Rare-event Probability Estimation via Empirical Likelihood Maximization

Alvin Huang
Supervisor: Dr. Zdravko Botev

Submitted in partial fulfillment of the requirements of the degree of
Bachelor of Science Honours,
School of Mathematics and Statistics,
University of New South Wales

December 12, 2013
Acknowledgements

I would like to sincerely thank my supervisor Dr. Zdravko Botev for his patient guidance, advice and tutelage he has provided throughout my Honours year. I am grateful for him taking time to share his interesting research ideas that has perked my interests. Also, I must express my gratitude to my colleagues Aaron and Victor for keeping me company and taking the same journey in 2013. Discussions with them are insightful and helped me keep on track. I would also like to thank the School of Mathematics and Statistics at UNSW for the fruitful experience over the last 5 years. Finally, I would like to thank my family for their care and support.
Abstract

We explore past and recent developments in rare-event probability estimation with a particular focus on a novel Monte Carlo technique – Empirical Likelihood Maximization (ELM). This is a versatile method that involves sampling from a sequence of densities using MCMC and maximizing an empirical likelihood. The quantity of interest, the probability of a given rare-event, is estimated by solving a convex optimization program related to likelihood maximization. Numerical experiments are performed using this new technique and benchmarks are given against existing robust algorithms and estimators.

Keywords: Rare-event probability estimation; Monte Carlo methods; Importance sampling; Markov chain Monte Carlo; Empirical Likelihood; Convex Optimization
Chapter 1

Introduction

One of the hallmark problems in Monte Carlo simulation is the efficient estimation of a high-dimensional integral with the form

\[ Z = \int f(x)H(x) \, dx = E_f H(X), \]  

(1.1)

where \( H : \mathbb{R}^d \to \mathbb{R} \) is an arbitrary real-valued function and \( X = (X_1, \ldots, X_d) \) is a \( d \)-dimensional random vector with joint probability density function \( f \).

These integrals arise in various contexts such as in Bayesian statistics, insurance risk, financial mathematics, statistical mechanics, queuing analysis, reliability theory, and combinatorial problems in computer science (see [2], [15]). An important class of estimation problems with the form (1.1) is rare-event probability estimation. For this particular class of problems, \( H \) is the indicator function

\[ H(x) = I\{S(x) \geq \gamma\} = \begin{cases} 1 & \text{if } S(x) \geq \gamma \\ 0 & \text{if } S(x) < \gamma, \end{cases} \]

where \( S \) is a real-valued function and \( \gamma \) is a level or threshold parameter. That is, \( H \) represents the occurrence of some given event. Define \( \ell \) to be the probability of this event, then

\[ \ell = \int f(x)I\{S(x) \geq \gamma\} \, dx = E_f I\{S(x) \geq \gamma\} = P_f(S(X) \geq \gamma). \]  

(1.2)

Of interest is when the probability \( \ell \) happens to be very small, say, with order less than \( 10^{-4} \). In such cases, we define the event \( \{S(X) \geq \gamma\} \) to be a rare-event and \( \ell \) to be a rare-event probability. For example, \( \ell \) may represent the probability of ruin for an insurance company such that \( S(X) \) is the aggregate claims received and \( \gamma \) is their current capital level. Other examples include telecommunications where \( \ell \) the probability of buffer overflow, or in reliability theory where \( \ell \) is the probability of failure before some time \( \gamma \).
These probabilities are represented by high-dimensional integrals whose integrands are non-smooth. In effect, numerical integration techniques such as traditional quadrature rules and quasi-Monte Carlo methods are typically inapplicable. Thus, we resort to Monte Carlo simulation, whose convergence is usually unaffected by the dimensions of the integration and the smoothness of the integrand.

In this thesis, we utilize Monte Carlo methods to estimate these high-dimensional integrals. Particular focus is placed on a novel Monte Carlo technique, Empirical Likelihood Maximization (ELM). This is a versatile method that involves sampling from a sequence of densities using Markov chain Monte Carlo (MCMC) and maximizing an empirical likelihood. The (rare-event) probability of interest is estimated by solving a convex optimization program similar to maximizing a likelihood function.

The remaining chapters of the thesis are organized as follows. In chapter 2, we cover essential background knowledge and introduce another Monte Carlo method for rare-event estimation, Markov Chain Importance Sampling (MCIS). We also present a survey of other common rare-event probability estimation techniques. In chapter 3, we draw analogies to the maximum likelihood estimation method to motivate the key mechanism underlying the ELM approach. This will be followed up by the formulation of the ELM procedure. Further, in chapter 4, we perform numerical experiments using ELM and benchmark against existing robust estimators. In addition, critical analysis of strengths and shortcomings of the ELM approach is given. Finally, in chapter 5, we provide concluding remarks and directions for future research. To preserve the flow of the thesis, we delegate various miscellaneous concepts and proofs to the appendix.
Chapter 2

Background

2.1 Efficiency

In this section, we explore the notion of efficiency in the context of rare-event probability estimation. This is important as we need a criteria to benchmark the performance of estimators. A more efficient estimator will require smaller sample size to achieve a given degree of performance or accuracy. Firstly, the concept of relative error is introduced which we extend by incorporating simulation time. This will allow us to establish and compare efficiency of estimators in typical rare-event environments.

In statistical inference, efficiency of estimators is often characterized by its variance or mean square error which are key quantities in establishing the accuracy of the estimator. However, in a rare-event context, both the magnitude of variances and estimates are typically very small. Hence, we need to define a similar measure which captures the variability relative to the magnitude of the quantity we are trying to estimate. In other words, we need to borrow the idea of relative accuracy to give an indication of the estimator’s performance relative to its magnitude. There exist several definitions in the literature [1, 16]. One measure we will be considering is the relative error defined below.

Relative Error: The relative error of an estimator ̂ℓ is defined as its standard deviation divided by the quantity of interest ℓ. To formulate this, suppose ̂ℓ is an unbiased estimator of the rare-event probability

\[ ℓ = P_f(S(X) ≥ γ) = E_f[I\{S(X) ≥ γ\}], \quad (2.1) \]

where \( f \) is a density function, \( S \) is an arbitrary real-valued function, \( X \) is a random vector and \( γ \) is a level or threshold parameter. The quantity of interest is the expectation of the event \( \{S(X) ≥ γ\} \) occurring under the
density \( f \). Thus, the relative error of the estimator \( \hat{\ell} \) is

\[
\text{RE}(\hat{\ell}) = \frac{\sqrt{\text{Var}(\hat{\ell})}}{\ell}.
\]  

(2.2)

The relative variance or squared relative error is

\[
\text{RV}(\hat{\ell}) = \frac{\text{Var}(\hat{\ell})}{\ell^2}.
\]  

(2.3)

To compute the above quantities, we may need to estimate \( \ell \) and \( \text{Var}(\hat{\ell}) \) by using realizations of the estimator. Suppose we have \( N \) independent replications, \( Z_1, \ldots, Z_N \), of some random variable \( Z \) for which \( \mathbb{E}[Z] = \ell \). Then an unbiased estimator of \( \ell \) is

\[
\hat{\ell} = \frac{1}{N} \sum_{i=1}^{N} Z_i.
\]  

(2.4)

Further, note that

\[
\text{Var}(\hat{\ell}) = \text{Var} \left( \frac{1}{N} \sum_{i=1}^{N} Z_i \right) = \frac{1}{N} \text{Var}(Z).
\]

Hence, the relative error of this estimator \( \hat{\ell} \) is

\[
\text{RE}(\hat{\ell}) = \frac{\sqrt{\text{Var}(Z)}}{\ell \sqrt{N}}.
\]  

(2.5)

This quantity can be estimated from the iid sample \( Z_1, \ldots, Z_N \) by replacing \( \ell \) by its estimator \( \hat{\ell} \) and the variance by its sample counterpart

\[
\hat{\text{Var}}(Z) = \frac{1}{N} \sum_{i=1}^{N} (Z_i - \hat{\ell})^2.
\]

Similarly, the relative variance or squared relative error is

\[
\text{RV}(\hat{\ell}) = \frac{\text{Var}(Z)}{N\ell^2} = \text{RE}(\hat{\ell})^2.
\]  

(2.6)

It is important to note that the above efficiency measures ignore the simulation time, say \( \tau \), to generate one realization of \( Z \). Two estimators may have similar relative error but significantly different computational time. We can take into account the computational time by defining the relative time variance product below.
Relative Time Variance Product (RTVP): In order to benchmark such estimators according to their efficiency, we must take into account simulation time along with its relative error. Assume each of the $N$ replications takes equal amount of time $\tau$ to simulate. Then $N\tau$ is the total time to generate $N$ replications of $Z$. Note $\tau$ is commonly measured in seconds. One approach to account for computational cost is to multiply the relative variance or squared relative error \((2.6)\) by the total computational time $N\tau$. It is given by

$$\text{RTVP}(\hat{\ell}) = N\tau \times \text{RV}(\hat{\ell}) = \frac{\tau \text{Var}(Z)}{\ell^2}. \quad (2.7)$$

Hence, the relative time variance product of $\hat{\ell}$ is independent of the number of replications $N$. It is also known as the work normalized squared relative error. This is an important measure that will be used throughout our numerical experiments to compare estimators with differing computational costs and relative errors.

2.2 Importance Sampling

In typical rare-event settings, naive Monte Carlo methods are not viable due to inefficiency. A well known example is the Crude Monte Carlo (CMC) method where high computational costs are required to achieve a reasonable degree of relative error. This is illustrated in example A.1.2 of the appendix. Efficiency, as measured by the size of RTVP, can be improved by either decreasing the simulation time to generate a replication or reducing the variance of the estimator. One popular approach for the latter is importance sampling. It is a widely used variance reduction technique in the estimation of \((1.1)\) and \((1.2)\).

Given a fixed degree of relative error, the variance reduction achieved through importance sampling is usually high enough such that its total computational effort is many orders of magnitude less than the CMC method. In many cases, the reduction in variance typically outweighs any increases in computational complexity yielding a smaller RTVP.

Importance sampling can be viewed as a special ‘acceleration’ technique that allows us to change the probability measure. In effect, we can sample from another probability measure where rare-events occur more frequently. The procedure is described in detail below.

2.2.1 Methodology

Suppose $g$ is another probability density such that $Hf$ is dominated by $g$. In other words, for any $x$ such that $g(x) = 0$ then $H(x)f(x) = 0$. We can
rewrite (1.1) as
\[ Z = \int H(x) \frac{f(x)}{g(x)} \, dx = \mathbb{E}_g \left[ H(X) \frac{f(X)}{g(X)} \right]. \]

Thus, we take the expectation with respect to \( g \) instead of \( f \) and choose \( g \) such that a reduction in variance of \( \hat{Z} \) is achieved. Consequently, if \( X_1, X_2, \ldots, X_N \) is an iid population from \( g \) then an unbiased estimator for \( Z \) is
\[ \hat{Z} = \frac{1}{N} \sum_{k=1}^{N} Z_k \quad \text{with} \quad Z_k = H(X_k) \frac{f(X_k)}{g(X_k)}. \] (2.8)

We call estimator \( \hat{Z} \) the importance sampling estimator and \( g \) the importance sampling density. The ratio of densities
\[ W(x) = \frac{f(x)}{g(x)} \]
is the likelihood ratio. If \( \hat{\sigma} \) is the sample standard deviation of \( Z_1, \ldots, Z_N \), then the relative error of \( \hat{Z} \) is
\[ RE(\hat{Z}) = \frac{\hat{\sigma}}{\sqrt{N}|\hat{Z}|}, \]
and an asymptotic normal approximation \( 1 - \alpha \) confidence interval for \( Z \) is
\[ \left( \hat{Z} - z_{1-\alpha/2} \frac{\hat{\sigma}}{\sqrt{N}}, \hat{Z} + z_{1-\alpha/2} \frac{\hat{\sigma}}{\sqrt{N}} \right), \]
where \( z_\omega \) is the \( \omega \)-quantile of the standard normal distribution.

The main difficulty in importance sampling is choosing an appropriate importance sampling density \( g \) which yields an estimator with small variance. A poor choice of \( g \) may compromise the quality of the estimate and its confidence intervals, see [2]. This motivates the need for an optimal importance sampling density which minimizes the variance of the estimator in (2.8). Such importance sampling density exists in theory and is given in the next subsection below. We also note here that a good choice of \( g \) should give the estimator (2.8) finite variance. That is,
\[ \mathbb{E}_g H^2(X) \frac{f^2(X)}{g^2(X)} = \mathbb{E}_f H^2(X) \frac{f(X)}{g(X)} < \infty. \]

If possible, \( g \) should be chosen such that it does not have lighter tails than \( f \) and the likelihood ratio \( f/g \) is bounded.

6
2.2.2 Minimum-Variance Density

The optimal importance sampling density $\pi$ is one that should minimize the variance of $\hat{Z}$. Thus, it is the solution to the functional minimization program

$$\min_g \text{Var}_g \left( H(X) \frac{f(X)}{g(X)} \right) \quad (2.9)$$

It is well known and can be shown, for example in [19], that the solution to this optimization program is the minimum-variance importance sampling density

$$\pi(x) = \frac{|H(x)| f(x)}{\int |H(x)| f(x) \, dx}. \quad (2.10)$$

In particular, if $H(x) \geq 0$ or $H(x) \leq 0$ for all $x$ then

$$\pi(x) = \frac{H(x) f(x)}{Z} \quad (2.11)$$

is the zero-variance importance sampling density. That is, our estimator $\hat{Z}$ is constant under $\pi$ and in this case

$$\text{Var}_\pi(\hat{Z}) = \text{Var}_\pi(H(X)W(X)) = 0.$$  

Example 2.2.1. In the context of rare-event estimation, we have assumed that

$$H(x) = \mathbb{I}\{S(x) \geq \gamma\}$$

which is an indicator function that maps to the set $\{0, 1\}$. Thus, the property $H(x) \geq 0$ for all $x$ is satisfied. We can write the minimum (zero) variance importance sampling pdf in the form of (2.11). That is,

$$\pi(x) = \frac{f(x) \mathbb{I}\{S(x) \geq \gamma\}}{\ell}, \quad (2.12)$$

with $\ell = \mathbb{P}_f(S(X) \geq \gamma)$.

We note that in both (2.10) and (2.11), the optimal density $\pi(x)$ depends on the unknown quantity $Z$ as

$$\int |H(x)| f(x) \, dx = Z - 2 \int_{H(x) < 0} H(x) f(x) \, dx.$$  

Hence, in practice, the evaluation of the minimum-variance importance sampling density $\pi$ is usually not possible and cannot be directly used as an importance sampling density. However, the minimum-variance density will play a significant role during the formulation of the ELM procedure in chapter 3.

A good importance sampling density $g$ should still be ‘close’ to the minimum variance density $\pi$. This motivates the concept of the Kullback-Leibler distance given below.
2.2.3 Kullback-Leibler Distance

The optimal importance sampling density $\pi$ in (2.10) or (2.11) is difficult to evaluate and cannot be used directly. Instead, consider an importance sampling density $g$ that is ‘close’ to $\pi$. The closeness between two probability density functions $f$ and $g$ is commonly measured by the Kullback-Leibler Cross-Entropy Distance which takes the form

$$d(f, g) = \mathbb{E} \left[ \log \left( \frac{f(X)}{g(X)} \right) \right] = \int f(x) \log \left( \frac{f(x)}{g(x)} \right) dx$$

$$= \int f(x) \log (f(x)) \, dx - \int f(x) \log (g(x)) \, dx.$$

(2.13)

The Kullback-Leibler distance is a special case of the $\phi$-divergence distance with $\phi(x) = -\log(x)$.

This is explained further in appendix A.3 with other special cases of the $\phi$-divergence distance.

In theory, we would like to minimize the distance between the the minimum-variance density $\pi$ and our importance sampling estimator $g$

$$d(\pi, g) = \int \pi(y) \log \left( \frac{\pi(y)}{g(y)} \right) dy.$$  

(2.14)

However, without any restrictions, the optimal solution is $g(y) = \pi(y)$ which again is not useful. Instead, restrict the search space to product form

$$g(y) = \prod_{i=1}^{d} g_i(y_i).$$

Given this condition, it can be shown (see Lemma 2.3.1) that the best importance sampling density (in cross-entropy sense) is the product of the marginal densities from the minimum variance density

$$g(y) = \prod_{i=1}^{d} \pi_i(y_i).$$

This is inspired by the mean field approximation method in physics [13] and is the backbone of the MCIS method discussed in the next section.

2.3 MCIS Estimator

In this section, we introduce an adaptive importance sampling procedure for rare-event probability estimation. The estimator combines two distinct and
widely used Monte Carlo simulation methods, Markov chain Monte Carlo (MCMC) and importance sampling (IS), into a single algorithm. This is discussed in detail in [4]. The main steps of this algorithm are summarized as follows

1. **MC Stage:** Construct the semi-parametric importance sampling density using MCMC samples from the zero-variance density (2.12).

2. **IS Stage:** Use the constructed importance sampling density to deliver an estimator for $\ell$ by (2.8).

### 2.3.1 Product of Marginals Model

Instead of minimizing the Kullback-Leibler CE distance (2.14) over a simple parametric family of densities, we minimize it over all densities of product form

$$
g(y) = \prod_{i=1}^{d} g_i(y_i), \quad g_i \in \mathcal{G} = \left\{ g : \mathbb{R} \to [0, \infty) \mid \int g(y) \, dy = 1 \right\}.\tag{2.15}
$$

That is, we solve the functional optimization program

$$
\min_{g_i \in \mathcal{G}} d(\pi, g) = \int \pi(y) \log \left( \frac{\pi(y)}{\prod_{i=1}^{d} g_i(y_i)} \right) \, dy.\tag{2.15}
$$

**Lemma 2.3.1.** Suppose that $\pi_i(y_i)$ is the marginal of the zero-variance density $\pi(y)$. Then, the solution to functional optimization program (2.15) is

$$
g_i(y_i) = \pi_i(y_i) \quad \text{for all} \quad i = 1, \ldots, d.
$$

That is, the best importance sampling density (in the cross-entropy sense) within the space of all product-form densities is the product of marginals for (2.11). This is given by

$$
g(y) = \prod_{i=1}^{d} \pi_i(y_i).\tag{2.16}
$$

**Proof.** The proof of this lemma is given in appendix B.1. □
2.3.2 MCIS Methodology

Suppose we have generated a population from the zero-variance density using the MCMC sampler. That is,
\[ \mathbf{X}_1, \ldots, \mathbf{X}_n \overset{\text{approx}}{\sim} \pi(\mathbf{x}), \quad \mathbf{X}_i = (X_{i1}, \ldots, X_{id}). \]

We would like to evaluate (2.16). Since the marginals are typically not available in closed form, we use the Markov chain output to estimate each marginal density \( \pi_i(y_i) \) by
\[
\hat{\pi}_i(y_i) = \frac{1}{n} \sum_{k=1}^{n} \pi(y_i|\mathbf{X}_{k,-i}), \tag{2.17}
\]
where
- the population \( \mathbf{X}_1, \ldots, \mathbf{X}_n \) from \( \pi \) is obtained approximately by the Gibbs Sampler;
- the vector \( \mathbf{X}_{k,-i} \) is the same vector as \( \mathbf{X}_k \) with the \( i \)-th element removed;
- \( \pi(x_i|\mathbf{X}_{k,-i}) \) is the the conditional density of \( x_i \) given all other components of \( \mathbf{X}_k \).

The estimator (2.17) is motivated by the identity
\[
\mathbb{E}_\pi[\hat{\pi}_i(y)] = \frac{1}{n} \sum_{k=1}^{n} \mathbb{E}_\pi[\pi(y|\mathbf{X}_{k,-i})] = \mathbb{E}_\pi[\pi(y|\mathbf{X}_{-i})]
\]
\[
= \mathbb{E}_\pi[\pi(y|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_d)]
\]
\[
= \int \pi(y|x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_d) \pi(\mathbf{x}) \, dx
\]
\[
= \int \frac{\pi(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_d)}{\pi(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_d)} \pi(\mathbf{x}) \, dx
\]
\[
= \int \frac{\pi(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_d)}{\pi(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_d)} \left( \int \pi(x_1, \ldots, x_d) \, dx_1 \right) \, dx_{-i}
\]
\[
= \int \pi(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_d) \, dx_{-i}
\]
\[
= \pi_i(y).
\]

We assumed that the conditional densities \( \pi(x_i|\mathbf{X}_{-i}) \) are available in closed form. Thus, we can use the Gibbs sampling algorithm (see appendix A.5.2) to obtain \( \mathbf{X}_1, \ldots, \mathbf{X}_n \overset{\text{approx}}{\sim} \pi(\mathbf{x}) \). Thus, in the MC stage we can construct the semi-parametric importance sampling density
\[
\hat{g}(y) = \prod_{i=1}^{d} \hat{\pi}_i(y_i) = \prod_{i=1}^{d} \frac{1}{n} \sum_{k=1}^{n} \pi(y_i|\mathbf{X}_{k,-i}). \tag{2.18}
\]
Generating $\mathbf{Y} \sim \hat{g}(\mathbf{y})$ is straightforward as each $Y_i$ is generated from the mixture $\tilde{\pi}_i(y_i) = \frac{1}{n} \sum_{k=1}^{n} \pi(y_i \mid X_k, -i)$ and independently from other components of $\mathbf{Y}$ given $\{X_k\}$. In the IS stage, we generate from the iid population $\mathbf{Y}_1, \ldots, \mathbf{Y}_m$ from $\hat{g}$ and then deliver the estimator

$$\hat{\ell}_{\text{MCIS}} = \frac{1}{m} \sum_{k=1}^{m} \mathbb{1}\{S(\mathbf{Y}_k) \geq \gamma\} \frac{f(\mathbf{Y}_k)}{g(\mathbf{Y}_k)}.$$  (2.19)

Note here that the MCIS estimator uses the MCMC samples from the zero-variance density indirectly by first estimating $g$ and then $\ell$. The estimator $\hat{g}$ is semi-parametric as opposed to non-parametric since $\hat{g}$ does not converge to $\pi$ as $n \uparrow \infty$ unless the true $\pi$ is in product form.

2.4 Survey of Other Monte Carlo Techniques

A survey of rare-event simulation techniques and recent advances is given in [2, 12, 15]. In this section, we give a brief overview of the other common Monte Carlo methods used in rare-event probability estimation. This includes the adaptive importance sampling algorithms, conditioning methods for heavy tails and the splitting method.

2.4.1 Adaptive importance sampling schemes

These schemes allow for the adaptive or automatic selection of the importance sampling density. One example is the MCIS procedure described previously in section 2.3. Other examples include the cross-entropy and variance minimization methods, see [14, 15]. We describe the well known cross-entropy method in more detail below.

**Parametric Cross-Entropy Method:** The key idea behind the cross-entropy method is to choose the importance sampling density $g$ in a specified parametric class of densities $\mathcal{G}$ such that the Kullback-Leibler divergence (2.13) between the optimal importance sampling density $\pi$ (2.10) and $g$ is minimal. This is described in detail in [14]. We want to find a $g \in \mathcal{G}$ that minimizes

$$d(\pi, g) = \mathbb{E}_\pi \left[ \log \frac{\pi(\mathbf{X})}{g(\mathbf{X})} \right] = \int \pi(\mathbf{y}) \log \left( \frac{\pi(\mathbf{y})}{g(\mathbf{y})} \right) d\mathbf{y}. \quad (2.20)$$

In most cases of interest the nominal probability density $f$ is parametrized by a finite-dimensional vector $\mathbf{u}$; that is, $f(\mathbf{x}; \mathbf{u})$. It is then customary to choose the importance sampling density $g$ in the same family of probability densities. In other words, we would like $g(\mathbf{x}) = f(\mathbf{x}; \mathbf{v})$ for some reference
parameter $v$. The cross-entropy minimization procedure then reduces to finding an optimal reference parameter $v^*$:

$$v^* = \arg\min_v \int \pi(x) \log \left( \frac{\pi(x)}{f(x; v)} \right) \, dx = \arg\max_v \int \pi(x) \log f(x; v) \, dx. \quad (2.21)$$

In practice, the integral in (2.21) is estimated from preliminary simulations so that we can obtain the estimator $\hat{v}^*$

$$\hat{v}^* = \arg\max_v \sum_{i=1}^{n} \log(f(X_i, v)),$$

where $X_1, \ldots, X_n$ is an approximate sample from the zero-variance density $\pi$ (2.12) obtained using MCMC sampling. For example, we can use the Gibbs Sampler; see algorithm A.5.2 from the appendix. Hence, we can deliver the cross-entropy estimator of (1.2) as

$$\hat{\ell}_{CE} = \frac{1}{m} \sum_{k=1}^{m} \mathbb{I}\{S(Y_k) \geq \gamma\} \frac{f(Y_k; u)}{f(Y_k; \hat{v}^*)}, \quad Y_1, \ldots, Y_m \text{ iid } f(y; \hat{v}^*). \quad (2.23)$$

2.4.2 Conditioning methods for heavy tails

These are specialized and highly efficient algorithms designed for the estimation of probabilities arising from heavy-tailed random variables. For characteristics of heavy-tailed distributions, see appendix A.2. We focus on the Asmussen-Kroese conditional estimator [15, §10.3] below. This is one of the estimators that will be used during the numerical benchmark problem in section 4.1.

Asmussen-Kroese (AK) Conditional Estimator: This is an efficient estimator of rare-event probabilities that performs well for distributions with subexponential properties. It caters for estimation problems of the form

$$\ell(\gamma) = \mathbb{P}(S_d(X) \geq \gamma) = \mathbb{P}(X_1 + \cdots + X_d \geq \gamma),$$

where the $\{X_i\}$ are iid with cdf $F$ from the subexponential class of distributions. We are interested in the case where the threshold $\gamma$ is large, making probability $\ell(\gamma)$ small. This estimator is based on a conditioning idea which exploits the subexponential property

$$\lim_{\gamma \to \infty} \frac{\mathbb{P}(S_d \geq \gamma)}{d \mathbb{P}(X_1 \geq \gamma)} = 1.$$

Typically, with the subexponential property, the rare-event primarily occurs due to a single variable exceeding the threshold value. This is in contrast
with the light-tailed case where the rare-event occurs usually when the majority of variables take on a large value. The algorithm given below outputs the estimator and is based on the identity

\[ \ell(\gamma) = d \mathbb{P}(S_d(X) \geq \gamma, X_d = \max_j X_j) = d \mathbb{E} \left[ \tilde{F} \left( \left( \gamma - \sum_{j=1}^{d-1} X_j \right) \vee \max_{j \neq d} X_j \right) \right], \]

where \( \tilde{F}(x) = 1 - F(x) \) and \( a \vee b = \max\{a, b\} \). This is explained in more detail in [3].

Algorithm 2.4.1. (AK Conditional Estimator for \( \mathbb{P}(S_d(X) \geq \gamma) \))

1. Generate \( X_1, \ldots, X_n \sim \mathcal{F} \).

2. Compute

\[ Y = d \tilde{F} \left( \left( \gamma - \sum_{j=1}^{d-1} X_j \right) \vee \max_{j < d} X_j \right). \]

3. Using \( N \) independent replications of \( Y \), deliver the unbiased estimator

\[ \hat{\ell}(\gamma) = \frac{1}{N} \sum_{k=1}^{N} Y_k. \quad (2.24) \]

Hence, the AK conditional estimator based on one replication is given by

\[ \hat{\ell}_{AK} = d \tilde{F} \left( \left( \gamma - \sum_{j=1}^{d-1} X_j \right) \vee \max_{j < d} X_j \right), \quad X_1, \ldots, X_{d-1} \sim \mathcal{F}. \]

We are interested in the Weibull distribution with scale parameter 1 and shape parameter \( \alpha < 1 \). This will be used in the numerical example later in section 4.1. In this case, the estimator (2.24) has vanishing relative error (see appendix A.1 for definition of vanishing relative error) when \( 0 < \alpha < \log(3/2)/\log(3) \approx 0.369 \). For more detail, see [15, §10.3].

2.4.3 Splitting

The splitting method, sometimes recognized as Sequential Monte Carlo (SMC), is another simulation technique for the estimation of rare-event probabilities [7, 9]. The main objective is to generate more occurrences of the rare-event by splitting the sample paths of (a possibly artificially induced) Markov chain into multiple copies at various stages of the simulation. This method uses a decomposition of the state space into nested subsets so that the rare event is represented as the intersection of a nested sequence of events. The probability of the rare event is the product of conditional
probabilities, each of which can be estimated much more accurately than the rare-event probability itself. These methods are less effective than the conditioning methods used in section 2.4.2 so we do not discuss them further. We note, however, that it may be possible to use the splitting method to generate approximate samples from $\pi$ as an alternative to MCMC.
Chapter 3

Theory of Empirical Likelihood Maximization

In the MCIS algorithm, we have used the MCMC samples from the zero-variance density $\pi$ in (2.12) indirectly to first estimate the importance sampling density $g$ and then the (rare-event) probability $\ell$. While the estimator (2.19) works well on a number of prototypical problems, we strive to do better by using the MCMC samples directly. This motivates a novel Monte Carlo technique – *Empirical Likelihood Maximization (ELM)*. It is a versatile approach to rare-event probability estimation where we sample from a sequence of densities and maximize an empirical likelihood. This method uses the MCMC samples directly. It is inspired by Bayesian methodology, where the objective is to estimate normalizing constants of posterior densities via MCMC sampling.

We begin by drawing some analogies to the maximum likelihood estimation method used ubiquitously in statistics. The aim is to explain and motivate the key mechanism underlying the ELM approach with the familiar maximum likelihood principle.

3.1 Method of Maximum Likelihood

Suppose we are given iid observed vectors $x_1, \ldots, x_n$ which we believe come from an unknown probability density function $g_0(\cdot)$. However, we also believe that $g_0(\cdot)$ belongs to a certain parametric family of distributions $\{g(\cdot|\theta) : \theta \in \Theta\}$ such that

$$g_0(\cdot) = g_0(\cdot|\theta_0),$$

for some unknown value $\theta_0$. We wish to estimate the true unknown $\theta_0$. A desirable estimator $\hat{\theta}$ is one that is close in some sense to the true value $\theta_0$. One way to obtain such an estimator is to use the maximum likelihood method.
To proceed, we specify the joint probability density function for the observed vectors. For an iid sample, this is given by

\[ g(x_1, \ldots, x_n | \theta) = \prod_{i=1}^{n} g(x_i | \theta). \]

Consider a different perspective where we hold the observed values \( x_1, \ldots, x_n \) fixed, and allow \( \theta \) to vary. This is known as the likelihood function and is given by

\[ L(\theta | x_1, \ldots, x_n) = \prod_{i=1}^{n} g(x_i; \theta). \]

The maximum likelihood method searches the parameter space \( \{ \theta \in \Theta \} \) to find a value \( \hat{\theta} \) that maximizes \( L(\theta | x_1, \ldots, x_n) \). We refer to this value as the maximum likelihood estimator (MLE) of \( \theta \) if it exists. In other words, we need to solve the following optimization program to obtain the MLE

\[ \hat{\theta} = \arg\max_{\theta} L(\theta | x_1, \ldots, x_n). \]

It is often more convenient to work with the logarithm of the likelihood function; referred to as the log-likelihood. Since logarithm is a monotonically increasing function, the same MLE \( \hat{\theta} \) can also be obtained by maximizing the log-likelihood function

\[ \hat{\theta} = \arg\max_{\theta} \sum_{i=1}^{n} \log g(x_i; \theta). \]

The key statistical properties of the MLE are as follows.

- **Consistency** – The MLE \( \hat{\theta} \) is consistent. That is, the estimator \( \hat{\theta} \) converges in probability to its true value as \( n \to \infty \)

\[ \hat{\theta} \xrightarrow{p} \theta_0. \]

- **Asymptotic Normality** – Under some regularity conditions, the MLE \( \hat{\theta} \) converges in distribution to a normally distributed random vector as \( n \to \infty \)

\[ \sqrt{n}(\hat{\theta} - \theta_0) \overset{d}{\to} N(0, I^{-1}(\theta_0)), \quad (3.1) \]

where \( I^{-1}(\theta_0) \) is the inverse of the Fisher information matrix.

- **Efficiency** – The MLE \( \hat{\theta} \) is asymptotically efficient as it attains the Cramér-Rao lower-bound

\[ \lim_{n \to \infty} \text{Var}(\hat{\theta}) I(\theta) = 1. \]
Motivating the ELM approach: In the maximum likelihood estimation method, we found an estimator of the unknown parameter $\theta$ by maximizing the likelihood function. The ELM procedure shares the same fundamental idea such that we maximize an (empirical) likelihood to obtain an estimator of the unknown (rare-event) probabilities of interest. One key difference is, instead of using a likelihood, we use an empirical likelihood constructed from simulated data from known distributions or distributions known up to a multiplicative constant.

In the ELM method, we will use a sequence of probability densities with known or unknown normalizing constants. An important step is to embed the normalizing constants as parameters in the empirical likelihood. These normalizing constants are treated as unknown parameters that are to be estimated. Hence, the empirical likelihood will be a function of embedded normalizing constants that do not affect the simulated observations. Maximizing the empirical likelihood or empirical log-likelihood with respect to such parameters will allow us to obtain an estimator of the (rare-event) probability. That is, we want to solve the following optimization program

$$\hat{\ell} = \operatorname*{argmax}_\ell \left\{ -\hat{D}(\ell; x) \right\},$$

where $-\hat{D}(\ell; x)$ is an empirical log-likelihood function and $\ell$ is as vector of embedded parameters containing the quantity of interest. Using knowledge of one known normalizing constant, we can retrieve estimates for the remaining parameters. The resulting estimators would inherit consistency and asymptotic normality properties of the MLE. More concrete and detailed mathematical explanations of the ELM procedure is given below.

3.2 ELM Formulation

Suppose we have a sequence of densities

$$f_t(x) = \frac{w_t(x)}{\ell_t} = \frac{f(x)H_t(x)}{\ell_t}, \quad t = 1, \ldots, s, \quad (3.2)$$

where $f$ is a known density, $\{H_t\}$ are known functions and $\ell_t$ are normalizing constants to $\{w_t\}$. A density $f_t$ in the given sequence $(3.2)$ is a reference density if it has a known normalizing constant that can be calculated analytically. We assume the existence of at least one reference density $f_t$, say $f_1$, whose corresponding normalizing constant $\ell_1$ is known.

Now, we embed the rare-event probability of interest, say $\ell_s$, into this sequence. This can be achieved by using the zero-variance density $(2.12)$

$$f_s(x) = \frac{f(x)\mathbb{I}\{S(x) \geq \gamma\}}{\ell_s}.$$
Hence, the problem of interest is to estimate the normalizing constant
\[ \ell_s = \mathbb{E}_f[H_s(X)] = P_f(S(X) \geq \gamma). \]
To proceed, we need to introduce a key condition required by the supports of the given sequence of densities.

**Connectedness:** Suppose we are given a graph with \(s\) nodes. Let there be an edge between two distinct nodes \(i\) and \(j\) if and only if
\[ \mathbb{E}_f[\mathbb{I}\{H_i(X) > 0\} \times \mathbb{I}\{H_j(X) > 0\}] > 0. \]  
(3.3)
If there exists a path between any two nodes \(i\) and \(j\), then the condition (3.3) is known as Vardi’s connectivity condition [8, 20] on the supports of \(\{f_t\}\). This connectivity condition is needed for justifying the results presented in the subsequent sections.

**Mixture Model:** Now, assume that iid samples from the sequence of densities \(\{f_t\}\) have been simulated exactly, each with corresponding fixed sample size \(n_t\). In other words, we have generated
\[ X_{t,1}, \ldots, X_{t,n_t} \overset{iid}{\sim} f_t(x) \quad \text{for all} \quad t = 1, \ldots, s. \]
Conceptually, this is similar to sampling
\[ n = n_1 + \cdots + n_s \]
random variables with stratification from the mixture
\[ f(x) = \frac{1}{n} \sum_{t=1}^{s} n_t f_t(x) = \sum_{t=1}^{s} \lambda_t f_t(x), \quad \lambda_t \overset{def}{=} \frac{n_t}{n}. \]
Note \(\lambda_t\) is fixed for each \(t = 1, \ldots, s\) and is defined as the proportion of samples obtained from density \(t\). Now, to group the observations together, let the pooled sample be
\[ X_1, \ldots, X_n \]
where the first \(n_1\) samples are realizations from \(f_1\), the next \(n_2\) are samples from \(f_2\) and so on. In practice, sampling may need to be done approximately (e.g. via MCMC, see appendix A.5) if exact sampling methods (see appendix A.4) are not viable. In the following step of the ELM procedure, we will use the samples to construct the empirical likelihood.
Empirical Likelihood: To proceed, we define the vector of parameters $z = (z_1, \ldots, z_n) = \left(-\log \left(\frac{\ell_1}{\lambda_1}\right), -\log \left(\frac{\ell_2}{\lambda_2}\right), \ldots, -\log \left(\frac{\ell_s}{\lambda_s}\right)\right) = -\log \left(\frac{\ell}{\lambda}\right)$.

Now, consider the empirical likelihood of the observed simulated data $X_{k,1}, \ldots, X_{k,n_k}$, $k = 1, \ldots, s$, as a function of the parameter vector $z$

$$
\hat{L}(z) = \prod_{k=1}^s \prod_{j=1}^{n_k} f_k(X_{k,j})
$$

$$
= \prod_{k=1}^s \prod_{j=1}^{n_k} \frac{w_k(X_{k,j})}{\lambda_k e^{-z_k}}, \quad \text{(as } \ell_k = \lambda_k e^{-z_k})
$$

$$
= \prod_{k=1}^s \prod_{j=1}^{n_k} \frac{w_k(X_{k,j})}{\lambda_k e^{-z_k} \bar{f}(X_{k,j})} \prod_{k=1}^s \prod_{j=1}^{n_k} \bar{f}(X_{k,j}). \quad (3.4)
$$

Further, the partial empirical log-likelihood (by [8]) is

$$
\hat{l}(z) = \log \left(\prod_{k=1}^s \prod_{j=1}^{n_k} \frac{w_k(X_{k,j})}{\lambda_k e^{-z_k} \bar{f}(X_{k,j})}\right)
$$

$$
= \sum_{k=1}^s \sum_{j=1}^{n_k} \left( \log w_k(X_{k,j}) + z_k - \log \lambda_k - \log \bar{f}(X_{k,j}) \right)
$$

$$
= -\sum_{j=1}^n \log \bar{f}(X_j) + \sum_{k=1}^s n_k z_k + C
$$

$$
= -\sum_{j=1}^n \log \left(\sum_{k=1}^s w_k(X_j)e^{z_k}\right) + \sum_{k=1}^s n_k z_k + C, \quad (3.5)
$$

where

$$
C = \sum_{k=1}^s \sum_{j=1}^{n_k} \log w_k(X_{k,j}) + \sum_{k=1}^s n_k \log \lambda_k
$$

is a constant independent of the parameters $z$. Define

$$
\hat{D}(z) \overset{\text{def}}{=} \sum_{j=1}^n \log \left(\sum_{k=1}^s w_k(X_j)e^{z_k}\right) - \sum_{k=1}^s n_k z_k, \quad (3.6)
$$

which is a convex function of $z$. This will be our objective function in the optimization program formulated below.
Optimization: We observe that maximizing the partial log-likelihood in (3.5) is equivalent to minimizing (3.6). Under Vardi’s connectivity condition (3.3) and assumption of an iid sample, it is shown [8, 20] that the maximum of this partial empirical log-likelihood is same as the maximum of the complete empirical likelihood (3.4).

Also, under the same conditions, it is shown [8, 20] that the unique nonparametric likelihood estimate of \( \hat{z} \) (and hence \( \hat{\ell} \)) is the solution to the convex optimization program

\[
\hat{z} = \arg\min_z \hat{D}(z),
\]

with \( \ell_1 \) known and \( z_1 \) fixed. This restriction is required to obtain an unique and optimal solution \( \hat{z} \) from the parameter space. We will see later, in section 3.4, that such restriction can be replaced by adding in a homogeneous constraint in the optimization.

Solving the unconstrained program (3.7) will allow us to obtain an optimal solution \( \hat{z} \) and hence \( \hat{\ell} \) up to a multiplicative constant. Having known \( \ell_1 \), we are able to recover the remaining \( \hat{\ell} \) and hence \( \hat{\ell}_s \). The idea is similar to recovering an unique eigenvector by knowing its first component (an eigenvector corresponding to an eigenvalue of multiplicity unity is unique up to a multiplicative constant). One way to execute this convex optimization is to compute the gradient of the objective function and equate it to zero. This leads to the moment-matching equations described in detail below.

3.3 Moment-Matching Equations

Without loss of generality, assume the known normalizing constant \( \ell_1 \) of the reference density \( f_1 \) is equal to unity. Then, the gradient is a \((s-1) \times 1\) vector with components

\[
|\nabla \hat{D}|_{\ell}(z) = \sum_{j=1}^{n} \frac{w_t(X_j)e^{\hat{z}_t}}{\sum_{k=1}^{s} w_k(X_j)e^{\hat{z}_k}} - n_t, \quad \text{for} \ t = 2, \ldots, s. \tag{3.8}
\]

Hence, we equate these to zero and solve the \( s - 1 \) dimensional system for the unknowns \( z_2, \ldots, z_s \)

\[
|\nabla \hat{D}|_{\ell}(\log(\lambda_1), z_2, \ldots, z_s) = 0, \quad \text{for} \ t = 2, \ldots, s.
\]

In other words, solve the following nonlinear system for \( \hat{z}_2, \ldots, \hat{z}_s \)

\[
\frac{1}{n} \sum_{j=1}^{n} \frac{w_t(X_j)e^{\hat{z}_t}}{\sum_{k=1}^{s} w_k(X_j)e^{\hat{z}_k}} = \lambda_t, \quad \text{for} \ t = 2, \ldots, s. \tag{3.9}
\]

The solution of these equations will include \( \hat{z}_s \) and hence this will give us an estimator of the rare-event probability of interest \( \hat{\ell}_s \). Alternatively, the
nonlinear system $[\nabla \hat{D}]_t(\hat{z}) = 0$ for $t = 2, \ldots, s$ can be written in terms of the vector $\ell$

$$\hat{\ell}_t = \frac{1}{n} \sum_{j=1}^{n} \frac{w_t(X_j)}{\sum_{k=1}^{s} \lambda_k f_k(X_j)} = \frac{1}{n} \sum_{j=1}^{n} \frac{\lambda_t w_t(X_j)}{\sum_{k=1}^{s} \lambda_k f_k(X_j)}, \quad t = 2, \ldots, s. \tag{3.10}$$

This resembles the generalized method of moments (GMM) (see [10]) for estimating parameters in statistical models. A brief description is given below.

**Generalized Method of Moments:** Suppose we have $n$ iid observations $X_1, \ldots, X_n$ where each observation is an $d$-dimensional multivariate random variable. We assume the data comes from a certain statistical model with an unknown parameter $\theta \in \Theta$. The aim of the estimation problem is to find the ‘true’ value of this parameter, $\theta_0$. In order to apply GMM there should exist a vector-valued function $g(X, \theta)$ such that

$$m(\theta_0) \overset{\text{def}}{=} \mathbb{E}_{\theta_0}[g(X, \theta_0)] = 0.$$

The central idea behind GMM is to replace the theoretical expected value with its empirical first moment (the sample average)

$$\hat{m}(\theta) = \frac{1}{n} \sum_{i=1}^{n} g(X_i; \theta),$$

and then to minimize the norm of this expression with respect to $\theta$. Similarly, the identity [3.10] can be rewritten such that it resembles the matching of $s$ empirical first moments. In other words, we solve the following system of non-linear equations

$$\hat{m}_{t-1}(\ell) \overset{\text{def}}{=} \frac{1}{n_t} \sum_{j=1}^{n} \frac{\lambda_t w_t(X_j)}{\sum_{k=1}^{s} \lambda_k f_k(X_j)} - \ell_t = 0, \quad t = 2, \ldots, s.$$

However, instead of minimizing the norm, this nonlinear system can be solved exactly by using Jacobi or Gauss-Seidel iterations given below in algorithm 3.3.1. Executing this will allow us to obtain $\ell$ up to a multiplicative constant. In effect, using the knowledge of $\ell_1$, we can determine the remaining $\ell_t$ for $t = 2, \ldots, s$. This process is similar to finding eigenvalues via power iterations.

**Algorithm 3.3.1.** (Jacobi Iterations)

Require: Matrix $A$ and initial starting point $\ell = (\ell_1, \ldots, \ell_s) = (1, \ldots, 1)$

Set $\epsilon = \infty$ and $\ell^* = \ell$

While $\epsilon > \text{tol}$ do

  For $i = 2, \ldots, s$ do
\begin{align*}
\ell_i &= \sum_{j=1}^{n} \frac{A_{i,j}}{\sum_{k=1}^{s} A_{k,j} n_k / \ell_i}, \quad A_{i,j} \overset{def}{=} H_i(X_j).
\end{align*}

End do
\[
\epsilon = \max_i \frac{\ell_i - \ell^*}{\ell_i},
\]
Set \( \ell^* = \ell \).

End do

Return the vector of estimated probabilities \( \hat{\ell} = \ell \).

Although the Jacobi algorithm above works satisfactorily, it sometimes takes many iterations to converge. This is why, in subsequent sections, we offer an faster alternative to minimizing \( \hat{D} \) — direct optimization of \( \hat{D} \) using MATLAB’s \texttt{fmincon} routine.

### 3.4 Constrained Maximum Empirical Likelihood

Instead of solving the moment-matching equations to obtain \( \hat{\ell} \), it is sometimes easier to use an optimization routine to minimize \( \hat{D} \) directly. In effect, we are able to add in constraints along with the optimization. This is a powerful approach to incorporate any analytical knowledge for the given sequence of densities \( \{ f_t, t = 1, \ldots, s \} \) which is one of the key strengths of the ELM method. We begin by explaining how equality constraints can be added, followed by inequality constraints.

#### Equality Constraints:
Recall that we required the first component of \( z, z_1 \), to be fixed in the optimization program (3.7). This is needed to find an unique solution \( \hat{z} \) in the parameter space. Alternatively, we can add in a homogeneous constraint to alleviate this restriction

\[ \lambda_1 z_1 + \lambda_2 z_2 + \cdots + \lambda_s z_s = 0. \] (3.11)

This linear constraint is homogeneous in the sense that the linear combination of \( z_1, \ldots, z_s \) is equal to zero. Solving the optimization problem using only this homogeneous constraint is equivalent to solving the Jacobi iterations in 3.3.1.

We can also add in an inhomogeneous linear constraint where the linear combination of \( z_1, \ldots, z_s \) does not need to be zero. This will allow us to incorporate any analytical knowledge of reference densities with known normalizing constants. For example, suppose that two of the normalizing constraints, say \( \ell_1 \) and \( \ell_2 \), are known analytically. Then we can add in the following equality constraint

\[ z_1 - z_2 = \log(\lambda_1) - \log(\lambda_2) + \log(\ell_2) - \log(\ell_1). \]

This will allow us to use \( \ell_1 \) and \( \ell_2 \) to reduce variance and improve the quality of the estimator. Hence, we can solve the following constrained optimization
to estimate the remaining $\ell_t, t = 3, \ldots, s$

$$\min_{\mathbf{z}} \hat{D}(\mathbf{z})$$
subject to:
$$\lambda_1 z_1 + \cdots + \lambda_s z_s = 0,$$
$$z_1 - z_2 = \log(\lambda_1) - \log(\lambda_2) + \log(\ell_2) - \log(\ell_1).$$

Solving the constrained program (3.12) will allow us to obtain an optimal solution $\mathbf{z}$ and hence $\hat{\ell}$ up to a multiplicative constant. Using the analytical knowledge of known $\ell_1$ or $\ell_2$, we are able to recover the remaining $\ell$ and hence $\hat{\ell}_s$. It is also easy to extend this idea by adding in more equality constraints which would allow for additional idiosyncratic knowledge of relationships amongst reference densities.

**Inequality Constraints:** Similar to the equality constraints, it is also simple to add inequality constraints. This will allow us to include any analytical knowledge of lower or upper bounds to the probabilities that are to be estimated. For example, suppose one of the reference densities, say $f_1$, has a normalizing constant $\ell_1$ which serves as a lower bound to $\ell_s$. That is, $\ell_1 \leq \ell_s$.

To incorporate this, we just need to add the inequality constraint

$$-z_1 + z_2 \leq -\log(\lambda_1) + \log(\lambda_2).$$

Hence, along with the homogeneous constraint (3.11), our constrained optimization program becomes

$$\min_{\mathbf{z}} \hat{D}(\mathbf{z})$$
subject to:
$$\lambda_1 z_1 + \cdots + \lambda_s z_s = 0,$$
$$-z_1 + z_2 \leq -\log(\lambda_1) + \log(\lambda_2).$$

Again, solving this constrained program delivers an optimal solution $\mathbf{z}$ and hence $\hat{\ell}$ up to a multiplicative constant. Hence, we are able to recover $\hat{\ell}_s$ by using analytical knowledge of known components of $\ell$.

### 3.4.1 Standard Errors and Asymptotic distribution

It is possible to derive a central limit theorem for the estimator $\hat{\mathbf{z}}$ (which solves the constrained (empirical) likelihood problem (3.12)), similar to the classical likelihood one given in (3.1). This will allow us to derive asymptotic standard errors for $\hat{\mathbf{z}}$ using the corresponding Fisher information matrix.
However, such limiting results will rely on the iid assumption on the sample $X_1, \ldots, X_n$ and we will be using an approximate dependent sample from MCMC. Of course, it is possible to construct central limit results in the MCMC case involving estimation of an autocorrelation function, but this is beyond the scope of an Honours thesis. For this reason and to keep things simple, we leave such derivations as future research. In this thesis, we will estimate standard errors from repeated simulation runs of the ELM algorithm providing us with a sequence of $\hat{z}_1, \ldots, \hat{z}_K$. 
Chapter 4

Application to Rare-event Probability Estimation

In this chapter, we illustrate the effectiveness and versatility of the ELM approach by focusing on a numerical example: sum of heavy-tailed iid Weibull random variables. Computationally, this is of interest because there exist algorithms that efficiently estimate \( \ell \) for certain values of \( \alpha \) but not its entire range. For example, the Asmussen-Kroese conditional estimator described in section 2.4 has vanishing relative error property when
\[
0 < \alpha < \frac{\log(3/2)}{\log(3)} \approx 0.369,
\]
see appendix A.1. It is known that this estimator is less efficient and can fail when \( \alpha \) falls outside the given range. We will demonstrate that the ELM estimation method performs well for all values of \( \alpha < 1 \). This is a numerical benchmark problem from [3] and is stated below.

4.1 Sum of heavy-tailed iid Weibull random variables

Suppose that \( X = (X_1,\ldots,X_d) \) is a \( d \)-dimensional independent and identically distributed random vector where each component is from the \( \text{Weib}(\alpha,1) \) distribution for some fixed shape parameter \( \alpha \). Denote the joint \( \text{Weib}(\alpha,1) \) distribution by
\[
f(x) = \prod_{i=1}^{d} \alpha x_i^{\alpha-1}e^{-x_i^{\alpha}}, \quad x_i \geq 0 \quad \forall i \in \{1,\ldots,d\}.
\]

Each random component from this distribution is heavy-tailed when \( \alpha < 1 \). It is of interest to estimate the (rare-event) probability
\[
\ell = \mathbb{E}_f(\|S(X) \geq \gamma\|) = \mathbb{P}(S(X) \geq \gamma),
\]
with
\[ S(X) = \sum_{i=1}^{d} X_i. \]

For example, this may represent the probability of default for an insurance company where \( S(X) \) is the sum of all claims received and \( \gamma \) is their current level of capital.

We consider two different ELM schemes. The first ELM scheme given in \[4.2\] below uses 4 densities and incorporates equality constraints. The densities included are ones which we believe work best in obtaining an efficient ELM estimator. Further, the second ELM given in \[4.4.2\] uses 2 densities and incorporates an inequality constraint. We will present a novel approach in obtaining a lower bound to our (rare-event) probability of interest. This is the underlying mechanism behind the second ELM scheme allowing us to utilize an inequality constraint.

### 4.2 ELM Scheme A: 4 densities with equality constraints

We apply the procedure as described in section \[3.2\] and \[3.4\] with \( s = 4 \) densities.

#### 4.2.1 ELM Densities

The four ELM densities, to be introduced, are key ingredients in obtaining an efficient ELM estimator \( \hat{\ell}_s \). Firstly, we embed the (rare-event) probability of interest \( \ell_s \) into \( f_s \) by using the zero-variance density from \[2.12\]. To motivate the inclusion of the other densities, consider rewriting the (rare-event) probability as

\[ \ell_s = P(S(X) \geq \gamma) = P(M(X) \geq \gamma) + P(S(X) \geq \gamma, M(X) < \gamma), \]

where
\[ M(X) = \max_{i \in \{1, \ldots, d\}} X_i = X_{[d]}. \]

This separates the (rare-event) probability into its main ‘heavy-tailed’ component of \( \ell_s \)

\[ P(M(X) \geq \gamma), \]

which represents the probability that the largest element of the random vector exceeds the \( \gamma \) threshold, and its residual component of \( \ell_s \)

\[ P(S(X) \geq \gamma, M(X) < \gamma) \]
which represents the probability that the sum of the elements exceeds the
$\gamma$ threshold but its largest component $X_{[d]}$ will not. Generally, for heavy-
tailed distributions, the magnitude of the main components greatly exceeds
the residual component. That is,
$$
\lim_{\gamma \to \infty} \frac{P(S(X) \geq \gamma, M(X) < \gamma)}{P(M(X) \geq \gamma)} \to 0.
$$
This is particularly the case when $\alpha$ is close to zero for this $\text{Weib}(\alpha, 1)$ ex-
ample. However, the residual component becomes more significant when
the distribution becomes less heavy-tailed (i.e. when $\alpha$ is close to 1). The
Asmussen-Kroese conditional estimator given in section 2.4 captures the
main ‘heavy-tailed’ contribution very well but ignores the residual compo-
nent.

The first two densities are reference densities such that their normalizing
constants $\ell_1$ and $\ell_2$ are known analytically. We can use such information as
a correction factor during the constrained empirical likelihood optimization.
The normalizing constant of the third density $\ell_3$ is unknown but the samples
attained allow us to capture the residual component of the probability of
interest. More detail of these four densities are given below.

1. Reference density: Main ‘heavy-tailed’ component (sum of the indica-
tors)
$$
f_1(x) = \frac{f(x) \sum_{i=1}^{d} I \{x_i \geq \gamma\}}{\ell_1} = \frac{w_1(x)}{\ell_1},
$$
where
$$
\ell_1 = \sum_{i=1}^{d} P(X_i \geq \gamma) = d \cdot P(X \geq \gamma) = d(1 - F_X(\gamma)) = de^{-\gamma^\alpha},
$$
with $X \sim \text{Weib}(\alpha, 1)$. This density is introduced to capture the main
contribution of $\ell_s$. It has normalizing constants that represent the sum
of the probabilities for each component exceeding the $\gamma$ threshold.

2. Reference density: Product of marginals from the zero-variance den-
sity (2.12)
$$
f_2(x) = \prod_{i=1}^{d} \hat{\pi}(x_i) = \frac{w_2(x)}{\ell_2},
$$
where $\hat{\pi}(x_i)$ are the marginal densities of $f_s$. Note $\ell_2 = 1$. This
is the importance sampling density used in the MCIS method (see
section 2.3).

3. Density: Residual component
$$
f_3(x) = \frac{f(x) I \{S(X) \geq \gamma, M(X) < \gamma\}}{\ell_3} = \frac{w_3(x)}{\ell_3},
$$
where
\[ M(x) = \max_{i \in \{1, \ldots, d\}} x_i = x_{[d]}. \]

This density captures the residual component of \( \ell_s \). For each sample obtained from \( f_3 \), the sum of the elements will exceed the \( \gamma \) threshold but its largest component \( X_{[d]} \) will not. Here, the normalizing constant \( \ell_3 \) is unknown and represents the residual (rare-event) probability.

4. Target density: minimum (zero) variance density from (2.12), \( s = 4 \),
\[ f_s(x) = \frac{f(x)I\{S(x) \geq \gamma\}}{\ell_s} = \frac{w_s(x)}{\ell_s}, \]
where
\[ \ell_s = \mathbb{E}_f[\mathbb{I}(S(X) \geq \gamma)] = \mathbb{P}(S(X) \geq \gamma) \]
is unknown and represents our (rare-event) probability of interest.

### 4.2.2 Sampling

In this subsection, we describe the sampling procedures used to generate random realizations from each of our four densities. Denote \( n_t \) to be the corresponding sample sizes for each density \( t = 1, \ldots, 4 \) respectively and define our pooled sample to be \( X_1, \ldots, X_n \) where \( n = n_1 + \cdots + n_4 \).

**Sampling from \( f_1 \):** Sampling iid copies from the reference density \( f_1 \) is easy by using its mixture representation
\[ f_1(x) = \sum_{j=1}^d \frac{\mathbb{P}(X_j \geq \gamma) f(x)I\{x_j \geq \gamma\}}{\ell_1}, \]
where we select component \( J \) with probability \( \mathbb{P}(J = j) = \mathbb{P}(X_j \geq \gamma)/\ell_1 \).
Given \( J = j \), we sample from the truncated joint pdf
\[ \frac{f(x|x_j)f(x_j)I\{x_j \geq \gamma\}}{\mathbb{P}(X_j \geq \gamma)}, \]
where each \( X_i \) for \( i \neq j \) has iid \text{Weib}(\alpha, 1) and \( X_j \) has truncated pdf
\[ \frac{f(x_j)I\{x_j \geq \gamma\}}{\mathbb{P}(X_j \geq \gamma)}, \quad 0 < \gamma \leq x_j < \infty. \]

The algorithm below describes how this can be implemented.

**Algorithm 4.2.1.**

1. Select \( j \in \{1, \ldots, d\} \) with equal probability.
2. Generate $U \sim \text{Uniform}(0, 1)$.

3. Set $X_j := (\gamma - \ln U)^{1/\alpha}$ as our sample from truncated Weibull.

4. Set $X_i := (-\ln U)^{1/\alpha}$ for all $i \neq j$ as our sample from Weibull.

5. Return $X = (X_1, \ldots, X_d)$.

Here, we obtain $n_1$ replications $X_1, \ldots, X_{n_1}$ from $f_1$.

**Sampling from $f_2$:** As $f_2$ is a product of independent marginals, we can sample each element of $X$ independently. Note that $\hat{\pi}_i(x_i)$ can be interpreted as a mixture of conditional densities using samples $X_j$ from $f_s$. Hence, it suffices to sample from one of $f_s(x_i|X_{j,-i}), j = 1, \ldots, n_s$ with equal probability. Each element of our sample will be from marginal or truncated Weib($\alpha, 1$). To attain a sample $X = (X_1, \ldots, X_d)$, we utilize the following algorithm.

**Algorithm 4.2.2.**
For all $i = 1, \ldots, d$, perform the following

1. Generate $j \in \{1, \ldots, n_s\}$ with equal probability.

2. To sample from $f_s(x_i|X_{j,-i})$, compute $c_{ji} = \sum_{k \neq i} X_{jk}$.

3. Generate $U \sim \text{Uniform}(0, 1)$.

4. Set
   
   $x_i := \begin{cases} 
   (-\log U)^{1/\alpha} & \text{if } c_{ji} > \gamma, \\
   [(\gamma - c_{ji})^\alpha - \log U]^{1/\alpha} & \text{if } c_{ji} \leq \gamma,
   \end{cases}$

   where $(y)_+ = \max\{0, y\}$.

Here, we generate $n_2$ samples $X_1, \ldots, X_{n_2}$ from $f_2$.

**Sampling from $f_3$:** We apply the Gibbs Sampler, see algorithm [A.5.2]
Let $X_{-i} = (X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, x_d)$ and $a_+ := \max\{0, a\}$. The conditional densities for all components $i = 1, \ldots, d$ are truncated Weib($\alpha, 1$) given by

$$f_3(x_i|X_{-i}) \propto f(x_i), \quad \left( \gamma - \sum_{j \neq i} X_j \right)_+ \leq x_i < \gamma,$$

where $f(x_i)$ is the Weib($\alpha, 1$) density. To sample from the conditional densities, we can use the inverse-transform method for truncated distributions
For these truncated Weibull, our samples are obtained by setting for all $i = 1, \ldots, d$

$$X_i = [- \log (e^{-a} - U(e^{-a} - e^{-\gamma}))]^{\frac{1}{\alpha}},$$

where

$$a = \left( \gamma - \sum_{j \neq i} X_j \right)_+$$

and

$$U \sim \text{Uniform}(0, 1).$$

Here, we generate $n_3$ samples $X_1, \ldots, X_{n_3}$ from $f_3$.

**Sampling from $f_s$:** This is a special case of the above, where the conditional densities for all $i = 1, \ldots, d$ are truncated $\text{Weib} (\alpha, 1)$ of the form

$$f_s(x_i | X_{-i}) \propto f(x_i), \quad \left( \gamma - \sum_{j \neq i} X_j \right)_+ \leq x_i < \infty.$$

To sample from these conditional densities, set for all $i = 1, \ldots, d$

$$X_i = \left( (\gamma - \sum_{j \neq i} x_j)^\alpha - \ln U \right)^{\frac{1}{\alpha}}.$$

Here, we generate $n_s$ samples $X_1, \ldots, X_{n_s}$ from $f_s$. Hence, we obtain the pooled sample $X_1, \ldots, X_n$. We can now proceed to formulate the empirical likelihood and perform the optimization.

### 4.2.3 Empirical Likelihood Optimization

Suppose we have generated $n_t$ samples for each density $t = 1, \ldots, 4$ and have obtained a pooled sample $X_1, \ldots, X_n$ where $n = n_1 + \cdots + n_4$. Define the vector of parameters

$$z = (z_1, \ldots, z_n) = (- \log (\ell_1/\lambda_1), \ldots, - \log (\ell_s/\lambda_s)).$$

We want to solve the following constrained nonlinear convex optimization program

$$\min_{\mathbf{z}} \hat{D}(\mathbf{z})$$

subject to:

$$\sum_{i=1}^{n} \lambda_i z_i = 0$$

$$z_1 - z_2 = E_{12},$$

(4.1)
where
\[ E_{12} = \log(\lambda_1) - \log(\lambda_2) + \log(\ell_2) - \log(\ell_1) = \log(\lambda_1) - \log(\lambda_2) - \log(d) + \gamma^\alpha. \]

The objective function is
\[
\hat{D}(z) = \sum_{j=1}^{n} \log \left( \sum_{t=1}^{s} w_t(X_j)e^{z_t} \right)
\]
\[
= \sum_{j=1}^{n} \log \left( f(X_j) \sum_{k=1}^{d} \mathbb{I}\{X_{jk} \geq \gamma\} e^{z_1} + \frac{f_2(X_j)}{f(X_j)} e^{z_2} \right.
\]
\[
+ f(X_j) \mathbb{I}\{S(X_j) \geq \gamma, M(X_j) < \gamma\} e^{z_3}
\]
\[
+ \left. f(X_j) \mathbb{I}\{S(X_j) \geq \gamma\} e^{z_4} \right) \right)
\]
\[
= \sum_{j=1}^{n} \log \left( \sum_{k=1}^{d} \mathbb{I}\{X_{jk} \geq \gamma\} e^{z_1} + \frac{f_2(X_j)}{f(X_j)} e^{z_2} \right.
\]
\[
+ \mathbb{I}\{S(X_j) \geq \gamma, M(X_j) < \gamma\} e^{z_3}
\]
\[
+ \mathbb{I}\{S(X_j) \geq \gamma\} e^{z_4} \right) + \sum_{j=1}^{n} \log f(X_j).
\]

Hence, solving this will allow us to obtain an optimal solution \( \hat{z} \), and \( \hat{\ell} \) up to a multiplicative constant. Using the analytical knowledge of known \( \ell_1 \) or \( \ell_2 \), we are able to recover the remaining \( \hat{\ell} \). Thus, we are able to obtain the estimator of the (rare-event) probability of interest \( \hat{\ell} \). For implementation purposes, we can use the following lemma to speed up the computation time in evaluating the product of marginals from the zero-variance density (2.12).

**Lemma 4.2.3.** The \( i \)th marginal of the zero-variance density can be written as
\[
\hat{\pi}_i(x_i) = \frac{f(x_i)}{n_s} \sum_{j=1}^{n_s} e^{C_{ji}} \mathbb{I}\{x_i \geq C_{ji}\}, \tag{4.2}
\]
where
\[
C_{ji} = \max \left\{ 0, \gamma - \sum_{k \neq i} X_{jk} \right\},
\]
are calculated constants from samples of \( f_s \). It follows that
\[
\frac{f_2(x)}{f(x)} = \prod_{i=1}^{d} \frac{1}{n_s} \sum_{j=1}^{n_s} e^{C_{ji}} \mathbb{I}\{x_i \geq C_{ji}\}, \ x = (x_1, \ldots, x_d). \tag{4.3}
\]
Proof. The proof of lemma 4.2.3 has been delegated to appendix B.2.

The computational cost measured by CPU time will depend on its implementation in software. An efficient implementation to evaluate (4.3) can be achieved by sorting the vector $[C_{ji}], j = 1, \ldots, n_s$, and pre-computing the cumulative sum $\sum_{j=1}^k e^{C_{ij}}$, for each $k = 1, \ldots, n_s$. This is followed by simply looking up which index boundaries the samples fall into. Using vectorization and suitable index ‘look-up’ functions may greatly speed up the computational process. For example, the \texttt{histc} function in \texttt{MATLAB} outputs the bin count which is an efficient way to look up the values in (4.3). See page 63 of the appendix for this implementation in \texttt{MATLAB}.

We can improve the accuracy of each marginal density $\hat{\pi}_i(x_i)$, by exploiting symmetry and iid properties of our samples from $f_s$. This is achieved by increasing the sample size used from $n_s$ to $n_s \times d$. By utilizing all elements of each random sample, a better estimator is

$$\hat{\pi}(x_i) = \frac{f(x_i)}{n_s d} \sum_{i=1}^{n_s} \sum_{j=1}^{d} e^{C_{ij}} \mathbb{I}\{x_i \geq C_{ji}\}.$$  

Further, it also possible to reduce the computational costs by taking a sub-sample of existing $X_1, \ldots, X_s$ to cut down the number of summation terms in (4.3) but this will reduce the estimator’s accuracy. There exists a trade-off between variance reduction and computational speed.

4.3 Numerical Results with Discussion A

For the problem stated in section 4.1 comprehensive simulations were run to compare the performance of the ELM estimator against the MCIS (2.19) and the AK conditional estimator (2.24). Firstly, the shape parameter $\alpha$ is varied from 0.1 to 0.9 in order to illustrate changes displayed by the estimators as the distribution \text{Weib}(\alpha, 1) becomes less heavy-tailed when $\alpha$ approaches 1. Similarly, the rarity parameter $\gamma$ is increased such that the probability of the rare-event becomes smaller. The aim is to capture any numerical display of second order efficiency properties (see appendix A.1) by the estimators.

The numerical package \texttt{MATLAB} is used to implement the given ELM schemes. The function \texttt{fmincon} from the optimization toolbox is used to solve the constrained nonlinear convex optimization program. Simulations were ran on a quad core, 2.83GHz computer. See appendix C.1 for the main program and functions used in \texttt{MATLAB}. The relative error (2.5) and relative time variance product (2.7) for each of the estimators are given. These are key quantities used to compare efficiency between the estimators. The estimate $\hat{\ell}$ (to three significant figures) and the average CPU time (in seconds) are also given.
We begin by examining the results from the ELM scheme with 4 densities. Further, to illustrate the trade off between variance reduction and computational speed, we also give results for an ELM scheme with 3 densities ($f_3$ discarded). For a fair comparison, the sample sizes used to compute the MCIS and AK conditional estimators are adjusted accordingly.

### 4.3.1 ELM Scheme with 4 densities

**Parameters:** We consider the case when $d = 10$. Equal sample sizes of $n_t = 10^4$ from each density $f_t$, $t = 1, \ldots, 4$ are used. Hence, the pooled sample size is $n = 4 \times 10^4$. A random sub-sample of size $0.5 \times n_s$ is taken from $X_1, \ldots, X_n$, to estimate the product of marginals $f_2$. We have used $K = 100$ iterations of the ELM algorithm to estimate its empirical standard error. Both the MCIS and AK conditional estimators used an equivalent sample size of $n = 4 \times 10^4$. The MCIS estimator used sample size of $m = 0.5 \times n$ to estimate the product of marginals importance sampling density.

**Discussion of Results:** The numerical results of the ELM scheme are given in Table 4.1 below. We observe that the ELM is superior to the MCIS estimator with overall smaller relative error and RTVP. The ELM estimator performed better than the AK conditional estimator with smaller relative error and RTVP when $\alpha = 0.6, 0.8, 0.9$. Similarly, the MCIS estimator had lower RTVP than the AK conditional estimator when $\alpha = 0.8, 0.9$. However, the AK conditional estimator outperforms both the ELM and MCIS estimators when $\alpha = 0.1, 0.2$. These results are expected as it is known that the AK conditional estimator performs very well when $0 < \alpha < 0.369$.

Further, for $\alpha = 0.1, 0.2$, the AK conditional estimator is observed to enjoy the property vanishing relative error properties when $\gamma \rightarrow \infty$ (so that $\ell \downarrow 0$). Similarly, we observe that the ELM and MCIS estimators display numerical bounded relative error properties where $\text{RE}(\ell(\gamma)) < K$ for some $K$ independent of $\gamma$. The performance of the ELM estimator seems to be consistent for the whole range of $\alpha < 1$ with relative error of order $10^{-3}$. This is also true for the MCIS estimator with relative error of order $10^{-2}$. All three estimators have smaller relative error and RTVP when $\alpha$ is small. This suggests these estimators are more efficient when the distribution is more heavy-tailed.

On average, the computational time required for the ELM method is 6.6 times higher than the MCIS procedure and 2.6 times higher than computing the AK conditional estimator. After close examination of the ELM procedure, we found the Gibbs sampling procedure for $f_3$ to be the culprit for the bottleneck in computational speed. It is worthwhile to investigate whether discarding $f_3$ will improve the RTVP for the ELM procedure. Hence, we perform another simulation run of the same ELM scheme but with $f_3$ removed.
4.3.2 ELM Scheme with 3 densities

**Parameters:** The density that captures the residual component of the rare-event \( f_3 \) has been omitted. Hence, the overall sample size has been reduced to \( n = 3 \times 10^4 \) for each of the estimation methods. Other algorithmic parameters have remained the same.

**Discussion of Results:** Table 4.2 shows the results from another simulation study using 3 densities instead of 4 in the ELM scheme \((f_3 \text{ discarded})\). We observe there is an overall increase in the relative error for the ELM estimator. The is due to the removal of \( f_3 \) in the ELM procedure. During the initial runs of searching suitable ELM densities, we have found that the inclusion of \( f_3 \) contributes to some degree of variance reduction. However, by removing such density, we obtain significant gains in computational speed (average CPU time of 0.35s compared with 1.92s). The bottleneck from the original scheme of 4 densities seem to result from sampling from such density. On average, the ELM scheme with 3 densities is approximately 5.5 times faster than the ELM scheme with 4 densities. Further, by comparing average CPU time, the ELM procedure now runs faster than computing the AK conditional estimator (0.54s) but slower than calculating the MCIS estimator (0.24s).

This ELM estimator has obtained significant gains in efficiency. Overall, by comparing similar levels of \( \alpha \) and \( \gamma \), the RTVP for the ELM with 3 densities is smaller than the ELM procedure with 4 densities. This suggests the variance reduction achieved by adding in \( f_3 \) is not high enough to warrant the increase in computational time. Due to a decrease in sample size, the computational time is also slightly reduced for obtaining the MCIS and AK estimators. Note we still observe similar patterns in the relative performance of the three estimators.
Table 4.1: Comparison of the ELM method (4 densities) with the MCIS and AK conditional estimator for different $\alpha$ and $\gamma$.

| $\alpha = 0.9$ | Relative Error | RTVP |
|----------------|----------------|------|
| $\gamma$      | $\ell$        | ELM  | MCIS | AK   |
| 30             | 1.63/10^4     | 1.66/10^4 | 1.29/10^4 | 5.73/10^4 | 6.45/10^4 | 4.61/10^4 | 2.3/10^4 |
| 40             | 6.25/10^7     | 2.00/10^3 | 1.68/10^2 | 3.24/10^1 | 8.00/10^6 | 9.15/10^5 | 7.91/10^2 |
| 50             | 2.25/10^9     | 2.27/10^3 | 1.86/10^2 | 2.99/10^1 | 1.12/10^5 | 1.16/10^5 | 6.76/10^2 |
| 60             | 7.01/10^{12}  | 2.70/10^3 | 2.26/10^2 | 2.98/10^1 | 1.47/10^5 | 1.73/10^4 | 6.64/10^2 |
| 90             | 1.54/10^{19}  | 3.33/10^3 | 2.86/10^2 | 3.92/10^1 | 2.17/10^5 | 2.68/10^4 | 1.16/10^1 |

| $\alpha = 0.8$ | Relative Error | RTVP |
|----------------|----------------|------|
| $\gamma$      | $\ell$        | ELM  | MCIS | AK   |
| 30             | 1.59/10^4     | 1.61/10^4 | 1.17/10^4 | 2.29/10^4 | 4.91/10^6 | 4.38/10^5 | 3.96/10^4 |
| 50             | 1.02/10^6     | 2.01/10^3 | 1.72/10^2 | 1.03/10^1 | 7.95/10^6 | 9.17/10^5 | 7.93/10^3 |
| 100            | 1.44/10^{14}  | 3.31/10^3 | 2.65/10^2 | 2.16/10^1 | 2.15/10^5 | 2.19/10^4 | 3.47/10^2 |
| 150            | 1.33/10^{21}  | 3.53/10^3 | 4.10/10^2 | 1.12/10^1 | 2.54/10^5 | 5.08/10^4 | 9.35/10^3 |
| 200            | 5.17/10^{28}  | 3.04/10^3 | 2.37/10^2 | 5.99/10^2 | 1.89/10^5 | 1.84/10^4 | 2.68/10^3 |

| $\alpha = 0.6$ | Relative Error | RTVP |
|----------------|----------------|------|
| $\gamma$      | $\ell$        | ELM  | MCIS | AK   |
| 100            | 9.47/10^8     | 2.02/10^4 | 1.49/10^4 | 2.11/10^4 | 7.65/10^6 | 6.22/10^5 | 3.20/10^4 |
| 150            | 7.86/10^8     | 2.35/10^3 | 1.56/10^2 | 3.63/10^2 | 9.26/10^6 | 6.12/10^5 | 9.30/10^4 |
| 200            | 1.35/10^9     | 1.74/10^3 | 1.64/10^2 | 1.16/10^2 | 5.12/10^6 | 8.41/10^5 | 1.03/10^4 |
| 500            | 1.84/10^{17}  | 1.03/10^3 | 1.21/10^2 | 3.59/10^3 | 2.29/10^6 | 3.72/10^5 | 9.29/10^6 |
| 1000           | 7.01/10^{27}  | 1.03/10^3 | 1.17/10^2 | 2.09/10^3 | 1.95/10^6 | 3.47/10^5 | 3.16/10^6 |

| $\alpha = 0.2$ | Relative Error | RTVP |
|----------------|----------------|------|
| $\gamma$      | $\ell$        | ELM  | MCIS | AK   |
| 10^3           | 1.97/10^4     | 4.42/10^3 | 7.81/10^1 | 1.02/10^1 | 3.29/10^6 | 1.56/10^5 | 7.63/10^4 |
| 10^6           | 4.64/10^4     | 3.41/10^3 | 6.97/10^1 | 5.34/10^1 | 2.37/10^6 | 1.17/10^5 | 2.05/10^7 |
| 10^9           | 1.31/10^6     | 3.83/10^3 | 6.64/10^1 | 8.38/10^1 | 2.59/10^6 | 1.18/10^5 | 5.08/10^9 |
| 10^12          | 1.23/10^10    | 4.01/10^3 | 6.33/10^1 | 2.08/10^1 | 2.83/10^6 | 1.05/10^5 | 3.19/10^10 |
| 10^15          | 5.13/10^{17}  | 3.51/10^3 | 6.35/10^1 | 1.97/10^1 | 2.43/10^6 | 9.95/10^5 | 2.87/10^{12} |

| $\alpha = 0.1$ | Relative Error | RTVP |
|----------------|----------------|------|
| $\gamma$      | $\ell$        | ELM  | MCIS | AK   |
| 10^{10}        | 4.54/10^{10}  | 3.57/10^3 | 6.26/10^1 | 1.27/10^1 | 2.17/10^6 | 1.06/10^5 | 1.20/10^8 |
| 10^{11}        | 3.41/10^{10}  | 3.37/10^3 | 6.30/10^1 | 5.72/10^1 | 2.12/10^6 | 1.12/10^5 | 2.37/10^9 |
| 10^{12}        | 1.31/10^{10}  | 3.41/10^3 | 6.29/10^1 | 6.52/10^1 | 2.45/10^6 | 1.02/10^5 | 3.02/10^{11} |
| 10^{13}        | 2.16/10^{10}  | 3.36/10^3 | 6.35/10^1 | 1.54/10^1 | 1.94/10^6 | 1.01/10^5 | 1.69/10^{12} |
| 10^{14}        | 1.85/10^{11}  | 3.10/10^3 | 6.33/10^1 | 7.35/10^1 | 1.81/10^7 | 1.10/10^5 | 3.87/10^{15} |

Average CPU Time (s)  
| ELM  | MCIS | AK  |
|------|------|-----|
| 1.92 | 0.29 | 0.73 |
| $\alpha$ = 0.9 | $\ell$ | Relative Error | RTVP |
|-------------|-------|---------------|-----|
| $\gamma$    |       | ELM | MCIS | AK  | ELM | MCIS | AK  |
| 30          | 1.33/10^4 | 1.84/10^4 | 1.51/10^4 | 7.69/10^4 | 3.83/10^4 | 5.78/10^4 | 3.16/10^4 |
| 40          | 6.25/10^7 | 2.13/10^3 | 1.82/10^2 | 2.11/10^1 | 1.31/10^6 | 8.30/10^5 | 2.37/10^2 |
| 50          | 2.25/10^9 | 2.52/10^3 | 2.15/10^2 | 9.46/10^1 | 1.94/10^6 | 1.20/10^5 | 4.77/10^1 |
| 60          | 7.01/10^{12} | 2.75/10^3 | 2.50/10^2 | 2.74/10^1 | 3.18/10^6 | 1.59/10^4 | 4.01/10^2 |
| 90          | 1.54/10^{19} | 4.49/10^3 | 3.36/10^2 | 8.91/10^1 | 6.82/10^6 | 2.85/10^4 | 4.23/10^1 |

| $\alpha$ = 0.8 | $\ell$ | Relative Error | RTVP |
|-------------|-------|---------------|-----|
| $\gamma$    |       | ELM | MCIS | AK  | ELM | MCIS | AK  |
| 30          | 1.59/10^4 | 1.66/10^4 | 1.37/10^4 | 2.69/10^4 | 8.20/10^5 | 4.58/10^5 | 3.88/10^4 |
| 50          | 1.02/10^6 | 2.50/10^3 | 1.98/10^2 | 1.18/10^1 | 2.05/10^6 | 9.66/10^5 | 7.53/10^3 |
| 100         | 1.44/10^{14} | 4.11/10^3 | 2.99/10^2 | 3.86/10^1 | 6.03/10^{10} | 2.19/10^4 | 7.97/10^2 |
| 150         | 1.33/10^{21} | 4.03/10^3 | 4.84/10^2 | 1.60/10^1 | 6.31/10^{10} | 5.68/10^4 | 1.37/10^2 |
| 200         | 5.17/10^{28} | 3.17/10^3 | 3.42/10^2 | 5.40/10^1 | 4.52/10^{10} | 2.82/10^4 | 1.55/10^1 |

| $\alpha$ = 0.6 | $\ell$ | Relative Error | RTVP |
|-------------|-------|---------------|-----|
| $\gamma$    |       | ELM | MCIS | AK  | ELM | MCIS | AK  |
| 100         | 9.47/10^{10} | 2.13/10^4 | 1.83/10^4 | 2.67/10^4 | 1.31/10^6 | 7.76/10^5 | 3.82/10^4 |
| 150         | 7.86/10^{8} | 2.06/10^3 | 1.81/10^2 | 1.95/10^1 | 1.25/10^6 | 7.75/10^5 | 2.93/10^4 |
| 200         | 1.35/10^{9} | 1.99/10^3 | 2.22/10^2 | 4.33/10^1 | 1.13/10^6 | 1.26/10^4 | 1.01/10^3 |
| 500         | 1.84/10^{17} | 1.18/10^3 | 1.65/10^2 | 3.90/10^1 | 4.72/10^{10} | 6.03/10^5 | 8.12/10^6 |
| 1000        | 7.01/10^{27} | 0.17/10^4 | 1.41/10^2 | 2.65/10^1 | 2.79/10^{10} | 4.78/10^{10} | 3.89/10^6 |

| $\alpha$ = 0.2 | $\ell$ | Relative Error | RTVP |
|-------------|-------|---------------|-----|
| $\gamma$    |       | ELM | MCIS | AK  | ELM | MCIS | AK  |
| 10^3        | 1.97/10^{10} | 4.55/10^3 | 8.83/10^2 | 1.17/10^1 | 5.98/10^5 | 1.78/10^5 | 7.71/10^4 |
| 10^5        | 4.64/10^{10} | 3.98/10^3 | 7.85/10^2 | 6.45/10^1 | 5.32/10^{10} | 1.40/10^5 | 2.34/10^7 |
| 10^6        | 1.31/10^{10} | 3.40/10^3 | 7.48/10^2 | 2.14/10^1 | 4.07/10^{10} | 1.31/10^5 | 2.56/10^8 |
| 10^7        | 1.23/10^{10} | 3.46/10^3 | 7.35/10^2 | 1.65/10^1 | 3.94/10^{10} | 1.26/10^5 | 1.53/10^9 |
| 10^8        | 5.13/10^{10} | 3.46/10^3 | 7.33/10^2 | 2.54/10^1 | 4.53/10^{10} | 1.20/10^5 | 3.54/10^{12} |

| $\alpha$ = 0.1 | $\ell$ | Relative Error | RTVP |
|-------------|-------|---------------|-----|
| $\gamma$    |       | ELM | MCIS | AK  | ELM | MCIS | AK  |
| 10^{10}     | 4.54/10^{10} | 3.73/10^3 | 7.29/10^2 | 1.23/10^1 | 6.67/10^{10} | 1.19/10^{10} | 8.29/10^{10} |
| 10^{11}     | 3.41/10^{10} | 3.21/10^3 | 7.29/10^2 | 6.88/10^1 | 4.28/10^{10} | 1.19/10^{10} | 2.62/10^{10} |
| 10^{12}     | 1.31/10^{10} | 3.65/10^3 | 7.25/10^2 | 8.07/10^1 | 4.42/10^{10} | 1.17/10^{10} | 3.59/10^{11} |
| 10^{13}     | 2.16/10^{10} | 3.61/10^3 | 7.25/10^2 | 5.24/10^1 | 3.70/10^{10} | 1.14/10^{10} | 1.51/10^{12} |
| 10^{14}     | 1.85/10^{11} | 3.50/10^3 | 7.22/10^2 | 2.01/10^1 | 4.57/10^{10} | 1.15/10^{10} | 2.22/10^{16} |

| Average CPU Time (s) | ELM | MCIS | AK |
|----------------------|-----|-----|-----|
| 0.35                 | 0.24 | 0.54 |
4.4 ELM Scheme B: 2 densities with inequality constraint

In section 3.4, we have mentioned that by maximizing the empirical likelihood subject to both equality and inequality constraints, the efficiency of the estimator $\hat{z}$ can be improved. In this section, we present a new variational optimization method for obtaining a lower bound for rare-event probabilities of interest. The method not only provides a lower bound on $\ell$, which can be incorporated into the likelihood optimization, but as a byproduct it provides an excellent reference density to be used in the ELM sequence of densities. We now explain how we obtain these lower bounds before giving the details of the ELM algorithm that exploits the lower bound. The main finding is summarized in Theorem 4.4.1 below.

4.4.1 Variational Lower Bound on $\ell$

Here, we introduce a novel method in obtaining a lower bound for our (rare-event) probability $\ell$. This serves as the key ingredient in implementing ELM scheme B in section 4.4.2 below. Firstly, we need the following observation to rewrite our Weib($\alpha, 1$) random variables in terms of Exp(1) random variables.

The (rare-event) probability $\ell = P(X_1 + \cdots + X_d \geq \gamma)$, with $X_i \sim \text{Weib}(\alpha, 1)$ and $0 < \alpha < 1$, is equivalent to $\ell = P(Y_1^{1/\alpha} + \cdots + Y_d^{1/\alpha} \geq \gamma)$, $Y_i \sim \text{Exp}(1)$.

Now, we use the following theorem to establish a lower bound $\ell_L$ for the (rare-event) probability of interest $\ell$. That is, $\ell_L \leq \ell$. This will be used to construct a reference density in section 4.4.2 below.

**Theorem 4.4.1.** For any positive value of the variational parameters $\lambda_1, \ldots, \lambda_d$, we have a lower bound to $\ell$ given by

$$\ell_L = P(S_L(Y; \lambda) \geq \gamma) = Q\left(\beta; \alpha \gamma + (1 - \alpha) \sum_i \lambda_i^{1/\alpha}\right),$$

(4.4)

where

$$S_L(y; \lambda) \overset{def}{=} \frac{1}{\alpha} \sum_i \lambda_i^{1/\alpha - 1} y_i - \frac{1 - \alpha}{\alpha} \sum_i \lambda_i^{1/\alpha}$$

(4.5)

and the function $Q(\beta; \gamma) = (1, 0, \ldots, 0) e^A \gamma 1$ is defined via the matrix
exponential with

\[ A = \begin{pmatrix} -\beta_1 & \beta_1 & 0 & \cdots & 0 \\ 0 & -\beta_2 & \beta_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & -\beta_{d-1} & \beta_{d-1} \\ 0 & \cdots & 0 & 0 & -\beta_d \end{pmatrix} \]

and

\[ \beta_j = \frac{\lambda^{1/\alpha - 1} + \cdots + \lambda_d^{1/\alpha - 1}}{d - j + 1}, \quad j = 1, \ldots, d. \]

Proof. The proof of this theorem is given in appendix B.3.

Note the function \( Q(\beta; \gamma) \) is the tail probability of the generalized Erlang distribution [15, §4.2.13]. It is the distribution of a linear combination of \( d \) independent exponential random variables with equal or unequal rates. This quantity can be calculated analytically. Ideally, we want to obtain a lower bound probability \( \ell_L \) close to the true (rare-event) probability. This can be achieved by solving the following maximization program

\[
\lambda^* = \arg\max_{\lambda > 0} P(S_L(Y; \lambda) \geq \gamma).
\]

Here \( \lambda^* \) is the optimal value of the variational parameters that maximizes the lower bound probability. Hence, we can establish a ELM scheme which uses a reference density whose normalizing constant represents the lower bound of the (rare-event) probability of interest.

4.4.2 Formulation of Scheme B

Firstly, we find the optimal solution \( \lambda^* \) to the maximization program given by \((4.6)\). This optimization problem may be difficult to solve as it is non-linear and not convex. However, any heuristic approach such as the cross-entropy method for optimization [15, §13.2] would work. Maximizing this probability will give a lower bound to the probability of interest \( \ell \) which can be incorporated in the ELM scheme.

We apply the procedure as described in section 3.2 and 3.4 with \( s = 2 \) densities. Suppose that \( f(x) \) is the joint density of iid \( \text{Exp}(1) \) distribution. Consider the following densities

1. Lower bound reference density

\[
f_1(x) = \frac{f(x) I \{ x[1] \lambda_1^{(1/\alpha - 1)} + \cdots + x[d] \lambda_d^{(1/\alpha - 1)} \geq \gamma \} \ell_1}{\ell_1} = \frac{w_1(x)}{\ell_1},
\]
where
\[ \gamma^* = \alpha \gamma + (1 - \alpha) \sum_i \lambda_i^{1/\alpha}, \]
and \( x_{[1]}, x_{[2]}, \ldots, x_{[d]} \) represents the order statistics. Here, \( \ell_1 \) is known analytically and is given by

\[ \ell_1 = Q(\beta^*; \gamma^*), \]

with
\[ \beta_j^* = \frac{\lambda_j^{1/\alpha - 1} + \cdots + \lambda_d^{1/\alpha - 1}}{d - j + 1}, \quad j = 1, \ldots, d. \]

An important point is that while the ELM can exploit this lower bound reference density, the reference density by itself cannot be used as an importance sampling density.

2. Zero-variance density

\[ f_2(x) = \frac{f(x) \mathbb{I}\left\{ x_{[1]}^{1/\alpha} + \cdots + x_{[d]}^{1/\alpha} \geq \gamma \right\}}{\ell_2} := \frac{w_2(x)}{\ell_2}, \]

where \( \ell_2 \) is the unknown (rare-event) probability of interest.

Again we emphasize an important advantage of the ELM method: since the reference density \( f_1 \) exhibits light tailed behavior (see appendix A.2) and its support is a subset of the support of \( f_s \), it cannot be used as an importance sampling density to estimate \( f_s \). Nevertheless, it makes for an excellent ELM density.

4.4.3 Sampling

For this scheme there is no need to generate from \( f_1 \). Note that

\[ \{ S_L(Y; \lambda) \geq \gamma \} \subseteq \{ S(Y) \geq \gamma \}, \]

and thus samples attained from \( f_1 \) and \( f_2 \) always satisfies the condition imposed by the zero variance density

\[ \mathbb{I}\left( x_{[1]}^{1/\alpha} + \cdots + x_{[d]}^{1/\alpha} \geq \gamma \right). \]

However, we will need to sample from \( f_1 \) if we change this ELM, say, by adding another reference density. To proceed, we will need to use the Gibbs Sampler to generate samples from

\[ g(z) = \frac{f(x) \mathbb{I}\left\{ \sum_{j=1}^d z_j \beta_j^* \geq \gamma^* \right\}}{\ell_1}. \]
In the Gibbs Sampler, generating from the conditional distribution
\[ g(z_i | z_{-i}) \propto f(z_i), \quad \max \left( 0, \frac{1}{\beta^*_i} \left[ \gamma^* - \sum_{j \neq i} z_j \beta^*_j \right] \right) \leq z_i < \infty, \]
can be achieved by setting the \( i \)th component as
\[ z_i := \left[ \max \left( 0, \frac{1}{\beta^*_i} \left[ \gamma^* - \sum_{j \neq i} z_j \beta^*_j \right] \right) - \log U \right], \quad U \sim \text{Uniform}(0, 1). \]
Finally, realizations from \( f_1 \) can be attained by setting the \( i \)th component as
\[ x_{[i]} := \sum_{j=1}^{i} \frac{z_j}{d-j+1}. \]
Sampling from \( f_2 \) is straightforward by using the Gibbs Sampler and then sorting the components of generated \( x_j \)'s in increasing order.

\textbf{4.4.4 Empirical Likelihood Optimization}

Now, parametrize
\[ z = (z_1, z_2) = (- \log(\ell_1/\hat{\lambda}_1), - \log(\ell_2/\hat{\lambda}_2)), \]
where \( \hat{\lambda}_1 \) and \( \hat{\lambda}_2 \) are the proportion of samples attained from \( f_1 \) and \( f_2 \) respectively. The objective function is
\[ \hat{D}(z) = \sum_{j=1}^{n} \log \left( \sum_{t=1}^{2} w_t(X_j)e^{z_t} \right). \]
Note, in this case, \( \hat{D}(z) \) can be simplified, because it depends on the sufficient statistic \( \hat{p} \), where \( \hat{p} \) denotes the number of samples \( X_j \)'s that satisfy
\[ X_{[1]} \lambda_1^{(1/\alpha-1)} + \cdots + X_{[d]} \lambda_d^{(1/\alpha-1)} \geq \gamma^*. \]
Then, we have
\[ \hat{D}(z) = \sum_{j=1}^{n} \log \left( \sum_{t=1}^{2} w_t(X_j)e^{z_t} \right) \]
\[ = \hat{p} \log(e^{z_1} + e^{z_2}) + (n - \hat{p})z_2 + \sum_{j=1}^{n} \log f(X_j) \]
\[ \overset{\text{def}}{=} \hat{Q}(z) \]
constant
We can then minimize \( \hat{Q}(z) \) instead of \( \hat{D}(z) \). Hence, we solve the following convex optimization program to obtain \( \hat{z} \) and thus \( \hat{\ell} \) up to a multiplicative constant.

\[
\hat{z} = \min_z \hat{Q}(z)
\text{subject to:}
\tilde{\lambda}_1 z_1 + \tilde{\lambda}_2 z_2 = 0,
-z_1 + z_2 \leq -\log(\tilde{\lambda}_1) + \log(\tilde{\lambda}_2).
\]

Using knowledge of \( \ell_1 \), we can recover an estimate \( \hat{\ell}_2 \) of the (rare-event) probability.

We require the equality constraint to obtain an unique solution from the parameter size. The inequality constraint allows us to use the analytical knowledge that \( \ell_1 \leq \ell_2 \).

### 4.5 Numerical Results with Discussion B

Similar to ELM scheme A, we perform comprehensive simulations using ELM scheme B for the problem stated in section 4.1. We vary the shape parameter \( \alpha \) from 0.1 to 0.9 in order to capture any changes displayed by the estimators as \( \alpha \) approaches 1. Further, we increase the rarity parameter \( \gamma \) steadily so the probability of the rare-event becomes smaller. Again, the aim is to capture any numerical display of second order efficiency properties (c.f. Appendix A.1) by the estimators.

We use the cross-entropy method for optimization \cite{15, §13.2} to obtain the lower bound probability. See appendix C.2 for the main program and functions used in MATLAB. We output the lower bound probability and estimated (rare-event) probability for each \( \gamma \) and \( \alpha \). The relative error (2.5) and relative time variance product (2.7) for each of the estimators are given. Further, the total CPU time to obtain the ELM estimator is shown.

**Parameters:** In order to benchmark with ELM scheme A, we take similar algorithmic and problem parameters. The dimension considered is \( d = 10 \). Sample sizes of \( n_t = 2 \times 10^4 \) for each density \( f_t, t = 1, 2 \). Total pooled sample size of \( n = 4 \times 10^4 \). We use \( K = 100 \) iterations to estimate the ELM relative error.

**Discussion of Results:** The numerical results of the lower bound ELM scheme are given in Table 4.3 below. When \( \alpha \) is close to zero, we see that the lower bound probabilities are very close to the (rare-event) probabilities. As \( \alpha \) increases towards 1, the lower bound becomes less tight and the relative error is larger. This suggests the proposed ELM estimator performs slightly
worse when the distribution is less heavy-tailed. However, we observe a significant reduction in the overall relative error when comparing to the results of ELM scheme A. Hence, the addition of the lower bound constraint seem to provide a better ELM estimator than the scheme with only equality constraints. We still observe numerical bounded relative error properties for this ELM scheme.

Compared to ELM scheme A, we observe higher CPU time to calculate the ELM estimator. The bottleneck is due to solving the maximization problem for the optimal $\lambda^*$ and lower bound probability $\ell_1$. There seem to be some inconsistency in the optimization time in finding lower bound probability. Generally, it varies between 8 to 10 seconds observed for $\alpha = 0.8$ and 0.9. However, the optimization takes significantly less time for $\alpha = 0.1$ and large values of $\gamma$ for $\alpha = 0.2$. Instead using cross-entropy optimization, we might want to consider using a gradient optimization procedure in the future. This may decrease the overall computational time and further improve efficiency. Note, the overall RTVP for each $\gamma$ and $\alpha$ is still smaller for ELM scheme B than ELM scheme A. This suggests that ELM scheme B, with the inclusion of lower bound, is superior to the ELM scheme that only incorporates equality constraints.
4.6 Critical Analysis of ELM Method

In this section, we give a critical analysis of the proposed ELM method when applied to rare-event probability estimation. The proposed ELM method has advantages in terms of versatility, general purpose, and usage of flexible constraints. It formulates a convex optimization program which is typically easy to solve. Conversely, the ELM approach can be hindered by high computational costs. Dependent MCMC samples used may impact on the estimation. Good reference densities can be difficult to find (but are always easier to find than a good importance sampling density).
Strengths:

1. Versatility and General Purpose – In the ELM procedure, we can include the samples of any probability density that is known or known up to a multiplicative constant and use it directly in the estimation. A major advantage of ELM densities is that their choice is less restricted than the choice of valid importance sampling densities. The only requirement is the connectivity condition on the supports of the given sequence of densities. Of course, the density (or densities) should somehow be related to the integral we are trying to estimate. In this respect, the situation is similar to the use of control variables in Monte Carlo variance reduction — the inclusion of any control variable reduces the variance, but if the control variable is not highly correlated to the quantity of interest, then the additional computational cost may not be justified.

2. Flexibility of Constraints – We can add in constraints in the optimization to account for any analytical knowledge of reference densities and their relationships. This would typically reduce the variance of the estimator, again similar to the usage of control variates in Monte Carlo methods. For instance, in ELM scheme B, we have seen the inclusion of a lower bound reference density improved the efficiency of the ELM estimator. In the future, it may be possible to find a reference density whose normalizing constant is the upper bound to the rare-event probability. This would give a 100% confidence interval for our estimator and its knowledge can be simply incorporated by using an inequality constraint.

3. Convex Optimization – The ELM procedure results in the optimization of a convex (negative) log-likelihood function. If the constraints added are linear, we will still have a convex optimization program. Convexity typically makes the optimization easier to solve. For example, any local minimum solution found must also be a global minimum.

Shortcomings:

1. Computational Cost – In the ELM method, a significant portion of the computational cost occurs from sampling from densities. The computational speed may be hindered by densities that are difficult to sample from. Typically, this is due nature of MCMC algorithms. For example, the Gibbs sampling of the probability density $f_3$ in ELM scheme A increased the average CPU time approximately 5.5 times. Significant computational costs will increase RTVP which can severely impact efficiency.
2. Dependent Samples – The theory of ELM relies on assuming iid samples. However, in practice, we need to employ MCMC methods to sample from the zero variance density and any density difficult to sample from. Hence, the resulting samples are from a Markov chain which will have some degree of dependence. We are unsure of the impact on estimation. One solution may be looking at the autocorrelations of the samples and discarding certain number of realizations to remove dependence. This is not ideal as it would significantly increase the computational cost. Alternatives like splitting and sequential Monte Carlo methods may be better and need to be explored.

3. Availability of Densities – For a specific problem, it is sometimes difficult to obtain an appropriate sequence of densities. Nevertheless, this is easier than finding a good importance sampling density. Of course, good ELM densities are often inspired by, but not limited to, existing importance sampling methodologies.
Chapter 5

Conclusion

Empirical Likelihood Maximization (ELM) is a novel Monte Carlo method used in rare-event probability estimation. The underlying mechanism involves the construction of an empirical likelihood using samples from a sequence of probability densities which may or may not be sampled sequentially. These densities have normalizing constants which we treat as unknown parameters to be estimated. In particular, the (rare-event) probability of interest is embedded in the empirical likelihood via the zero-variance density. Similar to likelihood maximization, we solve an optimization program to recover the estimator of the (rare-event) probability. A nice feature of the empirical likelihood is its convexity ensuring that we easily obtain a global solution to the optimization.

There exist many Monte Carlo methods used in rare-event probability estimation which are specialized algorithms and would only perform well in certain scenarios while poorly in others. The proposed ELM method is very flexible and can be easily adapted to obtain an efficient estimator for a general problem. Another advantage of the ELM method is its ability to utilize analytical knowledge of reference densities and their interrelationships. The idea is similar to using a control variate – a variance reduction technique used in Monte Carlo methods. This is achieved by simply adding linear equality or inequality constraints in the convex optimization.

Numerical experiments have been performed using this new technique and benchmarks are made against existing algorithms and estimators. The performance of the ELM approach has been consistently reliable and fares well against both the MCIS and AK conditional estimator. We have observed numerical bounded relative error properties for the ELM approach. However, one shortcoming of the ELM procedure is its computational speed to sample from difficult densities via MCMC. When considering different ELM schemes, one needs to consider the trade-off between variance reduction and computational cost. Some possible directions of future research are given below.
Future Research

1. From our numerical experiments, we have observed the ELM estimator exhibits bounded relative error properties. We would like to make theoretical developments in establishing higher order efficiency properties. In particular, we want to establish the asymptotic behaviour of the ELM estimator as \( \gamma \to \infty \).

2. Currently, the relative error of the ELM estimate must be computed empirically from independent runs of the ELM scheme. If we had iid samples, we could use the Fisher information matrix corresponding to \( \hat{D}(z) \) to obtain an asymptotic result. However, the dependence in MCMC sampling makes it difficult to obtain a central limit result for the empirical likelihood. The current approach is inadequate and time consuming. We would like to develop an analytical formula for computing the relative error when we have linear constraints in the optimization and with MCMC sampling. This may involve looking at the asymptotic behaviour as the events become rarer (\( \gamma \to \infty \)) and when sample size becomes larger (\( n \to \infty \)).

3. We would like to explore alternatives to MCMC sampling. One possibility is the Sequential Monte Carlo (SMC) method. As mentioned in section 2.4.3, the SMC or splitting method does not yield competitive estimators compared to the AK estimator. However, using splitting or SMC only for approximate sampling from the zero-variance density \( \pi \) in (2.12) may be advantageous. Finally, we would like to explore the possibility of using upper bound inequality constraints. While we have found a method to obtain lower bounds, we are not presently able to find a variational approach that yields useful upper bounds (standard approaches like Chebyshev or Chernoff bounds are too loose to be useful).
Appendix A

Miscellaneous Concepts

A.1 Efficiency

The probability of interest $\ell$ usually depends on a rarity parameter $\gamma$ such that $\ell$ approaches zero as the threshold level $\gamma$ approaches infinity. That is,

$$\ell = \ell(\gamma) = P(S(X) \geq \gamma)$$

where

$$\ell(\gamma) \downarrow 0 \quad \text{as} \quad \gamma \to \infty.$$ 

Thus, it is of interest to quantify the behaviour of an estimator as the event of interest becomes more rare. A key issue is whether the estimator’s accuracy will deteriorate as the rare-event probability $\ell$ approaches zero. We introduce the following second order efficiency measures from [15, §10.1].

**Definition A.1.1.** Consider a random variable $Z = Z(\gamma)$ with the above property. That is,

$$E(Z(\gamma)) = \ell(\gamma) \to 0^+ \quad \text{as} \quad \gamma \to \infty.$$ 

1. The given estimator has (asymptotically) vanishing relative error if

$$\limsup_{\gamma \to \infty} \frac{Var(Z(\gamma))}{\ell^2(\gamma)} = 0.$$ 

Here, the relative error approaches zero as the event becomes more rare.

2. The given estimator has bounded relative error if

$$\limsup_{\gamma \to \infty} \frac{Var(Z(\gamma))}{\ell^2(\gamma)} \leq K < \infty,$$

where $K$ is a constant that does not depend on $\gamma$. The relative error is bounded regardless of the rarity of the event.
3. The given estimator is **logarithmically efficient** if

\[ \limsup_{\gamma \to \infty} \frac{\text{Var}(Z(\gamma))}{\ell^{2-\epsilon(\gamma)}} = 0 \]

for all \( \epsilon > 0 \), or equivalently if

\[ \liminf_{\gamma \to \infty} \left| \frac{\ln \text{Var}(Z(\gamma))}{\ln \ell^2(\gamma)} \right| \geq 1. \]

Here, the second moment of the estimator and the square of its mean approach zero at the same exponential rate.

The vanishing relative error is a stronger condition than the bounded relative error. Both these conditions are stronger than logarithmic efficiency. An estimator with vanishing error property is the Asmussen-Kroese conditional estimator (see 2.4.2) for distributions with subexponential properties. Generally, it is more difficult to find an estimator with bounded relative error than a logarithmically efficient estimator.

**Example A.1.2.** (Inefficiency of CMC Estimator) The naive Crude Monte Carlo (CMC) estimator is given by

\[ Z = Z(\gamma) = \mathbb{I}\{S(X) \geq \gamma\}. \]

For this estimator

\[ EZ^2 = EZ = \ell \]

and thus

\[ \lim_{\gamma \to \infty} \frac{\log EZ^2}{\log \ell^2} = \frac{\log \ell}{\log \ell^2} = \frac{1}{2} < 1. \]

Hence, the CMC estimator is not logarithmically efficient. For small \( \ell \), the relative error \( \epsilon \) of the CMC estimator satisfies

\[ \epsilon = \sqrt{\frac{\text{Var}(\ell)}{\text{E}\ell}} = \sqrt{\frac{1}{N} \frac{\text{Var}Z}{\ell}} = \sqrt{\frac{1 - \ell}{N\ell}} \approx \sqrt{\frac{1}{N\ell}}. \]

Suppose \( \ell = 10^{-6} \). Then to estimate \( \ell \) accurately with relative error of 1%, we require a sample size of

\[ N \approx \frac{1}{\epsilon^2 \ell} = 10^{10}. \]

The computational cost is too high and using the CMC estimator becomes an infeasible proposition for rare-events. Thus, this motivates the need for improved methods and alternate estimators of rare-event probabilities.

\[ \square \]

We briefly note there exist potential problems with the second-order efficiency measures as they do not convey any information about the accuracy of the estimator \( \hat{\sigma}^2 \) of the true variance \( \sigma^2 \). This issue is addressed by the introduction of higher-order efficiency measures as outlined in [15, §10.1] and [16].
A.2 Tail properties of distributions

A random variable $X$ with cdf $F$ is said to have a (right) light-tailed distribution if its moment generating function is finite for some $t > 0$. That is,
\[ \mathbb{E}(e^{tX}) \leq c < \infty. \] (A.1)

Otherwise, when $\mathbb{E}(e^{tX}) = \infty$ for all $t > 0$, $X$ is said to have heavy-tailed distribution. We only consider the right tail of the distribution as the left tail can be treated equivalently. Note for every $x$
\[ \mathbb{E}e^{tX} \geq \mathbb{E}e^{tX}\mathbb{1}(X > x) \geq e^{tx}\mathbb{P}(X > x). \]

It follows that for any light-tailed distribution satisfying (A.1)
\[ \mathbb{P}(X > x) \leq ce^{-tx}. \]

That is, if $X$ has a light-tailed distribution, then $\bar{F}(x) = 1 - F(x)$ decays at an exponential rate or faster. Similarly, heavy-tailedness property is equivalent to $\lim_{x \to \infty} e^{tx} F(x) = \infty$ for all $t > 0$. On the other hand, any distribution with bounded support is light-tailed. Examples of light-tailed distributions with unbounded support include Exponential, Geometric, Poisson, Gamma, and Weibull($\alpha, \lambda$) for $\alpha \geq 1$. Further, there are additional properties that imply heavy-tailedness. They are given below in the following order of generality:

Regularly varying $\Rightarrow$ Subexponential $\Rightarrow$ Long-tailed $\Rightarrow$ Heavy-tailed.

1. A distribution is said to be long-tailed if
\[ \lim_{x \to \infty} \frac{\bar{F}(x + t)}{\bar{F}(x)} = 1, \quad \text{for all } t. \]

2. A distribution on the interval $(0, \infty)$ is said to be subexponential if, with $X_1, \ldots, X_n \overset{iid}{\sim} F$
\[ \lim_{x \to \infty} \frac{\mathbb{P}(X_1 + \cdots + X_n > x)}{\mathbb{P}(X_1 > x)} = n, \quad \text{for all } n, \]
or equivalently,
\[ \lim_{x \to \infty} \frac{\mathbb{P}(X_1 + \cdots + X_n > x)}{\mathbb{P}(\max\{X_1, \ldots, X_n\} > x)} = 1. \]

3. A distribution is called regularly varying if
\[ \bar{F}(x) = \frac{L(x)}{x^\alpha} \]
for some $\alpha > 0$ and some function $L$ that satisfies $L(tx)/L(x) \to 1$ as $x \to \infty$ for all $t > 0$. 

50
Examples of regularly varying distributions include Student-t, Cauchy, and Pareto. Two families of distributions that are subexponential but not regularly varying are Log-Normal and $\text{Weib} (\alpha, \lambda)$ for $\alpha < 1$. For more detailed class properties of heavy-tailed distributions see [6].

### A.3 Distance Measures

Instead of Kullback-Leibler distance, there exist other distance or divergence measures between probability density functions. An important class of distance measures used to capture the “closeness” between two pdfs $f$ and $g$, is the Csiszár’s $\phi$-divergence

$$d(f, g) = \int f(x) \phi \left( \frac{g(x)}{f(x)} \right) \, dx,$$

(A.2)

where $\phi : \mathbb{R}^+ \to \mathbb{R}$ is twice continuously differentiable with $\phi(1) = 0$ and $\phi''(x) > 0$ for all $x > 0$. Note that $\phi$ is a convex function. Special cases [15] of the $\phi$-divergence include most of the information-theoretic distances such as Burg CE distance, Kullback-Leibler CE distance, Hellinger distance and Pearson $\chi^2$ distance. They are given below

- **Burg CE distance:**
  $$d(f, g) = \int g(x) \log \frac{g(x)}{f(x)} \, dx.$$

- **Kullback-Leibler CE distance:**
  $$d(f, g) = \int f(x) \log \frac{f(x)}{g(x)} \, dx.$$

- **Hellinger distance:**
  $$d(f, g) = 2 \int \left( \sqrt{f(x)} - \sqrt{g(x)} \right)^2 \, dx.$$

- **Pearson $\chi^2$ discrepancy measure:**
  $$d(f, g) = \frac{1}{2} \int \frac{[f(x) - g(x)]^2}{g(x)} \, dx.$$

Formally, $d(f, g)$ is not a distance between $f$ and $g$ as $d(f, g) \neq d(g, f)$ in general. However, it is useful to think of it as a distance measure because

$$d(f, g) \geq 0$$

and $d(f, g) = 0$ if and only if $g(x) = h(x)$. This follows from Jensen’s inequality and convexity of $\phi$. Namely,

$$D(f, g) = \mathbb{E}_f \left[ \phi \left( \frac{g(X)}{f(X)} \right) \right] \geq \phi \left( \mathbb{E}_f \left[ \frac{g(X)}{f(X)} \right] \right) = \phi(1) = 0.$$
A.4 Random Variable Simulation

Estimation and optimization problems in applied probability and statistics often concern sampling from specified target distributions. Standard sampling methods include the inverse-transform method, the acceptance-rejection method. These methods are exact, that is, the generated random variables are distributed exactly according to the target distribution.

A.4.1 Inverse Transform Method

Let $X$ be a random variable with cdf $F$. Since $F$ is a non-decreasing function, the inverse cdf $F^{-1}$ can be defined as

$$\{F^{-1}(y) = \inf\{x : F(x) \geq y\}, \quad 0 \leq y \leq 1\}.$$ 

Let $U \sim \text{Uniform}(0, 1)$. The cdf of the inverse transform $F^{-1}(U)$ is given by

$$P(F^{-1}(U) \geq x) = P(U \leq F(x)) = F(x).$$

Hence, to simulate a random variable from $X$ with cdf $F$, draw $U \sim \text{Uniform}(0, 1)$ and set $X = F^{-1}(U)$.

Algorithm A.4.1. (Inverse-Transform Method)

1. Generate $U \sim \text{Uniform}(0, 1)$.
2. Deliver $X = F^{-1}(U)$.

It is sometimes of interest to apply the inverse transform method to sample from a truncated probability density. More detail is given below.

**Truncation:** Suppose $G_A$ and $G_B$ are two distributions on sets $A$ and $B \subset A$, respectively. Let

$$X \sim G_A, \quad Z \sim G_B.$$ 

If the conditional distribution of $X$ given $X \in B$ coincides with the distribution of $Z$, then the latter distribution is said to be the truncation of $G_A$ to $B$. For this case if $f_X$ is the pdf of $X$, then the pdf of $Z$ is

$$f_Z(z) = \frac{f_X(z)}{\int_B f_X(x) \, dx}, \quad z \in B.$$ 

In the continuous univariate case, the truncation of a density $f(x)$ to an interval $[a, b]$ gives the pdf

$$f_Z(z) = \frac{f(z)}{\int_a^b f(x) \, dx}, \quad a \leq x \leq b.$$
Writing in terms of cdfs, we have
\[ F_Z(z) = \frac{F(z) - F(a^-)}{F(b) - F(a^-)}, \quad a \leq x \leq b, \]
where \( F(a^-) = \lim_{x \uparrow a} F(x) \). When the generation of \( X \) can be readily performed via the inverse-transform method, we can simulate from the truncated distribution using algorithm A.4.2 below.

**Algorithm A.4.2.** (Truncation with the Inverse-Transform Method)

1. Generate \( U \sim \text{Uniform}(0, 1) \).
2. Deliver \( Z = F^{-1}(F(a^-) + U(F(b) - F(a^-))) \).

The only difference with the inverse-transform algorithm A.4.1 is in step 2 where the argument of \( F^{-1} \) is uniformly distributed on the interval \((F(a^-), F(b))\) rather than on \((0, 1)\).

**A.4.2 Acceptance-Rejection Method**

The inverse-transform method will fail when the inverse of the cdf does not have a closed form. An alternative exact method of sampling is the acceptance-rejection method. It is one of the most useful methods for sampling from general distributions. The method is based on the following observation.

**Theorem A.4.3.** (Acceptance-Rejection) Let \( f(x) \) and \( g(x) \) be two pdfs such that for some \( C \geq 1 \), \( Cg(x) \geq f(x) \) for all \( x \). Let \( X \sim g(x) \) and \( U \sim \text{Uniform}(0, 1) \) be independent. Then, the conditional pdf of \( X \) given \( U \leq f(x)/(Cg(x)) \) is \( f(x) \).

The proof is given chapter 3 of [15]. The density \( g(x) \) is called the proposal pdf and we assume that it is easy to generate random variables from it. The acceptance-rejection method can be formulated below in algorithm A.4.4.

**Algorithm A.4.4.** (Acceptance-Rejection)

1. Generate \( X \) from \( g(x) \).
2. Generate \( U \) from \( \text{Uniform}(0, 1) \) independently of \( X \).
3. If \( U \leq f(X)/(Cg(X)) \), output \( X \). Else, reject \( X \) and return to step 1.

We generate \( X \sim g \) and accept it with probability \( f(X)/(Cg(X)) \), otherwise reject \( X \) and try again.
A.5 Markov chain Monte Carlo

Markov chain Monte Carlo (MCMC) is a generic method used for approximate sampling from an arbitrary distribution. The key idea is to generate a Markov chain whose limiting distribution is equal to the desired distribution. It is utilized in many cases where exact sampling is very costly or infeasible.

The idea of MCMC was first introduced by Metropolis, Rosenbluth, Rosenbluth, Teller and Teller (1953) in [17] as a method for efficient simulation of the energy levels of atoms in a crystalline structure. It is subsequently adapted by Hastings (1970) in [11] to focus upon statistical problems and is now frequently used in Bayesian statistics.

Here we focus two prominent MCMC algorithms: the Metropolis-Hastings algorithm and, its special case, the Gibbs Sampler. This algorithm has been applied several times during our numerical experiments in Chapter 4.

A.5.1 Metropolis-Hastings Algorithm

This is a general way to construct the MCMC sampler. It is analogous to rejection sampling where realizations are drawn from approximate distributions. However, instead of rejecting samples, we correct them so they asymptotically behave as random observations from the target distribution. The method induces a Markov chain by sequentially drawing candidate observations from a distribution conditional on the last observation.

**Formulation:** Suppose that we want to generate samples from an arbitrary multi-dimensional probability density

\[ f(x) = \frac{p(x)}{Z}, \quad x \in \mathcal{X} \subseteq \mathbb{R}^n, \]

where \( p(x) \) is a known positive function, and \( Z \) is a known or unknown normalizing constant. Let \( q(y|x) \) be the proposal density or candidate generating function. This is a Markov transition density describing how to go from state \( x \) to \( y \). When a process is at the point \( x \), the density generates a value \( y \) from \( q(y|x) \). Algorithm A.5.1 below illustrates how a Markov chain with state space \( \mathcal{X} \) and stationary distribution \( f(x) \) is constructed.

**Algorithm A.5.1.** (Metropolis-Hastings Algorithm) To sample from a density \( f(x) \) known up to a normalizing constant, initialize with some \( X_0 \) for which \( f(X_0) > 0 \). For each \( t = 0,1,2,\ldots,T-1 \), execute:

1. Given current state \( X_t \), generate \( Y \sim q(y|X_t) \)
2. Generate \( U \sim \text{Uniform}(0,1) \) and deliver

\[
X_{t+1} = \begin{cases} 
Y & \text{if } U \leq \alpha(X_t, Y) \\
X_t & \text{otherwise,}
\end{cases}
\]
where
\[
\alpha(x, y) = \min \left\{ \frac{f(y)q(x|y)}{f(x)q(y|x)}, 1 \right\}.
\]

The probability \(\alpha(x, y)\) is the acceptance probability. If the move is accepted, the process moves to \(Y\), otherwise it remains at \(X_t\). We can also replace \(f\) by \(p\) in the acceptance probability.

The resulting Metropolis-Hastings Markov chain is \(X_0, X_1, \ldots, X_T\) with \(X_T\) being approximately distributed as \(f(x)\) for large \(T\). In other words, if we run the chain for long enough, the realizations from the chain can be treated as a sample from the target distribution. However, the samples obtained would not be independent. One Metropolis-Hastings iteration is equivalent to generating a point from the transition density \(\kappa(x_{t+1}|x_t)\), where
\[
\kappa(y|x) = \alpha(x, y)q(y|x) + (1 - \alpha^*(x))\delta_x(y)
\]
with
\[
\alpha^*(x) = \int \alpha(x, y)q(y|x)dy \quad \text{and} \quad \delta_x(y) \text{ as the Dirac delta function}.
\]

The transition density is required to satisfy the detailed balance equation or also known as the reversibility condition
\[
f(x)\kappa(y|x) = f(y)\kappa(x|y).
\]
Hence, the density \(f\) must be the stationary probability density of the Markov chain. Now, suppose the event \(\{X_{t+1} = X_t\}\) has positive probability. That is, the transition density \(q\) satisfies the condition
\[
P(\alpha(X_t, Y) < 1|X_t) > 0
\]
and
\[
q(y|x) > 0
\]
for all \(x, y \in \mathcal{X}\). Then \(f\) must also be the limiting density of the Markov chain.

With exception of trivial cases, as long as the chain has a probability of eventually reaching any state from any other state, the MCMC algorithm will work. For an ergodic and irreducible Markov chain, the stationary distribution is also the limiting distribution of successive iterations from the chain.

### A.5.2 Gibbs Sampler

The Gibbs Sampler is a special case of the Metropolis-Hastings algorithm. The proposal or candidate generating function is just the conditional distribution
\[
q(x|y) = f(x|y).
\]
Further, the acceptance probability is 1. In other words, every proposal \( y \) is accepted with 
\[
\alpha(x, y) = 1 \quad \forall (x, y).
\]

A distinguishing feature that sets apart the Gibbs Sampler is that the underlying Markov chain is built by a sequence of conditional distributions. It is used widely whenever it is easy to sample from the conditional distributions of the joint density.

**Formulation:** Suppose that we want to sample a random vector \( X = (X_1, \ldots, X_n) \) from a target probability density \( f(x) \). Using the Bayesian notation, let 
\[
f(x_i|x_{-i}) = f(x_i|x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)
\]
represent the conditional probability density of the \( i \)-th component, \( X_i \), given all the other components except \( x_i \). Algorithm A.5.2 below describes the implementation of the Gibbs sampler.

**Algorithm A.5.2.** (Gibbs Sampler) Given an initial state \( X_0 \), iterate the following steps for \( t = 0, 1, \ldots \)

1. For a given \( X_t \), generate \( Y = (Y_1, \ldots, Y_n) \) as follows:
   (a) Draw \( Y_1 \) from \( f(x_1|X_{t,2}, \ldots, X_{t,n}) \).
   (b) Draw \( Y_i \) from \( f(x_i|Y_1, \ldots, Y_{i-1}, X_{t,i+1}, \ldots, X_{t,n}) \) for \( i = 2, \ldots, n-1 \).
   (c) Draw \( Y_n \) from \( f(x_n|Y_1, \ldots, Y_{n-1}) \).
2. Set \( X_{t+1} = Y \).

The transition probability density function is given by 
\[
\kappa_{1\to n}(y|x) = \prod_{i=1}^{n} f(y_i|y_1, \ldots, y_{i-1}, x_{i+1}, \ldots, x_n),
\]
where the subscript \( 1 \to n \) indicates that the components of vector \( x \) are updated in the systematic order \( 1 \to 2 \to 3 \to \cdots \to n \). Now if the transition density of the reverse move \( y \to x \), then the vector \( y \) is updated in order \( n \to n-1 \to \cdots \to 1 \). In this case, we have 
\[
\kappa_{n\to 1}(x|y) = \prod_{i=1}^{n} f(x_i|y_1, \ldots, y_{i-1}, x_{i+1}, \ldots, x_n).
\]

Algorithm A.5.2 is known as the systematic or coordinate-wise Gibbs Sampler. The completion of all the conditional sampling steps in the specified order is called a cycle.
There are alternative ways to update the components of \( \mathbf{X} \). In the random sweep or scan Gibbs sampler, a single cycle can consist of one or several coordinates selected uniformly from the integers 1, \ldots, n or a random permutation \( \pi_1 \to \pi_2 \to \cdots \to \pi_n \) of all coordinates. The resulting Markov chain \( \{ \mathbf{X}_t, t = 1, 2, \ldots \} \) is reversible. In the situation where a cycle consists of a single randomly selected coordinate, the random Gibbs sampler can be viewed as a Metropolis-Hastings sampler with transition function

\[
q(y|x) = \frac{1}{n} f(y_{i-1}|x_{i-1}) = \frac{1}{n} \frac{f(y)}{\sum_y f(y)},
\]

where \( y = (x_1, \ldots, x_{i-1}, y_i, x_{i+1}, \ldots, x_n) \). Since \( \sum_y f(y) \) can also be written as \( \sum_x f(y) \), we have

\[
\frac{f(y)q(x|y)}{f(x)q(y|x)} = \frac{f(y)f(x)}{f(x)f(y)} = 1,
\]

so in this case, the acceptance probability \( \alpha(x, y) \) is 1.
Appendix B

Miscellaneous Proofs

B.1 Lemma 2.3.1

Suppose that $\pi_i(y_i)$ is the marginal of the zero-variance density $\pi(y)$. Then, the solution to functional optimization program (2.15) is

$$g_i(y_i) = \pi_i(y_i) \quad \text{for all} \quad i = 1, \ldots, d.$$  

That is, the best importance sampling density (in the cross-entropy sense) within the space of all product-form densities is the product of marginals for (2.11). This is given by

$$g(y) = \prod_{i=1}^{d} \pi_i(y_i).$$  \hspace{1cm} (B.1)

Proof. Consider

$$d(\pi, g) = \int \pi(y) \log \left( \frac{\pi(y)}{\prod_{i=1}^{d} g_i(y_i)} \right) dy$$

$$= \int \pi(y) \log (\pi(y)) dy - \int \pi(y) \sum_{i=1}^{d} \log(g_i(y_i)) dy$$

$$= \int \pi(y) \log (\pi(y)) dy - \sum_{i=1}^{d} \int \pi_i(y_i) \log(g_i(y_i)) dy_i$$

$$= \sum_{i=1}^{d} \int \pi_i(y_i) \log \left( \frac{\pi_i(y_i)}{g_i(y_i)} \right) dy_i + K(\pi(y)),$$

where $K(\pi(y))$ consists of terms without $g_i$ for all $i = 1, \ldots, d$. Hence, the functional optimization program (2.15) is equivalent to

$$\min_{g \in \mathcal{G}} \int \pi_i(y_i) \log \left( \frac{\pi_i(y_i)}{g_i(y_i)} \right) dy_i, \quad \text{for all} \quad i = 1, \ldots, d.$$  \hspace{1cm} (B.2)
We recognize this as the cross-entropy distances between the marginals $\pi_i$ and $g_i$ for all $i$. These are zero if and only if $g_i = \pi_i$ for all $i$.

\section*{B.2 Lemma 4.2.3}

The $i$th marginal from the zero-variance density can be written as

$$\hat{\pi}_i(x_i) = \frac{f(x_i)}{n_s} \sum_{j=1}^{n_s} e^{C_{ji}} I\{x_i \geq C_{ji}\}, \quad (B.3)$$

where

$$C_{ji} = \max \left\{ 0, \gamma - \sum_{k \neq i} X_{jk} \right\},$$

are calculated constants from samples of $f_s$. It follows that

$$f_2(x) = \prod_{i=1}^{d} \frac{1}{n_s} \sum_{j=1}^{n_s} e^{C_{ji}} I\{x_i \geq C_{ji}\}, \quad x = (x_1, \ldots, x_d). \quad (B.4)$$

\textbf{Proof.} Recall that our product of marginals density is

$$f_2(x) = \prod_{i=1}^{d} \hat{\pi}_i(x_i), \quad x = (x_1, \ldots, x_d),$$

where $\hat{\pi}_i(x_i)$ are the marginal densities of $f_s$. Using samples $X_1, \ldots, X_{n_s}$ generated from $f_s$, we can estimate $\hat{\pi}_i(x_i)$ by their conditionals

$$\hat{\pi}_i(x_i) = \frac{1}{n_s} \sum_{j=1}^{n_s} f_s(x_i | X_{j,-i}),$$

where $X_{j,-i} = (X_{j,1}, \ldots, X_{j,i-1}, X_{j,i+1}, \ldots, X_{j,d})$ is the $j$-th sample without the $i$-th element. Now, by iid property of each sample

$$f_s(x_i | X_{j,-i}) = \begin{cases} f(x_i), & \text{if } \sum_{k \neq i} X_{jk} > \gamma \\ f(x_i) I\{x_i \geq \gamma - \sum_{k \neq i} X_{jk}\} / K_{ji}, & \text{if } \sum_{k \neq i} X_{jk} \leq \gamma, \end{cases}$$

where

$$K_{ji} = \mathbb{E}_f \left( I\left\{ X \geq \gamma - \sum_{k \neq i} X_{jk} \right\} \right) = \mathbb{P}\left( X \geq \gamma - \sum_{k \neq i} X_{jk} \right) = \exp\left( - \left( \gamma - \sum_{k \neq i} X_{jk} \right)^{\alpha} \right)$$

59
is the normalizing probability that a \( \text{Weib} (\alpha, 1) \) random variable exceeds some threshold. Now let us define
\[
C_{ji} = \max \left\{ 0, \gamma - \sum_{k \neq i} X_{jk} \right\}.
\]
Thus, we have
\[
\frac{f_a(x_i|X_{j,-i})}{f(x_i)} = \mathbb{I} \left\{ \sum_{k \neq i} X_{jk} > \gamma \right\} + \frac{1}{K_{ji}} \mathbb{I} \left\{ \sum_{k \neq i} X_{jk} \leq \gamma \right\} \mathbb{I} \left\{ x_i \geq \gamma - \sum_{k \neq i} X_{jk} \right\}
\]
\[
= \begin{cases} 
1 & \text{if } \sum_{k \neq i} X_{jk} > \gamma \\
0 & \text{if } \sum_{k \neq i} X_{jk} \leq \gamma \text{ and } x_i < \gamma - \sum_{k \neq i} X_{jk} \\
\frac{1}{K_{ji}} & \text{if } \sum_{k \neq i} X_{jk} \leq \gamma \text{ and } x_i \geq \gamma - \sum_{k \neq i} X_{jk}
\end{cases}
\]
\[= e^{C_{ji}} \mathbb{I} \{ x_i \geq C_{ji} \}.
\]
Hence,
\[
\hat{\pi}_i(x_i) = \frac{f(x_i)}{f(x)} \sum_{j=1}^{n_s} e^{C_{ji}} \mathbb{I} \{ x_i \geq C_{ji} \},
\]
which implies
\[
\frac{f_2(x)}{f(x)} = \prod_{i=1}^{d} \frac{1}{n_s} \sum_{j=1}^{n_s} e^{C_{ji}} \mathbb{I} \{ x_i \geq C_{ji} \}.
\]

\[\square\]

### B.3 Theorem 4.4.1

For any positive value of the variational parameters \( \lambda_1, \ldots, \lambda_d \), we have a lower bound to \( \ell \) given by
\[
\ell_L = \mathbb{P}(S_L(Y; \lambda) \geq \gamma) = Q \left( \beta; \alpha \gamma + (1 - \alpha) \sum \lambda_i^{1/\alpha} \right),
\]
where
\[
S_L(Y; \lambda) \overset{d=1}{=} \frac{1}{\alpha} \sum \lambda_i^{1/\alpha - 1} y_i - \frac{1 - \alpha}{\alpha} \sum \lambda_i^{1/\alpha}
\]
and the function \( Q(\beta; \gamma) = (1, 0, \ldots, 0) e^A \gamma \) is defined via the matrix exponential with
\[
A = \begin{pmatrix}
-\beta_1 & \beta_1 & 0 & \cdots & 0 \\
0 & -\beta_2 & \beta_2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
0 & \cdots & 0 & -\beta_{d-1} & \beta_{d-1} \\
0 & \cdots & 0 & 0 & -\beta_d
\end{pmatrix}.
\]
and
\[ \beta_j = \frac{\lambda_j^{1/\alpha - 1} + \cdots + \lambda_d^{1/\alpha - 1}}{d - j + 1}, \quad j = 1, \ldots, d. \]

**Proof.** Consider the convex function \( S(y) = y_1^{1/\alpha} + \cdots + y_d^{1/\alpha} \). For convex functions, we have the following inequality
\[ S(y) \geq S(\lambda) + \nabla S(\lambda)^\top (y - \lambda), \quad \text{for all valid } \lambda. \]

Hence,
\[ S(y) \geq \frac{1}{\alpha} \sum_i \lambda_i^{1/\alpha - 1} y_i - \frac{1 - \alpha}{\alpha} \sum_i \lambda_i^{1/\alpha} \overset{\text{def}}{=} S_L(y; \lambda) \]
is valid for all values of \( \lambda \geq 0 \). Thus, we have \( \{S_L(Y; \lambda) \geq \gamma\} \subseteq \{S(Y) \geq \gamma\} \). It follows that
\[ \ell_L = \sup_{\lambda \geq 0} \mathbb{P}(S_L(Y; \lambda) \geq \gamma) \leq \mathbb{P}(S(Y) \geq \gamma) = \ell. \]

We now exploit the following representation of the order statistic \( Y_{[1]}, \ldots, Y_{[d]} \) of \( \text{Exp}(1) \) exponential random variables (see [5]):
\[ Y_{[i]} = \sum_{j=1}^i \frac{Z_j}{d - j + 1}, \quad Z_1, \ldots, Z_d \overset{iid}{\sim} \text{Exp}(1). \]

Note that
\[ \sum_i Y_{[i]}^{1/\alpha - 1} = \sum_i \lambda_i^{1/\alpha - 1} \sum_{j=1}^i \frac{Z_j}{d - j + 1} = \sum_{j=1}^d Z_j \beta_j, \quad \beta_j = \frac{\lambda_j^{1/\alpha - 1} + \cdots + \lambda_d^{1/\alpha - 1}}{d - j + 1} \]
Therefore, we have
\[ \mathbb{P}(S_L(Y; \lambda) \geq \gamma) = \mathbb{P} \left( Y_{[1]}^{1/\alpha - 1} + \cdots + Y_{[d]}^{1/\alpha - 1} \geq \alpha \gamma + (1 - \alpha) \sum_i \lambda_i^{1/\alpha} \right) \]
\[ = \mathbb{P} \left( \sum_{j=1}^d Z_j \beta_j \geq \alpha \gamma + (1 - \alpha) \sum_i \lambda_i^{1/\alpha} \right) \]
\[ = Q \left( \beta; \alpha \gamma + (1 - \alpha) \sum_i \lambda_i^{1/\alpha} \right), \]
where the function $Q(\beta; \gamma) = (1, 0, \ldots, 0) e^{A\gamma} 1$ is defined via the matrix exponential with

$$A = \begin{pmatrix}
-\beta_1 & \beta_1 & 0 & \cdots & 0 \\
0 & -\beta_2 & \beta_2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
0 & \cdots & 0 & -\beta_{d-1} & \beta_{d-1} \\
0 & \cdots & 0 & 0 & -\beta_d
\end{pmatrix}.$$ 

Note $Q$ is the tail probability of a Generalized Erlang distribution. This is a specific case of a phase-type distribution. For more details, see [15, §4.2.13].
Appendix C

MATLAB Code

This appendix chapter will contain the main MATLAB programs and functions used in the numerical examples from sections 4.3 and 4.5.

C.1 ELM scheme A: 4 densities and equality constraints

The main program `weibull_main_A.m` computes the estimator from the ELM scheme A. Also, the MCIS estimator and AK-conditional estimator are computed along with their relative errors. This main program calls functions used to sample from the 4 densities and functions used for the constrained empirical likelihood optimization. This is illustrated in figure C.1 below.

![Figure C.1: Functions for ELM Scheme A](image)
C.1.1 Main program

Main program weibull_main_A.m

%%
% weibull_main_A.m
% Date: 23/10/2013
% Description: Main program to compute estimators from ELM scheme A, MCIS
% and AK procedure
%%
clear all, clc
format long g

%% Problem Parameters
d=10;
alpha=0.9;
gamma=90;

%% ELM Scheme 1: 4 densities with equality constraints

% Algorithmic Parameters
nt=[1,1,1,1]*10^4;
s=length(nt); n=sum(nt);
lam=nt/n;

% Gibbs Sampling from f_s (Conditional on the rare-event of interest)
Xs = fs_rand(gamma,alpha,d,nt(4));

% Gibbs Sampling from f_3
X3 = f3_rand(gamma,alpha,d,nt(3));

% Computing C_kj = max{0,gamma-sum_{j neq i}(X_ki)}
% Take a subsample of Xs to estimate product of marginals
m=nt*s*0.5;
Xss=Xs(randperm(m),:);
C = NaN(m,d);
sums=sum(Xss,2);
for i=1:d
    C(:,i)=max(0,gamma-sums+Xss(:,i));
end

% Setup for efficient calculation of the product of marginals
C=sort(C(:,1),’ascend’);
pi=cumsum(exp(C.’*alpha’))/(m*d);

% Sampling from f_2 (Reference density)
X2 = f2_rand(C,alpha,d,nt(2));

% Sampling from f_1 (Reference density)
X1 = f1_rand(gamma,alpha,d,nt(1));

% Pooled Samples
X = [X1; X2; X3; Xs];
% Coefficients in objective function D(z)
global Coeff
Coeff=NaN(4,n);
Coeff(1,:)=sum(X>=gamma,2);
Coeff(3,:)=sum(X,2)>=gamma & max(X,[1],2)<gamma;
Coeff(4,:)=sum(X,2)>=gamma;

% Coefficients from f2 (efficient implementation)
[~,bin]=histc(X,C);
bin(bin==0)=size(C,1);
Coeff(2,:) = prod(pi(bin),2)';

% Optimization Options
options = optimset('Algorithm','interior-point','TolX',10^-14,'TolFun',10^-10,...
'GradConstr','on','Display','iter-detailed','GradObj','on',...
'Hessian','user-supplied','HessFcn','hessianfcn',...
'MaxFunEvals',10^5,'MaxIter',10^5);

% Constrained Optimization with fmincon
[z,fval,exitflag,output] = fmincon(@(z)myfun(z,lam),zeros(1,4),...
[],[],[],[],[],@(z)mycon(z,lam,-gamma^alpha+log(d)),options);

% Extract ell from optimal z
ell_con=lam.*exp(-z);

% Retrieve ELM estimator using analytical knowledge of ell1
prob=exp(-gamma^alpha)*d;
ell_con=ell_con/ell_con(1)*prob;

% MCIS Estimator
N=n;
ell=NaN(N,1);
X_IS = f2_rand(C,alpha,d,N);

% Efficient Implementation
[dummy,binIS]=histc(X_IS,C);
binIS(binIS==0)=size(C,1);
ell = (sum(X_IS,2)>gamma)./prod(pi(binIS),2);
ell_is=mean(ell);
RE_is=std(ell)/mean(ell)/sqrt(nt(2));

% Asmussen-Kroese conditional estimator
N=n;
Y=NaN(N,1);
for i=1:N
x=(-log(rand(1,d-1)))'/(1/alpha);
Y(i)=d*exp(-max([x,gamma-sum(x)])^alpha);
end
ell_cond = mean(Y);
RE_cond = std(Y)/mean(Y)/sqrt(N);

65
C.1.2 Functions for Sampling

Function f1_rand.m to sample from $f_1(x)$

```matlab
%%
% Sampling from f1: main 'heavy-tailed' component
%%
function out = f1_rand(gamma, alpha, d, n)

out = NaN(n, d);
prob = ceil(rand(n, 1) * d);
for i = 1:n
    for j = 1:d
        if (j == prob(i))
            out(i, j) = (gamma^alpha - log(rand))^(1/alpha);
        else
            out(i, j) = (-log(rand))^(1/alpha);
        end
    end
end
```

Function f2_rand.m to sample from $f_2(x)$

```matlab
%%
% Sampling from f2: product of marginals from the zero-variance density
%%
function out = f2_rand(C, alpha, d, n)

% generate j in (1,...,ns*d) with equal probability
J = ceil(rand([n, d]) .* size(C, 1));

out = ((C(J).^alpha - log(rand([n, d])))^(1/alpha));
```

Function f3_rand.m to sample from $f_3(x)$

```matlab
%%
% Gibbs Sampling from f3: residual component
%%
function out = f3_rand(gamma, alpha, d, n)

x = ones(1, d) * gamma / d;
s = sum(x); out = NaN(n, d);
for i = 1:n
    for j = 1:d
        s = s - x(j);
        x(j) = trwbl(max(gamma - s, 0), gamma, alpha);
        s = s + x(j);
    end
    out(i, :) = x(randperm(d));
end
```
Function trwbl.m to sample from truncated Weibull distribution

%%%
% Simulating from general truncated Weibull(alpha,1) distribution
%%%
function out=trwbl(a,b,alpha)

out=a.^alpha-log(1-rand(size(a)).*(1-exp(a.^alpha-b^alpha)));
out=out.^(1/alpha);

Function fs_rand.m to sample from \( f_s(x) \)

%%%
% Gibbs Sampling from f4: zero-variance density
%%%
function out=fs_rand(gamma,alpha,d,n)

x=ones(1,d)*gamma/d;
s=sum(x); out=NaN(n,d);
for i=1:n
    for j=1:d
        s=s-x(j);
        x(j)=(-log(rand)+(max(0,gamma-s))^alpha)^(1/alpha);
        s=s+x(j);
    end
    out(i,:)=x(randperm(d));
end

C.1.3 Functions for Optimization

Function myfun.m to evaluate the objective function \( D(z) \)

%%%
% Objective function
%%%
function [fval,gradval]=myfun(z,lam)

global Coeff
z=z(:)';
[s,n]=size(Coeff);
y=exp(z)*Coeff;
fval=sum(log(y))/n-lam*z';
if nargout>1
    gradval=exp(z').*sum(Coeff./repmat(y,s,1),2)/n-lam';
end
%% Function to specify constraints
function [c,ceq,gradc,gradceq] = mycon(z,lam,p)

% 4 densities
% 2 constraints

if nargout > 2

end

Function hessianfcn.m to compute Hessian of objective function

%% Compute Hessian of objective function
function out=hessianfcn(z,Lagr_mult)

 global Coeff

out=out/n;

C.2 ELM scheme B: 2 densities with inequality constraint

The main program `weibull_main_B.m` computes the estimator for ELM scheme B. The lower bound probability is evaluated by calling `lowerbound.m`. Functions used to sample from $f_2(x)$ and to perform the constrained optimization are also provided below. This is illustrated in figure C.2 below.
C.2.1 Main program

------------- Main program weibull_main_B.m -------------

%%% weibull_main_B.m
%%% Date: 23/10/2013
%%% Description: Main program to compute the lower bound probability and
%%% estimator from ELM scheme B
%%% 

format long g
format compact
clear all
clc

% Problem Parameters
gamma=90;
d=10;
alpha=0.9;

% Calculate optimal lambda, lower bound ell_1 proobability
[prob, lamstar] = lowerbound(gamma, d, alpha);

% Calculate gamma star
gamstar = alpha*gamma + (1-alpha)*sum(lamstar.^(1/alpha));

% Algorithmic parameters
nt=[.1,1]*10^5;
s=length(nt); n=sum(nt);
lambda=nt/n;

% Gibbs Sampling from f_2 zero variance density
% Conditional on the rare-event of interest
X2 = f2_rand(gamma, alpha, d, nt(2));

global Coeff
Coeff=NaN(2,n);
Coeff(1,:) = [ones(1,nt(1))
             (sum(repmat(lamstar.^(1/alpha-1),nt(2),1).*X2,2)>= gamstar)'
            ];
Coeff(2,:) = ones(1,n);

% Optimization Options
options = optimset('Algorithm','interior-point','TolX',10^-14,'TolFun',10^-10,...
                    'GradConstr','on','Display','iter-detailed','GradObj','on',...
                    'Hessian','user-supplied','HessFcn',@hessianfcn,...
                    'MaxFunEvals',10^5,'MaxIter',10^5);
% Constrained Optimization
[z,fval,exitflag,output] = fmincon(@(z)myfun(z,lam),zeros(1,2),...
                                  [],[],[],[],@(z)mycon(z,lam),options);

ell_con=lam.*exp(-z);
el1l_con=ell_con/ell_con(1)*prob;
el1l_con

C.2.2 Functions for Lower Bound Probability

Function lowerbound.m to calculate \ell_L and optimal \lambda^*

%%% Solves the nonlinear optimization using cross entropy optimization
%%% to find lower bound probability and optimal lambda
%%% function [prob lamstar] = lowerbound(gamma,d,alpha)

se=3*ones(1,d); m=zeros(1,d); S_old=inf;
N=1000;
for iter=1:10^6
L=randn(N,d).*repmat(se,N,1)+repmat(m,N,1);
    for i=1:N
        S(i)=q(L(i,:)',gamma,alpha);
    end
    I=S>prctile(S,50);
    m=mean(L(I,:));
    se=std(L(I,:));
    if norm(max(S)-S_old)/max(S)<10^-6, break, end
    S_old=max(S);
end

lamstar = exp(m);
prob = q(m,gamma,alpha);

Function q.m

%%%
% Function to compute the tail probability of Generalized Erlang distribution

function out=q(lam, gamma, a)

d=length(lam);
beta = NaN(1,d);
for j=1:d
    beta(j)=sum(exp(lam(j:d)*(1/a-1)))/(d-j+1);
end

out=convolution(a*gamma+(1-a)*sum(exp(lam/a)),1./beta);

----------- Function convolution.m  --------------------------

%%
% Function to calculate the convolution P(A_1+...+A_b>t)

function ell=convolution(t,nu)
b=length(nu);
A=diag(-nu)+diag(nu(1:b-1),1);
A=expm(A*t);
el1=sum(A(1,:));
ell=ell.*(ell<1)*(ell>0);

C.2.3 Functions for Sampling

----------- Function f2_rand.m to sample from zero-variance density  ----------

%%
% Gibbs sampling from f2: zero-variance density

function X=f2_rand(gamma, alpha, d, n)

x=ones(1,d);
s=sum(x.^(1/alpha)); X=NaN(n,d);
for i=1:n
    for j=1:d
        s=s-x(j)^(1/alpha);
        x(j) = -log(rand) + max(0,(gamma-s))”alpha;
        s=s+x(j)^(1/alpha);
    end
    X(i,:) = x(randperm(d));
end
X = sort(X,2);

C.2.4 Functions for Optimization

----------- Function myfun.m to evaluate the objective function D(z)  ----------

71
% Objective function
function [fval,gradval]=myfun(z,lam)
    global Coeff
    z=z(:)';
    [s,n]=size(Coeff);
    y=exp(z)*Coeff;
    fval=sum(log(y))/n-lam*z';
    if nargout>1
        gradval=exp(z').*sum(Coeff./repmat(y,s,1),2)/n-lam';
    end

% Function mycon.m to specify optimization constraints
function [c,ceq,gradc,gradceq] = mycon(z,lam)
    c=z(2)-z(1)-(log(lam(2))-log(lam(1)));
    ceq=sum(lam.*z);
    if nargout > 2
        gradc = [-1;1];
        gradceq = [lam(1);lam(2)];
    end

% Function hessianfcn.m to compute Hessian of objective function
function out=hessianfcn(z,Lagr_mult)
    global Coeff
    z=z(:)';
    [s,n]=size(Coeff);
    B=exp(z)*Coeff;
    y=Coeff./repmat(B,s,1);
    out=diag(exp(z').*sum(y,2))-(y*y').*(exp(z')*exp(z));
    out=out/n;
Bibliography

[1] Søren Asmussen, Klemens Binswanger, and Bjarne Højgaard. Rare events simulation for heavy-tailed distributions. *Bernoulli*, 6(2):303–322, 2000.

[2] Søren Asmussen and Peter W. Glynn. *Stochastic Simulation, Algorithms and Analysis*. Springer, 2007.

[3] Søren Asmussen and Dirk P. Kroese. Improved algorithms for rare event simulation with heavy tails. *Advances in Applied Probability*, 38(2):545–558, 2006.

[4] Zdravko I. Botev, Pierre L’Ecuyer, and Bruno Tuffin. Markov chain importance sampling with applications to rare event probability estimation. *Statistics and Computing*, pages 1–15, 2012.

[5] Devroye. *Non-Uniform Random Variate Generation*. Springer, 1986.

[6] Paul Embrechts and Noel Veraverbeke. Estimates for the probability of ruin with special emphasis on the possibility of large claims. *Insurance: Mathematics and Economics*, 1(1):55–72, 1982.

[7] Marnix J.J. Garvels. *The splitting method in rare event simulation*. Universiteit Twente, 2000.

[8] Richard D. Gill, Yehuda Vardi, and Jon A. Wellner. Large sample theory of empirical distributions in biased sampling models. *The Annals of Statistics*, pages 1069–1112, 1988.

[9] Paul Glasserman, Philip Heidelberger, Perwez Shahabuddin, and Tim Zajic. Multilevel splitting for estimating rare event probabilities. *Operations Research*, 47(4):585–600, 1999.

[10] Lars P. Hansen. Large sample properties of generalized method of moments estimators. *Econometrica: Journal of the Econometric Society*, pages 1029–1054, 1982.

[11] W. Keith Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.
[12] Sandeep Juneja and Perwez Shahabuddin. Rare event simulation techniques: An introduction and recent advances. *Simulation, Handbooks in Operations Research and Management Science*, pages 291–350, 2006.

[13] Leo P. Kadanoff. More is the same: phase transitions and mean field theories. *Journal of Statistical Physics*, 137(5-6):777–797, 2009.

[14] Dirk P. Kroese, Reuven Y. Rubinstein, and Peter W. Glynn. The cross-entropy method for estimation. *Handbook of Statistics, edited by V. Govindaraju and CR Rao*, 31, 2010.

[15] Dirk P. Kroese, Thomas Taimre, and Zdravko I. Botev. *Handbook of Monte Carlo Methods*, volume 706. Wiley, 2011.

[16] Pierre L’ecuyer, Jose H. Blanchet, Bruno Tuffin, and Peter W. Glynn. Asymptotic robustness of estimators in rare-event simulation. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 20(1):6, 2010.

[17] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21:1087, 1953.

[18] Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the Monte Carlo method*, volume 707. Wiley, 2011.

[19] Reuven Y. Rubinstein and Benjamin Melamed. *Modern simulation and modeling*, volume 7. Wiley, 1998.

[20] Yehuda Vardi. Empirical distributions in selection bias models. *The Annals of Statistics*, pages 178–203, 1985.