

# An Introduction to Financial Mathematics with MATLAB

Dmitrii S. Silvestrov and Anatoliy A. Malyarenko

## Abstract

We describe a complex of programs called IFM, which will be used in the course “Introduction to financial mathematics” for students of the first year of a new Master educational program “Analytical finance” at the Mälardalen University.

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Elementary probability theory</b>	<b>3</b>
2.1	Getting started . . . . .	3
2.2	A model of price evolution . . . . .	4
2.3	Central Limit Theorem . . . . .	8
2.4	Geometric Brownian Motion . . . . .	11
<b>3</b>	<b>Elementary financial calculations</b>	<b>13</b>
3.1	Interest rates . . . . .	13
3.1.1	Problems . . . . .	16
3.2	Present value analysis . . . . .	16
3.2.1	Problems . . . . .	23
3.3	Rate of return . . . . .	23
3.3.1	Problems . . . . .	27
3.4	Pricing via arbitrage . . . . .	27
3.4.1	Problems . . . . .	32
3.5	The multi-period binomial model . . . . .	32
3.5.1	Problems . . . . .	37
3.6	The Black–Scholes formula . . . . .	39
3.6.1	Problems . . . . .	40
<b>A</b>	<b>Overcoming limitations</b>	<b>40</b>
A.1	The program <code>present_value</code> . . . . .	42
A.2	The program <code>ror</code> . . . . .	43
A.3	The program <code>mbm</code> . . . . .	44
A.3.1	Problems . . . . .	47

# 1 Introduction

The new Master education program “Analytical Finance” was introduced at the Mälardalen University in August, 2001. It gives students a general base in Mathematics, Business Administration, Economics and Computer Science. Advanced financial software is planned to be used in the teaching.

The course “Introduction to Financial Mathematics” is included in the program. It contains the elementary introduction to the theory of options pricing. All the necessary preliminary material including elementary probability, normal random variables and the geometric Brownian motion model is presented. The Black–Scholes theory of options as well as such general topics in finance as the time value of money and rate of return of an investment cash-flow sequence are part of the course. This material is covered in the first seven chapters of [4].

In order to support the course, the authors wrote a complex of MATLAB programs called IFM, which stands for **I**ntroduction to **F**inancial **M**athematics.

Why we choose MATLAB as a tool for the development of the complex IFM?

- ☞ MATLAB integrates mathematical computing, visualisation, and a powerful, but simple language.
- ☞ Open architecture. The user can easy manage MATLAB system: to add and remove MATLAB routines, both written by MATLAB team or by the user him (or her-)self.
- ☞ Graphical User Interface. This is a user interface built with graphical objects, such as buttons, text fields, sliders, and menus. In general, these objects already have meanings to most computer users. Applications that provide a Graphical User Interface are generally easier to learn and use since the person using the application does not need to know what commands are available or how they work.
- ☞ Mobility of the source code (Windows, Unix, Linux). The source code of MATLAB programs designed using IBM-compatible PC under MS Windows, will work in the same way on the Sun workstation under Unix or Linux.
- ☞ Special financial toolboxes. The toolbox is a set of MATLAB routines competed together. The Financial, Financial Derivatives, and Financial Time Series toolboxes contain a wide variety of computational and graphical tools for financial engineering.
- ☞ Simple interface to MAPLE®. MAPLE V, a symbolic computational language, can manipulate, solve, and evaluate mathematical expressions, both symbolically and numerically. MATLAB allows easy access to a kernel of MAPLE for symbolic computation (see, for example, [3]). We plan to use such an interface in other courses of the program “Analytical Finance”.

In this report we describe the complex. In Section 2, we first describe, how to start and quit MATLAB, and how to execute any program of the complex. Then we describe the programs for calculations in elementary probability theory. These calculations include a log-normal model of price evolution, an illustration of the central limit theorem and the simulation of the geometric Brownian motion.

In Section 3, the programs for elementary financial calculations are described. These include interest rates, present value analysis, rate of return, pricing the simplest options via arbitrage, multi-period binomial model, and the Black–Scholes formula.

Some programs of the complex contain limitations. For example, it is impossible to calculate the multi-period binomial model with more than 5 periods. One can overcome these limitations using the MATLAB Command Window directly. In Appendix A, we give the examples along with some description of the simplest MATLAB commands.

Some parts of the text need careful reading. Such places are emphasised by the special sign on the margins. You can see this sign here.



This complex is free software. To obtain IFM, contact Professor Dmitrii Silvestrov, Department of Mathematics and Physics, Mälardalen University, SE-72 123 Västerås, Sweden. E-mail: [dmitrii.silvestrov@mdh.se](mailto:dmitrii.silvestrov@mdh.se).

MATLAB is a registered trademark of The MathWorks, Inc. Other product or brand names are trademarks or registered trademarks of their respective holders.

The authors would like to thank Henrik Jönsson for careful reading of the manuscript and useful discussions.

## 2 Elementary probability theory

The programs in the complex IFM can be classified into two groups:

- ☞ Elementary probability theory;
- ☞ Elementary financial calculations.

In Subsection 2.1 we describe, how to start and quit MATLAB, and how to execute any program in the complex. In subsequent subsections we describe three programs, which belong to the first group.

### 2.1 Getting started

MATLAB® is a high performance language for technical computing. It integrates computation, visualisation and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. The name MATLAB stands for *matrix laboratory*.

The MATLAB documentation is available in PDF format at the following address: <http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml>

This section provides a brief introduction to starting and quitting MATLAB, and the commands to start programs of the complex IFM.

To start MATLAB, double-click the MATLAB shortcut icon  on your Windows desktop.

After starting MATLAB, the MATLAB desktop opens — see Fig. 1.

To end your MATLAB session, select **Exit MATLAB** from the **File** menu in the desktop, or type `quit` in the Command Window. You can find more information in Appendix A.

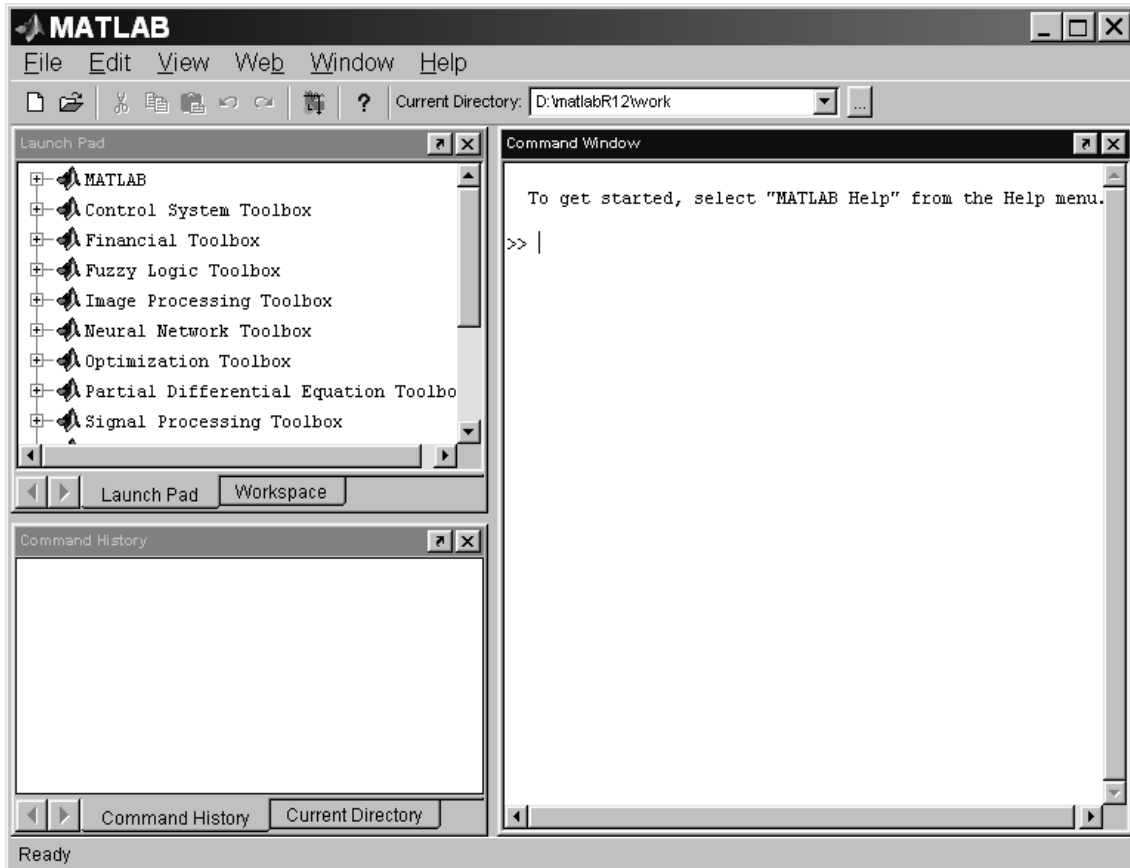


Figure 1: The MATLAB desktop

You can start any of the programs of the complex IFM, using one of the next two methods.

1. Type `i_fm` at the MATLAB prompt in the Command Window and press **Enter**. The dialog of the program `i_fm` will appear (Fig. 2). Choose a program from the list of available programs and press **Start** button. The dialog window of the corresponding program will appear.
2. Type the name of the program at the MATLAB prompt in the Command Window and press **Enter**. The names of all programs are listed in Table 1.

The programs `present_value`, `ror`, `mbm` and `black_scholes` require MATLAB Financial toolbox. All the other programs require only standard MATLAB installation.

*Remark 1.* A string “The list of all available programs” on Fig. 2 is called a *tooltip string*. Such a string appears when you put the mouse cursor over a user interface control. It gives the user a tip describing the corresponding control.

## 2.2 A model of price evolution

Let  $S(0)$  denote the initial price of some security. Let  $S(n)$ ,  $n \geq 1$  denote its price at the end of  $n$  additional weeks. A popular model for the evolution of these prices [4,

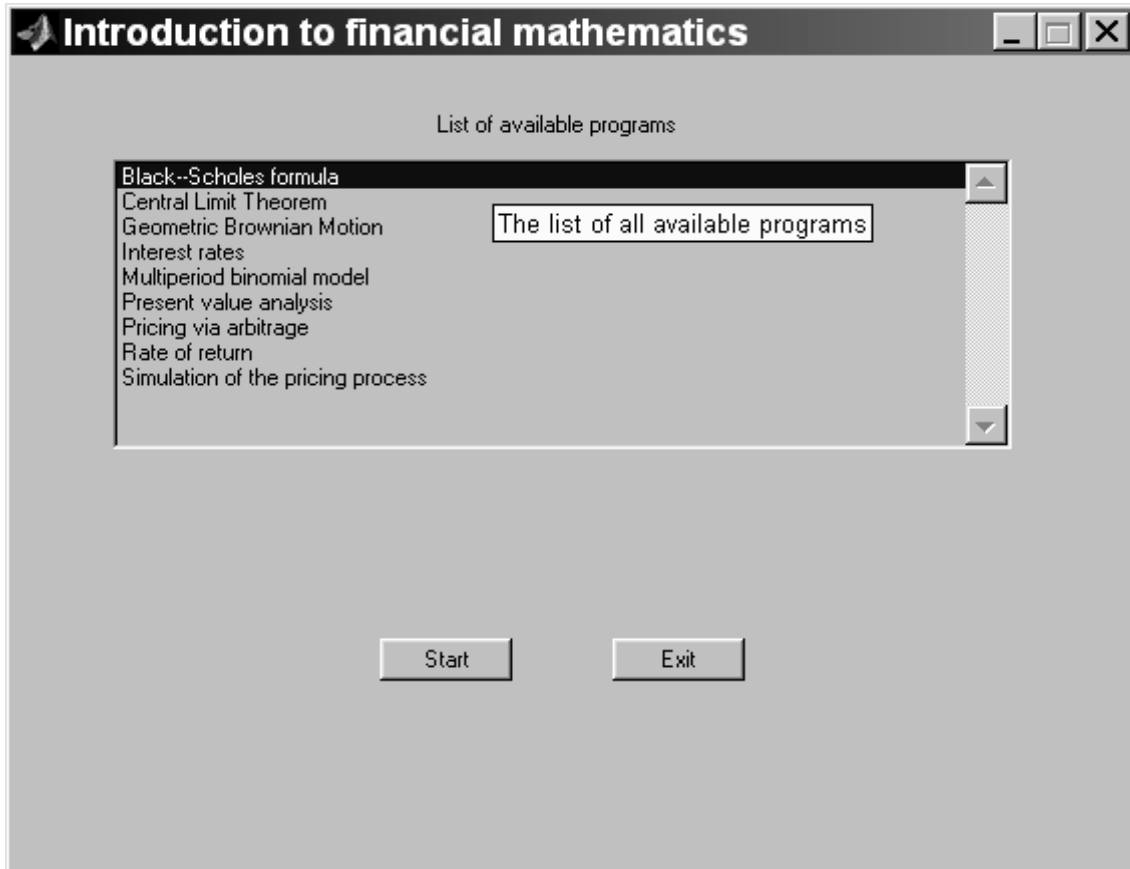


Figure 2: The Control Centre

Name	Description
ifm	Control centre
price_evolution	A model of price evolution
clt	An illustration of the Central limit theorem
gbm	Geometric Brownian motion
interest_rate	Interest rates
present_value	Present value analysis
ror	Rate of return
options_pricing	Pricing via arbitrage
mbm	Multiperiod binomial model
black_scholes	Black-Scholes formula

Table 1: Programs in the complex

Example 2.3d] assumes that the price ratios  $S(n)/S(n-1)$  are independent and identically distributed lognormal random variables. Recall that the random variable  $Y$  is called a *lognormal* random variable with parameters  $\mu$  and  $\sigma$  [4, p. 26] if  $\log(Y)$  is a normal random variable with mean  $\mu$  and variance  $\sigma^2$ .

The pricing process under consideration can be simulated with the help of the program `price_evolution` (Fig. 3).

The program `price_evolution` requires only standard MATLAB installation (no additional toolboxes are used).

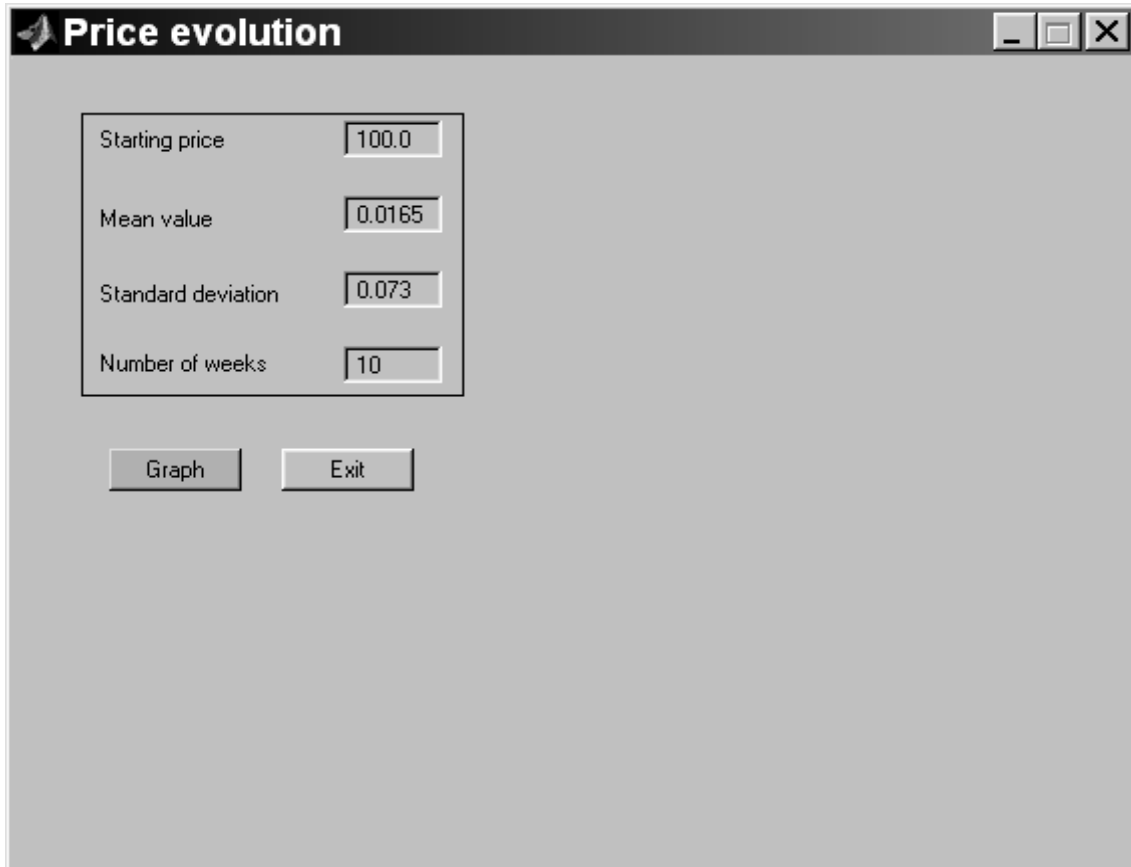


Figure 3: A window of the program `price_evolution`

The dialog window of the program `price_evolution` contains the *user interface controls*. The user interacts with the program, using these controls.

In the upper left corner of the dialog you can see a frame that encloses a group of *edit boxes*. Consider the functions of these boxes.

- ☞ **Starting price.** Here you can enter the value of parameter  $S(0)$ , i.e. the initial price of the security.
- ☞ **Mean value.** Here you can enter the value of parameter  $\mu$ , i.e. the mean of the logarithm of the random variable  $Y$ .
- ☞ **Standard deviation.** Here you can enter the value of parameter  $\sigma$ , i.e. the standard deviation of the logarithm of the random variable  $Y$ .

- ☞ **Number of weeks.** Here you can enter the length of the time interval of simulation of the evolution of the price, measured in weeks.

*Remark 2.* Some edit boxes in the programs of the complex have *default values*. In simple cases, you can start calculations without changing these values. We will refer to this feature as *solution of the standard problem*. Fig. 3 shows that the standard problem has the following values: starting price is equal to 100 (say, Swedish kronas), mean value is equal to 0.0165, standard deviation is equal to 0.073, and we want to simulate price evolution during 10 weeks.

*Remark 3.* Some edit boxes have prevention to non-correct input. For example, somebody tried to enter a negative value of the starting price into the corresponding edit box. The result is shown on Fig. 4. A warning message was produced, and the program returned the previous value into the edit box instead of the wrong value. In such cases, you must press the **OK** button in order to continue your work.

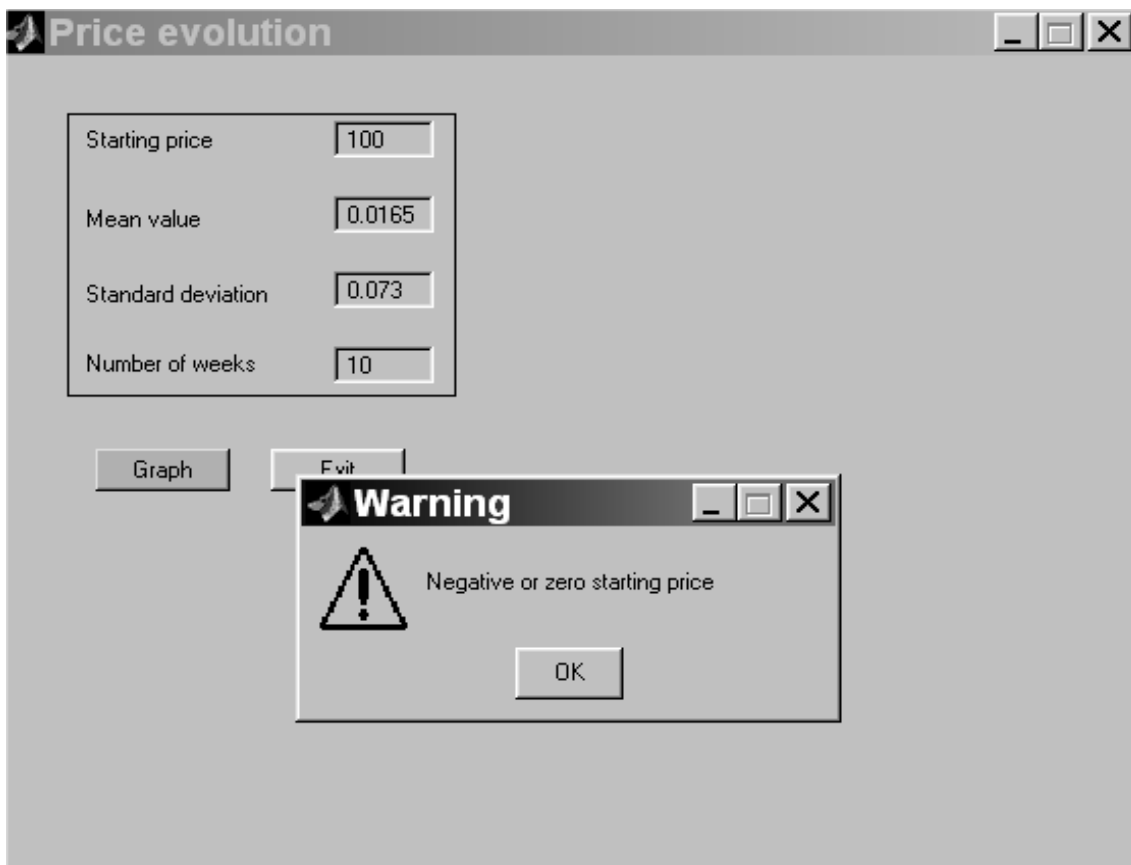


Figure 4: An example of incorrect input

Under the frame you can see two *push buttons*. The functions of these buttons are:

- ☞ **Graph.** A graph of the pricing process will be produced after pressing this button (Fig. 5).
- ☞ **Exit.** Exits the program.

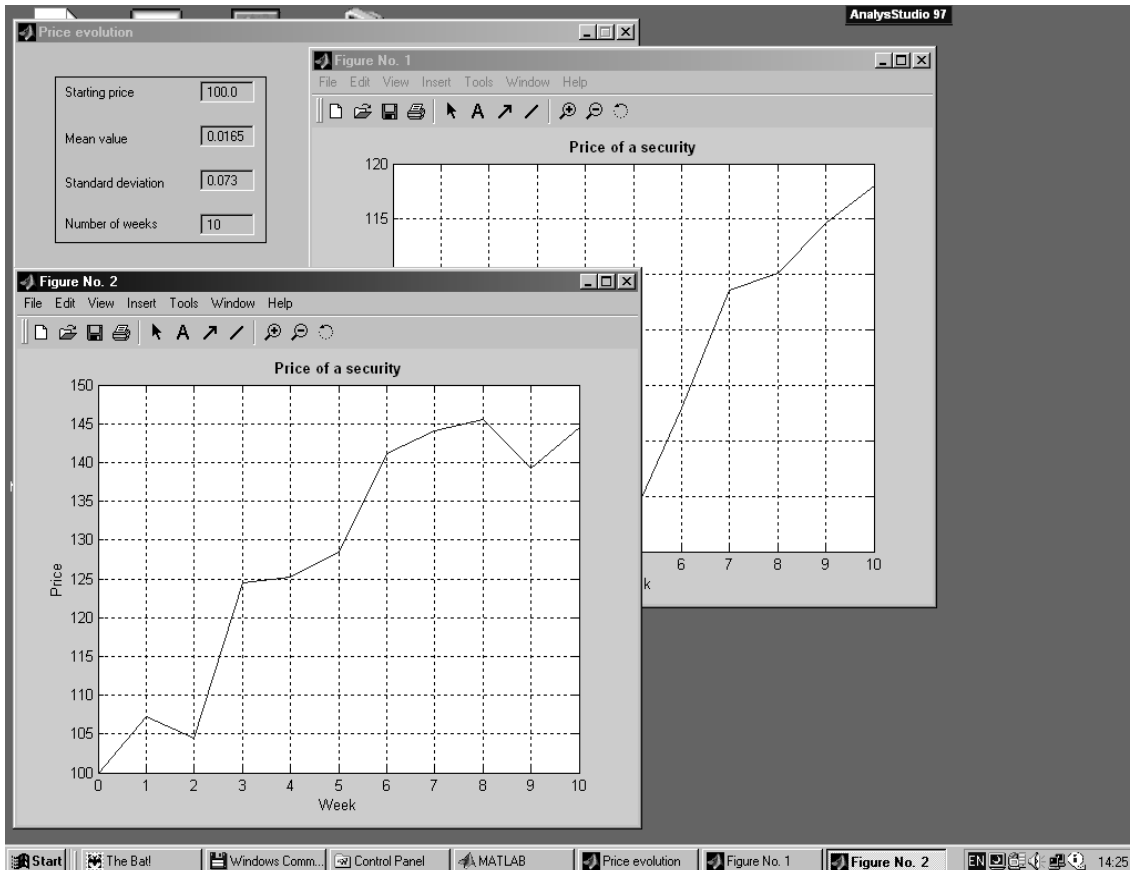


Figure 5: An output of the program `price_evolution`

As you can see from Fig. 5, the graph of the pricing process is drawn in a separate window. You can press the push button **Graph** several times and obtain several different graphs. Every graph is a stand-alone application, which occupies a separate place on the Windows taskbar and lives its own life. You can play with the menu system of any graph.

### 2.3 Central Limit Theorem

A rigorous formulation of the Central limit theorem needs sophisticated mathematical tools and is beyond the scope of this report. In our simplified approach, it is enough to know [4, Section 2.4] that if  $X_1, \dots, X_n, \dots$  is a sequence of independent and identically distributed random variables, each with expected value  $\mu$  and variance  $\sigma^2$ , then a random variable

$$Y_n = \frac{S_n - n\mu}{\sigma\sqrt{n}}$$

is approximately standard normal random variable. As a consequence, the sum

$$S_n = \sum_{k=1}^n X_k \tag{1}$$

is approximately a normal random variable with expected value  $n\mu$  and variance  $n\sigma^2$ .



Consider the following example. Let  $X_1$  be a *Bernoulli* random variable, i.e.,  $X_1 = 1$  with probability  $p$  and  $X_1 = 0$  with probability  $1 - p$ . In this case we have ([4, Examples 1.3c and 1.3f]):

$$EX_1 = p, \quad \text{Var } X_1 = p(1 - p).$$

According to the central limit theorem, a random variable

$$Y_n = \frac{S_n - np}{\sqrt{np(1 - p)}} \quad (2)$$

is approximately a standard normal random variable. Here  $S_n$  is defined by (1).

This example is illustrated by the program `clt` (Fig. 6).

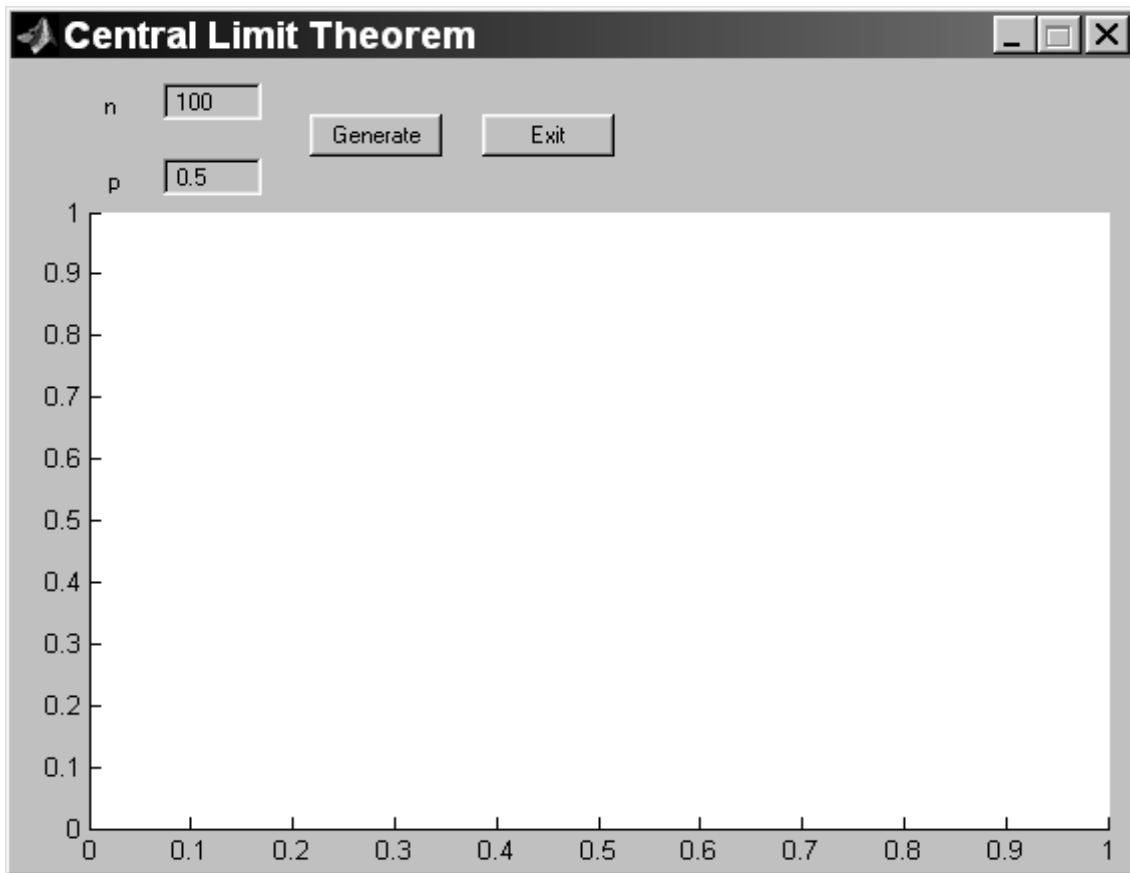


Figure 6: A window of the program `clt`

The program `clt` requires only standard MATLAB installation (no additional toolboxes are used).

Two edit boxes are situated in the left upper corner of the dialog. Their functions are:

- ☞ **n**. Here you can enter the number of independent Bernoulli random variables in the sum  $S_n$ . The standard problem has the value  $n = 100$ .
- ☞ **p**. Here you can enter the parameter of each of the independent Bernoulli random variables. The standard problem has the value  $p = 0.5$ .

Onto the right of the edit boxes you can see two push buttons. They have the following functions:

☞ **Generate.** A figure containing two graphs and their legend (Fig. 7) is produced. The solid line is the graph of the probability density of a normal distribution with parameters  $\mu = np$  and  $\sigma^2 = np(1 - p)$ . The dashed line is the graph of the distribution of the random variable  $S_n$ . It is drawn in the following way. We generate many (say,  $N$ ) realisations of the random variable  $S_n$ . Let  $k_0$  be the number of realisations taking the value 0, let  $k_1$  be the number of realisations taking the value 1, and so on up to  $k_n$ . We draw a dashed line through the points with coordinates

$$\left(0, \frac{k_0}{N}\right), \left(1, \frac{k_1}{N}\right), \dots, \left(n, \frac{k_n}{N}\right).$$

☞ **Exit.** Stops the program.

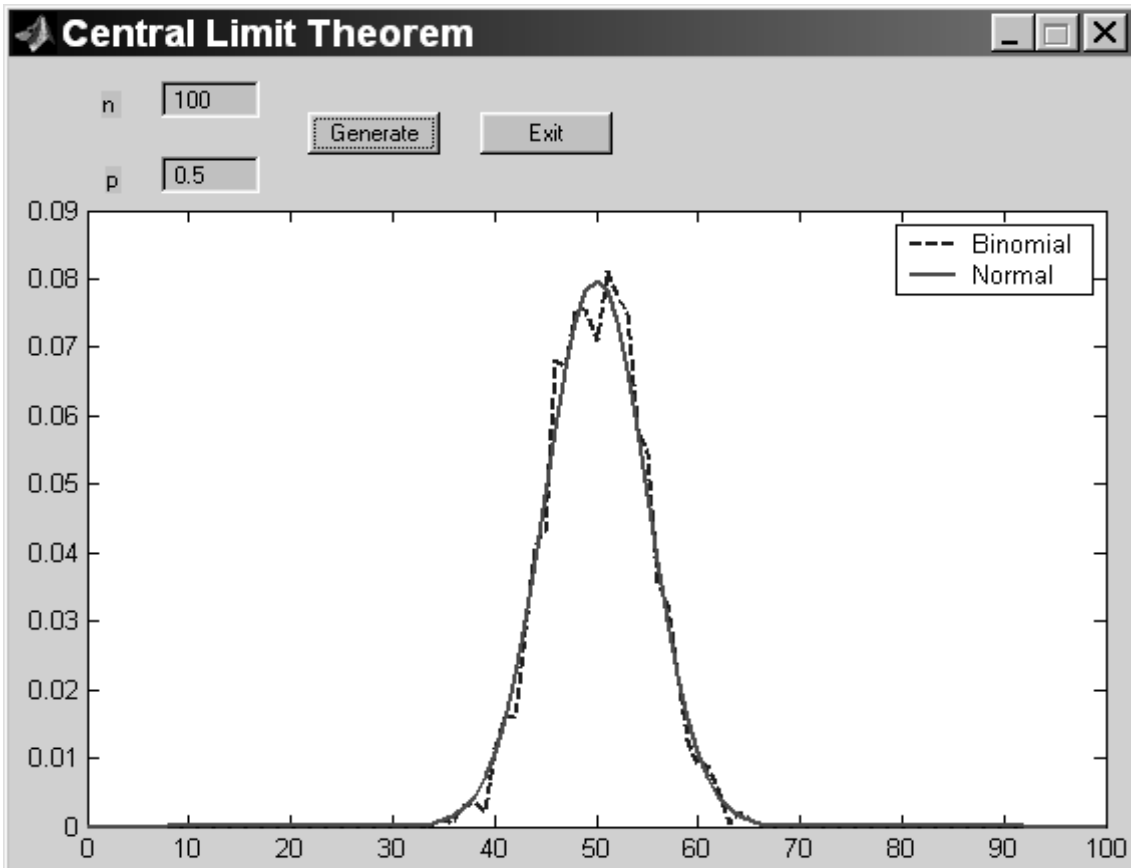


Figure 7: An output of the program `clt`

You can press the push button **Generate** several times using the same values of the variables  $n$  and  $p$ . The solid line on the graph will not change. The dashed line will be subject to small changes, because it is random.

## 2.4 Geometric Brownian Motion

Consider a collection of random variables  $S(y)$ ,  $0 \leq y < \infty$ . This collection follows a *geometric Brownian motion* with drift parameter  $\mu$  and volatility parameter  $\sigma$ , if for all non-negative values of  $y$  and  $t$ , the random variable

$$\frac{S(t+y)}{S(y)} \quad (3)$$

is independent of all random variables  $S(z)$ ,  $0 \leq z < y$ , and the logarithm of the random variable (3) is a normal random variable with mean  $\mu t$  and variance  $t\sigma^2$  [4, p. 32].

Geometric Brownian motion is a popular model of price evolution in *continuous time* (in contrast to a discrete time model from Section 2.2).

Suppose we want to build a computer model of the geometric Brownian motion. A computer can simulate values of any function only at some discrete set of points, say,  $n\Delta$ , where  $0 \leq n \leq N$ ,  $N$  is some number and  $\Delta$  denotes a small increment of time.


In order to simulate values  $S(n\Delta)$ ,  $0 \leq n \leq N$ , we can use a simpler model proposed in [4, p. 33–35]. The value  $S(0)$  is some non-random number which is known, because it denotes the initial price of a security. Now let  $Y_n$ ,  $1 \leq n \leq N$  be the sequence of independent Bernoulli random variables with parameter

$$p = \frac{1}{2} \left( 1 + \frac{\mu}{\sigma} \sqrt{\Delta} \right).$$

Our model can be calculated as

$$S(n) = \begin{cases} S(n-1)e^{\sigma\sqrt{\Delta}}, & \text{if the price goes up } (Y_n = 1), \\ S(n-1)e^{-\sigma\sqrt{\Delta}}, & \text{if the price goes down } (Y_n = 0). \end{cases} \quad (4)$$

As  $\Delta$  tends to 0, the model (4) tends to geometric Brownian motion. A rigorous proof of this fact is very complicated. A simplified proof can be found in [4, p. 33–35].

*Remark 4.* On page 32 and subsequent pages of [4] the author denotes the initial price by two different symbols, namely, by  $S(0)$  and  $s_0$ . We prefer to use only the first one. 

The model (4) is realised in the program `gbm` (Fig. 8).

The program `gbm` requires only standard MATLAB installation (no additional tool-boxes are used).

Consider the functions of the edit boxes of the program `gbm`.

- ☞ **Initial price.** Here you can enter the value of the parameter  $S(0)$ , i.e., the initial price of a security. The standard problem has the value  $S = 100$ .
- ☞ **Drift.** Here you can enter the value of the parameter  $\mu$ , i.e., the drift parameter of the geometric Brownian motion under simulation. The standard problem has the value  $\mu = 0.01$ .
- ☞ **Volatility.** Here you can enter the value of the parameter  $\sigma$ , i.e., the volatility parameter of the geometric Brownian motion under consideration. The standard problem has the value  $\sigma = 0.2$ .

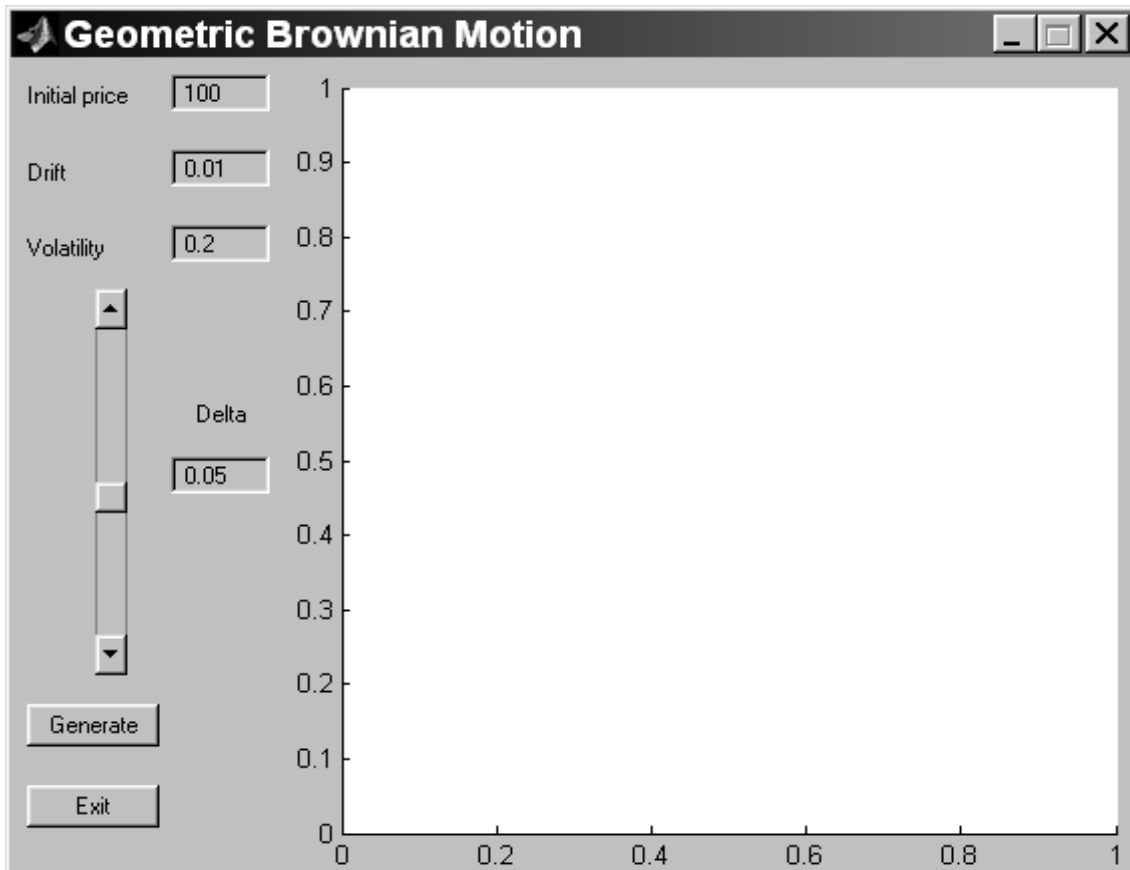


Figure 8: A window of the program gbm

- ☞ **Delta.** Here you can enter the value of the time increment  $\Delta$ . The standard problem has the value  $\Delta = 0.05$ . You can also change the value of the time increment using the *slider* on the left hand side of the **Delta** edit box. You can move the slider's bar by pressing the mouse button and dragging the slide, by clicking on the trough, or by clicking an arrow. The minimum slider (and edit box) value is equal to 0.01, the maximum slider and edit box value is equal to 0.1.

The functions of two push buttons in the lower left part of the dialog are:

- ☞ **Generate.** Generates a graph of the model (4). See Fig. 9.
- ☞ **Exit.** Stops the program.

You can press the push button **Generate** several times without changing model parameters. Every time you will obtain a graph of a new realisation of the random sequence (4).

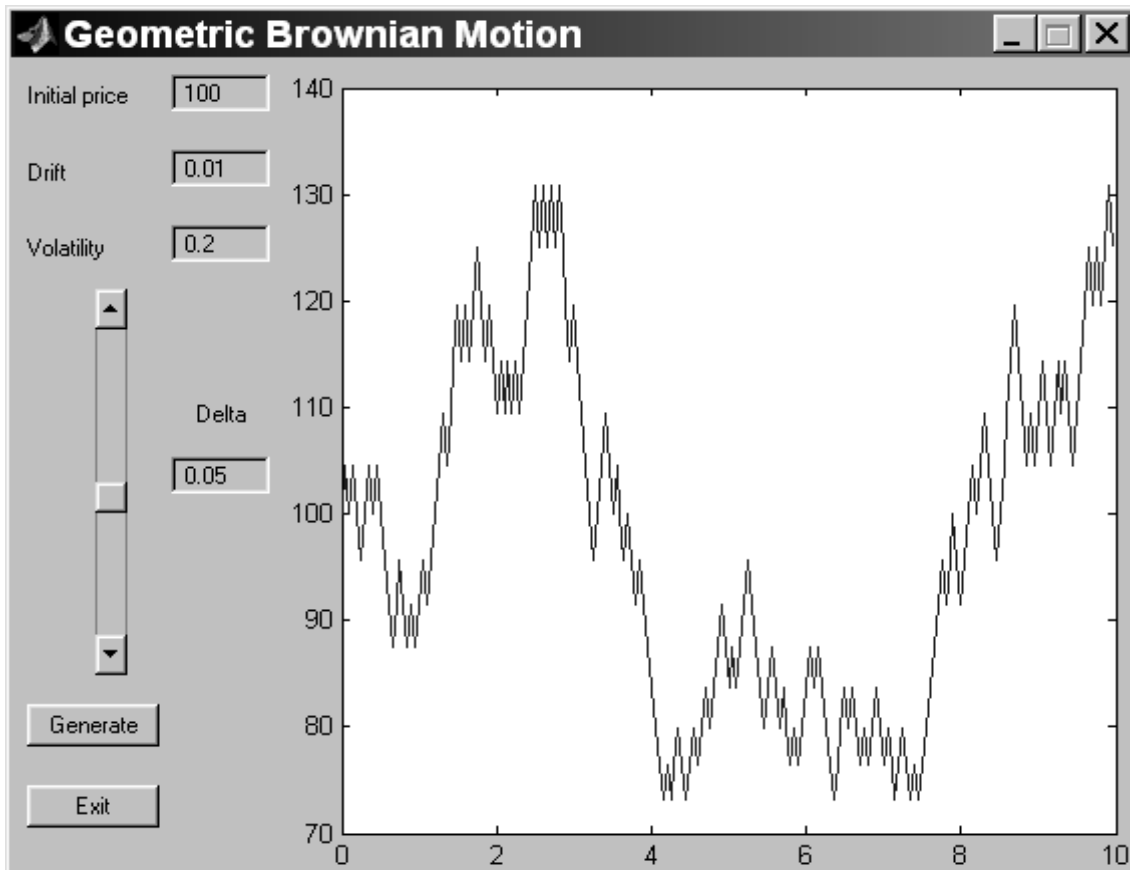


Figure 9: An output of the program gbm

### 3 Elementary financial calculations

#### 3.1 Interest rates

Recall that the *principal* is an amount of borrowed money which must be repaid along with some *interest*. Denote the principal by  $P$ . An *nominal annual interest rate* or *simple interest*  $r$  means that the amount to be repaid one year later is  $P(1 + r)$ .

Different financial institutions use various *compound interests*. For example, the interest can be compounded *semi-annually*. It means, that after six months you owe  $P(1 + r/2)$ , and after one year you pay  $P(1 + r/2)^2$ . Similarly, if the loan is compounded at  $n$  equal intervals in the year, then the amount owed at the end of the year is  $P(1 + r/n)^n$ .

In order to compare different compound interests we use the *effective annual interest rate*. The payment made on a one-year loan with compound interest is the same as if the loan called for simple interest at the effective annual interest rate. If we denote the effective annual interest rate by  $r_{\text{eff}}$ , then this definition can be expressed mathematically as

$$r_{\text{eff}} = (1 + r/n)^n - 1.$$

A *continuous compounding* is naturally referred as the limit of this process as  $n$  growth

larger and larger. In this case the amount owed at the end of the year is

$$P \lim_{n \rightarrow \infty} (1 + r/n)^n = Pe^r.$$

Similarly, if the principal  $P$  is borrowed for  $t$  years at a nominal interest rate of  $r$  per year compounded continuously, then the amount owed at time  $t$  is

$$P \lim_{n \rightarrow \infty} (1 + rt/n)^n = Pe^{rt}.$$

The program `interest_rate` (Fig. 10) calculates different compound interests and shows the corresponding graphs.

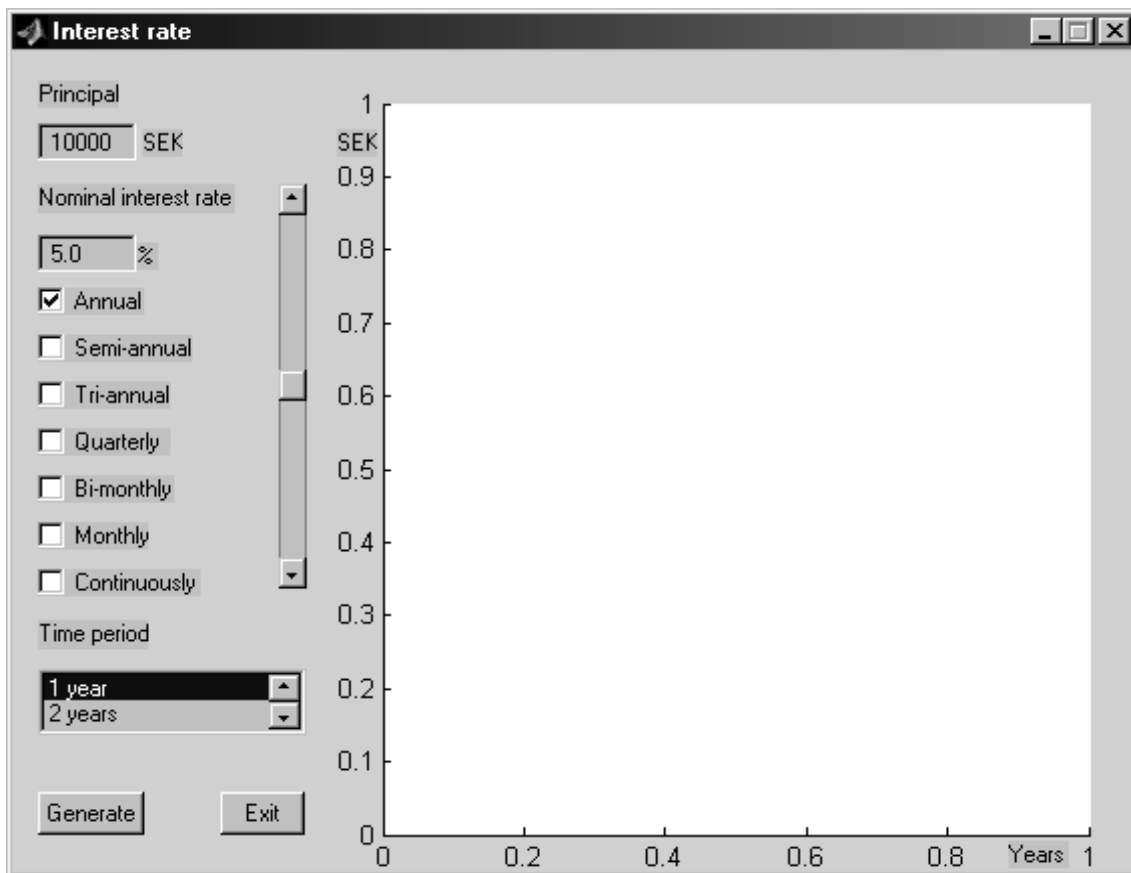


Figure 10: A window of the program `interest_rate`

The program `interest_rate` requires only standard MATLAB installation (no additional toolboxes are used).

Consider the functions of the two edit boxes in the upper left corner of the dialog.

- ☞ **Principal.** The amount of money which is borrowed. The standard problem has the value  $P = 10000$ .
- ☞ **Nominal interest rate.** The simple interest per year. The standard problem has the value  $r = 0.05$ , which corresponds to 5%, Here, as well as in all other programs of

the complex, you must enter percent values. The corresponding numerical value of  $r$  is calculated by the program itself. You can also change the value of the interest using the slider on the right hand side of the **Nominal interest rate** edit box. The minimum slider (and edit box) value is equal to 0%, the maximum slider and edit box value is equal to 10%.

Seven *checkboxes* are situated below the edit boxes. The checked state of any box means that the corresponding compound interest will be calculated. Consider these checkboxes in more details.

- ☞ **Annual.** Corresponds to the value  $n = 1$ , i.e. the simple interest. The interest is compound annually. The standard problem calculates this kind of interest.
- ☞ **Semi-annual.** Corresponds to the value  $n = 2$ . The interest is compound every 6 months. The standard problem does not calculate this kind of interest.
- ☞ **Tri-annual.** Corresponds to the value  $n = 3$ . The interest is compound every 4 months. The standard problem does not calculate this kind of interest.
- ☞ **Quarterly.** Corresponds to the value  $n = 4$ . The interest is compound every 3 months. The standard problem does not calculate this kind of interest.
- ☞ **Bi-monthly.** Corresponds to the value  $n = 6$ . The interest is compound every 2 months. The standard problem does not calculate this kind of interest.
- ☞ **Monthly.** Corresponds to the value  $n = 12$ . The interest is compound every month. The standard problem does not calculate this kind of interest.
- ☞ **Continuously.** Corresponds to the limit, when  $n$  grows larger and larger. The interest is compound continuously. The standard problem does not calculate this kind of interest.

A *list box* under the checkboxes contains five elements. It determines the borrowing time (in years) and can take values 1, 2, 3, 4 or 5 years. You can choose any of these terms from the list. The standard problem has value  $t = 5$  years.

The functions of the two push buttons in the lower left part of the dialog are:

- ☞ **Generate.** Generates a graph of the repay. See Fig. 11. If no checkboxes are checked, an error message is generated instead.
- ☞ **Exit.** Stops the program.

The graph (Fig. 11) shows the time dependence of different compound schemes chosen by the user. A legend explains which line corresponds to which scheme, and shows the corresponding effective interest rate.

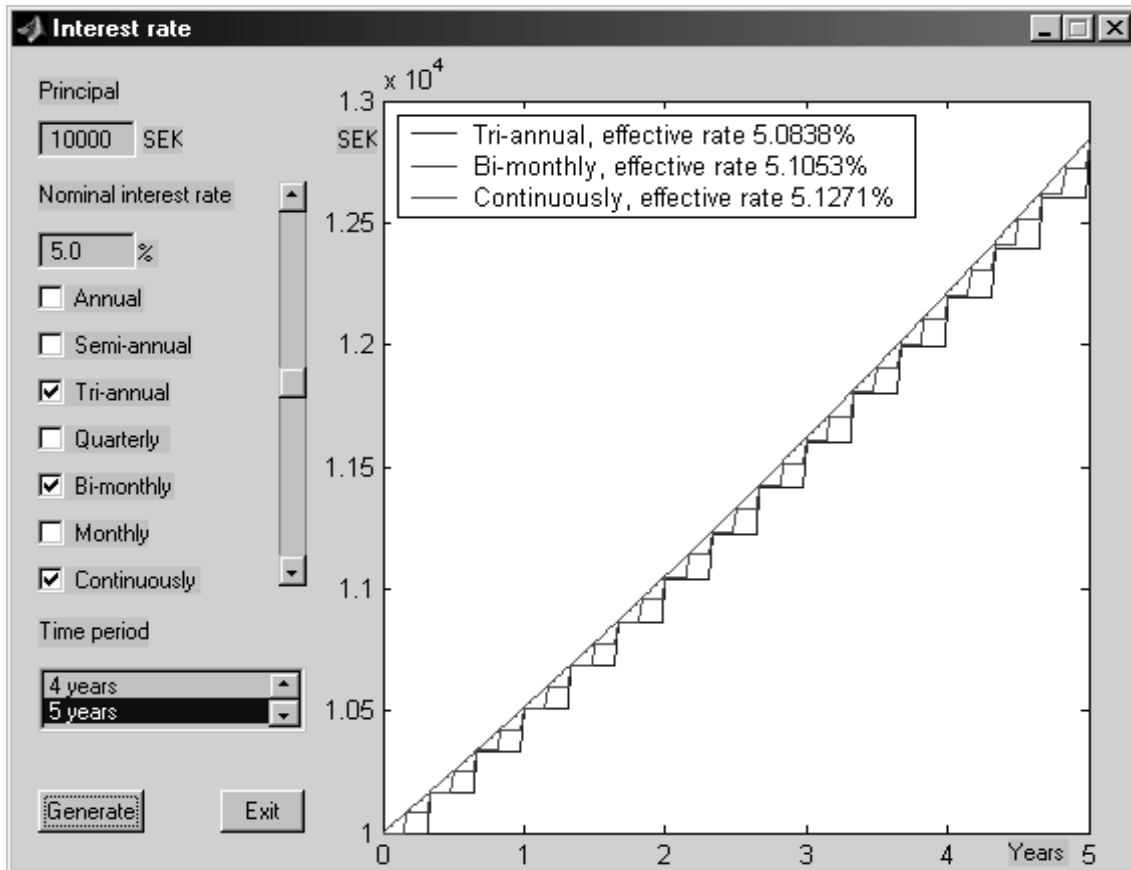


Figure 11: An output of the program `interest_rate`

### 3.1.1 Problems

**Problem 1.** What is the effective annual rate if a rate of 8 percent per year is compounded

- semi-annually?
- tri-annually?
- quarterly?
- bi-monthly?
- monthly?
- continuously?

### 3.2 Present value analysis

Consider the next example. You have a saving account earning 6% interest rate per year. Today is November 1. You need to pay to somebody \$201 on December 1. How much money should you put on your account today?



The monthly interest rate is equal to  $0.06/12 = 0.005$ . You can pay

$$\frac{201}{1 + 0.005} = 200$$

dollars today. On December 1 you will have

$$200 \times (1 + 0.005) = 201$$

dollars on your account — exactly as you need.

We say that \$200 is the *present value* of your payment of \$201 one month later from today. In this case it means, that you can pay \$200 today or \$201 one month later — the results will be the same. In other words, the *cash flows* from Table 2 are equal.

Cash flow 1		Cash flow 2	
Date	Payment	Date	Payment
November 1	-200	December 1	-201

Table 2: Two equal simple cash flows

Consider more complicated example. You obtain \$200 monthly into a saving account earning 6%. The payments are made at the end of the month for five years. What is the present worth of these payments?

The monthly interest rate is equal to  $0.06/12 = 0.005$ . Assume for simplicity that you will obtain \$200 only once one month later. It means, that today you can obtain an amount of

$$\frac{200}{1 + 0.005} \approx 199.00$$

dollars and one month later you will have an amount of \$200. Therefore the present value of this payment is equal to \$199.00

Assume now that you will obtain \$200 one month later and \$200 two months later. The present value of the first payment is still equal to \$199.00 and the present value of the second payment is equal to

$$\frac{200}{(1 + 0.005)^2} \approx 198.01$$

dollars. Indeed, today you can obtain an amount of \$198.01 and two months later you will have an amount of

$$198.01 \times (1 + 0.005)^2 = 200.$$

The present value of both payments is equal to

$$199.0 + 198.01 \approx 397.01$$

dollars.

As a result we obtain that the present value of our payment is equal to

$$200 \times (1 + 0.005)^{-1} + 200 \times (1 + 0.005)^{-2} + \dots + 200 \times (1 + 0.005)^{-60} \approx 10345.11.$$

In other words, the two cash flows from Table 3 are equal.

Cash flow 1		Cash flow 2	
Date	Payment	Date	Payment
Today	10345.11	Today + 1 month	200
		Today + 2 months	200
		...	...
		Today + 60 months	200

Table 3: Two equal more complicated cash flows

Present value enables us to compare different cash flows to see which is preferable. In our case the cash flow consists of equal payments which are paid periodically. We will call such a flow a *fixed cash flow*.

In our first example the cash flow contains only negative values, i.e., you should pay money. In the second example the cash flow contains only positive values, i.e., you receive money. More complicated cash flows can contain both positive and negative values, i.e., you both pay and receive money. Consider another example.

Year 1	\$2000	A cash flow (Table 4) represents the yearly income from an initial investment of \$10,000. The annual interest rate is 8%. How to calculate the present value of this <i>varying</i> cash flow? Let $x_j$ , $0 \leq j \leq 5$ be the sequence of payments ( $x_0 = -10,000$ is the initial investment). The present value is:
Year 2	\$1500	
Year 3	\$3000	
Year 4	\$3800	
Year 5	\$5000	

Table 4: Varying periodic cash flow

$$\sum_{j=0}^5 (1+r)^{-j} x_j \approx 1715.39,$$

where  $r$  denotes the annual interest rate.

The present value can be calculated with the help of the program `present_value` (Fig. 12).

The program `present_value` requires Financial toolbox.

An edit box **Annual interest rate** in the upper part of the dialog contains the value of the simple interest per year. The standard problem has the value  $r = 0.05$ , which corresponds to 5%. You can also change the value of the interest using a slider on the left hand side of the **Annual interest rate** edit box. The minimum slider (and edit box) value is equal to 0%, the maximum slider and edit box value is equal to 10%.

Just below these elements you can see two *frames* that enclose two groups of related controls. Consider the first group on the left side of the dialog. The controls of this group are related to the fixed cash flow.

First consider the functions of the three edit boxes in the upper part of the frame.

- ☞ **Number of months.** For simplicity, in the case of a fixed cash flow our program calculates only cash flows having a period equal to one month. In this edit box, you can enter the number of one-month periods. The standard problem has this value equal to 60.
- ☞ **Month payment.** Here you can enter the amount of money which you plan to pay or obtain monthly. The standard problem has this value equal to 200.

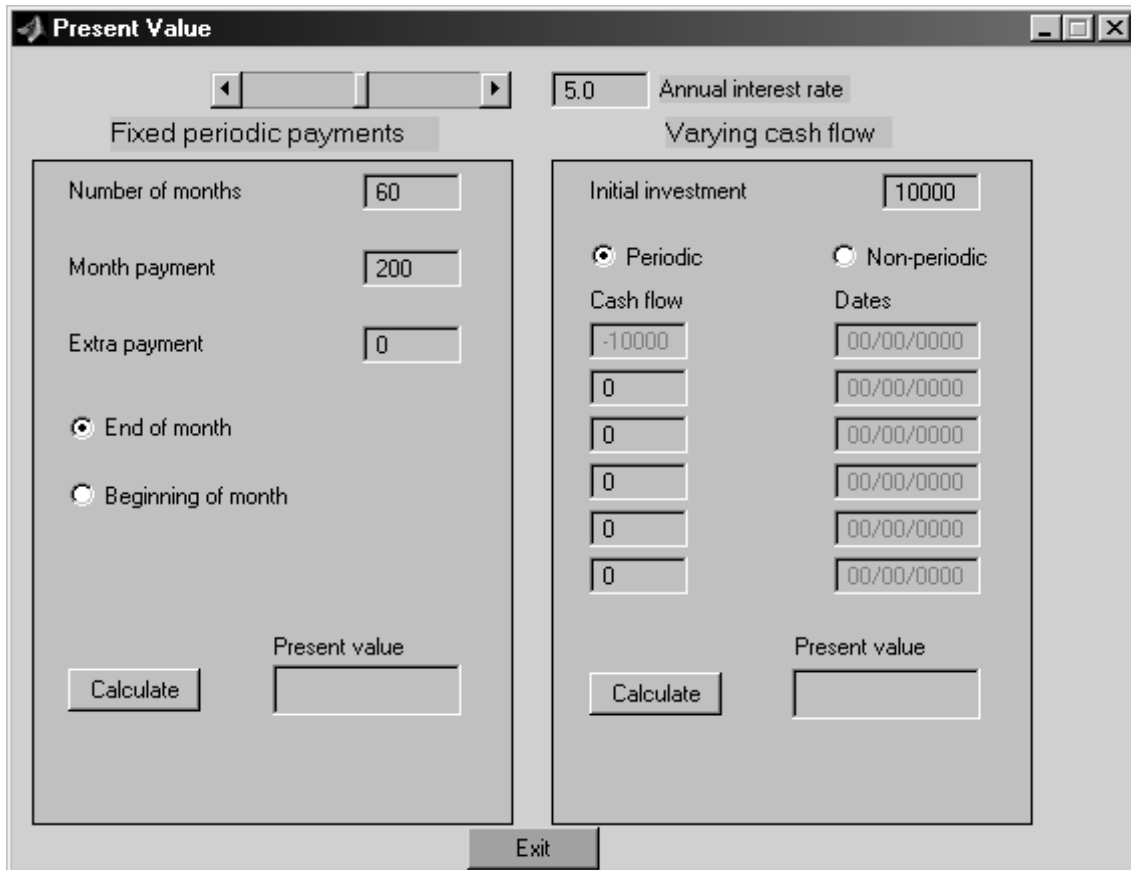


Figure 12: A window of the program `present_value`

☞ **Extra payment.** Some financial institutions propose an extra payment received in the last period. You can enter the value of such a payment in this edit box. The standard problem has this value equal to 0.

Just below the above described edit boxes, you can see a group of two related *radio buttons*. In contrast to checkboxes, only one radio button can be in a selected state at any given time. To activate a radio button, click a mouse button on the object.

Sometimes payment can be paid in the beginning of a period instead of at the end. In this case you can activate the **Beginning of month** radio button. In the standard problem, the radio button **End of month** is active.

The push button **Calculate** below is intended for calculation of the present value in the case of fixed periodic payments, or fixed cash flow. Press it after entering all data of your problem.

The result will appear in the edit box **Present value** (Fig. 13). In contrast to the above described edit boxes, this edit box is *disabled*. The digits inside it are in gray colour. You can not change the value inside, only the program can do this.

Consider, how the program `present_value` solves the example from Table 4 (see Fig. 14).

In this case you should use controls inside the *right* frame. The edit box **Initial investment** contains the value of an initial investment. In the standard problem, this value

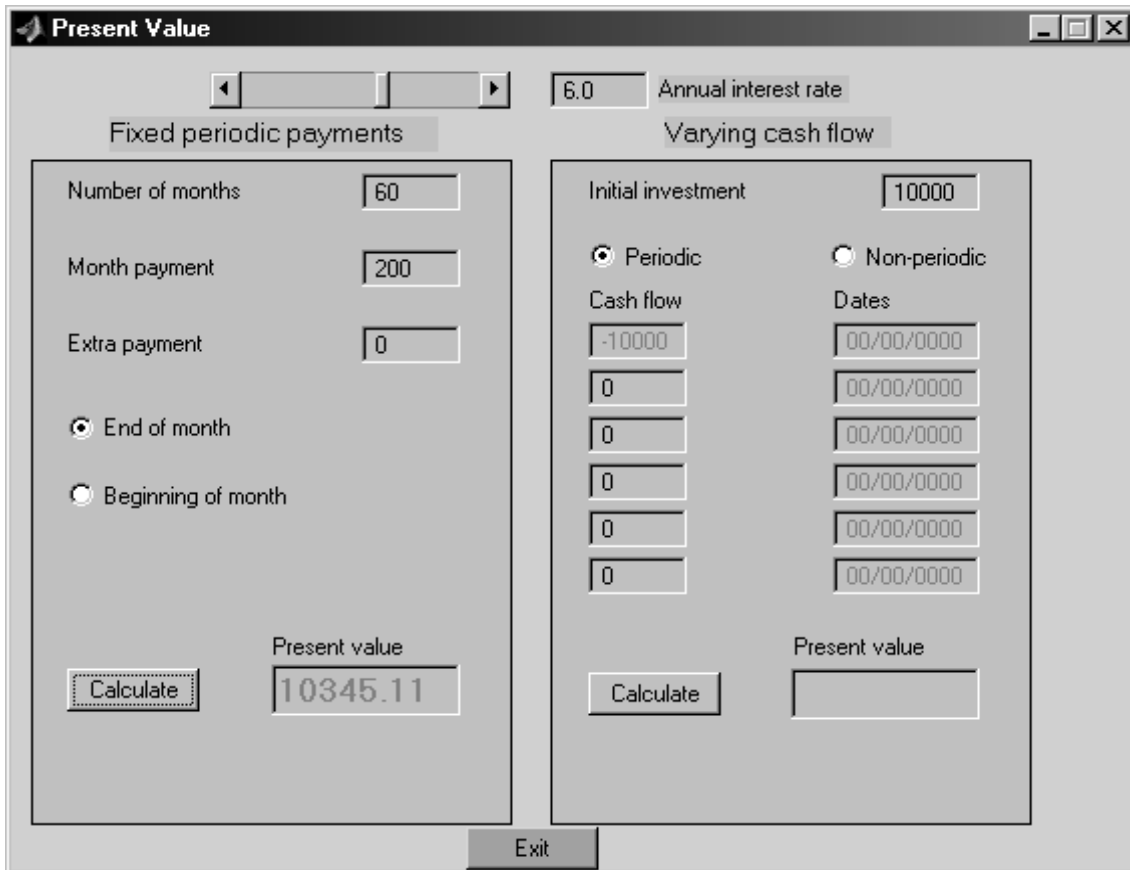


Figure 13: An output of the program `present_value`, case of fixed periodic payments

is equal to 10000.

The group of two related radio buttons below determines one of two different types of a varying cash flow. In our case the cash flow is varying, but periodic. For simplicity, the program calculates only one-year periodic cash flows. Therefore the radio button **Periodic** is active. In our next example we will calculate the present value of a *non-periodic* cash flow. The radio button **Non-periodic** will be active.

The group of six edit boxes under the endorsement **Cash flow** should contain the values of the initial investment and payments. The value of the initial investment is multiplied by  $-1$  and automatically copied from the **Initial investment** edit box to the first edit box of the group. This edit box is disabled. The other five edit boxes are enabled. By default they contain zero values. You should fill one or more of these edit boxes by values of payments, otherwise an error message is generated.

The push button **Calculate** is intended for calculation of the present value of the varying cash flow. After entering data of our example you can press this button. The answer will appear in the disabled edit box **Present value** inside the right frame (Fig. 14).

Consider an example of a varying non-periodic cash flow.

An investment of \$10,000 returns an irregular cash flow (Table 5). The annual interest rate is 9%. Calculate the present value of this cash flow.

In addition to previous notation, let  $t_j$  denotes the time of payments in years. Then the

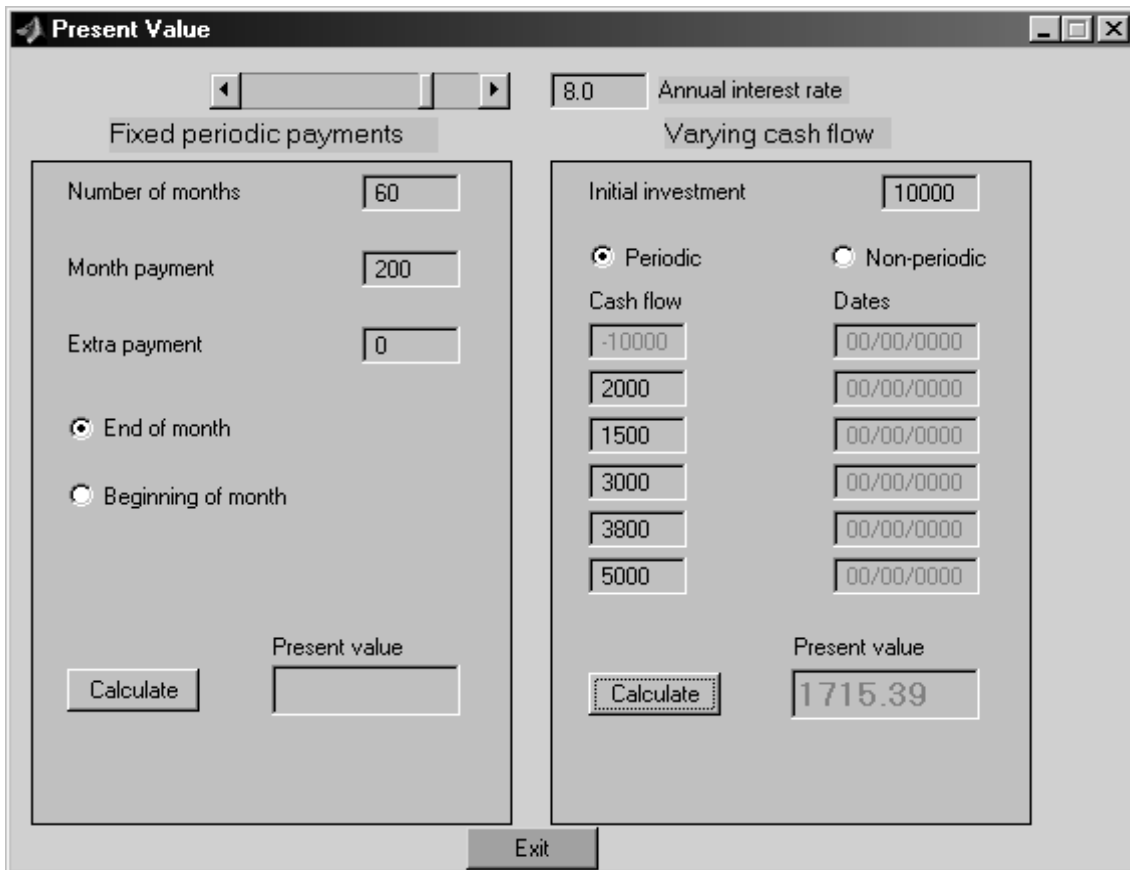


Figure 14: An output of the program present\_value, case of varying periodic payments

Cash flow	Dates
-\$10000	January 12, 1987
\$2500	February 14, 1988
\$2000	March 3, 1988
\$3000	June 14, 1988
\$4000	December 1, 1988

Table 5: Varying non-periodic cash flow

present value is

$$\sum_{j=0}^5 (1+r)^{-(t_j-t_0)} x_j \approx 142.16.$$

In this case, you should work more harder to calculate this expression by hand. Consider, how the program `present_value` solves this example ( see Fig. 15).

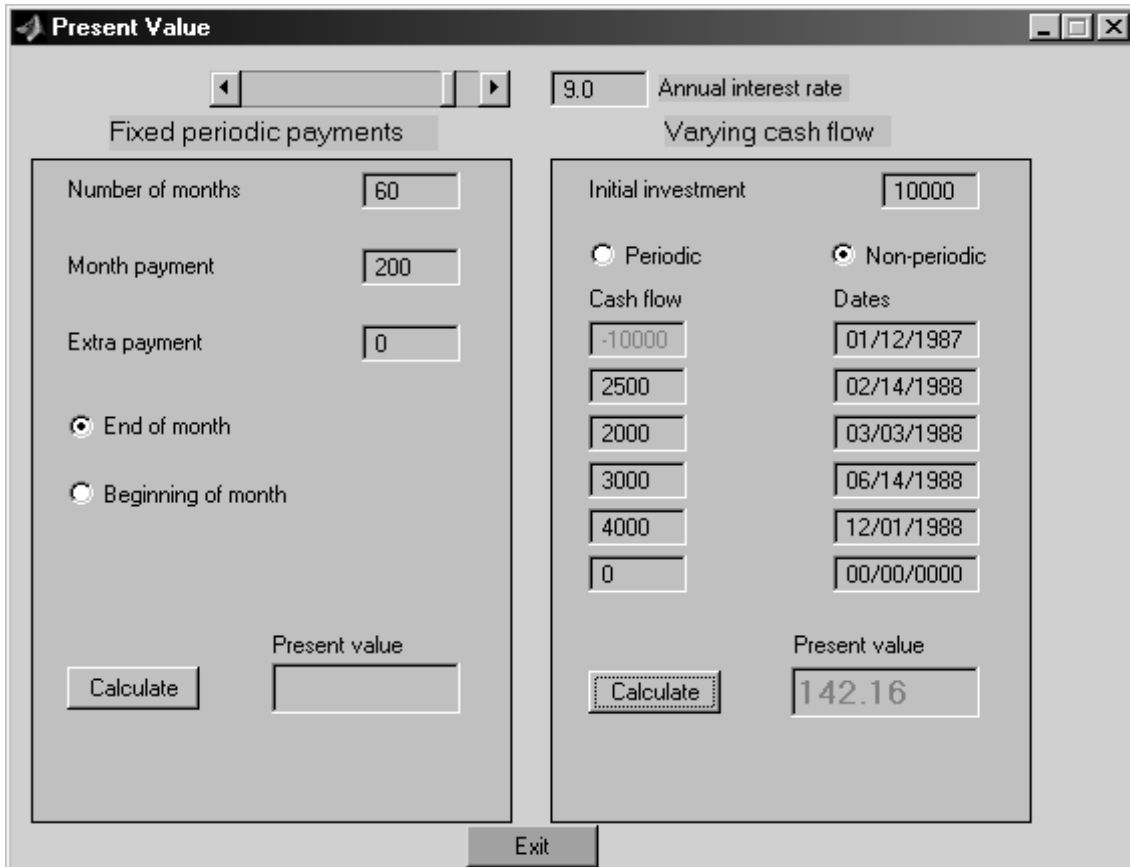


Figure 15: An output of the program `present_value`, case of varying non-periodic payments

As in the previous example, the edit box **Initial investment** contains the value of an initial investment. But now the radio button **Periodic** is not active. The radio button **Non-periodic** is active instead.

The group of six edit boxes **Cash flow** still contains values of the cash flow under consideration. Another group of six edit boxes, **Dates**, becomes enabled. You must enter dates of the initial investment and the values of all payments in these edit boxes.



*Remark 5.* Unfortunately, MATLAB follows the American convention for the format of dates. The string “12/01/1988” means December 1, *not* January 12!



*Remark 6.* Be careful when enter dates. The corresponding edit boxes do not control its input.

Finally, the push button **Exit** stops the program.

*Remark 7.* The program `present_value` has a limitation. You can not calculate the present value of a cash flow with 6 or more payments. For calculation with such flows you can use MATLAB Command Window directly. See Section A.1.

### 3.2.1 Problems

**Problem 2.** \$150 is paid monthly into a saving account earning 4%. The payments are made at the end of the month for ten years. What is the present value of these payments?

**Problem 3.** \$250 is paid monthly into a saving account earning 5%. The payments are made at the beginning of the month for seven years. What is the present worth of these payments?

**Problem 4.** A cash flow (Table 6) represents the yearly income from an initial investment of \$4,400. The annual interest rate is 6%. Calculate the present value of this cash flow.

**Problem 5.** An investment of \$10,000 returns an irregular cash flow (Table 7). The annual interest rate is 9%. Calculate the present value of this cash flow.

Dates	Cash flow
Initial	-\$4,400
Year 1	\$800
Year 2	\$800
Year 3	\$1800
Year 4	\$800
Year 5	\$1400

Table 6: Cash flow, problem 4

Cash flow	Dates
-\$10,000	January 1, 2002
\$800	March 10, 2003
\$800	April 14, 2004
\$1800	June 26, 2005
\$800	August 31, 2006
\$1400	November 15, 2007

Table 7: Cash flow, problem 5

### 3.3 Rate of return

Initial	-\$100,000
Year 1	\$10,000
Year 2	\$20,000
Year 3	\$30,000
Year 4	\$40,000
Year 5	\$50,000

Table 8: The yearly income from an initial investment of \$100,000

Consider the next example. Some financial organisation proposed you to make an initial investment of \$100,000. They promised that you will obtain the sequence of yearly incomes shown in Table 8.

Another financial organisation proposed you to put the same amount to the bank and to obtain the yearly interest rate  $r$ . What proposition is better?

To solve this problem, we must calculate the present value of the cash flow defined by an initial investment of \$100,000, the incomes in Table 8 and the yearly interest rate  $r$ . Three possibilities can happen:

1. The present value is less than zero.
2. The present value is equal to zero.
3. The present value is greater than zero.

In the first case the initial investment exceeds the total of the amounts received. Therefore we loose money under the conditions of the first proposition, and the second proposition is better.

In the third case the total of the amounts received exceeds the initial investment. Therefore we obtain a gain under the conditions of the first proposition, and the first proposition is better.

In the second case, however, the propositions are equivalent.

The *rate of return* of the investment can be defined as the interest rate  $r$  that makes the present value of the cash flow defined by an initial investment and the payments equal to zero.

*Remark 8.* It is easy to see that this definition is the same as a definition in [4, p. 51]. We prefer this kind of definition, because in MATLAB the value of an initial investment, multiplied by  $-1$ , should be the first element of the vector representing the cash flow.

Let  $b_0, b_1, \dots, b_n$  denote the periodic cash flow sequence, in which  $b_0 < 0$  denotes the initial investment. If the interest rate per one period is equal to  $r$ , then the present value of this cash flow is equal to

$$P(r) = \sum_{j=0}^n b_j(1+r)^{-j}.$$

By definition, the rate of return per period of the investment is that value  $r^* > -1$ , for which

$$P(r^*) = 0. \tag{5}$$

In our case  $n = 5$ , and the rate of return should be determined numerically. Consider, how the program `ror` (Fig. 16) solves this problem.

The program `ror` requires Financial toolbox.

The two radio buttons in the upper part of the dialogue determine the type of cash flow under consideration. If the radio button **Yearly cash flow** is active, then we consider (for simplicity) a cash flow having the period equal to one year. If the radio button **Non-periodic cash flow** is active, then the cash flow is irregular.



The group of six edit boxes under the radio button **Yearly cash flow** should contain the values of the initial investment and payments. The first edit box should contain the value of the initial investment, multiplied by  $-1$ . By default, the other five edit boxes contain zero values. You should fill one or more of these edit boxes by values of payments, otherwise an error message is generated.

The push button **Calculate** in the lower left corner calculates the rate of return. Press it after you have entered the values of the initial investment and payments. The result will



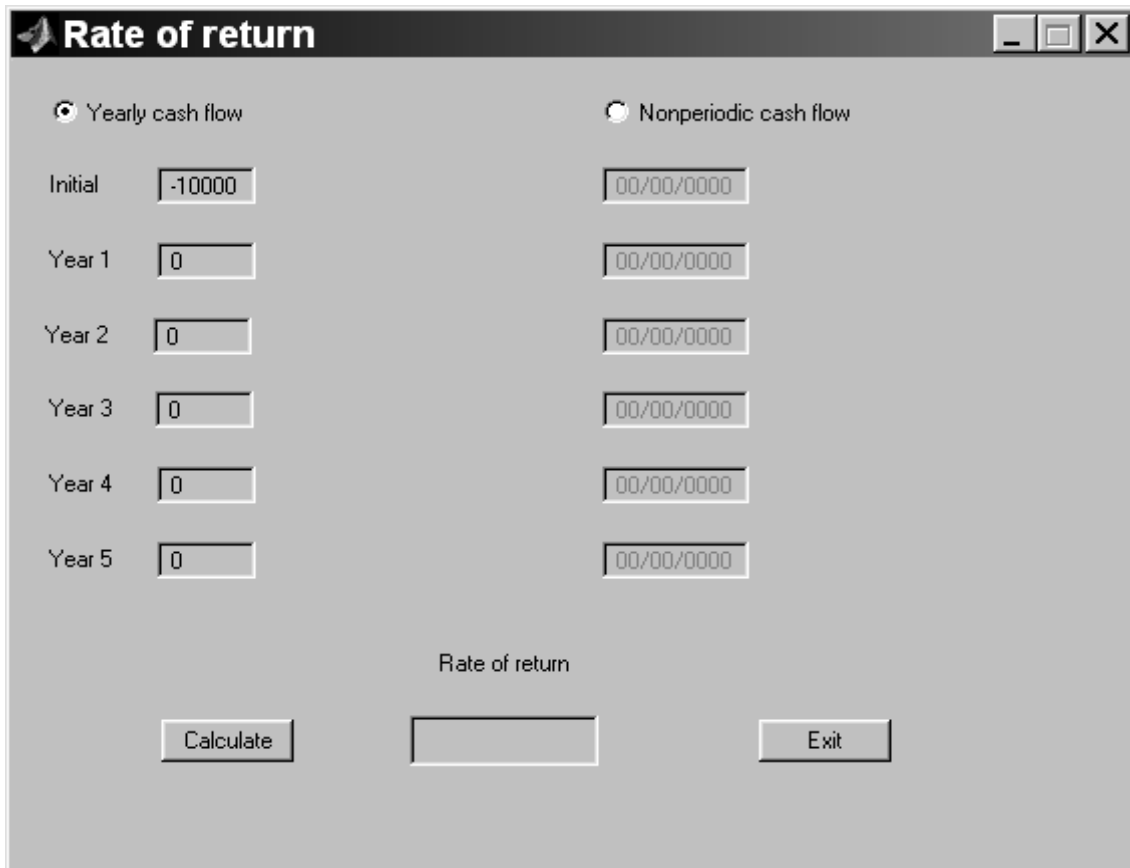


Figure 16: A window of the program ror

appear in the disabled edit box **Rate of return**. In our case we entered the value of an initial investment, the values of payments from Table 8 and obtained the result (Fig. 17).

$$r^* \approx 0,1201 \text{ (12.01\%)}$$

You can quit the program ror by pressing the push button **Exit** in the lower right corner of the dialog.

Consider a more complicated example. An investment of \$10,000 returns non-periodic cash flow shown in Table 9. Calculate the rate of return for this non-periodic cash flow.

	<b>Cash flow</b>	<b>Dates</b>
$b_0$	-\$10000	January 12, 2000
$b_1$	\$2500	February 14, 2001
$b_2$	\$2000	March 3, 2001
$b_3$	\$3000	June 14, 2001
$b_4$	\$4000	December 1, 2001

Table 9: The non-periodic cash flow from an initial investment of \$10,000

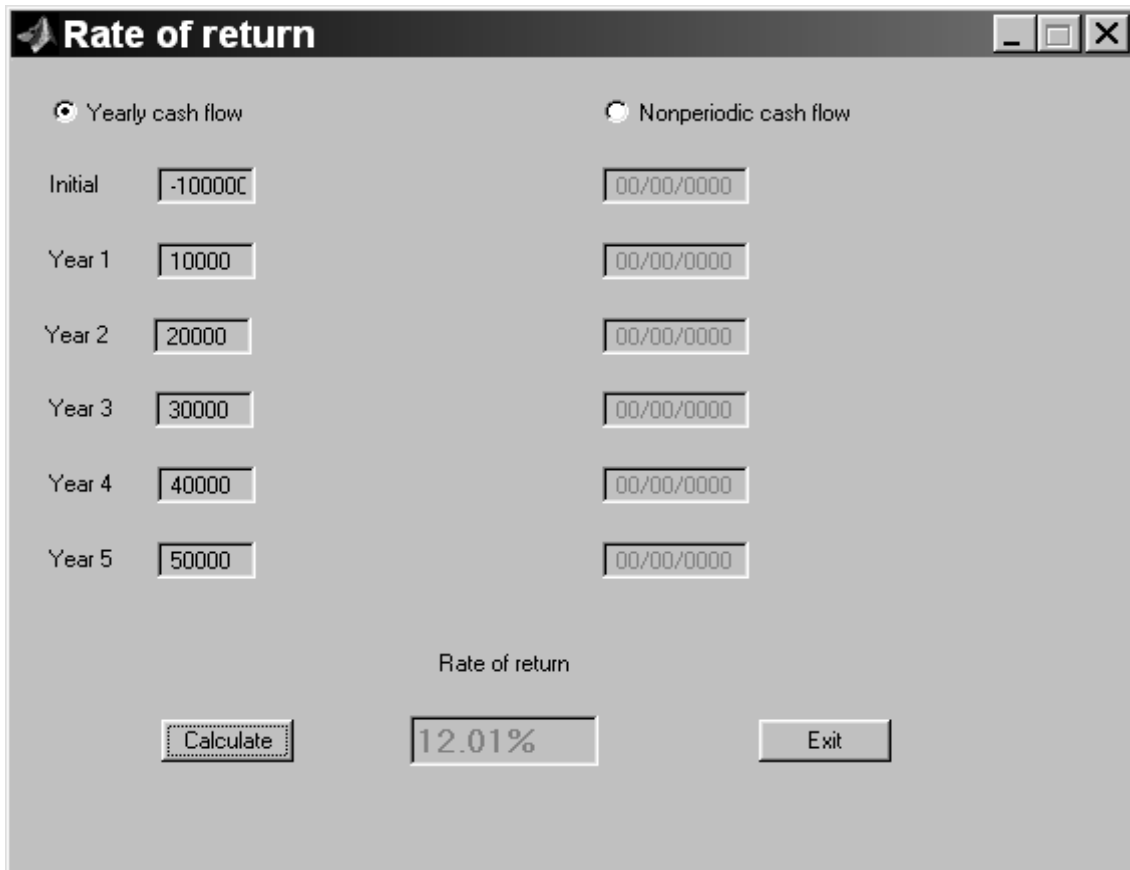


Figure 17: An output of the program `ror`, case of periodic payments

Let  $t_0$  denotes the date of the initial investment (measured in years A.D.). For example, in our case  $t_0 = 1999 + \frac{12}{366} \approx 1999.0328$ . Denote by  $t_j$ ,  $1 \leq j \leq 4$  the dates of payments. Let  $b_0$  denotes the initial investment, multiplied by  $-1$ , and  $b_j$ ,  $1 \leq j \leq 4$  denote payments. Then the rate of return  $r^*$  should be equal to the root of the equation

$$\sum_{j=0}^4 b_j (1+r)^{-(t_j-t_0)} = 0.$$

This equation is much more complicated than the equation (5). Consider its solving with the help of the program `ror` (Fig. 18).

First, we activate the radio button **Non-periodic cash flow**. Second, we enter the value of an initial investment, multiplied by  $-1$ , into the first edit box of the left column. Third, we entered values of payments into next edit boxes of the left column. Now, the edit boxes of the right column are enabled (in the previous example they were disabled). We entered dates of the initial investment and payments into the edit boxes of the right column. After pressing the push button **Calculate**, we obtained the result

$$r^* \approx 0.1009 \text{ (10.09\%)}.$$

*Remark 9.* The program `ror` has a limitation. You can not calculate the present value of a cash flow with 6 or more payments. For calculation with such flows you can use MATLAB Command Window directly. See Section A.2.

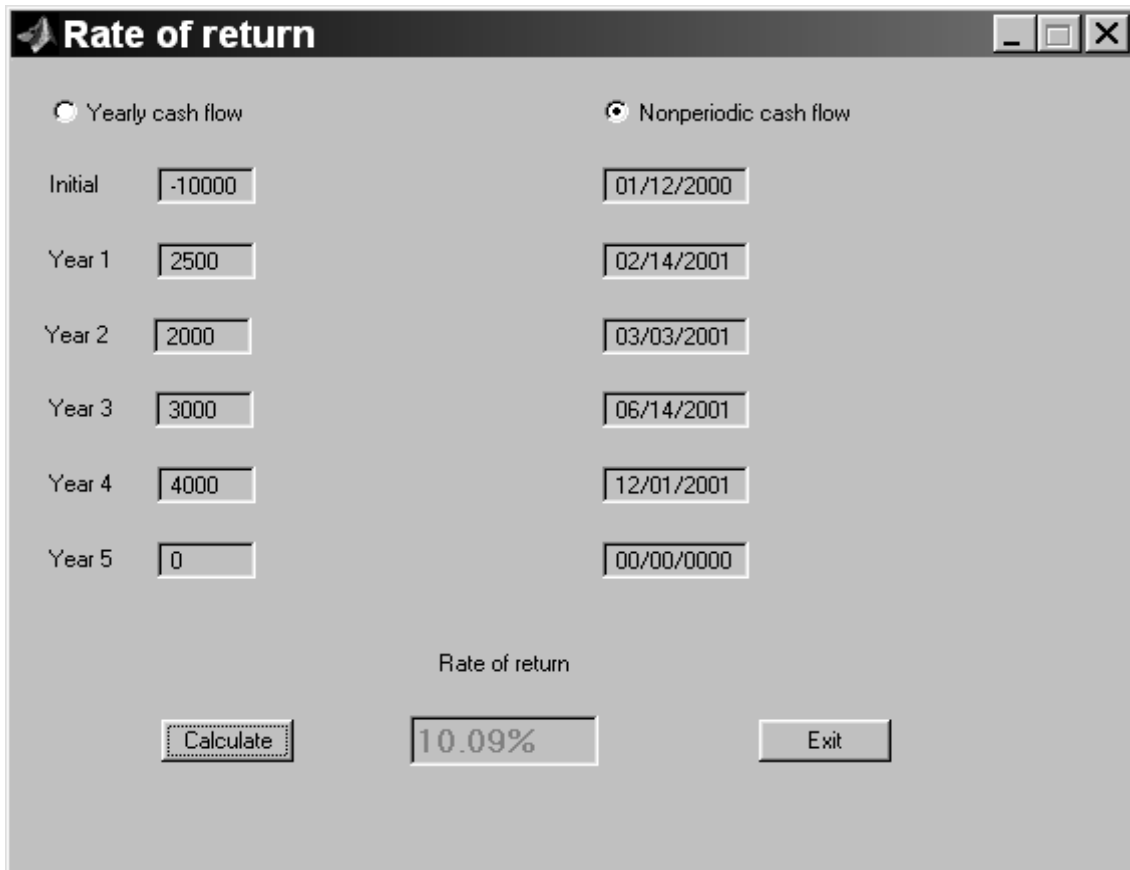


Figure 18: An output of the program ror, case of non-periodic payments

### 3.3.1 Problems

**Problem 6.** The initial investment of \$4,400 returns the yearly cash flow shown in Table 6. You can both borrow and save money at the yearly interest rate of 6%. Is this a worthwhile investment for you?

**Problem 7.** The initial investment of \$10,000 returns the irregular cash flow shown in Table 7. You can both borrow and save money at the yearly interest rate of 7%. Is this a worthwhile investment for you?

## 3.4 Pricing via arbitrage

Recall that an *option* gives the buyer the right, but not the obligation, to buy or sell a security under specified terms. An option that gives the right to buy is called a *call option*. An option that gives the right to sell is called a *put option*. Consider the example of a call option.

Suppose that the nominal interest rate per time period is  $r$ . Let the present price of the security be \$100 per share. After one time period it will be either \$200 or \$50 (fig. 19). In what follows, we will refer to these possible outcomes as *states of nature*. You can think about the two states of nature as collections of circumstances which will cause the

price of the security to be as above. At the present time, it is not known which state will be realised after one time period. It is known only that one (and only one) of these states will occur. There is no assumption made about the probability of each state's occurrence, except that each state has a positive probability of occurrence. In this model, the states capture the uncertainty about the price of the security after one time period.

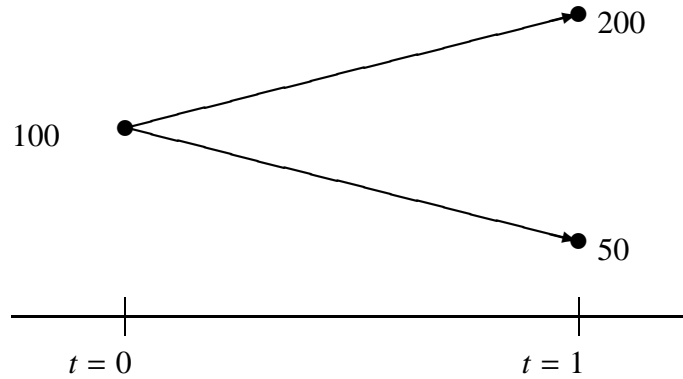


Figure 19: Possible security prices at time 1

For any  $y$ , at a cost of  $cy$  you can purchase at time 0 the  $y$  call options to buy  $y$  shares of the stock at time 1 for the price of \$150 per share. In addition, you can purchase  $x$  shares of the security at time 0. For what values of  $c$  exists an arbitrage possibility?

Recall that an *arbitrage* is a sure-win betting scheme [4, p. 63]. The vector  $(x, y)$  is called a *portfolio*. In our case the portfolio consists of the security and the options.

*Step 1.* We choose  $y$  so that the value of our portfolio at time 1 does not depend on the state of nature. In the first state of nature, when the price of security at time 1 is \$200 per share, the  $x$  shares of the security are worth  $200x$  and the  $y$  units of options to buy the security at a share price of \$150 are worth  $(200 - 150)y = 50y$ . Therefore the value of our portfolio at time 1 is equal to  $200x + 50y$ .

On the other hand, in the second state of nature, when the price of security at time 1 is \$50 per share, then the  $x$  shares are worth  $50x$  and the  $y$  units of options are worthless. Therefore the value of our portfolio at time 1 is equal to  $50x$ .

That is, we choose  $y$  so that

$$200x + 50y = 50x, \quad \text{or} \quad y = -3x,$$

and the value of our portfolio at time 1 is equal to  $50x$  no matter what is the state of nature.

*Step 2.* At time 0 we purchase  $x$  units of security and  $-3x$  units of options. The cost of this transaction is

$$100x + cy = (100 - 3c)x.$$

If the cost of the transaction is positive, i.e.,  $(100 - 3c)x > 0$ , then it should be borrowed from a bank, to be repaid with interest  $r$  at time 1. Therefore our gain is equal to

$$\begin{aligned} \text{gain} &= 50x - (100 - 3c)x(1 + r) \\ &= (1 + r)x[3c - 100 + 50(1 + r)^{-1}]. \end{aligned} \tag{6}$$

On the other hand, if the cost of transaction is negative, then the amount received,  $-(100 - 3c)x$ , should be put in the bank to be withdrawn at time 1. Therefore our gain is determined by (6) no matter of the sign of transaction.

Thus, if  $3c = 100 - 50(1 + r)^{-1}$ , then the gain is zero. Otherwise we can **guarantee a free lunch** (no matter what the price of the security at time 1). Indeed, suppose that  $r = 0,05$  or 5%. Consider two cases.

*Case 1.*  $3c < 100 - 50(1 + r)^{-1}$ , for example,  $c = 15$ . The option is too cheap. At time 0 we sell *in short* one share of the security ( $x = -1$ ) and obtain \$100. Selling in short means that we sell a security that we do not own. We buy 3 options ( $y = 3$ ) at a cost of \$45 and put the amount \$55 in the bank. At time 1 we withdraw the amount  $55(1 + r) = \$57.75$  from the bank.

In the first state of nature the stock's price is \$200. We exercise our options, buy 3 shares of the security at a cost of \$450, return one share which we borrowed at time 0 and sell 2 shares at a cost of \$400. Our gain is  $57.75 - 50 = \$7.75$  and we go to have our free lunch.

In the second state of nature the stock's price is \$50. We do not exercise our options. Instead we buy 1 share of the security and return it to the owner. Our gain is  $57.75 - 50 = \$7.75$  and we go to have our free lunch.

*Case 2.*  $3c > 100 - 50(1 + r)^{-1}$ , for example,  $c = 20$ . The option is too expensive. At time 0 we borrow from a bank \$40. We sell in short 3 options at a cost of \$60 ( $y = -3$ ) and buy 1 share of the security ( $x = 1$ ).

In the first state of nature the stock's price is \$200. The options' owner realises the options. We are obliged to buy 3 shares at a cost of \$600 and sell them to the options' owner at a cost of 450. Then we sell our share at a cost of \$200. The amount earned is \$50, but we have to return the loan  $40 \times (1 + 0.05) = \$42$  to the bank. Our gain is  $50 - 42 = \$8$  and we go to have our free lunch.

In the second state of nature the stock's price is \$50. The options' owner does not realise the options. We sell our share for \$50, return the loan of \$42 and go to have our free lunch.

The model under consideration contains one time period and only two possible outcomes. Therefore sometimes it is called a *one-period binomial model*.

The program `options_pricing` performs these calculations (Fig. 20).

The program `options_pricing` requires only standard MATLAB installation (no additional toolboxes are used).

The window of the program `options_pricing` can be considered as consisting of the left and right hand sides. The left hand side called **Problem** contains user interface controls for formulation of the problem. The solution of the problem appears in the user interface controls on the right hand side of the window called **Solution**.

Consider the user interface controls on the left hand side first. The slider and edit box **Period interest rate** in the upper left corner contain value of the interest rate per period under consideration. Standard problem has a value  $r = 0.05$ , which corresponds to 5%. The minimum slider (and edit box) value is equal to 0%, the maximum slider and edit box value is equal to 10%.

The frame under the above described group contains three edit boxes.

☞ **Now.** This edit box contains the initial price of one share of the security. The standard problem has this value equal to 100.

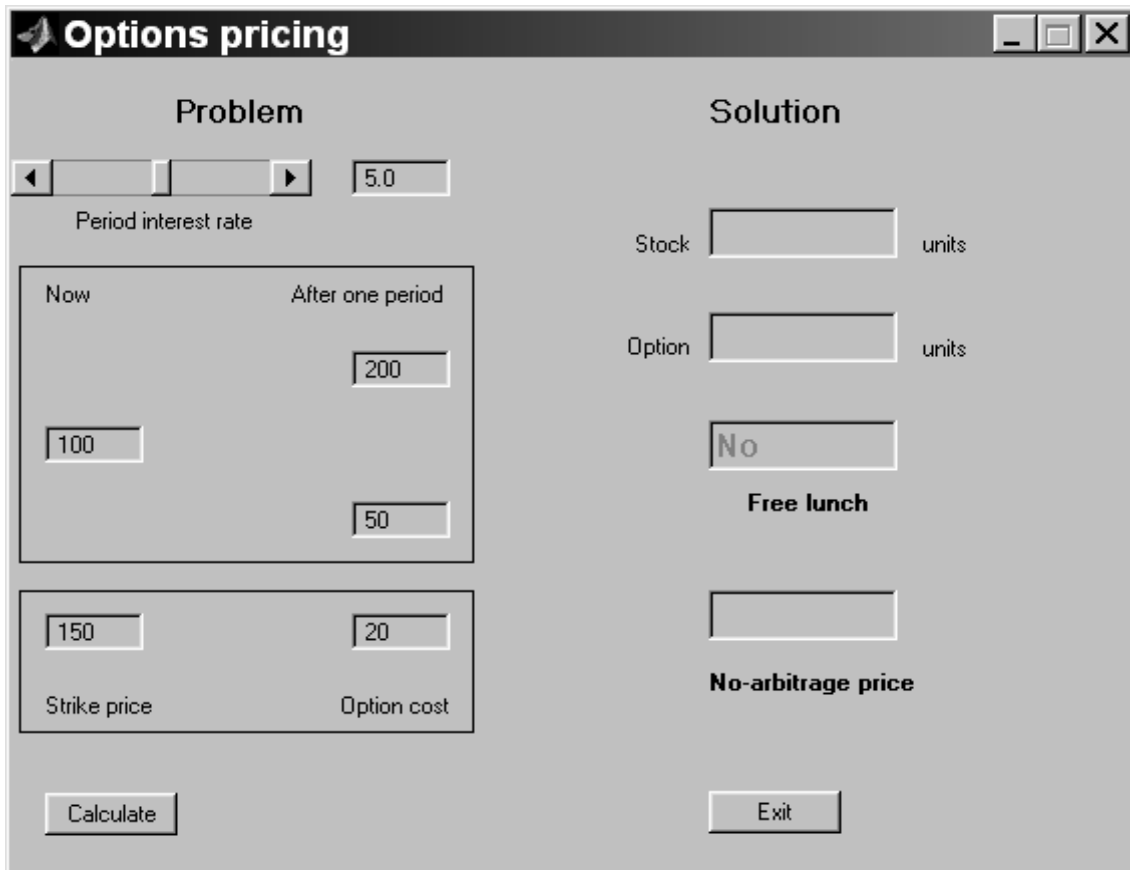


Figure 20: The window of the program options\_pricing

☞ **After one period.** Two edit boxes under this caption contain possible values of the price of one share of the security one time period later. The standard problem has values 200 and 50.

The next frame contains two edit boxes.

☞ **Strike price.** This edit box contains value of the strike price of the option. The standard problem has this value equal to 150.

☞ **Option cost.** This edit box contains the price of the option. The standard problem has this value equal to 20.

The push button **Calculate** calculates the arbitrage possibility. Consider the results of solution of the standard problem in the right hand side of the program window (Fig. 21). Four disabled edit boxes are in the right hand side of the program window.

☞ **Stock ... units.** The edit box enclosed by these captions contains the number of stocks in our portfolio. In the case of too expensive option it always contains the value 1. Of course, the player can always multiply all values in the edit boxes on the right hand side of the window by any positive number.

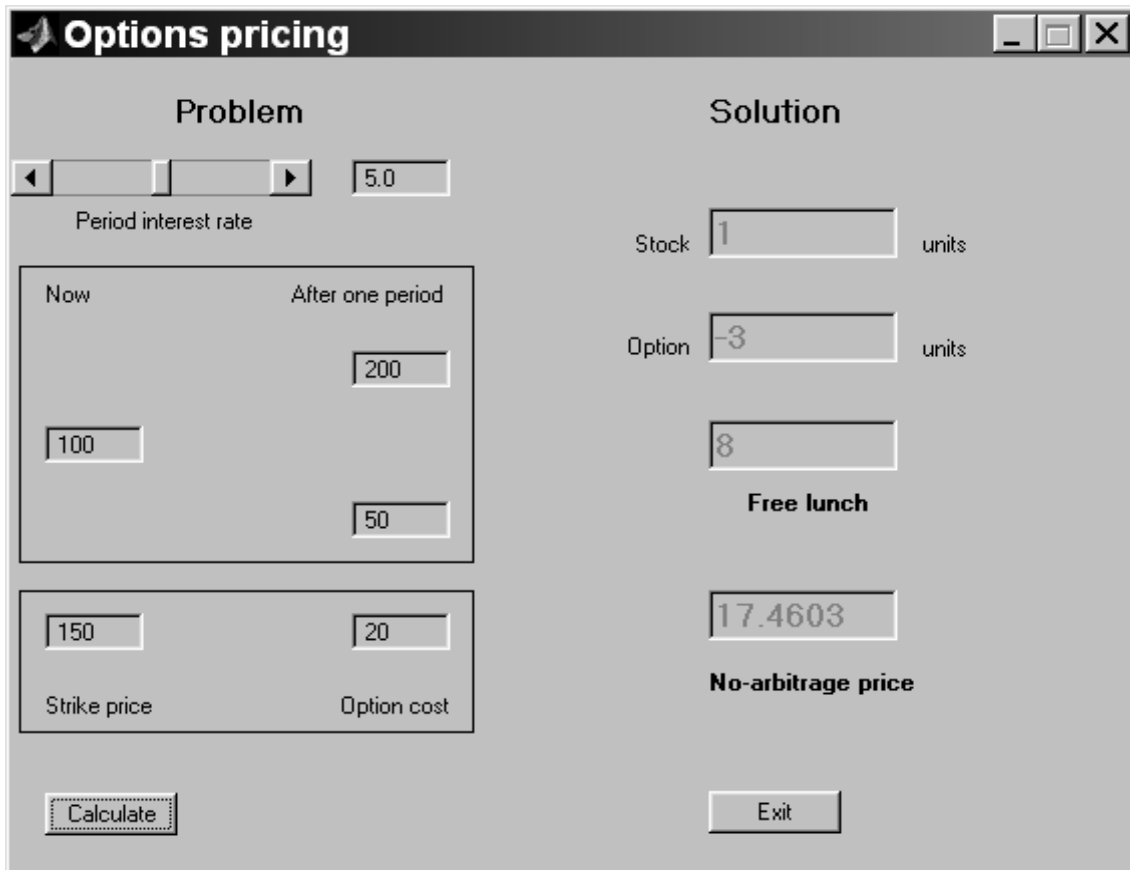


Figure 21: An output of the program `options_pricing`, case of too expensive option

- ☞ **Option ... units.** The edit box enclosed by these captions contains the number of options in our portfolio. In our example, this value is negative. It means, that we should sell options in short.
- ☞ **Free lunch.** This edit box contains the value of our gain.
- ☞ **No-arbitrage price.** This edit box contains the value of the only option cost that does not result in an arbitrage. This price is called *no-arbitrage* or *risk-neutral* price. *Pricing* of an option means the calculation of its no-arbitrage or risk-neutral price.

Finally, the push button **Exit** stops the program.

Consider the case of a too cheap option (Fig. 22). We changed the value in the edit box **Option cost** only. In this case the first edit box always contains the value  $-1$ , It means that the player should sell one share of the security in short. The second edit box contains the value 3, i.e., we buy three options. The value of our gain and the non-arbitrage price of the option are contained in the third and fourth edit box respectively.

We can check how the program works in the following way. Substitute the value of the no-arbitrage price into the edit box **Option cost** (Fig. 23). The first two edit boxes do not contain any values. It means that no portfolio can bring a positive gain. The edit box

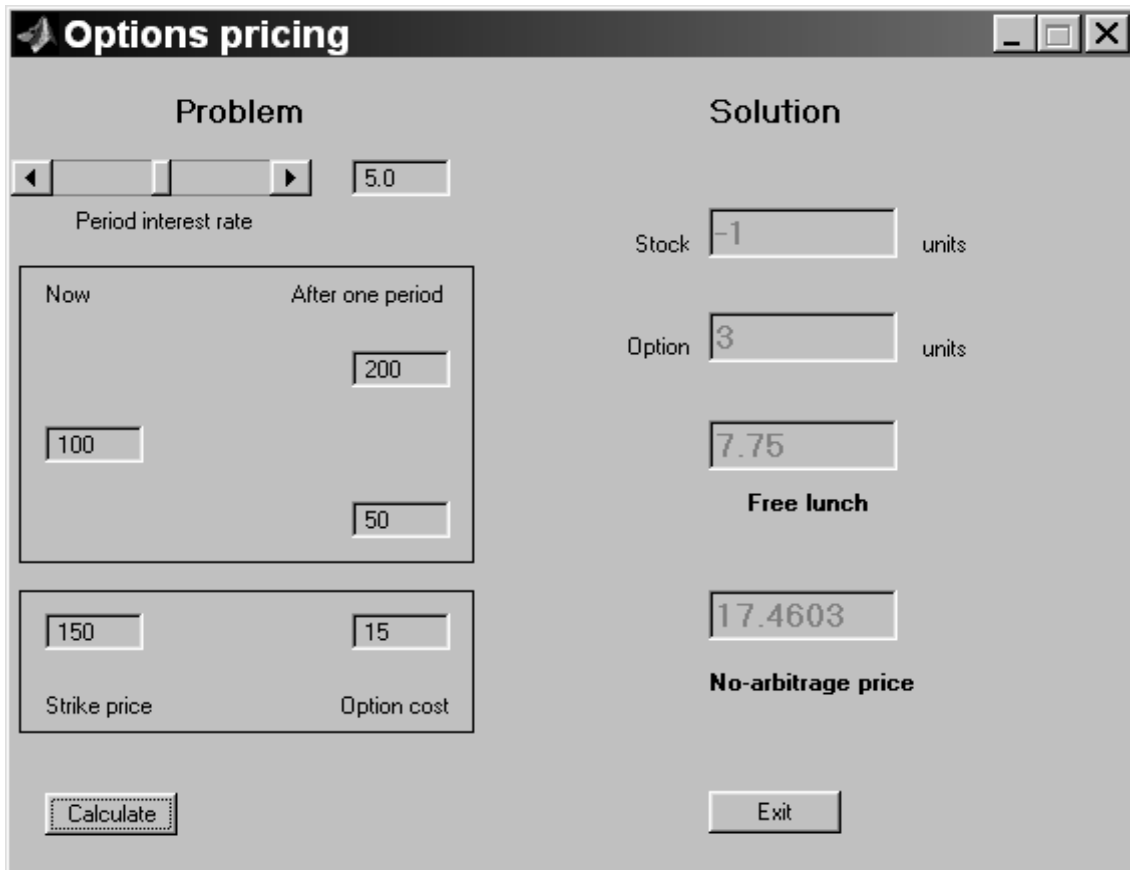


Figure 22: An output of the program `options_pricing`, case of too cheap option

**Free lunch** contains the word **No**. As in all the previous cases, the edit box **No-arbitrage price** contains the corresponding value.

### 3.4.1 Problems

**Problem 8.** The nominal interest rate per time period is  $r$ . The present price of the security is  $P_0$  dollars per share. After one time period it will be either  $P_1$  or  $P_2$  dollars per share. At a cost of  $c$  dollars you can purchase at time 0 the call option to buy one share of the stock at time 1 for the price of  $S$  dollars per share.

Describe the sure-win betting scheme for the cases from Table 10. Find the no-arbitrage price of the call option for every case.

## 3.5 The multi-period binomial model

In Section 3.4 we supposed that there is only one period of time. Consider the case when there are  $n$  periods.

Consider a five-month ( $t = 0.4167$ ) American put option when the initial price of the non-dividend paying stock is  $S(0) = \$50$ , the strike price is  $K = \$50$ , the risk-free interest is 10% per annum, and the volatility is 40% per annum ( $\sigma = 0.4$ ). Divide the life of the



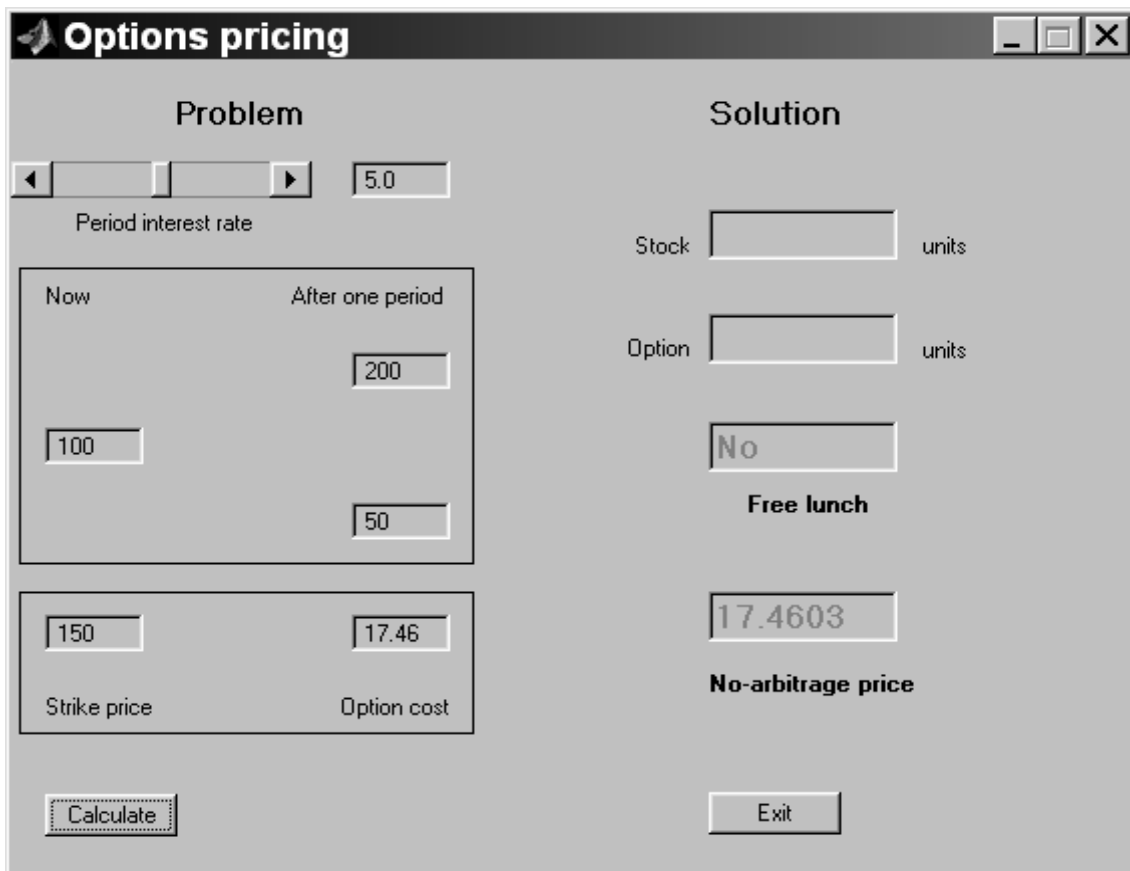


Figure 23: An output of the program options\_pricing, case of no-arbitrage price

$r$	$P_0$	$P_1$	$P_2$	$c$	$S$
5	60	50	80	9	60
3	105	90	130	9	110
1	108	92	115	4	110
4	113	108	115	7	113
9	65	20	90	8	80
2	70	60	100	5	80
6	120	75	165	21	125
8	100	60	200	26	115

Table 10: One-period binomial models

option into  $n = 5$  equal periods of length  $t/n$ . Suppose that the price of a security can change only at the times  $t_k = kt/n$ ,  $k = 1, 2, \dots, n$  and that the option can be exercised only at one of the times  $t_k$ . Moreover, suppose that the security price  $S(k+1)$  at  $k+1$  time periods later is either  $uS(k)$  or  $dS(k)$ . How to find the risk-neutral price of this option?

Recall that *American* option can be exercised at any time up to expiration time, whereas *European* option can be exercised **only** at the expiration time.

In contrast to the one-period binomial model described in Section 3.4, the model under consideration is called a *multi-period binomial model*.

We want the above described process to approximate the geometrical Brownian motion when  $n$  grows. According to (4) that happens if

$$u = e^{\sigma\sqrt{t/n}}, \quad d = e^{-\sigma\sqrt{t/n}}. \quad (7)$$

According to [4, p. 92]

$$\begin{aligned} P\{S(k+1) = uS(k)\} &= p = \frac{1 + rt/n - d}{u - d}, \\ P\{S(k+1) = dS(k)\} &= 1 - p = \frac{u - 1 - rt/n}{u - d}. \end{aligned} \quad (8)$$

According to [4, formulae (7.11) and (7.12)], the possible values of the price of the put option at time  $t_n$  is equal to

$$V_n(j) = \max\{K - u^j d^{n-j} S(0), 0\}, \quad (9)$$

if  $j$  of the first  $n$  price movements were increases and  $n - j$  were decreases. The possible values of the price of the put option at time  $t_k$ ,  $k = n - 1, n - 2, \dots, 0$  are calculated as

$$V_k(j) = \max\{K - u^j d^{k-j} S(0), \beta p V_{k+1}(j+1) + \beta(1-p)V_{k+1}(j)\}, \quad (10)$$

if there were  $j = 0, \dots, k$  increases and  $k - j$  decreases. The first term in figure brackets of (10) denotes the return if we exercise the option in moment  $t_k$  at node  $j$ . The second term denotes the return if we do not exercise the option in moment  $t_k$  at node  $j$ .  $\beta$  denotes the *discount factor* per period. According to [4, p. 94] it is equal to

$$\beta = e^{-rt/n}. \quad (11)$$

Using these formulae, we obtain

$$V_0(0) \approx 4.488.$$

Calculations by hand using these formulae can be computationally messy. Consider how the MATLAB program `mbm` (Fig. 24) solves this problem.

The program `mbm` requires Financial toolbox.

The frame in the upper part of the window of the program `mbm` contains two sliders, five edit boxes and a pair of mutually exclusive radio buttons. Consider the functions of these controls.

☞ **Price.** This edit box contains the value of the initial price of the stock,  $S(0)$ . The standard problem has the value  $S(0) = 50$ .

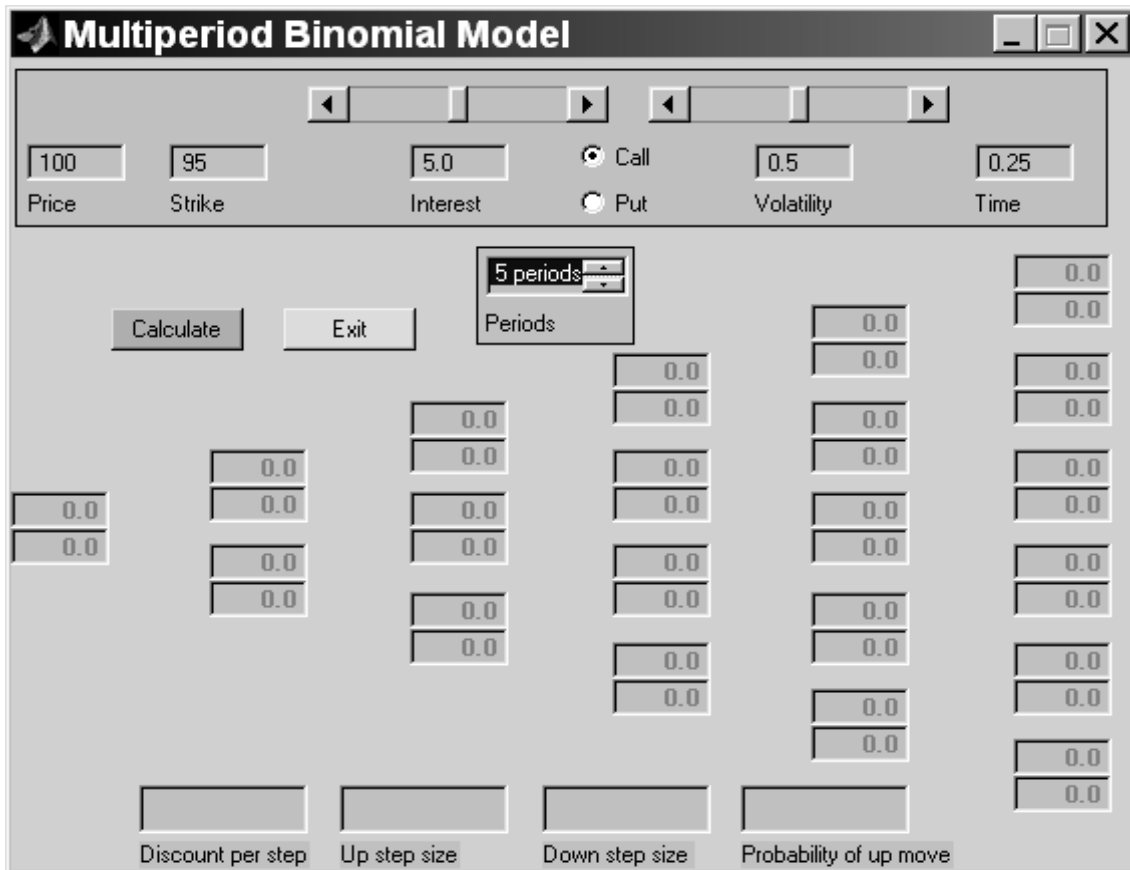


Figure 24: A window of the program mbm

- ☞ **Strike.** This edit box contains the value of the strike price of the security,  $K$ . The standard problem has the value  $K = 50$ .
- ☞ **Interest.** This edit box and the slider over it are responsible for the value of the annual risk-free interest rate,  $r$ . The standard problem has the value  $r = 0.05$ , which corresponds to 5%. The minimum slider (and edit box) value is equal to 0%, the maximum slider and edit box value is equal to 10%.
- ☞ **Volatility** This edit box and the slider over it are responsible for the value of the annualised volatility,  $\sigma$ . The standard problem has the value  $\sigma = 0.5$ , which corresponds to 50%. The minimum slider (and edit box) value is equal to 0%, the maximum slider and edit box value is equal to 100%.
- ☞ **Time.** This edit box contains the value of the length of life of the option,  $t$ , measured in years. The standard problem has the value  $t = 0.25$  or 3 months.

The mutually exclusive radio buttons **Call** and **Put** define the type of the option. The standard problem considers a call option.

The list box **Periods** just under the frame contains five elements. This control determines the value of  $n$ . In our model, the possible values are  $1 \leq n \leq 5$ . The standard problem has the value  $n = 5$ .

The push buttons perform the next functions.

- ☞ **Calculate.** Calculates the price of the option, using formulae (7)–(10).
- ☞ **Exit.** Stops the program.

The result of calculations is shown on Fig. 25. The group of 42 disabled edit boxes fills the triangle shape and represents the binomial tree. This tree has 21 nodes. Every node consists of two edit boxes.

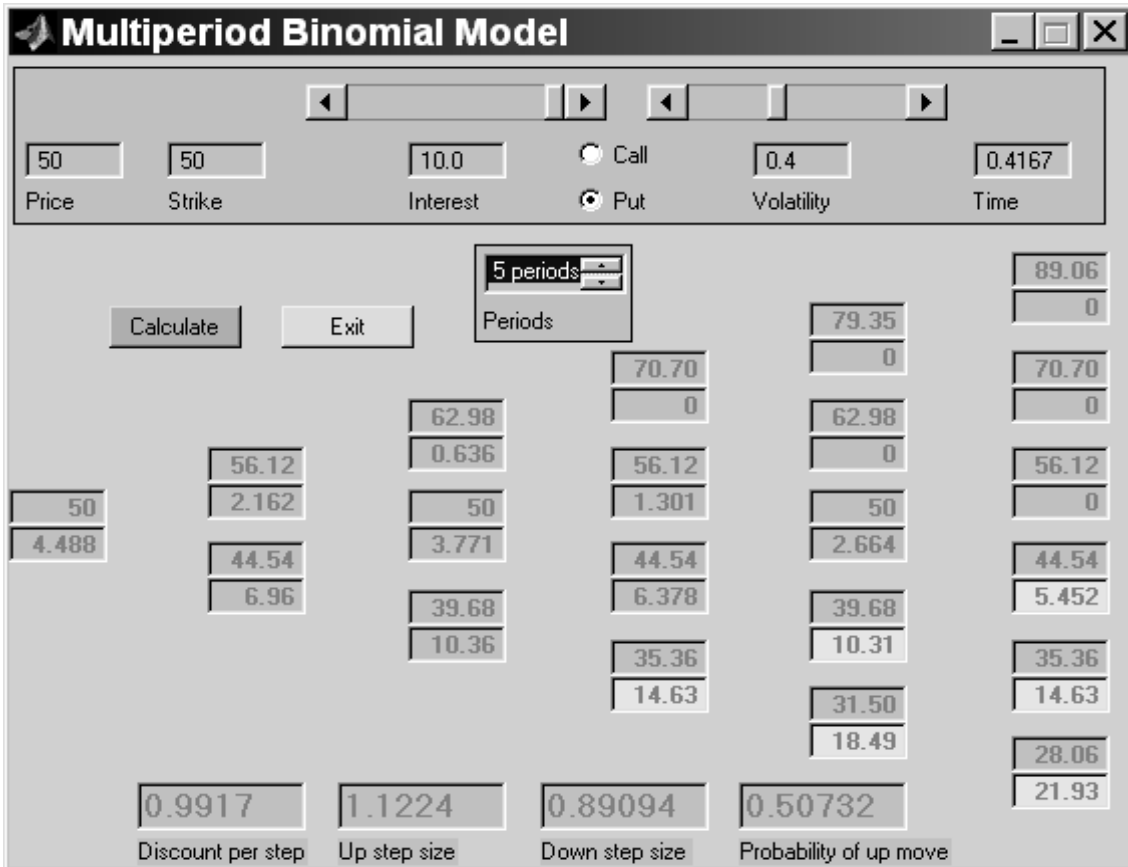


Figure 25: An output of the program mbm, case of put option

The upper edit boxes of the  $k$ th column ( $k = 0, 1, \dots, n$ ) shows all the possible values of the price of the security at time  $t_k$ . The price of the security at the  $j$ th node ( $j = 0, 1, \dots, k$ ) is calculates as  $S(0)u^j d^{k-j}$ . The nodes are counted from the bottom to the top.

The lower edit boxes show the time- $t_k$  expected return of the put, given that the put has not been exercised before time  $t_k$ , that the price is determined by the value in the corresponding upper edit box, and that an optimal policy will be followed from time  $t_k$  onward. In particular, the lower edit box in the 0th column shows the approximate value of the risk-neutral price of the put option.

The lower edit box is highlighted, if the return from exercising the option at  $t_k$  is greater than expected return if we keep the option at least until  $t_{k+1}$ . The holder of the

option should immediately exercise the option if it is in the highlighted edit box, and vice versa.

The four disabled edit boxes in the bottom of the window show the values of different parameters.

- ☞ **Discount per step.** Shows the value of  $\beta$ , calculated by (11).
- ☞ **Up step size.** Shows the value of  $u$ , calculated with the help of the first equation in (7).
- ☞ **Down step size.** Shows the value of  $d$ , calculated with the help of the second equation in (7).
- ☞ **Probability of up move.** Shows the value of  $p$ , calculated by (8).

Consider the example of an American call option. Let all the values be the same as in the previous example of the put option. How to calculate the risk-neutral price?

The parameters  $u$ ,  $d$ ,  $\beta$ ,  $p$  and  $S(t_k)$  are calculated by the same formulas (11)–(9). However,  $V_k(j)$ , the possible time- $t_k$  expected values of the call, are calculated as

$$V_n(j) = \max\{u^j d^{n-j} S(0) - K, 0\},$$

and for  $k = n - 1, \dots, 0$  they are calculated as

$$V_k(j) = \max\{u^j d^{k-j} S(0) - K, \beta p V_{k+1}(j+1) + \beta(1-p)V_{k+1}(j)\},$$

$j = 0, \dots, k$ . Using these formulas, we obtain

$$V_0(0) \approx 6.359.$$

Fig. 26 shows the results of calculations. Note that the radio button **Call** is active now. Also note that all the highlighted edit boxes are in the last column. Indeed, according to [4, Proposition 5.2.1], one should never exercise the American call option before its expiration time.

*Remark 10.* A program `mbm` has a limitation. You can not calculate the multi-period binomial model with 6 or more periods. For calculation of such models you can use MATLAB Command Window directly. See Section A.3.

### 3.5.1 Problems

**Problem 9.** Consider a five-month ( $t = 0.4167$ ) American option when the initial price of the stock is  $S(0)$ , the strike price is  $K$ , the risk-free interest is  $r\%$  per annum, and the volatility is  $\sigma$  per annum. Divide the life of the option into 5 equal periods. Suppose that the price of a security can change only at the times  $t_k = kt/5$ ,  $k = 1, 2, \dots, 5$  and that the option can be exercised only at one of the times  $t_k$ .

Calculate the risk-neutral price of both call and put options for the cases from Table 11. For put options describe all the moments, when the holder of the option should immediately exercise the option. Also calculate the up step size, down step size and probability of up move for every case.

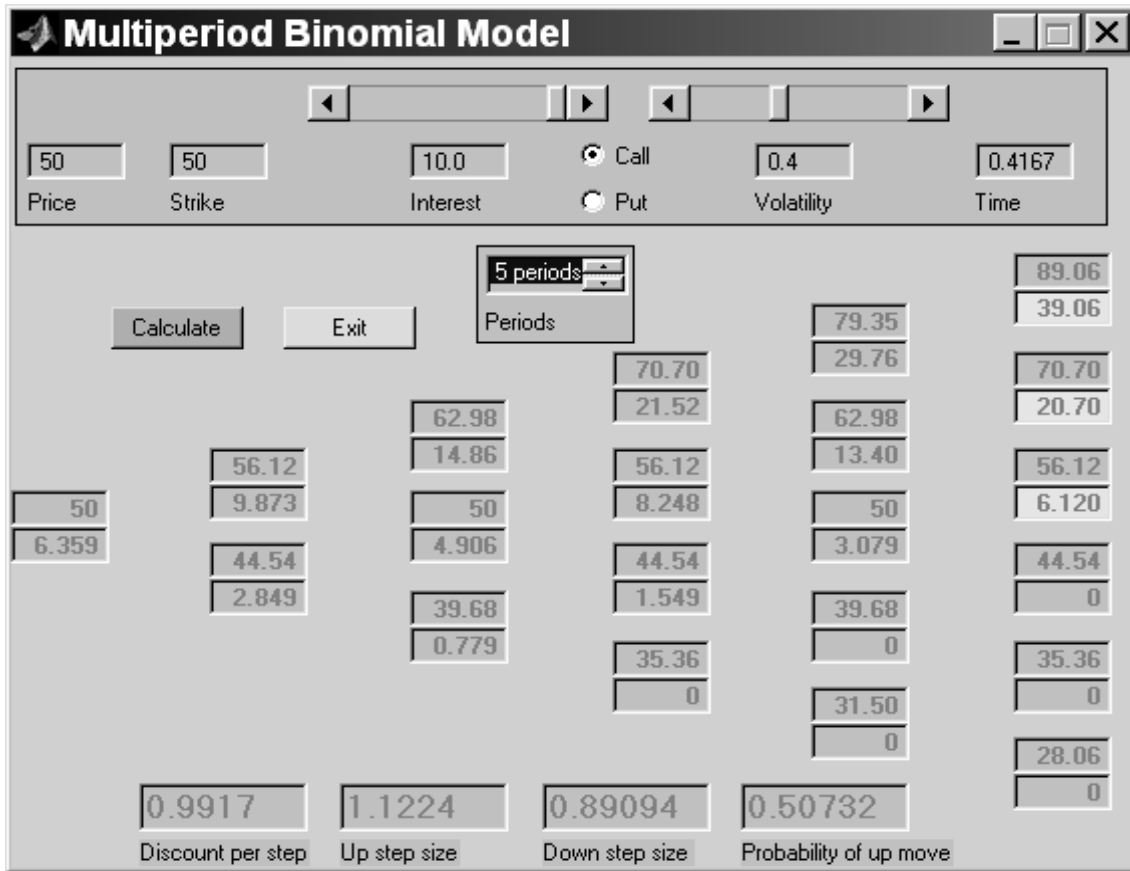


Figure 26: An output of the program mbm, case of call option

$S(0)$	$K$	$r$	$\sigma$
40	39	8	0.3
50	50	10	0.1
100	100	8	0.4
50	49	10	0.1
80	80	5	0.5
40	40	8	0.9
50	51	5	0.2
25	27	10	0.6

Table 11: Data for pricing of American options

### 3.6 The Black–Scholes formula

Consider the next problem. The initial price of the security is  $S(0) = \$100$ , the exercise price of the option is  $K = \$95$ , the risk-free interest rate is  $r = 10\%$ , the time to maturity of the option is  $t = 0.25$  years, and the volatility of the security is  $\sigma = 50\%$ . Calculate the value of European type call ( $C$ ) and put ( $P$ ) options.

Black–Scholes formula gives

$$C = S(0)\Phi(\omega) - Ke^{-rt}\Phi(\omega - \sigma\sqrt{t}),$$

where

$$\omega = \frac{rt + \sigma^2 t/2 - \log(K/S(0))}{\sigma\sqrt{t}},$$

and  $\Phi(\omega)$  is the standard normal distribution function. According to put–call option parity

$$P = C + Ke^{-rt} - S(0).$$

Using these formulae, we obtain  $C \approx 13.70$  and  $P \approx 6.35$ .

Consider how the MATLAB program `black_scholes` (Fig. 27) solves this problem.

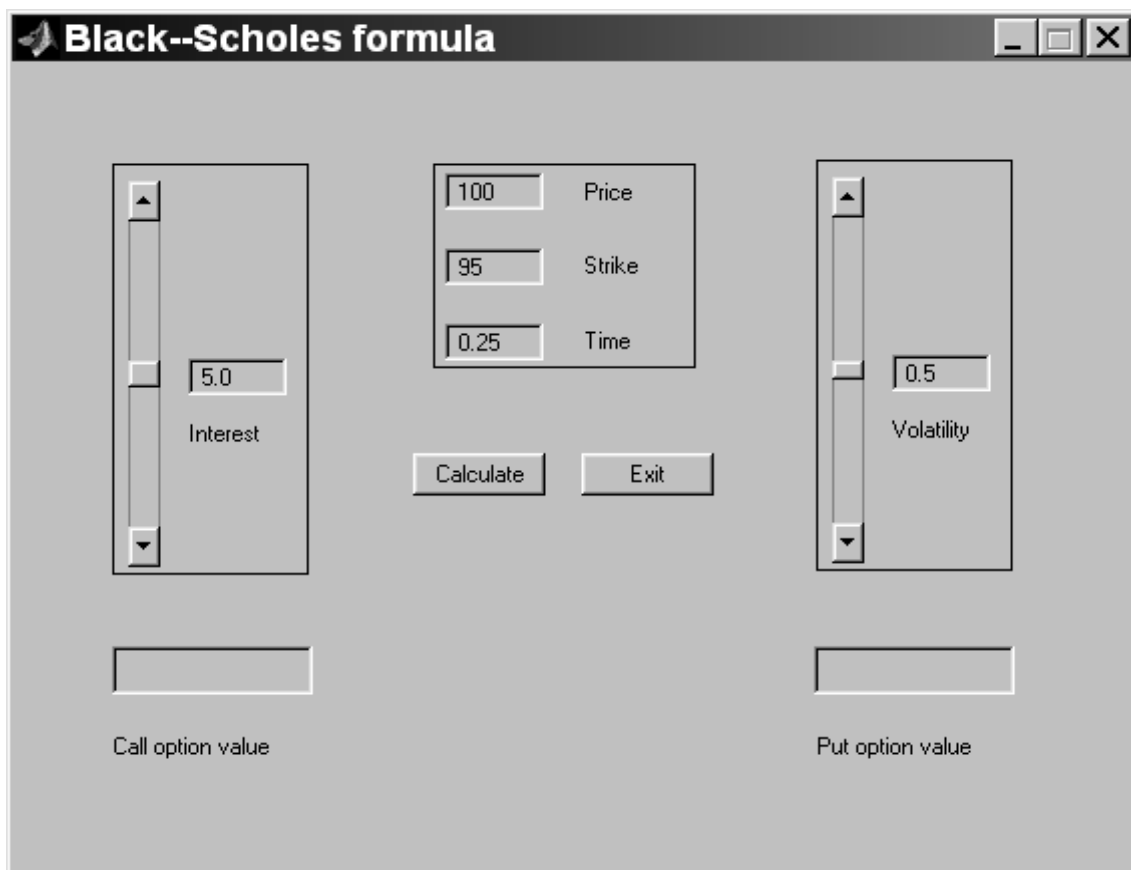


Figure 27: A window of the program `black_scholes`

The program `black_scholes` requires the Financial toolbox.

The frame in the left hand side of the window contains the edit box **Interest** and the corresponding slider. They are responsible for the value of the annual risk-free interest rate,  $r$ . The standard problem has the value  $r = 0.05$ , which corresponds to 5%. The minimum slider (and edit box) value is equal to 0%, the maximum slider and edit box value is equal to 10%.

The frame in the left hand side of the window contains the edit box **Volatility** and the corresponding slider. They are responsible for the value of the annualised volatility,  $\sigma$ . Standard problem has a value  $\sigma = 0.5$ , which corresponds to 50%. The minimum slider (and edit box) value is equal to 0%, the maximum slider and edit box value is equal to 100%.

The frame in the upper part of the window contains three edit boxes.

- ☞ **Price.** Contains the value of the initial price of the security,  $S(0)$ . The standard problem has the value  $S(0) = 100$ .
- ☞ **Strike.** Contains the value of the strike price,  $K$ . The standard problem has the value  $K = 95$ .
- ☞ **Time.** Contains the time to maturity of the option in years,  $t$ . The standard problem has value  $t = 0.25$ .

The push buttons perform the following functions.

- ☞ **Calculate.** Calculates values of call and put options, using the Black-Scholes formula and put-call option parity.
- ☞ **Exit.** Stops the program.

The results of calculations of our example are shown on Fig. 28. The disabled edit box **Call option value** contains the value of the call option,  $C$ . The disabled edit box **Put option value** contains the value of a put option,  $P$ .

### 3.6.1 Problems

**Problem 10.** Calculate the value of call and put options for the cases from Table 11. The time to maturity of the option is  $t = 0.4167$  years. Compare your results with results of the solution of Problem 9.

## A Overcoming limitations

Some programs in the complex contain limitations. You can overcome these limitations, using MATLAB Command Window directly. Consider some examples.



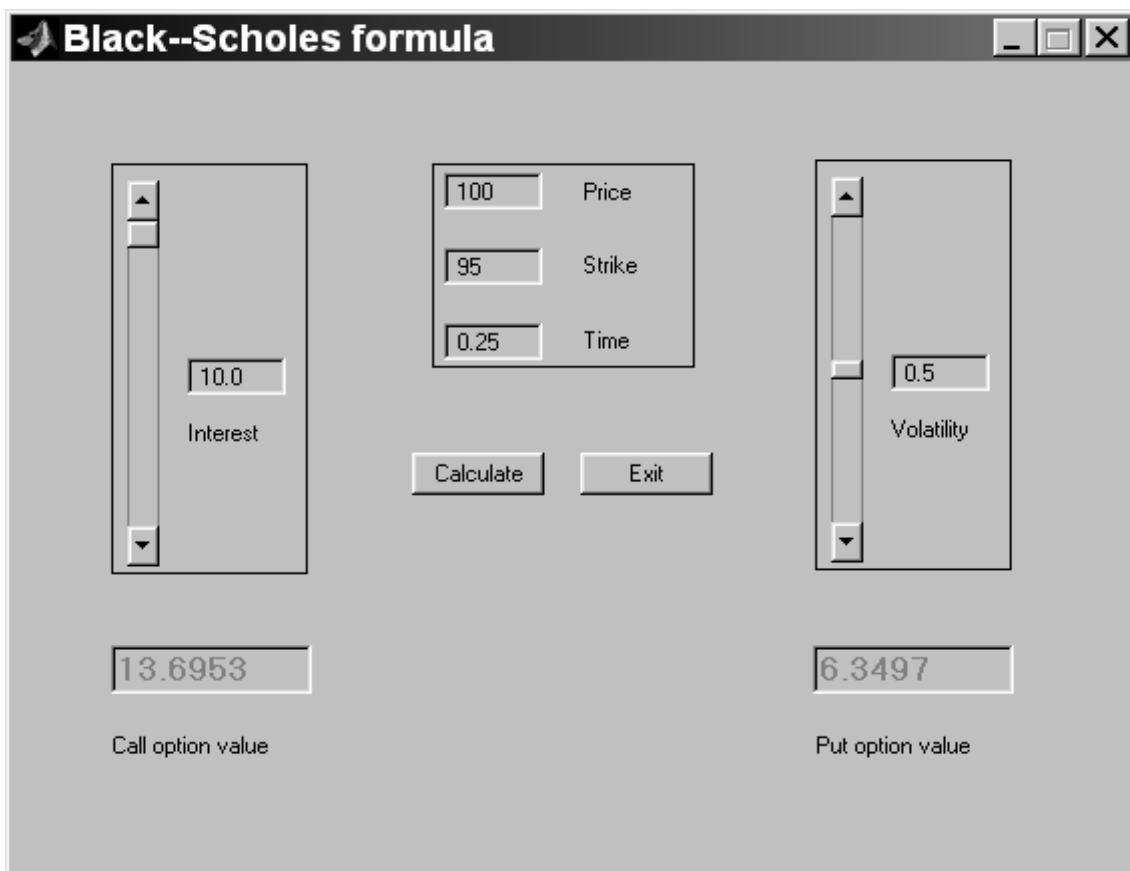


Figure 28: An output of the program black\_scholes

Year 1	\$2000
Year 2	\$1500
Year 3	\$3000
Year 4	\$3800
Year 5	\$5000
Year 6	\$6000

Table 12: A long varying periodic cash flow

## A.1 The program `present_value`

Using the program `present_value`, you can not calculate the present value of a cash flow containing more than 5 payments. The Command Window can be used instead. Consider the next example.

The cash flow (Table 12) represents the yearly income from an initial investment of \$15,000. The annual interest rate is 8%. How to calculate the present value of this varying cash flow?

We can not use the program `present_value` directly, because our cash flow contains more than 5 payments. Instead we can make direct use of the MATLAB Command Window.

The first time MATLAB starts, the desktop appears as shown in Fig. 29. The window in the left top corner is called *Launch Pad*. It contains a list of tools, demos and documentation of your MATLAB configuration and provides easy access to them. The window in the right side is called *Command Window*. You can use it to enter variables and run functions and M-files. *M-files* are text files containing MATLAB code. In particular, any program of the complex (Table 1) is contained in a M-file. You can run this file by typing its name in MATLAB Command Window.

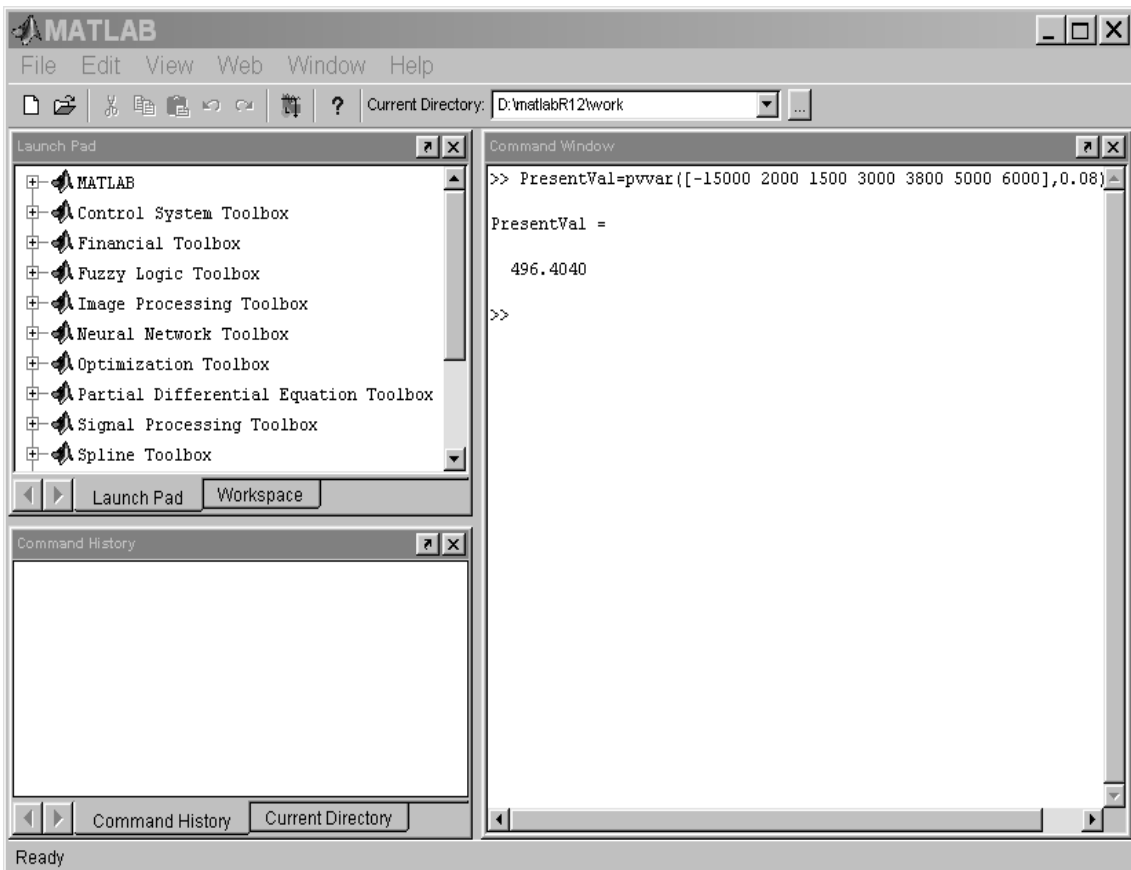


Figure 29: Finding present value of a long cash flow

Input the next command into the MATLAB Command Window (Fig. 29).

```
PresentVal=pvvar([-15000 2000 1500 3000 3800 5000 6000],0.08)
```

After pressing **Enter** you will obtain the result:

```
PresentVal =
```

```
496.4040
```

Specifically, we introduce a new MATLAB *variable* named `PresentVal`. You can think of a variable as a named place in the computer's memory. Every variable should have some value. In our case we called MATLAB *function* `pvvar`. Functions are M-files that can accept input arguments and return output arguments. The function `pvvar` is contained in the Financial toolbox.

We passed two input arguments to this function. The value of the first argument is equal to `[-15000 2000 1500 3000 3800 5000 6000]`. This is the vector of cash flows. The initial investment is included as the initial cash flow value (a negative number). The value of the second argument is equal to `0.08`. It is the yearly interest rate.

The function `pvvar` returned an output argument (the present value of a cash flow representing by its first input argument with the yearly interest rate representing by its second argument). The value of the output argument was written into the variable `PresentVal`. It was also written in the Command Window (Fig. 29).

Other examples of calculations of present values can be found in [1, pp. 4-224–4-226]. In order to download this document, point your browser to the address shown at page 3.

## A.2 The program `ror`

Using the program `ror`, you can not calculate the rate of return of a cash flow containing more than 5 payments. The Command Window can be used instead.

Consider the next example. Let us calculate the rate of return from an initial investment of \$15,000 and payments shown in Table 12. There are 6 payments, and we can not use the program `ror`. Let us use the MATLAB Command Window instead.

Input the next command into the MATLAB Command Window (Fig. 30).

```
Return=irr([-15000 2000 1500 3000 3800 5000 6000])
```

After pressing **Enter** you will obtain the result:

```
Return =
```

```
0.0888
```

or approximately 8.88%.

We called the MATLAB function `irr`. This function is contained in the Financial toolbox. A vector representing the cash flow was passed to the function `irr` as its unique input argument. After calculations, the function `irr` returned the value of the rate of return as its output argument and wrote it into the variable `Return` in the computer's memory. MATLAB wrote the value of the variable `Return` for you in the Command Window. This variable was also written into the *workspace*. The MATLAB workspace consists of

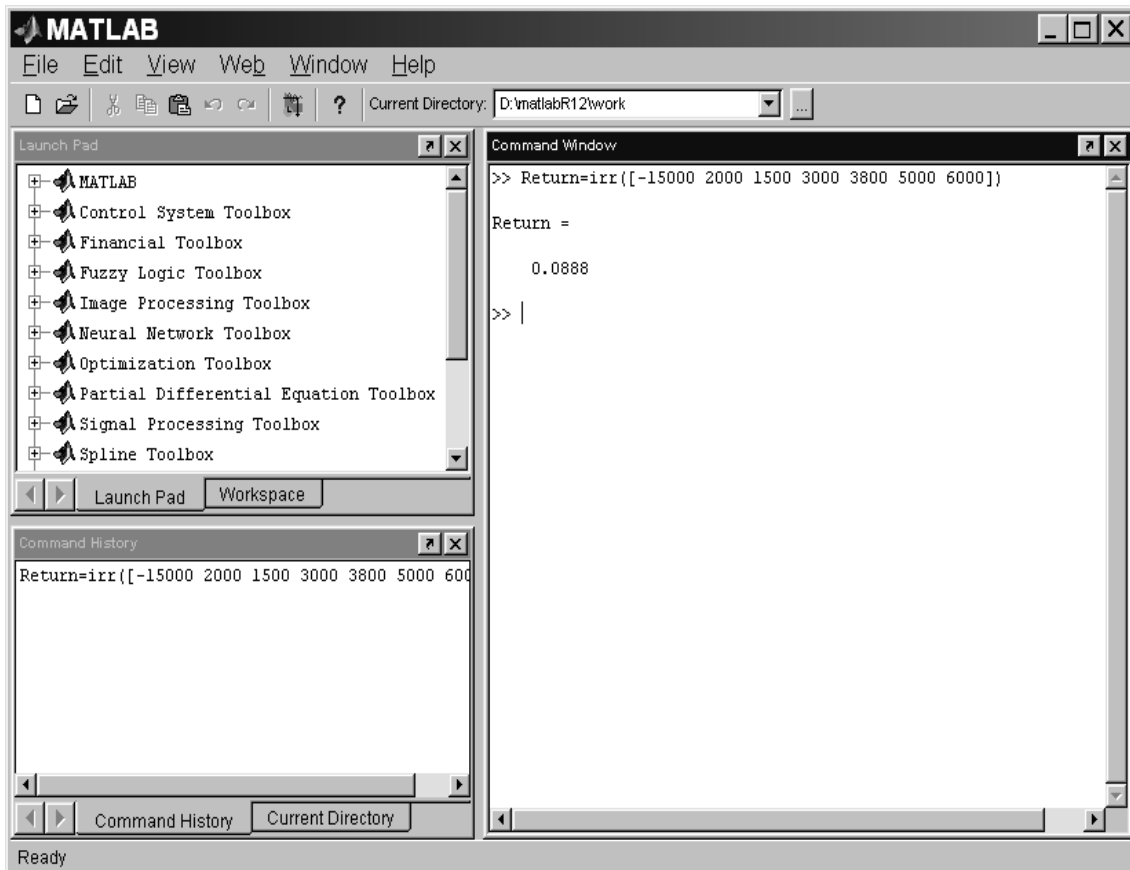


Figure 30: Finding rate of return of a long cash flow

the set of variables built up during a MATLAB session and stored in the memory. You can use the variables of the workspace in subsequent commands.

Other examples of calculations of rates of return can be found in [1, pp. 4-166, 4-260–4.261].

### A.3 The program mbm

Using the program mbm, you can not make calculations with return of a cash flow containing more than 5 payments. The Command Window can be used instead.

Consider the example of an American put option from Section 3.5. Assume we want to calculate the risk-neutral price of this option using 100 periods.

Input the next command into the MATLAB Command Window (Fig. 31)

```
[AssetPrice,OptionValue]=binprice(50,50,0.1,0.4167,...
0.4167/100,0.4,0);
```

We called the MATLAB function binprice. This function is contained in the Financial toolbox. We passed seven parameters to the function binprice. These parameters are shown in Table 13.

In the MATLAB Command Window, we used three dots ... to indicate that the statement continues at the next line. The fifth parameter is adjusted so that the length of each

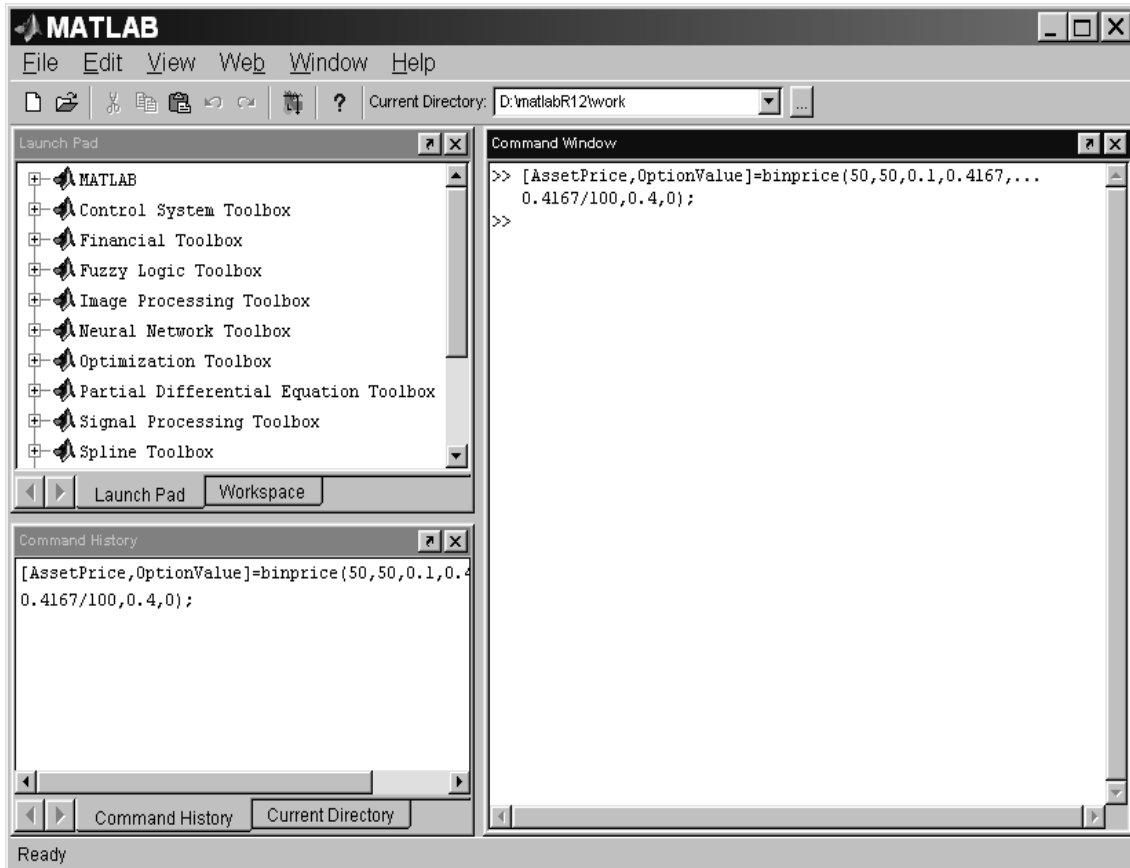


Figure 31: A command for calculation of the 100-period binomial model

Number	Meaning
1	The initial price of the security
2	The exercise price of the option
3	The risk-free interest rate
4	The option's exercise time in years
5	The length of one period
6	The annualised volatility
7	Specifies whether the option is a call (1) or a put (0)

Table 13: The parameters of the MATLAB function binprice

interval is consistent with the exercise time of the option. The option's exercise time divided by the length of one period equals an integer number of periods.

The function `binprice` returns two output arguments. The first output argument is the matrix of the security's prices. The second output argument is the matrix of the option's prices. We used the semicolon `;` to suppress the MATLAB's output. Without semicolon two huge matrices  $100 \times 100$  would appear in the Command Window.

The approximate value of the risk-neutral price of the option contains in the variable `OptionValue(1,1)`. Enter this variable into the Command Window and press **Enter**. We obtain the result (Fig. 32).

`ans =`

4.2782

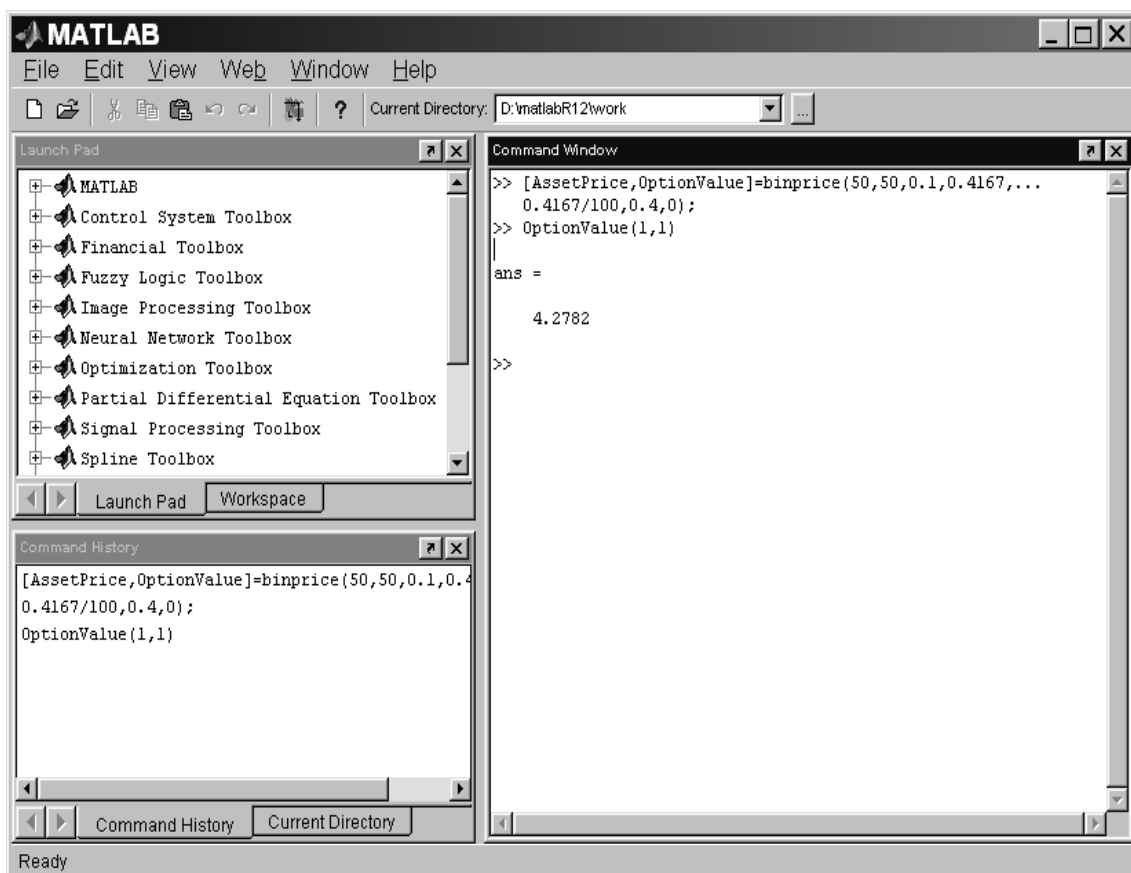


Figure 32: The result of calculation of the 100-period binomial model

The difference between two values ( $V_0(0) \approx 4.488$  for  $n = 5$  and  $V_0(0) \approx 4.2782$ ) is essential. Note that another software, DerivaGem [2, p. 393], gives the same answer as MATLAB.

You can calculate the American call option with 100 periods yourselves. The only difference is: the value of the seventh parameter should be equal to 1.

Other examples of binomial put and call pricing can be found in [1, p. 4-37–4-38] and in [2, Chapter 16].

### A.3.1 Problems

**Problem 11.** Calculate the value of call and put options for the cases from Table 11. The time to maturity of the option is  $t = 0.4167$  years. Divide it onto 100 equal parts. Compare your results with results of the solution of Problems 9 and 10.

## References

- [1] *Financial Toolbox for Use with MATLAB*, User's Guide, Version 2.1.2, The Math-Works, Inc., September 2000.
- [2] Hull, J. C. *Options, Futures & Other Derivatives*, Fourth Edition, Prentice Hall, Upper Saddle River, 2000.
- [3] Prisman, E. Z. *Pricing Derivative Securities: an Interactive Dynamic Environment with Maple V and Matlab*, Academic Press, 2000.
- [4] Ross, S. M. *An introduction to Mathematical Finance: Options and Other Topics*, Cambridge University Press, 1999.

## List of Figures

1	The MATLAB desktop . . . . .	4
2	The Control Centre . . . . .	5
3	A window of the program <code>price_evolution</code> . . . . .	6
4	An example of incorrect input . . . . .	7
5	An output of the program <code>price_evolution</code> . . . . .	8
6	A window of the program <code>clt</code> . . . . .	9
7	An output of the program <code>clt</code> . . . . .	10
8	A window of the program <code>gbm</code> . . . . .	12
9	An output of the program <code>gbm</code> . . . . .	13
10	A window of the program <code>interest_rate</code> . . . . .	14
11	An output of the program <code>interest_rate</code> . . . . .	16
12	A window of the program <code>present_value</code> . . . . .	19
13	An output of the program <code>present_value</code> , case of fixed periodic payments	20
14	An output of the program <code>present_value</code> , case of varying periodic payments . . . . .	21
15	An output of the program <code>present_value</code> , case of varying non-periodic payments . . . . .	22
16	A window of the program <code>ror</code> . . . . .	25
17	An output of the program <code>ror</code> , case of periodic payments . . . . .	26
18	An output of the program <code>ror</code> , case of non-periodic payments . . . . .	27
19	Possible security prices at time 1 . . . . .	28
20	The window of the program <code>options_pricing</code> . . . . .	30
21	An output of the program <code>options_pricing</code> , case of too expensive option	31
22	An output of the program <code>options_pricing</code> , case of too cheap option . . . . .	32

23	An output of the program <code>options_pricing</code> , case of no-arbitrage price .	33
24	A window of the program <code>mbm</code> . . . . .	35
25	An output of the program <code>mbm</code> , case of put option . . . . .	36
26	An output of the program <code>mbm</code> , case of call option . . . . .	38
27	A window of the program <code>black_scholes</code> . . . . .	39
28	An output of the program <code>black_scholes</code> . . . . .	41
29	Finding present value of a long cash flow . . . . .	42
30	Finding rate of return of a long cash flow . . . . .	44
31	A command for calculation of the 100-period binomial model . . . . .	45
32	The result of calculation of the 100-period binomial model . . . . .	46

## List of Tables

1	Programs in the complex . . . . .	5
2	Two equal simple cash flows . . . . .	17
3	Two equal more complicated cash flows . . . . .	18
4	Varying periodic cash flow . . . . .	18
5	Varying non-periodic cash flow . . . . .	21
6	Cash flow, problem 4 . . . . .	23
7	Cash flow, problem 5 . . . . .	23
8	The yearly income from an initial investment of \$100,000 . . . . .	23
9	The non-periodic cash flow from an initial investment of \$10,000 . . . . .	25
10	One-period binomial models . . . . .	33
11	Data for pricing of American options . . . . .	38
12	A long varying periodic cash flow . . . . .	41
13	The parameters of the MATLAB function <code>binprice</code> . . . . .	45