Abstract

We survey the EM algorithm and its Monte Carlo-based extensions.

- Replace "conditional distribution of the missing data given the observed data" with "missing data distribution"
- Caffo et al. (2005) use M_k for the Monte Carlo size at iteration k. This is a useful definition for discussing other papers too.
- I have changed my iteration labels. $\hat{\theta}_{k-1}$ is now the maximizer of the current MCEM objective function, and $\hat{\theta}_{k-1}$ was used to construct that objective function. Watch for this when editing.
- Parameter space has not yet been defined.
- Is the "a" in EM algorithm capitalized or not?

1 The EM Algorithm

We define three distributions which will be central to our study of missing data problems. Let Y be the observed data and X be the missing data. Note that X need not correspond to any actual real-world process, but may instead be a conceptual device which facilitates analysis of the data which were actually observed. We refer to the distribution of Y as the "observed data distribution", and write f for its density (or mass function). We refer to the joint distribution of Y and X as the "complete data distribution", and write f_c for its density. We refer to the conditional distribution of X given Y as the "missing data distribution", and write f_m for its density. Note that the missing data distribution is not the marginal distribution of the missing data, but rather its conditional distribution given the observed data.

The EM algorithm is a method for analyzing incomplete data which was formalized by Dempster et al. (1977). See McLachlan and Krishnan (2008) for an excellent book-length overview of the EM algorithm. We begin by discussing a probabilistic framework within which the EM algorithm is often applied. We then present the EM algorithm in detail. Finally, we discuss some limitations of this method. Throughout, we illustrate our presentation with a toy problem based on linear regression with a single, unobserved, covariate.

The EM algorithm consists of iterating two steps. First is the expectation, or "E", step, in which an objective function is constructed from the complete data likelihood. Second is the maximization, or "M", step, in which the previously computed objective function is maximized. These two steps are then alternated until some convergence criterion is met.

Whatever value of θ the algorithm converges to is used as our parameter estimate. We now go into more detail on each of the two steps.

The E-step of the EM algorithm is where we construct the objective function which will be used to update our parameter estimate. This objective function is the conditional expectation of the complete data likelihood, given the observed data. If our complete data can be partitioned into an observed component, Y, and a missing component, X, then our objective function at iteration k is given by

$$Q(\theta|\theta_{k-1}) = \mathbb{E}_{\theta_{k-1}}[\ell_c(\theta; y, X)|Y = y] \tag{1}$$

where ℓ_c is the log-likelihood of the complete data model. Note that the conditional expectation uses our parameter estimate from the previous iteration.

The M-step of the EM algorithm consists of maximizing the objective function constructed in the previous E-step. That is, we define $\theta_k = \underset{\theta}{\operatorname{argmax}}\, Q(\theta|\theta_{k-1})$. Typically, this optimization must be performed numerically via, e.g., gradient ascent or Newton's method. See Nocedal and Wright (2006) for details and other optimization algorithms.

We can combine the E- and M-steps of the EM algorithm into a single "update function". We write $M(\theta_{k-1}) = \underset{\theta}{\operatorname{argmax}}\, Q(\theta|\theta_{k-1})$. The EM algorithm can thus be viewed as the iterative application of this update function, M.

1.1 Properties

Section intro...

1.1.1 Ascent Property and Generalized EM

An important feature of the EM algorithm is its so-called "ascent property". This property says that an iteration of the EM algorithm (have I explicitly defined "EM iteration"? Do I need to?) never decreases the observed data likelihood. This is somewhat surprising, since EM updates are computed without ever evaluating the observed data likelihood.

Proposition 1.1 (Ascent Property of EM)**.** *Let $\theta \in \Theta$, and $\theta' = M(\theta)$ be the EM update from θ. Then $\ell(\theta') \geq \ell(\theta)$.*

Proof. We begin by noting that the following decomposition holds for any value of x:

$$\ell(\theta; y) = \ell_c(\theta; y, x) - \ell_m(\theta; y, x) \tag{2}$$

Subtracting the values of both sides at θ from their values at θ' and taking conditional expectations, we get

$$\begin{aligned} \ell(\theta'; y) - \ell(\theta; y) &= Q(\theta'|\theta) - Q(\theta|\theta) + \mathbb{E}_\theta[\ell_m(\theta; y, x) - \ell_m(\theta'; y, x)] & (3) \\ &= Q(\theta'|\theta) - Q(\theta|\theta) + \mathrm{KL}(\theta||\theta') & (4) \end{aligned}$$

where the last term in line 4 is the Kullback-Leibler (KL) divergence from the missing data distribution with $\theta = \theta$ to the same distribution with $\theta = \theta'$. Note that KL divergences are always non-negative, so we get

$$\ell(\theta'; y) - \ell(\theta; y) \geq Q(\theta'|\theta) - Q(\theta|\theta) \tag{5}$$

Finally, since θ' maximizes $Q(\cdot|\theta)$, we have $\ell(\theta'; y) - \ell(\theta; y) \geq 0$. $\square$

In our proof of the ascent property, we only required that $Q(\theta'|\theta) \geq Q(\theta|\theta)$, not that θ' maximize $Q(\cdot|\theta)$. This observation leads to the definition of the "Generalized EM Algorithm", which replaces the M-step with setting θ_k to any point in Θ such that $Q(\theta_k|\theta_{k-1}) \geq Q(\theta_{k-1}|\theta_{k-1})$.

1.1.2 Recovering Observed Data Likelihood Quantities

Under regularity conditions, it is possible to compute both the score vector and the observed information matrix of the observed data likelihood using complete data quantities. These regularity conditions consist of being able to interchange the order of differentiation and integration for various functions (awk? I wasn't feeling well when I wrote this.). Does it make sense to define $\mathcal{I}_c$ and $\mathcal{I}_m$ in the following proposition?

Proposition 1.2. *The following identities hold (regularity conditions!):*

(i) $S(\theta; y) = \mathbb{E}_\theta[S_c(\theta; y, X)|Y = y]$

(ii) $I(\theta) = \mathcal{I}_c(\theta) - \mathcal{I}_m(\theta)$
where $\mathcal{I}_c(\theta) := -\mathbb{E}_\theta\left[\nabla^2 \ell_c(\theta; y, X)|Y = y\right]$ *and* $\mathcal{I}_m(\theta) := -\mathbb{E}_\theta\left[\nabla^2 \ell_m(\theta; y, X)|Y = y\right]$

Proof. We start with expression (i). Let Ω be the complete data sample space. Let $\mathcal{Y}$ and $\mathcal{X}$ be the observed and missing data sample spaces respectively. For every $y \in \mathcal{Y}$, let

$\mathcal{X}(y) = \{x \in \mathcal{X} : (y,x) \in \Omega\}$. Note that $f(y;\theta) = \int_{\mathcal{X}(y)} f_c(y,x;\theta)dx$.

$$\begin{aligned}
\mathbb{E}_\theta[S_c(\theta;y,X)|Y=y] &= \int_{\mathcal{X}(y)} \nabla \ell_c(\theta;y,x) f_m(y,x;\theta)dx \\
&= \int_{\mathcal{X}(y)} \frac{f_m(y,x;\theta)}{f_c(y,x;\theta)} \nabla f_c(\theta;y,x)dx \\
&= \int_{\mathcal{X}(y)} \frac{1}{f(y;\theta)} \nabla f_c(\theta;y,x)dx \\
&= \frac{1}{f(y;\theta)} \int_{\mathcal{X}(y)} \nabla f_c(\theta;y,x)dx \\
&= \frac{1}{f(y;\theta)} \nabla \int_{\mathcal{X}(y)} f_c(\theta;y,x)dx \\
&= \frac{1}{f(y;\theta)} \nabla f(y;\theta) \\
&= S(\theta;y)
\end{aligned}$$

Proceeding now to (ii), we decompose the observed data log-likelihood as

$$\ell(\theta;y) = \ell_c(\theta;y,x) - \ell_m(\theta;y,x)$$

Differentiating twice and taking conditional expectations of both sides yields the required result. □

An alternative to Proposition 1.2 part (ii) which involves only conditional expectations of complete data quantities is given in the following proposition.

Proposition 1.3. *Let $\hat\theta$ be a stationary point of the observed data log-likelihood. Assuming regularity conditions, we can write the observed information of the observed data distribution at $\hat\theta$ as*

$$I(\theta) = \mathcal{I}_c(\theta) - \mathbb{E}_\theta[S_c(\theta)S_c(\theta)^T|Y=y] + S(\theta)S(\theta)^T \tag{6}$$

In particular, if $\hat\theta$ is a stationary point of the observed data log-likelihood, then

$$I(\hat\theta) = \mathcal{I}_c(\hat\theta) - \mathbb{E}_{\hat\theta}[S_c(\hat\theta)S_c(\hat\theta)^T|Y=y] \tag{7}$$

Proof. We follow the derivation of Louis (1982). For brevity, we write $f(\theta)$ and $f_c(\theta)$ for

$f(y;\theta)$ and $f(y,x;\theta)$ respectively. Consider the following two Hessians:

$$\nabla^2 \ell(\theta) = \nabla \left[\int_{\mathcal{X}(y)} \frac{\nabla f_c(\theta) dx}{f(\theta)} \right] \tag{8}$$

$$= \int_{\mathcal{X}(y)} \frac{\nabla^2 f_c(\theta)}{f(\theta)} dx - \frac{1}{f(\theta)^2} \left(\int_{\mathcal{X}(y)} \nabla f_c(\theta) dx \right) \left(\int_{\mathcal{X}(y)} \nabla f_c(\theta) dx \right)^T \tag{9}$$

$$= \mathbb{E}_\theta \left[\frac{\nabla^2 f_c(\theta)}{f_c(\theta)} \middle| Y = y \right] - \mathbb{E}_\theta \left[\frac{\nabla f_c(\theta)}{f_c(\theta)} \middle| Y = y \right] \mathbb{E}_\theta \left[\frac{\nabla f_c(\theta)}{f_c(\theta)} \middle| Y = y \right]^T \tag{10}$$

$$= \mathbb{E}_\theta \left[\frac{\nabla^2 f_c(\theta)}{f_c(\theta)} \middle| Y = y \right] - S(\theta; y) S(\theta; y)^T \tag{11}$$

$$\nabla^2 \ell_c(\theta) = \nabla \left(\frac{\nabla f_c(\theta)}{f_c(\theta)} \right) \tag{12}$$

$$= \frac{\nabla^2 f_c(\theta)}{f_c(\theta)} - S_c(\theta) S_c(\theta)^T \tag{13}$$

Combining lines 11 and 13, we get

$$\nabla^2 \ell(\theta) = \mathbb{E}_\theta[\nabla^2 \ell_c(\theta) | Y = y] + \mathbb{E}_\theta[S_c(\theta) S_c(\theta)^T | Y = y] - S(\theta; y) S(\theta; y)^T \tag{14}$$

Finally, evaluating line 14 at $\theta = \hat{\theta}$ makes the rightmost term vanish, thereby yielding the required expression. $\square$

Proposition 1.3 is known as Louis' standard error formula. Other decompositions for the observed information matrix of the observed data likelihood do exist; see, e.g., Oakes (1999); McLachlan and Krishnan (2008). However, the one due to Louis will be most useful to us later.

1.2 Example: Gene Frequency Estimation

Consider the problem of estimating allele frequencies based on observed phenotypes. Often, a single phenotype can be encoded by multiple genotypes with different configurations of dominant and recessive alleles. This is sometimes referred to as the problem of gene frequency estimation. Our analysis closely follows Example 2.4 from McLachlan and Krishnan (2008)[1]

We investigate a simplified model for blood type which consists of only the ABO blood group. See Chapter 5 of Dean (2005), for a detailed introduction to blood types. There are three alleles for this gene: A, B and O. Allele O is recessive, which alleles A and B

[1]Although these authors do give data in their example, there is little context around this data, and I have been unable to locate more information from their references. I have opted instead to use my own dataset.

Table 1: Observed frequency and theoretical probability of each blood type (Fujita et al., 1978)

Blood Type	O	A	B	AB
Random Variable	Y_1	Y_2	Y_3	Y_4
Observed Frequency	10	16	7	1
Probability	r^2	$p^2 + 2pr$	$q^2 + 2qr$	$2pq$

exhibit co-dominance. That is, genotypes AO and AA encode blood type A, genotypes BO and BB encode blood type B, genotype OO encodes blood type O, and genotype AB encodes blood type AB. Suppose that we seek to estimate the proportion of each allele within a population, based on a sample of individuals' phenotypes. Fujita et al. (1978) report blood types of 4,464,349 people in Japan collected between 1964 and 1975. This sample is so large that any standard errors are practically zero. To retain a reasonable level of uncertainty, we focus on a single administrative division, Oto, in Nara Prefecture. See Figure 1 for details.

Let Y_1, Y_2, Y_3 and Y_4 be the number of people with blood type O, A, B and AB respectively, and $Y = (Y_1, Y_2, Y_3, Y_4)$. Let r, p and q be the proportions of alleles O, A and B respectively within the population of interest. Since $r + p + q = 1$, we let $\theta = (p, q)$ be our target of inference. Pretending that the population size is fixed and that iid sampling was employed, Y follows a multinomial distribution with $n = 4{,}464{,}349$. Assuming homogeneous genetic mixing, the probability vector for Y is $\pi = (r^2, p^2+2pr, q^2+2qr, 2pq)$.

Maximizing the likelihood in this model involves solving the score equation, a system of two 3rd-degree polynomials in p and q. This can be done numerically, and gives estimates $p = 0.299$ and $q = 0.128$. These values match the ones given by Fujita et al. (1978). The information matrix and asymptotic covariance (inverse information matrix) are given by

$$I(\hat{\theta}) = \begin{bmatrix} 276 & 84.8 \\ 84.8 & 584 \end{bmatrix} \tag{15}$$

$$\hat{\Sigma}_{\text{MLE}} = \begin{bmatrix} 3.79 \cdot 10^{-3} & -5.49 \cdot 10^{-4} \\ -5.49 \cdot 10^{-4} & 1.79 \cdot 10^{-3} \end{bmatrix} \tag{16}$$

1.2.1 Complete Data

The problem of gene frequency estimation would be much simpler if we could observe individuals' genotypes. We consider augmenting the observed data Y by further classifying individuals by genotype. Let $X = (X_1, \ldots, X_6)$ be the genotypes of the individuals represented in Table 1. See Table 2.

Note that we can write Y in terms of X. Specifically, $Y_1 = X_1$, $Y_2 = X_2 + X_3$, $Y_3 = X_4 + X_5$ and $Y_4 = X_6$. This corresponds to summing components of X which

Table 2: Terminology and probabilities for our augmented version of the dataset in Fujita et al. (1978). We also give the blood type coded for be each genotype.

Genotype	OO	AO	AA	BO	BB	AB
Random Variable	X_1	X_2	X_3	X_4	X_5	X_6
Probability	r^2	$2pr$	p^2	$2qr$	q^2	$2pq$
Blood Type	O	A	A	B	B	AB

correspond to the same blood type. The distribution of X is multinomial, with the same sample size as Y, and probability vector given in Table 2.

See Appendix A.2 for the complete data likelihood function and its derivatives.

1.2.2 EM Algorithm

The gene frequency estimation problem fits nicely into the EM algorithm framework. In this section, we present key quantities and results of our analysis. See Appendix A, especially part A.3, for more details.

The EM objective function at iteration $k+1$ is

$$Q(\theta|\theta_k) \equiv \mathbb{E}_{\theta_k}(n_O|y)\log r + \mathbb{E}_{\theta_k}(n_A|y)\log p + \mathbb{E}_{\theta_k}(n_B|y)\log q \tag{17}$$

$$=: \nu_O^{(k)}\log r + \nu_A^{(k)}\log p + \nu_B^{(k)}\log q \tag{18}$$

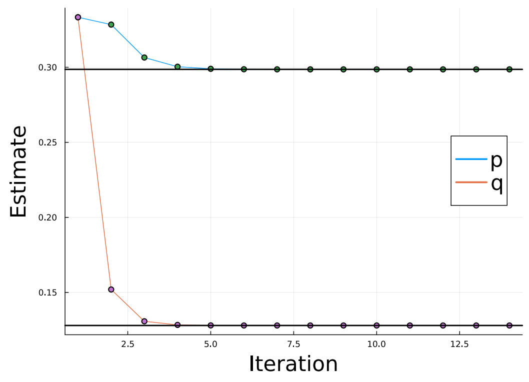
where $nu_O^{(k)}$, $\nu_A^{(k)}$ and $\nu_B^{(k)}$ are the expected number of O, A and B alleles respectively given $Y = y$ and $\theta = \theta_k$. Maximizing this objective function in p and q gives the following EM update function:

$$M(\theta_k) = \begin{pmatrix} p_{k+1} \\ q_{k+1} \end{pmatrix} \tag{19}$$

$$= \begin{pmatrix} \nu_A^{(k)}/2n \\ \nu_B^{(k)}/2n \end{pmatrix} \tag{20}$$

Starting with $\theta_0 = (1/3, 1/3)$ corresponding to equal proportions of the three alleles, Figure 1 gives trajectories of the EM estimates for p and q using the data in Figure 1. These estimates converge quite quickly to the maximizer of the observed data likelihood. Beyond computing the observed data MLE, we also need the standard error of this estimator. To this end, we compute the observed data information matrix using Louis' Method (see Proposition 1.3). The asymptotic covariance matrix of our MLE is then the inverse of this information matrix. Omitting details (see Appendices A.2 and A.3), both the observed data information matrix and asymptotic covariance match those obtained from the observed data likelihood.

Figure 1: Trajectory of EM estimates for p and q for the blood type example. Horizontal lines give the values of the MLE.



2 The Monte Carlo EM Algorithm

The Monte Carlo EM, or MCEM, algorithm was first proposed by Wei and Tanner (1990). This method proceeds by replacing the conditional expectation in the E-step of the EM algorithm with a Monte Carlo average. More precisely, at each iteration we generate observations from the conditional distribution of the missing data given the observed data, and average the complete data likelihood over this Monte Carlo sample. Formally, at a given iteration of the MCEM algorithm, let $X_1, \ldots, X_M$ be a Monte Carlo sample from the law of $X|Y = y$ with θ set to the value from the previous iteration, say θ_0. Write

$$\hat{Q}(\theta|\theta_0) = \sum_{i=1}^{M} w_i \ell_c(\theta; y, X_i) \tag{21}$$

$$:= \hat{\mathbb{E}} \ell_c(\theta; y, X) \tag{22}$$

where the w_i are sampling weights. Confirm that this operator notation isn't contradicted elsewhere. Under iid sampling we simply get $w_i = M^{-1}$ for every i, but more intricate sampling schemes may have more complicated weights. The estimate of θ is then the maximizer of the MCEM objective function: $\hat{\theta} = \operatorname{argmax}_\theta \hat{Q}(\theta|\theta_0)$. Write $\hat{\theta}_{k-1}$ for the kth MCEM estimate.

Provided that a valid sampling scheme is available for the missing data distribution, we can use Proposition 1.3 to estimate the observed data information matrix.

Proposition 2.1. *Under the conditions of Proposition 1.3, as well as any required for the sampler, we get*

$$-\hat{\mathbb{E}}_{\hat{\theta}} \nabla^2 \ell_c(\hat{\theta}) - \hat{\mathbb{E}}_{\hat{\theta}} S_c(\hat{\theta}) S_c(\hat{\theta})^T \to I(\theta) \tag{23}$$

Under stronger conditions, we also get asymptotic normality with variance obtained from importance sampling analysis.

The MCEM algorithm has the advantage of circumventing the challenge of computing potentially intractable conditional expectations for the EM algorithm. However, this analytical simplification does come at the cost of introducing some new computational problems. In this section, we outline the main problems faced by the MCEM algorithm and present various solutions which have been proposed in the literature. We focus primarily on practical aspects of the MCEM algorithm; see Neath (2013) for a survey of theoretical considerations.

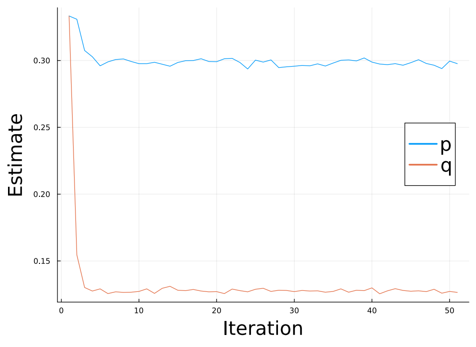
Two problems which have received considerable attention in the literature are how to choose the Monte Carlo sample size at each iteration, and when to terminate the MCEM algorithm. These were identified early by Wei and Tanner (1990), but did not receive systematic treatment until later. We here give a brief overview of different authors' approaches to solving these two problems. The rest of this section goes into more detail on each of the methods. Wei and Tanner (1990) suggest examining a plot of the parameter estimates across iterations, and either terminating or increasing the Monte Carlo size when the plot appears to stabilize. Chan and Ledolter (1995) use a pilot study to choose the Monte Carlo sample size, and terminate when a confidence interval for the improvement of the observed data log-likelihood between successive iterations contains zero. Booth and Hobert (1999) frame each MCEM iteration as an M-estimation problem targeting the deterministic EM update. They increase the Monte Carlo size if an asymptotic confidence interval for the EM update contains the previous iteration's parameter estimate, and terminate when multiple consecutive iterations' estimates have sufficiently small relative error. Caffo et al. (2005) build confidence bounds for the increment in the EM objective function at each iteration of the MCEM algorithm. They increase the Monte Carlo size until the lower bound is positive and terminate when the upper bound is sufficiently small.

In the rest of this section, we give more detail on each of the implementations introduced above. We also illustrate each method on the blood type dataset described in Section 1.2. The relevant conditional distribution and likelihood calculations are described in Appendix A.3.

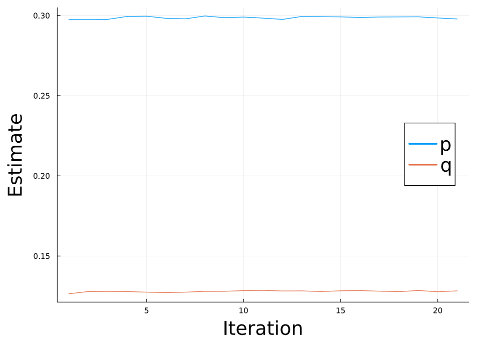
2.1 Early Work (Wei and Tanner, 1990)

In their seminal work, Wei and Tanner (1990) propose the MCEM algorithm and present a simple implementation. They illustrate that the complete data gradient and Hessian are easily obtained at each iteration from the Monte Carlo sample and, following Louis (1982), give an estimator for the observed data information matrix. Regarding convergence, Wei and Tanner recommend plotting the parameter estimates across iterations and stopping when the estimates appear to stabilize around some constant. When this stabilization is detected, one can either declare convergence and stop, or increase the Monte Carlo size and continue iterating until the estimate trajectory again stabilizes.

Figure 2: Trajectory of MCEM estimates of p and q for the blood type example. The horizontal lines correspond to maximum likelihood estimates.



(a) $M = 100$



(b) $M = 1000$

In order to apply the MCEM algorithm to estimate allele frequencies in the blood type problem, we must specify the number of iterations, K, and the Monte Carlo size for each iteration, M. Starting conservatively, we use $K = 50$ and $M = 100$. Figure 2a gives trajectories of the MCEM estimates of p and q. These estimates appear to converge quickly to a stationary mean, but there is still some uncertainty around this mean. As such, we run MCEM for another 20 iterations with $M = 1000$, staring with the final value from our first run. See Figure 2b. The trajectories from our second run are much more stable around their means. We use the final values from these trajectories as our estimates: $\hat{p} = 0.298$ and $\hat{q} = 0.128$. These values closely match the maximizer of the observed data likelihood.

2.2 Running a Pilot Study (Chan and Ledolter, 1995)

Building on the ideas of Wei and Tanner, Chan and Ledolter (1995) develop a method for both choosing the Monte Carlo size and deciding when to terminate the MCEM algorithm. This method is based on an identity which allows us to estimate observed data likelihood ratios by Monte Carlo averages of complete data likelihood ratios. More precisely, we write

$$\frac{\mathcal{L}(\theta_1; y)}{\mathcal{L}(\theta_2; y)} = \mathbb{E}_{\theta_2}\left[\frac{\mathcal{L}_c(\theta_1; y, X)}{\mathcal{L}_c(\theta_2; y, X)} \middle| Y = y\right] \tag{24}$$

See Chan and Ledolter (1995) for a derivation. To apply Equation 24 to the MCEM algorithm, we replace the conditional expectation on the right-hand side with a Monte Carlo average from the corresponding conditional distribution. This adjustment allows us to estimate log-likelihood ratios from the observed data distribution, without ever directly evaluating the observed data likelihood.

The method of Chan and Ledolter can be divided into two parts. The first part is a pilot study, in which we compute a standard error for our log-likelihood ratio estimator near the MLE, and determine what Monte Carlo size is required to get this standard error below a pre-specified threshold. In the second part, we increase the Monte Carlo size appropriately, and continue iterating until a confidence interval for the true observed data log-likelihood ratio contains zero. This two-part procedure reflects the suggestion of Wei and Tanner (1990) to run MCEM until it appears to stabilize, then increase the Monte Carlo sample size to get a more precise estimate.

The pilot study portion of Chan and Ledolter's method proceeds with a fixed, "moderately large" Monte Carlo size. In addition to tracking the parameter estimates across iterations, we also record the estimated log-likelihood ratio of the current estimate relative to the starting point of the algorithm. This ratio is computed by keeping a running cumulative sum of all one-step log-likelihood ratios.

Note that the observed data log-likelihood ratio is estimated by a Monte Carlo average. In order to avoid bias, we do not use the same Monte Carlo sample to estimate this ratio as we used to compute the current parameter update. Instead, we generate the Monte Carlo sample which will be used to compute the next iteration's parameter update, and use this new sample to estimate the log-likelihood ratio[2]. Since Monte Carlo variability is independent across the two iterations' samples (something, something, conditional independence), we remove any possible bias from our estimate of the log-likelihood ratio. We are then free to use this new Monte Carlo sample to compute the next iteration's parameter update.

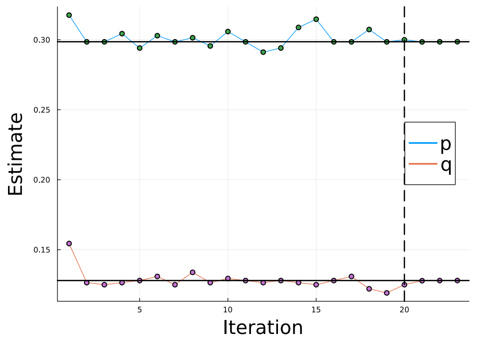
Upon completion of the pilot study (i.e., after a pre-specified number of iterations), we select the estimate with maximal estimated observed data likelihood. We also select a few nearby estimates (Chan and Ledolter suggest the 10 which follow the maximizer), and perform a small number of single-iteration MCEM runs from each of the chosen estimates. We compute the estimated observed data log-likelihood ratio for each of these MCEM updates, and get the variance of these updates for each parameter estimate. We then pool variances across parameter estimates into a standard error for our log-likelihood ratio estimator. Finally, we use this standard error, along with its known scaling rate with the Monte Carlo sample size[3], to determine what size is needed to get our standard error below the pre-specified threshold.

With this newly determined Monte Carlo size, we return to the optimal parameter estimate from our pilot study and continue iterating the MCEM algorithm. At each step now, we also construct a confidence interval for the true observed data log-likelihood ratio

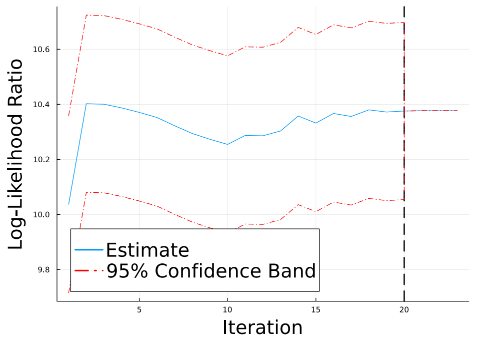
[2]Due to our Monte Carlo sample using the new parameter estimate rather than the old, we must actually estimate the reciprocal of the likelihood ratio which we want, then multiply its log by -1. This is reflected in the formulas of Chan and Ledolter but not discussed explicitly.

[3]Provided that the parameter estimates used to compute our standard error are close enough to the MLE, Chan and Ledolter show that the standard error scales like $1/M$ rather than the usual $1/\sqrt{M}$, where M is the Monte Carlo size.

Figure 3: MCEM estimates of p and q for the blood type example, based on the method of Chan and Ledolter (1995). The vertical dashed line shows the end of the pilot study. The horizontal lines correspond to maximum likelihood estimates.



(a) Estimate trajectories



(b) Estimated log-likelihood ratio trajectory

corresponding to the current parameter update (i.e., between two consecutive parameter values, not from the starting point to the current estimate). We terminate the algorithm when such a confidence interval contains zero. This corresponds to no evidence of an improvement in the observed data likelihood.

Applying Chan and Ledolter's method to our blood-type dataset, we get the parameter estimate trajectory shown in Figure 3a. Figure 3b gives the trajectory of estimated observed data log-likelihood ratios relative to the starting point of the algorithm ($\hat{p}_0 = \hat{q}_0 = 1/3$), along with pointwise 95% Wald-type confidence bands using the standard error computed after the pilot study and scaled appropriately for the Monte Carlo sample size. We use a stopping rule of 10^{-6}. Note that this standard error only applies near the maximizer of the observed data MLE.

2.3 Uncertainty Quantification for the Parameter Estimate (Booth and Hobert, 1999)

Booth and Hobert (1999) use a somewhat different approach from either Wei and Tanner (1990) or Chan and Ledolter (1995) to understand the behaiour of the MCEM algorithm. The method of Booth and Hobert is based on quantifying Monte Carlo uncertainty of the MCEM update as an approximation to the update which would be made by the deterministic EM algorithm from the same starting point. They then recommend starting the MCEM algorithm with a small Monte Carlo size, and adding more observations only when the parameter estimates are no longer changing discernibly across iterations. More formally, Booth and Hobert suggest building a confidence interval for the EM update based on the Monte Carlo variability of the MCEM update at each iteration. If this interval contains the previous iteration's parameter estimate, then the parameter updates are too small relative to the amount of Monte Carlo variability and more samples are required. These authors similarly recommend assessing convergence by checking for small relative error in the parameter updates. To account for the possibility of Monte Carlo variability leading to two consecutive estimates being similar before the algorithm has 'converged', they suggest waiting until the relative error is small for three consecutive iterations.

The confidence interval used to quantify Monte Carlo uncertainty within an iteration is obtained by framing each step of the MCEM algorithm as the solution of an M-estimation problem. This allows us to inherit the desirable properties of M-estimators; specifically, asymptotic normality (see, e.g. van der Vaart, 1998). Following the usual M-estimator construction and assuming that the relevant regularity conditions hold, we are able to estimate the asymptotic variance of the MCEM parameter estimator at each iteration. Note that this standard error is based on the Monte Carlo variability within an iteration; it does not measure sampling variability due to the observed data.

More formally, write $\tilde{\theta}_k$ for the EM update based on $\hat{\theta}_{k-1}$. Note that $\hat{\theta}_{k-1}$ is held fixed. Analysis of a single MCEM iteration is done conditional on the previous iteration (awk?). Unless stated otherwise, all expectations are taken with $\theta = \hat{\theta}_{k-1}$. Assuming sufficient smoothness and moment conditions, we get the following expression for the MCEM update:

$$\sqrt{M}(\hat{\theta}_k - \tilde{\theta}_k) = -\sqrt{M}\left[\nabla^2 Q(\tilde{\theta}_k|\hat{\theta}_{k-1})\right]^{-1}\left[\nabla \hat{Q}(\tilde{\theta}_k|\hat{\theta}_{k-1})\right] + o_p(1) \tag{25}$$

where M is the Monte Carlo size and ∇ denotes differentiation with respect to the left argument of Q or $\hat{Q}$. Note that the first expression on the right-hand side is the inverse Hessian of the EM objective function (fixed) while the second is the gradient of the MCEM objective function (an average). Thus, $\hat{\theta}_k$ is asymptotically normal with asymptotic variance

$$\left[\nabla^2 Q(\tilde{\theta}_k|\hat{\theta}_{k-1})\right]^{-1} \mathbb{V}\left[S_c(\tilde{\theta}_k)|Y = y\right]\left[\nabla^2 Q(\tilde{\theta}_k|\hat{\theta}_{k-1})\right]^{-1} \tag{26}$$

$$\approx \left[\nabla^2 \hat{Q}(\hat{\theta}_k|\hat{\theta}_{k-1})\right]^{-1} \hat{\mathbb{E}}\left[S_c(\hat{\theta}_k)S_c(\hat{\theta}_k)^T|Y = y\right]\left[\nabla^2 \hat{Q}(\hat{\theta}_k|\hat{\theta}_{k-1})\right]^{-1} \tag{27}$$

where S_c is the complete data score vector, and $\hat{\mathbb{E}}$ is the Monte Carlo average over the missing data, with $\hat{\theta}_k$ held fixed (remove comma?). Note that there is no first moment term in the conditional variance of S_c because $\hat{\theta}_k$ is a maximizer of $\hat{\mathbb{E}}[\ell_c|Y = y]$.

Based on the above discussion, we can build an asymptotic confidence interval for $\tilde{\theta}_k$, the EM update based on the MCEM estimate from iteration k. Booth and Hobert recommend checking whether this interval contains $\hat{\theta}_{k-1}$ and, if so, increasing M for the next iteration. Specifically, they suggest starting the next iteration with M/r more points, with $r = 3, 4$ or 5 working well in their examples (this whole discussion is pretty awkward).

To assess convergence of the MCEM algorithm, Booth and Hobert present two criteria. The first is a familiar measure of relative error in parameter estimates between consecutive iterations:

$$\max_j \left(\frac{\left|\hat{\theta}_{k,j} - \hat{\theta}_{k-1,j}\right|}{\left|\hat{\theta}_{k-1,j}\right| + \delta_1} \right) < \delta_2 \tag{28}$$

where δ_1 and δ_2 are small positive constants, and the subscript j ranges over components of θ. Booth and Hobert suggest using $\delta_1 = 10^{-3}$ and δ_2 between $2 \cdot 10^{-3}$ and $5 \cdot 10^{-3}$. See (Citation Needed) (probably Searle et al., 2006, p. 436, or Marquardt, 1963) for why condition (28) has this particular form.

Alternatively, since Booth and Hobert apply their method to the analysis of generalized linear mixed models, where pathologies may arise due to parameter estimates being too close to a boundary, they propose a second stopping rule:

$$\max_j \left(\frac{\left|\hat{\theta}_{k,j} - \hat{\theta}_{k-1,j}\right|}{\sqrt{\mathbb{V}\hat{\theta}_{k-1,j}} + \delta_1'} \right) < \delta_2' \tag{29}$$

How do we estimate the variance here? The purpose of condition (29) is to detect when estimated variance components are very close to zero and the numerical precision needed to satisfy condition (28) requires a prohibitive amount of computation.

We apply the method of Booth and Hobert (1999) to our blood type example, with the settings recommended in their paper. Specifically, they suggest setting $\alpha = 0.25$, $k = 3$ (alternatively, 4 or 5), $\delta_1 = 0.001$, and $\delta_2 = 0.002$ (or as high as 0.005). We also start with a Monte Carlo size of 10. Figure 4 gives trajectories of the MCEM estimates, as well as the Monte Carlo size used to obtain each of these estimates. Note how the trajectory stabilizes as the MC size increases. The final estimate from this method is $\hat{p} = 0.299$, $\hat{q} = 0.128$; again, very close to the MLE.

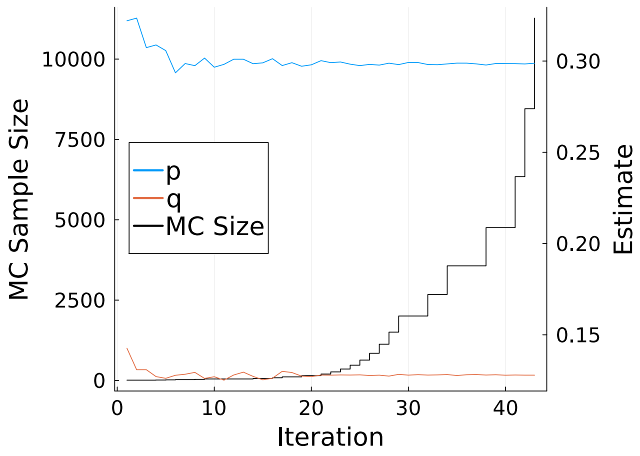
2.4 (Levine and Casella, 2001)

I don't think there is enough innovation on the MCEM algorithm in this paper. Their real contributions are on a way to apply importance sample (**??**) and a way to do UQ with MCMC sampling (4.3). These can both just be mentioned in their respective sections.

The method of Levine and Casella (2001) has two main innovations. First is a way to save computation time by sharing a single Monte Carlo sample across all MCEM iterations. Second is a way to apply the uncertainty quantification ideas of Booth and Hobert (1999) when using Markov Chain Monte Carlo sampling (see Section 4.3).

Importance sampling is a method for computing

Figure 4: Trajectory of estimates for p and q, as well as Monte Carlo sample sizes, from the method of Booth and Hobert.



2.5 Uncertainty Quantification for the Objective Function (Caffo et al., 2005)

The approach of Caffo et al. (2005) is similar in spirit to that of Booth and Hobert (1999). Both sets of authors seek to quantify Monte Carlo uncertainty in the MCEM algorithm as an approximation to the EM algorithm. The difference is that where Booth and Hobert measure uncertainty in the parameter estimate, Caffo et al. focus on uncertainty in the objective function. Specifically, Caffo et al. base their analysis on asymptotic normality of the MCEM increment:

Proposition 2.2. *Let $\Delta\hat{Q}(\hat{\theta}_k|\hat{\theta}_{k-1}) = \hat{Q}(\hat{\theta}_{k-1}|\hat{\theta}_{k-1}) - \hat{Q}(\hat{\theta}_k|\hat{\theta}_{k-1})$. Define $\Delta Q(\hat{\theta}_k|\hat{\theta}_{k-1})$ similarly. Let M_k be the Monte Carlo size at iteration k. Then, assuming some regularity conditions,*

$$\sqrt{M_k}\left[\Delta\hat{Q}(\hat{\theta}_k|\hat{\theta}_{k-1}) - \Delta Q(\hat{\theta}_k|\hat{\theta}_{k-1})\right] \rightsquigarrow N(0, \Sigma_k) \tag{30}$$

As $M_k \to \infty$, where Σ_k is an asymptotic covariance matrix.

Proof. Following Caffo et al. (2005), we write

$$\begin{aligned}
\sqrt{M_k}\left[\Delta\hat{Q}(\hat{\theta}_k|\hat{\theta}_{k-1}) - \Delta Q(\hat{\theta}_k|\hat{\theta}_{k-1})\right] &= \sqrt{M_k}\left[\Delta\hat{Q}(\theta_k|\hat{\theta}_{k-1}) - \Delta Q(\theta_k|\hat{\theta}_{k-1})\right] \\
&+ \sqrt{M_k}\left[\Delta\hat{Q}(\hat{\theta}_k|\hat{\theta}_{k-1}) - \Delta Q(\theta_k|\hat{\theta}_{k-1})\right] \qquad (31) \\
&+ \sqrt{M_k}\left[\Delta\hat{Q}(\theta_k|\hat{\theta}_{k-1}) - \Delta Q(\hat{\theta}_k|\hat{\theta}_{k-1})\right] \qquad (32) \\
&=: A_k + B_k + C_k \qquad (33)
\end{aligned}$$

First, note that A_k depends on the current Monte Carlo sample only through $\hat{Q}$, and is thus asymptotically normal by the ordinary Central Limit Theorem. However, B_k and C_k require a more careful analysis.

Use Taylor's Theorem when you're more awake. □

Provided that we are able to estimate Σ_k, Proposition 2.2 allows us to build asymptotic confidence intervals for the EM increment, ΔQ. Recall that in Section 1.1.1, we defined the Generalized EM algorithm by requiring that $\Delta Q \geq 0$, and showed that this requirement guarantees the ascent property. While the stochastic nature of the MCEM algorithm makes it impossible to guarantee that the EM increment is positive, we are able to use Proposition 2.2 to construct asymptotic confidence bounds for ΔQ. Provided that we can estimate Σ_k, we can then be reasonably confident that $\Delta Q > 0$ (awk).

Estimating the asymptotic variance under iid or importance sampling is fairly straightforward. Importance sampling however, is somewhat more complicated; particularly when a normalizing constant must be estimated. Caffo et al. give a formula for importance sampling based on the Delta Method. They also give some guidance for calculating standard errors based on MCMC sampling, which we do not go into here. See Section XX for some details.

We now return to the key MCEM problems of choosing the Monte Carlo size and when to terminate. For the former, Caffo et al. advise constructing a lower confidence limit for the EM increment, ΔQ. If this limit is positive, then we proceed to the next iteration. If not, then we augment the Monte Carlo sample at the current iteration (with, say, M_k/r, with r some small positive integer as in Booth and Hobert, 1999), and compute a new confidence bound. At the next iteration, Caffo et al. advise using a starting Monte Carlo sample which is at least as large as the final sample from the previous iteration. In fact, a larger sample may be required based on extrapolating the MC variability from the previous iteration (clarify; I don't fully understand what they're doing here).

Caffo et al. base their termination criterion on stopping when there is evidence that the algorithm is no longer yielding sufficient improvement in the EM objective function. Specifically, they start by choosing a tolerance, $\tau > 0$, then calculate an upper confidence limit for the EM increment at each iteration. If this upper confidence limit is below τ, then we declare that there is little room for improvement left in the EM objective, and terminate our algorithm.

3 Alternatives to the MCEM Algorithm

In this section, we outline a few alternatives to the MCEM algorithm for maximizing the likelihood of an incomplete dataset. Examples include the Monte Carlo Maximum Likelihood method of Geyer (1994), and Variational Inference (Blei et al., 2017; Tsikas et al., 2008).

3.1 Monte Carlo Maximum Likelihood

3.2 Stochastic Approximation

3.3 Variational Methods

This section was written by memory (i.e. without looking anything up). The idea is to more efficiently get words on the page, then I can fix things later.

Variational inference is a set of methods for approximating intractable densities (Citation Needed). The literature on variational inference is vast. See Blei et al. (2017) for a recent survey paper. Gelman et al. (2013, Section 13.7) give a textbook-level overview focusing on applications to Bayesian inference. Numerous authors have discussed the connection between variational inference and the EM algorithm; see, e.g., Neal and Hinton (1998); Tsikas et al. (2008).

The central idea of variational inference is to frame the target density as the exact solution of a functional optimization problem, where the decision variable typically ranges over densities. The domain of functions is then restricted to some set which is easier to work with. Finally, optimization is performed over this restricted class of functions. The result of this optimization is then our approximation to the target density. This discussion doesn't make it clear whether our estimator is the optimizer or the optimal objective function.

More generally, the optimization problem described above may be just one in a sequence of problems which may form part of an iterative algorithm. Herein lies the connection to the EM algorithm. In the M-step of EM, we compute expectations with respect to a particular conditional distribution. To embed this procedure in the variational inference framework, we define an optimization problem whose solution is the same conditional distribution.

We now give some details. First, let q be some probability density for the missing data,

X. We can write the observed data log-likelihood as:

$$\begin{aligned}\ell(\theta; y) &= \log f_c(y, x; \theta) - \log f_m(y, x; \theta) && (34)\\ &= \log\left[\frac{f_c(y, x; \theta)}{q(x)}\right] - \log\left[\frac{f_m(y, x; \theta)}{q(x)}\right] && (35)\\ &= \mathbb{Q}\log\left[\frac{f_c(y, X; \theta)}{q(X)}\right] - \mathbb{Q}\log\left[\frac{f_m(y, X; \theta)}{q(X)}\right] && (36)\\ &=: F(q, \theta) + \mathrm{KL}(q \to f_m(\theta)) && (37)\\ &\geq F(q, \theta) && (38)\end{aligned}$$

where line (36) holds because the left-hand side does not depend on x, and the last line follows from non-negativity of the KL divergence. The first term in line (37) is called the "evidence lower-bound" (ELBO), as well as the "variational free energy", depending on the field of application (Citation Needed). The latter name comes from an analogue in physics (Citation Needed). The former name comes from applications to Bayesian inference, where the observed data log-likelihood can be seen as an evidence if we take Y to be the data and X to be the parameters.

The EM algorithm can be re-framed as alternately maximizing F with respect to q and θ. To see why, first recall that the KL divergence between two distributions is non-negative, and is zero if and only if the two distributions are equal. Starting at a fixed parameter value, θ_0, maximizing F with respect to q is accomplished by setting $q(x) = f_m(y, x; \theta_0)$, as this minimizes the KL divergence in (37) and the left-hand side is constant in q. The resulting ELBO is equal to $Q(\theta|\theta_0) + \xi$, where Q is the EM objective function and ξ is constant in θ. Maximizing the ELBO in θ is therefore equivalent to maximizing the EM objective function. This maximization gives a new parameter value, which is used as input to the next iteration. See Theorem 1 of Neal and Hinton (1998) for a more formal proof.

While the EM algorithm is obtained by maximizing q over an unrestricted class of densities, other procedures can be formulated by restricting this class. One popular example is the "mean-field" approximation, which involves optimizing over the class of densities which factor over their arguments. That is, the class of functions $\mathcal{Q}_{MF} := \{\text{Densities } q : q(X_1, \ldots, X_p) = \prod_{j=1}^{p} q_j(X_j)\}$.

A major advantage of the mean-field approximation is that an iterative algorithm exists for finding the density, q, which maximizes the ELBO. This algorithm performs coordinate ascent, and the coordinate updates are closely related to computation of the full conditional distributions in Gibbs sampling (Citation Needed). Write $q^{(k)} = \prod q_j^{(k)}$ for the current value of q, and $\mathbb{Q}_{-j}^{(k)}$ for expectation with respect to all the missing variables except j, with distributions from $q^{(k)}$ (awk?). The update formula is

$$q_j^{(k+1)} \propto \exp\left[\mathbb{Q}_{-j}^{(k)} \ell_c(y, x_j, X_{-j})\right] \tag{39}$$

where X_{-j} is all the missing variables other than X_j. See Section 2.4 of Blei et al. (2017) for a derivation of (39). The overall algorithm consists of repeatedly cycling through updating each coordinate's distribution until some convergence criterion is met (how is convergence assessed?).

Note that, so far, our discussion of how to compute the mean-field approximate density for X has not addressed θ. To apply mean-field variational inference to EM-type problems, we substitute the mean-field density into the ELBO and maximize over θ. This new value of θ is fed back into (39), giving us a different complete data likelihood function and, hence, a new optimal density.

4 Simulation

An obstacle to implementing the MCEM algorithm which was not addressed in Section 2, and which is also relevant to many of the methods described in Section 3, is how to generate the necessary Monte Carlo samples. It is in general a hard problem to simulate from arbitrary conditional distributions. In this section, we discuss several methods for simulating the required observations at each step of the MCEM algorithm. All the topics that we cover here have their own vast literature, which we cannot hope to cover in their entirety. We instead give only a brief overview, focusing on aspects which are particularly relevant to use with the MCEM algorithm, and direct the reader to other, more focused, reviews.

4.1 Importance Sampling

Broadly speaking, importance sampling is a framework for approximating intractable expectations. A typical use case is when we want to evaluate the expected value of some function, $\mathbb{F}h$, and this expectation is not just analytically intractable, but the distribution bF is impossible (or impractical) to sample from. The latter restriction prevents us from using ordinary Monte Carlo integration. Instead, we re-write $\mathbb{F}h$ as $\mathbb{G}[h \cdot (f/g)]$ for some new distribution, $\mathbb{G}$, where f and g are the densities of $\mathbb{F}$ and $\mathbb{G}$ respectively. Provided that $\mathbb{G}$ is easy to sample from, we can estimate this alternative expression for our target expectation via Monte Carlo integration with samples drawn from $\mathbb{G}$. We call $\mathbb{F}$ the target distribution and $\mathbb{G}$ the proposal distribution.

A classic reference on importance sampling and other Monte Carlo methods is the book by Robert and Casella (2004); particularly Chapters 3 and 4. Chapter 8 of the book by Chopin and Papaspiliopoulos (2020) gives a more current overview of importance sampling, with a focus on its application to Sequential Monte Carlo. Elvira and Martino (2022) give a survey of modern methods for extending the importance sampling framework, with an emphasis on two main approaches: multiple importance sampling and adaptive importance sampling. See Elvira et al. (2019) for a review of multiple importance sampling methods, and Bugallo et al. (2017) for more on adaptive importance sampling. Agapiou et al. (2017)

give a survey paper level treatment of some more theoretical considerations for importance sampling.

In the rest of this section, we describe a few simple modifications which can ease implementation and improve performance when using importance sampling with the MCEM algorithm.

When using importance sampling with the MCEM algorithm, our target distribution is the missing data distribution. In some settings, this distribution may be difficult to describe exactly. However, the integrand is the (log-)likelihood for the complete data distribution, so it is reasonable to expect that we can evaluate this complete data density. From the definition of conditional probability, the missing data density is proportional to the complete data density, provided that we treat the latter as a function of the missing data and hold the observed data fixed. The proportionality constant here is the observed data density, so it is unlikely that we will be able to normalize the missing data density exactly (otherwise, we could just work directly with the observed data likelihood).

A simple modification of importance sampling, which circumvents the need for normalized densities, is to compute the importance weights as normal, then normalize them to sum to one. This is referred to as "auto-normalized", or "self-normalized", importance sampling (see, e.g., Elvira and Martino, 2022). The reason self-normalized importance sampling works is that the unknown normalizing constant cancels in the numerator and denominator of our normalized weights. In fact, our proposal distribution can also be unnormalized, and the ratio of the two normalizing constants cancels when we normalize our weights.

There are, however, disadvantages of self-normalized importance sampling compared to the exact importance sampling which we can do when both target and proposal densities are known exactly. One important limitation is that our estimator of $\mathbb{F}h$ is no longer unbiased, and is instead only asymptotically unbiased. In fact, as a ratio estimator, the standard error of a self-normalized importance sampling estimator can only be determined asymptotically. This is fine in problems where it is easy to sample from our proposal distribution, $\mathbb{G}$, but in high dimensional problems for example, even simulating from $\mathbb{G}$ may be costly, and more care must be taken with the discrepancy between asymptotic results and finite-sample behaviour.

It is well-known that the performance of an importance sampling estimator depends on how closely the proposal distribution matches the target (Robert and Casella, 2004). One thing that can go wrong is if the importance weight (i.e. the likelihood ratio between these two distributions) does not have sufficiently many finite moments (Agapiou et al., 2017). A simple modification to our importance weights which guarantees infinitely many finite moments is to specify a threshold value, and truncate any weights which fall above the threshold (i.e., set weights which fall above the threshold equal to the threshold value). This "truncated importance sampling" method is proposed and analyzed by Ionides (2008). Two strategies are proposed in this work for selecting the threshold: the first is to simply use the square root of the Monte Carlo size, $\sqrt{M}$, while the latter is based on unbiased risk estimation and gives a value better tailored to the specific problem. Note that these recommendations are based on exact importance sampling. If self-normalization is used, the general recommendation is instead to truncate at $\sqrt{M}$ times the mean of the un-normalized weights. Truncated importance sampling represents an example of the bias-variance trade-off. Truncating weights at some threshold reduces the variance of our importance sampling estimator, but also introduces bias. This trade-off highlights the importance of selecting an appropriate threshold value: too large and the variance reduction will be negligible, but too small and the bias will be unacceptable.

Vehtari et al. (2022) propose an alternative method for handling large importance weights, called Pareto Smoothed Importance Sampling (PSIS). The idea of this method is to fit a Generalized Pareto Distribution to the largest importance weights, then replace those large weights with quantiles of the fitted distribution. One of the parameter's fitted values also serves as a useful diagnostic for how closely our proposal distribution matches the target. An advantage of PSIS over the truncated importance sampling method of Ionides (2008) is that the bias here is smaller, although it is not clear in general which method has better mean-squared error (see Vehtari et al., 2022, for extensive numerical comparisons).

Further discuss application to MCEM? Illustrate application to MCEM?

An approach proposed by Levine and Casella (2001) seeks to use importance sampling to save computing time when running the MCEM algorithm. This computational efficiency is especially important in their context where Markov Chain Monte Carlo sampling is used (see 4.3), so generating a single sample takes quite some time. Their idea is to generate a single sample from the missing data distribution with some reference value for θ, θ_{ref}. This sample is then re-used at every MCEM iteration, with conditional expectations under the current parameter value computed by taking importance ratios with respect to θ_{ref}. This raises the question of whether a single proposal distribution can be adequate for every target distribution along the MCEM trajectory. To address this concern, Levine and Casella suggest running MCEM for a few iterations with fresh importance samples and only drawing the sample which will be used for their method after a sufficiently long "burn-in" period (they suggest running for one minute).

4.2 Particle-Based Methods

It might be worth cutting this section. I can describe SMC, but I can't say much about how it relates to MCEM besides being another Monte Carlo algorithm. That being said, the work of Golchi and Campbell (2016) is worth mentioning in the context of MCEM. Also, the work of Moffa and Kuipers (2014), which is cited by Golchi and Campbell. I will need to re-read both of these and look into who they are cited by.

4.3 Markov Chain Monte Carlo

The core idea of Markov Chain Monte Carlo (MCMC) sampling is to construct a Markov Chain from which we can simulate, and which has stationary distribution equal to the target distribution. Popular methods to construct such a Markov Chain are Gibbs sampling and the Metropolis-Hastings algorithm. See Gelman et al. (2013) or Robert and Casella (2004) for excellent textbook-length overviews. A popular implementation of MCMC sampling is the `Stan` programming language (Stan Development Team, 2022), and its `R` interface, `RStan` (Stan Development Team, 2023).

Both Gibbs sampling and Metropolis-Hastingsstart with a random variable, $X = (X_1, \ldots, X_d)$, which we wish to simulate. Let f be the density of X. These methods proceed by iteratively simulating draws of the vector X from some distribution based on the draw from the previous iteration.

Gibbs sampling is based on successively sampling each component of X conditional on all the other components. The order in which this conditioning is performed is a bit subtle, however. When generating X_i, we condition on the values of $X_1, \ldots, X_{i-1}$ from the current iteration and the values of $X_{i+1}, \ldots, X_d$ from the previous iteration. Once we reach the end of X, we start a new iteration. More generally, we can group the components of X, and simulate an entire group conditional on the others, provided that we follow the structure outlined above (awk).

The Metropolis-Hastings algorithm closely resembles rejection sampling. We start each iteration by generating a candidate value of X from some proposal distribution (this distribution may depend on the previous iteration's value of X). Write $J(x|x_0)$ for the proposal density, where x_0 is the value of X from the previous iteration. Next, we define an acceptance probability, $r := f(x)J(x_0|x)/f(x_0)J(x|x_0)$, and accept the proposed value of X with probability $r \wedge 1$. With probability $(1 - r) \vee 0$, we reject the proposed x and instead set the current iteration's value to x_0. We then proceed to the next iteration. Note that rejecting still adds an observation to our Monte Carlo sample, this value just happens to be identical to the one proceeding it.

I'm not super comfortable with this material. I want to acknowledge it and use it to motivate the more complicated adjustments to make MCEM work with MCMC, but I also don't want to say anything silly. I'm also going to need references. Limit theory for estimators based on MCMC sampling are more complicated than the similar theory for estimators based on iid or importance sampling (SMC?). This additional complexity stems from the dependence between draws from an MCMC sampler. While convergence for iid and importance sampling can be established using the Law of Large Numbers and the classical Central Limit Theorem, similar results for MCMC sampling make use of the Ergodic Theorem and Markov Chain Central Limit Theorem. A challenge in implementing the Markov Chain Central Limit Theorem is that the asymptotic variance depends on pairwise covariances between points in the chain which are arbitrarily far apart. Estimation of this asymptotic variance is therefore challenging in practice.

Levine and Casella (2001) propose a method to simplify the analysis of estimates based on MCMC sampling. Their approach consists of subsampling the original chain using Poisson spacings with increasing mean. The result is that a target function averaged over the subsampled chain satisfies a classical Central Limit Theorem (i.e. one with no covariance terms in the asymptotic variance). We can estimate the mean and standard error of our subsample estimator using the entire chain and construct confidence intervals for the mean of the target function. Levine and Casella apply this strategy for constructing confidence intervals along the same lines as Booth and Hobert (1999). Here, we construct a confidence interval for the gradient in the EM objective function, and increase the Monte Carlo size if the confidence interval for the current iteration contains the subsample estimate from the previous iteration. There's still something here I don't understand. What parameter is this CI for? In the paper, the CI is given by Equation 2.8 and justified by Theorem 1. A problem I'm having is that the thing which Theorem 1 asserts is asymptotically normal does not contain any parameters.

Appendix A Likelihood for Gene Frequency Estimation

In this appendix, we present details for the analysis of our example of estimating gene frequency. See Section 1.2 for formulation of the model and definition of notation.

A.1 Observed Data Likelihood, Score and Information

Let π_i be the probability of blood type i. The observed data likelihood for our model can be written as follows:

$$\begin{aligned} \ell(\theta; y) &= \log \binom{n}{y} + \sum y_i \log \pi_i(\theta) && (40) \\ &\equiv \sum y_i \log \pi_i && (41) \\ &\equiv 2y_1 \log r + y_2 \log(p^2 + 2pr) + y_3 \log(q^2 + 2qr) + y_4 \log pq && (42) \end{aligned}$$

where we use $\equiv$ to denote equality up to additive constants which do not depend on θ.

Differentiating ℓ wrt θ gives the observed data score.

$$\begin{aligned} S(\theta; y) &= \begin{pmatrix} \partial_p \ell(\theta; y) \\ \partial_q \ell(\theta; y) \end{pmatrix}, \text{where} && (43) \\ \partial_p \ell(\theta; y) &= -\frac{2y_1}{r} + \frac{2ry_2}{p^2 + 2pr} - \frac{2qy_3}{q^2 + 2qr} + \frac{y_4}{p} && (44) \\ \partial_q \ell(\theta; Y) &= -\frac{2y_1}{r} - \frac{2py_2}{p^2 + 2pr} + \frac{2ry_3}{q^2 + 2qr} + \frac{y_4}{q} && (45) \end{aligned}$$

Solving the score equation, $S(\theta) = 0$, thus reduces to solving a system of two polynomials in p and q. Since p and q are proportions, we reject any roots outside the unit simplex.

Differentiating ℓ again and multiplying by -1 gives the observed data information matrix. To simplify notation, let $p_y = p^2 + 2pr$ and $q_y = q^2 + 2qr$.

$$\begin{aligned} I(\theta; y) &= -\begin{bmatrix} \partial_p^2 \ell(\theta; y) & \partial_{p,q} \ell(\theta; y) \\ \partial_{p,q} \ell(\theta; y) & \partial_q^2 \ell(\theta; y) \end{bmatrix}, \text{where} && (46) \\ \partial_p^2 \ell(\theta; y) &= \frac{2y_1}{r^2} + \frac{2y_2(p_y + 2r^2)}{p_y^2} + \frac{4y_3 q^2}{q_y^2} + \frac{y_4}{p^2} && (47) \\ \partial_{p,q} \ell(\theta; y) &= \frac{2y_1}{r^2} + \frac{2y_2 p^2}{p_y^2} + \frac{2y_3 q^2}{q_y^2} && (48) \\ \partial_q^2 \ell(\theta; y) &= \frac{y_1}{r^2} + \frac{4y_2 p^2}{p_y^2} + \frac{2y_3(q_y + 2r)}{q_y^2} + \frac{y_4}{q^2} && (49) \end{aligned}$$

The asymptotic standard error of our MLE is I^{-1}, evaluated at the estimate.

A.2 Complete Data Likelihood, Score and Information

The complete data distribution for our model can be written as follows. Write ρ_i for the probability of genotype i. See Table 2 for the values of these probabilities.

$$\begin{aligned}
\ell_c(\theta; y, x) &= \log \binom{n}{x} + \sum x_i \log \rho_i(\theta) && (50)\\
&\equiv \sum y_i \log \rho_i && (51)\\
&\equiv 2x_1 \log r + x_2 \log pr + 2x_3 \log p + x_4 \log qr + 2x_5 \log q + x_6 \log pq && (52)\\
&= (2x_1 + x_2 + x_4) \log r + (x_2 + 2x_3 + x_6) \log p + (x_4 + 2x_5 + x_6) \log q && (53)\\
&= n_O \log r + n_A \log p + n_B \log q && (54)
\end{aligned}$$

where n_O, n_A and n_B are the number of times allele O, A and B arise respectively in the sampled genotypes. Note that ℓ_c depends on y only through x, so we suppress y from our notation for complete data quantities. The complete data score function is

$$\begin{aligned}
S_c(\theta; x) &= \begin{pmatrix} \partial_p \ell_c(\theta; x) \\ \partial_q \ell_c(\theta; x) \end{pmatrix}, \text{where} && (55)\\
\partial_p \ell_c(\theta; x) &= \frac{x_2 + 2x_3 + x_6}{p} - \frac{2x_1 + x_2 + x_4}{r} = \frac{n_A}{p} - \frac{n_O}{r} && (56)\\
\partial_p \ell_c(\theta; x) &= \frac{x_4 + 2x_5 + x_6}{q} - \frac{2x_1 + x_2 + x_4}{r} = \frac{n_B}{q} - \frac{n_O}{r} && (57)
\end{aligned}$$

Notice that the score is linear in x. To make this relationship explicit, we write $S_c(\theta; x) = M(\theta)x$, where $M(\theta) \in \mathbb{R}^{2\times 6}$ consists of the coefficients on x in (56) and (57). We will make use of this linearity later.

Next, we give the information matrix for the complete data.

$$\begin{aligned}
I_c(\theta; x) &= -\begin{bmatrix} \partial_p^2 \ell_c(\theta; x) & \partial_{p,q} \ell_c(\theta; x) \\ \partial_{p,q} \ell_c(\theta; x) & \partial_q^2 \ell_c(\theta; x) \end{bmatrix}, \text{where} && (58)\\
\partial_p^2 \ell_c(\theta; x) &= \frac{x_2 + 2x_3 + x_6}{p^2} + \frac{2x_1 + x_2 + x_4}{r^2} = \frac{n_A}{p^2} + \frac{n_O}{r^2} && (59)\\
\partial_{p,q} \ell_c(\theta; x) &= \frac{2x_1 + x_2 + x_4}{r^2} = \frac{n_O}{r^2} && (60)\\
\partial_q^2 \ell_c(\theta; x) &= \frac{x_4 + 2x_5 + x_6}{q^2} + \frac{2x_1 + x_2 + x_4}{r^2} = \frac{n_B}{q^2} + \frac{n_O}{r^2} && (61)
\end{aligned}$$

A.3 Missing Data Distribution

Many quantities which arise in the EM and MCEM algorithms depend on the missing data distribution (i.e. the conditional distribution of X given $Y = y$). This distribution is best described componentwise in X. First, note that X_1 and X_6 occur in Y, so we have $X_1 = y_1$

and $X_6 = y_4$. Next, we have that $X_2 + X_3 = y_2$ and $X_4 + X_5 = y_3$. Thus, we can write $X_2 \sim \text{Bin}(y_2, 2pr/(p^2 + 2pr))$ and $X_4 \sim \text{Bin}(y_3, 2qr/(q^2 + 2qr))$. Finally, we recover X_3 and X_5 by subtracting X_2 from y_2 and X_4 from y_3 respectively.

We make frequent require the first few conditional moments of X. We list them here for convenience. Let $\alpha_1 = 2pr/(p^2 + 2pr)$ be the probability parameter for the binomial distribution of X_2 given Y, and $\alpha_2 = 1 - \alpha_1$. Similarly, let $\beta_1 = 2qr/(q^2 + 2qr)$ correspond to X_4 and $\beta_2 = 1 - \beta_1$. Notation in this section is new. I'm not sure I will use it.

$$\mathbb{E}(X|Y = y) = \left(y_1, y_2\alpha_1, y_2\alpha_2, y_3\beta_1, y_3\beta_2, y_4\right)^T \tag{62}$$

$$=: \mu_m \tag{63}$$

$$\mathbb{V}(X|Y = y) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & y_2\alpha_1\alpha_2 & -y_2\alpha_1\alpha_2 & 0 & 0 & 0 \\ 0 & -y_2\alpha_1\alpha_2 & y_2\alpha_1\alpha_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & y_3\beta_1\beta_2 & -y_3\beta_1\beta_2 & 0 \\ 0 & 0 & 0 & -y_3\beta_1\beta_2 & y_3\beta_1\beta_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{64}$$

$$=: \Sigma_m \tag{65}$$

$$\mathbb{E}(XX^T|Y = y) = \Sigma_m + \mu_m\mu_m^T \tag{66}$$

Conditional expectations of the number of alleles of each kind will be of particular interest.

$$\begin{aligned}
\nu_O &:= \mathbb{E}(n_O|y) && (67)\\
&= 2y_1 + \frac{y_2 pr}{p^2+2pr} + \frac{y_3 qr}{q^2+2qr} && (68)\\
&= 2y_1 + y_2\left(\frac{\rho_2}{\rho_2+\rho_3}\right) + y_3\left(\frac{\rho_4}{\rho_4+\rho_5}\right) \qquad \left(= 2y_1 + y_2\left(\frac{\rho_2}{\pi_2}\right) + y_3\left(\frac{\rho_4}{\pi_3}\right)\right) && (69)
\end{aligned}$$

$$\begin{aligned}
\nu_A &:= \mathbb{E}(n_A|y) && (70)\\
&= \frac{2y_2 pr}{p^2+2pr} + \frac{2y_2 p^2}{p^2+2pr} + y_4 && (71)\\
&= y_2\left(\frac{\rho_2}{\rho_2+\rho_3} + 2\frac{\rho_3}{\rho_2+\rho_3}\right) + y_4 \qquad \left(= y_2\left(\frac{\rho_2}{\pi_2} + 2\frac{\rho_3}{\pi_2}\right) + y_4\right) && (72)\\
&= y_2\left(1 + \frac{p^2}{p^2+2pr}\right) + y_4 && (73)
\end{aligned}$$

$$\begin{aligned}
\nu_B &:= \mathbb{E}(n_B|y) && (74)\\
&= \frac{2y_3 qr}{q^2+2qr} + \frac{2y_3 q^2}{q^2+2qr} + y_4 && (75)\\
&= y_3\left(\frac{\rho_4}{\rho_4+\rho_5} + 2\frac{\rho_5}{\rho_4+\rho_5}\right) + y_4 \qquad \left(= y_3\left(\frac{\rho_4}{\pi_3} + 2\frac{\rho_5}{\pi_3}\right) + y_4\right) && (76)\\
&= y_3\left(1 + \frac{q^2}{q^2+2qr}\right) + y_4 && (77)
\end{aligned}$$

A.4 EM Algorithm

In order to apply the EM algorithm, we must construct and optimize the EM objective function. That is, we must compute $Q(\theta|\theta_0) = \mathbb{E}_{\theta_0}\left[\ell_c(\theta; y, X)|Y = y\right]$. The conditional distribution of X given $Y = y$ is best described componentwise in X. First, note that X_1 and X_6 occur in Y, so we have $X_1 = y_1$ and $X_6 = y_4$. Next, we have that $X_2 + X_3 = y_2$ and $X_4 + X_5 = y_3$. Thus, we can write $X_2 \sim \text{Bin}(y_2, 2pr/(p^2 + 2pr))$ and $X_4 \sim \text{Bin}(y_3, 2qr/(q^2 + 2qr))$. Finally, we recover X_3 and X_5 by subtracting X_2 from y_2 and X_4 from y_3 respectively.

Before writing out the EM objective function, we first compute conditional expectations of the sample allele counts.

$$\begin{aligned}
\mathbb{E}(n_O|y) &= 2y_1 + \frac{y_2 pr}{p^2+2pr} + \frac{y_3 qr}{q^2+2qr} && (78)\\
&= 2y_1 + y_2\left(\frac{\rho_2}{\rho_2+\rho_3}\right) + y_3\left(\frac{\rho_4}{\rho_4+\rho_5}\right) \quad \left(= 2y_1 + y_2\left(\frac{\rho_2}{\pi_2}\right) + y_3\left(\frac{\rho_4}{\pi_3}\right)\right) && (79)\\
&=: \nu_O && (80)
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}(n_A|y) &= \frac{2y_2 pr}{p^2+2pr} + \frac{2y_2 p^2}{p^2+2pr} + y_4 && (81)\\
&= y_2\left(\frac{\rho_2}{\rho_2+\rho_3} + 2\frac{\rho_3}{\rho_2+\rho_3}\right) + y_4 \quad \left(= y_2\left(\frac{\rho_2}{\pi_2} + 2\frac{\rho_3}{\pi_2}\right) + y_4\right) && (82)\\
&= y_2\left(1 + \frac{p^2}{p^2+2pr}\right) + y_4 && (83)\\
&=: \nu_A && (84)
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}(n_B|y) &= \frac{2y_3 qr}{q^2+2qr} + \frac{2y_3 q^2}{q^2+2qr} + y_4 && (85)\\
&= y_3\left(\frac{\rho_4}{\rho_4+\rho_5} + 2\frac{\rho_5}{\rho_4+\rho_5}\right) + y_4 \quad \left(= y_3\left(\frac{\rho_4}{\pi_3} + 2\frac{\rho_5}{\pi_3}\right) + y_4\right) && (86)\\
&= y_3\left(1 + \frac{q^2}{q^2+2qr}\right) + y_4 && (87)\\
&=: \nu_B && (88)
\end{aligned}$$

The EM objective function can be written as

$$\begin{aligned}
Q(\theta|\theta_0) &:= \mathbb{E}_{\theta_0}[\ell_c(\theta; X)|Y=y] && (89)\\
&\equiv \nu_O^{(0)} \log r + \nu_A^{(0)} \log p + \nu_B^{(0)} \log q && (90)
\end{aligned}$$

where a superscript zero denotes that the quantity is computed by taking an expectation under θ_0. Maximizing Q analytically with respect to θ gives the following expression for the EM update:

$$\begin{aligned}
M(\theta_{k-1}) &= \begin{pmatrix} \hat{p}_k \\ \hat{q}_k \end{pmatrix} && (91)\\
&= \begin{pmatrix} \nu_A^{(k-1)}/2n \\ \nu_B^{(k-1)}/2n \end{pmatrix} && (92)
\end{aligned}$$

where a superscript $k-1$ denotes that the quantity is computed by taking an expectation under θ_{k-1}. Note that the equation for a stationary point of the EM algorithm, $M(\theta) = \theta$, has identical roots to the observed data score equation, $S(\theta) = 0$ (confirm this).

A.4.1 Asymptotic Standard Error

Recall that the EM algorithm computes the MLE, which has asymptotic covariance matrix equal to the inverse Fisher information matrix evaluated at the true parameter value. In practice, we estimate this covariance with the inverse of the observed information matrix evaluated at the MLE. Using Proposition 1.3, we can calculate the observed information matrix using conditional expectations of quantities derived from the complete data likelihood.

To this end, we need to evaluate the conditional expectations in expression (7) of Proposition 1.3. It will be convenient for us to write $S_c(\theta) =: \mathscr{S}(\theta)X$. Then

$$\mathcal{I}_c(\hat{\theta}) = \begin{bmatrix} \frac{\nu_A}{p^2} + \frac{\nu_O}{r^2} & \frac{\nu_O}{r^2} \\ \frac{\nu_O}{r^2} & \frac{\nu_B}{q^2} + \frac{\nu_O}{r^2} \end{bmatrix}, \text{and} \tag{93}$$

$$\begin{aligned} \mathbb{E}_{\hat{\theta}}[S_c(\hat{\theta})S_c(\hat{\theta})^T | Y = y] &= \mathscr{S}(\hat{\theta})\mathbb{E}_{\hat{\theta}}\left[XX^T | Y = y\right]\mathscr{S}(\hat{\theta}) && (94)\\ &= \mathscr{S}(\hat{\theta})(\Sigma_m + \mu_M\mu_M^T)\mathscr{S}(\hat{\theta}) && (95)\\ & && (96) \end{aligned}$$

While it is possible to expand the above expressions, they quickly become too long to easily interpret. We instead leave these as computational formulas and use them as a guide for writing `R` or `Julia` code.

A.5 Monte Carlo EM

Check for “Citation Needed” before publishing.

References

S. Agapiou, O. Papaspiliopoulos, D. Sanz-Alonso, and A. M. Stuart. Importance sampling: Intrinsic dimension and computational cost. *Statistical Science*, 32(3), 2017.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: a review for statisticians. *Journal of the American Statistical Association*, 112(518), 2017.

James G. Booth and James P. Hobert. Maximizing generalized linear mixed model likelihoods with an automated monte carlo em algorithm. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 61(1), 1999.

Mónica F. Bugallo, Víctor Elvira, Luca Martino, David Luengo, Joaquín Míguez, and Petar M. Djurić. Adaptive importance sampling: The past, the present, and the future. *IEEE Signal Processing Magazine*, 34(4), 2017.

Brian S. Caffo, Wolfgang Jank, and Galin L. Jones. Ascent-based Monte Carlo expectation-maximization. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 67(2), 2005.

K. S. Chan and Johannes Ledolter. Monte Carlo EM estimation for time series models involving counts. *Journal of the American Statistical Association*, 90(429), 1995.

Nicolas Chopin and Omiros Papaspiliopoulos. *An Introduction to Sequential Monte Carlo*. Springer, 2020.

Citation Needed.

Laura Dean. Blood groups and red cell antigens [internet]. Technical report, National Center for Biotechnology Information (US), 2005.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1), 1977.

Víctor Elvira, Luca Martino, David Luengo, and Mónica Bugallo. Generalized multiple importance sampling. *Statistical Science*, 34(1), 2019.

Víctor Elvira and Luca Martino. Advances in importance sampling. *arXiv:2102.05407v3*, 2022.

Yoshiko Fujita, Masako Tanimura, and Katumi Tanaka. The distribution of the ABO blood groups in Japan. *Japanese Journal of Human Genetics*, 23, 1978.

Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. CRC Press, third edition, 2013.

Charles J. Geyer. On the convergence of Monte Carlo maximum likelihood calculations. *Journal of the Royal Statistical Society. Series B (Methodological)*, 56(1), 1994.

Shirin Golchi and David A. Campbell. Sequentially constrained Monte Carlo. *Computational Statistics and Data Analysis*, 97, 2016.

Edward L. Ionides. Truncated importance sampling. *Journal of Computational and Graphical Statistics*, 17(2), 2008.

Richard A. Levine and George Casella. Implementations of the Monte Carlo EM algorithm. *Journal of Computational and Graphical Statistics*, 10(3), 2001.

Thomas A. Louis. Finding the observed information matrix when using the EM algorithm. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 44(2), 1982.

Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2), 1963.

Geoffrey J. McLachlan and Thriyambakam Krishnan. *The EM algorithm and extensions*. Wiley, 2nd edition, 2008.

Giusi Moffa and Jack Kuipers. Sequential Monte Carlo EM for multivariate probit models. *Computational Statistics and Data Analysis*, 72, 2014.

Radford M. Neal and Geoffrey E. Hinton. *A view of the EM algorithm that justifies incremental, sparse, and other variants*, pages 355–368. Springer, 1998.

Ronald C. Neath. On convergence properties of the Monte Carlo EM algorithm. *Advances in Modern Statistical Theory and Applications*, 10, 2013.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 2nd edition, 2006.

David Oakes. Direct calculation of the information matrix via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 61(2), 1999.

Christian P. Robert and George Casella. *Monte Carlo statistical methods*. Springer, second edition, 2004.

Shayle R. Searle, George Casella, and Charles E. McCulloch. *Variance Components*. Wiley Interscience, 2006.

Stan Development Team. Stan modelling language users guide and reference manual, 2022. URL `http://mc-stan.org/`. Version 2.32.

Stan Development Team. RStan: the R interface to Stan, 2023. URL `https://mc-stan.org/`. R package version 2.21.8.

Dimitris G. Tsikas, Aristidis C. Likas, and Nikolaos P. Galatsanos. The variational approximation for Bayesian inference. *IEEE Signal Processing Magazine*, 25(6), 2008.

A. W. van der Vaart. *Asymptotic Statistics*. Cambridge University Press, 1998.

Aki Vehtari, Daniel Simpson, Andrew Gelman, Yuling Yao, and Jonah Gabry. Pareto smoothed importance sampling, 2022.

Greg C. G. Wei and Martin A. Tanner. A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85(411), 1990.

Index