binGroup2: Statistical Tools for Infection Identification via Group Testing

by Christopher R. Bilder, Brianna D. Hitt, Brad J. Biggerstaff, Joshua M. Tebbs, and Christopher S. McMahan

Abstract Group testing is the process of testing individual items as an amalgamation, rather than separately, to determine the binary status for each item. Its use has been especially important during the COVID-19 pandemic through testing specimens for SARS-CoV-2. The adoption of group testing for this and many other applications is because members of a negative testing group can be declared negative with potentially only one test. This subsequently leads to significant increases in laboratory testing capacity. Whenever a group testing algorithm is put into practice, it is critical for laboratories to understand the algorithm's operating characteristics, such as the expected number of tests. Our paper presents the **binGroup2** package that provides the statistical tools for this purpose. This R package is the first to address the identification aspect of group testing for a wide variety of algorithms. We illustrate its use through current COVID-19 and chlamydia/gonorrhea applications of group testing.

Introduction

The COVID-19 pandemic has shown the need for accessible, fast, and reliable testing for severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) and for infections in general. To meet this need, many laboratories have turned to the use of group testing. Group testing, also known as pooled testing, specimen pooling, batch testing, and bulk testing, involves combining portions of multiple specimens from different individuals into a "group" and testing this group as if it were a single specimen. If the group tests negative, all individuals represented within it generally can be declared negative. Thus, for a group of size 10, it takes only one test to determine all 10 individuals are infection free, whereas it would take 10 tests if each specimen was tested separately. Alternatively, if a group tests positive, retesting in some form is needed to determine who is positive or negative within the group. Dorfman (1943) proposed the original retesting method that involved simply retesting each group member separately using the remaining portions of their specimens. Since this seminal paper, many other group testing algorithms have been proposed. These algorithms are categorized into hierarchical and non-hierarchical approaches corresponding to whether specimens are tested in non-overlapping or overlapping groups, respectively, at each stage of the algorithm (Hitt et al. 2019).

A large number of research papers (e.g., Abdalhamid et al. 2020; Hogan et al. 2020; Barathidasan et al. 2022), articles in the news media (e.g., Abdelmalek 2020; Mandavilli 2020; Anthes 2022), and Emergency Use Authorizations given by the US Food and Drug Administration (LabCorp 2021; Verily Life Sciences 2021; Yale University 2022) have shown group testing is one of the important tools available to mitigate the crisis caused by COVID-19. Our new **binGroup2** package for the R software environment provides the statistical tools that researchers and laboratories need for group testing. This package is available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=binGroup2`. By its name, one can surmise that it is the second, large-scale implementation of the package. The **binGroup** package (Bilder et al. 2010b) focused on estimation using the unique type of data that arises through group testing. Its applications ranged from estimating an overall infection prevalence to estimating a regression model for individual-specific infection probabilities as a function of risk factors. Bilder et al. (2010b) encouraged researchers to provide new functions so that there would be one main package for group testing in R. This resulted in new functions for estimation and a few functions focused on the identification aspect of group testing (i.e., determine a positive/negative outcome for each individual). Over time, these additions resulted in many different usage styles among functions, making use of the package inconsistent and more difficult. The new **binGroup2** package was created to align function syntax and, most importantly, to incorporate a new suite of functions for identification. These new functions provide the information that laboratories need to increase their testing capacity to its fullest potential.

Our paper focuses on these identification functions of **binGroup2** because they represent the main new contribution of the package. There are two other R packages on CRAN that also examine the identification aspect of group testing but for different settings. First, the **gtcorr** package (Lendle 2011) examines hierarchical and non-hierarchical algorithms for a homogeneous population when there is a constant correlation among individuals within a group. This package implements the work of Lendle et al. (2012) that uses single-infection assays only. Second, the **mMPA** package (Liu and Xu 2018) examines three hierarchical algorithms, where two are for homogeneous populations and one is for heterogeneous populations. This package implements the work of Liu et al. (2017) for single-infection

assays that return an additive count outcome (e.g., viral load) when applied to a group. Both **gtcorr** and **mMPA** are useful for their specialized situations. Our package encompasses a total of 27 different algorithms that include homogeneous/heterogeneous populations and hierarchical/non-hierarchical algorithms. These algorithms are constructed for the most common setting of a binary test response (positive/negative) from independent individuals within a group. We allow for the possibility of testing error and single or multiple-infection assays. Our package implements the work of more than ten papers on the identification aspect of group testing (e.g., Kim et al. 2007; McMahan et al. 2012b; Hou et al. 2020).

An outline of our paper is as follows. The next section provides a review of group testing algorithms that are commonly used by laboratories. The third section discusses how to calculate important operating characteristics, such as the expected number of tests, for a group testing algorithm by using **binGroup2**. This section also demonstrates how one can choose the most efficient implementation of an algorithm. Finally, we conclude with an overview of changes to estimation functions and present future extensions of the package. Throughout our text, we phrase our discussion in the context of testing humans for infectious diseases. Applications of group testing for this purpose outside of COVID-19 include testing for influenza (Van et al. 2012), gonorrhea (Ando et al. 2021), HIV (Kim et al. 2014), and West Nile virus (American Red Cross 2022). The package can also be used in a wide variety of other areas where group testing is implemented, such as infectious disease testing for farm animals (Nebraska Veterinary Diagnostic Center 2022), detection of human exposure to pollutants (Thai et al. 2020), determination of virus presence in insect carriers (Zhao and Rosa 2020), development of new pharmaceuticals (Salzer et al. 2016), and security of computer networks (Thai 2011).

Algorithms

Laboratories select a group testing algorithm by examining 1) the expected number of tests needed to make a positive/negative determination for every individual, 2) the expected accuracy of the positive/negative determinations, 3) the information available on individuals tested, and 4) the ease of implementation. The **binGroup2** package provides computations needed to address 1) through 3), while laboratories need to address 4) relative to their work environment. Overall, there is not one group testing algorithm that is best for all situations, which has resulted in many different hierarchical and non-hierarchical algorithms being used in practice.

Hierarchical algorithms

Hierarchical group testing algorithms involve testing a group and splitting its members into smaller, non-overlapping sub-groups for retesting if the original group tests positive. These sub-groups are split further whenever they test positive. If a group/sub-group tests negative, its members are declared negative, so that no further testing is performed upon them. Overall, testing can continue until each group member is tested individually.

A recent example of a hierarchical algorithm comes from Lohse et al. (2020) to test for SARS-CoV-2. Individuals were put into non-overlapping groups of size 30. If a group tested positive, three non-overlapping groups of size 10 were formed from its members. If one of these sub-groups tested positive, each of its members were retested separately. This hierarchical algorithm is referred to as a three-stage algorithm because three distinct stages are defined for it.

Hierarchical algorithms can have more or fewer stages than used by Lohse et al. (2020). For example, Abdalhamid et al. (2020) used two stages for SARS-CoV-2 detection by testing in groups of size 5 and subsequently retesting each member of a positive group separately. More than three stages are possible as well, but applications are much rarer due to logistics, larger chance for error, and delay in obtaining the positive/negative individual outcomes. With respect to the last reason, this delay can be significant for nucleic acid amplification tests which can take hours to complete. For example, the Centers for Disease Control and Prevention's assay to detect SARS-CoV-2 takes approximately four hours to complete.

Non-hierarchical algorithms

Non-hierarchical algorithms were developed to minimize the number of retests needed after an initial stage of testing. The most prominent of these algorithms is referred to as array testing (also known as matrix pooling). For this algorithm, specimens are arranged in a grid. Groups are formed by row and by column, and each of these groups are subsequently tested. Individual specimens that lie at the intersections of positive testing rows and positive testing columns are retested separately in a second stage to determine a positive/negative outcome for each of them. All other specimens are

declared negative. Because the accuracy of infectious disease assays is not perfect, ambiguities may occur, resulting in positive rows (columns) without any positive columns (rows). In those situations, members of positive rows (columns) should be retested separately (Kim et al. 2007). A recent example of an array testing algorithm is the use of a 5×5 array by LabCorp for SARS-CoV-2 detection (LabCorp 2021).

Array testing has a number of variants in application. For example, a master group containing portions of all specimens within an array can be tested first. If this master group is negative, all individuals are quickly declared negative. If the master group is positive, row and column groups are tested as usual.

Additional considerations

The probability an individual has an infection plays a very important role in determining the number of tests needed by any group testing algorithm. For a homogeneous population of individuals tested, we can define p as this probability. Equivalently, this p is the overall infection prevalence. In general, the lower (higher) the p, the lower (higher) the expected number of tests needed for group testing.

In many situations, additional information is available on each individual to be tested. This information can be used to determine an individual-specific probability of infection, say p_i, which can be incorporated into the group testing algorithm to reduce the number of tests (Bilder et al. 2010b; Lewis et al. 2012). For example, testing is performed by public health laboratories across the United States for *Chlamydia trachomatis* (CT) and *Neisseria gonorrhoeae* (NG), the bacteria that lead to chlamydia and gonorrhea, respectively. Clinics collecting specimens also obtain important information on their patients, including recent sexual history and whether a patient has symptoms, and this information can be incorporated into a statistical model to estimate individual-specific probabilities of infection. In general, group testing algorithms that take advantage of this type of additional information are referred to as being "informative" hierarchical/non-hierarchical algorithms (McMahan et al. 2012a; McMahan et al. 2012b).

Group testing algorithms can also be used with multiplex assays. Thus, rather than testing for only one infection, multiple infections can be detected simultaneously. For example, Roche was the first to receive an Emergency Use Authorization for their SARS-CoV-2 and influenza multiplex assay (Roche 2020). Also, the widely adopted Aptima Combo 2 Assay (Hologic 2022) is used for CT and NG detection via group testing. Algorithms with multiplex assays are implemented in the same way as for single-infection assays, but now a group that tests positive for *at least one* infection leads to retesting for *all* infections in the next stage. For example, if a group tests positive for CT and negative for NG in a two-stage hierarchical algorithm, each group member is retested for both bacteria in the second stage. Testing is performed in this manner, rather than retesting for CT only, due to the dedicated testing platforms used.

Identification

Identifying infected individuals is most often the primary goal of infectious disease testing. Prior to implementing a particular group testing algorithm for this purpose, laboratories need to know the expected number of tests. This information is used for planning purposes to make sure enough resources and staff are available to implement testing. Because cost is often directly proportional to the number of tests, the expected number of tests can be used for budgeting purposes as well.

Define T as the number of tests required to determine the positive/negative outcomes for I individuals. These I individuals may be those represented within one initial group for hierarchical testing or in one array for array testing. For the simplest algorithm, two-stage hierarchical testing, suppose a single-infection assay is applied to a homogeneous population. The expected number of tests is

$$E(T) = 1 + I\left[S_e^{(1)} + (1 - S_p^{(1)} - S_e^{(1)})(1-p)^I\right],$$

where $S_e^{(1)}$ and $S_p^{(1)}$ are the first-stage sensitivity and specificity, respectively, of the assay. The optimal testing configuration (OTC) is the group size that minimizes $E(T)/I$, the expected number of tests per individual. This testing configuration represents the most efficient group size for a laboratory. In other words, it is the group size that increases their testing capacity to its fullest potential, because resources saved from its application can be used to test more specimens by the same algorithm. Unfortunately, there is not a closed form expression for the OTC, but grid searches are sufficient to find it.

Expressions for $E(T)$ become much more complicated with other group testing algorithms. This is especially the case for informative algorithms and when multiplex assays are used. The OTC continues to be of great interest for these algorithms as well, but now one needs to optimize within a stage

Table 1: Values for `algorithm` in `opChar1()` and `opChar2()`. A five-stage informative hierarchical algorithm (`ID5`) is also available for two infections. Informative array testing is not included for `opChar2()` because closed-form expressions of operating characteristics have not been proposed in the literature.

algorithm	Description
A2	Array testing
A2M	Array testing with master group
D2	Two-stage hierarchical
D3	Three-stage hierarchical
D4	Four-stage hierarchical
IA2	Informative array testing
ID2	Two-stage, informative hierarchical
ID3	Three-stage, informative hierarchical
ID4	Four-stage, informative hierarchical

(individuals could be put into groups of unequal sizes) and over multiple stages. We refer interested readers to the works of Kim et al. (2007), McMahan et al. (2012a), McMahan et al. (2012a); Black et al. (2015), Bilder et al. (2019), and Hou et al. (2020) for specific expressions and calculation details. These references also provide expressions for accuracy measures. This includes the pooling sensitivity, the probability a truly positive individual is found to be positive from the group testing algorithm, and the pooling specificity, the probability a truly negative individual is found to be negative from the group testing algorithm. These probabilities are not necessarily equal to an assay's stated sensitivity and specificity because individuals are tested in one or more groups. For example, the pooling sensitivity for a two-stage hierarchical algorithm can be shown to be $S_e^{(1)} S_e^{(2)}$, where $S_e^{(2)}$ is the sensitivity of the assay at the second stage (Johnson et al. 1991, Kim et al. 2007, and Hitt 2020).

Main functions

There are two main sets of functions in **binGroup2** used for identification. First, the `opChar1()` and `opChar2()` functions calculate operating characteristics for a group testing algorithm. These functions are for single-infection and two-infection assays, where the number in the name designates the number of infections. Calculations for three or more infection assays are discussed in the concluding section of the paper. The syntax for `opChar1()` is

```
opChar1(algorithm, p = NULL, probabilities = NULL, Se = 0.99,
    Sp = 0.99, hier.config = NULL, rowcol.sz = NULL, alpha = 2,
    a = NULL, print.time = TRUE, ...)
```

The required `algorithm` argument specifies the chosen group testing algorithm. Possible values for this argument are listed in Table 1. For example, a value of `"D2"` indicates two-stage hierarchical testing in a homogeneous population (the "D" is for its originator, Robert Dorfman). The remaining arguments are dependent on the algorithm chosen or are optional. To indicate the probability of infection, one value can be given for `p` or a vector of potentially different probabilities can be given for `probabilities`. Rather than providing a specific vector of probabilities, the `p` and `alpha` arguments can be used together for informative group testing algorithms to specify a beta distribution from which the expected values of order statistics are found. For this case, `p` represents the expected value of a beta random variable and `alpha` represents a shape parameter. This type of specification is helpful when probabilities of infection can be characterized well by a beta distribution.

For hierarchical algorithms, a group membership matrix (Bilder et al. 2019) must be provided for `hier.config` to detail the testing of each individual. In this matrix, the rows correspond to the stages of testing, the columns correspond to each individual to be tested, and the cell values specify the group number of each individual at each stage. We provide an example of its use shortly. For non-hierarchical algorithms, the `rowcol.sz` argument must be provided for the row and column size of a square array.

Additional arguments include `Se` and `Sp` for the sensitivity and specificity of the assay, respectively. If a single value is given for one of these arguments, this value is used for each stage. Otherwise, a vector of values can be given in order of stage. The `a` argument specifies individuals for which to compute accuracy measures. By default, accuracy measures are computed for all individuals. Finally, `print.time` allows users to turn off information regarding the duration of calculations.

The `opChar2()` function follows a similar syntax so we do not provide it here. The main difference

involves p becoming a vector of joint probabilities of infection. For example, `p = (0.90,0.03,0.02,0.05)` represents $(p_{--}, p_{+-}, p_{-+}, p_{++})$, where p_{ab} is the probability of being positive/negative $(+/-)$ for infections a and b. Also, the `probabilities` argument value becomes a $4 \times I$ matrix of these probabilities. Similarly, the `alpha` argument becomes the parameter vector for a Dirichlet distribution.

The second set of functions are `OTC1()` and `OTC2()` that find the OTC corresponding to single-infection and two-infection assays, respectively. The syntax for `OTC1()` is

```
OTC1(algorithm, p = NULL, probabilities = NULL, Se = 0.99, Sp = 0.99,
    group.sz, obj.fn = "ET", weights = NULL, alpha = 2, trace = TRUE,
    print.time = TRUE, ...)
```

The syntax is similar to `opChar1()` as well, so we highlight the main differences only. There is no group membership matrix or row/column size specified because `OTC1()` searches for the OTC. Instead, the `group.sz` argument specifies the initial group sizes to search over. For example, a value of `3:10` searches over initial group sizes of 3 to 10 for hierarchical testing or array testing (row/column size).

The `obj.fn` argument of `OTC1()` indicates which objective function to minimize when searching for the OTC. Earlier in this section, we focused on using the expected number of tests per individual as this objective function `obj.fn = "ET"`. While this is used most often in practice, other objective functions are possible (Hitt et al. 2019). For example, Graff and Roeloffs (1972) proposed to minimize a linear combination of the expected number of tests and the number of misclassified individuals (false positives, false negatives). This objective function is specified using `"GR"`, and the coefficients in this linear combination are given in the `weights` argument.

The `OTC2()` function follows a similar syntax as `OTC1()` with changes like those for `opChar2()` compared to `opChar1()`. We next provide examples using these functions for current group testing applications.

Operating characteristics

We illustrate the use of `opChar1()` for the three-stage hierarchical testing algorithm of Lohse et al. (2020). The group membership matrix for this application is the 3×30 matrix below.

```
> group.member <- matrix(data = c(rep(1, times = 30), rep(1:3, each = 10),
     1:30), nrow = 3, ncol = 30, byrow = TRUE)
> group.member[, 11]

[1]  1  2 11
```

For example, the 11th individual is tested initially in a group of size 30 that includes every specimen (one group overall). In the second stage, this individual is tested in the second sub-group of 10 individuals if its first-stage group is positive. In the third stage, this individual is tested separately if its second-stage group is positive.

We need to specify p or p_i for each individual. In actual practice, these values will be unknown. However, very good point estimates are usually available from past testing results in high volume clinical specimen settings where group testing is used. For the Lohse et al. (2020) application, the observed prevalence of SARS-CoV-2 was 0.0193, so we use this value here for p. By specifying `"D3"` for `algorithm` to represent three-stage hierarchical testing, we invoke `opChar1()` as follows.

```
> library(package = "binGroup2")
> save.Lohse <- opChar1(algorithm = "D3", p = 0.0193, Se = 1, Sp = 1,
     hier.config = group.member, print.time = FALSE)
> summary(save.Lohse)


Algorithm: Non-informative three-stage hierarchical testing

Testing configuration:
Stage 1: 30
Stage 2: 10,10,10

Expected number of tests: 7.64
Expected number of tests per individual: 0.2547

Accuracy for individuals:
     PSe    PSp   PPPV   PNPV Individuals
```

```
1 1.0000 1.0000 1.0000 1.0000         All

Overall accuracy of the algorithm:
     PSe    PSp   PPPV   PNPV
1 1.0000 1.0000 1.0000 1.0000

PSe denotes the pooling sensitivity.
PSp denotes the pooling specificity.
PPPV denotes the pooling positive predictive value.
PNPV denotes the pooling negative predictive value.
```

The sensitivity (`Se`) and the specificity (`Sp`) are set to 1 for each stage because the assay accuracy is not stated in Lohse et al. (2020) or in the product insert of the assay (Diagnostics 2022). While the algorithm's accuracy values produced will not be useful, this approach is often followed in practice to focus on the expected number of tests regardless of false positives/negatives.

The generic `summary()` function uses the corresponding method function for the class to summarize the operating characteristics. For example, the expected number of tests per individual is $E(T)/I = 0.2547$. A more compact summary based on the expected number of tests is available with `ExpTests()`.

```
> ExpTests(save.Lohse)

  ExpTests ExpTestsPerIndividual PercentReductionTests PercentIncreaseTestCap
1   7.6403               0.2547                74.53                292.66
```

Group testing requires $1 - E(T)/I = 75\%$ fewer tests on average than testing each specimen separately. In turn, this leads to a $100\{1/[E(T)/I] - 1\}\% = 293\%$ increase in testing capacity on average when applying the algorithm to a continuous stream of specimens. This type of large increase in testing capacity is why laboratories have implemented group testing during the COVID-19 pandemic.

Our second example focuses on the Aptima Combo 2 Assay and its use by the State Hygienic Laboratory (SHL) at the University of Iowa. The SHL tests thousands of female swab specimens each year in groups of size four using a two-stage hierarchical algorithm. We focus here instead on their male urine specimens because group testing is not currently implemented. The reason is due to concern that their infection prevalence may be too large for group testing to be beneficial. Bilder et al. (2019) provided a Dirichlet distribution with parameter vector $\alpha = (10.99, 0.18, 2.04, 0.31)$ to describe a vector of the joint probabilities of infection for CT and NG that take into account risk factors, such as symptoms and exposure to an infected individual. Using a first-stage group size of 5, we want to calculate how well three-stage hierarchical testing would perform when allowing for differences among probabilities of infection for individuals. To begin, we simulate what one potential set of probabilities of infection could be using the Dirichlet distribution and the `rdirichlet()` function of the **rBeta2009** package (Cheng et al. 2012). These probabilities are ordered by the probability of having at least one infection.

```
> library(package = "rBeta2009")
> set.seed(3789)
> p.unordered <- t(rdirichlet(n = 5, shape = c(10.99, 0.18, 2.04, 0.31)))
> p.ordered <- p.unordered[, order(1 - p.unordered[1, ])]
> round(p.ordered, 4)

       [,1]   [,2]   [,3]   [,4]   [,5]
[1,] 0.9714 0.8841 0.8356 0.8291 0.7009
[2,] 0.0001 0.0006 0.0000 0.0238 0.0000
[3,] 0.0274 0.0879 0.1643 0.1292 0.2991
[4,] 0.0011 0.0274 0.0001 0.0178 0.0000
```

Next, we create the group membership matrix for three-stage informative hierarchical testing using the OTC found in Bilder et al. (2019). Because individuals 4 and 5 are tested separately in the second stage, a `NA` is used for them in the third stage of the group membership matrix.

```
> group.member <- matrix(data = c(rep(1, times = 5), 1, 1, 1, 2, 3, 1, 2,
     3, NA, NA), nrow = 3, ncol = 5, byrow = TRUE)
> group.member

     [,1] [,2] [,3] [,4] [,5]
[1,]    1    1    1    1    1
[2,]    1    1    1    2    3
[3,]    1    2    3   NA   NA
```

Lastly, the `opChar2()` function calculates the operating characteristics for the algorithm. The `Se` and `Sp` matrices provide the sensitivity and specificity values, respectively, for each infection at each stage (ICBSS 2014). Because these accuracies are equal across the stages, the sensitivity (specificity) for CT and NG could be instead included as a single vector of length 2 for the `Se` (`Sp`) argument of `opChar2()`. Alternatively, if accuracies were different across stages, the full matrices provide a general way to include this information.

```
> Se <- matrix(data = rep(c(0.979, 0.985), times = 3), nrow = 2, ncol = 3,
     dimnames = list(Infection = 1:2, Stage = 1:3))
> Sp <- matrix(data = rep(c(0.985, 0.996), times = 3), nrow = 2, ncol = 3,
     dimnames = list(Infection = 1:2, Stage = 1:3))
> save.SHL <- opChar2(algorithm = "ID3", probabilities = p.ordered, Se = Se,
     Sp = Sp, hier.config = group.member, print.time = FALSE)
> summary(save.SHL)


Algorithm: Informative three-stage hierarchical testing

Testing configuration:
Stage 1: 5
Stage 2: 3,1,1

Expected number of tests: 3.59
Expected number of tests per individual: 0.7182

Disease 1 accuracy for individuals:
     PSe    PSp   PPPV   PNPV Individuals
1 0.9770 0.9958 0.2111 1.0000           1
2 0.9778 0.9961 0.8784 0.9994           2
3 0.9784 0.9958 0.0124 1.0000           3
4 0.9729 0.9915 0.8330 0.9988           4
5 0.9787 0.9913 0.0012 1.0000           5

Disease 2 accuracy for individuals:
     PSe    PSp   PPPV   PNPV Individuals
1 0.9585 0.9990 0.9643 0.9988           1
2 0.9633 0.9992 0.9940 0.9952           2
3 0.9575 0.9994 0.9969 0.9917           3
4 0.9725 0.9979 0.9879 0.9953           4
5 0.9714 0.9984 0.9961 0.9879           5

Overall accuracy of the algorithm:
     PSe    PSp   PPPV   PNPV
1 0.9749 0.9941 0.7039 0.9996
2 0.9669 0.9988 0.9931 0.9941

PSe denotes the pooling sensitivity.
PSp denotes the pooling specificity.
PPPV denotes the pooling positive predictive value.
PNPV denotes the pooling negative predictive value.
```

The expected number of tests per individual is 0.7182. Because this value is less than 1, group testing would be better on average than testing each individual separately. The PSe and PSp values in the output are the pooling sensitivity and specificity, respectively. Additionally, values are given for the pooling positive predictive value (PPPV) and the pooling negative predictive value (PNPV) (see Altman and Bland 1994 and Hitt et al. 2019).

Of course, not every set of 5 individuals has these 5 joint probabilities of infection. When this group testing algorithm was applied to the data from which the Dirichlet distribution was estimated, Bilder et al. (2019) showed that the number of tests decreased by approximately 26% on average.

Optimal testing configuration

Returning to the three-stage hierarchical algorithm of Lohse et al. (2020), suppose again that $p = 0.0193$. The OTC is found using the following code.

```
> OTC.Lohse <- OTC1(algorithm = "D3", p = 0.0193, Se = 1, Sp = 1,
     group.sz = 3:20, obj.fn = "ET", print.time = FALSE)

Initial Group Size = 3
Initial Group Size = 4
Initial Group Size = 5
Initial Group Size = 6
Initial Group Size = 7
Initial Group Size = 8
Initial Group Size = 9
Initial Group Size = 10
Initial Group Size = 11
Initial Group Size = 12
Initial Group Size = 13
Initial Group Size = 14
Initial Group Size = 15
Initial Group Size = 16
Initial Group Size = 17
Initial Group Size = 18
Initial Group Size = 19
Initial Group Size = 20

> summary(OTC.Lohse)


Algorithm: Non-informative three-stage hierarchical testing

Optimal testing configuration:
   Stage 1 Stage 2
ET      16 4,4,4,4

Expected number of tests:
   E(T)  Value
ET 3.27 0.2045

E(T) denotes the expected number of tests.
Value denotes the objective function value per individual.

Overall accuracy of the algorithm:
      PSe    PSp   PPPV   PNPV
ET 1.0000 1.0000 1.0000 1.0000

PSe denotes the pooling sensitivity.
PSp denotes the pooling specificity.
PPPV denotes the pooling positive predictive value.
PNPV denotes the pooling negative predictive value.
```

The `summary()` function summarizes the OTC. The first-stage group size is 16, the second stage has 4 groups of size 4, and the third stage uses individual testing. The expected number of tests per individual is 0.2045. Therefore, the OTC decreases the expected number of tests per individual further by almost 20% $((0.2547 - 0.2045)/0.2547)$ over the testing configuration used by Lohse et al. (2020). In terms of expected test capacity, one can show with `ExpTests(OTC.Lohse)` that the OTC leads to a 389% increase in testing capacity when compared to testing each specimen separately. Again, this is significantly better than the testing configuration chosen by Lohse et al. (2020) and helps to show the importance of choosing a testing configuration.

The `OTC1()` function searches for the optimal set of group sizes at *each* first-stage group size specified in the `group.sz` argument. By default, the function's progress is printed during its running (`trace = TRUE`). We take this approach to finding the OTC, rather than optimizing over all group sizes initially, because a laboratory may prefer a sub-optimal testing configuration due to their work environment. The `Config()` function provides information about these sub-optimal testing configurations. This function extracts each of the best testing configurations by initial group size and returns them as a data frame sorted by the value of the objective function ($E(T)/I$ because the default was used in `OTC1()`).

```
> Config(OTC.Lohse)
```

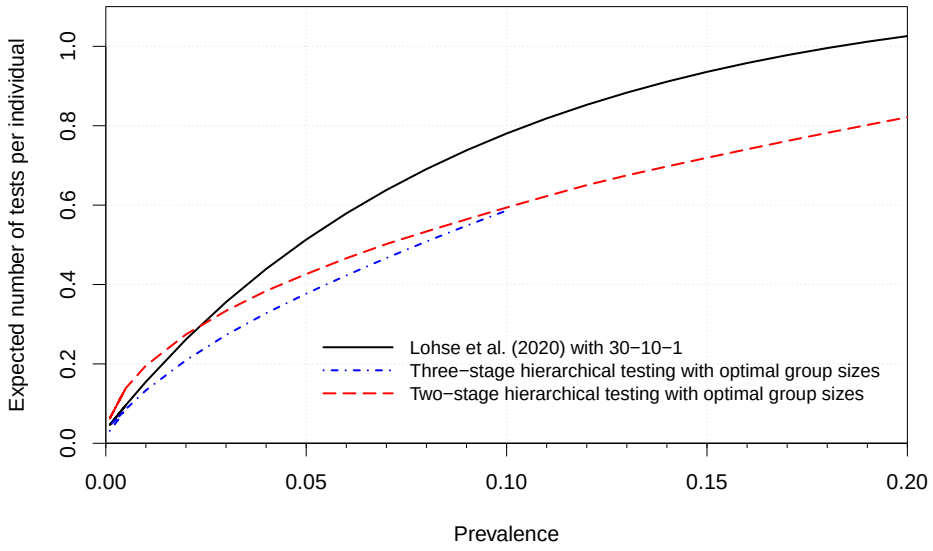

Figure 1: Expected number of tests per individual as a function of infection prevalence. Three-stage hierarchical testing using the OTC is not plotted for prevalences greater than 0.10 because two stages results in a lower expected number of tests.

```
   I    config     ET  value PSe PSp PPPV PNPV
1 16   4,4,4,4 3.2714 0.2045   1   1    1    1
2 17   5,4,4,4 3.4922 0.2054   1   1    1    1
3 15   4,4,4,3 3.0842 0.2056   1   1    1    1
4 20 4,4,4,4,4 4.1138 0.2057   1   1    1    1
5 19 4,4,4,4,3 3.9176 0.2062   1   1    1    1
```

Adding `top.overall = TRUE` to `Config()` provides the same information but with the top testing configurations overall rather than for each initial group size.

The most efficient outcome from the OTC depends on the chosen value of p. As mentioned earlier in this section, very good point estimates for p are often available. Still, it would be of interest to determine what would occur for other choices of p. The **binGroup2** package provides the tools needed by experienced R users for this more in depth examination. Our Appendix A provides additional code using `OTC1()` to produce Figure 1. The testing configuration chosen by Lohse et al. (2020) is never optimal. In fact, two-stage hierarchical group testing is more efficient than Lohse et al. (2020) for $p > 0.03$.

Our second example focuses on SARS-CoV-2 detection again, but now in the context of LabCorp's array testing. LabCorp (2021) examined the potential performance of their approach using possible prevalences between 0.001 and 0.15, so we begin with using a prevalence of 0.05. Also, LabCorp (2021) estimates that approximately 0.023 of all positives would be missed using array testing rather than individual testing. This is not quite the sensitivity of the assay, but we will use 0.977 and 1 as the sensitivity for the first and second stages, respectively, for illustration purposes. We will also use 1 as the specificity for illustration purposes. The expected number of tests per individual for the algorithm is found first using `opChar1()` with `algorithm = "A2"`.

```
> save.LabCorp <- opChar1(algorithm = "A2", p = 0.05, rowcol.sz = 5,
     Se = c(1 - 0.023, 1), Sp = 1, print.time = FALSE)
> ExpTests(save.LabCorp)

  ExpTests ExpTestsPerIndividual PercentReductionTests PercentIncreaseTestCap
1  12.0763               0.4831                51.69                107.02
```

The expected number of tests per individual is 0.48, resulting in an expected 107% increase in testing capacity.

Could LabCorp do better? Below is the code used to find the OTC for a range of group sizes from 3 to 10 with array testing.

```
> OTC.LabCorp.Array <- OTC1(algorithm = "A2", p = 0.05, Se = c(0.977,
     1), Sp = 1, group.sz = 3:10, obj.fn = "ET", trace = FALSE,
     print.time = FALSE)
> summary(OTC.LabCorp.Array)


Algorithm: Non-informative array testing without master pooling

Optimal testing configuration:
   Row/column size Array size
ET              10        100

Expected number of tests:
    E(T)  Value
ET 37.20 0.3720

E(T) denotes the expected number of tests.
Value denotes the objective function value per individual.

Overall accuracy of the algorithm:
      PSe    PSp   PPPV   PNPV
ET 0.9550 1.0000 1.0000 0.9976

PSe denotes the pooling sensitivity.
PSp denotes the pooling specificity.
PPPV denotes the pooling positive predictive value.
PNPV denotes the pooling negative predictive value.

> ExpTests(OTC.LabCorp.Array)

       ExpTests ExpTestsPerInd PercentReductionTests PercentIncreaseTestCap
opt.ET  37.1953         0.3720                 62.80                 168.85
```

The OTC for array testing is a 10×10 array that has an expected number of tests per individual of 0.37 and an increase in testing capacity of 169%. This is a significant improvement over the 5×5 array. One could perform similar calculations using two-stage hierarchical testing with `algorithm = "D2"` in `OTC1`. This results in an OTC that uses an initial group size of 5 and leads to an expected number of tests per individual of 0.42.

Because these calculations rely on $p = 0.05$, we include results and the corresponding more advanced code to investigate additional values of p in Appendix B. In most cases, the 5×5 array is not the OTC.

Additional functions

Not all group testing algorithms fit well within a general computing framework. For this reason, we include a few additional functions outside of those discussed previously. In particular, the `halving()` and `Sterrett()` functions provide alternative ways to use hierarchical testing algorithms. The former function calculates operating characteristics when positive testing groups are split only in half (Litvak et al. 1994; Black et al. 2012). The latter function calculates operating characteristics for algorithms that retest only one specimen at a time from a positive group (Sterrett 1957; Bilder et al. 2010a). Once a positive is found, the remaining specimens are retested again in a group. The idea behind this strategy is there will likely be only one positive (or few positives) in the original group.

Conclusion

The **binGroup2** package provides researchers and laboratories with the statistical tools needed to implement group testing most effectively. An earlier version of this package was used as well by *The New York Times* to help readers understand the benefits from group testing during the COVID-19 pandemic (Bui et al. 2020). Similar to Bilder et al. (2010b), we encourage researchers to submit their own functions to us to be included within the package. This will allow users to have one overall package rather than many packages that may duplicate the work of others.

The identification functions are for one and two-infection assays. While there are a few three and four-infection assays for infectious disease detection that have been recently released (e.g., the BD

Max CT/GC/TV assay (BD 2022) that tests for pathogens which lead to chlamydia, gonorrhea, and trichomoniasis), these assays are used currently much less than single and two-infection assays. Also, derivations for operating characteristics become much more complex for more than two infections, so there are no closed-form expressions available for these types of assays. For example, Hou et al. (2020) needed Monte Carlo simulation for a three-infection assay. We anticipate that it will be useful to include Monte Carlo simulation-based estimates of operating characteristics in future versions of **binGroup2** as these types of multiplex assays become more widely used. New research is needed though to find efficient computational approaches when searching for the OTC due to these simulation aspects.

While not the focus of this paper, the estimation functions from **binGroup** have been simplified and expanded upon using a coherent style. The new `propCI()` function combines three previous functions into one to calculate point estimates and confidence intervals for an infection prevalence. The new `propDiffCI()` provides similar functionality for the difference of two infection prevalences. Both functions incorporate new research since Bilder et al. (2010b). In particular, the bias correction methods of Hepworth and Biggerstaff (2017) are included to account for the long-standing problem of bias for point estimates in group testing (Swallow 1985). The `gtReg()` function combines three previous **binGroup** functions used to estimate regression models with group testing data that arise from different testing algorithms. The `gtSim()` function also combines three previous **binGroup** functions to simulate group testing data with covariates that can arise from different testing algorithms.

Acknowledgments

The authors thank Jeffrey Benfer and Kristopher Eveland at the SHL and Peter Iwen and Baha Abdalhamid at the Nebraska Public Health Laboratory for their consultation on CT/NG and SARS-CoV-2 testing. This research was supported by Grant R01 AI121351 from the National Institutes of Health. The views expressed in this article are those of the authors and do not necessarily reflect the official policy or position of the United States Air Force Academy, the Air Force, the Department of Defense, or the U.S. Government.

Appendix A: Details for Figure 1

The code in this appendix creates Figure 1. This figure presents a comparison of the three-stage hierarchical testing algorithm of Lohse et al. (2020) to using the OTC for two- and three-stage hierarchical testing. We use the `OTC1()` function to find the OTCs for the prevalences of 0.001, 0.005, and 0.01 to 0.20 by 0.01 with a maximum group size of 64. The large number of prevalences and large group sizes will result in a significant amount of computational time. For readers interested in testing the code, we recommend using a few prevalence values and a maximum group size of 30 instead as given in the code here.

```
> # Use these prevalences when testing the code
> p.seq <- seq(from = 0.05, to = 0.06, by = 0.01)
> # Prevalences used in the paper: c(0.001, 0.005, seq(from =
> # 0.01, to = 0.20, by = 0.01))
>
> # Group membership matrix for Lohse et al. (2020)
> group.member <- matrix(data = c(rep(1, times = 30), rep(1:3,
    each = 10), 1:30), nrow = 3, ncol = 30, byrow = TRUE)
>
> # Save results from for loop here
> save.res <- matrix(data = NA, nrow = length(p.seq), ncol = 7)
> save.res.SecondOf3Stages <- matrix(data = NA, nrow = length(p.seq),
    ncol = 2)
>
> counter <- 1
>
> # Use this group size when testing the code
> max.gp.size <- 30
> # Maximum group size used in the paper: 64
>
> for (p in p.seq) {
    # Lohse et al. (2020) 30-10-1 testing configuration
    res.Lohse <- opChar1(algorithm = "D3", p = p, Se = 1, Sp = 1,
```

```
        hier.config = group.member, print.time = FALSE)
    # Find OTCs for two- and three-stage hierarchical testing
    res.D2 <- OTC1(algorithm = "D2", p = p, Se = 1, Sp = 1, group.sz = 3:max.gp.size,
        obj.fn = "ET", trace = FALSE, print.time = FALSE)
    res.D3 <- OTC1(algorithm = "D3", p = p, Se = 1, Sp = 1, group.sz = 3:max.gp.size,
        obj.fn = "ET", trace = FALSE, print.time = FALSE)
    # Save results
    save.res[counter, ] <- c(p, res.Lohse$Config$Stage1, res.Lohse$value,
        res.D2$opt.ET$OTC$Stage1, res.D2$opt.ET$value, res.D3$opt.ET$OTC$Stage1,
        res.D3$opt.ET$value)
    save.res.SecondOf3Stages[counter, ] <- c(p, paste(res.D3$opt.ET$OTC$Stage2,
        collapse = " "))
    counter <- counter + 1
 }
>
> # Expected number of tests per individual and corresponding
> # group sizes
> colnames(save.res) <- c("p", "LohseSize", "LohseEff", "TwoStage.Stage1",
    "TwoStageEff", "ThreeStage.Stage1", "ThreeStageEff")
> save.res2 <- as.data.frame(save.res)
> save.res2$ThreeStage.Stage2 <- save.res.SecondOf3Stages[, 2]
>
> # Compare the expected number of tests per individual
> plot(x = save.res2$p, y = save.res2$LohseEff, type = "l", xlab = "Prevalence",
    ylab = "Expected number of tests per individual", col = "black",
    panel.first = grid(), ylim = c(0, 1.1), lwd = 2, yaxs = "i",
    xaxs = "i", xlim = c(0, 0.2))
> lines(x = save.res2$p, y = save.res2$TwoStageEff, lwd = 2, lty = "longdash",
    col = "red")
>
> # Check if two-stage is more efficient than three-stage. If
> # so, don't plot because a laboratory would not want to
> # implement three stages if this occurred.
> check <- save.res2$TwoStageEff > save.res2$ThreeStageEf
> lines(x = save.res2$p[check], y = save.res2$ThreeStageEff[check],
    lwd = 2, lty = "dotdash", col = "blue")
>
> axis(side = 1, at = seq(from = 0, to = 0.2, by = 0.01), tck = -0.01,
    labels = FALSE)
> legend(x = 0.05, y = 0.3, legend = c("Lohse et al. (2020) with 30-10-1",
    "Three-stage hierarchical testing with optimal group sizes",
    "Two-stage hierarchical testing with optimal group sizes"),
    lty = c("solid", "dotdash", "longdash"), bty = "n", bg = "white",
    col = c("black", "blue", "red"), lwd = 2, seg.len = 4, cex = 0.9)
```

Appendix B: LabCorp's array testing

LabCorp (2021) described the use of arrays for up to size 5×5. We present code in this appendix to create Figure 2 that provides evidence that a 5×5 array size is not an optimal choice. For this plot, we calculated the OTC for array testing by minimizing the expected number of tests per individual for the prevalences of 0.001, 0.005, and 0.01 to 0.20 by 0.01. Because two-stage hierarchical testing is also frequently used for SARS-CoV-2 detection, we also found the OTC for this algorithm. Throughout these calculations, we conservatively limit the maximum group size to 10.

```
> # Prevalences used in the paper
> p.seq <- c(0.001, 0.005, seq(from = 0.01, to = 0.2, by = 0.01))
>
> # Save results from for loop here
> save.res <- matrix(data = NA, nrow = length(p.seq), ncol = 7)
>
> counter <- 1
>
> for (p in p.seq) {
```

```
    # LabCorp (2021) 5x5 array
    res.LabCorp <- opChar1(algorithm = "A2", p = p, rowcol.sz = 5,
        Se = c(0.977, 1), Sp = 1, print.time = FALSE)
    # Find OTCs for array testing and two-stage hierarchical
    # testing
    res.A2 <- OTC1(algorithm = "A2", p = p, Se = c(0.977, 1),
        Sp = 1, group.sz = 3:10, obj.fn = "ET", trace = FALSE,
        print.time = FALSE)
    res.D2 <- OTC1(algorithm = "D2", p = p, Se = c(0.977, 1),
        Sp = 1, group.sz = 3:10, obj.fn = "ET", trace = FALSE,
        print.time = FALSE)
    # Save results
    save.res[counter, ] <- c(p, res.LabCorp$Config$Array.dim,
        res.LabCorp$value, res.A2$opt.ET$OTC$Array.dim, res.A2$opt.ET$value,
        res.D2$opt.ET$OTC$Stage1, res.D2$opt.ET$value)
    counter <- counter + 1
 }
>
> # Expected number of tests per individual and the
> # corresponding group sizes
> colnames(save.res) <- c("p", "LabCorpSize", "LabCorpEff", "ArrayOTC",
    "ArrayEff", "DorfOTC", "DorfEff")
> save.res2 <- as.data.frame(save.res)
>
> # Compare the expected number of tests per individual
> plot(x = save.res2$p, y = save.res2$LabCorpEff, type = "l", xlab = "Prevalence",
    ylab = "Expected number of tests per individual", col = "black",
    panel.first = grid(), ylim = c(0, 1.1), lwd = 2, yaxs = "i",
    xaxs = "i", xlim = c(0, 0.2))
> lines(x = save.res2$p, y = save.res2$DorfEff, lwd = 2, lty = "longdash",
    col = "red")
> lines(x = save.res2$p, y = save.res2$ArrayEff, lwd = 2, lty = "121252",
    col = "darkgreen")
> axis(side = 1, at = seq(from = 0, to = 0.2, by = 0.01), tck = -0.01,
    labels = FALSE)
> legend(x = 0.07, y = 0.3, lty = c("solid", "longdash", "121252"),
    legend = c("LabCorp with 5x5 array", "Array testing with optimal group sizes",
        "Two-stage hierarchical testing with optimal group sizes"),
    bty = "n", bg = "white", col = c("black", "red", "darkgreen"),
    lwd = 2, seg.len = 4, cex = 0.9)
```

Bibliography

B. Abdalhamid, C. Bilder, E. McCutchen, S. Hinrichs, S. Koepsell, and P. Iwen. Assessment of specimen pooling to conserve SARS CoV-2 testing resources. *American Journal of Clinical Pathology*, 153:715–718, 2020. URL https://doi.org/10.1093/ajcp/aqaa064. [p1, 2]

M. Abdelmalek. With all eyes on coronavirus testing, some researchers say 'group testing' could make up the shortage. *ABC News*, 2020. URL https://abcnews.go.com/Health/eyes-coronavirus-testing-researchers-group-testing-make-shortage/story?id=70658896. May 13; Retrieved April 27, 2022. [p1]

D. Altman and J. Bland. Diagnostic tests 2: Predictive values. *BMJ*, 309:102, 1994. URL https://doi.org/10.1136/bmj.309.6947.102. [p7]

American Red Cross. Infectious disease testing. 2022. URL https://www.redcrossblood.org/biomedical-services/blood-diagnostic-testing/blood-testing.html. Retrieved April 27, 2022. [p2]

N. Ando, D. Mizushima, K. Watanabe, M. Takano, D. Shiojiri, H. Uemura, A. Takahiro, Y. Yanagawa, Y. Kikuchi, S. Oka, and H. Gatanaga. Modified self-obtained pooled sampling to screen for *Chlamydia trachomatis* and *Neisseria gonorrhoeae* infections in men who have sex with men. *Sexually Transmitted Infections*, 97:324–328, 2021. URL http://dx.doi.org/10.1136/sextrans-2020-054666. [p2]

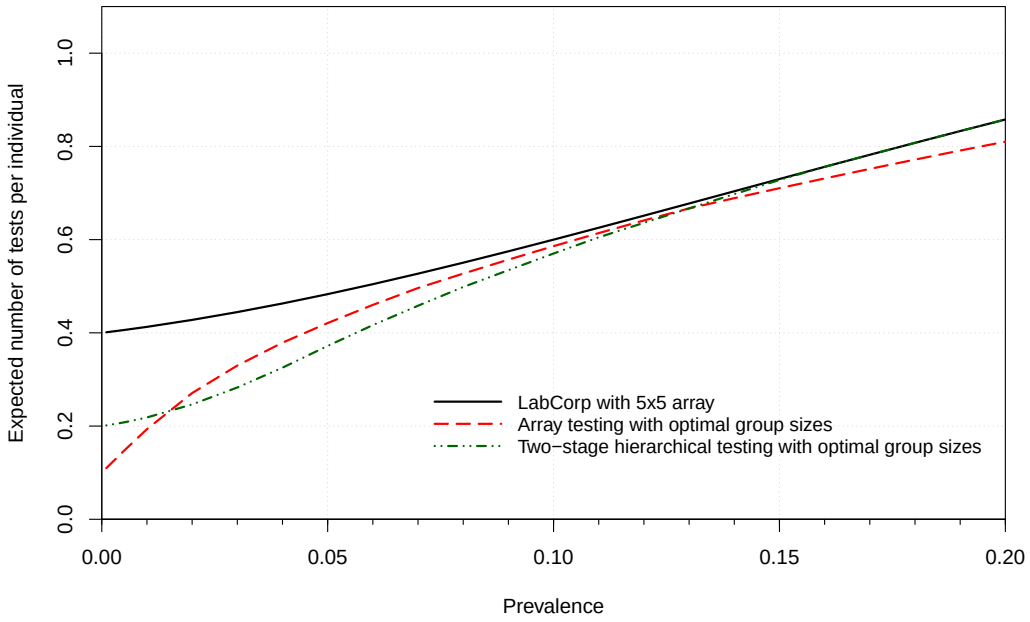

Figure 2: Expected number of tests per individual as a function of infection prevalence for LabCorp's setting.

E. Anthes. A CDC airport surveillance program found the earliest known US cases of Omicron subvariants. *The New York Times*, 2022. URL https://www.nytimes.com/2022/03/24/health/cdc-us-ba2.html. March 24; Retrieved April 27, 2022. [p1]

R. Barathidasan, F. Sharmila, R. Raj, G. Dhanalakshmi, G. Anitha, and R. Dhodapkar. Pooled sample testing for COVID-19 diagnosis: Evaluation of bi-directional matrix pooling strategies. *Journal of Virological Methods*, page 114524, 2022. URL https://doi.org/10.1016/j.jviromet.2022.114524. [p1]

BD. BD Max CT/NG/TV. 2022. URL https://moleculardiagnostics.bd.com/syndromic-solutions/womens-health-stis/CT-GC-TV. Retrieved April 27, 2022. [p11]

C. Bilder, J. Tebbs, and P. Chen. Informative retesting. *Journal of the American Statistical Association*, 105: 942–955, 2010a. URL https://doi.org/10.1198/jasa.2010.ap09231. [p10]

C. Bilder, B. Zhang, F. Schaarschmidt, and J. Tebbs. binGroup: A package for group testing. *The R Journal*, 2:56–60, 2010b. URL https://journal.r-project.org/archive/2010-2. [p1, 3, 10, 11]

C. Bilder, J. Tebbs, and C. McMahan. Informative group testing for multiplex assays. *Biometrics*, 75: 278–288, 2019. URL https://doi.org/10.1111/biom.12988. [p4, 6, 7]

M. Black, C. Bilder, and J. Tebbs. Group testing in heterogeneous populations by using halving algorithms. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 61:277–290, 2012. URL https://doi.org/10.1111/j.1467-9876.2011.01008.x. [p10]

M. Black, C. Bilder, and J. Tebbs. Optimal retesting configurations for hierarchical group testing. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 64:693–710, 2015. URL https://doi.org/10.1111/rssc.12097. [p4]

Q. Bui, S. Kliff, and M. Sanger-Katz. How to test more people for coronavirus without actually needing more tests. *The New York Times*, 2020. URL https://www.nytimes.com/interactive/2020/07/27/upshot/coronavirus-pooled-testing.html. July 27; Retrieved April 27, 2022. [p10]

C. Cheng, Y. Hung, and N. Balakrishnan. *rBeta2009: The Beta Random Number and Dirichlet Random Vector Generating Functions*, 2012. URL https://CRAN.R-project.org/package=rBeta2009. [p6]

A. Diagnostics. Realstar. 2022. URL https://www.altona-diagnostics.com/en/products/reagents-140/reagents/realstar-real-time-pcr-reagents/realstar-sars-cov-2-rt-pcr-kit-ruo.html. Retrieved April 27, 2022. [p6]

R. Dorfman. The detection of defective members of large populations. *Annals of Mathematical Statistics*, 14:436–440, 1943. [p1]

L. Graff and R. Roeloffs. Group testing in the presence of test error; an extension of the Dorfman procedure. *Technometrics*, 14:113–122, 1972. [p5]

G. Hepworth and B. Biggerstaff. Bias correction in estimating proportions by pooled testing. *Journal of Agricultural, Biological and Environmental Statistics*, 22:602–614, 2017. [p11]

B. Hitt. *Group Testing Identification: Objective Functions, Implementation, and Multiplex Assays*. PhD thesis, University of Nebraska-Lincoln, 2020. [p4]

B. Hitt, C. Bilder, J. Tebbs, and C. McMahan. The objective function controversy for group testing: Much ado about nothing? *Statistics in Medicine*, 38:4912–4923, 2019. URL https://doi.org/10.1002/sim.8341. [p1, 5, 7]

C. Hogan, M. Sahoo, and B. Pinsky. Sample pooling as a strategy to detect community transmission of SARS-CoV-2. *Journal of the American Medical Association*, 323:1967–1969, 05 2020. URL https://doi.org/10.1001/jama.2020.5445. [p1]

Hologic. Aptima STIs. 2022. URL https://www.hologic.com/hologic-products/diagnostic-solutions/aptima-stis. Retrieved April 27, 2022. [p3]

P. Hou, J. Tebbs, D. Wang, C. McMahan, and C. Bilder. Array testing for multiplex assays. *Biostatistics*, 21:417–431, 2020. URL https://doi.org/10.1093/biostatistics/kxy058. [p2, 4, 11]

ICBSS. *Iowa Community-Based Screening Services Procedures Manual*, 2014. URL http://www.shl.uiowa.edu/dcd/iippmanual.pdf. Retrieved April 27, 2022. [p7]

N. Johnson, S. Kotz, and X. Wu. *Inspection Errors for Attributes in Quality Control*. CRC Press, 1991. [p4]

H. Kim, M. Hudgens, J. Dreyfuss, D. Westreich, and C. Pilcher. Comparison of group testing algorithms for case identification in the presence of test error. *Biometrics*, 63:1152–1163, 2007. URL https://doi.org/10.1111/j.1541-0420.2007.00817.x. [p2, 3, 4]

S. Kim, H. Kim, H. Kim, H. Ann, J. Kim, H. Choi, M. Kim, J. Song, J. Ahn, N. Ku, D. Oh, Y. Kim, S. Jeong, S. Han, J. Kim, D. Smith, and J. Choi. Pooled nucleic acid testing to identify antiretroviral treatment failure during HIV infection in Seoul, South Korea. *Scandinavian Journal of Infectious Diseases*, 46:136–140, 2014. URL https://doi.org/10.3109/00365548.2013.851415. [p2]

LabCorp. Emergency Use Authorization summary for COVID-19 RT-PCR test. 2021. URL https://www.fda.gov/media/136151/download. Retrieved April 27, 2022. [p1, 3, 9, 12]

S. Lendle. *gtcorr: Calculate Efficiencies of Group Testing Algorithms with Correlated Responses*, 2011. URL https://CRAN.R-project.org/package=gtcorr. Retrieved April 27, 2022. [p1]

S. Lendle, M. Hudgens, and B. Qaqish. Group testing for case identification with correlated responses. *Biometrics*, 68:532–540, 2012. URL https://doi.org/10.1111/j.1541-0420.2011.01674.x. [p1]

J. Lewis, V. Lockary, and S. Kobic. Cost savings and increased efficiency using a stratified specimen pooling strategy for *Chlamydia trachomatis* and *Neisseria gonorrhoeae*. *Sexually Transmitted Diseases*, 39:46–48, 2012. URL https://doi.org/10.1097/OLQ.0b013e318231cd4a. [p3]

E. Litvak, X. Tu, and M. Pagano. Screening for the presence of a disease by pooling sera samples. *Journal of the American Statistical Association*, 89:424–434, 1994. URL https://doi.org/10.1080/01621459.1994.10476764. [p10]

T. Liu and Y. Xu. *mMPA: Implementation of Marker-Assisted Mini-Pooling with Algorithm*, 2018. URL https://CRAN.R-project.org/package=mMPA. Retrieved April 27, 2022. [p1]

T. Liu, J. Hogan, M. Daniels, M. Coetzer, X. Yizhen, B. Gerald, A. DeLong, L. Ledingham, M. Orido, L. Diero, and R. Kantor. Improved HIV-1 viral load monitoring capacity using pooled testing with marker-assisted deconvolution. *Journal of Acquired Immune Deficiency Syndromes*, 75:580, 2017. URL https://doi.org/10.1097/QAI.0000000000001424. [p1]

S. Lohse, T. Pfuhl, B. Berkó-Göttel, J. Rissland, T. Geißler, B. Gärtner, S. Becker, S. Schneitler, and S. Smola. Pooling of samples for testing for SARS-CoV-2 in asymptomatic people. *The Lancet Infectious Diseases*, 20:1231–1232, 2020. URL https://doi.org/10.1016/S1473-3099(20)30362-5. [p2, 5, 6, 7, 8, 9, 11]

A. Mandavilli. Federal officials turn to a new testing strategy as infections surge. *The New York Times*, 2020. URL https://www.nytimes.com/2020/07/01/health/coronavirus-pooled-testing.html. July 1; Retrieved April 27, 2022. [p1]

C. McMahan, J. Tebbs, and C. Bilder. Informative Dorfman screening. *Biometrics*, 68:287–296, 2012a. URL https://doi.org/10.1111/j.1541-0420.2011.01644.x. [p3, 4]

C. McMahan, J. Tebbs, and C. Bilder. Two-dimensional informative array testing. *Biometrics*, 68: 793–804, 2012b. URL https://doi.org/10.1111/j.1541-0420.2011.01726.x. [p2, 3]

Nebraska Veterinary Diagnostic Center. Diagnostic tests & fees, 2022. URL https://vbms.unl.edu/nvdc-tests-fees. Retrieved April 27, 2022. [p2]

Roche. Roche receives FDA Emergency Use Authorization for the cobas SARS-CoV-2 & Influenza A/B Test for use on the cobas 6800/8800 Systems. 2020. URL https://www.roche.com/media/releases/med-cor-2020-09-04.htm. Retrieved April 27, 2022. [p3]

E. Salzer, E. Nixon, G. Drewes, F. Reinhard, G. Bergamini, and C. Rau. Screening pools of compounds against multiple endogenously expressed targets in a chemoproteomics binding assay. *Journal of Laboratory Automation*, 21:133–142, 2016. URL https://journals.sagepub.com/doi/full/10.1177/2211068215595355. [p2]

A. Sterrett. On the detection of defective members of large populations. *The Annals of Mathematical Statistics*, 28:1033–1036, 1957. [p10]

W. Swallow. Group testing for estimating infection rates and probabilities of disease transmission. *Phytopathology*, 75:882–889, 1985. [p11]

M. Thai. *Group Testing Theory in Network Security: An Advanced Solution*. Springer, 2011. [p2]

P. Thai, A. Banks, L. Toms, P. Choi, X. Wang, P. Hobson, and J. Mueller. Analysis of urinary metabolites of polycyclic aromatic hydrocarbons and cotinine in pooled urine samples to determine the exposure to PAHs in an Australian population. *Environmental Research*, 182:109048, 2020. URL https://doi.org/10.1016/j.envres.2019.109048. [p2]

T. Van, J. Miller, D. Warshauer, E. Reisdorf, D. Jernigan, R. Humes, and P. Shult. Pooling nasopharyngeal/throat swab specimens to increase testing capacity for influenza viruses by PCR. *Journal of Clinical Microbiology*, 50:891–896, 2012. URL https://doi.org/10.1128/JCM.05631-11. [p2]

Verily Life Sciences. Emergency Use Authorization summary: Verily COVID-19 RT-PCR test for use with the Verily COVID-19 Nasal Swab Kit. 2021. URL https://www.fda.gov/media/141951/download. Retrieved April 27, 2022. [p1]

Yale University. Yale School of Public Health, Department of Epidemiology of Microbial Diseases SalivaDirect for use with DTC Kits assay EUA summary. 2022. URL https://www.fda.gov/media/151841/download. Retrieved April 27, 2022. [p1]

K. Zhao and C. Rosa. Thrips as the transmission bottleneck for mixed infection of two orthotospoviruses. *Plants*, 9:509, 2020. URL https://doi.org/10.3390/plants9040509. [p2]

Christopher R. Bilder
Department of Statistics
University of Nebraska-Lincoln
Lincoln, NE 68583, USA
chris@chrisbilder.com
www.chrisbilder.com

Brianna D. Hitt
Department of Mathematical Sciences
United States Air Force Academy
Colorado Springs, CO 80840, USA
brianna.hitt@afacademy.af.edu

Brad J. Biggerstaff
Division of Vector-Borne Diseases
Centers for Disease Control and Prevention
Fort Collins, CO 80521, USA
bkb5@cdc.gov

Joshua M. Tebbs
Department of Statistics
University of South Carolina
Columbia, SC 29208, USA
tebbs@stat.sc.edu

Christopher S. McMahan
School of Mathematical and Statistical Sciences
Clemson University
Clemson, SC 29634, USA
mcmaha2@clemson.edu