside the parameter space, we can see the effect of the barrier term in forcing the estimates towards the interior, and the MLE is successfully found.

4.2. Simulations

In order to further explore the stability of the computational methods, we used each method to fit the model on 1000 simulated samples generated by taking parametric bootstrap replications from the simulated data discussed in the previous section.

The combinatorial EM, parameter-expanded EM and adaptive barrier algorithms all converged for every sample, although the EM-type algorithms required a large number of iterations to satisfy the convergence criterion in some cases. In one sample, the sequence of estimates from `glm` showed chaotic behavior and did not converge, but `glm2` avoided this problem. Both methods reported convergence to the same estimate in every other sample.

Despite the modified Fisher scoring and adaptive barrier approaches reporting convergence, the resulting estimate was sometimes suboptimal, similar to what was observed in the original sample. The estimate returned by the EM-type algorithms had a greater likelihood than the estimates from the other approaches in every sample. In particular, the likelihood ratio of the EM estimate compared to that obtained using `glm2` exceeded 1.05 in more than 10% of cases, leading to an average error of -0.019 in the estimate of the θ_2 parameter from `glm2`. The adaptive barrier approach generally found numerically the same estimate as the EM algorithms, although there were four cases in which the estimate of the θ_2 parameter was more than 0.01 away from the MLE with smaller likelihood.

5. Speed

In unconstrained problems, Newton's algorithm has quadratic local convergence (Lange 2013, p. 294). In practice, Fisher scoring will usually perform similarly, particularly for large sample sizes when the expected information approximates the observed information matrix well. This is reflected in the fact that the default maximum number of iterations used by `glm.fit` is 25.

The algorithms underlying the adaptive barrier approach also generally have fast convergence in the unconstrained case. For example, quasi-Newton methods such as BFGS have superlinear local convergence (Broyden, Dennis Jr, and Moré 1973).

In contrast, the EM algorithm typically takes a large number of iterations to converge. Its local convergence is linear, inversely related to the proportion of missing information. Because the EM-type algorithms are the main novel method included in the **logbin** package, we discuss an approach for improving its speed without compromising its stability.

5.1. turboEM algorithms

The **turboEM** package (Bobb and Varadhan 2018) provides a general interface to several algorithms that aim to accelerate the convergence of EM algorithms. In **logbin**, the `accelerate` argument of the `logbin` function can be used to specify one of these algorithms (`"squarem"`, `"pem"` or `"qn"`) to speed up either the individual components of the combinatorial EM algorithm or the single parameter-expanded EM algorithm described in Section 3.2.

We have observed that the default parameters generally give good behavior when using each of these acceleration schemes, but they can be altered by the user via the `control.method` argument to the `logbin` function; we refer the reader to the **turboEM** package documentation for specific details. The sections below briefly outline the acceleration algorithms, given an EM mapping $\hat{\boldsymbol{\theta}}_{c+1} = F(\hat{\boldsymbol{\theta}}_c)$.

SQUAREM

The SQUAREM algorithm (Varadhan and Roland 2008) is a globally convergent iteration scheme for accelerating an EM algorithm. Its basis is a vector generalization of Steffensen's method for scalar fixed-point problems, which Varadhan and Roland call the STEM scheme:

$$\hat{\boldsymbol{\theta}}_{c+1} = \hat{\boldsymbol{\theta}}_c - \alpha_c(F(\hat{\boldsymbol{\theta}}_c) - \hat{\boldsymbol{\theta}}_c),$$

where α_c is a step length that minimizes some measure of discrepancy between the zeros resulting from two different linear approximations of the residual function $r(\boldsymbol{\theta}) = F(\boldsymbol{\theta}) - \boldsymbol{\theta}$.

Analogous to the way in which the Cauchy-Barzilai-Borwein method (Raydan and Svaiter 2002) improves upon the Cauchy method for a linear problem, the SQUAREM algorithm is defined by forcing its recursive error relation to have the same (squared) form when compared to the STEM algorithm. This gives

$$\hat{\boldsymbol{\theta}}_{c+1} = \hat{\boldsymbol{\theta}}_c + 2\alpha_c r(\hat{\boldsymbol{\theta}}_c) + \alpha_c^2 v_c,$$

where $v_c = r(F(\hat{\boldsymbol{\theta}}_c)) - r(\hat{\boldsymbol{\theta}}_c) = F(F(\hat{\boldsymbol{\theta}}_c)) - 2F(\hat{\boldsymbol{\theta}}_c) + \hat{\boldsymbol{\theta}}_c$.

Given the current estimate $\hat{\boldsymbol{\theta}}_c$, the update requires two applications of the EM step. The SQUAREM scheme can be made globally convergent by using a "back-tracking" strategy to modify the step length α_c so that each iteration is guaranteed to increase the observed-data log-likelihood.

Parabolic EM

Parabolic extrapolation of the EM algorithm (Berlinet and Roland 2009) was designed to avoid the problem of "stagnation", in which the parameter estimates move quickly to a neighborhood of the maximum, but final convergence to a stationary point is very slow. Given three past values $\hat{\boldsymbol{\theta}}_{c-2}$, $\hat{\boldsymbol{\theta}}_{c-1}$, $\hat{\boldsymbol{\theta}}_c$, we consider a Bézier curve $M(t)$ controlled by these points.

A grid search along this curve is performed for different $t \in \mathbb{R}$, as long as $L(M(t))$ is increasing. We retain the point $M(\hat{t})$ that maximizes the likelihood, and use this to choose a new set of three points with which to define the next Bézier curve. This approach requires two applications of the EM step at each iteration to choose the next control points, as well as potentially multiple evaluations of the likelihood in performing the grid search. It preserves the property of increasing the likelihood at each iteration.

Quasi-Newton

Zhou, Alexander, and Lange (2011) described an acceleration scheme based on Newton's method for finding a root of the residual function $r(\boldsymbol{\theta})$:

$$\hat{\boldsymbol{\theta}}_{c+1} = \hat{\boldsymbol{\theta}}_c - \left[I - \mathrm{d}F(\hat{\boldsymbol{\theta}}_c)\right]^{-1} r(\hat{\boldsymbol{\theta}}_c).$$

The quasi-Newton approach replaces $\mathrm{d}F$ in the above equation with a low-rank matrix M, which is calculated using q secant approximations derived from previous iterations. Thus the method requires q initial EM updates, at which point quasi-Newton updating can commence based on the $q - 1$ secant pairs.

Although this algorithm can violate the likelihood ascent property of the EM algorithm, if the quasi-Newton update fails to increase the likelihood at any iteration we can revert to the usual EM update from our current estimate.

Number of covariates	5		10		15	
Relative risk	0.8	1.0	0.8	1.0	0.8	1.0
`glm2`	0.01	0.01	0.01	0.01	0.01	0.02
`ab`	0.01	0.02	0.02	0.03	0.04	0.04
`em`	0.08	0.07	0.53	0.25	4.32	2.26
`em` + `squarem`	0.03	0.03	0.08	0.06	0.31	0.29
`em` + `pem`	0.05	0.04	0.14	0.09	0.51	0.33
`em` + `qn`	0.03	0.03	0.09	0.06	0.48	0.52

Table 2: Average time in seconds for each algorithm to converge to the MLE of a log-binomial model in simulated data.

5.2. Simulations

We compared the speed of the different computational approaches using a simulation study similar to that described by Marschner (2014). In short, we considered a simple log-binomial model as in Equation 1 with an intercept and $J - 1 = 5$, 10 or 15 binary covariates. The baseline risk was fixed at $\exp(\theta_1) = 0.6$ and the relative risk for each covariate was $\exp(\theta_j) = 0.8$ or 1, $j = 2, \ldots, J$.

In each scenario, we produced 1000 datasets with sample size $n = 500$, and used the `logbin` function to find the MLE via modified Fisher scoring (`method = "glm2"`), adaptive barrier (`method = "ab"`) and various methods based on the EM algorithm. Although in general the combinatorial EM algorithm (`method = "cem"`) converged to the MLE in a reasonable amount of time for a single dataset, the total time across all simulations became prohibitive, particularly as the number of parameter subspaces to be searched (2^{J-1}) grew. For this reason, we examined only the parameter-expanded version using the `method = "em"` option. We also applied each of the **turboEM** acceleration schemes described in Section 5.1 to the parameter-expanded EM algorithm.

Table 2 shows the average time taken for each algorithm to converge to the MLE for each scenario, using a computer with a 3.40 GHz processor. While Fisher scoring was consistently very fast, even as the number of covariates increased, the adaptive barrier algorithm was only marginally slower in all scenarios. Unsurprisingly, the EM approach was substantially slower, with the deficit increasing with the number of covariates. However, the acceleration schemes provided a considerable improvement, with the SQUAREM algorithm proving to be the best.

6. Parameter space

The parameter space over which the log-binomial likelihood must be maximized depends on the covariate space for which we require the fitted probabilities to lie in $[0, 1]$. That is, the parameter space is

$$\Theta_{\mathcal{X}} = \{\boldsymbol{\theta} : 0 \leq p(\boldsymbol{x}; \boldsymbol{\theta}) \leq 1, \boldsymbol{x} \in \mathcal{X}\}$$

for some $\mathcal{X}$. As discussed by McCullagh (2005) in the context of the proportional odds model, at a minimum this $\mathcal{X}$ must include the observed covariate vectors, but it could be much larger.

In the **logbin** package, the Fisher scoring and adaptive barrier methods both ensure risks are valid for $\mathcal{X}_1 = \bigcup_{i=1}^{n} \boldsymbol{x}_i$, that is, the observed covariate vectors. This constraint is applied by R's `glm` and `glm2` functions, as well as the `GENMOD` procedure in SAS and `binreg` in Stata.

In fact, because $\Theta_{\mathcal{X}}$ corresponds to non-positivity constraints on a linear combination of $\boldsymbol{x} \in \mathcal{X}$, it can be shown that $\Theta_{\mathcal{X}} = \Theta_{\mathcal{X}^*}$, where $\mathcal{X}^* = \text{Conv}(\mathcal{X})$, the convex hull of $\mathcal{X}$. So methods that maximize over $\Theta_{\mathcal{X}_1}$ will ensure that fitted risks are valid for covariate vectors that lie within the convex hull of the set of observed covariate vectors, but could produce probabilities greater than 1 for a covariate vector $\boldsymbol{x} \not\subseteq \mathcal{X}_1^*$, even if the individual components of $\boldsymbol{x}$ are each within their observed ranges.

Figure 2 shows two simple scenarios in which this could have an impact. In (a), we consider a model with an intercept and two continuous covariates $(x_1, x_2) \in [0, 1]^2$:

$$\log p(\boldsymbol{x}; \boldsymbol{\theta}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2.$$

The black circles mark 10 observed covariate vectors, and the shaded area shows the convex hull $\mathcal{X}_1^*$. Methods that maximize the likelihood over $\Theta_{\mathcal{X}_1}$ will ensure that the fitted probability will be within $[0, 1]$ for every point in the shaded area. One such valid parameter vector is $\boldsymbol{\theta}^* = (-0.25, -0.5, 0.5)$, which gives $p(\boldsymbol{x}^*; \boldsymbol{\theta}^*) \leq \exp(-0.05) = 0.951$ for all $\boldsymbol{x}^* \in \mathcal{X}_1^*$, taking its maximum value along the line $x_2^* - x_1^* = 0.4$. However, the point $\boldsymbol{x}' = (0.2, 0.8)$, marked by a red square, gives a fitted risk of $p(\boldsymbol{x}'; \boldsymbol{\theta}^*) = \exp(0.05) = 1.051$.

In Figure 2(b), we consider a model with one categorical covariate and one continuous covariate:

$$\log p(\boldsymbol{x}; \boldsymbol{\theta}) = \theta_{x_1} + \theta_3 x_2,$$

where $x_1 \in \{1, 2\}$, and the black circles denote observed values. The parameter space $\Theta_{\mathcal{X}_1}$ ensures that fitted probabilities will be valid for x_2 values along the grey lines, where the range depends on the level of x_1. In such a situation, the interpretation of the exponentiated categorical parameters as relative risks is only relevant at certain levels of x_2. For example, the parameter vector $\boldsymbol{\theta}^* = (-0.5, -0.3, 0.45)$ is within $\Theta_{\mathcal{X}_1}$ and we would usually interpret $\exp(\theta_2^* - \theta_1^*) = 1.221$ as the relative risk associated with a change from $x_1 = 1$ to $x_1 = 2$ while keeping x_2 constant. However, the fitted risk at $(1, 0.8)$ is $\exp(-0.14) = 0.869$, and applying a relative risk greater than $1/0.869 = 1.15$ will give a risk greater than 1.

An alternative choice for $\mathcal{X}$ is the Cartesian product of the observed ranges of each covariate. That is,

$$\mathcal{X}_2 = \prod_{a=1}^{A} \{1, \ldots, c_a\} \times \prod_{b=A+1}^{A+B} [v_b^{(0)}, v_b^{(1)}],$$

where $v_b^{(0)} = \min_i x_{ib}$ and $v_b^{(1)} = \max_i x_{ib}$. The corresponding parameter space $\Theta_{\mathcal{X}_2}$ is maximized over by the EM-type algorithms provided by the **logbin** package with `method = "em"` or `"cem"`.

In Figure 2 $\mathcal{X}_2$ is surrounded by a black dashed line, and we can see that the unobserved covariate vectors discussed previously (marked by red squares) are inside this covariate space. Thus the parameter vector that produced invalid fitted probabilities for these covariate vectors will not be a member of $\Theta_{\mathcal{X}_2}$.

Since $\mathcal{X}_1^* \subseteq \mathcal{X}_2$, then $\Theta_{\mathcal{X}_2} \subseteq \Theta_{\mathcal{X}_1^*}$. This means that

$$\max_{\boldsymbol{\theta} \in \Theta_{\mathcal{X}_2}} L(\boldsymbol{\theta}) \leq \max_{\boldsymbol{\theta} \in \Theta_{\mathcal{X}_1^*}} L(\boldsymbol{\theta}),$$

that is, the maximum likelihood estimate $\hat{\boldsymbol{\theta}}^{(2)}$ in $\Theta_{\mathcal{X}_2}$ cannot have a higher likelihood than the maximum likelihood estimate $\hat{\boldsymbol{\theta}}^{(1)}$ in $\Theta_{\mathcal{X}_1^*}$. Clearly, if $\Theta_{\mathcal{X}_1^*} = \Theta_{\mathcal{X}_2}$ then the MLEs coincide,

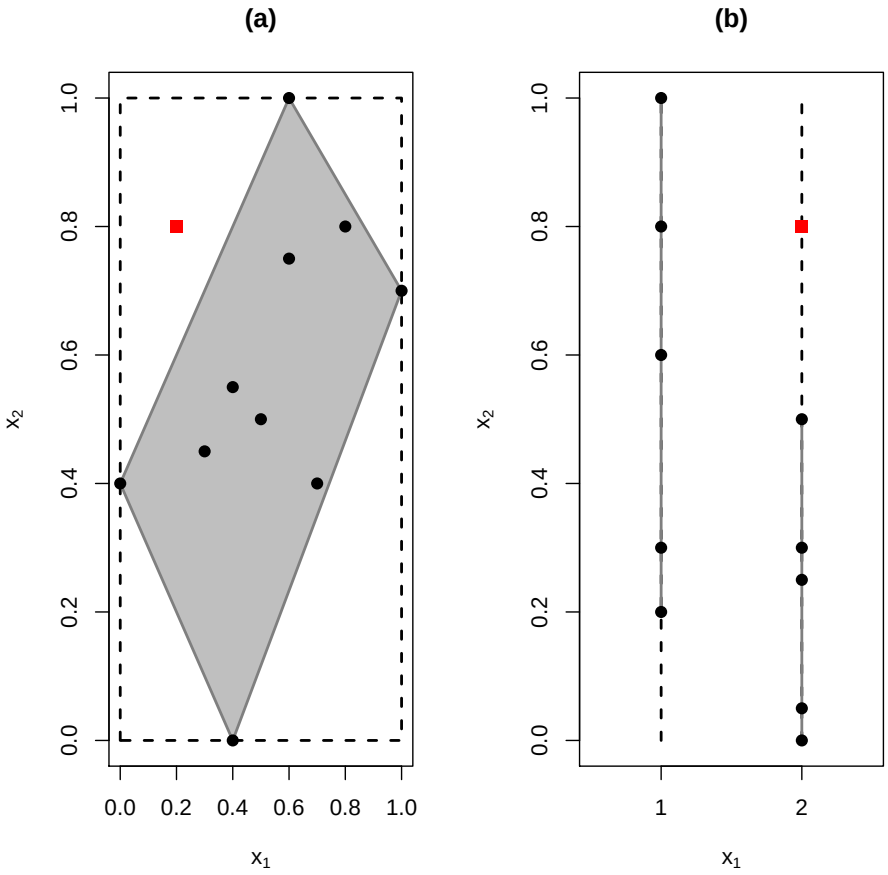

Figure 2: Examples of the covariate space considered by the Fisher scoring and adaptive barrier methods (grey) compared to that considered by the EM approaches (black, dashed) for models with (a) two continuous covariates, and (b) one categorical and one continuous covariate. Observed covariate values are marked as black circles.

that is $\hat{\boldsymbol{\theta}}^{(1)} = \hat{\boldsymbol{\theta}}^{(2)}$. For models that only contain categorical covariates, this will occur if we have at least one observation in each cell of the categorical cross-tabulation.

In general, due to the concavity of the log-likelihood function, we know that if $\hat{\boldsymbol{\theta}}^{(2)}$ is in the interior of $\Theta_{\mathcal{X}_2}$, then $\hat{\boldsymbol{\theta}}^{(1)} = \hat{\boldsymbol{\theta}}^{(2)}$. If $\hat{\boldsymbol{\theta}}^{(2)}$ is on the boundary of its parameter space $\Theta_{\mathcal{X}_2}$, it is possible that $\hat{\boldsymbol{\theta}}^{(1)} \neq \hat{\boldsymbol{\theta}}^{(2)}$, in which case the fitted risks under a model with $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}^{(1)}$ will be invalid for some $\boldsymbol{x} \in \mathcal{X}_2 \setminus \mathcal{X}_1^*$.

Either parameter space could be more appropriate in particular scenarios. In situations with small sample sizes where the covariates are uncorrelated, it may be more reasonable to consider the parameter space that is based on $\mathcal{X}_2$, in order to avoid estimates that give invalid risks for covariate patterns that were not observed simply by chance. On the other hand, if the covariates are strongly related to each other, it would not make sense to impose parameter constraints that ensure valid fitted probabilities for implausible covariate combinations.

An extreme example of this occurs if the covariates are structurally dependent on one another, such as in the case where they are separate indicator variables for the levels of a single categorical covariate. If entered this way into the model (rather than as a single `factor`), the

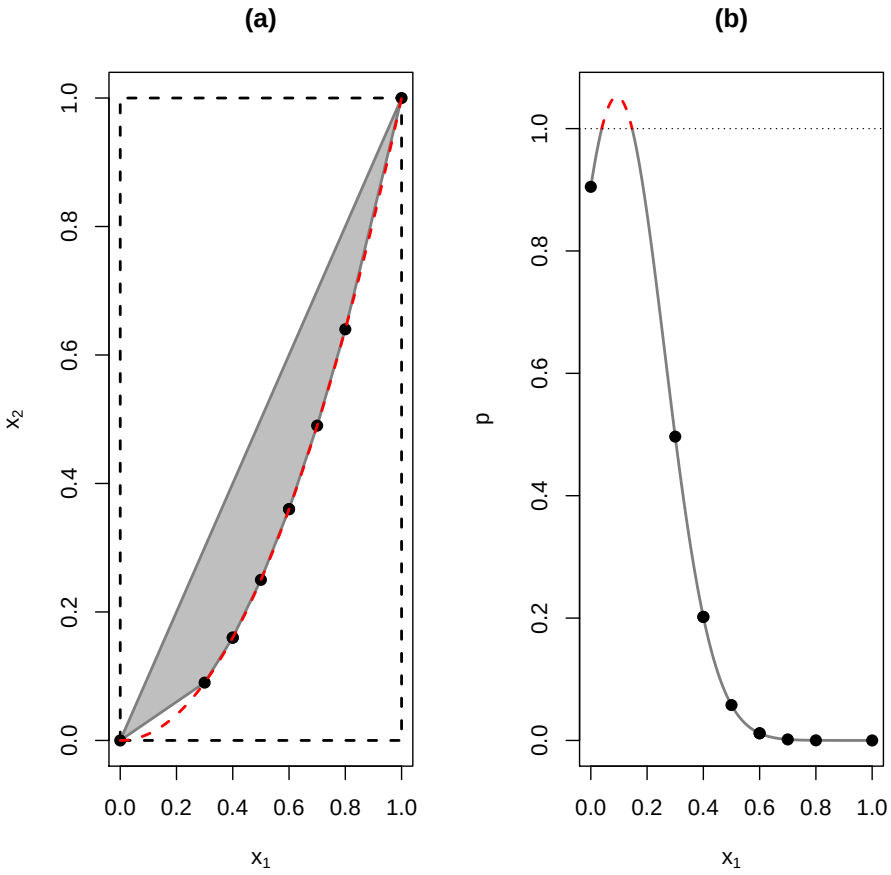

Figure 3: Examples of the (a) covariate space considered by the Fisher scoring and adaptive barrier methods (grey) compared to that considered by the EM approaches (black, dashed) for a model with a quadratic term, and (b) a hypothetical curve that produces invalid fitted risks. Observed covariate values are marked as black circles.

EM-type algorithms will consider them to be separate continuous covariates each with a range of $[0, 1]$, and $\mathcal{X}_2$ will include a covariate vector in which more than one of these covariates is non-zero, overly constraining the parameter space.

Another such example occurs when we wish to model a quadratic relationship between a covariate and the log-probability of an event. In a model with a single covariate x_1, the typical way to achieve this is to define $x_2 = x_1^2$ and include both in the linear predictor:

$$\log p(\boldsymbol{x}; \boldsymbol{\theta}) = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2.$$

However, neither choice of parameter space is ideal in this case. Figure 3(a) shows each covariate space for a hypothetical set of covariate vectors. $\mathcal{X}_2$, surrounded by a dashed black line, is most clearly inadequate here: it includes covariate pairs such as $(1, 0)$ that do not lie on the quadratic curve. The resulting parameter space $\Theta_{\mathcal{X}_2}$ would overly constrain our parameter vector to ensure that fitted risks are valid for these impossible combinations.

The parameter space $\Theta_{\mathcal{X}_1^*}$ is a vast improvement, but may still be problematic. Because the curve $x_2 = x_1^2$ is itself convex, the convex hull $\mathcal{X}_1^*$ of the observed covariate vectors – shown by the shaded area – is bounded by straight lines that lie wholly within the curve.

This means that the covariate vector for *any* unobserved x_1 value lies outside $\mathcal{X}_1^*$, and so $\Theta_{\mathcal{X}_1^*}$ will contain parameter vectors that produce fitted risks within $[0, 1]$ at the observed x_1 but can have invalid fitted risks at intermediate values. An example of such a curve, with $\boldsymbol{\theta}^* = (-0.1, 3.25, -17.5)$ is shown in Figure 3(b), which is in $\Theta_{\mathcal{X}_1^*}$ since $p(x_1; \boldsymbol{\theta}^*) \leq 1$ for all observed x_1, but gives $p(x_1; \boldsymbol{\theta}^*) > 1$ for $x_1 \in (0.04, 0.15)$.

The purpose of this discussion is to highlight the point made by McCullagh (2005) that alternative parameter spaces may be of interest, beyond the space $\Theta_{\mathcal{X}_1}$ implemented in standard software. Through its `"em"` and `"cem"` methods, **logbin** provides the facility to impose alternative parameter space restrictions, while also allowing standard parameter space assumptions using its other methods. This flexibility is a useful feature of the package, and to our knowledge is not available in any other package.

7. Semi-parametric extensions

As described by Donoghoe and Marschner (2015), it is straightforward to extend the EM-type algorithms for log-binomial regression to include J' flexible semi-parametric components via B-splines. Specifically, the model in Equation 1 becomes

$$\log p_i = \log p(\boldsymbol{x}_i; \boldsymbol{\theta}) = \sum_{j=1}^{J} \theta_j x_{ij} + \sum_{j=J+1}^{J+J'} f_j(x_{ij}),$$

and each unknown f_j is parameterized by constraining it to belong to the space of B-splines:

$$f_j(x) = \sum_{k=1}^{K_j} \theta_{j,k} B_{j,k}(x),$$

where B are the B-spline basis functions defined by a set of knots that determine the continuity of the curve at chosen turning points (Ramsay 1988). These can be calculated recursively (De Boor 1978), implemented in R via the function `splines::splineDesign`.

Semi-parametric log-binomial models can be fitted in **logbin** by using the `logbin.smooth` function. We illustrate its use with an augmented version of the heart attack data described in Section 3.1. In their supplementary material, Marschner and Gillett (2012) provide this data, which has patient age in years rather than in broad categories.

B-spline terms can be included in the `logbin.smooth` formula by using `B()`, which has two additional arguments. `knot.range` is used to specify the number of internal turning points in the curve, placed at evenly spaced quantiles of the observed covariate values. Figure 4 shows the fitted risk of death by age for models of increasing complexity, from a linear age term up to a semi-parametric model with 10 internal turning points.

An information criterion can be used to select the "best" model, and following the suggestion of Donoghoe and Marschner (2015) we choose the model with the lowest small-sample corrected Akaike information criterion, AIC_c (Burnham and Anderson 2002). In this case, the smooth model with one internal turning point has the lowest AIC_c. This entire process is automated by `logbin.smooth` if `knot.range` is provided as a vector, with the parameter-expanded EM algorithm and SQUAREM acceleration scheme used to speed up convergence:

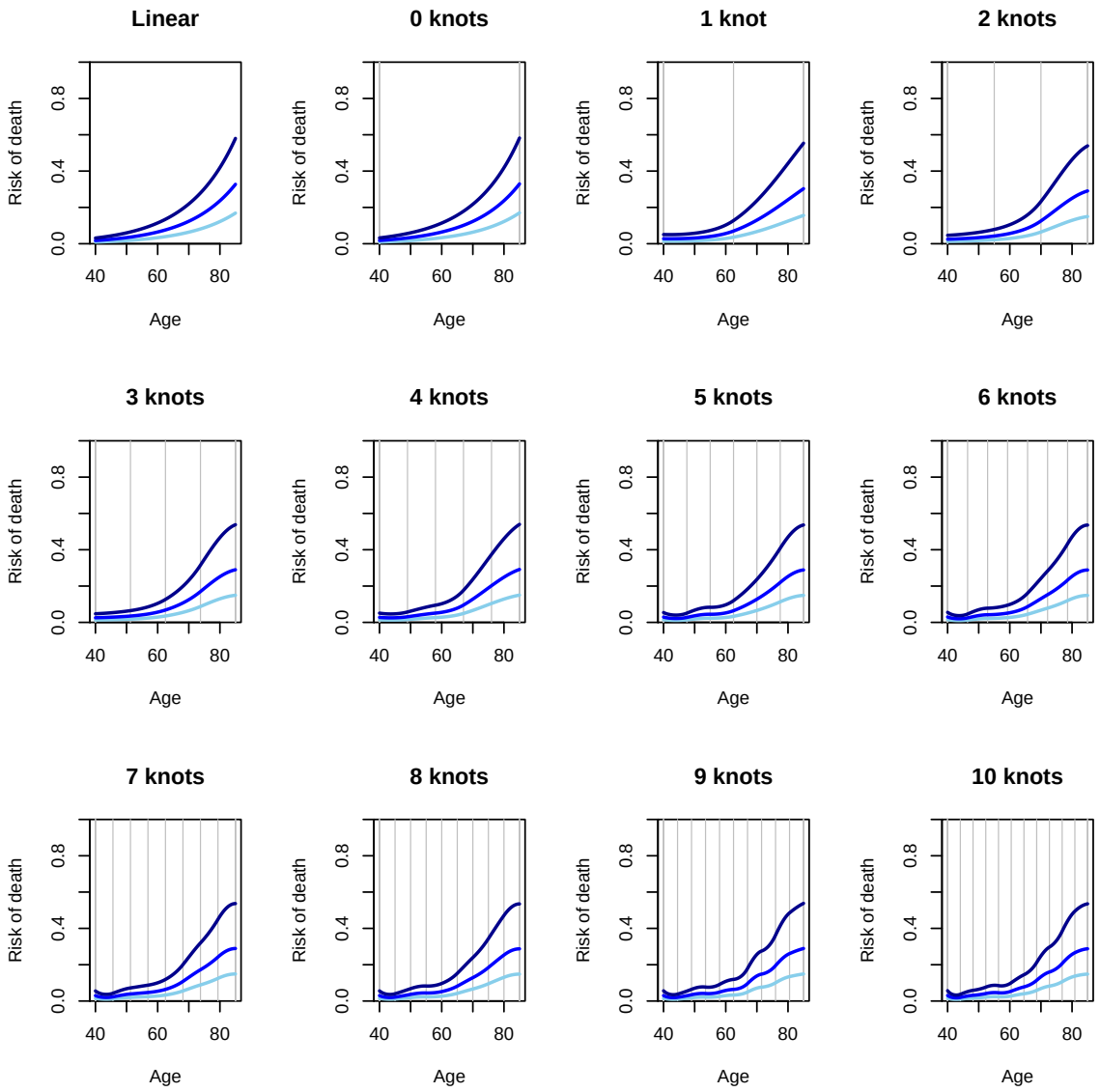

Figure 4: Fitted risk of death by age for individuals from Western countries with <2 hour treatment delay after mild (light blue), moderate (blue) and severe (dark blue) heart attacks, based on models with increasing complexity for the semi-parametric effect of age. The vertical grey lines show the location of the turning points for the B-splines.

```
R> m.bestsmooth <- logbin.smooth(cbind(deaths, patients - deaths) ~
+     factor(severity) + factor(onset) + factor(region) +
+     B(age, knot.range = 0:10), data = heart2, method = "em",
+     accelerate = "squarem")
```

Alternatively, the user can specify the `knots` argument to the `B` function, providing a vector that contains the locations of the internal turning points. A `plot` method for class '`logbin.smooth`' has been defined so that fitted semi-parametric models can be easily visualized. The `at` argument must provide a `data.frame`, in which each row determines the levels of the other covariates in the model at which the fitted risks will be calculated.

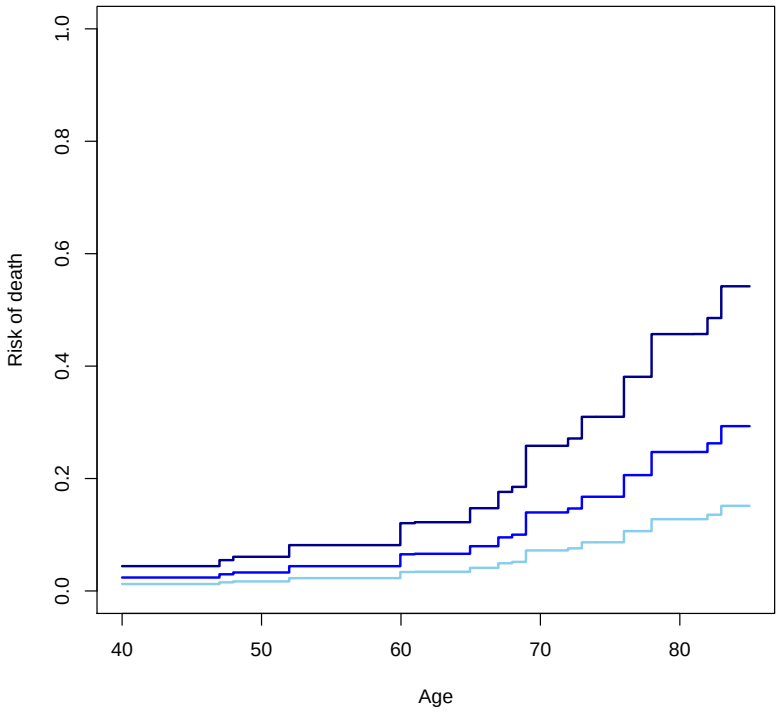

Figure 5: Fitted risk of death by age for individuals from Western countries with <2 hour treatment delay after mild (light blue), moderate (blue) and severe (dark blue) heart attacks, based on a model with an isotonic relationship between age and risk of death.

The `mono` argument can be used in a call to `logbin.smooth` in the same way as it is for `logbin`. Constraining the B-spline parameters to be monotonically non-decreasing is equivalent to using I-spline basis functions with non-positivity constraints on each coefficient, which will produce a monotonically non-decreasing smooth curve (Tutz and Leitenstorfer 2007; Donoghoe and Marschner 2015). Alternatively, non-smooth monotonic relationships can be included by specifying an `Iso()` term in the `formula` for `logbin.smooth`. In this case, the unknown f_j is parameterized as a step function that may increase at each unique observed covariate value. The result from fitting such a model to our example is shown in Figure 5.

8. Discussion

Relative risk regression is an important alternative to logistic regression for binomial outcomes, as it provides an effect measure that is arguably easier to interpret than the odds ratio. However, standard methods for maximum likelihood estimation can encounter difficulties with the log-binomial model, signalling the need for specialized statistical software to reliably fit such models.

The **logbin** package presented in this paper provides an interface in R to algorithms that perform maximum likelihood estimation for log-binomial models. Specifically, EM-type algorithms described by Marschner and Gillett (2012) and Donoghoe and Marschner (2016) and

an adaptive barrier approach based on the method of Lange (1994) and suggested by Lumley *et al.* (2006) and de Andrade and Carabin (2011) all provide stable convergence to the MLE, as demonstrated in Section 4.2. The usual modified Fisher scoring algorithm, as implemented in the `glm` function or **glm2** package (Marschner 2018) can also be accessed via **logbin**.

The speed of the adaptive barrier approach is comparable to that of the Fisher scoring algorithm. The combinatorial EM algorithm can be considerably slower in large models, but this can be substantially improved by applying the underlying EM algorithm to an overparameterized model (Donoghoe and Marschner 2016). Either can be further accelerated by using one of the algorithms implemented in the **turboEM** package (Bobb and Varadhan 2018). As shown by simulations in Section 5.2, a combination of these ideas can make the loss in speed negligible.

Some care must be taken to identify the parameter space that is being considered by the maximization routine, as this can cause differences to occur in the MLE found by competing approaches in boundary cases. Finally, the EM-type approaches can be used to include semi-parametric terms, providing additional flexibility that would usually require the use of additional packages such as **gam** (Hastie 2018).

The **logbin** package addresses the calls of Lumley *et al.* (2006) and Marschner (2015) that methods for log-binomial regression be made readily available in standard statistical software. We hope that this encourages the appropriate use of the relative risk in situations where it may have been avoided solely for computational reasons.

References

ASSENT-2 Investigators (1999). "Single-Bolus Tenecteplase Compared with Front-Loaded Alteplase in Acute Myocardial Infarction: The ASSENT-2 Double-Blind Randomised Trial." *The Lancet*, **354**(9180), 716–722. doi:10.1016/s0140-6736(99)07403-6.

Berlinet A, Roland C (2009). "Parabolic Acceleration of the EM Algorithm." *Statistics and Computing*, **19**(1), 35–47. doi:10.1007/s11222-008-9067-x.

Bobb JF, Varadhan R (2018). ***turboEM**: A Suite of Convergence Acceleration Schemes for EM, MM and Other Fixed-Point Algorithms*. R package version 2018.1, URL https://CRAN.R-project.org/package=turboEM.

Broyden CG, Dennis Jr JE, Moré JJ (1973). "On the Local and Superlinear Convergence of Quasi-Newton Methods." *IMA Journal of Applied Mathematics*, **12**(3), 223–245.

Burnham KP, Anderson DR (2002). *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. 2nd edition. Springer-Verlag, New York.

Davies HTO, Crombie IK, Tavakoli M (1998). "When Can Odds Ratios Mislead?" *BMJ*, **316**(7136), 989–991. doi:10.1136/bmj.316.7136.989.

de Andrade BB, Carabin H (2011). "On the Estimation of Relative Risks via Log Binomial Regression." *Revista Brasiliera de Biometria*, **29**(1), 25–46.

De Boor C (1978). *A Practical Guide to Splines*. Applied Mathematical Sciences. Springer-Verlag, New York.

Deddens JA, Petersen MR, Lei X (2003). "Estimation of Prevalence Ratios When PROC GENMOD Does Not Converge." In *Proceedings of the 28th Annual SAS Users Group International Conference*, Paper 270-28.

Dempster AP, Laird NM, Rubin DB (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society B*, **39**(1), 1–38.

Donoghoe MW (2018). **logbin**: *Relative Risk Regression Using the Log-Binomial Model*. R package version 2.0.4, URL https://CRAN.R-project.org/package=logbin.

Donoghoe MW, Marschner IC (2015). "Flexible Regression Models for Rate Differences, Risk Differences and Relative Risks." *International Journal of Biostatistics*, **11**(1), 91–108. doi:10.1515/ijb-2014-0044.

Donoghoe MW, Marschner IC (2016). "Fast Stable Relative Risk Regression Using an Over-parameterised EM Algorithm." In JF Dupuy, J Josse (eds.), *Proceedings of the 31st International Workshop on Statistical Modelling*, volume 1, pp. 93–98.

Forrow L, Taylor WC, Arnold RM (1992). "Absolutely Relative: How Research Results Are Summarized Can Affect Treatment Decisions." *The American Journal of Medicine*, **92**(2), 121–124. doi:10.1016/0002-9343(92)90100-p.

Grimes DA, Schulz KF (2008). "Making Sense of Odds and Odds Ratios." *Obstetrics & Gynecology*, **111**(2, Part 1), 423–426. doi:10.1097/01.aog.0000297304.32187.5d.

Hardin J, Cleves M (1999). "Generalized Linear Models: Extensions to the Binomial Family." *Stata Technical Bulletin*, **50**, 21–25.

Hastie T (2018). **gam**: *Generalized Additive Models*. R package version 1.16, URL https://CRAN.R-project.org/package=gam.

Holcomb Jr WL, Chaiworapongsa T, Luke DA, Burgdorf KD (2001). "An Odd Measure of Risk: Use and Misuse of the Odds Ratio." *Obstetrics & Gynecology*, **98**(4), 685–688. doi:10.1016/s0029-7844(01)01488-0.

IBM Corporation (2016). *IBM SPSS Statistics V24.0 Documentation*, chapter Generalized Linear Models Estimation. IBM Corporation, Armonk.

Lacy CR, Barone JA, Suh DC, Malini PL, Bueno M, Moylan DM, Kostis JB (2001). "Impact of Presentation of Research Results on Likelihood of Prescribing Medications to Patients with Left Ventricular Dysfunction." *The American Journal of Cardiology*, **87**(2), 203–207. doi:10.1016/s0002-9149(00)01317-5.

Lange K (1994). "An Adaptive Barrier Method for Convex Programming." *Methods and Applications of Analysis*, **1**(4), 392–402. doi:10.4310/maa.1994.v1.n4.a1.

Lange K (2013). *Optimization*. Springer Texts in Statistics, 2nd edition. Springer-Verlag, New York. doi:10.1007/978-1-4614-5838-8.

Lumley T, Kronmal R, Ma S (2006). "Relative Risk Regression in Medical Research: Models, Contrasts, Estimators, and Algorithms." Working Paper 293, URL http://biostats.bepress.com/uwbiostat/paper293.

Marschner IC (2011). "**glm2**: Fitting Generalized Linear Models with Convergence Problems." *The R Journal*, **3**(2), 12–15.

Marschner IC (2014). "Combinatorial EM Algorithms." *Statistics and Computing*, **24**(6), 921–940. doi:10.1007/s11222-013-9411-7.

Marschner IC (2015). "Relative Risk Regression for Binary Outcomes: Methods and Recommendations." *Australian & New Zealand Journal of Statistics*, **57**(4), 437–462. doi:10.1111/anzs.12131.

Marschner IC (2018). ***glm2**: Fitting Generalized Linear Models*. R package version 1.2.1, URL https://CRAN.R-project.org/package=glm2.

Marschner IC, Gillett AC (2012). "Relative Risk Regression: Reliable and Flexible Methods for Log-Binomial Models." *Biostatistics*, **13**(1), 179–192.

McCullagh P (2005). "Proportional-Odds Model." In P Armitage, T Colton (eds.), *Encyclopedia of Biostatistics*, volume 6, pp. 4277–4280. John Wiley & Sons, Chicester.

McCullagh P, Nelder JA (1989). *Generalized Linear Models*. Monographs on Statistics and Applied Probability, 2nd edition. Chapman and Hall, London.

Ramsay JO (1988). "Monotone Regression Splines in Action." *Statistical Science*, **3**(4), 425–441.

Raydan M, Svaiter BF (2002). "Relaxed Steepest Descent and Cauchy-Barzilai-Borwein Method." *Computational Optimization and Applications*, **21**(2), 155–167. doi:10.1023/a:1013708715892.

R Core Team (2018). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Sackett DL, Deeks JJ, Altman DG (1996). "Down with Odds Ratios!" *Evidence-Based Medicine*, **1**(6), 164–166.

SAS Institute Inc (2013). *The SAS System, Version 9.4*. SAS Institute Inc., Cary. URL http://www.sas.com/.

Schouten EG, Dekker JM, Kok FJ, Cessie SL, Van Houwelingen HC, Pool J, Vandenbroucke JP (1993). "Risk Ratio and Rate Ratio Estimation in Case-Cohort Designs: Hypertension and Cardiovascular Mortality." *Statistics in Medicine*, **12**(18), 1733–1745. doi:10.1002/sim.4780121808.

StataCorp (2015). *Stata 14 Base Reference Manual*. Stata Press, College Station.

Tutz G, Leitenstorfer F (2007). "Generalized Smooth Monotonic Regression in Additive Modeling." *Journal of Computational and Graphical Statistics*, **16**(1), 165–188. doi:10.1198/106186007x180949.

Varadhan R, Roland C (2008). "Simple and Globally Convergent Methods for Accelerating the Convergence of Any EM Algorithm." *Scandinavian Journal of Statistics*, **35**(2), 335–353. doi:10.1111/j.1467-9469.2007.00585.x.

Wacholder S (1986). "Binomial Regression in GLIM: Estimating Risk Ratios and Risk Differences." *American Journal of Epidemiology*, **123**(1), 174–184. doi:10.1093/oxfordjournals.aje.a114212.

Williamson T, Eliasziw M, Fick GH (2013). "Log-Binomial Models: Exploring Failed Convergence." *Emerging Themes in Epidemiology*, **10**(1), 1–10. doi:10.1186/1742-7622-10-14.

Yu B, Wang Z (2008). "Estimating Relative Risks for Common Outcome Using PROC NLP." *Computer Methods and Programs in Biomedicine*, **90**(2), 179–186. doi:10.1016/j.cmpb.2007.12.010.

Zhou H, Alexander D, Lange K (2011). "A Quasi-Newton Acceleration for High-Dimensional Optimization Algorithms." *Statistics and Computing*, **21**(2), 261–273. doi:10.1007/s11222-009-9166-3.

Zou G (2004). "A Modified Poisson Regression Approach to Prospective Studies with Binary Data." *American Journal of Epidemiology*, **159**(7), 702–706.

Affiliation:

Mark W. Donoghoe
Stats Central
Mark Wainwright Analytical Centre
UNSW Sydney, New South Wales 2052, Australia
E-mail: markdonoghoe@gmail.com

Ian C. Marschner
Department of Statistics
Faculty of Science and Engineering
Macquarie University
New South Wales 2109, Australia
and
NHMRC Clinical Trials Centre
University of Sydney
New South Wales 2006, Australia
E-mail: Ian.Marschner@mq.edu.au

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
August 2018, Volume 86, Issue 9 *Submitted:* 2017-03-08
doi:10.18637/jss.v086.i09 *Accepted:* 2017-10-29