УДК 519.644

Адаптивная процедура вычисления двукратных интегралов

Т. А. Ломоносов*

Адаптивная процедура численного интегрирования широко известна, когда подынтегральная функция есть функция одного переменного. Задача численного интегрирования в многомерных областях куда менее освещена, особенно, когда речь идет о разработке адаптивных процедур.

Рассмотрена задача вычисления двойных интегралов с заданной точностью, простроена адаптивная процедура и описана ее эффективная рекурсивная реализация методом объектно-ориентированного программирования. При этом проанализированы квадратурные формулы классов PB (положительно на границе) и PI (положительно внутри), т. е. все веса положительны, а все узлы находятся строго внутри области интегрирования, либо, как в первом случае, хотя бы один узел находится на границе области.

Соответственно, в первом случае есть надежда на то, что некоторые узлы можно использовать повторно, а во втором есть полная гарантия того, что результат получится в пределах заданной точности.

Рассматриваемая задача является актуальной и позволит с большей долей точности решать уравнения математической физики с помощью метода конечных элементов. Настоящая работа преследует следующую цель: создать эффективный алгоритм численного интегрирования в двумерном случае, который будет давать приемлемые результаты на довольно широком классе функций. Практически интересные интегралы, приведенные в настоящей работе, имеют физический смысл и находят свое применение в области лазерной рефрактографии.

Ключевые слова: двумерное численное интегрирование, адаптивная процедура, быстро осциллирующие функции.

Введение

В области лазерной рефрактографии [1] возникает задача вычисления семейства двукратных интегралов вида:

$$A(x,y,z) = -\frac{ikz}{2\pi}\int\limits_{-\infty}^{\infty}\int\limits_{-\infty}^{\infty}\frac{e^{ikR(x,y,z)}}{R^2(x,y,z)}A_0(\xi,\eta)e^{-ikln(\xi,\eta)}\,d\xi d\eta, \quad (1)$$

где i — мнимая единица;

$R(x,y,z) = \sqrt{(x-\xi)^2+(y-\eta)^2+z^2}$; $k >> 1$, l и A_0, n — заданные физические параметры и функции. Функция A_0 быстро убывает на бесконечности.

Рассмотрим алгоритм адаптивного вычисления двукратных интегралов, обобщающий на двумерный случай известную вертикальную процедуру из [2]. Используем формулы численного интегрирования по треугольникам алгебраических порядков точности $d = 3, 5, 7, 11$ из [3 — 6]. Опишем эффективную рекурсивную программную реализацию алгоритма. Покажем применение для вычисления серий интегралов типа (1) и продемонстрируем эффективность алгоритма.

Используемые квадратурные формулы

Широко известна квадратурная формула по треугольнику T для $d = 3$. Пусть a_i — вершины; a_{ij} — середины сторон; a_{123} — центр тяжести T, а $|T|$ — площадь T. Тогда формула такова (см., например [3]):

$$\int_T f(x)dx \approx \frac{|T|}{60}\left(3\sum_{i=1}^{3} f(a_i) + 8\sum_{1\le i<j\le 3} f(a_{ij}) + 27 f(a_{123})\right).$$

Однако апробация данной формулы для вычисления интегралов типа (1) приводит к чрезмерно большим затратам времени, поэтому также были применены фор-

* aladdin93@bk.ru

мулы для $d = 5, 7, 11$ из [4 — 6], что последовательно привело к заметному уменьшению времени расчета.

Пусть Δ — треугольник с вершинами $V_0 = (0, 0)$, $V_1 = (1, 0)$, $V_2 = (0, 1)$. Следуя [6], построим квадратурную формулу вида:

$$\int_\Delta f(x,y)dxdy \approx \sum_{k=1}^{N_0}\lambda_{k,0} f(x_{k,0}, y_{k,0}) + \sum_{k=1}^{N_1}\lambda_{k,1} f(x_{k,1}, 0) + \sum_{k=1}^{N_2}\lambda_{k,2} f(0, y_{k,2}) + \sum_{k=1}^{N_3}\lambda_{k,3} f(x_{k,3}, 1-x_{k,3}) + \mu_0 f(V_0) + \mu_1 f(V_1) + \mu_2 f(V_2). \quad (2)$$

На многочленах одной переменной введем линейные функционалы:

$$\langle \mathcal{L}_s, g\rangle = \int_\Delta g(x) w_s(x,y)dxdy - \sum_{k=1}^{N_0}\lambda_{k,0} w_s(x_{k,0}, y_{k,0}) g(x_{k,0}),\ s = 1,3;$$

$$\langle \mathcal{L}_2, g\rangle = \int_\Delta g(y) w_1(y,x)dxdy - \sum_{k=1}^{N_0}\lambda_{k,0} w_1(y_{k,0}, x_{k,0}) g(y_{k,0}),$$

где $w_1(x, y) = x(1 - x - y)$; $w_3(x, y) = xy$; $\lambda_{k,j}$, $x_{k,0}$, $y_{k,0}$ — такие же, как и в (2).

Алгоритм построения квадратурной формулы порядка точности d таков:

Шаг 1. Построим квадратурную формулу порядка точности $d - 3$ для вычисления интеграла с весом $xy(1 - x - y)$, все узлы которой лежат внутри Δ и положим $\lambda_{k,0} = \lambda^*_{k,0} / [x_{k,0} y_{k,0} (1 - x_{k,0} - y_{k,0})]$.

Шаг 2. Построим квадратурные формулы:

$$\langle \mathcal{L}_s, g\rangle = \sum_{k=1}^{n}\lambda^*_{k,s} g(x_{k,s}),\ s = 1,3, \langle \mathcal{L}_2, g\rangle = \sum_{k=1}^{n}\lambda^*_{k,2} g(y_{k,2}),$$

где $g(x)$ — многочлен одной переменной степени не выше $2n - 3$, где $d = 2n - 1$. Они дают узлы квадратурной формулы на границе Δ (кроме вершин), причем следует положить:

$$\lambda_{k,s} = \frac{\lambda^*_{k,s}}{x_{k,s}(1 - x_{k,s})};\quad s = 1,3;\quad \lambda_{k,2} = \frac{\lambda^*_{k,2}}{y_{k,2}(1 - y_{k,2})}.$$

Шаг 3. Веса μ_0, μ_1, μ_2 можно найти из условия обращения формулы (2) в равенство на функциях $f(x, y) - 1, x, y$ и решения получившейся системы линейных алгебраических уравнений.

В частном случае этот алгоритм можно применить для получения симметричных квадратурных формул Гаусса–Лобатто, узлы которых инвариантны относительно перестановки $(x, y, 1 - y)$. Они выглядят следующим образом:

$$\int_\Delta f(x,y)dxdy \approx \sum_{k=1}^{M_0} A_k [f(u_{k,0}, v_{k,0}) + f(v_{k,0}, w_{k,0}) + f(w_{k,0}, u_{k,0})] + \sum_{k=1}^{M_1} B_k [f(u_{k,1}, 0) + f(0, 1-u_{k,1}) + f(1-u_{k,1}, u_k)] + C[f(V_0) + f(V_1) + f(V_2)],$$

где $w_{k,0} = 1 - u_{k,0} - v_{k,0}$.

Так, формула для $d = 5$ с 12 узлами отвечает параметрам $M_0 = 1$; $M_1 = 2$; $u_{1,0} = v_{1,0} = \frac{1}{21}(7 - \sqrt{7})$; $A_1 = \frac{7}{720}(14 - \sqrt{7})$; $u_{1,k} = \frac{1}{42}(21 + (-1)^k \sqrt{21(4\sqrt{7} - 7)})$ при $k = 1, 2$, $B_1 = B_2 = \frac{1}{720}(4 + \sqrt{7})$, $C = \frac{1}{720}(8 - \sqrt{7})$.

В [4] рассматривается построение квадратурных формул для интегралов вида $\int_\Omega f(x,y)dxdy$, где Ω — некоторая область на плоскости. Авторы заинтересованы в квадратурных формулах, которые используют менее чем m_2 узлов и имеют алгебраический порядок точности $2m - 1$, где m — заданное натуральное число. Сначала изучается случай Ω-прямоугольника, а затем полученный результат переносится на случай треугольника. Данный переход осуществляется путем выбора четверки ортогональных относительно него многочленов, в то время как для прямоугольника достаточно подобрать пару.

Введем несколько вспомогательных определений. Напомним сначала, что многочлен $P_m(x, y)$ степени m (по совокупности переменных) называется ортогональным относительно области Ω, если равенство $\int_\Omega P_m(x,y) Q_{m-1}(x,y)dxdy = 0$ верно для всех многочленов Q_{m-1} степени не выше $m - 1$.

Определение. *Многочлены и имеют конечный общий корень, если и .*

Определение. *Многочлены $S_1(x, y)$ и $S_2(x, y)$ степени m имеют общий корень в бесконечности, если $\exists y^* \in \mathbb{C} : S_1(\cdot, y^*) \not\equiv 0, S_2(\cdot, y^*) \not\equiv 0$, и многочлены $S_1(\cdot, y^*)$ и $S_2(\cdot, y^*)$ имеют степень $m - 1$.*

В [4] даны следующие две теоремы.

Пусть $\Omega = (a, b)\times(c, d)$.

Теорема 1. *Пусть S_1 и S_2 — многочлены двух переменных степени m, ортогональные относительно Ω и имеющие ровно m^2 попарно различных общих корней v_k, причем ни один из них не является корнем в бесконечности. Тогда можно найти такие постоянные $A_k > 0$, что*:

$$\int_\Omega P_{2m-1}(x,y)dxdy = \sum_{k=1}^{m^2} A_k P_{2m-1}(v_k)$$

для любых многочленов P_{m-1} степени не выше $2m - 1$.

Ортогональными относительно квадрата $\Omega = (-1, 1)^2$ являются, например, многочлены $P_m(x, y) = L_k(x)L_{m-k}(y)$, где $L_s(t)$ — многочлен Лежандра степени s, $0 \le k \le m$.

Теорема 2. *При $m \ge 3$ можно построить квадратурную формулу порядка точности $d = 2m - 1$, использующую менее m^2 узлов и с положительными весами.*

Доказательство опирается на использование ортогональных многочленов и применение теоремы (1).

При построении соответствующей квадратурной формулы с положительными весами для Ω-прямоугольного треугольника необходимо наложить дополнительные условия. Во-первых, общие корни многочленов S_1 и S_2 из теоремы 1 должны быть вещественны.

Далее, пусть $J \subset \mathbb{R}$ — наибольший интервал такой, что если $\lambda \in J$, то многочлены $S_4(x, y; \lambda) = S_1(x, y) + \lambda S_3(x, y)$ и $S_2(x, y)$ также удовлетворяют теореме 1 для $\Omega = \Delta$, где S_3 — некоторый ортогональный относительно Ω многочлен, причем множество корней S_3 не совпадает с множествами корней ни S_1, ни S_2 (но может пересекаться с ними). Тогда достаточные условия для построения квадратурной формулы со свойствами из теоремы 2 следующие:

1. Веса $A_k(\lambda)$, полученные по теореме 1 для $S_4(x, y; \lambda)$ и $S_2(x, y)$, положительны для любого $\lambda \in J$.

2. Если $\lambda^* \in \partial J$, то многочлены $S_4(x, y; \lambda^*)$ и $S_2(x, y)$ имеют конечное число общих корней, и они конечны в смысле определения 1.

3. Общие корни многочленов $S_4(x, y; \lambda)$ и $S_2(x, y)$ как функции от λ непрерывны при $\lambda = \lambda^*$, а также существует $\lim\limits_{\lambda \to \lambda^*, \lambda \in J} A_k(\lambda)$.

В табл. 1 даны узлы и веса формулы с $d = 7$ для треугольника $\Omega = \Delta$, такого же, как и выше [4, с. 813].

Таблица 1

Узлы и веса квадратурной формулы с $d = 7$

Координаты узлов	Веса
(0,064634109801617; 0,064634109801617)	0,0263321501360460
(0,250478764260821; 0,250478764260821)	0,0666750609902085
(0,405288113134598; 0,405288113134598)	0,0598398472297514
(0,483428507060240; 0,405288113134598)	0,0302244308027287
(0,0490241549057468; 0,312418129002285)	0,0387139102462897
(0,312418129002285; 0,0490241549057468)	0,0387139102462897
(0,0272654917225016; 0,649829918830148)	0,0223103130816147
(0,649829918830148; 0,0272654917225016)	0,0223103130816147
(0,00748092005042521; 0,922929224698637)	0,00930956404694027
(0,922929224698637; 0,00748092005042521)	0,00930956404694027
(0,166718687651425; 0,775796880494268)	0,0365382927009296
(0,775796880494268; 0,166718687651425)	0,0365382927009296
(0,151969575382297; 0,569101341800312)	0,0515921753448585
(0,569101341800312; 0,151969575382297)	0,0515921753448585

4. Легко видеть, что формула порядка точности d для заданного треугольника посредством аффинного преобразования преобразуется в формулу того же порядка точности для любого другого треугольника.

В [5] выбирается равносторонний треугольник Δ, вписанный в единичную окружность, с одной из медиан на оси абсцисс. Он является D_3-областью, т.е. инвариантен относительно поворота на угол $2\pi/3$ вокруг начала координат и отражения относительно оси абсцисс. Важную роль играет переход к полярным координатам $x = \rho\cos\varphi$, $y = \rho\sin\varphi$.

Далее рассмотрим функции $f(\rho, \varphi)$ с $0 \le \rho \le 1$ и 2π-периодические по φ.

Определение. *Функция $f(\rho, \varphi)$ называется D_3-функцией, если она удовлетворяет свойству*

$$f(\rho,\varphi) = f(\rho,\varphi+\frac{2\pi}{3}) = f(\rho,-\varphi).$$

Определение. *Квадратурная формула Q, рассматриваемая как элемент пространства $(C(\Delta))^*$, называется D_3-формулой, если для любых функций $f \in C(\Delta)$ верны равенства*

$$\langle Q, f(\rho,\varphi)\rangle = \left\langle Q, f(\rho,\varphi+\frac{2\pi}{3})\right\rangle = \langle Q, f(\rho,-\varphi)\rangle.$$

Определение. *Базовой D_3-формулой называется функционал $Q = (\rho, \alpha)$, действующий по правилу*

$$\langle Q(\rho,\alpha), f\rangle = \frac{1}{6}\sum_{j=1}^{3}\left[f(\rho,\alpha+\frac{2\pi j}{3}) + f(\rho,-\alpha+\frac{2\pi j}{3})\right].$$

Эта формула использует шесть значений f при $\rho \ne 0$ и $\cos 3\alpha \ne 1$, три значения при $\rho \ne 0$ и $\cos 3\alpha = 1$ (в узлах, лежащих на трех медианах Δ) и одно значение при $\rho = 0$ (тогда $\langle Q(0, \alpha), f\rangle = f(0, 0)$).

Построим квадратурную D_3-формулу Q, которую представим в виде линейной комбинацией базовых квадратурных формул $Q(\rho_i, \alpha_i)$:

$$\langle Q, f\rangle = \sum_{i=1}^{n} w_i \langle Q(\rho_i,\alpha_i), f\rangle,$$

где (ρ_i, α_i) — полярные координаты узлов квадратурной формулы; w_i — соответствующие им веса.

Чтобы получить формулу порядка точности d для нахождения параметров w_i, ρ_i, α_i, надо решить систему моментных уравнений:

$$\left\langle Q, \rho^j e^{ik\varphi}\right\rangle = v_{j,k} = \frac{1}{|\Delta|}\int_{\Delta}\rho^{j+1}e^{ik\varphi}\,d\rho d\varphi.$$

где $0 \le j \le d$, $|k| \le j$, $k + j$ — четно.

Переход к полярным координатам дает большое упрощение при конструировании формул, благодаря следующим результатам [5].

Теорема. *Если л не кратно 3, то* $v_{j,k} = 0$ *и* $\langle Q, \rho^j e^{ik\varphi}\rangle = 0$ *при любом j. Для любых j, k имеем* $v_{j,-k} = v_{j,k}$ *и* $\langle Q, \rho^j e^{-ik\varphi}\rangle = \langle Q, \rho^j e^{ik\varphi}\rangle$.

Любая квадратурная D^3-формула может быть записана в виде:

$$\langle Q, f\rangle = w_0 f(0,0) + \sum_{i=1}^{n_1} w_i \langle Q(\rho_i, 0), f\rangle + \sum_{i=n_1+1}^{n_1+n_2} w_i \langle Q(\rho_i, \alpha_i), f\rangle,$$

где $\cos 3\alpha_i \neq 1$ при $i \geq n_1 + 1$. Количество ее узлов равно $\nu(Q) = n_0 + 3n_1 + 6n_2$, где $n_0 = 1$ при $w_0 \neq 0$, либо $n_0 = 0$ при $w_0 = 0$. Для нахождения параметров $\{w_i\}_{i=n_0}^{n_1+n_2}$, $\{\rho_i\}_{i=1}^{n_1+n_2}$, $\{\alpha_i\}_{i=n_1+1}^{n_1+n_2}$ следует решить систему нелинейных уравнений:

$$w_0 + \sum_{i=1}^{n_1} w_i + \sum_{i=n_1+1}^{n_1+n_2} w_i = \nu_{0,0} \equiv 1;$$

$$\sum_{i=1}^{n_1} w_i \rho_i^j + \sum_{i=n_1+1}^{n_1+n_2} w_i \rho_i^j \cos 3k\alpha_i = \nu_{j,3k},$$

где $2 \leq j \leq d$, $0 \leq 3k \leq j$, $k + j$ — четно.

В табл. 2 приведены узлы формулы 11-го порядка точности и их кратности, а в табл. 3 — веса, соответствующие узлам из строк табл. 2 [5]. Для узлов указаны пары ζ_1, ζ_2 их барицентрических координат $(\zeta_1, \zeta_2, \zeta_3)$; напомним, что $\zeta_3 = 1 - \zeta_1 - \zeta_2$. Здесь $\nu(Q) = 28$.

Кратность m данного узла означает, сколько различных узлов он порождает при последовательных поворотах на угол $2\pi/3$ вокруг начала координат и отражениях относительно оси абсцисс. Так, узел с $(\zeta_1, \zeta_2, \zeta_3) = (1/3,1/3,1/3)$ является центром тяжести Δ и при указанных преобразованиях переходит сам в себя, поэтому для него $m = 1$.

Таблица 2

Узлы квадратурной формулы с $d = 11$ в барицентрических координатах и их кратности

ζ_1	ζ_2	m
0,3333333333333333	0,3333333333333333	1
0,09480217181434233	0,2598914092828833	3
0,8114249947041546	0,09428750264792270	3
0,01072644996557060	0,49463677501721470	3
0,5853132347709715	0,2073433826145142	3
0,1221843885990187	0,4389078057004907	3
0	0,858870281826364	6
0,04484167758913055	0,06779376548825502	6

Таблица 3

Веса квадратурной формулы с $d = 11$

w_i
0,087977301116221900
0,026232934661208570
0,11424471598180600
0,05656634416839376
0,2164790926342230
0,20798741611661160
0,04174302699803440
0,2463378925757316

Построение адаптивного алгоритма

Реализованный в работе адаптивный алгоритм является обобщением на двумерный случай вертикальной процедуры из [2]. Пусть Ω — многоугольник, разбитый на треугольники, и требуется вычислить интеграл по треугольнику T. Формула правила Рунге практической оценки погрешности для прямоугольного равнобедренного треугольника T_h с катетом h такова:

$$R_h = \frac{|I_{h/\sqrt{2}} - I_h|}{2^{(d+1)/2} - 1},$$

где I_h — квадратурная формула для вычисления интеграла по T_h алгебраического порядка точности d; $I_{h/\sqrt{2}}$ — ее составной вариант по объединению двух треугольников с катетами $h/\sqrt{2}$, полученных разбиением T_h пополам с помощью высоты, проведенной из вершины прямого угла.

Алгоритм использует переход к вычислению интеграла по тому подтреугольнику, который лежит слева от этой высоты. Оба подтреугольника можно получить из исходного с помощью поворота и сжатия. Пусть x_v — вершина треугольника, x_a — вектор, направленный от вершины прямого угла к середине гипотенузы, x — текущая точка треугольника. Тогда соответствующая точка левого (правого) треугольника получается по формуле:

$$Ux = \begin{pmatrix} -0,5 & \pm 0,5 \\ \mp 0,5 & -0,5 \end{pmatrix}(x - x_v) + x_v + x_a,$$

где в матрице берутся верхние (нижние) знаки.

В начале работы алгоритм обращается к интегрированию по всему треугольнику T.

Исходная область интегрирования может быть неограниченной, в частности, совпадать с $\mathbb{R}^2$. В послед-

нем случае для вычисления интеграла с точностью $\varepsilon > 0$ сначала следует подобрать многоугольник Ω_ε, такой, что модуль интеграла по $\mathbb{R}^2 \setminus \Omega$ меньше $\varepsilon/2$, а интеграл по Ω_ε вычислять с точностью $\varepsilon/2$.

Программная реализация

Описанный выше алгоритм является рекурсивным. Это обеспечивает возможность его компактной реализации на языке C++ с применением методов объектно-ориентированного программирования (ООП). Вычисление интегралов реализовано в виде шаблонного класса, что позволяет легко использовать программу как для вещественно-, так и комплекснозначных функций. Класс TriangleIntegral представляет собой набор членов и методов для вычисления интегралов в заданном треугольнике.

```
template <class T> class TriangleIntegral {
private: double A1, double B1, double C1, bool Left;
RPoint GetUpper(RPoint X, double xv, double yv, double AscX,
double AscY);
RPoint GetLower(RPoint Y, double xv, double yv, double AscX,
double AscY);
public: typedef T(*OscFunc3)(RPoint X);
int FuncCalls, OscFunc3 osf, double Cathetus;
RPoint GetStartLower(RPoint X);
RPoint GetStartUpper(RPoint X);
RPoint GetStartLowerInv(RPoint X);
TriangleIntegral(OscFunc3 f, double c);
void Trapeze(RPoint A, RPoint B, RPoint C, T fA, T fB, T fC,
double eps, T &I);
void ThirdDegree(RPoint Point[7], T fPoint[7], double eps,
T &I);
void FifthDegree(RPoint Point[12], T fPoint[12], double eps,
T &I);
void SeventhDegree(RPoint Point[14], T fPoint[14], double eps,
T &I, RPoint Vertex[3]); void EleventhDegree(RPoint Point[14],
T fPoint[14], double eps, T &I, RPoint Vertex[3]);};
```

Методы класса GetUpper и GetLower выполняют пересчет координат узлов, изначально заданных в глобальном массиве для стандартного треугольника Δ с вершинами (0, 0), (1, 0), (0, 1), в координаты соответствующих узлов в преобразованных треугольниках. Методы ThirdDegree, FifthDegree, SeventhDegree и EleventhDegree реализуют двумерные квадратурные формулы 3-го, 5-го, 7-го и 11-го порядка точности соответственно. Программная реализация последнего метода класса EleventhDegree представлена ниже.

```
template <class T> void
TriangleIntegral<T>::EleventhDegree(RPoint Point[31],T
fPoint[31], double eps, T &I, RPoint Vertex[3]) {
double h = Length(Vertex[0],Vertex[1]);
RPoint V[3];
RPoint Median[3];
Median[0] = DistanceX(0.5, Vertex[0],Vertex[1]);
Median[1] = DistanceX(0.5, Vertex[1],Vertex[2]);
Median[2] = DistanceX(0.5, Vertex[2],Vertex[0]);
bool InCircle = false;
for (int i = 0; i <= 2; i++)
 if (Median[i].norm2() <= Cath)
 { InCircle = true; }
if (!InCircle) return;
RPoint D[56];
T fD[56];
Min++;
double AbsX = Median[1].x - Vertex[0].x;
doubleAbsY =Median[1].y - Vertex[0].y;
for (int i = 0; i < 28; i++) {
 D[i] =
GetLower(Point[i],Vertex[0].x,Vertex[0].y,AbsX,AbsY);
 fD[i] = osf(D[i]);
}
for (int i = 0; i < 28; i++) {
 D[i+28] =
GetUpper(Point[i],Vertex[0].x,Vertex[0].y,AbsX,AbsY);
 fD[i+28] = osf(D[i+28]);
}
FuncCalls += 56;
T Ih = 0;
for (int i = 0; i < 28; i++)
 Ih += weight11[i]*fPoint[i];
Ih *= (h*h)/2;
T Ih2A = 0;
for (int i = 0; i < 28; i++)
 Ih2A += weight11[i]*fD[i];
Ih2A *= (h*h)/4.0;
T Ih2B = 0;
for (int i = 0; i < 28; i++)
 Ih2B += weight11[i]*fD[i+28];
Ih2B *= (h*h)/4.0;
T r = Ih - Ih2A - Ih2B;
if (abs(r) < 63*eps/(Cathetus*Cathetus))
 I += Ih2A + Ih2B;
else {
 for (int i = 0; i < 28;i++) {
 Point[i] = D[i];
 fPoint[i] = fD[i];
}
for (int i = 0; i <= 3; i++) {
 V[i] =
GetLower(Vertex[i],Vertex[0].x,Vertex[0].y,AbsX,AbsY); }
EleventhDegree(Point,fPoint,eps,I,V);
for (int i = 0; i < 28;i++) {
Point[i] = D[i+28];
fPoint[i] = fD[i+28];
}
for (int i = 0; i < 3; i++) {
V[i] =
GetUpper(Vertex[i],Vertex[0].x,Vertex[0].y,AbsX,AbsY);
}
EleventhDegree(Point, fPoint, eps, I,V);
}
}
```

В данной реализации метода в качестве области интегрирования выбирается круг, поскольку подынтегральная функция в (1) имеет удобный вид для интегрирования в полярных координатах.

Сама процедура является основной: ее рекурсивное выполнение обеспечивает существенную компактность

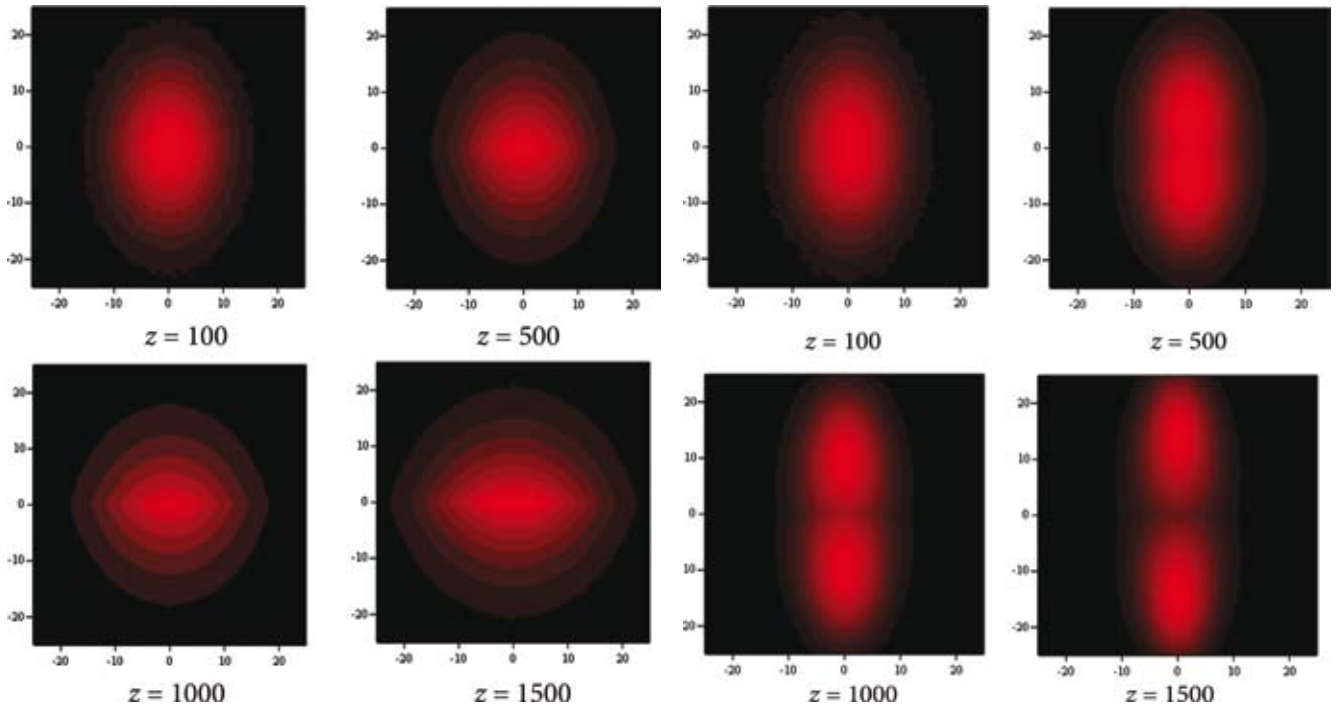

Рис. 1. Круглый пучок с цилиндрической неоднородностью с параметрами $a_1 = a_2 = 5$ мм; левый график отвечает выбору знака «+», а правый — знака «–»

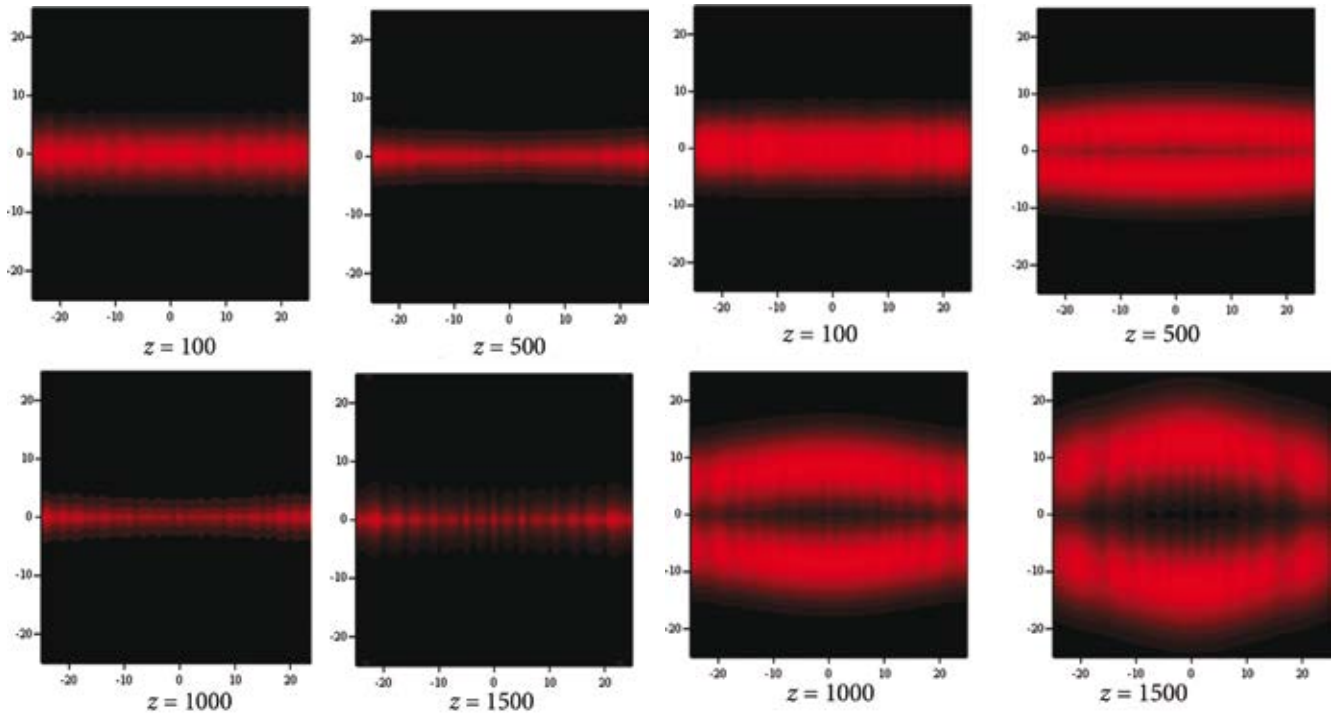

Рис. 2. Плоский пучок с цилиндрической неоднородностью с параметрами $a_1 = 30$ мм, $a_2 = 5$ мм; левый график отвечает выбору знака «+», а правый — знака «–»

кода по сравнению с нерекурсивными аналогами. Она интуитивно понятна даже тем, кто не знаком с языком.

Пустые строки используются как визуальный разделитель разных блоков алгоритма: выделяется блок инициализации, за ним идет блок проверки принадлежности элементарного треугольника кругу, по которому идет интегрирование, и т. д.

Эта же компактность обеспечивает (при необходимости) достаточно простой переход к использованию квадратурных формул более высокого порядка точности.

Таблица 4

Относительное время работы алгоритма при разных значениях *z* в зависимости от порядка точности *d*

z,мм	3	5	7	11
1000	1	4,4	14,1	24,2
800	1	7,4	16,2	29,6
500	1	10,6	25,1	45,3
300	1	13,3	31,9	60,2
Суммарно	1	11,2	27,0	50,0

Благодаря одному из классических принципов ООП, можно создать классы-наследники данного класса для разных областей, и проводить триангуляцию произвольной области. Например, можно создать класс, который будет вычислять интеграл по квадрату.

```
template<classT>classSquareIntegral:publicTriangleIntegral<T>
{
private: double Left, Top, Right, Bottom;
public: SquareIntegral(double l,double t,double r,double b,TriangleIntegral<T>::OscFunc3 f, double C): TriangleIntegral<T>(f,C) {
Left = l; Top = t; Right = r; Bottom = b;
}
T computeIntegralTrapeze(double eps);
T computeIntegralFifth(double eps);
T computeIntegralSeventh(double eps);
T computeIntegralEleventh(double eps);
}
```

Список формальных параметров процедуры вычисления значительно отличается от такового в базовом классе и содержит только желаемую точность. Поэтому интерфейс этого класса дружествен по отношению к пользователю. Естественно, метод базового класса EleventhDegree вызывается внутри метода computeIntegralEleventh.

Результаты вычислений

Приведем результаты вычисления интегралов типа (1) при нескольких *z*. Значения *x* и *y* меняются от 0 до 2,5 мм с шагом 0,1 мм. Расчеты выполнены с точностью $\varepsilon_0 = 0{,}005$. Значения параметров: $k = 2\pi/\lambda$, $\lambda = 0{,}6328\cdot10^{-3}$ мм.

В табл. 4 представлено относительное время вычислений интеграла (1) при разных *z* в зависимости от *d* для довольно типичных функций $A_0(\xi,\eta) = \exp\{-\frac{\xi^2+\eta^2}{w^2}\}$ и $n(\xi,\eta) = n_0 - \Delta n \exp\{-\frac{\xi^2+\eta^2}{a^2}\}$, где $w = 1{,}5$ мм, $a = 5$ мм, $n_0 = 1$, $\Delta n = 0{,}001$ при $l = 0{,}5$ мм. Для формулы с $d = 3$ время условно принято равным 1, а для формул с другими *d* указан коэффициент его уменьшения по отношению к случаю $d = 3$. В частности, переход от $d = 3$ к $d = 11$ многократно сокращает время вычислений.

Проведены также расчеты с прежней A_0 и более сложной:

$$n(\xi,\eta) = n_0 \pm \Delta n \exp\{-\frac{\xi^2}{a_1^2} - \frac{(\eta-3)^2}{a_2^2}\}$$

при $l = 5$ мм мм. Результаты представлены на рис. 1, 2.

Автор признателен проф. А.А. Злотнику за постановку задачи и постоянную помощь в работе.

Исследование выполнено при поддержке гранта Российского научного фонда (соглашение № 14-11-00306).

Литература

1. **Евтихиева О.А., Расковская И.Л., Ринкевичюс Б.С.** Лазерная рефрактография. М: Физматлит, 2008.

2. **Бахвалов Н.С., Жидков Н.П., Кобельков Г.М.** Численные методы. М.: Бином, Лаборатория знаний, 2011.

3. **Сьярле Ф.** Метод конечных элементов для эллиптических задач. М: Мир, 1980.

4. **Franke R.** Obtaining cubatures for rectangles and other planar regions by using orthogonal polynomial // Math. Comp. 1971. V. 25. P. 803 — 813.

5. **Lyness J.N., Jespersen D.** Moderate degree symmetric quadrature rules for the triangle // J. Inst. Math. Appl. 1975. V. 15. N 1. P. 19 — 32.

6. **Yuan Xu.** On Gauss-Lobatto integration on the triangle // SIAM J. Numer. Anal. 2011. V. 49. P. 541 — 548.

Статья поступила в редакцию 17.02.2016