CHAPTER - V
A. INTRODUCTION

The economic behavior is often handled through game theoretical techniques. Major contributions in this area are of hybrid nature. Here we present game theoretical tools coupled with genetic algorithm to solve share trading time series problems. Due to huge amount of data to be analyzed in our study, we compress the data to a manageable level by suitable sampling techniques along with sampling weights. Main issue in statistical design and analysis is the estimate of parameters. Here we apply standard estimation procedures in our study.

B. GAME THEORETICAL TOOLS

Many practical problems require decision making in a competitive situation. It is helpful when two or more individuals or organizations with conflicting objectives try to make decisions. In such situations decision made by one decision maker affects the decision made by one or more of the remaining decision makers. This theory is applicable to a wide variety of situations such as two players struggling to win at chess, candidates fighting on election, firms struggling to maintain their market shares, etc. It is based on the minimax principle put forward by Von Neumann [82] which implies that each competitor will act so as to minimize his maximum loss or maximize his minimum gain.

A game, in which the sum of payments to all the players, after the play of the game is zero, is called zero-sum game. Here the gain of players that win is exactly equal to the loss of players those lose. If the number of players in a zero-sum game is two, it is known as two-person zero-sum game. The gains resulting from a two-person zero-sum game can be represented in the matrix form, usually called payoff matrix.
Consider two players. We can adopt two strategies—defects and co-operate. This is identical with a problem in game theory which we have four possibilities in which these two players can play given by the following table:

<table>
<thead>
<tr>
<th>Player – 2</th>
<th>Co-operate</th>
<th>Defect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Co-operate</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Defect</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 5.1: A typical payoff matrix.*

This is in accordance with Iterate Prisoner's dilemma problem. The Iterated Prisoner's dilemma problem was studied by Axelrod [3, 4] and programmed by Forrest [28]. The prisoner's dilemma is a classic; some might say the archetypal, problem of conflict and cooperation. In its simplest form, each of two players has a choice of cooperating with the other or defecting. Depending on the two players' decisions each receives payoff according to a payoff matrix similar to the one shown in figure 5.2. When both players cooperate they are both rewarded at an equal intermediate level (the reward, $R$). When only one player defects, he receives the highest level payoff (the temptation, $T$) while the other player gets the sucker's just deserts (the sucker, $S$). When both players defect they each receive an intermediate penalty (the penalty, $P$).

The prisoner's dilemma has often been cited as a simple yet realistic model of the inherent difficulty of achieving cooperative behavior when rewards are available for the successful miscreant. The problem is called the prisoner's dilemma because it is an abstraction of the situation felt by a prisoner who can either cut a deal with the prosecutor and thereby rat on his partner in crime (defect) or keep silent and thereby tell nothing of the misdeed (cooperate).
The problem is made more interesting by playing it repeatedly with the same player or group of players, thereby permitting partial time histories of behavior to guide future cooperation-defection decisions. This so-called iterated prisoner's dilemma has drawn interest from game theorists for a number of years. Computer tournaments [4] have pitted different computer procedures against one another in two recent round robin contests. In both contests, a very simple strategy called "tit for tat" was the overall winning among 76 total entrants. As the name implies, tit for tat simply cooperates on the first move and then does whatever its opponent did on the previous move. To do this we appeal to Genetic algorithm.

The decision rules were set to depend upon the behavior of both parties during the previous three moves. On each of these moves there are, of course, four possibilities; both players can cooperate (CC or R for reward), the other player can defect (CD or S for sucker), the first player can defect (DC or T for temptation), or both players can defect (DD or P for penalty). To code a particular strategy, we code the particular behavioral sequence as a three-letter string.

For example, RRR would represent the sequence where both parties cooperated over the three moves and SSP would represent the sequence where the first player was played for a sucker twice and finally defected. The three letter sequence was then used to generate a number between 0 and 63 by treating the code as an integer base 4 where the behavioral alphabet is decoded in the following way.

\[
CC = R = 0, \quad DC = T = 1, \quad CD = S = 2, \quad DD = P = 3.
\]
In this way for example three mutual defections \((PPP)\) would decode to a 63. Using the coding it is possible to define a particular strategy (over the past three moves) as a 64 bit binary string of \(C\)'s and \(D\)'s where the \(i^{th}\) \(C\) or \(D\) corresponds to the \(i^{th}\) behavioral sequence. Using this sequence for example, a \(D\) in position 0 would be decoded as a rule of the from \(RRR \rightarrow D\) and a \(C\) in the third position would be decoded as a rule of the form \(RRP \rightarrow C\).

Actually, the situation is somewhat more complex than stated. Since the set of rules generated by a 64 bit string depends upon the past three plays; behavior at the game's beginning is indeterminate. To get around these problems six bits (Six \(C\)'s and \(D\)'s) were added to the coding to specify a strategy's premises or assumptions about pregame behavior. The six bits were used sequentially to simply specify the assumed behavior of both players prior to the game beginning. In this way the normal rules could be used in conjunction with the premises to specify opening game as well as middle game behavior. Together each of the 70 bit strings thus represented a particular strategy with 64 bits for the rules and six bits for the premises.

In our problem every position in the string will identify the reward, penalty, temptation, and sucker. We get the point of cross over between two strings in different possibilities until we identify sucker.

The position of cross over in the string having the value 'S' gives the minimum value for the fitness of the string.

C. ADVANCED COMPUTING THEORY

The evaluation of distributions of random sums or compound processes has been ubiquitous in actuarial mathematics for many decades. It typically arises as the preferred method of modelling the probability of ruin and the loss distributions for insurance claims that can be assumed to arrive according to, for example, a Poisson or negative binomial compound process. That is, they are typically considered when modelling the distribution of the total claims incurred.
in a fixed period of time. What makes the explicit computation of these loss
distributions difficult is that the conditional distribution of the amount of total claim
or loss given a certain number of claims \( m \) has occurred involves an \( m \)-fold
convolution of the severity distribution.

As alluded to in the introduction, Monte Carlo simulation is usually
employed to approximate these \( m \)-fold convolutions. However, at the extreme
values the mean annual number of events is large or small the standard Monte
Carlo approach becomes extremely computationally inefficient. Additionally, for
any mean arrival rate, if one is trying to achieve a given accuracy for an estimate
of Value at Risk (VaR) or Expected Shortfall (ES), the computational effort
required to obtain accurate tail estimates of the annual loss distribution can be
significant. Below, we propose a simulation procedure which helps to reduce this
computational burden.

In Operational Risk one is typically concerned with rare and infrequent
events which, if they do occur, can have catastrophic consequences for the annual
loss of a given year. This typically corresponds to the situation in which the mean
annual loss is small and the mean severity of the given losses is very large. Here we
will focus on the most important case for Operational Risk and that is the
infrequent yet catastrophic event situations. We will demonstrate that our
approach is a very efficient means of accurately estimating VaR (a tail quantile of
the loss distribution) and \( ES \) deep in the tails of the annual loss distribution.
We will additionally point out that given we know the starting point \( y_s \) for the
\( \text{VaR}_\alpha(Y) = y_s \) at which one wants to calculate the expected shortfall,

\[
ES_\alpha(Y) = \frac{1}{1 - \alpha} \int \text{VaR}_\alpha(Y) du
\]

\[
= E_{y_s}[Y | Y > y_s]
\]

\[
= \frac{1}{1 - E_{y_s}(y_s)} \int_y^\infty y f_Y(y) dy
\]

In such cases our method provides a solution without computing the
annual loss distribution for the domain $[0, y_s]$. In other words our procedure begins with the estimation of the annual loss distribution on $[y_s, \infty)$. ES is an important measure of risk since it has the property of coherence, which is not the case for VaR.

In practice we may need to calculate a VaR$_\alpha(Y)$ to get $y_s$ first before calculation of $ES$, in these cases we present a fast and efficient algorithm to perform both the calculation of $y_s$ and the calculation of $ES$. However, there will be some cases in which the value of $y_s$ is known in advance. The initial value of $y_s$ could be known from a previous VaR calculation.

Additionally computation of $ES$ without the construction of the entire loss distribution can be valuable for insurance purposes. If a haircut is known to occur after a level $y_s$ then one may be interested in efficiently calculating the Expected loss ignoring the insurance policy and then calculating the Expected loss including the insurance deductions and comparing the excess to the uninsured expected loss to decide between insurance policies.

Our target is to evaluate the compound distribution for the annual loss which is described by equation

$$Y = \sum_{i=1}^{M} X_i.$$  \hspace{1cm} \text{...5.1}

In these situations actuarial techniques can prove to be effective means of evaluating an annual loss distribution point-wise. The approach we will review here is the most popular of these known as the Panjer recursion.

If the severity distributions are discrete, then efficient, deterministic techniques based upon the z-transform may be employed but this approach does not generalize to continuous severity distributions. Alternatively, the Panjer recursion provides a recursive expression for evaluation of the coefficients $c_k$ of the probability generating function $P(z)$.

Although, in some settings, discretisation of a continuous severity distribution might be justifiable, this is not the preferred approach to most Operational Risk models. However, the Panjer recursion approach mentioned
above may also be applied in a continuous setting, leading to the recursion:

\[ f_Y(x) = p_I f_x(x) + \int_0^x \left( a + \frac{by}{x} \right) f_x(y) f_Y(x-y) \, dy \]  ...5.2

where \( a, b \) and \( p_I \) parameterize the frequency distribution of the compound process. There are many approaches to evaluate this expression. The property of the Panjer recursion that we will exploit in developing our simulation algorithms is that equation 5.2 can be recognized as a Volterra equation of the second kind Panjer [54], Wilmott [85]. In general the Volterra integral equation of the second kind takes the form

\[ f(x) = g(x) + \int_0^x K(x, x_1, f(x_1)) \, dx_1. \]

In the case of the Panjer recursion we have a linear Volterra equation in which

\[ K(x, x_1, f(x_1)) = k(x, x_1) f(x_1) \]

This gives

\[ f(x) = g(x) + \int_0^x k(x, x_1) f(x_1) \, dx_1 \]  ...5.3

allowing us to make explicit the association between the Volterra equation of the second kind and the Panjer recursion. To do this we make the following identifications,

\[ x_1 = x - y \]

\[ g(x) = p_I f_x(x) \]

\[ k(x, x_1) = \left( a + b \frac{x-x_1}{x} \right) f_x(x-x_1) \]

\[ f(x_1) = f_Y(x_1) \]

Working with the Volterra integral equation of the second kind we can obtain the following representation,

\[ f(x) = g(x) + \int_0^x K(x, x_1) f(x_1) \, dx_1 \]
and we recognize that this equation can be represented also as,

\[ f(x) = g(x) + \int_0^x r(x, x_1) g(x_1) dx_1 \]

in which \( r \) is the resolvent kernel for the Volterra equation of the second kind which, under the condition given below, may be expressed as the following Von Neumann series expansion

\[ r(x, x_1) = \sum_{n=1}^{\infty} k^n(x, x_1) \]

where

\[ k^1(x, x_1) = k(x, x_1) \quad \text{and} \]

\[ k^n(x, x_1) = \int_0^x k(x, u) k^{n-1}(u, x_1) du \quad n = 2, 3, 4 \ldots \]

Applying this series expansion to equation 5.3 gives,

\[ f(x_0) = g(x_0) + \sum_{n=1}^{\infty} \int_0^{x_{0-1}} \cdots \int_0^{x_n} g(x_n) \prod_{i=1}^{n} k(x_{i-1}, x_i) dx_{i-1} \]

where we use the notation \( x_{\ell,n} = (x_1, \ldots, x_n) \).

In order to simplify expressions, it is useful to define the following notation to describe the domain of integration. The conditional one-dimensional domains of integration are defined by, \( D_k(x_{k-1}) = [0, x_{k-1}] \), and we define the domain of integration of the \( n^{th} \) term in the summation as:

\[ D_{\ell,n}(x_0) = \{(x_1, \ldots, x_n) : x_0 > x_1 > \ldots > x_n\} \]

adopting the convention that \( D_{1,0}(x_0) = \{\phi\} \).

Doing so allows us to write the previous expression in the form:

\[ f(x_0) = g(x_0) + \sum_{n=1}^{\infty} \int_{D_{\ell,n}(x_0)} \prod_{i=1}^{n} k(x_{i-1}, x_i) g(x_n) dx_{i-1} \]
with this representation valid whenever the right hand side is finite. We also define

$$\tilde{D}(D_0) = \{(x_0, x_1, \ldots, x_n) : D_0 \ni x_0 \geq x_1 \geq \cdots \geq x_n \},$$

where $D_0$ corresponds to a region of values over which we wish to characterize annual loss distribution (typically an interval $[x_a, x_b]$), for later use.

To conclude this section we shall mention some of the approaches that have to be proposed for evaluation of Volterra equations of the second kind and related fixed domain problems, the Freedholm integral equation of the second kind. In general this is a large and diverse literature spanning many different disciplines. The most commonly used approaches include quadrature methods for solving the integrals in the Panjer recursion, Runge-Kutta methods, Collocation and Galerkin methods which are based on polynomial splines or piecewise-polynomial densely defined approximations and also importance sampling techniques.

Our approach can most easily be associated with the importance sampling approach. In particular we utilize some concepts from Doucet [20] to interpret the standard Von Neumann expansion of the Panjer recursion as an expectation with respect to a probability distribution defined on a union of subspaces of variable dimension. We then utilize both importance sampling and trans-dimensional Markov Chain Monte Carlo algorithms to simulate from the density with which the expectation is defined.

Under this framework we develop two novel algorithms for simulation of an annual loss distribution, we present consistent and unbiased estimators for both point-wise and interval estimates of the annual loss distribution when evaluating VaR and also calculation of $ES$.

**Algorithm 1:** Importance Sampling for Panjer Recursions [54]:

1. Simulate $N$ independent Markov Chain paths $\{X_{0,n}^{(i)} \}_{i=1,N}$ until absorption, where $X_{\pi(k)}^{(i)} = d$.
2. Calculate Importance Sampling Weights.
If evaluation of the annual loss density at a point is desired, at the value $x_0$, then this weight is given by:

$$W(X_{0:n^{(i)}}) = \begin{cases} \prod_{s=1}^{n^{(i)}} \frac{k(X_{s-1}^{(i)}, X_s^{(i)})}{M(X_{s-1}^{(i)}, X_s^{(i)})} \frac{g(X_{n^{(i)}}^{(i)})}{P_d} & n^{(i)} \geq 1 \\ \frac{1}{\mu(X_0^{(i)})} \frac{g(X_0^{(i)})}{P_d} & n^{(i)} = 0. \end{cases}$$

Whilst, if $X_0$ is being sampled from some distribution $\mu$ in order to characterize $f$ over some interval, then the importance weight function becomes:

$$W(X_{0:n^{(i)}}^{(i)}) = \begin{cases} \frac{1}{\mu(X_0^{(i)})} \prod_{s=1}^{n^{(i)}} \frac{k(X_{s-1}^{(i)}, X_s^{(i)})}{M(X_{s-1}^{(i)}, X_s^{(i)})} \frac{g(X_{n^{(i)}}^{(i)})}{P_d} & n^{(i)} \geq 1 \\ \frac{1}{\mu(X_0^{(i)})} \frac{g(X_0^{(i)})}{P_d} & n^{(i)} = 0. \end{cases}$$

iii). If one is interested only in evaluating the annual loss distribution point wise at $x_0$, then we have the estimate,

$$\hat{f}(x_0) = \frac{1}{N} \sum_{i=1}^{N} W(x_0, X_{0:n^{(i)}})$$ \hspace{1cm} \ldots 5.4$$

Otherwise, if approximating the annual loss distribution over some interval, such as when one is interested in calculation of $ES$, use the empirical estimate given by

$$\hat{f}(x_0) = \frac{1}{N} \sum_{i=1}^{N} W(X_{0:n^{(i)}}) \delta_{X_0^{(i)}}(x_0)$$ \hspace{1cm} \ldots 5.5$$

where $\delta_{X_0^{(i)}}$ is the Dirac-delta mass located at $X_0^{(i)}$.

These algorithms provide an unbiased point wise estimate equation 5.4 of the annual loss at $x_0 = x$ and the empirical estimate equation 5.5 can be used to compute an unbiased estimate of the integral of any test function with respect to $f(x)$ over an interval $D_0$. 

115
Equations 5.4 and 5.5 can be seen to be an importance sampling procedure; equation 5.4 exists on the space $\bigcup_{n=0}^{\infty} \{n\} \times \hat{D}_{t,n}(x_0)$ and equation 5.5 on the space $\bigcup_{n=0}^{\infty} \{n\} \times \hat{D}_{0,n}(D_0)$. In the first case the importance sampling distribution takes the form $p(n, n_{t,n}) = p(n) p_n(x_{t:n})$ with
\[
p(n) = P_r(X_{t:n} \in D_{t:n}(x_0), X_{n+1} = \{d\}) = (1 - P_d)^n P_d
\]
and
\[
p_n(x_{t:n}) = \frac{M(x, x_i) \prod_{k=2}^{n} M(x_{k-1}, x_k)}{(1 - P_d)^n}
\]
and the changes required to obtain the distribution used in the second case are obvious.

Hence, we have associated the original Panjer recursion with an expectation and then formulated an algorithm to provide an unbiased and consistent approximation of this annual loss distribution given by either equation 5.4 or equation 5.5.

Although we have successfully demonstrated one mechanism for obtaining the expectation of interest via importance sampling, it is known that whenever importance sampling is used, it is important to employ a good proposal distribution which leads to an estimator of low variance. If the Monte Carlo variance of the importance weights is large, then it will not be an efficient means of estimating the integrals comprising the expectation. As argued in Doucet [20], this can be difficult to enforce when using importance sampling on the path space, commonly known as sequential importance sampling. We will consider a principled approach to choosing an importance function, and to obtaining samples with this distribution.

**Algorithm 2: Reversible Jump Markov Chain Monte Carlo for Importance Distribution Initialization:**

i). For $i = 1$ set $(n^{(i)}, X_{0:n}^{(i)})$ deterministically as $n^{(i)} = 1$ and $X_{0}^{(i)} = x$.
ii). Update Move:

Step 1: Set \( n^{(i)} = n^{(i-1)} \).

Step 2: Sample uniformly index \( J \sim U\{1, \ldots, n^{(i)}\} \).

Step 3: Sample proposed update for \( J^{th} \) element, \( X^*_j \sim q_u(X^{(i-1)}_J) \).

Step 4: Evaluate the acceptance probability (note that \( X^{(i-1)}_{0:n^{(i)} \setminus J} \) should be interpreted in the natural manner as \( (X^{(i-1)}_0, X^{(i-1)}_{J+1:n^{(i)}}) \))

\[
\alpha((n^{(i)}, X^{(i-1)}_{0:n^{(i)}}, X^{(i-1)}_{0:n^{(i)} \setminus J}, X^*_j)) = \min\left\{ 1, \frac{P_{opt}(n^{(i)}, X^{(i-1)}_{0:n^{(i)} \setminus J}, X^*_j) H_u(X^*_j, X^{(i-1)}_J)}{P_{opt}(n^{(i)}, X^{(i-1)}_{0:n^{(i)}}, X^*_j) H_u(X^{(i-1)}_J, X^*_j)} \right\}
\]

Step 5: Sample uniform random variate \( U \sim U[0,1] \)

Step 6: If \( U \leq \alpha((n^{(i)}, X^{(i-1)}_{0:n^{(i)}}, X^{(i-1)}_{0:n^{(i)} \setminus J}, X^*_j)) \) then set

\( (n^{(i)}, X^{(i-1)}_{0:n^{(i)}}) = (n^{(i)}, X^{(i-1)}_{0:n^{(i)} \setminus J}, X^*_j) \).

Otherwise set

\( (n^{(i)}, X^{(i-1)}_{0:n^{(i)}}) = (n^{(i)}, X^{(i-1)}_{0:n^{(i)}}) \)

iii). Sample uniform random variate \( U \sim U[0,1] \)

If \( U < p_b \) then

iv). Birth Move:

Step 1: Sample an index uniformly to add a new component at

\( J \sim U\{1, \ldots, n^{(i)} + 1\} \)

Step 2: Sample new component's value \( X^*_j \sim q_b(\cdot) \).

Step 3: Evaluate the acceptance probability

\( X^{(1)}_i = x / 2 \), Repeat for \( i \geq 1 \).
\[ \alpha \left( \left( n^{(i-1)}, X_{0,n}^{(i-1)} \right), \left( n^{(i-1)} + 1, X_{0,n}^{(i-1)}, X^*_j \right) \right) = \min \left\{ \frac{P_{opt} \left( n^{(i-1)} + 1, \left( X_{0,j-1}, X^*_j, X_{j+1:n}^{(i-1)} \right) \right) P_d}{P_{opt} \left( n^{(i-1)}, X_{0,n}^{(i-1)} \right) q_b \left( X^*_j \right) P_b} \right\} \]

Step 4: If \( t/ \leq \alpha \left( \left( n^{(i-1)}, X_{0,n}^{(i-1)} \right), \left( n^{(i-1)} + 1, X_{0,n}^{(i-1)}, X^*_j \right) \right) \), then set
\[ \left( n^{(i)}, X_{0,n}^{(i)} \right) = \left( n^{(i-1)} + 1, \left( X_{0,j-1}, X^*_j, X_{j+1:n}^{(i-1)} \right) \right) \]
 Otherwise set
\[ \left( n^{(i)}, X_{0,n}^{(i)} \right) = \left( n^{(i-1)}, X_{0,n}^{(i-1)} \right) \]

else.

v). Death Move:

Step 1: Sample an index uniformly to delete an existing component
\[ J \sim U \{ 1, \ldots, n^{(i-1)} \} \]

Step 2: Evaluate the acceptance probability
\[ \alpha \left( \left( n^{(i-1)}, X_{0,n}^{(i-1)} \right), \left( n^{(i-1)} - 1, X_{0,n}^{(i-1)} \right) \right) = \min \left\{ \frac{P_{opt} \left( n^{(i-1)} - 1, \left( X_{0,j-1}, X_{j+1:n}^{(i-1)} \right) \right) q_b \left( X_{j}^{(i-1)} \right) P_b}{P_{opt} \left( n^{(i-1)}, X_{0,n}^{(i-1)} \right) P_d} \right\} \]

Step 3: If \( t/ \leq \alpha \left( \left( n^{(i-1)}, X_{0,n}^{(i-1)} \right), \left( n^{(i-1)} - 1, X_{0,n}^{(i-1)} \right) \right) \), then set
\[ \left( n^{(i)}, X_{0,n}^{(i)} \right) = \left( n^{(i-1)} - 1, \left( X_{0,j-1}, X_{j+1:n}^{(i-1)} \right) \right) \]
 Otherwise set
\[ \left( n^{(i)}, X_{0,n}^{(i)} \right) = \left( n^{(i-1)}, X_{0,n}^{(i-1)} \right) \]

vi). If \( i < M \), go to (ii).

The only two quantities we need to specify to apply Algorithm 2 to approximate a particular annual loss distribution, is the Markov transition kernel that will be used for the within subspace moves \( q_u \left( X_{j}^{(i-1)}, x \right) \) and also the
distribution for the birth proposal $q_p$. Here $q_w (X^{(i-1)}_j , x)$ denotes the probability of updating the $J^{th}$ element of the current state of the Markov Chain, $(X^{(i-1)}_{k:d-1})$, from $X^{(i-1)}_j$ to some value $x \in D_j \left(X^{(i-1)}_j \right)$. Additionally, the notation $q_b$ denotes the probability density from which the new proposed birth component will be sampled, when proposing to move from subspace $D_{1:n} (x_0)$ to $D_{1:n+1} (x_0)$.

We will utilize a symmetric Gaussian kernel for the within subspace moves leading to a Random Walk Metropolis (RWM) algorithm. For the birth move we now present the birth proposal distribution which is optimal in the sense that it minimizes the variance of the ratio within the acceptance probability of a birth and death move. This will be achieved if we recognize that we can always make the following factorization,

$$P_{\text{opt}} (X_{1:n+1}) = P_{\text{opt}} (x_{n+1} | x_{1:n}) P_{\text{opt}} (x_{1:n}).$$

Then we can easily see that the proposal for the birth move minimizing the variance of the ratio appearing in the acceptance probability is given by $P_{\text{opt}} (x_{n+1} | x_{1:n})$. In our setting this can be shown to take the form,

$$P_{\text{opt}} (x_{n+1} | x_{1:n}) = \frac{f_X (x_{n+1}) }{f_X (x_n)} k(x_n, x_{n+1}) \propto f_X (x_{n+1}) \left( a + b \frac{x_n - x_{n+1}}{x_n} \right) f_X (x_n - x_{n+1}),$$

for $x_{n+1} \in [0, x_n]$. ...5.7

Hence, we propose to sample from an approximation to this distribution using a simple empirical cdf estimate. We construct a fast, crude estimate over a uniform grid using a right end-point rule, using 20 points to construct the piecewise estimate. Alternatively, one could use rejection sampling since $P_{\text{opt}} (x_{n+1} | x_{1:n})$ is supported on $[0, x_n]$ and typically the severity distribution, have an analytic bounding distribution.

The approximation of the annual loss distribution, once we have drawn samples $\{n^{(i)}, X^{(i)}_{j,l,d} \}$ approximately distributed as the optimal importance sampling distribution, is given by Doucet [20].
\[ \hat{f}(x) = f_0(x) + \hat{c}_{opt}. \]  

To perform this calculation one can approximate the optimal normalizing constant as,

\[ \hat{c}_{opt} = \frac{\hat{c}_{1, opt}}{\hat{p}_{\lambda, opt}}, \]

providing that it is possible to estimate \( c_{1, opt} \). This can be done in advance and we utilize a right end point trapezoidal rule, but any numerical integration scheme could be used. We obtain the estimate of \( p_{1, opt} \) as the proportion of the total number of states explored by the Markov chain which lie in \( D_{1;1}(x_0) \).

Summarizing this, we have developed machinery to allow us to obtain accurate estimates of an annual loss distribution. This was achieved by importance sampling in the case of Algorithm 1 and the methodology to sample from a more principled importance distribution in Algorithm 2.