УДК 681.3.06:681.323(519.6)

Я.Е. РОММ, И.В. ЗАИКА

ЧИСЛЕННАЯ ОПТИМИЗАЦИЯ НА ОСНОВЕ АЛГОРИТМОВ СОРТИРОВКИ С ПРИЛОЖЕНИЕМ К ДИФФЕРЕНЦИАЛЬНЫМ И НЕЛИНЕЙНЫМ УРАВНЕНИЯМ ОБЩЕГО ВИДА

Ключевые слова: *численная оптимизация, параллельная локализация экстремумов, приближенное решение систем нелинейных уравнений, сортировка, устойчивость при возмущении параметров.*

Введение. В статье представлены конструктивные алгоритмы вычисления экстремумов функций и приближенных решений систем нелинейных уравнений, включая системы алгебраических и трансцендентных уравнений. Также рассмотрены алгоритмы вычисления экстремумов разностных решений систем обыкновенных дифференциальных уравнений (ОДУ), аналоги которых применимы к уравнениям в частных производных. Вычисления построены по единой схеме на основе сортировки последовательности с взаимно однозначным соответствием входных и выходных индексов. Схема инвариантна относительно размерности, вида задачи и сводится к допускающему распараллеливание циклическому поиску локально экстремальных элементов последовательности. При этом входная последовательность зависит от вида задачи. Для поиска экстремумов функции одной действительной переменной эта последовательность составлена из значений функции на равномерной сетке. При решении этой задачи для системы ОДУ входными являются последовательности разностных значений каждой переменной. Для однородной системы трансцендентных уравнений используется последовательность значений норм левой части на многомерной сетке. На основе схемы программируется локализация одновременно всех экстремумов входной последовательности при произвольно заданных радиусах окрестности локализации. В случае непрерывного решения задачи к локализованным значениям выполняется спуск посредством выбора наименьшего значения на равномерной сетке с итеративным сужением локализованной области. Нули функции идентифицируются как минимумы модулей значений при условии их достаточной малости. В общем случае область поиска нулей и экстремумов подразделяется на фрагменты, границы которых проверяются на наличие экстремума или нуля по схеме на основе сортировки в отличие от способа, описанного для частного случая в [1].

Идентификация экстремальных элементов числовой последовательности на основе сортировки. Используем устойчивую [2] внутреннюю сортировку одномерного массива по ключу (ниже — сортировку) с взаимно однозначным соответствием входных и выходных индексов. Сортировка строится с помощью матриц сравнений (МС) [3, 4], разновидность которых определяется ниже. Пусть по отношению $\le$ требуется упорядочить элементы массива

$$\{c_l\}_{l=1}^{n},\ n<\infty. \tag{1}$$

Массив (1) располагается горизонтально над таблицей и вертикально слева от нее. МС принимает вид

	c_1	c_2	…	c_n
c_1	α_{11}	α_{12}	…	α_{1n}
c_2	α_{21}	α_{22}	…	α_{2n}
…	…	…	…	…
c_n	α_{n1}	α_{n2}	…	α_{nn}

(2)

Здесь

$$\alpha_{ij} = \begin{cases} 1, & c_i < c_j, \\ 0, & c_i = c_j, \\ -1, & c_j < c_i, \end{cases} \tag{3}$$

где $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n$. Значения α_{ij} из (3) целесообразно заменить символами $+$, 0, $-$. Например, для массива $C = (7.3, 2.1, 8.1, 2.1, 8)$ МС (2) примет вид

	7.3	2.1	8.1	2.1	8
7.3	0	–	+	–	+
2.1	+	0	+	0	+
8.1	–	–	0	–	–
2.1	+	0	+	0	+
8	–	–	+	–	0

Чтобы определить номер в отсортированном массиве j-го элемента входного массива, достаточно подсчитать число нулей и плюсов в j-м столбце над диагональю (включая диагональный элемент) и суммировать это число с числом плюсов j-й строки ниже диагонали. Если обозначить отсортированный массив $c1$, то получим

$$c1[3] = c[1],\ c1[1] = c[2],\ c1[5] = c[3],\ c1[2] = c[4],\ c1[4] = c[5].$$

Процедура *sort*00 (Delphi) данной сортировки:

```
const n=5; a: array [1..n] of real = (7.3,2.1,8.1,2.1,8);
type vect=array [1..n] of real; vect1=array [1..n] of integer;
var c, c1: vect; e: vect1;
procedure sort00(var c,c1: vect; var e: vect1);
var i,j,k: integer;
begin
for j:=1 to n do begin k:=0;
for i:=1 to j do if c[j]>=c[i] then k:=k+1;
for i:=j+1 to n do if c[j]>c[i] then k:=k+1;
c1[k]:=c[j]; e[k]:=j;
end; end;
```

Все операции сравнения в (2) взаимно независимы и инвертируют знак относительно диагонали — сортировка максимально параллельна с оценкой временной сложности $T(N^2/2) = O(1)$ [4]. Для идентификации экстремальных элементов существенно, что при каждом индексе k элемента $c1[k]$ отсортированного массива в процедуре *sort*00 выполняется присваивание

$$c1[k] := c[j] \tag{4}$$

и запоминается индекс этого же элемента во входном массиве

$$e[k] := j, \tag{5}$$

на выходе сортировки элементы $c1[k]$ и $e[k]$ имеют одинаковый индекс k.

Локально минимальный в ε-окрестности элемент c_j последовательности (1) далее определяется согласно неравенствам

$$c_j < c_{j-\ell} \wedge c_j \le c_{j+\ell},\ \ell = 1, 2, \ldots, \varepsilon,$$

а локально максимальный элемент — согласно неравенствам

$$c_{j+\ell} < c_j \wedge c_{j-\ell} \le c_j,\ \ell = 1, 2, \ldots, \varepsilon.$$

Определения адаптированы применительно к отношению ≤, по которому элементы упорядочены данной сортировкой. При равенстве отсортированных элементов первый из них считается наименьшим, последний — наибольшим.

Пусть требуется идентифицировать все локально экстремальные элементы массива (1). Выполняется его сортировка *sort*00, в результате которой массив (1) переводится в массив

$$C1=(c1[1], c1[2],\ldots, c1[n]), \tag{6}$$

для которого выполнены соотношения (4), (5). При каждом k условие локализации минимального элемента (здесь и ниже относительно только данного определения) проверяется системой неравенств

$$|e[k]-e[k-L]|>\varepsilon,\ L=1,2,\ldots,k-1. \tag{7}$$

Выполнение одновременно всех неравенств (7) означает, что внутри ε-окрестности входного элемента массива (1) с входным индексом $e[k]$ нет элемента из массива (6), меньшего по значению, чем элемент $c1[k]$. Проверку выполняет оператор локализации минимума:

```
k:=1; while k<=n do begin for L:=1 to k-1 do if abs(e[k-L]-e[k])<=eps0 then goto 22; writeln(' ',c1[k],' ',e[k]);
22: k:=k+1 end;
```

В этом фрагменте n — число элементов из (1), а $eps0=\varepsilon$. При условии, что $eps0$ меньше половины расстояния между ближайшими локально минимальными элементами, идентифицируются одновременно все локально минимальные элементы в окрестностях радиуса $eps0$.

Пример 1. Все локально минимальные элементы последовательности $C=(7.3, 2.1, 8.1, 2.1, 8)$ с радиусами окрестностей локализации $eps0=1$ и $eps0=2$ идентифицирует следующая программа:

```
Program extrim;
{$APPTYPE CONSOLE}
uses SysUtils;
label 22, 24;
const n=5; eps0=1; {eps0=2;} a: array [1..n] of real = (7.3,2.1,8.1,2.1,8);
type vect=array [1..n] of real; vect1=array [1..n] of integer;
  var i,k,l: integer; c, c1: vect; e: vect1;
procedure sort00 (var c,c1: vect; var e: vect1);
```

Описание процедуры *sort* 00

```
begin
for i:=1 to n do c[i]:=a[i];
sort00(c,c1,e); for i:=1 to n do write (' ',c[i]:2:4,' '); writeln;
k:=1; while k <= n do begin
for L:= 1 to k-1 do
if abs(e[k]- e[k-L])<= eps0 then goto 22;
writeln (' min=',c1[k]:2:4,' ',e[k],' ');writeln;
22: k:=k+1 end; readln;
end.
```

Результат работы программы при $eps0=1$:

Минимум	Номер минимума во входном массиве
Min=2.10000000000000E+0000	2
Min=2.10000000000000E+0000	4

При $eps0=2$ получим

Min=2.1000000000000E+0000	2

Идентификацию всех локально максимальных элементов последовательности можно произвести по этой же программе, добавив оператор локализации максимума с обратным направлением просмотра элементов отсортированного массива

$$|e[k]-e[k+L]|>\varepsilon,\ L=1,2,\ldots,n-k, \tag{8}$$

в программной форме:

```
k:=1; while k<=n do begin for L:=1 to n-k do if abs(e[k+L]-e[k])<= eps0 then goto 24; writeln ('  ',c1[k],' ',e[k]);
24:  k:=k+1  end;
```

Результат работы модифицированной программы для $eps0=1$:

Максимум	Номер максимума во входном массиве
Max=7.300000000000E+0000	1
Max=8.100000000000E+0000	3
Max=8.000000000000E+0000	5

При $eps0=2$ получим

Max=8.100000000000E+0000	3

Оператор локализации идентифицирует все экстремальные элементы последовательности по построению после однократной сортировки со свойством устойчивости, опираясь на взаимно однозначное соответствие входных и выходных индексов сортируемых элементов. Оператор максимально распараллеливается по всем индексам L и k с оценкой $T(n^2)=O(1)$. Идентификация экстремумов выполняется с помощью обработки только этих индексов, поэтому описываемый способ не зависит от типа последовательности и исключает накопление погрешности.

Вычисление экстремумов функций одной переменной. Пусть последовательность вида (1) образована дискретизацией функции одной действительной переменной. Тогда локализованные на основе сортировки значения являются взаимно отделенными приближениями экстремумов. С целью уточнения их значений к ним можно выполнить спуск. Пусть для функции

$$y=f(x) \tag{9}$$

рассматриваемого вида на произвольном отрезке $[x^{(0)}, x^{(N)}]$ в области ее определения требуется найти все локальные экстремумы при произвольно заданном, но достаточно малом радиусе окрестности локализации. Строится равномерная сетка $x_l=x^{(0)}+Lh,\ L=0,1,\ldots,N,\ h=\dfrac{x^{(N)}-x^{(0)}}{N}$, в узлах которой значения функции интерпретируются как элементы массива

$$c[i]=f(x_{i-1}),\ i=1,2,\ldots,N. \tag{10}$$

Массив (10) поступает на вход сортировки $sort00$. К выходу процедуры $sort00$, как и в программе *extrim*, подсоединяется циклический оператор проверки (7), в котором с учетом шага сетки модифицируется правая часть:

$$|e[k]-e[k-L]|>\varepsilon/h,\ L=1,2,\ldots,k-1. \tag{11}$$

Идентификация локального минимума (в виде приближения к его числовому значению) выполнится, поскольку неравенства (11) эквивалентны неравенствам

$$|e[k]\times h-e[k-L]\times h|>\varepsilon,\ L=1,2,\ldots,k-1.$$

Последнее означает, что на равномерной сетке расстояние между текущим узлом $x^{(0)}+e[k]\times h$ и любым другим ее узлом $x^{(0)}+e[k-L]\times h$ больше радиуса ε для тех значений функции, которые в отсортированном массиве предшествуют $c[k]=f(x_{k-1})$. Таким образом, система неравенств (11) соответствует наименьшему в смысле отношения порядка $\le$ значению $f(x_{k-1})$ в окрестности радиуса ε.

После локализации выполняется спуск к минимальному значению внутри окрестности локализованной точки путем выбора наименьшего значения в сужаемой окрестности с фиксированным числом элементов равномерной сетки (ниже в программе это число равно mm). Сужение продолжается до тех пор, пока не будет достигнута требуемая точность приближения. Спуск представлен программным фрагментом (сужение радиуса окрестности с коэффициентом 1.2, наименьшее значение выбирается процедурой $\min 1$ с указанием индекса минимума *ee*):

```
xk:=x0+e[k]*h; eps1:=eps0; xk0:=xk-eps1; xk1:=xk+eps1; h0:=2*eps1/mm; while eps1 > eps do
begin x:=xk0; min1(x,ee); eps1:=eps1/1.2; xk0:=xk0+ee*h0-eps1; xk1:=xk0+ee*h0+eps1; h0:=2*eps1/mm end;
```

Нули функции локализуются как минимумы модулей $c[i]=|f(x_{i-1})|$, $i=1,2,\ldots,N$, с проверкой малости (при поиске нулей многочлена в силу принципа максимума модуля [5] проверка малости излишня).

Идентификацию всех локальных максимумов можно выполнить на основе циклической проверки неравенств (8), где по аналогии с (11) правую часть следует заменить выражением ε / h. Процедура $\min 1$ заменяется процедурой $\max 1$, аналогично спуску осуществляющей приближение к наибольшему значению внутри окрестности локализованной точки.

Если отрезок $[X_{00}, X_{NN}]$ поиска экстремумов шире исходного (справа), то в конце блока инструкций происходит выполнение циклического смещения отрезка $[x^{(0)}, x^{(N)}]$ на его длину до тех пор, пока не будет пройден весь априори заданный отрезок $[X_{00}, X_{NN}]$. При этом дополнительно проверяются на экстремумы границы смещаемого отрезка. Сортировка $sort\,00$ с некоторым формальным видоизменением и оператор локализации экстремумов на каждом шаге смещения применяются к последовательности значений функции в заключительных узлах предыдущего отрезка и начальных узлах текущего отрезка. При наличии экстремума на границе смещаемого отрезка к нему выполняется спуск путем выбора наименьшего значения в сужаемой вокруг его локализованного значения окрестности. В результате все экстремумы и нули идентифицируются по значению и по индексу как внутри, так и на границах смещаемого отрезка инвариантно относительно размера априори заданного отрезка $[X_{00}, X_{NN}]$. Это позволяет выполнять идентификацию экстремумов параллельно по всем непересекающимся отрезкам, на которые разбивается полный отрезок поиска, затем параллельно «склеить» результаты по всем границам составляющих отрезков. С точностью до формальных оговорок инвариантность схемы сохраняется относительно вида функции, характера экстремумов и априори заданной границы погрешности вычисления. Эксперимент показывает устойчивость локализации и высокую точность вычисления одновременно всех экстремумов и нулей функции на произвольном отрезке из области ее определения [6–9].

Ниже дана схема идентификации экстремумов функции двух переменных. Она строится аналогично схеме для одной переменной, но имеет принципиальные изменения.

Локализация и вычисление экстремумов функции двух действительных переменных. Для приближенного вычисления всех экстремумов действительной функции

$$z = f(x, y) \tag{12}$$

от двух действительных переменных вначале задаются текущие отрезки $[x^{(0)}, x^{(N)}]$ и $[y^{(0)}, y^{(M)}]$, образующие прямоугольники внутри области ее определения. Внутри прямоугольников строится равномерная сетка

$$h=|x^{(N)}-x^{(0)}|/N,\ x_\ell = x^{(0)}+\ell h,\ \ell=0,1,\ldots,N,\ y_\ell = y^{(0)}+\ell h,\ \ell=0,1,\ldots,M.$$

Для каждого значения $j = 0, 1, \ldots, N$ выполняется проход в направлении оси OY вдоль j-го столбца прямоугольной сетки, во время которого определяется минимальное по элементам строк значение функции (12) в столбце

$$c[j] = \min_{\substack{j=\text{const} \\ 1 \le i \le M}} f(x_j, y_i).$$

Этот минимум заносится на вход сортировки как j-й элемент одномерного массива. Ввиду квадратичного роста обрабатываемой информации для ускорения последовательной реализации метода используется устойчивая сортировка слиянием по МС [1, 3] с взаимно однозначным соответствием входных и выходных индексов, которая в последовательном варианте имеет временную сложность $T(1) = O(N \log_2 N)$ и вместе с тем эффективно распараллеливается: $T(N^2 / 4) = O(\log_2 N)$. Ниже процедура *sort* этой сортировки без изменений заимствуется из [1].

К выходу процедуры *sort* подсоединяется оператор локализации минимума. Для текущего узла, определяемого индексом, записанным в элемент массива $e[k]$, оператор идентифицирует каждый узел двумерной сетки, в проекции $eps\,0$-окрестности которого на ось OX отсутствуют узлы, доставляющие значения элементов входной последовательности, которые предшествуют $c[e[k]]$ в отсортированном массиве. Значение локализованной абсциссы минимума $xk := x0 + e[k] * h$ рассматривается как привязка к координатам двумерного минимума. Абсцисса xk фиксируется, и в ней по аналогичной одномерной схеме локализуется ордината yk двумерного минимума. Точка (xk, yk) рассматривается как приближение к искомым координатам минимума функции (12). Выполняется спуск к наименьшему значению в окрестности локализованной точки. Он осуществляется выбором наименьшего значения в сужаемой окрестности приближения с фиксированным числом элементов в сетке (причем, по обеим переменным) пока не достигается требуемая точность. С целью повышения точности спуск повторяется дважды [1, 6].

Аналогичное применение оператора локализации максимума обеспечивает локализацию точки максимума. Приближение к наибольшему значению в окрестности локализованной точки осуществляется выбором наибольшего значения в итеративно сужаемой окрестности.

Для исследования всей области поиска экстремумов функции (12) выполняется циклическое смещение отрезков $[x^{(0)}, x^{(N)}]$ и $[y^{(0)}, y^{(M)}]$ на их длину вдоль строк и столбцов сетки до тех пор, пока не будет пройдена вся априори заданная область. Дополнительная проверка наличия экстремумов в окрестностях границ смещаемых промежутков выполняется по схеме, аналогичной схеме для одномерного случая, — используются элементы массива, образуемого значениями функции слева и справа от левой границы каждого смещаемого отрезка $[x^{(0)}, x^{(N)}]$ и $[y^{(0)}, y^{(M)}]$. Как и для одномерного случая, это позволяет выполнять идентификацию экстремумов параллельно по всем прямоугольникам рассматриваемого вида, которые при объединении покрывают область поиска, затем параллельно «склеить» результаты по всем границам составляющих прямоугольников.

Для вычисления нулей идентифицируются минимумы модуля функции с оценкой малости найденных значений [1, 6].

Таким образом, начальная схема применима для функции двух действительных переменных. С ее помощью локализуются и вычисляются все экстремумы произвольной функции данного вида в произвольно фиксированной области на плоскости, входящей в область определения функции. Радиус окрестности локализации задается также произвольно, но фиксируется с учетом ограничения, согласно которому он должен быть меньше половины расстояния между ближайшими сосед-

ними экстремумами искомого вида (максимумами либо минимумами). Как и в одномерном случае, ограничение вытекает из построения схемы. Экспериментальное соотношение между радиусом $eps0$ окрестности локализации и шагом сетки h может заимствоваться из [1], где оно используется для вычисления комплексных нулей многочлена.

Пример 2. Все локальные минимумы функции $z = -e^{-(\sqrt{x^2+y^2}-2)^2} - e^{-(\sqrt{x^2+y^2}-3)^2} + 0.1\sin(xy)$ в квадрате $x \in [-5,5]$, $y \in [-5,5]$ с радиусом окрестности локализации $eps0 = 1$ идентифицирует следующая программа:

```
Program funcXYmin;
{$APPTYPE CONSOLE}
uses SysUtils;
label 22, 23;
const eps=1.1e-444; eps0=1; h=eps0/67; n00=400; x00=-5; x11=5; y00=-5; y11=5;
nn=n00+round(n00/2)+1; mm=64;
type vect1=array [1..2*nn+nn] of extended;
   vect2=array [1..2*nn+nn] of longint;
var i,ii,j,k,k1,r,ee,ee1,tty,nn0,k0k,k0k0x,k0k0y,L : longint; c,a1: vect1; e,e3, e33: vect2;
    x,x0,x1,xk,xk0,xk1,hx,hy,min,eps1,eps11,eps12,eps13: extended;
    y,y0,y1,yk,yk0,yk1, ykk0,aak,bbk,hh,p: extended;
   n00n: longint; a01x,a01y,c01x,c01y: vect1; e01x,e01y: vect2;
      {задание функции}
function func (x,y:extended):extended;
begin func:= -exp(-sqr((sqrt(x*x+y*y)-2)))-exp(-sqr(sqrt(x*x+y*y)-3))+0.1*sin(x*y); end;
procedure minx (var x,y, min:extended; var ee:integer);
begin min:=func(x,y); ee:=0; for i:=1 to mm do begin x:=xk0+i*hx;
IF min > func(x,y) then begin min:=func(x,y);ee:=i end; end; end;
procedure miny (var x,y, min:extended; var ee1:integer);
begin min:=func(x,y); ee1:=0; for i:=1 to tty do begin y:=ykk0+i*hy;
IF min > func(x,y) then begin min:=func(x,y); ee1:=I; end; end; end;
      {процедура сортировки слиянием}
procedure sort(var nn0:longint; var c: vect1; var e: vect2);
```

Описание процедуры *sort*

```
    {процедура сортировки подсчетом}
procedure sort00 (var n00n: longint; var a01,c01: vect1; var e01: vect2);
```

Описание процедуры *sort* 00; n00n — число сортируемых элементов, a01 — входной, c01 — выходной массивы; e01 — массив выходных индексов

```
begin
x0:=x00; x1:=x11; y0:=y00; y1:=y11; nn0:=n00; hh:=nn0*h; n00n:=4;
while x0 <= x11 do begin while y0 <= y11 do begin
k0k0x:=1; for r:=1 to nn0 do begin
x:=x0+r*h; ykk0:=y0; y:=y0; tty:=n00;hy:=h;
miny (x,y, min,ee1); a1[r]:= min; if (r > nn0-n00n div 2) and (k0k0x <= n00n div 2) then
begin a01x[k0k0x]:= min; k0k0x:=k0k0x+1; end; end;
sort(nn0, a1, e3); k:=1; while k <= nn0 do
begin for r := 1 to k-1 do IF abs(e3[k]-e3[k-r]) <= eps0/h then goto 23;
if (e3[k] <= 1) or( e3[k] >= nn0-1) then begin
sort00(n00n,a01x,c01x,e01x); k0k := 1; while k0k<= n00n do begin
for L := 1 to k0k-1 do IF abs(e01x[k0k - L] - e01x[k0k]) <= eps0/h then goto 23;
k0k:=k0k+1; end; end;
xk:= x0+e3[K]*h;
k0k0y:=1; for r:=1 to nn0 do begin y:=y0+r*h; a1[r]:=func(xk,y);
if (r > nn0-n00n div 2) and (k0k0y <= n00n div 2) then begin a01y[k0k0y]:=func(xk,y);
k0k0y:=k0k0y+1; end; end;
sort( nn0, a1, e33); k1:=1; while k1 <= nn0 do begin for r := 1 to k1-1 do
IF abs(e33[k1]-e33[k1-r]) <= eps0/h then goto 22;
if (e33[k1] <= 1) or ( e33[k1] >= nn0-1) then begin
sort00(n00n,a01y,c01y,e01y); k0k := 1; while k0k <= n00n do begin
for L := 1 to k0k-1 do IF abs(e01y[k0k - L] - e01y[k0k]) <= eps0/h then goto 22;
k0k:=k0k+1; end; end;
yk:= y0+e33[K1]*h;
```

```
{спуск к наименьшему значению функции}
eps1:=eps0; eps11:=eps0; xk0:=xk-eps1; xk1:=xk+eps1; hx:=abs(2*eps1)/mm; y:=yk;
spuskx(eps1,xk0,xk1,hx,y);
yk0:=yk-eps11; yk1:=yk+eps11; hy:=abs(2*eps11)/mm; x:=xk0+ee*hx+eps1;
spusky( eps11,yk0,yk1,hy,x); eps12:=eps0/1.2;
xk0:=x-eps12; xk1:=x+eps12; hx:=abs(2*eps12)/mm; y:=yk0+ee1*hy+eps11;
spuskx( eps12, xk0,xk1,hx,y); eps13:=eps0/1.2;
yk0:=yk0+ee1*hy-eps13; yk1:=yk0+2*eps13; hy:=abs(2*eps13)/mm;
x:=xk0+ee*hx+eps12; spusky( eps13,yk0,yk1,hy,x);
writeln (',x:30:5' '); writeln (',yk0:30:5' ',func(x,yk0));writeln;
22: k1:=k1+1 end;
23: k:=k+1 end;
y0:=y0+hh;ii:=1; if (k0k0y <= n00n) then begin y:=y0+ii*h; a01y[k0k0y]:=func(x,y);
ii:=ii+1; k0k0y:=k0k0y+1; end; end; k0k0y:=1; x0:=x0+hh;
for r:=1 to n00n do begin x:=x0+r*h; ykk0:=y0; y:=y0; tty:=n00;hy:=h;
miny (x,y,min,ee1);
a01x[k0k0x]:=min; k0k0x:=k0k0x+1; end; k0k0x:=1; y0:=y00; end; readln; end.
```

Результаты работы программы представлены в табл 1.

Таблица 1

Значения аргументов		Значения минимумов функции z
x	y	
– 2.41477	0.65000	– 1.6576010915
– 2.07453	– 1.71591	– 1.5695232027
……	……	……
1.87936	1.88013	– 1.5763022602
1.87957	5.84920	– 0.1000509240

Для вычисления нулей достаточно в подпрограмме *func* сформировать значения модуля функции:

```
function func (x,y:extended):extended;
begin func:=abs(-exp(-sqr((sqrt(x*x+y*y)-2)))
   -exp(-sqr(sqrt(x*x+y*y)-3))+0.1*sin(x*y)); end;
```

Результаты работы программы представлены в табл 2. Рассмотренным способом находятся действительная и мнимая части комплексных нулей многочлена [1], квадрат модуля которого можно определять умножением на комплексно-сопряженное значение многочлена, вычисляемое в арифметике комплексных чисел [7–9].

Таблица 2

Значения аргументов		Значения нулей функции z
x	y	
– 4.9552450E+0000	– 4.4776119E-0002	0.00000000000000E+0000
– 4.95798840E+0000	– 2.5373134E+0000	3.44107134822060E-0020
– 4.1243246E+0000	– 3.8059701E+0000	4.00223067577657E-0020
……	……	……
6.7463688E+0000	4.6567164E+0000	7.25056330178458E-0020
4.8166294E+0000	6.5223880E+0000	3.23831094870378E-0020

Существенным является качество одновременной идентификации всех нулей и экстремумов функций двух переменных в окрестностях произвольного (достаточно малого) радиуса без априорного указания области отдельного нуля или экстремума, области всего их множества и начального приближения. Математические методы, как правило [10, 11], используют данные параметры. В предложенной схеме все нули и экстремумы идентифицируются по значению и местоположению. Согласно численным экспериментам [6, 9, 12–14] устойчивость и точность приближения аналогичны достигаемым в одномерном случае.

Локализация экстремумов функции нескольких действительных переменных. Относительно действительной функции n действительных переменных $\psi = f(x_1, x_2, \ldots, x_n)$ предполагается, что повторные пределы совпадают с пределами одновременно по всем переменным в каждой точке экстремума. Изложен-

ная схема обобщается на случай данной функции следующим образом. Вводится многомерная равномерная сетка с узлами переменных, расположенными вдоль прямых, параллельных соответственным осям декартовых координат. Вначале выбирается переменная x_1 и для каждого ее узлового значения на прямой, параллельной соответственной этой переменной оси, находится минимум (максимум) функции ψ, который берется по $n-1$ другим переменным. Все такие минимумы (максимумы) поступают на вход сортировки в виде последовательности, оператор локализации идентифицирует среди ее элементов все локально минимальные (максимальные) элементы — по значению и по индексу. Каждое локализованное значение x_1 фиксируется в качестве привязки к искомому локальному экстремуму функции n переменных. При каждом значении фиксированной таким способом координаты для оставшихся $n-1$ других координат возобновляется процесс, полностью идентичный изложенному выше для n координат. Конкретно для каждого фиксированного значения локализованной координаты x_1 выбираются минимумы (максимумы) по $n-2$ переменным при каждом узловом значении переменной x_2, взятом на прямой равномерной сетки, параллельной соответственной этой переменной оси. Минимумы образуют новую последовательность, которая поступает на вход сортировки, с помощью оператора локализации идентифицируются все значения приближений второй координаты искомых локальных экстремумов. Далее, при последовательном фиксировании каждого сочетания пары уже локализованных координат процесс возобновляется для оставшихся $n-3$ координат и продолжается до тех пор, пока в заданном порядке не будут локализованы все независимые переменные функции ψ. Как и в случае двух переменных, схема позволяет идентифицировать все локальные экстремумы в окрестности произвольного радиуса, который меньше половины наименьшего расстояния между экстремумами, без априорного определения области локализации и начальных приближений. К локализованным приближениям можно выполнить спуск, аналогичный спуску для двух переменных. Нули идентифицируются как минимумы модуля функции с оценкой малости значения. Программа, реализующая схему для функции трех переменных, приводится в работе [15], для четырех и пяти переменных — в [6].

Если область локализации имеет большой размер, то необходимо поочередно смещать отрезки $[x_1^{(0)}, x_1^{(N)}]$, $[x_2^{(0)}, x_2^{(M)}], \ldots, [x_n^{(0)}, x_n^{(R)}]$ на их длину до тех пор, пока вся область не окажется пройденной. Проверка экстремумов в окрестностях границ смещаемых отрезков должна выполняться по аналогии с двумерной схемой: используются элементы массива узловых значений по обе стороны от левой границы каждого из смещаемых отрезков. Отсюда вытекает возможность идентификации экстремумов параллельно по всем элементам рассматриваемого вида, которые при объединении покрывают область поиска, затем параллельно «склеить» результаты по всем границам элементов.

Численное решение нелинейных систем уравнений общего вида. Применим рассматриваемую схему к приближенному решению систем нелинейных уравнений. Пусть исходная система преобразована к виду

$$f_i(x_1, x_2, \ldots, x_n) = 0, \; i = \overline{1, n}. \tag{13}$$

Здесь левые части являются действительными функциями от n действительных переменных. Совокупность функций левой части (13) образует n-мерную вектор-функцию аргументов $x_1, x_2, \ldots, x_n$. Требуется приближенно найти все решения системы (13) в многомерном параллелепипеде $[x_1^{(0)}, x_1^{(L_1)}]$,

$[x_2^{(0)}, x_2^{(L_2)}]$, …, $[x_n^{(0)}, x_n^{(L_n)}]$, входящем в область определения данной вектор-функции. Строится многомерная сетка $x_{j\ell_j} = x_j^{(0)} + \ell_j h_j$, $h_j = \dfrac{x_j^{(L_j)} - x_j^{(0)}}{L_j}$, $\ell_j = 0, 1, \ldots, L_j$, $j = \overline{1, n}$. Во всех ее узлах вычисляются значения канонической нормы (для определенности $||Z|| = \sum_{i=1}^{n} |z_i|$) рассматриваемой вектор-функции, которые выражаются элементами n-мерного массива

$$c_{j\ell_j} = |f_1(x_{j\ell_j})| + |f_2(x_{j\ell_j})| + \ldots + |f_n(x_{j\ell_j})|, \tag{14}$$

где индексы соответствуют узлам многомерной сетки. Левая часть (14) интерпретируется как дискретизированная функция n переменных, ее минимумы можно идентифицировать программно по изложенной схеме. При этом все нули нормы функции из левой части (13) идентифицируются среди минимумов по заданной границе $\tilde{\varepsilon}$ малости: $0 \le \min\limits_{j\ell_j} c_{j\ell_j} \le \tilde{\varepsilon}$.

Пример 3. Приближенное решение системы

$$\begin{cases} x^2 + y^2 - z = 0, \\ x^2 + y^2 - z^2 = 0, \\ \ln x - \sqrt{y} + 0.8 = 0 \end{cases}$$

в параллелепипеде $x \in [0.1, 1]$, $y \in [0, 1]$, $z \in [0, 1]$ вычисляется по следующей программе:

```
Program UravnNelin;
{$APPTYPE CONSOLE}
Uses SysUtils;
label 21,22,23;
const eps0=0.1; h=eps0/23; x00=0.1; x11=1; y00=0; y11=1; z00=0; z11=1;
n00=trunc(4/h); nn=n00+round(n00/2)+1;
type vect1=array [1..2*nn+nn] of extended;
vect2=array [1..2*nn+nn] of longint;
var i,j,k,k1,r,rx,ry,rz,nn0,kx,ky,kz : longint;
c,a1,az: vect1; e, ex, ey,ez: vect2; x,x0,x1,xk,minx,minz: extended;
y,y0,y1,yk,z0,z1,z,zk: extended;
function f1 (x,y,z:extended):extended;
begin f1:= x*x+y*y-z ; end;
function f2 (x,y,z:extended):extended;
begin f2:= x*x+y*y-z*z; end;
function f3 (x,y,z:extended):extended;
begin f3:= ln(x)-sqrt(y)+0.8; end;
function func (x,y,z:extended):extended;
var p: extended; i1: integer;
begin func:= abs(f1(x,y,z))+abs(f2(x,y,z))+abs(f3(x,y,z)); end;
procedure minyz (var x,y,z,minx:extended);
begin minx:=func(x,y,z);
for i:=1 to trunc((y11-y00)/h) do
for j:=1 to trunc((z11-z00)/h) do
begin y:=y0+i*h; z:=z0+j*h;
IF minx > func(x,y,z) then minx:=func(x,y,z) end end;
procedure minyy (var x,y,z,minz:extended);
begin minz:=func(x,y,z); for i:=1 to trunc((y11-y00)/h) do
begin y:=y0+i*h; IF minz > func(x,y,z) then minz:=func(x,y,z) end end;
procedure sort(var nn0:longint; var c: vect1; var e: vect2);
```

Описание процедуры *sort*

```
begin x0:=x00; x1:=x11; y0:=y00; y1:=y11;z0:=z00; z1:=z11; nn0:=n00{-3};
for rx:=1 to nn0 do begin x:=x0+rx*h; minyz (x,y,z,minx); a1[rx]:=minx; end;
```

```
sort( nn0, a1, ex); kx:=1; while kx <= nn0 do begin
for rx := 1 to kx-1 do IF abs( ex[kx]-ex[kx-rx] ) <= eps0/h then goto 21;
xk:= x0+ex[kx]*h; for rz:=1 to nn0 do begin
z:=z0+rz*h; x:=xk; minyy (x,y,z,minz); az[rz]:=minz; end;
sort( nn0, az, ez); kz:=1; while kz <= nn0 do begin
for rz := 1 to kz-1 do IF abs(ez[kz]-ez[kz-rz]) <= eps0/h then goto 22; zk:= z0+ez[kz]*h;
for ry:=1 to nn0 do begin y:=y0+ry*h; a1[ry]:=func(xk,y,zk) end;
sort( nn0, a1, ey); ky:=1; while ky <= nn0 do begin
for ry := 1 to ky-1 do IF abs(ey[ky]-ey[ky-ry]) <= eps0/h then goto 23;
yk:= y0+ey[ky]*h; writeln (' ', xk:30:4,' '); writeln (' ', yk:30:4,' ');
writeln (' ', zk:30:4,' ', f1(xk,yk,zk):2:4,' ',f2(xk,yk,zk):2:4,' ',f3(xk,yk,zk):2:4); writeln;
23: ky:=ky+1; end;
22: kz:=kz+1; end;
21: kx:=kx+1; end; writeln('end'); readln; end.
```

В результате получим значения, представленные в табл. 3. Для нелинейных систем с неединственным решением схема позволяет программно локализовать все минимумы нормы (14) и среди них выделить наиболее близкие к нулю. Их локализация в окрестности наперед заданного радиуса означает приближение к решению по норме с точностью до значения радиуса. Например, для системы

Таблица 3

Значения аргументов			Значения компонент
x	y	z	
0.8870	0.4609	1.0000	$f1 = -0.0009$ $f2 = -0.0009$ $f3 = 0.0012$

$$\begin{cases} x^3 + y^3 - 8xy = 0, \\ x\ln(y) - y\ln(x) = 0, \end{cases}$$

которая имеет три решения, предложенная схема дает семь значений минимумов нормы, из них три наименьшие, достаточно близкие к нулю (подчеркнуты), приближают искомые решения (табл. 4).

Таблица 4

Значения аргументов		Значения компонент $\frac{f1}{f2}$
x	y	
1.94007	3.67910	$\frac{-0.0000000000}{0.0890364151}$
2.07596	**3.77313**	$\frac{\mathbf{-0.0000000000}}{\mathbf{0.0006942907}}$
2.49393	4.01045	$\frac{-0.0000000000}{-0.2011612746}$
3.67927	1.94030	$\frac{0.0000000000}{-0.0888810927}$
3.77324	**2.07612**	$\frac{\mathbf{0.0000000000}}{\mathbf{-0.0005957791}}$
4.01049	2.49403	$\frac{0.0000000000}{0.2011954085}$
4.00000	**4.00000**	$\frac{\mathbf{-0.0000000000}}{\mathbf{0.0000000000}}$

Схема без изменения может применяться для приближенных решений систем линейных и нелинейных алгебраических, а также трансцендентных уравнений [6], по способу компьютеризации отличаясь от аналогов из [10, 16]. Такой метод на входе использует значения функции, которые с учетом общности ограничений могут быть получены в результате решения некоторой другой задачи в реальном времени. Существенным ограничением при этом является требование достаточной малости шага дискретизации. Отличительным качеством (как и в двумерном случае) является отсутствие начальных приближений, а также отсутствие области локализации экстремумов, используемых в математических методах [11, 12, 16]. Численный эксперимент показывает применимость схемы в случае сложных рельефов вектор-функций из левой части (13) [6], включая разрывы первого рода.

Идентификация экстремумов и нулей разностных решений систем ОДУ. Рассмотрим задачу Коши для уравнения

$$\frac{dy}{dt} = f(t, y),\ y(t_0) = y_0, \tag{15}$$

относительно которого предполагается выполнение всех условий существования и единственности. Кроме того, предполагается сходимость метода Эйлера с равномерным шагом к решению задачи (15) на отрезке $[t_0, T]$:

$$y_{i+1} = y_i + f(t_i, y_i)h,\ t_i = t_0 + ih,\ i = 0, 1, \ldots, N;\ h = \frac{T - t_0}{N};\ \lim_{N\to\infty} y_i = y. \tag{16}$$

Разностные значения из (16) рассматриваются как элементы массива

$$c[i] = y(t_{i-1}),\ i = 1, 2, \ldots, N, \tag{17}$$

который сортируется с присоединением оператора локализации экстремумов. Обработка массива (17) практически не отличается от обработки массива (10) для функции одной переменной, за исключением того, что в случае разностной схемы нельзя выполнить спуск. При данном способе локализация по индексу шага означает окончательную идентификацию экстремума. Поэтому погрешность вычисления экстремумов решения уравнения (15) полностью определяется погрешностью разностного приближения и не превысит ее. В этих границах имеют место устойчивость локализации и вычисления экстремумов решения (16) [6, 14]. Для поиска нулей достаточно взять модули значений (17) и из их минимумов выбрать близкие к нулю. Как и в случае функции одной переменной, необходимо смещение текущего отрезка по всему промежутку поиска экстремумов с дополнительной проверкой границ смещаемого отрезка на экстремум. Проверка выполняется сортировкой некоторого числа элементов в окрестности левой границы текущего отрезка с присоединением оператора локализации. В итоге достигается инвариантность схемы относительно размеров исследуемого отрезка $[t_0, T]$. Метод Эйлера можно заменить разностным методом более высокого порядка. Численный эксперимент с кодом программ на Delphi представлен в [6].

Схема без принципиальных изменений переносится на случай двух и более уравнений посредством ее буквального повторения для каждого уравнения в отдельности. Предполагается, что для задачи Коши

$$\frac{dY}{dt} = F(t, Y),\ Y(t_0) = Y_0, \tag{18}$$

где $F(t,Y) = (f_1(t,Y), f_2(t,Y), \ldots, f_n(t,Y))$, $Y = (y_1(t), y_2(t), \ldots, y_n(t))$, $Y_0 = (y_{01}, y_{02}, \ldots, y_{0n})$, выполнены все условия существования и единственности. При этом имеет место сходимость к решению рассматриваемого разностного метода на отрезке $[t_0, T]$. На вход сортировки подаются разностные приближения (для простоты по методу Эйлера) каждой компоненты решения

$$y_{k(i+1)} = y_{ki} + f_k(t_i, y_{1i}, y_{2i}, \ldots, y_{ni}) \cdot h,\ ck[i] = y_{ki}(t_{i-1}),\ k = 1, 2, \ldots, n; \tag{19}$$

$$t_i = t_0 + ih,\ i = 0, 1, \ldots, N,\ h = \frac{T - t_0}{N}.$$

В результате применения метода идентифицируются все локальные экстремумы каждой переменной y_k, дискретизированной в виде последовательности $ck[i]$, в окрестностях произвольного (достаточно малого) радиуса на всем отрезке $[t_0, T]$. Нули разностных приближений компонент будут идентифицированы, если на вход подавать модули значений (19) с оценкой на выходе малости минимумов. По-прежнему необходимо смещение текущего отрезка по всему промежутку поиска экстремумов с проверкой границ смещаемого отрезка на экст-

ремум. Вместо метода Эйлера могут применяться разностные методы высшего порядка. Численный эксперимент в [6] показывает возможность идентификации экстремумов с точностью до формата представления данных в языке программирования.

Для идентификации экстремумов нормы решений формируется одномерный массив значений нормы $\| Y(t_i) \|$ разностного решения системы (18), например

$$c[i] = \sqrt{|y_1[i-1]|^2 + |y_2[i-1]|^2 + \ldots + |y_n[i-1]|^2}, \quad i = 1, 2, \ldots, N. \tag{20}$$

С использованием процедуры *sort* и оператора локализации к одномерному массиву (20) применяется схема идентификации экстремумов числовой последовательности. Нули разностного приближения вектор-функции решения системы (18) определяются как достаточно малые минимумы нормы (20).

Определение экстремумов возмущений разностных решений систем ОДУ при вариации параметров. Рассматриваемый подход можно применить для определения экстремумов возмущений решений систем ОДУ в разностной форме, в том числе при вариации параметров. Идентификации экстремумов норм мультипликативных [17, 18] преобразований разностных решений позволяет оценить экстремальное отклонение системы от устойчивого состояния. Устойчивость в смысле Ляпунова понимается по определению из [19]. Пусть дана система ОДУ общего вида

$$\frac{dY}{dt} = F(t, Y, a_1, a_2, a_3), \; Y(t_0) = Y_0, \tag{21}$$

где $Y(t)$ определяются как в (18), $t_0 \le t \le T$. Для простоты рассматриваются три варьируемых числовых параметра: a_1, a_2, a_3 в диапазоне $a_{10} \le a_1 \le a_{11}$, $a_{20} \le a_2 \le a_{21}$, $a_{30} \le a_3 \le a_{31}$ (аналогично могут решаться задачи с четырьмя и более параметрами). Возмущенные начальные данные обозначаются $\widetilde{Y}(t_0) = \widetilde{Y}_0$, соответственное возмущенное решение — $\widetilde{Y}(t)$. Требуется найти все экстремальные значения возмущений при вариации параметров в разностном приближении решений. Для этого в границах вариации достаточно оценить норму отклонения от нуля разности между возмущенным и невозмущенным разностными решениями системы (21).

Способ основывается на изложенной схеме идентификации экстремумов функции четырех переменных. В качестве дискретизируемой функции выступает норма вычисляемых по разностной схеме значений $Y(t) - \widetilde{Y}(t)$, при этом в качестве одной независимой переменной выступает t, в качестве трех других — варьируемые параметры. При выборе, например, евклидовой нормы на вход метода поступает элемент массива

$$c[i, a_1, a_2, a_3] = \sqrt{\sum_{k=1}^{n} |y_k[i-1] - \widetilde{y}_k[i-1]|^2}, \quad i = 1, 2, \ldots, N. \tag{22}$$

Значения $Y(t), \widetilde{Y}(t)$ синхронно вычисляются по разностной схеме отдельно для каждого набора дискретизированных значений параметров. Шаг изменения независимой переменной, если это не нарушает корректности решения, выбирается на порядок меньше шага дискретизации параметров (с целью ускорения вычислений). Дискретизированные значения левой части (22) задают четырехмерный массив с элементами $c[i, j, \ell, r]$. Согласно (22) $c[i, j, \ell, r] = c[i, a_1, a_2, a_3]$ при значениях параметров $a_1 = a_1[j]$, $a_2 = a_2[\ell]$, $a_3 = a_3[r]$, индексы соответствуют номерам шагов дискретизации. В остальном схема идентификации экс-

тремумов функции четырех переменных не изменяется. Найденные по такой схеме экстремумы характеризуют разностное приближение меры отклонения возмущенного решения от невозмущенного. Видоизменение условия (8) в форме неравенства $|e[k+L]-e[k]|\ge 1,\ L=1,2,\dots,n-k,$ в результате применения метода дает глобальный максимум; аналогичное изменение условия (7) дает глобальный минимум [12]. С этими изменениями идентифицируются глобальные экстремумы возмущения решения системы (21) на отрезке $t_0 \le t \le T$ при всех дискретных значениях трех параметров, в результате определяются значения параметров и момент времени, при которых экстремумы возмущений наиболее «опасны» для системы управления.

Для общей оценки возмущение решения системы (21) преобразуется к виду

$$\widetilde{y}_{k(i+1)} - y_{k(i+1)} = \prod_{\ell=0}^{i} (1+D_{i-l}^{(k)}h)(\widetilde{y}_{k0} - y_{k0}) + \sum_{r=0}^{i-1} \prod_{\ell=0}^{i} (1+D_{i-l}^{(k)}h)\, w_{k(i-r-1)} + w_{ki},$$

где h дано в (19), $D_i^{(k)} = \dfrac{f_k(t_i,\widetilde{Y}_i) - f_k(t_i,Y_i)}{\widetilde{y}_{ki} - y_{ki}}$, w_{ki} — остаточные члены разложения разности решений по формуле Тейлора на i-м шаге метода Эйлера, $k=1,2,\dots,n$. На основе данного преобразования оценивается не только экстремальное возмущение при рассматриваемой вариации параметров, но и устойчивость невозмущенного решения [6, 17–20]. При этом оценке подлежит норма вектора, компонентами которого являются $\prod_{\ell=0}^{i} (1+D_{i-l}^{(k)}h),\ k=1,2,\dots,n.$

Аналогичное преобразование для линейной однородной системы

$$\frac{dY}{dt} = A(t, a_1, a_2, a_3)Y,\ Y(t_0) = \vec{0}$$

примет вид $\widetilde{Y}_{i+1} - Y_{i+1} = \prod_{\ell=0}^{i+1} (E + hA(t_\ell, a_1, a_2, a_3))(\widetilde{Y}_0 - Y_0)$ [17, 18]. Для оценки устойчивости данной системы достаточно анализировать экстремумы значений $\left\| \prod_{\ell=0}^{i+1} (E + hA(t_\ell, a_1, a_2, a_3)) \right\|$ при вариации дискретизированных параметров. Глобальный экстремум непосредственно указывает меру отклонения системы от устойчивого состояния на промежутке решения в границах вариации. Соответственные численные эксперименты с использованием различных разностных схем и полные коды программ приводятся в [6, 20, 21]. В [21] данный подход применяется к оценке возмущений энергетических систем большой мощности при возмущении параметров. Обоснование опирается на критерии, излагаемые в [17, 18]. Такой подход соотносится с возможностью анализа устойчивости линейной системы на основе нулей характеристической функции. Например, для системы управления с идеальным запаздыванием по времени (рис. 1) характеристическое уравнение имеет вид $s^3 + 2s^2 + s + e^{-t_0 s} = 0$. Нули трансцендентной функции $f(s) = s^3 + 2s^2 + s + e^{-t_0 s}$ можно найти по описанной выше схеме, если $f(s)$ преобразовать в функцию двух переменных умножением на комплексно-сопряжен-

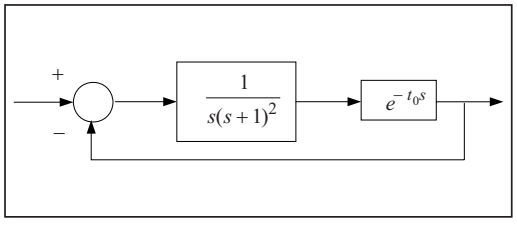

Рис. 1

ное значение. При вариации t_0 с шагом 0.01 вычисляются следующие попарно располагаемые значения действительной и мнимой частей нулей: (0.01: −0.12, −0.74; −0.12, 0.74; −1.75, −4.56); (0.02: −0.12, −0.74; −0.12, 0.74; −1.75, −4.57); … ; (0.51: −2.32, 0.71; −2.32, −0.71; −1.64, −2.35); (0.52: 1.97E−003, 0.71; 1.97E−003, −0.71; −1.64, 0); (0.53: −1.63, 0; 4.16E−003, 0.71; 4.16E−003, −0.71); (0.54: −1.63, 0; 6.34E−003, 0.71; 6.34E−003, −0.71); (0.55: 8.51E−003, 0.71; 8.51E−003, −0.71; −1.62, 0); (0.56: 1.07E−002, 0.71; 1.07E−002, −0.71; −1.62, −2.57); …

Таким образом, на отрезке $[0, 0.51]$ имеет место асимптотическая устойчивость системы (отрицательные действительные части нулей), на отрезке $[0.52, 0.55]$ имеет место неасимптотическая устойчивость (нулевые действительные части), далее следует область неустойчивости, которая меняет характер с ростом t_0.

Подход к поиску на аналогичной основе нулей и экстремумов разностных решений уравнений в частных производных излагается в [6, 13].

Обобщенная локализация экстремумов на основе сортировки. Идентификация экстремальных элементов последовательности согласно условиям (7), (8) выполняется исключительно по индексам, поэтому может быть отнесена к последовательности элементов произвольного множества, если на нем априори задано отношение порядка $\le$. Возможно самостоятельное определение программистом функции сравнения сложных объектов и выделение среди них экстремальных в смысле данного сравнения. По метрике, которая должна быть априори определена на множестве, идентифицируется значение экстремального элемента или объекта. Выделенные по экстремальным признакам объекты можно сопоставлять с помощью функции сравнения и метрического значения с пороговыми элементами, эталонными объектами, определять экстремальную меру сходства и отличия. При таком подходе возможно осуществлять синтез алгоритмов поиска [22] и распознавать объекты сложной природы при наличии помех [23–25]. В силу идентификации согласно условиям (7), (8) одновременно всех экстремумов представляется информация о топологии взаимного положения объектов распознавания. Информация включает множество всех локальных экстремумов во всех окрестностях локализации при всех априори заданных радиусах в многомерном пространстве с метрикой. В каждой окрестности экстремум идентифицируется по индексам и метрическому значению. Достигается полнота образа рельефа пространственной поверхности и расположения на ней объектов. Могут быть идентифицированы не только отдельные экстремумы, но и любое их подмножество с определяемой индексами конфигурацией взаимного расположения. На этой основе осуществимо распознавание целевых объектов и их подмножеств с заданными экстремальными свойствами.

Заключение. Изложенная локализация экстремума функции n переменных в окрестности произвольного (достаточно малого) радиуса ε_0 означает одновременное вычисление точки экстремума с границей абсолютной погрешности ε_0 в метрике R_n. При оценке устойчивости не обязательно выполнять спуск — знак действительной части нулей характеристической функции определится непосредственно в результате локализации. Это, в частности, относится к случаю локализации нулей характеристического полинома матрицы коэффициентов системы линейных ОДУ. С точностью до размерности представленные схемы различаются лишь способом представления и дискретизации данных на входе метода в зависимости от вида задачи. С их помощью можно оценить локально и глобально экстремальное отклонение решения дифференциальной системы от устойчивого состояния. Схемы отличаются построением на основе сортировки, автоматичностью программной локализации экстремумов функций, а также разностных решений дифференциальных уравнений, возможностью приближенного решения систем алгебраических и трансцендентных уравнений. Программная реализация схем инвариантна относительно размеров области исследования и распараллеливается по фрагментам области.

СПИСОК ЛИТЕРАТУРЫ

1. Р о м м Я. Е. Локализация и устойчивое вычисление нулей многочлена на основе сортировки. I // Кибернетика и системный анализ. — 2007. — № 1. — С. 165–183.
2. В и р т Н. Алгоритмы и структуры данных. — М.: Мир, 1989. — 360 с.
3. Р о м м Я. Е. Параллельная сортировка слиянием по матрицам сравнений. I // Кибернетика и системный анализ. — 1994. — № 5. — С. 3–23.
4. Р о м м Я. Е. Параллельная сортировка слиянием по матрицам сравнений. II // Там же. — 1995. — № 4. — С. 13–37.
5. М а р к у ш е в и ч А. И., М а р к у ш е в и ч Л. А. Введение в теорию аналитических функций. — М.: Просвещение, 1997. — 320 с.
6. З а и к а И. В. Разработка и исследование схем оптимизации на основе алгоритмов сортировки с приложением к идентификации экстремумов решений дифференциальных уравнений: Автореф. дис. ... канд. техн. наук. — Таганрог: ТРТУ, 2007. — 19 с.
7. Р о м м Я. Е., З а и к а И. В., Л а б и н ц е в а А. А. Безусловная численная оптимизация при вариации параметров. I. — Таганрог, 2008. — 31 с. Деп. в ВИНИТИ 04.03.2008, № 193-В2008.
8. Р о м м Я. Е., З а и к а И. В., Л а б и н ц е в а А. А. Безусловная численная оптимизация при вариации параметров. II. — Таганрог, 2008. — 44 с. Деп. в ВИНИТИ 04.03.2008 № 194-В2008.
9. Р о м м Я. Е., З а и к а И. В., Л а б и н ц е в а А. А. Безусловная оптимизация на основе сортировки с приложением к компьютерному анализу устойчивости систем управления // Изв. вузов. Северо-Кавказ. регион. Техн. науки. Сер. «Управление, вычислительная техника и информатика». — 2008. — № 6. — С. 11–17.
10. Б е р е з и н И. С., Ж и д к о в Н. П. Методы вычислений. — М.: Физматгиз, 1962. — Т.2. — 640 с.
11. В а с и л ь е в Ф. П. Численные методы решения экстремальных задач. — М.: Наука, 1988. — 552 с.
12. Р о м м Я. Е. Локализация и устойчивое вычисление нулей многочлена на основе сортировки. II // Кибернетика и системный анализ. — 2007. — № 2. — С. 161–175.
13. Р о м м Я. Е., З а и к а И. В., Т ю ш н я к о в а И. А. Идентификация экстремумов функций на основе сортировки с приложением к вычислительным схемам алгебры, анализа и распознаванию изображений // Проблеми програмування: Матеріали 5-ї Міжнар. наук.-практ. конф. з програмування УкрПРОГ'2006. 23–25 травня 2006 р., Україна, Київ. — 2006. — № 2–3. — С. 708–717.
14. Р о м м Я. Е., З а и к а И. В. Программная локализация экстремумов функций и разностных приближений решений дифференциальных уравнений // Изв. вузов. Северо-Кавказ. регион. Техн. науки. Спец. выпуск «Математическое моделирование и компьютерные технологии», 2005. — С. 55–61.
15. Р о м м Я. Е., З а и к а И. В., С о л о в ь е в а И. А. Метод программной оптимизации в приложении к математическим моделям экономики. Сб. докл. IV-й Междунар. науч.-практ. конф. «Проблемы регионального управления, экономики, права и инновационных процессов в образовании». Т. 2 (8–10 сентября 2005 г., ТИУиЭ). — Таганрог, 2005. — С. 17–26.
16. Д е м и д о в и ч Б. П., М а р о н И. А. Основы вычислительной математики. — М.: Физматгиз, 1963. — 660 с.
17. Р о м м Я. Е. Мультипликативные критерии устойчивости на основе разностных решений обыкновенных дифференциальных уравнений // Кибернетика и системный анализ. — 2006. — № 1. — С. 128–144.
18. Р о м м Я. Е. Моделирование устойчивости по Ляпунову на основе преобразований разностных схем решений обыкновенных дифференциальных уравнений // Изв. РАН. Мат. моделирование. — 2008. — **20**, № 12. — С. 105–118.
19. Ч е з а р и Л. Асимптотическое поведение и устойчивость решений обыкновенных дифференциальных уравнений. — М.: Мир, 1964. — 478 с.
20. К а т р и ч С. А. Разработка и исследование схем программного моделирования устойчивости решений нелинейных дифференциальных уравнений на основе разностных методов: Автореф. дис. ... канд. техн. наук. — Таганрог: ТРТУ, 2006. — 20 с.
21. Б у л а н о в С. Г. Разработка и исследование методов программного моделирования устойчивости систем линейных дифференциальных уравнений на основе матричных мультипликативных преобразований разностных схем: Автореф. дис. ... канд. техн. наук. — Таганрог: ТРТУ, 2006. — 20 с.
22. Б е л о к о н о в а С. С. Разработка и исследование схем детерминированного поиска на основе сортировки с приложением к идентификации оцифрованных объектов различных типов: Автореф. дис . ... канд. техн. наук. — Таганрог: ТТИ ЮФУ, 2007. — 18 с.
23. Р ю м и н О. Г. Разработка и исследование алгоритмов распознавания изображений на основе определения экстремальных признаков замкнутых контуров с помощью сортировки: Автореф. дис. ... канд. техн. наук. — Таганрог: ТТИ ЮФУ, 2008. — 16 с.
24. Р о м м Я. Е., Д о р д о п у л о А. И., З а я р н ы й В. В. Программная локализация нулей многочленов с приложением к идентификации объектов по данным гидроакустической локации. — Таганрог: Изд-во ТГПИ, 2005. — 217 с.
25. Р о м м Я. Е., З а я р н ы й В. В. Компьютерная обработка сигналов гидроакустической локации и идентификация объектов растровых изображений // Системы управления и информационные технологии. — 2007. — № 4, Вып. 2(30). — С. 290–295.

Поступила 25.06.2009