# Modeling Loss Emergence and Settlement Processes

CAS Loss Simulation Model Working Party Summary Report

**Abstract:** The CAS Loss Simulation Model Working Party (LSMWP) has been charged by the Committee on Dynamic Risk Modeling Committee with creating a simulation model of the processes of loss emergence and settlement, commonly known as loss development, that underlie the loss "triangles" and other statistics used to estimate loss reserves. The goal was to create a tool that researchers could use to generate claims that would be summarized into loss development triangles and complete rectangles which could then be used to test loss reserving methods and models.

**Motivation.** Actuaries need tools that will enable them to better understand the underlying loss development process and will aid them in determining what methods and models work best in different reserving situations.

**Method**. The LSMWP first developed a prototype model that met its basic objectives. It then engaged a consultant to develop an open-source model that could be applied and further developed by the CAS membership.

**Results**. The open-source model developed by the consultant is documented in this paper, along with the testing which validated the model.

**Conclusions**. A valuable tool has been created for use by the actuarial community in its research work on reserving methods and models. We encourage actuaries to perform additional validation tests of the model and contribute model enhancements.

**Availability**. This report and the associated model are available by clicking on the LSMWP link on the Committee on Dynamic Risk Modeling (DRM) page on the CAS Web Site (www.casact.org/research/drm).

**Keywords:** loss reserving; simulation model; reserve variability; reporting pattern; payment pattern; open-source model.

## TABLE OF CONTENTS

# 1. INTRODUCTION

The work of the LSMWP evolved in the following stages:

- Survey of existing literature and preparation of an appropriate bibliography. This bibliography includes papers that provide guidance concerning how alternative reserving methods and models may be tested.

- Develop methods of testing simulated data to assure the data generated represent "real" results that could be produced by a company and, therefore, used in testing the relative worth of various reserve models.

- Develop prototype model that has key features desired in final open-source model.

- Develop and document open-source model through work with a consultant.

- Test open-source model by evaluating output to determine if it could be distinguished from real data and if it is consistent with actuarial assumptions.

- Develop procedures to enable users to develop model parameters that fit their own data.

The model testing and the procedures for fitting the model to a user's own data are contained in two sections. The model generates both detailed claim transaction data and aggregate data in the form of "triangles." Section 3 discusses testing of the aggregate data features. Section 6 tests the detailed data and also describes procedures for fitting the model to actual detailed data.

The sections below elaborate on the above developmental stages.

## 1.1 Research Context

Actuaries need tools that will enable them to better understand the underlying loss development process and will aid them in determining what methods and models work best in different reserving situations.

## 1.2 Objective

Our objective is to make a valuable tool available for use by the actuarial community in its research work on reserving methods and models. We encourage actuaries to perform additional validation tests of the model and contribute model enhancements.

## 1.3 Disclaimer

While this paper is the product of a CAS Working Party, its findings do not represent the official view of the Casualty Actuarial Society. Moreover, while we believe the approaches we describe are very good examples of how to address the issue of loss development, we do not claim they are the only acceptable ones.

While we made a reasonable effort to test and validate the new open-source model, users should do their own independent testing and validation. We cannot assure users that the model is completely valid or free from error. We look forward to working with users to correct any errors and enhance it so that it is more useful to the actuarial community.

## 1.4 Outline

The remainder of the paper proceeds as follows. Section 2 will discuss our survey of existing literature and present our bibliography. Section 3 will discuss our recommended method of testing simulated data to assure the data generated represent "real" results that could be produced by a company and, therefore, used in testing the relative worth of various reserve models. Section 4 will present the model features that were developed in the prototype model and ultimately programmed in the new open-source model. Section 5 will present summary features and documentation of the new open-source model. Finally, Section 6 will present test results of the new open-source model, both evaluating output to determine if it is consistent with actuarial assumptions and fitting data to common distributions. Section 7 will present a list of potential applications of the model as well as model enhancements that we hope the actuarial community will undertake, and Section 8 will present our conclusions.

# 2. SURVEY OF EXISTING LITERATURE

As we began our survey work, we decided to associate our findings with four basic categories of articles that might prove of value to those who ultimately use the model developed from the LSMWP efforts:

General Interest

Building Models

Testing Simulated Data

Testing Reserve Methods

Our final list of readings for consideration is provided as the bibliography at the end of this paper. While we do not represent this to be an exhaustive list, we think we have a good balance of the four areas.

In most cases, the articles suggested reference additional publications for consideration of the interested reader as she or he delves deeper into one or more aspects of the issues associated with development of a model generating loss data and testing the simulated output from that model.

## 3. STATISTICAL TESTS OF SIMULATED MODEL OUTPUT

We began by asking questions that one would ask when inspecting the output from any model supposedly representing potential loss results for a line of business:

- Do aggregate results appear reasonable in terms of losses (loss ratios) generated in a given year and the distribution of losses (loss ratios) over multiple years?
- Do incremental changes appear "logical" with respect to the line of business being modeled?
- Do the distributions (claim count, severity, etc.) produced appear "reasonable?"

While these questions would properly be asked by someone with reserving experience when presented with real data by line of business, this type of review lacks the level of objective analysis required to use data generated from a model for the purpose of evaluating the value of different reserving models (methods).

Depending on the amount of real data available, many modelers suggest holding back data for use in testing a model. In this approach, a model is parameterized based on a subset of the total actual data, then tested against the unused portion of data. This "control" data is compared with what comes out of the model to determine if the model "accurately" represents the real world.

The ability to apply the "control data" approach depends on the amount and granularity of data being modeled. Unfortunately, it is often difficult to amass enough data to apply this approach. Therefore, modelers have on occasion "created" more control data. Fundamentally, this is done by building a set of data or supplementing it with randomly selected data elements from the initial set of data available to the modeler. There are a host of potential pitfalls with this approach, not the least of which is making certain you have, in fact, generated a randomly selected set of data, but it does have its merits.

After reviewing the discussion of statistical testing methods in our bibliography, we chose to recommend a testing approach brought to our attention by Glenn Meyers. Previously, we had received input from experts in model building suggesting we test the simulated data against actual data by comparing the differences at various incremental points of development. If these differences were randomly distributed with a mean of zero, then one could conclude the simulated data would serve as a good proxy for real data.

Mr. Meyers has authored two papers documented in the literature search results that provide detailed explanation of a way to assess the ability of data from a model to represent "real" data: "Estimating Predictive Distributions for Loss Reserve Models" and "Thinking Outside the Triangle."

In the following, we will provide both general and specific insights gained while working with the Meyers approach, along with some more general thoughts related to testing of data.

## 3.1 PP-Plots

A commonly used approach to testing data output from a model is to use the Kolmogorov-Smirnov (K-S) test. This test is described in the book, *Loss Models: From Data to Decisions*, by Klugman, Panjer, and Willmot[1] and in Wikipedia.

An approach that is similar to graphically analyzing a sample of data output from the model is to construct PP-Plots. The points $\left(\frac{i}{n+1}, F_i\right)$ are plotted for $i$=1,…,$n$. One evaluates whether these points are within a fixed percentage of the 45° line. This is the approach recommended and documented by Meyers in his papers and subsequently adopted by the LSMWP in its tests of the open-source loss simulation model developed for the CAS. We recommend the Meyers papers cited above for the insights shared on interpreting PP-Plots.

## 3.2 The Meyers Approach

The approach taken by Meyers was as follows:

- Starting with a real set of data, a model is constructed which can be used to generate accident year payments by settlement lags. It is used to construct "typical" accident year (AY) triangles and rectangles (all AYs fully developed) we face during normal reserve reviews.
- Using the model, generate a significant set of simulated AY rectangles (500 in this example) for use in testing output from the model and how well it represents real data.
- Produce as additional output a set of simulated triangles (nine for this example) and randomly insert a real triangle to be tested with the others.

[1] [11] in bibliography.

- For this example, parameters for the model were selected to purposely NOT fit the real data. Therefore, we'd expect the real triangle to not pass a goodness-of-fit test, while the simulated triangles would do so.

The PP-Plot tests were performed as follows:

- A set of 500 simulated rectangles was generated with each simulation including 10 AYs. For each AY, premium and incremental paid losses for each of 10 settlement lags were calculated. (While the premium is not needed for our example, it was an integral part of the model, as it used an expected loss ratio to determine the total losses ultimately paid for an AY.)
- We used Excel commands to reformat the output into a summary of paid losses by settlement lag for the 5000 AYs provided.
- We are given the 10 candidates triangles (nine simulated and one real triangle) to test.
- Candidate data was reformatted, percentiles determined and then sorted for ease of output as PP-Plot graphs. (Note: For a given candidate, the percentile of the accident year emergence for each lag is calculated from the distribution of simulated loss emergences for the same accident year and lag. Also, for data points outside the percentile range determined by the 500 simulations, a value of zero is substituted.)
- Appendix C, Sheet 1-10: Candidate PP-Plots are provided. We think it's fairly obvious which of the candidate triangles represents the real data.

## 3.3 Additional Comments

An underlying tenet of the highlighted approach to testing the ability of a model to represent real data is: If real data passes the PP-Plot with the Kolmogorov-Smirnov (KS) test using data simulated by the model, then we can infer that a formula/method that works well (passes appropriate tests) for data from the loss simulation model will work well on real data. We are comfortable with this assumption but it should not be accepted without some reflection.

# 4. BASIC FEATURES IN THE PROTOTYPE MODEL

The prototype model contained the following basic features that were ultimately programmed in the new open-source model:

- (1) Observation period: We assume that the relevant loss process involves accidents or occurrences between dates $t_0$ and $t_1$. The simulator tracks transactions until accidents are settled.
- (2) Time intervals: We assume that parameters are constant throughout calendar months but may change from one month to next. Lags are measured in days.
- (3) Exposures: The user may specify a measure of exposure for each month. By default, the system assumes constant unit exposure. The purpose of the exposure parameter is to allow the user to account for a principal source of variation in monthly frequencies.

- (4) Events: Each claim may be described by the dates and amounts of the events it triggers: the accident date, the report date and an initial case reserve, zero or more subsequent valuation dates and case reserves changes, zero or one payment date and amount, and zero or one recovery date and amount.
- (5) Distributions: For most variables, the user may specify a distribution and associated parameters.
- (6) Frequency: The user may specify monthly claim frequency as a Poisson distribution with mean proportional to earned exposure, or as a Negative Binomial distribution with mean proportional to earned exposure and variance proportional to the mean frequency (which implies that the variance is also proportional to the earned exposure.
- (7) Report lag: The lag between occurrence and reporting is assumed to be distributed Exponential, Lognormal, Weibull, or Multinomial. The Multinomial distribution allows the user to define proportions of claims reporting within one month, two months, and so on.
- (8) The lags between reporting and payment, between one valuation date and the next, and between payment and recovery or adjustment, are also assumed to be distributed Exponential, Lognormal, Weibull, or Multinomial.
- (9) Size of loss: The actual size of the loss to the insured, independent of responsibility for payment, is distributed Lognormal, Pareto, or Weibull.
- (10) Case reserve factor: Case reserves are assumed to equal the actual size of loss, adjusted for the minimum, the maximum, the deductible, and the probability of closure without payment, all multiplied by an adequacy factor. This factor is assumed to be distributed Lognormal. The user may specify the mean factor at four points in time between the report and payment dates.
- (11) Fast-track reserve: A value may be assigned to each loss at first valuation, independent of regular case reserves and case reserve factor.
- (12) Initial payment factor: The initial payment of each loss not closed without payment is assumed to equal the actual size of loss, adjusted for the minimum, the maximum, the deductible, multiplied by a payment adequacy factor (PAF). The PAF determines the size of any subsequent adjustment or recovery.
- (13) Second-level distributions: The LSMWP models the drift in parameter values that may take place for many reasons but chiefly because of business turnover. It has developed an autoregressive model to reflect parameter drift.
- (14) Monthly vectors of parameters: For nearly all distributional parameters, the user may specify a single value or a vector of values.
- (15) Frequency Trend and Seasonality: The user may specify monthly trend and seasonality factors for frequency that are applied to means.
- (16) Severity Trend: The user may specify monthly trend factors for severity.
  - The "main" trend is allowed to operate up to the accident date and a fraction of this trend, defined by Butsic's "alpha" parameter, is allowed to operate between accident and payment dates.
  - Case reserves before the adequacy factor are centered on the severity trended to the payment date.
- (17) Lines and Loss Types: The prototype model recognizes that loss data often involves a mixture of coverages or loss types or both with quite different frequencies, lags, and

severities. Therefore, it allows the user to specify a two-level nested hierarchy of simulation specifications, with one or more "Lines" each containing one or more "Types."
  - A typical Line might be "Auto"; typical Types within that Line might be "APD," "AL-BI," and "AL-PD."
  - This hierarchy allows the user to set up any reasonable one- or two-level classification scheme.
  - Accident frequencies are modeled at the Line level and loss counts per accident are distributed among Types using a discrete distribution.
- (18) Lines and Loss Types Example: An Automobile occurrence might give rise to a single Auto Physical Damage (APD) claim with probability 0.4, to a single Auto Property Damage Liability (AL-PD) claim with probability 0.2, to a single APD and a single AL-PD claim with probability 0.2, to a single Auto Bodily Injury Liability (AL-BI) claim with probability 0.1, to two AL-BI claims with probability 0.05, etc.
- (19) Correlations: The prototype model makes it possible to request correlated samples of certain variables without fully specifying their joint distribution. These variables are (a) the mean frequencies across Lines and (b) the size of loss and report lag within a Type. To specify correlated frequencies among lines, the user specifies the marginal frequency distribution for each Line and then specifies a correlation matrix and copula to determine the joint distribution.
- (20) Clustering: The prototype simulator allows a selectable fraction of loss sizes and a selectable fraction of case reserves to be rounded to two significant digits, imitating clustering around round numbers frequently observed.
- (21) Output: The prototype simulator produces output as comma-delimited (csv) text files or by launching an instance of Excel and populating it with worksheets. In both cases, the possible output tables include claim and transaction files (together displaying the complete loss history), all the usual triangles, a table of large losses, a summary of the simulation specifications, and a summary of the frequency derivation by month.

## 5. DOCUMENTATION OF OPEN-SOURCE MODEL

The help files in the new open-source model fully document the implementation of all of the features that were included in the prototype model. These help files are organized into comprehensive model documentation on the CAS Web Site at http://www.casact.org/research/lsmwp/losshelp/index.cfm?fa=main. The documentation within the model and on the CAS Web Site will be kept up to date as this open-source software is enhanced.

Comprehensive program instructions are provided in Appendix A, and are also included within the model and in the CAS Web Site model documentation.

# 6. TESTING DETAILED OUTPUT AND FITTING THE MODEL

This section describes tests performed on the Simulator's output. The basic procedure for these tests was to run the Simulator with parameters that generate random claims with their transactions, from distributions whose densities or distribution functions or both functions are easily computed. Then we test the output files to see how closely the empirical distributions match the theoretical distributions.

An important point is that properly designed and documented statistical tests on simulator output also enable the user to fit models to real data. The user would transform the real data into the same format as the "claims" and "transactions" files output from the simulator. Then the user can run the same tests that we have developed for testing simulator output. Many of these tests include maximum likelihood estimates of the parameters. The users can generate the same maximum likelihood estimates on their own data. To enable this, we are including much of the source code used to perform the tests.

This section deals with the claims at their ultimate values, not with age-to-age loss development. Chapter 3 discusses the aggregate reserve triangles generated by summarizing the reserve transactions as of specific accident periods and development dates.

## 6.1 Data Formats

The simulator generates a Claims file and a Transaction file. The Claims file contains one record per claim. A sample of the format follows:

| Simulation No | Occurrence No | Claim No | Accident Date | Report Date | Line | Type |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 20000104 | 20000227 | 1 | 1 |
| 1 | 2 | 1 | 20000105 | 20000818 | 1 | 1 |
| ………. | | | | | | |
| 1000 | 324 | 1 | 20001222 | 20010120 | 3 | 3 |
| 1000 | 325 | 1 | 20001215 | 20010224 | 3 | 3 |

The Transaction file may have multiple transactions per claim, and has format:

| Simulation No | Occurrence No | Claim No | Date | Transaction | Case Reserve | Payment |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 20000227 | REP | 2000 | 0 |
| 1 | 1 | 1 | 20000413 | RES | 89412 | 0 |
| 1 | 1 | 1 | 20000417 | CLS | -91412 | 141531 |
| 1 | 2 | 1 | 20000818 | REP | 2000 | 0 |
| …….. | ………. | …….. | ……… | | | |

In this case, the first three records in the Transaction file correspond to the first record in the Claims file. The amounts in the "Case Reserve" column are incremental reserve changes, while the "Payment" column shows the payment made on each date. Thus, on 4/17/2000, Occurrence 1 – Claim No 1 closed, had zero case reserve (i.e., the sum of the Case Reserve column) and a cumulative paid amount of $141,531.

For purposes of testing ultimate severity and number of claims, the two files were merged into an "Ultimate Loss" file whose fields are listed:

| Simulation. No | Occurrence. No | Claim. No | Accident. Date | Report. Date | Line | Type | Case. Reserve | Payment |
|---|---|---|---|---|---|---|---|---|

This file has the same number of records as the Claims file. For each Simulation/Occurrence/Claim combination, the Case Reserve is zero, since all records are at ultimate value, and the Payment is the sum of all the Payment amounts from the Transaction file.

Many of the tests were performed on this "Ultimate Loss" file. **If the Modelers can format their own data into this format, then they can use the LSMWP source code to parameterize the Simulator.**

## 6.2 Description of Testing

The LSMWP Testing Group ran tests on the model as it was being developed. One reason for doing this is that these tests helped debug the model's calculations. The model simulates claims and transactions using the "R" language. Part of the "debugging" consisted of making sure the R commands work the way the LSMWP members think they work.. The subsections of section 6.2 each describe specific groups of tests that were performed.

### 6.2.1 Test of Elementary Frequencies, Trend, and Zero-modification

The model allows the user to input annual frequency by line and year, to specify the probability distribution for the types of claims within a line, and to specify "P(0)," the probability that a claim closes without payment. This section describes the final test among a series of tests of these parameters. The size of loss distribution was not used in this test except that a status indicator was generated for each claim, with "CNP" (resp. CWP) meaning that the claim was closed without (resp. with) payment.

A partial description of this test follows. The full list of parameters is in the Appendix B.

Test was run 10/27/2009. Project name: Frequency Test

Purpose: Test frequency with trend. Two types within one line.

- One Line with annual frequency distributed Poisson with mean 120
- Set claim/acc distribution matrix as follows:
  Prob =75% that one Type 1 claim is generated.
  Prob =25% that one Type 2 claim is generated.
- Freq Trend: 1.02 constant throughout
- P(0) = 0.4, EstP(0) = 0.4 for each Type.
- Accident Years: 2000-2002
- Random Seed: 16807
- Frequency correlation copula: normal Correlation=c( ) Dim = 1

Run: 1,000 simulations.

What would we expect from this run? If we think of the months *k* in the three accident years as numbered 1 through 36, the Poisson $\lambda_k$ for each month[2] fits the formula:

$\lambda_k = 10\,(1.02)^{k/12}$, where 10 = 120/12.

The total number of expected claims in each simulation is $\lambda = 10 \sum_{k=1}^{36} (1.02)^{k/12} = 371.2144835$. With 1,000 simulations, the total expected claim count equals $EN$ = 371,214. The actual number generated was $N$ = 371,198. The closeness of the actual and expected counts indicates that the Poisson frequency and trend parameters are operating in the manner expected.

[2] The simulator converts annual frequency to monthly frequency when performing the simulations.

The parameters were chosen so that a claim has probability of 0.75 (0.25) of being Type 1 (Type 2) and probability 0.60 (0.40) of having status CWP (CNP). A very important property of Poisson distributions states that the number of claims in 1,000 simulations for each Type($x$) x Status($y$) category are mutually independent Poisson random variables with mean $EN$ * Pr[Type=$x$]*Pr[Status=$y$].[3]

**<u>Chi-Square test for independence of Type and Status on claim counts.</u>**

To use the Chi-square test, we must construct data with both expected and actual claim counts. From the input parameters, there is only one Line, with a claim for this line being Type 1 (Type 2) 75% (25%) of the time. Also, the probability of the claim being Status CWP (CNP) is 60% (40%). Generally we wish to compare the actual and expected claims by unique "cells." Here the cells are defined by unique combination of all the predictors. Here are some S-PLUS[4] statements that produce all possible combinations:

```
temp1a <- unique(datacc$Simulation.No)
temp1b <- unique(datacc$Accident.Year)
temp1b2 <- unique(datacc$Accident.Month)
temp1c <- unique(datacc$Line)
temp1d <- unique(datacc$Type)
temp1e <- unique(datacc$Status)
temp1 <- list(
        Simulation.No=temp1a, Accident.Year =temp1b, Accident.Month=temp1b2,
             Line=temp1c,Type=temp1d,Status=temp1e)
temp2 <- expand.grid(temp1)
        for ( j in 1:length(temp2)) {
        if(is.factor(temp2[,j]))
                temp2[,j] <- as.character( temp2[,j])
        }
temp7 <- order(temp2$Simulation.No,temp2$Accident.Year,temp2$Accident.Month,
                   temp2$Line,temp2$Type,temp2$Status)
temp3 <- temp2[temp7,]
temp3$timeyear <-
        timeyear(base.year,temp3$Accident.Year,temp3$Accident.Month)
```

The "expand.grid" statement creates a data frame[5] with all possible combinations of the individual variables. It is important to note that the expand.grid statement produces more unique combinations of the individual variables than are found in the actual data. For this run, there are 144,000 combinations (1000 x 3 x 12 x 1 x 2 x 2). The last statement calls a user-defined function

[3] [11] pp. 103-104

[4] SPlus and R are implementations of the statistical language S.

[5] Think of a data frame in R or SPlus as a matrix, with each column representing either a predictor or a value and each row representing a combination of predictors or an observation.

"timeyear," which converts accident year and month to a numeric variable that corresponds to the length of the trend period over which the simulator's "trend" parameter operates.

To apply Chi-square testing, we want to calculate the expected number of claims for each record in the data frame and to tabulate the actual number of claims. The total claims for each simulation and accident year/month equals 10 * $(1.02)^{\text{timeyear}}$. Then the expected count for each type and status combination is just this number multiplied by the probabilities mentioned earlier. After this "expected count" dataframe is built, it can be merged with the actual counts to produce a dataframe, which we will call "ccstacked," with both actual and expected counts. Here is the image of the first few records:

Data frame "ccstacked"

| Simulation. No | Accident. Year | Accident. Month | Line | Type | Status | timeyear | expec.count | count |
|---|---|---|---|---|---|---|---|---|
| 1 | 2000 | 1 | 1 | 1 | CNP | 0.083333 | 3.004955 | 2 |
| 1 | 2000 | 1 | 1 | 1 | CWP | 0.083333 | 4.507432 | 6 |
| 1 | 2000 | 1 | 1 | 2 | CNP | 0.083333 | 1.001652 | 0 |
| 1 | 2000 | 1 | 1 | 2 | CWP | 0.083333 | 1.502477 | 3 |
| 1 | 2000 | 2 | 1 | 1 | CNP | 0.166667 | 3.009918 | 2 |
| 1 | 2000 | 2 | 1 | 1 | CWP | 0.166667 | 4.514877 | 4 |
| 1 | 2000 | 2 | 1 | 2 | CNP | 0.166667 | 1.003306 | 1 |

Proper application of the Chi-Square test requires that the expected counts from the theoretical distribution be modified so that the total number of claims and the marginal probabilities match the actual data.

| | **Actual Counts** | | | | | **Expected Counts** | | |
|---|---|---|---|---|---|---|---|---|
| | Type 1 | Type 2 | Margin | | | Type 1 | Type 2 | Margin |
| CNP | 111,066 | 37,007 | 0.398906 | | CNP | 111,029.0 | 37,044.0 | 0.398906 |
| CWP | 167,268 | 55,857 | 0.601094 | | CWP | 167,305.0 | 55,820.0 | 0.601094 |
| Margin | 0.749826 | 0.250174 | 371,198 | | | 0.749826 | 0.250174 | 371,198 |

$$\chi^2 = \sum_i \sum_j \frac{(Actual_{ij} - Expected_{ij})^2}{Expected} = 0.0819.$$

$\Pr[\chi^2 > 0.0819] = 0.775.$

The independence of Type and Status on counts is supported.

If we instead use the crosstabs command in S-PLUS, we obtain the output:

```
> temp1 <- crosstabs( count  ~  Status + Type ,
+           data=datacc.stack,
+           na.action=na.fail,                    ##### No missing data should exist
+           drop.unused.levels=F
+      )
> temp1
Call:
crosstabs(formula = count ~ Status + Type, data = datacc.stack, na.action =
       na.fail, drop.unused.levels = FALSE)
371198 cases in table
+-----------+
|N          |
|N/RowTotal |
|N/ColTotal |
|N/Total    |
+-----------+
Status     |Type
           |1          |2          |RowTotal   |
-----------+-----------+-----------+-----------+
CNP        |1.11066e5  |3.7007e4   |148073     |
           |0.75       |0.25       |0.4        |
           |0.4        |0.4        |           |
           |0.3        |0.1        |           |
-----------+-----------+-----------+-----------+
CWP        |1.67268e5  |5.5857e4   |223125     |
           |0.75       |0.25       |0.6        |
           |0.6        |0.6        |           |
           |0.45       |0.15       |           |
-----------+-----------+-----------+-----------+
ColTotal   |278334     |92864      |371198     |
           |0.75       |0.25       |           |
-----------+-----------+-----------+-----------+
Test for independence of all factors
       Chi^2 = 0.081902898185 d.f.= 1 (p=0.77473503489)
       Yates' correction was not used
```

Thus, we can see that the "crosstabs" command produces the Chi-square analysis.

**Use of Poisson Generalized Linear Model (GLM) for testing.**

The hypotheses underlying the model parameterization imply that covariates such as Type and Status have multiplicative effects on the number of claims. Such situations are conveniently modeled using a log-linear model such as a Poisson GLM with either categorical or numeric or both predictor variables. This model is more flexible than just relying on cross-tabulations. Since this is a powerful predictive modeling technique, I will describe its use here in some detail. In particular, we will discuss how the Type and Status variables affect the claim counts and whether there is an interactive effect.

In the Poisson GLM, the dependence of $\mu_i = E(Y_i)$, the variable of interest, is related to the covariates through the relationship: $\log \mu_i = \sum_{j=1}^{p} \beta_j x_{ij}$ for $i$ =1, 2, …*n*.. Here *p* is the number of covariates, $n$ is the number of observations for which the fitting takes place, and $\beta$ is the vector of coefficients determined by the model.[6]

To model the effects of Type and Status, we can use the data frame "ccstacked" described earlier. We run a "full" model, using Type, Status, and their interaction as predictors, and a "reduced model" without the interactive effect. If our model parameters work the way we think they should, then the interactive term should add very little predictive power and the reduced model should fit very well. The details of the actual runs are contained in Appendix B. We outline the results here.

The S-PLUS statement for the reduced model is

```
model5x<- glm(count ~   + Type + Status,
     data = temp.datacc.stack,
     family = poisson,
     x=T)
summary(model5x,correlation=F)
```

The predicted value for this fitted model is

$$\log \hat{\mu} = 1.126272675 - 1.097685774 * \mathit{Type} + 0.410026757 * \mathit{Status} \tag{6.2.1}$$

How do we interpret these equations? Type and Status in the "ccstacked" file each take two possible values. When converted to the indicator variables, the translations are that *Type* = 0 (1) when the original variable Type = 1 (2), and that *Status* = 0 (1) when the original variable Status = CNP (CWP). There are 36,000 combinations of simulation number, accident year, and accident month in the data.

Equation (6.2.1) gives the log of the predicted number of claims by Type and Status for each of these combinations. The sum of the predicted values for the four combinations of Type and Status equals 10.31. Multiplying this by 36,000 produces the total number of claims 371,198. This is almost exactly equal to the a priori expected number of claims.

The coefficient of − 1.097685774 for Type implies that

$\dfrac{\Pr[\text{Type} = 2]}{\Pr[\text{Type} = 1]} = \exp(-1.097685774) \approx 0.3336$. Indeed, this ratio is 1/3 from the input parameters.

The coefficient of 0.410026757 for Status implies that

$\dfrac{\Pr[\text{Status}=CWP]}{\Pr[\text{Status}=CNP]} = \exp(0.410026757) \approx 1.507$. The ratio of the input values is 1.500.

[6] *Generalized Linear Models* by McCullagh and Nelder [15] is an excellent reference on GLMs and Poisson GLMs.

Thus, we have good idea that the model fits the data well.

Can the model be improved by including the interactive variable *Type*Status* ?

To check this, we run the model statement

```
model6x<- glm(count ~  Type + Status + Type*Status                       ,
     data = temp.datacc.stack,
     family = poisson,
     x=T)
```

To test the additional value of the interactive variable we compare the deviations:

```
> anova(model5x,model6x,test="Chi")
Analysis of Deviance Table

Response: count

                          Terms Resid. Df   Resid. Dev         Test Df
1                + Type + Status    143997   160969.366
2 Type + Status + Type * Status    143996   160969.284 +Type:Status  1

      Deviance     Pr(Chi)
1
2 0.0819088429 0.774727081
```

The "Resid.dev" is the residual deviance as defined for GLMs and will not be discussed further here. What is interesting is that the reduction in deviance from introducing the interactive variable and the corresponding probability exactly matches those from the earlier "crosstabs" calculation. Thus we cannot reject the null hypothesis that there is no interactive term.

In the actual frequency testing, we ran GLMs that include the trend period as a numeric predictor, since we are assuming 2% annual frequency trend. These models provided better predictions, as we would expect. Even in the presence of a numeric time variable, the *type*status* interaction is not indicated.

### 6.2.2 Testing Size of Loss Distributions

The simulator was run May 12, 2010, to produce output on which one can test the distribution of claim severities (i.e., size of loss). The simulator produced claims from three different lines, with the severity distributions by line set as lognormal, Pareto, and Weibull. The frequency parameters assure that the number and size of claims by lines are mutually independent.

**<u>Description of Run and Parameters</u>**

Set up a test for univariate severity distributions and some time distributions.
Project name: Test severities 20100512
Purpose: Test univariate ultimate severities. Set up three lines of business with no correlation in frequency among the three lines. For each simulation, the number of occurrences by line by accident year is as described in the frequency parameters below.

- 3 Lines. For each line, every occurrence generates one claim of one type.
- Zero trend for frequency, zero trend for severity.

Accident Years: 2000-2001
Random Seed: 16807
Frequency correlation -- uncorrelated

Each line has same frequency parameters
Annual Frequency: Poisson(**600**)
Monthly exposure: (1), zero trend, no seasonality
Set claim/acc distribution matrix as follows:
**each occurrence generates 1 claim of one type**
**P(0) = 0.0, EstP(0) = 0.0**

Lags: Irrelevant for this run except for report lag. We are not testing the reserve change process.
Report lag: Exponential with Rate = 1/365. Mean=365 days. Max=3650.
Payment lag: Maximum one day
Inter-valuation lag: Maximum one day

Correlation of Amount with lag: normal Correlation=c( ) Dim=2
Reserve adequacy: Irrelevant, leave at default values.

Run: 100 simulations.

Severity parameters – these vary by Line.

- Line 1 Type 1: Lognormal, mean=100,000, stnd. dev. 100,000, max 10,000,000. This means that the input lognormal parameters are $\mu$ = meanlog = 11.16636357, $\sigma$ = sdmean = 0.832549779.
- Line 2, Type 2: Pareto with $\alpha$ (shape) =6, $\theta$ (scale) = 500,000. This results in a mean of 100,000 and standard deviation 122,474.5.
- Line 3, Type 3: Weibull with $\theta$ (scale) = 95,000, $\tau$ (shape) = 0.9. This results in mean of 99,957 and standard deviation of 111,256.

Expected total # claims = 600 (freq) x 100 (# sims) x 3 (lines) x 2 (years) = 360,000.
Actual # claims: 359,819.

**Results of simulation**

The output of the simulation was summarized to the "Ultimate Loss" file format described above in section 6.1.[7] This file contains one record for each claim, with the ultimate value of the claim in the "payment" field and the case reserve equal to zero, since all claims are at ultimate value.

The work in the test was divided up between two people. Joe Marker ran the simulator, created the summary file, and wrote the main body of this section. Yuting Yang did the testing work described below, wrote the R code, and produced the graphs.

The R project included four libraries that can be added using the statements.

```
library(stats4)
library(MASS)
library(actuar)
library(graphics)
```

Three separate vectors, named ultloss1, ultloss2, and ultloss3, were created using R. For *k* = 1 to 3, ultloss*k* is the vector of loss sizes for the claims from line *k*. To test whether the simulator actually generates claims sizes according to the parameters input, we fit each of the vectors to the lognormal, Pareto, and Weibull distributions, and then conducted goodness-of-fit tests.

First, exploratory data analysis was performed on the three data sets. This included:

- Calculating the standard statistics
- Generating histograms, empirical densities, empirical log densities, and empirical cdfs.

Appendix B contains all the graphs.

The following illustrates how the R commands produced the graphs for Line 1.

```
hist(ultloss1,main=“Histogram of observed data of Line 1,”
       freq=FALSE,breaks=10000,xlim=c(0,1050000))
plot(density(ultloss1),main=“Density estimate of Line 1,”xlim=c(-1000,600000))
plot(density(log(ultloss1)),main=“LogDensity estimate of Line 1,”xlim=c(0,20))
plot(ecdf(ultloss1),main=“Empirical cdf of Line 1,”xlim=c(0,1e+06))
```

This analysis of the graphs suggests that Line 1 has a lognormal density. Lines 2 and 3 have similar shapes, but Line 2 has more data in the tail, which suggests a heavier-tailed distribution.

The initial guess for the severity distributions is:

Line 1 – Lognormal, Line 2 – Pareto, Line 3 – Weibull.

The remaining steps in fitting the data consisted of:

1. Calculating the maximum likelihood estimates for the three distributions.

[7] The name of this file is "ultloss20100520.csv".

2. Informally looking at the fits by constructing and examining the Q-Q Plots for the candidate distributions.

3. Producing a more formal goodness-of-fit test by binning both the expected and empirical distributions and then performing chi-square tests.

**Calculate the maximum likelihood estimates of the parameters.**

The next step was to use maximum likelihood estimation (m.l.e.) to determine the optimal parameters for each line. We used the R command "fitdistr" for the calculations. However, prior to using this command, we calculated the negative loglikelihoods (n.l.l.) using the density functions. For each observed value *x*, let's review the both the p.d.f. *f*(*x*) and the corresponding negative loglikelihood – *ln f*(*x*).[8]

a) If $X$ has the lognormal distribution with parameters μ and σ, then $Y = \ln X$ has the normal distribution with parameters μ and σ. For the m.l.e., it is more convenient to fit observations $y = \ln x$ than it is to fit $x$. We have

$$f_Y(y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{z^2}{2}}, \text{ where } z = \frac{y-\mu}{\sigma}, \text{ and n.l.l.} = \ln(\sigma\sqrt{2\pi}\,) + \frac{z^2}{2}.$$

b) For $X$-distributed Pareto[9] with shape α and scale θ, we have

$$f(x) = \alpha\,\theta^{\alpha}\,(x+\theta)^{-(\alpha+1)}, \quad \text{and n.l.l.} = -\,(\ln\alpha + \alpha\ln\theta) + (\alpha+1)\ln(x+\theta)$$

c) For $X$-distributed Weibull with shape τ and scale θ, we have

$$f(x) = \frac{\tau x^{\tau-1}}{\theta^{\tau}} \exp\left[-(x/\theta)^{\tau}\right], \quad \text{and n.l.l.} = -\ln\tau + \tau\ln\theta - (\tau-1)\ln x + (x/\theta)^{\tau}$$

The parameter values are then determined by finding the parameter values that maximize the total n.l.l.

For Line 1, the R code below illustrates the use of the "fitdistr" command to fit *ln X* to a normal distribution:

```
fit1.ln <- fitdistr(log(ultloss1),"normal")

fit1.ln$estimate  #   mean        sd
                  #11.1659376  0.8361509

-fit1.ln$loglik   #148761.9
```

[8] The parameterizations here follow those from Appendix A of [11].

[9] Pareto refers to Pareto Type II, Lomax.

The output of any R command is an "object" with "properties." The first line above gives this object the name "fit1.ln". The vector fit1.ln$estimate of length 2 contains the optimal parameter values μ= `11.1659376` and σ = `0.8361509`. The number−fit1.ln$loglik is the minimized negative loglikelihood.

The same command was applied to the Line 2 data for both the Pareto and Weibull distributions.

```
## 2.1-Pareto ##

fit.p2<-fitdistr(ultloss2.0,dpareto,list(shape=6,scale=500000))
       ## list() provides initial values for optimization

fit.p2$estimate #  shape       scale
                #5.97635e+00 5.00000e+05

-fit.p2$loglik  #1500363

## 2.2-Weibull (second method slightly better) ##

fit2.w <- fitdistr(ultloss2.0,"weibull")

fit2.w$estimate  #   shape        scale
                 # 9.056193e-01 9.750673e+04

fit2.w$loglik    # -1500950

fit.w2<- fitdistr(ultloss2.0,dweibull,list(shape=.9097626,scale=95000))

fit.w2$estimate #   shape        scale
                # 9.009281e-01 9.500000e+04

-fit.w2$loglik  # 1500926
```

The Line 3 results are shown:

```
## 3.1-Pareto ##

fit.p3<-fitdistr(ultloss3.0,dpareto,list(shape=7,scale=6.026793e+05))

fit.p3$estimate #   shape        scale
                #6.966806e+00 6.026793e+05

-fit.p3$loglik  #1499343

## 3.2-Weibull (first method slightly better) ##

fit.w3 <- fitdistr(ultloss3.0,"weibull")

fit.w3$estimate  #   shape        scale
                 # 9.052532e-01 9.907429e+04

-fit.w3$loglik   # 1498920
```

For a set of observations, it is tempting to compare the n.l.l. from different models to label one of them as best. There is some logic in doing this for the Pareto and Weibull distributions, since they each have two parameters. However, there are better methods to differentiate them, which we discuss below.

It would be wrong to compare an n.l.l. from fitting the lognormal to either the n.l.l. for the Weibull or Pareto. This is because the observed variable *ln X* for fitting the lognormal is on a totally different scale than the variable *X* used for the Weibull and Pareto.

**<u>QQ Plots</u>**

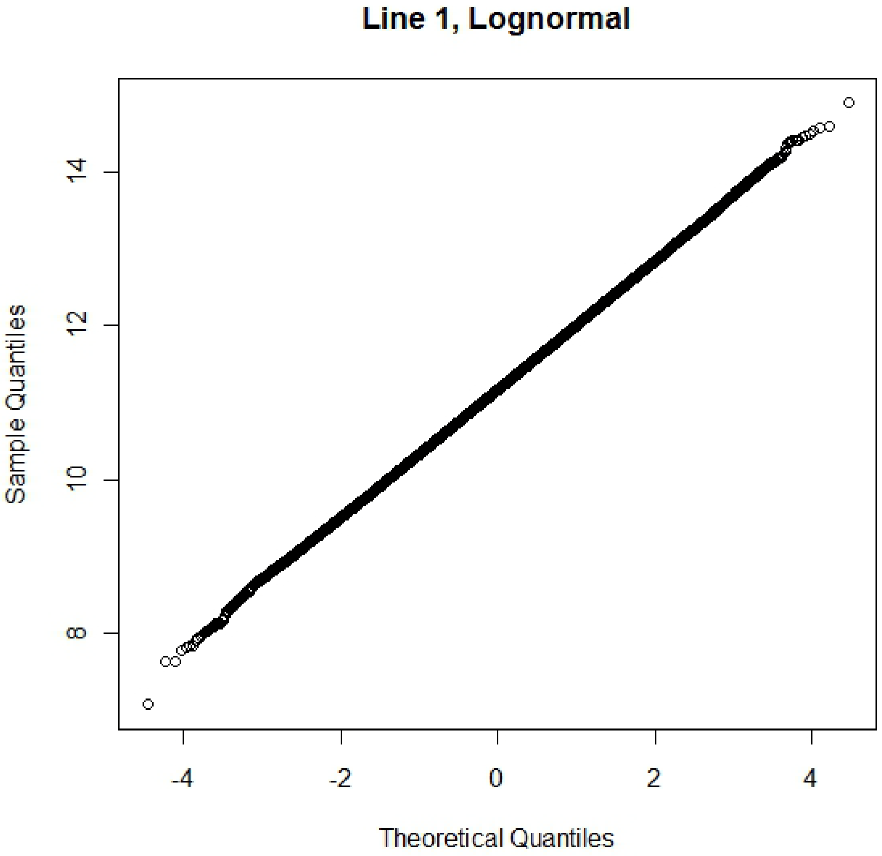
The QQ plot provides one of the best ways to visually compare two distributions. If one of the distributions is the empirical distribution of the observed values and the other is a c.d.f. of a random sample of the same size from the fitted distribution, then the closer the plot is to a 45 degree line, the better the fit.

The following R statements create a QQ-plot for the Line 2 data and the fitted Weibull distribution.[10]

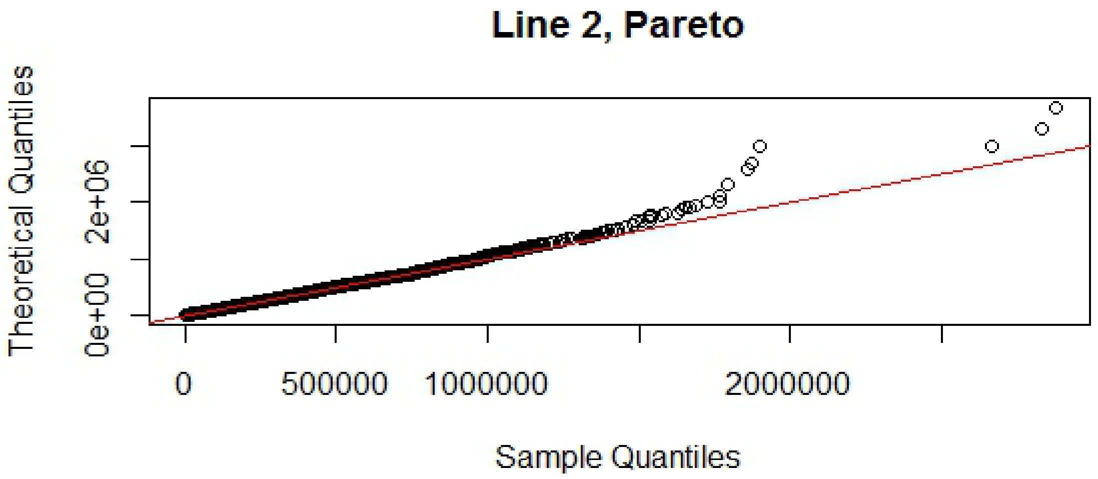
```
thqua.w2 <- rweibull(n2,shape=fit.w2$estimate[1],scale=fit.w2$estimate[2])

qqplot(ultloss2,thqua.w2,xlab="Sample Quantiles," ylab="Theoretical Quantiles,"
main="Line 2, Weibull")

abline(0,1,col="red")
```

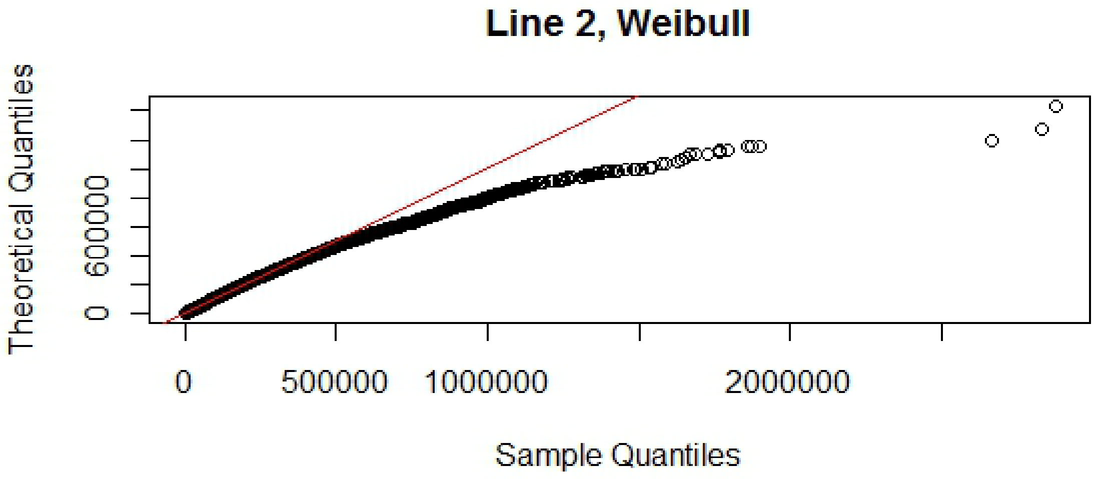
The QQ-plots for the fitted distributions for each line follow.

[10] These plots show quantiles of the observed values versus quantiles of a random sample from the fitted distribution. More commonly, the corresponding quantiles of the fitted distribution are used rather than a sample. See Section 5.1 of Venables and Ripley. [30]

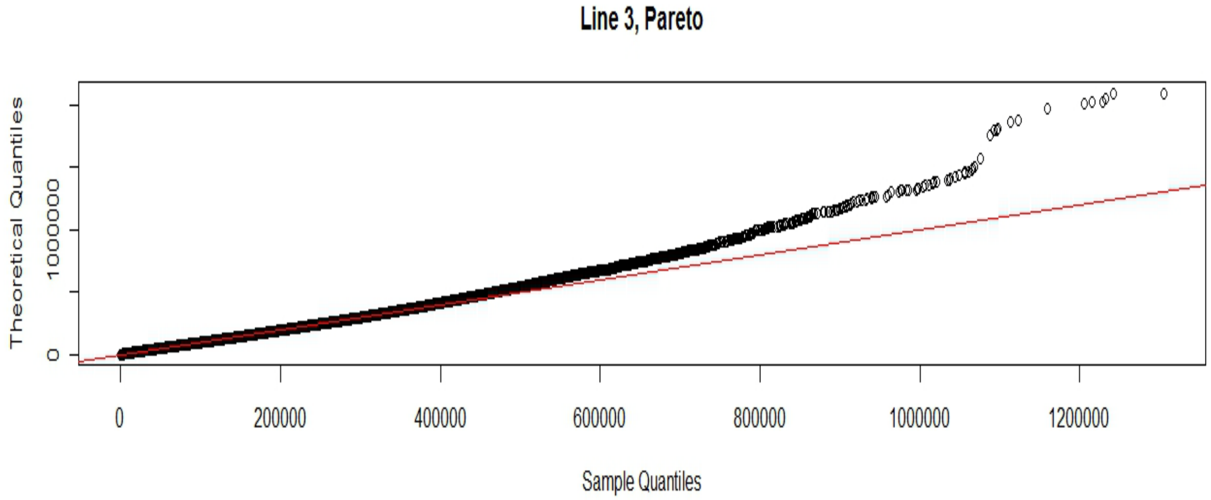

Line 1, Lognormal
Sample Quantiles
8
10
12
14
-4
-2
0
2
4
Theoretical Quantiles

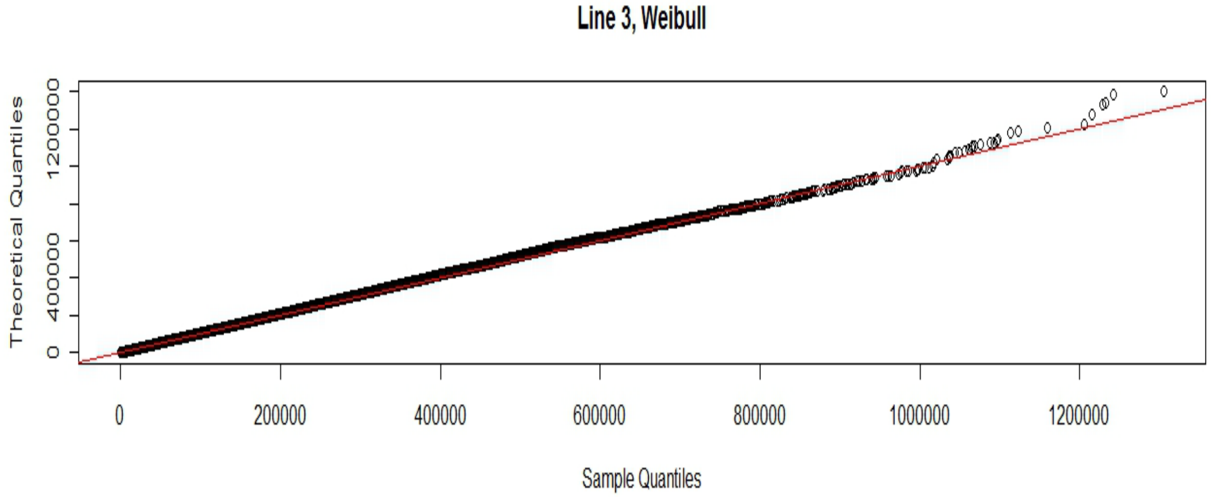
For Line 2, the Pareto fits the data much better over a longer range than the Weibull distribution.





For Line 3, the Weibull distribution is best.





The good fits for each of the three Lines indicates that the chosen models fit the ultimate loss size well. We can go further to test more formally the hypotheses that these models fit the data.

### Goodness-of-Fit tests

We discuss the use of the Chi-square test for the Pareto fit to Line 2. First, define the break points and set up the bins according to these break points.

```
#2.3.1 Pareto Chi-Square #
m = mean(ultloss2)
s = sqrt(var(ultloss2))

ult2.cut <- cut(ultloss2.0,
    breaks = c(0,m-s/2,m,m+s/4,m+s/2,m+s,m+2*s,2*max(ultloss2))) ##binning data
table.ult2 <- table(ult2.cut)                    ## binned data table
ult2.os <- c(as.vector(table.ult2))              ## vectorization

b = length(ult2.os)

labs.2 <- levels(ult2.cut)                       ## extract the breakpoints
break.2 <- cbind(lower = as.numeric(sub(“\\((.+),.*,” “\\1,” labs.2)),
    upper = as.numeric(sub(“[^,]*,([^]]*)\\],” “\\1,” labs.2)))

as.numeric(sub(“[^,]*,([^]]*)\\],” “\\1,” labs.2)))
```

See the R code in Appendix B for an illustration of the previous two R statements.

To calculate the expected number of claims in each bin, use ppareto to generate the c.d.f. values at the break points. Note that we need to define an “excess” interval to contain the expected number of claims larger than the last break point. Here we have chosen the last break point to be twice the maximum observed value.

```
ult2.p <- ppareto(break.2,shape=fit.p2$estimate[1],scale=fit.p2$estimate[2]) ##
    Pareto cdf values at break points

ult2.prob <- (ult2.p[,2]-ult2.p[,1])[1:b-1]  ## Probabilities of each interval
ult2.ex <- n2.0*c(ult2.prob,1-sum(ult2.prob))
    ## Expected frequency of each interval and the “excess” interval
```

The expected # claims by interval are in `ult2.ex` and the observed # claims are in `ult.os`.

```
E.2 = ult2.ex
O.2 = ult2.os

x.sq.2 = (E.2-O.2)^2/E.2

cbind(E.2,O.2,x.sq.2)       ## expected, observed, and chi-square of each interval
after full adjustment

##### Test Statistic Calculation ###########
#___________________________________________
#          E.2   O.2     x.sq.2
#[1,] 43993.890 44087 0.19705959
#[2,] 35651.989 35680 0.02200752
#[3,] 10493.758 10323 2.77864169
#[4,]  7240.583  7269 0.11152721
#[5,]  9277.383  9164 1.38570182
#[6,]  8063.576  8176 1.56743997
#[7,]  5289.820  5312 0.09299630
```

Next we calculate chi-squared statistic and its critical value, which shows that the null hypothesis that the Pareto model fits the data cannot be rejected. **It is important to note that the degrees of freedom are not six as we might expect, but rather 6-2=4.** This is because the Pareto distribution used for the expected values was the best fit Pareto distribution from the data, which is a two-parameter distribution.[11]

```
##chi-square test statistic##

df=length(E.2)-1-2           ## df = 4

chi.sq.2 <- sum(x.sq.2)    ## test statistic
chi.sq.2                   ## 6.155374

qchisq(.95,df)             ## critical value ## 9.487729

1-pchisq(chi.sq.2,df)      ## p-value  ## 0.1878414
```

Using the chi-squared test in R directly would produce a wrong *p*-value, to wit:

```
## chi-square goodness-of-fit test from R ##

chisq.test(O.2,p=E.2/n2.0)

#####################################################
#    Chi-squared test for given probabilities
# data:  O.2
# X-squared = 6.1554, df = 6, p-value = 0.406
######################################################
```

## Summary

The initial motivation for the procedures in this section was to test the output of the simulator to see whether it fit the severity distribution specified by the input parameters. We first used graphs to select candidates for fitted distributions. For example, for Line 2, this led to selecting Weibull and Pareto as candidates. Recall that the size of loss distribution for the Line 2 simulation was set as Pareto with α (shape) =6, and θ (scale) = 500,000. The m.l.e. estimates for the Pareto parameters are $\hat{\alpha} = 5.97635$ and $\hat{\theta} = 500{,}000$.

The Q-Q plots of the observed values versus sample values from the fitted theoretical distribution gives a visual indication of which distribution best fits the observed values for each line. Appendix B contains all the graphs. Next we used the chi-square test to decide whether the selected models fit the data well. We did not calculate other appropriate tests such as Kolmogorov-Smirnov or Anderson-Darling.[12]

**One can also use the procedures in this section to fit a set of actual claim size data, once it is converted to a format similar to the "ultloss" file from this chapter.** To help with this, we

[11] See, for example, Devore [4].

[12] See pp. 448-458 in Klugman, Panjer, and Wilmot [11], for a fuller discussion of these tests.

included in Appendix B all the R statements used to fit and test the various distributions we tested for each line.

### 6.2.3 Testing Correlated Frequencies

In order to make sure the Copula feature in the model is appropriate, we implemented statistical techniques to test the Gaussian Copula setting based on correlated set of frequency data simulated by the model. The simulation run results files "c_20100720_1900.csv" and "bymonth.csv" are used, which have one-year frequency data for three lines. The number of simulation is 1,000. The correlation assumption is as below:

| Correlation | Line 1 | Line 2 | Line 3 |
|---|---|---|---|
| Line 1 | 1 | 0 | 0.99 |
| Line 2 | 0 | 1 | -0.01 |
| Line 3 | 0.99 | -0.01 | 1 |

Some key parameters for the simulator run that produced the dataset for this test include:

Annual frequency for each line is Poisson with mean 96 occurrences.

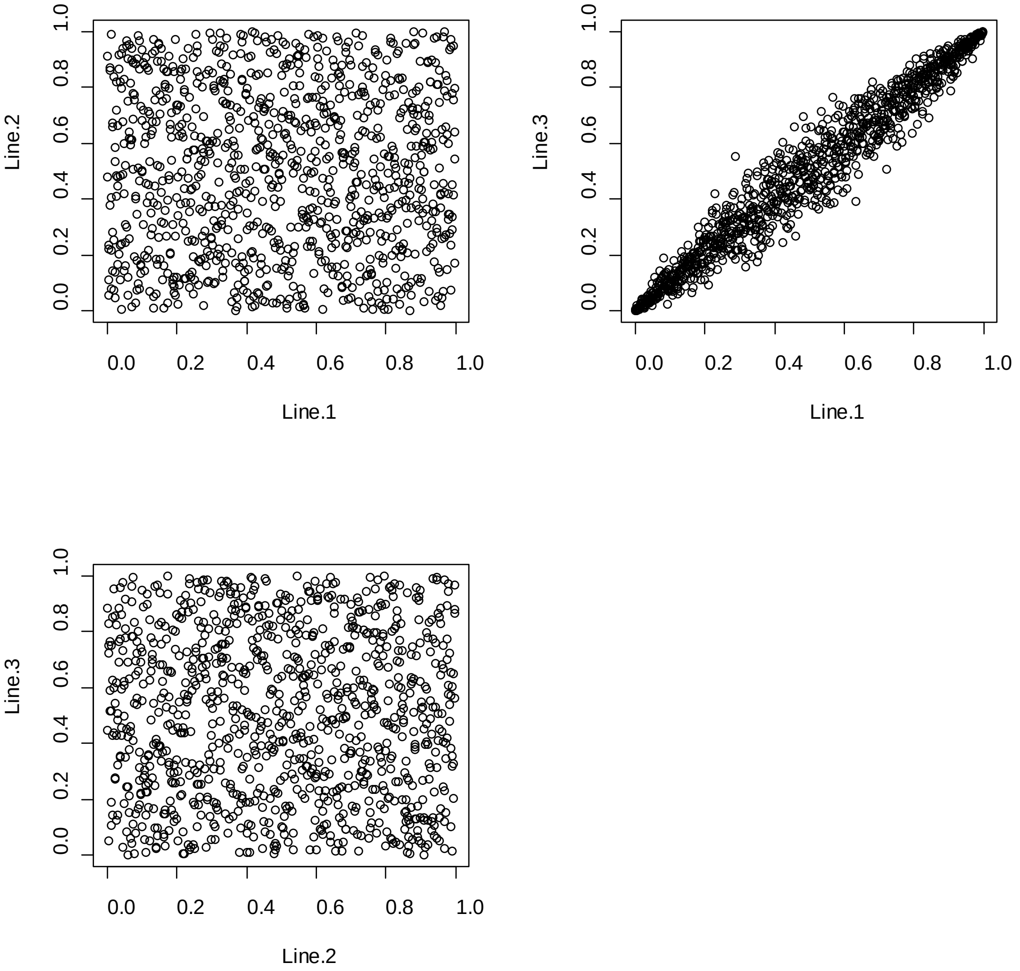
Each occurrence generates exactly one claim.

The simulator produces claims for accident year 2000 only.

**Descriptive statistics**

The annual frequency number for each line and each simulation is extracted based on the run results file. In addition, the first month frequency number is also extracted for a parallel test run. The two sets do not show too much difference regarding testing results. The test results below are based on annual frequency only. See Appendix B for the layout of the data set used for this test.

A scatter-plot of the empirical cumulative distribution function for each pair of lines is displayed below, showing a high positive correlation between line 1 and line 3. This is in line with our assumption for the correlation between Line 1 and Line 3: 0.99. Other pairs' correlations cannot be identified, which is also consistent with our parameter assumption.



### *Test of Correlation*

(1) Fit simulated data set (data pair) to normal copulas.

Two methods have been used for the copula fitting. We can see that Rho 1, Rho 2, and Rho 3 MLE are very close to our assumption (0, 0.99, -0.01).

### *Using maximum likelihood method*

The estimation is based on the maximum likelihood and a sample of size 1000.

```
          Estimate  Std. Error       z value  Pr(>|z|)
rho.1 -0.002112605 0.031977597   -0.06606516 0.9473259
rho.2  0.979258746 0.000921392 1062.80366235 0.0000000
rho.3 -0.010486832 0.031974114   -0.32797880 0.7429277
```

The maximized loglikelihood is 1591.565

The convergence code is 0.

### *Using Inversion of Kendall's tau*

The estimation is based on the inversion of Kendall's tau and a sample of size 1000.

```
          Estimate  Std. Error     z value  Pr(>|z|)
rho.1 -0.01420116 0.033302051  -0.4264349 0.6697910
rho.2  0.97843954 0.001595654 613.1904797 0.0000000
rho.3 -0.01938295 0.033034985  -0.5867400 0.5573783
```

(2) <u>Apply goodness-of-fit test.</u>

a) The empirical copula (from simulation model) and hypothesized copula are compared under the null hypothesis that they are from the same copula. Cramér-von-Mises (CvM) statistic *S* is used and the *p*-value is estimated using parametric bootstrapping method (simulated *p* value)

$$S_n = \int_{[0,1]^d} \mathbb{C}_n(\mathbf{u})^2 \mathrm{d}C_n(\mathbf{u}) = \sum_{i=1}^{n} \{C_n(\hat{\mathbf{U}}_i) - C_{\theta_n}(\hat{\mathbf{U}}_i)\}^2$$

$$C_n(\mathbf{u}) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(U_{i1} \le u_1, \ldots, U_{id} \le u_d),$$

$$\mathbf{u} = (u_1, \ldots, u_d) \in [0, 1]^d.$$

This test can be realized by using function "gofCopula". Please note that, due to the simulation number chosen to do parametric bootstrapping, the run takes a while to finish.

The goodness-of-fit test using CvM statistics was intended to be applied at dimension 3. However, the run speed is very slow and impractical. Only 100 simulations for calculating the *p* value take more than 20 minutes to finish. Therefore, I used two dimension tests three times to accomplish it. Only 100 simulations for parametric bootstrapping will take around 10 minutes for two dimensions. Ties will produce an incorrect *p* value calculation. We used pseudo observations to randomly break the ties as suggested in the article "Modeling Multivariate Distributions with Continuous Margins Using the copula R Package" by Ivan Kojadinovic and Jun Yan (published in *Journal of Statistical Software,* May 2010, Vol. 34, Issue 9).

The results below show that the simulated correlated frequencies can fit to normal copula very well.

Line 1&2

Parameter estimate(s): -0.002100962
Cramér-von Mises statistic: 0.0203318 with *p*-value 0.4009901

Line 1&3

```
Parameter estimate(s): 0.97926
Cramer-von Mises statistic: 0.007494245 with p-value 0.3811881
```

Line 2&3

```
Parameter estimate(s): -0.01049841
Cramér-von Mises statistic: 0.01614539 with p-value 0.5891089
```

b) Kolmogorov-Smirnov (K-S) test. Only two dimensions are currently the maximum number allowable in R. Based on fitted copula, a random data set is generated and the empirical data and generated data are used to do two-2-sample K-S test. The null hypothesis is that the two data sets are from the same continuous distribution. The statistic is $D_{K-S} = \max_i \left| F_E(x_i) - F_H(x_i) \right|$. The *p* value will be calculated exactly if the sample size is less than 10,000.

Line 1&2 (Null hypothesis: correlation coefficient = 0)

```
        Two-sample Kolmogorov-Smirnov test

data:  x12 and y12
D = 0.016, p-value = 0.96
alternative hypothesis: two-sided
```

Line 1&3 (Null hypothesis: correlation coefficient = 0.99)

```
        Two-sample Kolmogorov-Smirnov test

data:  x13 and y13
D = 0.014, p-value = 0.9895
alternative hypothesis: two-sided
```

Line 2&3 (Null hypothesis: correlation coefficient = -0.01)

```
        Two-sample Kolmogorov-Smirnov test

data:  x23 and y23
D = 0.0135, p-value = 0.9933
alternative hypothesis: two-sided
```

The *P* value of the K-S test shows that we cannot reject the null hypothesis. This is also quite consistent with what we can spot from the scatter plots.

(3) Conclusion

Based on the scatter plots, normal copula fitting parameters, and also the goodness-of-fit test using CvM statistics, we can conclude that the correlation in simulated frequency data is consistent with our assumption.

## 7. POTENTIAL APPLICATIONS AND MODEL ENHANCEMENTS

The CAS Committee on Dynamic Risk Modeling plans to hold a 2011 Call Paper Program soliciting enhancements to the model, additional testing as well as papers applying the model to test alternative loss reserving methods and models. The following model enhancements would be welcome:

1. Include covariates as categorical or numeric variables in setting the parameters for the distributions. For example, in the modeling for claim size, if "state" is a categorical variable that affects the parameter of the distribution, we can include this information as input for the model, rather than setting up a separate "type" for each state. The covariate should be passed to the output detailed claim file.
2. Additional pairs (or groups) of variables whose sample values are correlated.
3. Input and output to and from standard database programs.

The following additional tests of the Loss Simulation Model would be welcome, and could be performed while testing alternative loss reserving methods and models:

1. Poisson frequencies have been tested, while negative binomial frequencies have not been tested.
2. Correlations of mean frequencies across lines of business have been tested, while correlations between size of loss and report lag within a Type have not been tested.
3. Ultimate claim values have been tested, while the sequence of loss reserve changes has not been tested.
4. The Single Payment Model has been tested, while the Periodic Payments Model and the Multiple Random Payments Model have not been tested.
5. The mechanics of certain parameters such as alpha, severity trend, inertia, etc. have been verified by examining the code. However, output from runs using these parameters has not been verified.
6. Apply the testing approach described in Section 3.2 above to assess the ability of simulated data from the Loss Simulation Model to represent "real" data.

Other tests, enhancements or topics not on these lists are also welcome as long as they address something specific to the Loss Simulation Model or its intended uses.

## 8. CONCLUSIONS

The LSMWP has developed a model that we hope will become a valuable tool in researching reserving methods and models. We hope that actuaries will use this model to:

- Better understand the underlying loss development process.
- Determine which methods and models work best in different reserving situations.
- Reflect this knowledge in evolving loss reserving practices.

# BIBLIOGRAPHY

**Categories**

1. General Interest
2. Building Models
3. Testing Simulated Data
4. Testing Reserving Methods

**Abbreviations**

CAS – Casualty Actuarial Society
IAA – International Actuarial Association
SOA – Society of Actuaries

[1] Arata, David A., "Computer Simulation and the Actuary: A Study in Realizable Potential," *Proceedings of the CAS* LXVIII, 1981, pp. 24-64, http://www.casact.org/pubs/proceed/proceed81/81024.pdf.
**Type of Material:** Refereed Paper/Article
**Categories:** 1,2
**Abstract:** This paper argues that computer simulation is an underappreciated and, therefore, underutilized casualty actuarial resource. In so contending, "Computer Simulation and the Actuary" discusses five applications of Monte Carlo computer simulation to everyday actuarial problems: establishing full credibility standards; testing the solidity of new, limited purpose insurance companies; pricing difficult or catastrophic exposures; customizing casualty insurance charges and excess loss premium factors: and developing loss reserve confidence intervals. Illustrations of appropriate simulation solutions to each of these problems are provided.

[2] Bickerstaff, David R., "Automobile Collision Deductibles and Repair Cost Groups: The Lognormal Model," *Proceedings of the CAS* LIX, 1972, pp. 68-102, http://www.casact.org/pubs/proceed/proceed72/72068.pdf.
**Topics:** Collision losses, lognormal distribution
**Type of Material:** Refereed Paper/Article
**Categories:** 1,2
**Comments:** This paper applies the theory of lognormal distributions to the study of collision losses, which are left-truncated and shifted. The paper discusses many properties of the lognormal, such as the distribution of moments.

[3] Bratley, P., B.L. Fox, and L.E. Schrage, *A Guide to Simulation,* 2nd ed., New York: Springer, 1987.
**Topics:** Risk Theory
**Type of Material:** Book
**Categories:** 1,2,3

[4] Devore, Jay L., *Probability and Statistics for Engineering and the Sciences,* 5th Ed., Belmont, CA: Duxbury Thomson Learning, 2000.
**Topics:** Probability and Statistics
**Type of Material:** Book
**Categories:** 2
**Comments:** This is a practical book covering probability and statistics. It has a large number of well-explained examples.

[5] Faraway, Julian J., *Extending the Linear Model with R,* Boca Raton, FL: Chapman and Hall/CRC, 2006.
**Topics:** Non-Linear models, GLM
**Type of Material:** Book
**Categories:** 1,2
**Comments:** This is an advanced textbook covering topics beyond linear models, such as generalized linear models, mixed-effects models, and non-parametric models. The book uses the programming language R and assumes knowledge of linear models. The author has written a predecessor book titled *Linear Models with R.*

# BIBLIOGRAPHY

**Categories**

1. General Interest
2. Building Models
3. Testing Simulated Data
4. Testing Reserving Methods

**Abbreviations**

CAS – Casualty Actuarial Society
IAA – International Actuarial Association
SOA – Society of Actuaries

[6] Frees, Edward W. and Emiliano A. Valdez, "Understanding Relationships Using Copulas," *North American Actuarial Journal* 2 (1), 1998, pp. 1-25, www.soa.org/library/journals/north-american-actuarial-journal/1998/january/naaj9801_1.pdf - 2009-05-07.
**Topics:** Copulas and related topics
**Type of Material:** Refereed Paper/Article
**Categories:** 1,2,3
**Comments:** An excellent introduction to copulas, this article includes fitting models to data, simulation, and several insurance company data applications.

[7] Heckman, Philip E., and Glenn G. Meyers, "The Calculation of Aggregate Loss Distributions from Claim Count Distributions," *Proceedings of the CAS* LXX, 1983, pp. 22-61, http://www.casact.org/pubs/proceed/proceed83/83022.pdf.
**Topics:** Reserves
**Type of Material:** Refereed Paper/Article
**Categories:** 2,3
**Comments:** This paper discusses aggregate loss distributions from the perspective of collective risk theory. An accurate, efficient and practical algorithm is given for calculating cumulative probabilities and excess pure premiums. The input required is the claim severity and claim count distributions. One of the main drawbacks of the collective risk model is the uncertainty of the parameters of the claim severity and claim count distributions. Modifications of the collective risk model are proposed to deal with these problems. These modifications are incorporated into the algorithm. Examples are given illustrating the use of this algorithm. They include calculating the pure premium for a policy with an aggregate limit; calculating the pure premium of an aggregate stop-loss policy for group life insurance; and calculating the insurance charge for a multi-line retrospective rating plan, including a line which is itself subject to an aggregate limit.

[8] Hogg, R.V, J.W. McKean, and A.T. Craig, *Introduction to Mathematical Statistics*, 6th ed., Upper Saddle River, NJ: Prentice Hall, 2004.
**Topics:** Probability and Statistics
**Type of Material:** Book
**Categories:** 1,2
**Comments:** A probability and statistics textbook frequently used in university statistics classes, this book contains chapters on Statistical Inference, Maximum Likelihood Methods, and Inferences about Normal Models.

[9] Hogg, Robert V. and Elliott A. Tanis, *Probability and Statistical Inference*, 7th ed., Upper Saddle River, NJ: Prentice Hall, 2006.
**Topics:** Probability and Statistics
**Type of Material:** Book
**Categories:** 1,2
**Comments:** This book on probability and statistics is easier to understand than the Hogg, McKean, and Craig book.

# BIBLIOGRAPHY

**Categories**

1. General Interest
2. Building Models
3. Testing Simulated Data
4. Testing Reserving Methods

**Abbreviations**

CAS – Casualty Actuarial Society
IAA – International Actuarial Association
SOA – Society of Actuaries

[10] Klugman, Stuart, and Jacques Rioux, "Toward a Unified Approach to Fitting Loss," *North American Actuarial Journal* 10:1, SOA: Schaumburg, IL, 2006, http://www.soa.org/library/journals/north-american-actuarial-journal/2006/january/naaj0601-6.pdf.
**Topics:** Reserves
**Type of Material:** Refereed Paper/Article
**Categories:** 2,3
**Comments:** There are two components to fitting models: selecting a set of candidate distributions and determining which member fits best. It is important to have the candidate set be small to avoid overfitting. Finite mixture models using a small number of base distributions provide an ideal set. Because actuaries fit models for a variety of situations, particularly with regard to data modifications, it is useful to have a single approach. Though not optimal or exact for a particular model or data structure, the method proposed in this paper should be reasonable for most all settings. A computer program implementing these models and techniques is provided.

[11] Klugman, Stuart, Harry Panjer, and Gordon Willmot, *Loss Models: From Data to Decisions,* Third ed., New York: John Wiley & Sons, Inc., 2008.
**Topics:** Probability Distributions
**Type of Material:** Book
**Categories:** 1,2,3
**Comments:** *(Included in the CAS Syllabus)* This book is devoted to the problem of fitting parametric probability distributions to data. This treatment unifies loss modeling in one book. Emphasis is made on the distribution of single losses related to claims experienced against various types of insurance policies. The book includes five sets of insurance data as examples.

[12] Kojadinovic, Ivan, and Jun Yan, "Modeling Multivariate Distributions with Continuous Margins Using the copula R Package," *Journal of Statistical Software* 34, No. 9, May 2010, http://www.jstatsoft.org/v34/i09/paper.
**Topics:** Copulas, including discrete variables
**Type of Material:** Refereed Paper/Article
**Categories:** 1,2,3
**Comments:** A thorough discussion of using copulas in R, this article describes how to adapt copulas to discrete random variables. Includes information on goodness-of-fit tests as well as illustrations using insurance loss data.

[13] Kreps, Rodney E., and Jeffrey A. Englander, "Simulation Models for Reserve and Surplus Analysis," Casualty Loss Reserve Seminar Transcript, Arlington, VA: CAS, 1993.
**Topics:** Reserves
**Type of Material:** Meeting transcription
**Categories:** 2
**Comments:** Simulation models have a long history of use in the actuarial profession for estimating confidence intervals for loss reserves and pricing estimates. More recently, models have been developed which can be used to derive probability levels for insurance company surplus. In this session we will describe procedures for modeling loss reserve uncertainty using simulation models. Models utilizing claim count and claim severity distribution will be discussed. The issue of parameter variance and its estimation will be presented. Techniques for estimating payout patterns and its impact on discounted reserve variability will be presented. A more comprehensive model of insurance company operations which can be used to assess insurance company solvency will then be presented. The items modeled will include premiums, claims outstanding, investment returns, business cycles, and effect of inflation.

# BIBLIOGRAPHY

| Categories | | Abbreviations |
|---|---|---|
| 1. General Interest | 3. Testing Simulated Data | CAS – Casualty Actuarial Society |
| 2. Building Models | 4. Testing Reserving Methods | IAA – International Actuarial Association |
| | | SOA – Society of Actuaries |

[14] Lyons, Daniel K., “A Note on Simulation of Claim Activity for Use in Aggregate Loss Distributions” *CAS Forum*, Spring 1994, Vol. 1, pp. 357-392, http://www.casact.org/pubs/forum/94spforum/94spf357.pdf .
**Topics:** Reserves; Ratemaking; Reinsurance Research
**Type of Material:** Non-Refereed Paper/Article
**Categories:** 2
**Comments:** Aggregate loss distributions have been used in a number of different applications over the last few years. These applications have usually focused on the distribution of losses at ultimate or final values and have not studied how losses move to ultimate values over time. The approach outlined in this note models claim activity through the use of transition matrices. Individual claim activity is then incorporated into an aggregate loss simulation model to determine a number of distributions of interest. Keywords: confidence estimates, loss development, IBNR, reinsurance research, size of loss distributions.

[15] McCullagh, P., and J.A. Nelder, *Generalized Linear Models,* 2nd ed*.*, Boca Raton, FL: Chapman & Hall/CRC, 1989.
**Topics:** Predictive Modeling, GLMs
**Type of Material:** Book
**Categories:** 1,2,3
**Comments:** This is an excellent reference on GLMs and Poisson GLMs and is often considered the definitive early work on generalized linear models.

[16] Meyers, Glenn, “Estimating Predictive Distributions for Loss Reserve Models,” CAS *Forum,* Fall 2006*,* pp. 159-203, 2006, http://www.casact.org/pubs/forum/06fforum/163.pdf.
**Topics:** Reserves
**Type of Material:** Refereed Paper/Article
**Categories:** 2,3,4
**Comments:** This paper demonstrates a Bayesian method for estimating the distribution of future payments of individual insurers. The main features of this method are: (1) the stochastic loss reserving model is based on the collective risk model; (2) predicted loss payments derived from a Bayesian methodology utilizing large/stable insurers as its prior information; and (3) applying tests on large numbers of insurers and demonstrating its predictions are within the statistical bounds expected for a sample of its size. It concludes with an analysis of reported reserves and their subsequent developments in terms of the predictive distribution calculated by this Bayesian methodology.

[17] Meyers, Glenn, “Thinking Outside the Triangle,” *ASTIN Colloquium Paper,* 2007, http://www.actuaries.org/ASTIN/Colloquia/Orlando/Papers/Meyers.pdf.
**Topics:** Reserves
**Type of Material:** Refereed Paper/Article
**Categories:** 2,3,4
**Comments:** Overfitting is the result of having a model that is too complex for the amount of data available, which is typically the case when a loss reserve model has a large number or parameters on a smallish triangle of data. This paper uses simulation methods to diagnose overfitting in estimating the predictive distribution of loss reserves by the method of maximum likelihood. This paper then shows how to use a Bayesian fitting methodology to overcome overfitting using prior information that is “outside the triangle.”

## BIBLIOGRAPHY

**Categories**

1. General Interest
2. Building Models
3. Testing Simulated Data
4. Testing Reserving Methods

**Abbreviations**

CAS – Casualty Actuarial Society
IAA – International Actuarial Association
SOA – Society of Actuaries

[18] Narayan, P., and T. Warthen, "A Comparative Study of the Performance of Loss Reserving Methods Through Simulation," *CAS Forum,* Summer 1997, Vol. 1, pp. 175-195, http://www.casact.org/pubs/forum/97sforum/97sf1175.pdf.
**Topics:** Reserves
**Type of Material:** Non-Refereed Paper/Article
**Categories:** 2,4
**Comments:** This paper employs four methods to simulate loss development triangles and uses the simulated triangles to compare two traditional actuarial reserving methods against three regression methods of reserving. Section IV's "Comparison of Procedures" provides good insights into using simulation to test reserving methods.

[19] Panjer, Harry H., "The Aggregate Claims Distribution and Stop-Loss Reinsurance," *Transactions of the SOA* 32, 1980, pp. 523-535, http://www.soa.org/library/research/transactions-of-society-of-actuaries/1980/january/tsa80v3215.pdf.
**Topics:** Risk Theory
**Type of Material:** Refereed Paper/Article
**Categories:** 2
**Comments:** This paper describes a method for approximating the distribution of aggregate claims for a group life insurance contract. It is found that a compound Poisson process appropriately modeled the aggregate claims distribution, based on the collective risk assumption, which states that each life that leaves the group by death claim is immediately replaced by a life with identical mortality characteristics. The resulting method produces accurate and useful results, and is relatively easy to use as long as the number of distinct amounts of insurance in the group is not too great. In practice, this condition is achieved by rounding all face amounts to be integral multiples of some convenient unit.

[20] Patrik, Gary, "Estimating Casualty Insurance Loss Amount Distributions," *Proceedings of the CAS* LXVII, 1980, pp. 57-109, http://www.soa.org/library/research/transactions-of-society-of-actuaries/1980/january/tsa80v3215.pdf.
**Topics:** Reserves
**Type of Material:** Refereed Paper/Article
**Categories:** 2,3
**Comments:** This paper concentrates upon probability model-building and statistical techniques for estimating and testing the model parameters. A general procedure for selecting a "best" parameterized model based upon loss amount data. This solves only part of a broader problem, which is to estimate loss amount distributions for future coverage periods or future final-valued loss amount distributions for past coverage periods where the losses are not all settled or even known. To solve this broader problem it is necessary to specify models of the overall insurance loss processes, defining how the future relates to the past and how the individual insured relates to the whole insurance portfolio.

# BIBLIOGRAPHY

| **Categories** | | **Abbreviations** |
|---|---|---|
| 1. General Interest | 3. Testing Simulated Data | CAS – Casualty Actuarial Society |
| 2. Building Models | 4. Testing Reserving Methods | IAA – International Actuarial Association |
| | | SOA – Society of Actuaries |

[21] Pentikäinen, Teivo, and Jukka Rantala, "A Simulation Procedure for Comparing Different Claims Reserving Methods" *CAS Forum*, Fall 1995, pp. 128-156, http://www.casact.org/pubs/forum/95fforum/95ff128.pdf .
**Topics:** Reserves
**Type of Material:** Non-Refereed Paper/Article
**Categories:** 2,4
**Comments:** The estimation of outstanding claims is one of the important aspects in the management of the insurance business. Exploration of the inaccuracies involved is traditionally based on a post-facto comparison of the estimates against the actual outcomes of the settled claims. However, until recent years it has not been usual to consider the inaccuracies inherent in claims reserving in the context of more comprehensive (risk theoretical) models. Important parts of the technique that are outlined in this paper can be incorporated into over-all risk theory models to introduce the uncertainty involved with technical reserves as one of the components in solvency and other analyses. The idea in this paper is to describe a procedure by which one can explore how various reserving methods react to fictitious variations, fluctuations, trends, etc., which might influence the claims process and how they reflect on the variables indicating the financial position of the insurer.

[22] Robertson, John P., "The Computation of Aggregate Loss Distributions," *Proceedings of the CAS* LXXIX, 1992, pp. 57-133 http://www.casact.org/pubs/proceed/proceed92/92057.pdf.
**Topics:** Reserves
**Type of Material:** Refereed Paper/Article
**Categories:** 2
**Comments:** This paper provides an application of the Fourier transform as a tool for computation of aggregate loss distributions from arbitrary frequency and severity distributions. The paper offers a complete algorithm and provides examples to allow its implementation in various computer languages. The final section contains a discussion of excess loss distributions where computation is not limited to the fast Fourier transform based algorithm.

[23] Rollins, John W., "Performance Testing Aggregate and Structural Reserving Methods: A Simulation Approach," *CAS Forum,* Summer 1997, Vol. 1, pp. 137-174, http://www.casact.org/pubs/forum/97sforum/97sf1137.pdf.
**Topics:** Reserves
**Type of Material:** Non-Refereed Paper/Article
**Categories:** 2
**Comments:** Aggregate financial data histories are extensively used by actuaries in projecting ultimate liabilities, but the claim occurrence, reporting, and settlement process which generates these data is not perfectly understood, rarely modeled directly, and not incorporated into the structure of most popular reserving methods. This paper utilizes today's computer applications to create an automated simulation tool. This module allows the user to choose statistical assumptions for each element of the claims process and specify the structure of the simulation experiment. It then generates random paid loss and claim count histories based on these inputs. The module is used to perform an experimental test of the performance of aggregate versus structural reserving methods. The methods chosen are the paid loss development method and a new "closed claim cost" structural method. In each trial, a database of ten accident years at ten annual evaluations is simulated. Then both methods are run at five successive calendar year evaluations of the simulated data. Several error functions are tabulated at each valuation date and the speed of the approach of each method's indication to the true value of the aggregate costs is examined.

# BIBLIOGRAPHY

| Categories | | Abbreviations |
|---|---|---|
| 1. General Interest | 3. Testing Simulated Data | CAS – Casualty Actuarial Society |
| 2. Building Models | 4. Testing Reserving Methods | IAA – International Actuarial Association |
| | | SOA – Society of Actuaries |

[24] Sargent, Robert G., "Validation and Verification of Simulation Models," *Proceedings of the 1999 Winter Simulation Conference,* Hanover, MD: Institute for Operations Research and the Management Sciences, 1999, pp. 39-48, http://www.informs-sim.org/wsc99papers/005.PDF.
**Topics:** Model Validation and Verification
**Type of Material:** Refereed Paper/Article
**Categories:** 3
**Comments:** Referenced in S.M. Ross Simulation Text (2002 Academic Press, 6277 Sea Harbor Drive, Orlando, FL 32887), this article defines model validation and verification. The section on operational validity describes approaches helpful in assessing testing model results against known data points.

[25] Schiegl, M., "On the Safety Loading for Chain Ladder Estimates: A Monte Carlo Simulation Study," *ASTIN Bulletin* 32:1, 2002, pp. 107-128*,* IAA: Brussels, Belgium 2002, http://www.casact.org/library/astin/vol32no1/107.pdf.
**Type of Material:** Refereed Paper/Article
**Categories:** 4
**Comments:** A method of analyzing the risk of taking a too low reserve level by the use of Chain Ladder method is developed. We give an answer to the question of how much safety loading in terms of the Chain Ladder standard error has been added to the Chain Ladder reserve in order to reach a specified security level in loss reserving. This is an important question in the framework of integrated risk management of an insurance company. Furthermore we investigate the relative bias of Chain Ladder estimators. We use Monte Carlo simulation technique as well as the collective model of risk theory in each cell of run-off table. We analyze deviation between Chain Ladder reserves and Monte Carlo simulated reserves statistically. Our results document dependency on claim number and claim size distribution types and parameters.

[26] Scollnik, David P.M., "Modeling Size of Loss Distributions for Exact Data in WinBUGS," *Journal of Actuarial Practice* 10 (1&2), Lincoln, NE: Absalom Press, 2001, pp. 202-227.
**Topics:** Reserves
Ratemaking; Reinsurance Research
**Type of Material:** Refereed Paper/Article
**Categories:** 3
**Comments:** This paper discusses how the statistical software WinBUGS can be used to implement a Bayesian analysis of several popular severity models applied to exact size-of-loss data. The particular models targeted are the gamma, inverse gamma, loggamma, lognormal, (two-parameter) Pareto, inverse (two-parameter) Pareto, Weibull, and inverse Weibull distributions. It is possible to implement additional size-of-loss models (including those for truncated data) using analogous methods.

# BIBLIOGRAPHY

**Categories**

1. General Interest
2. Building Models
3. Testing Simulated Data
4. Testing Reserving Methods

**Abbreviations**

CAS – Casualty Actuarial Society
IAA – International Actuarial Association
SOA – Society of Actuaries

[27] Stanard, James N., "A Simulation Test of Prediction Errors of Loss Reserve Estimation Techniques," *Proceedings of the CAS* LXXII, 1985, pp.124-148, http://www.casact.org/pubs/proceed/proceed85/85124.pdf.
**Topics:** Reserves; Risk Theory
**Type of Material:** Refereed Paper/Article
**Categories:** 2,4
**Comments:** Contains fully described simulation models.
**Abstract:** This paper uses a computer simulation model to measure the expected value and variance of prediction errors of four simple methods of estimating loss reserves. Two of these methods are new to the *Proceedings*. The simulated data triangles that are tested are meant to represent sample sizes typically found in individual risk rating situations. The results indicate that the commonly used age-to-age factor approach gives biased estimates and is inferior to the three other methods tested. Theoretical arguments for the source of this bias and a comparison of two of the methods are presented in the Appendices.

[28] Tableman, Mara, and Jong Sung Kim, *Survival Analysis Using S: An Analysis of Time-to-Event Data,* Boca Raton, FL: Chapman and Hall/CRC, 2005.
**Topics:** Survival Models
**Type of Material:** Book
**Categories:** 1,2
**Comments:** This book covers parametric and non-parametric survival models and provides modeling examples in the statistical language S (note: the language R is closely related to S). Survival models are useful in modeling time-to-event variables such as report lag and settlement lag and also claim size distribution. They handle censored data well.

[29] Vaughan, Richard L., "Some Extensions of J.N. Stanard's Simulation Model for Loss Reserving," *CAS Forum*, Fall 1998, pp. 415-498, http://www.casact.org/pubs/forum/98fforum/vaughn.pdf .
**Topics:** Reserves
**Type of Material:** Non-Refereed Paper/Article
**Categories:** 2,4
**Comments:** The loss process model and simulation procedures proposed by James M. Stanard in 1985 are extended in numerous ways, including provision for serial autocorrelation of parameters, mixtures of claim types, conditional selection of sample points, and a much greater variety of reserving methods. The extended model is used to explore many questions arising in practical loss reserving and to assist the loss reserver in choosing the best estimator for particular data conditions.

[30] Venables, W.N., and B.D. Ripley, *Modern Applied Statistics with S*, 4th Ed., New York: Springer-Verlag, 2002.
**Topics:** Statistics and R language
**Type of Material:** Book
**Categories:** 2
**Comments:** This book is the most well-known book covering R (and the related language S+) and its application to data analysis. The first part of the book discusses the structure and basic elements of the language.

# BIBLIOGRAPHY

**Categories**

1. General Interest
2. Building Models
3. Testing Simulated Data
4. Testing Reserving Methods

**Abbreviations**

CAS – Casualty Actuarial Society
IAA – International Actuarial Association
SOA – Society of Actuaries

[31] Venter, Gary G., "Testing the Assumptions of Age-To-Age Factors," *Proceedings of the CAS* LXXXV, 1998, pp. 807-847, http://www.casact.org/pubs/proceed/proceed98/980807.pdf .
**Type of Material:** Refereed Paper/Article
**Categories:** 2,4
**Comments:** The use of age-to-age factors applied to cumulative losses has been shown to produce least-squares optimal reserve estimates when certain assumptions are met. Tests of these assumptions are introduced, most of which derive from regression diagnostic methods. Failures of various tests lead to specific alternative methods of loss development.

[32] Venter, Gary G., "Transformed Beta and Gamma Distributions and Aggregate Losses," *Proceedings of the CAS* LXX, 1983, pp. 156-193, http://www.casact.org/pubs/proceed/proceed83/83156.pdf.
**Type of Material:** Refereed Paper/Article
**Categories:** 2
**Abstract:** Distribution functions are introduced based on power transformations of beta and gamma distributions, and properties of these distributions are discussed. The gamma, beta, F, Pareto, Burr, Weibull, and loglogistic distributions are considered. The transformed gamma is used to model aggregate distributions by matching moments. The transformed beta is used to account for parameter uncertainty in this model. Calculation procedures are discussed and APL program listings are included. The transformed gamma distribution is compared to exact methods of computing the aggregate distribution function based on the entire frequency and severity distributions.

## Acknowledgment

The authors acknowledge Mark Shapland for his vision in creating the LSMWP and for his encouragement and very helpful advice as work proceeded. The authors also acknowledge Richard Vaughan and Huan Zhu for the very major contributions they made in developing and testing a prototype model during the LSMWP's planning phase.

## Supplementary Material

See Section 5 above on Documentation of Open-Source Model. Also, please refer to the results of our search of the actuarial literature summarized in the Bibliography.

**Abbreviations and notations**

AL-BI, Auto Liability Bodily Injury
AL-PD, Auto Liability Property Damage
APD, Auto Physical Damage
AY, accident year
DRM Committee, Committee on Dynamic Risk Modeling
GLM, Generalized Linear Model
K-S test, Kolmogorov-Smirnov test
LSMWP, Loss Simulation Model Working Party
MLE, Maximum Likelihood Estimation
NLL, Negative Log-Likelihood
P-P plot, Probability-Probability or Percent-Percent plot
Q-Q plot, a Quantile-Quantile plot
R, an implementation of the S programming language
S, a statistical programming language
S-PLUS, a commercial implementation of the S programming language sold by TIBCO Software Inc.

## Working Party Oversight, Model Building, and Testing Team Members

**Robert Bear** is principal and founder of RAB Actuarial Solutions LLC, which offers the following consulting services: (1) loss reserve studies and research on loss reserving methods and models; (2) development of dynamic risk models to facilitate evaluation of profitability as well as risk load and capital needs; (3) insurance and reinsurance pricing, including reinsurance commutation, excess pricing, and price monitoring studies; (4) resolution of loss reserve and coverage disputes subject to insurance arbitration, reinsurance arbitration, or mediation; and (5) actuarial and reinsurance expert witness and litigation support.

Bear currently serves as Chairperson of the CAS Dynamic Risk Modeling Committee. He previously served as Chairperson of the Reinsurance Association of America Actuarial Committee and as President of Casualty Actuaries in Reinsurance. He has authored several CAS discussion papers and articles on reinsurance pricing, loss reserving, and risk modeling issues. Additional information is available on his Web site, www.rabsolutions.net.

**Joseph Marker,** FCAS. MAAA, teaches Actuarial Science and Financial Mathematics at the University of Michigan. Joe is principal and co-founder of Marker Actuarial Services, LLC, which he and his wife Connie started in 2001. Marker Actuarial provides property-casualty actuarial and management consulting services. Actuarial consulting concentrates on predictive modeling, pricing, loss modeling, financial analysis, actuarial management, and service to small and mid-size insurers. Additional information is available at www.markeractuarial.com.

Prior to consulting, Joe worked for four organizations in various actuarial capacities for 28 years, the last fifteen years as Chief Actuary at two regional insurance companies.

He is a past president of the Midwestern Actuarial Forum. He published the paper, "Studying Policy Retention using Markov Chains" (*PCAS* LXXXV, 1998). Joe also co-authored the paper "Rating Claims-Made Insurance Policies" (CAS *Discussion Paper Program* 1980). This paper was part of the CAS exam syllabus for many years.

**Hai You** is the software developer of the Loss Simulation Model that is the subject of this paper and has co-authored sections of this paper that document the model as well as the associated documentation within the program and on the CAS LSMWP Web page. He also contributed software that assisted with the testing of the Loss Simulation Model.

As vice president of technology at Goouon, Hai is responsible for actuarial modeling, product development and innovative solutions. He is the author of ReserveMaster, the cutting-edge loss reserving platform for P&C insurers. Hai developed the Illustration Copula and contributed to the CAS. His three-day-project of Two Stage Bootstrap in R language has had more than 100 downloads since its creation. Before co-founding Goouon, Hai has served as an IT consultant in various insurance companies for 10 years, where he got inside knowledge of the core systems such as underwriting, claims, and ratemaking. Hai has mastered various technologies including, computer languages, securities, databases, 3-D, e-mail, UI, and data communication. He also believes that Object-Oriented-R is 10 times stronger than R. Just for fun, he developed his own games like ChessBot, Maze (using AStar), and Mine Digger, among others.

Goouon is an actuarial engineering company where people integrate actuarial knowledge and analytical solutions into services. The company offers consulting services on pricing, reserving, statistical modeling, data mining, and customized actuarial solutions. Besides ReserveMaster, the company also provides tailor-cut Loss Data Processing, Health Insurance Solutions, iERM-Insurance Enterprise Risk Management and Depict - a drag-and-drop Enterprise Workflow Management platform. Please visit http://www.goouon.com for additional information.

**Glenn Meyers** is Vice President of Research for ISO Innovative Analytics. He earned a Ph.D. in mathematics from the State University of New York at Albany and is a Fellow of the Casualty Actuarial Society and a member of the American Academy of Actuaries. Before joining ISO in 1988, Glenn worked at CNA Insurance Companies and the University of Iowa.

Glenn's current responsibilities at ISO include the development of insurance scoring products using predictive modeling techniques. Prior responsibilities have included working on ISO Capital Management products, increased limits, catastrophe ratemaking, and reinsurance products.

Glenn's work has been published in *Proceedings of the Casualty Actuarial Society* and the CAS peer-reviewed journal, *Variance*. He also writes "Brainstorms," a regular column in the *Actuarial Review*. His papers have won numerous awards, and he is a frequent speaker at CAS meetings and seminars.

His service to the CAS includes membership on various education and research committees. He has served on the CAS Board of Directors and currently serves as the IAA's delegate to ASTIN. His work on the LSMWP was to develop statistical tests to compare the outcomes of a simulation model with outcomes produced by real data.

**Curtis Parker**, FCAS, CPCU, is Vice President and Chief Actuary for Grange Insurance Companies. Curt is an actuary with 36 years of varying experiences in the property/casualty insurance industry, including personal and commercial lines underwriting, pricing, product research and development, market planning, reserving, and predictive modeling. He has volunteered over the years on examination, ratemaking, and reserving committees and seminar panels. He served as chairperson for the subcommittee of the LSMWP, which completed research and documented literature providing general and more technical readings related to loss simulation considerations and techniques. His subcommittee was also charged with development of initial methods for testing output from the model with respect to the ability to distinguish it from actual historic data. The subcommittee is indebted to Glenn Myers for his significant contributions with this later portion of the subcommittee's charge.

**Kailan Shang** works in the area of financial risk management in AIA. Prior to this, he worked as a pricing actuary in a life insurance company for two and a half years. Years of actuarial and risk management experience has allowed him to get a broad exposure, including economic capital, market-consistent embedded value (MCEV), financial engineering, dynamic management options, dynamic policyholder behavior modeling, product development and management, U.S. GAAP reporting, dynamic solvency testing, and the like.

As an FSA, CFA, PRM and SCJP, he is also an enthusiast of actuarial research through both volunteer works and funded research program. He participated in the IAA Comprehensive Actuarial Risk Evaluation project and he is now working as the lead researcher on the SOA research project, "Valuation of Embedded Option in Pension Plan.".

**Yuting Yang** is a graduate student in actuarial science and financial mathematics at the University of Michigan. Prior to attending Michigan, she earned a master's degree in mathematics from the University of California at Davis. At UC-Davis, she did research in probability, random matrices, and convex geometry. Yuting did most of the work in testing the severity distributions in section 6.2.2.

## Working Party Members

Robert A. Bear, Co-Chairperson
Mark R. Shapland, Co-Chairperson

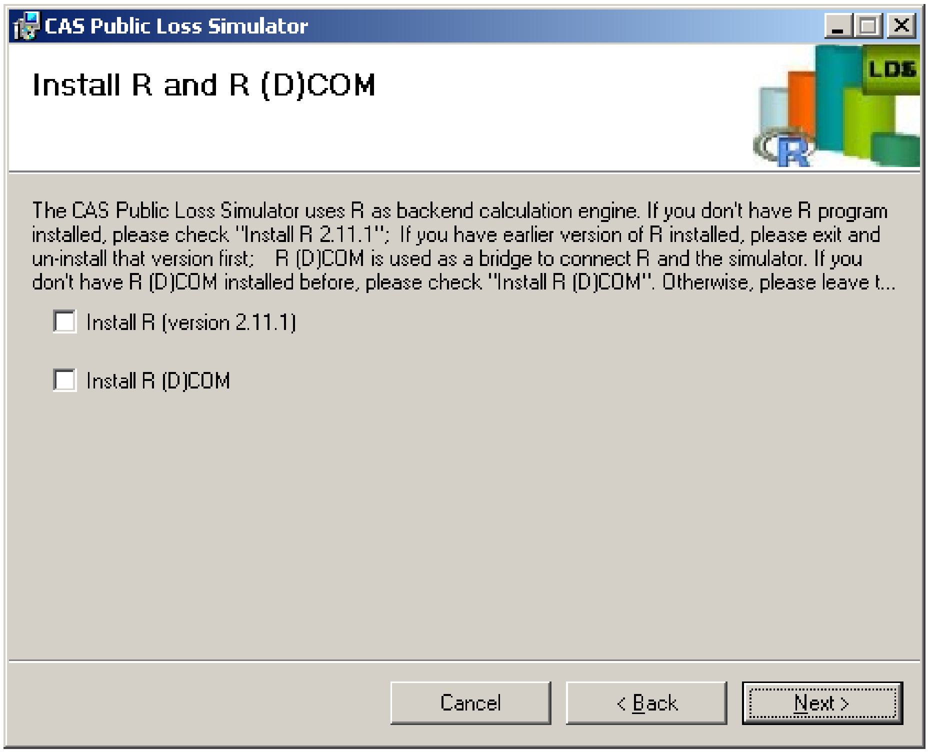
Ramzi AbuJamra
Shobhit Awasthi
Hassan A. Ayoub
Glen Barnett
Nebojsa Bojer
Bhaskar Chattaraj
Denise L. Cheung
Wei Chuang
Kevin M. Cleary
Catherine Cresswell
Salvatore Forte
Bradford S. Gile
Spencer M. Gluck
Songling Guo
Thomas Hartl
Ping-Hung Hsieh
Nicole Huang
Turab Hussain
Li Hwan Hwang
Jan I. Iwanik
Julia Jacobi
Shiwen Jiang
Nancy A. Kelley
Stephen Jacob Koca
Andrew M. Koren
Scott C. Kurban
Kin Hoe Lee
Kin Yee Lee
Stephen L. Lienhard
Joseph O. Marker
Glenn G. Meyers
Jonathan E. Miller
Jie Min
F. James Mohl
Curtis M. Parker
Marco Pirra
Arlie J. Proctor
Ralph Stephen Pulis
Keith A. Rogers
Manalur S. Sandilya
Kailan Shang
Catherine E. Staats
Christopher M. Steinbach
Lin Yee Tan
Varsha A. Tantri
Jack T. Tower
Daniel M. Van der Zee
Justin M. VanOpdorp
Richard L. Vaughan
Gary G. Venter
Yuting Yang
Yuanhe (Edward) Yao
Bo Zhou
Hongbo Zhou
Huan Zhu
Jane E. Fulton, CAS Staff Liaison

# APPENDIX A

## 1) How to install the Public Loss Simulation Model

a. To install the Loss Simulator Model to your computer, please download and run the windows installation package LossSimulatorSetup.msi from the following link http://www.casact.org/Research/LSMWP/.

b. Following the initial screen instructions, the installation package will ask you two questions, as shown in Picture (1). The model runs R as the background calculation engine, and requires R (D) COM as a bridge component between R and the front end application. If you have not installed these two services before, please check them and continue. If you have earlier version of R installed, please exit and uninstall that version first, then come back and check the "Install R (version 2.11.1)." Ignore the "Install R" option if you have higher version of R installed already.



CAS Public Loss Simulator

Install R and R (D)COM

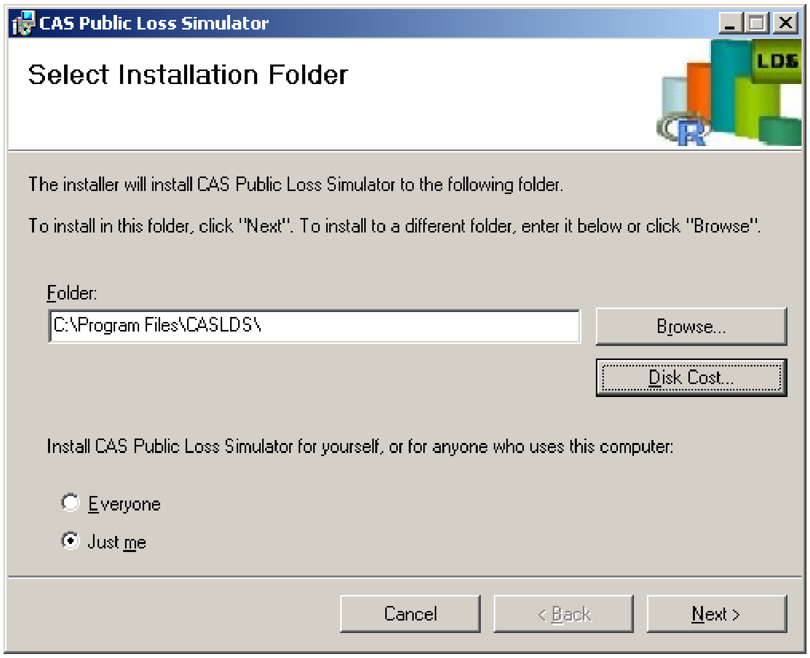
The CAS Public Loss Simulator uses R as backend calculation engine. If you don't have R program installed, please check "Install R 2.11.1"; If you have earlier version of R installed, please exit and un-install that version first; R (D)COM is used as a bridge to connect R and the simulator. If you don't have R (D)COM installed before, please check "Install R (D)COM". Otherwise, please leave t...

□ Install R (version 2.11.1)

□ Install R (D)COM

**Picture (1). Installation Options**

c. Then provide an installation location, as shown in Picture (2).



CAS Public Loss Simulator

Select Installation Folder

The installer will install CAS Public Loss Simulator to the following folder.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

Folder: C:\Program Files\CASLDS\

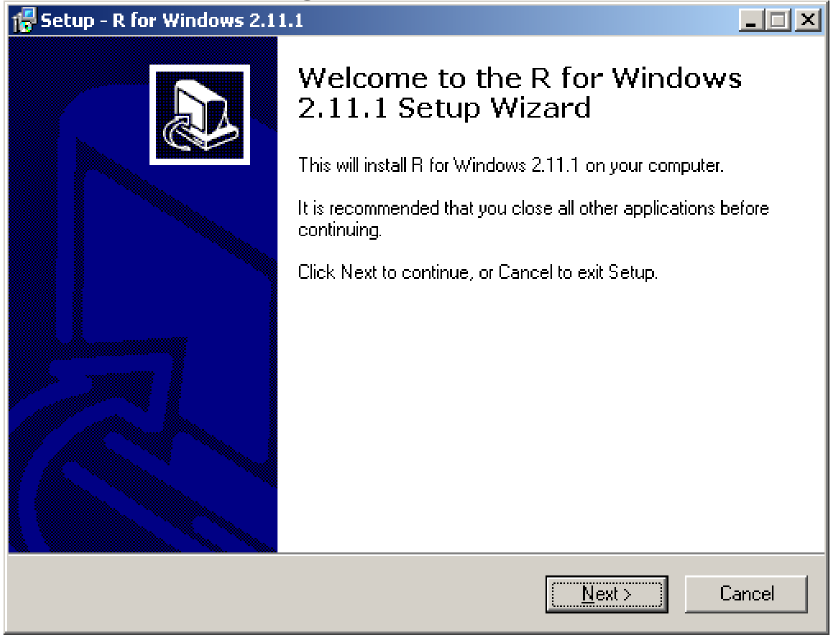
Install CAS Public Loss Simulator for yourself, or for anyone who uses this computer:

○ Everyone

● Just me

**Picture (2). Choose installation directory.**

d. Click the next button to continue the installation. **If you choose to install R 2.11.1 and R (D)COM, the following two screens will start automatically.**



Setup - R for Windows 2.11.1

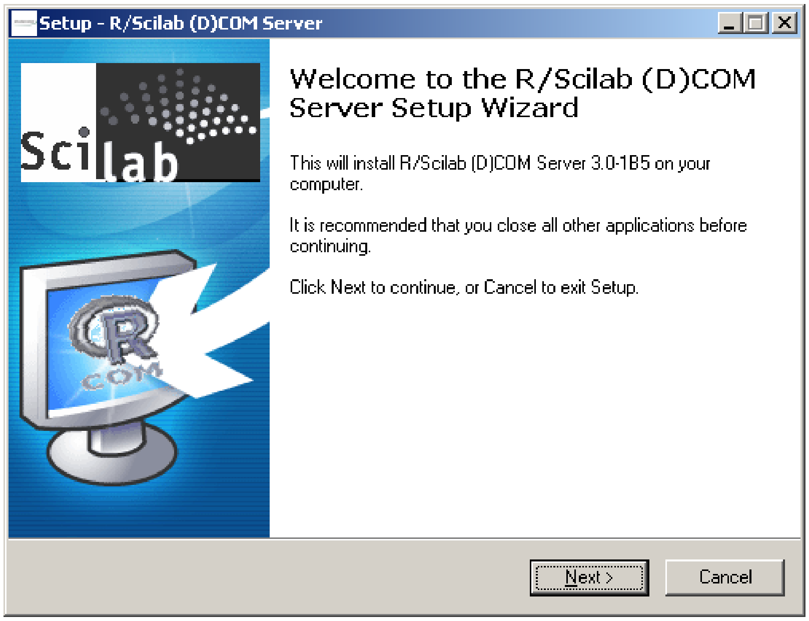
Welcome to the R for Windows 2.11.1 Setup Wizard

This will install R for Windows 2.11.1 on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

**Picture (3). Install R 2.11.1.**



Setup - R/Scilab (D)COM Server

Welcome to the R/Scilab (D)COM Server Setup Wizard

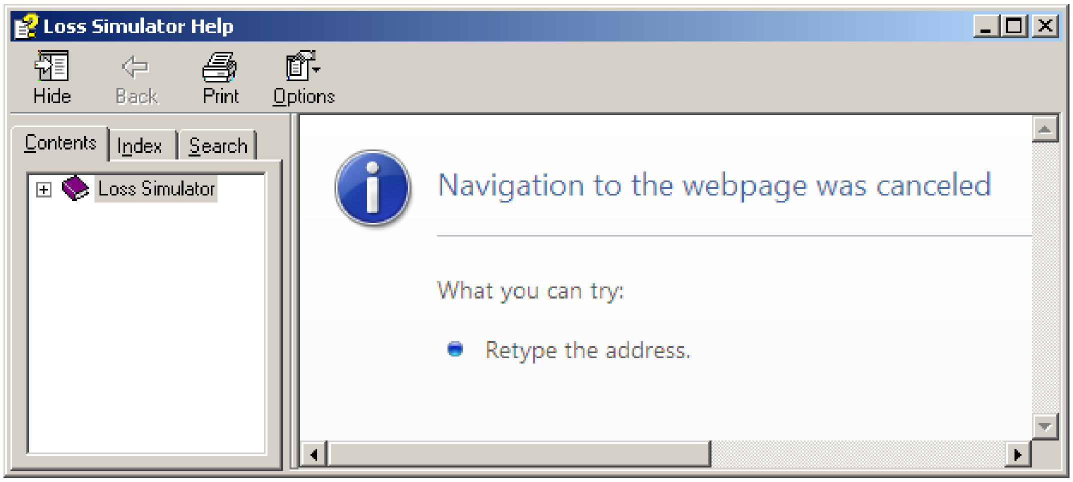
This will install R/Scilab (D)COM Server 3.0-1B5 on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

**Picture (4). Install R (D) COM.**

e. There is one more thing to be aware: besides the CAS online help at http://www.casact.org/research/lsmwp/losshelp/index.cfm?fa=main, Loss Simulator also contains a windows html help file as an attached help system. But due to Windows security restriction (http://support.microsoft.com/kb/902225), you may not be able to see the help content when launching help from the Simulator, and get the following error page instead:



Loss Simulator Help

Navigation to the webpage was canceled

What you can try:

- Retype the address.

**Picture (5). Possible error when trying to open help file from Loss Simulator help menu.**

In this case, you can just simply go to the **model installation folder** and right click the help file named **LossSimulator.chm**, and then select **Properties,** click **Unblock.** After this action, you will be able to see the help content.

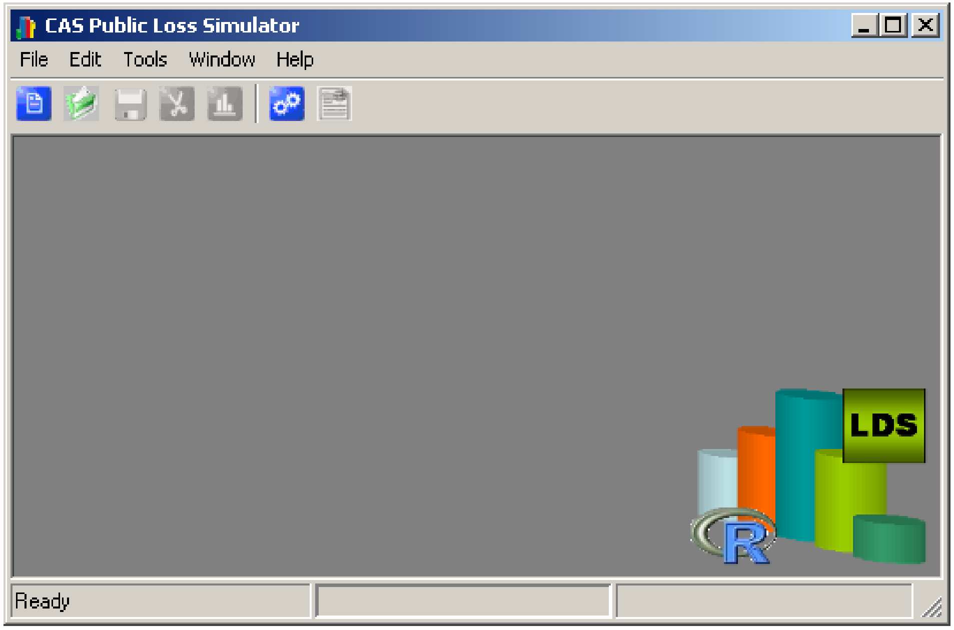
## 2) How to run the Public Loss Simulation Model

Public Loss Simulator.lnk

After installation, you will find the Public Simulator icon on both your desktop and Start Menu. You can just double click it to launch the application.

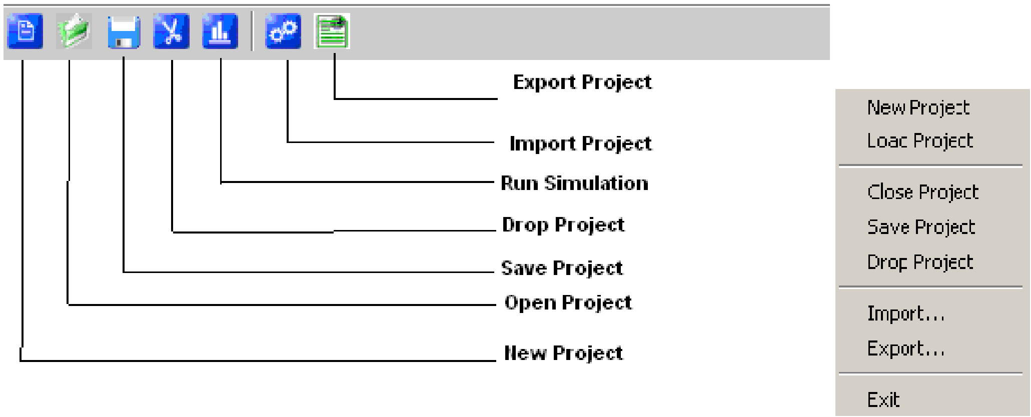
### 2.1 System Overview

Initial screen will be like Picture (6). The application is developed within Windows UI standard, so that it contains menus and tool bar buttons associated with each menu items.



**Picture (6). Public Loss Simulator main window**

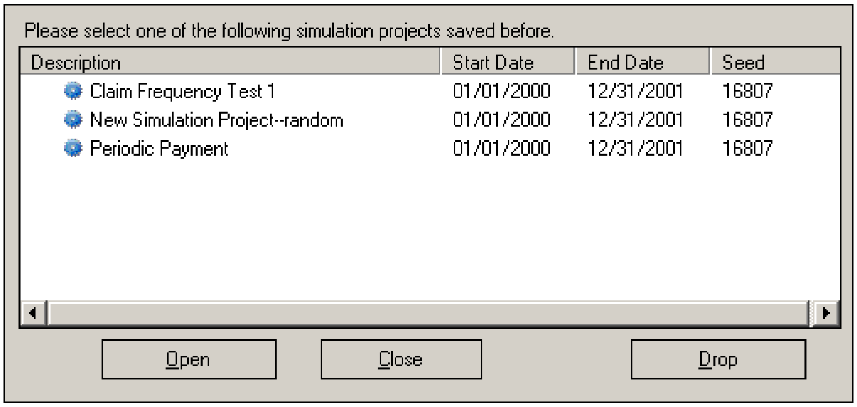
Picture (7) shows a brief explanation of each menu item and the later chapters will explain each of them in detail.



**Picture (7). System Tool Bars and Menus**

**File, New Project:** Initialize a brand new simulation project, which contains default settings for one line, with one Single Payment type. You can save it later.

**File, Load Project:** Open a previously saved simulation project from the database.

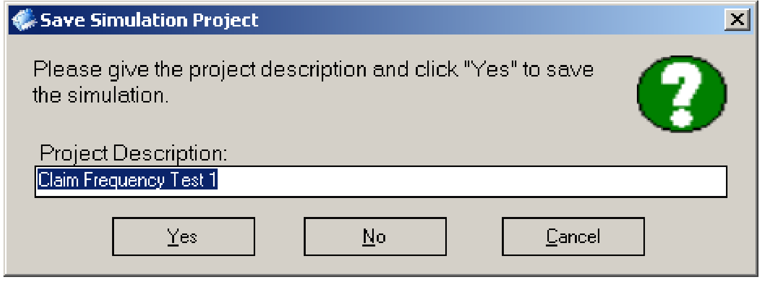

Please select one of the following simulation projects saved before.

| Description | Start Date | End Date | Seed |
|---|---|---|---|
| Claim Frequency Test 1 | 01/01/2000 | 12/31/2001 | 16807 |
| New Simulation Project--random | 01/01/2000 | 12/31/2001 | 16807 |
| Periodic Payment | 01/01/2000 | 12/31/2001 | 16807 |

**Picture (8). Load previous saved project from database.**

**File, Close Project:** Close the current opened project. If system detects any change to the project properties, it will first ask for saving the project.

**File, Save Project:** Save the current opened project into database. When saving a simulation project, all the simulation properties will be saved into database. If it is initialized from a new project menu, system will ask you for a project description, as shown in Picture (9). System has a default MS Access database attached; you can configure the simulator to connect to any kind of database server also.



Save Simulation Project

Please give the project description and click "Yes" to save the simulation.

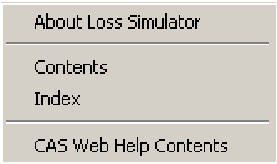
Project Description: Claim Frequency Test 1

**Picture (9). Project Save Window will pop up asking for project description.**

**File, Drop Project:** Permanently delete a project previously saved in the database. This menu item is enabled when a project is open.

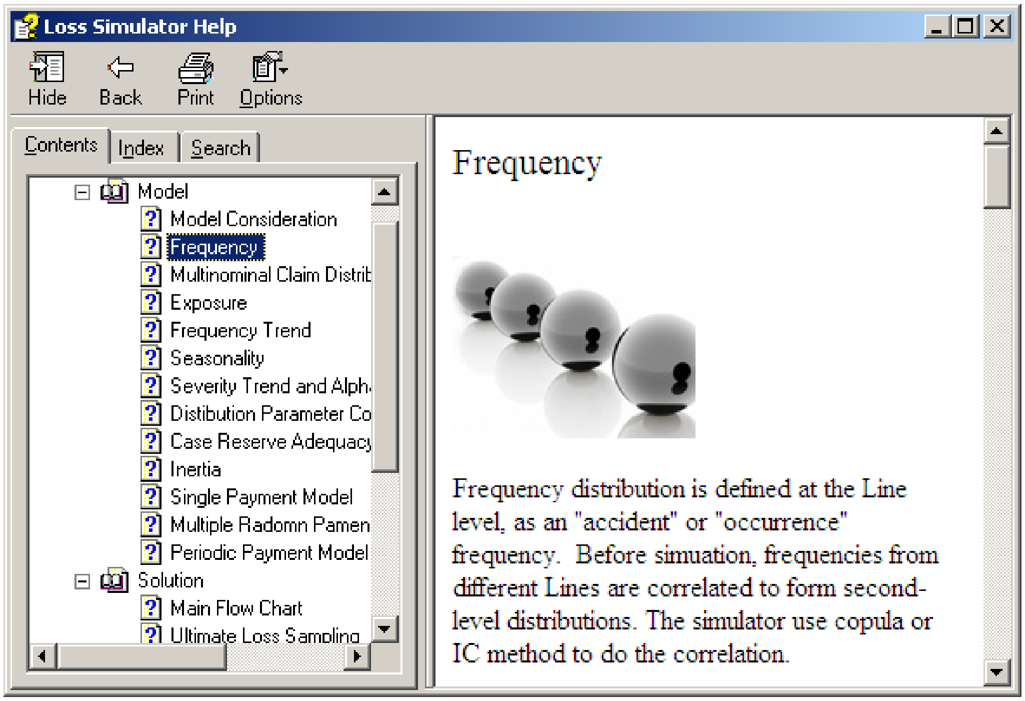
**File, Import… :**
**File, Export… :**
These two menus enable project sharing among users. You can export a project into a XML file and e-mail it to another person. That person can import it and run simulation from his workstation, with all the project properties set by you. Or you can export the project and later import it again for different testing stage purpose. In that case, it is pretty similar to saving project to the database and loading project back.

This is the help menu.



**Help, CAS Web Help Contents:** This will lead users to the help contents from CAS Loss Simulator Working Party Web Site.

**Help, Contents:** This will open the attached simulator help system, as shown in Picture (10). **If you cannot see any help contents from right side panel, please refer to section 7.1 (e) in this paper for proper configuration**.

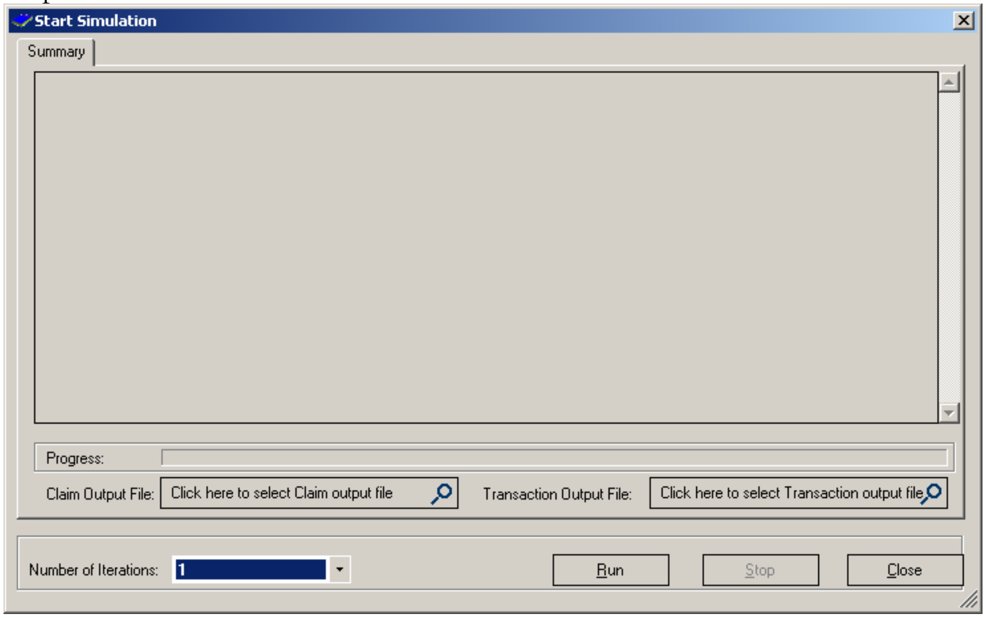


Frequency

Frequency distribution is defined at the Line level, as an "accident" or "occurrence" frequency. Before simuation, frequencies from different Lines are correlated to form second-level distributions. The simulator use copula or IC method to do the correlation.

**Picture (10). Loss Simulator Help system**

**Tools, Run Simulation:** This menu will let you start to run the simulation. As shown in Picture (11), you need to provide a claim file name, a transaction file name, and depending upon your needs, choose how many iterations you prefer. For a full-scale simulation, you may try 1000 iterations. That could take hours to finish, generating hundred thousands of claims and millions of transactions. We will explain the simulation result in detail from later chapter.



Start Simulation

Progress:

Claim Output File: Click here to select Claim output file

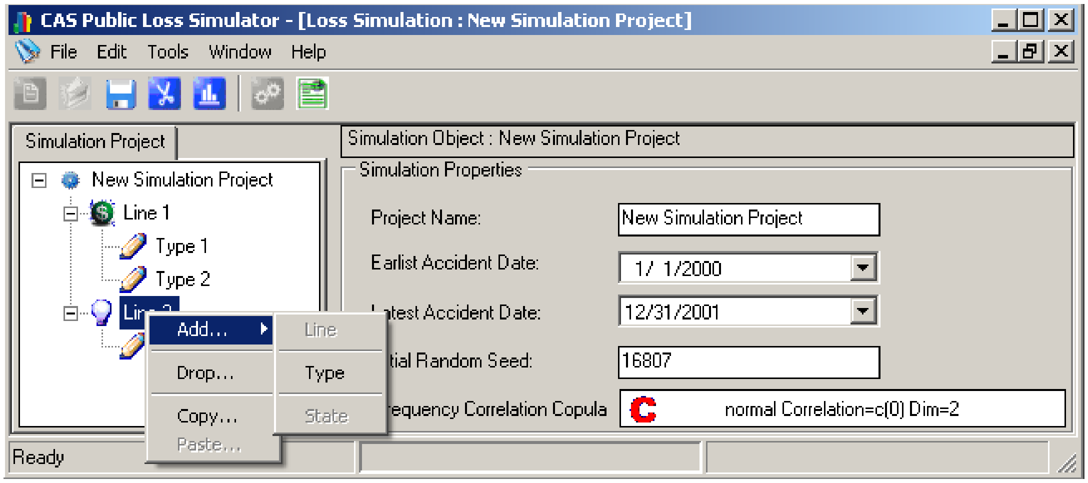
Transaction Output File: Click here to select Transaction output file

Number of Iterations: 1

**Picture (11). Run Simulation window**

## 2.2 Simulation Project

A simulation project is one where modeler enters properties for line of businesses and types and initializes the simulation. A typical project can contain multiple lines of business (LOB), with each line of business consisting of one or more types of claims. Type can be treated as coverage in real case. A tree structure is the best way to describe the relationships among them, as shown in Picture (12).



Simulation Object : New Simulation Project

Simulation Properties

Project Name: New Simulation Project

Earlist Accident Date: 1/ 1/2000

…test Accident Date: 12/31/2001

…tial Random Seed: 16807

…equency Correlation Copula: normal Correlation=c(0) Dim=2

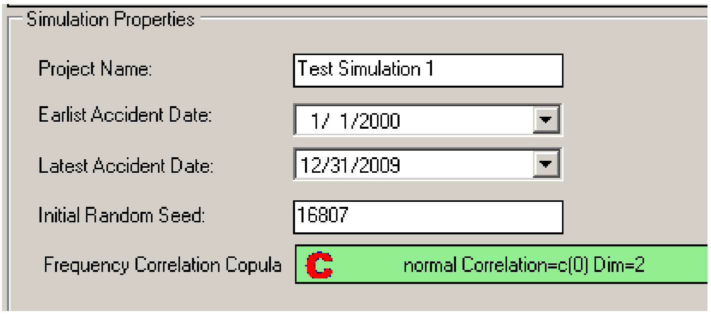
**Picture (12). A simulation project is constructed by tree structure.**

**The left panel in the project window uses popup menus** to allow user to add, drop, update, copy, and paste nodes to build up a company business model; while the right panel of the project displays properties of the selected node from left side. It could be a project properties, line level properties, or type level properties. This section of the paper won't give a detailed explanation of each property listed, most of which are explained in the help file or the CAS Web Site online help for this model.

## 2.3 Overall Simulation Properties

According to the tree relationship described in previous section, the simulation project will have properties that control line level correlations, as well as main simulation features such as the earliest accident date and the latest accident date.

As seen in Picture (13) below, when you double click the Simulation Project node, you will see the screen on the right side panel. You can define a 10-year accident date range, starting from 01/01/2000 to 12/31/2009, as shown in the sample. This is a common time frame for most loss reserve analysis. In the later simulation stage, system will loop through each year and apply the line level properties, such as frequencies, etc., to generate synthetic claims and transactions.



Simulation Properties

Project Name: Test Simulation 1

Earlist Accident Date: 1/ 1/2000

Latest Accident Date: 12/31/2009

Initial Random Seed: 16807

Frequency Correlation Copula: normal Correlation=c(0) Dim=2

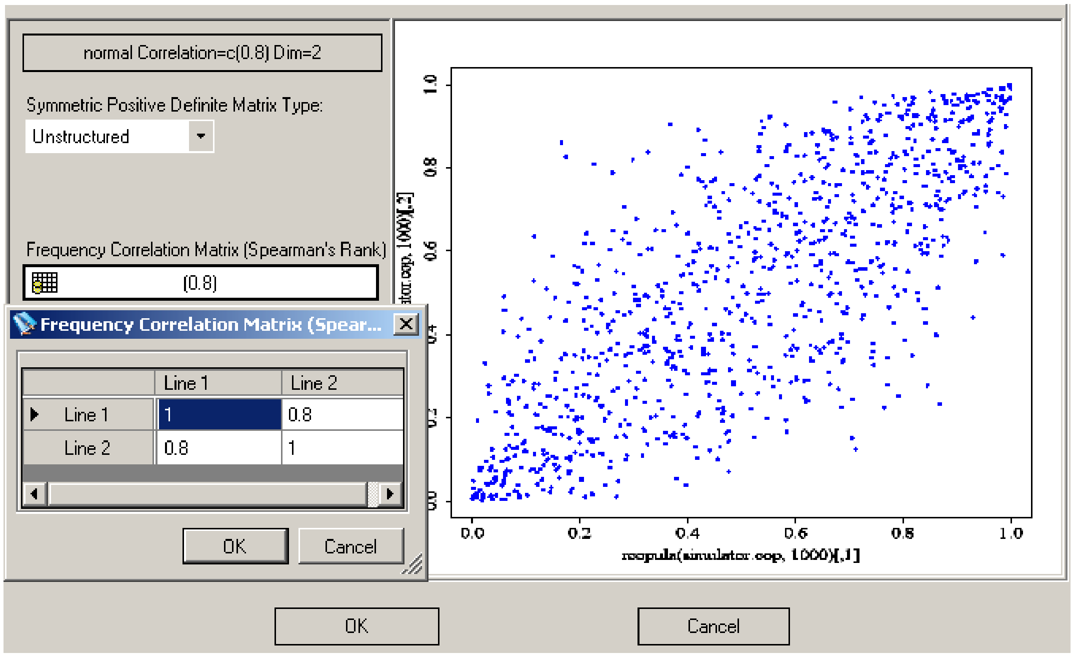
**Picture (13). Overall Simulation Properties**

The **Initial Random Seed** (or **seed state**, or just **seed**) is a number used to initialize a pseudorandom number generator. The choice of a good random seed is crucial in the field of computer security. However, in our simulator it is mainly used for testing the simulated result. By fixing the seed, we ensure that users can generate the same result with the same seed number when the simulation is run under the same assumptions. Please note that the simulation results are still truly randomly generated when we permit the seed to be randomly selected.

The correlation among line level frequencies is defined by **Copula**. Please go to the help system to read more about how the copula is applied in the simulator on a multivariate distribution. Or you can read more copula applications from R help.

You can **ignore** copula correlation with the default value set, as seen in Picture (13). The green button shows property of a two-dimension Normal Copula, but correlation is c(0). The button is enabled only for multiple lines of businesses. In our example, it has two dimensions because the sample created two lines of businesses under the simulation project, and so there are two marginal frequency distributions to be correlated. The **c(**0**)** correlation means there is no correlation between the variants. Thus when simulating, system will instead bypass the copula calculation and run the typical distribution individually defined by those variants.

When you are confident enough to apply Copula for correlations, you can click the green button above, and see the Picture (14) for a typical copula screen. You will notice a two-dimensional copula scatter plot on the right side of the screen. In this sample, it is a **normal copula**. Please pay attention to the shape of the plot. The simulator will show copula plot up to three dimensions. The plot will be refreshed once you change any property. Correlation Matrix can be entered by clicking the button of "Frequency Correlation Matrix". It is actually the Spearman's Rank matrix for normal copula, as sampled. The matrix is symmetrical, so that you can just give upper-left values, and the system will fill in the lower-right corner values automatically.



normal Correlation=c(0.8) Dim=2

Symmetric Positive Definite Matrix Type: Unstructured

Frequency Correlation Matrix (Spearman's Rank) (0.8)

Frequency Correlation Matrix (Spear...

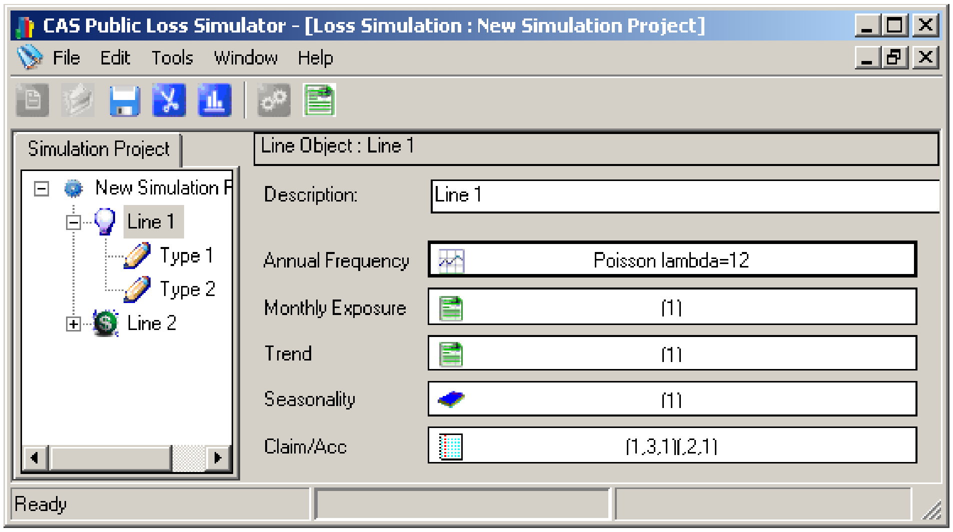
|  | Line 1 | Line 2 |
|---|---|---|
| Line 1 | 1 | 0.8 |
| Line 2 | 0.8 | 1 |

**Picture (14). Define a Copula between LOB frequencies.**

The marginal distributions of this copula will be the line level Frequencies.

## 2.4 Line Level Properties

The Simulator defines most of frequency related properties at the line of business level, as shown in Picture (15). They are Annual Frequency, Monthly Exposure, Trend, Seasonality, and Multinomial Claim Distribution among types. The help system has provided very detailed explanations on each property topic and how monthly frequency is generated from an Annual Frequency by these properties. The help system also illustrates the detail workflow of: monthly frequency ->occurrences->claims->claims for each type. So, in this section of the paper, we won't go through them in detail again, but instead, will focus on how to enter those values.



Line Object : Line 1

Description: Line 1

Annual Frequency: Poisson lambda=12

Monthly Exposure: [1]

Trend: [1]

Seasonality: [1]

Claim/Acc: [1,3,1][,2,1]

**Picture (15). Line level properties**

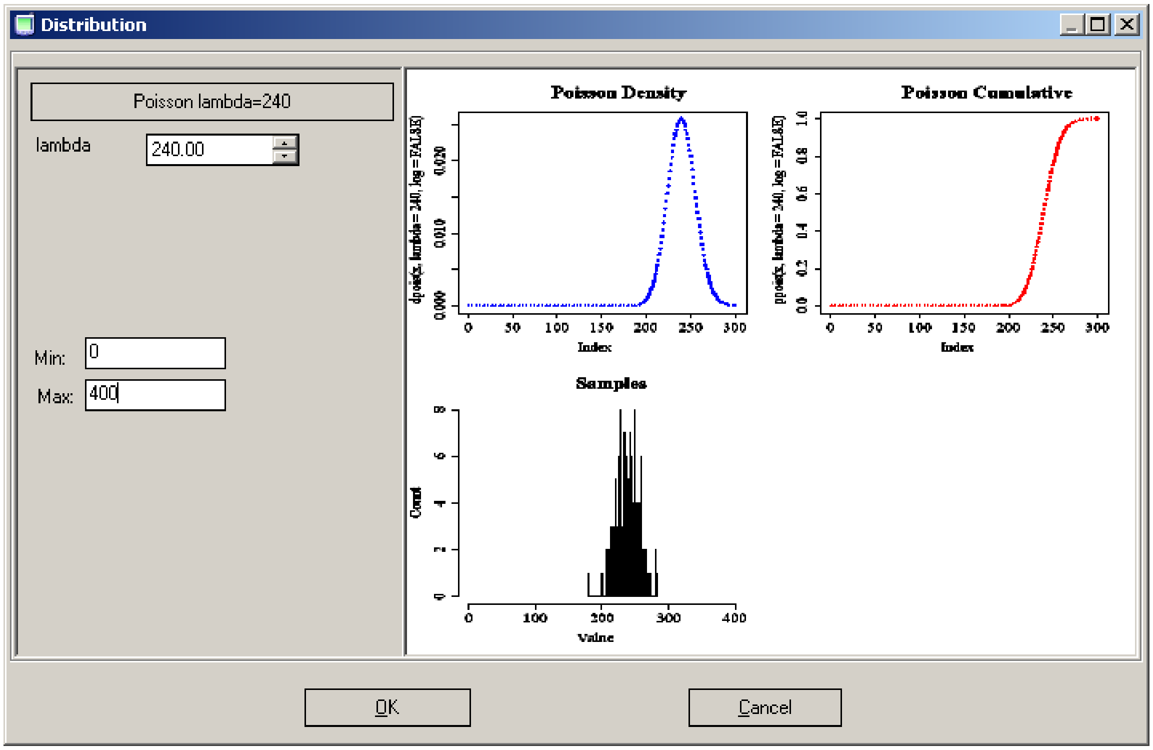
### 2.4.1) How to define distribution in simulator

**Annual Frequency** is a distribution value. The button on the right side will show the little icon, and lists out the distribution property, such as, it is a Poisson distribution with lambda = 12. This is a very small sample. You can change it by clicking the button, and you will see the screen as Picture (16).

In fact, all the distributions, either continuous or discrete, are defined with screens similar to Picture (16). Any distribution defined in the simulator will have properties such as name, parameters, and min/max value. The upper-left corner contains a button that shows the currently selected distribution. You can change the distribution by clicking it and picking from a dropdown list. Then you can change the parameter(s) for the selected distribution. Every time you make a change, you will notice that distribution graphs such as Density, CDF, and a Histogram are refreshed on the right side of the panel.

The **min/max values** are applied when sampling from the defined distribution, so that any generated random number will be limited to the [min, max] range. With this said, when the associated distribution is the severity distribution, the max value also physically represents the **coverage limit**.

The Histogram shown in the picture is a visual representation, between the [min, max] values, of the possible random numbers sampled in the future simulation run.



Poisson lambda=240

lambda 240.00

Min: 0

Max: 400

**Picture (16). A Poisson distribution**

All the distributions in the simulator are defined in the same manner as is done in the R language. For example, the user enters the meanlog (μ) and sdlog (sigma) parameters for all lognormal distributions in the model. Thus, if you want to define the size of loss to be lognormal with mean $100,000 and standard deviation $100,000, you need to perform a **conversion calculation** first.

To achieve the given size of loss distribution above, you can set the parameters:

sigma = sdlog = {ln(1+CV^2)}^.5, where CV = mean/(standard deviation).
Since CV = 100,000/100,000 = 1 in our example, sigma = {ln(2)}^.5 = .833 and
*μ = meanlog = ln(mean) – (sigma^2)/2 = ln(100,000) - .693/2 = 11.166 .*

Please recall that ln(*x*) represents the natural logarithm of *x*, while exp(*x*) = *e*^*x*, the constant *e* raised to the power *x*.

This will produce a size of loss *X* with mean

$$EX = \exp(\mu + \frac{1}{2}\sigma^2) \approx 100{,}000 \text{ and second moment } E(X^2) = \exp(2\mu + 2\sigma^2) \approx 2 \text{ x } 10^{10}.$$

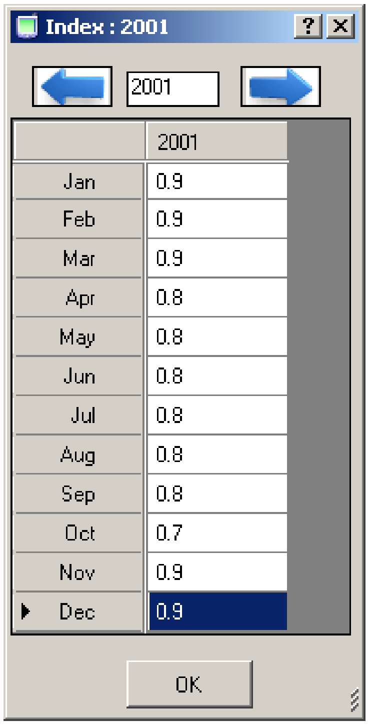
This results in Var(*X*) $=2 \text{ x } 10^{10} - 10^{10}$ and Standard Deviation(*X*) = 100,000.

### 2.4.2) How to define simulation property by month

Most of the properties defined in simulator are **monthly** values. That means their value will change by month and detail calculations are also carried out at monthly level. For instance, trend, exposure, seasonality, most of the severity distributions and lags, case reserves, etc., are all defined by month. Line level frequency is defined annually for simplicity, but it is still divided into monthly frequency in the initial simulation calculations.

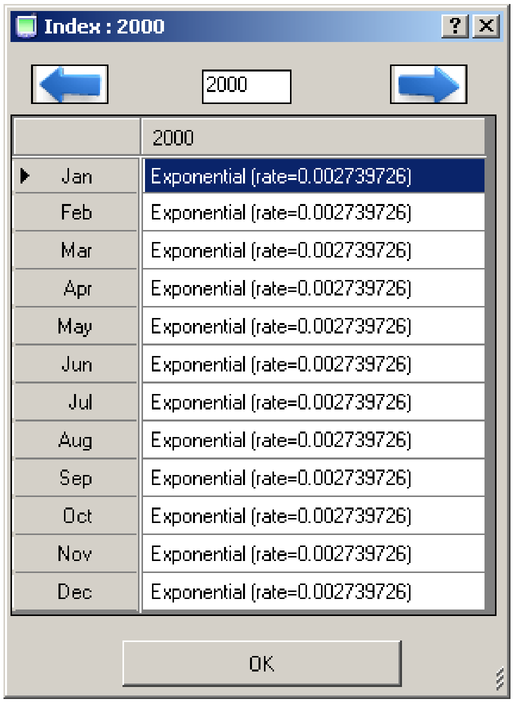
This may cause some confusion, especially if you define severity distribution or lags by month. By default, the Simulator will set those values to all be the same value. When you change a value for one particular month, that value is automatically carried through to all subsequent months. This feature gives user a flexibility to treat all kinds of probabilities in their business, and provides a powerful ability to change the distribution for a certain variant. Please see Picture (18) for a Payment Lag Distribution example defined in next chapter.

At the line level, an example of frequency trend defined by each month in 2001 is illustrated by Picture (17). You can navigate to a different year using the blue arrows. Please be aware that every time you make a change to one cell, the values after that month will be changed automatically, and will remain changed across years. So, if you just want to change only one month's value, you should be sure to change the value for later months to the previous value.



| | 2001 |
|---|---|
| Jan | 0.9 |
| Feb | 0.9 |
| Mar | 0.9 |
| Apr | 0.8 |
| May | 0.8 |
| Jun | 0.8 |
| Jul | 0.8 |
| Aug | 0.8 |
| Sep | 0.8 |
| Oct | 0.7 |
| Nov | 0.9 |
| Dec | 0.9 |

**Picture (17). Example of monthly frequency trend**



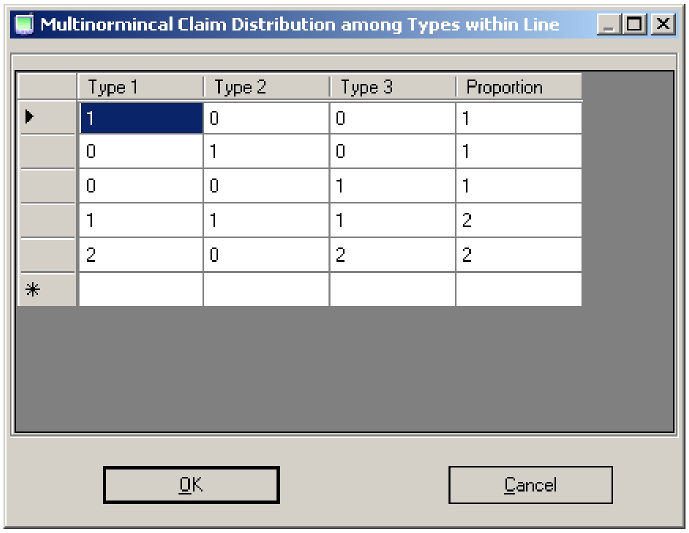
| | 2000 |
|---|---|
| Jan | Exponential (rate=0.002739726) |
| Feb | Exponential (rate=0.002739726) |
| Mar | Exponential (rate=0.002739726) |
| Apr | Exponential (rate=0.002739726) |
| May | Exponential (rate=0.002739726) |
| Jun | Exponential (rate=0.002739726) |
| Jul | Exponential (rate=0.002739726) |
| Aug | Exponential (rate=0.002739726) |
| Sep | Exponential (rate=0.002739726) |
| Oct | Exponential (rate=0.002739726) |
| Nov | Exponential (rate=0.002739726) |
| Dec | Exponential (rate=0.002739726) |

**Picture (18). Example of Payment Lag for each month**

### 2.4.3) How to define claim/acc value

The number of claims of each Type arising from an occurrence is modeled via a multinomial distribution. This allows for the possibility of multiple claims from the same occurrence, either of the same Type or of multiple Types. It typically introduces correlations between frequencies across pairs of Types within the same Line.

Picture (19) demonstrates a screen example of the multinomial distribution parameters entered for a line with three types defined.



| Type 1 | Type 2 | Type 3 | Proportion |
|---|---|---|---|
| 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 2 |
| 2 | 0 | 2 | 2 |
| | | | |

**Picture (19). Example of defining multinomial claim distribution among types**

In this case, the multinomial matrix is marked with blue in the following table. For example, if you have 600 occurrences generated in some month for this line, then the Simulator will run this R command at backend to allocate the 600 occurrences to each possible combination of claim types.

```
> rmultinom(1, 600, c(1,1,1,2,2))
     [,1]
[1,]   95
[2,]   85
[3,]   80
[4,]  178
[5,]  162
```

| Type1 | Type2 | Type3 | Proportion | Normalized Probability | Occurrence |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 0.1429 | 95 |
| 0 | 1 | 0 | 1 | 0.1429 | 85 |
| 0 | 0 | 1 | 1 | 0.1429 | 80 |
| 1 | 1 | 1 | 2 | 0.2857 | 178 |
| 2 | 0 | 2 | 2 | 0.2857 | 162 |

To interpret the above table, imagine that we have a die that is loaded so that the normalized probabilities in the above table represent the probability of landing on each of five sides of the die, while the probability of landing on the sixth side is 0. The Occurrence column summarizes the results of tossing the die 600 times. If we repeated this process 1,000 times and summed the resulting Occurrence columns from these simulations, we would expect that approximately 14.29% of the total occurrences generated would have one Type 1 claim and no Type 2 or Type 3 claims, 14.29% of the total occurrences generated would have one Type 2 claim and no Type 1 or Type 3 claims, 14.29% of the total occurrences generated would have one Type 3 claim and no Type 1 or Type 2 claims, 28.57% of the total occurrences generated would have one claim of each Type, and 28.57% of the total occurrences generated would have two Type 1 claims, no Type 2 claims and two Type 3 claims. **Claims generated from same occurrence will have the same accident date.**

The user can enter more combinations in the cells marked with * on the left, as shown in the Picture (19). They can also just highlight a row and hit delete to remove a proportion. However, please note that if you add another row of data such as (1, 0, 0, 3), since the allocation of "1, 0, 0" already exists, the Simulator will combine them into one allocation as (1, 0, 0, 4).

Please refer to the Loss Simulator Help section 5.4 - Multinomial Claim Distribution among Types for more technical model details on this topic.

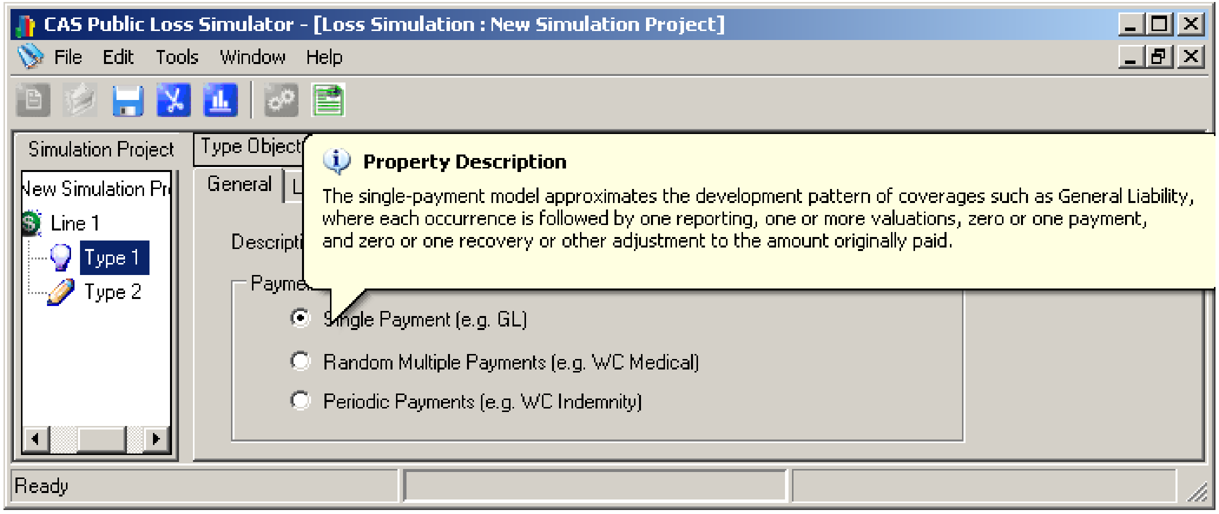
## 2.5 Type Level Properties

The Type object defined in the Simulator can be used to define coverage for a line. It will contain most of the required coverage properties such as payment pattern, lags, severity, case reserve activities, and recovery adjustment properties.

### 2.5.1) Three Payment Patterns

This version of simulator contains three payment patterns defined in the first tab of each Type object, as sampled in Picture (20). **Loss Simulator contains another set of help system that, when you put your mouse over a property (most of them are property labels), you will see a popup tool tip help window for that property, explaining the**

**physical meaning of that property.** Each payment pattern option has its own completely different simulation algorithm. Please refer to the help section online at: http://www.casact.org/research/lsmwp/losshelp/index.cfm?fa=main for a detailed discussion of each payment pattern option.



**Property Description**

The single-payment model approximates the development pattern of coverages such as General Liability, where each occurrence is followed by one reporting, one or more valuations, zero or one payment, and zero or one recovery or other adjustment to the amount originally paid.

● Single Payment (e.g. GL)

○ Random Multiple Payments (e.g. WC Medical)

○ Periodic Payments (e.g. WC Indemnity)

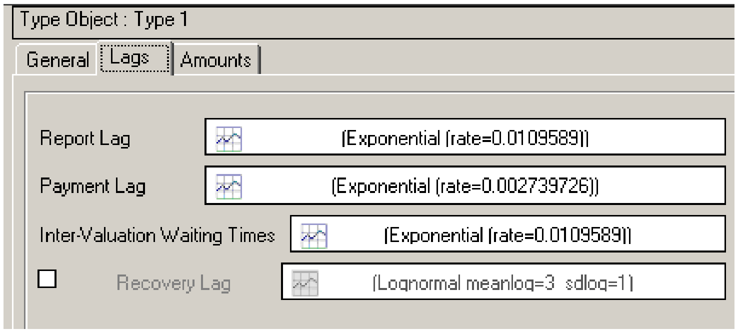
**Picture (20). Three payment patterns defined for each Type object**

Once you click one of the payment pattern radio buttons, you will notice that the remaining two tabs of the Type object will also be changed associated with the selected payment pattern. We will explain them one by one, associating each property with each simulation algorithm.

### 2.5.2) Single Payment Pattern

A Type object of Single Payment Pattern will have the following two screens, as seen in Picture (21) and Picture (22), to define lags, severity properties, and recoveries.

### 2.5.2.1) Single Payment Pattern Lags



Report Lag: (Exponential (rate=0.01095891)

Payment Lag: (Exponential (rate=0.002739726))

Inter-Valuation Waiting Times: (Exponential (rate=0.0109589))

☐ Recovery Lag: (Lognormal meanlog=3 sdlog=1)

**Picture (21). Lags defined for single Payment Type object**

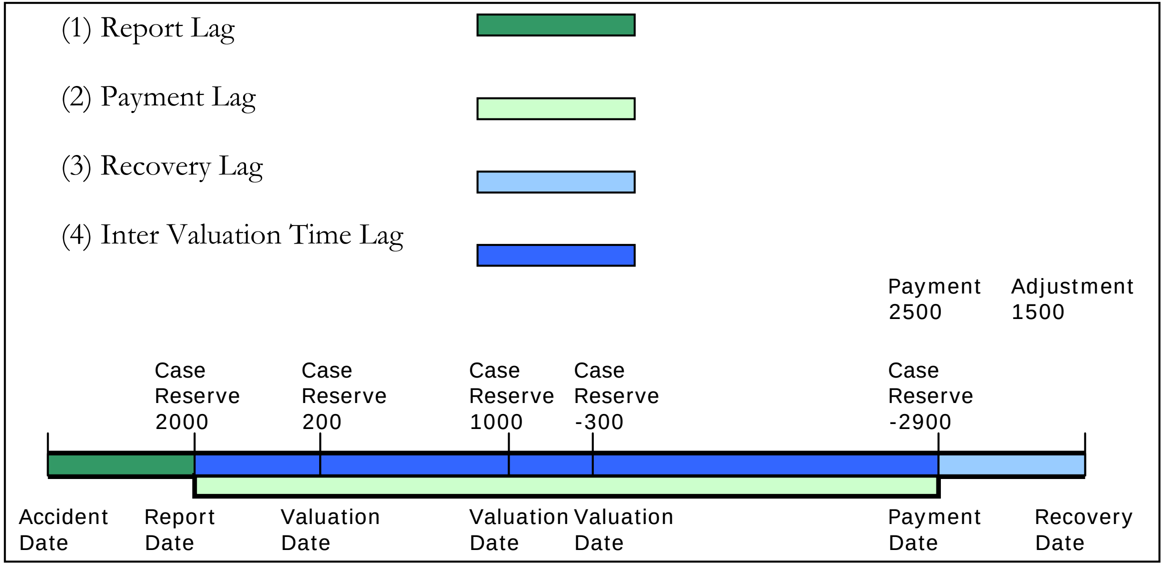
**Report Lag,** a monthly defined continuous distribution, helps to generate Report Date of a claim from the accident date.

**Payment Lag,** a monthly defined continuous distribution, helps to generate Payment Date of the claim from the Report Date.

**Inter-Valuation Waiting Times**, a monthly defined continuous distribution, will define each Valuation Date (Case Reserve Date) between the Report Date and the final Payment Date.

**Recovery Lag**, a monthly defined continuous distribution, will define the Recovery Date after the Payment Date if the checkbox is selected.

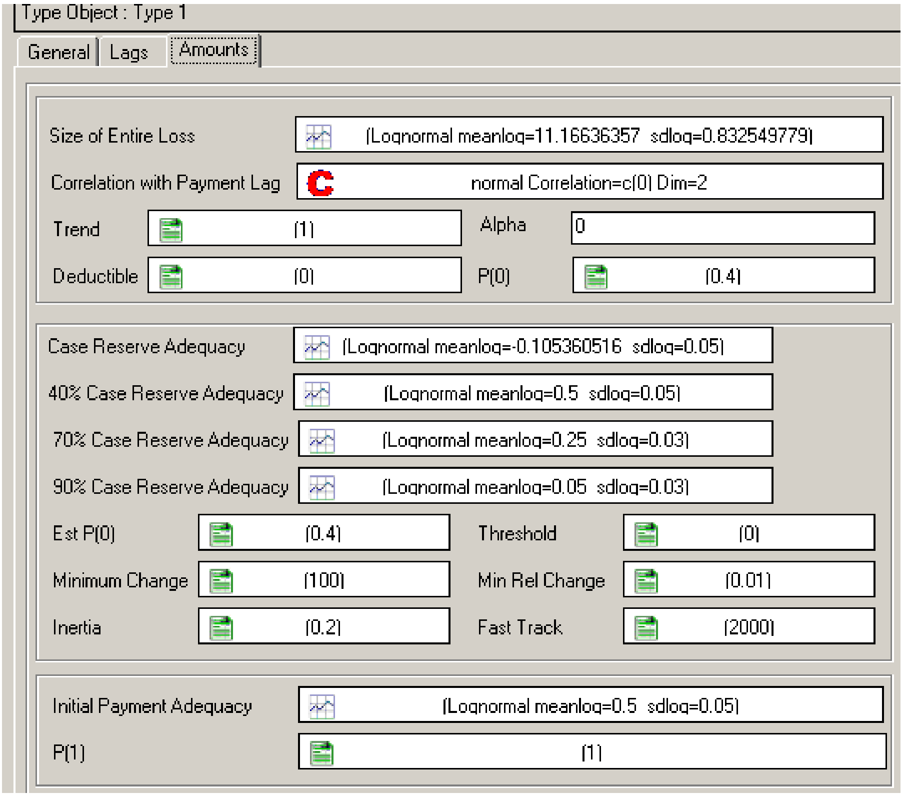
The following picture illustrates the above four properties.



**Picture (22). Illustration of Single Payment Pattern option in simulation**

### 2.5.2.2) Single Payment Pattern Severity Properties

Picture (23) shows the properties to define severity, case reserve, and recovery (if selected). We will explain each property in detail.



Size of Entire Loss: [Lognormal meanlog=11.16636357 sdlog=0.832549779]
Correlation with Payment Lag: normal Correlation=c(0) Dim=2
Trend: [1]
Alpha: 0
Deductible: [0]
P(0): [0.4]

Case Reserve Adequacy: [Lognormal meanlog=-0.105360516 sdlog=0.05]
40% Case Reserve Adequacy: [Lognormal meanlog=0.5 sdlog=0.05]
70% Case Reserve Adequacy: [Lognormal meanlog=0.25 sdlog=0.03]
90% Case Reserve Adequacy: [Lognormal meanlog=0.05 sdlog=0.03]
Est P(0): [0.4]
Threshold: [0]
Minimum Change: [100]
Min Rel Change: [0.01]
Inertia: [0.2]
Fast Track: [2000]

Initial Payment Adequacy: [Lognormal meanlog=0.5 sdlog=0.05]
P(1): [1]

**Picture (23). Severity and recovery (if enabled) for single Payment Type object**

**Size of Entire Loss** is a monthly defined continuous distribution that defines the severity distribution of the Type. It will determine the final claim payment. This variable can be correlated using a Copula object with the payment lag defined in the Type. However, NO correlation is selected in Picture (23).

**P(0),** a monthly defined value, defines the constant probability of closure without payment, for reasons other than failure to exceed the deductible.

The cumulative trend factors (cum) are calculated to the accident and payment dates, and then the trend multiplier is calculated as follows:

$$\text{trend} = \left(cum_{acc\_date}\right)\left(\frac{cum_{pmt\_date}}{cum_{acc\_date}}\right)^{\alpha} = \left(cum_{acc\_date}\right)^{1-\alpha}\left(cum_{pmt-date}\right)^{\alpha}$$

**We are applying the full trend factor from the average accident date underlying the assumed loss distribution to the accident date of the occurrence, and a portion "alpha" of the trend from the accident date to the payment date. This is an application of Butsic's alpha parameter concept from his May 1981 CAS *Discussion Paper Program* titled, "The Effect of Inflation on Losses and Premiums for Property-Liability Insurers."**

Then the final payment of the claim is calculated by multiplying the trend factor and the simulated claim from the Size of Loss property and then applying the policy limit and deductible.

### Case Reserve Adequacy

Case Reserve Adequacy at valuation time *t* is a lognormal random variable with

$\mu$ = (meanlog at time $t$) and s = (sdlog at time $t$). Here $t$ is the fraction:

$$t = \frac{\text{valuation time minus report date}}{\text{payment date minus report date}} = \frac{\text{Valuation lag}}{\text{Payment lag}}$$

The user enters the *meanlog* for times $t$ = 0%, 40%, 70%, and 90% (the time 0 value is labeled "case reserve adequacy" in the model). The *meanlog* for time 1.0 is set at 0.0. For other values of *t*, the Simulator applies linear interpolation to calculate values of *meanlog*. The modeler also input *sdlog*s for times 0%, 40%, 70%, and 90%, which provides flexibility in controlling the variance of the reserve adequacy factor.

On the Simulator screen, the user determines the case reserve adequacy parameters by selecting four lognormal distributions labeled as Case Reserve Adequacy, 40% Case Reserve Adequacy, 70% Case Reserve Adequacy, and 90% Case Reserve Adequacy, as displayed in Picture (23).

### Threshold and EstP(0)

The Simulator introduces a non-negative threshold value that is associated with EstP(0) and is used when case reserves are set at each valuation time. If the claim's ultimate size of loss value is strictly below this threshold, we would not apply the EstP(0) adjustment in setting the case reserve. If the user enters a zero value for threshold, then the model will apply the "EstP(0)," adjustment to all claims.

The algorithm for estimating the case reserve value of a claim is as follows:

```
(1) If threshold <= 0 OrElse payment > threshold Then
```

*value = (1 - EstP0 at ValuationDate)* payment * multiplier(apply policy limit and deductible)*

*(2) If (threshold > 0 AndAlso payment <= threshold) Then*
*value = payment * multiplier - deductible*

*(Multiplier is the value simulated from the interpolated case reserve adequacy lognormal distribution.)*

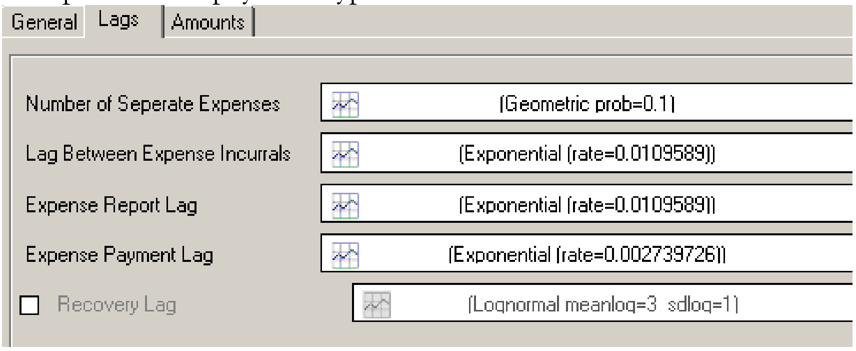
**Recovery.** Recovery is optional.
By checking the Recovery Lag Checkbox, as shown in Picture (21), you enable the recovery calculation. **P(1)** is the probability that the claim closes with the initial payment amount, and is constant for all claims. The default value is 1, which means the initial payment covers the full payment amount and there is no recovery (even if you selected the checkbox).

**InitialPaymentAdequacy** is a monthly defined distribution that is used when P(1) is not 1. It represents the ratio of the ultimate payment after recoveries to the initial payment, and so defines the recovery amount as *Recovery= Initial Payment * (1- ratio).*

### 2.5.3) Multiple Random Payment Pattern

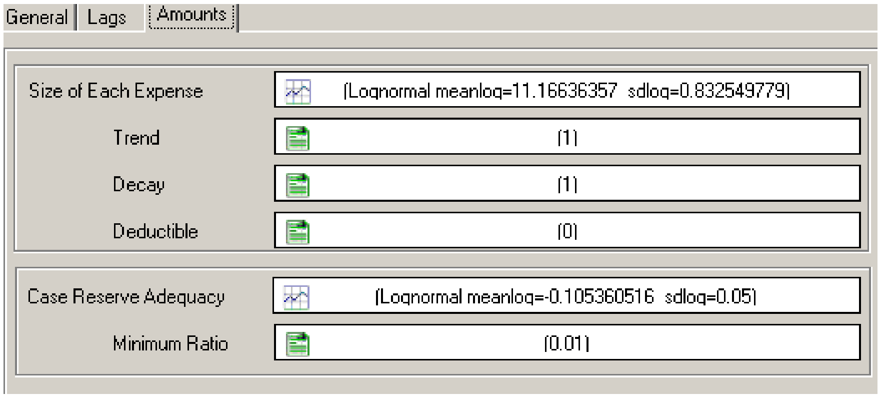
The multiple random payments model approximates the development pattern of coverage such as Medical Payments, where each occurrence is followed by a random number of reimbursable incurred expenses. Each expense is followed by one reporting and one payment. The final expense payment (which may or may not be the final expense incurred) is followed by zero or one recovery or other adjustment to the total of all previous payments.

The following pictures will show the screen of defining lags and severities properties of a multiple random payment type.



Number of Seperate Expenses (Geometric prob=0.1)
Lag Between Expense Incurrals (Exponential (rate=0.0109589))
Expense Report Lag (Exponential (rate=0.0109589))
Expense Payment Lag (Exponential (rate=0.002739726))
□ Recovery Lag (Lognormal meanlog=3 sdlog=1)

**Picture (24). Lags defined for Multiple Random Payment Type object**



Size of Each Expense (Lognormal meanloq=11.16636357 sdloq=0.832549779)

Trend (1)

Decay (1)

Deductible (0)

Case Reserve Adequacy (Lognormal meanloq=-0.105360516 sdloq=0.05)

Minimum Ratio (0.01)

**Picture (25). Size of Loss properties defined for Multiple Random Payment Type**

The multiple-random-payments model approximates the development pattern of coverages such as Medical Payments, where each occurrence is followed by a random number of reimbursable incurred expenses. Each expense is followed by one reporting and one payment. The final expense payment (which may or may not be the final expense incurred) is followed by zero or one recovery or other adjustment to the total of all previous payments.

The number of reimbursable expenses per claim is assumed to follow a geometric distribution or a multinomial distribution specified by the user.

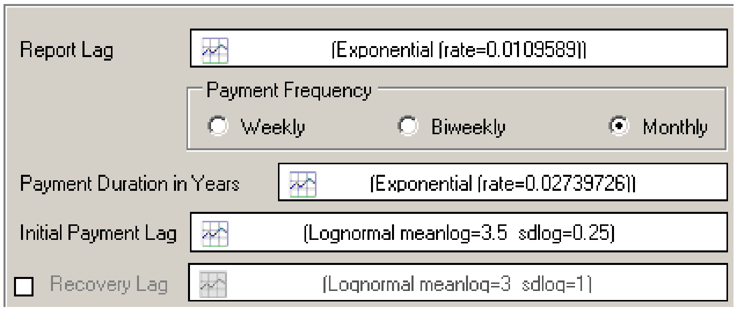
The severity parameters describe the distribution of severities for each individual expense, except that the deductible and maximum apply to all expenses in aggregate. There is a single trend factor (actually annual trend rate) applied to each expense through its incurral date, and there is decay factor allowing the user to specify a declining mean from one expense to the next expense arising from the same claim.

Case reserves are assumed to be revalued at each payment date. Their adequacy is measured relative to all expenses that have been or will be incurred but have not yet been paid, subject to a minimum that allows a reserve to be carried between the last payment and the recovery date. The “P(2 sig dig)” entry represents the probability that a case reserve will be estimated to a nearby “round” number—in this case rounding to two significant digits—rather than its exact value.

Recoveries are modeled as one-time adjustments to correct errors in the original amounts paid. For this purpose each amount paid is treated as an adequacy factor times the actual severity after application of the deductible and the maximum. Payment errors are reflected in the distribution of this factor less 1.00. In particular, if the adequacy factor is greater than 1.00, the initial payment will be too great and will produce a future recovery, represented as a negative payment. The simulator generates both the original overpayment and underpayment, spread uniformly across all payment dates, and the later recovery or adjustment.

### 2.5.4) Periodic Multiple Payment Pattern

The periodic payments model approximates the development patterns of coverage such as Group Long-Term Disability or the wage-replacement provisions of Workers Compensation, where each occurrence is followed by a random number of regular periodic payments of equal amounts or of equal amounts subject to periodic inflation adjustments. The final payment is followed by zero or one recovery of any payments that were inadvertently made beyond the termination of disability.



Report Lag (Exponential (rate=0.0109589))

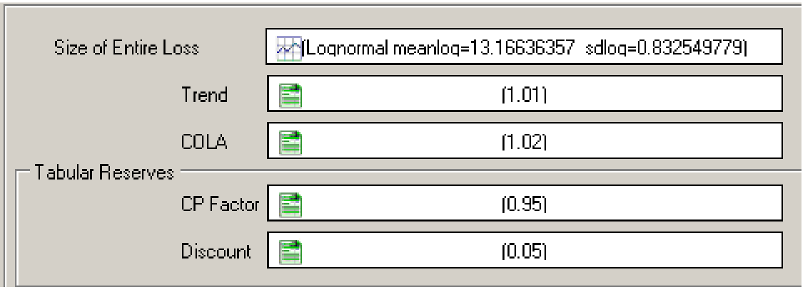
Payment Frequency: ○ Weekly ○ Biweekly ● Monthly

Payment Duration in Years (Exponential (rate=0.02739726))

Initial Payment Lag (Lognormal meanlog=3.5 sdlog=0.25)

□ Recovery Lag (Lognormal meanlog=3 sdlog=1)

**Picture (26). Lags defined for Multiple Periodic Payment Type**



Size of Entire Loss (Lognormal meanlog=13.16636357 sdlog=0.832549779)

Trend (1.01)

COLA (1.02)

Tabular Reserves

CP Factor (0.95)

Discount (0.05)

**Picture (27). Severity and case reserve defined for Multiple Periodic Payment Type**

**Size of Loss** is typically related to salaries and may be approximated by a suitable distribution with a minimum and maximum. There is a provision for trend which affects the payment size at time of occurrence, and for COLA factors which affects individual claims at annual intervals following commencement of payments.

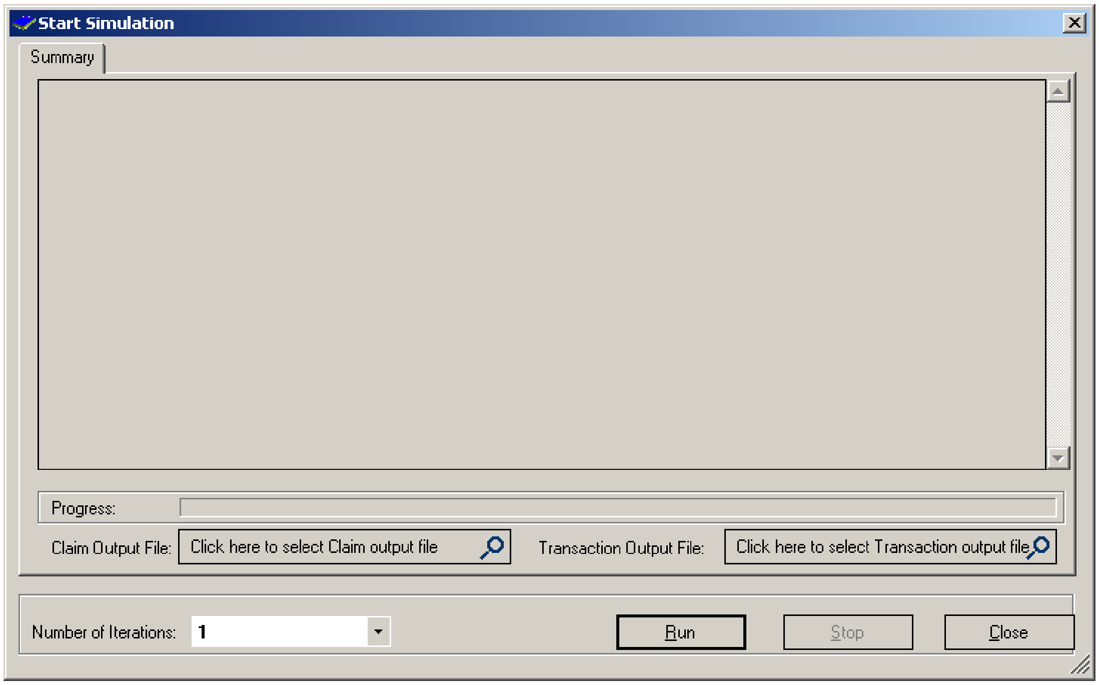
**Trend** is the change in mean payment size for newly incurred claims. It is expressed as an annual factor but applied monthly.

**COLA** is the change in payment size, at the end of the first and each subsequent year, for a given claim already in payment status.

**CP factor** is the ratio of the periodic continuance probabilities assumed in the tabular reserves to the same probabilities implicit in the payment duration distribution.

## 2.6 Run Simulation

You can only start the simulation with an open project; otherwise, the simulation menu and the toolbar buttons are all disabled. Once you feel comfortable enough for the model configuration, you can go to Tools->Run Simulation, or simply click the 📊 button from toolbar. The "Run Simulation" window pops up as shown in Picture (28)



Number of Iterations: 1

**Picture (28). Run simulation**

### 2.6.1) Provide Claim and Transaction Output File

The Simulator will need a claim and a transaction output file. So from the above screen, you can click the two buttons individually and provide the **CSV** file name from the Windows File Save Dialog. The Simulator will then generate two **CSV** files after the simulation is done.

Here is an example of how the claim CSV file looks like (if opened with Excel):

```
Simulation 6/9/2010 11:51:33 AM
Simulation  Occurrence  Claim  Accident   Report
No          No          No     Date       Date      Line    Type
         1           1      1  20000117   20001103     1       1
         1           2      1  20000112   20000418     1       1
         1           3      1  20000108   20000320     1       1
         1           4      1  20000129   20000201     1       1
```

And here is a sample transaction file:

```
Transactions 6/9/2010 11:51:33 AM
Simulation  Occurrence  Claim
No          No          No     Date      Transaction  Case Reserve   Payment
         1           1      1  20001103  REP                  2000         0
         1           1      1  20010313  RES                 15439         0
         1           1      1  20010425  RES                  1330         0
         1           1      1  20010915  RES                  2938         0
         1           1      1  20020429  RES                 -3870         0
         1           1      1  20020501  RES                  -650         0
         1           1      1  20020524  RES                  -484         0
         1           1      1  20020602  RES                  -198         0
         1           1      1  20020706  RES                  -632         0
         1           1      1  20020821  RES                  -920         0
         1           1      1  20020906  RES                  -542         0
         1           1      1  20021004  CLS                -14411      1233
         1           2      1  20000418  REP                  2000         0
         1           2      1  20000918  RES                 13113         0
         1           2      1  20010405  RES                  3830         0
         1           2      1  20010913  RES                 -1334         0
         1           2      1  20020101  RES                 -2803         0
         1           2      1  20020205  RES                  -950         0
         1           2      1  20020319  CLS                -13856         0
         1           3      1  20000320  REP                  2000         0
         1           3      1  20000408  RES                 29554         0
         1           3      1  20000511  RES                  5778         0
         1           3      1  20000624  RES                  7127         0
         1           3      1  20001004  RES                  -984         0
         1           3      1  20010108  CLS                -43475     10866
```

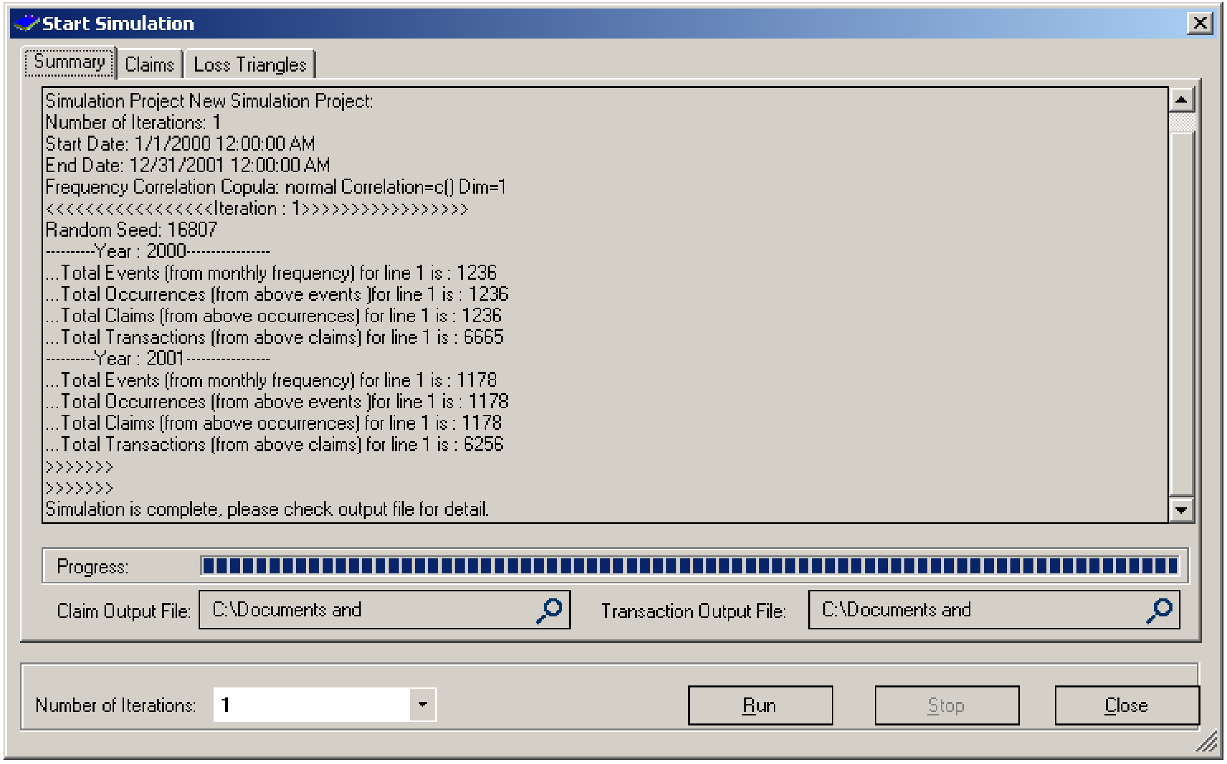
### 2.6.2) Number of Iterations

This controls how many iterations the Simulator will run. To get a feel for the simulator, you can simply choose 1 from the dropdown selection, and the simulation will finish after 1 iteration.

For real simulation, you can choose up to 1000 iterations. This could require many hours of CPU time, depending upon the complexity of your parameterization of the model and your computer speed. The final result may contain hundreds of thousands of claims and millions of transactions. In this case, you can let the program run overnight. However, you can cancel the process at any time by clicking the “Stop” button.

### 2.6.3) On Screen Simulation Output

The Run Simulation window will show the calculation progress and, at regular intervals, displays the simulation summary text. For a large simulation, the summary text will be refreshed after each 100 iterations of the simulation process, so it may not give you the full picture in this case.



```
Simulation Project New Simulation Project:
Number of Iterations: 1
Start Date: 1/1/2000 12:00:00 AM
End Date: 12/31/2001 12:00:00 AM
Frequency Correlation Copula: normal Correlation=c() Dim=1
<<<<<<<<<<<<<<<<<Iteration : 1>>>>>>>>>>>>>>>>>
Random Seed: 16807
---------Year : 2000----------------
...Total Events (from monthly frequency) for line 1 is : 1236
...Total Occurrences (from above events )for line 1 is : 1236
...Total Claims (from above occurrences) for line 1 is : 1236
...Total Transactions (from above claims) for line 1 is : 6665
---------Year : 2001----------------
...Total Events (from monthly frequency) for line 1 is : 1178
...Total Occurrences (from above events )for line 1 is : 1178
...Total Claims (from above occurrences) for line 1 is : 1178
...Total Transactions (from above claims) for line 1 is : 6256
>>>>>>>
>>>>>>>
Simulation is complete, please check output file for detail.
```

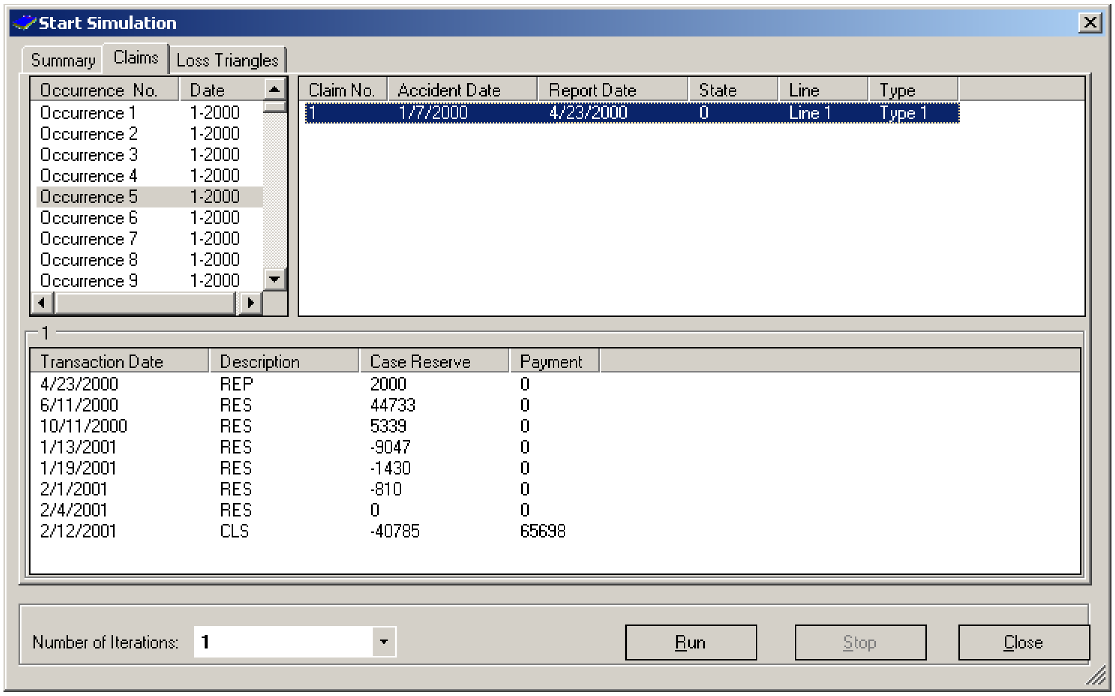
Claim Output File: C:\Documents and

Transaction Output File: C:\Documents and

Number of Iterations: 1

**Picture (29). Simulation progress screen with summary text**

**If you select "The Number of Iteration" to be 1,** at the end, **the Simulator** will automatically present results similar to Picture (30). There you can scroll down and browse any occurrence, then browse the claims within that occurrence, and finally the transactions inside that claim.



| Occurrence No. | Date |
|---|---|
| Occurrence 1 | 1-2000 |
| Occurrence 2 | 1-2000 |
| Occurrence 3 | 1-2000 |
| Occurrence 4 | 1-2000 |
| Occurrence 5 | 1-2000 |
| Occurrence 6 | 1-2000 |
| Occurrence 7 | 1-2000 |
| Occurrence 8 | 1-2000 |
| Occurrence 9 | 1-2000 |

| Claim No. | Accident Date | Report Date | State | Line | Type |
|---|---|---|---|---|---|
| 1 | 1/7/2000 | 4/23/2000 | 0 | Line 1 | Type 1 |

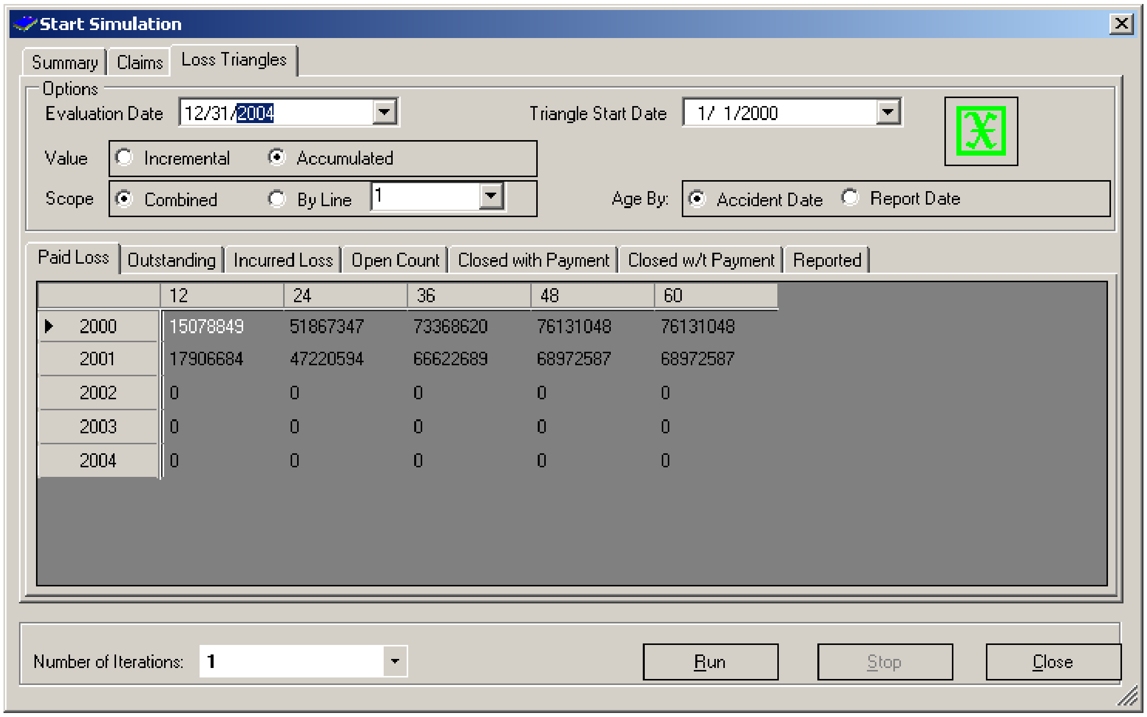
1

| Transaction Date | Description | Case Reserve | Payment |
|---|---|---|---|
| 4/23/2000 | REP | 2000 | 0 |
| 6/11/2000 | RES | 44733 | 0 |
| 10/11/2000 | RES | 5339 | 0 |
| 1/13/2001 | RES | -9047 | 0 |
| 1/19/2001 | RES | -1430 | 0 |
| 2/1/2001 | RES | -810 | 0 |
| 2/4/2001 | RES | 0 | 0 |
| 2/12/2001 | CLS | -40785 | 65698 |

Number of Iterations: 1

**Picture (30). Occurrence, Claim, and Transactions (if only 1iteration is executed)**

**If you run the Simulator with one iteration,** the program will also generate a variety of loss triangles as shown in Picture (31). The user may export the triangles to an Excel file. Please note that the "triangles" are presented in "rectangle" format, and that the user may also reconfigure the triangles by choosing different properties on this screen.



Options

Evaluation Date 12/31/2004

Triangle Start Date 1/ 1/2000

Value ○ Incremental ● Accumulated

Scope ● Combined ○ By Line 1

Age By: ● Accident Date ○ Report Date

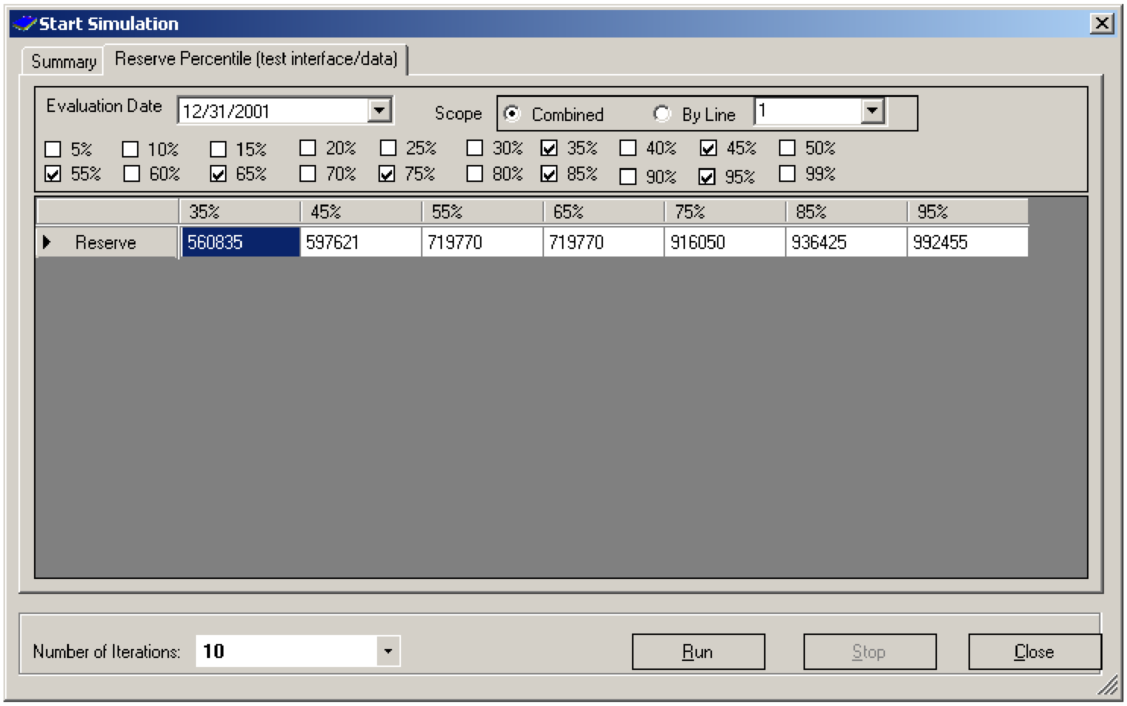
| | 12 | 24 | 36 | 48 | 60 |
|---|---|---|---|---|---|
| 2000 | 15078849 | 51867347 | 73368620 | 76131048 | 76131048 |
| 2001 | 17906684 | 47220594 | 66622689 | 68972587 | 68972587 |
| 2002 | 0 | 0 | 0 | 0 | 0 |
| 2003 | 0 | 0 | 0 | 0 | 0 |
| 2004 | 0 | 0 | 0 | 0 | 0 |

Number of Iterations: 1

**Picture (31). Loss Triangles from the output (if only 1iteration is executed)**

### 2.6.4) Reserve Percentile Output

**If the simulation has more than 100 iterations,** the Simulator will generate a Reserve Percentile table from the simulation results.



Start Simulation

Summary | Reserve Percentile (test interface/data)

Evaluation Date 12/31/2001

Scope ● Combined ○ By Line 1

□ 5% □ 10% □ 15% □ 20% □ 25% □ 30% ☑ 35% □ 40% ☑ 45% □ 50%
☑ 55% □ 60% ☑ 65% □ 70% ☑ 75% □ 80% ☑ 85% □ 90% ☑ 95% □ 99%

| | 35% | 45% | 55% | 65% | 75% | 85% | 95% |
|---|---|---|---|---|---|---|---|
| Reserve | 560835 | 597621 | 719770 | 719770 | 916050 | 936425 | 992455 |

Number of Iterations: 10

**Picture (32). Reserve Percentiles from the output (if 100+ iterations are executed)**

## 3) Simulation Example

The CAS Loss Simulator Working Party Testing Group, led by Professor Joe Marker, has conducted several testing scenarios to evaluate the simulation parameterization and simulation results. One scenario tests the severity distributions (i.e., size of loss). We select this test in order to illustrate how to set up the parameters to simulate a company's business.

The simulated results contain the following:

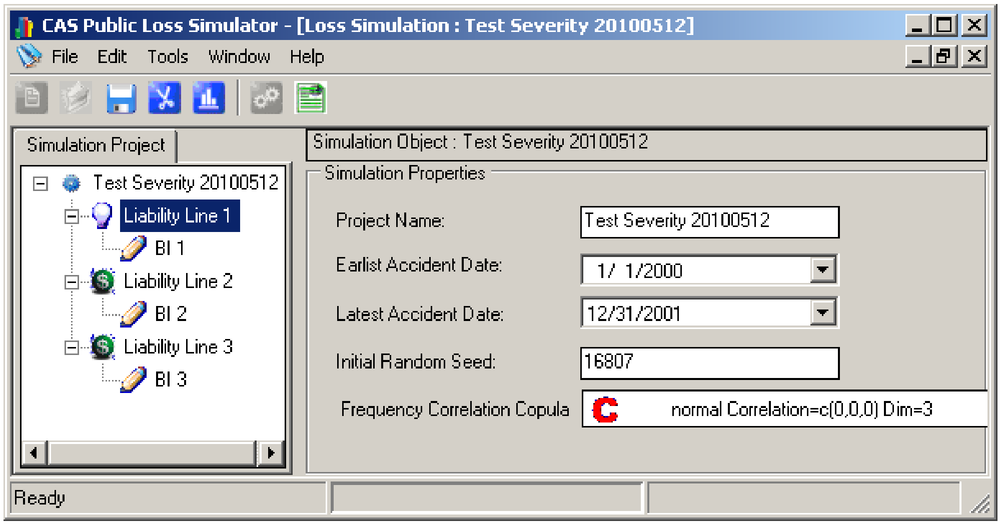
| File Name | File Description |
|---|---|
| c.csv | claims output by the simulator |
| t.csv | transactions output by the simulator |
| ultloss 20100512.csv | the file containing ultimate loss for each claim |
| Test severities 20100512.xml | XML file containing the parameters used to run the model (This file can be imported into your Simulator directly to see the following parameter definitions, and will produce the same results as displayed in the above c.csv and t.csv files.) |

These files can be downloaded from the following link:
**http://www.goouon.com/loss_simulator/file/simulation20100520**

### 3.1 Simulation Project Level Parameters Setup

Set up three lines of business with no correlation in frequency among the three lines. If you import the "Test severities 20100512.xml" file, you will see the exact same screens illustrated below, starting from Picture (33).

| | |
|---|---|
| Project Name | Test Severity 20100512 |
| Accident Years | 2000-2001 |
| Initial Random Seed | 16807 |
| Frequency correlation copula | (None) normal Correlation=c( ) Dim = 1 |
| Line of Business (3) | Liability Line 1<br>Liability Line 2<br>Liability Line 3 |
| Coverage | One BI Coverage (for each line), single payment model |



Simulation Object : Test Severity 20100512

Simulation Properties

Project Name: Test Severity 20100512

Earlist Accident Date: 1/ 1/2000

Latest Accident Date: 12/31/2001

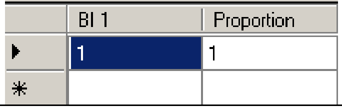
Initial Random Seed: 16807

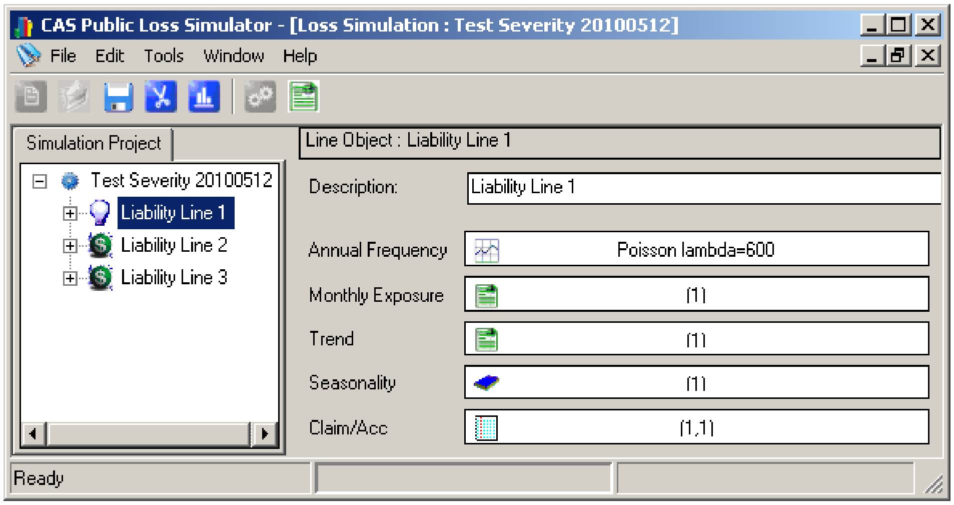
Frequency Correlation Copula: normal Correlation=c(0,0,0) Dim=3

**Picture (33). Severity Test with Project Level Parameters Setup**

### 3.2 Line Level Parameters Setup

For each line, an occurrence generates one claim of one unique type for each line.
Zero trend for frequency; zero trend for severity. Most parameters are the same for all lines. We use Line 1 as representative of all three lines in the table below (please also refer to Picture (34)):

| | |
|---|---|
| Annual Frequency | Poisson(**600**) |
| Monthly Exposure | (1)—None |
| Trend | (1)—None |
| Seasonality | (1)—None |
| Claim/acc matrix | **each occurrence generates 1 claim of one type**<br><br>BI 1 \| Proportion<br>1 \| 1 |
| Expected # claims | = 600 (freq) x 100 (# sims) x 3 (lines) x 2 (years) = 360,000. |
| Actual # claims from the simulation result | 359,819 |



Line Object : Liability Line 1

Description: Liability Line 1

Annual Frequency: Poisson lambda=600

Monthly Exposure: (1)

Trend: (1)

Seasonality: (1)

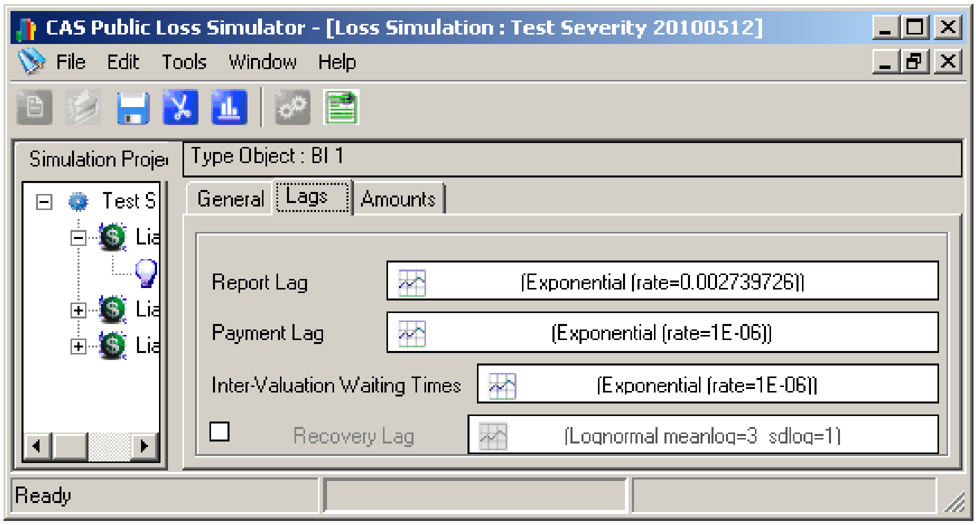
Claim/Acc: (1,1)

**Picture (34). Line Level Parameters**

### 3.3 Coverage Level Parameters Setup

Lags are irrelevant for this run except for report lag. We are not testing the reserve change process.

| | |
|---|---|
| Report lag | Exponential with Rate = 1/365, mean=365 days.<br>Max=3650. |
| Payment lag | Maximum one day (irrelevant) |
| Inter-valuation lag | Maximum one day (irrelevant) |
| Recovery lag box | Not checked (irrelevant) |
| | |
| Correlation of Amount with lag | (None): normal Correlation=c( ) Dim=2 |
| Alpha | 0 |
| Inertia | 0 |
| P(0) | 0 |
| EstP(0) | 0 |
| | |
| Case Reserve adequacy | Irrelevant, leave at default values. |
| Case Reserve Interpolation | Irrelevant, leave with default |
| Minimum change | 0 |
| Min Rel Chg | 0 |
| | |
| P(1) | Not set because recovery lag was not checked. |
| Initial payment adequacy | Not set because recovery lag was not checked. |



Type Object : BI 1

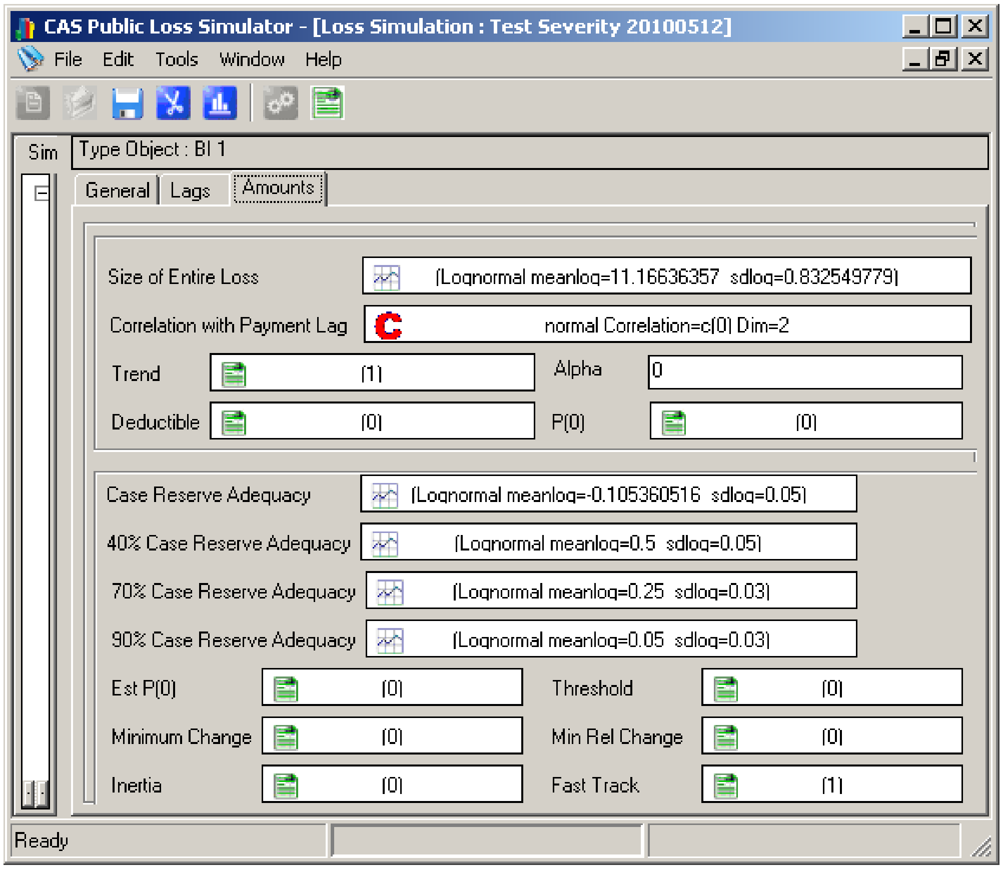
Report Lag (Exponential (rate=0.002739726))

Payment Lag (Exponential (rate=1E-06))

Inter-Valuation Waiting Times (Exponential (rate=1E-06))

□ Recovery Lag (Lognormal meanlog=3 sdlog=1)

**Picture (35). Coverage Level Lag Parameters**


CAS Public Loss Simulator - [Loss Simulation : Test Severity 20100512]
File
Edit
Tools
Window
Help
Sim
General
Lags
Amounts
Ready

Type Object : BI 1

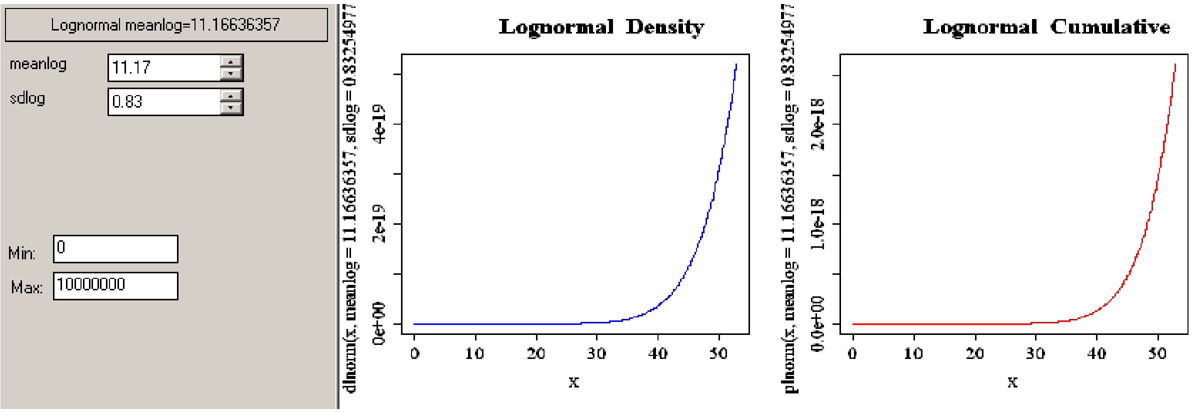
| | | | |
|---|---|---|---|
| Size of Entire Loss | (Lognormal meanlog=11.16636357 sdlog=0.832549779) | | |
| Correlation with Payment Lag | normal Correlation=c(0) Dim=2 | | |
| Trend | (1) | Alpha | 0 |
| Deductible | (0) | P(0) | (0) |
| Case Reserve Adequacy | (Lognormal meanlog=-0.105360516 sdlog=0.05) | | |
| 40% Case Reserve Adequacy | (Lognormal meanlog=0.5 sdlog=0.05) | | |
| 70% Case Reserve Adequacy | (Lognormal meanlog=0.25 sdlog=0.03) | | |
| 90% Case Reserve Adequacy | (Lognormal meanlog=0.05 sdlog=0.03) | | |
| Est P(0) | (0) | Threshold | (0) |
| Minimum Change | (0) | Min Rel Change | (0) |
| Inertia | (0) | Fast Track | (1) |

**Picture (36). Coverage Level Amounts Parameters**

## 3.4 Coverage Level Severity Distributions

Size of Loss (severity) Parameters varies by Line.

| | | |
|---|---|---|
| BI 1 for Liability Line 1<br>Picture (37) | Lognormal<br>mean=100,000<br>standard. dev.=100,000<br>max =10,000,000 | This means that the input lognormal parameters are<br>$\mu$ = meanlog =11.16636357,<br>$\sigma$ = sdmean = 0.832549779. |
| BI 2 for Liability Line 2<br>Picture (38) | Pareto<br>$\alpha$ (shape) =6<br>$\theta$ (scale) = 500,000<br>max =10,000,000 | This results in a mean of 100,000 and stnd deviation of 122,474.5. |
| BI 3 for Liability Line 3<br>Picture (39) | Weibull<br>$\theta$ (scale) = 95,000<br>$\tau$ (shape) = 0.9<br>max =10,000,000 | This results in mean of 99,957 and stnd.dev. of 111,256. |



Lognormal meanlog=11.16636357

meanlog 11.17

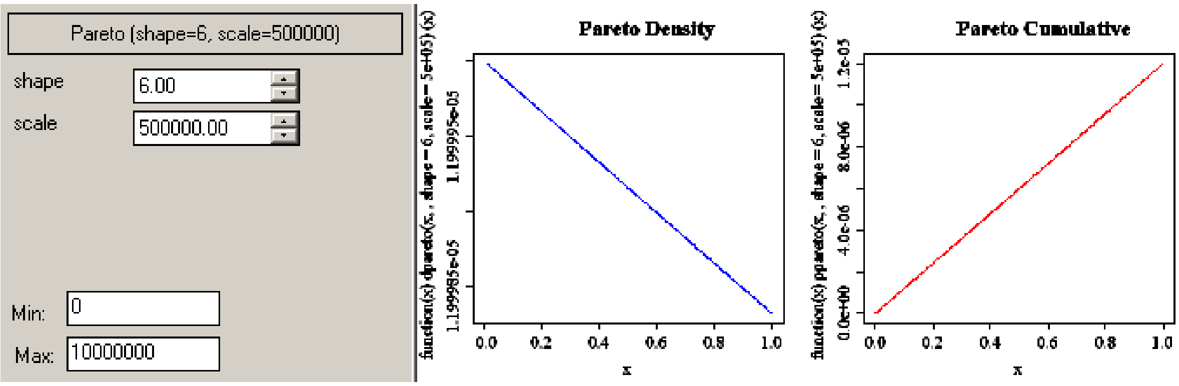
sdlog 0.83

Min: 0

Max: 10000000

Lognormal Density
dlnorm(x, meanlog = 11.16636357, sdlog = 0.83254977
0e+00
2e-19
4e-19
0
10
20
30
40
50
x
Lognormal Cumulative
plnorm(x, meanlog = 11.16636357, sdlog = 0.83254977
0.0e+00
1.0e-18
2.0e-18
0
10
20
30
40
50
x

**Picture (37). BI 1 for Line 1, Lognormal, mean=100,000 standard. dev.=100,000, max =10,000,000**



Pareto (shape=6, scale=500000)

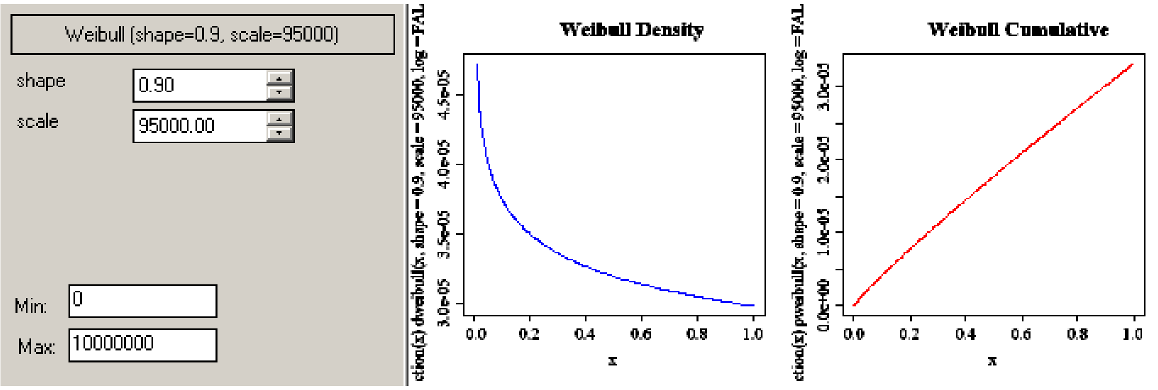
shape 6.00

scale 500000.00

Min: 0

Max: 10000000

**Picture (38). BI 2 for Line 2, Pareto , α (shape) =6, θ (scale) = 500,000, max =10,000,000**



Weibull (shape=0.9, scale=95000)

shape 0.90

scale 95000.00

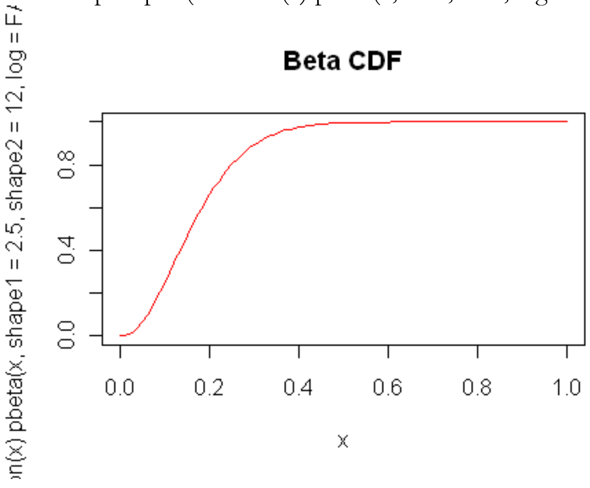
Min: 0

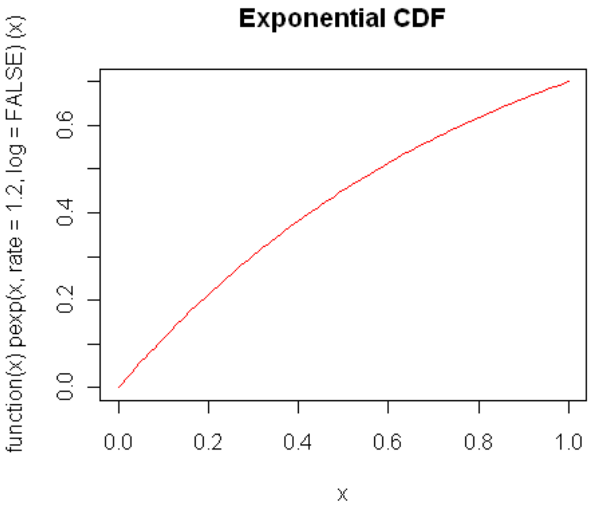
Max: 10000000

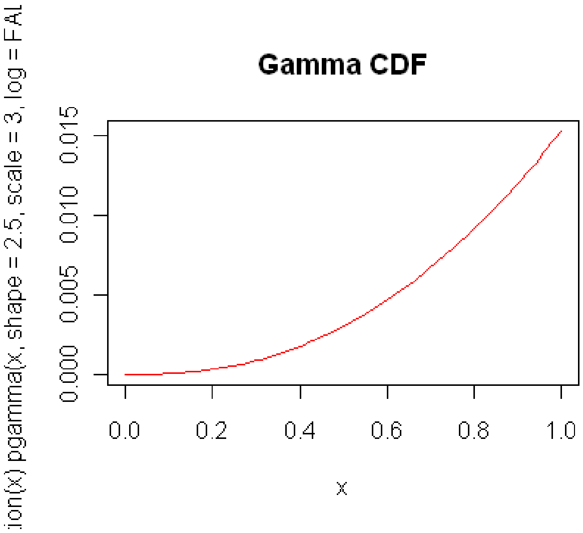
**Picture (39). BI 3 for Line 3, Weibull, θ (scale) = 95,000, $\tau$ (shape) = 0.9, max =10,000,000**

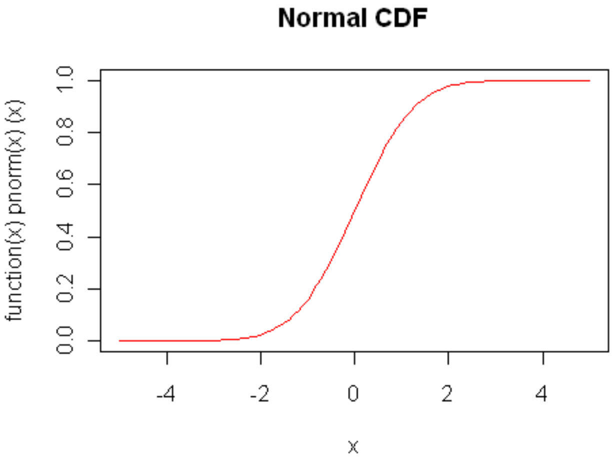
### 3.5 Number of Iterations: 100.

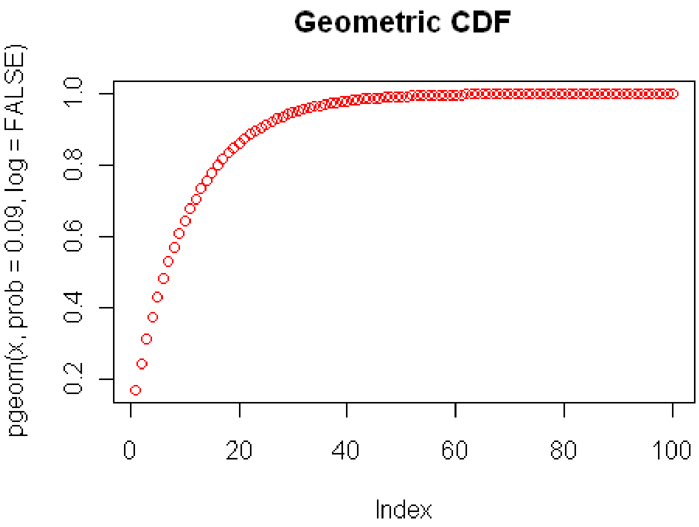
## 4) Distributions Used in the Public Loss Simulator and Their Parameterizations

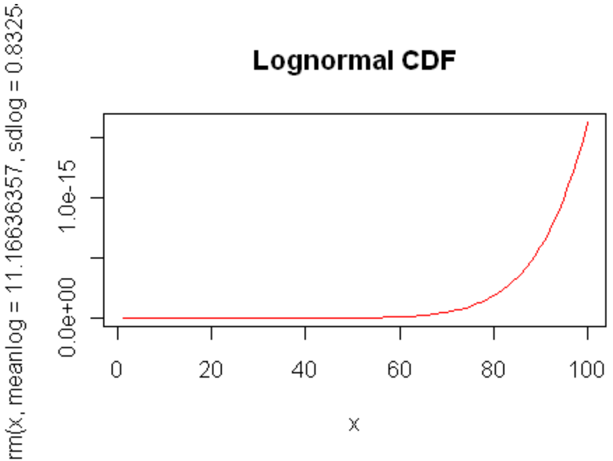
| Name | R Representation and Explanation | |
|---|---|---|
| **Beta** | Density | dbeta(x, shape1, shape2, ncp = 0, log = FALSE) |
| | Probability Function | pbeta(q, shape1, shape2, ncp = 0, lower.tail = TRUE, log.p = FALSE) |
| | Quantile Function | qbeta(p, shape1, shape2, ncp = 0, lower.tail = TRUE, log.p = FALSE) |
| | Random Generation | rbeta(n, shape1, shape2, ncp = 0) |
| | Details | Before defining the Beta distribution, it is convenient to define the Beta function by: $B(a, b) = \int_0^1 t^{a-1}(1-t)^{b-1}dt$ for $a>0$, $b>0$. It turns out that $B(a, b) = \frac{\Gamma(a)\ \Gamma(b)}{\Gamma(a+b)}$.<br>Then $f(x) = \frac{1}{B(a,b)}x^{a-1}(1-x)^{b-1}$ for $a > 0$, $b > 0$ and $0 \le x \le 1$ , is the density for the Beta distribution with shape1 = *a* and shape2 = *b*. The boundary values at *x*=0 and *x*=1 are defined by continuity (as limits).<br>The mean is $\frac{a}{a+b}$ and the variance is $\frac{a\,b}{(a+b)^2\,(a+b+1)}$.<br>Pbeta is the cumulative distribution function *F*(*x*) and is closely related to the incomplete beta function, defined by $B(x; a, b) = \int_0^x t^{a-1}(1-t)^{b-1}dt$ .<br>pbeta(*x*, *a*, *b*) = *F*(*x*) = $\frac{B(x;a,b)}{B(a,b)}$ |
| | CDF Sample: plot(function(x) pbeta(x, 1.25,12.0 , log=FALSE), main = "Beta CDF," col="red")<br><br> | |

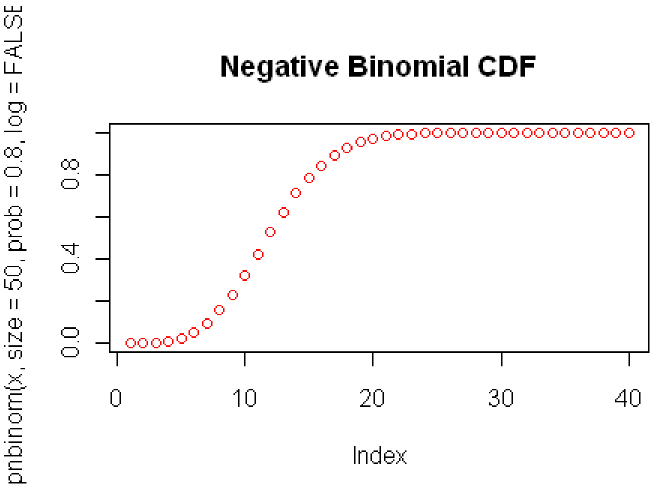
| Name | R Representation and Explanation | |
|---|---|---|
| **Exponential** | Density | dexp(x, rate = 1, log = FALSE) |
| | Probability Function | pexp(q, rate = 1, lower.tail = TRUE, log.p = FALSE) |
| | Quantile Function | qexp(p, rate = 1, lower.tail = TRUE, log.p = FALSE) |
| | Random Generation | rexp(n, rate = 1) |
| | Details | If rate is not specified, it assumes the default value of 1.<br>The exponential distribution with rate $\lambda$ has density<br>$f(x) = \lambda e^{-\lambda x}$ for $x \geq 0.$ |
| | CDF Sample: plot(function(x) pexp(x, rate=1.2, log=FALSE), main = "Exponential CDF," col="red")<br><br> | |

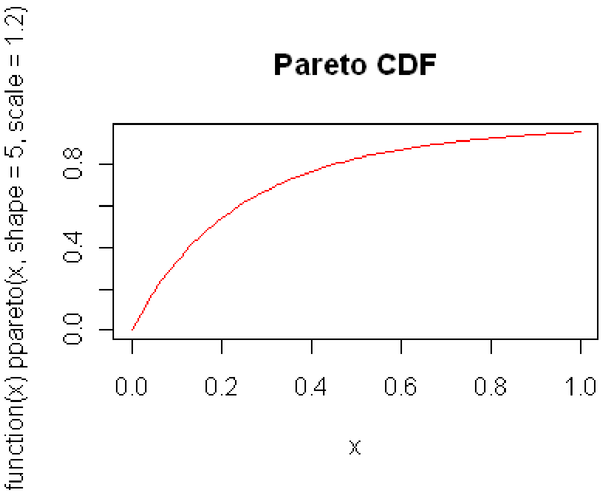
| Name | R Representation and Explanation | |
|---|---|---|
| **Gamma** | Density | dgamma(x, shape, rate = 1, scale = 1/rate, log = FALSE) |
| | Probability Function | pgamma(q, shape, rate = 1, scale = 1/rate, lower.tail = TRUE,log.p = FALSE) |
| | Quantile Function | qgamma(p, shape, rate = 1, scale = 1/rate, lower.tail = TRUE,log.p = FALSE) |
| | Random Generation | rgamma(n, shape, rate = 1, scale = 1/rate) |
| | Details | The Gamma distribution with parameters shape = *a* and scale = *s* has density<br>$f(x)= \frac{x^{a-1}\exp(-x/s)}{s^{a}\,\Gamma(a)}$, for $x \geq 0$, $a > 0$ and $s > 0$.<br>(Here $\Gamma(a)$ is the function implemented by **R**'s gamma( ) and defined in its help. Note that $a=0$ corresponds to the trivial distribution with all mass at point 0.)<br>The mean and variance are $E(X) = a*s$ and $Var(X) = a*s^2$.<br>The cumulative hazard $H(t) = - log(1 - F(t))$ is -pgamma(*t*, ..., lower = FALSE, log = TRUE).<br>Note that for small values of shape (and moderate scale) a large part of the mass of the Gamma distribution is on values of *x* so near zero that they will be represented as zero in computer arithmetic. So rgamma can well return values which will be represented as zero. (This will also happen for very large values of scale since the actual generation is done for scale=1.) |
| | CDF Sample: plot(function(x) pgamma(x, shape=2.5, scale==3, log=FALSE), main = "Gamma CDF," col="red")<br><br> | |

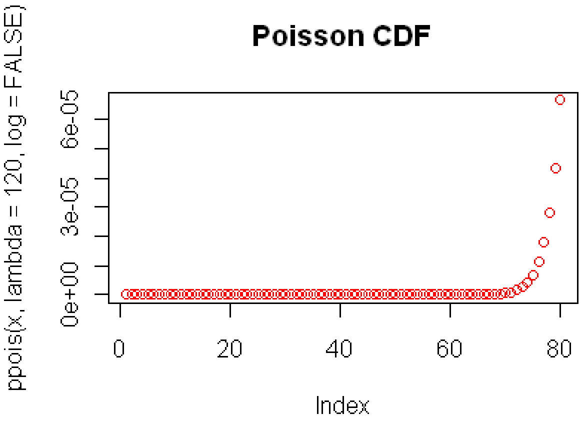
| | R Representation and Explanation | |
|---|---|---|
| **Gaussian (Normal)** | Density | dnorm(x, mean = 0, sd = 1, log = FALSE) |
| | Probability Function | pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE) |
| | Quantile Function | qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE) |
| | Random Generation | rnorm(n, mean = 0, sd = 1) |
| | Details | The normal distribution has density<br>$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-z^2 / 2)$, where $z = \frac{x - \mu}{\sigma}$<br>where $\mu$ is the mean of the distribution and $\sigma$ the standard deviation. |
| | CDF Sample: plot(function(x) pnorm(x), -5, 5,main = “Normal CDF,” col=“red”)<br> | |

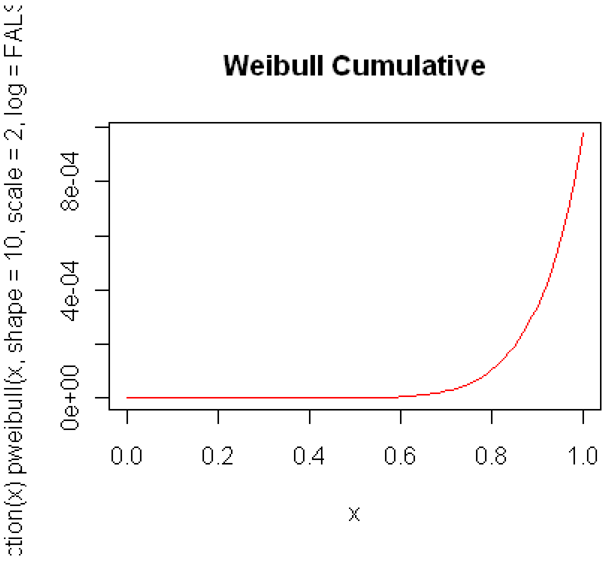
| Name | R Representation and Explanation | |
|---|---|---|
| **Geometric** | Density | dgeom(x, prob, log = FALSE) |
| | Probability Function | pgeom(q, prob, lower.tail = TRUE, log.p = FALSE) |
| | Quantile Function | qgeom(p, prob, lower.tail = TRUE, log.p = FALSE) |
| | Random Generation | rgeom(n, prob) |
| | Details | The geometric distribution with prob = *p* has density<br>$$p(x) = p(1-p)^x$$<br>for $x = 0, 1, 2, \ldots, 0 < p \leq 1$.<br>If an element of *x* is not integer, the result of pgeom is zero, with a warning. |
| | CDF Sample: x<-1:100<br>plot(pgeom(x, prob=0.09, log=FALSE), main = "Geometric CDF," col="red")<br><br> | |

| Name | R Representation and Explanation | |
|---|---|---|
| **Lognormal** | Density | dlnorm(x, meanlog = 0, sdlog = 1, log = FALSE) |
| | Probability Function | plnorm(q, meanlog = 0, sdlog = 1, lower.tail = TRUE, log.p = FALSE) |
| | Quantile Function | qlnorm(p, meanlog = 0, sdlog = 1, lower.tail = TRUE, log.p = FALSE) |
| | Random Generation | rlnorm(n, meanlog = 0, sdlog = 1) |
| | Details | A lognormal random variable *X* is one for which *Y* = ln(*X*) is normally distributed. The parameters *μ* and *σ* are the mean and standard deviation of ln(*X*). We can think of *X* as exp(*Y*), where *Y* has the normal distribution. *X* has density<br>$f(x) = \frac{1}{x\sigma\sqrt{2\pi}} \exp(-z^2/2), \text{ where } z = \frac{\ln x - \mu}{\sigma}$<br>The mean is *EX* = *exp(μ + 1/2 σ^2)*, the median is *med(X)* = *exp(μ)*, and the variance *Var(X)* = *exp(2*μ + σ^2)*(exp(σ^2) - 1)* and hence the coefficient of variation is *sqrt(exp(σ^2) - 1)* which is approximately *σ* when that is small (e.g., *σ* < *1/2*).<br>For example, the user enters the meanlog (μ) and sdlog (s) parameters for all lognormal distributions in the model. Thus, if you want to define the size of loss to be lognormal with mean 100,000 and standard deviation 100,000, you need to do a conversion calculate first.<br>To achieve the given size of loss distribution above, you can set the parameter<br>μ = meanlog = 11.16636357 and<br>s = sdlog = 0.832549779<br>since this produces a size of loss *X* with E*X* $= \exp\left(\mu + \frac{1}{2}\sigma^2\right) \approx$ 100,000.<br>and second moment E$X^2$ $= \exp\left(2\mu + 2\sigma^2\right) \approx$ 2 x $10^{10}$<br>This results in Var(*X*) ˜ 2 x $10^{10}$ - $10^{10}$ and s(*X*) ˜ 100,000. |
| | CDF Sample plot(x, plnorm(x, meanlog = 11.16636357, sdlog=0.832549779), type=“l,” main = “Lognormal CDF,” col=“red”)<br><br> | |

| Name | R Representation and Explanation | |
|---|---|---|
| **NegativeBinomial** | Density | dnbinom(x, size, prob, mu, log = FALSE) |
| | Probability Function | pnbinom(q, size, prob, mu, lower.tail = TRUE, log.p = FALSE) |
| | Quantile Function | qnbinom(p, size, prob, mu, lower.tail = TRUE, log.p = FALSE) |
| | Random Generation | rnbinom(n, size, prob, mu) |
| | Details | The negative binomial distribution with size = *n* and prob = *p* has density<br>$$\binom{x+n-1}{n-1} p^n (1-p)^x = \frac{\Gamma(x+n)}{x!\Gamma(n)} p^n (1-p)^x$$<br>for $x = 0, 1, 2, \ldots$, where *n* is a real number, $n > 0$ and $0 < p \leq 1$.<br>If *n* is a positive integer, this represents the number of failures before the *n*th success in a series of independent Bernoulli trials with probability of success *p*. The mean is $\frac{n(1-p)}{p}$ and the variance is $\frac{n(1-p)}{p^2}$. A negative binomial distribution results from a Poisson distribution whose mean has a gamma prior distribution with scale (1 - *p*)/*p* and shape *n*. (This definition allows non-integer values of size.)<br>An alternative parameterization (often used in ecology) is by the *mean* mu, and size, the *dispersion parameter*, where prob = size/(size+mu). The variance is mu + mu^2/size in this parameterization or *n (1-p)/p^2* in the first one. |
| | CDF Sample: x<-(1:40)<br>plot(pnbinom(x, size=50, prob=0.8, log=FALSE), main = “Negative Binomial CDF,” col=“red”)<br><br> | |

| Name | R Representation and Explanation | |
|---|---|---|
| **Pareto** | Density | dpareto(x, shape, scale, log = FALSE) |
| | Probability Function | ppareto(q, shape, scale, lower.tail = TRUE, log.p = FALSE) |
| | Quantile Function | qpareto(p, shape, scale, lower.tail = TRUE, log.p = FALSE) |
| | Random Generation | rpareto(n, shape, scale) |
| | Details | The Pareto distribution with parameters shape = *a* and scale = *θ* has density:<br>$f(x) = \alpha\,\theta^{\alpha}\,(x+\theta)^{-(\alpha+1)}$<br>for $x > 0$, $a > 0$ and $\theta > 0$. |
| | CDF Sample: plot(function(x) ppareto(x, shape=5, scale=1.2), main = "Pareto CDF," col="red")<br><br> | |

| Name | R Representation and Explanation | |
|---|---|---|
| **Poisson** | Density | dpois(*x*, lambda, log = FALSE) |
| | Probability Function | ppois(*q*, lambda, lower.tail = TRUE, log.*p* = FALSE) |
| | Quantile Function | qpois(*p*, lambda, lower.tail = TRUE, log.*p* = FALSE) |
| | Random Generation | rpois(*n*, lambda) |
| | Details | The Poisson distribution with parameter λ has density<br>$$p(x) = e^{-\lambda} \frac{\lambda^{x}}{x!}$$<br>for $x = 0, 1, 2, \ldots$ . The mean and variance are $E(X) = Var(X) = \lambda$.<br>If an element of *x* is not integer, the result of dpois is zero, with a warning. *p(x)* is computed using Loader's algorithm.<br>The quantile is right continuous: qpois(*p*, lambda) is the smallest integer *x* such that $P(X \le x) \ge p$.<br>Setting lower.tail = FALSE allows to get much more precise results when the default, lower.tail = TRUE would return 1, see the example below. |
| | CDF Sample: x<-(1:80)<br>plot(ppois(*x*, lambda=120, log=FALSE), main = "Poisson CDF," col="red")<br><br> | |

| Name | R Representation and Explanation | |
|---|---|---|
| **Weibull** | Density | dweibull(*x*, shape, scale = 1, log = FALSE) |
| | Probability Function | pweibull(*q*, shape, scale = 1, lower.tail = TRUE, log.*p* = FALSE) |
| | Quantile Function | qweibull(*p*, shape, scale = 1, lower.tail = TRUE, log.*p* = FALSE) |
| | Random Generation | rweibull(*n*, shape, scale = 1) |
| | Details | The Weibull distribution with shape parameter $\tau$ and scale parameter $\tau$ has density $f(x)$ and c.d.f. $F(x)$ given by<br>$f(x) = \frac{\tau x^{\tau-1}}{\theta^{\tau}} \exp\left[-(x/\theta)^{\tau}\right], \quad F(x) = 1 - \exp\left[-(x/\theta)^{\tau}\right] \quad \text{for } x > 0.$<br>The mean is $E(X) = \theta\,\Gamma(1 + 1/\tau)$, and $Var(X) = \theta^{2} * [\Gamma(1 + 2/\tau) - \{\Gamma(1 + 1/\tau)\}^{2}]$ |
| | CDF Sample: plot(function(*x*) pweibull(*x*, shape=10, scale=2.0, log=FALSE), main = "Weibull Cumulative," col="red")<br><br> | |

## 5) Public Loss Simulator Customized to Your Own Needs

The CAS Public Loss Simulation Model is an open-source model, which means you can use the available source code, containing VB.NET and embedded R language, and make modifications to fit your own company's reserving needs. The source code contains sufficient coding documentation that would enable an experienced programmer or modeler to begin customizing the model.

### 5.1 Pre-requirements

- To compile the program, you need to have Microsoft Visual Basic IDE (Integrated Development Environment). This can be downloaded and installed for free from Microsoft Web Site from the following link: http://www.microsoft.com/express/windows/.

- To run it with R engine, you need to have at least R 2.11.1 installed. The installation process mentioned at the beginning of this section will install R 2.11.1 binary for you, or you can download and install the R runtime from the following CRAN web link: http://cran.r-project.org/bin/windows/base/.

- The Loss Simulator employs the Object-Oriented Programming concept, which is implemented in VB.NET.

- Basic or advanced R programming skills are needed to implement modifications. The Loss Simulator uses embedded R code for all the statistical calculations. Additional R packages may be incorporated in future enhancements. The R (D)COM package is used to bridge the R engine and VB.NET.

### 5.2 Object-Oriented Programming

Object Orientation (OO) is one of the most popular programming concepts in computer science. It includes concepts such as class, object, instance, inheritance, abstraction, polymorphism, encapsulation, overriding, overloading, and the like. Visual Basic 6.0 or VB6.0 (VBA in Excel) is an Object-Based language, since it has introduced concepts like "implements," "class," and "encapsulation." However, it is hard to classify it as an OO language.

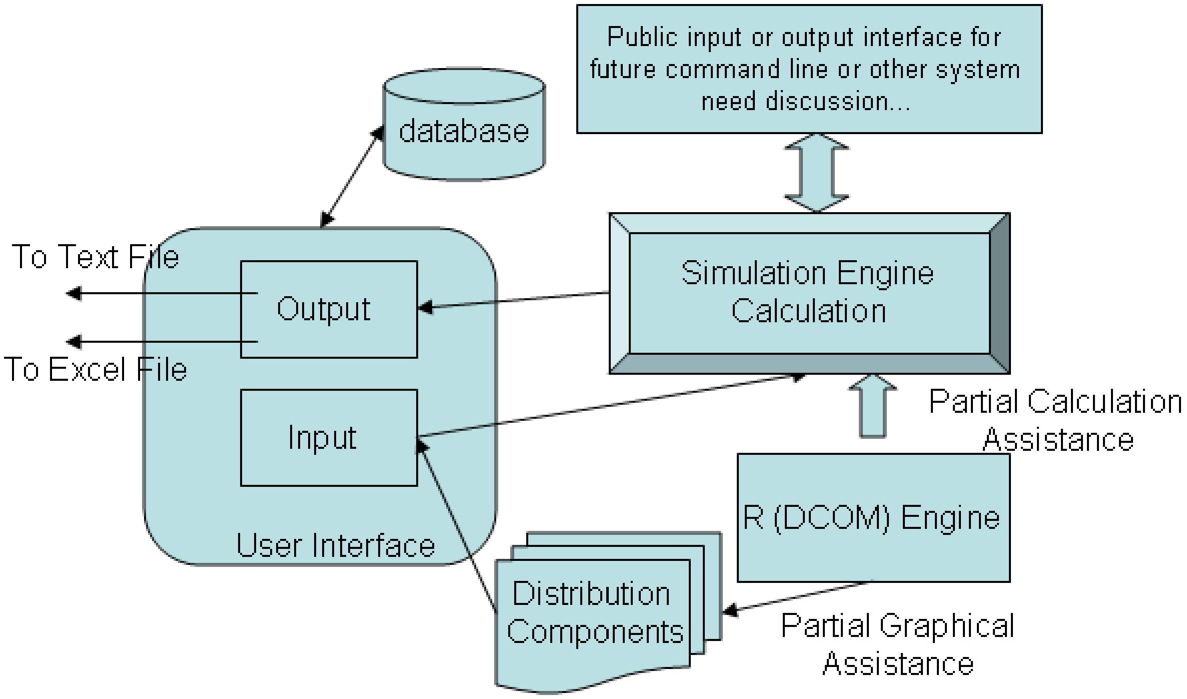
What is the benefit of an OO design? Such a design permits everybody (object) to do its own job.

- For example, if you want to design a car, you can separate the car into tires (big, small, wide, or narrow? Bridgestone?), engine (V8, V12?), body (shape, green, red, silver?), exhaust, electrical systems, etc. Each of these parts of the car is an object and each object has its own properties, methods, or actions. You can assembly all of them together to make a car.

- Another benefit of OO design is that the program can easily achieve multitask jobs (or technically, multithreading) and object sharing. For example, when you drive a car, you must use at least one hand to control the steering, one foot to pedal the gas, two eyes to watch the road, and perhaps one hand to change the radio station. Object (information) sharing is at the brain level. It is very hard (although doable) to simulate this process in some other modular programming languages.

- In addition to the simulation project (object); in the Loss Simulator you will find as the following:
    - The Project has locations (states) and each location has multiple LOB (object).
    - The LOB contains Types or Coverage (object) as its own property (it does not matter how many).
    - LOB or Type have all kinds of distribution objects such as severity, frequency, lags, etc., and numbers (like limits, deductibles, etc.) as properties.
    - When we turn the wheel, we will have actions or methods of these LOB or Type objects (e.g., Monte Carlo simulation and correlation). At the same time, the system will produce occurrences, claims, and transactions (object again, with dollars, days, time, description, etc., as properties).
    - Triangles or rectangles (another object) then can be assembled from these claim data. The triangle object has years and ages as its basic properties and will handle its own output format and development process. The triangle object is a good example of inheriting properties from a grid object.
    - Each object will handle its own database transaction and its own on-screen display, and will communicate to its peer objects.

### 5.3 System Architecture


Public input or output interface for future command line or other system need discussion...
database
To Text File
Output
To Excel File
Input
User Interface
Simulation Engine Calculation
Partial Calculation Assistance
R (DCOM) Engine
Distribution Components
Partial Graphical Assistance

### 5.4 System COM Components

Major components in the system are programmed within COM (Component Object Model), which will greatly benefit the future system development, component communications, system deployment, and customizations of each individual component object, yet leave others intact. COM objects can also be loaded into different machines or servers, thus you can assemble a distributed system in a multi-tier environment.

Each component is organized into package in the IDE. Loss Simulator system has CASLDS package, Distribution Object package, Database Object package, Public Reference package, R Engine package, and Simulation Object package.

The CASLDS package handles all of the system interface. It has customized controls and forms to load and display each simulation object. The main entrance of the program is from frmMain.vb, where the system will verify licenses (which are free in our case), R installation, R D(COM) installation, and required R packages, etc. A simulation is also carried out from this package, while the frmSimulationResult.vb form will dynamically display simulation progress, messages, triangles, claims, and percentile tables.

The Distribution Object package contains all the built-in distributions and Copulas. If your customization project requires additional distributions, you will need to work on this package also. Otherwise, the system default distributions, listed in the section 4 of this Appendix A, should be sufficient for many applications.

The Database Object package helps load and save the whole simulation project into any database servers using OleDB.NET technology. The installation comes with a default MS Access database, systematically handled by AccessData.vb class. You can link the system to any other kind of relational database server, providing a handler for each type of database by yourself. Besides the MS Access handler, system provides a default MySQL database handler, named MySQLData.vb.

Modelers don't need to touch the R Engine and Public Reference packages. The R Engine package provides wrapper objects that will utilize R D(COM) interface to run R commands, returning results and messages. Public Reference is a helper package which contains objects such as: system event handler, matrix, hash table, XML, and some globally used constants.

Most of the major actuarial related objects and processes are built inside the Simulation Object package, where the user can find objects like Line, Type, Simulation, Claim, Transaction, and Triangle, etc. This is also where modelers will have to focus on most of their customization work, and so there is a need to read and understand the program in this package in detail, probably line-by-line.

### 5.5 How is R language implemented in Loss Simulator?

The R language is a free resource language, with many statistical packages available and ready to use. The other reasons the Loss Simulator uses R as a backend calculation engine include its speed and reliability, so that the working party members don't have to spend a huge amount of resources building up the statistical utilities.

There is a long learning curve to master the R language. R is a vector language, such that R programmers should have to think in a vector way, not a modular way. Please refer to CRAN links on how to use and program in R.

R D(COM) service is used to bridge R Engine and VB.NET. But due to R D(COM)'s immaturity and limitation, e.g., limited primitive data returning ability, this phase of the Loss Simulator is not using a fully functional, separated, and independent R package for the complicated claim simulation. Even though we have provided a caslds R package, which contains the fully functional multiple random payment pattern claim simulation, its results (dataframe or matrix data type in R) cannot be simply returned back to front end.

Another reason for not using a fully independent R package is due to its limitation on Object Oriented programming. Even though people declare R is OO language, it really has less power to handle complicated data types and dynamic needs. R is very strong at statistical calculations, so we will have to use that strength wisely, by building the complicated front end with VB.NET.

Let us use a simple example to illustrate the difficulty by using an independent and powerful R. In Loss Simulator, most of the distributions are stored as a monthly value. Let us think about how to build this data structure in R. And suppose we have made the following section of R code (it is fully functional).

```
###############################################
##This function will store dist objects or any other objects by month
##usage: obj<-dateHash(new("norm," p1=1.20, p2=0.5))
###############################################
dateHash<-function(x){
        l=480    #40 years
        dates<-seq(simulationStartDate, by="month," length.out=l)
        z<-rep(list(x), length(dates))
        names(z)<-dates
        z
}
```

With any object *x* you passed, the function will return a vector of objects, separated monthly and all equal to *x*. Bear in mind that, you have to pass an object, probably a distribution object. Thus you have to wrap your own distribution object in R. Then you have to think about how to let the VB.NET front end handle the user distribution selection dynamically. You also have to make sure that the user is able to set the rest of the monthly values back to its original, and more and more.

So, by comparing the pros and cons, we decided to make the solution simple as, e.g. define object in VB.NET, yet carry real statistical calculation in R. In reviewing the following piece of VB.NET code, you will see how R is incorporated:

```
    ‘ This function will help to draw pictures for the user selection
    Public Overrides Function getRPicString() As String
       Return "par(mfrow = c(2, 2)) " & vbLf & _
              "x<-seq(0, " & (meanlog + 50 * sdlog) & ," length=500)" & vbCrLf & _
              "plot(x, dlnorm(x, " & " meanlog =" & meanlog & ," sdlog=" & sdlog & "), type=""l"," main
= ""Lognormal  Density"," col=""blue"")" & vbLf & _
              "plot(x, plnorm(x, " & " meanlog =" & meanlog & ," sdlog=" & sdlog & "), type=""l"," main
= ""Lognormal  Cumulative"," col=""red"") " & vbCrLf & _
              "hist(" & getRSampleString(200) & ," breaks=1000, xlim = c(" & Min & ," " & Max & "),
main=""Samples"," xlab=""Value"," ylab=""Count"," col='blue1')"
    End Function
```

```
'random sampling, using R's rlnorm function
Public Overrides Function getRSampleString(Optional ByVal n As Integer = 1000) As String
   Return "rlnorm(" & n & ," meanlog =" & meanlog & ," sdlog=" & sdlog & ")"
End Function
```

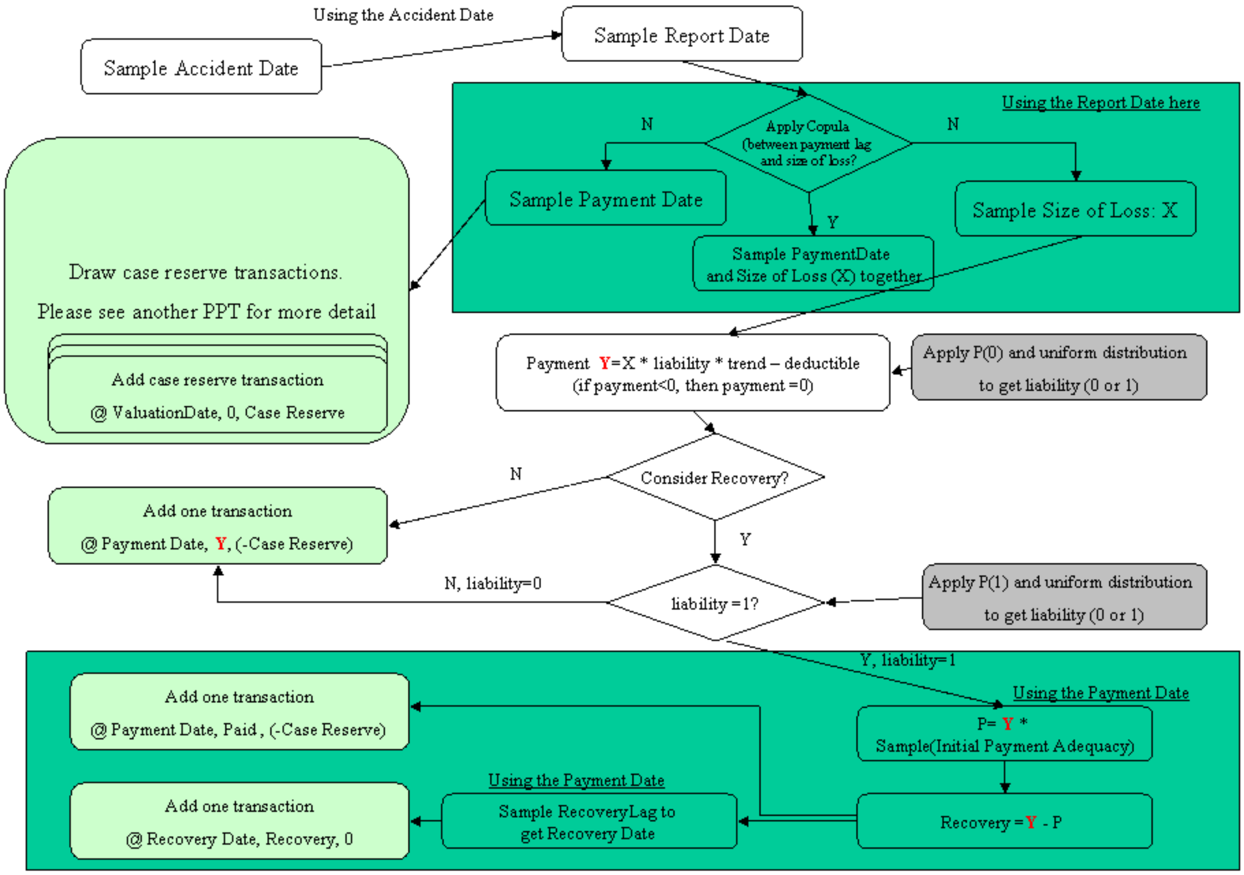
By issuing a simple call to R, you can get the same result as if you had programmed in a fully independent R package.  However, we believe that this approach yields a more powerful solution.

Future development of the Loss Simulator will likely use more independent R packages. For example, during the test of the simulator, the testing group directed by Joe Marker has developed many R packages for curve fittings and distribution parameterizations. At the same time, the developers have already transformed those codes into Object Oriented R. There will be great benefit to reuse those packages for the next phase of development.

### 5.6 Sample Flowchart to Calculate Ultimate Loss and Case Reserves for Single Payment Type

The following sample chart is also in the help file. The reason we put it here is to help programmers understand how the simulation is carried out for the ultimate loss, case reserves, and adjustments, in a single payment pattern model.

**How to calculate the ultimate loss from Claim Severity**


Sample Accident Date
Using the Accident Date
Sample Report Date
Using the Report Date here
Apply Copula (between payment lag and size of loss?
N
N
Y
Sample Payment Date
Sample Size of Loss: X
Sample PaymentDate and Size of Loss (X) together
Draw case reserve transactions.
Please see another PPT for more detail
Add case reserve transaction
@ ValuationDate, 0, Case Reserve
Payment Y=X * liability * trend – deductible
(if payment<0, then payment =0)
Apply P(0) and uniform distribution
to get liability (0 or 1)
Consider Recovery?
N
Y
Add one transaction
@ Payment Date, Y, (-Case Reserve)
N, liability=0
liability =1?
Apply P(1) and uniform distribution
to get liability (0 or 1)
Y, liability=1
Using the Payment Date
P= Y *
Sample(Initial Payment Adequacy)
Add one transaction
@ Payment Date, Paid , (-Case Reserve)
Add one transaction
@ Recovery Date, Recovery, 0
Using the Payment Date
Sample RecoveryLag to
get Recovery Date
Recovery = Y - P

# APPENDIX B

## 6.2.1 Test of Elementary Frequencies, Trend, and Zero-modification

This is a complete description of the parameters for this run:

Test run 10/27/2009

Project name: Frequency Test

Purpose: Test frequency with trend. Two types within one line.

- One Line with annual frequency Poisson(120)
- Set claim/acc distribution matrix as follows:
  Prob =75% that one Type 1 claim is generated.
  Prob =25% that one Type 2 claim is generated.
  - Freq Trend: 1.02 constant throughout
  - P(0) = 0.4, EstP(0) = 0.4 for each Type.
  - Accident Years: 2000-2002
  - Random Seed: 16807
  - Frequency correlation copula: normal Correlation=c( ) Dim = 1

Other frequency parameters
Monthly exposure: (1)
Seasonality: 1.0

The severity parameters were not used in this run. They are presented here only for completeness.

Type 1 and Type 2 severity are lognormal with different parameters:
Type 1: Lognormal, mean=100,000, stnd. dev. 100,000, max 1,000,000.
Type 2: Lognormal, mean 10,000, stnd. dev. 5,000, max 1,000,000.
Zero trend for severity.

Lags: Irrelevant for this run. We are not testing.
Report lag: Exponential with Rate = 4/365, mean=365/4 days. Max=365.
Payment lag: Exponential with Rate = 1/365, mean=365 days. Max = 700.
Inter-valuation lag: Exponential with Rate = 4/365, mean = 365/4 days. Max=365.

Correlation of Amount with lag: normal Correlation=c( ) Dim=2
Reserve adequacy: Irrelevant, leave at default values.
Recovery lag box: Not checked.
Alpha = 0 Inertia=0.2

Interpolation parameters: irrelevant
P(1) not set because recovery lag was not checked.
Minimum change = Min Rel Chg = 0.

Initial payment adequacy not set because recovery lag was not checked.

Run: 1,000 simulations.

Actuaries need tools that will enable them to better understand the underlying loss development process and will aid them in determining what methods and models work best in different reserving situations.

Section 6.2.1 discussed a series of GLM runs that investigated the effect of predictors on claim counts. This section provides more detail on the S-PLUS statements used.

Following is the full output from the S-PLUS "glm" command defining the "reduced" model5x:

```
> model5x<- glm(count ~   + Type + Status,
+      data = temp.datacc.stack,
+      family = poisson,
+      x=T)
>
>
> summary(model5x,correlation=F)

Call: glm(formula = count ~  + Type + Status, family = poisson, data = 
       temp.datacc.stack, x = TRUE)
Deviance Residuals:
        Min           1Q        Median          3Q        Max 
 -3.0487245 -0.817678553 -0.0481310816 0.498450005 4.53059039

Coefficients:
                   Value    Std. Error     t value 
(Intercept)  1.126272675 0.00276627012  407.144865
       Type -1.097685774 0.00378962189 -289.655751
     Status  0.410026757 0.00335189741  122.326762

(Dispersion Parameter for Poisson family taken to be 1 )

    Null Deviance: 273221.814 on 143999 degrees of freedom

Residual Deviance: 160969.366 on 143997 degrees of freedom

Number of Fisher Scoring Iterations: 5
```

Following is the result of the "glm" command for the full model model6x:

```
> ####
> #### The only possible variable to add is Type*Status
> ####   Just create the bigger model and compare
> ####       No need to use stepAIC here.
> #####
> model6x<- glm(count ~  Type + Status + Type*Status                       ,
+      data = temp.datacc.stack,
+      family = poisson,
+      x=T)
>
>
> summary(model6x)

Call: glm(formula = count ~ Type + Status + Type * Status, family = poisson, data = 
       temp.datacc.stack, x = TRUE)
Deviance Residuals:
```

```
         Min           1Q        Median          3Q        Max
 -3.04838755 -0.817233217 -0.0487132512 0.497838102 4.52978802

Coefficients:
                     Value    Std. Error        t value
(Intercept)  1.12660568075 0.00300060918  375.458986266
       Type -1.09901753526 0.00600212647 -183.104694918
     Status  0.40947269733 0.00387066987  105.788587162
Type:Status  0.00221507211 0.00773994871    0.286186924

(Dispersion Parameter for Poisson family taken to be 1 )

    Null Deviance: 273221.814 on 143999 degrees of freedom

Residual Deviance: 160969.284 on 143996 degrees of freedom

Number of Fisher Scoring Iterations: 5

Correlation of Coefficients:
             (Intercept)         Type       Status
       Type -0.499924351
     Status -0.775217024  0.387549868
Type:Status  0.387678174 -0.775473675 -0.500089861
>
```

Results of the test to see whether interactive variable can be dropped:

```
>
> anova(model5x,model6x,test=“Chi”)
Analysis of Deviance Table

Response: count

                         Terms Resid. Df Resid. Dev         Test Df
1                + Type + Status    143997 160969.366
2 Type + Status + Type * Status    143996 160969.284 +Type:Status  1

      Deviance     Pr(Chi)
1
2 0.0819088429 0.774727081
>
>
```

One way to test whether a Poisson GLM is appropriate is to see how close the dispersion parameter is to 1.0. Following is an execution of this process.[13]

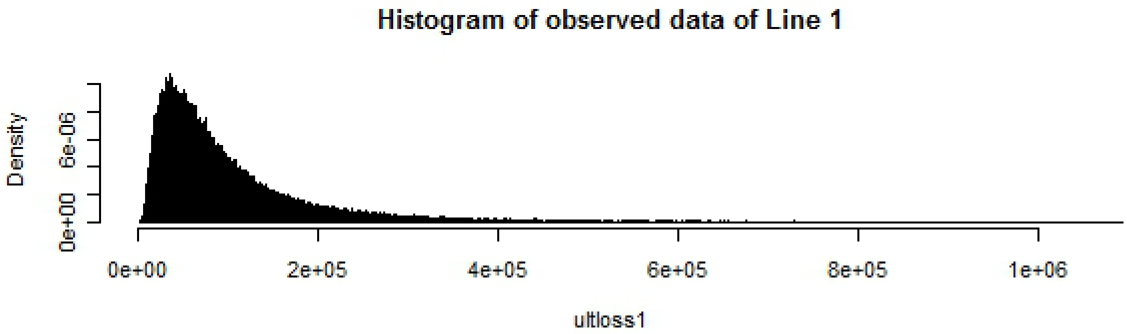
```
> ####
> ####   Estimate dispersion parameter - see Faraway p 60.
> ####
>  temp.pearson <- residuals(model5x,type=“pearson”)  ###  (y-fitted y)/(y^0.5)
> model5x$df.residual
[1] 143997
> tempdp <- sum(temp.pearson^2)/model5x$df.residual
>     cat(“\n     Estimate   of   dispersion   parameter   using   Pearson   residuals.
disp=,”tempdp,”\n”)

  Estimate of dispersion parameter using Pearson residuals.  disp= 1.00653317880341
>
>
```

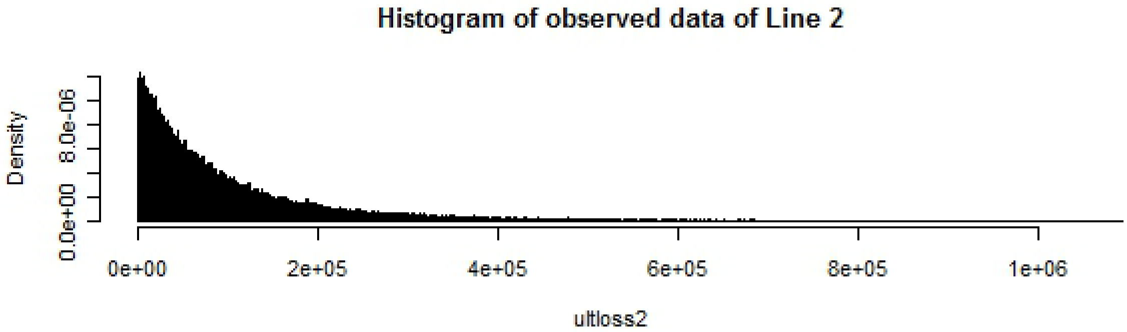
[13] [5] Faraway 2006, p. 60.

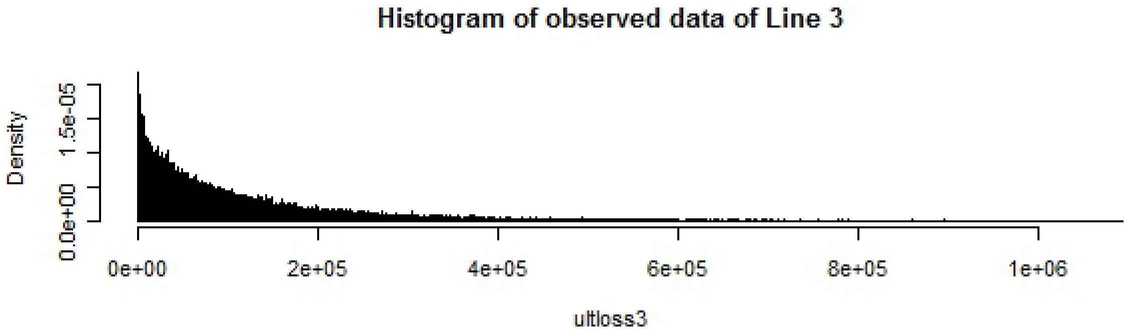
```
> temp.pearson <- residuals(model6x,type=“pearson”)     ### (y-fitted y) / (y^0.5)
> model6x$df.residual
[1] 143996
> tempdp <- sum(temp.pearson^2)/model6x$df.residual
>     cat(“\n     Estimate   of   dispersion   parameter   using   Pearson   residuals.
disp=,”tempdp,”\n”)

  Estimate of dispersion parameter using Pearson residuals.  disp= 1.00659609377679
>
> ####
> ####   Dispersion in both cases very close to 1.0, supporting Poisson model
> ####
>
```

## Appendix to section 6.2.2 Test of Severity Distributions

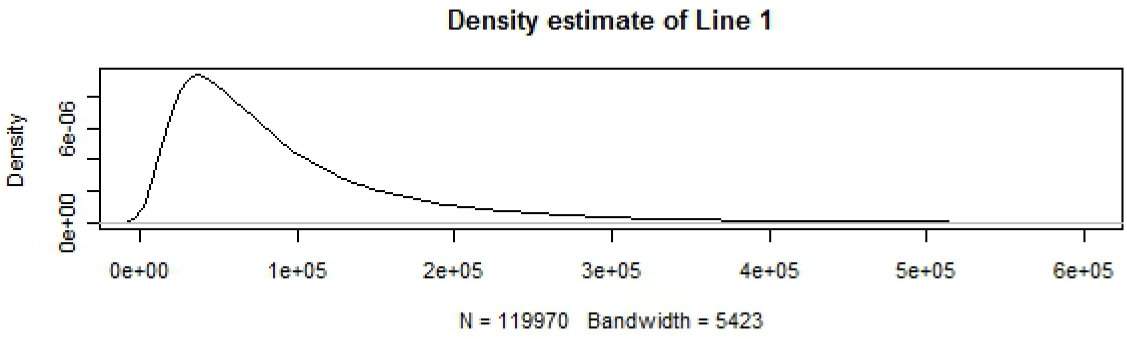
Following is the set of histograms, empirical densities and log densities, and empirical cdfs for the size of loss for each line.

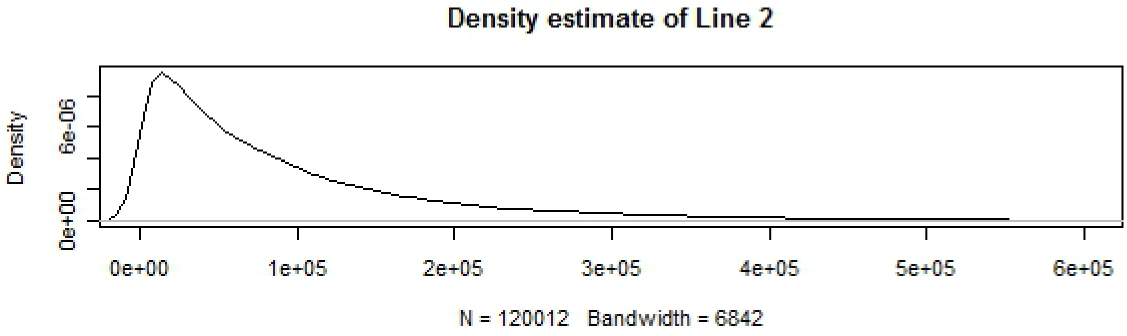

Histogram of observed data of Line 1

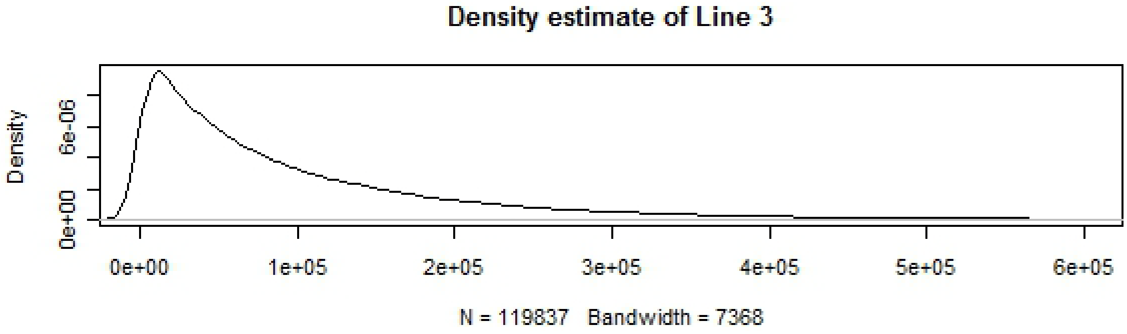

Histogram of observed data of Line 2


Histogram of observed data of Line 3

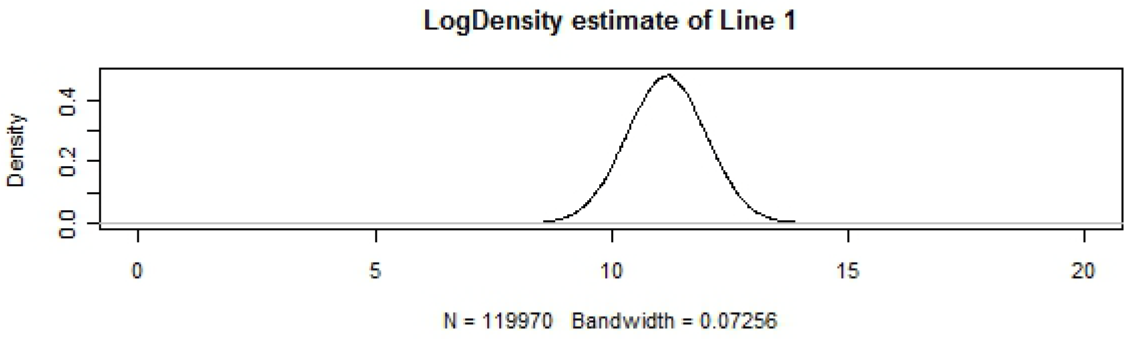
**Density estimates**

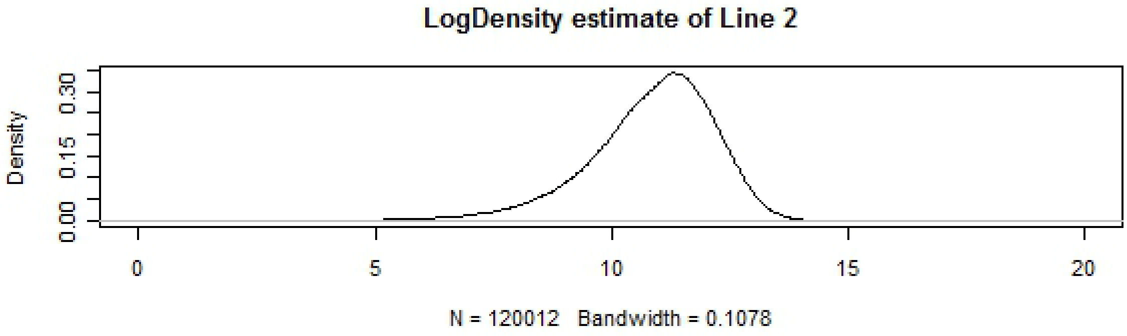


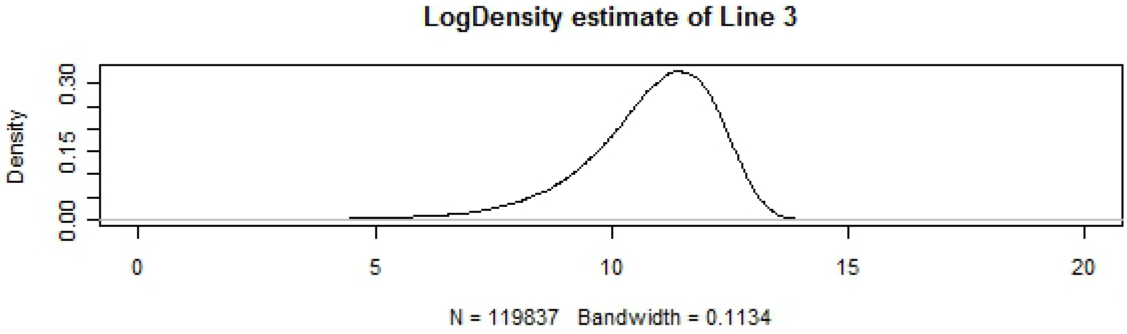




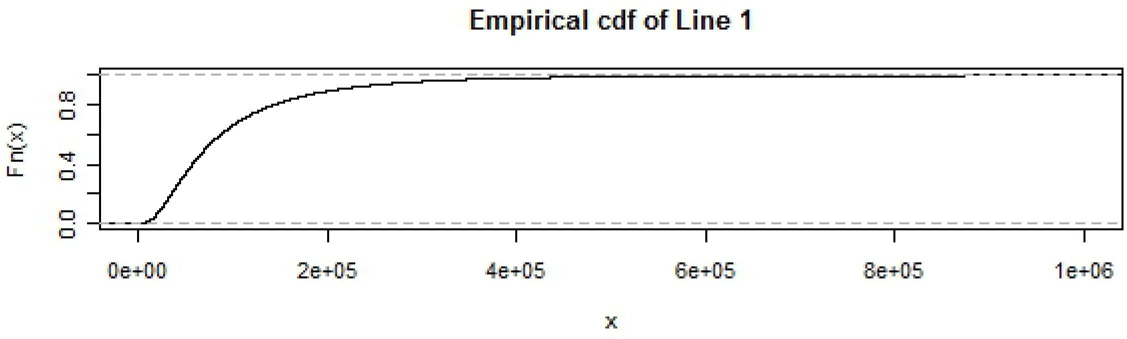
## Log densities

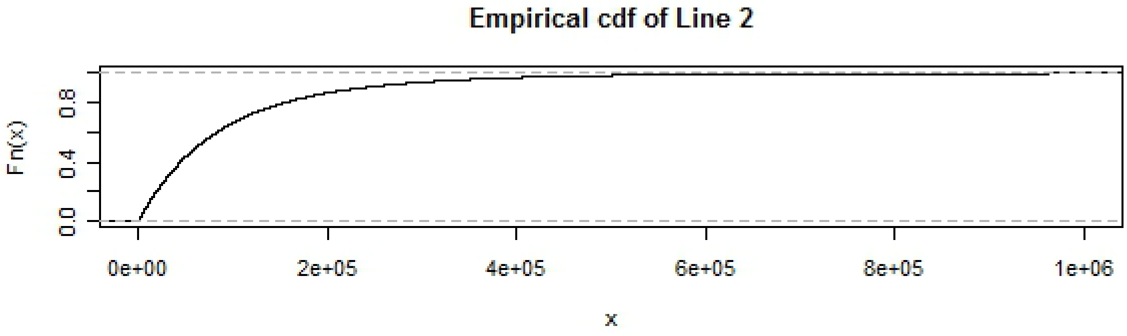

LogDensity estimate of Line 1

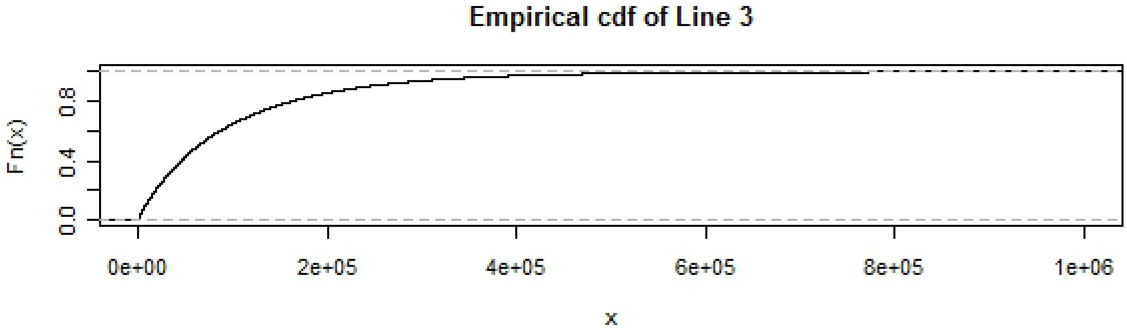

LogDensity estimate of Line 2


LogDensity estimate of Line 3

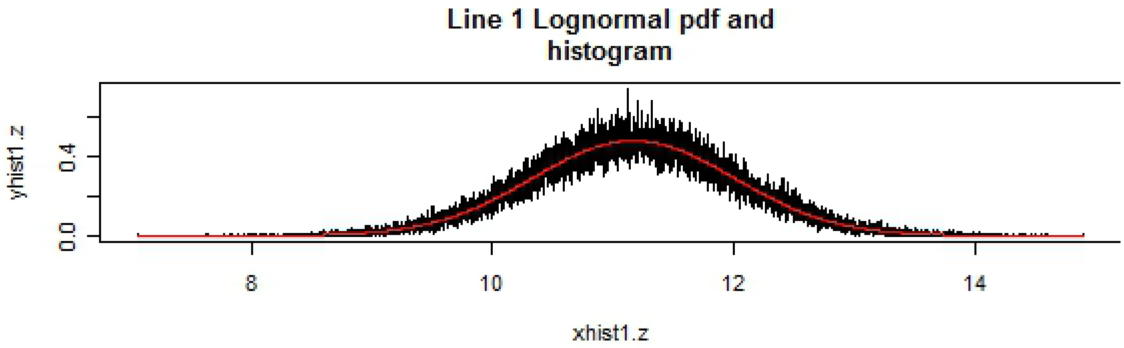
## Empirical distribution functions

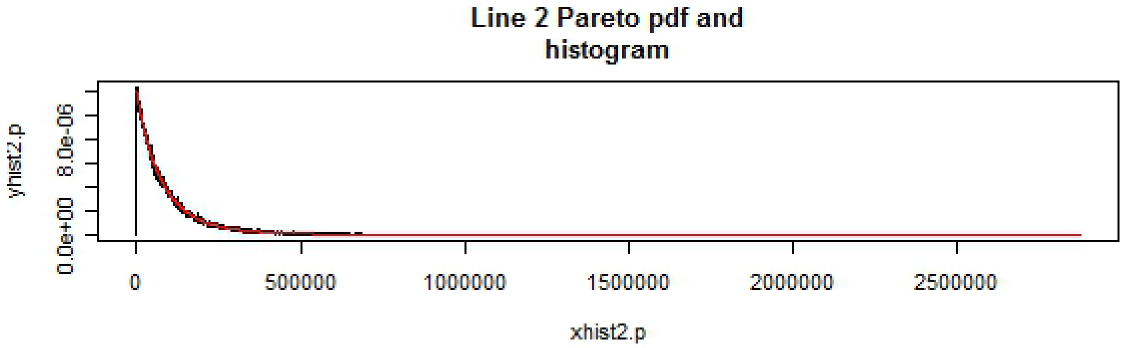


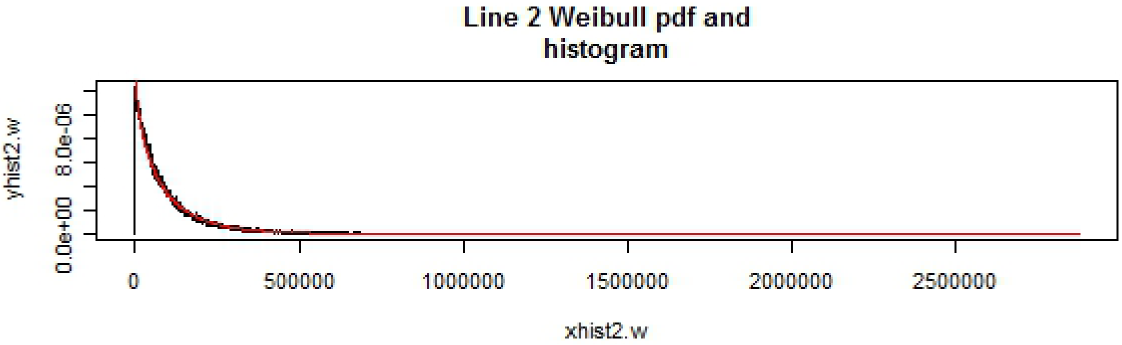




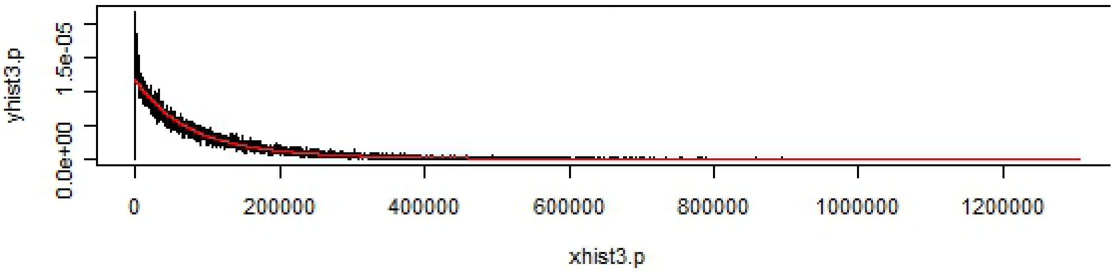
We next show the histograms and density functions for each model that produced a reasonable fit to the observed data.



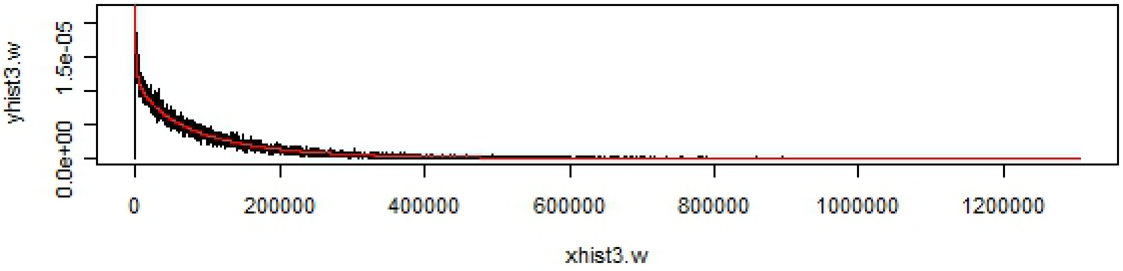




**Line 3 Pareto pdf and histogram**



**Line 3 Weibull pdf and histogram**



Following is the complete R code for the severity testing.

```
library(stats4)
library(MASS)
library(actuar)
library(graphics)

############ Part I: Exploratory Data Analysis ############

ultloss1<-as.vector(scan(“line 1 ultloss.txt”))
ultloss2<-as.vector(scan(“line 2 ultloss.txt”))
ultloss3<-as.vector(scan(“line 3 ultloss.txt”))

mean1 <- mean(ultloss1)
var1 <- var(ultloss1)

mean2 <- mean(ultloss2)
var2 <- var(ultloss2)

mean3 <- mean(ultloss3)
var3 <- var(ultloss3)

n1<-length(ultloss1)
n2<-length(ultloss2)
n3<-length(ultloss3)

##############################################################################
#          Histograms, empirical density/log density, empirical cdf          #
##############################################################################

par(mfrow=c(3,1)) # Histograms
hist(ultloss1,main=“Histogram       of       observed       data       of       Line
1,”freq=FALSE,breaks=10000,xlim=c(0,1050000))
hist(ultloss2,main=“Histogram       of       observed       data       of       Line
2,”freq=FALSE,breaks=10000,xlim=c(0,1050000))
hist(ultloss3,main=“Histogram       of       observed       data       of       Line
3,”freq=FALSE,breaks=10000,xlim=c(0,1050000))

par(mfrow=c(3,1)) # Density
plot(density(ultloss1),main=“Density estimate of Line 1,”xlim=c(-1000,600000))
plot(density(ultloss2),main=“Density estimate of Line 2,”xlim=c(-1000,600000))
plot(density(ultloss3),main=“Density estimate of Line 3,”xlim=c(-1000,600000))

par(mfrow=c(3,1)) # Log Density
plot(density(log(ultloss1)),main=“LogDensity estimate of Line 1,”xlim=c(0,20))
plot(density(log(ultloss2)),main=“LogDensity estimate of Line 2,”xlim=c(0,20))
plot(density(log(ultloss3)),main=“LogDensity estimate of Line 3,”xlim=c(0,20))

par(mfrow=c(3,1)) # Empirical Cumulative Distribution Function
plot(ecdf(ultloss1),main=“Empirical cdf of Line 1,”xlim=c(0,1e+06))  ## lightest tail
plot(ecdf(ultloss2),main=“Empirical cdf of Line 2,”xlim=c(0,1e+06))  ## heaviest tail
plot(ecdf(ultloss3),main=“Empirical cdf of Line 3,”xlim=c(0,1e+06))  ## heavier tail

############ Part II: Maximum Likelihood Estimates of Parameters ############

########################################################
#       Pareto (shape=alpha, scale=theta)              #
#  pdf f(x) = alpha*theta^alpha/((x+theta)^(alpha+1))  #
########################################################

####################################################################
#               Weibull(shape=tao, scale=lambda)                   #
# pdf f(x) = tao*x^(tao-1)/(lambda^tao)*exp(-(x/lambda)^(tao))      #
# nll=-log(tao/(lambda^tao))-(tao-1)*log(x)+(x/k)^tao               #
#####################################################################

###############################################################
#                  lognormal(miu,sigma)                       #
#   pdf f(x)=1/(sqrt(2*pi)*sigma)*exp(-(x-miu)^2/(2*sigma^2)) #
#   nll = log(sqrt(2*pi)*sigma)+(x-miu)^2/(2*sigma^2)         #
###############################################################


ultloss1.0 <- ultloss1[ultloss1!=0]
n1.0 <- length(ultloss1.0)

ultloss2.0 <- ultloss2[ultloss2!=0]
n2.0 <- length(ultloss2.0)

ultloss3.0 <- ultloss3[ultloss3!=0]
n3.0 <- length(ultloss3.0)


                             ##########
                             # Line 1 #
                             ##########

## From the exploratory analysis, it is clear that Line 1 comes from lognormal.

fit1.ln <- fitdistr(log(ultloss1),”normal”)
fit1.ln$estimate  #   mean        sd
                  #11.1659376  0.8361509
-fit1.ln$loglik   #148761.9

fit.ln1 <- fitdistr(log(ultloss1.0),dnorm,list(mean=miu1,sd=sigma1))
fit.ln1$estimate  #   mean        sd
                  #11.1659362  0.8361455
-fit.ln1$loglik   #148761.9

## QQ plot ##
qqnorm(log(ultloss1),main=“Line 1, Lognormal”)
abline(0,1,col=“red”)


## Chi-Square Test ##

ult1.cut <- cut(log(ultloss1),breaks = seq(0,9,10,11,12,13,22)) ## binning data
table.ult1 <- table(ult1.cut)                        ## binned data table
ult1.os <- c(as.vector(table.ult1))                  ## vectorization

b = length(ult1.os)

labs <- levels(ult1.cut)                             ## extract the breakpoints
break.1  <-  cbind(lower  =  as.numeric(sub(“\\((.+),.*,”  “\\1,”  labs)),upper  =
as.numeric(sub(“[^,]*,([^]]*)\\],” “\\1,” labs)))

ult1.ln <- pnorm(break.1,mean=fit.ln1$estimate[1],sd=fit.ln1$estimate[2])

ult1.prob <- (ult1.ln[,2]-ult1.ln[,1])[1:b-1]     ## cut off at the last bin to absorb
all tail prob
ult1.ex <- n1*c(ult1.prob,1-sum(ult1.prob))

#cbind(ult1.ex,ult1.os)

E.1 = ult1.ex
O.1 = ult1.os

x.sq.1 = (E.1-O.1)^2/E.1
```

```
#cbind(E.1,O.1,x.sq.1)

##chi-square test statistic##

df=length(E.1)-1-2     ## df = 3

chi.sq.1 <- sum(x.sq.1)          ## test statistic
chi.sq.1                         ## 3.594133
qchisq(.95,df)                   # 7.814728

1-pchisq(chi.sq.1,df)              # 0.308757

## Chi-Square Test in R ##
chisq.test(O.1,p=E.1/n1)

## Chi-squared test for given probabilities
## X-squared = 3.5941, df = 5, p-value = 0.6092

##############################################
# Calculation for the test statistic
#____________________________________________
#           E.1   O.1     x.sq.1
#[1,]   575.073   531 3.377707e+00
#[2,]  9213.966  9254 1.739471e-01
#[3,] 40759.725 40740 9.545298e-03
#[4,] 50315.005 50316 1.969319e-05
#[5,] 17410.264 17434 3.236081e-02
#[6,]  1695.968  1695 5.528125e-04
##############################################

                            ##########
                            # Line 2 #
                            ##########

## 2.1-Pareto ##

fit.p2<-fitdistr(ultloss2.0,dpareto,list(shape=6,scale=500000))  ##  list()  provides
initial values for optimization
fit.p2$estimate #  shape       scale
                #5.97635e+00 5.00000e+05
-fit.p2$loglik  #1500363

## 2.2-Weibull (second method slightly better) ##

fit2.w <- fitdistr(ultloss2.0,”weibull”)
fit2.w$estimate  #   shape         scale
                 # 9.056193e-01 9.750673e+04
fit2.w$loglik    # -1500950

fit.w2 <- fitdistr(ultloss2.0,dweibull,list(shape=.9097626,scale=95000))
fit.w2$estimate #    shape        scale
                # 9.009281e-01 9.500000e+04
-fit.w2$loglik  # 1500926

## QQ plot ##
par(mfrow=c(2,1))
thqua.p2 <- rpareto(n2,shape=fit.p2$estimate[1],scale=fit.p2$estimate[2])
qqplot(ultloss2,thqua.p2,xlab=“Sample          Quantiles,”          ylab=“Theoretical
Quantiles,”main=“Line 2, Pareto”)
abline(0,1,col=“red”)

thqua.w2 <- rweibull(n2,shape=fit.w2$estimate[1],scale=fit.w2$estimate[2])
qqplot(ultloss2,thqua.w2,xlab=“Sample   Quantiles,”   ylab=“Theoretical   Quantiles,”
main=“Line 2, Weibull”)
```

```
abline(0,1,col="red")

## 2.3 Chi-Square Test ##

#2.3.1 Pareto Chi-Square #
m = mean(ultloss2)
s = sqrt(var(ultloss2))

ult2.cut <- cut(ultloss2.0,breaks = c(0,m-
s/2,m,m+s/4,m+s/2,m+s,m+2*s,2*max(ultloss2))) ##binning data
table.ult2 <- table(ult2.cut)                    ## binned data table
ult2.os <- c(as.vector(table.ult2))           ## vectorization

b = length(ult2.os)

labs.2 <- levels(ult2.cut)                       ## extract the breakpoints
break.2  <-  cbind(lower  =  as.numeric(sub("\\((.+),.*,"  "\\1,"  labs.2)),upper  =
as.numeric(sub("[^,]*,([^]]*)\\],"  "\\1," labs.2)))   See note below[14]

ult2.p <- ppareto(break.2,shape=fit.p2$estimate[1],scale=fit.p2$estimate[2]) ## Pareto
cdf values at break points

ult2.prob <- (ult2.p[,2]-ult2.p[,1])[1:b-1]  ## Probabilities of each interval
ult2.ex <- n2.0*c(ult2.prob,1-sum(ult2.prob))  ## Expected frequency of each interval
and the "excess" interval

#cbind(ult2.ex,ult2.os)         ## expected and observed frequencies

E.2 = ult2.ex
O.2 = ult2.os

x.sq.2 = (E.2-O.2)^2/E.2

#cbind(E.2,O.2,x.sq.2)         ## expected, observed, and chi-square of each interval
after full adjustment

##chi-square test statistic##

df=length(E.2)-1-2           ## df = 4

chi.sq.2 <- sum(x.sq.2)    ## test statistic
```

[14] For example, suppose that labs.2 consists of seven intervals:

```
"(0,3.97e+04]"
"(3.97e+04,1e+05]"
"(1e+05,1.31e+05]"
"(1.31e+05,1.61e+05]"
"(1.61e+05,2.22e+05]"
"(2.22e+05,3.43e+05]"
"(3.43e+05,5.75e+06]"
```

Then break.2 looks like:

```
      lower           upper
[1,]      0           39700
[2,]  39700          100000
[3,] 100000          131000
[4,] 131000          161000
[5,] 161000          222000
[6,] 222000          343000
[7,] 343000         5750000
```

```
chi.sq.2                    ## 6.155374

qchisq(.95,df)              ## critical value ## 9.487729

1-pchisq(chi.sq.2,df)       ## p-value  ## 0.1878414

## chi-square goodness-of-fit test from R ##

chisq.test(O.2,p=E.2/n2.0)

#####################################################
#    Chi-squared test for given probabilities
# data:  O.2
# X-squared = 6.1554, df = 6, p-value = 0.406
######################################################


##### Test Statistic Calculation ###########
#____________________________________________
#          E.2   O.2     x.sq.2
#[1,] 43993.890 44087 0.19705959
#[2,] 35651.989 35680 0.02200752
#[3,] 10493.758 10323 2.77864169
#[4,]  7240.583  7269 0.11152721
#[5,]  9277.383  9164 1.38570182
#[6,]  8063.576  8176 1.56743997
#[7,]  5289.820  5312 0.09299630
###########################################

### 2.3.2 Weibull ###
m = mean(ultloss2)
s = sqrt(var(ultloss2))

ult2.cut           <-           cut(ultloss2.0,breaks           =           c(0,m-
s/2,m,m+s/4,m+s/2,m+s,m+2*s,2*max(ultloss2))) ##binning data
table.ult2 <- table(ult2.cut)                   ## binned data table
ult2.os <- c(as.vector(table.ult2))           ## vectorization

b = length(ult2.os)

labs.2 <- levels(ult2.cut)                      ## extract the breakpoints
break.2  <-  cbind(lower  =  as.numeric(sub("\\((.+),.*,"  "\\1,"  labs.2)),upper  =
as.numeric(sub("[^,]*,([^]]*)\\],” "\\1," labs.2)))

ult2.w     <-     pweibull(break.2,shape=fit.w2$estimate[1],scale=fit.w2$estimate[2])
##Weibull cdf values at break points

ult2.prob <- (ult2.w[,2]-ult2.w[,1])[1:b-1]  ## Probabilities of each interval
ult2.ex <- n2.0*c(ult2.prob,1-sum(ult2.prob))  ## Expected frequency of each interval
and the "excess" interval

#cbind(ult2.ex,ult2.os)         ## expected and observed frequencies before adjustment

E.2 = ult2.ex
O.2 = ult2.os

x.sq.2 = (E.2-O.2)^2/E.2

#cbind(E.2,O.2,x.sq.2)        ## expected, observed, and chi-square of each interval
after full adjustment

##chi-square test statistic##

df=length(E.2)-1-2          ## df = 4
```

```
chi.sq.2 <- sum(x.sq.2)    ## test statistic
chi.sq.2                   ## 270.3838

qchisq(.95,df)             ## critical value  ##  9.487729

1-pchisq(chi.sq.2,df)      ## p-value  ## 0

# chi-square goodness-of-fit test from R #
chisq.test(O.2,p=E.2/n2.0)

#####################################################
#  Chi-squared test for given probabilities
#data:  O.2
#X-squared = 270.3838, df = 6, p-value < 2.2e-16
##################################################

## Test Statistic Calculation #############
#_________________________________________
#         E.2     O.2     x.sq.2
#[1,] 43917.914 44087  0.6509914
#[2,] 33982.992 35680 84.7434487
#[3,] 10551.628 10323  4.9538155
#[4,]  7532.284  7269  9.2028627
#[5,] 10023.960  9164 73.7763860
#[6,]  9007.970  8176 76.8402384
#[7,]  4994.251  5312 20.2160763
###########################################


                            ##########
                            # Line 3 #
                            ##########

## 3.1-Pareto ##

fit.p3<-fitdistr(ultloss3.0,dpareto,list(shape=7,scale=6.026793e+05))
fit.p3$estimate #   shape       scale
                #6.966806e+00 6.026793e+05
-fit.p3$loglik  #1499343

## 3.2-Weibull (first method slightly better) ##

fit.w3 <- fitdistr(ultloss3.0,"weibull")
fit.w3$estimate  #    shape       scale
                 # 9.052532e-01 9.907429e+04
-fit.w3$loglik   # 1498920

fit3.w <- fitdistr(ultloss3.0,dweibull,list(shape=0.9,scale=100000))
fit3.w$estimate #   shape       scale
                #9.067305e-01 9.999992e+04
-fit3.w$loglik  #1498955

## QQ plot ##
par(mfrow=c(2,1))
thqua.p3 <- rpareto(n3,shape=fit.p3$estimate[1],scale=fit.p3$estimate[2])
qqplot(ultloss3,thqua.p3,xlab="Sample          Quantiles,"          ylab="Theoretical
Quantiles,"main="Line 3, Pareto")
abline(0,1,col="red")

thqua.w3 <- rweibull(n3,shape=fit.w3$estimate[1],scale=fit.w3$estimate[2])
qqplot(ultloss3,thqua.w3,xlab="Sample   Quantiles,"   ylab="Theoretical   Quantiles,"
main="Line 3, Weibull")
abline(0,1,col="red")

## 3.3 Chi-Square Test ##
```

```
# 3.3.1 Pareto #
m = mean(ultloss3)
s = sqrt(var(ultloss3))
M=max(ultloss3)

ult3.cut <- cut(ultloss3,breaks = c(seq(0,1000000,200000),2000000)) ##binning data
table.ult3 <- table(ult3.cut)                  ## binned data table
ult3.os <- c(as.vector(table.ult3))           ## vectorization

b=length(ult3.os)

labs.3 <- levels(ult3.cut)                     ## extract the breakpoints
break.3  <-  cbind(lower  =  as.numeric(sub("\\((.+),.*,"  "\\1,"  labs.3)),upper  =
as.numeric(sub("[^,]*,([^]]*)\\],” “\\1,” labs.3)))

ult3.p <- ppareto(break.3,shape=fit.p3$estimate[1],scale=fit.p3$estimate[2])

ult3.prob <- (ult3.p[,2]-ult3.p[,1])[1:b-1]
ult3.ex <- n3*c(ult3.prob,1-sum(ult3.prob))

#cbind(ult3.ex,ult3.os)

E.3 = ult3.ex
O.3 = ult3.os

x.sq.3 = (E.3-O.3)^2/E.3

#cbind(E.3,O.3,x.sq.3)

# Chi-Square Test Statistic #

df=length(E.3)-1-2  # df = 3

chi.sq.3 <- sum(x.sq.3)    ## chi-square test statistic
chi.sq.3                   ## 275.7469

qchisq(.95,df)             ##critical value 7.814728

1-pchisq(chi.sq.3,df)      ##p-value  ## 0

# chi square test in R #

chisq.test(O.3,p=E.3/n3)

########################################################
#     Chi-squared test for given probabilities
#  data:  O.3
# X-squared = 275.7469, df = 5, p-value < 2.2e-16
########################################################

####### Test Statistic Calculation #######
#_________________________________________
#            E.3    O.3     x.sq.3
#[1,] 103561.6008 102680   7.504906
#[2,]  12820.7894  13997 107.908432
#[3,]   2481.6241   2535   1.148034
#[4,]    639.8169    491  34.613767
#[5,]    201.5347    102  49.158580
#[6,]    131.6340     32  75.413160
##########################################


# 3.3.2 Weibull
m = mean(ultloss3)
```

```
s = sqrt(var(ultloss3))
M=max(ultloss3)

ult3.cut <- cut(ultloss3,breaks = c(seq(0,1000000,200000),2000000)) ##binning data
table.ult3 <- table(ult3.cut)                   ## binned data table
ult3.os <- c(as.vector(table.ult3))           ## vectorization

b=length(ult3.os)

labs.3 <- levels(ult3.cut)                      ## extract the breakpoints
break.3  <-  cbind(lower  =  as.numeric(sub(“\\((.+),.*,”  “\\1,”  labs.3)),upper  =
as.numeric(sub(“[^,]*,([^]]*)\\],” “\\1,” labs.3)))

ult3.w <- pweibull(break.3,shape=fit.w3$estimate[1],scale=fit.w3$estimate[2])

ult3.prob <- (ult3.w[,2]-ult3.w[,1])[1:b-1]
ult3.ex <- n3*c(ult3.prob,1-sum(ult3.prob))

#cbind(ult3.ex,ult3.os)

E.3 = ult3.ex
O.3 = ult3.os

x.sq.3 = (E.3-O.3)^2/E.3

#cbind(E.3,O.3,x.sq.3)

# Chi-Square Test Statistic #

df=length(E.3)-1-2

chi.sq.3 <- sum(x.sq.3)    ## chi-square test statistic
chi.sq.3                   ## 70.21185

qchisq(.95,df)             ##critical value  ## 7.814728

1-pchisq(chi.sq.3,df)      ##p-value 3.885781e-15

# chi square test in R #

chisq.test(O.3,p=E.3/n3)
#####################################################
#      Chi-squared test for given probabilities
# data:  O.3
# X-squared = 70.2119, df = 5, p-value = 9.259e-14
######################################################

### Test Statistic Calculation #################
#_______________________________________________
#           E.3      O.3     x.sq.3
#[1,] 101709.59203 102680  9.2586314
#[2,]  14641.17032  13997 28.3416822
#[3,]   2759.98729   2535 18.3404039
#[4,]    567.24209    491 10.2475765
#[5,]    122.92003    102  3.5604243
#[6,]     36.08824     32  0.4631346
#################################################
```

## Appendix to section 6.2.3—Testing Correlated Frequencies

The following material is the “R Code” used to produce the results in 6.2.3.

```
library(MASS)
library(methods)
library(mvtnorm)
library(scatterplot3d)
library(mnormt)
library(sn)
library(pspline)
library(copula)

# import data
#annual frequency for each line and each simulation
datar<-read.csv(“D:/LSMWP/byyear.csv”)
summary(datar)
n<-length(datar$Line.1)
set.seed(123)
x<- sapply(datar, rank, ties.method = “random”) / (n + 1)
x12<-subset(x,select=-Line.3)
x13<-subset(x,select=-Line.2)
x23<-subset(x,select=-Line.1)
plot(x12)
plot(x13)
plot(x23)
write.csv(x12,”D:/LSMWP/x12ry.csv”)
write.csv(x23,”D:/LSMWP/x23ry.csv”)
write.csv(x13,”D:/LSMWP/x13ry.csv”)

#Set up copula object for copula distribution and goodness-of-fit test later
normal.cop <- normalCopula(c(0,0,0),dim=3,dispstr=“un”)

#Copula fit with prespecified type.

date()
fit.normal<-fitCopula(normal.cop,x,method=“ml”)
fit.normal

fit.normal<-fitCopula(normal.cop,x,method=“itau”)
fit.normal

date()

#Copula Goodness-of-fit test

date()

normal2.cop <- normalCopula(c(0),dim=2,dispstr=“un”)
gofCopula(normal2.cop, x12, N=100, method = “mpl”)
gofCopula(normal2.cop, x13, N=100, method = “mpl”)
gofCopula(normal2.cop, x23, N=100, method = “mpl”)


#gofCopula(normal.cop, x, N=100, method = “mpl”)
#gofCopula(normal.cop, x, N=100, method = “itau”)
date()

#K-S test.
normal.fit12<-normalCopula(0, dim=2)
normal.fit13<-normalCopula(0.99,dim=2)
normal.fit23<-normalCopula(-0.01,dim=2)

y12<-rcopula(normal.fit12,n)
y23<-rcopula(normal.fit23,n)
y13<-rcopula(normal.fit13,n)
ks.test(x12,y12)
ks.test(x13,y13)
ks.test(x23,y23)
```

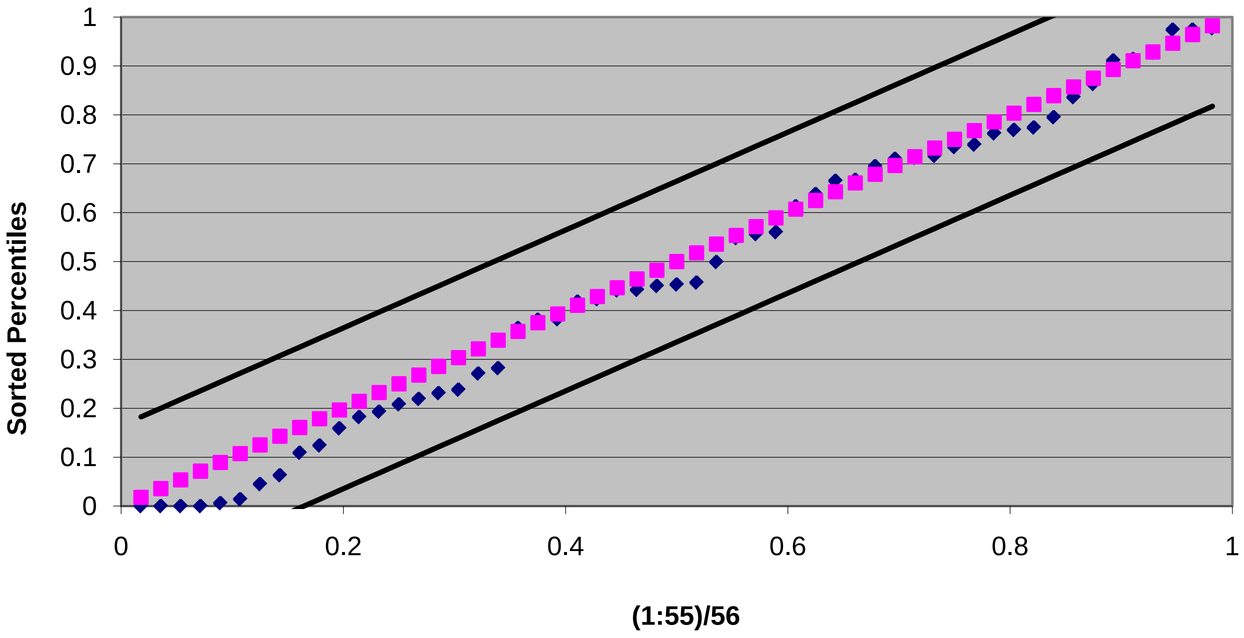
The following table shows the first three records and last two records from "D:/LSMWP/byyear.csv," the dataset used in the R code above.

| Line 1 | Line 2 | Line 3 |
|---:|---:|---:|
| 114 | 95 | 117 |
| 89 | 85 | 90 |
| 94 | 119 | 99 |
| …. | …. | …. |
| 94 | 113 | 94 |
| 105 | 97 | 105 |

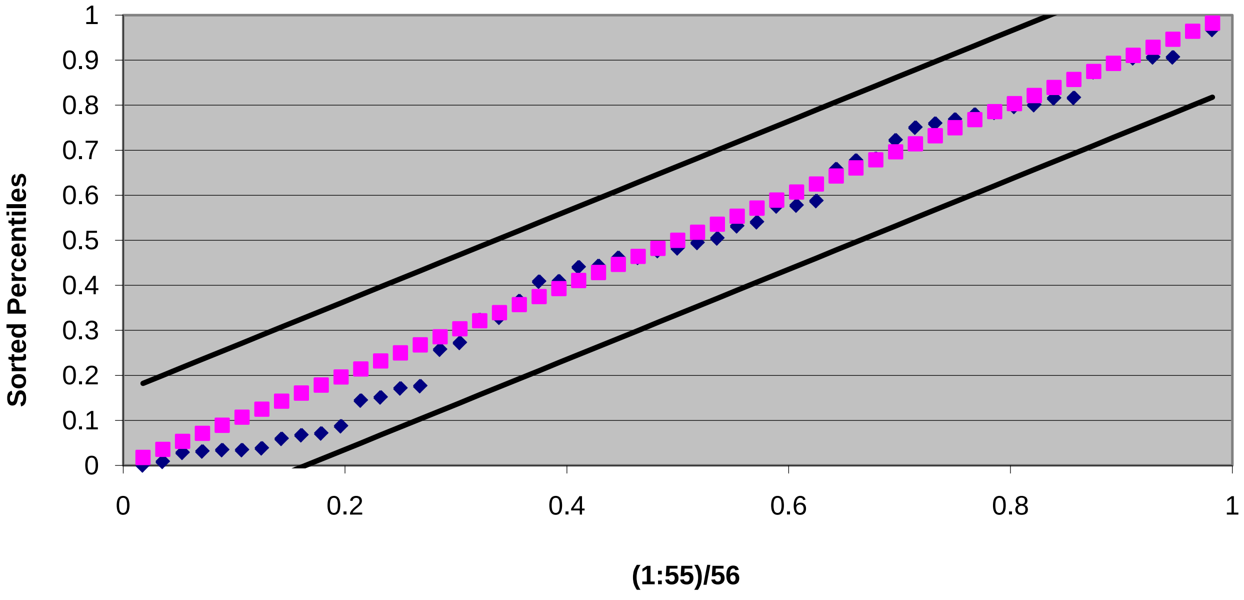
Each row represents the number of claims by line for one simulation.

## Appendix C

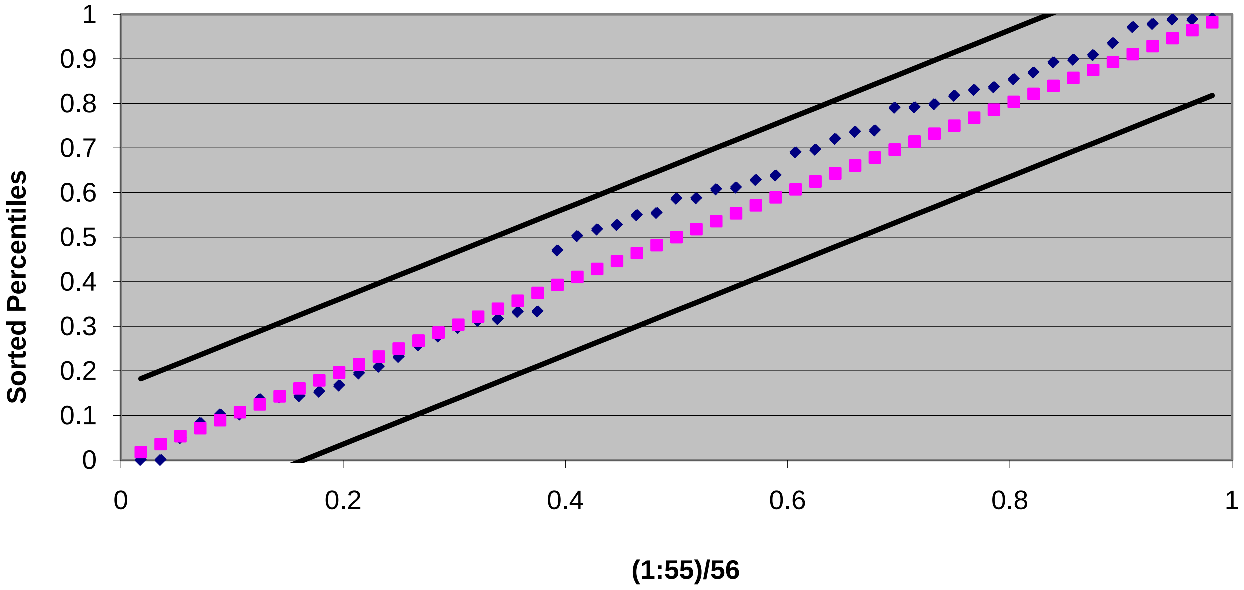
### Candidate 1: Appendix C.1



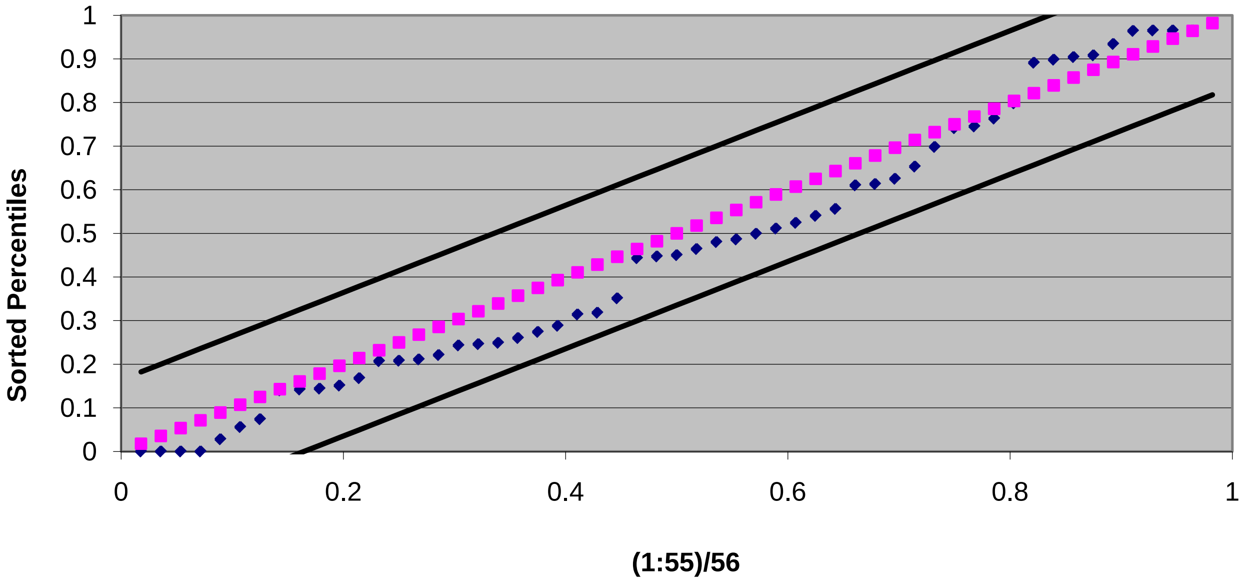
### Candidate 2: Appendix C.2



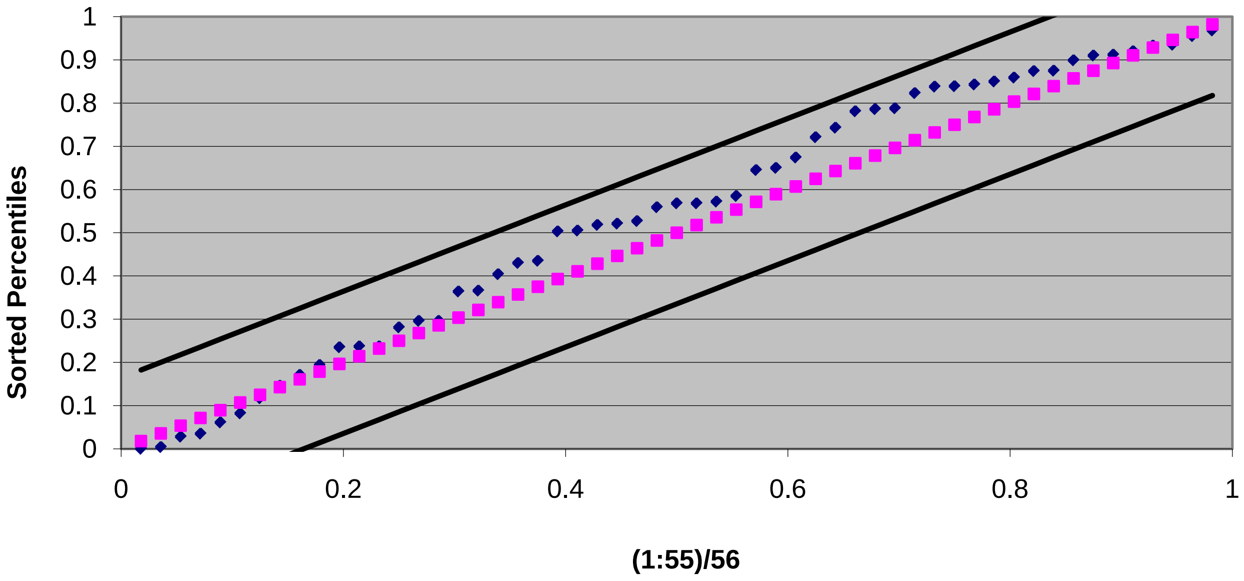
## Candidate 3: Appendix C.3



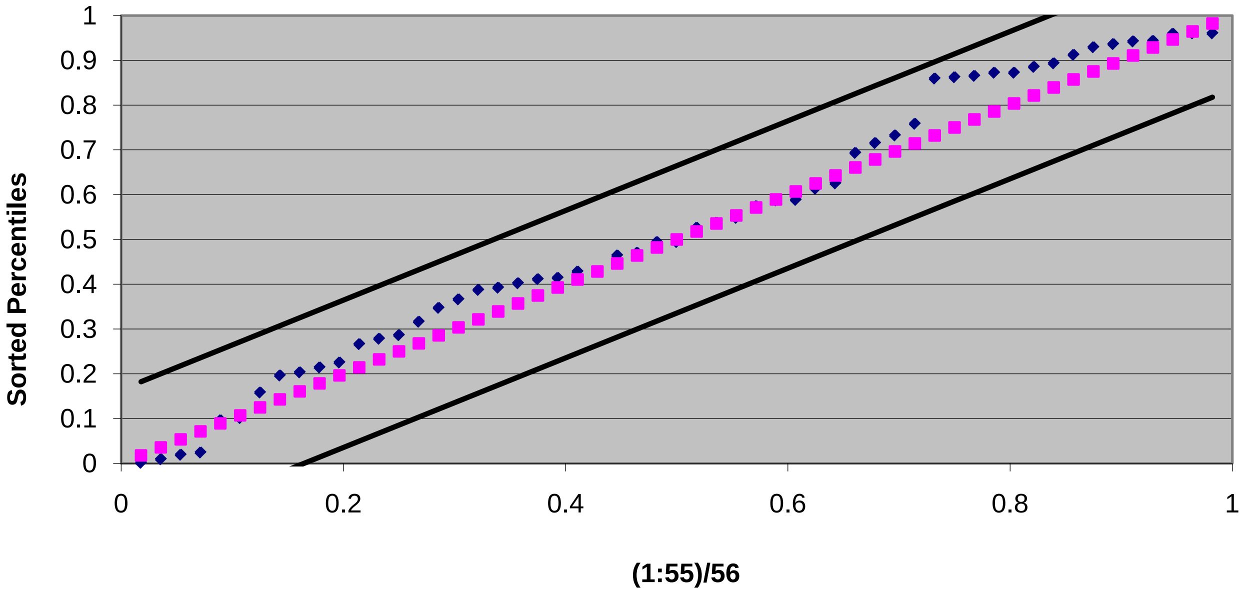
## Candidate 4: Appendix C.4



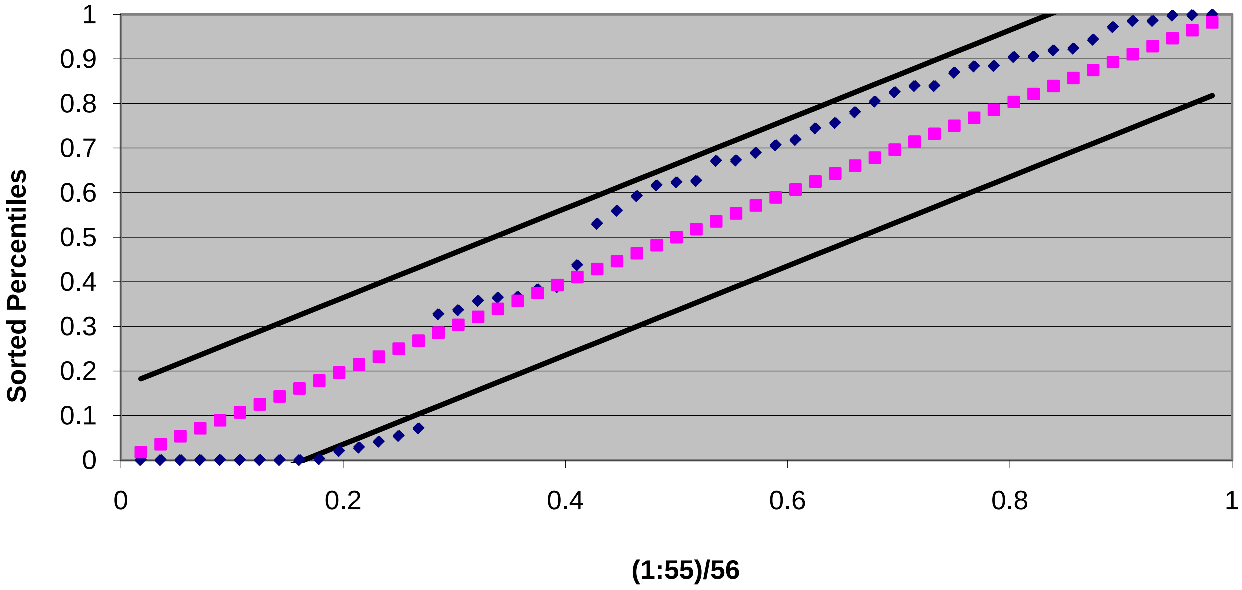
## Candidate 5: Appendix C.5



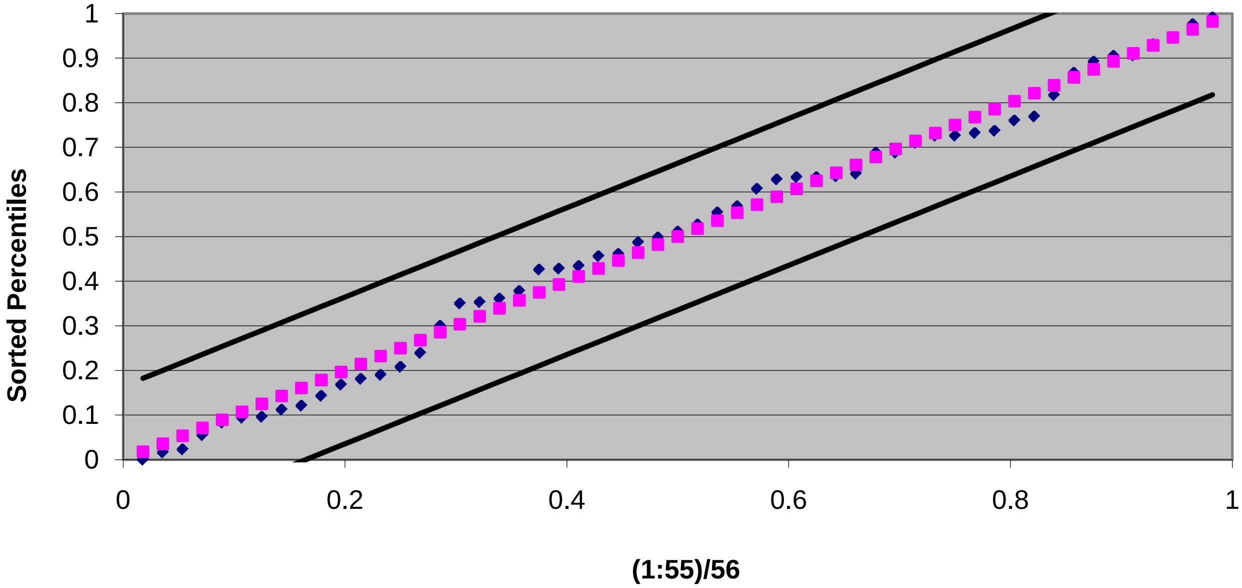
## Candidate 6: Appendix C.6



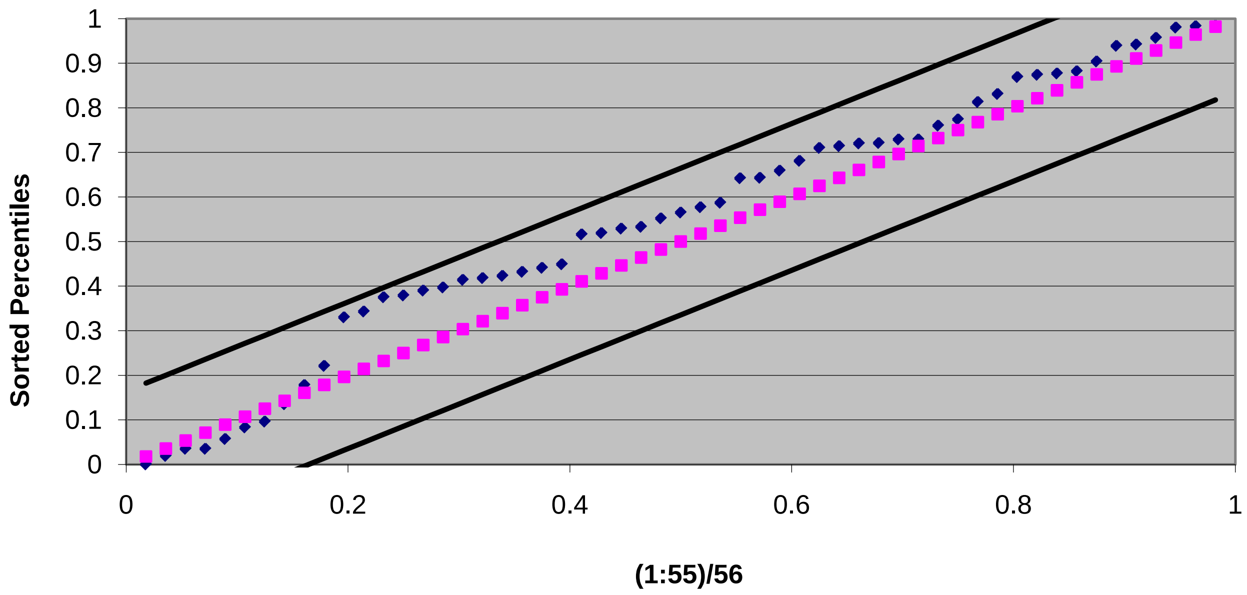
## Candidate 7: Appendix C.7



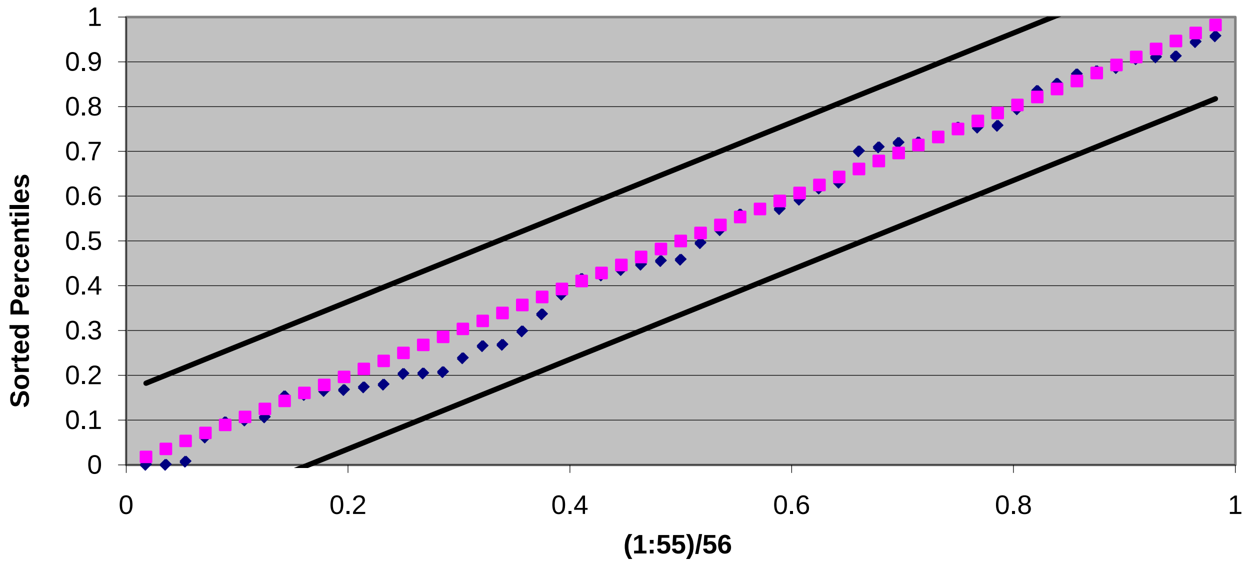
## Candidate 8: Appendix C.8



## Candidate 9:  Appendix C.9



## Candidate 10:  Appendix C.10


Sorted Percentiles
(1:55)/56