Approximate Bayesian Inference Based on Expected Evaluation

Ishi Jain
220464

Nandini Bhattad
220693

Parnapalli Sarath Kumar
220754

Abstract

In this report, we present our implementation of the methodology proposed in the paper, "Approximate Bayesian Inference Based on Expected Evaluation" by Hammer et al. (2024) [1], when the likelihood function is not available or tractable. Unlike traditional approaches like Approximate Bayesian Computation (ABC) or Bayesian Synthetic Likelihood (BSL), which start with a simulation algorithm and derive an approximate posterior from it, Expected Evaluation method takes a different route. It begins by defining an interpretable approximate posterior based on the distribution of an expected discrepancy between observed and simulated data. We implemented the accompanying MCMC algorithm to sample from this posterior and tested it on a synthetic dataset. This report gives a detailed explanation of the proposed method, implementation process and discusses the performance of the method.

Existing Work

Bayesian inference aims to compute the posterior distribution:

$$f(\theta|x) \propto f(\theta)f(x|\theta),$$

where x is the observed data, θ the parameter vector, $f(x|\theta)$ the likelihood, and $f(\theta)$ the prior. Usually the likelihood function is available and tractable and most techniques to evaluate the posterior distribution relies on this.

When the likelihood function $f(x|\theta)$ is computationally intractable or unavailable, traditional Bayesian methods such as Metropolis-Hastings cannot be directly applied. To overcome this, likelihood-free inference techniques have been developed, with Approximate Bayesian Computation (ABC) and Bayesian Synthetic Likelihood (BSL) being two of the most widely used approaches.

Approximate Bayesian Computation (ABC)

Approximate Bayesian Computation (ABC) methods aim to approximate the posterior distribution without evaluating the likelihood function directly. The basic idea is to simulate data $\tilde{x} \sim f(x|\theta)$ for candidate parameter values θ, and accept those θ values for which the simulated data $\tilde{x}$ is close to the observed data x.

Closeness is typically measured via a discrepancy function $\rho(s(x), s(\tilde{x}))$, where $s(\cdot)$ is a vector of summary statistics. The classic ABC rejection algorithm accepts a proposed θ if:

$$\rho(s(x), s(\tilde{x})) \leq \epsilon$$

where $\epsilon > 0$ is a user-specified tolerance threshold. The resulting approximate posterior is:

$$\tilde{f}_\epsilon(\theta|x) \propto f(\theta) \int f(\tilde{x}|\theta)\, \mathbb{I}(\rho(s(x), s(\tilde{x})) \leq \epsilon)\, d\tilde{x}$$

Limitations of ABC:

- The choice of ϵ critically affects accuracy: small ϵ leads to better approximations but lower acceptance rates.
- The use of summary statistics introduces information loss unless they are sufficient.
- Computationally expensive due to repeated simulations and high rejection rates.
- Resulting posterior is often noisy and depends heavily on the kernel or threshold used.

Bayesian Synthetic Likelihood (BSL)

Bayesian Synthetic Likelihood (BSL) takes a different approach by modeling the distribution of summary statistics $s(x)$ as multivariate normal:

$$s(x) \sim \mathcal{N}(\mu(\theta), \Sigma(\theta))$$

Given simulated datasets $\tilde{x}_1, \ldots, \tilde{x}_N \sim f(x|\theta)$, one can estimate the sample mean $\hat{\mu}(\theta)$ and covariance $\hat{\Sigma}(\theta)$ of the summary statistics, and define the synthetic likelihood as:

$$\tilde{f}(s(x)|\theta) = \mathcal{N}(s(x); \hat{\mu}(\theta), \hat{\Sigma}(\theta))$$

This synthetic likelihood can be inserted into the Bayesian framework, enabling standard MCMC sampling. BSL does not rely on thresholds like ABC and can be more statistically efficient.

Limitations of BSL:

- Assumes the summary statistics are approximately multivariate normal, which may not hold in practice.
- Requires repeated simulations at each iteration to estimate $\hat{\mu}(\theta)$ and $\hat{\Sigma}(\theta)$.
- Less flexible than ABC in handling non-Gaussian summary distributions.
- Sensitive to the choice of summary statistics, which still need to be informative.

Both ABC and BSL provide powerful tools for likelihood-free inference but have inherent trade-offs between flexibility, efficiency, and theoretical guarantees. The method by Hammer et al. (2024), which we implement in this report, addresses some of these issues by starting with a defined approximate posterior using an expected evaluation framework.

Just like BSL, the proposed method differs from ABC in that it does not rely on thresholding in any way. Like BSL, the setup includes taking an expected value with respect to the generated variable $\tilde{x}$. However, in BSL the expectation is of an auxiliary likelihood function and the expected value is used directly as the approximate likelihood, whereas in this setup the expectation is of a more freely chosen evaluation function and the expected value is just a new stochastic variable for which an approximate likelihood function can be defined.

Approximating the likelihood

The key idea is to define an evaluation function $s(x, \tilde{x})$ that quantifies the discrepancy between the observed data x and synthetic data $\tilde{x} \sim f(\tilde{x}|\theta)$. Instead of using $s(x, \tilde{x})$ directly, as in ABC methods, the approach constructs an expected evaluation:

$$r(x, \theta) = \mathbb{E}[s(x, \tilde{x}) \mid x, \theta] = \int s(x, \tilde{x}) f(\tilde{x} \mid \theta)\, d\tilde{x}$$

This expected evaluation $r(x, \theta)$, which is a scalar function of θ, is then used to define an approximate likelihood $g(r(x, \theta) \mid \theta)$, leading to the following form of the approximate posterior distribution:

$$g(\theta \mid x) \propto f(\theta)\, g(r(x,\theta) \mid \theta)$$

This formulation introduces two approximations:

- Conditioning on $r(x,\theta)$ rather than the full data x, which introduces information loss unless $r(x,\theta)$ is sufficient.
- Approximating the true but intractable distribution of $r(x,\theta)$ with a tractable distribution $g(r \mid \theta)$, often chosen to be Gaussian for simplicity.

If $s(x,\tilde{x})$ is constructed as a sample mean or similar statistic, then $r(x,\theta)$ can be viewed as a sum of weakly dependent random variables. Under suitable conditions, the Central Limit Theorem justifies a normal approximation for $r(x,\theta)$. Thus, the approximate likelihood $g(r \mid \theta)$ is typically modeled as:

$$g(r(x,\theta) \mid \theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(r(x,\theta)-\mu)^2}{2\sigma^2}\right)$$

This approach does not require any thresholding or kernel smoothing, unlike ABC. It also avoids reliance on auxiliary models, which is done in BSL methods. By formulating an explicit likelihood, the resulting posterior distribution is smooth, interpretable, and more principled in its construction.

In our implementation, we followed this methodology to define the approximate likelihood and used MCMC to estimate posterior quantities based on unbiased likelihood estimators, as outlined in the paper.

Estimating the posterior

Once the approximate likelihood $g(r(x,\theta) \mid \theta)$ is defined, the next step is to estimate the posterior distribution:

$$g(\theta \mid x) \propto f(\theta)\, g(r(x,\theta) \mid \theta)$$

However, since $r(x,\theta) = \mathbb{E}[s(x,\tilde{x}) \mid x,\theta]$ is itself an expectation, we cannot compute $g(r(x,\theta) \mid \theta)$ exactly. To overcome this, we implemented a posterior estimation strategy based on Markov chain Monte Carlo (MCMC), using an unbiased estimator $\hat{g}(r(x,\theta) \mid \theta, u)$ of the approximate likelihood.

Here, u denotes the collection of random variables used to compute the unbiased estimator. The estimator satisfies the following:

$$\mathbb{E}_u[\hat{g}(r(x,\theta) \mid \theta, u)] = g(r(x,\theta) \mid \theta)$$

To ensure valid inference despite potential negative values of the estimator, an augmented target distribution is defined over (θ, u):

$$\breve{g}(\theta, u \mid x) \propto f(\theta)\ |\hat{g}(r(x,\theta) \mid \theta, u)|\ h(u \mid \theta)$$

where $h(u \mid \theta)$ is the distribution used to sample u. Sampling from this target is done via a Metropolis–Hastings algorithm.

Given samples $\{(\theta_k, u_k)\}_{k=1}^{M}$ from $\breve{g}(\theta, u \mid x)$, posterior expectations are estimated as:

$$\widehat{\mathbb{E}}[\psi(\theta)] = \frac{\sum_{k=1}^{M} \psi(\theta_k)\, \mathrm{sign}(\hat{g}(r(x,\theta_k) \mid \theta_k, u_k))}{\sum_{k=1}^{M} \mathrm{sign}(\hat{g}(r(x,\theta_k) \mid \theta_k, u_k))}$$

where $\psi(\theta)$ is any statistic that we want to estimate which is a function of μ and σ. This estimator is unbiased and consistent under regularity conditions. In our implementation, this MCMC scheme was used to estimate posterior quantities for the approximate model defined in the previous section.

Unbiased estimate for the approximate normal likelihood

When modeling $r(x,\theta)$ using a normal likelihood, we wish to compute:

$$g(r(x,\theta)|\theta) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(r(x,\theta)-\mu)^2}{2\sigma^2}\right)$$

Since $r(x, \theta)$ must be estimated via simulation, a Hermite polynomial expansion of the standard normal density is used to build an unbiased estimator.

Hermite Expansion and Truncation

Using a Taylor expansion of the Gaussian density:

$$\phi(t) = \sum_{n=0}^{\infty} \frac{(-1)^n}{n!} \phi(v) H_n(v)(t - v)^n$$

Substituting $t = \frac{r(x,\theta)-\mu}{\sigma}$ and $v = \frac{\hat{r}-\mu}{\sigma}$, the approximate likelihood can be rewritten as a series in $(r(x, \theta) - \hat{r})/\sigma$. To make the sum computable, it is truncated randomly using a stopping time τ:

$$\tilde{g}(r(x, \theta)|\theta) = \frac{1}{\sigma}\phi\left(\frac{\hat{r} - \mu}{\sigma}\right) \sum_{n=0}^{\tau} \frac{(-1)^n}{n!P(\tau \geq n)} H_n\left(\frac{\hat{r} - \mu}{\sigma}\right)\left(\frac{r(x, \theta) - \hat{r}}{\sigma}\right)^n$$

Monte Carlo Estimation of Evaluation Terms

We estimate $r(x, \theta)$ via:

$$\hat{r} = \frac{1}{\nu} \sum_{i=1}^{\nu} s(x, \tilde{x}_i), \quad \tilde{x}_i \sim f(\cdot|\theta)$$

We then compute unbiased estimates of the standardized term $\frac{r(x,\theta)-\hat{r}}{\sigma}$ using:

$$\hat{r}_{\text{std}}^{n,i}(x, \theta) = \frac{1}{m} \sum_{j=1}^{m} \frac{s(x, \tilde{x}^{n,i,j}) - \hat{r}}{\sigma}$$

These terms are used in the Hermite series to compute the unbiased estimator.

Final Estimator

Putting it all together:

$$\hat{g}(r(x, \theta)|\theta) = \frac{1}{\sigma}\phi\left(\frac{\hat{r} - \mu}{\sigma}\right) \sum_{n=0}^{\tau} \frac{(-1)^n}{n!P(\tau \geq n)} H_n\left(\frac{\hat{r} - \mu}{\sigma}\right) \prod_{i=1}^{n} \hat{r}_{\text{std}}^{n,i}(x, \theta)$$

The estimator depends on three hyperparameters:

- ν: Number of samples for $\hat{r}$
- m: Number of samples per product terms
- $p(\tau)$: Distribution for the stopping time

A geometric distribution is often used for τ, with minimum truncation level τ_0 and decay rate p:

$$p(\tau) = \begin{cases} 0 & \text{if } \tau < \tau_0 \\ p(1-p)^{\tau-\tau_0} & \text{if } \tau \geq \tau_0 \end{cases}$$

This construction ensures an unbiased estimate of the approximate likelihood usable in the Metropolis–Hastings algorithm.

Simulations

We implemented the methodology described in the paper and compared the approximate posterior obtained using the proposed Expected Evaluation method with the true posterior distribution.

We considered the model $f(x \mid \theta)$ to be a normal distribution with unknown mean μ and unknown standard deviation σ. The parameters μ and σ^2 are assumed to be a priori independent, with $\mu \sim \mathcal{N}(0, 25)$ and $\sigma \sim \text{Uniform}(0, 10)$. Observed data x_{obs} of size $n = 250$ was generated from the true parameters $\mu = 2$, $\sigma = 2$.

Exact Posterior Estimation

We implemented a Metropolis-Hastings (MH) sampler using the exact likelihood. The sampler proposes new parameter values using a Gaussian random walk and accepts them based on the standard MH acceptance rule. This gives us samples from the true posterior distribution $f(\mu, \sigma^2 \mid x)$, which we use as the gold standard.

Classification Based Evaluation Function

The `classify.R` script defines the classify function, which plays a central role in estimating a smooth discrepancy between observed and generated data. It sets up a binary classification problem where the observed data (`xObs`) are labeled as class 0 and the generated data (`xGenerated`) as class 1.

The function uses k-fold cross-validation (default $k = 5$) to train a classifier on the concatenated dataset. Two classification methods are supported:

- **Logistic Regression**: A logistic model is trained using a quadratic feature basis (`x` and `x^2`) for improved flexibility. This is the default classification method.
- **K-Nearest Neighbors (KNN)**: A KNN classifier is trained with k neighbors specified via the `classifyParam` argument.

In each fold, the model is trained on a subset of the data and evaluated on the remaining fold. Depending on the `type` argument (0 or 1), the classifier is tested on either observed or generated data. The function computes the posterior class probabilities for the test set and returns the average of these probabilities across all folds.

This averaged classification probability quantifies the distinguishability between the observed and generated samples and is used within the `rhathat` function to construct smooth, differentiable estimates of posterior density ratios.

Approximate Posterior Using Expected Evaluation

To implement the approximate inference method from the paper, we used the classification-based evaluation function explained above. We distinguished between observed data x and synthetic data $\tilde{x} \sim f(\cdot \mid \theta)$ using a logistic classifier. This yields a classification accuracy score, used to define $r(x, \theta) = \mathbb{E}[s(x, \tilde{x}) \mid \theta]$.

To evaluate the approximate likelihood, we implemented a Hermite polynomial expansion-based estimator $\hat{g}(r(x, \theta) \mid \theta)$, which uses classification-based discrepancy scores, Monte Carlo estimation for unbiasedness and random truncation using a geometric distribution.

An MCMC sampler then uses this estimated likelihood to accept or reject proposed parameter values. To ensure valid inference, sign-correction terms were applied when estimating posterior expectations, as described in the paper.

Posterior Summaries and Comparison

The true posterior probabilities were computed using numerical integration. We computed posterior summaries such as the mean and standard deviation of μ and σ for both the exact and approximate posterior samples. We used a burn-in of 500 iterations, and the remaining samples were aggregated using the sign-weighted expectation formula:

$$\mathbb{E}[\psi(\theta)] \approx \frac{\sum_{k=1}^{M} \psi(\theta_k) \cdot \text{sign}(\hat{g}(r(x, \theta_k)))}{\sum_{k=1}^{M} \text{sign}(\hat{g}(r(x, \theta_k)))}$$

where $\psi(\theta)$ is any statistic that we want to estimate which is a function of μ and σ. We also estimated the posterior over a 2D grid of (μ, σ) values defined by $[1.0, 1.2, \ldots, 3.0] \times [1.0, 1.2, \ldots, 3.0]$ using both methods and computed the L1 distance between the approximate and true posterior grids. This provided a quantitative measure of how close the approximation was to the exact posterior.

Results

In the original paper while approximating the posterior using expected evaluation the number of iterations used were 1e5. The error in absolute value estimating posterior probabilities was 0.25. However, due to computational barriers we ran MCMC for 1000, 2000 and 4000 iterations and compared the absolute error values in estimating posterior probabilities. The following results were obtained:

No. of Iterations	Absolute Error Value
1000	1.247
2000	0.9759
4000	0.8542

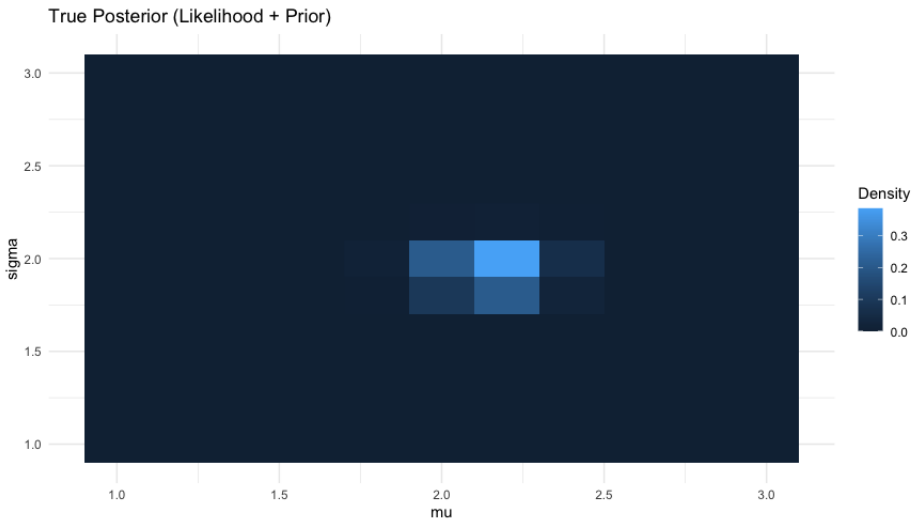
Table 1: Absolute error values for different iteration counts

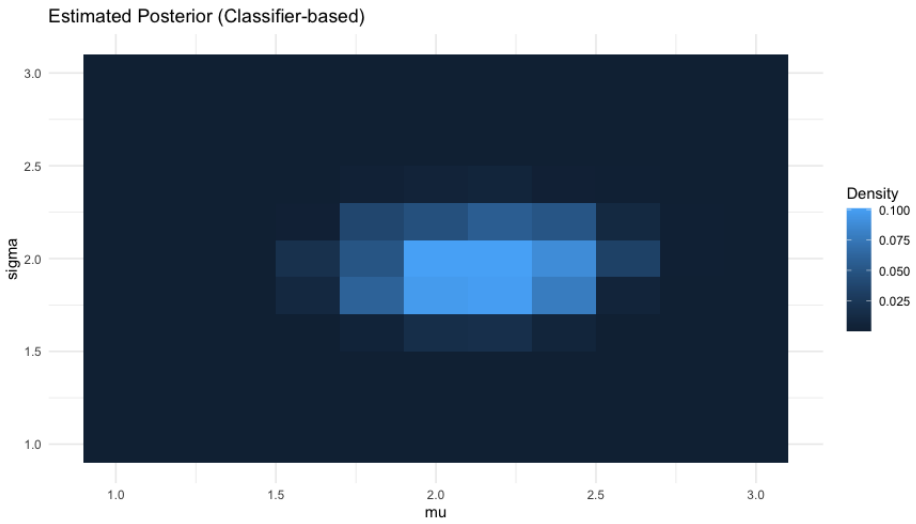
As it can be seen from the table, we got a gradual decrease in error with increase in number of iterations. There is a decrement of absolute error by ≈ 0.4 upon increasing the iterations from 1e3 to 4e3.

No. of Iterations	Mu_approx	Mu_exact	Sigma_approx	Sigma_exact
1000	1.953327	1.976519	1.841436	1.973765
2000	1.790145	1.802951	1.98301	1.877619
4000	2.144216	2.143811	1.908124	1.92724

Table 2: Estimated and true values of μ and σ for different iteration counts

The following density plots were obtained for the true and estimated posterior probabilities with 4000 iterations:





Improvements

The programming language could be shifted to C++ and the number of iterations can be increased to 1e5 or even more to reproduce exact or even better results.

Contribution

- Contribution to finding papers: The three potential research papers that were mailed were found by Nandini and Sarath.
- Contribution to understanding the methodology: All the team members went through the selected papers to understand the methodology.
- Contribution to coding: The coding for the file toy.R was done by Ishi and Nandini. The coding for the files H.R, classify.R and rhathat.R which are called in toy.R was done by Sarath.
- Generating the figures and tables: Since the implementation was computationally demanding, we divided the work to find out the errors for different number of iterations and generated the respective plots.
- Writing the report: The report was written by Ishi and Nandini.

References

[1] https://projecteuclid.org/journals/bayesian-analysis/volume-19/issue-3/Approximate-Bayesian-Inference-Based-on-Expected-Evaluation/10.1214/23-BA1368.full

[2] https://en.wikipedia.org/wiki/Hermite_polynomials

[3] https://en.wikipedia.org/wiki/Approximate_Bayesian_computation

[4] https://arxiv.org/abs/2305.05120