ИЗВЕСТИЯ
ТОМСКОГО ОРДЕНА ОКТЯБРЬСКОЙ РЕВОЛЮЦИИ
И ОРДЕНА ТРУДОВОГО КРАСНОГО ЗНАМЕНИ
ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА ИМ. С. М. КИРОВА

Том 277 1977

ПРИМЕНЕНИЕ СЛУЧАЙНОГО ПОИСКА В ОДНОЙ ЗАДАЧЕ НЕЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

В. В. ЗАХАРОВ, В. П. ИВАНЧЕНКОВ, В. А. ФИЛИНОВА

(Представлена научным семинаром лаборатории вычислительной техники и автоматизации НИИ ЯФ ЭА)

Для решения задач нелинейного программирования небольшой размерности и с небольшим числом ограничений кажется естественным привлекать методы статистического поиска, обеспечивающие простые машинные алгоритмы. В настоящей статье предлагается применить для этого метод случайного поиска [1] в рамках одной схемы функций штрафа [2].

1. Постановка задачи

Пусть $F(x)$ и $f_j(x)$ ($j \in I$, $I=\{1, 2, ..., m\}$) — выпуклые функции векторного аргумента $x=(x_1, ..., x_n)$. В работе рассмотрена задача: найти

$$\min_X \{F(X) \mid f_j(X) \geqslant 0,\ j \in J\}. \qquad (1)$$

Следует подчеркнуть, что сходимость рассматриваемого варианта метода штрафных функций доказана в [3] для выпуклых функций F, f_j. Тем не менее в этой же работе были рассмотрены примеры решения невыпуклых задач «хорошего поведения», т. е. таких, в которых кривизна поверхностей $F(x)=\text{const}$, $f_f(x)=\text{const}$ не была слишком большой. Метод функций штрафа был выбран в настоящей работе в сочетании со случайным поисковым движением по той причине, что он позволял надеяться на приближенное решение рассматриваемой далее практической задачи, в отношении выпуклости которой не было уверенности.

2. Описание метода

Решение задачи в [2], [3] сводится к поэтапной минимизации функции

$$A_m(X)=\sum_{j=1}^{m^1} f_j(X), \qquad (2)$$

представляющей собой штраф за приближение к границе области определения R функции $F(X)$.

В настоящей работе на $(k+1)$-м этапе поиска $\min\limits_{X\in R} A_m(X)$ (область $R=\{f_j(X)\geqslant 0,\ j\in J\}$) функция (2) дополнялась ограничением:

$$f_{m+1}(X)=F(X^{0k})-F(X)\geqslant 0,$$

где X^{0k} — точка траектории поиска на k-м этапе, в которой функция $A_{m+1}(X)=A(X)$ достигает наименьшего значения.

Модификация использованного метода штрафа состояла еще в том, что на k-м этапе не ставилась задача отыскания $\min A(X)$ с большой точностью. Более того, выход из k-го этапа минимизации происходил тогда, когда разрешающая способность метода случайного поиска (шаг метода) уменьшалась до некоторой фиксированной величины τ.

3. Алгоритм

Рассмотрим алгоритмическую реализацию метода на некотором, k-м этапе. Основной задачей этапа является получение точки $X^{0,k+1}$ такой, что $F(X^{0,k+1})<F(X^{0k})$. Спуск к этой точке осуществляется по поверхности штрафа $A(X)$ методом случайного поиска с шагом τ. Не трудно видеть, что $f_{m+1}(X^{0k})=0$ и $A(X^{0k})=+\infty$. Поэтому неравенство для дополнительного $(m+1)$-го ограничения имеет смысл взять в виде

$$f_{m+1}(X)=F(X^{0k})-F(X)+\varphi\geqslant 0,$$

где $\varphi>0$.

Теперь уже можно искать $\min A(X)$ методом случайного поиска. В соответствии с известным алгоритмом случайного поиска с пересчетом [1] принимаем точку X^k за начальную точку траектории поиска на k-м этапе: $X^k\equiv X_0^k$. В кубической окрестности точки (ребро куба равно τ) X_0^k производим равномерное рассеяние точек до тех пор, пока не найдется хотя бы одна из них, в которой значение $A(X)$ меньше, чем $A(X_0^k)$. Обозначим эту удачную точку через X_1^k, вновь примем ее за центр куба с ребром τ и вновь будем производить равномерно распределенные испытания в этом кубе и т. д.

Для параметра τ введем адаптацию к условиям поиска: величину τ будем уменьшать в q раз, если нарушено хотя бы одно из $(m+1)$ ограничений; величину τ будем увеличивать в q раз в противном случае. Начальное значение полагалось равным $\overline{\tau}$. Если выполнялось $\tau<\underline{\tau}$, то k-й этап считался оконченным и полагалось $X_L^k=X^{k+1}$, причем, очевидно,

$$A(X^k=X_0^k)>A(X_1^k)>\cdots>A(X_{L-1}^k)>A(X_L^k=X^{k+1}).$$

Можно предположить (и практические вычисления это подтверждают), что на заключительных этапах поиск будет происходить в узком овраге. Вследствие этого в кубе с центром X_l^k необходимо провести чрезвычайно много испытаний, прежде чем выполнится $A(X_l^k)>A(X_{l+1}^k)$. Против такой ситуации введено следующее видоизменение вычислительного алгоритма. Если количество испытаний в окрестности X_1^k превысило $N_э$ (зацикливание), то из точки X_0^k производится вторично спуск по описанному алгоритму до тех пор, пока не выполнится

$\tau<\underline{\tau}$ (т. е. естественное окончание этапа) либо вновь произойдет зацикливание в точке $X^k_{l_2}$. Пусть $A(X^k_{l_2})<A(X^k_{l_1})$. В этом случае в качестве X^k_L выбирается точка

$$X^k_L=\frac{X^k_{l_2}-X^k_R}{2},$$

где точка X^k_R есть точка пересечения границы области R с прямой $X^k_{l_2}$ $X^k_{l_1}$ в направлении убывания $A(X)$ вдоль этой прямой. На этом k-й этап также оканчивается.

Вся процедура решения задачи (1) продолжается до тех пор, пока не будет исчерпано заданное заранее количество вычислений функции $F(X)$, равное N.

4. Блок-схема и ее описание

Более детально строение алгоритма можно уяснить из приводимой блок-схемы. Поясним ее.

Блок 1 осуществляет формирование команд программы с учетом числа переменных n и числа ограничений m.

Блок 2. Признак d определяет номер спуска из начальной точки X^k_L; если $d=1$, то из точки X^k_L спуск совершен уже дважды. Рабочий шаг τ в начале k-го этапа поиска принимает значение, равное $\underline{\tau}$. В блоке 2 вычисляются начальные значения $F(X)$ и $A(X)$.

Блок 3. Точка X^k — центр поиска на k-м этапе. Начальное значение X^k равно X^k_0.

Блок 4. Счетчик (τ) необходим для подсчета числа последовательных точек, в которых $A(X^\xi)>A(X^k_l)$, где X^ξ — случайная, равномерно распределенная точка в кубе с центром $X^k_{l^\xi}$.

Блоки 5—6. производят вычисление $f_j(X^\xi)$. Если условие $f_j(X^\xi)\geqslant 0$ не выполняется, то шаг τ уменьшается в q раз (блок 17).

Блок 7. Счетчик (F) контролирует число вычислений $F(X)$, если $\mathrm{cr}\,(F)=N$, то происходит печать найденных $F_{\min}$ и $X_{\min}$. Поиск прекращается (блок 18).

Блоки 8—9. M — текущее минимальное значение $F(X)$, X^M — точка, соответствующая M.

Блок 10. Рабочий шаг τ увеличивается в q раз, если для точки X выполнилось m ограничений.

Блок 11 проверяет: $A(X^\xi)<A(X^k_l)$. Если да, то→блок 26: $A(X^k_{l+1}):=A(X^\xi)$, $X^k_{l+}:=X^\xi$, иначе $\mathrm{cr}\,(\tau):=\mathrm{cr}\,(\tau)+1$.

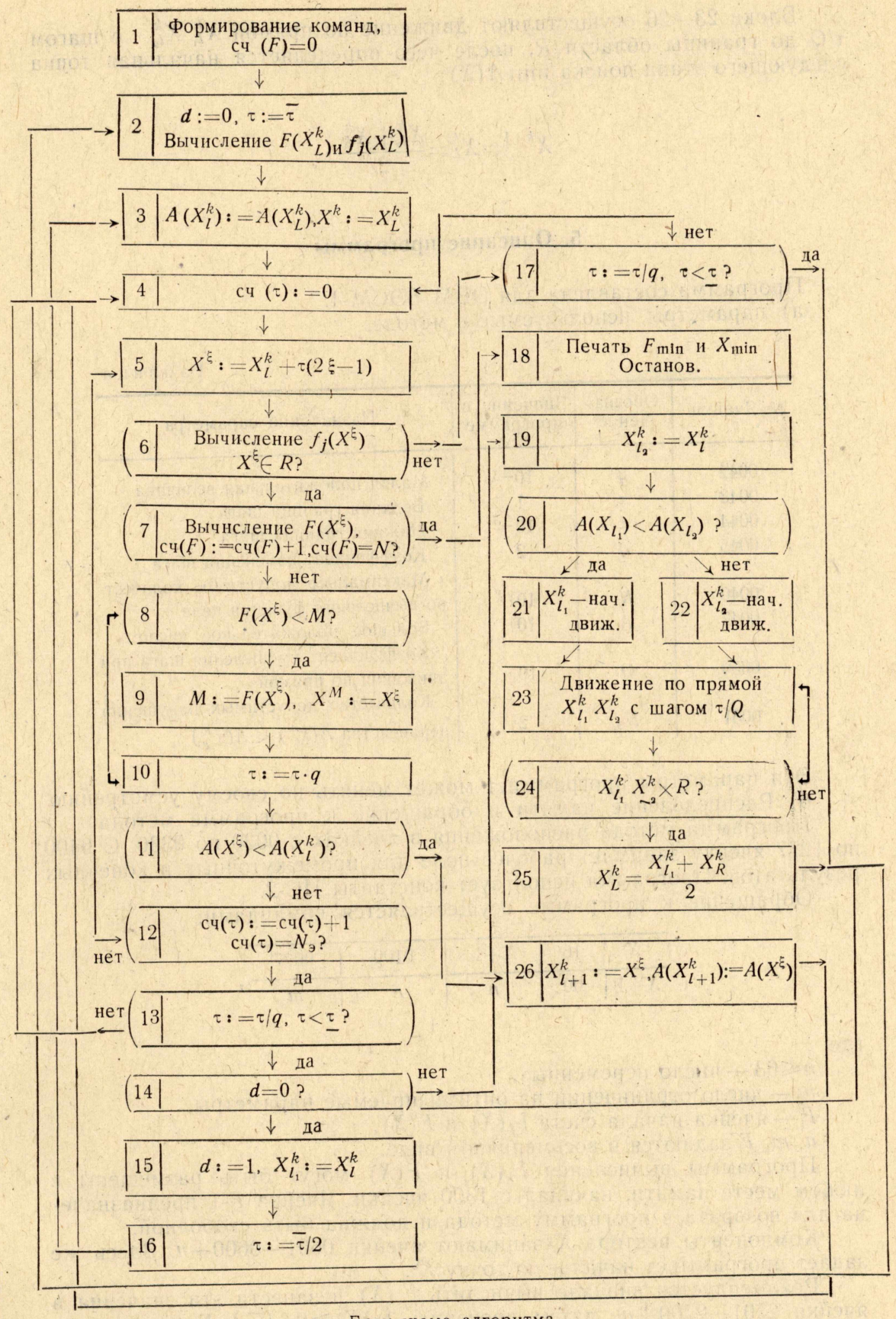
Блоки 12—16. Если $\mathrm{cr}\,(\tau)=N_э$, то шаг τ уменьшается в q раз, иначе (в блоке 14) проверяется: $d=0$?

Если да, то $X^k_{l_1}:=X^k$ и $d=1$.

С шагом $\tau=\tau/2$ еще раз осуществляется спуск до получения точки $X^k_{l_2}$.

Блоки 19—22. Если $d\neq 0$, то $X^k_{l_2}:=X^k_l$.

Делается проверка $A(X^k_{l_1})<A(X^k_{l_2 2})$, чтобы выбрать направление движения по прямой $X^k_{l_1}$ $X^k_{l_2}$.



Блок-схема алгоритма

Блоки 23—26 осуществляют движение по прямой $X^k_{l_1}\ X^k_{l_2}$ с шагом τ/Q до границы области R, после чего определяется начальная точка следующего этапа поиска $\min A(X)$:

$$X^{k+1}=X^k_L=\frac{X^k_{l_2}+X^k_R}{2}.$$

5. Описание программы

Программа составлена для ЭВМ БЭСМ-4.

а) параметры, используемые в методе.

Таблица

№ Ячейки	Обозначение	Значение в программе	Назначение параметра
0042	φ	10^{-5}	Малая положительная величина
0043	τ	1	Верхняя граница шага
0044	τ	10^{-2}	Нижняя граница шага
0045	q	2	Коэффициент изменения шага
0046	N	10^3	Максимально допустимое количество вычислений функции цели
0047	A	10^5	Большое положительное число
0050	Q	10	Коэффициент уменьшения шага при движении по прямой
0051	$N_э$	2	Количество допустимых нарушений неравенства $A(x^{\xi}) < A(x^{\kappa}_e)$

Эти параметры программист может менять по своему усмотрению.

б) Распределение памяти и обращение к программе метода.

Программа метода расположения в ячейках с 0030 по 0330. С 0400 по 1277 ячейку занимает рабочее поле для промежуточных и конечных результатов. Программа использует константы ИС-2.

Обращение к программе осуществляется командами:

X	16	$X+1$	0100	0077
$X+1$	00	n	F	m

,

где

$n\leqslant 63$ — число переменных,

m — число ограничений на оптимизируемые параметры,

F — ячейка начала счета $f_j(X)$ и $F(X)$,

n, m, F задаются в восьмеричном виде.

Программы вычисления $f_j(X)$ и $F(X)$ могут быть размещены в любом месте памяти, начиная с 1300 ячейки. Ячейка F-1 предназначена для возврата в программу метода и должна быть свободной.

Компоненты вектора X занимают ячейки 0601—0600+n. Здесь же задает программист начальную точку X^0.

Рекомендуется вначале вычислить $f_j(X)$ и занести эти значения в ячейки 2701—2700+m, затем проверить $f_j(X)\geqslant 0$, $j\in J$. Если хотя бы одно из ограничений не выполнилось, необходимо передать управление

в ячейку 0076, иначе вычислять $F(X)$. Значение $F(X)$ необходимо записать в 0600 ячейку. По окончании счета $F(X)$ передать управление в ячейку F—1.

Текущее минимальное значение функции и точка, ему соответствующая, хранятся в ячейках 0400—0400+n. Рабочий шаг τ хранится в 0060 ячейке.

6. Пример применения метода к производственной задаче

Задача состояла в определении параметров тракта транспортировки заряженных частиц одного ускорителя, оптимальных с точки зрения минимума стоимости конструкции в целом. Тракт состоял из 13 участков: 7 квадрупольных линз и 6 участков дрейфа частиц. Каждый участок однозначно определяется набором параметров в [4]

$$k_i,\ l_i,\ \beta_i,\ \alpha_i\ \gamma_i\ (i=\overline{1,13}).$$

Величины β_i, α_i, γ_i вычисляются по следующим формулам.
Для горизонтальной плоскости фокусировки линз:

$$\beta_i=\gamma_{0i}\sin^2\Theta_i-2\alpha_{0i}\frac{1}{\sqrt{k_i}}\sin\Theta_i\cos\Theta_i+\beta_{0i}\cos^2\Theta_i,$$

$$\alpha_i=\gamma_0{}^i\frac{1}{\sqrt{k_i}}\cos\Theta_i\sin\Theta_i+\alpha_{0i}(\cos^2\Theta_i-\sin^2\Theta_i)+\beta_{0i}\sqrt{k_i}\sin\Theta_i\cos\Theta_i,$$

$$\gamma_i=\gamma_{0i}\cos^2\Theta_i+2\alpha_{0i}\sqrt{k_i}\cos\Theta_i\sin\Theta_i+\beta_{0i}k_i\cos^2\Theta_i,(i=1,\ldots,7).$$

Для участков дрейфа пучка частиц:

$$\left.\begin{aligned}\beta_i&=\beta_{0i}-2\alpha_{0i}l_i+\gamma_{0i}l_i^2,\\ \alpha_i&=\alpha_{0i}-\gamma_{0i}l_i\\ \gamma_i&=\gamma_{0i}\end{aligned}\right\}\begin{matrix}k=0.\\(i/2=1,\ldots,6).\end{matrix}$$

Для вертикальной плоскости фокусировки линз:

$$\beta_i=\gamma_{0i}\frac{1}{k_i}\mathrm{ch}^2\Theta_i-2\alpha_{0i}\frac{1}{\sqrt{k_i}}\mathrm{ch}\,\Theta_i\,\mathrm{ch}\,\Theta_i+\beta_{0i}\mathrm{ch}^2\Theta_i,$$

$$\alpha_i=-\gamma_{0i}\frac{1}{\sqrt{k_i}}\mathrm{ch}\,\Theta_i\,\mathrm{ch}\,\Theta_i+\alpha_{0i}(\mathrm{ch}^2\Theta_i+\mathrm{ch}^2\Theta_i)-\beta_{0i}\sqrt{k_i}\,\mathrm{ch}\,\Theta_i\cdot\mathrm{ch}\,\Theta_i,$$

$$\gamma_i=\gamma_{0i}\,\mathrm{ch}^2\Theta_i-2\alpha_{0i}\sqrt{k_i}\,\mathrm{ch}\,\Theta_i\,\mathrm{ch}\,\Theta_i+\beta_{0i}\,k_i\,\mathrm{ch}^2\Theta_i,(i=1,\ldots,7).$$

Здесь в формулах всюду $\Theta_i=l_i\sqrt{k_i}$.
В этих обозначениях функция цели $F(X)$ имеет вид:

$$F(X)=4\sum_{i=1}^{7}l_i\sqrt{C\cdot\beta^*_{\text{гор}.i}\cdot\beta^*_{\text{верт}.i}},\quad C=\text{const}.$$

здесь i — номер участка тракта с линзой, $\beta^*_{\text{гор}.i}$, $\beta^*_{\text{верт}.i}$ — максимальные на участке i в вертикальной или горизонтальной плоскости фокусировки линз. Переобозначим вектор переменных

$$X=(x_1=x_3,\ x_2=k_4,\ x_3=l_4,\ x_4=l_6,\ x_5=l_8,\ x_6=l_{10}).$$

На переменные x были наложены ограничения:

$$0{,}09 \leqslant x_1 \leqslant 0{,}64;\ 0{,}09 \leqslant x_2 \leqslant 1;\ 4 \leqslant x_3 \leqslant 10;$$
$$4 \leqslant x_4 \leqslant 10;\ 4 \leqslant x_5 \leqslant 9;\ 4 \leqslant x_6 \leqslant 9;$$
$$\sum_{i=3}^{6} X_i = 31{,}92.$$

Остальные параметры не менялись.

Для действующего тракта значения переменных были равны:

$$x_1 = x_2 = 0{,}438;\ x_3 = 8;\ x_4 = 7{,}96;$$
$$x_5 = 7{,}96;\ x_6 = 8;\ F(X) = 0{,}08230.$$

После минимизации $F(X)$ изложенным выше методом получили:

$$F(X) = 0{,}00135;$$
$$x_1 = 0{,}184;\ x_2 = 0{,}375;\ x_3 = 5{,}084;$$
$$x_4 = 8{,}460;\ x_5 = 8{,}895;\ x_6 = 9{,}480.$$

Машинное время, затраченное на поиск $F(X^*) = 0{,}00135$ из данной начальной точки, составило примерно 13 мин. Подходящий результат методом случайного поиска без всяких модификаций ранее был получен за 4 часа работы БЭСМ-4.

Алгоритм метода в том виде, как он приведен на блок-схеме, довольно чувствителен к оврагам, поэтому когда к имевшимся ограничениям были добавлены еще ограничения $\beta_7(X) \leqslant 0{,}0833$; $\gamma_7(X) \leqslant 33{,}3$, имевшие ярко выраженный овражный вид, то метод быстро зацикливал.

Таким образом, для повышения эффективности метода следует увеличить его разрешающую способность в овражной ситуации.

Приложение

Программа метода

0100		56	—	0303	0056	1	1	32	0001	0130	7777
1	4	55	—	7734	0072	2		00	—	—	0062
2	1	14	0064	0072	0072	3		72	—	0072	—
3	4	55	—	7732	0073	4		54	0107	0040	0053
4	4	55	—	7731	0074	5		07	0053	0040	0053
5		14	0114	0074	0074	6		13	0041	0053	0053
6		00	0047	—	0400	7		23	0053	—	0040
7		00	—	—	0067	0140		05	7762	0040	0054
0110		00	—	—	0061	1		02	0054	7761	0054
1		72	—	0072	—	2		05	0054	0060	0054
2	5	00	0600	—	0500	3	3	01	0054	0700	0600
3	1	32	0002	0112	7777	4	1	32	0002	0134	7777
4		00	—	—	0063	5		72	—	0073	—
5		00	0043	—	0060	6	3	16	0147	—	7777
6		72	—	0073	—	7		01	7761	0067	0067
7	3	16	0120	—	7777	0150		15	0067	0046	—
0120		56	0600	0300	0500	1		36	—	0272	—
1	2	04	7761	2700	0075	2		02	0600	0400	—
2		01	0075	0065	0065	3		76	—	0157	0070

3	1	32	0002	0121	7777
4		04	7761	0042	0075
5		01	0075	0065	0065
6		00	0065	—	0064
7		72	—	0072ч	—
0130	5	00	0500	—	0700
2		01	0075	0070	0070
3	1	32	0002	0161	7777
4		02	0700	0600	0075
5		01	0075	0042	0075
6		56	—	0312	—
7		01	0075	0070	0070
0170		02	0070	0064	—
1		76	—	0177	—
2		72	—	0072	—
3	5	00	0600	—	0700
4	1	32	0001	0173	7777
5		00	0070	—	0064
6		56	—	0132	—
7		01	0062	7761	0062
0200		15	0062	0051	—
1		36	—	0203	—
2		56	—	0133	—
3		04	0060	0045	0060
4		02	0060	0044	—
5		36	—	0207	—
6		56	—	0132	—
7		15	0063	—	—
0210		76	—	0221	—
1		00	7761	—	0063
2		72	—	0072	—
3	5	00	0700	—	1000
4	1	32	0002	0213	7777
5		00	0064	—	0066
6		04	0043	7762	0060
7		56	—	0126	—
6		56	—	0237	—
7		00	0311	—	0157
0260		72	—	0072	—
1	6	01	1000	1200	0075
2	1	04	0075	7762	0600
3	1	32	0002	0261	7777
4		01	0061	7761	0061
5		56	0307	0111	0076
6		04	0060	0045	0060
7		02	0060	0044	—
0270		76	—	0132	—
1		56	—	0207	—
2		72	—	0072	—
3		16	0274	7501	7610
4	1	72	0400	0027	0400
5		16	0276	7501	7610
6		72	0060	0027	0063
7		77	—	—	—
0300		72	—	0074	—
4		72	—	0072	—
5	5	00	0600	—	0400
6	1	32	0001	0155	7777
7		05	0060	0045	0060
0160		72	—	0074	—
1	2	04	7761	2700	0075
0220		00	—	—	—
1		72	—	0072	—
2	5	00	0700	—	1100
3	1	32	0002	0222	7777
4		04	0060	0050	0060
5		02	0066	0064	—
6		36	—	0235	—
7		00	—	—	—
0230		72	—	0072	—
1	4	00	1000	—	0075
2	5	00	1100	—	1000
3	1	00	0075	—	1100
4	1	32	0002	0231	7777
5		72	—	0072	—
6	5	00	1000	—	0600
7	5	01	0600	0060	0600
0240	6	02	0600	1000	0075
1	6	02	1077	0777	0057
2		05	0057	0075	0057
3	6	02	1100	1000	0075
4		04	0057	0075	0075
5	3	01	0075	0777	0577
6	1	32	0003	0240	7777
7		00	0310	—	0076
0250		16	0251	0145	0157
1		00	—	—	—
2		72	—	0072	—
3	5	00	0600	—	1200
4	1	32	0002	0253	7777
5		72	—	0072	—
3		36	—	0266	—
4		04	7761	0075	0075
5		01	7761	0056	0056
6		56	—	0167	—
7		15	1001	0701	—
0320		36	—	0323	—
1		72	—	0072	—
2		56	—	0222	—
3		04	0060	7760	0060
4		56	—	0126	—

1	56	—	0121	0065
0302	00	—	—	—
3	16	0304	7501	7610
4	52	0042	0042	0051
5	72	—	0077	—
6	56	0307	0101	0076
7	16	—	0266	—
0310	16	—	0257	—
1	05	0060	0045	0060
2	02	0075	—	—

ЛИТЕРАТУРА

1. Л. А. Растригин. Статистические методы поиска. М., «Наука», 1968.
2. C. W. Carroll. The CRST for Optimizing Nonlinear Restraines systems. Oprns. Res., v. 9, № 2, 1961.
3. A. V. Fiacco, G. P. Mc Cormick. Computational Algorithm for SUMT for Nonlinear programmung. Manag. Sc., v. 10, № 4, 1964.
4. В. П. Иванченков. Вопросы применения корреляционно-экстремальных систем в задачах электронной оптики. Диссертация. Томск, 1969.