

ALGORITHM OPTIMAL PENTRU PROBLEME DE OPTIMIZARE CU RESTRICȚII SPECIALE

Dr. Eugeniu GÂRLĂ, ASEM
e-mail: eugeniugarla@yahoo.com

În acest articol, este analizată o clasă de probleme de optimizare neliniară cu restricții speciale, se propune o metodă directă de rezolvare pentru problema auxiliară, pentru care este calculată complexitatea, se evaluează numărul maxim de operații elementare, este descris algoritmul optimal pentru efectuarea calculelor numerice. În studiu, se construiește un algoritmul optimal de rezolvare a problemei auxiliare a modelului PG, complexitatea acestui algoritmul este $O(nm^2, N)$, numărul de operații elementare este minimal. Matricea supusă inversării nu depinde de dimensiunea problemei n și are permanent dimensiunea constantă – $m \times m$, $m \ll n$. Astfel, modelul PG este absolut funcțional și, practic, „imun” la dimensiunea problemei de optimizare.

Cuvinte-cheie: complexitatea algoritmului, metode de optimizare

JEL: C0, C6

1. Considerații generale

În lucrul cu structuri de date, metode de optimizare, se folosesc modele complexe care presupun un consum foarte mare de resurse pentru rezolvare. Există mai mulți algoritmi de rezolvare a uneia și aceleași probleme și atunci ar trebui să se stabilească algoritmul care este mai performant. Se impune astfel a găsi o măsură a gradului de performanță sau de eficiență al algoritmilor. Două criterii stabilesc măsura performanței unui algoritmul: timpul în care se obține soluția problemei și spațiul de memorie utilizat pentru obținerea

OPTIMAL ALGORITHM FOR OPTIMIZATION PROBLEMS WITH SPECIAL RESTRICTIONS

PhD Eugeniu GÂRLĂ, ASEM
e-mail: eugeniugarla@yahoo.com

The present paper analyses a class of nonlinear optimization problems with special restrictions, we propose a direct method for solving the auxiliary problem, for which we calculate complexity, we also assesses the maximum number of elementary operations and describe the optimal algorithm for performing numerical calculations. The study builds an optimal algorithm for solving the auxiliary problem of PG model; the complexity of this algorithm is $O(nm^2, N)$, the number of elementary operations is minimal. Matrix inversion does not depend on the size of problem n and always has constant size – $m \times m$, $m \ll n$. Thus, the model PG is fully functional and practically “immune” to the size of the optimization problem.

Key words: algorithm complexity, optimization methods

JEL: C0, C6

1. General considerations

Complex models are used when working with data structures and optimization methods; they involve the consumption of a lot of resources. There are several algorithms for solving one and the same problem and then there should be settled an algorithm that would be more efficient. It is therefore necessary to find a measure of performance or efficiency of algorithms. Two criteria establish the measure of performance of an algorithm: the time during which the solution of the problem is found and the memory

ei. Dintre cele două resurse de calcul, spațiu și timp, cel mai des, cea critică este timpul de execuție. Analiza acestor parametri de eficiență a algoritmilor este cunoscută sub numele de analiza complexității algoritmilor. În majoritatea algoritmilor, volumul resurselor necesare depinde de dimensiunea problemei de rezolvat – n . Funcția f_0 , care dă ordinul de mărime al numărului de operații elementare și implicit al timpului de execuție, se va nota cu $f_0(n)$, iar algoritmul respectiv este notat $O(f_0(n))$. Această notație comportă un caracter asimptotic și determină o clasificare a algoritmilor după ordinul de complexitate. Un algoritm cu $(f_0(n), O(n))$ se numește liniar, cu $O(n^2)$ – pătratic, cu $O(n^3)$ – cubic, cu $O(n^k)$ – polinomial, cu $O(2^n)$ – exponențial. Nici algoritmii exponențiali, nici cei polinomiali cu grad mare nu pot fi utilizați în practică. De aici, algoritmii aplicabili pentru probleme de dimensiuni mari sunt doar cei din clasa $O(n^k)$ ($k \ll n$), a căror durată de execuție se consideră rezonabilă. Se cere menționat că, pentru dimensiunea mare a problemei, performanța hardware este insuficientă, mai esențială fiind îmbunătățirea ordinului algoritmului. La general, nu există nicio formulă universală pentru determinarea complexității. Aceasta se face de la caz la caz, ținând cont de particularitățile problemei. Pentru cele ce urmează: diferiți algoritmi care soluționează aceeași problemă formează clasa algoritmilor; operația cea mai des repetată se numește operația de bază, iar determinarea teoretică a numărului de repetări permite alegerea celui mai bun algoritm; algoritmul optimal este acel algoritm care efectuează cel mai mic număr de operații de bază dintre toți algoritmii clasei sale; o tehnică cunoscută de

space used to obtain it. Of the two calculation resources, space and time, most often, the runtime is critical. The analysis of these efficiency parameters of the algorithm is known as the complexity of the analysis algorithms. In most algorithms, the amount of resources needed depends on the size of the problem to be solved – n . Function f_0 , which gives the size of the number of elementary operations and the execution time, shall be noted as $f_0(n)$, while the respective algorithm is noted as $O(f_0(n))$. This notation has an asymptotic character and determines a classification of algorithms by order of complexity. An algorithm with $(f_0(n), O(n))$ is called linear with $O(n^2)$ – quadratic, with $O(n^3)$ – cubic, with $O(n^k)$ – polynomial with $O(2^n)$ – exponential. Neither the exponential algorithms, nor the polynomial with high degree can be used in practice. From here, algorithms applicable to large problems are just of the class $O(n^k)$ ($k \ll n$), whose execution time is considered reasonable. It is worth mentioning that for the large size of the problem, the hardware performance is insufficient, more essential being the improvement of the algorithm order. Generally, there is no universal formula for determining the complexity. This is done from case to case, taking into account the peculiarities of the problem. For the following: different algorithms that solve the same problem form the class of algorithms; the most often repeated operation is called basic operation; the theoretical determination of the number of repetitions allows selecting the best algorithm; the optimal algorithm is the one that performs the smallest number of basic operations of all algorithms for its class; a well-known technique in solving recurrence

rezolvare a recurenței complexității este metoda iterației, care transformă recurența în sumă de termeni dependenți doar de n și condițiile inițiale, iar suma, la rândul ei, se obține, deseori, prin operația de logaritmare; sunt utile relațiile:

$$\lim_{n \rightarrow \infty} \frac{n^k}{2^n} = 0; O(n^k) \subset O(2^n),$$

la calcularea complexității modelului contează doar ordinul superior al operațiilor efectuate în procesul algoritimizării, deci, se reține doar termenul care crește cel mai repede odată cu creșterea lui n , deoarece acest termen are impactul cel mai mare asupra timpului de execuție sau al spațiului ocupat al implementărilor algoritmului, ceilalți termeni devenind neglijabili pentru valori mari ale lui n ; pentru operația de multiplicare a două matrice pătrate, complexitatea este $O(n^3)$, respectiv, pentru înmulțirea a două matrice dreptunghiulare (una $m \times l$, alta $l \times n$), este $O(mln)$; inversarea matricială nu este mai complexă din punct de vedere al numărului de operații decât înmulțirea de matrice; frecvent se cere parantezarea optimală a produsului de matrice $A_1 \times A_2 \times \dots \times A_n$ de dimensiuni $d_0 \times d_1, d_1 \times d_2, \dots, d_{n-1} \times d_n$ pentru care numărul total de operații elementare să fie minim, în particular, parantezarea optimală se rezolvă pornind de la dreapta spre stânga: fie $A_1(n \times m), A_2(m \times m), A_3(m \times n), A_4(n \times 1)$, atunci, pentru produsul $A_3(m \times n) \Pi, A_4(n \times 1)$, se vor efectua înmulțiri și adunări respectiv

$$\rightarrow 2 \times n \times m - m,$$

pentru produsul $A_2(m \times m) \Pi A_3(m \times n) \Pi A_4(n \times 1)$, se vor efectua înmulțiri și adunări respectiv $\rightarrow 2 \times n \times m - m + 2 \times m \times m - m,$

iteration complexity is the method of iteration, that converts recurrence in the sum of terms dependent only on n and initial conditions, while the sum is obtained through the logarithm operation; in this respect the following relations are useful:

while calculating the model complexity only higher order operations counts, that are carried out in the process of algorithmisation, therefore having retained only the term which has the fastest growth together with the growth of n , because this term has the largest impact on execution time or space occupied in the implementation of the algorithm, the other terms becoming negligible for larger values of n ; for the operation of multiplication of two square matrices, the complexity is $O(n^3)$, respectively, for the multiplication of two rectangular matrix (one $m \times l$, another $l \times n$), is $O(mln)$; matrix inversion is not more complex in terms of the number of operations than multiplication of matrices; frequently requires bracketing of optimal product matrix $A_1 \times A_2 \times \dots \times A_n$ of size $d_0 \times d_1, d_1 \times d_2, \dots, d_{n-1} \times d_n$ for the total number of elementary operations to be minimal, in particular, bracketing optimal solved from right to left: let, $A_1(n \times m), A_2(m \times m), A_3(m \times n), A_4(n \times 1)$ then the product $A_3(m \times n) \Pi, A_4(n \times 1)$, will perform multiplication and congregation like

for product $A_2(m \times m) \Pi A_3(m \times n) \Pi A_4(n \times 1)$, will perform multiplication and gatherings that $\rightarrow 2 \times n \times m - m + 2 \times m \times m - m,$

în rezultat per total, pentru produsul | resulting in overall product for product

$$A_1(n \times m) \Pi A_2(m \times m) \Pi A_3(m \times n) \Pi$$

se vor efectua înmulțiri și adunări respectiv | will perform multiplication and sums respectively

$$\rightarrow 2 \times n \times m - m + 2 \times m \times m - m + 2 \times n \times m - n.$$

2. Modelul de optimizare

Problemele de optimizare neliniară, în control optimal și economie, sunt probleme de valori extreme cu condiții suplimentare, caracterizate prin aceea că numărul variabilelor este foarte mare. Este binecunoscut faptul că majoritatea modelelor de optimizare operează cu inversa pentru unele matrice. Dacă inversa s-ar afla, atunci soluția s-ar găsi în urma unor operații matematice elementare, dar efortul de calcul necesar inversării este foarte mare, există enorme probleme de stabilitate, totodată, nu există soluția în formă analitică, de aceea, majoritatea metodelor sunt metode iterative, adică de la iterație la iterație se construiește consecutivitatea

2. The optimization model

Nonlinear optimization problems, as well as those in optimal control and economy are matters of extreme values with additional conditions, characterized in that the number of variables is very high. It is well known that most optimization models operate with the reverse for some matrix. If the reverse is found out, then the solution would find the following basic mathematical operations, but the calculation effort required for reverse is very high; there is enormous stability issues, while there is no solution in analytical form, therefore, most methods are iterative methods, i.e. iteration to iteration builds the following sequence

$$f(x^0) \geq \dots \geq f(x^{k-1}) \geq f(x^k) \geq f(x^{k+1}), \geq \dots, \\ x^{k+1} = x^k + \alpha_k p_k,$$

unde α_k – număr, p_k – vector,

iar procesul de calcul se încheie, dacă diferența dintre două valori consecutive ale lui x satisfac inegalitatea $|x^{k+1} - x^k| < \varepsilon$, ε – număr mic dat, iar iterația aici incluzând rezolvarea problemei auxiliare – problemă-cheie, de care depinde esențial și eficiența problemei inițiale. Pornind de la aceste considerente, cercetările făcute de autor [1] s-au axat pe ideea selectării claselor de probleme de dimensiuni mari, care determină niște cazuri particulare, dar care acoperă o mare parte din problemele practice și pentru care, totodată, pot fi propuse scheme eficiente de calcul numeric. Fie problema inițială de optimizare neliniară în forma (modelul *PG*)

where α_k – number, p_k – vector,

and the calculation process ends, if the difference between two consecutive values of x satisfy the inequality $|x^{k+1} - x^k| < \varepsilon$, ε – given small number and iteration here including ancillary problem solving – key issue, and effectiveness which depends heavily on the initial problem. Based on these considerations, the research made by the author [1] focused on the idea of selecting classes of large problems that causes some cases, but covering much of the practical problems and which also may be the proposed effective numerical computation schemes. Whether the initial nonlinear optimization problem in the form (model *PG*)

$$\begin{aligned} & \min f(x) \quad (1) \\ & \left\{ \begin{aligned} & g_j(x) = 0, j = \overline{1, m} \dots (2) \\ & 0 \leq x_i \leq a_i, i = \overline{1, n} \dots (3) \end{aligned} \right. \end{aligned}$$

problema auxiliară (PA) în model este o problemă de programare pătratică cu matrice unitară

the auxiliary problem (PA) in the model is a quadratic programming problem with unitary matrix

$$\min \left(f_1(p) = 1/2 \|p\|^2 + (d, p) \right) \quad (4)$$

$$Hp = h, H = \begin{bmatrix} h_{11} & \dots & h_{1n} \\ \dots & \dots & \dots \\ h_{m1} & \dots & h_{mn} \end{bmatrix} \quad (5)$$

$$0 \leq p_i \leq a_i, i = \overline{1, \dots, n}. \quad (6)$$

unde $f(x), g_j(x)$, – funcții de x , continue, împreună cu derivatele lor, d, h – vectori, H – forma matricială a restricțiilor Jacobianului (2), în care $h_j(x) \equiv g_j'(x)$, însemnând derivata, iar componenta i a gradientului $h_j(x)$, notată respectiv cu h_{ji} . Nu se distinge aici separat cazul unui sistem de ecuații neliniare, la fel nu se abordează în detalii condițiile necesare de extremum, convergența modelului PG, ca atare, stabilitatea soluției, problema acumulării erorilor, tehnicile computeraie etc. (pentru rezultate teoretice și practice a se vedea, de exemplu [3;1]). În continuare, se examinează doar complexitatea modelului, în speță a celei mai dificile operații – complexitatea problemei auxiliare, celelalte operații ale modelului nefiind esențiale ca timp și spațiu. Trebuie să se remarcată că, în (2), se presupune $m \ll n$, altfel spus, restricții de tip “=” sunt foarte puține, adevărata dimensiune mare a problemei depinzând esențial de n . Pentru (2)-(3) se definește operatorul de proiecție:

where $f(x), g_j(x)$ – functions of x , continue along with their derivatives, d, h – vectors, H – matrix form of Jacobean restrictions (2) where $h_j(x) \equiv g_j'(x)$ meaning the derivative and the gradient component i of gradient $h_j(x)$, denoted respectively with h_{ji} . We cannot separately distinguish here a system of linear equations, the same does not address in detail the conditions necessary of extremum, PG convergence model, as such, the stability of the solution, the problem of accumulation of errors, computer techniques etc. (for theoretical and practical results see, for example [3, 1]). Next, we shall examine the complexity of the model and in case of the most difficult operations – Auxiliary complexity of the problem, other operations of the model is not essential as time and space. It should be noted that in (2), it is assumed that $m \ll n$, in other words, restrictions type “=” are very few true size of the problem depends essentially on n . For (2) – (3) is defined the projection operator:

$$P = G^T (GG^T)^{-1} G,$$

care alcătuiește matricea de proiectare din restricții după un algoritm stabilit. Operatorul

which forms the matrix design of restrictions set by an algorithm. The operator P has a

P are o serie de proprietăți remarcabile, fapt pentru care, deseori, constituie metoda de bază a rezolvării problemei auxiliare. Dar în cazul problemelor de dimensiuni mari examinate aici, dacă s-ar proceda frontal, atunci în G s-ar include și restricțiile paralelipipedice de tip „ \leq ”, fapt ce-ar îngreuna la maximum inversarea lui $(GG^T)^{-1}$, matrice de dimensiuni foarte mari $(m+2 \times n, m+2 \times n)$. Pentru operatorul de proiecție al modelului PG , însă, s-a reușit reducerea inversării la calculul matricei

number of remarkable properties, for which often constitutes the basic method of auxiliary problem solving. But if large issues examined here, if you would proceed front, then it would include restrictions rectangular type “ \leq ” which would make it more difficult to reverse of $(GG^T)^{-1}$ than his very large matrix $(m+2 \times n, m+2 \times n)$. Operator PG projection model, however, has managed to reduce to calculation matrix inversion

$$R^{-1} = (HH^T)^{-1},$$

unde H este compusă doar din restricții de tip “=” – matrice bandă, iar restricțiile de tip „ \leq ”, se iau H în considerare adăugând în H o coloană după o formulă, sau scoțând din H o coloană conform unei formule similare, altfel spus, s-a demonstrat că luarea în considerare a unei restricții în procedura de optimizare echivalează cu adăugarea unei coloane sau cu scoaterea coloanei respective din matricea restricțiilor de tip “=” . Pentru problema auxiliară, s-a demonstrat că la pasul k există: întotdeauna o direcție p nenulă:

where H is composed only of restrictions type “=” – matrix tape and restrictions of “ \leq ” are H considered adding to H a column after the formula, or taking out from H a column according to a formula similar, in other words, it was shown that consideration of a restriction optimization procedure is equivalent to a column or removing restrictions that column matrix type “=” . For the auxiliary problem, it has been demonstrated that in step k there is:

always a non-null p direction:

$$p^{k+1} = p^k - \alpha^k (I - P)(p^k + d)$$

- formulă de calcul al multiplicatorilor Lagrange:

- calculation formula of Lagrange multipliers:

$$\lambda^{m+j}_k = \{ p^k + d \}_i + (H^T_j, \lambda_k)$$

- soluție după un număr finit de pași $\leq N$, N – constantă
- număr α^k , găsit după o formulă analitică
- metodă de inversare a (HH^T) .

- solution after a finite number of steps $\leq N$, N – constant
- number α^k , found after an analytical formula
- method of reverse of (HH^T) .

Pentru evitarea calculării excesive a $(HH^T)^{-1}$, în procesele iterative, se aplică frecvent formula generală de recurență a lui Woodbury, sau cazul particular al acesteia –

In order to avoid calculating excessive $(HH^T)^{-1}$, in iterative processes frequently applied a common recurrence formula of Woodbury, or the particular case of it -

formula Sherman-Morrison cu modificății, care permit calcularea inversei noii matrice, la care s-a adăugat o coloană (vectorul H_j) și un rând (vectorul H^T_j), fără a precede la inversarea acesteia, ci folosind matricea inițială R^{-1} și operația produsului vectorial.

3. Rigurozități matematice

Rigurozitățile modelului de optimizare au la bază următoarele considerente.

3.1. Fie p^0_k punctul inițial, care satisface condițiile (5) – (6) și h_j liniar independenți. Având în vedere specificul restricțiilor problemei inițiale se presupune că un astfel de punct p^0_k de fiecare dată există, mai mult decât atât, deoarece $n \gg m$, la fiecare pas se pot lua în considerare toate m restricții de tip “=”, iar cele de tipul „≤”, se iau în evidență conform algoritmului descris mai sus. După cum s-a menționat, matricea G va conține, în rândurile corespundente restricțiilor de tipul “=”, doar ± 1 sau 0, ceea ce sugerează idei de simplificare la calcularea operatorului de proiecție P . Cu adevărat, dacă, de exemplu, o componentă $p^0_{j_k}$ a vectorului p^0_k iese la suprafață, adică $p^0_{j_k} = 0$, bunăoară (aici și în continuare, cazul $p^0_{j_k} = a_i$, se examinează analogic), atunci aceasta echivalează cu fixarea coloanei respective j în matricea H , corespunzătoare restricțiilor de tipul “=“.

formula Sherman-Morrison with modifications that allow the calculation of the inverse of the new matrix to which was added a column (vector H_j) and a row (vector H^T_j) without proceeding to its reverse, but using the original matrix R^{-1} and vector product operation.

3. Mathematical rigors

Stringencies of the optimization model are based on the following considerations.

3.1. Let p^0_k be the starting point, satisfying the conditions (5) – (6) and h_j linearly independent. Considering the specific restrictions of original problem, supposedly such a point p^0_k every time there is more than that, since $n \gg m$, every step can consider all m restrictions of “=” and those of type “≤” are taken out according to the algorithm described above. As mentioned above, the matrix G will contain in the corresponding rows restrictions of the type “=” only ± 1 or 0, which suggests ideas to simplify the calculation of projection operator P . Indeed, if, for example, a component $p^0_{j_k}$ of vector p^0_k comes out, that is $p^0_{j_k} = 0$, for example (here and hereinafter, the case $p^0_{j_k} = a_i$ is examined analogue), then this is **equivalent** to determining that column j of the matrix H , corresponding to restrictions such as “=”.

$$\begin{bmatrix} h_{11} \dots 0 \dots h_{1n} \\ \dots 0 \dots \\ h_{m1} \dots 0 \dots h_{mn} \end{bmatrix} = H \quad (7)$$

Pentru aceasta, este suficient să comparăm operatorii de proiecție respectivi P_L , P_{L-1} . P_{L-1} corespunde matricei $H_{L-1} = (H_{L,1}, \dots, H_{L,j-1}, H_{L,j+1}, \dots, H_{L,n})$, în care lipsește exact coloana j , iar P_L matricei (5) și care va avea coloana respectivă

For this reason it is sufficient to compare the respective projection operators P_L , P_{L-1} . P_{L-1} corresponding to matrix $H_{L-1} = (H_{L,1}, \dots, H_{L,j-1}, H_{L,j+1}, \dots, H_{L,n})$, where it misses column j , while P_L of matrix (5) and that will have the respective column

alcătuită din 0. La fel și întreaga (PA) arată respectiv

composed of 0. Similarly, the entire (PA) looks as follows

$$\begin{aligned} & \min \left(1/2 \left\| (z - p_k) \right\|^2 : H_L p_k = h \right) \\ & \min \left(1/2 \left\| (z - p_k) \right\|^2 : H_{L-1} p_k = h \right). \end{aligned}$$

Altfel spus, se efectuează succesiv trecerea de la o problemă de minimizare la alta cu o unitate mai mică a dimensiunii (subspațiu), adică trecerea restricției în rândul celor pasive a echivalat cu apariția coloanei din 0 în (7). Având P_L calculat în acest mod și micșorând cu mult dimensiunea matricei inițiale se poate găsi un nou punct

In other words, sequentially is performed the switch from one problem to another with a smaller unit size (subspace), that is passage of restriction among those liabilities amounted to 0 in appearance of the column (7). With P_L calculated in this way and reducing a lot the initial matrix size can find a new point

$$p_{k+1}^0 = p_k^0 - \alpha_k^o (I - P_L) f_1'(p_k^0) \tag{8}$$

care este desemnat drept inițial și cu care se procedează analogic. Situația se schimbă, dacă, la un moment dat, o deplasare din punctul inițial este imposibilă, cu alte cuvinte, s-a atins *min*. Atunci se pot formula următoarele condiții necesare pentru extremum, referitor la problema (4) – (6), pe componente

that is designated as the original and proceed with the analogue. The situation changes if, at a given moment, a shift of the starting point is not possible, in other words, it has reached *min*. Then the following conditions can be formulated for the extremum, on the problem (4) – (6), on components

$$\left[f_1' \left\{ (p_k^o) \right\}^j \right] = - \left(H_{L,j}^T, \lambda_k \right) - \dots - \lambda_k^{m+j} \{-1\}, \tag{9}$$

λ_k^i – multiplicatorii Lagrange sunt împărțiți în două grupuri: $\lambda_k = (\lambda_k^1 \dots \lambda_k^m)$, cei ce corespund restricțiilor de tip “=”, pentru care semnul nu contează și restricțiilor „≤”, și cei care se deduc din (9) după formula:

λ_k^i – Lagrange multipliers are divided into two groups: $\lambda_k = (\lambda_k^1 \dots \lambda_k^m)$, those corresponding to restrictions of type “=”, to which the sign does not matter and restrictions “≤” and those which are deducted from (9) by the formula:

$$\lambda_k^{m+j} = \left\{ f_1' \left(p_k^o \right) \right\}^j + \left(H_{L,j}^T, \lambda_k \right). \tag{10}$$

Dacă toți λ_k^{m+j} sunt nenegativi, atunci s-a găsit soluția pentru (4)- (6), în caz contrar, dacă cel puțin unul din aceștia, de exemplu λ_k^{m+j} , care corespunde $p_{j k}^0 = 0$, este negativ, atunci se efectuează operația inversă, în matricea H se readuce coloana j și se construiește deja operatorul de proiecție P_{L+1} ,

If all λ_k^{m+j} are non-negative, then we found the solution for (4) – (6), otherwise, if at least one of them, for example λ_k^{m+j} , which corresponds to $p_{j k}^0 = 0$ is negative, then is performed the inverse operation, in matrix H is brought back the column j and projection operator P_{L+1} is build, following the same

urmând aceeași procedură de calcul a *min* în noul subspațiu.

3.2. Este ușor de dedus că α^o_k din (4)-(6) are următoarea **formulă analitică** de calcul

$$\alpha^o_k = \min \left(p_{j\ k}^0 / \left((I - P_L) f_1'(p^0_k) \right)^j > 0, (p_{j\ k}^0 - a_j) / \left((I - P_L) f_1'(p^0_k) \right)^j < 0, 1 \right), \quad (11)$$

parcurgând toți $j = \overline{1, n}$. Evident că cel puțin pentru componenta j , pentru care $\lambda^{m+j}_k < 0$ a vectorului $(I - P_L) f_1'(p^0_k)$, este posibilă o direcție nenulă, deoarece produsul componente j a vectorului unitar $e = (0, \dots, 1, \dots, 0)$ cu această componentă va fi:

$$- \left((I - P_L) f_1'(p^0_k) \right)^j = - \lambda^{m+j}_k > 0.$$

Suplimentar, se observă ușor că $\alpha^o_k > 0$, deoarece valoarea $\alpha^o_k = 0$ a fost exclusă pentru componenta pentru care $p_{j\ k}^0 = 0$.

3.3. Fie $J(p^0_k)$ mulțimea de indici pasivi. În noul punct p^0_{k+1} , valoarea funcționalei f_l strict va descreește, iar mulțimea indicilor $J(p^0_{k+1})$ va include strict mulțimea $J(p^0_k)$. Aceasta înseamnă că *min*, în noul subspațiu, va fi atins în mod obligatoriu, deoarece mulțimea J nu se poate lărgi la infinit, ceea ce demonstrează că, **cel mult în N pași**, procesul de lărgire a mulțimii de indici se va întrerupe. Astfel, se observă lejer că schema generală de rezolvare conține o singură operație matematică complicată și voluminoasă în calcule numerice – alcătuirea operatorului de proiecție P_L , în care intră operația de inversare a matricei $(H_L H^T_L)^{-1}$.

Pornind de la specificul restricțiilor problemei inițiale, schema generală de rezolvare poate fi eficientizată în continuare, în primul rând, prin evitarea operației de inversare a matricei $(H_L H^T_L)^{-1}$ la fiecare iterație, utilizând formule de recurență din algebra liniară. Pentru aceasta, este suficient de reprezentat R_L sub forma

procedure for calculating the new subspace *min*.

3.2. It is easy to deduce that α^o_k from (4)-(6) has the following **analytical formula**

$$\alpha^o_k = \min \left(p_{j\ k}^0 / \left((I - P_L) f_1'(p^0_k) \right)^j > 0, (p_{j\ k}^0 - a_j) / \left((I - P_L) f_1'(p^0_k) \right)^j < 0, 1 \right), \quad (11)$$

going through all $j = \overline{1, n}$. Obviously, at least for the component j , where $\lambda^{m+j}_k < 0$ of the vector $(I - P_L) f_1'(p^0_k)$ is possibly a non-zero direction because the product of component j of the unit vector $e = (0, \dots, 1, \dots, 0)$ with this component will be:

$$- \left((I - P_L) f_1'(p^0_k) \right)^j = - \lambda^{m+j}_k > 0.$$

Additionally, it is easy to notice that $\alpha^o_k > 0$, as the value $\alpha^o_k = 0$ was excluded for component for which $p_{j\ k}^0 = 0$.

3.3. Let $J(p^0_k)$ be the crowd of passive indices. The new point p^0_{k+1} , the functional value f_l will strictly decrease, and the crowd indices $J(p^0_{k+1})$ will include strict set $J(p^0_k)$. This means that *min* in the new subspace will necessarily be achieved, as set J cannot expand indefinitely, which demonstrates that in **at most N steps** the process of set enlarging of indices will pause. Thus, there is a general scheme for resolving a single mathematical operation which is complicated and full of numerical calculations – P_L projection operator composition into that contains the matrix inversion operation $(H_L H^T_L)^{-1}$. Starting from the specific restrictions of the initial problem, the general scheme of solving the problem can be streamlined further, primarily by avoiding matrix inversion $(H_L H^T_L)^{-1}$ at each iteration operation, using recursion formulas from linear algebra. For this, it is sufficient to represented R_L in the form

$$R_L = H_{L,1}H_{L,1}^T + \dots + H_{L,n}H_{L,n}^T,$$

de unde rezultă imediat că orice permutare a coloanelor în R_L nu schimbă această matrice. fie acum că se adăugă o coloană în H_L , anume $H_{L,j}$, notat $H_{L+1} = (H_L, H_{L,j})$. Atunci

from which results immediately that any permutation of columns in R_L does not change this matrix. Event that now is added a column in H_L , namely $H_{L,j}$, noted $H_{L+1} = (H_L, H_{L,j})$. Then

$$R_{L+1} = H_L H_L^T + H_{L,j} H_{L,j}^T.$$

Aplicând acum pentru cazul particular R_{L+1} formula generală se obține:

So, applying for the particular case R_{L+1} the general formula, is obtained:

$$R_{L+1}^{-1} = R_L^{-1} - (1 + H_{L,j}^T R_L^{-1} H_{L,j})^{-1} R_L^{-1} H_{L,j} H_{L,j}^T R_L^{-1}.$$

Analogic, la scăderea din H_{L+1} , a coloanei $H_{L,j}$:

Analogically, decreasing from of H_{L+1} the column $H_{L,j}$:

$$R_{L+1}^{-1} = R_L^{-1} + (1 - H_{L,j}^T R_L^{-1} H_{L,j})^{-1} R_L^{-1} H_{L,j} H_{L,j}^T R_L^{-1}.$$

Ca rezultat, excelenta formulă recurentă din algebra liniară își găsește locul cuvenit în schema de optimizare sporind eficiența algoritmului de minimizare și evitând operația de inversare. Atunci însă, când inversarea este inevitabilă, bunăoară, la prima iterație, sau la acumularea erorii de calcul după un anumit număr de pași, în ipoteza că funcțiile ce figurează sunt derivabile până la ordinul doi inclusiv, fapt ce implică practic și simetria matricei $R_L = R_L^T$, schema de optimizare poate fi dezvoltată în continuare. Pentru astfel de matrice, în algebra liniară există metode de inversare și mai eficiente, cu economii esențiale de memorie operativă și volum de operații elementare. De exemplu, metoda lui Cholesky permite descompunerea R_L în produs de două matrice triunghiulare, una superior triunghiulară și alta inferior triunghiulară $R_L = S_L S_L^T$, cu inversarea prin factorizare a uneia din ele.

As a result, the excellent formula recurring from linear algebra finds its rightful place in the scheme of increasing efficiency optimization algorithm for minimizing and avoiding reversal operation. Then, when the reversal is inevitable, for example, at the first iteration, or at the accumulation of error calculation after a certain number of steps, assuming that the functions contained are differentiable up to order two inclusive, which basically involves matrix symmetry $R_L = R_L^T$, the optimization scheme can be further developed. For such a matrix, in linear algebra, there are reversal methods even more efficient, with essential memory savings and volume of elementary operations. For example, the Cholesky method of decomposition R_L in product of two triangular matrices, one upper and one lower triangular $R_L = S_L S_L^T$ reversing through factorization of one of them.

3.4. Foarte des în model se formulează problema de extremizare pe subspațiu în forma (4)-(5). Minimumul pentru (4)-(5) se obține **dintr-un pas**. Într-adevăr, dacă punem

3.4. Very often in the model is formulated a problem of extremity in the form of (4)-(5). The minimum for (4)-(5) is obtained in **one step**. Indeed, if we have

$P = H^T (HH^T)^{-1} H, x = (I - P)p$, I - matrice unitară și $x_0 = x + (I - P)p$ în (4)-(5), atunci derivând după p și luând în considerare proprietățile lui P , se obține în coordonatele inițiale

$$f'_1(x) = f'_1((I - P)p) = (I - P)p' = (I - P)(x_0 + d),$$

adică

$$x_1 - x_0 = - (I - P)f'_1(x_0),$$

iar pe de altă parte, utilizând metoda gradientilor conjugați pentru problema formulată după primul pas, vom avea la fel

$$x_1 - x_0 = - (I - P)f'_1(x_0),$$

de unde rezultă afirmația enunțată.

4. Complexitatea algoritmului

Din cele demonstrate, rezultă că operația de bază este calcularea matricei P , dimensiunile acesteia fiind date de

$$\begin{bmatrix} P \\ n \times n \end{bmatrix} = \begin{bmatrix} H^T \\ n \times m \end{bmatrix} X \left(\begin{bmatrix} H \\ m \times n \end{bmatrix} X \begin{bmatrix} H^T \\ n \times m \end{bmatrix} \right)^{-1} X \begin{bmatrix} H \\ m \times n \end{bmatrix}. \quad (12)$$

În procesul de calcul, însă, se profită din plin de descompunerea lui P în produs de matrice, efectuând succesiv de la dreapta spre stânga operația de înmulțire a matricei la vector, numărul operațiilor elementare micșorându-se simțitor. Pe lângă acest mare avantaj, mai există cel legat de stabilitate, deoarece anume operația inversării matricei generează multă instabilitate. Or, matricea supusă inversării are dimensiuni constante și mult mai mici, comparativ cu cele care se înmulțesc. În prealabil, se cuvine subliniat că, pentru calcularea produsului matricei (HH^T) , se vor utiliza $m^2 \times (2 \times n - 1)$ operații elementare. Mai sus, s-a arătat că, pentru multiplicarea a 4 matrice, parantezarea optimală se face prin $2 \times n \times m - m + 2 \times m \times m - m + 2 \times n \times m - n$ înmulțiri și adunări, ori componentele lui P din (12) au exact

$P = H^T (HH^T)^{-1} H, x = (I - P)p$, I - unitary matrix and $x_0 = x + (I - P)p$ in (4)-(5), then when derived after p and taking into account the properties of P can be obtained in the initial coordinates

namely

iar pe de altă parte, utilizând metoda gradientilor conjugați pentru problema formulată după primul pas, vom avea la fel

hence the stated statement.

4. The complexity of the algorithm

From the above mentioned results that the basic operation in calculating the matrix P , its size is given by

In the process of calculation however, is taken full advantage of the decomposition of P into the matrix product performing sequentially from the right to left matrix multiplication operation of the vector, the number of elementary operations decreasing significantly. Besides this big advantage, there is one related to stability, because only matrix inversion operation generates more instability. Or, matrix subject to inversion is subject to constant and much smaller dimensions compared to those that are multiplying. Initially, it should be noted that for the calculation of the matrix product (HH^T) we shall use $m^2 \times (2 \times n - 1)$ elementary operations. Above we have shown that, for multiplying 4 matrixes, optimal bracketing is done by $2 \times n \times m - m + 2 \times m \times m - m + 2 \times n \times m - n$ multiplications and sums, or

aceleași dimensiuni, deci, la minimizarea pe subspațiu este nevoie de același număr de operații. Atunci, per total, la rezolvarea (PA) se vor utiliza cel mult

$$N \times (m^2 \times (2 \times n - 1) + 2 \times m \times m - m + 2 \times n \times m - n + n \times (2 \times m - 1)) \quad (13)$$

operații elementare. Evident $N = 1$, dacă restricțiile de tipul „ \leq ” lipsesc. Din motivele expuse mai sus în (13), nu se are în vedere inversarea matricei, care nu depinde de parametrul n , ci numai de m , aceasta are permanent aceeași dimensiune constantă – $m \times m$, $m \ll n$, iar complexitatea operației este $O(m^3)$, totodată, multiplicările la vector din aceleași considerente au complexitatea liniară $O(mn)$, echivalentă cu $O(n)$. Atunci (13), finalmente, poate fi scrisă în forma:

$$N \times [2 \times m^2 \times n + 4 \times m \times n - 2 \times n - m - m^2]. \quad (14)$$

Revenind acum la (PA), (4)-(6) formal N depinde de numărul de restricții de tipul „ \leq ”, dar, deoarece acestea se referă la componentele variabilei x , urmează că sunt nu mai mult de $2 \times n$ și N , în consecință, depinde de n . La evaluarea lui N , se disting câteva cazuri. Din cele menționate anterior, rezultă că, în problema auxiliară, este rațional de introdus inițial toate restricțiile, în acest caz, la pasul k prezintă interes doar situația când câteva dintre acestea devin pasive, ieșind la suprafața paralelipipedului. Dacă se constată că multiplicatorul lui Lagrange corespunzător este nenegativ, se îndeplinesc condițiile de minimum și procesul de calcul numeric s-a încheiat. Dacă însă multiplicatorul lui Lagrange este negativ, atunci componenta respectivă se readuce în matricea H . Important este că acest proces este monoton, funcționala descrește de la pas la pas, iar fiecare pas se termină neapărat după un număr finit de operații de aflare a minimumului. Aceste concluzii decurg din teorema fundamentală formulată în [5], care demonstrează că operatorul de proiecție minimizează o formă pătratică cu

the components of P in (12) have exactly the same size, thus, minimizing the subspace requires the same number of operations. So, overall, in solving the (PA) will be used at most

elementary operations. Obviously $N = 1$ if restrictions of the type “ \leq ” are missing. For the reasons stated above in (13) is not taken into account the matrix inversion, which does not depend on the parameter n , but only m ; it always has the same constant size – $m \times m$, $m \ll n$ and the complexity of the operation is $O(m^3)$, while the vector multiplications of the same considerations have linear complexity $O(mn)$, equivalent to $O(n)$. Then (13), finally, can be written in the form

Turning now to (PA) (4)-(6) formal N depends on the number of restrictions such as “ \leq ” but because they relate to parts of the variable x , it follows that there are no less than $2 \times n$ and N , therefore, depends on n . In the evaluation of N , there can be distinguished several cases. From the foregoing, it follows that the auxiliary problem is rational to introduce initially restrictions; in this case, in step k of interest is only the situation when some of them become passive, then coming out to the surface of the cuboid. If it turns out that corresponding Lagrange multipliers are non-negative, they fulfil the minimum conditions and numerical computation process has ended. But if the Lagrange multiplier is negative, then the matrix component is brought back to matrix H . It is important that this process is monotonous, functional, decreasing step by step, and each step necessarily terminates after a finite number of operations of finding the minimum. These conclusions arise from the fundamental theorem formulated in [5], demonstrating that the operator projection minimizes a quadratic form with

matrice simetrică și pozitiv definită în cel mult n pași. Însă rigurozitățile matematice de mai sus îmbunătățesc esențial aceste rezultate, demonstrând că, în cazul matricei unitare, minimumul se atinge dintr-un singur pas, numărul de operații elementare reducându-se considerabil, ceea ce a făcut ca algoritmul elaborat să fie declarat optimal. Astfel, complexitatea acestui algoritm se poate scrie sub forma:

$$O(nm^2, N). \tag{15}$$

Suplimentar la aceste afirmații, componenta respectivă, odată readusă în matrice, determină multiplicatorul lui Lagrange corespunzător să-și păstreze pe întreg parcursul iterației semnul nenegativ în procesul de calcul, fapt dovedit și de formulele analitice (10)-(11), ceea ce exclude revenirea a doua oară la suprafață a componentei respective, menținând, totodată, valoarea numerică a acestei componente în interiorul paralelipipedului, grație alegerii adecvate a pasului α^k . Așadar, $N \leq 2 \times n$ pentru restricțiile de jos asupra variabilelor, analogic, aceeași estimare este valabilă și pentru restricțiile de sus, în consecință, per total $N \leq 4 \times n$. De observat comportamentul formării mulțimii de indici pentru această situație

$$J_{activ}(p_k^0) \setminus \{j\}, J_{pasiv}(p_k^0) \cup \{j\},$$

anume mulțimea indicilor pasivi mereu se lărgiște și includerea este strictă:

$$J_{pasiv}(p_{k+1}^0) \supset J_{pasiv}(p_k^0),$$

proces care poate dura numai un număr finit de ori, cumulat cu descreșterea funcționalei, aceasta înseamnă că mulțimile de indici $J(p_k^0)$ nu se pot repeta, ca rezultat – convergența algoritmului rezolvării problemei auxiliare într-un număr finit de operații.

5. Compararea cu alți algoritmi

Desigur, o astfel de evaluare optimală a complexității algoritmului PG pornește de la forma specială a restricțiilor. De aceea, pentru compararea cu alți algoritmi, sunt necesare

symmetric matrix and defined positively in most n steps. But mathematical rigors above essentially improve these results, demonstrating that in the case of unitary matrix, the minimum is reached in one step, reducing the number of elementary operations considerably, which made the developed optimal algorithm to be declared as optimal. Thus, the complexity of this algorithm can be written as:

In addition to these statements, the respective component, once returned to the matrix determines the Lagrange multiplier appropriate to retain throughout iteration the mark of non-negative in the calculation, which has been proven also by analytical formulas (10)-(11), which excludes returning a second time to the surface of the respective component, while maintaining the numerical value of this component within the cuboid, through appropriate choice of step α^k . So, $N \leq 2 \times n$ for lower restrictions on variables, analogically, the same estimation applies to the above restrictions, consequently, overall $N \leq 4 \times n$. The observed trend in setting of indices for the situation

namely passive indices crowd always broadens and inclusion is strict

a process which can take only a finite number of times, combined with decreased functionality, it means that the sets of indices $J(p_k^0)$ cannot be repeated; as a result – convergence algorithm in solving the problem in a finite number of auxiliary operations.

5. Comparison with other algorithms

Of course, such an assessment PG optimal algorithm complexity starts to form special restrictions. Therefore, comparing it with other algorithms requires some

câteva generalizări și estimări asimptotice. Este știut, bunăoară, că pentru problemele de optimizare de dimensiuni mari, încercarea de a le rezolva utilizând cea mai populară metodă – metoda simplex, deseori, se confruntă cu dificultăți, deoarece numărul de iterații, în acest caz, crește exponențial față de dimensiunea problemei. Ulterior, au fost propuși alți algoritmi, polinomiali în sensul complexității [4], dar departe de a fi și funcționali, de aceea, metoda – simplex rămâne, în continuare, cea mai des utilizată. Fie în continuare problema (4)-(6). Algoritmii direcți de rezolvare conduc la soluția exactă a sistemului. Metodele directe aduc sistemul prin transformări de echivalență, la un sistem particular (diagonal, triunghiular etc.), care se rezolvă prin metode elementare. Este cunoscut, la fel, că, în acest caz, complexitatea metodelor directe de rezolvare este de ordinul $O(n^3)$, spre deosebire de cele iterative pentru care complexitatea este de $O(n^2)$, dar ultimele construiesc doar un șir de aproximații pentru x , $\nu=1,2,3,\dots$ convergent în anumite condiții la soluția exactă, iar procesul iterativ se oprește când aproximația de ordin ν se încadrează între limitele unei precizii stabilite inițial, considerente din care acestea nu sunt examinate aici. În atare situație, se pot compara doar metode finite de rezolvare. Creșterea dimensiunii problemei și trecerea acesteia în clasa de probleme de dimensiuni mari, automat o transformă în problemă foarte dificilă de rezolvat, în speță la efectuarea calculului numeric. De aceea, este foarte important de propus metode și modele, care ar fi, într-un anumit sens „îmune” la dimensiunea problemei. Desigur că astfel de metode și modele sunt excepție de la regulă, căci majoritatea depind esențial de acest „blestem” al problemelor de optimizare. Modelul *PG* este anume un astfel de caz fericit, iar numărul minim de operații elementare necesare găsirii soluției permite declararea acestuia drept algoritm optimal pentru probleme de opti-

generalizations and asymptotic estimations. It is known, for instance, that for large optimization problems, trying to solve them using the most popular method – the simplex method often faces difficulties because the number of iterations in this case, increases exponentially compared to the size of the problem. Subsequently, other algorithms have been proposed, within the meaning of polynomial complexity [4], but far from being functional and, therefore, the simplex method still remains the most widely used. Let us consider the problems (4)-(6). The direct algorithms lead to finding the exact solution of the system. Direct methods bring equivalence transformations to the system, to a particular system (diagonal, triangular etc.), which are resolved by elementary methods. It is also known that in this case, the complexity of direct methods of resolution is of the order $O(n^3)$, as opposed to the iterative complexity for which the complexity is $O(n^2)$, but the last build only a number of approximations to x , $\nu=1,2,3,\dots$ convergent under certain conditions to the exact solution and the iterative process stops when the approximation ν order falls within the limits of accuracy set initial considerations of which are not examined here. In such a situation can be compared only finished solving methods. Increasing the size of the problem and its passage in large class of problems automatically turns very difficult problem to solve, namely the numerical calculations. It is therefore very important to propose methods and models that would be in some sense “immune” to the extent of the problem. Of course such methods and models are an exception, as most depend essentially on the “curse” of optimization problems. The *PG* model is exactly a happy event, and the minimum number of elementary operations needed to find its solution allows the declaration as optimal algorithm for nonlinear optimization problems with special

mizare neliniară cu restricții speciale, numărul de operații elementare fiind

$$4 \times n \times \left[2 \times m^2 \times n + 4 \times m \times n - 2 \times n - m - m^2 \right]. \quad (16)$$

De menționat că, în cazul când în (4)-(6) m tinde către n , modelul practic se transformă într-unul clasic liniar, complexitatea algoritmului devenind $O(n^4)$. Dacă se compară acest rezultat cu complexitatea din [4] – un record în domeniu, dată în forma $O(nm^5 \log^2 m)$, m, n – însemnând dimensiunile matricei restricțiilor și care în condiții similare crește până la $O(n^6 \log^2 n)$, se constată o diminuare a complexității pentru algoritmul descris mai sus cu peste două ordine, suplimentar, în algoritmul construit, se iau în evidență ușor și restricțiile asupra semnelor variabilelor.

6. Concluzii

În studiu, s-a construit un algoritm optimal de rezolvare a problemei auxiliare a modelului PG , funcțional din toate punctele de vedere. Complexitatea acestui algoritm este $O(nm^2, N)$, dată de (15), numărul de operații elementare este limitat de (16). Matricea supusă inversării nu depinde de dimensiunea problemei n , ci numai de m , altfel spus, matricea inversabilă are permanent aceeași dimensiune constantă – $m \times m$, $m \ll n$, iar complexitatea operației este $O(m^3)$. Pornind de la aceste evaluări, se poate concluziona că PG este absolut funcțional și practic „imun” [2] la dimensiunea problemei de optimizare.

restrictions, the number of elementary operations being

It should be noted that if (4)-(6) m tends to n , the model practically turns into a classic linear algorithm, the complexity becomes $O(n^4)$. If we compare this result with the complexity of [4] – a record in this field, given in the form $O(nm^5 \log^2 m)$, m, n – meaning the matrix dimensions restrictions and similar conditions of increase up to $O(n^6 \log^2 n)$, there is a decrease in the complexity of the algorithm described above by more than two orders with additional built in algorithm that take out easily as well as the restrictions on the sign variables.

6. Concluzions

In the study, we constructed an algorithm for solving the problem of auxiliary optimal model PG , functional in all respects. The complexity of this algorithm is $O(nm^2, N)$, given by (15), the number of elementary operations is limited by (16). The matrix subject to the reversal of the matrix does not depend on the size of the problem n , but only by m , that is, the inverse matrix that always has the same constant size – $m \times m$, $m \ll n$, and the operation complexity is $O(m^3)$. Based on these assessments, it can be concluded that PG is fully functional and practically “immune” [2] to the size of the optimization problem.

Referințe/References:

1. GÂRLĂ E. *Științificitatea strategiilor de dezvoltare economică*. Academia de Studii Economice a Moldovei, 2015, p. 255. ISBN 978-9975-75-762-1.
2. GÂRLĂ E. Model imun la dimensiunea problemei de optimizare, *Analele ASEM*, Ediția a IX-a, 2011, p. 255-259. ISBN 978-9975-75-567-2.
3. PSHENICHNIY B. *Nonsmooth optimization and nonlinear programming*. Pergamon Press, 1978.
4. KHACHIYAN L.G. *Polynomial algorithms in linear programming*. Computational Mathematics and Mathematical Physics, volume 20, Issue 1, 1980, pages 53-72.
5. PEARSON J. D. *Variable metric methods of minimization*. Research Analysis Corporation, McLean, Virginia, 1968.