A SEMI-SUPERVISED BAYESIAN MIXTURE MODELLING APPROACH FOR JOINT BATCH CORRECTION AND CLASSIFICATION

STEPHEN COLEMAN[*,1], **XAQUIN CASTRO DOPICO**[2], **GUNILLA B. KARLSSON HEDESTAM**[2], **PAUL D.W. KIRK**[†,1,3], **CHRIS WALLACE**[†,1,3]

[1] MRC Biostatistics Unit, University of Cambridge, U.K.

[2] Department of Microbiology, Tumor and Cell Biology, Karolinska Institutet, Sweden.

[3] Cambridge Institute of Therapeutic Immunology & Infectious Disease, University of Cambridge, U.K.

ABSTRACT

Systematic differences between batches of samples present significant challenges when analysing biological data. Such *batch effects* are well-studied and are liable to occur in any setting where multiple batches are assayed. Many existing methods for accounting for these have focused on high-dimensional data such as RNA-seq and have assumptions that reflect this. Here we focus on batch-correction in low-dimensional classification problems. We propose a semi-supervised Bayesian generative classifier based on mixture models that jointly predicts class labels and models batch effects. Our model allows observations to be probabilistically assigned to classes in a way that incorporates uncertainty arising from batch effects. We explore two choices for the within-class densities: the multivariate normal and the multivariate t. A simulation study demonstrates that our method performs well compared to popular off-the-shelf machine learning methods and is also quick; performing 15,000 iterations on a dataset of 500 samples with 2 measurements each in 7.3 seconds for the MVN mixture model and 11.9 seconds for the MVT mixture model. We

∗ Corresponding author: stephen.coleman@mrc-bsu.cam.ac.uk

† These authors provided an equal contribution.

apply our model to two datasets generated using the enzyme-linked immunosorbent assay (ELISA), a spectrophotometric assay often used to screen for antibodies. The examples we consider were collected in 2020 and measure seropositivity for SARS-CoV-2. We use our model to estimate seroprevalence in the populations studied. We implement the models in C++ using a Metropolis-within-Gibbs algorithm; this is available in the R package at `https://github.com/stcolema/BatchMixtureModel`. Scripts to recreate our analysis are at `https://github.com/stcolema/BatchClassifierPaper`.

Keywords SARS-CoV-2 · ELISA · Mixture model · Batch correction · Bayes · Assay data · Classification.

1 Background

Many biological assays are performed across sets of samples or *batches*. When the number of samples exceeds the batch size, then it is common to notice *batch effects*, systematic differences between assay readouts from different batches which may affect both their mean and scale. This is a prevalent problem, that may be addressed in a variety of ways depending on the planned downstream analysis. In discussing available options for batch correction, we will use the term "batch effect" to mean differences between samples arising from between-batch technical factors in the experiment, and the term "class effect" to refer to biological differences arising due to samples coming from distinct biological classes. We consider settings in which the objective is to classify unlabelled samples into predefined classes.

To analyse class effects we should also account for the batch effects. One common approach is to first correct for batch effects as part of a pre-processing or data cleaning step (which might be as simple as zero-centring the data; i.e., transforming each batch to have a common mean), and then to apply standard classification models to the resulting "cleaned" data (e.g., 2, 25, 32). However, such two-step approaches have been found to increase false positive rates because they may induce correlation between the cleaned observations which is typically not accounted for in downstream analysis (23). Further, when batch is confounded with class effects (due to unbalanced representation of classes across batches) then naive adjustment which ignores known biological classes in the data can lead to incorrect conclusions (22), and methods for adjustment which preserve differences attributable to known classes can lead to false positive results (29). An alternative approach is to incorporate batch information directly into downstream analyses, for example as a covariate in regression-based approaches. It has been shown that mixed effects models which share information between batches produce better calibrated quantitative data than independent analyses of each batch (35). However, only a subset of analytical approaches have been adapted to accommodate batch effects (e.g., 28, 30, 21),

and there has been a strong focus on high-dimensional settings (e.g., 17, 6, 1). Thus a need exists for a wider range of methods that can account for batch effects directly in low-dimensional data analysis.

Here we focus on the problem of assigning class labels using low-dimensional assay data generated across several batches. This is a common design in many assays that measure a small number of specific biomarkers such as enzyme-linked immunosorbent assay (ELISA) and flow cytometry data. If there are known classes in the population, then class-specific controls can be included in the assay, resulting in training examples for which the class labels are known. We are motivated in part by the specific problem of estimating seroprevalance of SARS-CoV-2 by classifying individuals into seropositive and seronegative classes at different points in time during the pandemic. Since batches tend to comprise samples collected at the same time point, and since seroprevalence is expected to vary through the course of the pandemic, we expect class membership to be imbalanced across batches – motivating the development of a joint classification and batch-correction model, rather than a 2-step approach. Insofar as we are aware, there is no appropriate method for classification using data with all of these characteristics.

To address this, we propose a semi-supervised Bayesian mixture model that explicitly models batch parameters and predicts class membership. The semi-supervised aspect means that observed labels from positive and negative controls are used in the model. The Bayesian framework allows our model to propagate the uncertainty arising from the batch effects to the class allocation probabilities for each item in the dataset. This provides a more complete quantification of the uncertainty in the final predictions, thereby enabling more informed interpretation.

This manuscript is organised as follows: in section 2 we describe our model; in section 3 we evaluate our model using simulated data, and compare to off-the-shelf machine learning methods; and in section 4 we apply the proposed method to two ELISA studies of seroprevalence of SARS-CoV-2 in Stockholm (7) (section 4.1) and Seattle (11) (section 4.3). We then conclude our manuscript in section 5 with a discussion of the contribution, limitations, and possible extensions to our model.

2 Model

2.1 Notation

We consider a study that collects P measurements for each of N individuals to form a dataset $X = (X_1, \ldots, X_N)$, where $X_n = [X_{n,1}, \ldots, X_{n,P}]^\top$ for all $n \in \{1, \ldots, N\}$. We assume that each individual has an associated observed batch label $b_n \in \{1, \ldots, B\} \subset \mathbb{N}$, where B is the total number of batches, and we write $b = [b_1, \ldots, b_N]^\top$ for the collection of all N batch labels. Note that as each individual belongs

to a single batch, we assume that all P measurements for each individual are part of the same batch. We wish to predict class labels for each individual, and write $c = [c_1, \ldots, c_N]^\top$ for the collection of all class labels. We assume that the number of classes, K, is known, so that each $c_n \in \{1, \ldots, K\}$, and introduce a binary vector $\phi = [\phi_1, \ldots, \phi_N]^\top$, such that $\phi_n = 1$ if and only if c_n is known.

2.2 Model specification

We use a K-component mixture model to describe the data X. The mixture model can be be written

$$p(X_n) = \sum_{k=1}^{K} \pi_k f(X_n|\theta_k) \qquad \text{independently for each } n = 1, \ldots, N, \tag{1}$$

where $\pi = [\pi_1, \ldots, \pi_K]^\top$ is the vector of component weights, $f(\cdot)$ is a parametric density function, and θ_k are the parameters of the k^{th} component. We assume each component describes a single and distinct class in the population and use the class labels to rewrite the model

$$p(X_n|c_n = k) = f(X_n|\theta_k). \tag{2}$$

We then introduce batch-specific parameters, $z = (z_1, \ldots, z_B)$ and expand $f(\cdot)$ to accommodate these. Then conditioning on the observed batch label we have

$$p(X_n|c_n = k, b_n = b) = f(X_n|\theta_k, z_b). \tag{3}$$

We focus on continuous data where each measurement has support across the entire real line. We consider the multivariate t density (**MVT**, density denoted $f_t(\cdot)$) and the multivariate normal (**MVN**, density denoted $f_{\mathcal{N}}(\cdot)$) as choices for f, but depending on the situation other choices could be more relevant and our model is not inherently restricted to these. We use $z_b = (m_b, S_b)$, choosing m_b to be a P-vector representing the shift in location due to the batch effects and S_b to be a scaling matrix. We assume the observed location of X_n is composed of a class-specific effect, μ_k, and a batch-specific effect, m_b, so $(X_n|c_n = k, b_n = b) = \mu_k + m_b + \epsilon_n$. Similarly we assume that the random noise, ϵ_n, is subject to class and batch specific effects Σ_k and S_b respectively.

More specifically, if we use a mixture of MVN densities, then our class parameters are $\theta_k = (\mu_k, \Sigma_k)$, where μ_k is the P-dimensional mean vector and Σ_k is the $P \times P$ covariance matrix. We assume

$$X_n|c_n = k, b_n = b \sim \mathcal{N}(\mu_k + m_b, \Sigma_k \oplus S_b). \tag{4}$$

We define the operator $\oplus$ for a $P \times P$ matrix, A, and a diagonal matrix B of equal dimension, as:

$$A \oplus B \coloneqq \begin{pmatrix} a_{1,1}b_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,P} \\ a_{2,1} & a_{2,2}b_{2,2} & a_{2,3} & \cdots & a_{2,P} \\ a_{3,1} & a_{3,2} & a_{3,3}b_{3,3} & \cdots & a_{2,P} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{P,1} & a_{P,2} & a_{P,3} & \cdots & a_{P,P}b_{P,P} \end{pmatrix}. \tag{5}$$

Similarly for a mixture of MVT densities, we assume

$$X_n|c_n = k, b_n = b \sim t_{\eta_k}(\mu_k + m_b, \Sigma_k \oplus S_b). \tag{6}$$

where η_k is the class-specific degrees of freedom.

In the likelihood function, only the combinations of the class and batch parameters, $\mu_k + m_b$ and $\Sigma_k \oplus S_b$, are identifiable, and the values of the class and batch specific effects are not. However, we assume that we have some prior information about the relative orders of magnitude of the class and batch effects and encode this in an informative prior, reducing the problem of identifiability with this additional constraint. If the magnitude of the between-batch variability is similar to or greater than the true biological effect, then we suspect that any analysis of such a dataset is untenable, or at least that the data are not appropriate for our model.

The full hierarchical model can be found in section 1 of the supplementary material. Here we include the choice of prior distributions for the class and batch effects:

$$\mu_k, \Sigma_k|\xi, \kappa, \nu, \Psi \sim \mathcal{N}\left(\mu_k|\xi, \frac{\Sigma_k}{\kappa}\right) \mathcal{IW}\left(\Sigma_k|\nu, \Psi\right), \tag{7}$$

$$m_{b,p}|\delta^2 \sim \mathcal{N}\left(0, \lambda\delta^2\right), \tag{8}$$

$$(S_b)_{p,p}|\alpha, \beta, S_{loc} \sim \mathcal{IG}(\alpha, \beta, S_{loc}), \tag{9}$$

$$\eta_k \sim \mathcal{G}(\epsilon, \zeta) \quad \text{(if the MVT density is being used)}. \tag{10}$$

$\mathcal{IW}$ denotes the inverse-Wishart distribution, $\mathcal{IG}$ denotes the inverse-Gamma distribution with a shape α, rate β and location S_{loc}, $\mathcal{N}$ signifies the Gaussian distribution parameterised by a mean vector and a covariance matrix and $\mathcal{G}$ denotes the Gamma distribution parameterised by a shape and rate. An empirical Bayes approach is used to set the hyperparameters for the class mean and covariance (details are included in section 2 of the Supplementary material, these follow the suggestions of 13). The δ^2 hyperparameter is set to the mean of the diagonal entries of the observed covariance in the data. S_{loc} is set to 1.0 to ensure that the

likelihood covariance matrix remains positive semi-definite. For the MVT mixture model, we choose the hyperparameters of the degrees of freedom to be $\epsilon = 20$ $\zeta = 0.1$ in line with suggestions from Juárez and Steel (19). This uninformative prior does not restrict η_k to small values, and enables the MVT mixture model to approximate the MVN model if the data are truly Gaussian. The remaining hyperparameters (λ, α and β) are user-specified, and we explore the impact of different choices on the final inference in sections 4.1 and 4.3. We investigate the impact of 3 different values for each of these parameters, reflecting an informative or constrained prior, a flexible, uninformed prior, and a choice in the middle-ground.

Sampling the batch and class parameters allows us to derive a batch-corrected dataset, Y, in each iteration. We define the p^{th} measurement for the n^{th} sample in Y as

$$(Y_{n,p}|c_n = k, b_n = b, \ldots) = \frac{X_{n,p} - m_{b,p} - \mu_{k,p}}{(S_b)_{p,p}} + \mu_{k,p}, \tag{11}$$

for all $n = \{1, \ldots, N\}, p = \{1, \ldots, P\}$. Note that Y will incorporate the uncertainty about the batch and class parameters, and the classification. This transformation is similar to the empirical Bayes batch correction suggested by Johnson et al. (18); however their method is a pre-processing step that is applied to each measurement in turn, whereas our model is jointly inferring class and batch effects and may be applied to the full dataset.

We perform inference using a Metropolis-within-Gibbs sampler as described in section 3 of the supplementary material.

3 Simulations

3.1 Simulation design

We wish to evaluate the performance of the MVN and MVT implementations of our model and compare these to the popular machine learning methods random forest (**RF**, 5), probabilistic support vector machine (**SVM**, 4) and logistic regression (without batch-correction, **LR**). We also include the case where each batch is separately mean centred and transformed to have unit variance with logistic regression then applied (**LR - BC**), to show the limitations of a naive batch correction. Our primary interest is in the ability of each method to infer the correct class, the uncertainty quantification about the classification point estimate and time to run the models. We are also interested in inferring the proportion of the second (smaller) class in the dataset; this is the same as seroprevalence in our real data examples.

To achieve this, we generate 10 datasets in each of 6 different scenarios. In all bar one the data are generated from a mixture of MVN distributions. We intend that the underlying class structure, once free of batch effects, is identifiable. Our aim is to show the importance of integrating batch correction into the classification method, since the success of Bayesian mixture models for classification (this has previously been demonstrated, see, e.g. 10). For each simulation we generate both a "batch-free" and an observed dataset. Each contains $P = 2$ measurements for each of 500 samples. The "batch-free" dataset is the observed dataset less the batch effects. It represents solely the influence of the class parameters. The class parameters are chosen to give a separation of 4 between the mean parameters in each dimension (e.g., $\mu_{1,1} = -2, \mu_{2,1} = 2$). We set the covariance matrix to $\sigma^2\mathbf{I}$, where $\sigma = 1.25$ in each class. In the default setting, our "Base case", we generate data from 5 batches. The entries of each batch shift were restricted to one of two option, $m_{b,p} \in (-0.5, 0.5)$. Similarly, $S_{b,p} \in (1.2, 1.5)$. The class weights are uneven, with the first class expected to contribute 75% of the samples with the remainder drawn from class 2. In this scenario the batches are all expected to have equal numbers of samples. We randomly select which class labels are observed, sampling uniformly across the data indices, $\{1, \ldots, N\}$. We expect one quarter of the labels to be observed, i.e. $\mathbb{E}\left(\sum_{n=1}^{N} \phi_n\right) = 0.25N = 125$. These labelled observations constitute the training set for the off-the-shelf methods.

Our six simulation scenarios are:

- Base case: The generic, base scenario; all other scenarios are variations of this, using the same choices for all bar a subset of parameters, with this subset varied to define the specific scenario.

- Batch-free: Similar to the Base case but no batch effects are present (i.e., $m_b = \mathbf{0}_P, S_b = \mathbf{I}$).

- Varying batch effects: the Base case with more variance among the batch effects, $m_{b,p} \in (-1.5, -0.5, 0.0, 0.5, 1.5), S_{b,p} \in (1.0, 1.25, 1.5, 1.75, 2.25)$.

- Varying class representation: the classes are imbalanced across batches, i.e, the expected proportion of each class varies across batches (note that this is a slightly different generating model, the class weights are batch specific). The first two batches contain a larger proportion of samples from class 1, the third batch is balanced and the final two batches have a greater proportion of samples from class 2.

- Varying batch size: rather than equally sized batches, the batches have varying proportions of the total sample. The expected proportions are $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{16}$.

- Multivariate t generated: the data are generated from a MVT mixture model rather than a MVN mixture model. One class is generated from a MVT with 3 degrees of freedom, the other has 5 degrees of freedom.

The parameters that differentiate the scenarios are summarised in table 1, with a more detailed description, along with visualisations of an example dataset for each scenario, provided in section 4 of the supplementary material.

Scenario	B	Class weights	m_b	S_b	Batch weights	η
Base Case	5	Across batch	± 0.5	1.2	Constant	NA
Batch-free	1	Across batch	0.0	1.0	Constant	NA
Varying class representation	5	Within batch	± 0.5	1.2	Constant	NA
Varying batch effects	5	Across batch	Varied	Varied	Constant	NA
Varying batch size	5	Across batch	± 0.5	1.2	Varying	NA
MVT generated	5	Across batch	± 0.5	1.2	Constant	(3, 5)

Table 1: Defining parameters of each simulation scenario.

We use implementations of the machine learning methods available in `R` (31). For the RF this is the `randomForest` package (24), for the SVM we use the `kernlab` package (20), and for LR we use the base implementation of LR contained in the `glm` function. We use the default parameters in each method, bar the SVM where we set `prob.model = TRUE` to build a model for calculating class probabilities. The default for a classification SVM in this package uses a Gaussian Radial Basis kernel function.

We use the data with observed labels as the training set for each of these methods and those with unobserved labels as a test set. We record the time taken to train the model and to predict the outcome for the test set.

3.2 Results

We assessed within-chain convergence by calculating the Geweke statistic (15), and removed chains which failed the diagnostic test. We then considered the trace plots for the complete log-likelihood in the remaining chains as a visual check to identify chains that had not converged. An example of the sequential reductions in chains by this process is shown in figures 8 and 9 of section 5 of the supplementary material.

We compared the models using the F1 score, considering the difference between the predicted labels to the true classes, and the squared Euclidean distance between the allocation probability matrix (a $N \times K$ matrix) to the one-hot-encoding of the true classes (figure 1 A and B). We found that our mixture model performed better or at least as well as the ML methods across all scenarios. When the data were generated from Gaussian distributions, the performance of the two versions of the mixture model performed very similarly. The MVT mixture model learned a large degree of freedom for each component, indicating that

this behaves as an approximation of the Gaussian mixture model when appropriate (figure 2). In contrast, in the Multivariate t generated scenario, the performance of the MVN mixture model had greater variation in performance than in any other scenario. Figure 2 also shows that parameter estimates were consistent across chains.

We also wanted a sense of how well our models would estimate the "seroprevalence" in our simulations. In this case we defined seroprevalence as the proportion of the smaller class in the dataset, and compared the models' estimate to the truth (figure 1 C). We found that the mixture models have a more narrow range in their estimates than the other models in the Base case, No batch effects, Varying batch effects and Varying batch size scenarios, with a similar range for the MVT mixture model in the other two scenarios indicating a more consistent behaviour than the other methods. The MVN mixture model exhibited good behaviour, except when misspecified as in the MVT generated data. We note that the MVT mixture model's estimate tended to be either centred on the true value (MVT generated, No batch effects, Varying class representation in figure 1 C) or else to be slightly lower (Base case, Varying batch effects, Varying batch size in figure 1 C). We also observed that when the batch effects were more varied and greater in magnitude, the SVM and RF had very long tails in their performance (Varying batch effects in figure 1 C). We saw similar behaviour for LR - BC in the F1 score and distance; the imbalance of classes across batches caused the naive batch-correction to be misleading and hence the method performed poorly. LR (without batch correction) was probably the strongest contender to the MVT mixture model in most of our scenarios. This method provided an estimate close to the true value in many simulations, but it has a wider range in its performance across simulations than the MVT.

Logistic regression applied after a batch correction matched the mixture model in performance in three settings: the Base case, the Varying batch effects and the Varying batch size scenarios. However, when the classes were imbalanced across batches, as in the varying class representation scenario, this naive batch correction method performed the worst of all methods. This behaviour for a pre-processing batch correction step and its disadvantages compared to incorporating the batch correction into the modelling is in keeping with results from Leek et al. (22), Li et al. (23).

The Varying class representation scenario was also the setting in which the off-the-shelf methods performed most similarly to our model under the F1 score, but under the squared Euclidean distance our mixture model still performed better, suggesting that the items misclassified by the mixture models had a high uncertainty associated with their classification.

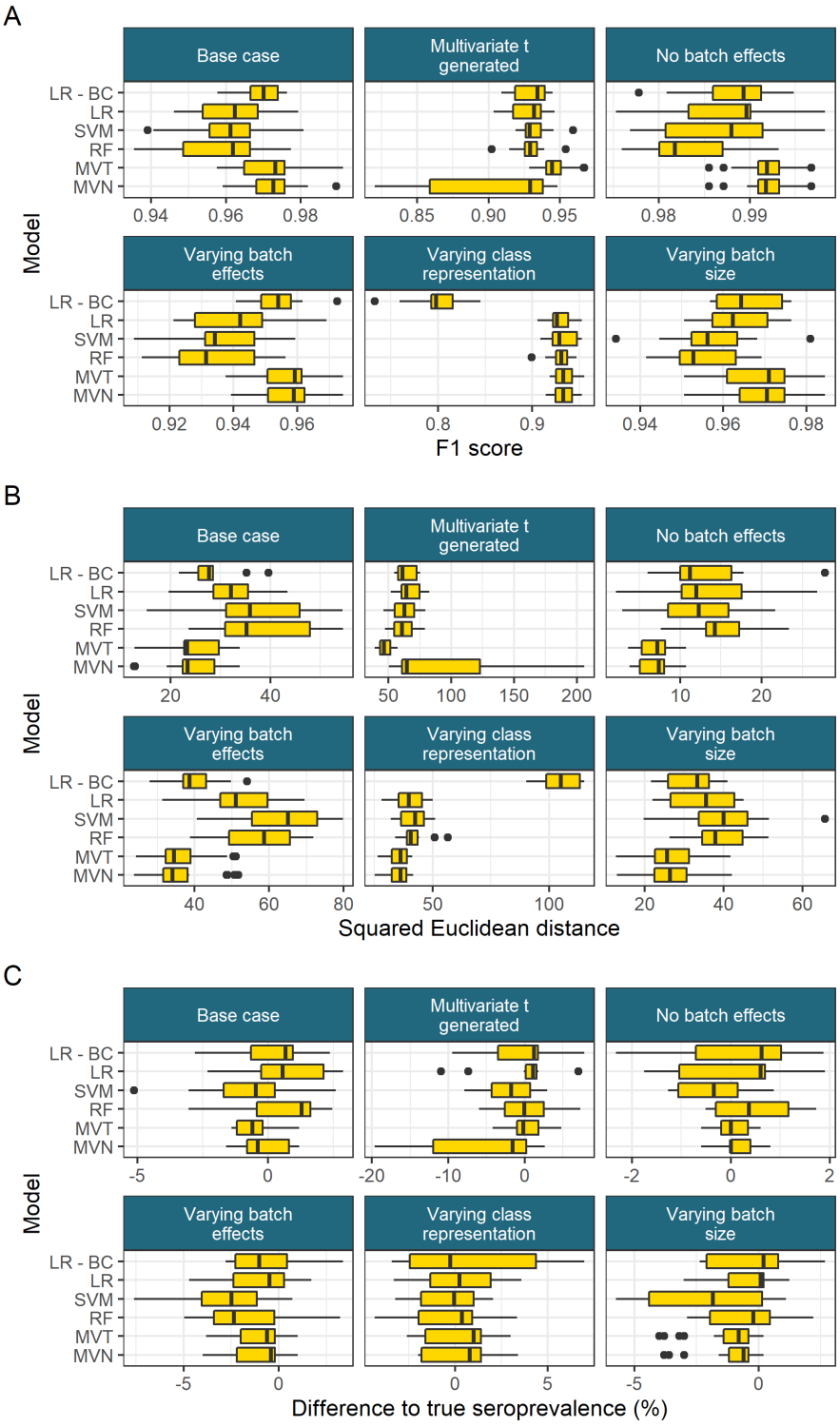

Figure 1: A) F1 score for the predicted classification to the true allocation in test datasets across simulations. B) Squared Euclidean distance between the allocation probability matrix and a one-hot-encoding of the true labels. C) The difference between the point estimate of seroprevalence and the truth across simulations.

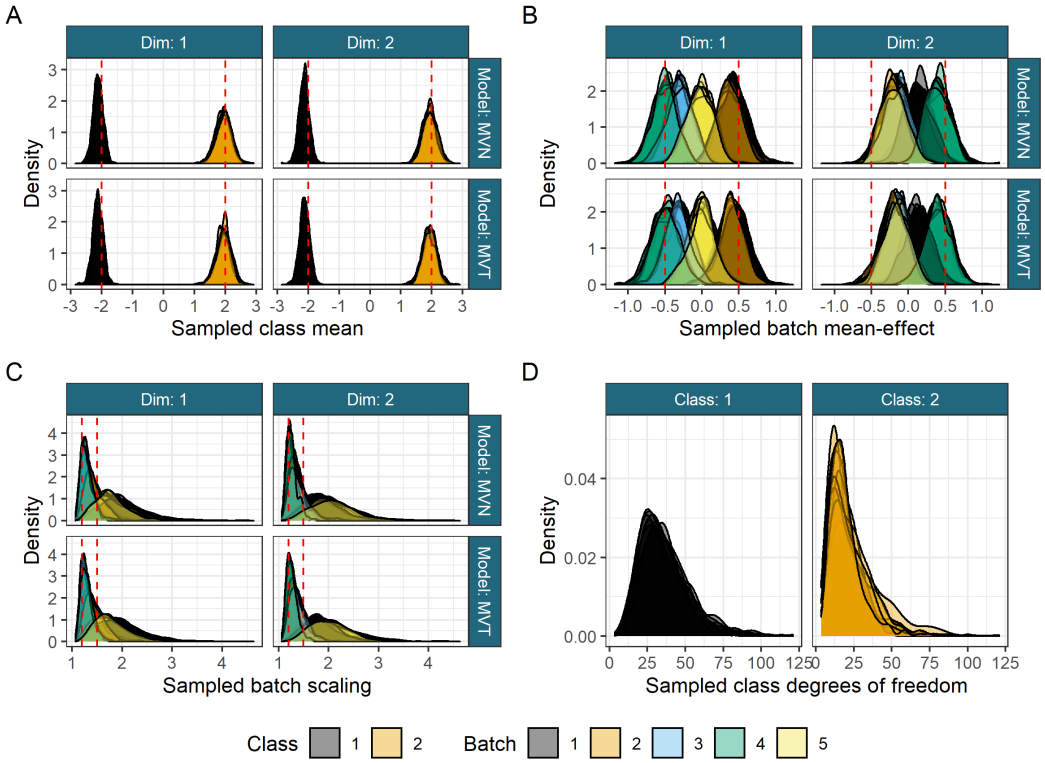

Figure 2: Sampled values for A) the class means, B) the batch mean-effect, C) the batch scaling effect and D) the class degrees of freedom for the well-behaved chains for the first simulated dataset in the Base case scenario. True values are shown by the dashed red vertical lines (as the data are generated from a MVN density there is no true degree of freedom, but larger values better approximate the MVN).

MCMC was slower than the machine learning approaches (table 2), but still reasonable, taking only 7 seconds for 15,000 MCMC iterations (more than enough for chains to converge) for the MVN mixture model and less than 12 seconds for the MVT.

Model	Average time (seconds)
LR	0.003
RF	0.027
SVM	0.049
MVN	7.28
MVT	11.9

Table 2: Average time for each model to converge or, for the Bayesian models, to perform 15,000 iterations across all model runs.

4 ELISA data examples

ELISA is an immunological assay used to measure antibodies, antigens, proteins and glycoproteins, and normally involves a reaction that converts the substrate into a coloured product, the optical density (**OD**)

which can be measured and is then used to determine the antigen concentration. One application is to assess seroprevalence of a disease within a population by measuring seropositivity of antibodies. It has a history of application to a wide range of diseases (e.g., 34, 3, 16, 27) and was used extensively to study seropositivity of antibodies to SARS-CoV-2 antigens used to estimate prevalence of cumulative infection and immunity (11, 26, 33). In such cases it is often possible to include known positive and negative controls as samples (these might be PCR-positive patients and historical samples collected before the pandemic began) and thus a subset of labels are observed.

We investigated the performance of our model on two recent examples of ELISA data, both from studies estimating seroprevalence of SARS-CoV-2. Based on the results from the simulations, we use the MVT as our choice of density, as it always matched or outperformed the MVN mixture in simulations (figure 1).

In the ELISA datasets we do not know the true seropositive status for the non-control data and cannot evaluate the model accuracy. Rather, we present these to demonstrate application of our model and highlight how diagnostic plots and results may be interpreted. In each case we run multiple chains and then use the sampled log-likelihood to assess within and across chain convergence.

Traditional analysis of ELISA data in seroprevalence studies makes dichotomous calls according to thresholds based on the sum of the sample mean and some number of standard deviations of the negative controls in each measurement. However, various choices of the number of standard deviations to use to define the decision boundary are present in the literature (e.g., compare 11, 26, 33).

4.1 Carlos Dopico *et al., 2021*

We used the dataset available from Castro Dopico et al. (7), with the $group$ variable representing the batch divisions. This dataset comprises the log-transformed normalised OD for IgG responses against stabilized trimers of the SARS-CoV-2 spike glycoprotein (SPIKE) and the smaller receptor-binding domain (RBD) in 2,100 sera samples from blood donors, 2,000 samples from pregnant volunteers, 595 historical negative controls, repeatedly sampled, and 149 PCR-positive patients (positive controls from 8). The data were generated across seven batches, with the positive controls contained in two of these. This, combined with our expectation that seropositivity should increase with time as more of the population were exposed to SARS-CoV-2, suggests that the batch and seropositivity frequency are dependent. Based on our simulation study, we would expect that a pre-processing batch normalisation would therefore produce misleading results.

We ran five chains of the MVT mixture model for 50,000 iterations for each of nine combinations of different choices for the hyperparameters of the batch effects in the model (choices in table 3, distributions in figure

3 A and B). The first 20,000 samples were removed as burn-in, and we thinned to every 100th sample to reduce auto-correlation.

	Value								
α	1	5	10	1	5	10	1	5	10
β	3	11	21	3	11	21	3	11	21
λ	0.01	0.01	0.01	0.10	0.10	0.10	1.00	1.00	1.00

Table 3: Hyperparameter combinations used in analysing the data from Castro Dopico et al. (7). The prior expected value of the batch scaling effect is the same for all choices of α and β. The choice of λ represents the scale we *a priori* expect for the batch shift effect.

We chose a representative chain for each hyperparameter combination to estimate the seroprevalence for each week of the year 2020 for which samples are available and compared these to the estimates from Castro Dopico et al. (7) (figure 3 C). Our point estimate was the mean posterior probability of allocation for the non-control data. This was highly consistent across hyperparameter choices and was contained within the confidence interval of the estimate provided by Castro Dopico et al. (7). However, our seroprevalence point estimates, particularly in later dates, were higher than the those from Castro Dopico et al. (7). Table 1 of the Supplementary material shows the point estimate from the ML methods used in the Simulation study, our MVT mixture model and that from the original paper. This shows that while our method provides higher point estimates than those from Castro Dopico et al. (7), the other ML methods (barring the SVM) provide estimates much closer to or even exceeding that from the MVT.

The seroprevalence estimates and their credible intervals were almost identical across hyperparameter choices, suggesting that the classification results are robust to different choices for these hyperparameters. We took a single chain with hyperparameter choice $\alpha = 5, \beta = 11$ and $\lambda = 0.1$ as a representative example. This value of λ represents our expectation that m_b should be approximately an order of magnitude smaller than μ_k. We used this to infer a point classification and a batch-corrected dataset (figure 4 B). Note that the data were on a similar scale to the observed data (figure 4 A), the lack of identifiablity for parameters in the likelihood function did not emerge as a problem here. The batch-corrected dataset was better visually separated into seronegative and seropositive classes than the observed data due to our batch-correction.

To confirm the batch-correction was working as intended we use repeated control samples from a particular patient, "Patient 4". A sample from Patient 4 was included in many plates as a positive control but discarded before our analysis because it was chosen for extremely high antibody levels and so is unrepresentative, even for the seropositive class. We hypothesised that appropriate batch-correction should bring the different measurements of this sample closer together, which is indeed what we observed after applying the correction learnt from the samples excluding Patient 4 (figure 5). Before correction, the batches had no overlap; there was a distance of 0.197 between the batch means. After correction the two batches overlapped with a

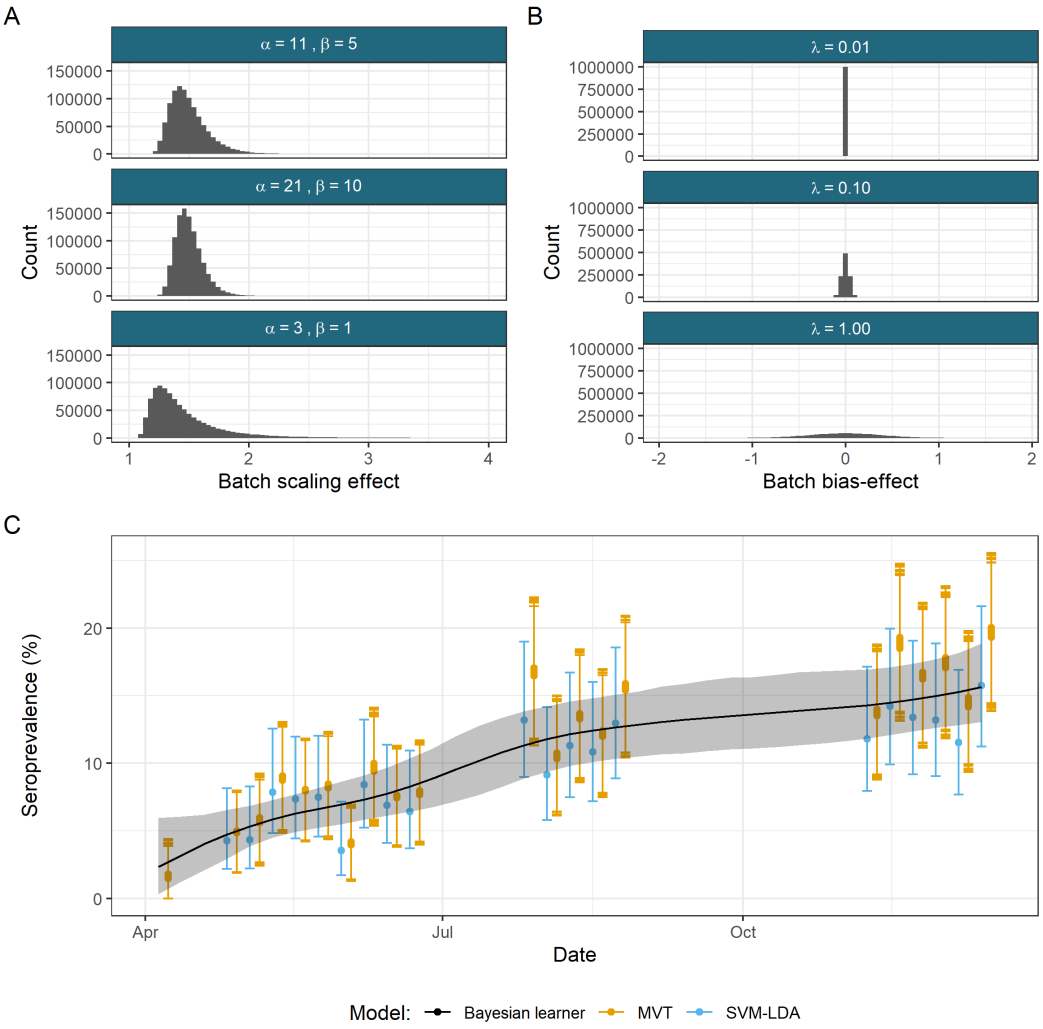

Figure 3: Effect of hyperparameter choices on seroprevalence estimates. One million draws from the prior distributions for the different hyperparameter choices for A) the batch scaling effect and B) the batch shift effect. In A) draws exceeding a value of 4 are hidden. This means that approximately 0.5% of the draws from the prior distribution with a shape of 3 and a scale of 1 are not shown. C) A comparison of the estimated seroprevlaence with population 95% confidence intervals for the MVT mixture model with nine different choices of hyperparameters for the batch-effect prior distributions and the estimates from Castro Dopico et al. (7) for the SVM-LDA ensemble model and the Bayesian learner from Christian and Murrell (9). The Bayesian learner is designed to estimate seroprevalence during an epidemic and provides a smooth, non-decreasing estimate across time. Its assumptions ensure a more consistent increase across time, whereas the SVM-LDA and MVT mixture models are not incorporating any explicit temporal information. The estimates from the mixture model have been moved 3 days to the right on the x-axis to reduce overlap.

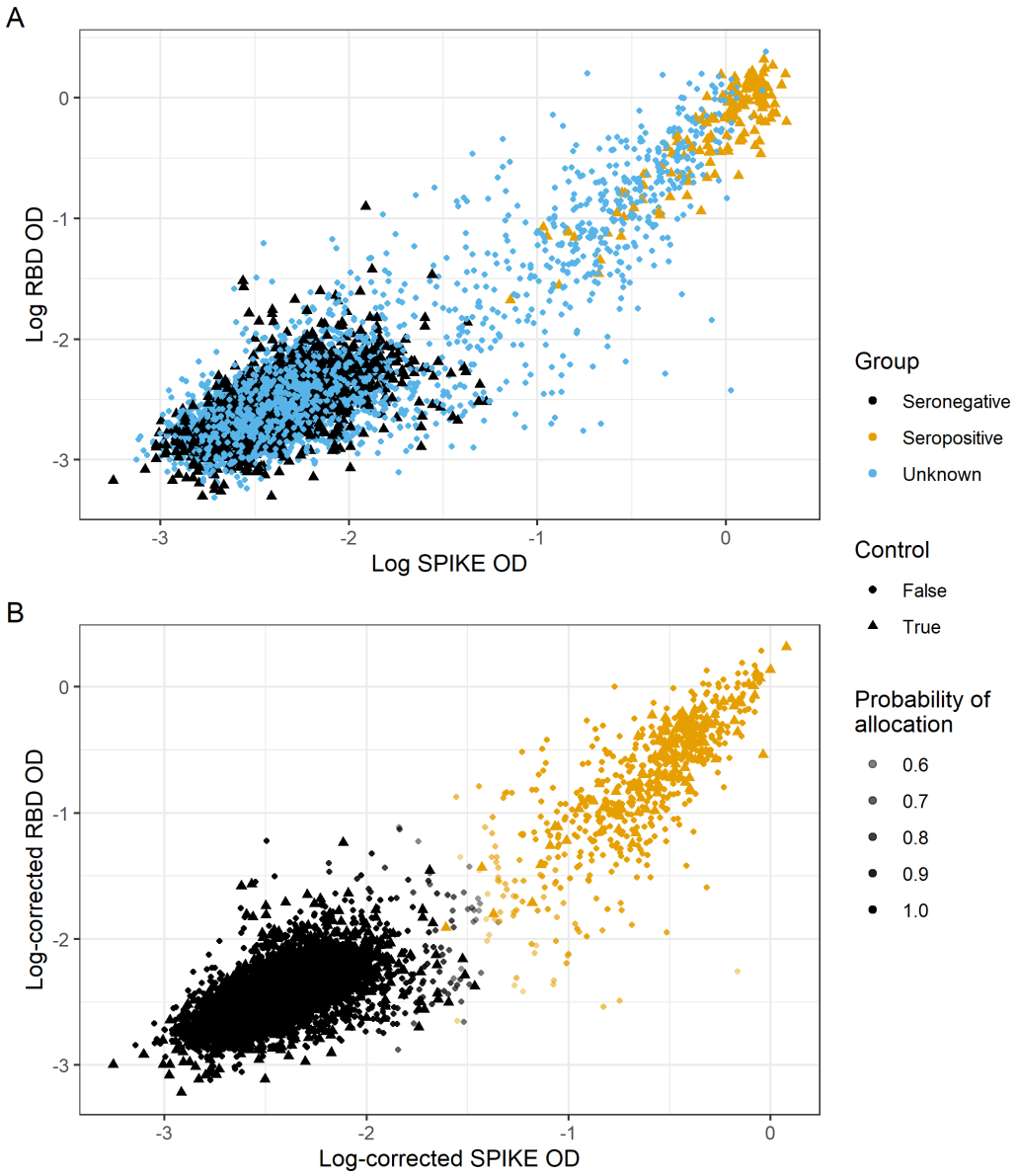

Figure 4: A) The observed data from Castro Dopico et al. (7) and B) the point estimate of the batch-corrected dataset from the MVT mixture model with $\alpha = 11, \beta = 5, \lambda = 0.1$. Points on both plots are coloured by the class. In the observed dataset non-control points are labelled "Unknown" and in the batch-corrected dataset these points are labelled with their inferred class.

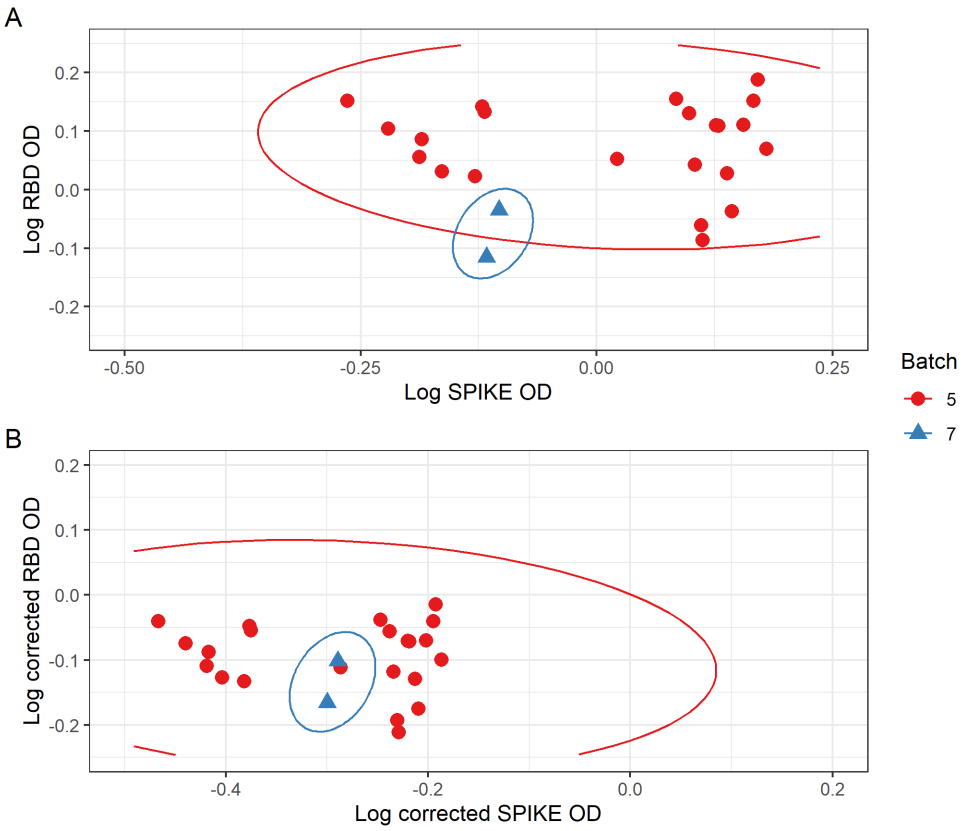

Figure 5: The samples from Patient 4 A) as observed and B) after batch correction, circled by batch.

distance of 0.040 between the means as the points moved closer together and towards the class mean (figure 5 would correspond to the upper right hand of figure 4 A and B). The correction also saw the variation among samples in each batch reduce and become more similar.

4.2 Pseudo-ELISA data

We wished to investigate the possibility that other known positive samples could be more extreme than the non-hospitalised donors. To examine this, we generated datasets from the model fitted in section 4.1. This also tests if the model has learnt representative parameters for the dataset, as our generated data should be very similar to the original data. We used the MCMC sample mean for each parameter except the class weights. For the class weights we used the inferred proportion of each class in each batch to preserve the problem of the imbalance of classes across batches. In the original data, the positive controls were more extreme members of the positive class, having sufficiently severe symptoms to have undergone PCR testing when such resources were severely constrained early in the pandemic. To reflect this in our data generation procedure, we increased the probability that samples with observed positive labels (i.e., the positive

controls) are from the tail of the distribution of the seropositive measurements which is furthest from the seronegative class, whereas the negative controls are sampled uniformly from the seronegative population. An example dataset is shown in figure 6 C, note how closely it resembles the true ELISA data in figure 4 A, suggesting that the model has learnt accurate values. See section 7 of the supplementary material for a deeper explanation of the generation process.

We performed a similar analysis to our original simulation study on these datasets, comparing our models to a range of off-the-shelf machine learning methods. Across all of the simulations, we found that our mixture models outperformed other methods under both the F1 score and the squared distance (figures 6 A, 6 B).

4.3 Dingens et al., 2020

As a final real data example, we analysed the ELISA data collected by Dingens et al. (11). This consisted of 1,891 measurements of antibodies to the SARS-CoV-2 RBD protein. 1,783 of these were from residual serum from Seattle Children's Hospital, with 52 pre-2020 samples used as negative controls and 52 samples from individuals with RT-PCR-confirmed infections as positive controls (figure 7 A). These data are different to the data from Castro Dopico et al. (7) in several ways. There is only a single antigen, there is a smaller ratio of controls to non-controls, particularly for the seronegative samples, and the controls do not appear to be representative of either class. The mean log OD of the negative controls is -1.91, whilst the dataset mean is -2.28 without controls. We analyse the log-transform of the OD using our MVT model for the same variety range of hyperparameter choices as in table 3. An example of a batch-corrected dataset is shown in figure 7 B. We show the comparison of the inferred seroprevalence in each batch for an example chain of each of these models as well as that estimated by Dingens et al. (11) (figure 7 C). The 9 different hyperparameter choices have almost identical seroprevalence estimates and are estimating higher levels of seroprevalence than the estimate provided by Dingens et al. (11).

5 Discussion

The results of our simulation study show that our mixture model consistently matches or outperforms several alternatives when applied to data with batch effects, across a range of data generating models. In the more specific scenario where data were generated from a converged chain that had been applied to the ELISA data from Castro Dopico et al. (7), we obtained the same findings, with our model again performing better than the off-the-shelf machine learning methods. We also see from our simulation study that we should use the MVT density over the MVN density, as the MVT can approximate the MVN quite well by learning a large

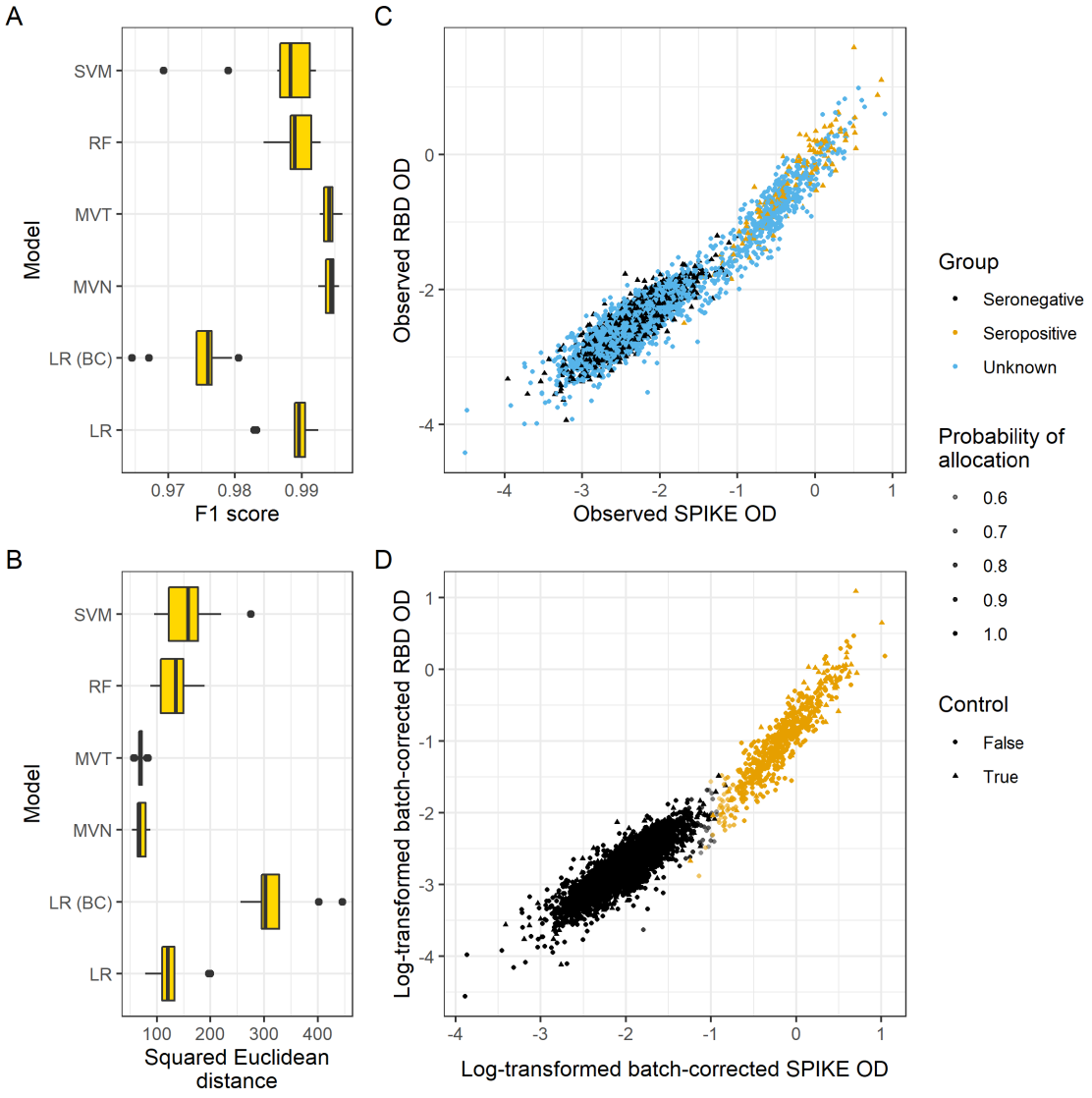

Figure 6: Model comparison for the ELISA-like simulations under the A) F1 score and B) Squared Euclidean distance between the probability allocation matrix and the true classification. B) An example of the simulated data and C) the corresponding inferred dataset for a representative chain of the MVT mixture model.

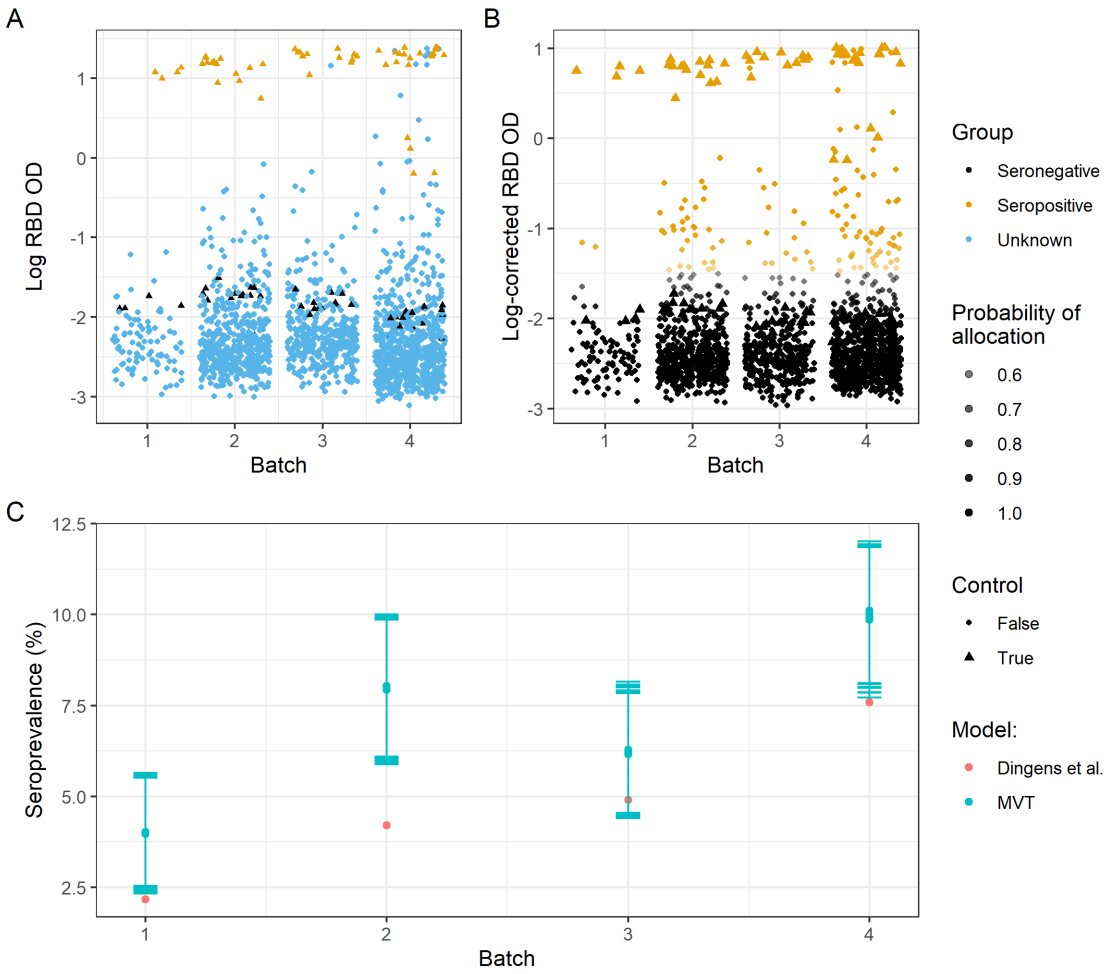

Figure 7: A) The observed data from Dingens et al. (11) and B) the point estimate of the batch-corrected dataset from the MVT mixture model with $\alpha = 11, \beta = 5, \lambda = 0.1$. Points on both plots are coloured by the class. In the observed dataset non-control points are labelled "Unknown" and in the batch-corrected dataset these points are labelled with their inferred class. C) A comparison of the seroprevalence estimate from the MVT mixture model with nine different choices of batch-effect hyperparameters and that from Dingens et al. (11). The error bars indicate the 95% credible interval for the seroprevalence estimates of the MVT mixture model in each batch; this is not available for the estimate from Dingens et al. (11).

degree of freedom, but also has additional flexibility as shown by the Multivariate t generated simulation scenario where the MVN mixture model behaved very inconsistently. The only cost of the MVT mixture model is the approximate 50% increase in runtime, but as our implementation is quite fast we believe that this is not a significant detractor. Based on these results we recommend the use of our MVT mixture model when the analyst suspects the classes in the data may be non-Gaussian.

In terms of estimating seroprevalence, our mixture model performed very well in our simulation study. Using the results shown in figure 1 C, we can try to gauge how well our method is performing in the ELISA data. We would argue that the most pertinent scenarios are the MVT generated (the ELISA data are non-Gaussian), the Varying batch effects and the Varying class representation scenarios. Our method estimates seroprevalence close to the truth, or slightly smaller, in these simulations. Based on this, we suspect that the high estimates of seroprevalence provided by our model (relative to those from the original papers) in the ELISA analyses are plausible.

In the Swedish dataset, we are reassured that the batch-correction is reasonable by our analysis of the patient 4 samples - these samples were used across several batches as positive controls; after applying the correction learnt on the dataset excluding these extreme samples they are no longer separable by batch and have moved towards the class mean. The data generated from our converged model also appears very similar to the observed data, suggesting that the model assumptions are reasonable, and that meaningful estimates of the parameters were obtained.

In the analysis using the data from Dingens et al. (11), the unrepresentative negative controls presented a problem. We believe that the preceding analyses show the potential advantages of our model over existing methods, but this dataset is a good example to show that our method is not a panacea that may overcome all problems - it remains vital to have useful and relevant data in order to perform meaningful inference (12). Any analysis that uses training data that appear to be drawn from a different population than the test data is unlikely to produce meaningful results. Furthermore, the data are not well-described by a pair of MVT distributions (even allowing for our additional flexibility with the batch parameters). This combination of model misspecification and misleading training data makes us skeptical of the inferred parameters.

We note, however, that in the simulation of pseudo-ELISA data, our method still performed strongly despite the positive controls not being representative of the general seropositive sample. In this case our model was correctly specified (the data are generated from a MVT mixture model). In general, we suspect that our method is useful if either the assumption that the labelled data represent their class well or that the model density choice is correct are slightly relaxed, but if both do not hold or if either is profoundly wrong then the model will perform poorly.

Since only the combined class and batch parameters, $\mu_k + m_b$ and $\Sigma_k \oplus S_b$, are identifiable, one might expect this to present challenges when fitting our model. While it is possible that the individual batch and class parameters never stabilise (note that their combinations should converge), running multiple chains helps to avoid this pitfall as one can use the trace plots for the complete likelihood to assess if the chains have reached a common mode in the likelihood surface even if the individual batch and class parameters do not converge. This is standard practice when using stochastic methods, so this aspect of the model should not introduce additional work to the recommended Bayesian workflow (14). Furthermore, from the similarity of the inferred parameters across multiple chains in the Base case simulation (figure 2), we have empirical evidence that this behaviour is not common . We also saw that the seroprevalence estimates and their credible intervals across different hyperparameter choices in the ELISA analyses were well-behaved and, as a result, so was the inferred allocation. This similarity across hyperparameter choice suggests that choosing between specific values is not too important, but we suspect that, if the sample size is smaller, having λ close to one could exacerbate the identifiability problem for the batch shift effect and the class mean. Therefore, we suggest setting $\lambda \leq 0.1$ to encourage these parameters to converge in the small sample setting (although note that their sum, $\mu_k + m_b$, should converge regardless).

We have developed a Bayesian method to predict class membership and perform batch-correction simultaneously, developing on the pre-processing, univariate method of Johnson et al. (18). Our method is intended for low-dimensional data, but the main limitation for higher dimensional data is computational (inverting the covariance matrix becomes very costly) rather than theoretical. Our model is not strictly limited to the semi-supervised setting either; it could be used for unsupervised learning. In this case we expect that the model will rely much more heavily on the distributional assumptions. Our work could be extended to include alternative densities, such as the skew multivariate t. We could extend the model to include batch-specific class weights, such as we used to generate the data in our Varying class representation simulation scenario, or a deeper hierarchy for the batch parameters, such as nested batches (e.g., this could represent scenarios where multiple plates are run at each of multiple time points or locations).

Funding

This work was funded by the MRC (MC UU 00002/4, MC UU 00002/13) and the Wellcome Trust (WT2200788) and supported by the NIHR Cambridge Biomedical Research Centre (BRC-1215-20014). The views expressed are those of the author(s) and not necessarily those of the NHS, the NIHR or the Department of Health and Social Care. This research was funded in whole, or in part, by the Wellcome Trust

[WT107881]. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

Acknowledgements

We would like to thank Dingens et al. (11), specifically Janet A. Englund and Jesse D. Bloom, for being willing to openly share their data and batch information.

Authors' contributions

SC, PK and CW all contributed to model design. SC implemented the model in C++ and built the R package with PK and CW contributing to debugging strategies. SC designed the simulation study and the pseudo-ELISA data. SC, PK and CW all contributed to the design of the Metropolis algorithm used to implement the model and the choice of proposal densities. PK, CW and SC all contributed to analysis and the interpretation of results. XD and GK generated data which CW cleaned. All authors read and approved the manuscript.

References

[1] Emanuele Aliverti, Kristian Lum, James E. Johndrow, and David B. Dunson. Removing the influence of group variables in high-dimensional predictive modelling. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 184(3):791–811, 2021. ISSN 1467-985X. doi: 10.1111/rssa.12613.

[2] O. Alter, P. O. Brown, and D. Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *Proceedings of the National Academy of Sciences*, 97(18):10101–10106, August 2000. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.97.18.10101.

[3] Stuart D. Blacksell, Richard G. Jarman, Robert V. Gibbons, Ampai Tanganuchitcharnchai, Mammen P. Mammen, Ananda Nisalak, Siripen Kalayanarooj, Mark S. Bailey, Ranjan Premaratna, H. Janaka de Silva, Nicholas P. J. Day, and David G. Lalloo. Comparison of Seven Commercial Antigen and Antibody Enzyme-Linked Immunosorbent Assays for Detection of Acute Dengue Infection. *Clinical and Vaccine Immunology*, 19(5):804–810, May 2012. ISSN 1556-6811, 1556-679X. doi: 10.1128/CVI.05717-11.

[4] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, July 1992. Association for Computing Machinery. ISBN 978-0-89791-497-0. doi: 10.1145/130385.130401.

[5] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, October 2001. ISSN 1573-0565. doi: 10.1023/A:1010933404324.

[6] Andrew Butler, Paul Hoffman, Peter Smibert, Efthymia Papalexi, and Rahul Satija. Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nature Biotechnology*, 36(5):411–420, May 2018. ISSN 1546-1696. doi: 10.1038/nbt.4096.

[7] X. Castro Dopico, S. Muschiol, M. Christian, L. Hanke, D. J. Sheward, N. F. Grinberg, J. Rorbach, G. Bogdanovic, G. M. Mcinerney, T. Allander, C. Wallace, B. Murrell, J. Albert, and G. B. Karlsson Hedestam. Seropositivity in blood donors and pregnant women during the first year of SARS-CoV-2 transmission in Stockholm, Sweden. *Journal of Internal Medicine*, May 2021. ISSN 1365-2796. doi: 10.1111/joim.13304.

[8] Xaquin Castro Dopico, Leo Hanke, Daniel J. Sheward, Sandra Muschiol, Soo Aleman, Nastasiya F. Grinberg, Monika Adori, Murray Christian, Laura Perez Vidakovics, Changil Kim, Sharesta Khoenkhoen, Pradeepa Pushparaj, Ainhoa Moliner Morro, Marco Mandolesi, Marcus Ahl, Mattias Forsell, Jonathan Coquet, Martin Corcoran, Joanna Rorbach, Joakim Dillner, Gordana Bogdanovic, Gerald M. McInerney, Tobias Allander, Ben Murrell, Chris Wallace, Jan Albert, and Gunilla B. Karlsson Hedestam. Probabilistic approaches for classifying highly variable anti-sars-cov-2 antibody responses. *medRxiv*, 2021. doi: 10.1101/2020.07.17.20155937. URL `https://www.medrxiv.org/content/early/2021/01/06/2020.07.17.20155937`.

[9] Murray Christian and Ben Murrell. Discriminative Bayesian Serology: Counting Without Cutoffs. *bioRxiv*, 2020. doi: 10.1101/2020.07.14.202150. URL `https://www.biorxiv.org/content/early/2020/07/14/2020.07.14.202150`.

[10] Oliver M. Crook, Claire M. Mulvey, Paul D. W. Kirk, Kathryn S. Lilley, and Laurent Gatto. A Bayesian mixture modelling approach for spatial proteomics. *PLOS Computational Biology*, 14(11):e1006516, November 2018. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1006516.

[11] Adam S. Dingens, Katharine H. D. Crawford, Amanda Adler, Sarah L. Steele, Kirsten Lacombe, Rachel Eguia, Fatima Amanat, Alexandra C. Walls, Caitlin R. Wolf, Michael Murphy, Deleah Pettie, Lauren Carter, Xuan Qin, Neil P. King, David Veesler, Florian Krammer, Jane A. Dickerson, Helen Y. Chu, Janet A. Englund, and Jesse D. Bloom. Serological identification of SARS-CoV-2 infections among children visiting a hospital during the initial Seattle outbreak. *Nature Communications*, 11(1): 4378, September 2020. ISSN 2041-1723. doi: 10.1038/s41467-020-18178-1.

[12] David B. Dunson. Statistics in the Big Data era: Failures of the machine. *Statistics & Probability Letters*, 136:4–9, 2018.

[13] Chris Fraley and Adrian E Raftery. Bayesian Regularization for Normal Mixture Estimation and Model-Based Clustering. *Journal of classification*, page 27, 2007.

[14] Andrew Gelman, Aki Vehtari, Daniel Simpson, Charles C. Margossian, Bob Carpenter, Yuling Yao, Lauren Kennedy, Jonah Gabry, Paul-Christian Bürkner, and Martin Modrák. Bayesian Workflow. *arXiv:2011.01808 [stat]*, November 2020.

[15] John Geweke et al. *Evaluating the Accuracy of Sampling-Based Approaches to the Calculation of Posterior Moments*, volume 196. Federal Reserve Bank of Minneapolis, Research Department Minneapolis, MN, 1991.

[16] Dane Granger, Heather Hilgart, Lori Misner, Jaime Christensen, Sarah Bistodeau, Jennifer Palm, Anna K. Strain, Marja Konstantinovski, Dakai Liu, and Anthony Tran. Serologic testing for Zika virus: Comparison of three Zika virus IgM-screening enzyme-linked immunosorbent assays and initial laboratory experiences. *Journal of clinical microbiology*, 55(7):2127–2136, 2017.

[17] Laleh Haghverdi, Aaron T. L. Lun, Michael D. Morgan, and John C. Marioni. Batch effects in single-cell RNA sequencing data are corrected by matching mutual nearest neighbours. *Nature biotechnology*, 36(5):421–427, June 2018. ISSN 1087-0156. doi: 10.1038/nbt.4091.

[18] W. Evan Johnson, Cheng Li, and Ariel Rabinovic. Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics*, 8(1):118–127, January 2007. ISSN 1468-4357, 1465-4644. doi: 10.1093/biostatistics/kxj037.

[19] Miguel A. Juárez and Mark F. J. Steel. Model-Based Clustering of Non-Gaussian Panel Data Based on Skew- *t* Distributions. *Journal of Business & Economic Statistics*, 28(1):52–66, January 2010. ISSN 0735-0015, 1537-2707. doi: 10.1198/jbes.2009.07145.

[20] Alexandros Karatzoglou, Alex Smola, Kurt Hornik, and Achim Zeileis. kernlab – an S4 package for kernel methods in R. *Journal of Statistical Software*, 11(9):1–20, 2004. URL `http://www.jstatsoft.org/v11/i09/`.

[21] Sharon X. Lee, Geoffrey J. McLachlan, and Saumyadipta Pyne. Modeling of inter-sample variation in flow cytometric data with the joint clustering and matching procedure: Modeling of Inter-Sample Variation. *Cytometry Part A*, 89(1):30–43, January 2016. ISSN 15524922. doi: 10.1002/cyto.a.22789.

[22] Jeffrey T. Leek, Robert B. Scharpf, Héctor Corrada Bravo, David Simcha, Benjamin Langmead, W. Evan Johnson, Donald Geman, Keith Baggerly, and Rafael A. Irizarry. Tackling the widespread and critical impact of batch effects in high-throughput data. *Nature Reviews. Genetics*, 11(10):733–739, October 2010. ISSN 1471-0064. doi: 10.1038/nrg2825.

[23] Tenglong Li, Yuqing Zhang, Prasad Patil, and W. Evan Johnson. Overcoming the impacts of two-step batch effect correction on gene expression estimation and inference. *bioRxiv*, page 2021.01.24.428009, January 2021. doi: 10.1101/2021.01.24.428009.

[24] Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002. URL `https://CRAN.R-project.org/doc/Rnews/`.

[25] J Luo, M Schumacher, A Scherer, D Sanoudou, D Megherbi, T Davison, T Shi, W Tong, L Shi, H Hong, C Zhao, F Elloumi, W Shi, R Thomas, S Lin, G Tillinghast, G Liu, Y Zhou, D Herman, Y Li, Y Deng, H Fang, P Bushel, M Woods, and J Zhang. A comparison of batch effect removal methods for enhancement of prediction performance using MAQC-II microarray gene expression data. *The Pharmacogenomics Journal*, 10(4):278–291, August 2010. ISSN 1470-269X, 1473-1150. doi: 10.1038/tpj.2010.57.

[26] Reuben McGregor, Alana L. Whitcombe, Campbell R. Sheen, James M. Dickson, Catherine L. Day, Lauren H. Carlton, Prachi Sharma, J. Shaun Lott, Barbara Koch, Julie Bennett, Michael G. Baker, Stephen R. Ritchie, Shivani Fox-Lewis, Susan C. Morpeth, Susan L. Taylor, Sally A. Roberts, Rachel H. Webb, and Nicole J. Moreland. Collaborative networks enable the rapid establishment of serological assays for SARS-CoV-2 during nationwide lockdown in New Zealand. *PeerJ*, 8:e9863, September 2020. ISSN 2167-8359. doi: 10.7717/peerj.9863.

[27] Caroline E. Mullis, Oliver Laeyendecker, Steven J. Reynolds, Ponsiano Ocama, Jeffrey Quinn, Iga Boaz, Ronald H. Gray, Gregory D. Kirk, David L. Thomas, and Thomas C. Quinn. High frequency of false-positive hepatitis C virus enzyme-linked immunosorbent assay in Rakai, Uganda. *Clinical infectious diseases*, 57(12):1747–1750, 2013.

[28] S. K. Ng, G. J. McLachlan, K. Wang, L. Ben-Tovim Jones, and S.-W. Ng. A Mixture model with random-effects components for clustering correlated gene-expression profiles. *Bioinformatics*, 22(14): 1745–1752, July 2006. ISSN 1367-4803, 1460-2059. doi: 10.1093/bioinformatics/btl165.

[29] Vegard Nygaard, Einar Andreas Rødland, and Eivind Hovig. Methods that remove batch effects while retaining group differences may lead to exaggerated confidence in downstream analyses. *Biostatistics (Oxford, England)*, 17(1):29–39, January 2016. ISSN 1465-4644. doi: 10.1093/biostatistics/kxv027.

[30] Sandhya Prabhakaran, Elham Azizi, Ambrose Carr, and Dana Pe'er. Dirichlet Process Mixture Model for Correcting Technical Variation in Single-Cell Gene Expression Data. *International Conference on Machine Learning*, page 10, June 2016.

[31] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2021. URL `https://www.R-project.org/`.

[32] Ronald P. Schuyler, Conner Jackson, Josselyn E. Garcia-Perez, Ryan M. Baxter, Sidney Ogolla, Rosemary Rochford, Debashis Ghosh, Pratyaydipta Rudra, and Elena W. Y. Hsieh. Minimizing Batch Effects in Mass Cytometry Data. *Frontiers in Immunology*, 10:2367, October 2019. ISSN 1664-3224. doi: 10.3389/fimmu.2019.02367.

[33] Daniel Stadlbauer, Fatima Amanat, Veronika Chromikova, Kaijun Jiang, Shirin Strohmeier, Guha Asthagiri Arunkumar, Jessica Tan, Disha Bhavsar, Christina Capuano, Ericka Kirkpatrick, Philip Meade, Ruhi Nichalle Brito, Catherine Teo, Meagan McMahon, Viviana Simon, and Florian Krammer. SARS-CoV-2 Seroconversion in Humans: A Detailed Protocol for a Serological Assay, Antigen Production, and Test Setup. *Current Protocols in Microbiology*, 57(1):e100, 2020. ISSN 1934-8533. doi: 10.1002/cpmc.100.

[34] A. Voller, D. Bidwell, G. Huldt, and E. Engvall. A microplate method of enzyme-linked immunosorbent assay and its application to malaria. *Bulletin of the World Health Organization*, 51(2):209, 1974.

[35] Brian W. Whitcomb, Neil J. Perkins, Paul S. Albert, and Enrique F. Schisterman. Treatment of Batch in the Detection, Calibration, and Quantification of Immunoassays in Large-scale Epidemiologic Studies. *Epidemiology (Cambridge, Mass.)*, 21(Suppl 4):S44–S50, July 2010. ISSN 1044-3983. doi: 10.1097/EDE.0b013e3181dceac2.

A semi-supervised Bayesian mixture modelling approach for joint batch correction and classification: Supplementary material

Stephen Coleman[1,*]
stephen.coleman@mrc-bsu.cam.ac.uk
Xaquin Castro Dopico[2]
xaquin.castro.dopico@ki.se
Gunilla B. Karlsson Hedestam[2]
gunilla.karlsson.hedestam@ki.se
Paul D.W. Kirk[1,3,†]
paul.kirk@mrc-bsu.cam.ac.uk
Chris Wallace[1,3,†]
cew54@cam.ac.uk

[1] MRC Biostatistics Unit
[3] Cambridge Institute of Therapeutic Immunology & Infectious Disease
University of Cambridge, U.K.

[2] Department of Microbiology, Tumor and Cell Biology
Karolinska Institutet, Sweden.

∗ Corresponding author.
† These authors provided an equal contribution.

Abstract

Description of the model, our choice of priors, and the sampling algorithm. Example of likelihood trace plots for model convergence. Description of how the simulated data is generated for both the main simulation study and the pseudo-ELISA simulation.

1 Model

Our data $X = (X_1, \ldots, X_N)$ is generated across B batches where the origin batch of each point is known and represented by the vector $b = (b_1, \ldots, b_N)$. We are interested in classifying X into K disjoint classes. We model X using a K component mixture model:

$$p(X|b_n = b, \theta, \psi) = \sum_{k=1}^{K} \pi_k f(X_n|\theta_k, z_b). \tag{1}$$

Here $f(\cdot)$ is the density function, $\pi = (\pi_1, \ldots, \pi_K)$ are the component or class weights, $\theta = (\theta_1, \ldots, \theta_K)$ are the parameters describing the classes and $z = (z_1, \ldots, z_B)$ are the parameters associated with the batches. We introduce an allocation variable, $c = (c_1, \ldots, c_N)$, to represent the class membership and assume that each class is represented by a single component of the mixture. Conditioning on c, our model is then

$$p(X_n|b_n = b, c_n = k, \theta, \psi) = f(X_n|\theta_k, z_b). \tag{2}$$

For us, c contains some observed values (alternatively, c contains missing values), this enables supervised or semi-supervised methods to infer the missing values. We introduce a binary vector, $\phi = (\phi_1, \ldots, \phi_N)$, indicating if the label of the n^{th} individual is observed or not. If we separate our dataset into subsets

$$X_{train} = \{X_n \in X : \phi_n = 1\}, \tag{3}$$
$$X_{test} = \{X_n \in X : \phi_n = 0\}. \tag{4}$$

and use X_{train} to train some classifier which predicts the labels of X_{test}, we would be in traditional prediction territory. However, the Bayesian framework enables us to integrate these steps, seamlessly incorporating information from the allocations from X_{test} into the class parameters while maintaining the information from X_{train}.

1.1 Multivariate Normal

Let f be the density function for the multivariate normal distribution, parametrised by a mean vector μ and a covariance matrix Σ.

We assume

$$\begin{aligned} &X_n|c_n, b_n, \ldots \sim \mathcal{N}(\mu_{c_n} + m_{b_n}, \Sigma_{c_n} \oplus S_{b_n}), \\ &\implies p(X_n|\cdot) = \left[(2\pi)^P |\Sigma_{c_n} \oplus S_{b_n}|\right]^{-1/2} \\ &\qquad \times \exp\left\{-\frac{1}{2}\left[X_n - (\mu_{c_n} + m_{b_n})\right]^T (\Sigma_{c_n} \oplus S_{b_n})^{-1} \left[X_n - (\mu_{c_n} + m_{b_n})\right]\right\}. \end{aligned}$$

We also assume that the batch effects have no correlation across dimensions. We restrict the covariance matrix, S_b, to being diagonal and assume independence between the entries of m_b.

Our hierarchical model is

$$\mu_k, \Sigma_k|\xi, \kappa, \nu, \Psi \sim \mathcal{N}\left(\mu_k|\xi, \frac{\Sigma_k}{\kappa}\right) \mathcal{IW}\left(\Sigma_k|\nu, \Psi\right), \tag{5}$$
$$m_{b,p}|\lambda, \delta^2 \sim \mathcal{N}\left(0, \lambda\delta^2\right), \tag{6}$$
$$(S_b)_{p,p}|\alpha, \beta, S_{loc} \sim \mathcal{IG}(\alpha, \beta, S_{loc}), \tag{7}$$
$$\pi|\gamma \sim Dir(\gamma/K, \ldots, \gamma/K), \tag{8}$$
$$c_n|\pi \sim Cat(\pi), \tag{9}$$
$$X_n|c_n = k, b_n = b, \mu_k, \Sigma_k, m_b, S_b \sim \mathcal{N}(\mu_k + m_b, \Sigma_k \oplus S_b). \tag{10}$$

$\mathcal{IW}$ denotes the inverse-Wishart distribution, $\mathcal{IG}$ denotes the inverse-Gamma distribution with a shape α, rate β and location S_{loc}. $\mathcal{N}$ is the Gaussian distribution, Dir is the Dirichlet distribution and Cat is the categorical distribution. As we assume independence of batch effects across dimensions, we model each entry of the b^{th} batch mean vector, $m_{b,p}$, and the b^{th} batch covariance matrix, $(S_b)_{p,p}$, using one dimensional distributions.

The total joint probability is

$$\begin{aligned} p(X, \mu, \Sigma, m, S, \pi, c|b) &= p(\pi|\gamma)p(X, c|\mu_k, \Sigma_k, m_b, S_b, b) \\ &\quad \times \prod_{k=1}^{K} p(\mu_k|\xi, \Sigma_k, \kappa)p(\Sigma_k|\nu, \Psi) \\ &\quad \times \prod_{b=1}^{B}\prod_{p=1}^{P} p(m_{b,p}|\lambda, \delta^2)p((S_b)_{p,p}|\alpha, \beta, S_{loc}) \\ &= f_{Dir}(\gamma) \prod_{n=1}^{N}\sum_{k=1}^{K} \pi_k f_{\mathcal{N}}(X_n|\mu_k + m_b, \Sigma_k \oplus S_b) \\ &\quad \times \prod_{k=1}^{K} f_{\mathcal{N}}(\mu_k|\xi, \Sigma_k, \kappa) f_{\mathcal{IW}}(\Sigma_k|\nu, \Psi) \\ &\quad \times \prod_{b=1}^{B}\prod_{p=1}^{P} f_{\mathcal{N}}(m_{b,p}|0, \lambda\delta^2) f_{\mathcal{IG}}((S_b)_{p,p}|\alpha, \beta, S_{loc}). \end{aligned}$$

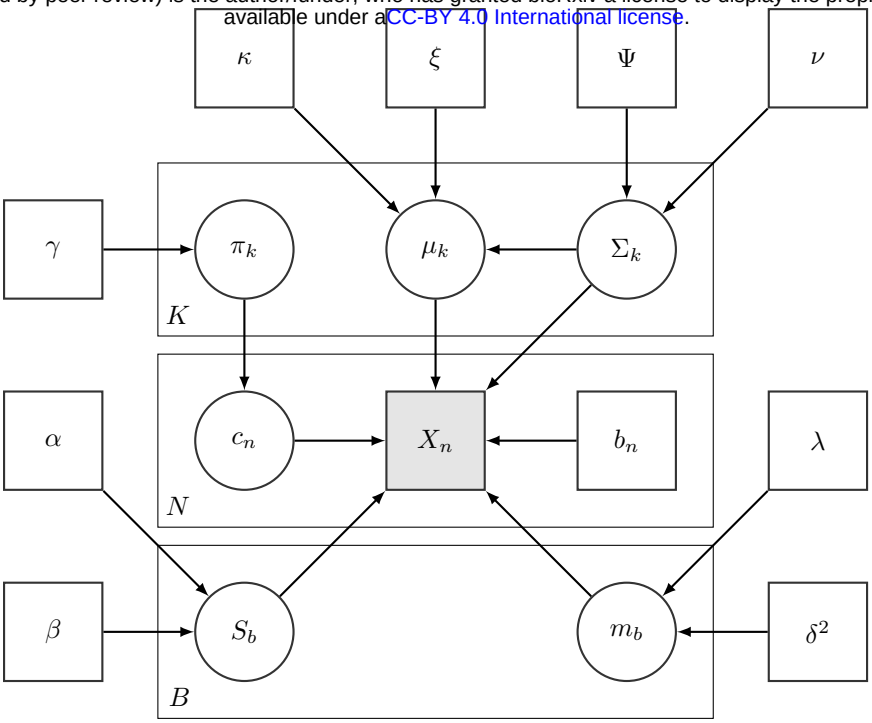

Figure 1: Directed acyclic graph for mixture of multivariate normal distributions with random effects.

1.2 Multivariate t

If we let f be the density function for the multivariate t (**MVT**) distribution, parametrised by a mean vector μ, a covariance matrix Σ and degrees of freedom, η, then the model remains as described in section 1.1 and equations 5, except the model likelihood changes and we introduce a prior distribution over η:

$$\eta_k \sim \mathcal{G}(\epsilon, \zeta), \tag{11}$$

$$X_n | c_n = k, b_n = b, \mu_k, \Sigma_k, \eta_k, m_b, S_b \sim t_{\eta_k}(\mu_k + m_b, \Sigma_k \oplus S_b). \tag{12}$$

here $\mathcal{G}$ denotes the Gamma distribution parametrised by a shape and rate.

The total joint probability for the mixture of MVT distributions is

$$\begin{aligned}
p(X, \mu, \Sigma, \eta, m, S, \pi, c|b) &= p(\pi|\gamma)p(X, c|\mu_k, \Sigma_k, m_b, S_b, b, \eta_k) \\
&\quad \times \prod_{k=1}^{K} p(\mu_k|\xi, \Sigma_k, \kappa)p(\Sigma_k|\nu, \Psi)p(\eta_k|\epsilon, \zeta) \\
&\quad \times \prod_{b=1}^{B}\prod_{p=1}^{P} p(m_{b,p}|\lambda\delta^2)p((S_b)_{p,p}|\alpha, \beta, S_{loc}) \\
&= f_{Dir}(\gamma) \prod_{n=1}^{N}\sum_{k=1}^{K} \pi_k f_t(X_n|\mu_k + m_b, \Sigma_k \oplus S_b, \eta_k) \\
&\quad \times \prod_{k=1}^{K} f_{\mathcal{N}}(\mu_k|\xi, \Sigma_k, \kappa) f_{\mathcal{IW}}(\Sigma_k|\nu, \Psi) f_{\mathcal{G}}(\eta_k|\epsilon, \zeta) \\
&\quad \times \prod_{b=1}^{B}\prod_{p=1}^{P} f_{\mathcal{N}}(m_{b,p}|0, \delta^2) f_{\mathcal{IG}}((S_b)_{p,p}|\alpha, \beta, S_{loc}).
\end{aligned}$$

1.3 Parameter interpretation

Note that the "batch" parameters should not be inferred as direct estimates of the effect the batches have on the true measures. As we are essentially performing a classification on the inferred batch-free

dataset,

$$(Y_{n,p}|c_n = k, b_n = b, \ldots) = \frac{X_{n,p} - m_{b,p} - \mu_{k,p}}{(S_b)_{p,p}} + \mu_{k,p}, \tag{13}$$

$$p(Y_n|\mu, \Sigma, \pi_k) = \sum_{k=1}^{K} \pi_k p(Y_n|\mu_k, \Sigma_k), \tag{14}$$

and the likelihood parameters of $\mu_k + m_b$ and $\Sigma_k \oplus S_b$ are not constrained in the likelihood, we recommend that users focus on the relative change in the measurements for batches, the inferred dataset and the inferred classification rather than the direct meaning of individual parameters.

2 Empirical Bayes

We use the suggestions of Fraley and Raftery (2007) for our choices of prior hyperparameters on the class parameters.

$$\xi = \frac{1}{N} \sum_{n=1}^{N} X_n, \tag{15}$$

$$\kappa = 0.01, \tag{16}$$

$$\nu = P + 2. \tag{17}$$

The choice of ξ is self-explanatory. κ can be viewed as the number of observations contributing to the prior. Fraley and Raftery (2007) choose a value based on experiments to acquire a BIC curve that is a smooth extension of the counterpart without a prior. The marginal prior distribution of μ_k is a Student's t distirbution centred at ξ with $\nu - P + 1$ degrees of freedom. ν is the smallest integer value for the degrees of freedom that gives a finite variance.

We set Ψ as a diagonal matrix. Let

$$\Sigma_0 = \frac{1}{N-1} \sum_{n=1}^{N} (X_n - \xi)(X_n - \xi)^T, \tag{18}$$

$$\bar{\sigma}_0^2 = \frac{1}{P} \sum_{p=1}^{P} (\Sigma_0)_{p,p}, \tag{19}$$

then

$$\Psi_{p,p} = \frac{\bar{\sigma}_0^2}{K^{2/P}}. \tag{20}$$

The logic is that the mixture components are expected, *a priori*, to each fill a common fraction of the total volume of space the data occupies.

For the concentration on the class weights, we use a flat prior with $\gamma = 1$. In our motivating exmple of ELISA data, we cannot use more information (such as the ratio of class members in the known data), as the negative controls are historical samples the number of which is chosen before the experiment and is not related to the expected seroprevalence in the dataset.

For the degrees of freedom for the MVT, η_k, we use an uniformative prior that offers a range of plausible values, $\epsilon = 2.0, \zeta = 0.1$ (Juárez and Steel, 2010).

3 Sampling algorithm

We use a *Metropolis-within-Gibbs* algorithm to sample our parameters. All parameters where the form of their posterior distribution is known are sampled via Gibbs sampling (Geman and Geman, 1984), the remaining parameters are sampled in a Metropolis-Hastings step (Metropolis et al., 1953; Hastings, 1970).

Algorithm 1: $sampler(X, I, c_0, fixed, b, K)$

Input:
Data X,
The number of iterations, I,
Initial classification, c_0,
Fixed labels, $fixed$,
Batch membership, b,
The number of classes to model, K,
The prior distributions for each parameter,
The likelihood function, $p(X|\cdot)$,
The proposal distributions for each class and batch parameter, $q(\theta)$.
Output: A Markov chain of accepted values for each of the sampled parameters.
begin
  `/* initialise parameters by drawing from the prior */`
  $sampleFromPriors()$;
  for $i = 1$ **to** I **do**
    `/* Update the class weights in a Gibbs step */`
    $\pi \leftarrow updateWeights(c, \gamma)$;
    `/* Update the class and batch parameters in a Metropolis-Hastings step */`
    for $k = 1$ **to** K **do**
      $\Sigma_k^i \leftarrow metropolisHastings(\Sigma_k^{i-1}, \nu_\Sigma, q_\Sigma(\cdot))$;
      $\mu_k^i \leftarrow metropolisHastings(\mu_k^{i-1}, \sigma_\mu^2 \mathbf{I}, q_\mu(\cdot))$;
    for $b = 1$ **to** B **do**
      for $p = 1$ **to** P **do**
        $(S_b^i)_{p,p} \leftarrow metropolisHastings(((S_b^{i-1})_{p,p}, \beta_S, q_S(\cdot))$;
      $m_b^i \leftarrow metropolisHastings(m_b^{i-1}, \sigma_m^2 \mathbf{I}, q_m(\cdot))$;
    `/* Update the class allocations */`
    $c \leftarrow updateAllocations(X, b, \pi, fixed)$;
    `/* Update the batch corrected data based on the current parameters. */`
    $Y \leftarrow batchCorrected(X, c, b, \mu, m, S)$;

Algorithm 2: $sampleFromPriors()$

Output: Initial values for class and batch parameters.
begin
  for $k = 1$ **to** K **do**
    $\Sigma_k \sim \mathcal{IW}(\nu, \Psi)$;
    $\mu_k \sim \mathcal{N}(\xi, \Sigma_k/\kappa)$;
  for $b = 1$ **to** B **do**
    for $p = 1$ **to** P **do**
      $(S_b)_{p,p} \sim \mathcal{IG}(\alpha, \beta, S_{loc})$;
      $m_{b,p} \sim \mathcal{N}(0, \delta^2)$;

Algorithm 3: *updateAllocation*$(X, b, \pi, fixed)$

Input:
X, the observed data,
b, the batch variable,
π, the class weights,
$fixed$, the binary vector indicating if the label is known.
Output: c, a new allocation vector.
begin
  for $n = 1$ **to** N **do**
    `/* If the item's class is unknown, update. */`
    if $fixed_n == 0$ **then**
      $ll \leftarrow logLikelihood(X_n, b_n)$;
      $ll \leftarrow ll + \log \pi$;
      `/* Handle overflow and normalise. */`
      $ll \leftarrow \exp(ll - \max(ll))$;
      $ll \leftarrow ll/sum(ll)$;
      `/* update class. */`
      $u \sim \mathcal{U}(0, 1)$;
      $c_n \leftarrow sum(u > cumsum(ll))$;

Algorithm 4: *updateWeights*(c, γ)

Input:
c, the current allocation,
γ, the prior concentration vector for the class weights.
Output: π, a new class weight vector.
begin
  for $k = 1$ **to** K **do**
    $members_k \leftarrow which(c == k)$;
    $N_k \leftarrow count(members_k)$;
    `/* the concentration for` pi_k `is the sum of the count of class members and the prior concentration. */`
    $\gamma \leftarrow \gamma_k + N_k$;
    $\pi_k \sim \mathcal{G}(\gamma, 1.0)$;
  `/* convert the weights from a Gamma random variable to a Dirichlet (or, if` $K = 2$`, a Beta) random variable. */`
  $\pi \leftarrow \pi/sum(\pi)$;

Algorithm 5: $batchCorrected(X, c, b, \mu, m, S)$

Input:
X, the observed dataset,
c, the allocation vector,
b, the batch label vector,
μ, the class means,
m, the batch effect on the class means,
S, the batch effect on the class standard deviations.
Output: Y, the batch-corrected dataset.
begin
  `/* Iteratve over points performing batch correction. */`
  for $n = 1$ **to** N **do**
    `/* Extract the current point's class and batch. */`
    $k \leftarrow c_n$;
    $b \leftarrow b_n$;
    `/* Remove the inferred batch effect. */`
    for $n = 1$ **to** N **do**
      $Y_{n,p} \leftarrow (X_{n,p} - \mu_{k,p} - m_{b,p})/(S_b)_{p,p} + \mu_{k,p}$;

Algorithm 6: $metropolisHastings(\theta, \sigma^2_{win}, q(\cdot))$

Input:
Current parameter value θ,
Proposal window, σ^2_{win},
The proposal distribution, $q(\theta, \sigma^2_{win})$,
The prior distribution for θ, $p(\theta)$,
The likelihood of θ, $p(X|\theta)$.
Output: A value θ^*.
begin
  `/* sample a proposal for` θ `*/`
  $\theta' \sim q(\theta, \sigma^2_{win})$;
  `/* calculate the accpetance probability (note that if` $q(\cdot)$ `is a symmetric distribution it cancels out) */`
  $\alpha \leftarrow \min\left(1, \frac{p(X|\theta')p(\theta')q(\theta|\theta')}{p(X|\theta)p(\theta)q(\theta'|\theta)}\right)$;
  $u \sim Unif(0, 1)$;
  if $u < \alpha$ **then**
    $\theta^* \leftarrow \theta'$;
  else
    $\theta^* \leftarrow \theta$;

3.1 Proposal distributions

For our batch and class parameters, we choose proposal densities that have an expectation of the current value and have the correct support. The class and batch means have a support (∞, ∞); this allows use of a Gaussian proposal distribution with a mean of the current value.

$$m_b^* \sim \mathcal{N}(m_b, \sigma_m^2 \mathbf{I}), \tag{21}$$

$$\mu_k^* \sim \mathcal{N}(\mu_k, \sigma_\mu^2 \mathbf{I}). \tag{22}$$

This density is symmetric and the relationship between the acceptance rate and the choice of the proposal window (σ_m^2 and σ_μ^2) is relatively intuitive, the acceptance rate will decrease as the window increases.

The batch standard deviations have a support of (S_{loc}, ∞). To ensure that proposed values remain in this range we use a Gamma proposal distribution with a shape of the current value divided by the rate, the rate set to some constant and a location of S_{loc}.

$$(S_b^*)_{p,p} \sim \mathcal{G}((S_b)_{p,p}/\beta_S, \beta_S, S_{loc}). \tag{23}$$

This proposal has an expected value of $(S_b)_{p,p}$. However, it is asymmetric and the acceptance rate increases as β_S increases. We propose all P members of S_b in each sampling step.

The class covariance matrices are the most difficult to sample. There are P^2 values to propose and must be positive semi-definite. We use a Wishart proposal to satisfy this

$$\Sigma_k^* \sim \mathcal{W}(\nu_\Sigma, \Sigma_k). \tag{24}$$

All of the proposal windows, $(\sigma_\mu^2, \sigma_m^2, \beta_S, \nu_\Sigma)$, are tuned aimming to achieve acceptance rates in the range $[0.1, 0.5]$ (Roberts and Rosenthal, 2001); if this is not possible we prioritise keeping acceptance rates above 0.1. This can involve multiple tuning runs of the sampler on each dataset.

4 Simulation study

We use a simulation study to test the model behaviour in examples where the generating model and the true labelling are known. We aim to explore

- the batch effects inferred by the model when none are present.
- the sampled distributions of the degree of freedom parameters in the mixture of multivariate t distributions.
- how the model behaves when there is some sort of inequality in the batches, e.g.,
  - different batch sizes,
  - different class representation in each batch, and
  - large difference in the magnitude of batch effects.

4.1 Design

Our study uses six different scenarios to test and benchmark behaviour. We use a *Base case* as the default scenario that each other scenario is a variation of. For example, the *No batch effects* scenario is the Base case with the batch means set to 0 and the batch standard deviations set to 1.0. We define each scenario by a set of parameters

N : the number of rows in the dataset,
P : the number of features in the dataset,
K : the number of classes in the dataset,
B : the number of batches in the dataset,
$\Delta\mu_{k,p}$: the cluster means before the batch effects,
$\sigma_{k,p}$: the cluster standard deviations before batch effects,
π_k : the expected class representations,
m_b : the batch effect on the means,
S_b : the batch effect on the standard deviations,
w_b : the expected proportion of the dataset in each batch.

We use the distance between cluster means in a single dimension, as this is the quantity of interest rather than specific values of μ_k.

To generate the datasets, we first sample batch and class labels based on w_b and π_k respectively. The measurements for each point are then generated from a Gaussian distribution defined by these labels (except in the *multivariate t generated* scenario where the generating distribution is the eponymous distribution). We use a diagonal covariance matrix for simplicity. Each column generated randomly permutes the parameters associated with each class and batch; this means that the different columns can contain different information.

$$b_n \sim Cat(w), \tag{25}$$
$$c_n \sim Cat(\pi), \tag{26}$$
$$Y_n \sim \mathcal{N}(\mu_{c_n}, \Sigma_{c_n}), \tag{27}$$
$$X_n \sim \mathcal{N}(Y_n + m_{b_n}, S_{b_n}). \tag{28}$$

4.1.1 Base case

The parameters defining each simulation in the scenario are

$$\begin{aligned} N &= 500, \\ P &= 2, \\ K &= 2, \\ B &= 5, \\ \Delta\mu_{k,p} &= 2, \\ \sigma_{k,p} &= 2, \\ \pi^T &= (0.75, 0.25), \\ m_b &= (-1)^b 0.5, \\ S_b &= 1.2, \\ w_b &= \frac{1}{5}. \end{aligned}$$

All the scenarios used these same parameters unless explicitly stated otherwise.

4.1.2 No batch effects

This scenario is aimed at measuring the bias of the inferred batch effects. We remove the batch effects from the generating model by using values

$$\begin{aligned} m_b &= 0.0, \\ S_b &= 1.0. \end{aligned}$$

Note the inferred values of S are restricted to the open interval $(1, \infty)$ in our sampler. Because of this we hope that the sampled batch scaling effect has a similar distribution across all batches rather than sampling a distribution centred on 1.0.

4.1.3 Varying batch size

This scenario investigates the behaviour of the model when the batch sizes are very different.

$$w^T = \left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{16}\right). \tag{29}$$

4.1.4 Varying batch effects

This scenario tests how successfully the model infers to differing batch effects in each batch, different magnitudes of batch effects (with some in the tails of the prior distribution) and the direction of the

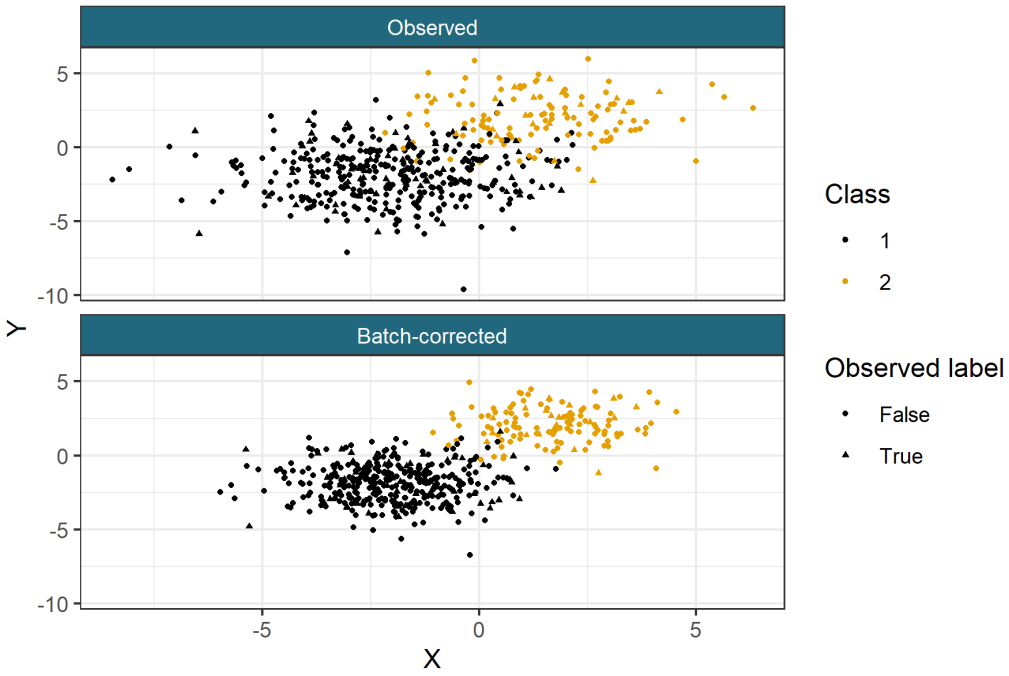

Figure 2: Example of a generated dataset from the Base case scenario.

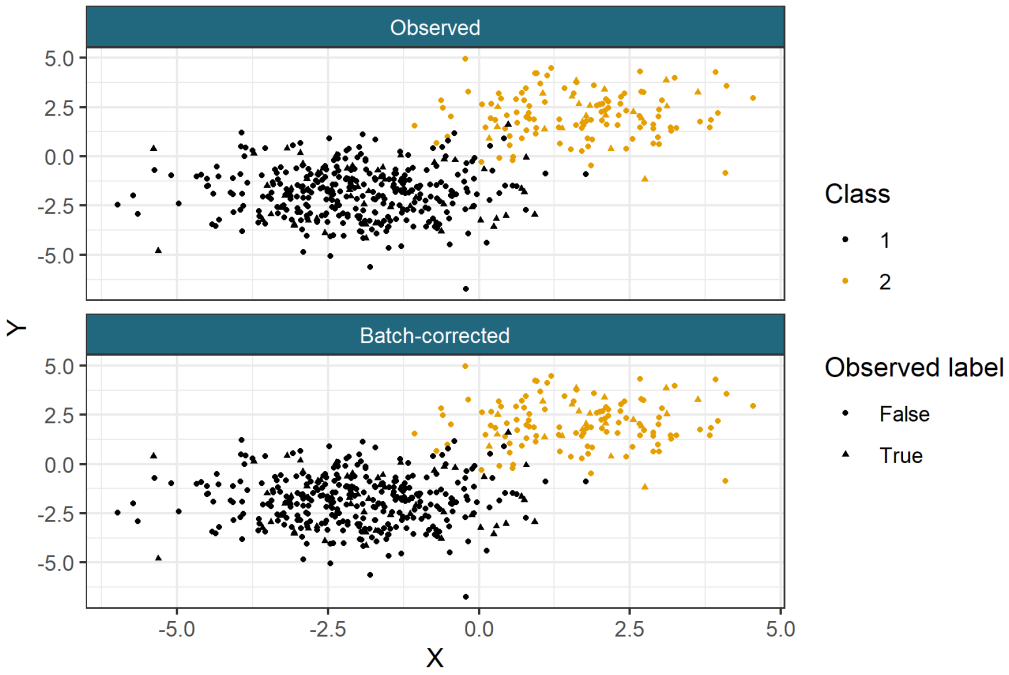

Figure 3: Example of a generated dataset from the No batch effects scenario. Note that the dataset is identical before and after batch-correction.

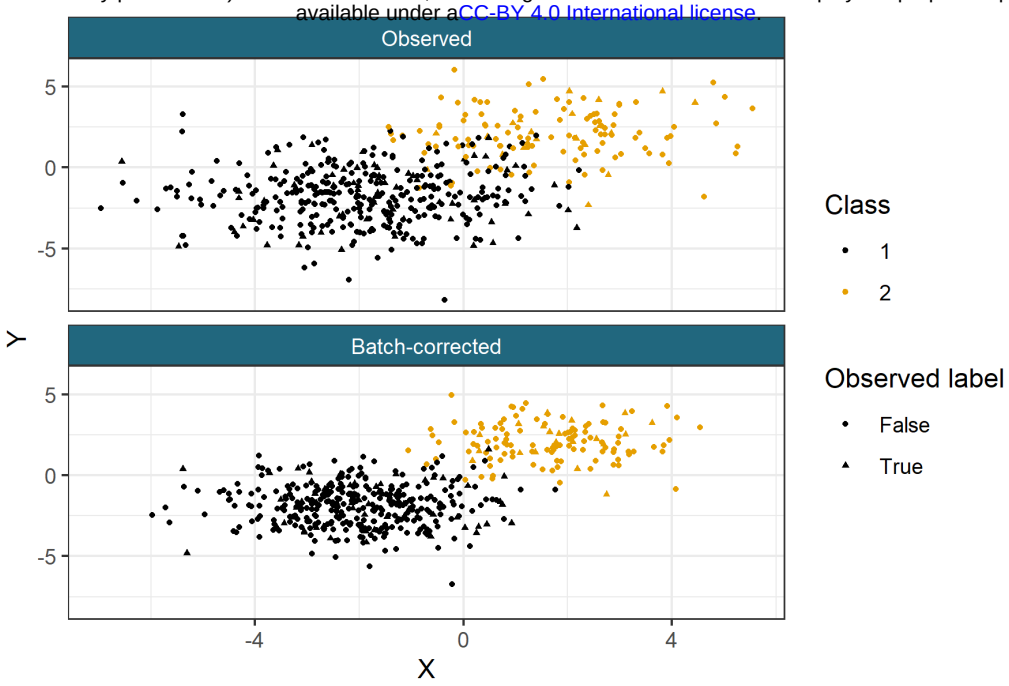

Figure 4: Example of a generated dataset from the Varying batch size scenario.

batch mean shift.

$$m_{b,p} \in [-1, -0.5, 0.0, 0.5, 1.0], \tag{30}$$

$$(S_b)p, p \in [1.1, 1.25, 1.4, 1.6, 2.0]. \tag{31}$$

4.1.5 Varying class representation across batches

In this scenario we investigate how the model responds to different expected representation of classes in each batch. This scenario might apply if the batches are collected across time and the proportion of each class in the population is expected to fluctuate. In this case the expected class proporitons vary across batches are therefore a $K \times B$ matrix,

$$\pi = \begin{pmatrix} 0.7 & 0.8 & 0.5 & 0.2 & 0.1 \\ 0.3 & 0.2 & 0.5 & 0.8 & 0.9 \end{pmatrix}. \tag{32}$$

In each batch one column of this matrix is used to sample the class membership. This introduces a dependency for c_n on b_n, i.e.,

$$c_n | b_n = b, \pi \sim Cat(\pi_b). \tag{33}$$

4.1.6 Multivariate t generated

This scenario generates the data from a multivariate t (**MVT**) distribution. This type of data is believed to be common in biology and we wish to investigate how well the model learns the degrees of freedom parameter and to compare the performance of the mixture of Gaussians model to the mixture of MVTs model.

$$Y_n | c_n = k \sim t_{\eta_k}(\mu_k, \Sigma_k), \tag{34}$$

$$\nu = (3, 5). \tag{35}$$

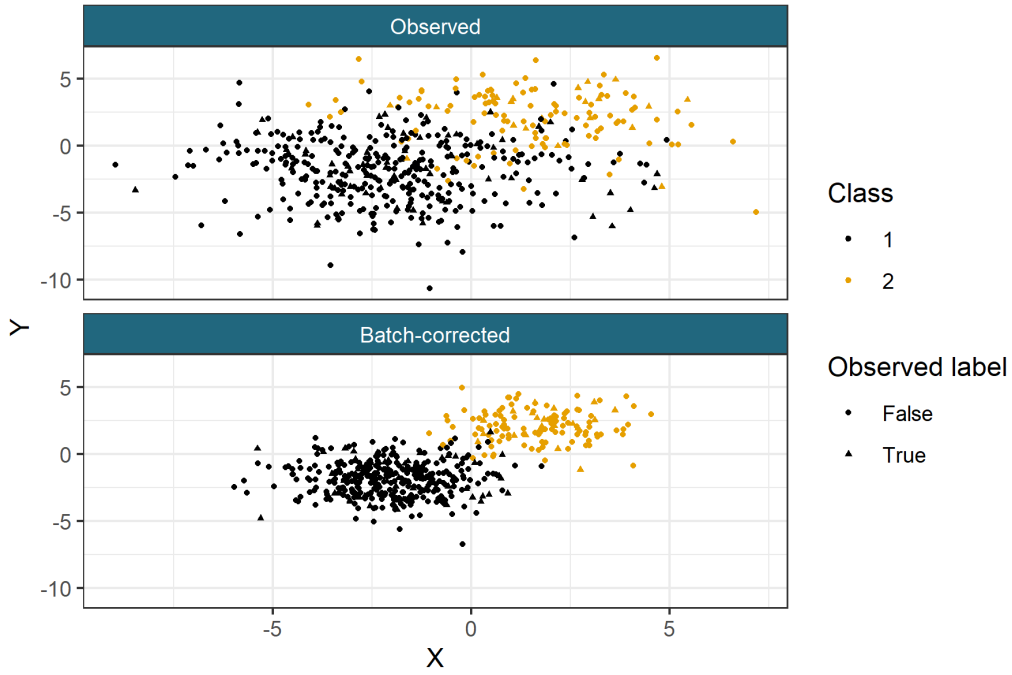

Figure 5: Example of a generated dataset from the Varying batch effects scenario.

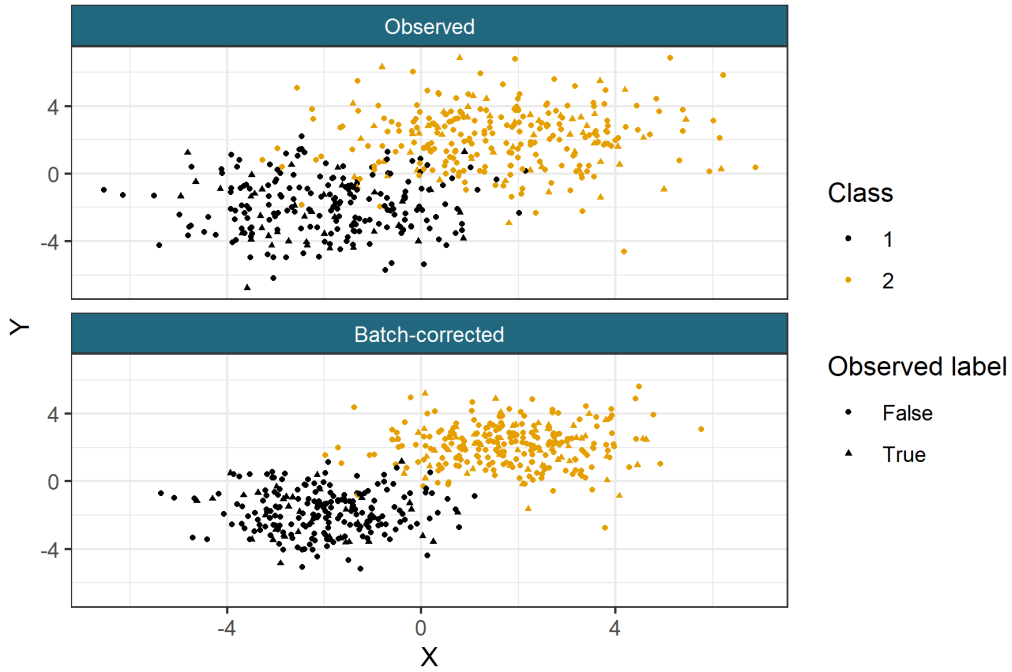

Figure 6: Example of a generated dataset from the Varying class representation across batches scenario.

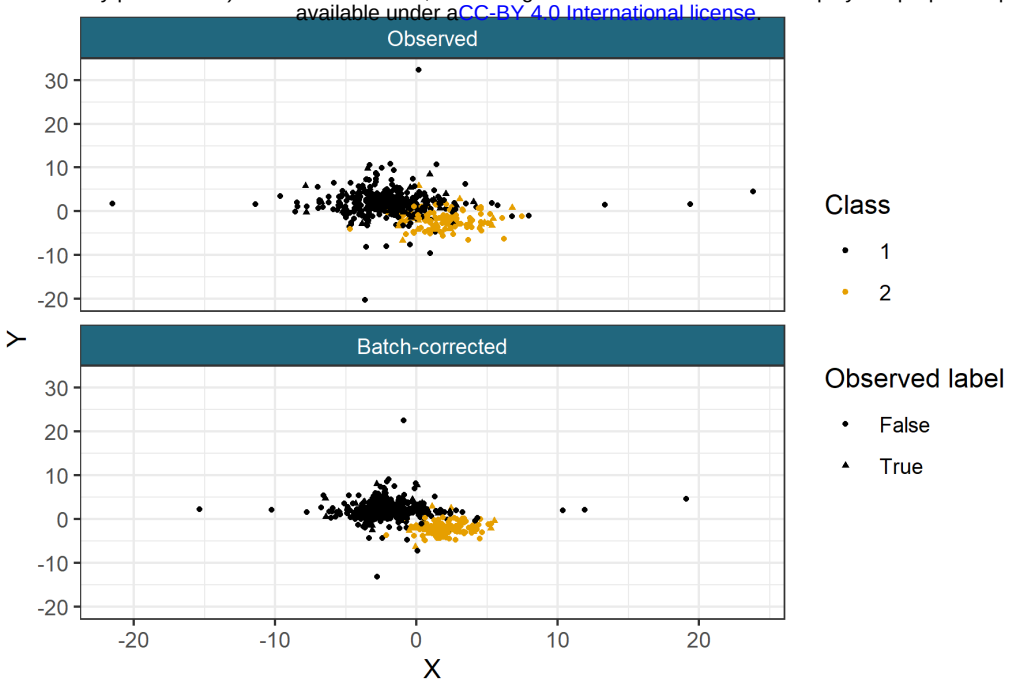

Figure 7: Example of a generated dataset from the MVT scenario.

5 Model convergence

For the simulated data we use the Geweke diagnostic for the complete log-likelihood after burn-in to assess within-chain convergence. We obtain a p-value by transforming the absolute value of the Z-scores with the Gaussian cumulative distribution function. We then discard all chains which have p-values below a threshold of 0.05. We then plot the complete log-likeihood and manually remove any chains that settled in a local mode. An example of this sequential reduction in chains is shown in figure 8 for the pseudo-ELISA simulation.

For the real data we visually inspect the complete log-likelihood trace plots and manually select which chains have converged to the same mode in the posterior distribution (possibly the global mode). As there are less chains performing the entire process manually is feasible for the real datasets. An example of this process is shown in figure 9.

6 Dopico *et al.*

Table 1 shows the seroprevalence estimate for the different methods in the data from Castro Dopico et al. (2021).

7 Pseudo-ELISA data

We use the mean posterior values from a converged chain from the MVT mixture model as the parameters to generate the ELISA-like data. For the class parameters, these are:

$$\Sigma_1 = \begin{pmatrix} 0.042 & 0.035 \\ 0.035 & 0.038 \end{pmatrix}, \qquad \Sigma_2 = \begin{pmatrix} 0.086 & 0.123 \\ 0.123 & 0.195 \end{pmatrix} \tag{36}$$

$$\mu_2 = \begin{pmatrix} -2.43 \\ -2.43 \end{pmatrix}, \qquad \mu_1 = \begin{pmatrix} -0.63 \\ -0.75 \end{pmatrix}, \tag{37}$$

$$\eta_1 = 7.02, \qquad \eta_2 = 13.35. \tag{38}$$

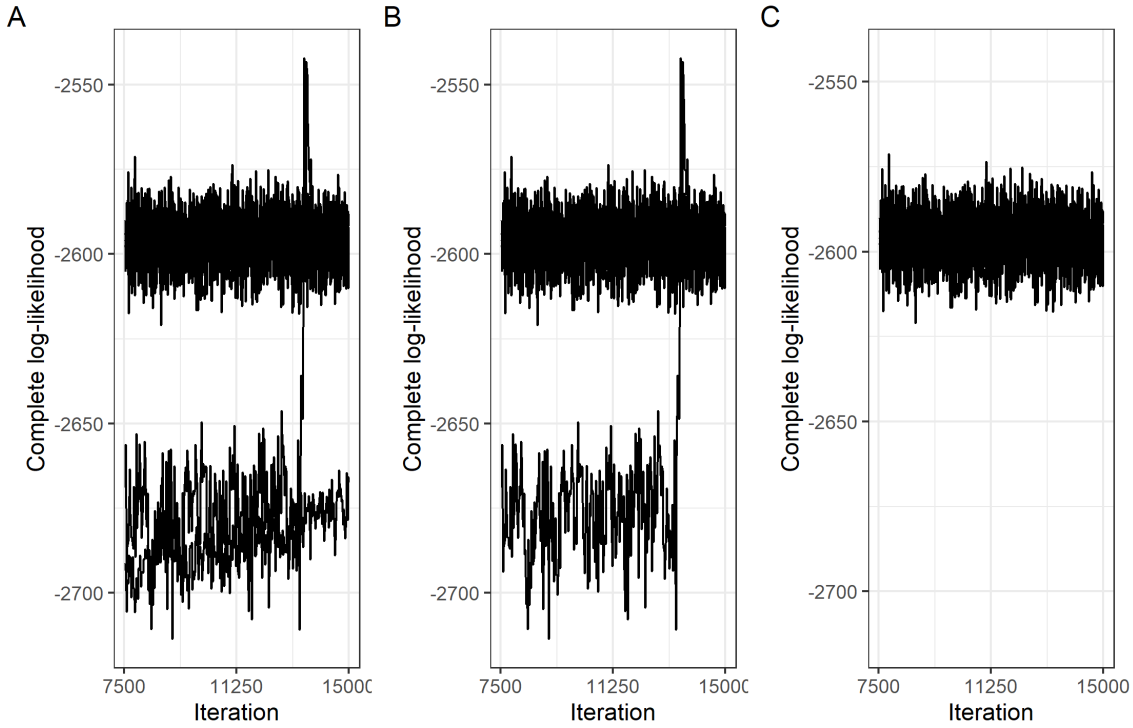

Figure 8: The complete log-likelihood for the MVN model in seventh simulation of the MVT generated data. A) All chains, B) the chains retained after using the Geweke diagnostic to asses within-chain convergence and C) the chains after manual curation.

and for the batch parameters,

$$S_1 = \begin{pmatrix} 1.28 & 0.0 \\ 0.0 & 1.21 \end{pmatrix}, \qquad m_1 = \begin{pmatrix} 0.03 \\ -0.09 \end{pmatrix}, \tag{39}$$

$$S_2 = \begin{pmatrix} 1.86 & 0.0 \\ 0.0 & 1.70 \end{pmatrix}, \qquad m_2 = \begin{pmatrix} 0.09 \\ -0.02 \end{pmatrix}, \tag{40}$$

$$S_3 = \begin{pmatrix} 1.36 & 0.0 \\ 0.0 & 1.28 \end{pmatrix}, \qquad m_3 = \begin{pmatrix} 0.01 \\ -0.13 \end{pmatrix}, \tag{41}$$

$$S_4 = \begin{pmatrix} 1.21 & 0.0 \\ 0.0 & 1.32 \end{pmatrix}, \qquad m_4 = \begin{pmatrix} 0.05 \\ -0.15 \end{pmatrix}, \tag{42}$$

$$S_5 = \begin{pmatrix} 1.58 & 0.0 \\ 0.0 & 1.40 \end{pmatrix}, \qquad m_5 = \begin{pmatrix} 0.11 \\ -0.09 \end{pmatrix}, \tag{43}$$

$$S_6 = \begin{pmatrix} 1.20 & 0.0 \\ 0.0 & 1.23 \end{pmatrix}, \qquad m_6 = \begin{pmatrix} 0.55 \\ 0.36 \end{pmatrix}, \tag{44}$$

$$S_7 = \begin{pmatrix} 1.25 & 0.0 \\ 0.0 & 1.26 \end{pmatrix}, \qquad m_7 = \begin{pmatrix} 0.10 \\ -0.10 \end{pmatrix}. \tag{45}$$

We use the predicted proportion of each batch as our batch-specific class weights,

$$\pi = \begin{pmatrix} 0.95 & 0.87 & 0.91 & 0.88 & 0.96 & 0.10 & 0.95 \\ 0.05 & 0.13 & 0.90 & 0.12 & 0.04 & 0.90 & 0.05 \end{pmatrix}. \tag{46}$$

Each column corresponds to a batch and each row is the class weight. We denote the class weights within a batch (i.e., one of these columns) by π_b. The probability of being drawn from a given batch is simply the observed proportion of items in each batch.

$$w = \begin{pmatrix} 0.18 & 0.18 & 0.06 & 0.28 & 0.15 & 0.02 & 0.13 \end{pmatrix}. \tag{47}$$

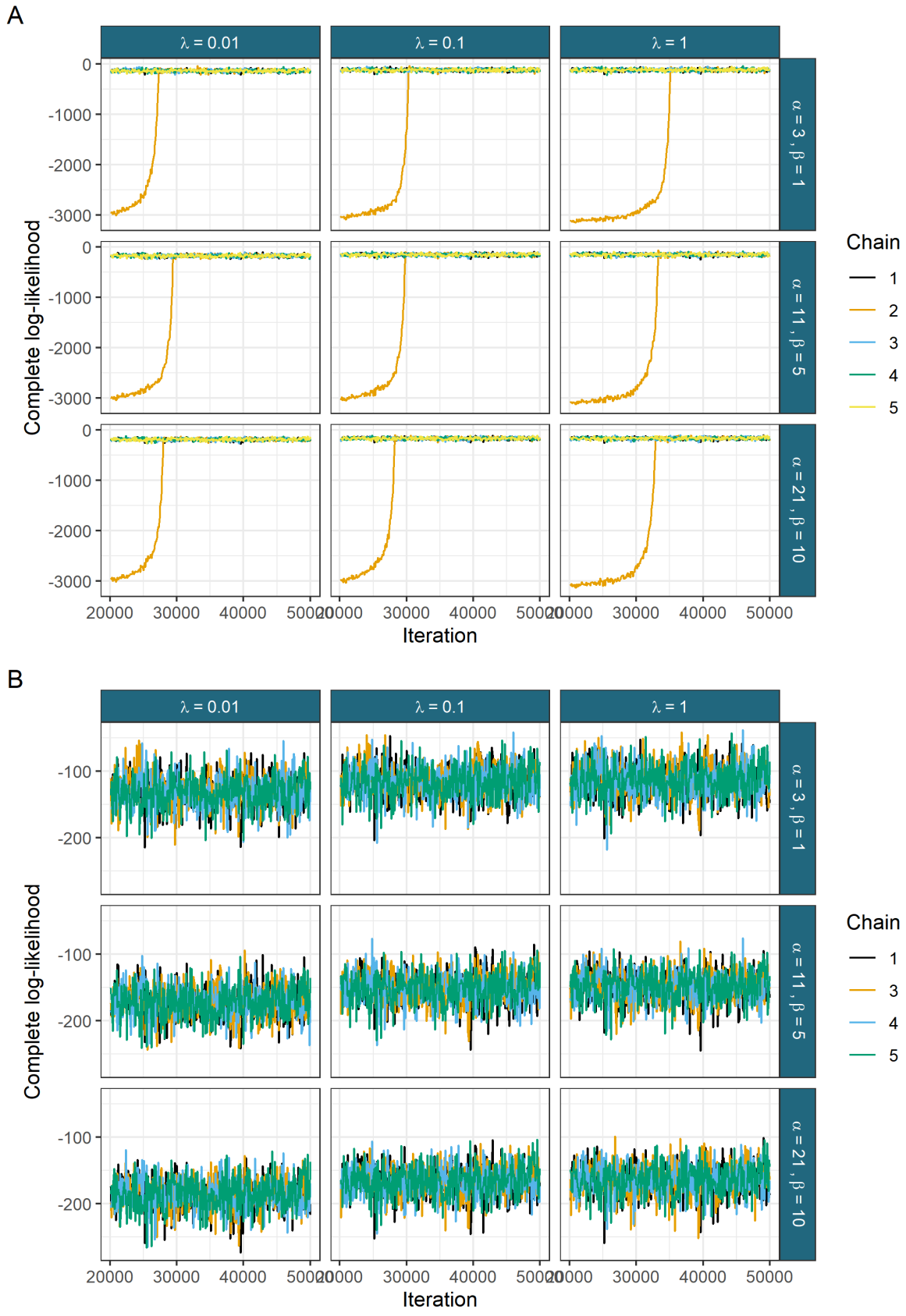

Figure 9: The complete log-likelihood for the MVT model in Stockholm ELISA data for A) all chains and B) the converged chains.

Date	SVM-LDA*	Bayesian learner*	MVT	RF	SVM	LR	LR - BC
2020/04/05	NA	**2.35**	1.60	**1.00**	**1.00**	**1.00**	2.00
2020/04/26	**4.26**	4.73	4.92	4.50	**5.00**	**5.00**	**5.00**
2020/05/03	**4.33**	5.34	5.74	4.50	4.50	5.00	**5.50**
2020/05/10	7.87	**5.86**	8.87	8.50	8.50	8.50	**10.50**
2020/05/17	7.36	6.28	8.05	7.50	**4.50**	8.00	**8.50**
2020/05/24	7.49	6.64	8.34	8.00	**6.50**	7.50	**9.50**
2020/05/31	3.54	**6.97**	4.15	4.00	**3.50**	4.00	5.00
2020/06/07	8.41	**7.31**	9.83	9.50	8.00	10.00	**10.50**
2020/06/14	**6.90**	7.75	7.55	7.00	7.00	7.00	**8.50**
2020/06/21	**6.44**	**8.30**	7.80	7.00	6.50	7.50	8.00
2020/07/26	13.20	**11.34**	**16.72**	15.00	11.50	16.50	16.00
2020/08/02	9.16	**11.79**	10.48	9.00	**8.50**	10.00	10.00
2020/08/09	11.30	12.15	13.43	12.00	**8.00**	13.00	**13.50**
2020/08/16	10.84	**12.43**	12.15	12.00	**10.50**	11.50	11.00
2020/08/23	12.97	12.65	**15.55**	15.50	**11.00**	15.00	15.00
2020/11/08	11.79	**14.28**	13.72	12.00	**11.50**	12.50	14.00
2020/11/15	**14.20**	14.47	**18.85**	15.50	14.50	17.00	18.50
2020/11/22	**13.37**	14.72	**16.44**	15.50	15.50	15.50	16.00
2020/11/29	**13.20**	14.98	**17.36**	15.00	15.00	16.00	16.50
2020/12/06	11.52	**15.29**	14.47	12.50	**11.00**	13.00	13.50
2020/12/13	15.73	**15.64**	**19.65**	18.00	16.00	18.00	19.00

Table 1: Seroprevalence estimates across time for each method in the data from Castro Dopico et al. (2021). The highest estimates at each data are coloured orange, the lowest are coloured blue. * from Castro Dopico et al. (2021).

We then generate a batch and class label for each item and then observed measurements conditioning on these labels, specifically for a given item index n:

$$b_n \sim Cat(w), \tag{48}$$
$$c_n|b_n = b \sim Cat(\pi_b), \tag{49}$$
$$X_n|c_n = k, b_n = b \sim t_{\eta_k}(\mu_k + m_b, \Sigma_k \oplus S_b). \tag{50}$$

For the seronegative class (we use the label of $c_n = 1$ for this class), the ϕ_n parameter indicating if the n^{th} item has an observed label is a Bernoulli random variable. For the seropositive class we introduce a bias to match the reality that it is more extreme observations that tend to have an observed label. To do this we find the most extreme value in each measurement, denoted X_{max}, (note that X_{max} is unlikely to be an observed value) and calculate the Euclidean distance between this and our observed values. We then sample ϕ according to:

$$p(\phi_n = 1|c_n = 1) = p(1-p), \tag{51}$$
$$p(\phi_n = 1|c_n = 2) = p(1-p)\exp\{-d(X_n, X_{max})\}, \tag{52}$$

where $p = \frac{1}{3}$. This values is chosen as the proportion of observed labels to the predicted labels is 0.332 for the seronegative class and 0.241 for the seropositive class. Our sampling process finds provides less observed seropositive labels than we have in the real data (the ratio of observed labels to true labels for the seropositive class had a mean of 0.16 across 500 simulated datasets), but we think representing the bias in the positive controls is more important than acquiring the exact proportion of training data.

References

X. Castro Dopico, S. Muschiol, M. Christian, L. Hanke, D. J. Sheward, N. F. Grinberg, J. Rorbach, G. Bogdanovic, G. M. Mcinerney, T. Allander, C. Wallace, B. Murrell, J. Albert, and G. B. Karlsson Hedestam. Seropositivity in blood donors and pregnant women during the first year of SARS-CoV-2 transmission in Stockholm, Sweden. *Journal of Internal Medicine*, May 2021. ISSN 1365-2796. doi: 10.1111/joim.13304.

Chris Fraley and Adrian E Raftery. Bayesian Regularization for Normal Mixture Estimation and Model-Based Clustering. *Journal of Classification*, 24(2):27, September 2007.

Stuart Geman and Donald Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6 (6):721–741, November 1984. ISSN 1939-3539. doi: 10.1109/TPAMI.1984.4767596. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, April 1970. ISSN 0006-3444. doi: 10.1093/biomet/57.1.97. URL `https://doi.org/10.1093/biomet/57.1.97`.

Miguel A. Juárez and Mark F. J. Steel. Model-Based Clustering of Non-Gaussian Panel Data Based on Skew- t Distributions. *Journal of Business & Economic Statistics*, 28(1): 52–66, January 2010. ISSN 0735-0015, 1537-2707. doi: 10.1198/jbes.2009.07145. URL `http://www.tandfonline.com/doi/abs/10.1198/jbes.2009.07145`.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, June 1953. ISSN 0021-9606. doi: 10.1063/1.1699114. URL `https://aip.scitation.org/doi/abs/10.1063/1.1699114`. Publisher: American Institute of Physics.

Gareth O. Roberts and Jeffrey S. Rosenthal. Optimal scaling for various Metropolis-Hastings algorithms. *Statistical Science*, 16(4):351–367, November 2001. ISSN 0883-4237, 2168-8745. doi: 10.1214/ss/1015346320. URL `https://projecteuclid.org/journals/statistical-science/volume-16/issue-4/Optimal-scaling-for-vario` Publisher: Institute of Mathematical Statistics.