**Course SJO750**

**Reliability analysis of marine structures**

**Computer excercises**

**Tutorial-R**

Department of Shipping and Marine Technology

Chalmers University of Technology

**Computer exercise 1 (Tutorial – R)**

Distributions of random variables in Matlab

**In this course, the reliability concepts and corresponding theoretical probability and statistical background are practiced by the Matlab software, which is also used for example to solve problems in the exercises. After you are familiar with these concepts and theories, you are encouraged to use DNV/Proban to solve these problems.**

In Matlab, many toolboxes are available for probability and statistical analysis, which are necessary for the reliability analysis of marine structures. In this course, the WAFO toolbox (<http://www.maths.lth.se/matstat/wafo/>) is adopted to practice your theoretical knowledge got from the course lectures. **Note that the techniques (MATLAB COMMANDS) given in the computer exercises will be needed to solve the projects in the assignments. Hence, please type them in the Matlab terminal and add comments after each useful command by yourself, rather than just copy and paste them. You must understand these commands.**

The first computer exercise is targeted to teach you how to calculate the empirical distribution of a random variable (RV), check the type of distribution for the RV, and estimate probabilities and quantiles of the distribution, based on the observed data (sample). You should also be able to construct the typical distributions used in the maritime industry. Obvious, the basic properties of a distribution, such as mean, variance, etc., are fundamental issues in this course. Finally, you are also encouraged to know how to compute a simple confidence interval and do hypothesis test, which will be further discussed in the next exercise.

# Theoretical knowledge -- preparatory exercise

Read the instructions in lecture 4 and 5, i.e. Chapter 5 in the course compendium.

## Distribution of a random variable *X*

Make sure you understand what the probability and density functions are and how they are related to the cumulative distribution function.

Let *X* denotes a continuous random variable, *X*∈[0, +∞), with a probability density function (pdf) f(x) and cumulative distribution function (cdf) *F(x)*.

* Please write down the relationship between *F(x)* and *f(x)*.
* Please write down the mean and variance of the random variable *X*.

## Empirical distribution of *X*

For a random variable *X* with a sample generated by (note that you will be taught later on how to simulate random variables in the Monte Carlo simulation lectures),

*>>x=normrnd(10,30,1,10000); % generate a sample of a normal R.V with N(10,302).*

* Compute the empirical distribution of X, *F1(x)*, and plot it in a figure.

*>>F1(:,1)=sort(x);*

*>>F1(:,2)=(1:10000)/10000;*

*>>figure(1); clf*

*>>plot(F1(:,1),F1(:,2),'b.')*

*>> hold on*

Alternatively, the calculation could be done by Matlab wafo toolbox command “***empdistr***”,

*>> [F X]=ecdf(x);*

*>>F2 = [X F];*

*>>plot(F2(1:50:end,1),F2(1:50:end,2),’ko’)*

Compare *F1(x)* and *F2(x)* to check the difference in the figure. They are also identical.

* Give the formula for the computation of mean and variance of the random variable *X*.
* Compute the values of mean and variance by direct formulas to get (EX1, VX1) and by Matlab commands to get (EX2, VX2). Compare with the real values in the simulation i.e. E[*X*]=10, V(*X*)=302=900

*>>EX1=sum(x)/10000*

*>>VX1=1/(10000-1)\*sum((x-EX1).^2)*

*>>EX2=mean(x)*

*>>VX2=var(x)*

## Quantiles of X

The concept of quantile is important in reliability analysis and extreme predictions. The quantile of a distribution could be defined in different ways. For the random variable *X*, the quantile as a number of *xα* is defined as,

F(*xα*)=P(X<*xα*)=α.

* From the above plot of the empirical distribution, i.e. Figure (1) in section 1.2, can you estimate the quantiles*x0.05* and *x0.95*.

*x0.05= ; x0.95=*

* Compare with the exact values, given by

*>>x005=norminv(0.05, 10, 30)*

*>>x095=norminv(0.95, 10, 30)*

# Distributions of two or more random variables

Let *X*, *Y* denote two independent random variables, while *X*, *Y*∈[0, +∞). The joint pdf and cdf are represented by *fX,Y(x,y)* and *FX,Y(x,y)*, respectively.

* Please write down the relation between the joint pdf and cdf of *X* and *Y*.
* Please write down the marginal distributions of *X* and *Y*, respectively.
* Please write down the expected value and variance of *X* and Y, respectively.

(Check Wikipedia or course compendium if you do not know the answers, but please first try to solve the problem by yourself)

Now, assume that the two random variables are dependents, let *ρX,Y* denote the correlation coefficient and *Cov(X,Y)* denote covariance between X and Y;

* Please write down the formula for computation of *ρX,Y*.

\rho_{X,Y}=\mathrm{corr}(X,Y)={\mathrm{cov}(X,Y) \over \sigma_X \sigma_Y} ={E[(X-\mu_X)(Y-\mu_Y)] \over \sigma_X\sigma_Y},

* If there are an observed sample for two random variables, i.e. (*xi*,*yi*), *i=1,…,n*. Please write down the formula for the sample correlation coefficient, *rxy* in terms of (*xi*,*yi*).


r_{xy}=\frac{\sum\limits_{i=1}^n (x_i-\bar{x})(y_i-\bar{y})}{(n-1) s_x s_y}
      =\frac{\sum\limits_{i=1}^n (x_i-\bar{x})(y_i-\bar{y})}
            {\sqrt{\sum\limits_{i=1}^n (x_i-\bar{x})^2 \sum\limits_{i=1}^n (y_i-\bar{y})^2}},


* Let ***X***=[*X1, X2, ….,Xn*] denote a random vector. Please write down the correlation matrix and covariance matrix of the random vector ***X***, using *ρXi,Yj*and*Cov(Xi,Yj)*  The correlation matrix is denoted by ***R***, while the covariance matrix is denoted by ***Σ***.
* Note that the correlation matrix ***Σ*** of a random vector, i.e. a series of random variables, could be easily computed by the Matlab routine, *corrcoef*, while *cov* is for covariance matrix. Here is an example.

*>>X1=normrnd(0,1,1000,1);*

*>>X2=normrnd(0,1,1000,1);*

*>>X=X1+0.3\*X2;*

*>>Y=-0.4\*X1+0.1\*X2;*

*>>Z=X1-X2;*

*>>% the random vector is [X Y Z], the correlation and covariance matrixes are*

*>>R=corrcoef([X Y Z])*

*>>SIGMA=cov([X,Y,Z])*

* Note that the two matrixes are symmetric. Please explain why?

# Common used distribution in maritime industry

In the maritime industry, a few distributions are commonly used for design and safety analysis. Without going too much into the details, the normal distribution is definitely the most popular distribution type. When values of observed data are too scattered, the lognormal distribution could be used to describe the data distribution. In the engineering applications, ship responses, for example motions, stress signals etc., are often assumed to be Gaussian for convenience even though it is not the real case. Then the local maxima (minima) of the response are often Rayleigh distributed. Further, the Weibull distribution is often used to describe the stress range distribution for long-term fatigue analysis, while Gumbel distribution to fit the yearly maxima for extreme predictions.

After this course, you are supposed to know the exact forms of some distributions, for example normal and lognormal. For others, you have to know at least how to compute the probability using the Matlab commands. Here the stress signals measured in a 4400TEU container ship will be used in the following exercise. You could download the data from the course website.

# Distributions for practical problems

You could check the Wikipedia for answers, but please remember it in the future.

* Write down the probability density function (pdf) for normal and lognormal distributions, as well as mean and variance of the distributions.
* Draw by hand the probability density function (pdf) and cumulative distribution function (cdf)of the two normal random variables, X ~ N(0, 1), while Y ~N(5, 100).

It is very convenient to build the pdf and cdf of random variables by Matlab commands. In this exercise, random variables of lognormal distribution, Rayleigh distribution, Weibull distribution, and Gumbel distribution will be constructed by Matlab. You need to know at least which parameters and ranges of a random variable are needed in the construction (check Wikipedia/course compendium if you do not know the answers). Here is an example for a Weibull random variable. You should be able to do examples for all other distributions.

*>>% choose parameters, i.e. scale and shape parameters by yourself*

*>> x = linspace(0,6,200);*

*>> pdf1 = wweibpdf(x,1,1); pdf2 = wweibpdf(x,2,2);*

*>> cdf1 = wweibcdf(x,1,1); cdf2 = wweibcdf(x,2,2);*

*>>figure(2)*

*>>plot(x,pdf1,’r’); hold on; plot(x,pdf2,’b’)*

*>>figure(3)*

*>>plot(x,cdf1,’r’); hold on; plot(x,cdf2,’b’)*

## Check the distribution type of measured stress signal

In the data “stress.dat”, the measure frequency is 2.5 Hz, i.e. time interval between two measurement points is 0.4 seconds. Load the data and check the distribution types of the stress signal, the distribution types of the local maxima (minima), the rainflow stress ranges, the assumed “yearly” maxima of the signal, using the probability plotting papers. Several probabilities papers are available in wafo toolbox. They are briefly introduced below (it will be further discussed in the next exercise.

1, normal distribution paper: >>wnormplot

2, Rayleigh distribution paper:>>wraylplot

3, Weibull distribution paper: >>wweibplot

4, Gumbel distribution paper: >>wgumbplot

If, in the plot, the observations seem to line up well along a straight line, it indicates that the chosen distribution for the probability plot indeed might serve as a good model for the observations.

Now we will test the distribution type of the measured stress signals. In this exercise you will also use routines in wafo toolbox, for example to get the local maxima, rainflow cycles etc.

* Load the stress signals and check the distribution of the time series. Please answer which distribution is best suitable for the singal.

*>>stress=load(‘stress.dat’);*

*>>stress=[(1:length(stress))’\*0.4 stress];*

*>>figure(4)*

*>>subplot(221); wnormplot(stress(:,2))*

*>>subplot(222); wraylplot(stress(:,2))*

*>>subplot(223); wweibplot(stress(:,2))*

*>>subplot(224); wgumbplot(stress(:,2))*

* Get the local maxima and minima of the signal, and check the distribution.

*>>tps=dat2tp(stress);*

*>>maxim=tps(2:2:end,2);*

*>>minim=tps(1:2:end,2);*

*>>figure(5)*

*>>subplot(221); wnormplot(maxim);*

*>>subplot(222); wraylplot(maxim);*

*>>subplot(223); wweibplot(maxim);*

*>>subplot(224); wgumbplot(maxim);*

* Get the rainflow stress ranges of the signal and do the same check.

*>>rfc=tp2rfc(tps);*

*>>Srfc=abs(rfc(:,2)-rfc(:,1));*

*>>figure(6)*

*>>subplot(221); wnormplot(Srfc)*

*>>subplot(222); wraylplot(Srfc)*

*>>subplot(223); wweibplot(Srfc)*

*>>subplot(224); wgumbplot(Srfc)*

* Divide the signal into 20 sections. Each section is assumed to be one year’s signal. Find the “yearly maxima”, check the distribution and analyze the results.

*>> N=length(stress(:,2));*

*>>index=linspace(1,N,21);*

*>>ym=[];*

*>>for i=1:20*

*>>ym=[ym; max(stress(index(i):index(i+1),2))];*

*>>end*

*>>figure(7)*

*>>subplot(221); wnormplot(ym)*

*>>subplot(222); wraylplot(ym)*

*>>subplot(223); wweibplot(ym)*

*>>subplot(224); wgumbplot(ym)*

## Distributions for regression and inference test

Standard normal distribution is the basis for the hypothesis test. The other distributions for the test are actually related with the standard normal distributions. Instead of investigating the hypothesis test, you will be asked to compute the 95% confidence intervals of random variables from these distributions.

* Standard normal distribution, X~N(mu,sigma2).

If mean (mu) and standard deviation (sigma) are known, the 95% CI is

*>>CI=[mu+norminv(0.025)\*sigma, mu+norminv(0.975)\*sigma]; %or as follows*

*>>CI=[norminv(0.025,mu,sigma), norminv(0.975,mu,sigma)]*

If for other cases, such as mu/sigma is unknown, please consult to the lecture notes.

* Chi-squared distribution

If *X1*, ..., *Xk* are independent, standard normal random variables, then the sum of their squares,, is distributed to the Chi-squared distribution with *k* degrees of freedom. This is usually denoted asor .

The *α*-quantile of Chi-squared distribution could be computed by “***chi2inv(1-α,k)***”***.***

* Student’s t-distribution

Let *x*1, ..., *xk* be the numbers observed in a sample from a continuously distributed population with expected value *μ*. The sample mean and sample variance are respectively

and.

The resulting *t-value* is: .

The *t*-distribution with *k* − 1 degrees of freedom is the [sampling distribution](http://en.wikipedia.org/wiki/Sampling_distribution) of the t-value when the samples consist of [independent identically distributed](http://en.wikipedia.org/wiki/Independent_identically_distributed) observations from a [***normal*** distribution.](http://en.wikipedia.org/wiki/Normal_distribution) The *α-*quantile of t-distribution could be computed by “***tinv(α,k-1)***”.

* F-distribution

Please find the properties by yourself.

**Computer exercise 2 (Tutorial – R)**

Fitting a distribution by maximum likelihood

This computer exercise will teach you the typical way to fitting a distribution by maximum likelihood method using the current commands from the existed Matlab code. In the example, the analyses steps will follow exact the same as that given in the lecture. More examples with less guidance will also be given for your own practice.

# Plot the data to check which type of distributions

Assume that we have a set of observations *x1*, *x2*, … , *xn*. Before we estimate any parameters, we must convince ourselves that the observations originate from the right family of distributions, e.g. normal, Gumbel, or Weibull. One way to get a rough idea, about which family of distributions may be suitable, is to display the observations in a probability plot. If you suspect that the data originate from, for instance, a normal distribution, then you should make a normal probability plot; if you instead suspect a Gumbel distribution, and then make a Gumbel probability plot. If, in the plot, the observations seem to line up well along a straight line, it indicates that the chosen distribution for the probability plot indeed might serve as a good model for the observations.

Statistics Toolbox provides *normplot* (for normal distribution), *weibplot* (for Weibull distribution); the WAFO toolbox furnishes you with *wgumbplot* (for Gumbel distribution). Acquaint yourself with the above-mentioned commands, for example

*>> dat1=randn(2000,1); % generate normal random variable, referred later on.*

*>>figure(1)*

*>>normplot(dat1)*

*>>weibplot(dat1) % Any error-message?*

*>> dat2=rand(3000,1); % generate Uniform distributed random variable.*

*>>normplot(dat2)*

*>>wgumbplot(dat2)*

*>> dat3=wweibrnd(2,2.3,1,3000); % generate Weibull distributed random variable.*

*>>wweibplot(dat3)*

*>>wgumbplot(dat3)*

Since normal distribution is most common used distribution in the industry, you could always first try if the observed data is normal distributed by *normplot*.

## Measurements of significant wave height *Hs* in the Atlantic Ocean

In oceanography and marine technology, statistical extreme-value theory has been used to a great extent. In design of offshore structures knowledge about “extreme” condition is important.

In the numerical examples above, we used artificial data, simulated from a distribution which we could control. We will now consider real measurements from the Atlantic Ocean. The data set contains so-called significant wave heights.

* Now, load the data set “***atlantic.dat***” and read about the measurements; then find the size of data, and plot it:

*>>atl=load('atlantic.dat');*

*>> help atlantic*

*>>size(atl)*

*>>plot(atl,'.')*

One knows that, roughly speaking, the registered so-called *Hs* behave, statistically, as if they were maximum wave-heights; therefore one can suspect them to originate from a Gumbel distribution, for instance.

* In the following, we will make different probability plots.

*>>normplot(atl)*

*>>normplot(log(atl))*

*>>wgumbplot(atl)*

*>>weibplot(atl)*

* Which distribution might be a satisfactory choice? Make comments to the plots! Consider also the log-normal distribution. (this will be used in the next section)

# Maximum likelihood method to fitting the probability distribution

Since the normal distribution is quite well recognized and systematically studied in the course lectures, in this section, we will use the Gumbel distribution, which is often used for extreme prediction, for an example to illustrate the method of maximum likelihood method. You will also be given some hints to use the method for fitting the Rayleigh and Weibull distribution (often for fatigue problems) by yourself.

Assume that we have a set of observations *x1*, *x2*, … , *xn*, from (for example) a Gumbel distribution, i.e. the cumulative distribution function (cdf) is

.

However, the parameters *μ* and *β* are not known. Then, one can use the maximum-likelihood method (ML method) to estimate the parameters from the sample.

* Write down the likelihood function for the example above.

L(μ; β;*x1*, *x2*, … , *xn*) =

## Estimation of parameters in a distribution function

In this section, you need to calculate the parameters suitable for the sample data, “***atlantic.dat***”. Firstly, please derive the maximum likelihood function and compute the parameters by the maximum likelihood method. Later on, you will see that these steps could be easily carried out by some existed Matlab routines.

* Derive the maximum likelihood estimates of the two parameters
* Please first write down the maximum likelihood function of the distribution,
* Then, the parameters could be estimated by the partially differentiate the maximum likelihood function with respect to different unknown parameters.
* Estimate the parameters.

=

=

## Estimation errors (Mean and variance of the estimated parameters)

In the WAFO toolbox the ML-method has been implemented in “***wgumbfit***”, “***wweibfit***”, and “***wraylfit***” for the purpose of estimating the parameters in a Gumbel, Weibull, and Rayleigh distributions, respectively.

* Please compute the two parameters by the WAFO routine “***wgumbfit***”, and then compare with that got from above direct calculation.

*>> [phat, covm]=wgumbfit(atl);*

The routine “***wgumbfit***” produces a figure, in which the empirical distribution is plotted together with the fitted Gumbel distribution where the parameters μ and β are computed by ML-method. The parameter estimates are given in the vector “***phat***”. The elements *phat(1)* and *phat(2)*, correspond toand, respectively. The fitting will also give the asymptotic variance and covariance (when the number of observations is “large”) viz

covm(1,1)=V(); covm(2,2)=V(); covm(1,2)=covm(2,1)=cov(,).

These values could be used to judge the efficient of the fitting, i.e. the estimation error.

* Compare the estimations by the two methods, which should give identical results.

## Estimation mean and errors of a quantile

The quantile from Gumbel distribution is often used for extreme prediction. Here the estimation of mean and standard deviation of a quantile is introduced.

**Mean of a quantile**

By means of and, estimate the upper 1% quantile, defined as the number *x0.01*, satisfying

 ⇔.

The value of *x0.01* could be obtained by solving

⇔.

This could lead to a reasonable estimate of *x0.01* as follows,

.

The value can be computed by Matlab as

*>>xhat=phat(2)-phat(1)\*log(-log(1-0.01))*

**Standard deviation of the quantile**

Since and, both are random variables, so is  according to above derivation. It is important to know the expectation and standard deviation of . The standard deviation indicates the dispersion of the estimate , and it is therefore important to get an idea of the value of the corresponding standard deviation. In most cases, it is impossible to find an exact value, and consequently, an approximation has to be done. Such an approximation is called a standard error. Based on the properties of variance, the standard error could be computed by

.

where the approximations of Var(), Var() and Cov(,) could be obtained from the above fit, the standard error is computed by

*>> c=-log(-log(1-0.01));*

*>>stderror=sqrt(covm(2,2)+c^2\*covm(1,1)+2\*c\*covm(2,1))*

Of course, just 50 values to estimate *x0.01* might be too small a sample. However, the standard error computed above is the only one we can get so far.

**2.4 (OPTIONAL) Please follow the steps from 2.1 to 2.3, fitting a distribution of Rayleigh distribution. The data is obtained from the measurements of stress local maximA.**

**Computer exercise 3 (Tutorial – R)**

Linear regression analysis

This computer exercise will teach you the typical way of carried out a linear regression analysis, which includes model determination, parameter estimate, diagnostic analysis and confidence interval of estimates. The assignment for this part will contain similar problems as this computer exercise.

# Understand the linear regression method

When dealing with two or more variables, the functional relation between the variables is often of interest. In almost any activities where experiments have been performed, linear regression might be an extremely useful tool for analysis. Even though the relationships between two variables are not strictly linear, it is still convenient to transfer into a linear regression problem. For example, the S-N curve for fatigue analysis is actually an exponential equation, i.e. *N*(*S*)*=αSm*, where *α* and *m* are the parameters needed to be regressed from the experiments, S is the stress range, and *N* is the number of *S* leading to fatigue failure of structures. But in the regression analysis, the logarithm is often taken for both sides of the equation. Then, it becomes log(*N*(*S*))=log*α*+*m*log*S*, a linear regression analysis could be carried out in this case.

Further we should note that linear regression does not mean that the two variables are linear related. For example, the quadratic relation between the dependent and independent variables *x*, *y*, i.e. *y*=*a*+*bx*+*cx*2, is still a linear regression problem. In the course and this computer exercise, we will focus on the linear regression analysis.

In the regression analysis, the parameters in the equation are often computed by the so-called Least Squares (LS) method. Suppose two variables (*X*, *Y*) have a simple linear relation as *Y*=*β0*+*β1X*. Further if the model is a probabilistic model, and a series of data (*xi*, *yi*), *i*=1,…,n (n>2) is available, the LS-method can estimate the parameters as and . In this case, the statistical relation is written as , where *ε* is the random variable denoted the residual from this regression, .

* Please list several major assumptions about the residual random variable for this simple linear regression analysis.

The matlab command “***regress****”* can be used for the regression analysis to get the parameters.

# Correlation and regression

## Correlation determination

Sometimes, we are only interested in understanding if a relationship between two variables is exists. In this case, we can use the correlation analysis. The correlation between two variables can be ranged from “-1≤*ρ*≤1”. It is often performed using the scatterplot to test the correlations.

If the correlation coefficient is close to +1, it means a strong positive relationship.

If the correlation coefficient is close to -1, it means a strong negative relationship.

If the correlation coefficient is close to 0 that means you have no correlation.

Here is an example to using scatter plot to show: Student’s height and weight in a classroom.





Fig. 1, Scatter plot of student’s height and weight to test their correlation

* Please guess the correlation *ρ* of Student’s height and weight as shown in Fig.1.

Upper left plot: *ρ =*

Upper right plot: *ρ =*

Lower left plot: *ρ =*

Lower right plot: *ρ =*

Note that the correlation analysis is only useful to determine the linear correlation.

## Linear regression analysis

If one is interested in describing the relation between two or more variables as a linear function, a regression analysis needed to be implemented. In this case, one should first determine the relation between the variables should be expressed as a deterministic model or probabilistic model. The deterministic model is an equation or set of equations that allow us to fully determine the value of the dependent variable from the values of the independent variables, while the probabilistic model is a method used to capture the randomness that is part of a real-life process. Since this course is designed for the reliability analysis, we will only focus on the probabilistic model.

# Regression analysis

In this section, we will show an example of a tensile test in a laboratory. In the lest, a given load *F* (kN) is applied on the specimen, and the deformation *X* (10-2mm) of the test is also measured. Some measurements have been saved in the file “***testdata.mat***”, which could be downloaded from the course website. Load the data in Matlab, check the variables, and plot the data.

*>>load testdata*

*>>figure(1)*

*>>plot(F,X,’b\*’)*

Judging from the plot, there seems to be (linear) relation: when the load increases, the deformation increase. However, there is also randomness occurring. For a given load (7.9 kN), four items were testes. Obviously, there is variability in the deformation. The measurement error is a source of randomness. In this situation, a naive probabilistic model is used to express the randomness,

*X*=*β0*+*β1F*+*ε*,

where the residuals are represented by ε, a normal random variable with ε∈N(0, σ2). We have 9 pairs of values (*Fi*, *Xi*), *i*=1,…,9. In the following regression analysis, the variable F is often called the independent variable, while X is called the dependent variable.

In the analysis, we firstly need to estimate the parameters in the model, i.e. the slope parameter *β0* and the intercept parameter *β1*. Typically, the goodness-of-fit is tested by the ANOVA. Since the regression is based on some major assumptions with respect to the residuals, i.e.*ε,* it is often needed to check these assumptions after the model fitting. It is also known as the regression diagnostics.

# Estimate parameters (Maximum Likelihood and Least squares)

In order to estimate the parameters *β0* and *β1,* one could use the Maximum Likelihood (ML) method assuming the residual is a Gaussian random variable with ε∈*N*(*0*, *σ2*).

* Please derive the formulas for the ML-estimators of *β0* and *β1*

Alternatively, for this simple linear regression analysis, the Least Squares (LS) estimators of these two parameters are given in the lecture.

* Please write the formulas for this case study.
* You could check for Gaussian ε, the ML- and LS-methods give the same estimators.

Actually, in Statistics Toolbox in Matlab, there is a routine “regress”, which performs the LS estimation. In this case, it could be implemented viz,

*>>b=regress(X, [ones(9,1) F])*

In the results, of *β0=b(1)* and *β1*= *b(2)*. Note that if there is a quadratic relation between F and X, such as *X* = *β0*+*β1F*+*β2F2*+ε. The parameters could be computed by

*>>b=regress(X, [ones(9,1) F F.\*F]) %β0 = b(1), β1 = b(2) and β2 = b(3)*

Plot the regression line in the same figure as the original data

*>>figure(1); hold on*

*>>xest=b(1)+b(2)\*F;*

*>>plot(F,xest,’r.-’)*

## Goodness-of-fit (ANOVA)

Analysis of Variance (ANOVA) consists of calculations that provide information about levels of variability within a regression model and form a basis for tests of significance. The F-test is used for comparisons of the components of the total deviation. For example, in one-way, or single-factor ANOVA, statistical significance is tested for by comparing the F test statistic. It could be done as follows:

In our case, first, assume the relation models between the variables X and Fare

X=*β0*+ε (1)

X=*β0*+*β1F*+ε (2)

The analysis of variance for the regression model (1) and (2) could be test by the goodness-of-fit to check which model is more suitable for the observed data, or directly check if the model (2), including one more parameter, will increase the regression accuracy. The determination of which model is good is done by the *F*-test.

In this test, one needs to compute the sum of squares for the regression model, errors (residuals), and compare with the F-parameters.

*>>SSR=sum((xest-mean(X)).^2);*

*>>SSE=sum((X-xest).^2);*

*>>SST=sum((X-mean(X)).^2);*

*>> % you could check if SST=SSR+SSE*

*>>F\_ratio=(SSR/1)/(SSE/7);*

*>>P=1-fcdf(F\_ratio, 1,7);*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ANOVA with one explatory (independent) variable** | | | | | |
| Source | DF (Deg. of Freedom) | SS  (Sum of Squares) | MS  (Mean Squares) | F-Ratio | P-Value |
| Regression | 1 | SSR=2120 | MSR=2120 | MSR/MSE=549 | 6.5E-08 |
| Errors | 7 | SSE=SS27 | MSE=3.86 |  |  |
| Total | 8 | SST=2147 |  |  |  |
|  |  |  |  |  |  |
| **P-value is the probability that the hypothesis“H0: 1=0” is true.** | | | | | |

In this case, we could reject the hypothesis. Hence, the model *X*=*β0*+*β1F* should be used to describe the relations. The statistical properties of the parameters will be analyzed later on.

## Regression diagnostics (check the residuals)

In this section, we have to check the major assumptions about the residuals in the regression estimations.

* Check the independent

If the sample is large enough, we could assume the residuals are approximately independent. In this example, although there are only 9 data sets available, we still assume that the regression residuals are independent.

* Check the normal distribution

This step could be done using normal probability paper plot.

*>>figure(2)*

*>>normplot(X-xest)*

* Check the constant variance

The scatterplot here is a useful tool to inspect the property. Besides, it could also help us to identify the violation of the assumption of linearity, potential outliers etc. which are not included in this course.

*>>figure(3)*

*>>plot(xest, X-xest,’bo’)*

Since the sample size is not large enough, it is actually not so easy to check if the variance is constant. But you will need to check that in the course assignment.

* Check the correlation between the residuals and independent variables

Again you could use the scatter plot of residuals against the independent variables.

*>>figure(4)*

*>>plot(F,X-xest,’r\*’)*

**Computer exercise 4 (Tutorial – R)**

Monte Carlo methods and simple applications

Monte Carlo methods are a class of computational algorithms that rely on repeated random sampling to compute their results. Monte Carlo methods are often used in computer simulations of physical and mathematical systems. These methods are most suited to calculation by a computer and tend to be used when it is infeasible to compute an exact result with a deterministic algorithm. This method is also used to complement the theoretical derivations.

In this computer exercise, you will learn how to generate/simulate independent random variables with given distribution. Further, you will also be realized that Monte Carlo methods could help you to solve problems in a very convenient way.

# Simulation of independent random variables

In the simulation of random variables, one should keep in mind that the value of cumulative probability *F* is ranged from 0 to 1. It could be recognized that the probability is actually uniformly distributed in [0, 1]. The Matlab routine “***rand***” can be used to generate the uniformly distributed random numbers from 0 to 1.

In order to generate a random variable with more general distribution type denote by *F*. Firstly, let *Y* be a uniformly distributed random variable (between 0 and 1). It could be generated by “***Y=rand(100,1)***” for example.

Then a random variable *X* is said to be distributed according to *F.* That is *Y=F(x)*. If the distribution *F(x)* is an invertible function, then the random variable *X* could be computed by the inverse function:

*X*=*F-1*(*Y*).

If, for instance, *F* is

Weibull distribution:⇔

Normal distribution: ⇔

Gumbel distribution: ⇔,

Then *X* is a Weibull, Normal or Gumbel distributed random variable, respectively.

First, we will use the above theory to generate 100 Weibull-distributed numbers:

*>>lambda=2; theta=0; k=3.6;*

*>>x=theta+lambda\*(-log(1-rand(100,1))).^(1/k);*

*>>figure(1)*

*>>plot(x,’b.’); hold on*

Or we could generate 100 normally-distributed numbers viz.

*>>mu=10; sigma=3;*

*>>x=mu+sigma\*norminv(rand(100,1));*

*>>figure(1); hold on*

*>>plot(x,’bo’)*

Eventually, generate 100 Gumbel-distributed numbers:

*>>mu=3.6; beta=2;*

*>>x=mu-beta\*log(-log(rand(100,1)));*

*>>figure(1); hold on*

*>>plot(x,’r\*’); grib on*

* What do the plots look like? Please make comments.

Note that this type of plot may indicate the “average” value and spreading, but in the first computer exercise, we have shown how to use the graphical plot to illustrate the distribution type of the data. For example, in the WAFO toolbox, one can use “***wweibplot***” to check the properties of Weibull distributed random numbers, while “***wnormplot***” and “***wgumbplot***” are corresponding to the Normal and Gumbel distributions.

* Please use the commands to plot the above generated random numbers.

# How to simulate independent R.Vs by Matlab intrinsic routines

Due to the common use of the Monte Carlo methods for the practical problems, the generations of random numbers from typical distribution have already been implemented in the Matlab Statistics Toolbox, and also WAFO toolbox. For example, to generate specific distributed numbers in Matlab, one can also make use of the commands “***wblrnd***” (Weibull distribution), “***normrnd***”(normal distribution), “***raylrnd***” (Rayleigh distribution) etc. In the WAFO toolbox, they can be implemented by “***wweibrnd***” (Weibull), ”***wnormrnd***” (Normal), ”***wgumbrnd***” (Gumbel), and “***wraylrnd***” (Rayleigh).

In the following, we will compare the generation of random numbers of different distributions by means of the above two method. You should note that it is only meaningful to compare the statistical properties of generated random variables. For example, when one wants to compare ship responses (motions or stress signals) under a sea state computed by a numerical code, with that obtained from the measurements, it does not prove anything if one only plots the times series of motions (stress) from the two sources and see the difference. Since the sea state is random as the input of the computation code, so are the outputs of the computation, i.e. stress or motions. In this case, you could compare the mean value and standard deviation, or even the maximum value of ship responses computed by numerical code with the measurement, in order to validate the computation. In the following case study, the Weibull distribution will be uses.

* Generations from two different approaches and compare the two generations.

*>>% generated random numbers are denoted as x1, while x2 for matlab direct generation*

*>>size=100;*

*>>lambda=2; theta=0; k=3.6;*

*>>x1=theta+lambda\*(-log(1-rand(size,1))).^(1/k);*

*>>x2=wweibrnd(lambda,k,size,1);*

*>>figure(2)*

*>>plot(x1,’b.’); hold on*

*>>plot(x2,’ro’)*

*>> % compare the mean, variance of the generated random numbers*

*>>[mean(x1) mean(x2) var(x1) var(x2) max(x1) max(x2)]*

Please give comments on the comparisons of the two types

* Increase the simulation size of the random numbers to check the convergence.

*>>size=100:500:10000; N=length(size);*

*>>stat=[];*

*>>lambda=2; theta=0; k=3.6;*

*>>for i=1:N*

*>>x1=theta+lambda\*(-log(1-rand(size(i),1))).^(1/k);*

*>>x2=wweibrnd(lambda,k,size(i),1);*

*>>stat=[stat; mean(x1) mean(x2) var(x1) var(x2)];*

*>>end*

*>>figure(3) % compare the mean value with the real mean*

*>>plot(size,stat(:,1),’b.’); hold on*

*>>plot(size,stat(:,2),’bo’);*

*>>plot(size,lambda\*gamma(1+1/k)\*ones(N,1),’k\*’) % real mean value*

*>>figure(4) % compare the variance with the real variance*

*>>plot(size,stat(:,3),’r.’); hold on*

*>>plot(size,stat(:,4),’ro’);*

*>>plot(size, (lambda^2\*gamma(1+2/k)-(lambda\*gamma(1+1/k))^2)\*ones(N,1),’k\*’)*

Please give your comments on the comparison.

* Please try the same steps with Gumbel or Rayleigh distributions.

## Other simulation techniques (resampling)

In principle, Monte Carlo methods are used for sampling random variables. Most often, the statistical properties of the random variables, for example, the distribution, are known. Sometimes, when we have a sample set for the random variable, without knowing the detailed distribution, the resampling techniques could be used for the simulation analysis. It is extremely useful to compute the mean, variance and confidence interval of the random variable. The following three methods are often adopted for the so-called resampling analysis.

* Jackknife method (mean and variance computation)
* Bootstraps (parametric and nonparametric)
* Importance simulations

A further understanding of these methods could be referred to the course website:

<http://www.scss.tcd.ie/Rozenn.Dahyot/>.

# Applications and examples

When using Monte Carlo methods to solve practical problems, it is always recommended to check the convergence of the solutions. In the following, we will introduce 3 different problems which could be easily solved by the Monte Carlo method.

## Approximating the value of π

In this procedure the domain of inputs is the square that circumscribes our circle. We generate random inputs by scattering grains over the square then perform a computation on each input (test whether it falls within the circle).

* The area of a circle with radius one is Ac=π\*r2=π.
* The area of a unit square is: As=2\*2=4
* The ratio between the two areas is R=Ac/As=π/4⇔π=4\*R.
* If we generate random points uniformly distributed (scattered) in [-1, 1] for both *x* and *y*. Then the ratio of points fall into the circle and points into the square is equal to R.

The Matlab could be written as follows:

*>> %first to draw the circle and square*

*>>theta=0:0:0.1:360;*

*>>x=sind(theta); y=cosd(theta);*

*>>figure(5);*

*>>plot(x,y); hold on*

*>>plot([-1 -1],[-1 1],'k-')*

*>>plot([-1 1],[-1 -1],'k-')*

*>>plot([1 1],[-1 1],'k-')*

*>>plot([-1 1],[1 1],'k-')*

*>>size=100;*

*>>x1=-1+2\*rand(size,1);*

*>> y1=-1+2\*rand(size,1);*

*>>plot(x1,y1,’b.’);*

*>>inds=find(x1.^2+y1.^2<1);*

*>>R=length(inds)/size;*

*>>PI=4\*R*

You could increase the size to check the convergence and compare with the real value of R.



Fig. 1, areas of a circle and a square (left) together with simulated random points

## Approximation of Integration

You could easily give the analytical solution of the integration. But when the integration become a bit complex, for example, , it might be extremely hard to get the analytical solution. If we assume the x is uniformly distributed in [0,1] with sample size N, i.e. *xi*, i=1,…,N, the integration could be approximated by

.

In the following, we will learn how to implement the two examples in Matlab. If you know Mathematics, you could use it to check the results from the Monte Carlo method.

*>> % first for the simple integration, you could check by the analytical solution*

*>>size=1000;*

*>>x=rand(size,1);*

*>>y1=sum(x.^2)/size*

*>>%check the denominator, x ca not be zero*

*>>inds=find(x==0); x(inds)=0.0001;*

*>>y2=sum(cos(x.^2)./log(x))/size*

Again, please increase the convergence by increase the sample size

## Mean and variance of the complex function of a random variable

Suppose there is a random variable X and its function of f(X), one is interested to compute the mean and variance of the function, i.e. E[f(x)] and Var(f(X)). For example, in maritime industry, such a problem is described in the assignment for this lecture. When the function is not so complex, you could have analytical solution. Alternatively, a convenient way is to use the so-called Gauss approximation shown below:



Perhaps, the most convenient and straightforward way is to use the Monte Carlo method. In this example, let X be a Normal distributed random variable with X~N(10, 25), we want to compute the mean and variance of.

Note that you could find the analytical solution for this case in the website:

<http://en.wikipedia.org/wiki/Normal_distribution>.

*>>size=1000; mu=10; sigma=5;*

*>>x=normrnd(mu,sigma,size,1);*

*>>y=x.^2.5+x.^2;*

*>>[mean(y) var(y)]*

So you can see it is so simple.

**Computer exercise 5 (Tutorial – R)**

Monte Carlo methods and advanced applications

In the real practice, two/several random variables are often correlated. Hence, in this computer exercise, you will be asked to generate correlated random variables, together with some complex problems using the generated correlated random variables. In order to generate multiple corrected random variables, the covariance matrix of these random variables is necessary. The properties of the covariance matrix denoted by ***Σ*** are listed as follows:

* it is a symmetric matrix such that ***Σ***=***Σ***T
* the diagonal elements (variance of these random variables) satisfy **Σ**i,i≥0
* it is positive semi-definite so that **x**T**Σx**≥0 for all **x**∈**R**.

In general, two methods are often used to simulate the correlated random variables, provided the covariance matrix of these correlated random variables. One is called the Cholesky decomposition method, and the other is often related with the Eigen system transformation. In principle, both methods are used to solve the following matrix equation with respect to the matrix ***C***,

=Σ.

where the  is used to multiply with the iid R.Vs (independent and identically distributed Random Variables) to get the correlated random variables. Because of the simple application of the Cholesky Decomposition, it is often used to get the matrix **C**. In Matlab, you could use the routine “***chol***” for such analysis.

Note that since the sum of iid normal R.Vs is still a normal random variable, whose mean and variance are also easy to be obtained, we focus on the generation of correlated normal random variables using the covariance decomposition and iid Normal random variables.

# Inputs for the simulations

In order to simulate a vector of correlated ***normal*** random variables ***Z***=[Z1, …, Zn]T, a full process is described as follows (Note in this exercise, we only give the most commonly used techniques or solutions. You may encounter other types of solutions in literatures. ):

1, generate *n* iid ***Normal*** random variables **X**=[*X1*, …, *Xn*]

2, compute the matrix ***C*** using either Cholesky or Eigen decomposition method

3, compute the correlated R.Vs by: ***Z***=***C***T***X***. After this step, you will get the correlated R.V with correct covariance matrix, but not mean values.

4, shift the mean values of the *n* correlated R.V. vector ***Z*** based on their properties.

5, validation and check the covariance and mean values of the vector ***Z***.

Obviously, there are some information needed to carry out the analysis as the above 5 steps. First of all, the distribution type of random vector ***Z*** is needed. In principle, the Monte Carlo simulation introduced here assumes ***Z*** or ln(***Z***) is normal distributed, because it is required that the generated random vector ***Z*** , i.e. sum of iid RVs ***X*,** should have the same distribution type as ***X***. In this case, the normal random variable can be easily applied for the simulation. The complex transformation is needed for other types of distributions, and will not discussed in this course. The validation of the fact that ***X*** and ***Z*** are all normal distributed can be done by:

*>>% 1, generate two iid normal distributed R.V*

*>>X1=normrnd(0,1,10000,1);*

*>>X2=normrnd(0,1,10000,1); % rewrite this command instead of use X2=X1, why?*

*>>Z=X1+X2;*

*>>figure(1)*

*>>wnormplot(X1);*

*>>wnormplot(X2);*

*>>wnormplot(Y);*

* What about the sum of independent Weibull distributed random variable? Please check using similar code as above and give comments.

Another mandatory input for the generation is the covariance/correlation matrix of the correlated random vector ***Z***. However, it is strongly recommended to use the covariance matrix instead of correlation for this simulation, since using the correlation matrix you need to adjust the variance of the random variables. The covariance matrix should satisfy the properties given previously.

Finally, the mean values of the random vector ***Z*** are also needed.

The Monte Carlo simulation will give the predefined covariance and mean of the correlated random variables. This is extremely useful for the random variables of normally (lognormally) distributed, since the two distributions are defined by the two parameters. For other types of distribution, for example, Rayleigh, Weibull distributions etc., very complex transformation are needed for the simulations. Hence, we will not discuss it in this course.

# Two methods to generate correlated Gaussian random variables

In this exercise, we will practice to use the two methods, i.e. Cholesky and Eigen decomposition, to simulate correlated random variables. Again, only the Normal random variables are discusses.

## Correlated Normal random variables

Suppose that we want to generate correlated random variables ***Z***=[*Z1, Z2, Z3*]T, whose covariance matrix ***Σ***=[1.0 0.5 0.5; 0.5 2.0 0.3; 0.5 0.3 1.5] and zero mean for all variables. It could be implemented by the codes viz

*>>sigma=[1.0 0.5 0.5; 0.5 2.0 0.3; 0.5 0.3 1.5];*

*>>size=1000;*

*>>C1=chol(sigma);*

*>>[V D]=eig(sigma);*

*>>C2=(V\*sqrt(D))';*

*>>C2=(V\*sqrt(D))';*

*>>X1=normrnd(0,1,1,size);*

*>>X2=normrnd(0,1,1,size);*

*>>X3=normrnd(0,1,1,size);*

*>>X=[X1; X2; X3];*

*>>Z1=C1'\*X;*

*>>Z2=C2'\*X;*

*>>% compare the covariance of simulated R.V.s by the two methods*

*>>cov(Z1')*

*>>cov(Z2')*

* Please compare the matrix ***C1*** and ***C2***, and also the covariance from the two methods with the given “sigma”. Give comments.
* You could also increase the simulation size to check the convergence problem.

## Non-zero mean Gaussian random vector

In the above simulation, the mean values of the correlated random variables are zero. If the mean values of the random variables are not zero, the most convenient way is to directly add the mean values to the generated random numbers. If the following, you will check if this will change the covariance matrix of the correlated random variables.

The covariance matrix is the same as used in section a, while the mean values of the random variables are E[***Z***]=[2 3 4]T.

*>>sigma=[1.0 0.5 0.5; 0.5 2.0 0.3; 0.5 0.3 1.5];*

*>>size=1000;*

*>>C=chol(sigma);*

*>>X1=normrnd(0,1,1,size);*

*>>X2=normrnd(0,1,1, size);*

*>>X3=normrnd(0,1,1,size);*

*>>X=[X1; X2; X3];*

*>>Z=C'\*X;*

*>>Z=Z+[2 ; 3; 4]\*ones(1,size);*

*>>mean(Z,2)*

*>>cov(Z’)*

* Please check the mean and covariance of simulated random numbers with the given values. Give your comments.

# Correlated log-normal random vector

 If *X* is a Normal random variable, then *Y* = [exp](http://en.wikipedia.org/wiki/Exponential_function)(*X*) has a log-normal distribution; likewise, if *Y* is log-normally distributed, then *X* = ln(*Y*) is normally distributed. This fact could be used to generate the correlated log-normally distributed random vector. For the simulation of correlated log-normal random vector ***Y=***[*Y1, Y2,…, Yn*]T (*Yi* are correlated), a convenient way is to use the covariance and mean of the logarithm of ***Y***, i.e. ***Z***=ln(***Y***).

For example, let ***Z***=ln***Y***=ln([Y1, Y2, Y3]T). Here ***Y*** is the correlated log-normally distributed random vector. In this study, only the Cholesky decomposition is used for the simulation, but you can try the Eigen transformation by yourself. The covariance of ***Z*** is the same as above, i.e. **Σ**=[1.0 0.5 0.5; 0.5 2.0 0.3; 0.5 0.3 1.5]. This exercise is divided into two parts, i.e. the random variables ***Z*** with zero or nonzero means.

1, Zero means of ***Z***

*>>clear all; close all; clc*

*>>sigma=[1.0 0.5 0.5; 0.5 2.0 0.3; 0.5 0.3 1.5];*

*>>size=1000;*

*>>C=chol(sigma);*

*>>X1=normrnd(0,1,1,size);*

*>>X2=normrnd(0,1,1, size);*

*>>X3=normrnd(0,1,1,size);*

*>>X=[X1; X2; X3];*

*>>Z=C'\*X;*

*>>Y=exp(Z);*

*>>cov(Y’)*

*>>mean(Y’,1)*

2, ***Z*** has the mean values of [2; 3; 4]

*>>Z=Z+[2 ; 3; 4]\*ones(1,size);*

*>>Y=exp(Z);*

*>>cov(Y’)*

*>>mean(Y’,1)*

* Please notice the changes of mean and covariance after taking the exponentials.

# Examples and Applications

## Expected and variance of random variables functions

If the random variable *X* is normally distributed, given mean and variance, and let *f(x)* denote a function of the random variable, it is shown in the last exercise that the most convenient way to compute the mean and variance of *f(x)* is using the Monte Carlo methods when the function *f(x)* becomes a bit complex. Further, if *Y*=*f*(*X1, X2*) where *X1*, *X2* are correlated random variables, how to compute the mean and variance of *Y*. In this case, even though the function *f* is not so complex, it is still not an easy task to give analytical results for the computation.

Now suppose that *Z1*, *Z2* are correlated normal random variables. The covariance matrix is ***Σ***=[3.5 1.3; 1.3; 5], and the mean values are E[*Z1*, *Z2*]=[1.5 5]. Compute the mean and variance of the function *f*(*Z1*, *Z2*)=*Z1*2+*Z1Z2*+*Z2*1.5.

*>>sigma=[3.5 1.3; 1.3 5];*

*>>mea=[1.5; 5];*

*>>size=10000;*

*>>C=chol(sigma);*

*>>X1=normrnd(0,1,1,size);*

*>>X2=normrnd(0,1,1,size);*

*>>X=[X1; X2];*

*>>Z=C’\*X;*

*>> Z=Z+mea\*ones(1,size);*

*>>Z1=Z(1,:); Z2=Z(2,:);*

*>>f=Z1.^2+Z1.\*Z2+Z2.^2;*

*>>[mean(f) var(f)]*

## Confidence interval of (function of) correlated random variables

If a random variable is normally distributed, the confidence interval can be computed using only the mean and variance of the random variable. However, when the random variable is not normally distributed, or even as above a complex function of several correlated normally distributed random variables, the computation of confidence interval should be computed using the original definition, i.e. P(*xα/2*≤*X*≤*x1-α/2*)=*1-α*. For example, the 95% confidence interval of the above function of two correlated random variables is,

*>>f=sort(f);*

*>>x025=f(size\*0.025);*

*>>x975=f(size\*0.975);*

*>>CI1=[x025, x975]*

* Please compare the above result with that computed assuming *f* is a normal distribution. Give your comments.

*>>figure*

*>>wnormplot(f);*

*>>CI2=[mean(f)-1.96\*std(f) mean(f)+1.96\*std(f)]*