groupTesting: An R package for group testing estimation

Md S. Warasi

Department of Mathematics and Statistics, Radford University, Radford, VA 24142, USA

email: msarker@radford.edu

Abstract: Group (pooled) testing has long been used for the monitoring and detection of infectious diseases. In group testing, pools comprised of individual biospecimens are amalgamated and tested initially, and then the individuals belonging to positive pools are retested to complete case identification. This procedure and its variants can offer substantial savings with regard to time and testing cost. Unfortunately, these savings come at the expense of a complex data structure; e.g., ambiguities due to imperfect testing and complex dependence caused by individuals potentially being tested in multiple pools. To account for these complex features, several advanced statistical methods have been proposed. Regretfully, these methods are non-trivial to implement, especially for non-statistician. Recognizing this as an important issue in surveillance programs, we have developed a user-friendly `R` package called `groupTesting` that can be used to analyze group testing data. In particular, our package consists of `R` functions which possess a great deal of versatility and generality, and can be used in estimating both proportions and binary regression functions. The computing efficiency of our package is greatly enhanced through the strategic implementation of compiled `Fortran` subroutines. The primary features of the `R` package are illustrated using HIV and chlamydia data.

Keywords: Binary regression; EM algorithm; Gibbs sampling; Pooling; Screening; Surveillance.

1 Introduction

When screening individuals for low-prevalence infectious diseases, group testing serves as a cost-efficient alternative to individual testing. Group testing was introduced by Dorfman (1943) to screen American soldiers for syphilis during the Second World War. Dorfman performed the tests in two stages, where pooled blood specimens were tested in stage 1 and positive pools were resolved by individual testing in stage 2. Since Dorfman's work, group testing has been widely used in screening individuals for HIV (Pilcher et al. 2005), HBV and HCV (Hourfar et al. 2008; Stramer et al. 2013), and chlamydia and gonorrhea (Lindan et al. 2005). Group testing has also been used in genetics (Chi et al. 2009), drug discovery (Kainkaryam and Woolf 2009), and animal disease testing (Dhand, Johnson, and Torbio 2010). To increase testing capacity, group testing has been identified as a useful strategy for influenza (Van et al. 2012) and SARS-CoV-2 (Abdalhamid et al. 2020; Hogan, Sahoo, and Pinsky 2020; Pilcher, Westreich, and Hudgens 2020).

Statistical literature on group testing can be split into two separate directions: case identification and estimation. The aim of case identification is to classify individuals as diseased or non-diseased through algorithms that involve retesting positive pools (Kim et al. 2007; Verdun et al. 2021). The aim of estimation is to estimate the probability of disease. In a homogeneous population (i.e., where the probability of disease for all subjects is identical), the goal is generally to estimate the population prevalence (Liu et al. 2012; Speybroeck et al. 2012; Ding and Xiong 2016; Haber, Malinovsky, and Albert 2018; Nguyen, Bish, and Aprahamian 2018). When applied to a heterogeneous population, the goal is to estimate individual-level disease probabilities using regression techniques. Recently, the regression problems have received a great amount of attention (Vansteelandt, Goetghebeur, and Verstraeten 2000; Chen, Tebbs, and Bilder 2009; Delaigle, Hall, and Wishart 2014; Wang et al. 2014; Wang, McMahan, and Gallagher 2015; McMahan et al. 2017; Liu et al. 2021). Our work focuses on likelihood-based estimation in both homogeneous and heterogeneous populations.

Group testing data is inherently complex because the test responses consist of (a) initial pooled responses, and (b) retesting responses which arise when positive pools are resolved, i.e., to classify individuals as diseased or not. Regardless of which type of test response is available, the goal is always to make inference on each individual. In a perfect world, a pool would test negative when all individuals in the pool are truly negative, and a pool would test positive when at least one individual in the pool is truly positive. However, because diagnostic assays are usually prone to errors (false negatives and false positives), test responses can be misclassified and thus the true statuses of both pools and individuals are latent. In such instances, group testing problems involve estimating a latent binary model for individuals' true disease statuses. This is especially challenging for group testing data when the models incorporate retesting responses or individual covariate information (e.g., age, sex, presence of symptoms, number of partners, etc.) in a regression context.

Estimation based on group testing has been broadly studied. The early developments involved simplified models using only initial pooled responses. The recent developments have considered more sophisticated models that not only incorporate initial pooled responses but also take advantage of retest responses observed naturally as part of the process of screening individuals for disease (Xie 2001; Zhang, Bilder, and Tebbs 2013; Zhang et al. 2019). Unfortunately, implementing these advanced techniques is non-trivial, especially for practitioners. This project aims at addressing these challenges and facilitating applications by introducing the `R` (R Core Team 2021) package `groupTesting` (Warasi 2021). We consider a general model framework that can accommodate *any* group testing data and accomplish estimation by using the expectation-maximization (EM) algorithm presented in Xie (2001). We explore the computing aspects of the EM algorithm with a Markov Chain Monte Carlo approach that does not depend on analytic calculation and can be used for both proportion and regression problems. Consequently, one can easily find estimates from the advanced group testing methods and can also compare the estimator qualities over different testing protocols. Two user-friendly `R` functions are provided in `groupTesting` to implement the estimation technique.

There are several software packages currently available for group testing estimation. For example, written for molecular xenomonitoring applications, the standalone software `PoolScreen` (Katholi and Barker 2010) and the `R` package `PoolTestR` (McLure 2021) can model binary pooled responses. However, they require that the assay used for diagnosis is perfect. The `R` package `pooling` (Domelen 2020) is designed for case control studies with continuous biomarker and thus not suited for our model. The only recognizable `R` packages consistent with our modeling framework are `binGroup` (Zhang et al. 2018) and its successor `binGroup2` (Hitt et al. 2020). However, a major limitation of `binGroup` and `binGroup2` is that these packages are designed only for problem-specific scenarios, such as 2-stage hierarchical testing (Dorfman 1943). Our `R` package overcomes these limitations and provides a general approach for modeling group testing data. To find the maximum likelihood estimate of the prevalence or regression coefficients, one can simply input the data into `groupTesting` in a particular manner, analogous to the `datalines` in SAS; i.e., `groupTesting` operates solely based on the data input, without requiring any information about the group testing protocol used. A package with such flexibility and generality is currently unavailable for group testing estimation.

`groupTesting` can use data arising from simple pooling as well as advanced pooling such as hierarchical testing with 2 stages (Dorfman 1943), 3 stages (Pilcher et al. 2005), 4 stages (Quinn et al. 2000) or even higher stages (Black, Bilder, and Tebbs 2015), array testing with 2 dimensions (Kim et al. 2007) and 3 dimensions (Kim and Hudgens 2009), and quality-control pooling (Gastwirth and Johnson 1994). Recently, numerous pooling strategies have been investigated to screen individuals for SARS-CoV-2 (Westreich, Diepstra, and Max 2020; Bish et al. 2021; Daniel et al. 2021; Ghosh et al. 2021; Lin et al. 2021; Mutesa et al. 2021). `groupTesting` can be used for modeling data from these or any other group testing protocols. The estimation framework we consider can be viewed as a generalization of many estimation methods available in the literature, including the ones with simple pooling (Vansteelandt et al. 2000; Bilder and Tebbs 2009; Hepworth and Watson 2009; Liu et al. 2012; Huang and Warasi 2017; Chatterjee and Bandyopadhyay 2020), hierarchical testing (Brookmeyer 1999; Xie 2001; Zhang et al. 2019), and array testing (Bilder et al. 2010; Zhang et al. 2013). These methods can be implemented using `groupTesting` in different estimation scenarios. For example, our work can be useful in monitoring the trends in SARS-CoV-2 prevalence.

The subsequent sections are organized as follows. In Section 2, we present the model and discuss how the EM algorithm in Xie (2001) can be used in our estimation framework. In Section 3, we describe the `R` functions provided in `groupTesting`. In Section 4, we illustrate `groupTesting` with HIV surveillance data from Vansteelandt et al. (2000) and chlamydia testing data obtained from the State Hygienic Laboratory at the University of Iowa. In Section 5, we conclude with a brief discussion.

2 Preliminaries

Suppose that N individuals are to be tested for a disease (e.g., HIV). Let $\widetilde{Y}_i$ denote the true status of individual i, for $i = 1, 2, ..., N$, where $\widetilde{Y}_i = 1$ if the ith individual is truly positive for the disease and $\widetilde{Y}_i = 0$ if otherwise. Let $\widetilde{Y}_i$'s be independent Bernoulli random variables. In a regression context, the covariate vector $\mathbf{x}_i = (1, x_{i1}, ..., x_{ir})^T$ is observed from the ith individual and related to $\widetilde{Y}_i$ as

$$\mathrm{pr}(\widetilde{Y}_i = 1|\mathbf{x}_i; \boldsymbol{\beta}) = g(\mathbf{x}_i^T\boldsymbol{\beta}), \tag{1}$$

where $\boldsymbol{\beta} = (\beta_0, \beta_1, ..., \beta_r)^T$ is an $(r + 1) \times 1$ vector of regression coefficients, g is a monotonic, differentiable, inverse link function from the generalized linear model (GLM) family, and T is the transpose of a matrix or vector. The joint probability mass function (PMF) for the vector of individual true statuses $\widetilde{\mathbf{Y}} = (\widetilde{Y}_1, \widetilde{Y}_2, ..., \widetilde{Y}_N)^T$ is

$$\pi(\widetilde{\mathbf{Y}}; \boldsymbol{\beta}) = \prod_{i=1}^{N} g(\mathbf{x}_i^T\boldsymbol{\beta})^{\widetilde{Y}_i}(1 - g(\mathbf{x}_i^T\boldsymbol{\beta}))^{1-\widetilde{Y}_i}. \tag{2}$$

Finding the maximum likelihood estimate (MLE) of $\boldsymbol{\beta}$ is of interest. However, this cannot be done directly from the population-level model in Equation (2) because $\widetilde{Y}_i$'s are unobservable, either due to pooling or testing error. In this article, $\boldsymbol{\beta}$ is estimated from the observable pooled testing data.

Pooled testing is performed in different ways. Generally, individual specimens are assigned to the initial pools, and then the tests are conducted in one or multiple stages. This procedure yields test responses from pools of the individuals who can be located in only one pool or in multiple pools. Suppose that the number of observed test responses is J. Let $\mathcal{P}_j \subseteq \{1, 2, ..., N\}$ be an index set of the individuals belonging to the jth pool, for $j = 1, 2, ..., J$. Denote by $\widetilde{Z}_j = I(\sum_{i\in\mathcal{P}_j} \widetilde{Y}_i > 0)$ the true status of pool j; i.e., $\widetilde{Z}_j = 1$ if at least one individual in the jth pool is truly positive and $\widetilde{Z}_j = 0$ if otherwise. Due to potential testing error, the true pooled statuses $\widetilde{Z}_j$'s are unobservable. Let Z_j denote the pooled test response observed in place of $\widetilde{Z}_j$. Let $S_{e_j} = \mathrm{pr}(Z_j = 1|\widetilde{Z}_j = 1)$ and $S_{p_j} = \mathrm{pr}(Z_j = 0|\widetilde{Z}_j = 0)$ denote the assay sensitivity and specificity, respectively, for the jth pool. We assume that there is no dilution effect due to pooling. We also assume that the test responses Z_j's are independent conditional on their true statuses, $\widetilde{Z}_j$'s. These assumptions are commonly used in group testing (Vansteelandt et al. 2000; Xie 2001).

With the assumptions, the joint PMF of the observed data $\mathbf{Z} = (Z_1, Z_2, ..., Z_J)^T$ conditional on the latent data $\widetilde{\mathbf{Y}}$ can be expressed as

$$\pi(\mathbf{Z}|\widetilde{\mathbf{Y}}) = \prod_{j=1}^{J} S_{e_j}^{Z_j \widetilde{Z}_j}(1 - S_{e_j})^{(1-Z_j)\widetilde{Z}_j} S_{p_j}^{(1-Z_j)(1-\widetilde{Z}_j)}(1 - S_{p_j})^{Z_j(1-\widetilde{Z}_j)}. \tag{3}$$

Then the likelihood function is

$$L(\boldsymbol{\beta}|\mathbf{Z}) = \sum_{\widetilde{\mathbf{Y}} \in \{0,1\}^N} \pi(\mathbf{Z}|\widetilde{\mathbf{Y}})\pi(\widetilde{\mathbf{Y}}; \boldsymbol{\beta}).$$

It is worth noting that $\mathbf{Z}$ is the collection of all test responses observed; i.e., $\mathbf{Z}$ may consist of only initial pooled responses or both initial pooled and retest responses. Consequently, $L(\boldsymbol{\beta}|\mathbf{Z})$ can be considered as a general-form of the likelihood functions that appeared in a large number of articles in the literature. Unfortunately, $L(\boldsymbol{\beta}|\mathbf{Z})$ is computationally intractable in many practical situations as it involves summation over 2^N terms. To circumvent this issue, Xie (2001) presented an expectation-maximization (EM) algorithm, which is the approach we espouse in calculating the MLE of $\boldsymbol{\beta}$. To better explain how our computing is performed, we briefly discuss the EM algorithm.

The EM algorithm proceeds iteratively where the individual true statuses in $\widetilde{\mathbf{Y}}$ are regarded as "missing data." Let $\boldsymbol{\beta}^{(d)}$ denote the current estimate of $\boldsymbol{\beta}$. Then $\boldsymbol{\beta}^{(d)}$ is updated in the M-step as

$$\boldsymbol{\beta}^{(d+1)} = \arg\max_{\boldsymbol{\beta}} \sum_{i=1}^{N} \left[E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}^{(d)}] \ln g(\mathbf{x}_i^T\boldsymbol{\beta}) + (1 - E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}^{(d)}]) \ln(1 - g(\mathbf{x}_i^T\boldsymbol{\beta})) \right], \tag{4}$$

which depends on the conditional expectation $E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}^{(d)}]$ to be evaluated in the E-step.

Here are some remarks about the E- and M-step. First, exact calculation of $E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}^{(d)}]$ is difficult or impossible in many pooling scenarios such as array testing (Xie 2001). In such settings, $E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}^{(d)}]$ can be calculated numerically using Gibbs sampling. Particularly, a large number of Gibbs samples are obtained from the conditional distribution $\widetilde{Y}_i|\mathbf{Z}, \widetilde{\mathbf{Y}}_{-i}, \boldsymbol{\beta}^{(d)} \sim$ Bernoulli(p_i^*) and then the mean of the samples is used as an estimate of $E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}^{(d)}]$, where $\widetilde{\mathbf{Y}}_{-i}$ is $\widetilde{\mathbf{Y}}$ excluding $\widetilde{Y}_i$ and p_i^* is provided in Appendix A. Second, the maximizer in Equation (4) can be found explicitly in many estimation problems, especially when no covariates are involved. When an explicit expression does not exist, standard optimization routines, such as `optim` in `R`, can be used. The EM algorithm is now described below.

EM ALGORITHM

1. Specify $\boldsymbol{\beta}^{(0)}$, an initial value of $\boldsymbol{\beta}$. Set $d = 0$.
2. (E-Step): For $i = 1, 2, ..., N$,

- sample $\widetilde{Y}_i^{(h)}$ from $\widetilde{Y}_i|\mathbf{Z}, \widetilde{\mathbf{Y}}_{-i}, \boldsymbol{\beta}^{(d)} \sim \text{Bernoulli}(p_i^*)$, for $h = 1, 2, ..., H$, where H is the number of Gibbs iterates;
- calculate the sample mean $H^{-1}\sum_{h=1}^{H} \widetilde{Y}_i^{(h)}$ as an estimate of the conditional expectation $E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}^{(d)}]$.

3. (M-Step): Calculate $\boldsymbol{\beta}^{(d+1)}$ from Equation (4).
4. Set $d = d+1$, and repeat steps 2-4 until $\max|\boldsymbol{\beta}^{(d+1)} - \boldsymbol{\beta}^{(d)}|$ is less than a small convergence tolerance.

When convergence is established, $\widehat{\boldsymbol{\beta}} = \boldsymbol{\beta}^{(d+1)}$ is found to be the MLE of $\boldsymbol{\beta}$. Note that a sufficient number of initial Gibbs iterates need to be discarded as a burn-in period before using H iterates for estimating $E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}^{(d)}]$ in the E-step.

The observed-information matrix can be calculated by an appeal to the missing data principle and the method in Louis (1982) as

$$\mathcal{I}(\boldsymbol{\beta}) = -\frac{\partial^2 Q(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}^T} - \text{cov}\left\{\frac{\partial \ell_C(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}}\middle|\mathbf{Z}, \boldsymbol{\beta}\right\}, \tag{5}$$

where $Q(\boldsymbol{\beta})$ and $\ell_C(\boldsymbol{\beta})$ are provided in Appendix B. The covariance matrix $\mathcal{I}(\boldsymbol{\beta})^{-1}$ is calculated at the MLE $\widehat{\boldsymbol{\beta}}$ as in Xie (2001), and then large-sample Wald inferences on $\boldsymbol{\beta}$ can be made conventionally. More information about the covariance matrix is shown in Appendix B.

With the assumptions adopted, the estimation method presented above encompasses a large range of models. However, the major challenge with its implementation is that $E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}^{(d)}]$ cannot be evaluated easily because an individual is potentially located in multiple pools. While $E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}^{(d)}]$ can be calculated analytically with simple pooling data, we use Gibbs sampling as a common approach in order to write a general software package. We use Gibbs sampling for calculating the covariance matrix $\mathcal{I}(\boldsymbol{\beta})^{-1}$ as well. A drawback of this approach is that it involves a great deal of computing. However, we overcome the computing challenge using compiled `Fortran` subroutines; see Section 3.

3 Software package

Developing a flexible and user-friendly software package for group testing data analysis is at the heart of this article. With this objective, we provide the `R` function `glm.gt` in `groupTesting` that can implement the EM algorithm presented in Section 2. The main features of `glm.gt` are shown in Table 1. We provide a customized version of `glm.gt` for the overall proportion estimation problems (i.e., when no covariates are observed). Several other useful functions are also provided in `groupTesting`; see the package vignette.

[Table 1 near here]

To input the data into glm.gt, we developed a data structure that combines the information observed in a group testing scheme. An example of such data input is shown in Table 2, which consists of information from a 3-stage hierarchical testing protocol (Kim et al. 2007), where $N = 12$ individuals are assigned to 2 non-overlapping initial pools in stage 1, and then tested subsequently with subpools of size 2 in stage 2 and individual retesting in stage 3. Columns 1-5 consist of the pooled test outcomes, pool sizes, sensitivities, specificities, and assays identification (ID) numbers, respectively. From column 6 onward, the pool member ID numbers need to be specified. Note that the ID numbers must start with 1 and increase consecutively up to $N = 12$. For smaller pools, incomplete ID numbers must be filled out by -9 or any non-positive numbers as shown in Table 2. The row name (i.e., pool ID) does not need to be specified because it is not used in glm.gt. One can easily see from the example in Table 2 why glm.gt is independent of the group testing protocol used.

[**Table 2 near here**]

With the data input structured in that manner, glm.gt internally keeps track of the pool members and extracts other information necessary for the EM algorithm. The Gibbs sampling is written entirely in Fortran. The Fortran subroutines are then called into R through the R function .Call, which requires using the internal architecture of programming C and R. This calling technique is difficult but usually more efficient in computing than the other calling techniques in R using function .C or .Fortran.

groupTesting can be useful in analyzing different types of group testing data, which is demonstrated in Section 4. Function glm.gt calculates the MLE $\widehat{\boldsymbol{\beta}}$, the covariate matrix $\mathcal{I}(\boldsymbol{\beta})^{-1}$ at the MLE, and Wald confidence intervals. Two functions are provided in groupTesting that can be useful in simulating hierarchical testing and array testing data. Some utility functions are also provided. To assess the performance of our work, we perform a simulation study in Appendix C. The simulation evidence shows that the package provides accurate estimates. Also, the data application results in Section 4.2 and the simulation evidence in Appendix C show that our programs are efficient.

4 Examples

We illustrate the package with two data sets obtained from infectious disease testing applications. We show how glm.gt and other functions that are provided in groupTesting can be used in such application scenarios.

4.1 HIV data

First, we illustrate glm.gt with data from a study of HIV surveillance on pregnant women in Kenya (Vansteelandt et al., 2000). The goal of the study was to monitor the trends in

HIV prevalence with adjustments for individual risk factors, such as age. Another goal was to investigate the feasibility of using group testing. The R package binGroup (Zhang et al. 2018) comprises the surveillance data from Vansteelandt et al. (2000), called hivsurv, which consists of individual HIV test outcomes (1 for positive and 0 for negative) and covariate information from 428 pregnant women. Using the individual test responses, Bilder et al. (2010) generated $\mathbf{Z}$, the vector of pooled responses, based on simple pooling as well as array testing, and they estimated the following latent GLM model

$$\text{logit}\{\text{pr}(\widetilde{Y}_i = 1|\mathbf{x}_i)\} = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2},$$

for $i = 1, 2, ..., 428$, where x_1 is age, x_2 is the highest level of completed education, and $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2)^T$ is the parameter of interest. We take the same approach but we use our R functions, provided in groupTesting, for simulation and estimation. The surveillance data can be extracted from binGroup as

```
> library(binGroup)
> data(hivsurv)
> x <- cbind(1, hivsurv[ ,c(3,5)])  # design matrix
> colnames( x ) <- c("Intercept", "Age", "Education")
> indv.resp <- hivsurv[ ,6]         # Individual HIV test outcomes
```

We show an application of glm.gt with the regression method that Vansteelandt et al. (2000) developed for modeling non-overlapping pooled responses. In doing so, we chronologically assign the individual test results to the initial pools and simulate pooled responses by using function hier.gt.simulation with pool size psz=5, sensitivity Se=0.99, and specificity Sp=0.95 as used in Bilder et al. (2010). A pool of size 3 is also used for the remainder individuals. The individual test results are input into hier.gt.simulation as Yt=indv.resp. Also, input are the number of stages S=1 and the assay ID number assayID=1. In the simulation, we treat the individual test results, indv.resp, as the individual true statuses, which is reasonable because testing error was negligible in those individual HIV tests; for more information, see Vansteelandt et al. (2000).

```
> library(groupTesting)    # Loading the package
> set.seed(123)
> IPT.gtData <- hier.gt.simulation(N=428, S=1, psz=5, Se=0.99, Sp=0.95,
+                                  assayID=1, Yt=indv.resp)$gtData
> ## Simulated data:
> tail(IPT.gtData)
        Z psz   Se   Sp Assay Mem1 Mem2 Mem3 Mem4 Mem5
Pool:81 0   5 0.99 0.95     1  401  402  403  404  405
Pool:82 1   5 0.99 0.95     1  406  407  408  409  410
Pool:83 0   5 0.99 0.95     1  411  412  413  414  415
Pool:84 1   5 0.99 0.95     1  416  417  418  419  420
Pool:85 1   5 0.99 0.95     1  421  422  423  424  425
Pool:86 0   3 0.99 0.95     1  426  427  428   -9   -9
```

The simulated data IPT.gtData complies with the data input structure suited for glm.gt. The EM algorithm is carried out with IPT.gtData and the design matrix x as follows. Using our utility function glmLink with argument fn.name="logit", we find the inverse logit link g as well as its first and second derivatives that are needed for estimating the covariance matrix. To approximate the expectations (used in the E-step and covariance matrix), ngit=10000 Gibbs iterates are used after discarding nburn=3000 initial iterates as a burn-in period. Confidence level for the Wald inference is specified as conf.level=0.95. A reasonable convergence tolerance and starting value, such as tol=1e-03 and beta0=c(0,0,0), need to be specified as well. We use the default values for other arguments, not mentioned here for brevity.

```
> # Fit the model:
> link.info <- glmLink(fn.name="logit")
> g <- link.info$g      # Inverse logit link g
> dg <- link.info$dg    # First derivative of g
> d2g <- link.info$d2g  # Second derivative of g
> IPT.res <- glm.gt(beta0=c(0,0,0), gtData=IPT.gtData, X=x,
+                   g=g, dg=dg, d2g=d2g, covariance=TRUE,
+                   nburn=3000, ngit=10000, tol=1e-03, conf.level=0.95)
```

The outputs are reported as a list object, IPT.res. The estimation summary (MLEs, standard errors, and 95% Wald confidence intervals) of the regression coefficients is

```
> IPT.res$summary
          Estimate Std.Err 95%lower 95%upper
Intercept   -2.782   1.610   -5.938    0.373
Age         -0.067   0.069   -0.202    0.068
Education    0.848   0.400    0.064    1.633
```

To fit the model with probit or complementary log-log link, one can use glmLink with argument fn.name="probit" or fn.name="cloglog". Also, any other link function from the GLM family can be specified into glm.gt. In the example above, we input the analytic first and second derivatives through arguments dg and d2g. When dg=NULL and d2g=NULL (default), a finite difference approximation will be used.

We briefly discuss how glm.gt can be used with more complex group testing data. Consider the square array without master pool testing (A2) protocol (Kim et al. 2007). Under this protocol, individual specimens are placed in $n \times n$ square arrays. Then the tests are performed on row/column pools as well as individual specimens for case identification. With the HIV surveillance data, we simulate the A2 pooling data using function array.gt.simulation with row/column size n=5. For illustration purposes, we assume that the tests are performed by two assays, where assay 1 is used for the row/column pools and assay 2 is for individual specimens. Suppose that the sensitivity and specificity for assay 1 are 0.99 and 0.95 as before and for assay 2 are 0.99 and 0.98. The simulation and estimation are performed as

```
> set.seed(123)
> A2.gtData <- array.gt.simulation(N=428, protocol="A2", n=5,
+                                  Se=c(0.99,0.99), Sp=c(0.95,0.98),
+                                  assayID=c(1,2), Yt=indv.resp)$gtData
> ## Fit the model:
> A2.res <- glm.gt(beta0=c(0,0,0), gtData=A2.gtData, X=x,
+                  g=g, dg=dg, d2g=d2g, covariance=TRUE,
+                  nburn=3000, ngit=10000, tol=1e-03, conf.level=0.95)

> ## Estimation summary:
> A2.res$summary
          Estimate Std.Err 95%lower 95%upper
Intercept   -3.669   0.958   -5.547   -1.791
Age         -0.009   0.034   -0.076    0.057
Education    0.643   0.216    0.219    1.066
```

The biggest strength of glm.gt is that it provides a flexible and general approach; i.e., its application is not limited to any specific group testing protocol. This is evident from the examples above, where the simulated data, IPT.gtData and A2.gtData, are observed from two different testing protocols but the model in Section 2 is fit invariantly. In the same manner, glm.gt can fit the model with any group testing data, regardless of its complexity. The only requirement is that the testing information needs to be arranged in the data input properly, as was discussed with the data example in Table 2. To view more examples, refer to Section 4.2, Appendix C, and the package vignette.

4.2 Chlamydia data

We show an application of groupTesting with chlamydia data collected at the State Hygienic Laboratory (SHL) at the University of Iowa. The data set consists of test responses from $N = 9810$ female subjects. Cervical swab specimens were collected from the subjects at different locations across the state and tested at the SHL in 2014. Individual covariate information was also recorded. The SHL usually uses 2-stage hierarchical testing for efficient screening, while individual testing is used occasionally. The tests are performed by the Aptima Combo 2 Assay (Gen-Probe, San Diego). To view more information about the data application, refer to McMahan et al. (2017).

We have 2350 initial pooled outcomes (2336 pools of size 4, 13 pools of size 3, and 1 pool of size 2) and 2813 retest outcomes by resolving the positive pools. Additionally, we have 425 test outcomes from the traditional individual testing. The test results are combined into a matrix object, gtData, to be entered into function glm.gt. In the data object, columns 1-2 consist of the test responses and pool sizes, respectively. The Aptima Combo 2 Assay sensitivity (0.942) and specificity (0.976), obtained from the product literature at www.hologic.com, are in columns 3-4. Because the same assay is used for all tests, the assay ID number in column 5 is 1. The individual ID numbers are arranged in column 6 onward as described in Section 3.

The individual covariates are combined into an $N \times 6$ matrix object, `x`, consisting of age (in years, x_1) and the binary (yes/no) covariates whether the individuals had symptoms (x_2), sexual contact with a partner who was positive for any STD in the previous year (x_3), a new partner in the last 90 days (x_4), and multiple partners in the last 90 days (x_5). The binary covariates are coded as 1 for yes and 0 for no. We fit the logistic regression model

$$\text{logit}\{\text{pr}(\widetilde{Y}_i = 1|\mathbf{x}_i)\} = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \beta_4 x_{i4} + \beta_5 x_{i5},$$

for $i = 1, 2, ..., N$.

Function `glm.gt` implements the EM algorithm with initial value `beta0=rep(0,6)`. For covariance matrix estimation, we make use of the numerical first and second derivatives of g using the default values `dg=NULL` and `d2g=NULL`. Other arguments are specified as shown with the examples in Section 4.1.

```
library(groupTesting)
> g <- glmLink(fn.name="logit")$g    # Inverse logit link
> res <- glm.gt(beta0=rep(0,6), gtData=gtData, X=x,
+               g=g, dg=NULL, d2g=NULL, covariance=TRUE,
+               nburn=3000, ngit=10000, tol=1e-03, conf.level=0.95)
```

The estimates with 95% Wald intervals are shown below. We have run the `R` code in an Intel 3.6GHz 32GB RAM machine. With `beta0=rep(0,6)`, the entire calculation is completed in 63 seconds. With a better starting point, the EM algorithm can converge even faster.

```
> res$summary
                  Estimate Std.Err 95%lower 95%upper
Intercept           -1.002   0.193   -1.380   -0.625
Age                 -0.077   0.008   -0.093   -0.061
Symptoms             0.435   0.086    0.266    0.604
Contact with STD     1.307   0.130    1.051    1.562
New partner          0.220   0.082    0.059    0.382
Multiple partners    0.349   0.109    0.135    0.563
```

The estimation summary is shown above, where each of the covariates has a significant effect and has an intuitive association with the probability of disease. For example, as age increases, the risk of chlamydia reduces. The estimation results are calculated from 5588 test responses, which amounts to a reduction of 43% of tests when compared to the situation where all subjects are tested individually.

5 Discussion

We have explored group testing estimation from a computing perspective. We calculate the likelihood-based estimates using the EM algorithm introduced in Xie (2001), where the E-step is approximated by a Markov Chain Monte Carlo technique. While this approach offers flexibility and produces accurate estimates, the computing task involved can be formidable, especially because group testing data is usually large. To alleviate the computing burden, we present a software package, `groupTesting`, which provides `R` functions with (nearly) optimized, compiled `Fortran` code.

An important feature of `groupTesting` is that any group testing data can be fed into its functions, without needing to specify information about the protocol used. This can be useful in different ways. For example, when data is collected based on multiple group testing protocols, one can combine the data sets into a single data input and still can find the estimates easily using `groupTesting`. An example of such amalgamation is shown in Section 4.2 with chlamydia data. Furthermore, one can use `groupTesting` to calculate the running estimates, which can be helpful in adapting the pooling design or developing an informative design such the ones in McMahan, Tebbs, and Bilder (2012a, 2012b).

The model we have considered has a basic structure, without allowing for other complexities such as dilution (Wang et al. 2015), measurement error in the covariates (Huang 2009), and random effects (Chen et al. 2009). However, our package can be extended to include these complexities as well as to accommodate group testing data with multiple infections (Tebbs, McMahan, and Bilder 2013).

Acknowledgments

We are grateful to Dr. Joshua M. Tebbs for providing the chlamydia data and to Dr. Christopher S. McMahan for his comments on the `R` package. We are also grateful to the reviewers for many helpful suggestions.

Disclosure statement

There are no conflicts.

Data availability statement

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

References

Abdalhamid, B., C. Bilder, E. McCutchen, S. Hinrichs, S. Koepsell, P. Iwen. 2020. Assessment of specimen pooling to conserve SARS-CoV-2 testing resources. *American Journal of Clinical Pathology* **153**:715-18. DOI:10.1093/AJCP/AQAA064.

Black, M., C. Bilder, J. Tebbs. 2015. Optimal retesting configurations for hierarchical group testing. *Journal of the Royal Statistical Society, Series C (Appl. Stat.)* **64**:693-710. DOI:10.1111/rssc.12097.

Bilder, C., J. Tebbs. 2009. Bias, efficiency, and agreement for group-testing regression models. *Journal of Statistical Computation and Simulation* **1**:67-80. DOI:10.1080/00949650701608990.

Bilder, C., B. Zhang, F. Schaarschmidt, J. Tebbs. 2010. binGroup: A Package for group testing. *The R Journal* **2**:56-60. DOI:10.32614/RJ-2010-016.

Bish, D., E. Bish, H. El-Hajj, H. Aprahamian. 2021. A robust pooled testing approach to expand COVID-19 screening capacity. *PLOS ONE* **16**:e0246285. DOI:10.1371/journal.pone.0246285.

Brookmeyer, R. 1999. Analysis of multistage pooling studies of biological specimens for estimating disease incidence and prevalence. *Biometrics* **55**:608-12. DOI:10.1111/j.0006-341x.1999.00608.x.

Chatterjee, A., T. Bandyopadhyay. 2020. Regression models for group testing: Identifiability and asymptotics. *Journal of Statistical Planning and Inference* **204**:141-52. DOI:10.1016/J.JSPI.2019.05.003.

Chen, P., J. Tebbs, C. Bilder. 2009. Group testing regression models with fixed and random effects. *Biometrics* **65**:1270-78. DOI:10.1111/j.1541-0420.2008.01183.x.

Chi, X., X. Lou, M. Yang, Q. Shu. 2009. An optimal DNA pooling strategy for progressive fine mapping. *Genetica* **135**:267-81. DOI: 10.1007/s10709-008-9275-5.

Daniel, E., B. Esakialraj, A. Muthuramalingam, R. Karunaianantham, L. Karunakaran, M. Nesakumar, M. Selvachithiram, S. Pattabiraman, S. Natarajan, S. Tripathy, L. Hanna. 2021. Pooled Testing Strategies for SARS-CoV-2 diagnosis: A comprehensive review. *Diagnostic Microbiology and Infectious Disease* **101**:115432. DOI:10.1016/j.diagmicrobio.2021.115432.

Delaigle, A., P. Hall, and J. Wishart. 2014. New approaches to nonparametric and semiparametric regression for univariate and multivariate group testing data. *Biometrika* **101**:567-85. DOI:10.1093/biomet/asu025.

Dhand, N., W. Johnson, J. Toribio. 2010. A Bayesian approach to estimate OJD prevalence from pooled fecal samples of variable pool size. *Journal of Agricultural, Biological, and Environmental Statistics* **15**:452-73. DOI:10.1007/s13253-010-0032-8.

Ding, J., W. Xiong. 2016. A new estimator for a population proportion using group testing. *Communications in Statistics - Simulation and Computation* **45**:101-14. DOI:10.1080/03610918.2013.854909.

Domelen, D. 2020. pooling: Fit poolwise regression models. R package version 1.1.2. DOI:10.1002/sim.3823.

Dorfman, R. 1943. The detection of defective members of large populations. *Annals of Mathematical Statistics* **14**:436-40. DOI:10.1214/aoms/1177731363.

Gastwirth, J., W. Johnson. 1994. Screening with cost-effective quality control: Potential applications to HIV and drug testing. *Journal of the American Statistical Association* **89**:972-81. DOI:10.1080/01621459.1994.10476831.

Ghosh, S., R. Agarwal, M. Rehan, S. Pathak, P. Agarwal, Y. Gupta, S. Consul, N. Gupta, Ritika, R. Goenka, A. Rajwade, and M. Gopalkrishnan. 2021. A compressed sensing approach to pooled RT-PCR testing for COVID-19 detection. *IEEE Open Journal of Signal Processing* **2**:248-64. DOI:10.1109/OJSP.2021.3075913.

Haber, G., Y. Malinovsky, P. Albert. 2018. Sequential estimation in the group testing problem. *Sequential Analysis* **37**:1-17. DOI:10.1080/07474946.2017.1394716.

Hepworth, G., R. Watson. 2009. Debiased estimation of proportions in group testing. *Journal of the Royal Statistical Society, Series C (Appl. Stat.)* **58**:105-21. DOI:10.1111/j.1467-9876.2008.00639.x.

Hitt, B., C. Bilder, F. Schaarschmidt, B. Biggerstaff, C. McMahan, J. Tebbs, B. Zhang, M. Black, P. Hou, P. Chen. 2020. binGroup2: Identification and estimation using group testing. R package version 1.0.2. https://CRAN.R-project.org/package=binGroup2/.

Hogan, C., M. Sahoo, B. Pinsky. 2020. Sample pooling as a strategy to detect community transmission of SARS-CoV-2. *Journal of the American Medical Association* **323**:1967-9. DOI:10.1001/jama.2020.5445.

Hourfar, M., C. Jork, V. Schottstedt, M. Weber-Schehl, V. Brixner, M. Busch, G. Geusendam, K. Gubbe, C. Mahnhardt, W. Mayr-Wohlfar, et al. 2008. Experience of German Red Cross blood donor services with nucleic acid testing: Results of screening more than 30 million blood donations for human immunodeficiency virus, hepatitis C virus, and hepatitis B virus. *Transfusion* **48**:1558-66. DOI:10.1111/j.1537-2995.2008.01718.x.

Huang, X. 2009. An improved test of latent-variable model misspecification in structural measurement error models for group testing data. *Statistics in Medicine* **28**:3316-27. DOI:10.1002/sim.3698.

Huang, X., W. Warasi. 2017. Maximum likelihood estimators in regression models for error-prone group testing data. *Scandinavian Journal of Statistics* **44**:918-31. DOI:10.1111/sjos.12282.

Kainkaryam, R., P. Woolf. 2009. Pooling in high-throughput drug screening. *Current Opinion in Drug Discovery and Development* **12**:339-50.

Katholi, C., J. Barker. 2010. PoolScreen 2.0 user's manual. University of Alabama, Birmingham, USA. https://sites.uab.edu/statgenetics/software/.

Kim, H., M. Hudgens, J. Dreyfuss, D. Westreich, C. Pilcher. 2007. Comparison of group testing algorithms for case identification in the presence of testing error. *Biometrics* **63**:1152-63. DOI:10.1111/j.1541-0420.2007.00817.x.

Kim, H., M. Hudgens. 2009. Three-dimensional array-based group testing algorithms. *Biometrics* **65**:903-10. DOI:10.1111/j.1541-0420.2008.01158.x.

Lin, Y., C. Yu, T. Liu, C. Chang, W. Chen. 2021. Positively correlated samples save pooled testing costs *IEEE Transactions on Network Science and Engineering* **8**. DOI:10.1109/TNSE.2021.3081759.

Lindan, C., M. Mathur, S. Kumta, H. Jerajani, A. Gogate, J. Schachter, J. Moncada. 2005. Utility of pooled urine specimens for detection of *Chlamydia trachomatis* and *Neisseria gonorrhoeae* in men attending public sexually transmitted infection clinics in Mumbai, India, by PCR. *Journal of Clinical Microbiology* **43**:1674-7. DOI:10.1128/JCM.43.4.1674-7.2005.

Liu, A., C. Liu, Z. Zhang, P. Albert. 2012. Optimality of group testing in the presence of misclassification. *Biometrika* **99**:245-51. DOI:10.1093/biomet/asr064.

Liu, Y., C. McMahan, J. Tebbs, C. Gallagher, C. Bilder. 2021. Generalized additive regression for group testing data. *Biostatistics* **22**:873-89. DOI:10.1093/biostatistics/kxaa003.

Louis, T. 1982. Finding the observed information matrix when using the EM algorithm. *Journal of the Royal Statistical Society, Series B (Methodology)* **44**:226-33. DOI:10.1111/j.2517-6161.1982.tb01203.x.

McLure, A. 2021. PoolTestR: Prevalence and regression for pool-tested (group-tested) data. R pacakage version 0.1.1. https://cran.r-project.org/web/packages/PoolTestR/index.html.

McMahan, C., J. Tebbs, C. Bilder. 2012a. Informative Dorfman screening. *Biometrics* **68**:287-96. DOI:10.1111/j.1541-0420.2011.01644.x.

McMahan, C., J. Tebbs, C. Bilder. 2012b. Two-dimensional informative array testing. *Biometrics* **68**:793-804. DOI:10.1111/j.1541-0420.2011.01726.x.

McMahan, C., J. Tebbs, C. Bilder. 2013. Regression models for group testing data with pool dilution effects. *Biostatistics* **14**:284-98. DOI:10.1093/biostatistics/kxs045.

McMahan, C., J. Tebbs, T. Hanson, C. Bilder. 2017. Bayesian regression for group testing data. *Biometrics* **73**:1443-52. DOI:10.1111/biom.12704.

Mutesa, L., P. Ndishimye, Y. Butera, J. Souopgui, A. Uwineza, R. Rutayisire, E. Ndoricimpaye, E. Musoni, N. Rujeni, T. Nyatanyi, et al. 2021. A pooled testing strategy for identifying SARS-CoV-2 at low prevalence. *Nature* **589**:276-80. DOI:10.1038/s41586-020-2885-5.

Nguyen, N., E. Bish, H. Aprahamian. 2018. Sequential prevalence estimation with pooling andcontinuous test outcomes. *Statistics in Medicine* **37**:2391-2426. DOI:10.1002/sim.7657.

Pilcher, C., S. Fiscus, T. Nguyen, E. Foust, L. Wolf, D. Williams, R. Ashby, J. O'Dowd, J. McPherson, B. Stalzer, et al. 2005. Detection of acute infections during HIV testing in North Carolina. *New England Journal of Medicine* **352**:1873-83. DOI:10.1056/NEJMoa042291.

Pilcher, C., D. Westreich, M. Hudgens. 2020. Group testing for severe acute respiratory syndrome- coronavirus 2 to enable rapid scale-up of testing and real-time surveillance of incidence. *Journal of Infectious Diseases* **222**:903-9. DOI:10.1093/infdis/jiaa378.

Quinn, T., R. Brookmeyer, R. Kline, M. Shepherd, R. Paranjape, S. Mehendale, D. Gadkari, R. Bollinger. 2000. Feasibility of pooling sera for HIV-1 viral RNA to diagnose acute primary HIV-1 infection and estimate HIV incidence. *AIDS* **14**:2751-7. DOI:10.1097/00002030-200012010-00015.

`R` Core Team. 2021. A language and environment for statistical computing. `R` Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org.

Speybroeck, N., C. Williams, K. Lafia, B. Devleesschauwer, D. Berkvens. 2012. Estimating the prevalence of infections in vector populations using pools of samples. *Medical and Veterinary Entomology* **26**:361-71. DOI:10.1111/j.1365-2915.2012.01015.x.

Stramer, S., E. Notari, D. Krysztof, R. Dodd. 2013. Hepatitis B virus testing by minipool nucleic acid testing: Does it improve blood safety? *Transfusion* **53**:2449-58. DOI:10.1111/trf.12213.

Tebbs, J., C. McMahan, C. Bilder. 2013. Two-stage hierarchical group testing for multiple infections with application to the Infertility Prevention Project. *Biometrics* **69**:1064-73. DOI:10.1111/biom.12080.

Van, T., J. Miller, D. Warshauer, E. Reisdorf, D. Jerrigan, R. Humes, P. Shult. 2012. Pooling nasopharyngeal/throat swab specimens to increase testing capacity for influenza viruses by PCR. *Journal of Clinical Microbiology* **50**:891-6. DOI:10.1128/JCM.05631-11.

Vansteelandt, S., E. Goetghebeur, T. Verstraeten. 2000. Regression models for disease prevalence with diagnostic tests on pools of serum samples. *Biometrics* **56**:1126-33. DOI:10.1111/j.0006-341x.2000.01126.x.

Verdun, C., T. Fuchs, P. Harar, D. Elbrachter, D. Fischer, J. Berner, P. Grohs, F. Theis, and F. Krahmer. 2021. Group testing for SARS-CoV-2 allows for up to 10-fold efficiency increase across realistic scenarios and testing strategies. *Frontiers in Public Health* **9**:583377. DOI:10.3389/fpubh.2021.583377

Wang, D., C. McMahan, M. Gallagher, B. Kulasekera. 2014. Semiparametric group testing regression models. *Biometrika* **101**:587-98. DOI: 10.1093/biomet/asu007.

Wang, D., C. McMahan, C. Gallagher. 2015. A general parametric regression framework for group testing data with dilution effects. *Statistics in Medicine* **34**:3606-21. DOI:10.1002/sim.6578.

Warasi, M. 2021. groupTesting: Simulating and modeling group (pooled) testing data. R package version 1.0.0. https://cran.r-project.org/web/packages/groupTesting.

Westreich, D., K. Diepstra, R. Max. 2020. Guidance on pooling of samples for SARS-CoV-2. Accessed October 25, 2021.
https://gillingscovid19.unc.edu/whitepaper/pooling-samples-covid-19.

Xie, M. 2001. Regression analysis of group testing samples. *Statistics in Medicine* **20**:1957-69. DOI:10.1002/sim.817.

Zhang, B., C. Bilder, J. Tebbs. 2013. Group testing regression model estimation when case identification is a goal. *Biometrical Journal* **55**:173-89. DOI:10.1002/bimj.201200168.

Zhang, B., C. Bilder, B. Biggerstaff, F. Schaarschmidt, B. Hitt. 2018. binGroup: Evaluation and experimental design for binomial group testing. R package version 2.2-1. https://CRAN.R-project.org/package=binGroup/.

Zhang, W., A. Liu, Q. Li, P. Albert. 2019. Incorporating retesting outcomes for estimation of disease prevalence. *Statistics in Medicine* **39**:687-97. DOI:10.1002/sim.8439.

Appendix A. Distribution used in Gibbs sampling

The individual true statuses $\widetilde{Y}_i$, conditional on $\mathbf{Z}, \widetilde{\mathbf{Y}}_{-i}, \boldsymbol{\beta}^{(d)}$, are sampled from Bernoulli(p_i^*) for the EM algorithm in Section 2, where $p_i^* = \zeta_1^i/(\zeta_1^i + \zeta_0^i)$,

$$\begin{aligned}
\zeta_1^i &= g(\mathbf{x}_i^T\boldsymbol{\beta}^{(d)}) \prod_{j\in\mathcal{A}_i} S_{e_j}^{Z_j}(1-S_{e_j})^{1-Z_j} \\
\zeta_0^i &= (1-g(\mathbf{x}_i^T\boldsymbol{\beta}^{(d)})) \prod_{j\in\mathcal{A}_i} \{S_{e_j}^{Z_j}(1-S_{e_j})^{1-Z_j}\}^{I(\sum_{i'\in\mathcal{P}_{ij}} \widetilde{Y}_i'>0)} \\
&\quad \times\{(1-S_{p_j})^{Z_j} S_{p_j}^{1-Z_j}\}^{I(\sum_{i'\in\mathcal{P}_{ij}} \widetilde{Y}_i'=0)},
\end{aligned}$$

$\mathcal{A}_i = \{j : i \in \mathcal{P}_j\}$, and $\mathcal{P}_{ij} = \{i' \in \mathcal{P}_j : i' \neq i\}$. Note that McMahan et al. (2017) used this distribution for posterior sampling in their Bayesian group testing model.

Appendix B. Covariance matrix estimation

The observed-information matrix in Equation (5) is

$$\mathcal{I}(\boldsymbol{\beta}) = -\frac{\partial^2 Q(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}^T} - \text{cov}\left\{ \frac{\partial \ell_C(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} \middle| \mathbf{Z}, \boldsymbol{\beta} \right\},$$

where $Q(\boldsymbol{\beta}) = E[\ln L_C(\boldsymbol{\beta}|\mathbf{Z}, \widetilde{\mathbf{Y}})|\mathbf{Z}, \boldsymbol{\beta}]$, $\ell_C(\boldsymbol{\beta}) = \ln L_C(\boldsymbol{\beta}|\mathbf{Z}, \widetilde{\mathbf{Y}})$, $L_C(\boldsymbol{\beta}|\mathbf{Z}, \widetilde{\mathbf{Y}}) = \pi(\widetilde{\mathbf{Y}}; \boldsymbol{\beta})\pi(\mathbf{Z}|\widetilde{\mathbf{Y}})$ is the complete likelihood, and $\pi(\widetilde{\mathbf{Y}}; \boldsymbol{\beta})$ and $\pi(\mathbf{Z}|\widetilde{\mathbf{Y}})$ are given in Equations (2) and (3).

For ease of presentation, let $\mathbf{x}_i = (x_{i0}, x_{i1}, ..., x_{ir})^T$ and $p_i = g(\mathbf{x}_i^T\boldsymbol{\beta})$. The intercept model in Section 2 can be found by setting $x_{i0} = 1$, for $i = 1, 2, ..., N$. Then $\partial^2 Q(\boldsymbol{\beta})/(\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}^T)$ is an $(r+1) \times (r+1)$ matrix. The (s, t)th component of the matrix is

$$\frac{\partial^2 Q(\boldsymbol{\beta})}{\partial\beta_s\partial\beta_t} = -\sum_{i=1}^{N} \left[\left\{ \frac{\mu_i}{p_i^2} + \frac{(1-\mu_i)}{(1-p_i)^2} \right\} \frac{\partial p_i}{\partial\beta_s}\frac{\partial p_i}{\partial\beta_t} + \frac{(p_i - \mu_i)}{p_i(1-p_i)}\frac{\partial^2 p_i}{\partial\beta_s\partial\beta_t} \right],$$

where $\mu_i = E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}]$ and

$$\frac{\partial p_i}{\partial\beta_s} = g'(\mathbf{x}_i^T\boldsymbol{\beta})x_{is}$$

$$\frac{\partial^2 p_i}{\partial\beta_s\partial\beta_t} = g''(\mathbf{x}_i^T\boldsymbol{\beta})x_{is}x_{it},$$

for $s = 0, 1, ..., r$ and $t = 0, 1, ..., r$. The expectation $E[\widetilde{Y}_i|\mathbf{Z}, \boldsymbol{\beta}]$ is estimated by using the Gibbs samples of $\widetilde{Y}_i$ from its conditional distribution as described in Section 2. Note that $g'(u)$ and $g''(u)$ are the first and second derivatives of $g(u)$ with respect to u. For the logit link, $g(u) = e^u/(1+e^u)$, $g'(u) = g(u)(1-g(u))$, and $g''(u) = g'(u)(1-2g(u))$. Similarly, these expressions can be easily found for other link functions as well. Combining the information above, one finds the first term of $\mathcal{I}(\boldsymbol{\beta})$.

To derive the second term of $\mathcal{I}(\boldsymbol{\beta})$, we have $\ell_C(\boldsymbol{\beta})$ up to a constant as

$$\ell_C(\boldsymbol{\beta}) = \sum_{i=1}^{N} \left[\widetilde{Y}_i \ln g(\mathbf{x}_i^T\boldsymbol{\beta}) + (1-\widetilde{Y}_i)\ln(1-g(\mathbf{x}_i^T\boldsymbol{\beta})) \right].$$

Then the vector of the first derivatives of $\ell_C(\boldsymbol{\beta})$ can be expressed as

$$\mathbf{V} = \frac{\partial\ell_C(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = \sum_{i=1}^{N} \frac{(\widetilde{Y}_i - p_i)g'(\mathbf{x}_i^T\boldsymbol{\beta})\mathbf{x}_i}{p_i(1-p_i)}.$$

Because finding an explicit expression of the covariance of $\mathbf{V}$ can be very difficult, we again approximate it using the Gibbs sampling. To briefly discuss this, suppose that $\mathbf{Y}^{(h)} = (\widetilde{Y}_1^{(h)}, \widetilde{Y}_2^{(h)}, ..., \widetilde{Y}_N^{(h)})^T$ is the hth Gibbs sample of $\mathbf{Y}$ and $\mathbf{V}^{(h)}$ is $\mathbf{V}$ evaluated at $\mathbf{Y}^{(h)}$, for $h = 1, 2, ..., H$. When H is large, the sample covariance of $\mathbf{V}$ calculated from $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, ..., \mathbf{V}^{(H)}$

is an estimate of $\text{cov}\{\partial\ell_C(\boldsymbol{\beta})/\partial\boldsymbol{\beta}|\mathbf{Z},\boldsymbol{\beta}\}$. Using the first and second term of $\mathcal{I}(\boldsymbol{\beta})$, we find an estimate of $\mathcal{I}(\boldsymbol{\beta})^{-1}$ at the MLE $\boldsymbol{\beta} = \widehat{\boldsymbol{\beta}}$. The simulation results in Table 3 show that this approach to estimating the covariance matrix $\mathcal{I}(\boldsymbol{\beta})^{-1}$ is highly accurate.

Appendix C. Simulation evidence

To assess the quality of our estimates, we perform a simulation study with initial pooled testing (IPT), 2-stage hierarchical testing (H2), 3-stage hierarchical testing (H3), 2-stage array testing without master pool (A2), and 2-stage array testing with master pool (A2M); Kim et al. (2007) described how pooled testing is performed according to these protocols. Individual testing (IND) is also used for benchmarking purposes.

For individual true statuses, we assume the models

M1. $\text{logit}\{\text{pr}(\widetilde{Y}_i = 1|x_{i1})\} = \beta_0 + \beta_1 x_{i1}$, $\boldsymbol{\beta} = (\beta_0, \beta_1)' = (-3, 0.5)'$;

M2. $\text{logit}\{\text{pr}(\widetilde{Y}_i = 1|x_{i1}, x_{i2})\} = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2}$, $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2)' = (-3, 1, -1)'$;

M3. $\text{logit}\{\text{pr}(\widetilde{Y}_i = 1|x_{i1})\} = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i1}^2$, $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2)' = (-3, 2, -0.5)'$;

where $x_{i1} \sim \mathcal{N}(0, 1)$ and $x_{i2} \sim \text{Bernoulli}(0.5)$. We simulate $N = 2700$ individual true statuses, $\widetilde{Y}_i$'s, from the models using `R` as shown below with model M2. Note that McMahan, Tebbs, and Bilder (2013) used the models for simulation with different parameter configurations.

```
> ## Individual true statues with model M2:
> param <- c(-3,1,-1)
> x <- cbind(1, rnorm(n=2700,mean=0,sd=1), rbinom(n=2700,size=1,prob=0.5))
> colnames(x) <- c("Intercept", "x1", "x2")
> pReg <- exp(x%*%param)/(1+exp(x%*%param))
```

For IPT, H2, and H3, we use the initial pool size 6. To resolve the positive pools, H2 uses individual testing in the next stage while H3 uses an intermediate stage for pooled testing with pool size 2 before implementing individual testing. For A2 and A2M, we use 6×6 arrays and resolve the positive pools as suggested in Kim et al. (2007). These pool sizes are chosen so that the number of individuals, $N = 2700$, can be evenly distributed to the initial pools. The assay sensitivity and specificity that we use are $S_{e_j} = 0.95$ and $S_{p_j} = 0.98$, which are consistent with the application in Section 4.2. The simulation is conducted using our functions `hier.gt.simulation` and `array.gt.simulation` as follows.

```
> library(groupTesting)

> ## Individual testing:
> IND.gtData <- hier.gt.simulation(N=2700, p=pReg, S=1, psz=1,
+                             Se=0.95, Sp=0.98, assayID=1)$gtData

> ## Initial pooled testing:
> IPT.gtData <- hier.gt.simulation(N=2700, p=pReg, S=1, psz=6,
+                             Se=0.95, Sp=0.98, assayID=1)$gtData

> ## 2-stage hierarchical:
> H2.gtData <- hier.gt.simulation(N=2700, p=pReg, S=2, psz=c(6,1),
+                             Se=c(0.95,0.95), Sp=c(0.98,0.98),
+                             assayID=c(1,1))$gtData

> ## 3-stage hierarchical:
> H3.gtData <- hier.gt.simulation(N=2700, p=pReg, S=3, psz=c(6,2,1),
+                             Se=c(0.95,0.95,0.95), Sp=c(0.98,0.98,0.98),
+                             assayID=c(1,1,1))$gtData

> ## 2-stage array testing without master pool:
> A2.gtData <- array.gt.simulation(N=2700, p=pReg, protocol="A2", n=6,
+                             Se=c(0.95,0.95), Sp=c(0.98,0.98),
+                             assayID=c(1,1))$gtData

> ## 2-stage array testing with master pool:
> A2M.gtData <- array.gt.simulation(N=2700, p=pReg, protocol="A2M", n=6,
+                             Se=c(0.95,0.95,0.95), Sp=c(0.98,0.98,0.98),
+                             assayID=c(1,1,1))$gtData
```

The design matrix `x` and each of the data `IND.gtData`, `IPT.gtData`, `H2.gtData`, `H3.gtData`, `A2.gtData`, and `A2M.gtData` can be input into `glm.gt` to find the MLE of the parameter $\boldsymbol{\beta}$ exactly as shown in Sections 4.1 and 4.2. For example, the model can be fit with `x` and `A2M.gtData` as follows, where `g`, `dg`, and `d2g` are as in Section 4.1.

```
> A2M.res <- glm.gt(beta0=c(0,0,0), gtData=A2M.gtData, X=x,
+                   g=g, dg=dg, d2g=d2g, covariance=TRUE,
+                   nburn=3000, ngit=10000, tol=1e-03, conf.level=0.95)
```

We repeat the above simulation to generate 500 data sets under each of the pooling configurations. The estimation results are summarized in Table 3. We show estimation bias (Bias), average standard error (SE), standard deviation (SD) of the 500 MLEs, average elapsed time (in seconds), and average number of tests expended. We also show empirical coverage probability (Cov) of the 95% Wald confidence intervals. The calculation is performed in an Intel 3.6GHz 32GB RAM machine.

Here are some remarks on the estimation results.

- Overall, the maximum likelihood estimates are unbiased.
- SD/SE is close to 1; i.e., the standard errors are estimated well.
- The empirical coverage probabilities are close to the nominal level 0.95.
- Function `glm.gt` offers efficient computing; the overall elapsed time is well below 60 seconds.

Table 1: Salient features of `glm.gt`.

(a) Fits the GLM models and provides the MLE of $\boldsymbol{\beta}$.
(b) Calculates the covariance matrix at the MLE $\widehat{\boldsymbol{\beta}}$.
(c) Provides the Wald confidence intervals.
(d) Can accommodate any group testing data.
(e) Does not require any information about the group testing protocol used.
(f) Can accommodate any link function g from the GLM family.
(g) Uses the finite difference approximation when gradients are not supplied.

Table 2: An example of the data input used in `glm.gt`.

	Z	psz	Se	Sp	Assay	Mem1	Mem2	Mem3	Mem4	Mem5	Mem6
Pool:1	1	6	0.90	0.92	1	1	2	3	4	5	6
Pool:2	0	6	0.90	0.92	1	7	8	9	10	11	12
Pool:3	1	2	0.95	0.96	2	1	2	-9	-9	-9	-9
Pool:4	0	2	0.95	0.96	2	3	4	-9	-9	-9	-9
Pool:5	1	2	0.95	0.96	2	5	6	-9	-9	-9	-9
Pool:6	0	1	0.92	0.90	3	1	-9	-9	-9	-9	-9
Pool:7	1	1	0.92	0.90	3	2	-9	-9	-9	-9	-9
Pool:8	0	1	0.92	0.90	3	5	-9	-9	-9	-9	-9
Pool:9	0	1	0.92	0.90	3	6	-9	-9	-9	-9	-9

Table 3: Regression estimation results from 500 simulated data sets. Also, the reduction in the average number of tests, when compared to individual testing, is shown in parentheses. The margin of error for the coverage probability estimates with 99% confidence level is 0.03.

Model	Parameter		IND	IPT	H2	H3	A2	A2M
M1	$\beta_0 = -3.00$	Bias (Cov)	−0.01(0.97)	−0.02(0.96)	−0.01(0.95)	−0.01(0.95)	−0.01(0.95)	−0.01(0.96)
		SD (SE)	0.12(0.12)	0.14(0.14)	0.10(0.10)	0.10(0.10)	0.10(0.10)	0.11(0.11)
	$\beta_1 = 0.50$	Bias (Cov)	0.00(0.94)	0.00(0.93)	−0.01(0.95)	−0.01(0.96)	0.00(0.94)	0.00(0.95)
		SD (SE)	0.11(0.11)	0.22(0.22)	0.10(0.10)	0.09(0.10)	0.10(0.09)	0.10(0.10)
	Avg. elapsed time		16	28	10	13	16	24
	Avg. number of tests		2700	450	1200(56%)	1081(60%)	1207(55%)	1091(60%)
M2	$\beta_0 = -3.00$	Bias (Cov)	0.00(0.95)	−0.04(0.95)	−0.01(0.95)	−0.01(0.95)	−0.02(0.95)	−0.01(0.93)
		SD (SE)	0.17(0.17)	0.25(0.25)	0.15(0.15)	0.15(0.15)	0.15(0.14)	0.16(0.15)
	$\beta_1 = 1.00$	Bias (Cov)	−0.01(0.95)	0.01(0.96)	0.00(0.93)	0.00(0.96)	0.01(0.96)	0.00(0.94)
		SD (SE)	0.13(0.13)	0.19(0.20)	0.11(0.11)	0.11(0.11)	0.11(0.11)	0.11(0.11)
	$\beta_2 = -1.00$	Bias (Cov)	−0.03(0.96)	−0.04(0.97)	−0.01(0.97)	−0.02(0.95)	−0.01(0.96)	0.00(0.96)
		SD (SE)	0.25(0.26)	0.50(0.50)	0.22(0.23)	0.23(0.22)	0.23(0.22)	0.22(0.22)
	Avg. elapsed time		21	44	13	17	20	31
	Avg. number of tests		2700	450	1155(57%)	1044(61%)	1182(56%)	1046(61%)
M3	$\beta_0 = -3.00$	Bias (Cov)	−0.02(0.96)	−0.06(0.95)	−0.01(0.94)	−0.02(0.95)	−0.01(0.95)	−0.01(0.96)
		SD (SE)	0.17(0.17)	0.30(0.30)	0.15(0.14)	0.14(0.14)	0.14(0.14)	0.14(0.14)
	$\beta_1 = 2.00$	Bias (Cov)	0.04(0.93)	0.13(0.94)	0.03(0.94)	0.03(0.96)	0.03(0.95)	0.03(0.96)
		SD (SE)	0.32(0.31)	0.66(0.66)	0.26(0.26)	0.24(0.24)	0.25(0.24)	0.26(0.25)
	$\beta_2 = -0.50$	Bias (Cov)	−0.02(0.94)	−0.06(0.94)	−0.01(0.94)	−0.01(0.95)	−0.01(0.95)	−0.02(0.94)
		SD (SE)	0.14(0.14)	0.29(0.28)	0.12(0.12)	0.11(0.11)	0.12(0.11)	0.13(0.12)
	Avg. elapsed time		21	51	14	18	22	33
	Avg. number of tests		2700	450	1487(45%)	1351(50%)	1388(49%)	1345(50%)