УДК 681.3.06

ОБРАБОТКА КАРТ ИЗОЛИНИЙ ДВУМЕРНЫХ ФУНКЦИЙ

Л.Г. Белиовская[1], Н.А. Богомолов[1], А.Д. Ковалев[1]

Рассматриваются вопросы построения и обработки линий уровня вещественных функций двух переменных. Для сглаживания изолиний используются параметрические соприкасающиеся сплайны. Предлагается алгоритм построения профилей двумерных функций. Обсуждается реконструкция функции по заданному семейству изолиний различных уровней.

Ключевые слова: линии уровня функций, сплайны, профили двумерных функций, реконструкция функций, инструментальные программные средства, интерполяция.

Данная работа продолжает серию публикаций, посвященных описанию Инструментального комплекса поддержки Базовой технологии создания систем обработки и отображения сложно организованной информации, разрабатываемого в НИВЦ МГУ (см., например, [1]). В ней содержатся описания операторов Главного управляющего языка Инструментального комплекса, предназначенных для построения и обработки линий уровня функций двух переменных.

1. Построение изолиний двумерной функции. Команда izoline_make Главного управляющего языка строит линии уровня $f(x, y) = c_k$, $k = 1, 2, \ldots, nc$, для функции двух переменных $f(x, y)$, заданной в узлах прямоугольной неравномерной сетки

$$\{x_i, y_j\}, \quad i = 1, 2, \ldots, nx\,, \quad j = 1, 2, \ldots, ny\,.$$

Эта двумерная сетка разбивает область определения функции на прямоугольные ячейки $\big(x_{i-1} \leqslant x \leqslant x_i\,,\ y_{j-1} \leqslant y \leqslant y_j\big)$. Доопределим функцию $f(x_i, y_j)$ на ребрах ячеек, используя линейную интерполяцию. Тогда линиями уровня будут ломаные, проходящие через точки, лежащие на ребрах ячеек, в которых $f(x, y) = c_k$, $k = 1, 2, \ldots, nc$.

Будем различать незамкнутые изолинии, начинающиеся и заканчивающиеся на границе области, и замкнутые линии, лежащие целиком внутри области определения функции. Задача построения изолиний решается следующим образом.

В начале строим незамкнутые изолинии. Для обнаружения начальных точек незамкнутых изолиний осуществляем обход по границе. Как только начальная точка обнаружена, изолиния отслеживается до конца, т.е. до ее выхода на границу. Чтобы исключить повторное проведение изолиний, в процессе отслеживания регистрируется факт ее прохождения через данное ребро. При этом считается, что через каждое ребро нельзя провести более одной изолинии заданного уровня. Затем строятся замкнутые изолинии. Информация о пройденных ребрах сохраняется и используется для определения конца изолинии. При обнаружении начальной точки рассматривается поведение линии внутри ячейки сетки. Если линия вошла в ячейку, то она должна выйти из нее через одно из оставшихся трех ребер. Проверяя соотношение

$$\big(f(\bar{x}, \bar{y}) - c_k\big) * \big(f(\bar{\bar{x}}, \bar{\bar{y}}) - c_k\big) \leqslant 0 \tag{1}$$

, где $(\bar{x}, \bar{y})$ и $(\bar{\bar{x}}, \bar{\bar{y}})$ — координаты вершин ребра, определяем, через какое ребро линия вышла. Далее рассматривается ячейка, соседняя с этим ребром, и операция повторяется. Координаты точки пересечения линии уровня с ребром находятся с помощью линейной интерполяции. Эти координаты запоминаются в массиве izo.

Массив rebro характеризует ребра сетки области. Порядковый номер элемента массива rebro однозначно определяет ребро сетки. Нумерация ребер осуществляется следующим образом. Сначала нумеруются все горизонтальные ребра. Пусть (i, j) и $(i+1, j)$ — координаты точек, характеризующих горизонтальное ребро, исходящее из точки (i, j). Тогда номер ребра равен $j(ny - 1) + i + 1$, $i = 0, \ldots, nx - 2$, $j = 0, \ldots, ny - 1$. Общее число горизонтальных ребер равно $(nx - 1) * ny$. Затем нумеруются вертикальные ребра. Пусть (i, j) и $(i, j+1)$ — координаты точек, характеризующих вертикальное ребро, исходящее из точки (i, j). Тогда номер ребра равен $(nx-1)*ny+i(ny-1)+j+1, i = 0, \ldots, nx-1, j = 0, \ldots, ny-2$. Общее число вертикальных ребер равно $(ny-1)*nx$. Таким образом, сетка разбиения области имеет $(nx-1)*ny+(ny-1)*nx = 2*nx*ny-ny-nx$

[1] Научно-исследовательский вычислительный центр, Московский государственный университет им. М.В. Ломоносова, 119899, Москва; e-mail: nbogom@mail.ru, kovalev_ad@list.ru

© Научно-исследовательский вычислительный центр МГУ им. М. В. Ломоносова

ребер. Это число и определяет длину одномерного массива rebro. Если в процессе построения изолиний элемент этого массива отличен от 0, то это означает, что через соответствующее ребро прошла изолиния.

При поиске линий уровня на сетке могут встретиться вырожденные ситуации. Во-первых, возможно проведение линии уровня через все ребра ячейки и во-вторых, линия проходит точно через вершину ячейки. В первом случае из трех возможных вариантов соединения отдается предпочтение тому, который предполагает прохождение изолинии вдоль диагонали, построенной через те узлы ячейки, где значения функции больше значения строящегося уровня. Вырожденность второго типа приводит либо к повторному проведению линии, либо к ее обрыву. Чтобы избежать этого, к вырожденному узлу добавляется некоторое малое число delta. В результате рассматриваемая вырожденность устраняется.

Обход границы начинается с точки (0,0) так, что область все время остается слева. Начальная координата изолинии определяется в процессе последовательного перемещения по границе с одновременной проверкой условия (1). Заметим, что каждая ячейка сетки идентифицируется номером (i, j) ее левого нижнего узла.

После того как построены все незамкнутые изолинии, проводится последовательный просмотр всех горизонтальных ребер ячеек для выявления точки, принадлежащей замкнутой изолинии. После обнаружения такой точки, изолиния отслеживается до конца, т.е. до возврата в ту же начальную точку. Как только построены все линии, соответствующие уровню c_k, переходим к следующему уровню c_{k+1}.

Команда Главного управляющего языка для построения изолиний заданных уровней двумерной функции имеет вид

```
izoline_make matr #fn1 #fn2 nx ny x y level nlev,
```

где matr — исходный массив значений (типа double) двумерной функции в узлах сетки;
#fn1, #fn2 — номера открытых файлов для записи координат изолиний и их атрибутов;
nx,ny — размерности массивов координат узлов сетки по осям x и y;
x,y — массивы координат узлов сетки по осям x и y соответственно;
level — массив значений уровней изолиний (типа double);
nlevel — размерность массива значений уровней изолиний.

Реализация этой команды основана на Си-функции tkxl_all_level , которая вычисляет координаты точек изолиний всех заданных уровней.

```
void tkxl_all_level(int f1, int f2, int nx, int ny, double*x,
                double*y, double*matr, double*lev, int nlev),
```

где nx — размерность массива координат узлов сетки по оси x;
ny — размерность массива координат узлов сетки по оси y;
x — массив координат узлов сетки по оси x;
y — массив координат узлов сетки по оси y;
matr — массив значений (типа double) двумерной функции в узлах сетки;
lev — массив значений изолиний;
nlev — размерность массива значений изолиний.

Данная Си-функция, многократно обращаясь к другой Си-функции tkxl_level, которая находит значения координат точек изолинии одного уровня, строит всю карту изолиний.

Рассмотрим функцию tkxl_level вычисления координат точек изолиний одного уровня. Обращение к ней имеет вид

```
static int tkxl_level(int nx, int ny, double val, double*x,
                      double*y,int f1, int f2, double*matr,
                     double**izo, double delta, int number),
```

где nx, ny — размерность сетки по осям x и y соответственно;
val — значение уровня;
∗x и ∗y — указатели на массивы координат точек сетки по осям x и y соответственно;
∗matr — указатель на двумерный массив значений функции в узлах сетки;
∗∗izo — указатель на указатель массива координат точек вычисленной изолинии;
delta — значение отступа от узла сетки;
f1, f2 — файлы с координатами изолиний и их атрибутами;
number — номер фрагмента изолинии (сквозная нумерация по всем уровням).

Данная Си-функция осуществляет последовательный обход точек границы области, а затем просматривает все горизонтальные ребра ячеек сетки в поиске начала изолинии. Как только начало изолинии найдено,

вычисляется массив координат изолинии, а затем из него формируется текстовый файл с координатами изолинии и файл атрибутов. После того как вычисленные координаты точек изолинии занесены в файл, необходимо освободить заказанную под вспомогательные массивы память.

В своей работе Си-функция tkxl_level использует два основных вспомогательных массива rebro и izo. Вспомогательный массив типа int rebro содержит характеристики ребер сетки. Его элементы принимают значение 0, если через ребро ячейки не прошла еще изолиния, или 1, если через ребро ячейки уже прошла изолиния. Нумерация ребер производится по следующему правилу: вначале нумеруются все горизонтальные ребра, а затем — вертикальные. Размерность i_rebro массива rebro равна 2*nx*ny–nx–ny+1, где nx и ny число точек сетки по осям x и y соответственно. Другой массив izo (типа double) содержит координаты изолиний. Си-функция tkxl_fragm вычисляет этот массив и, при необходимости, увеличивает его размерность с помощью Си-функции tkb_vmn_buf_realloc из состава Инструментального комплекса.

Расчет коодинат точек изолинии представляет собой наиболее сложную и громоздкую часть рассматриваемого блока команд. Под фрагментом изолинии мы понимаем связную последовательность звеньев изолинии. Си-функция tkxl_fragm формирует массив с координатами фрагмента изолинии. Обращение к ней имеет вид

```
static int tkxl_fragm(int i, int j, int k, int nx, int ny,
              double val, double*x, double*y, double*matr,
            double**izo, int*rebro, int nnn, double delta),
```

где i, j — координаты рассматриваемой ячейки сетки;

k — характеристика ребра ячейки, через которое может проходить изолиния: k = 0, если ребро нижнее; k = 1, если левое; k = 2, если верхнее; k = 3, если ребро правое;

nx — размерность сетки по оси x;

ny — размерность сетки по оси y;

val — значение линии уровня;

∗x и ∗y — указатели массивов координат узлов сетки по оси x и по оси y;

∗matr — указатель на массив значений двумерной функции в узлах сетки;

∗∗izo — указатель на указатель массива с координатами изолинии;

∗rebro — вспомогательный массив характеристик ребер сетки; его элемент равен 0, если через ребро не прошла еще изолиния, или 1, если через ребро уже прошла изолиния;

nnn — параметр поиска: nnn=1 для незамкнутой изолинии и nnn=2 для замкнутой изолинии;

delta — значение отступа от узла сетки.

Опишем вкратце алгоритм данной Си-функции. При просмотре текущего ребра проводится целый комплекс проверок, в том числе: является ли ребро граничным, проходит ли через него изолиния заданного уровня, может ли проходить эта изолиния через какую-либо вершину данного ребра. Если последнее из условий выполняется, то значение функции в этой вершине уменьшается на величину отступа delta. Затем проверяется, может ли проходить изолиния через данное ребро. Если может, то с помощью линейной интерполяции находятся ее координаты. Далее выбирается следующее ребро для проверки прохождения через него изолинии.

Си-функция tkxl_fragm использует массив izo типа double, который содержит координаты изолиний. Размерность его равна i_izo и при первоначальном обращении полагается равной 100, а при необходимости увеличивается с помощью Си-функции tkb_vmn_buf_realloc. Возвращаемым параметром tkxl_fragm является число точек в только что построенном фрагменте изолинии.

Вспомогательная Си-функция tkxl_write_data формирует файлы с координатами и их атрибутами:

```
void tkxl_write_data(int vfn1, int vfn2, double**izo,
               int n, int n1, int number, double val),
```

где int vfn1 — файл с координатами изолиний;

int vfn2 — файл с атрибутами файла vfn1;

double∗∗izo — указатель на указатель массива координат точек построенной изолинии;

int n — число точек в данном фрагменте изолиний;

int n1 — номер фрагмента (сквозная нумерация по всем изолиниям одного уровня);

int number — номер фрагмента (сквозная нумерация по всем уровням);

double val — значение уровня.

2. Сглаживание изолиний двумерной функции. Среди приближений ломаной линии гладкими кривыми выделяют гладкие линии двух типов: эрмитовый локальный описывающий сплайн и параметрический соприкасающийся сплайн. Как показали экспериментальные расчеты, при использовании для сглаживания эрмитовых локальных сплайнов изолинии разных уровней могут пересекаться за счет ошибок в

определении значений производных в узлах интерполяции. Использование же соприкасающихся сплайнов свободно от этих недостатков [2].

Предположим, что на плоскости задана ломаная своими вершинами $p_0(x_0, y_0), \ldots, p_n(x_n, y_n)$. Построим составную кривую, соприкасающуюся с данной ломаной Для этого потребуем, чтобы:

1) каждому звену ломаной $\{p_{k-1}, p_k\}$, $k = 1, \ldots, n$, принадлежал ровно один узел интерполяции $q_k = (\bar{x}_k, \bar{y}_k)$;

2) в узлах интерполяции q_k интерполяционная кривая касалась бы звеньев ломаной;

3) на каждом элементарном промежутке между двумя соседними узлами кривая являлась бы сегментом параболы;

4) интерполяционная кривая "сохраняла бы форму" ломаной, т.е. направления выпуклости кривой и ломаной должны совпадать на каждом элементарном промежутке.

Пусть ломаная не замкнута. Положение узла интерполяции q_k на каждом звене $\{p_{k-1}, p_k\}$ ломаной удобно задать с помощью параметра L_k:

$$L_k = \frac{|p_{k-1} - q_k|}{|q_k - p_k|}, \quad k = 1, \ldots, n,$$

который представляет собой отношение, в котором узел интерполяции q_k делит отрезок $[p_{k-1}, p_k]$. Тогда координаты узлов интерполяции определяются равенствами

$$\begin{aligned} \bar{x}_k &= \frac{x_{k-1} + L_k x_k}{1 + L_k}, \\ \bar{y}_k &= \frac{y_{k-1} + L_k y_k}{1 + L_k}. \end{aligned} \tag{2}$$

В результате получим уравнение параболического соприкасающегося сплайна в виде

$$\begin{aligned} x &= (\bar{x}_k - x_k)(1-t)^2 + (\bar{x}_{k+1} - x_k)t^2 + x_k, \\ y &= (\bar{y}_k - y_k)(1-t)^2 + (\bar{y}_{k+1} - y_k)t^2 + y_k, \end{aligned}$$

где $t \in [0, 1]$, $k = 1, \ldots, n$ и $\bar{x}_k$, $\bar{y}_k$ определяются из (2). Для каждого k при изменении t от 0 до 1, точка на параболе пробегает от узла q_k до q_{k+1}.

Можно использовать многочлены более высокой степени:

$$\begin{aligned} x &= (\bar{x}_k - x_k)(1-t)^m + (\bar{x}_{k+1} - x_k)t^m + x_k, \\ y &= (\bar{y}_k - y_k)(1-t)^m (\bar{y}_{k+1} - y_k)t^m + y_k, \end{aligned} \tag{3}$$

где $t \in [0, 1]$, $k = 1, \ldots, n$ и $m = 2, 3, \ldots$. При $m = 2$ имеем параболический сплайн, кубическая соприкасающаяся кривая получается при $m = 3$. При больших значениях m вершины кривой располагаются ближе к вершинам ломаной. В предельном случае (при m стремящемся к бесконечности) интерполяционный сплайн стремится занять положение ломаной.

Рассмотрим более подробно алгоритм построения параметрического соприкасающегося сплайна порядка m.

Вначале предположим, что ломаная не замкнута. Совместим два узла интерполяции q_1 и q_n с вершинами p_0 и p_n, т.е. положим

$$\begin{aligned} \bar{x}_1 &= x_0, \quad \bar{y}_1 = y_0, \\ \bar{x}_n &= x_n, \quad \bar{y}_n = y_n. \end{aligned} \tag{4}$$

Пусть $L_k = 1$ при $k = 2, 3, \ldots, n-1$ для внутренних звеньев ломаной. Тогда

$$\begin{aligned} \bar{x}_k &= \frac{x_{k-1} + x_k}{2}, \\ \bar{y}_k &= \frac{y_{k-1} + y_k}{2}, \end{aligned} \tag{5}$$

где $k = 2, 3, \ldots, n-1$. В результате формулы (3) определяют искомый сплайн, если узлы $\bar{x}_k$ и $\bar{y}_k$ вычислять по формулам (5) для $k = 2, 3, \ldots, n-1$ или по формулам (4) для $k = 1$ и $k = n$.

Пусть ломаная замкнута. В этом случае узлы выбираются в середине каждого звена. Недостающий узел определяется следующим образом: вводится вершина p_{n+1} дополнительной ломаной, совпадающая с вершиной p_1; тогда новый узел q_k будет иметь координаты $\big((x_0 + x_1)/2\big)$, $\big((y_0 + y_1)/2\big)$. Теперь искомый сплайн опять строится по формулам (3) для так выбранного набора узлов при всех $k = 1, 2, \ldots, n$.

Для сглаживания уже построенных изолиний служит команда izoline_spline:

```
izoline_spline #fn1 #fn2 var_name,
```

где #fn1 — номер файла с координатами изолиний, открытого ранее на чтение командой fopen; #fn2 — номер файла с координатами сглаженных изолиний, открытого ранее на запись командой fopen;

izo_env — имя переменной, содержащей характеристики сплайна; в ней в формате "2d" задаются: m – порядок сплайна и mm – количество точек в сплайне.

Эта команда основана на Си-функции tkxl_spline, которая сглаживает изолинии методом построения параметрического соприкасающегося сплайна порядка m. Обращение к ней имеет вид

```
int tkxl_spline( int f1, int f3, int m, int mm ),
```

где f1 — файл с координатами изолиний;

f3 — файл с координатами сглаженных изолиний;

m — порядок сплайна (степень приближения к ломаной);

mm — число точек в сплайне (до 30).

Данная Си-функция последовательно считывает координаты точек фрагмента изолинии в массив и определяет, замкнут ли этот фрагмент или нет. Если ломаная изолинии замкнута, тогда выберем узлы интерполирования в середине каждого звена. Вводим дополнительную вершину ломаной, совпадающую с начальной, и следовательно, новый узел. Если ломаная не замкнута, тогда совместим два узла сглаживающей прямой с начальной и конечной точками ломаной. Выберем узлы интерполирования для внутренних точек в середине каждого звена. Для полученных узлов интерполирования вычисляем точки сплайна, учитывая порядок сплайна (степень приближения к ломаной и число промежуточных точек при сглаживании). Затем записываем координаты вычисленных точек в выходной текстовый файл.

3. Построение профиля двумерной функции. Для отрезка с известными координатами требуется найти все координаты точек пересечения с изолиниями, причем координаты точек изолиний известны заранее.

Сложность реального алгоритма определяется прежде всего требованиями к его быстродействию. Пусть AB — отрезок, вдоль которого строится профиль. Если его конечная и начальная точка совпадают, то работа программы прекращается с соответствующей диагностикой. Рассмотрим также расположение самого отрезка относительно осей координат. Если отрезок расположен вдоль оси абсцисс, т.е. ybeg = yend, то параметр iy полагается равным 1. При расположении отрезка вдоль оси ординат (xbeg = xend) параметр ix полагается равным 1. Если отрезок не параллелен ни одной из осей, то как ix, так и iy полагаются равными 0.

Затем для двух несовпадающих точек (x_1, y_1) и (x_2, y_2), определяющих одно из звеньев изолинии, необходимо выяснить, пересекает ли данное звено исследуемый отрезок. Представим последний в виде нескольких равных полуинтервалов

$$\big[(x_i,\, y_i);\, (x_{i+1},\, y_{i+1})\big)\,, \quad i = 0,\, 1,\, \ldots\,,\, \mathrm{np}\,. \tag{6}$$

Выделим случаи совпадения конечных точек звена изолинии с левым концом полуинтервала (6) и параллельности звена одной из осей. Для конечной точки отрезка профиля (xend,yend) проверку выполним особо.

Далее необходимо было бы решать такую трудоемкую задачу: определить, находятся ли точки данного звена изолинии в разных полуплоскостях, определяемых прямой AB. Если это так, то надо построить через одну из точек звена и оба конца полуитнтервала (6) прямые линии l_1 и l_2 и проверить, принадлежит ли другая точка звена изолинии области, ограниченной отрезком AB и построенными прямыми. При положительном результате этой проверки вычисляются координаты точки пересечения. Отметим, что многие звенья изолиний будут, как правило, располагаться в значительном удалении от исследуемого полуинтервала и многие трудоемкие проверки будут напрасны. С целью исключения их и улучшения в связи с этим быстродействия программы, введем предварительные быстрые проверки расположения координат концов звена изолинии относительно полуинтервала. Если обе абсциссы (ординаты) звена меньше абсциссы (ординаты) левого конца полуинтервала или обе абсциссы (ординаты) звена больше абсциссы (ординаты) правого конца полуинтервала, то такое звено мы не рассматриваем и переходим к анализу следующего звена изолинии.

Теперь проведем более тщательное изучение возможности пересечения звена изолинии с полуинтервалом (6). Для этого исследуем подробнее возможные случаи расположения полуинтервала профиля AB. Возможны следующие варианты:

а1) полуинтервал расположен параллельно оси Oy (ix = 1); тогда уравнение прямой имеет вид $x = x_i$;

б1) полуинтервал расположен параллельно оси Ox (iy = 1); тогда уравнение прямой имеет вид $y = y_i$;

в1) в общем случае уравнение прямой имеет вид

$$(x - x_i)/(x_{i+1} - x_i) = (y - y_i)/(y_{i+1} - y_i)\,.$$

Для звена изолинии возможны аналогичные варианты:

а2) звено расположено вдоль оси Oy; тогда уравнение для точек звена имеет вид $x = \bar{x}_1$;

б2) звено расположено вдоль оси Ox; тогда уравнение для точек звена представимо в виде $y = \bar{y}_1$;

в2) в общем случае уравнение для точек звена имеет вид

$$(x - \bar{x}_1)/(\bar{x}_2 - \bar{x}_1) = (y - \bar{y}_1)/(\bar{y}_2 - \bar{y}_1)\,.$$

При совместном расположении полуинтервала профиля и звена изолинии в случае а1 и а2 результативной окажется ситуация, когда $x_i = \bar{x}_1$. Тогда общая часть их полностью войдет в профиль. Аналогично и для случая совместного расположения б1 и б2.

Рассмотрим все оставшиеся комбинации:

а1б2) если ордината $\bar{y}_1$ удовлетворяет условию $(y_i - \bar{y}_1) \times (y_{i+1} - \bar{y}_1) < 0$, то $(x_i,\, \bar{y}_1)$ есть искомая точка пересечения;

б1а2) если абсцисса $\bar{x}_1$ удовлетворяет условию $(x_i - \bar{x}_1) \times (x_{i+1} - \bar{x}_1) < 0$, то $(\bar{x}_1,\, y_i)$ есть искомая точка пересечения;

а1в2) если для $\bar{y}_0 = \bar{y}_1 + (x_i - \bar{x}_1) \times (\bar{y}_2 - \bar{y}_1)/(\bar{x}_2 - \bar{x}_1)$ справедливо $(y_i - \bar{y}_0) \times (y_{i+1} - \bar{y}_0) < 0$, то $(x_i,\, \bar{y}_0)$ есть искомая точка пересечения;

б1в2) если для $\bar{x}_0 = \bar{x}_1 + (y_i - \bar{y}_1) \times (\bar{x}_2 - \bar{x}_1)/(\bar{y}_2 - \bar{y}_1)$ справедливо $(x_i - \bar{x}_0) \times (x_{i+1} - \bar{x}_0) < 0$, то $(\bar{x}_0,\, y_i)$ есть искомая точка пересечения;

в1а2) если для $\bar{y}_{00} = y_i + (\bar{x}_1 - \mathrm{xbeg}) \times (\mathrm{yend} - \mathrm{ybeg})/(\mathrm{xend} - \mathrm{xbeg})$ справедливо $(\bar{y}_1 - \bar{y}_{00}) \times (\bar{y}_2 - y_{00}) < 0$, то $(\bar{x}_1,\, \bar{y}_{00})$ есть искомая точка пересечения;

в1б2) если для $\bar{x}_{00} = \mathrm{xbeg} + (\bar{y}_1 - \mathrm{ybeg}) \times (\mathrm{xend} - \mathrm{xbeg})/(\mathrm{ybeg} - \mathrm{yend})$ справедливо $(\bar{x}_1 - \bar{x}_{00}) \times (\bar{x}_2 - \bar{x}_{00}) < 0$, то $(\bar{x}_{00},\, \bar{y}_1)$ есть искомая точка пересечения;

в1в2) если для

$$xx = \frac{(\bar{x}_1 + (\mathrm{ybeg} - \bar{y}_1 - \mathrm{xbeg} \times (\mathrm{yend} - \mathrm{ybeg})/(\mathrm{xend} - \mathrm{xbeg})))}{(1 - (\mathrm{yend} - \mathrm{ybeg}) \times (\bar{x}_2 - \bar{x}_1)/((\mathrm{xend} - \mathrm{xbeg}) \times (\bar{y}_2 - \bar{y}_1)))}$$

справедливо $(x_i - xx) \times (x_{i+1} - xx) < 0$, то тогда искомая точка имеет координты $(xx,\, (xx - \mathrm{xbeg}) \times (\mathrm{yend} - \mathrm{ybeg}) + \mathrm{ybeg})$.

Для уменьшения времени счета, помимо предварительной проверки координат точек, необходимо вести обработку фрагментов изолиний, полностью помещенных в оперативную память машины. С этой целью разработана специальная Си-функция, которая заносит в оперативную память целые фрагменты изолиний, связанных с определенным значением двумерной функции.

После того как профиль на отрезке AB построен, граничные точки его, как правило, не определены, т.е. значения двумерной функции в точках A и B не найдены. Экстраполяция граничных точек осуществляется по следующему правилу. Если при приближении к концу отрезка функция значений уровней профиля монотонно убывает (возрастает), то значение профиля в граничной точке полагаем равным полусумме значений функции в последней известной точке, ближайшей к данной граничной, и следующего по уменьшению (увеличению) уровня. Если при этом значение профиля в ближайшей к данной граничной известной точке наименьшее (наибольшее), то значение в граничной точке полагаем равным значению в ближайшей к ней уже известной точке уменьшенному (увеличенному) на половину разницы между наименьшим (наибольшим) значением уровня и следующим за ним по увеличению (уменьшению) уровнем. Если при приближении к концу отрезка функция значений уровней остается постоянной, то значение профиля в граничной точке полагаем равным значению функции в ближайшей к данному концу, уже определенной точке.

В случае небольшого числа изолиний на карте или недостаточной длины выбранного отрезка профиля возможен вариант, при котором для заданного отрезка профиля не найдется ни одной точки пересечения с изолиниями. В этом случае Си-функция находит изолинию, ближайшую к данному отрезку, и полагает, что профиль имеет во всех точках ее значение. Величина минимального расстояния до изолинии в этом случае определяется.

На основе описанного алгоритма была разработана команда izoline_profile:

```
izoline_profile #fn1 #fn2 otr_coords xyv_prof,
```

где #fn1 — номер файла с координатами точек изолиний, открытого ранее на чтение командой fopen;

#fn2 — номер файла с атрибутами изолиний (включая значения уровней), открытого ранее на чтение командой fopen;

otr_coords — имя переменной, содержащей координаты отрезка, вдоль которого строится профиль; в ней в формате “4G” задаются абсцисса и ордината начальной точки отрезка профиля, а также абсцисса и ордината конечной точки отрезка профиля;

xyv_prof — имя переменной, содержащей массив точек построенного профиля, в котором в порядке возрастания координат расположены сами точки профиля со значениями двумерной функции в них; таким образом, сам массив состоит из троек чисел, первые два числа которых являются координатой точки профиля, а третье — значением функции в этой точке.

Команда основана на Си-функции tkxl_profile_all , которая и строит профиль двумерной функции на заданном отрезке по изолиниям:

```
int tkxl_profile_all( int f1, int f2, TKA_4D otr_coords,
                      double **prof_data, int np),
```

где f1, f2 — номера файлов с координатами точек и атрибутами изолиний;

TKA_4D otr_coords — координаты двух произвольных точек концов отрезка, на котором строится профиль otr_coords.x1, otr_coords.y1, otr_coords.x2, otr_coords.y2;

np — число точек в разбиении отрезка профиля; если np $\leqslant 0$, то профиль строится с автоматическим выбором шага (100 точек);

double**prof_data — указатель на указатель массива, содержащий координаты точки пересечения изолинии с отрезком и значение самой функции в этой точке.

Эта функция возвращает общее число точек пересечения всех изолиний с отрезком. Если нет точек пересечения с изолиниями, то Си-функция возвращает число точек пересечения, равное 2, и по указателю (*prof_data+6) можно определить значение расстояния до ближайшей изолинии.

Исходными параметрами программы являются координаты двух точек: (otr_coords.x1, otr_coords.y1) начальная и (otr_coords.x2, otr_coords.y2) конечная точка отрезка,вдоль которого строится профиль двумерной функции; файлы с координатами фрагментов изолиний и соответствующими им значениями уровней; значение шага разбиения исходного отрезка на подинтервалы. Шаг разбиения задается пользователем программы, который в какой-то мере знаком с особенностями двумерной функции или с ее картой изолиний. Как правило, шаг задается равным минимальному расстоянию между изолиниями. В случае задания неположительного шага программа работает с автоматическим выбором шага, разбивая отрезок на 100 равных частей.

Остановимся подробнее на алгоритме данной Си-функции. Вначале необходимо вычислить относительную степень близости точек и выбрать ось (Ox или Oy), которую будем считать в дальнейшем базовой: критерий выбора — максимальная длина отрезка построения профиля по этой оси. Затем, если отрезок вырождается в точку, выходим из СИ-функции. Потом проводим вычисление координат точек разбиения отрезка построения профиля на np равных частей и выделяем память под эти массивы. В процессе просмотра всех фрагментов изолиний, считывая текущее значение линии уровня, формируем массив значений уровней, расположенных по возрастанию без повторений.

Затем, используя Си-функцию tkxl_get_izoline , записываем фрагмент изолинии из файла f1 в массив, повторяющиеся точки опускаем. Поиск всех точек пересечения данного фрагмента изолинии с отрезком, по которому строится профиль, проводим с помощью Си-функции tkxl_profile1 . После того, как часть точек профиля найдена, проверяя необходимость увеличения памяти массива со значениями и координатами строящегося профиля, расположенного по адресу prof_data , записываем все координаты построенных точек профиля и значение функции в них. Таким образом исследуем все фрагменты изолиний всех уровней, записывая координаты найденных точек пересечения изолиний с отрезком и со значениями уровней в массив.

После того как вид профиля уже в целом определен, проводим экстраполяцию граничных точек профиля по правилу, описанному выше. Проводим окончательную сортировку всех точек профиля уже с вновь добавленными граничными точками. Осталось лишь удалить все точки с близкими координатами и освободить всю вспомогательную память, заказанную данной Си-функцией.

В результате работы программы формируется массив типа double троек чисел, каждая из которых представляет координату (два числа) и значение двумерной функции в этой точке. В обращении к данной Си-функции ей передается лишь адрес этого массива. Выделяет память для массива точек профиля и при необходимости увеличивает ее сама Си-функция. Освободить же память необходимо после использования значений координат точек профиля с помощью Си-функции tkb_vmn_buf_free.

Приведем вид обращения и описание параметров наиболее важной Си-функции реализации команды tkxl_profile1, выполняющей поиск всех точек пересечения данного фрагмента изолинии с отрезком, по которому строится профиль:

```
int tkxl_profile1(double* xy_prof, int i_prof,
       double* izoline_coords, int np_izoline_coords,
       double*x, double*y, int np_xy,double xbeg,
       double ybeg, double xend, double yend, double eps,
```

```
        int ix, int iy ),
```

где double∗xy_prof — указатель на массив координат точек пересечения изолинии с отрезком;

double∗izoline_coords — указатель на массив координат фрагмента изолинии в оперативной памяти;

int np_izoline_coords — число точек во фрагменте изолинии;

double∗x, double∗y — указатели на массивы абсцисс и ординат точек разбиения отрезка;

int np_xy — размерность массивов координат точек разбиения отрезка;

double xbeg, double ybeg, double xend,double yend — координаты двух концов отрезка, на котором строится профиль;

double eps — степень близости точек;

int ix = 0 , если абсциссы точек не совпадают, 1, если совпадают;

int iy = 0 , если ординаты точек не совпадают, 1, если совпадают.

Си-функция возвращает число точек пересечения. Если нет точек пересечения с изолиниями, то Си-функция возвращает число точек пересечения равное 0 и по указателю ∗xy_prof можно определить значение расстояния до ближайшей изолинии данного уровня.

4. Реконструкция двумерной функции. Пусть задано семейство изолиний различных уровней. Необходимо построить соответствующую двумерную функцию $f(x, y)$ в узлах заданной неравномерной сетки: $\{x_0, x_1, \ldots, x_n\} \times \{y_0, y_1, \ldots, y_m\}$.

Возьмем произвольный узел x_i сетки по оси Ox и найдем точки пересечения прямой $x = x_i$ с изолиниями всех уровней. С помощью Си-функции tkxl_profile_all , описанной выше, получим профиль вдоль отрезка $[(x_i, y_0); (x_i, y_m)]$ с экстраполяцией по ближайшему известному. Си-функция net_section линейной интерполяцией вычисляет значения профиля в точках сетки по оси Oy и вычисляет относительную погрешность интерполяции. Аналогично поступаем и для остальных узлов сетки по оси Ox.

Приведем вид команды реконструкции матрицы по известным изолиниям:

```
  izoline_matr #fn1 #fn2 nx ny x y matrix delta,
```

где #fn1 — номер файла с координатами точек изолиний двумерной функции, открытого ранее на чтение командой fopen;

#fn2 - номер файла с атрибутами изолиний (включая значения уровней), открытого ранее на чтение командой fopen;

nx, ny — размерности массивов узлов сетки по осям x и y;

x, y — массивы узлов сетки по осям x и y соответственно;

matrix — массив значений типа double точек реконструированной двумерной функции в узлах сетки;

delta — массив типа int значений погрешности реконструированной двумерной функции в узлах сетки.

Описание массива delta приведено ниже. Рассмотренная команда основана на Си-функции tkxl_matr, которая имеет вид:

```
int tkxl_matr(int f1, int f2, double*x, double*y, int nx,
              int ny, double*field, int*delta),
```

где f1, f2 — номера файлов с координатами точек и атрибутами изолиний;

doudle∗x — указатель на массив узлов неравномерной сетки по x, упорядоченных по возрастанию;

doudle∗y — указатель на массив узлов неравномерной сетки по y, упорядоченных по возрастанию;

int nx, int ny — размерность сетки по осям x и y соответственно;

double∗field — указатель реконструированного массива значений двумерной функции в точках сетки (nx-1)∗j+i, где i – индекс по оси x, j – индекс по оси y;

int∗delta — указатель массива значений погрешности реконструированной двумерной функции в соответствующих узлах сетки.

Данная Си-функция многократно обращается в цикле к Си-функции построения профиля tkxl_profilre_all, описанной выше, и затем, используя построенный профиль, линейной интерполяцией находит значения двумерной функции в узлах сетки. Последнее осуществляет Си-функция net_section, имеющая вид:

```
void net_section(double*xy, double*prof, int n, double*x,
            double**prof1, int nx, int**del, double eps),
```

где double∗xy — указатель массива упорядоченных по возрастанию аргументов профиля;

double∗prof — указатель массива значений профиля для них;

int n — число точек в профиле;

double∗x — указатель массива точек сетки;

double**prof1 — адрес указателя массива значений профиля в точках сетки;
int nx — число точек сетки;
int**del — адрес указателя массива значений погрешности вычислений в точках сетки;
double eps — степень близости точек.

Эта Си-функция для каждого узла сетки находит ближайшие точки профиля слева и справа и линейной интерполяцией вычисляет значение в данном узле сетки. Заметим, что после работы Си-функции tkxl_matr необходимо командой tkb_vmn_buf_free освободить память, заказанную для массива точек профиля.

СПИСОК ЛИТЕРАТУРЫ

1. *Арушанян О.Б., Богомолов Н.А., Волченскова Н.А., Ковалев А.Д., Синицын М.Н.* Общее описание Главного управляющего языка базовой технологии создания систем обработки и отображения сложно организованной информации // Библиотеки и пакеты прикладных программ. М.: Изд-во Моск. ун-та, 1996.
2. Автоматизация геолого-маркшейдерских графических работ. М.: Недра, 1990.

Поступила в редакцию
21.01.2000