Numerical methods for the VaR computation

Mohamed Ben Alaya

January 13th, 2014

1 Elementary Introduction to Scilab

Scilab (contraction of Scientific Laboratory) is a free software developed jointly by INRIA and ENPC. You can download it from

http ://scilabsoft.inria.fr/

The purpose of these notes is to help you begin to use Scilab. You should worked at the computer and freely experiment with examples. It is adapted for numerical computation that allows you to quickly perform all resolutions and graphical representations commonly encountered in applied mathematics. Nevertheless, Scilab is not a computer algebra system, such as Maple, it can calculate

$$\det \begin{pmatrix} 1 & 3 \\ -4 & 2 \end{pmatrix}$$

but not

$$\det \begin{pmatrix} a & 3 \\ -4 & 2 \end{pmatrix}$$

Scilab works like MATLAB with essentially one kind of object a rectangular numerical matrix with possibly complex entries. All variables are represented by matrices : scalars are matrices, the row vectors are matrices, column vectors are matrices.

The Scilab instructions can be directly typed in the Scilab window to see how they are executed. However, they may also be written in a file `filename.sce`, then executed by the instruction `exec("filename.sce","c")`.

Every line starting with // is a line of comments and is not executed by Scilab. You can also put the comment in the same line of an instruction after the symbol //. Successive instructions are separated by carriage return.

1.1 Scalars, vectors and matrices

```
A=[1,2,3;4,5,6;7,8,9]  // define a matrix with 3 rows and 3 columns
```

The variable A is a matrice 3×3. A matrice with only one row or one column is a vector and a 1×1 matrice is a scalar. The elements of a matrix can be referenced by their indices. Semicolons may also be used at the end of an instruction (in a file or in Scilab console). In this case it means that the result(s) is(are) not displayed. Conversely use comma (,) to get the display.

```
x=[-1.3 sqrt(3) 3*4/5];
x(5)=abs(x(-1));
x
```

Note that the dimension of x has been automatically adjusted. In Scilab, objects are never declared or allocated explicitly ; they have a dynamic type and their sizes can dynamically change according to applied operators or functions. We have to use a correct dimension for our instruction.

```
 r=[10,11,12];
 B=[A;r]
y=[x,r]
```

We can extract a submatrix from a given matrix.

```
A(:,3) // extracts the third column of A
A(2,:) // extracts the second row of A
A(:,2:3) // extracts a submatrix using columns from 2 to 3.
```

Scilab contains usual operations for matrices. The operation $A + B$ stands for the addition of matrices A and B, $A - B$ the subtraction, $A * B$ the multiplication, A/B the division, A' the transpose of A and A^n the power n of A. Besides, add a scalar λ to a matrix A returns a matrix with the scalar λ is added to all the terms of A. Similarly, multiply A by λ returns a matrix with terms of A are multiplied by the scalar λ.

```
x=[1 2 3 4];
x+1
2*x
sum(x)
```

We can apply a function to a matrix.

```
x=[1 4 9 16];
sqrt(x)
```

Operators whose name starts with the dot symbol "." generally stand for term-by-term operations. For example, $C = A. * B$ is the term-by-term multiplication of matrix A and matrix B, which is the matrix C with entries $C(i, j) = A(i, j) * B(i, j)$. Similarly, $C = A./B$ is the term-by-term division of matrix A by matrix B .

```
A
A./A
A.*A
sqrt(ans)
```

By default the result is assigned to the variable `ans` ("answer") which contains the last computation non assigned.

1.2 Some useful commands

In a session all variables are global and stored.

```
x=ones(1,100);          // $x$ is not  displayed
x                       // the vector $x$ is well defined
```

To avoid confusion with variable names you can use commands `who` and `whos`. In fact, using Scilab commands `who` and `whos`, you can get the name and size of dynamic objects present in the current session and you can use the command `clear` to destroy some variables and free the corresponding memory.

```
a=[1,2]; A=[1,2;3,4];  // $a$ and $A$ are assigned
1+1                    // the default variable \verb+ans+ is assigned
who                    // you get all variables
whos()                 // you get more détails
clear a
who                    // the variable $x$ has disappeared
clear
who                    // all variables $a$, $A$ and  \verb+ans+ have disappeared
xbasc()                // clear the current window and the associated recorded graphics
```

Available memory for a Scilab session can be controlled through the use of the `stacksize` command which return the available memory (number of double) and maximum number of variables. We can increase Scilab available memory using the command `stacksize(n)` where n is an integer. For more details, use `help`. To access the online help, click on `help` in the menu bar, or type in the console `help`. See also the command `apropos(key)` it looks for Scilab help files containing keywords `key` in their short description section.

```
help help
help apropos
apropos matrix         // help files containing keywords matrix
help matrix            // help for function matrix
```

The " :" operator allows to define vectors of numbers whose coordinates are in arithmetic sequence. We give the "beginning value" :step : "ending value". It is possible that "ending value" is not reached. If the step is not mentioned, its default value is 1.

```
3:10 // define a row vector of integers
      //which increments in steps of $1$ from $3$ and $10$
1:2:10// increments in steps of 2 from 1 to 10
```

Two iterative control structures exist in Scilab. They are the `for` loop and the `while` iterator.

```
for k=1:6 // increments in steps of 1 from 1 to 6
x(k)=2*k
x(k)=x(k)+1
end
```

The `while` iterator is especially useful when it is not known ahead of time how many iterations are to be performed. It has the following syntax :

```
y=1 ;
while y<14
y=2*y
end
```

The `plot2d` command generates two-dimensional graphs. Given vectors $x = [x1, ..., xn]$ and $y = [y1, ..., yn]$, `plot2d(x,y)` will plot the points $(x1, y1), ..., (xn, yn)$. There are several options for connecting the points, which can be specified using optional calling arguments. The default in `plot2d` is to connect them with a straight line. We can also choose the color.

```
// graphiques
r=rand(1,100)
plot2d(r)
xbasc()
plot2d(1:100,r,-5)
s=rand(1,100);
xbasc()
plot2d(r,s,-2)
```

To plot an histogram we use the command `histplot` for example the following program generates a random vector with a uniform distribution and plots this vector as an histogram.

```
// histogram
r=rand(1,100)
xbasc()
histplot(10,r)
xbasc()
histplot(0:0.2:1,r)
```

1.3 Scilab function

One may define a Scilab function interactively. However, it is more practical and flexible to write the Scilab code in a file with a text editor. One can mix Scilab instructions and functions definitions in a same file. The beginning of a Scilab function is function [<arguments de retour>]=<nom>(<arguments d'ent r ´ee>) and the code ends with the keyword endfunction.

```
function [x] = s_normal(s,mu,sigma)
// simulation of normal random number
// s = number of simulations
//--------------------
x=[];
y=rand(2,s);
for i=1:s do ;
```

```
x=[x,mu + sigma*sqrt(-2*log(y(1,i)))*cos(2*%pi*y(2,i))];
end;
x=x';
```

By convention Scilab script file names are terminated with the suffix `sce` but if the script contains only function definitions then the `sci` is used. In that case the command `getf` can be used instead of `exec` to load the functions. The file extensions Une fonction est chargée en tapant sous Scilab `sci` and `sce` are just conventions since Scilab functions `exec` and `getf` do not check file name suffixes. `getf("nom_du_fichier.sce","c")`.

```
// load the  function s_normal
getf("filename.sce","c")
// use of the function s_normal
s_normal(100,-1,5)
```

2 Simulation

From a mathematical point of view, we suppose that the function `rand(m,n)` generates a $m \times n$ matrix with independent identically distributed (i.i.d.) random variables with uniform distribution on the interval $[0, 1]$. More precisely, we suppose that the coefficients of the generated matrix are the realisation of these random variabales.

2.1 Simulation of a discrete distribution

We consider a random variable with a distribution supported by a finite set E

$$E = \{x_i, \quad i = 1 \cdots n\} \quad \text{et} \quad \mathbb{P}(X = x_i) = p_i, \quad i = 1 \cdots n.$$

Let U be a random variable with uniform distribution on the interval $[0, 1]$. It is easy to prove that the random variable Y defined below has the same distribution as X :

$$Y = \sum_{i=1}^{n} x_i \mathbf{1}_{[m_{i-1}, m_i[}(U) \quad \text{avec} \quad m_i = \sum_{j=1}^{i} p_j \quad i = 1 \cdots n \quad \text{et} \quad m_0 = 0.$$

- Simulate a sample of size n with a Bernoulli distribution for different parameter p.
- Simulate a sample of size n with an uniform distribution on $\{1, 2, 3, 4, 5, 6\}$
- Simulate a sample of size n with a binomial distribution for different parameter n and p.

Simulation of the binomial distribution $B(n, p)$ One can use properties of the distribution of a random variable X. Let X be a binomial random variable with parameters p and n

$$\mathbb{P}(X = k) = C_n^k p^k (1 - p)^{n-k}.$$

Let $U_i \quad i = 1, \cdots, n$ be n independent random variables with uniform distribution on the interval $[0, 1]$. If we put

$$Y = \sum_{i=1}^{n} \mathbf{1}_{\{U_i < p\}},$$

then the random variable Y represents the number of successes in a sequence of n independent yes/no experiments, each of which yields success with probability p. Hence Y and X have the same distribution and we can use this representation to simulate a binomial distribution.

2.2 Simulation of a distribution with a probability density f

Let X be a random variable with distribution function F, $F(x) = \mathbb{P}(X \leq x)$. We suppose F continuous and strictly increasing which is the case when $f = F'$ is a positive probability density function, $f > 0$. Hence F is invertible and if we put $Y = F^{-1}(U)$ where F^{-1} is the inverse of F then the random variable Y has the same distribution as X.

- Simulate a sample of size n with a Cauchy distribution for parameter 1.
- Simulate a sample of size n with an exponentiel distribution for parameter 1.
- Simulate a sample of size n with a probability density function $3x^2\mathbf{1}_{[0,1]}$.

Simulation of an exponentiel distribution with parameter $\lambda > 0$ The probability density function of an exponential distribution is $\lambda e^{-\lambda x}\mathbf{1}_{x\geq 0}$. Let U be a random variable with uniform distribution on the interval $[0, 1]$ and put $X = -\dfrac{1}{\lambda}\log(U)$ then show that X has un an exponentiel distribution with parameter λ.

2.3 Simulation of a Gaussian distribution

Our aim now is to simulate a random variable with standard Gaussian distribution called also standard normal distribution and denoted by $\mathcal{N}(0, 1)$. The probability density function of a standard Gaussian distribution is $\frac{1}{\sqrt{2\pi}}e^{-x^2/2}$. One relies on the Box-Müller method, which is probably the most efficient method to simulate couples of independent bi-variate normal distributions.

1. Let R and θ be two independent random variables and put $X = R\cos(\theta)$ and $Y = R\sin(\theta)$. Show that if R^2 has an exponentiel distribution with parameter $\frac{1}{2}$ and θ has a uniform distribution on the interval $[0, 2\pi]$ then the couple (X, Y) has a probability density function $\frac{1}{2\pi}e^{-\frac{x^2+y^2}{2}}$.
2. Let (U_1, U_2) be a couple of independent random variables with a uniform distribution on the interval $[0, 1]$ and deduce that $(\sqrt{-2\log(U_1)}\cos(2\pi U_2), \sqrt{-2\log(U_1)}\sin(2\pi U_2))$ is a couple of independent random variable with standard Gaussian distribution $\mathcal{N}(0, 1)$.

Our aim now is to simulate a correlated Gaussian random variables, more precisely we will simulate a centred Gaussian couple with covariance matrix $\begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$. Let $\begin{pmatrix} G_1 \\ G_2 \end{pmatrix}$ be $\mathbb{R}^2$-valued satndard $\mathcal{N}\left(0, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right)$ Gaussian vector and show that $\begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{pmatrix}\begin{pmatrix} G_1 \\ G_2 \end{pmatrix}$ has $\mathcal{N}\left(0, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}\right)$ distribution. Simulate a sample of size n with this Gaussian distribution for different parameter ρ.

`vecteur_gaussien.sce`

```
function [] = gaussian_vector(rho)
  if abs(rho) >1
disp('the correlation must be between -1 and 1!')
disp('aborting...')
return
  end
  xbasc();
  n=1000;
  u1=rand(1,n);
  u2=rand(1,n);
  g1= ... // complete
  g2= ... // complete
  A=[1,0;rho,sqrt(1-rho^2)];
  z=A*[g1;g2];
  x=z(1,:);
  y=z(2,:);
  plot2d(x,y,-1)
endfunction
```

Remark The matrix $\mathcal{L} = \begin{pmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{pmatrix}$ is the Cholesky decomposition of $M = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$ (i.e. $\mathcal{L}\mathcal{L}^T = M$ and $\mathcal{L}$ is lower triangular). More general, this method allows us to simulate a centred $\mathbb{R}^d$-valued Gaussian vector with covariance matrix Γ since we have its Cholesky decomposition $\mathcal{L}$ the lower triangular matrix satisfying $\mathcal{L}\mathcal{L}^T = \Gamma$. In fact, if G is a centred $\mathbb{R}^d$-valued standard Gaussian vector $\mathcal{L}G$ has the distribution $\mathcal{N}(0, \Gamma)$.

If X is positive definite, then R = chol(X) produces an upper triangular matrix R such that R'*R = X.

chol(X) uses only the diagonal and upper triangle of X. The lower triangular is assumed to be the (complex conjugate) transpose of the upper. Thanks to Scilab the function `chol` provides the Cholesky decomposition of symmetric positive definite matrix. This function returns $\mathcal{L}^T$. The algorithm to calculate the Cholesky decompsition $\mathcal{L}$ of a symmetric positive definite matrix A is given by

$$\begin{array}{l} \text{pour } k = 1..size \text{ faire} \\ \quad \mathcal{L}_{k,k} = \sqrt{A_{k,k} - \sum_{j<k} \mathcal{L}_{k,j}^2} \\ \quad \text{pour } i = k+1..size \text{ faire} \\ \quad\quad \mathcal{L}_{i,k} = \dfrac{A_{i,k} - \sum_{j<k} \mathcal{L}_{k,j}\mathcal{L}_{i,j}}{\mathcal{L}_{k,k}} \end{array}$$

2.4 Strong Law of Large Number

The basic principle of the so-called Monte Carlo method is to implement on a computer the Strong Law of Large Number (SLLN). Let $(Y_1, \ldots, Y_n)$ be a sample of size n with a uniform distribution on the interval $[0, 1]$. By using the function `cumsum`, calculate the vector $\bar{Y} = (\bar{Y_1}, \ldots, \bar{Y_n})$ of empirical means where $\bar{Y_i} = \frac{1}{i}\sum_{k=1}^{i} Y_k$ then plot the function $i \longmapsto \bar{Y_i}$.

2.5 Central Limit Theorem

The rate of convergence in the Strong Law of Large Number is ruled by the Central Limit Theorem. We consider a sample $(Z_1, \ldots, Z_n)$ with Z_i sont i.i.d random variables with the same law as $\sqrt{12p}\left(\frac{1}{p}\sum_{i=1}^{p} U_i - \frac{1}{2}\right)$, the random variables $(U_i, i \leq p)$ are i.i.d with uniform distribution on the interval $[0, 1]$. Use the following function to plot the histogram of $(Z_i, i \leq n)$ with nc classes. Vary n, p, nc we can take $p = 1$, $p = 12$with large and small values of nc. We will take n of order 1000.

TCL.sce

```
function [] = tcl(n, p, nc)
  xbasc();
  X=rand(n,p);
  Z=sqrt(12/p)*(sum(X,'c') - p/2); // sum of columns, centred  and
                                   // renormalised
  histplot(nc,Z)
  C=[-5:1/1000:+5];
  plot2d(C,exp(-C.^2/2)/sqrt(2*%pi),3) // plobability density of standad distribution
endfunction
```

Exercice : Let $(Y_1, \ldots, Y_n)$ be a sample of size n-échantillon, thanks to Central Limit Theorem, built an asymptotic confidence interval $[\bar{Y}_n - r_n, \bar{Y}_n + r_n]$ where r_n is a function of our simulations $r_n = r_n(Y_1, \ldots, Y_n)$. In the case of uniform distribution on the interval $[0, 1]$ calculate the radius r_n. For $1 \leq i \leq n$, plot the evolution of $i \longmapsto \bar{Y}_i$, the upper bound $i \longmapsto \bar{Y}_i + r_i$ and the lower bound $i \longmapsto \bar{Y}_i - r_i$ with different colors.

2.6 Kernel density estimation

We aim to estimate the probability density function of the distribution of a sample with size n. The first approach to estimate the probability density is to use a normalized histogram. More precisely, divide the sample space into a number of bins and approximate the density p at the center of each bin by the fraction of points in the data that fall into the corresponding bin by the size of the sample normalized by the bin width

$$p(x) \approx \frac{1}{n} \frac{\#\text{points in the same bin}}{\text{bin width}}.$$

The histogram is a very simple form of density estimation but has several drawbacks. A more efficient method is given by the non parametric kernel density estimatior. Let f be a probability density function and $(X_1, \cdots, X_n)$ be a sample size of i.i.d random variables with porbability density f. We introduce a family of kernel $(K_h)_{h>0}$ satisfying :

$$\begin{cases} K_h(x) \geq 0 & \text{pour tout } h > 0 \text{ et } x \in \mathbb{R}..... \\ \int_{\mathbb{R}} K_h(x)dx = 1 & \text{pour tout } h > 0 \\ K_h \xrightarrow[h \to 0]{} \delta_0. & \end{cases}$$

We usually use the kernel $K_h(x) = K(x/h)$ where K is the Gaussian kernel $K(x) = \frac{1}{\sqrt{2\pi}} \exp(-x^2/2)$ or the Epanechnikov's kernel $K(x) = \frac{3}{4}(1-x^2)\mathrm{I}_{]-1,1[}(x)$. We estimate the probability density f by the function

$$\hat{f}_n(x) = \frac{1}{nh_n} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h_n}\right).$$

The parameter $(h_n)_n$ is called the bandwidth and a judicious choice is of order $n^{-1/5}$.

We consider two independent standard Gaussian random variables X and Y. Let Z equal to X with probability $\frac{1}{3}$ and $aY + b$ with probability $\frac{2}{3}$ with $a > 0$ and $b \in \mathbb{R}$. The probability density of Z is given by

$$f(x) = \frac{2}{3a\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-b}{a}\right)^2} + \frac{1}{3\sqrt{2\pi}} e^{-\frac{x^2}{2}}.$$

Use the following program to study the kernel estimator where the exact density appears in red.

noyau.sce

```
clear;
// nc the number of classes in the histogram
// n the size of the sample
// hn of the bandwidth of order n^(-1/5)
function [] = estim_noyau(n,nc,hn)
  xbasc();
  a=1;
  b=3;
// two independent uniform random variables
  U=rand(n,1);
  V=rand(n,1);
// two independent Gaussian random variables
  X = sqrt(-2*log(U)).*cos(2*%pi*V);
  Y = sqrt(-2*log(U)).*sin(2*%pi*V);
// Z = X with proba 1/3 and aY+b with proba 2/3
  epsilon = rand(n,1);
  Z = (a*Y+b).*(epsilon > 1/3) + X .* (epsilon <= 1/3);
// histogram of the simulate variable, nc=number of classes
  histplot(nc,Z)
// estimation of the density  by the kernel method,
// Epanechnikov's kernel :
  C=[min(Z)-1:1/n:max(Z)+1];
  for i=1:length(C)
    B(i)=1/(n*hn)*3/4*sum((1-((C(i)-Z)/hn).^2).*(-1<(C(i)-Z)/hn).*((C(i)-Z)/hn<1));
  end
```

```
plot2d(C,B,2)
// the exact density
  f = (2/(3*a) * exp(-((C-b)/a).^2/2) + 1/3 * exp(-C.^2/2))/sqrt(2*%pi);
  plot2d(C,f,5);
endfunction
```

3 Value at Risk

We consider a portfolio such as for instance a collection of stocks, bonds or risky loans or the overall position of a financial institution. The value of the portfolio at time t is denoted $V(t)$. Given a time horizon Δt the profit over the interval $[t, t+\Delta t]$ is given by $V(t+\Delta t)-V(t)$ and its distribution is called the profit-and-loss distribution (P& L). The loss over the interval is then

$$L_{[t,t+\Delta t]} = -(V(t+\Delta t) - V(t))$$

and its distribution is called the loss distribution. Typical values of Δt is one day (or 1/250 years as we have approximately 250 trading days in one year), ten days, one month or one year. We may introduce a discrete parameter $n = 0, 1, 2, \cdots$ and use $t_n = n\Delta t$ as the actual time and use the notation $L_{n+1} = -(V_{n+1} - V_n)$ where we write V_n for $V(n\Delta t)$.

Example Consider a portfolio of d stocks with $(S_t^1, \cdots, S_t^d)$ their price at time t. If the portfolio contain H_t^i units of stock number i then the value of the portfolio at time t

$$V_t = \sum_{i=1}^{d} H_t^i S_t^i.$$

In financial statistics one often tries to find a statistical model for the evolution of the stock prices for example a model for $S_{n+1,i} - S_{n,i}$. Hence we can be able to compute the loss distribution L_{n+1}.

Remark Similarly, we can consider the portfolio return $\frac{V(t+\Delta t)-V(t)}{V(t)}$ as interest variable.

We modelize our loss function by a random variable. In order to control the risk related to the portfolio variation we define a function from a set of random variables to the real numbers. It is called a risk measure. In 1988 the first Basel Accord on Banking Supervision initiated an important step toward an international minimal capital standard and in 1996 an amendment to Basel I prescribes a so-called standardized model for market risk with an option for lager banks to use internal Value-at-Risk models.

Value at Risk The aim is to introduce the widely used risk measure known as Value-at-Risk. For a loss function L, the Value at Risk with confidence level $\alpha \in]0, 1[$ is

$$VaR_\alpha(L) \stackrel{\text{def}}{=} \inf\{m \in \mathbb{R} \mid \mathbb{P}(L \leq m) \geq \alpha\}$$

Intuitively, saying the $VaR_{95\%}$ of a portfolio is 100 means that the loss function L will be larger than 100 with probability less then 5%. The VaR_α informs us about the risk at a given level α, but not give us the distribution of the loss random variable L.

Recall The distribution function F of the random variable L is defined by $F(x) = \mathbb{P}(L \leq x)$. The function F is always nondecreasing, "càdlàg" (French acronym for right continuous with left left limits) and $\lim_{x\to+\infty} F(x) = 1$, $\lim_{x\to-\infty} F(x) = 0$. One can always associate to F its canonical left inverse called also the generalized inverse defined on the open unit interval $(0, 1)$ by

$$F^-(u) \stackrel{\text{def}}{=} \inf\{x \in \mathbb{R} \mid F(x) \geq u\}$$

In statistics the α-quantile of the random variable L, called also the quantile of order α, written ξ_α is the number $F^-(\alpha)$. When F is increasing and continuous on the real line then F has an inverse function denoted by F^{-1} and $F^{-1} = F^-$. Clearly, we can replace $\mathbb{R}$ by any interval $[a, b] \in \mathbb{R}$. In the discrete case, when L takes finitely many values in $E = \{x_1, \cdots, x_n\} \subset \mathbb{R}$ with $x_1 < x_2 < \cdots < x_n$, the "càdlàg" distribution function F is a step function defined by

$$F(x) = \mathbb{P}(L = x_1) + \cdots + \mathbb{P}(L = x_k), \quad \forall x \in [x_k, x_{k+1}[, k \in \{1, \cdots, n-1\}.$$

Then, one checks that the generalized inverse distribution function is given by

$$\forall u \in (0, 1), \quad F^-(u) = x_k \text{ if } \mathbb{P}(L = x_1) + \cdots + \mathbb{P}(L = x_{k-1}) < u \leq \mathbb{P}(L = x_1) + \cdots + \mathbb{P}(L = x_k).$$

Hence, $VaR_\alpha(L)$ is the quantile of order α of ξ_α and when F is invertible then

$$\mathbb{P}(L \leq VaR_\alpha) = \alpha \quad \text{and} \quad \mathbb{P}(L > VaR_\alpha) = 1 - \alpha.$$

Hence $VaR_\alpha(L)$ is given by the smallest number such that the probability that the loss L exceeds this number is no larger than $1 - \alpha$. Common confidence levels α are 95%, 99% and 99.9%, depending on the application.

3.1 Value at Risk of a log-normal distribution

A positive random variable X is called Log-normal if $\log X \sim \mathcal{N}(\mu, \sigma^2)$. Estimate the Value at Risk for different confidence level (1%, 5%, 10%) of a log-normal distribution. Plot the distribution function called also the cumulative distribution function of a random variable with log-normal distribution. For simulation, we will take $\mu = -1$ and $\sigma^2 = 0.09$.

```
stacksize(2*10^8);
N=1000; // number of simulation
moyenne=-1;
sigma2=0.09;
Y=grand(1,N,'nor',moyenne,sigma2) ;
// simulation of Gaussian variable
Z=sort(exp(Y));
// decreasing sort of the simulated realizations of a Log-Normal r.v.
```

```
values=  Z(?:-1:1);// ? the first argument is the symbol dollar

// increasing sorting

 // the VaR is quantile
alpha=0.99;
quantile=values(int(alpha*N));
disp('VaR_('+string(alpha) +')(Z)='+string(quantile))

alpha=0.95;
quantile=values(int(alpha*N));
disp('VaR_('+string(alpha) +')(Z)='+string(quantile))

alpha=0.90; // level
quantile=values(int(alpha*N));
disp('VaR_('+string(alpha) +')(Z)='+string(quantile))

xset('window',10) ; xbasc(); plot2d(values,linspace(0,1,N));
plot2d(values,alpha*ones(values),style=5);
xtitle('Empirical cumulated distribution of a Log-Normal random variable
L:...log(L) ~ N(' +string(moyenne) +',' +string(sigma2)+ ')')

xset('window',11) ; xbasc(); histplot(10,Z);
xtitle('Empirical histogram of a Log-Normal random variable L:: ...
log(L) ~ N(' +string(moyenne) +',' +string(sigma2)+ ')')
```

3.2 Comparison of the VaR under different laws

Compare the quantiles of the normal law and Laplace law and note that the parametric method is sensitive to the choice of parametric distribution family.

```
stacksize(2*10^8);
N=10000; // number of simulation
moyenne=1;
sigma2=9;
Y1=grand(1,N,'nor',0,1) ;
Y2=grand(1,N,'exp',1);
epsilon=2*grand(1,N,'bin',1,0.5)-1 ;
Y2=epsilon.*Y2;
Y2=Y2/sqrt(2);
Y1=sqrt(sigma2)*Y1+moyenne;
Y2=sqrt(sigma2)*Y2+moyenne;
// simulation of Gaussian distribution
Z1=sort(Y1);
Z2=sort(Y2);
```

```
// decreasing sort of the simulated realizations of Normal r.v.
values1=  Z1($:-1:1);
values2=  Z2($:-1:1);
// increasing sorting
// the VaR is a quantile
alpha=0.99;
quantile1=values1(int(alpha*N));
quantile2=values2(int(alpha*N));
disp('VaR_('+string(alpha) +')(Z1)='+string(quantile1))
disp('VaR_('+string(alpha) +')(Z2)='+string(quantile2))

alpha=0.95;
quantile1=values1(int(alpha*N));
quantile2=values2(int(alpha*N));
disp('VaR_('+string(alpha) +')(Z1)='+string(quantile1))
disp('VaR_('+string(alpha) +')(Z2)='+string(quantile2))

alpha=0.90; // level
quantile1=values1(int(alpha*N));
quantile2=values2(int(alpha*N));
disp('VaR_('+string(alpha) +')(Z1)='+string(quantile1))
disp('VaR_('+string(alpha) +')(Z2)='+string(quantile2))

xset('window',10) ; xbasc();
plot2d(values1,linspace(0,1,N));
plot2d(values2,linspace(0,1,N),5);
plot2d(values1,alpha*ones(values1),style=5);
plot2d(values2,alpha*ones(values2),style=5);
xtitle('Empirical cumulated distribution of a Normal random variable and Laplace')


xset('window',10) ; xbasc();
alpha=[0.01:0.01:0.99]
plot2d(alpha,values1(int(alpha*N)), 5);
```

3.3 Parametric method and confidence interval

Exercice : recall the parametric method to built an estimator $\hat{\xi}_\alpha$ of a quantile ξ_α $\alpha \in]0,1[$, based on $\hat{\mu}$ the estimator of the mean μ and $\hat{\sigma}$ the estimator the standad deviation σ of the a random variable L. Using the Monte Carlo approach, built an empirical confidence interval to ξ_α, according to a sample $(\hat{\xi}_\alpha^1, \cdots, \hat{\xi}_\alpha^M)$ of size $M \in \mathbb{N}$ with the same distribution of $\hat{\xi}_\alpha$ obtained by simulation of the couples $((\hat{\mu}_1, \hat{\sigma}_1), \cdots, (\hat{\mu}_M, \hat{\sigma}_M))$. Illustrate these results by numerical simulations.

3.4 Nonparametric method

Using the order statistics of the sample we can estimate the quantile of order α by the empirical quantile.

3.4.1 Exact confidence interval

We have the following result.

Theorem 1 *Let L be a random variable with continue cumulative distribution function F and generalized inverse $F^{-1}(p) = \inf\{x \in \mathbb{R} : F(x) \geq p\}$. Let $(L_k)_{1\leq k\leq n}$ be a sample of size n with the same distribution as the random variable L and $(L_{(k)})_{1\leq k\leq n}$ the corresponding order statistics. We let $L_{(0)} = -\infty$ and $L_{(n+1)} = \infty$. We have for all $0 \leq k^- \leq k^+ \leq n$ and all $p \in]0,1[$,*

$$\mathbb{P}(L_{(k^-)} \leq F^{-1}(p) \leq L_{(k^++1)}) = \mathbb{P}(k^- \leq \mathcal{B}(n,p) \leq k^+).$$

Exercice : To built a confidence intervall with level confidence x for the quantile $F^{-1}(p)$ we determine k^+ and k^- such that $\mathbb{P}(\mathcal{B}(n,p) > k^+) \leq \bar{x}/2$ and $\mathbb{P}(\mathcal{B}(n,p) < k^-) \leq \bar{x}/2$ with $\bar{x} = 1-x$. To determine k^+ we search for the smallest k such that $\mathbb{P}(\mathcal{B}(n,p) > k) \leq \bar{x}/2$. This is also the first k at which $F_{\mathcal{B}(n,p)}(k) \leq 1-\bar{x}/2$. So it is exactly the number of $i \in \{0,\cdots,n\}$ with $F_{\mathcal{B}(n,p)}(i) < 1-\bar{x}/2$. We determine k^- similarly. Develop a program providing us a confidence interval to VaR_α of a lost function L. Illustrate this result by numerical simulations.

3.4.2 Empirical confidence interval

Let L be a random variable and $(L_1,\cdots,L_n)$ be a sample of size $n \in \mathbb{N}$ with the same distribution as L. We can built an empirical confidence interval to the quantile ξ_α, $\alpha \in]0,1[$, according to the following central limit theorem for the empirical estimator $\hat{\xi}_\alpha$.

Theorem 2 *Let L be a random variable with probability density f. We assume that f is continuous at point ξ_α and $f(\xi_\alpha) > 0$. The empirical estimator $\hat{\xi}_\alpha$ satisfies*

$$\frac{\sqrt{n}}{\sigma_\alpha}(\hat{\xi}_\alpha - \xi_\alpha) \xrightarrow[n\to\infty]{loi} \mathcal{N}(0,1) \quad avec \quad \sigma_\alpha = \frac{\sqrt{\alpha(1-\alpha)}}{f(\xi_\alpha)}.$$

Exercice :

1. Built a confidence interval for VaR_α for a given level. Illustrate this result by numerical simulations. For a large value of n, $n \geq 30$ in practice, we can use the central limit theorem above and approximate f by the kernel method.
2. Plot the empirical mean and the empirical median with respect to the size of the sample for a standard Gaussian random variable $\mathcal{N}(0,1)$. Built the associated empirical confidence intervals of these both estimators and plot their upper and lower bounds in the same graph with different colors.

3.5 Value at Risk of financial model

We consider a financial model with d assets. We assume that the ith asset price is given by the Black and Scholes model and it is solution to $\frac{dS_t^i}{S_t^i} = rdt + \sigma dW_t^i, S_0^i = x_i$. with W_t^i a Brownian motion. In numerical simulations we will take $x_i = 100 + (-1)^i * i$, the interest rate $r = 0.05$, the volatility $\sigma = 0.3$ and $d = 10$. Finally, we will suppose the different Brownian motion independent.

1. Simulate the process $(S_t^1, \ldots, S_t^d)$ at points $t_k = k\delta t$, $t_k \in [0, T]$, with $\delta t = 1/250$ and $T = 4$.
2. For each day $t_k \in]0, T]$ the return on the ith asset is given by

$$, R_t^i = \frac{S_{t_{k-1}}^i - S_{t_k}^i}{S_{t_{k-1}}^i}$$

   and the portfolio return

$$R_t = \sum_{i=1}^{d} p_i r_t^i.$$

   with $\sum_{i=1}^{n} p_i = 1$. In numerical simulations we will take $p_1 = \cdots = p_d = 1/d$, simulate R at points $t_k = k\delta t$.
3. The historical VaR with level 99% of the portfolio is the quantile of the portfolio returns R_{t_k} at points $t_k = k\delta t$. For example, let $k \in \{1, \cdots, 1000\}$, we consider the order statistic $R_{(1)}, \cdots, R_{(1000)}$ and $VaR_{0.99}^{hist} = R_{(10)}$. Similarly, calculate $VaR_{0.95}^{hist}$.