ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ
1973 год Выпуск 56

УДК 518.12:681.3.06

# АЛГОРИТМ И ПРОГРАММА УСЛОВНОЙ ОПТИМИЗАЦИИ

В.К. Королев

## I. Формулировка задачи

Рассмотрим следующую задачу оптимизации

$$\min\{f(x) \mid Ax \le c,\ \varphi(x,\alpha)=0,\ \alpha_H \le \alpha \le \alpha_k\}. \qquad (I)$$

Здесь

$$x^T=(x_1,\dots,x_n);\quad A=\|a_{ij}\|_{(m\times n)};\quad c^T=(c_1,\dots,c_m);$$

$$f(x)=\alpha\cdot V(x)+\tilde{f},$$

где

$$\alpha=\begin{cases}+1, & \tilde{f}=0,\\ -1, & \tilde{f}>0.\end{cases}$$

Задача (I) есть задача отыскания (при каждом фиксированном $\alpha\in[\alpha_H,\alpha_k]$) минимума и максимума функции $V(x)$ на поверхности $\varphi=\{x \mid \varphi(x,\alpha)=0\}$ в допустимой области $R=\{x \mid Ax\le c\}$.

Предполагается, что функции $V(x)$ и $\varphi(x)$, по крайней мере, дважды непрерывно дифференцируемы.

Геометрический смысл задачи иллюстрируется примерами I и 2 (см. стр. 65).

Задача (I) возникает, например, в электронике при построении областей рабочих сигналов для интегральных схем [I]; аналогичные задачи рассматриваются в работах [2] (энергетика) и [3] (математическая экономика). От стандартной формулировки задач нелинейного программирования задача (I) отличается наличием ограничения-равенства. Заметим, что в задаче с линейными



границами такое выделение нелинейного ограничения-равенства естественно. Если бы мы попытались с его помощью исключить одну из компонент вектора $x$ (что не всегда возможно), то пришли бы к задаче с нелинейными границами, решать которую было бы не проще. Необходимость разрабатывать какие-то эвристические способы решения задачи (I) вызывается следующими причинами. Во-первых, строгие теоретические критерии (например, типа необходимых и достаточных условий Куна-Таккера [4,5]) обоснованы лишь для задач выпуклого программирования, в то время как наша задача (I) является, вообще говоря, многоэкстремальной (см. стр. 65). Во-вторых, математические методы поиска решения (например, метод Эрроу-Гурвица [6] интегрирования системы дифференциальных уравнений, описывающих траекторию движения к решению) мало эффективны при практических расчетах (см., например, [4,7]). И, наконец, трудности чисто технического характера, возникающие при практической реализации того или иного алгоритма, заставляют заменять некоторые строгие приемы или критерии эвристическими.

Предлагаемый ниже алгоритм следует подходам, изложенным в работах [3] и [2]. В первой из них предлагается комбинировать задачи "ухода" с поверхности $\varphi$ и "спуска" на нее. Во второй работе эта идея используется в сочетании с приемами градиентных методов с малым шагом, причем "уход" совершается в касательной плоскости к поверхности $\varphi$. Наша модификация этих подходов заключается в привлечении приемов полношагового градиентного метода [5] и метода Розена [4] проектирования градиента.

Кратко алгоритм описан в [I]. С учетом последующих усовершенствований и более подробно алгоритм излагается ниже.

## 2. Алгоритм

Общая блок-схема алгоритма приведена на рис. I. Здесь прямоугольники обозначают арифметические операторы (или целые блоки), ромбики – логические, кружки – метки в программе. Вместо общепринятых "да" и "нет" на выходах логических операторов мы считаем более удобным писать "I" и "0".

На блок-схеме не показаны два внешних цикла: по $\alpha$ и по $\alpha$, так что приведенная на рис. I блок-схема относится лишь к решению задачи

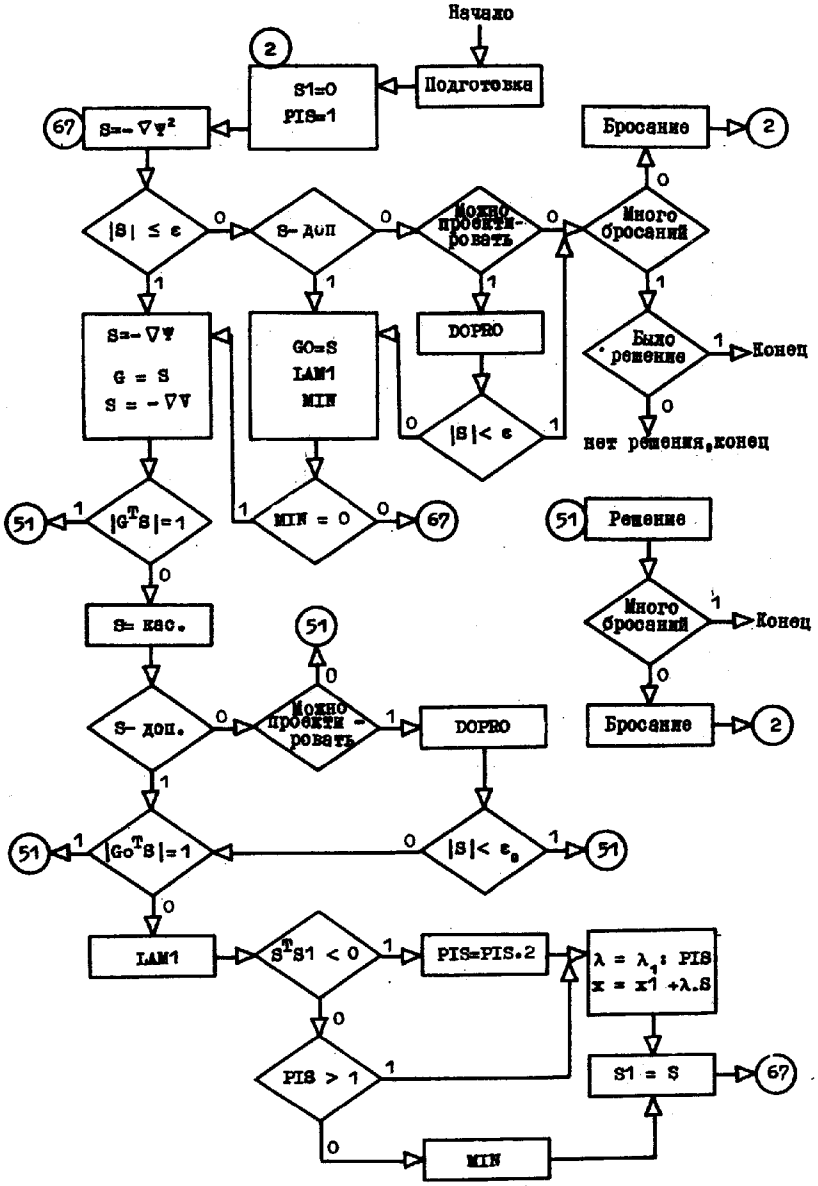


Рис. I. Общая блок схема алгоритма

$$\min\{V(x) \mid Ax \leq 0,\ \varphi(x) = 0\}. \qquad (I')$$

Дадим пояснения к общей блок-схеме. После ввода исходных данных проводится необходимая подготовка к работе программы: заготавливаются некоторые переменные; введенные координаты допустимой точки $x_s$ пересылаются в рабочий массив $x$; очищается массив $s_1$, в котором будет храниться вектор $s$ от предыдущей итерации поиска, а также обновляется переменная $\pi_s$, участвующая в колебательном движении к минимуму.

Для спуска на поверхность $\varphi$ составляется характеристическая функция [3]

$$\bar{\varphi}(x) \begin{cases} = 0, & \varphi(x) = 0, \\ > 0, & \varphi(x) \neq 0. \end{cases}$$

Например, можно положить $\bar{\varphi}(x) = \varphi^2(x)$. Теперь задача спуска на $\varphi$ сводится к нахождению нулевого минимума функции $\bar{\varphi}(x)$.

Вычисляется вектор-антиградиент этой функции: $s = -\nabla\bar{\varphi}$, и если он мал, то мы идем на вычисление вектора, касательного к $\varphi$. В противном случае определяем, является ли вектор $s$ допустимым. Если нет, но его можно спроектировать на некоторое (нарушаемое вектором $s$) граничное многообразие $\bar{R}$ ненулевой размерности, то $s$ проектируется на $\bar{R}$. Если эта проекция мала или размерность $\bar{R}$ равна нулю, проверяем, много ли было случайных бросаний точки в допустимую область $R$. Если нет, то производим это бросание и идем на 2; если много и при этом ни разу не было получено решения, то информация об этом печатается; если было решение, поиск прекращается - конец.

После того, как получен подходящий [5] вектор $s$, этот вектор запоминается в массиве $g_0$, по направлению $s$ определяется расстояние до границы области $\lambda'$ и решается задача одномерной минимизации ($\min f$ по направлению $s$).

Если найденный минимум ненулевой, то возвращаемся на 67. В противном случае вычисляем антиградиент функции $\varphi(x)$, запоминаем его ($g = -\nabla\varphi$) и вычисляем антиградиент функции $V(x)$ ($s = -\nabla V$).

Далее проверяем векторы $g$ и $s$ на коллинеарность. Если $|g^T s| = 1$, то мы имеем решение (точнее, стационарную точку). В

противном случае имеется возможность уменьшения $V$ вдоль поверхности $\varphi$. Вектор $s$ проектируется на касательную плоскость к поверхности $\varphi$, и определяется допустимость этого нового вектора $s$. Аналогично описанному выше проводится, если нужно, проектирование $s$ на нарушаемое им граничное многообразие $\bar{R}$ и в тех случаях, если эта проекция мала или коллинеарна вектору $g_0$, мы имеем решение.

Если $|g_0^T s| \neq 1$, определяется $\lambda'$ – шаг до границы и проверяется, нет ли колебательности при движении к минимуму. Если нет (то есть $s^T s_1 \geq 0$) и не было до этого (то есть $\pi_s = 1$), то на направлении $s$ отыскивается $\min V$, запоминается $s$ (в массиве $s_1$) и мы снова идем на 67.

Если $s^T s_1 < 0$, то $\pi_s$ удваивается, по направлению $s$ производится шаг величиной $\frac{\lambda'}{\pi_s}$, запоминается $s$ и управление передается на 67. Если $s^T s_1 \geq 0$, но $\pi_s > 1$ (то есть были колебания на предыдущих итерациях), то $\pi_s$ не увеличивается, но по-прежнему применяется "осторожная" тактика движения по направлению $s$.

Таким образом, в случае колебательного характера движения к минимуму мы застрахованы от бросков от границы до границы области $R$, колебания будут затухать, приводя в окрестность стационарной точки.

Заметим, что задача (I′) может иметь несколько изолированных решений или целое их многообразие; кроме того, процесс может остановиться в точке, не являющейся решением задачи (см. примеры I и 2). Поэтому с целью "прощупывания" всей области $R$ и увеличения надежности отыскания решения в алгоритме используется комбинация регулярного локального поиска со случайными бросаниями точки в допустимую область.

Описанная блок-схема алгоритма соответствует общей организации процесса оптимизации. Программа, реализующая этот алгоритм, написана на языке АКИ [8] и содержит I2 подпрограмм, выполняющих повторяющиеся процедуры. Ниже даются комментарии к подпрограммам. Порядок следования комментариев соответствует расположению подпрограмм в автокодовой программе (см.стр.7I). После названия подпрограммы в скобках указываются ее метка и шифр.



Подпрограмма дифференцирования (50. $GS$) осуществляет вычисление антиградиента функции $f(x)$: $s = -\nabla f$. Дифференцирование по $x_j$ производится с помощью безразностных формул [9] по трем точкам, причем шаг $\Delta x$ уменьшается до тех пор, пока приращения функции в точках $x + \Delta x$ и $x + \frac{\Delta x}{2}$ не станут достаточно малыми. Отметим, что для квадратичной функции получается точное значение производной (в пределах точности машины) при любом исходном значении $\Delta x$. В этой подпрограмме вычисляется также норма вектора $s$:

$$-g_s \equiv \| s \| = \sum_{i=1}^{n} s_i^2 .$$

Подпрограмма проверки $s$ на допустимость (55. $DOP$) для граничных точек $x$ (то есть тех, для которых выполняется хотя бы одно равенство $A_i x = c_i$) определяет, выводит ли вектор $s$ за пределы области $R$. Если да, то соответствующий вектор $A_i^T$ пересылается в двумерный массив $Q$ и запоминается $k_1$ – число таких векторов. Вектор $s$ допустим в точке $x$, если $k_1 = 0$.

Подпрограмма вычисления граничного шага (23. $\lambda'$), во-первых, запоминает текущее значение вектора $x$ в массиве $x_1$ и, во-вторых, для нарушенных ограничений (то есть тех, для которых $A_i s > 0$) вычисляет шаг до границы:

$$\lambda' = \min_i \left\{ \frac{c_i - A_i x}{A_i s} \right\} .$$

Подпрограмма одномерной минимизации (29. $\min(f)$) является одной из основных. Ее блок-схема приведена на рис. 2 вместе с блок-схемой подпрограммы контроля шага, пересчета $x$ и $f(x)$ и проверки конца минимизации (39. $\lambda - \varphi$). Первая из этих подпрограмм многократно обращается ко второй.

Обе подпрограммы совместно решают задачу

$$\min_{0 \leq \lambda \leq \lambda'} \left\{ \varphi(\lambda) \equiv f(x + \lambda s) \right\} \qquad (2')$$

при фиксированных векторах $x$ и $s$. Отметим, что в процессе решения задачи (I′) приходится дважды решать задачу (2′): для

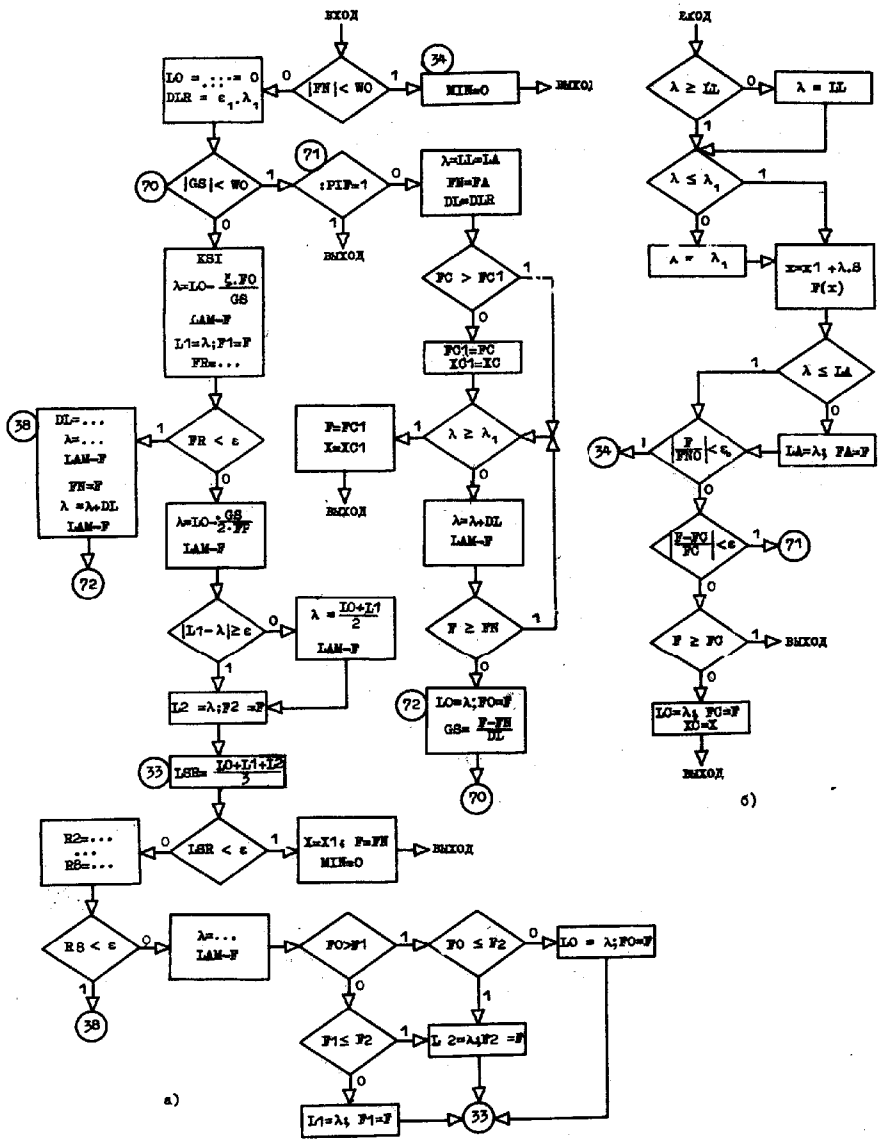


Рис. 2. Блок-схемы подпрограмм:
а) MIN(F); б) LAM-F



поиска минимума функции $V(x)$ и при спуске на $\varphi$. Если в первом случае нас устраивает любой локальный минимум, то во втором – только нулевой. Этим и объясняются описываемые ниже различия в подпрограмме "$min(f)$" при работе с функциями $V(x)$ и $\bar{\varphi}(x)$.

При входе в подпрограмму "$min(f)$" может оказаться, что текущее значение $\varphi_0$ минимизируемой функции мало; в этом случае сразу идем на выход, присвоив нулевое значение переменной $min$. В противном случае некоторые переменные обновляются: $\lambda_0$ – исходное значение $\lambda$; $\lambda_\alpha$ и $\lambda_\ell$ – соответственно правое и левое значения $\lambda$; $\varphi_0 = \varphi(\lambda_0)$; $\varphi_\alpha = \varphi(\lambda_\alpha)$; $\varphi_c$ и $\varphi_{c_1}$ – ординаты локального и глобального минимумов; $min$ – признак (не)нулевого минимума; $\Delta\lambda_z$ – приращение $\lambda$ для пошагового поиска.

Далее проверяем, мала ли величина $g_s$. Если да (то есть мы находимся в окрестности локального минимума) и, кроме того, минимизируется функция $V(x)$ (признак: $\pi_f = 1$), то мы идем на выход. Для функции $\bar{\varphi}(x)$ (признак: $\pi_f = 0$) в этой ситуации организуется пошаговый поиск (см.ниже) для отыскания нулевого минимума.

Если $g_s$ не мало, то, разлагая $\varphi(\lambda)$ в степенной ряд в точке $\lambda_0$:

$$\varphi(\lambda) = \varphi(\lambda_0) + \varphi'(\lambda_0)(\lambda - \lambda_0) + \frac{1}{2}\varphi''(\lambda_0)(\lambda - \lambda_0)^2 + \ldots, \quad (2)$$

мы находим абсциссу точки нулевого минимума линейной части разложения (2):

$$\lambda = \lambda_0 - \frac{\varphi(\lambda_0)}{\varphi'(\lambda_0)} = \lambda_0 - \frac{\varphi_0}{g_s}$$

В качестве первого, грубого приближения решения (2') мы берем

$$\lambda_1 = \lambda_0 - \xi\frac{\varphi_0}{g_s}, \quad \varphi_1 = \varphi(\lambda_1),$$

где $\xi$ – поправка (псевдослучайное число с равномерным распределением на [0,1]).

Теперь, имея две точки $(\lambda_0, \varphi_0)$ и $(\lambda_1, \varphi_1)$ и значение производной $g_s$ в первой точке, строим параболу, аппроксимирующую $\varphi(\lambda)$. Для разложения (2) получаем:

$$\frac{1}{2}\varphi''(\lambda_0) \equiv \varphi_2 = \frac{\varphi_1 - \varphi_0 - \varphi_3(\lambda_1 - \lambda_0)}{(\lambda_1 - \lambda_0)^2}.$$

Может оказаться, что $\varphi_2 \leq 0$ (парабола вогнутая или вырождается в прямую). В этих случаях применяется пошаговый поиск.

Если $\varphi_2 \geq \varepsilon$, то легко найти минимум параболы:

$$\lambda = \lambda_0 - \frac{\varphi_3}{2\varphi_2}.$$

Заметим, что если минимизируемая выпуклая функция квадратична, то это значение $\lambda$ дает решение (2'), а все дальнейшее будет служить лишь проверкой правильности решения.

Может случиться, что два значения $\lambda$, найденные из линейной и квадратичной аппроксимаций, совпадают. Тогда мы полагаем $\lambda = \frac{\lambda_0 + \lambda_1}{2}$.

Последующая процедура минимизации сводится к построению параболы по трем точкам и нахождению ее минимума. Найдем $\varphi(\lambda)$ и положим $\lambda_2 = \lambda$, $\varphi_2 = \varphi(\lambda_2)$. Нетрудно получить формулу для минимума параболы, проходящей через три точки: $(\lambda_0, \varphi_0)$, $(\lambda_1, \varphi_1)$ и $(\lambda_2, \varphi_2)$:

$$\lambda = \lambda_0 + \frac{1}{2}\,\frac{(\varphi_1 - \varphi_0)(\lambda_2 - \lambda_0)^2 - (\varphi_2 - \varphi_0)(\lambda_1 - \lambda_0)^2}{(\varphi_1 - \varphi_0)(\lambda_2 - \lambda_0) - (\varphi_2 - \varphi_0)(\lambda_1 - \lambda_0)}. \qquad (3)$$

Перед тем как считать по этой формуле, необходимо убедиться, что минимум параболы существует, то есть знаменатель в (3) $\geq \varepsilon$. Иначе парабола вогнута или вырождается, и мы идем на пошаговый поиск. Кроме того, неприятен случай, когда мы находимся очень близко от исходной точки (то есть величина $\lambda_{ср} = \frac{\lambda_0 + \lambda_1 + \lambda_2}{3}$ мала). В этом случае, очевидно, значения $\varphi_1$ и $\varphi_2$ мало отличаются от $\varphi_0$, и мы выходим из подпрограммы, приняв в качестве решения исходную точку.

После того как по формуле (3) вычислено $\lambda$ и найдено $\varphi = \varphi(\lambda)$, из четырех точек $(\lambda_0, \varphi_0)$, $(\lambda_1, \varphi_1)$ $(\lambda_2, \varphi_2)$ и $(\lambda, \varphi)$ отбрасывается точка с наибольшей ординатой и процесс (3) зацикливается на 33. Критерии окончания минимизации находятся в подпрограмме "$\lambda - \varphi$".

Опишем, как происходит пошаговый поиск. В процессе работы подпрограммы "$\lambda - \varphi$" запоминается наименьшее текущее значение $\varphi_c$ функции $\varphi(\lambda)$. Если найден локальный минимум этой функции, то проверяется, не меньше ли он найденного ранее минимума $\varphi_{c_1}$ (в начальный момент $\varphi_{c_1} >> 1$). Если меньше, то он запоминается, и если отрезок $[\lambda_0, \lambda']$ еще не исчерпан, то переменная $\lambda$ получает приращение $\Delta\lambda$, вычисляется $\varphi(\lambda + \Delta\lambda)$ и этот процесс повторяется до тех пор, пока $\varphi(\lambda)$ не начнет уменьшаться (то есть пока мы не перевалим через очередной "бугор" в область следующего минимума). Вычисляем в этой точке величину $\varphi_3$ и, оказавшись в первоначальной ситуации, возвращаемся на 70. Если отрезок $[\lambda_0, \lambda']$ пройден до его правого конца без уменьшения $\varphi(\lambda)$, то мы выходим из подпрограммы, приняв в качестве минимума точку $(\lambda_{c_1}, \varphi_{c_1})$.

Перед началом пошагового поиска мы сразу же отсекаем область ранее найденного минимума, полагая $\lambda_e = \lambda_\alpha$.

В тех случаях, когда нельзя получить минимум аппроксимирующих парабол, мы делаем шаг по $\lambda$ так, чтобы оказаться в ситуации, аналогичной началу минимизации (см. операторы 38 и 72).

Подпрограмма "$\lambda - \varphi$" (блок-схема на рис. 2,б), во-первых, ограничивает величину $\lambda$ в пределах $[\lambda_e, \lambda']$, во-вторых, в точке $\lambda$ пересчитывается вектор $x$ и функция $f(x)$. Кроме того, запоминается точка $(\lambda_\alpha, \varphi_\alpha)$ для самого правого из текущих значений $\lambda$.

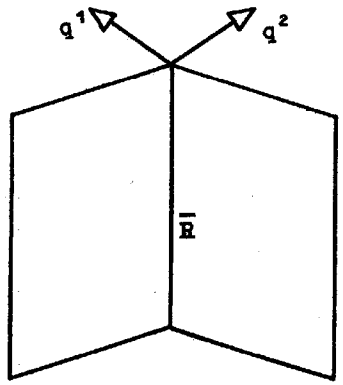


Рис. 3

Критерии окончания минимизации включают: 1) проверку относительной малости $\varphi$ или 2) проверку совпадения (с точностью до $\varepsilon$) значения $\varphi$ и наименьшего найденного ранее и сохраняемого значения $\varphi_c$.

Подпрограмма вычисления матрицы проектирования Розена (60.*MAPR*) реализует алгоритм вычисления матрицы $A_R$, с помощью которой вектор $s$, не являющийся допустимым в точке $x$, проектируется на нарушенное граничное многообразие $\bar{R}$. Приведем вывод формулы для $A_R$, следуя [4].

В подпрограмме "*DOP*", как отмечалось выше, формируется матрица $Q=(q^1,\dots,q^{k_1})$, столбцами которой являются линейно независимые строки матрицы $A$. Если $R^0$ – подпространство пространства $E^n$, порожденное векторами $q^j$, то $\bar{R}$ есть ортогональное дополнение подпространства $R^0$, так что $E^n = R^0 + \bar{R}$ (рис.3). Любой вектор из $E^n$ может быть единственным образом представлен в виде суммы векторов из $R^0$ и $\bar{R}$, например,

$$s = s^0 + \bar{s}, \quad s^0 \in R^0, \quad \bar{s} \in \bar{R}. \tag{4}$$

Так как $R^0$ порождено векторами $q^j$, то

$$s^0 = \sum_j \alpha_j q^j = Q\alpha. \tag{5}$$

Итак,

$$s = Q\alpha + \bar{s}. \tag{6}$$

Так как $\bar{s} \in \bar{R}$, то $(q^j)^T\bar{s} = 0$, $j = 1,\dots,k_1$. Следовательно, умножая (6) слева на $Q^T$, получаем:

$$Q^T s = Q^T Q\alpha.$$

Отсюда, поскольку матрица $Q^TQ$ невырожденная, находим

$$\alpha = (Q^TQ)^{-1}Q^Ts,$$

а из (5)

$$s^0 = Q(Q^TQ)^{-1}Q^Ts.$$

Наконец, из (4)

$$\bar{s} = s - s^0 = [E - Q(Q^TQ)^{-1}Q^T]\,s.$$

Таким образом,

$$A_R = E - Q(Q^TQ)^{-1}Q^T$$

Вычисление обратной матрицы производится в подпрограмме "82. *OBMAT*", работающей по способу Рутисхаузера. Блок-схема этой процедуры и пояснения к ней даны в [10] и здесь не приводятся.

Подпрограмма проектирования на границу (121. *PROGR*) умножает матрицу $A_R$ на вектор $s$, то есть проектирует его на $\bar{R}$.

Подпрограмма нормализации (75. *NORM*) нормирует вектор $s$ так, чтобы было

$$\sum_{i=1}^{n} s_i^2 = 1.$$



Подпрограмма допустимого проектирования (25. *DOPRO*). Поскольку принадлежность точки $x$ границе области $R$ определяется с точностью до $\varepsilon$, то возможна ситуация, когда проекция $s$ на $\bar{R}$ оказывается недопустимой. В этом случае приходится повторять проектирование до тех пор, пока $\bar{R}$ не станет одним и тем же для двух соседних проектирований.

Подпрограмма случайного бросания (62. *BROS*), используя датчик равномерно распределенных на [0,1] псевдослучайных чисел, производит случайное бросание точки $x$ в $n$-мерный куб с вписанной в него областью $R$ и выбирает ту точку, которая удовлетворяет неравенству $Ax \le c$.

Единственным нестандартным блоком в программе является подпрограмма вычисления функции $f(x)$ (40. *F*). Собственно нестандартными здесь являются операторы, вычисляющие функции $V(x)$ и $\varphi(x)$. Если $\pi_f = 1$, то, вычислив $f = \alpha \cdot V + \tilde{f}$, мы идем на выход. При $\pi_f = 0$ вычисляется $f = \varphi(x)$. Если при этом $\pi_{pq} = 0$ (признак того, что мы находимся на "дне" функции $\bar{\varphi}(x)$), мы идем на выход. При $\pi_{pq} = 1$ вычисляется $f = \varphi^2(x)$.

Полный текст автокодовой программы приведен на стр. 71.

## 3. Примеры

Рассмотрим два примера, использовавшиеся в качестве тестов при отладке алгоритма, которые могут служить иллюстрацией как самого процесса оптимизации, так и возможностей алгоритма.

1. Двумерный тест. Положим

$$V(x) = \frac{-x_1^2 + x_2 + 0{,}2}{0{,}76}; \quad \varphi(x) = \frac{x_1 - x_2^2 + 0{,}2}{0{,}78} - D \equiv U(x) - D.$$

Линии уровня $V(x) = const$ и $U(x) = const$ приведены на рис. 4. Допустимая область $R$, показанная на этом рисунке, описывается с помощью матриц

$$A^T = \left\| \begin{matrix} -1; & 0; & 1; & 1; & 0; & -1 \\ 0; & -1; & -1; & 0; & 1; & 1 \end{matrix} \right\|,$$

$c^T = (-0{,}35;\ -0{,}35;\ 0{,}15;\ 0{,}7;\ 0{,}7;\ 0{,}15)$.

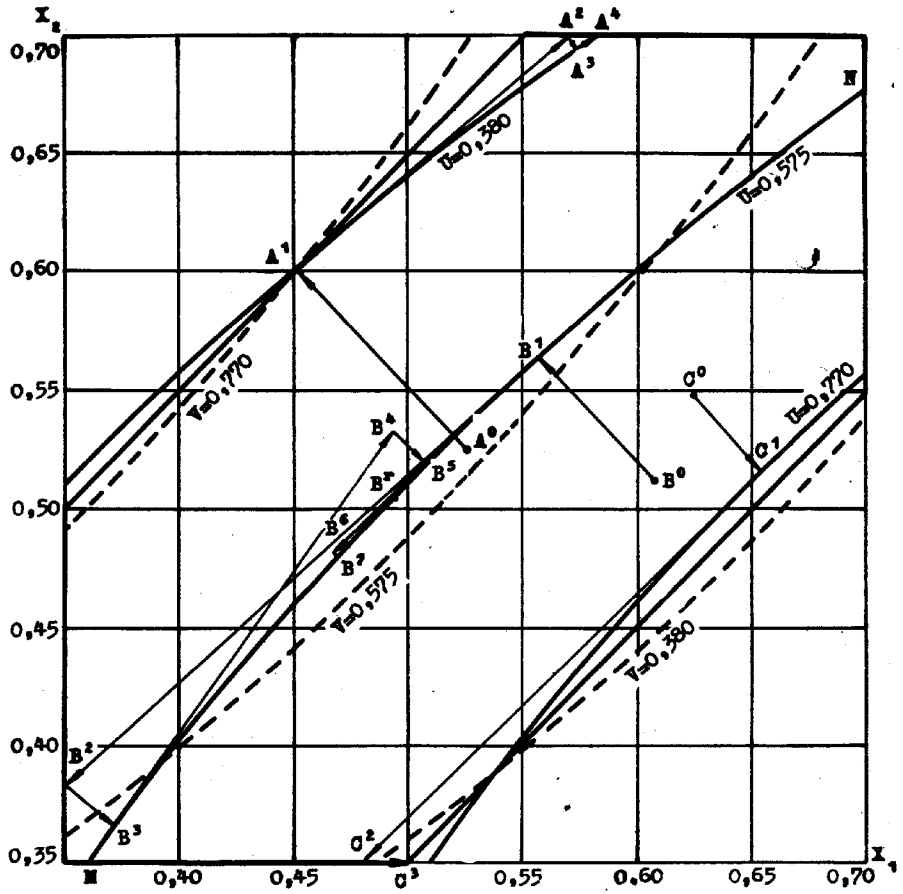


Рис. 4

Проследим на этом примере, как работает программа. (Результаты получены на ЭЦВМ "Минск-22" с 7-ю десятичными знаками; мы приводим лишь 4 значащих цифры).

а) Поиск $\min V$ на линии $D = 0{,}38$ (см. рис.4). Процесс начинается из точки

$$A^0 = (0{,}525;\ 0{,}525).$$

Спуск на $\varphi$ приводит в

$$A^1 = (0{,}4518;\ 0{,}6018).$$



Отсюда – движение по касательной в точку

$$A^2 = (0{,}57;\ 0{,}7).$$

Из нее спускаемся на $\psi$ :

$$A^3 = (0{,}573;\ 0{,}6958).$$

И, наконец, по касательной приходим в точку

$$A^4 = (0{,}5787;\ 0{,}7),$$

являющуюся решением (с точностью до $\varepsilon \sim 10^{-5}$).

б) Вычисление $\max V$ на линии $D = 0{,}575$. Исходная точка

$$B^0 = (0{,}6064;\ 0{,}5728).$$

Спуск на $\psi$ :

$$B^1 = (0{,}5559;\ 0{,}5647).$$

Движение по касательной:

$$B^2 = (0{,}35;\ 0{,}3824).$$

Снова спуск на $\psi$ :

$$B^3 = (0{,}3711;\ 0{,}3662).$$

Из этой точки начинается колебательный процесс движения к максимуму. По касательной:

$$B^4 = (0{,}4934;\ 0{,}5331).$$

Спуск:

$$B^5 = (0{,}5065;\ 0{,}5191).$$

По касательной:

$$B^6 = (0{,}4674;\ 0{,}4814).$$

Спуск:

$$B^7 = (0{,}4681;\ 0{,}4807).$$

И так далее, пока не приходим в точку максимума

$$B^* = (0{,}4921;\ 0{,}505).$$

в) Пример неудачного поиска $\max V$ на линии $D = 0{,}77$. Начиная из точки

$$C^0 = (0{,}624;\ 0{,}548),$$

спускаемся в

$$C^1 = (0{,}6524;\ 0{,}5169),$$

откуда по касательной выходим на границу:

$$C^2 = (0{,}4798;\ 0{,}35).$$

При попытке спуститься на $\varphi$ мы попадаем в угловую точку области:

$$C^3 = (0{,}5;\ 0{,}35),$$

и на этом процесс заканчивается из-за невозможности спроектировать вектор $-s$ на многообразие нулевой размерности. Выход из

подобной ситуации, как отмечалось выше, – в повторении поиска из новой случайно выбранной допустимой точки.

Обратим внимание на многоэкстремальность этой задачи. Например, для $D = 0{,}575$ мы имеем два минимума – в точках $M$ и $N$ (см. рис. 4), из них в $N$ – более глубокий.

2. Т р е х м е р н ы й  т е с т . Допустимая область $R$ описывается неравенствами (см.рис. 5):

$$x_1 \geqslant 0,\quad x_2 \geqslant 0,\quad x_3 \geqslant 0,$$
$$0{,}5x_1 + x_2 + x_3 - 1 \leqslant 0.$$

Ограничение-равенство:

$$\psi \equiv x_1^2 + x_2^2 - 0{,}25 = 0.$$

Функция $V = x_1^2 + x_2^2 + x_3^2$.

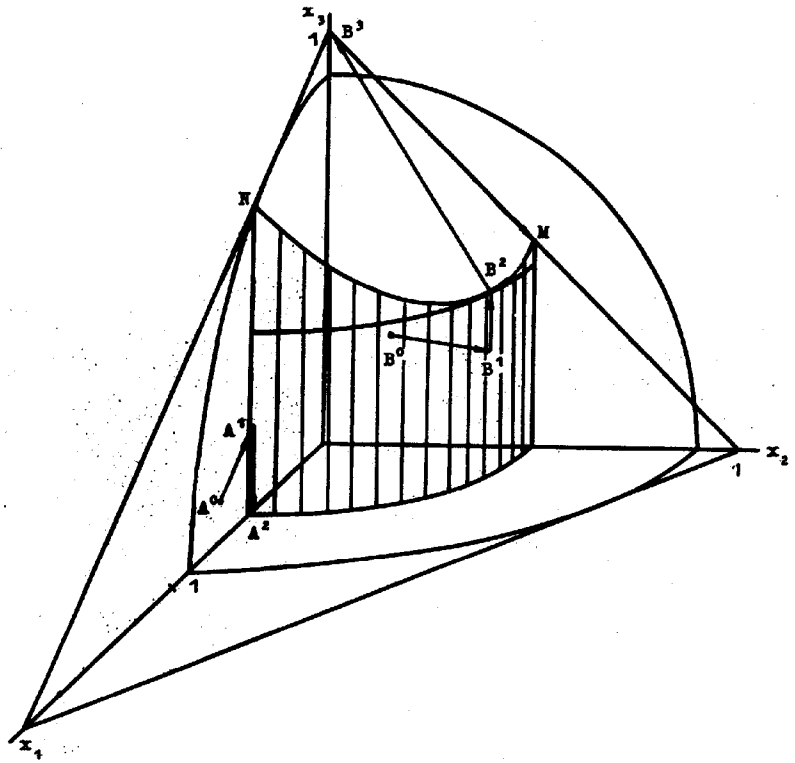


**Рис.5. Иллюстрация к примеру 2.**



Таким образом, задача состоит в отыскании на цилиндре $\psi$ (в пределах допустимой области – тетраэдра) точки, наименее и наиболее удаленной от начала координат.

а) Поиск $\min V$ из начальной точки

$$A^0 = (1{,}054;\ 0{,}1083;\ 0{,}2166).$$

Спуск на $\psi$ :

$$A^1 = (0{,}4974;\ 0{,}05108;\ 0{,}2166).$$

Отсюда – движение в касательной к цилиндру плоскости – в точку-решение:

$$A^2 = (0{,}4974;\ 0{,}05108;\ 0).$$

Очевидно, решением задачи на минимум будет целое многообразие – пересечение цилиндра с плоскостью $x_3 = 0$. Точки, лежащие на этом многообразии, можно получить, начиная поиск из различных исходных точек области $R$ .

б) Поиск $\max V$ из начальной точки

$$B^0 = (0{,}1;\ 0{,}2;\ 0{,}3).$$

Спускаемся на $\psi$ :

$$B^1 = (0{,}2237;\ 0{,}4474;\ 0{,}3).$$

Отсюда по касательной приходим в

$$B^2 = (0{,}2237;\ 0{,}4474,\ 0{,}4408).$$

Проекция градиента $V$ на границу приводит в

$$B^3 = (0;\ 0{,}0001305;\ 0{,}9998).$$

И, наконец, – спуск на $\psi$ , в точку

$$M = (0;\ 0;\ 0{,}4999),$$

доставляющую локальный максимум функции $V$ на поверхности $\psi$ .

В точку второго максимума мы попадем , если исходная случайная точка окажется в "области влияния" этого максимума. Например, из точки

$$C^0 = (1{,}035;\ 0{,}07[illegible]4;\ 0{,}1411)$$

аналогично описанному в п.б) процесс поиска приводит в точку

$$N = (0{,}4998;\ 0;\ 0{,}7500),$$

являющуюся точкой глобального максимума.

В заключение отметим, что описанная программа использовалась при построении областей рабочих сигналов для интегральной цепи логических инверторов [1].

## Литература


I. БЕЛЯЕВ Е.И.,КОРОЛЕВ В.К. Анализ интегральной цепи логических инверторов с простой корреляцией параметров."Вычислительные системы", Новосибирск, 1972, вып. 50, с. 30–43.

2. ГРОБМАН Д.М., СМИРНОВ Ю.И. Экономичное распределение нагрузок суточного графика для электростанций смешанной энергосистемы. –"Известия АН СССР, Энергетика и автоматика",М., 1959, № 4.

3. БАГРИНОВСКИЙ К.А. Об одном способе минимизации выпуклой функции на невыпуклом множестве. –Научные труды Новосибирского государственного университета, сер.экономическая, вып.8, 1966.

4. ХЕДЛИ Дж. Нелинейное и динамическое программирование. М., "Мир", 1967.

5. ЗОЙТЕНДЕЙК Г. Методы возможных направлений. М.,ИЛ,1963.

6. ЭРРОУ К., ГУРВИЦ Л., УДЗАВА Х. Исследования по линейному и нелинейному программированию, М., ИЛ, 1962.

7. КАПЛАН А.А. Численные методы решения задач выпуклого программирования. –В сб.: "Оптимальное планирование", Новосибирск, 1970, вып. 17.

8. ПРОХОРОВ В.И. и др. Основы программирования для ЭЦВМ, М., "Высшая школа", 1967.

9. БЕРЕЗИН И.С., ЖИДКОВ Н.П. Методы вычислений, М., Физматгиз, 1962, т. I.

10. КОРОЛЕВ В.К. Универсальная программа оптимизации электрических цепей. –"Вычислительные системы", Новосибирск, 1970, вып. 40, с. 82–113.



Поступила в ред.-изд.отд.

17 апреля 1973 года






```
----------
   ЛИСТ  01

01          МИН Ф=ПСИ*
02       МАС X(8),X1(8),XC(8),XC1(8),S(3),S1(8),G(8),G1(8),
03       G0(8),Q(64 8.8),Q1(64 8.8),AR(34 8.8),B(64 8.8)*
04    3.ВВОД EPS0,EPS,EPS1,DN,DD,DK,FM,XM,W0,
05       ED(64 N.N),A(192 M.N),C(24),XS(3):M,N,N1*
06       ВЫ4 :N2=N-1*
07       ВЫ4 PIPG=1*
08    1.ВЫ4 D=DG*
09   78.ВЫ4 ALF=ALF FF=FT*
10       ВЫ4 :N1N=0 N1E=0*
11       ВЫ4 FC2=-18*
12   66.ВЫ4 X/I/=XS/I/*
13       ПОВ 66 I=1 (1) N*
14       ВЫ4 :PIF=0*
15       ВЫП 40*
16       ВЫ4 FNO F*
----------
   ЛИСТ  02

01    2.ВЫ4  1/I/=0*
02       ПОВ 2 I=1 (1) N*
03       ВЫ4 PIS=1*
04   67.ВЫ4 :PIF=0*
05       ВЫП 50*
06       ЕСЛИ GS1 (=EPS TO 74*
07       ВЫП 55*
08       ЕСЛИ :K1 =0 TO 13*
09       ЕСЛИ :K1 =N TO 5*
10       ВЫП 25*
11       ЕСЛИ GS1 (EPS TO 5*
12   13.ВЫ4 G0/I/=S/I/*
13       ПОВ 13 I=1 (1) N*
14       ВЫП 23*
15       ВЫП 29*
16       ЕСЛИ MIN 10,5 TO 67*
----------
   ЛИСТ  03

01   74.ВЫ4 PIPG=0*
02       ВЫП 50*
03       ВЫ4 PIPG=1*
04       ВЫ4 :PIF=1*
05   63.ВЫ4 G/I/=S/I/*
06       ПОВ 63 I=1 (1) N*
07       ВЫП 50*
08       ВЫ4 R1=0*
09   45.ВЫ4 R1=R1+G/I/.S/I/*
10       ПОВ 45 I=1 (1) N*
11       ЕСЛИ MOD(1-MOD(R1)) (EPS TO 51*
12       ВЫ4 R1=0 R2=0 GS=0*
13   64.ВЫ4 R1=R1+G/I/.S/I/ R2=R2+G/I/'2.
14       ПОВ 64 I=1 (1) N*
15   68.ВЫ4 S/I/=S/I/.R2-G/I/.R1 GS=GS+S/I/'2*
16       ПОВ 68 I=1 (1) N*
```

```
----------
   ЛИСТ   4

01      ВЫП 75*
02      ВЫП 55*
03      ВЫ4 R=1 0*
04      ЕСЛИ :K1 =0 TO 57*
05      ЕСЛИ :K1 =N2 TO 51*
06      ВЫП 25*
07      ЕСЛИ GS1  EPS0 TO 51*
08  57.ВЫ4 R01=R01+G0/I/.S/I/*
09      ПОВ 57 I=1 (1) N*
10      ЕСЛИ MOD(1-MOD(R01)) (EPS TO 51*
11      ВЫ4 R=0*
12  58.ВЫ4 R=R+S/I/.S1/I/*
13      ПОВ 58 I=1 (1) N*
14      ВЫП 23
15      ЕСЛИ R (0 TO 53*
16      ЕСЛИ PIS )1 TO 61*
----------
   ЛИСТ  05

01      ВЫП 29*
02      ПЕР 12*
03  53.ВЫ4 PIS=PIS.2*
04  61.ВЫ4 LAM=LAM1:PIS*
05  59.ВЫ4 X/I/=LAM.S/I/+X1/I/*
06      ПОВ 59 I=1 (1) N*
07  12.ВЫ4 S1/I/=S/I/*
08      ПОВ 12 I=1 (1) N*
09      ПЕР 67*
10   5.ЕСЛИ :N1N =N1 TO 22*
11      ВЫ4 :N1N=N1N+1*
12      ВЫП 62*
13      ПЕР 2*
14  22.ЕСЛИ :N1E )0 TO 26*
15      НАП NA BPM ALA,0,0*
16      ПЕР 26*
----------
   ЛИСТ  06

01  51.ЕСЛИ MOD(F-FC2) (EPS TO 69*
02      ВЫ4 FC3=(F-FF):ALF*
03      НАП NA BPM ALA,0,FC3*
04      ВЫ4 FC2=F*
05  69.ЕСЛИ :N1E =N1 TO 26*
06      ВЫ4 :N1E=N1E+1*
07      ВЫП 62*
08      ПЕР 2*
09  26.ПОВ 78 ALA=1 (-2).FT=0 (FM).2*
10      ПОВ 1 DG=DN (DD) I=DK*
11      КОН *
12  50.ПОД GS*
13      ВЫ4 4 *
14      ВЫ4 PIG=0 GS=0 FN=F DX=EPS1*
15  16.ВЫ4 X/I/=X/I/+DX*
16      ВЫП 40*
```



```
----------
   ЛИСТ  07

01      ВЫ4 FD1=F DX=DX:2 X/I/=X/I/-DX*
02      ВЫП 40*
03      ВЫ4 FD2=F*
04 127.ВЫ4 X/I/=X/I/-DX DF=(4.FD2-3.FN-FD1):(2.DX)
05      DDF1=MOD(FD1-FN) DDF2=MOD(FD2-FN)*
06      ЕСЛИ DDF1.DDF2 (EPS TO 19*
07      ВЫ4 FD1=FD2 DX=DX:2 X/I/=X/I/+DX*
08      ВЫП 40*
09      ВЫ4 FD2=F*
10      ПЕР 127*
11  19.ВЫ4 DX=DX.2 S/I/=-DF GS=GS-DF.DF*
12      ПОВ 16 I=1 (1) N*
13      ВЫП 75*
14      ВЫХ GS*
15  55.ПОД DOP*
16      ВЫ4 :K1=0*
----------
   ЛИСТ  08

01   4.ВЫ4 R1= /I/*
02   6.ВЫ4 R1=R1-A/I,J/.X/J/*
03      ПОВ 6 J=1 (1) N*
04      ЕСЛИ R1 )EPS1 TO 8*
05      ВЫ4 R2=0*
06   7.ВЫ4 R2=R2+A/I,J/.S/J/*
07      ПОВ 7 J=1 (1) N*
08      ЕСЛИ -R2 )EPS0 TO 8*
09      ВЫ4 :K1=K1+1*
10   9.ВЫ4 Q/J,K/=A/I,J/*
11      ПОВ 9 J=1 (1) N*
12      ПОВ 9 K=K1 (1).1*
13   8.ПОВ 4 I=1 (1) M*
14      ВЫХ DOP*
15  23.ПОД LAM1*
16  73.ВЫ4 X1/I/=X/I/
----------
     СТ  09

01      ПОВ 73 I=1 (1) N*
02      ВЫ4 LAM1=1E18*
03  11.ВЫ4 R1=0*
04  10.ВЫ4 R1=R1+A/I,J/.S/J/*
05      ПОВ 10 J=1 (1) N*
06      ЕСЛИ R1 (EPS0 TO 14*
07      ВЫ4 R2=C/I/*
08  15.ВЫ4 R2=R2-A/I,J/.X/J/*
09      ПОВ 15 J=1 (1) N*
10      ЕСЛИ R2 (EPS1 TO 14*
11      ВЫ4 WAM=R2:R1*
12      ЕСЛИ WAM )=LAM1 TO 14*
13      ВЫ4 LAM1=WAM*
14  14.ПОВ 11 I=1 (1) M*
15      ВЫХ LAM1*
16  29.ПОД MIN(F)*
```

```
----------
   ЛИСТ  10

01      ЕСЛИ МО (FN) (WO TO 34*
02      ВЫ4 LO=O LA=C LL=O FO=FN FA=FN
03      FC=*18 FC1=*18 MIN=1 DLR.EPS1.LAM1*
04   70.ЕСЛИ MOD( S) (WO TO 71*
05      БИБ PROG 54(KSI)*
06      ВЫ4 LR=-KSI.FO:GS LAM=LO+LR*
07      ВЫП 39*
08      ВЫ4 L1=LAM F1=F FR=((F1-FO):LR-GS):LR*
09      ЕСЛИ FR (EPS TO 38*
10      ВЫ4 LAM=LO-GS:2:FR*
11      ВЫП 39*
12      ЕСЛИ MOD(L1-LAM) I=EPS TO 37*
13      ВЫ4 LAM=(LO+L1):2*
14      ВЫП 39*
15   37.ВЫ4 L2=LAM F2=F
16   33.ВЫ4 LSR=(LO+L1+L2):3*
----------
   ЛИСТ  11

01      ЕСЛИ LSR (EPS TO 91*
02      ВЫ4 R2=F1-FO R3=F2-FO R4=L2-LO R5=L1-LO R2=R2.R4 R3=R3.R5
03      R6=R2 R3 R7=R4.R5.(L1-L2) R8=R6:R7 R8=R8.LSR.LSR:FN*
04      ЕСЛИ R8 (EPS TO 38*
05      ВЫ4 R7=R2.R4-R3.R5 LAM=R7:2:R6+ C*
06      ВЫП 39*
07      ЕСЛИ FO  F1 TO 41*
08      ЕСЛИ F1 (=F2 TO 42*
09      ВЫ4 L1=LAM F1=F*
10      ПЕР 33*
11   41.ЕСЛИ FO (=F2 TO 42*
12      ВЫ4 LO=LAM FC=F*
13      ПЕР 33*
14   42.ВЫ4 L2=LAM F2=F*
15      ПЕР 33*
16   34.ВЫ4 MIN=C*
----------
   ЛИСТ  12

01      ПЕР 47*
02   38.ВЫ4 DL=-DLR LAM=(LL+LA):2*
03      ВЫП 39*
04      ВЫ4 FN=F LAM=LAM+DL*
05      ВЫП 39*
06      ПЕР 72*
07   71.ЕСЛИ :PIF =1 TO 47*
08      ВЫ4 LAM=LA LL=LA FN=FA DL=DLR*
09      ЕСЛИ FC )=FC1 TO 43*
10      ВЫ4 FC1=FC FC=*18*
11   48.ВЫ4 XC1/I/=XC/I/*
12      ПОВ 48  =1 (1) N*
13   43.ЕСЛИ LAM )=LAM1 TO 27*
14      ВЫ4 LAM=LAM+DL*
15      ВЫП 39*
16      ЕСЛИ F )=FN TO 44*
```



```
----------
   ЛИСТ  13

01   72.ВЫ4 LO=LAM FO=F GS=(F-FN):DL*
02      ПЕР 70*
03   44.ВЫ4 FN=F*
04      ПЕР 43*
05   27.ВЫ4 X/I/=XC1/I/*
06      ПОВ 27 I=1 (1) N*
07      ВЫ4 F=FC1*
08      ПЕР 47*
09   91.ВЫ4 X/I/=X1/I/*
10      ПОВ 91 I=1 (1) N*
11      ВЫ4 F=FN MIN=0
12   47.ВЫХ MIN(F)*
13   39.ПОД LAM=F*
14      ЕСЛИ LAM )=LL TO 30*
15      ВЫ4 LAM=LL*
16   30.ЕСЛИ LAM (=LAM1 TO 31*
----------
   ЛИСТ  14

01      ВЫ4 LAM=LAM1*
02   31.ВЫ4 X/I/=LAM.S/I/+X1/I/*
03      ПОВ 31 I=1 (1) N*
04      ВЫП 40*
05      ЕСЛИ LAM (=LA TO 35*
06      ВЫ4 LA=LAM FA=F*
07   35.ЕСЛИ MOD(F:FNO) (EPSO TO 34*
08      ЕСЛИ MOD(F-FC1:FC) (EPS TO 71*
09      ЕСЛИ F )=FC TO 32*
10      ВЫ4 LC=LAM FC=F*
11   80.ВЫ4 XC/I/=X/I/*
12      ПОВ 80 I=1 (1) N*
13   32.ВЫХ LAM=F*
14   60.ПОД МАРЗ*
15   29.ВЫ4 Q1/I,J/=Q/J,I/
16      ПОВ 26 J=1 (1) N*
----------
   ЛИСТ   5

01      ПО  29 I=1 (1) K1*
02   79.ВЫ4 B/I,J/=C*
03  122.ВЫ4 B/I,J/=B/I,J/+Q1/I,K/.B/K,J/*
04      ПОВ 122 K=1 (1) N*
05      ПОВ 79 J=1 (1) K1*
06      ПОВ 79 I=1 (1) K1*
07      ВЫП 82*
08  123.ВЫ4 AR/I,J/ C*
09  124.ВЫ4 AR/I,J/=AR/I,J/+Q/I,K/.B/K,J/*
10      ПОВ 124 K=1 (1) K1*
11      ПОВ 123 J=1 (1) K1*
12      ПОВ 123 I=1 (1) N*
13  125.ВЫ4 Q/I,J/=C*
14  126.ВЫ4 Q/I,J/=Q/I,J/+AR/I,K/.Q1/K,J/*
15      ПОВ 126 K=1 (1)  1*
16      ПОВ 125 J=1 (1)  N*
```

```
----------
   ЛИСТ  16

01      ПОВ 125 I=1 (1) N*
02   36.ВЫ4 AR/I,J/=ED/I,J/-Q/I,J/*
03      ПОВ 36 J=1 (1) N*
04      ПОВ 36 I=1 (1) N*
05      ВЫ  121*
06      ВЫХ МАРR*
07   82.ПОД ОВМАТ*
08   83.ВЫ4 :AR/4,I/=0 AR/5,I/=0*
09      ПОВ 83 I=1 (1) K1*
10   84.ВЫ4 R2=0*
11      ВЫ4 :M1=0 M4=0 M5=0*
12   85.ВЫ4 :M1=M1+1*
13   86.ЕСЛИ :M1 =AR/4,K/ TO 90*
14      ПОВ 86 K=1 (1) K1*
15      ВЫ4 :M2=0*
16   87.ВЫ4 :M2=M2+1*
----------
   ЛИСТ  17

01   88.ЕСЛИ :M2 =AR/5,L/ TO 89*
02      ПОВ 88 L=1 (1) K1*
03      ЕСЛИ MOD(B/I,J/) (=R2 TO 89*
04      ВЫ4 R2=B/I,J/*
05      ВЫ4 :M4=M1 M5=M2*
06   89.ПОВ 87 J=1 (1) K1*
07   90.ПОВ 85 I=1 (1) K1*
08      ЕСЛИ :M4 )  TO 92*
09      КОН *
10   92.ВЫ4 :AR/4,K/=M4*
11      ПОВ 92 K=M5 (1).1*
12   93.ВЫ4 :AR/5,L/=M5*
13      ПОВ 93 L=M4 (1).1*
14      ВЫ4 :M1=0*
15   94.ВЫ4 :M1=M1+1*
16      ЕСЛИ :M1 =M4 TO 95 INA4E 99*
----------
   ЛИСТ  18

01   95.ВЫ4 :M2=0*
02   96.ВЫ4 :M2=M2+1*
03      ЕСЛИ :M2 =M5 TO 97*
04      ВЫ4 AR/1,J/=-B/I,J/:R2 B/I,J/=0*
05      ПЕР 98*
06   97.ВЫ4 AR/1,J/=1:R2 B/I,J/=0*
07   98.ПОВ 96 J=1 (1) K1*
08   99.ПОВ 94 I=1 (1) K1*
09      ВЫ4 :M2=0*
10  100.ВЫ4 :M2=M2+1*
11      ЕСЛИ :M2 =M5 TO 101 :NA4E 105*
12  101.ВЫ4 :M1=0*
13  102.ВЫ4 :M1=M1+1*
14      ЕСЛИ :M1 =M4 TO 103*
15      ВЫ4 AR/2,I/=B/I,J/ B/I,J/=0*
16      ПЕР 104*
```



```
----------
   ЛИСТ  19

01  103.ВЫ4 AR/2,I/=1*
02  104.ПОВ 102 I=1 (1) K1*
03  105.ПОВ 100 J=1 (1) K1*
04  106.ВЫ4 B/I,J/=B/I,J/+AR/2,I/.AR/1,J/*
05      ПОВ 106 J=1 (1) K1*
06      ПОВ 106 I=1 (1) K1*
07      ПОВ 84 K1*
08      ВЫ4 :M1=0*
09  107.ВЫ4 :M1=M1+1 AR/3,K/=M1*
10      ПОВ 107 K=1 (1) K1*
11      ВЫ4 :M1=0*
12  108.ВЫ4 :M1=M1+1*
13      ЕСЛИ :M1 =K1 TO 113*
14      ЕСЛИ :M1 =AR/4,L/ TO 112*
15  109.ЕСЛИ :MOD(AR/4,L/-AR/3,K/) )0 TO 111*
16  110.ВЫ4 R2=B/I,L/ B/I,L/=B/I,K/ B/I,K/=R2*
----------
   ЛИСТ  20

01      ПОВ 110 I=1 (1) K1*
02      ВЫ4 :AR/3,K/=AR/3,L/*
03      ПЕР 112*
04  111.ПОВ 109 K=M1 (1) K1*
05  112.ПОВ 108 L=1 (1) K1*
06  113.ВЫ4 :M1=0*
07  114.ВЫ4 :M1=M1+1 AR/3,K/=M1*
08      ПОВ 114 K=1 (1) K1*
09      ВЫ4 :M1=0*
10  115.ВЫ4 :M1=M1+1*
11      ЕСЛИ :M1 =K1 TO 120*
12      ЕСЛИ :M1 =AR/5,L/ TO 116*
13  11 .ЕСЛИ :MOD(AR/5,L/-AR/3,K/) )0 TO 11 *
14  117.ВЫ4 R2=B/L,J/ B/L,J/=B/K,J/ B/K,J/=R2*
15      ПОВ 117 J=1 (1) K1*
16      ВЫ4 :AR/3,K/=AR/3,L/*
----------
   ЛИСТ  21

01      ПЕР 119*
02  118.ПОВ 116 K=M1 (1) K1*
03  119.ПОВ 115 L=1 (1) K1*
04  120.ВЫХ ОВМАТ*
05  121.ПОД PROGR*
06   49.ВЫ4 G1/I/=0*
07   52.ВЫ4 G1/I/=G1/I/+ R/I,J/.S/J/*
08      ПОВ 52 J=1 (1) N*
09      ПОВ 49 I=1 (1) N*
10      ВЫ4 GS=0*
11   54.ВЫ4 S/I/=G1/I/ GS=GS+G1/I/.S/I/*
12      ПОВ 54 I=1 (1) N*
13      ВЫП 75*
14      ВЫХ PROG*
15   75.ПОД NORM*
16      ВЫ4 GS=-(-GS)'(1:2) GS1=-GS*
```

```
---------
   ЛИСТ  22

01       ЕСЛИ GS1 )=EPS TO 76*
02       ВЫЧ PIG=1*
03       ПЕР 77*
04    76.ВЫЧ S/I/=S/I/: S1*
05       ПОВ 76 I=; (1) N*
06    77.ВЫХ NORM*
07    25.ПОД DOPRO*
08    18.ВЫЧ :K2=K1*
09       ВЫП 60*
10       ВЫП 55*
11       ЕСЛИ :K2 =K1 TO 24*
12       ПЕР 18*
13    24.ВЫХ DOPRO*
14    62.ПОД BROS*
15    17.БИБ PROG 54(KSI)*
16       ВЫЧ S/I/=KSI.XM*



---------
   ЛИСТ  23

01       ПОВ 17 I=1 (1) N*
02    46.ВЫЧ R1=C/I/*
03    20.ВЫЧ R1=R1-A/I,J/.S/J/*
04       ПОВ 20 J=1 (1) N*
05       ЕСЛ  R1 (0 TO 17*
06       ПОВ 46 I=1 (1) M*
07    21 ВЫЧ X/I/=S/I/*
08       ПОВ 21 I=1 (1) N*
09       ВЫХ BROS*
10    40.ПОД F*
11       ВЫЧ V=V(X)*
12       ВЫЧ F=ALF.V+FF*
13       ЕСЛИ :PIF =1 TO 56*
14       ВЫЧ F=PSI(X)*
15       ЕСЛИ PIPG (0,5 TO 56*
16       ВЫЧ F=F'2*



---------
   ЛИСТ  24

01    56.ВЫХ F*
02       НАЧ 3*
```