V.I. Reizlin

NUMERICAL METHODS OF CONTINUOUS OPTIMIZATION

Guidelines for the laboratory workshop
for graduate students of 09.04.01
"Computer science and engineering»

On the subject "Optimization Methods"

Tomsk 2014

LAB # 1

ONE-DIMENSIONAL OPTIMIZATION METHODS

The aim of this work: study of methods for one-dimensional search, as well as the study of the influence of algorithm parameters of appropriate methods for their effectiveness.

1. Job description

To solve the optimization problem in which the characteristic measure is a function of one variable, you can use a variety of methods. The choice of the solution of optimization problems depends on the various assumptions and assumptions about the nature and properties of the function. The following are some of the known methods for one dimensional optimization.

1.1. The methods of exclusion ranges

These methods are focused on finding the optimum within a specified interval to determine optimum function of one variable by sequential deletion of podyntervalov and, therefore, by reducing the search interva [l, 2].
To begin your search by using these methods, you must set the interval that contains the optimum. You can then apply the procedure for the search interval to obtain updated estimates of absolute coordinates. The subinterval, an exclusion on the each step depends on the location of the sampling points x_1 and x_2 in the search interval. Since the location of the point of optimum, a priori, it is appropriate to suggest that the location of test points shall ensure that the reduction of the interval in the same regard. In addition, in order to improve the efficiency of the algorithm must require that a particular relationship is maximized. This strategy is sometimes called a Minimax search strategy.

1.1.1. The method of dividing the interval in half

This method allows you to exclude exactly half the interval for each iteration. This is sometimes called a three-point search on equal intervals, because its implementation is based on the choice of three test points, uniformly distributed in the interval following is a search algorithm to search for finding a minimum of the function *f(x)* in the interval *(a, b).*

Step 1. Put $xm = (a+b)/2$ and $L = b - a$ Calculate the value. *f(xm)*.

Step 2. Put $x1 = a + L/4$ and $x2 = b - L/4$ Thus, the point $x1$, $x2$ and xm divide the interval *(a, b)* into four equal parts. Calculate values *f(x1)* and *f(x2)* .

Step 3. Compare *f(x1)* and *f(xm)*.

If $f(x1) < f(xm)$ delete the interval *(xm, b)* by putting $b = xm$ The new midpoint interval. search becomes point x 1. Therefore, it is necessary to put an $xm=x1$. go to step 5. If $f(x,1) \geq f(xm)$, go to step 4.

Step 4. Compare *f(x2)* and *f(xm)*.

If $f(x2) < f(xm)$ delete the interval *(a, xm)* by putting $a= xm$ Since the midpoint of the new range becomes point $x2$, put $xm = x2$ 5. go to step ... If $f(x2) \geq f(xm)$ delete intervals *(a, x1)* and *(x2, b)* Put $a= x1$ and $b = x2$ New interval midpoint. continues to be *xm* 5. go to step.

Step 5. Calculate $L = b - a$ If the value of $|L|$ small finish search. Otherwise, return to step 2.

1.1.2. The method of golden section

In contrast to the above in the method of golden section at each iteration is only one value of the objective function. Here is a particular implementation of this method.

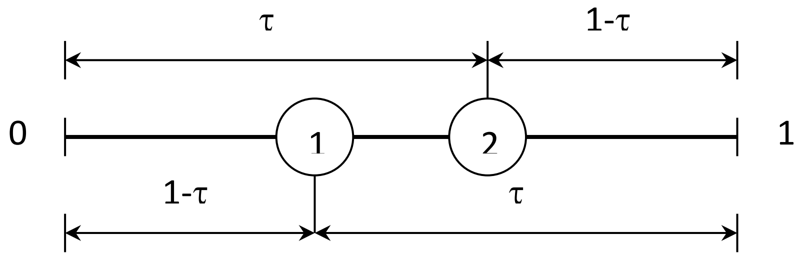

Figure 1.1. Search through golden section method

Consider a symmetrical two test points on the original range of unit length, which is shown in fig 1.1. Sample points are positioned from the interval boundary points τ The symmetric location points remaining after the interval length is always τ regardless of which of the values of the functions in the pilot locations are smaller. Suppose that is right podynterval figure 1.2 shows the remaining podynterval length τ contains one sampling point located at a distance (1-τ) from the left endpoint.

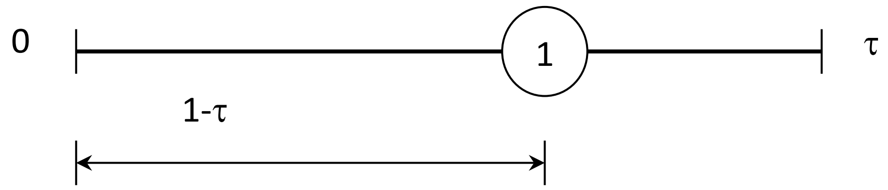

Figure 1.2. The intervals obtained by the golden section

To the symmetry of the search pattern persisted, distance (1-τ) must be τ the second part of the length of the interval (which is equal to τ) for. τ the next test point is placed on the distance equal to τ part of the length of the interval from the right perimeter point interval (Figure 1.3).

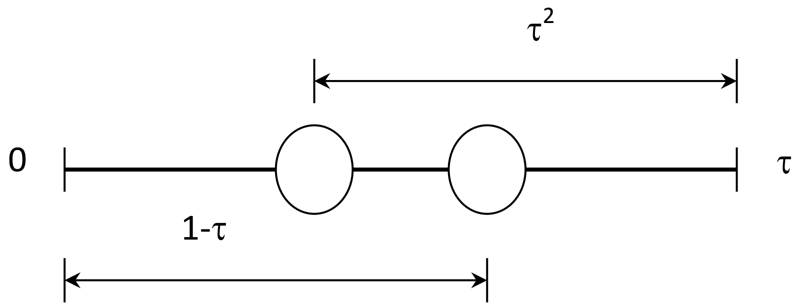

Figure 1.3. The symmetry of the golden section interval

This means that if you select τ in accordance with condition $1-\tau = \tau^2$ the search model, shown in Figure 1.1, is preserved when you move to a reduced interval, which is shown in Figure 1.3. Solving this quadratic equation, we get

$$\tau = (-1 \pm \sqrt{5}) / 2,$$

where a positive decision τ = 0.61803... . Search scheme in which trial points divide the interval, known as search using the golden section. Algorithm to search through this method is as follows.

Step 1. Put *x0 = a, x3 = b*.

Step 2. Put *x1 = a+ t1-(b-a)* Calculate the value. *f(x1)*.

Step 3. Put *x2 = a+ t2 · (b - a)*. Calculate the value of *f(x2)*.

Step 4. Compare *f(x2)* and *f(x1)*. If *f(x1) > f(x2)* delete the interval *(x0, x1)* by putting *L = x3- x1, x0 = x1, x1 = x2, x2 = x0 + t2 · L , f(x1) = f(x2)*

Calculate the value. $f(x2)$ go to step 5. If $f(x1) < f(x2)$ delete the interval $(x2, x3)$ by $L = x2 - x0$, $x3 = x2$, $x2 = x1$, $x1 = x0 + t1 \cdot L$, $f(x2) = f(x1)$ Calculate the value. $f(x1)$. go to step5.

Step 5. Determine the amount of computation functions- k. If $k < N$ and $| L | > E$, go to step 4. Otherwise, end the search.

1.1.3. Fibonacci Method

Fibonacci Method has the same sequence of intervals that the exceptions and the method of golden section. Difference is selecting the starting points and in the size exclusion starting point interval. $x1$ Fibonacci method is calculated by the formula [1].

$$x1 = ((b-a) \bullet F(N-1) + E-(-1)^N)/f N + a \quad (1.1)$$

where fi -Fibonacci numbers $i = 0, 1, 2, ..., the N$;

N -the number of computing functions;

E- the specified accuracy.

As can be seen from equation *(1.1),* to get a starting point you must know in advance the number of calculation functions, which will be in the search result is the optimum value with the specified precision. Below is a search algorithm for the method.

Step 1. Put $x0 = a$, $x3 = b$. Calculate the value of $x1$ according to the formula (1.1). Calculate the value of $f(x1)$.

Step 2. Put $x2 = x0 - x1 + x3$ Calculate the value. $f(x2)$.

Step 3. Compare $f(x2)$ and $f(x1)$.

If $f(x2) > f(x1)$, compare $x1$ and $x2$ If $x2 > x1$ delete the interval $(x2, x3)$ by putting $x3 = x2$. Go to step 4. If $x2 < x1$ delete the interval $(x0, x2)$ by putting $x0 = x2$. Go to step 4. If $f(x2) < f(x1)$, compare $x1$ and $x2$ If $x1 > x2$ delete the interval $(x1, x3)$ by putting $x3 = x1$, $x1 = x2$, $f(x1) = f(x2)$. Go to step 4.

If $x1 < x2$ delete the interval $(x0, x1)$ by putting $x0 = x1$, $x1 = x2$, $f(x1) = f(x2)$. go to step 4.

Step 4. Determine the amount of computation functions- k If $k < N$ go to step 2, otherwise complete the search.

1.2. the polynomial approximation and methods point estimation

The basic idea of the method is the possibility of approximation of smooth functions of polynomial and then using an approximating polynomial for the estimation of the optimum point coordinates//1.2. Necessary conditions for the effective implementation of this approach are the unimodal′nost′ and the continuity of the function. According to the Weierstrass theorem on approximation, if a function is continuous in some interval, then it's with any degree of accuracy can be approximated by a polynomial of sufficiently high order. Therefore, if the function is an unimodal and found a polynomial that approximates it quite accurately, the y-coordinate of the point of optimum function can be estimated by calculating the coordinates of the point of optimum polynomial. According to the theorem of Weierstrass, the quality of the estimates of absolute coordinates, using an approximating polynomial can be improved in two ways: using higher-order polynomial and decrease the interval approximation. The second method, generally speaking, is more preferable because building an approximating polynomial order above the third becomes a very complicated procedure, while decreasing the interval is an assumption about the unimodal function, is not particularly difficult.

1.2.1. Estimation Method using quadratic approximation

The simplest polynomial interpolation is a quadratic approximation, which is based on the fact that the function to take the minimum value in the inner point of the interval must be at least quadratic. If a function is linear, the optimal value can be achieved only in one of the two boundary points of the interval. Therefore, when implementing a method of estimation using a quadratic approximation assumes that in a limited range can be approximated by a quadratic polynomial function, and then use the approksimiruûŝuû scheme for the evaluation of constructed the coordinates of a point of true minimum functions.

If you specify a sequence of points x_1, x_2, x_3 and are known to correspond to the points values of functions f_1, f_2, f_3, it is possible to define constants *and*$_0$*and*$_1$ and $_2$ *and* in such a way that the value of a quadratic function

$$q(x) = a_0 + a_1(x - x_1) + a_2(x - x_1)(x - x_2)$$

coincide with the values of the *f(x)* the three Go to the calculation of points. *q (x)* in each of the three designated points. First of all, because

$$f_1 = f(x_1) = q(x_1) = a_0,$$

we have $a_0 = f_1$.

Further, since

$$f_2 = f(x_2) = q(x_2) = f_1 + a_1(x_2 - x_1),$$

we get

$$a_1 = (f_2 - f_1)/(x_2 - x_1). \qquad (1.2)$$

Finally, if $x = x_3$

$$f_3 = f(x_3) = q(x_3) = f_1 + (f_2 - f_1)/(x_2 - x_1)(x_3 - x_1) + a_2(x_3 - x_1)(x_3 - x_2).$$

Allowing the last equation on a_2, we get

$$a_2 = \frac{1}{x_3 - x_2}\left(\frac{f_3 - f_1}{x_3 - x_1} - \frac{f_2 - f_1}{x_2 - x_1}\right). \qquad (1.3)$$

Thus, the three given points and the corresponding values of the function, you can estimate the parameters a_0*and*$_1$ and *and*$_2$ an approximating polynomial using the above formulas.
If the precision of the approximation of functions, ranging from x_1 to x_3 using a polynomial is sufficiently high, in accordance with the strategy of search you can use polynomial constructed for evaluation of optimum point coordinates. Fixed points of functions of one variable is determined by equating to zero as its first derivative and subsequent finding roots of equations. In this case, from the equation

$$\frac{dq}{dx} = a_1 + a_2(x - x_2) + a_2(x - x_1) = 0$$

You can get the

$$\bar{x} = (x_2 + x_1)/2 - (a_1 / 2 \cdot a_2). \qquad (1.4)$$

Because the function *f(x)* on the interval has the property of unimodal and approximating a quadratic polynomial is also a unimodal function, it can be expected that the $\bar{x}$ will prove to be an acceptable estimate of the true optimum point coordinates x^* [2]. Diagram of the algorithm optimal point can be described as follows.

Step 1. Put $x1 = a$, $x3 = b$, $x2 = (x3-x1)/2$.

Step 2. Calculate *f(x1), f(x2), f(x3).*

Step 3. A three-point *x 1 , x 2 , x 3* calculate parameters *a1, a2,* using equation (1.2) and (1.3), i.e.

$$a1 = \frac{f(x2) - f(x1)}{x2 - x1},$$

$$a2 = \frac{1}{x3 - x2}\left(\frac{f(x3) - f(x1)}{x3 - x1} - \frac{f(x2) - f(x1)}{x2 - x1}\right).$$

Step 4. Calculate the optimum $\bar{x}$ using parameters *a1* and *a2,* using the formula (1.4).

1.2.2. Powell's Method

This method, developed by Powell, is based on the consistent application of the procedures of evaluation using a quadratic approximation scheme of the algorithm can be described as follows: Let x_1 - the starting point, Δx - the value selected for the step axis x.

Step 1. Calculate $x_2 = x_1 + \Delta x$.

Step 2. Calculate $f(x1)$ and $f(x2)$.

Step 3. If $f(x1) > f(x2)$, put $x_3 = x_1 + 2\Delta x$ If $f(x1) \leq f(x2)$, put $x_3 = x_1 - \Delta x$.

Step 4. Calculate $f(x3)$ and find

$$Fmin = min\{f1, f2, f3\},$$

$Xmin$ = dot xi that corresponds to the $Fmin$.

Step 5. A three-point x_1, x_2, x_3 calculate $\bar{x}$ using the equation (1.2), (1.3), (1.4).

Step 6. Check the end of the search. is the difference $Fmin - f(\bar{x})$ small enough? Is the difference $Xmin - \bar{x}$ small enough? When both conditions are true, complete search; otherwise, go to step 7.

Step 7. Choose the "best" point $(Xmin$ or $\bar{x})$ and the two points on either side of the Label that points to the natural order, and go to step 4.

The first implementation step 5 interval that contains the minimum may not necessarily be established. The obtained point $\bar{x}$ the point can be x_3 In order to avoid too much probability should be moved after step 5 extra checking and, in the case where a $\bar{x}$ is too far from the x_3, replace $\bar{x}$ point coordinate which is calculated based on a predetermined step length.

1.3. The methods using derived

All covered in previous sections of the search methods are based on assumptions about unimodal and, in some cases, the continuous study of the target function. It is useful to assume that if in addition to introduce a requirement for continuity differentiability condition function, the efficiency of search procedures can significantly increase. Recall that a necessary condition for the existence of a local minimum of the function at some point *z* zero is the first derivative of the function at that point, i.e. $f'(z) = df\ / dx|_{x=z} = 0$.

If the function *f(x)* contains members that include *x* in the third and higher degrees, the direct receipt of analytical solutions of the equation $f'(x) = 0$ it may be difficult in such cases are approximate methods of incremental search fixed points of functions *f*.

1.3.1. Newton-Raphson Method

In the context of a Newton-Raphson assumes the function *f* twice differentiable. the algorithm starts at the point x_1, which represents the starting value (or initial evaluation) the coordinates of the stationary point or square root equations $f'(x) = 0$ Then build linear approximation of the function $f'(x)$ at the point x_1, and the point at which approximates a linear function becomes zero is taken as the next approximation. If point x_{to} adopted as the current approach the fixed point, then a linear function, approximating the function $f'(x)$ at a point $x_{to\ a}$ is written in the form

$$\tilde{f}'(x; x_k) = f'(x_k) + f''(x_k)(x - x_k). \qquad (1.5)$$

On the right side of equation (1.5) to zero, we get the following approximation:

$$x_{k+1} = x_k - [f'(x_k) / f''(x_k)]. \qquad (1.6)$$

Figure 1.4 illustrates the fundamental steps of Newton's method. Unfortunately, depending on the choice of the starting point and the type of function as the algorithm can converge to the true stationary point and diverge [2].

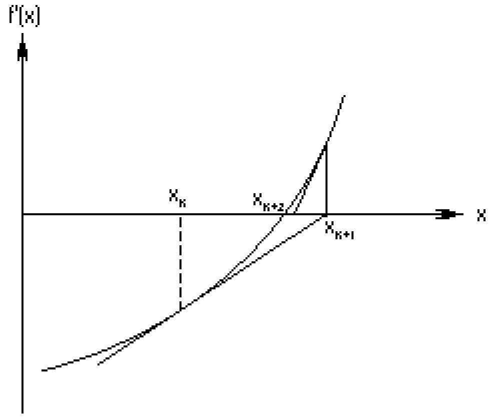

Figure 1.4 - The Newton-Raphson method

The following is the algorithm using the Newton-Raphson method search.

Step 1. Determine the starting point $x1$.

Step 2. Calculate $f'(x_k)$, $f''(x_k)$ and the following approximation formula (1.6).

Step 3. Check for end of search $| f'(x_k)| > E$, then go to step 2.

1.3.2. The midpoint Method

If the function $f'(x)$ unimodal search in the specified interval, the optimum point is the point at which the $f'(x) = 0$ If you can figure out how the value of the function and its derivative, then to find the root of the equation $f'(x) = 0$ you can use the effective exclusion algorithm intervals at each iteration where is considered only one trial period. For example, if a point z is inequality $f'(z) < 0$, given assumptions about unimodal′nosti naturally argue that point minimum may not be to the left of point z. In other words, the interval $x \leq z$ should be deleted, on the other hand, if $f'(z) > 0$, then the minimum point cannot be to the right of z and interval of $x \geq z$ you can delete [2]the reasoning underlying the logical structure of the midpoint method, sometimes called the search of Bolzano.
We define two points L and R in such a way that $f'L < 0$ and $f'(R) > 0$ Fixed point is located between the L and R. Calculate the value of the derivative of a function at the midpoint of the interval $z = (L+R)/2$ If $f'(z) > 0$ the interval (z,R) you can exclude from the search interval. On the other hand, if the $f'(z) < 0$ then you can delete the interval (L, z). The following is a formal description of the steps of the algorithm.

Step 1. Put $R = b$, $L = a$; the $f\,' a < 0$ and $f\,' b > 0$.

Step 2. Calculate $z = (R+L)/2$ and $f\,'(z)$.

Step 3. If $| f\,'(z) | \leq E$ Finish search. Otherwise, if the $f\,'(z) < 0$, put $L = z$ and go to step 2 If the $f\,'(z) > 0$, put $R = z$ and go to step 2.

It should be noted that the logical structure of the search as described by process of elimination of intervals is based only on the research of derivative regardless of the values that the derivative is.

1.3.3. The secant Method

The secant Method, a combination of Newton's method and the general pattern of exclusion ranges, is focused on finding a root of the equation $f\,'(x)$ in the interval (a, b), If, of course, the root exists.
Suppose that in the process of finding a fixed point of the function $f(x)$ in the interval (a, b) found two points L and R in which the characters derived different. In this case, the secant method algorithm approximates function $f\,'(x)$ «clipping straight» (straight line joining two points) and find the point at which a graph of secant $f\,'(x)$ crosses the x-axis (fig. 1.5). Therefore, the following approximation to a stationary point x * is defined by the formula

$$z = R - \frac{f'(R)}{[f'(R) - f'(L)]/(R - L)} . \qquad (1.7)$$

If $| f\,'(z)| \leq E$, the search must be complete; otherwise, you must select one of the points is L or R in such a way that marks the derivative at this point and point z were different, then repeat the basic step of the algorithm [2].

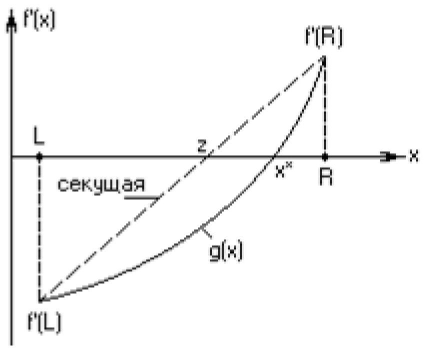

Figure 1.5 - Secant Method

The algorithm for this method is shown below.

Step 1. We put the $R = b$, $L = a$; the $f\,' a < 0$ and $f\,' b > 0$.

Step 2. Calculate the current approximation *z* to a minimum according to the formula (1.7). Calculate $f\,'(z\,)$.

Step 3. If $|\,f\,'(z)| \leq E$, Finish search. Otherwise, if $f\,'(z) > 0$, put $R=z$ and go to step 2. If $f\,'(z) < 0$, put $L=z$ and go to step 2.

After the algorithms you can start developing software complex, defining the first objective that must be fulfilled by the program.

2. The implementation of software complex

Complex for the study and research of one-dimensional search is implemented in your environment MATLAB.

After running the complex appears onscreen form that contains a certain number of Windows that you can use to specify methods for solving tasks option and the parameters of the study of algorithms.

The first three Windows <Section> <method> <Option> allow you to select the method the task group, respectively, within a given group-a method of solution and option that specifies the type of the target function.

The following window for displaying graphs of functions, the range of her study, referring to the objective function, the number of iterations N , to find the optimum, error, step are active in addressing the tasks the user can change the above settings, the settings of the target function and carry out studies of individual custom functions, using the step by step or automatic mode in step the user consistently implements virtual key to run.

The research results are presented to the user in a graphical form and in the form of a sequence of iterations with relevant quantitative estimates that appear in certain Windows screen form.

Background information explaining the characteristics of the implementation of each of the techniques is also displayed in the screen window of the form.

3. Implementation of the work

1. To familiarize themselves with the work of software.
2. To find a solution to test and compare with the received solution.
3. Get the job options for work.
4. Investigate the effects of parameters specified targets on the effectiveness of decisions.
5. Examine the effects specified accuracy on performance measures for methods.
6. Make a report with the results of the study methods.

A report must contain the results of the decision, the purpose of the given version of the target function, complemented by appropriate analytical calculations. In addition, you must submit a form for analysis, the results of the study of various parameters on the efficiency of methods. Bring the results of the decision of the user-defined function.

LAB # 2

METHODS OF MULTIVARIATE OPTIMIZATION

The purpose of the work: a study of the features of solution of optimization tasks using multivariate optimization without restrictions.

1. **Basic provisions**

To find extremum target functions of many variables, you can use a variety of methods [1, 2, 3, 4]. Depending on the specifics of the extremum search methods of multidimensional optimization can be divided into two groups: methods that use the actual values of the target functions and methods with the use of derivatives. The paper deals with the methods of the second group. The methods of this group is the use of an iterative procedure

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} \cdot s(x^{(k)}), \quad (1)$$

where $x^{(k)}$ -the current value of the argument of the function;

$(\alpha)^{k}$ is a parameter describing the length of the step;

$s(x^{(k)}) = s^{(k)}$ -Search direction in the N -dimensional space managed variables.

In determining the amount of and directions allocate a number of optimization techniques.

1.1. Gradient methods

Gradient methods in contrast to direct methods of search, search procedures that use only values of the objective function in the investigated points, presuppose the existence of derivative function. This reduces the number of required calculations inside functions.

If the search direction $s(x^{(k)})$ take direction antigradient function

$$s(x^{(k)}) = -\nabla f(x^{(k)}),$$

where $\nabla f = (\frac{\partial f}{\partial x_1}; \frac{\partial f}{\partial x_2}; ...; \frac{\partial f}{\partial x_n})$ -the gradient of the function,
We'll get out of (1) the ratio of the

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} \nabla f(x^{(k)}), \qquad (2)$$

program determines the method of steepest descent, or Cauchy method. In this method, the value of $\alpha^{(k)}$ on each iteration, computed by the task of finding the minimum functions $f(x^{(k+1)})$ along the lines of $\nabla f(x^{(k)})$ by means of a one-dimensional search method.

The method has high reliability and stability. However, the method has some drawbacks [1].

Taking as a parameter $\alpha^{(k)}$ in (1) a positive number α get computing scheme

$$x^{(k+1)} = x^{(k)} - \alpha \nabla f(x^{(k)}), \qquad (3)$$

which defines the simple gradient method in finding the minimum of multidimensional functions. The method has several disadvantages, including the following: first, you need to select the right value α; and secondly, the method is slow convergence to a minimum because of the smallness of the gradient ∇f in the neighborhood nearly stationary area.

Along with the above to solve optimization problems using Gauss-Seidel method, known under the name of the wise descent. The essence of the method is to find the minimum functions sequentially for each coordinate. Let the prioritized change coordinates $x_1, x_2, ..., x_n$ coinciding with the order of their indices 1, 2, ..., N. First change one component x_1 while keeping all other coordinates fixed by some amount Δx and determine the amount of the increment Δf . If It is therefore a step in the wrong direction. We change direction and move until $\Delta f < 0$ does not change the sign of a value. Then proceed to change the other coordinates x_2 and so on until it finds a specified precision point minimum. When you implement the other computational schemes.

The method is simple implementation. However, compared to other methods require more $\Delta f > 0$ to search the minimum functions.

1.2. Newton's Method

The above methods are based on consistent linear approximation of the objective function and require calculation of values of the objective function and its derivatives at each step, the number of which can be very large [1,3]. For a more general strategy to bring about the second derivatives $f(x)$. This information can be obtained when a quadratic approximation of functions, when its Taylor series expansion takes account of a number of members up to the second order, inclusive. Using the results of approximation leads to Newton's method is implemented by the formula

$$x^{(k+1)} = x^{(k)} - \nabla^2 f(x^{(k)})^{-1} \nabla f(x^{(k)}), \qquad (4)$$

where $\nabla^2 f(x)$ -Hessian (Hesse’s matrix).

Newton's method finds a quadratic rate of convergence, which is the inequality

$$\left\| \varepsilon^{(k+1)} \right\| \le c \left\| \varepsilon^{(k)} \right\|^2,$$

where c is a constant related to the conditions of the Hessian matrix.

$$\varepsilon^{(k)} = x^{(k)} - x^* ,$$

where x^* -solution.

Convergence of Newton's method depends largely on the choice of the initial approximation $x^{(0)}$. The method converges whenever choice $x^{(0)}$ is carried out in accordance with condition [1]

$$\left\| \varepsilon^{(0)} \right\| < 1/c \, .$$

Quadratic convergence speed due to the fact that the study of nekvadratičnyh functions of Newton's method is highly reliable/1, 3/. If the point located at a considerable distance from the point of Newton's method step is often too high, which can lead to non-convergence. Method can be quite easy to modify to ensure the reduction of the target function from iteration to iteration and search along a straight, as in the Cauchy. Iteration sequence is built in accordance with the formula

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} \nabla^2 f(x^{(k)})^{-1} \nabla f(x^{(k)}) \quad . \tag{6}$$

Choice $\alpha^{(k)}$ is carried out in such a way as to

$$f(x^{(k+1)}) \to \min \, .$$

This ensures that the inequality

$$f(x^{(k+1)}) \le f(x^{(k)}) \tag{7}$$

This technique is called modified Newton's method and when calculating the exact values of the first and second derivatives does not pose significant difficulties, is safe and effective.

1.3. Conjugate gradient Methods

These methods, with positive properties of Cauchy and Newton are based on calculating values only first derivatives, are highly reliable searches x^* from a remote location $x^{(0)}$ and quickly converge around point minimum. The build procedure methods Conjugate directions, for which the quadratic function approximation $f(x)$ and the value component of the gradient.

So, we believe that the objective function is quadratic:

$$f(x) = q(x) = a + b^T x + 1/2 \, x^T C x \, ,$$

and iteration are run according to formula (1), i.e.

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} \cdot s(x^{(k)}) .$$

Search direction on every iteration is defined by the following relations:

$$s^{(k)} = -g^{(k)} + \sum_{i=0}^{k-1} \gamma^{(i)} s^{(i)} , \tag{8}$$

$$s^{(0)} = -g^{(0)}, \tag{9}$$

where

$$g^{(k)} = \nabla f(\mathrm{x}^{(\mathrm{k})}) = \nabla \mathrm{g}(\mathrm{x}^{(\mathrm{k})}) = \mathrm{Cx}^{(\mathrm{k})} + b . \tag{10}$$

Using the quadratic function

$$\Delta g(x) = g(x^{(k)}) - g(x^{(k-1)}) = C(x^{(k)} - x^{(k-1)}) = C\Delta x$$

and c-contingency-directions $s^{(i)}$ (i= 1, 2, ...), the ratio can be obtained for computing parameter $\gamma^{(i)}$ in the formula (8) [1, 3], using the general formula (8) determine the direction of the search has the following form:

$$s^{(k)} = -g^{(k)} + \frac{\left\|g^{(k)}\right\|^2}{\left\|g^{(k-1)}\right\|^2} s^{(k-1)} . \tag{11}$$

If $f(x)$ -quadratic function, to find the point of minimum required N -1 of such areas and to N searches along a straight line. If the same function $f(x)$ is not quadratic, the number of destinations and related searches is increasing.

1.4. Other methods

In addition to those described above methods to meet the challenges of the unconditional multivariate optimization is used and many others [1, 2, 3, 6 .7, 8] which include quasi-Newtonian methods. These methods have the positive qualities of Newton's method, however, only use the first derivative. All methods of the specified class build vectors search referrals by using formula (1), which $s(x^{(k)})$ is written in the form

$$s(x^{(k)}) = A^{(k)} \nabla f(x^{(k)}) , \tag{12}$$

where $A^{(k)}$ is the matrix of order N*N, which is called a metric. Search methods along the lines determined by this formula are known as variable metric methods yet, because the matrix a is changed at each iteration. Changing matrix $A^{(k)}$ due to its proximity to the matrix of the inverse Hessian matrix. To approximate the inverse Hessian matrix, the ratio of recurrent

$$A^{(\kappa+1)} = A^{(\kappa)} + A_c^{(k)}, \qquad (13)$$

where $A_c^{(k)}$-adjustment matrix.

This matrix allows each step to improve the task.

Among the known methods of kvazin′ûtonovskih Dèvidona method-Flettčera-Powell (DPF), method Brojdena-Fletcher-Šènno, etc. [1, 3, 6, 7, 8].

The above methods have different performance indicators. Some research questions the effectiveness of the methods covered in this lab.

2. The implementation of software complex

Complex for the study and research of methods optimization is implemented nonjudgmental environment MATLAB.

After running the complex appears onscreen form that contains multiple Windows that you can use to specify methods of problem solving. Enhancing the multidimensional optimization leads to the emergence of a new on-screen forms to perform laboratory work. Window screen form allow you to specify any function, modify its parameters and the parameters of the study of algorithms to output tasks.

The top window screen form allows you to select the function, and in the next window, you can change its settings. The following window for setting the initial approach, accuracy, number of iterations, and step size are active. The study involves a step-by-step and automatic solution, and the results are displayed both graphically and in terms of quantitative evaluations. You can specify and three-dimensional graphics.

Background information explaining the characteristics of the implementation of each of the techniques is also displayed in the screen window of the form.

3. Implementation of the work

1. Get acquainted with the work of the software.
2. To find a solution to test and compare with the received solution.
3. Get the job options for work.
4. Investigate the effects of parameters specified targets on the effectiveness of decisions.
5. Examine the effects of the initial approximations, the accuracy of the indicators of performance techniques.

6. Get a solution for the user-defined function.
7. To issue a report with the results of the study methods.

LAB # 3

CONDITIONAL METHODS OF OPTIMIZATION

The purpose of the work: a study of the features of solution of optimization tasks using conditional methods of optimization.

1. Basic provisions

To find extremum target functions of many variables, you can use a variety of methods [1, 3, 5, 7]. This paper discusses how conditional optimization. Constraints on variables that may take the form of equations and inequalities. It addresses methods for solving constraint problems of both species.

1.1. The Adapted method Hook-1996

For solution of optimization tasks with constraints can be used method Hook-1996. The General procedure for finding the minimum of the objective function by changing the variables conditioning test valid decisions determined by the limitations of the task. And if the variable is outside the valid range, the objective function is given some great predefined value that adequately failure when looking for direction in the space of parameters. In the case of conditioning variables the General procedure feasible solution does not change [2].

1.2. Methods of Lagrange multipliers

To find extremum function

$$f(x) = f(x_1, x_2, ..., x_n) \qquad (1)$$

with restrictions

$$g_i(x_1, x_2, ..., x_n) = 0, \qquad i = 1, 2, ..., m. \quad (2)$$

You can use the classic method of conditional optimization functions of several variables [2, 5]. We believe that the functions $f(x_1, x_2, ..., x_n)$ and

$g_i(x_1, x_2, ..., x_n)$ together with its continuous first partial derivatives. The function is to solve the problem

$$F(x_1, x_2, ..., x_n, \lambda_1, \lambda_2, ..., \lambda_m) = f(x_1, x_2, ..., x_n) + \sum_{i=1}^{m} \lambda_i \cdot q_i(x_1, x_2, ..., x_n), \quad (3)$$

define partial derivatives $\frac{\partial F}{\partial x_j}(j = 1, 2, ..., n),\ \frac{\partial F}{\partial \lambda_i}(i = 1, 2, ..., m)$ and are equal to zero, resulting in a system of equations:

$$\begin{cases} \frac{\partial F}{\partial x_j} = \frac{\partial f}{\partial x_j} + \sum_{i=1}^{m} \lambda_i \frac{\partial g}{\partial x_j} = 0 \\ \frac{\partial F}{\partial x_j} = g_i(x_1, x_2, ..., x_n) = 0 \end{cases} \text{ where } i = 1, 2, ..., m \text{ and, } j = 1, 2, ..., n. \quad (4)$$

Function (3) is called the Lagrangian function, and the number of λ_i -Lagrange multipliers. If the function $f(x) = f(x_1, x_2, ..., x_n)$ at the point of $X^{(0)} = (x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)})$ has the extremum, then there exists a vector $\Lambda^{(0)} = (\lambda_1^{(0)}, \lambda_2^{(0)}, ..., \lambda_m^{(0)})$ that point $(x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)}, \lambda_1^{(0)}, \lambda_2^{(0)}, ..., \lambda_m^{(0)})$ is the solution of the system. Therefore, solving the system, get lots of points at which the function $f(x)$ can have extreme values.

You can use the Lagrangian method in the case of restrictions in the form of inequalities. By introducing additional variables, constraints are inequalities can be converted to the equation, with additional variables restricted neotricatel′nosti.

1.3. Method of penalty functions

The method of numerical solution of optimization problems with constraints method of conversion-based optimization tasks [1, 2, 3, 9].

The task can be formulated as follows:

minimize $f(x), \quad x \in R^N$ (1)

with restrictions $g_i(x) \geq 0,\ j = 1, 2, ..m,$ (2)

$h_i(x) = 0, i = 1, 2, ...l.$ (3)

The essence of the method is to convert the source to the target function (1) by incorporating features from the limits (2) and (3), thus obtaining the unconditional optimization, which you can use known techniques [1, 2, 3].

The transformed function defined by

$$P(x,R) = f(x) + \Omega(R, g(x), h(x)), \qquad (4)$$

where Ω -penalty function of restrictions and R -penalty parameter.
There are different types of fines and various procedures taking into account constraints when you navigate to a task without optimization.

1. *Quadratic penalty*

This type of penalty is used to account for the limitations of equations and has the form

$$\Omega = R(h(x))^2 . \qquad (5)$$

In the minimization of this penalty prevents rejection of the values of $h(x)$ from the ground up. It is easy to see that if you increase the R the stationary point of a function P(x, R) is coming to the decision x^* because in the limit $h_k(x^{(t)}) = 0$ where t = 1, 2, ..., T. Function. Ω is continuous and has continuous derivatives.

2. Logarithmic fine

This and the following penalties shall take into account the constraints of inequality. Logarithmic penalty has the form

$$\Omega = -R \ln[g(x)] \qquad (6)$$

The fine is positive for all x such that 0 < g(x) < 1, and negative if g(x)> 1. In this case, the internal points of the acceptable solutions will be given preference. Logarithmic barrier function is fine, not described in the invalid pixels (i.e. such that g(x) < 0). Therefore, the initial step is to ensure penetration into the permissible range. Since the transformed problem is solved one of the numerical methods, it is possible the emergence of unacceptable points in the course of the decision (for example, as a result of a great first step in the search for the one-dimensional). In this regard, special measures shall be provided to prevent this situation or its detection and repair. Iterative process starts from a valid positive initial point R (R= 10 or R= 100). following the decision of each subtask unconditional optimization decreases and in the limit of zero.

3. The penalty specified by reciprocal function

This type of penalty

$$\Omega = R[1/g(x)] \tag{7}$$

has no negative values in the acceptable area. Like the previous one, is a barrier to a fine in the valid points near the border values of fine positive and quickly wanes when advancing into the permissible area. On the border value of P(x, R) and its gradient is not defined. As in the previous case, you may receive incorrect points.

4. The penalty-type square cutting

$$\Omega = R \cdot \langle g(x) \rangle^2 , \tag{8}$$

where $\langle a \rangle = \begin{cases} a, & if \;\; a \le 0, \\ 0, & if \;\; a > 0. \end{cases}$ (9)

Note that this fine external. Invalid point do not create difficulties in this case compared with valid. The distinction between them is that in valid and boundary points penalty is 0. this type of penalty is convenient because P(x, R) is continuous and everywhere. The calculation is carried out with positive R rising from iteration to iteration.

In this lab, discusses the challenges of non-linear programming method of penalty functions taking into account constraints-constraints-equations and inequalities.

Algorithm for solving the problem can be represented as the following sequence of steps

Step 1. Specify the initial data N, J, k , $\varepsilon_{1,} \varepsilon_{2,} \varepsilon_{3,}$ $x^{(0)}$, $R^{(0)}$, where

ε_1 – the end of a one-dimensional search (if it is used in the procedure of unconditional optimization);

ε_2 - the end of the procedure the unconditional optimization;

ε_3 – the end of the algorithm;

$x^{(0)}$ is the starting point;

$R^{(0)}$ -initial vector of penalty parameters.

Step 2. Build the penalty function

$$P(x,R) = f(x) + \Omega(R, g(x), h(x)).$$

Step 3. Find $x^{(t+1)}$ that delivers the extremum $P(x^{(t+1)}, R^{(t)})$ with a fixed $R^{(t)}$. As a starting point is used $x^{(t)}$, and as the end step parameter – the constant ε_2.

Step 4. Check whether the condition $\left|P(x^{(t+1)}, R^{(t)}) - P(x^{(t)}, R^{(t-1)})\right| \le \varepsilon_3$. If it is, put $x^* = x^{(t+1)}$ and finish the process address; otherwise, go to step 5.

Step 5. Put $R^{(t+1)} = R^{(t)} + \Delta R$ in accordance with any conversion rule and proceed to step 2.

There are no clear guidelines on choosing R. Either R is incremented by a certain number ΔR, either as R use increasing degrees of some number (e.g. 10, 100, 1000, etc.).

When implementing step 3 can be used to optimize any procedure: Hook-Jeeves method, simplex search. If it is possible to calculate $\frac{\partial P}{\partial x_i}$, then apply any gradient search procedure.

2. The implementation of software complex

Center for study and research methods in the optimization of conditional MATLAB.

After the launch complex on the screen that appears, you click form, the corresponding conditional optimization techniques, increasing which leads to another screen form.

Work in the window that appears is similar to working with Windows of the preceding paragraphs, the complex is provided with relevant information and does not require further explanation.

3. Implementation of the work

1. To familiarize themselves with the work of software.
2. To find a solution to the previously selected test tasks for all kinds of fines.
3. Get a job at the teacher to complete the work.
4. Investigate the influence of fines and their parameters on the accuracy of decisions.
5. Execute the report with the results of decisions and research methods.

LAB No. 4

SOLUTION OF LINEAR PROGRAMMING PROBLEMS

Aim: **to study the methods of the solution of linear programming problems and their application to the study of applications**.

1. General linear programming problem

The General objective of linear programming (LP) is to find the extreme values (maximum or minimum) of linear function

$$Z = C_1x_1 + C_2x_2 + \ldots + C_nx_n \qquad (1.1)$$

from physical variables when imposed restrictions:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n \le (=,\ge) b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2j}x_j + \ldots + a_{2n}x_n \le (=,\ge) b_2 \\ \ldots \\ a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n \le (=,\ge) b_i \\ \ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mj}x_j + \ldots + a_{mn}x_n \le (=,\ge) b_m \end{cases}, \qquad (1.2)$$

$$x_j \ge 0 \ \ (j = 1, 2, \ldots, n), \qquad (1.3)$$

where a_{ij}, b_i and C_j are the defined constants.

A linear function that is extreme value, commonly referred to as *the target function*.

The restrictions can simultaneously meet marks *is less than or equal to, equal to, greater than or equal to*.

The overall objective has several forms of writing.

Vector linear programming task form has the following form:

minimize (maximize) linear function

$$Z = \mathbf{CX} \tag{1.4}$$

with restrictions

$$\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \ldots + \mathbf{A}_n x_n \le (=, \ge) \mathbf{A}_0, \ \mathbf{X} \ge 0, \tag{1.5}$$

where $\mathbf{C} = (c_1, \quad c_2, \quad \ldots, \quad c_n)$;

$\mathbf{X} = (x_1, \quad x_2, \quad \ldots, \quad x_n)$;

$\mathbf{CX}$ is the scalar product.

Vectors

$$\mathbf{A}_1 = \begin{pmatrix} a_{11} \\ a_{21} \\ \cdots \\ a_{m1} \end{pmatrix}, \quad \mathbf{A}_2 = \begin{pmatrix} a_{12} \\ a_{22} \\ \cdots \\ a_{m2} \end{pmatrix}, \quad \cdots, \quad \mathbf{A}_n = \begin{pmatrix} a_{1n} \\ a_{2n} \\ \cdots \\ a_{mn} \end{pmatrix}, \quad \mathbf{A}_0 = \begin{pmatrix} b_1 \\ b_2 \\ \cdots \\ b_m \end{pmatrix} \tag{1.6}$$

are the coefficients of the unknowns and free members.

Matrix form of linear programming task involves finding the minimum (maximum) value of linear function

$$Z = \mathbf{CX} \tag{1.7}$$

with restrictions

$$\mathbf{AX} \le (=, \ge) \mathbf{A}_0, \ \mathbf{X} \ge 0, \tag{1.8}$$

where $\mathbf{C} = \begin{pmatrix} c_1 & c_2 & \cdots & c_n \end{pmatrix}$ is the string-matrix;

$$\mathbf{X} = \begin{pmatrix} x_1 \\ x_2 \\ \cdots \\ x_m \end{pmatrix}, \ \mathbf{A}_0 = \begin{pmatrix} b_1 \\ b_2 \\ \cdots \\ b_m \end{pmatrix}$$ – the matrix-column;

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$ is the matrix of coefficients of the system constraints.

Example 1

Find the maximum value of a linear function $Z = -x_1 + x_2 + 3x_3$ with restrictions

$$\begin{cases} 2x_1 & -x_2 & +x_3 & \leq 1 \\ 4x_1 & -2x_2 & +x_3 & \geq -2. \\ 3x_1 & & +x_3 & \leq 5 \end{cases}$$

The vectors are of the form:

$$\mathbf{C} = \begin{pmatrix} -1 & 2 & 3 \end{pmatrix},$$

$$\mathbf{X} = \begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix},$$

$$\mathbf{A}_1 = \begin{pmatrix} 2 \\ 4 \\ 3 \end{pmatrix}, \quad \mathbf{A}_2 = \begin{pmatrix} -1 \\ -2 \\ 0 \end{pmatrix}, \quad \mathbf{A}_3 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \quad \mathbf{A}_0 = \begin{pmatrix} 1 \\ -2 \\ 5 \end{pmatrix}.$$

The matrix system of restrictions is as follows:

$$\mathbf{A} = \begin{pmatrix} 2 & -1 & 1 \\ 4 & -2 & 1 \\ 3 & 0 & 1 \end{pmatrix}.$$

2. Simplex method of linear programming

2.1. Standard form of linear programming tasks

Linear programming problems presented in different ways, can be reduced to standard form. Standard form is to minimize the objective function

$$Z = C_1 x_1 + C_2 x_2 + \ldots + C_n x_n \qquad (2.1)$$

in view of the limitations of equalities:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2j}x_j + \ldots + a_{2n}x_n = b_2 \\ \ldots \\ a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n = b_i \\ \ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mj}x_j + \ldots + a_{mn}x_n = b_m \end{cases}, \qquad (2.2)$$

$$x_j \geq 0 \ \left(j = 1, 2, \ldots, n \right), \qquad (2.3)$$

$$b_i \geq 0 \ \left(i = 1, 2, \ldots, m \right). \qquad (2.4)$$

If a task in the form of (2.1) to (2.4) the condition $m > n$ then the LP is reduced to the solution of systems of equations (2.2). This task will not have solutions if the condition (2.3) is not running or the system of equations has no solution. The LP will not have decisions and when $m < n$ the system of restrictions has no solution.

Here are the tasks for which the condition C_j.

To the task of finding the maximum value of the objective function (1.1) move on to the task of minimization, enough to take all coefficients the target function with signs. To navigate back after finding the minimum result so you must invert.

For the transition from the type constraint *is less than or equal* to the equity in it, you must enter additional nonnegative variable with sign «plus»:

$$a_{i1}x_1 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n \le b_1$$
$$\Rightarrow \quad a_{i1}x_i + \ldots + a_{ij}x_j + \ldots + a_{in}x_n + x_{n+1} = b_i \quad (2.5)$$

For the transition from the type constraint *is greater than or equal* to the equity in it, you must enter additional nonnegative variable with a minus sign:

$$a_{i1}x_1 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n \ge b_1$$
$$\Rightarrow \quad a_{i1}x_i + \ldots + a_{ij}x_j + \ldots + a_{in}x_n - x_{n+1} = b_i \quad (2.6)$$

In each inequality is it's $(n+i)$ an additional variable.

All equality with negative free members, divided into -1 in order to satisfy the condition (2.4).

Example 2

Convert task, as set out in example 1 to the standard form. The system of restrictions is as follows:

$$\begin{cases} 2x_1 & -x_2 & +x_3 & \le 1 \\ 4x_1 & -2x_2 & +x_3 & \ge -2. \\ 3x_1 & & +x_3 & \le 5 \end{cases}$$

Multiplying the second constraint by -1 for the non-negativity conditions of the free terms in the constraints:

$$\begin{cases} 2x_1 & -x_2 & +x_3 & \le 1 \\ -4x_1 & +2x_2 & -x_3 & \le 2. \\ 3x_1 & & +x_3 & \le 5 \end{cases}$$

We will introduce in the first inequality more variable $x_4 \ge 0$ with sign Plus, the second $-x_5 \ge 0$ and in the third $-x_6 \ge 0$ also the plus. The result will be a system of restrictions in standard form:

$$\begin{cases} 2x_1 & -x_2 & +x_3 & +x_4 & & & =1 \\ -4x_1 & +2x_2 & -x_3 & & +x_5 & & =2. \\ 3x_1 & & +x_3 & & & +x_6 & =5 \end{cases}$$

While these restrictions will need to find the minimum value of the function with coefficients:

$$Z = x_1 - x_2 - 3x_3.$$

2.2. The concept plan for the linear programming problem

PL task, presented in a standard format, introduced the concept *of the plan*.

The plan, or *a valid solution to* the linear programming problem, referred to as a vector $\mathbf{X} = (x_1, \quad x_2, \quad \ldots, \quad x_n)$ that satisfies the conditions (2.2), (2.3) and (2.4).

The plan is called the *baseline* if the vectors $\mathbf{A}_i \quad (i = 1, 2, \ldots, m)$ in the decomposition type (1.5) with positive coefficients x_i (standard forms) are linearly independent. Number of positive reference component of the plan may not exceed m. The support plan is called non-degenerate if it contains m positive components, otherwise the support program is called degenerate.

The optimal plan or the optimal solution of linear programming tasks is referred to as a plan that delivers the lowest value of linear function (2.1).

The set of all linear programming tasks plans (if they exist) is convex bi-hex profile. Each corner point of the polyhedron solutions meets the basic plan. Each basic plan is determined by the system m linearly independent vectors contained in the system of n vectors . $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n$

In order to find the optimal plan to explore the only anchor plans. The upper bound of the number of backup plans contained in this task, is determined by the number of combinations C_n^m. When large m and n find the best plan, turning over all support plans task very difficult.

Simplex method provides a schema that allows a smooth transition from one key to another plan. This method, based on known reference plan tasks for a finite number of steps to get the optimal plan. Each of the steps (or iterations) is to find a new plan, which is less than the value of the linear function the same as in the previous plan. If the task has no plans or linear function is not limited to the polyhedron solutions the simplex method allows you to set this in the decision.

2.3. Building support programs

To build initial support plan should be made in the system of restrictions (2.2) *m* linearly independent vectors. The most simple is made by converting the system of restrictions so that it appeared unit vectors (submatrix unit):

$$\begin{cases} x_1 \qquad\qquad\qquad +a_{1,m+1}x_{m+1}+\ldots+a_{1n}x_n=b_1 \\ \quad x_2 \qquad\qquad +a_{2,m+1}x_{m+1}+\ldots+a_{2n}x_n=b_2 \\ \qquad \ldots \\ \qquad\qquad x_m \quad +a_{m,m+1}x_{m+1}+\ldots+a_{mn}x_n=b_n \end{cases}, \ b_i \geq 0 \ \left(j=1,2,\ldots,m\right). \quad (2.7)$$

In vector form system (2.7) is as follows:

$$x_1\mathbf{A}_1+x_2\mathbf{A}_2+\ldots+x_m\mathbf{A}_m+x_{m+1}\mathbf{A}_{m+1}+\ldots+x_n\mathbf{A}_n=\mathbf{A}_0, \qquad (2.8)$$

where

$$\mathbf{A}_1=\begin{pmatrix}1\\0\\\cdots\\0\\0\end{pmatrix}, \quad \mathbf{A}_2=\begin{pmatrix}0\\1\\0\\\cdots\\0\end{pmatrix}, \quad \cdots, \quad \mathbf{A}_m=\begin{pmatrix}0\\0\\\cdots\\0\\1\end{pmatrix};$$

$$\mathbf{A}_{m+1}=\begin{pmatrix}a_{1,m+1}\\a_{2,m+1}\\\cdots\\a_{m,m+1}\end{pmatrix}, \quad \cdots, \quad \mathbf{A}_n=\begin{pmatrix}a_{1n}\\a_{2n}\\\cdots\\a_{mn}\end{pmatrix}, \quad \mathbf{A}_0=\begin{pmatrix}b_1\\b_2\\\cdots\\b_m\end{pmatrix}.$$

Vectors $\mathbf{A}_1,\mathbf{A}_2,\ldots,\mathbf{A}_m$ are linearly independent unit vectors of *m*-dimensional space. They form the basis of this space. Any of the vectors $\mathbf{A}_j$ can be represented as a linear combination of basis vectors, and the only way:

$$\mathbf{A}_j=\sum_{i=1}^{m}x_{ij}\mathbf{A}_i, \ i=0,1,2,\ldots n. \qquad (2.9)$$

If the decomposition (2.7) for the unknown base select $x_1,x_2,\ldots,x_m$ free unknown $x_{m+1},\ldots,x_n$ equates to zero and, given that $b_i \geq 0 \ \left(j=1,2,\ldots,m\right)$ and vectors $\mathbf{A}_1,\mathbf{A}_2,\ldots,\mathbf{A}_m$ are single and get an initial basic plan:

$$\mathbf{X}_0 = \left(x_1 = b_1; \quad x_2 = b_2; \quad \ldots \quad x_m = b_m; \quad x_{m+1} = 0; \quad \ldots \quad x_n = 0 \right). \qquad (2.10)$$

If you ask a certain amount $\theta > 0$ the vector

$$\mathbf{X}_1 = \left(x_1 - \theta x_{1,m+1}; \quad x_2 - \theta x_{2,m+1}; \quad \ldots \quad x_m - \theta x_{m,m+1}; \quad \theta; \quad 0 \quad \ldots \quad 0 \right)$$

also is a plan if its negative features.

Since $\theta > 0$, all components of the vector $\mathbf{X}_1$, which include non-positive $x_{i,m+1}$ are non-negative. You must define the $\theta > 0$ under which for all $x_{i,m+1} > 0$ the following condition is satisfied:

$$x_i - \theta x_{i,m+1} \geq 0. \qquad (2.11)$$

From (2.11) should be $\theta \leq x_i / x_{i,m+1}$. Therefore, the vector $\mathbf{X}_1$ is the Plan tasks for any θ that satisfies a condition

$$0 < \theta \leq \min \left[x_i / x_{i,m+1} \right], \qquad (2.12)$$

where the minimum is taken *i* for which $x_{i,m+1} > 0$.

Because the basic plan may not contain *m+1* positive component, so in $\mathbf{X}_1$ it is necessary to pay zero in at least one of the component. If you put

$$\theta = \theta_0 = \min \left[x_i / x_{i,m+1} \right], \qquad (2.13)$$

the component of the plan $\mathbf{X}_1$ that is at least becomes zero, the transition to the new support plan:

$$\mathbf{X}_1 = \left(0; \quad x_2`; \quad \ldots \quad x_m`; \quad x_{m+1}`; \quad 0 \quad \ldots \quad 0 \right). \qquad (2.14)$$

2.4 Finding optimal plan

If a linear programming problem (2.1)-(2.4) has the plans and each of its basic plan is not degenerate, then for the support program (2.10) the following relation is satisfied:

$$x_1 \mathbf{A}_1 + x_2 \mathbf{A}_2 + \ldots + x_m \mathbf{A}_m = \mathbf{A}_0, \qquad (2.15)$$

$$x_1C_1 + x_2C_2 + \ldots + x_mC_m = Z\left(\mathbf{A}_0\right), \tag{2.16}$$

where all $x_i > 0$;

$Z\left(\mathbf{A}_0\right)$ – the value of the objective function corresponding to this plan.

Decomposition of a vector $\mathbf{A}_j$ on this basis the only vectors:

$$x_{1j}\mathbf{A}_1 + x_{2j}\mathbf{A}_2 + \ldots + x_{mj}\mathbf{A}_m = \mathbf{A}_j \;\; \left(j = 1, 2, \ldots, n\right). \tag{2.17}$$

This corresponds to a single value decomposition of linear function

$$x_{1j}C_1 + x_{2j}C_2 + \ldots + x_{mj}C_m = Z_j \;\; \left(j = 1, 2, \ldots, n\right), \tag{2.18}$$

where Z_j – the value of the linear function if it instead of unknown substitute the corresponding coefficients of decomposition *j*-th vector by basis.

If, for a specific plan $\tilde{\mathbf{X}}$ decomposition of all vectors $\mathbf{A}_j$ $\left(j = 1, 2, \ldots, n\right)$ satisfies the condition

$$Z_j - C_j \leq 0, \tag{2.19}$$

the plan is the best.

Inequality (2.19) are a condition of optimality of the plan tasks to be solved by finding the minimum value of linear functions and values $Z_j - C_j$ referred to as *estimate* of the plan.

Thus, in order to plan tasks to be optimal, it is necessary and sufficient that its estimates were non-positive.

3. Algorithm of simplex method

3.1. Build Simplex table

The next step after the task is reduced to the standard form, the system of restrictions highlighted submatrix unit and received initial support program in the form (2.10), it is necessary to examine this plan optimality. The vectors $\mathbf{A}_j$ $\left(j = 1, 2, \ldots, n\right)$ of the system are decomposed on the basis vectors $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_m$ and count values assessments $Z_j - C_j$. The basis of an isolated, so the coefficients of the decomposition of a vector $\mathbf{A}_j$ on the baseline are its components, i.e. $x_{ij} = a_{ij}$ $\left(i = 1, 2, \ldots, m; j = 1, 2, \ldots, n\right)$. This means that columns $\mathbf{A}_j$ $\left(j = 1, 2, \ldots, n\right)$ of the table to record the type of constraint matrix (2.7). Further calculations easier to carry out if the condition and initial data obtained after the first key plan, written in simplex table (table 1).

Table 1

i	Basis	**C** Basis	$\boldsymbol{a}_0$	c_1	c_2	…	c_l	…	C_m	C_{m+1}	…	c $to_{(j)}$	…	C_k	…	C_n
				$\boldsymbol{a}_1$	$\boldsymbol{a}_2$	…	$\boldsymbol{a}_L$	…	$\boldsymbol{a}_{(m)}$	$\boldsymbol{a}_{m+1}$	…	$\boldsymbol{a}_J$	…	$\boldsymbol{a}_k$	…	$\boldsymbol{a}_N$
1	$\boldsymbol{a}_1$	c_1	X_1	1	0	…	0	…	0	$x_{1,\,m+1}$	…	x_{1j}	…	$x_{1\,k}$	…	x_{1n}
2	$\boldsymbol{a}_2$	c_2	X_2	0	1	…	0	…	0	$x_{2,\,m+1}$	…	x_{2j}	…	$x_{2\,k}$	…	x_{2n}
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
L	$\boldsymbol{a}_L$	c_l	x_l	0	0	…	1	…	0	$x_{L,\,(m)+1}$	…	x_{lj}	…	$\boldsymbol{x_{lk}}$	…	x_{ln}
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
(m)	$\boldsymbol{a}_{(m)}$	C_m	x_m	0	0	…	0	…	1	$x_{m,\,m+1}$	…	x_{mj}	…	x_{mk}	…	x_{mn}
$M+1$	Z_j-C_j		Z_0	0	0	…	0	…	0	Z_{m+1}- C_{m+1}	…	Z_j- C_j	…	Z_k- C_k	…	Z_n- C_n

In the column **C** of *basis* are the coefficients of linear function corresponding to the vectors of the basis. In the column $\mathbf{A}_0$ – initial basic plan $\mathbf{X}_0$. It is as a result of calculations produced optimum plan and columns $\mathbf{A}_j$ $\left(j = 1, 2, \ldots, n\right)$ are coefficients of the decomposition *j*-th vector on the basis described in the future through $\mathbf{X}_j$.

In (*m*+*1*)-th row in the column $\mathbf{A}_0$ recorded values of linear function $Z\left(\mathbf{X}_0\right)$. It makes the results the basic plan, and columns $\mathbf{A}_j$ – value assessments $Z_j - C_j$.

Function $Z\left(\mathbf{X}_0\right)$ and $Z_j = Z\left(\mathbf{X}_j\right)$ find substituting a linear function of the key components of the plan, respectively, and the coefficients of the expansion *j*-th vector of vectors, basis, so those values in table 1 can be obtained as the dot product:

$$Z\left(\mathbf{X}_0\right) = \mathbf{C}_б \mathbf{X}_0 = \sum_{i=1}^{m} C_i x_i\,, \tag{3.1}$$

$$Z_j = \mathbf{C}_б \mathbf{X}_j = \sum_{i=1}^{m} C_i x_{ij}\,,\ \ j = 1, 2, \ldots, n\,, \tag{3.2}$$

where C_j – coefficients of linear function corresponding to the vectors of the basis.

Example 3.1

We shall make the initial Simplex table for tasks in example 2.

Vectors $\mathbf{A}_4$, $\mathbf{A}_5$ and $\mathbf{A}_6$, form a single submatrix and form the basis of the initial plan, the free unknown are zero. The result is an initial basic plan:

$$\mathbf{X}_0^{(1)} = (0;\quad 0;\quad 0;\quad 1;\quad 2;\quad 5).$$

Let's calculate the value of $(m+1)$ row:

$$Z(\mathbf{X}_0) = \mathbf{C}_б \mathbf{X}_0 = 0;$$

$$Z_1 = \mathbf{C}_б \mathbf{X}_1 = 0;\quad Z_2 = \mathbf{C}_б \mathbf{X}_2 = 0;\quad Z_3 = \mathbf{C}_б \mathbf{X}_3 = 0;$$

$$Z_1 - C_1 = 0 - 1 = -1;\quad Z_2 - C_2 = 0 + 1 = 1;\quad Z_3 - C_3 = 0 + 3 = 3.$$

Table 2

i	Basis	**c** basis	***a₀***	*C₁= 1*	*c₂= -1*	*C₃= -3*	*C₄= 0*	*C₅= 0*	*c₆= 0*
				$\boldsymbol{a}_1$	$\boldsymbol{a}_2$	$\boldsymbol{a}_3$	$\boldsymbol{a}_4$	$\boldsymbol{a}_5$	$\boldsymbol{a}_6$
1	$\boldsymbol{a}_4$	*0*	*1*	*2*	*-1*	**1**	*1*	*0*	*0*
2	$\boldsymbol{a}_5$	*0*	*2*	*-4*	*2*	*-1*	*0*	*1*	*0*
3	$\boldsymbol{a}_6$	*0*	*5*	*3*	*0*	*1*	*0*	*0*	*1*
m +1	Z_j-C_j		*0*	*-1*	*1*	*3*	*0*	*0*	*0*

3.2 Search allowing the simplex table

After the table 1 analyses $(m+1)$ string. If for all $j = 1, 2, \ldots, n$ value assessments $Z_j - C_j$ less than zero, the basic plan $\mathbf{X}_0$ is optimal and minimum value of the linear function is $Z(\mathbf{X}_0)$. Otherwise, you can include the basis vector corresponding to a positive assessment, build a basic plan, which corresponds to a lower value of the linear function.

If the number of positive evaluations, the basis should be included in the vector, which corresponds to the max $\left[\theta_{0j}\left(Z_j - C_j\right)\right]$because the decreasing value of the objective function is the value of $\theta_{0j}\left(Z_j - C_j\right)$. Maximum is taken on the *j* for which the score is positive and θ_{0j} to be determined for each *j*. This allows for this step, go to the top of the decisions related to the polyhedron the greatest decrease in linear functions.

Let the basis is included *k*-th vector, then excluded from the basis of the vector, which corresponds to the $\theta_{0k} = \min[x_i / x_{ik}]$ $(x_{ik} > 0)$. If this condition is fulfilled for the vector basis $\mathbf{A}_l$. Element x_{lk} is called *an allow*and and column and line, at the intersection of which it is located, is *directing*. New reference plan represents the basis consisting of $\mathbf{A}_1, \ldots, \mathbf{A}_{l-1}, \mathbf{A}_k, \mathbf{A}_{l+1}, \ldots, \mathbf{A}_m$.

3.3. The transition to the new reference plan

To calculate the new basic plan and test it for continuity, you must distribute all vectors vectors new basis. The new basic plan and expansion of vectors in the basis when $j = 1, 2, \ldots, n$ is defined by the equations:

$$\begin{cases} x'_{ij} = x_{ij} - \dfrac{x_{ij}}{x_{lk}} & (i \neq l) \\ x'_{ij} = \dfrac{x_{ij}}{x_{lk}} & (i = j) \end{cases}, \qquad (3.3)$$

which are the formulas of Gauss-Jordan total exclusions.

So to get the rates of decomposition of vectors $\mathbf{A}_0$, $\mathbf{A}_j$ $(j = 1, 2, \ldots, n)$ on the new basis vectors, value assessments of new reference plan and values of linear function, you need to divide all the elements of the guide lines to allow entry and by producing a full conversion of the Gauss-Jordan method, using the converted string to compose a new Simplex table.

3.4 Change the end Conditions

If a simplex table $(m+1)$ row of all ratings $(Z_j - C_j) \leq 0$, the resulting plan is optimal. If there are positive, then you look for the next basic plan.

If at least one positive assessment of the coefficients of the expansion x_{ij} of the vector, the nonlinear function is not limited to the polyhedron solutions. The linear function can be arbitrarily small value.

The process continues until the optimal plan, or to establish an linear function problem. If estimates of optimal plan evaluation, only the zero reference vectors, it talks about the uniqueness of the optimal plan.

If the system limitations of linear programming task contains a single basis, then the challenge should be approximately iterations.

Example 3.2

Will continue the consideration of the tasks of the example 3.1.

In $(m+1)$ row of the original table (table 2), there are two positive reviews. They correspond to vectors $\mathbf{A}_2$ and $\mathbf{A}_3$. This means that the original plan is not optimal and it could be improved by incorporating in the basis vector by max $\left[\theta_{0j}\left(Z_j - C_j\right) > 0\right]$. Among the coefficients of expansion of vectors $\mathbf{A}_2$ and $\mathbf{A}_3$ on the basis there are positive, so $\theta_{02} > 0$ and $\theta_{03} > 0$ which exclude from the basis of vectors exist. Find these values:

$$\theta_{02} = 2/2\,;$$

$$\theta_{03} = \min\left(1/1, \quad 5/1\right) = 1;$$

$$\theta_{02}\left(Z_2 - C_2\right) = 1 \cdot 1 = 1;$$

$$\theta_{03}\left(Z_3 - C_3\right) = 1 \cdot 3 = 3;$$

$$\max\left(1, \quad 3\right) = 3.$$

Thus allowing the element is the number of *1* standing in the first row and third column. This means that vector $\mathbf{A}_3$ include in the basis, and the vector $\mathbf{A}_4$ deleted.

Make second Simplex table (table 3).

Table 3

i	Basis	**c** basis	***a****0*	$C_1 = 1$	$c_2 = -1$	$C_3 = -3$	$C_4 = 0$	$C_5 = 0$	$c_6 = 0$
				$\boldsymbol{a}_1$	$\boldsymbol{a}_2$	$\boldsymbol{a}_3$	$\boldsymbol{a}_4$	$\boldsymbol{a}_5$	$\boldsymbol{a}_6$
1	$\boldsymbol{a}_3$	*-3*	*1*	*2*	*-1*	*1*	*1*	*0*	*0*
2	$\boldsymbol{a}_5$	*0*	*3*	*-2*	**1**	*0*	*1*	*1*	*0*
3	$\boldsymbol{a}_6$	*0*	*4*	*1*	*1*	*0*	*-1*	*0*	*1*
m +1	Z_j-C_j		*-3*	*-7*	*4*	*0*	*-3*	*0*	*0*

Let us count the new rail line. The old items guide lines divide the allow entry (*1*) and with the help of the received string will make one conversion method of total exclusions, i.e., add second and subtract from third.

The values in the $(m+1)$ row are similar to the example 3.1.

In table 3 is the second basic plan $\mathbf{X}_0^{(2)} = (0;\quad 0;\quad 1;\quad 0;\quad 3;\quad 4)$, which corresponds to the value of the linear function $Z\left(\mathbf{X}_0^{(2)}\right) = -3$. The second plan is not optimal, because the score in $(m+1)$ row is $(Z_2 - C_2) = 4 > 0$.

Define $\theta_{02} = \min(3/1,\quad 4/1) = 3$. The number of 1, standing at the intersection of the second row and second column is an allow element vector $\mathbf{A}_5$ excluded from the baseline. Make up a third of the simplex table (table 4).

Table 4

i	Basis	**c** basis	***a₀***	$C_1 = 1$	$c_2 = -1$	$C_3 = -3$	$C_4 = 0$	$C_5 = 0$	$c_6 = 0$
				a₁	***a₂***	***a₃***	***a₄***	***a₅***	***a₆***
1	***a₃***	*-3*	*4*	*0*	*0*	*1*	*2*	*1*	*0*
2	***a₂***	*-1*	*3*	*-2*	*1*	*0*	*1*	*1*	*0*
3	***a₆***	*0*	*1*	**3**	*0*	*0*	*-2*	*-1*	*1*
m +1	Z_j-C_j		*-15*	*1*	*0*	*0*	*-7*	*-4*	*0*

In table 4 is the third basic plan $\mathbf{X}_0^{(3)} = (0\quad 3;\quad 4;\quad 0;\quad 0;\quad 1)$, which corresponds to the value of the linear function $Z\left(\mathbf{X}_0^{(3)}\right) = -15$. The third plan is the same is not optimal, because $(Z_1 - C_1) = 1 > 0$. Is the next simplex table 5.

Table 5

i	Basis	**c** basis	***a₀***	$C_1 = 1$	$c_2 = -1$	$C_3 = -3$	$C_4 = 0$	$C_5 = 0$	$c_6 = 0$
				a₁	***a₂***	***a₃***	***a₄***	***a₅***	***a₆***
1	***a₃***	*-3*	*4*	*0*	*0*	*1*	*2*	*1*	*0*
2	***a₂***	*-1*		*0*	*1*	*0*			
3	***a₁***	*1*		*1*	*0*	*0*			
m +1	Z_j-C_j			*0*	*0*	*0*			

In table 5, it can be concluded that the plan

$$\mathbf{X}_0^{(4)} = (1/3;\quad 11/3;\quad 4;\quad 0;\quad 0;\quad 0)$$

is the best and only. The minimum value of a linear function is at the point and equal . $\mathbf{X}_0^{(4)}$ и равно $-46/3$.

4. Dual linear programming problems

4.1. Direct and dual problem

With each task is closely related to other linear programming linear task, called a dual or less in relation to the reference or direct [1, 5]. Will define the dual tasks in relation to the original problem of linear programming to find the maximum value of the function

$$Z = c_1x_1 + c_2x_2 + \ldots + c_nx_n \tag{4.1}$$

with restrictions

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \le b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \le b_2 \\ \ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \le b_m \end{cases} \tag{4.2}$$

$$(j = 1, 2, \ldots, n). \tag{4.3}$$

Task of finding the minimum value of the function

$$Z^{'} = b_1x_1 + b_2x_2 + \ldots + b_mx_m \tag{4.4}$$

with restrictions

$$\begin{cases} a_{11}y_1 + a_{21}y_2 + \ldots + a_{m1}y_m \le c_1 \\ a_{12}y_1 + a_{22}y_2 + \ldots + a_{m2}y_m \le c_2 \\ \ldots \\ a_{1n}y_1 + a_{2n}y_2 + \ldots + a_{mn}y_m \le c_m \end{cases}, \tag{4.5}$$

$$y_j \ge 0 \ \ (i = 1, 2, \ldots, m). \tag{4.6}$$

is called the dual task (4.1) and (4.3).

Tasks (4.1) and (4.3) and (4.4) to (4.6) form a couple of tasks, called linear programming dual pair.

The comparison of two targets shows that the dual problem in relation to the reference shall be drawn up according to the following rules:

1. The target function has the opposite type of extremum.

2. the matrix of coefficients of the dual tasks resulting from the transposition of the matrix of coefficients of the original problem.

3. The number of variables in the dual task (4.4) to (4.6) is equal to the number of relationships in the system (4.2) the original problem and the number of constraints in the system (4.5)-the number of variables in the original problem.

4. Coefficients of the unknowns in the objective function (4.4) the dual tasks are free members into the system (4.2), and the right parts in the system (4.5) the dual objectives-unidentified factors in the objective function (4.1).

Many of the original linear programming shall be drawn up in the form of a source or of dubious tasks, so it makes sense to talk about a pair of dual linear programming tasks.

Between optimal plans of couples of ambiguous objectives there is a link that installs a duality theorem [5]:

If a pair of dual tasks, one has the best plan, and the other has a solution, with extreme values of linear functions is the ratio

$$\min Z = \max Z' .$$

If a linear function is one of the objectives is not limited to, the other has no solution.

4.2 Model dual tasks

There are asymmetric and symmetric dual task. The asymmetrical dual tasks of system constraints of the original problem is given in the form of equations, and the dual-in the form of inequalities, and the variables may be negative. In the symmetric constraint system tasks as source and dual tasks set inequalities, with the dual variables are neotricatel′nosti condition.

Mathematical model of a pair of twin tasks can have one of the following types.

Asymmetrical task

The initial problem	Dual problem
1. $Z_{\min} = CX$;	$Z'_{\max} = YB$;
$AX = B$;	$YA \le C$.
$X \ge 0$.	
2. $Z_{\max} = CX$;	$Z'_{\min} = YB$;
$AX = B$;	$YA \ge C$.
$X \ge 0$.	

Symmetric problem

The initial problem	Dual problem
3. $Z_{\min} = CX$;	$Z'_{\max} = YB$;
$AX \ge B$;	$YA \le C$;
$X \ge 0$.	$Y \ge 0$.
4. $Z_{\max} = CX$;	$Z'_{\min} = YB$;
$AX \le B$;	$YA \ge C$;
$X \ge 0$.	$Y \ge 0$.

Here

$X = (x_1, x_2, ..., x_n)^T$ – the matrix-column;

$C = (c_1, c_2, ..., c_n)$ – the matrix-a string;

$B = (b_1, b_2, ..., b_m)^T$ – matrix column;

$Y = (y_1, y_2, ..., y_m)$ – the string matrix;

$A = (a_{ij})_{mxn}$ – the matrix of coefficients of the system constraints.

Finally, here is the following theorem, which is used in the evaluation of the duality of linear programming problems.

The theorem. *If the substitution component of the optimal plan in restrictions of the original problem i-th constraint applies to inequality, the i-th component of the optimal program of the dual problem is zero.*
If the i-th component of the optimal program of the dual problem is positive, the i-th constraint is satisfied its initial problem the best solution as a strict equality.

5. Description of the laboratory complex

The complex is a software module written in Borland C++ Builder 3. The complex is designed to solve the problems of linear programming and carrying out laboratory work on studying of the simplex method for solving the problem.

Programmatic form consists of four pages: "model", "table", "Results" and "Task".

5.1 Page "model"

"Model" page is for task models in source form. Canonicalization is required. Conversion routine tasks implemented programmatically. The page includes four areas for input and two buttons (fig. 1).

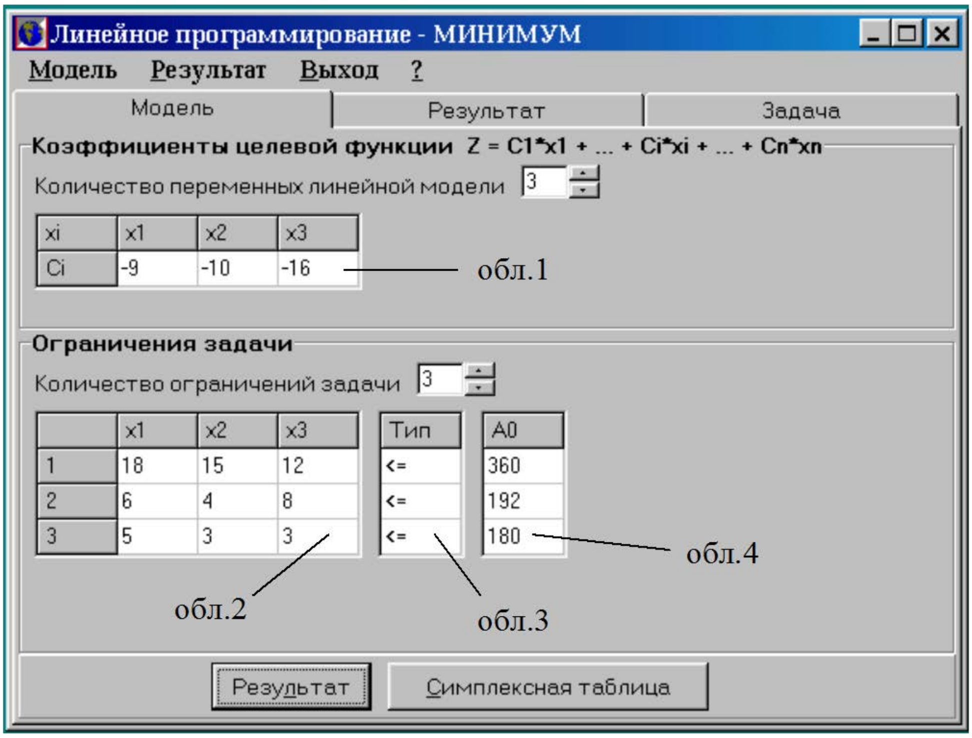

Figure. 1. The page “Model”

In region 1 are the values of the coefficients C_i the target of linear function (string matrix C).

Region 2 is a matrix ***A*** task constraints.

In region 3, you specify the type of limitations for each row of the matrix $\mathbf{A}$. The value of each field can be "==" for type constraints *as well*, "> =" – type restrictions for *greater-or-equal to* , and < = "-for *less-or-equal* in any order. You can set the keyboard type or cycle through the values by double-clicking the left mouse button.

Region 4 – vector $\mathbf{A}_0$ of free members enforcement tasks.

All numeric data is entered in floating point format. Accuracy of calculations 10^{-6}. Writing fractions are not allowed. The correctness of the input is checked automatically.

The number of model variables (n) can be changed within the range of 2 to 100, the number of restrictions (m). – from 1 to 100. Introduced the model can be saved to disk in a specific format, and then use the model by entering them from the file. These operations are performed using the menu.

By clicking the "result" shows the results of the task, when correctly its performances (go to the page "Results").

When you click the "task" table is converted to a simplex in canonical form is determined by the initial defensive plan, and presented a simplex table (go to the page "table").

5.2 Page “Table”

On this page (fig. 2) is a simplex table, a brief comment and three buttons.

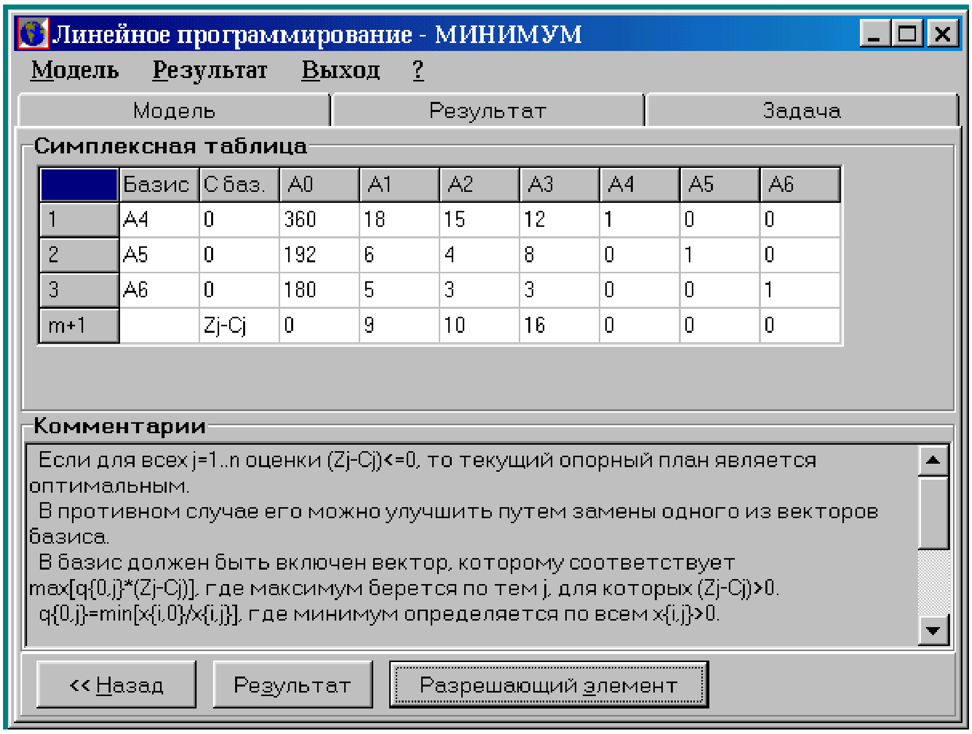

Figure. 2. Page “Table”

This page shows a step-by-step solution of the problem. Each step consists of defining an element and conversion tables. To do this on the page have corresponding buttons.

When you click the "Allow" element is defined in the current table and allow entry is highlighted.

By clicking "recalculate the table recalculation Simplex table corresponding to any element. The table contains all values of TRIMs, that does not affect the accuracy of the final result. In the final step, the message is displayed as a result of solving the problem.

The "back" button returns to the previous page. To calculate the final result at any step, you can navigate to the result page.

Result button duplicates the corresponding button on the previous page.

5.3 Page “Result”

Results page (Figure 3) contains the text wording generalized linear programming tasks in accordance with the model specified on the "model", and the results of the decision:

- the minimum value of the linear function, if it exists;
- the optimal values of variables.

As a model, you can save the result as text to disk and read from the disk stored results using the menu.

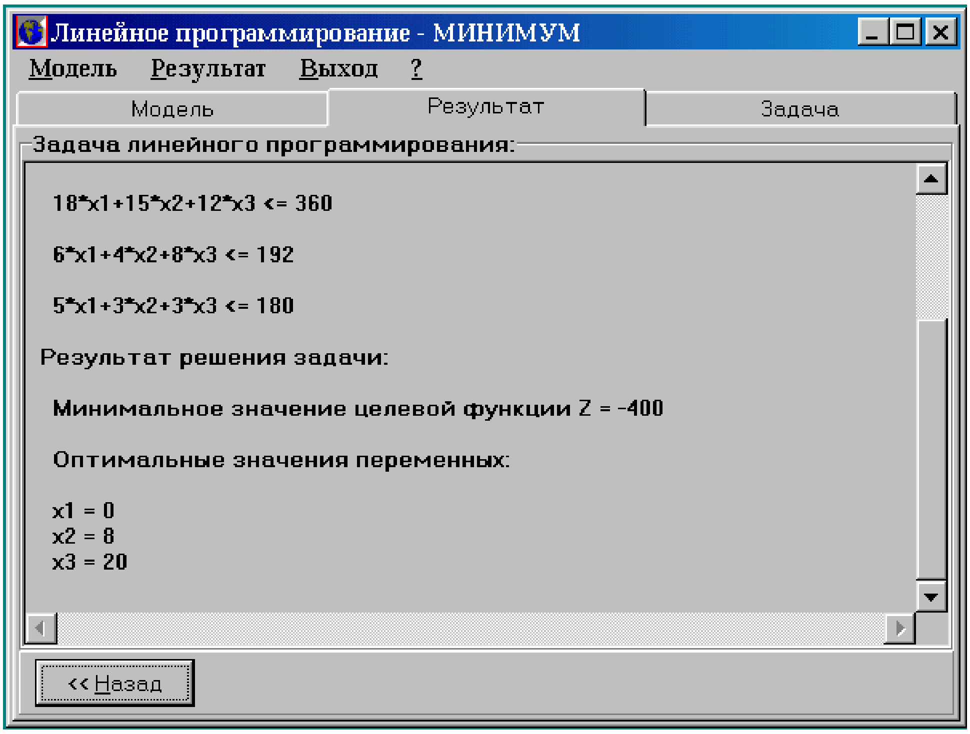

Figure. 3. Page “Result”

5.4 The page “Task”

This page (Figure 4) is only used when you run the laboratory work. On the Task, you can view the text of the task, in accordance with the specified option. The library includes three levels of 10 options for each.

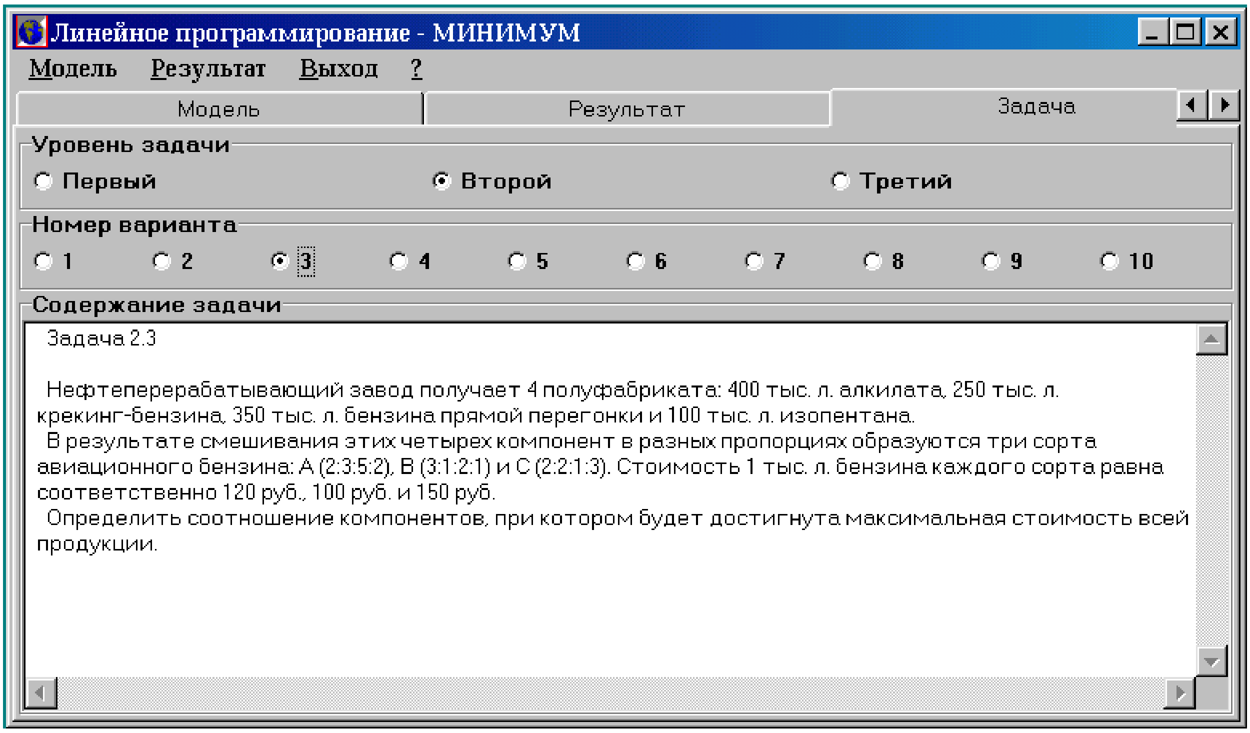

Figure. 4. The page “Task”

5.5 Main menu

The menu contains four items:

- the "Model" is designed for operations with task models;

-the "Result" is designed to perform operations on the results of tasks;

- by selecting "exit" to exit the program;

-Click "?" displays help.

The sub-item "Model-> New" clears the input fields and navigates to a page "model".

Subparagraph "Model-> Open" can be considered a model from disk that was saved using the sub-item "Model-> save. Model files have the format "*.mdl". Change the content model file will cause an error in the program.

The sub-item "Result-> open" displays the result on the page the text stored in text form. You may open any text file and modify it.

Subparagraph "Result-> Calculate" is similar to the corresponding button on the page.

The sub-item "Result-> Check" is used only when you run the lab work and is intended for checking of models of the optimal value of the objective function.

5.6 How to solve a problem using the software module

To solve the problem, you must perform the following steps.

- Run the file Lin_Prog.exe
- Go to the "model"
- Set the number of variables in the objective function
- Enter the objective function coefficients vector
- Set the number of task constraints
- Enter the constraint matrix
- Specify the type of the constraint for each row of the matrix constraints
- Introduce column constraint system free members
- Check the correctness of
- Get result in automatic or step mode
- Store the result in a file

When you enter a model from a file, the number of variables and constraints are placed automatically.

The order of execution of work

1. The software system on the example of verification (test) task.

2. Confirm using the complex decision of linear programming tasks selected previously.

3. Get a job teaching at the three levels.

4. Build a model of the dual tasks and get solutions.

5. Make a report stating the results of the objectives justifying the construction of models.

Bibliography

1. Reklejtis G., Rejvindran A., Règsdel K. Optimization technique: in 2 's book, book 1. Translation from English. -M.: Mir, 1986.-300 p.
2. Bundy B. Optimization methods.-M.: Radio and Communications, 1988.-128p.
3. Himmelblau D. Applied Nonlinear programming.-M.: Mir, 1975.-534p.
4. Y. Kuznetsov and Others. Mathematical programming.-M.: Graduate School. 1980.-300 p
5. N.N. Moiseev etc. Optimization techniques.-M.: Nauka, 1978.-352 p.
6. Degtyarev Yu.I. Operations Research.-M.: Graduate School., 1986.-320p.
7. Gill, F. et al. Practical optimization.-Moscow: Mir, 1985.-510p.
8. Taha H. Introduction to operations research.-Moscow: Mir, 1985.-232 p.
9. Akulich I.I. Matamatičeskoe programming.-M.: Graduate School., 1986. -319p.