УДК 517.55, 519.6

М. В. Бураченко

АЛГОРИТМ ПОНИЖЕНИЯ КРАТНОСТИ ИНТЕГРАЛА РАЦИОНАЛЬНОЙ АЛГЕБРАИЧЕСКИ ТОЧНОЙ ДИФФЕРЕНЦИАЛЬНОЙ ФОРМЫ С КВАЗИЭЛЛИПТИЧЕСКИМ ЗНАМЕНАТЕЛЕМ

Изложен метод понижения кратности интеграла рациональной алгебраически точной дифференциальной формы с квазиэллиптическим знаменателем вписан алгоритм и представлена его реализация для системы компьютерной алгебры Maple 9.

В данной статье в качестве основного объекта исследования будем рассматривать интеграл вида

$$I = \int\limits_{R^n} \frac{P(x)}{Q(x)}dx = \int\limits_{R^n} \frac{P(x_1, ..., x_n)}{Q(x_1, ..., x_n)}dx_1...dx_n \ , \quad (1)$$

где $P(x)$ и $Q(x)$ – полиномы. Предполагается, что Q не обращается в нуль на R^n и интеграл (1) сходится абсолютно. Такие интегралы встречаются в различных областях математики и теоретической физики. В частности, к ним приводятся фейнмановские интегралы в квантовой теории поля [1], [2]; некоторые интегралы такого типа выражают топологические заряды инстантонных полей [3].

В работе [4] была предложена формула для понижения кратности подобных интегралов в случае, когда $Q(x)$ – эллиптический полином, не обращающийся в нуль на R^n, а подынтегральная форма алгебраически точна. В работе [5] этот результат был обобщен на случай квазиэллиптического знаменателя.

В данной статье, развивающей идеи работы [5], излагается метод понижения кратности интеграла (1) в случае квазиэллиптического знаменателя, описывается соответствующий алгоритм и представляется его реализация для системы компьютерной алгебры Maple 9, позволяющей вычислять интеграл (1) до конца в пространстве R^2 и понижать кратность интеграла на единицу в пространстве R^3.

Основные определения и постановка задачи

Введем, следуя [5], некоторые определения и условные обозначения, необходимые для формулировки основного результата.

Ниже будут рассмотрены интегралы вида (1), где P и Q – полиномы с рациональными коэффициентами. Это ограничение на коэффициенты связано с возможностью реализации алгоритмов. С теоретический точки зрения допустимы коэффициенты из любого упорядоченного поля характеристики нуль.

Пусть полином $Q(x)$ представлен в виде

$$Q(x) = \sum_{\alpha \in N^n} c_\alpha x^\alpha = \sum_{(\alpha_1,...,\alpha_n) \in N^n} c_{\alpha_1...\alpha_n} x_1^{\alpha_1} ...x_n^{\alpha_n} \ ,$$

где N – множество натуральных чисел с добавленным нулем. Носителем Q будем называть множество $\mathrm{supp}Q = \{\alpha \in N^n \mid c_\alpha \neq 0\}$.

Многогранником Ньютона $\Delta = \Delta(Q)$, соответствующим полиному Q, называется выпуклая оболочка в R^n для носителя Q.

Срезкой Q_v полинома Q в направлении ковектора $v \in R^{n*}$ (здесь R^{n*} означает сопряженное пространство к R^n) будем называть полином $Q_v = \sum\limits_{\alpha \in \Delta^v} c_\alpha x^\alpha$, где $\Delta^v = \left\{ w \in \Delta : \langle w, v \rangle = \min\limits_{u \in \Delta} \langle u, v \rangle \right\}$ – грань Δ в направлении v.

Определение 1. Полином Q называется квазиэллиптическим, если для любого ненулевого ковектора $v \in R^{n*}$ срезка Q_v не обращается в нуль в торе $(R \setminus \{0\})^n$.

Замечание 1. Так как число граней многогранника Δ конечно, то условие квазиэллиптичности достаточно проверять лишь для конечного числа срезок Q_v.

При дальнейшем изложении будем полагать, что для многогранника Ньютона Δ, соответствующего полиному Q, справедливо следующее предложение:

Предположение 1. Многогранник $\Delta(Q)$ содержит по ребру на каждой координатной оси Ox_i, $1 \le i \le n$.

Каждому многограннику Ньютона Δ можно поставить в соответствие двойственный конический полиэдр (веер) Σ_Δ. Дадим его определение.

С каждым ковектором $v \in R^{n*}$ связана грань Δ^v многогранника Δ, на которой скалярное произведение $\langle v, u \rangle$ минимально на Δ. Два ковектора будем считать эквивалентными, если связанные с ними грани совпадают. Замыкание классов эквивалентности ковекторов, связанных с гранью Δ_i, образует конус в R^{n*}, порожденный элементами решетки Z^{n*}. Этот конус называется двойственным к грани Δ_i. Совокупность конусов, двойственных всем граням многогранника Δ, образует веер Σ_Δ, который называется двойственным к многограннику Δ. Мы будем рассматривать только полные вееры, для которых объединение входящих в них конусов есть все пространство R^{n*}.

Очевидно, что образующими ребер веера Σ_Δ являются внутренние нормали v_j гиперграней $\Delta_j^{(n-1)} \in \Delta$, а набор $v^{j_1}, ..., v^{j_k}$ таких нормалей порождает конус $\sigma\left(v^{j_1}, ..., v^{j_k}\right)$ в Σ_Δ тогда и только

тогда, когда соответствующие грани $\Delta_{j_1}^{(n-1)}, ..., \Delta_{jk}^{(n-1)}$ имеют непустое пересечение $\Delta_{j_1,...,j_k}$. Размерность конуса σ будет дополнительной к размерности грани $\Delta_{j_1,...,j_k}$, т. е. $\dim\sigma = n - \dim\Delta_{j_1,...,j_k}$.

Конус, число ребер которого равно его размерности, назовем симплициальным. Симплициальный k-мерный конус $\sigma\left(v^1, ..., v^k\right)$ называется примитивным, если параллелепипед $\prod = \left\{\sum_{l=1}^{k} \lambda_v v^1, 0 \le \lambda_v < 1\right\}$ не содержит целых точек (точек, все координаты которых целые числа), отличных от точки 0, или, что то же самое, векторы v^1, .., v^k можно дополнить до базиса решетки Z^{n*}. При $k = n$ это условие эквивалентно тому, что матрица, образованная векторами v^1, .., v^k – унимодулярна, т. е. ее определитель равен ± 1. Веер будем называть примитивным, если любой его конус симплициален и примитивен.

Пусть d – число гиперграней многогранника $\Delta(Q)$ и Σ_Δ – веер, двойственный $\Delta\left(Q\right)$. Обозначим через $v^1,...,v^d$ минимально целочисленные образующие ребер веера Σ_Δ. Если веер Σ_Δ не является примитивным, то можно построить примитивный веер $\widetilde{\Sigma}_\Delta$, состоящий из примитивных конусов, причем каждый конус $\widetilde{\Sigma}_\Delta$ лежит внутри некоторого конуса из Σ_Δ, и если конус из Σ_Δ – примитивный, то он не измельчается. Пусть в результате такого подразбиения к векторам v^1, …, v^d добавились векторы v^{d+1}, …, v^{d+r}.

Для каждого вектора v^k, $1 \le k \le d$, определим матрицу $C_k = C_{k_1...k_{n-1}k} = \left(v^{k_1}, ..., v^{k_{n-1}}, v^k\right)$, столбцами которой служат векторы $v^{k_1}, ..., v^{k_{n-1}}$, порождающие вместе с v^k конус в $\widetilde{\Sigma}_\Delta$. Поскольку $\widetilde{\Sigma}_\Delta$ – примитивный веер, то все матрицы C_k унимодулярны, т. е. $\det C_k = \pm 1$, $1 \le k \le d$.

С каждой матрицей C_k свяжем биномиальную замену координат $x = y^{C_k}$, что в терминах координат $\left(v_1{}^k, ..., v_n{}^k\right)$ векторов v^k означает следующее:

$$\begin{cases} x_1 = y_1^{v_1^{k_1}} ... y_{n-1}^{v_1^{k_{n-1}}} y_n^{v_1^k}, \\ ..... \\ x_n = y_1^{v_n^{k_1}} ... y_{n-1}^{v_n^{k_{n-1}}} y_n^{v_n^k}. \end{cases}$$

На подынтегральное выражение в (1) будем смотреть как на дифференциальную форму, и записывать ее в виде

$$\omega = \frac{P(x_1, ..., x_n)}{Q(x_1, ..., x_n)} dx_1 \wedge ... \wedge dx_n. \qquad (2)$$

Определение 2. Дифференциальная форма ω называется алгебраически точной, если существует рациональная $(n-1)$ – форма φ, дифференциал $d\varphi$ которой равен ω. Такую форму φ будем называть рациональной первообразной для формы ω.

Определение 3. Пусть φ – рациональная форма степени p, т. е. $\varphi = \sum_{1 \le i_1 < ... < i_p \le n} \varphi_{i_1...i_p}(x)\, dx_{i_1} \wedge ... \wedge dx_{i_p}$, где коэффициенты $\varphi_{i_1...i_p}(x)$ – рациональные функции переменной $x \in R^n$. Если знаменатели всех функций $\varphi_{i_1...i_p}$ являются степенями квазиэллиптического полинома Q, то мы говорим, что форма φ не растет на бесконечности, когда для каждой гиперграни Δ_j^{n-1} многогранника $\Delta(Q)$ форма $\varphi\left(y^{C_i}\right)$ не имеет полюса на гиперплоскости $y_n = 0$.

Введем обозначения: $\left|v^k\right| = v_1^k + ... + v_n^k$,

$$G_\pm = \left\{\left(y_1, ... y_{n-1}\right) \in R^n : y_1^{\left|v^{k_1}\right|_{-1}} \cdot ... \cdot y_{n-1}^{\left|v^{k_{n-1}}\right|_{-1}} \begin{matrix} > \\ < \end{matrix} 0\right\}.$$

В указанных обозначениях справедлива следующая теорема:

Теорема 1. Пусть Q – квазиэллиптический полином, не обращающийся в нуль на R^n, и интеграл (1) абсолютно сходится. Если дифференциальная форма (2) имеет рациональную первообразную φ, которая не растет на бесконечности, то для интеграла (1) справедлива формула

$$I = (-1)^n \sum_{k=1}^{d+r} \left(1 + (-1)^{|v_k|}\right) \det C_k \left(\int\limits_{G_+} - \int\limits_{G_-} \right) \left[\varphi\left(y^{C_k}\right)\right]\Big|_{y_n = 0}, \qquad (3)$$

где суммирование ведется по всем $(n-1)$-мерным граням $\Delta_k^{n-1}(Q)$.

Доказательство этой теоремы в рамках данной работы не приводится, но его можно найти в [5].

Для случая $n = 2$ формулу (3) можно переписать в виде

$$I = 2 \sum_{k:\left|v^k\right| - \text{четн}} \int_R \left[\varphi\left(y^{C_k}\right)\right]\Big|_{y_2 = 0}.$$

Алгоритм понижения кратности интеграла рациональной алгебраически точной дифференциальной формы с квазиэлептическим знаменателем

На основании вышеизложенного представим алгоритм понижения кратности интеграла (1). Все вспомогательные алгоритмы, выделенные латинскими буквами, будут описаны далее, после основного алгоритма.

QuasiellipticIntegration(P, Q, x)

Input: $P = P(x_1, \dots x_n)$ – полином, зависящий от n переменных, – числитель подынтегральной формы интеграла (1);

$Q = Q(x_1, \dots x_n)$ – полином, зависящий от n переменных, – знаменатель подынтегральной формы интеграла (1);

$x = [x_1, \dots x_n]$ – список переменных;

Output: интеграл вида (3), кратность которого $(n - 1)$;

1) проверка того, что Q не обращается в нуль в R^n:

t_1 : = PolyNot0(Q, x);

if t_1 = false then error `знаменатель обращается в нуль`;

2) проверка квазиэллиптичности полинома Q:

t_2 : = Quasielliptic(Q, x);

if t_2 = false then error `знаменатель не квазиэллиптический`;

3) проверка условия предположения 1:

Δ : = NewtonPolyhedron(Q, x);

if найдется $1 \le I \le n$, такой что не существует $s \in N$, такого что $\left(0, \dots, \underset{i}{s}, \dots, 0\right) \in \Delta$ then error `условие предположения 1 не выполняется`;

4) проверка того, что дифференциальная форма с коэффициентом $\frac{P}{Q}$ алгебраически точная, и вычисление ее рациональной первообразной φ, не растущей на бесконечности:

φ : = RationalPrimitive (P, Q, x) φ: = = RationalPrimitive$\left(Q, P, x\right)$;

if $\varphi = [\underbrace{0, \dots, 0}_{n}]$ then error `для исходной подынтегральной формы не существует рациональной первообразной, не растущей на бесконечности`;

5) проверка абсолютной сходимости интеграла (1):

t_3 = AbsoluteConvergence(P, Q, x);

if t_3 = false then error `интеграл не сходится абсолютно`;

6) Σ : = NormalFan(Q, x) – веер в R^{n*}, двойственный многограннику Δ, конуса которого порождены минимальными целочисленными ковекторами v^k, k = 1,...,d внутренних нормалей к $(n - 1)$-мерным граням Δ_k^{n-1} многогранника Δ;

7) $\widetilde{\Sigma}$: = PrimitiveNormalFan(Σ) – примитивный веер, полученный в результате простого подразбиения веера Σ (Q) добавлением к его образующим ковекторов $v^{d+1} \dots v^{d+r}$;

8) каждому ковектору v^k, k = 1, …, d, сопоставим унимодулярную матрицу $C_k = \left(v^{k_1}, \dots, v^{k_{n-1}}, v^k\right)$, так чтобы ковекторы $v^{k_1}, \dots, v^{k_{n-1}}, v^k$ образовывали конус в $\widetilde{\Sigma}(Q)$;

9) с каждой матрицей C_k, $1 \le k \le d + r$ свяжем биномиальную замену переменных $x = y^{C_k}$;

10) return (интеграл кратности $(n - 1)$, записанный по формуле (2))

end.

Пока интеграл, получившийся в результате работы алгоритма, удовлетворяет условиям теоремы 1 и предположения 1, его кратность можно понижать, используя тот же алгоритм. Но если итоговый интеграл не будет удовлетворять какому-либо из условий, то его можно попытаться вычислить стандартными средствами.

Разберем подробнее вспомогательные алгоритмы, встречающиеся в основном алгоритме, и особенности их реализации в системе компьютерной алгебры Maple.

Построение многогранника Ньютона и двойственного ему веера. При рассмотрении QuasiellipticIntegration, в частности при проверке условий квазиэллиптичности знаменателя, алгебраической точности подынтегральной формы и абсолютной сходимости интеграла, нам понадобятся алгоритмы построения многогранника Ньютона (алгоритм NewtonPolyhedron) и двойственного ему веера нормалей (алгоритм NormalFan). Напомним, что многогранником Ньютона полинома называется выпуклая оболочка точек носителя этого полинома.

Алгоритмы построения многогранника Ньютона полинома и двойственного ему веера осуществляются в два этапа: на первом этапе определяется носитель полинома, а на втором находится его выпуклая оболочка.

Для построения выпуклой оболочки множества точек n-мерного пространства используется алгоритм, базирующийся на алгоритме Моцкина–Бургера [6], и его реализация П. А. Буровским [7] в виде Maple-программ Convexhull и FullConvexhull программного пакета convexhull.

Программа Convexhull по заданному списку точек и размерности пространства строит список точек исходного множества, попадающих на каждую из гиперграней выпуклой оболочки точек исходного списка, и список образующих нормальных конусов соответствующих гиперграней. Программа FullConvexhull строит описание граней всех размерностей и нормальных конусов к ним.

Для построения многогранника Ньютона и двойственного ему вектора нормалей (алгоритмы NewtonPolyhedron и NormalFan соответственно)

мы будем пользоваться какой-либо одной из этих процедур в зависимости от полноты требуемых данных.

Если исходный интеграл берется по пространству R^2, то подключение пакета convexhull не требуется. Для построения многогранника Ньютона (это будет выпуклый многоугольник) используется стандартная Maple-процедура convexhull программного модуля simplex. Для построения веера нормалей векторы, получающиеся при обходе многоугольника против часовой стрелки, поворачиваются на угол 90^0 (если (x, y) – координаты вектора-стороны многоугольника, то $(-y, x)$ – координаты внутренней нормали к этой стороне) и минимизируются т. е. для каждого вектора вычисляется наименьший общий делитель его координат и все координаты на него делятся.

Проверка того, что знаменатель не обращается в нуль, и квазиэллиптичность знаменателя. На знаменатель $Q(x)$ подынтегральной формы интеграла (1) накладываются два условия:

Условие 1. Полином $Q(x)$ не обращается в нуль в R^n.

Условие 2. Полином $Q(x)$ является квазиэллиптическим полиномом, т. е. для любого ненулевого ковектора $v \in R^{n*}$ срезка Q_v не обращается в нуль в торе $(R \setminus \{0\})^n$.

Для проверки этих условий используется так называемый алгоритм цилиндрического алгебраического разбиения, или сокращенно CAD-алгоритм (от англ. *Cylindrical Algebraic Decomposition*). Этот алгоритм является инструментом для решения систем полиномиальных уравнений и неравенств в R^n и заключается в построении такого разбиения пространства R^n, чтобы в каждой компоненте разбиения сохранялись знаки всех полиномов исходной системы. Под разбиением пространства будем понимать конечную совокупность непересекающихся областей пространства, объединение которых равно этому пространству

Впервые такой метод был предложен в 1948 г. Тарским, но он не был реализован практически. В 1973 г. Коллинз предложил конструктивный метод. В рамках данной статьи алгоритм был реализован в виде Maple-программы CAD.

Введем некоторые определения, необходимые для формулирования основных положений метода. Все определения даются согласно работам [8, 9]. Более детальное описание алгоритма можно найти в [9].

Входными данными для алгоритма является множество полиномов $\Phi \subset \overline{Q}[x_1, ..., x_n]$, где $\overline{Q}$ – поле алгебраических чисел, и список переменных $[x_1, ..., x_n]$.

CAD-алгоритм состоит из трех основных этапов: проекции, построения базы и расширения.

Этап проекции состоит из $(n - 1)$-го шага. На каждом шаге строится новое множество полиномов, зависящих от числа переменных, на единицу меньшего, чем полиномы множества, полученного на предыдущем шаге. Множества нулей полиномов, полученных на каждом шаге этапа, – это проекции особых точек множеств полиномов предыдущего шага, т. е. точек самопересечений, изолированных точек и точек, в которых касательные вертикальны. Переход от множества полиномов к множеству их проекций осуществляется при помощи оператора proj. Чтобы его записать, нам понадобятся некоторые условные обозначения.

Пусть $F = \{f_1, \ldots, f_r\}$, где $f_i(x_1, \ldots, x_n) \in \overline{Q}[x_1, ..., x_n]$, $1 \le i \le r$. Эти полиномы можно рассматривать как полиномы, зависящие от одной переменной x_n, коэффициентами которых являются полиномы от $(n - 1)$-й переменной $x_1, \ldots, x_{n-1}$, т. е. $f_i(x_1, \ldots, x_n)$, $1 \le i \le r$, представляется в виде $f_i(x_1, ..., x_{n-1}, x_n) = f_{i,d_i}(x_1, ..., x_{n-1}) x_n^{d_i} + ... + f_{i,0}(x_1, ..., x_{n-1}) \in \overline{Q}[x_1, ..., x_{n-1}][x_n]$ и $f_{i,k}(x_1, ..., x_{n-1})$ – коэффициент i-го полинома при k-й степени переменной, x_n – полином, зависящий от переменных $x_1, \ldots, x_{n-1}$.

Обозначим $\hat{f}_i^k(x_1, ..., x_n) = \sum_{j=0}^{k} f_i^j(x_1, ..., x_{n-1}) x_n^j$, где $0 \le k \le d_i$.

Обозначим символом D_{x_n} оператор дифференцирования по переменной x_n.

Матрицу $\mathrm{mat}(g_1, \ldots, g_k) = [g_{i,\, l-g}]$ размера $k \times k$, где $l = 1 + \max_{1 \le i \le k}(\deg(g_i))$, назовем матрицей, ассоциированной с полиномами $g_1, \ldots, g_k$.

Для полинома g степени s по переменной x_n и полинома h степени m по переменной x_n определим $\mathrm{psc}_l^{x_n}(g, h) = \det(M_l)$, где матрица M_l состоит из l первых столбцов матрицы $\mathrm{mat}\left(x^{s-l-1}g, ..., g, x^{m-l-1}h, ..., h\right)$, ассоциированной с полиномами $x^{s-l-1}g, ..., g, x^{m-l-1}h, ..., h$.

Тогда

$$\mathrm{proj}(F) = \mathrm{proj}_1(F) \cup \mathrm{proj}_2(F) \cup \mathrm{proj}_3(F),$$

где $\mathrm{proj}_1 = \left\{ f_i^k(x_1, ..., x_{n-1}) \mid 1 \le i \le r, 0 \le k \le d_i \right\}$;

$$\mathrm{proj}_2 = \left\{ \mathrm{psc}_l^{x_n}\left(\hat{f}_i^k(x_1, ..., x_n), D_{x_n}\left(\hat{f}_i^k(x_1, ..., x_n)\right)\right) \mid 1 \le i \le r, 0 \le l < k \le d_i \right\};$$

$$\mathrm{proj}_3 = \left\{ \mathrm{psc}_m^{x_n}\left(\hat{f}_i^{k_i}(x_1, ..., x_n), \hat{f}_j^{k_j}(x_1, ..., x_n)\right) \mid 1 \le i < j \le r, 0 < k_i, k_j \le d_i, 0 \le m \le \min(k_i, k_j) \le d_i \right\}.$$

Определенный таким образом оператор proj рекурсивно применяется $(n - 1)$ раз к исход-

ному множеству полиномов, т. е. $\text{proj}^0(\Phi) = \Phi$, $\text{proj}^1(\Phi) = \text{proj}(\text{proj}^0(\Phi))$, …, $\text{proj}^j(\Phi) = \text{proj}(\text{proj}^{j-1}(\Phi))$, …, $\text{proj}^{n-1}(\Phi) = \text{proj}(\text{proj}^{n-2}(\Phi))$. Последнее множество – множество полиномов, зависящих от одной переменной x_1, – конечный результат этапа проекции.

На *этапе построение базы* строится множество пробных точек разбиения пространства R^1. В качестве входных данных выступает множество полиномов $\text{proj}^{n-1}(F)$ – это множество полиномов, зависящих от одной переменной, получившееся в результате этапа проекции, и сама переменная, от которой зависят данные полиномы. Вычисляются действительные корни полиномов множества $\text{proj}^{n-1}(F)$. Эти корни в общем случае являются алгебраическими числами и могут быть представлены в виде пары полинома и интервала, на котором этот полином имеет единственный корень – данное алгебраическое число, причем концами интервала являются рациональные числа, а полином выбирается таким образом, чтобы его степень была наименьшей. Пусть у нас получилось следующее множество алгебраических чисел $\xi_1 \in (u_1, v_1], \ldots, \xi_s \in (u_s, v_s]$, где все $u_j, v_j \in \overline{Q}$. Пробные точки выбираем следующим образом: $\alpha_1 = u_1$, $\alpha_2 = \xi_1, \ldots, \alpha_{2s+1} = v_s + 1$. Последнее множество и является результатом этапа построения базы.

Цель *этапа расширения* заключается в построении разбиения пространства R^n. В качестве входных данных выступает множество пробных точек, полученное на этапе построения базы: $\text{base}_0 = \{\alpha_1, \ldots, \alpha_{2s+1}\}$. Этап расширения содержит $(n - 1)$ шагов.

На первом шаге точки множества base_0 подставляются в полиномы множества $\text{proj}^{n-2}(F)$ – проекции, полученной на $(n - 2)$-м шаге этапа проекции. В результате такой подстановки для каждой точки α_j, $1 \le j \le 2s + 1$ получается множество полиномов, зависящих от одной переменной x_2. К ним снова применяется этап построения базы. Пусть в результате для каждой точки α_j, $1 \le j \le 2s + 1$ получилось множество пробных точек $\left\{\beta_1^j, \ldots, \beta_{2l_i+1}^j\right\}$. Тогда множество пробных точек разбиения R^2 организуется следующим образом:

$$\left\{\begin{array}{l}\left\{(\alpha_1, \beta_1^1), \ldots, (\alpha_1, \beta_{2l_1+1}^1), (\alpha_2, \beta_1^2), \ldots,\right.\\ \left.\left\{(\alpha_1, \beta_{2l_2+1}^2), \ldots, (\alpha_{2s+1}, \beta_1^{2s+1}), \ldots, (\alpha_{2s+1}, \beta_{2l_{2s+1}+1}^{2s+1})\right\}\right.\end{array}\right\}.$$

На втором шаге эти точки подставляются в полиномы из множества $\text{proj}^{n-3}(F)$, и аналогичная процедура повторяется. Таким образом, после выполнения $(n - 1)$-го шага мы получим множество пробных точек пространства R^n. Это и есть результат CAD-алгоритма.

Корректность рассмотренного CAD-алгоритма в рамках данной статьи не доказывается. Это доказательство можно найти, например, в [9].

Вернемся теперь к проверке условий 1 и 2.

Достаточным условием того, что полином не обращается в нуль в R^n, является то, что все входящие в его состав мономы $c_\alpha x_1^{\alpha_1} \ldots x_n^{\alpha_n}$ имеют положительные коэффициенты c_α и четные степени α_i по каждой переменной x_i, а свободный член полинома не равен нулю. Если это условие не выполняется, то для проверки условия 1 используется CAD-алгоритм. С его помощью строится множество пробных точек для этого полинома; затем вычисляются значения исходного полинома в этих точках, и если ни в одной из этих точек полином не обращается в нуль, то можно заключить, что полином не обращается в нуль в R^n. Если же хотя бы в одной пробной точке значение полинома равно нулю, то условие 1 не выполнено.

Таким образом, алгоритм PolyNot0 выглядит следующим образом:

PolyNot0(Q, x)

Input: $Q = Q(x_1, \ldots, x_n)$ – полином, зависящий от n переменных;

$x = [x_1, \ldots, x_n]$ – список переменных;

Output: true, если полином Q не обращается в нуль в R^n, false – в противном случае;

if (все мономы полинома Q имеют положительные коэффициенты и четные степени
по каждой переменной) and (свободный член полинома Q равен нулю) then return (true) else

1) SamplePoints : = CAD($\{Q\}$, x) – множество пробных точек разбиения пространства R^n;

2) ValueQ – множество значений полинома Q в точках множества SamplePoints;

3) if 0 не содержится в множестве ValueQ then return (true) else return (false) end if

end if

end.

Достаточным условием квазиэллиптичности полинома является то, что все входящие в его состав мономы $c_\alpha x_1^{\alpha_1} \ldots x_n^{\alpha_n}$ имеют положительные коэффициенты c_α и четные степени α_i по каждой переменной x_i. Если это условие не выполняется, то при помощи CAD-алгоритма нужно построить множество пробных точек для всех срезок этого полинома (не имеет значения, рассматривать эти полиномы вместе или по отдельности), исключить из этого множества точки, у которых хотя бы одно координата равна нулю (это будет соответствовать исключению координатных осей из R^n, так как в данном случае нас интересует только тор $(R \setminus \{0\})^n$), подставить оставшиеся точки в срезки исходного полинома и убедиться, что они не обращаются в нуль ни в одной точке из оставшихся точек. Алгоритм Quasielliptic выглядит таким образом:

Quasielliptic(Q, x)

Input: $Q = Q(x_1, \ldots, x_n)$ – полином, зависящий от n переменных;

$x = [x_1, \ldots, x_n]$ – список переменных;

Output: true, если полином Q квазиэллиптический, false – в противном случае;

if (все мономы полинома Q имеют положительные коэффициенты и четные степени

по каждой переменной) then return (true) else

1) $\Delta(Q)$:=NewtonPolyhedron(Q,x) – многогранник Ньютона полинома Q (здесь нас будут интересовать все грани многогранника Ньютона, поэтому мы будем использовать процедуру FullConvexhull);

2) $F = \{f_1, \ldots, f_k\}$ – множество срезок, соответствующих граням многогранника $\Delta(Q)$;

3) SamplePoints : = CAD($\{f_1, \ldots, f_k\}$,x) – множество пробных точек разбиения пространства R^n;

4) SamplePointsInTor – подмножество точек из SamplePoints, у которых ни одна координата не равна нулю;

5) ValueQ – множество значений полиномов множества F в точках множества SamplePoints;

6) if 0 не содержится в множестве ValueQ then return (true) else return (false) end if

end if

end.

Вычисление рациональной первообразной, не растущей на бесконечности. Алгоритм нахождения рациональной первообразной (RationalPrimitive), не растущей на бесконечности, основывается на методе неопределенных коэффициентов Остроградского и осуществляется в три этапа:

– определяются степени мономов, входящих в состав коэффициентов первообразной;

– выписываются полиномы с неопределенными коэффициентами и найденными степенями;

– решается уравнение на коэффициенты.

В качестве входных данных для этого алгоритма выступают полиномы $P(x)$, $Q(x)$ и список переменных x. Если исходная дифференциальная форма алгебраически точна и имеет не растущую на бесконечности рациональную первообразную, то алгоритм возвращает значение этой первообразной, если нет, то возвращается список из n нулей.

Рассмотрим теперь все этапы алгоритма более подробно.

Допустим, что дифференциальная форма $\omega = \frac{P(x_1, \ldots, x_n)}{Q(x_1, \ldots, x_n)} dx_1 \ldots dx_n$ алгебраически точна и φ – ее рациональная первообразная, не растущая на бесконечности. Представим ω в виде $\widetilde{\omega} = \frac{\widetilde{P}(x)}{\widetilde{Q}^{2k}(x)} dx$ следующим образом: если найдется полином $\widetilde{Q}(x)$ и натуральное число k, такие что $\widetilde{Q}^{2k}(x) = Q(x)$, то тогда $\widetilde{P}(x) = P(x)$; если найдется полином $\widetilde{Q}(x)$ и натуральное число $k > 1$, такие что $\widetilde{Q}^{2k}(x) = Q(x)$, то тогда $\widetilde{P}(x) = P(x) \cdot \widetilde{Q}(x)$; если таких $\widetilde{Q}(x)$ и k не существует, то тогда $\widetilde{Q}(x) = Q(x)$, $k = 1$, $\widetilde{P}(x) = P(x) \cdot Q(x)$.

Рациональную первообразную φ будем искать в виде

$$\frac{1}{B(x)} \sum_{i=1}^{n} A_i(x) dx_1 \wedge \ldots [i] \ldots \wedge dx_n ,$$

где $x = (x_1, \ldots, x_n)$, A_i – некоторые полиномы; $B \equiv \widetilde{Q}^{2k-1}$ – квазиэллиптический полином, не обращающийся в нуль на R^n.

Пусть $\Delta = \Delta(B)$ – многогранник Ньютона. Очевидно, что многогранники $\Delta(Q)$ и $\Delta(B)$ гомотетичны. Таким образом, если $\Delta(Q)$ удовлетворяет условию предположения 1, т. е. содержит по ребру на каждой координатной оси в R^n, то и $\Delta(B)$ удовлетворяет этому условию. Веера нормалей, двойственные к $\Delta(Q)$ и $\Delta(B)$, совпадают. Пронумеруем векторы веера нормалей, двойственного $\Delta(B)$ таким образом, чтобы $v^i = \left(0, \ldots, \underset{\hat{i}}{1}, \ldots, 0\right) \in R^{n*}$, $i = 1, \ldots, n$.

Обозначим через Δ^0 множество внутренних точек многогранника Δ, а через Δ_k^0 – множество точек, лежащих в относительной внутренности $(n-1)$-мерной грани $\Delta_k = \Delta_k^{(n-1)}(B)$, двойственной вектору v^k.

В работе [10] доказывается теорема о том, что в предположении 1 для того чтобы дифференциальная форма φ не росла на бесконечности, ее коэффициенты должны иметь вид

$$A_i(x) = (-1)^i x_i \sum_{k=n+1}^{d} v_i^k A^k(x), \; I = 1, \ldots, n,$$

где A_i^0 и A^k – полиномы, многогранники Ньютона которых удовлетворяют условиям $\Delta\left(A_i^0\right) + I_i \subset \Delta^0 \cup \Delta_0^i$, $\Delta\left(A^k\right) + I \subset \Delta_k^0$, здесь $I = (1, \ldots, 1) \in R^n$, $I_i = \left(1, \ldots, \underset{\hat{i}}{0}, \ldots 1\right) \in R^n$, а $v_1^k, \ldots, v_n^k$ – координаты вектора, двойственного грани $\Delta_k^{n-1}(B)$.

Исходя из этой теоремы, определим возможные степени полиномов $A_i(x)$, $I = 1, \ldots, n$, коэффициенты которых не определены. Затем эти полиномы подставляем в уравнение

$$\sum_{i=1}^{n} (-1)^{i-1} \left(B(x) \frac{\partial A_i(x)}{\partial x_i} - (2k-1) A_i(x) \frac{\partial B(x)}{\partial x_i} \right) - P(x) = 0$$

и решим уравнение на коэффициенты.

Если в результате решения этого уравнения все неопределенные коэффициенты получились равными нулю, то рациональной первообразной φ, не растущей на бесконечности, не существует,

и мы возвращаем список из n нулей; если же не все неопределенные коэффициенты равны нулю, то возвращается список коэффициентов первообразной.

Абсолютная сходимость интеграла. В [2] доказывается следующая теорема:

Теорема 2. Если Q – квазиэллиптический полином, не обращающийся в нуль на R^n, то интеграл (1) абсолютно сходится тогда и только тогда, когда $I + \Delta(P) \subset \Delta^0(Q)$, т. е. когда сдвиг многогранника $\Delta(P)$ вдоль вектора $I = (1, \ldots, 1) \in R^n$ лежит во внутренности $\Delta^0(Q)$ многогранника $\Delta(Q)$.

На ее основе составлен следующий алгоритм:

AbsoluteConvergence(Q, P, x)

Input: $P = P(x_1, \ldots, x_n)$ – полином, зависящий от n переменных, числитель подынтегральной формы интеграла (1);

$Q = Q(x_1, \ldots, x_n)$ – полином, зависящий от n переменных, – знаменатель подынтегральной формы интеграла (1);

$x = [x_1, \ldots, x_n]$ – список переменных;

Output: true, если интеграл (1) сходится абсолютно,

false – в противном случае;

1) $\Delta(P)$: = NewtonPolyhedron(P, x) (используется процедура Convexhull);

2) *verticesP* – список вершин многогранника Ньютона $\Delta(P)$;

3) образовать список *verticesP_I*, сдвинув все вершины списка *verticesP* на вектор I (увеличить все координаты всех точек списка *verticesP* на 1);

4) $\Delta(Q)$: = NewtonPolyhedron(Q, x) (используется программа Convexhull); $\left\{\Delta_1^{n-1}(Q), \ldots, \Delta_d^{n-1}(Q)\right\}$ – гиперграни $\Delta(Q)$;

5) $\{v_1, \ldots, v_d\}$: = NormalFan(Q, x) – веер нормалей, двойственный $\Delta(Q)$;

6) для каждой точки $p \in$ *verticesP_I* и каждой гиперграни $\Delta_k^{n-1}(Q)$ вычислить скалярное произведение вектора, отпущенного из произвольной точки гиперграни $\Delta_k^{n-1}(Q)$ в точку p, и вектора v^k, $1 \le k \le d$;

7) if (все скалярные произведения строго положительны) then return (true) else return(false) end if

end.

Простое разбиение веера нормалей. Рассмотрим алгоритм PrimitiveNormalFan, преобразующий веер нормалей в примитивный веер нормалей в результате простого подразбиения. На входе в алгоритм имеется множество конусов Σ, образующих веер, на выходе – множество примитивных конусов, организующих простое подразбиение веера Σ.

Алгоритм состоит из двух этапов.

На первом этапе исходное множество конусов Σ преобразуется во множество симплициальных конусов Σ' следующим образом: пусть $C_0 = \left(v_0^1, \ldots, v_0^m\right)$ – не симплициальный конус, т. е. $m > n$, тогда находим вектор $v = \sum_{i=1}^{m} v^i$, минимизируем его (сокращаем все его координаты v^i на их наибольший общий делитель) и добавляем ко всем группам из (n –1) векторов множества $\left\{v_0^1, \ldots, v_0^m\right\}$, образующих конусы размерности (n – 1). Данная процедура применяется ко всем несимплициальным конусам множества Σ, начиная с конусов размерности 3 и заканчивая конусами размерности n. В результате мы получим множество симплициальных конусов Σ'. Заметим, что так как в пространстве R^2 все конусы симплициальны, то в случае интегрирования по пространству R^2 первый этап автоматически опускается.

На втором этапе, пока во множестве Σ' найдется конус $C_0 = \left(v_0^1, \ldots, v_0^n\right)$, такой что матрица, составленная из векторов $v_0^1, \ldots, v_0^n$, не является унимодулярной, находим точку p с целыми координатами, лежащую внутри параллелепипеда (или на его гранях, но не совпадающую ни с одной его вершиной), построенного на векторах $v_0^1, \ldots, v_0^n$; затем проводим из начала координат в точку p вектор v. Далее конструируем множество конусов $C' = \{C_i\}$, где $C_i = (v^1, \ldots, v^{i-1}, v, v^{i+1}, \ldots, v^n)$, $i = 1, \ldots, n$; исключаем из множества C' конусы, размерность которых меньше размерности пространства; в заключение добавляем оставшиеся в C' конусы к множеству Σ', а конус C_0 из этого множества исключаем. В результате повторения такой процедуры получим множество примитивных конусов, которые и образуют примитивный веер.

Рассмотренный алгоритм корректен, если исходить из простых геометрических соображений.

Компьютерная реализация и примеры вычислений

Пакеты quasielliptic2 и quasielliptic3 на языке системы Maple разработаны для понижения кратности интегралов вида (1), удовлетворяющих условиям теоремы 1, по пространствам R^2 и R^3 соответственно. В них реализованы все алгоритмы, рассмотренные выше (за исключением алгоритма Моцкина–Бургера и алгоритма построения выпуклой оболочки, реализованногох П. А. Буровским [7]) в виде одноименных процедур. Все эти процедуры могут вызываться пользователем непосредственно, так как они имеют самостоятельную ценность и могут быть полезны при решении других задач.

Основной алгоритм реализован в процедуре QuasiellipticIntegration. Для ее вызова необходимо ввести следующие входные данные:

– P – полином с рациональными коэффициентами, зависящий от двух или трех переменных, – числитель коэффициента подынтегральной формы;

– Q – полином с рациональными коэффициентами, зависящий от двух или трех переменных, – знаменатель коэффициента подынтегральной формы, причем если существует полином с рациональными коэффициентами Q' и натуральное число d, такие что $Q = Q'^{d}$, то вместо полинома Q в качестве исходных данных можно ввести пару $[Q', d]$;

– var – список переменных, от которых зависят полиномы P и Q.

Если регулярная рациональная первообразная для подынтегральной дифференциальной формы заранее известна, то список ее коэффициентов можно ввести четвертым аргументом для ускорения работы программы.

Если исходный интеграл не удовлетворяет какому-либо из условий теоремы 1 или условию предположения 1, то выдается сообщение об ошибке и информация о том, какое из условий нарушено. Если все условия для исходного интеграла выполнены, то результатом работы процедуры для случая двух переменных является значение исходного интеграла. Для случая интегрирования по пространству R^3 основной результат работы процедуры – интеграл кратности 2. Если этот интеграл вычисляется до конца стандартными средствами Maple 9, то выдается его значение; если же он до конца не вычисляется, но сводится к одномерному интегралу, то выдается именно этот интеграл. Если стандартными средствами Maple 9 двумерный интеграл не упрощается, но имеет вид (1) и удовлетворяет условиям теоремы 1, то к нему можно применить процедуру QuasiellipticIntegration, но уже из пакета quasielliptic2, и вычислить этот интервал до конца.

Рассмотрим примеры работы этой процедуры. Все вычисления производились в среде Maple 9 на компьютере на базе Athlon XP 1700+, 512 Mb RAM.

Сначала приведем примеры вычисления интегралов по пространству R^2. Нам потребуется пакет quasielliptic2. Загрузим его следующим образом:

```
> read (quasielliptic2):
```

Пример 1. Вычислим интеграл

$$\int_{R^2} \frac{x^2 + 4x^2y^2 + 4x^4y^4 + x^4y^6 + x^6y^4 + y^2}{\left(1 + x^2 + x^2y^4 + x^4y^4 + x^4y^2 + y^2\right)} dxdy .$$

Введем входные данные:

```
> P1 : = x^2+4*x^2*y^2+4*x^4*y^4+ x^4*y^6+
+x^6*y^4+y^2;
```

$$P1 := x^2 + 4x^2y^2 + 4x^4y^4 + x^4y^6 + x^6y^4 + y^2$$

```
> Q1 : = (1+x^2+x^2*y^4+x^4*y^4+ x^4*y^2+
+y^2)^2;
```

$$Q1 := (1 + x^2 + x^2y^4 + x^4y^4 + x^4y^2 + y^2)^2$$

```
> var : = [x, y];
```

$$var := [x, y]$$

Вызовем процедуру QuasiellipticIntegration:

```
> t : = time():
QuasiellipticIntegration (P1, Q1, var);
TimeOfCalculation : = time() – t;
```

$$2\pi$$

TimeOfCalculation: = 0.100

Пример 2. Вычисляем интеграл

$$\int_{R^2} \frac{x^2 - y^2 - 2x^4y^4 + 2x^6y^6}{\left(1 + x^2 + y^2 + x^4y^2 + x^4y^6\right)^2} dxdy .$$

Введем входные данные:

```
> P2 : = x^2-y^2-2*x^4*y^4+2*x^6*y^6;
```

$$P2 := x^2 - y^2 - 2x^4y^4 + 2x^6y^6$$

```
> Q2 : = (1+x^2+y^2+x^4*y^2+x^4*y^6)^2;
```

$$Q2 := (1 + x^2 + y^2 + x^4y^2 + x^4y^6)^2$$

Вызовем процедуру QuasiellipticIntegration:

```
> t : = time():
QuasiellipticIntegration (P2, Q2, var);
TimeOfCalculation : = time() – t;
```

$$\pi - \frac{1}{2}\pi\sqrt{2}$$

TimeOfCalculation: = 0.091

Пример 3. Вычислим интеграл

$$\int_{R^2} \frac{yx}{\left(1 + x^3 + 2x^2y^2 + 3xy^4 + y^6\right)^3} dxdy .$$

Введем входные данные:

```
> P3 : = y*x;
```

$$P3 := yx$$

```
> Q3 : = (1+x^3+2*x^2*y^2+3*x*y^4+y^6)^3;
```

$$Q3 := (1 + x^3 + 2x^2y^2 + 3xy^4 + y^6)^3$$

Вызовем процедуру QuasiellipticIntegration:

```
> t : = time():
QuasiellipticIntegration (P3, Q3, var);
TimeOfCalculation : = time() – t;
```

Error, (in QuasiellipticIntegration) differential form denominator is not quasielliptic polynomial

TimeOfCalculation: = 0.080

Процедура возвращает сообщение о том, что знаменатель подынтегральной дифференциальной формы не является квазиэллиптическим полиномом. Это действительно так, поскольку в точке (–1, –1) полином Q3 обращается в нуль.

Рассмотрим примеры работы процедуры QuasiellipticIntegration для случая трех переменных. Загрузим пакет quasielliptic3:

```
> read (quasielliptic3):
```

Пример 4. Вычислим интеграл

$$\int_{R^3} \frac{4x_3^2 + 3x_1^2x_3^2 + 3x_2^2x_3^3}{\left(1 + x_1^2 + x_2^2 + x_3^4 + x_1^4x_3^2 + x_2^4x_3^2\right)^2} dx_1dx_2dx_3 .$$

Введем входные данные:

```
> P1 : = 4*x3^2+3*x1^2*x3^2+3*x2^2*x3^2;
```

$$P1 := 4x3^2 + 3x1^2x3^2 + 3\,x2^2x3^2$$

> Q1 : = (1+x1^2+x2^2+x3^4+x1^4*x3^2+
+x2^4*x3^2)^2

$$Q1 := (1 + x1^2 + x2^2 + x3^4 + x1^4 x3^2 + x2^4 x3^2)^2$$

> var : = [x1, x2, x3];

$$var := [x1, x2, x3]$$

Вызовем процедуру QuasiellipticIntegration:

> t := time():

QuasiellipticIntegration (P1, Q1, var);

TimeOfCalculation : = time() – t;

$$-\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} -\frac{1}{1+y2^2+y1^4}\,dy2dy1 = \frac{1}{2}\pi B\left(\frac{1}{4},\frac{1}{4}\right)$$

$$TimeOfCalculation := 11.236$$

Пример 5. Вычислим интеграл

$$\int_{R^3} \frac{1+x_4^3}{\left(1+x_1^2+x_2^2+x_3^4+x_1^2x_3^2+x_2^2x_3^2\right)^2}\,dx_1dx_2dx_3 .$$

Введем входные данные:

> P2 : = 1+x3^4;

$$P2 := 1 + x3^4$$

> Q2 : = (1+x1^2+x2^2+x3^4+x1^2*x3^2+
+x2^2*x3^2)^2;

$$Q2 := (1 + x1^2 + x2^2 + x3^4 + x1^2x3^2 + x2^2x3^2)^2$$

Вызовем процедуру QuasiellipticIntegration:

> t : = time():

QuasiellipticIntegration (P2, Q2, var);

TimeOfCalculation : = time() – t;

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{1}{1+y2^2+y1^2+y2^2y1^2}\,dy2dy1 =$$

$$= \pi^2 - \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} -\frac{1}{1+y2^2+y1^2+y2^2y1^2}\,dy2dy1 = \pi^2$$

$$TimeOfCalculation := 5.338$$

Таким образом алгоритмы, представленные в данной статье, позволяют понижать кратность интегралов рациональных функций со знаменателем специального вида, а в некоторых случаях точно вычислять такие интегралы. Это существенным образом расширяет класс вычисленных интегралов, что может быть полезно при решении различных математических и физических задач. Все представленные выше алгоритмы реализованы автором в системе компьютерной алгебры Maple 9 для двумерного и трехмерного случая. Приведенные примеры вычисленных интегралов могут использоваться в качестве тестовых для сравнения различных методов приближенного интегрирования.

Библиографический список

1. Фам, Ф. Введение в топологические исследования особенностей Ландау / Ф. Фам. М. : Мир, 1967. 184 с.

2. Хуа, Р. Гомологии и феймановские интегралы / Р. Хуа, В. Теплиц. М. : Мир, 1969. 223 с.

3. Сергеев, А. Г. Теория твисторов и классические калибровочные поля / А. Г. Сергеев. // Монополи: Топологические и вариационные методы : сб. ст. (1983–1989). М. : Мир, 1989. С. 492–555.

4. Цих, А. К. Интегралы рациональных функций по пространству R^n / А. К. Цих // Докл. АН СССР. 1989. Т. 307. № 6. С. 1325–1329.

5. Ермолаева, Т. О. Интегрирование рациональных функций по R^n с помощью торических компактификаций и многомерных вычетов / Т. О. Ермолаева, А. К. Цих // Мат. сб. 1996. № 9. С. 45–64.

6. Черников, С. Н. Линейные неравенства / С. Н. Черников. М. : Наука, 1968. 488 с.

7. Буровский, П. А. Алгоритм Моцкина–Бургера и вычисление выпуклых оболочек точек n-мерного пространства / П. А. Буровский // Вестн. краснояр. гос. ун-та. 2005. № 1. С. 85–92. (Серия «Физико-математические науки»).

8. Jistrand, M. Cylindrical Algebraic Decomposition – an Introduction : technical rep. / M. Jistrand ; Lincöping Univ. Lincöping, Sweden, 1995. 38 p.

9. Mishra, B. Algorithmic Algebra. Texts and Monographs in Computer Science / B. Mishra. Berlin : Springer-Verlag, 1991. 416 с.

10. Кочеткова, Т. О. Вычисление интегралов некоторых рациональных функций с помощью торических компактификаций : дис. … канд. физ.-мат. наук / Т. О. Кочеткова. Красноярск, 1998. 74 с.

M. V. Burachenko

ALGORITHM OF INTEGRAL MULTIPLICITY REDUCING FOR RATIONAL ALGEBRAICALLY EXACT DIFFERENTIAL FORM WITH QUASIELLIPTIC DENOMINATOR

It is considered a method of integral multiplicity reducing for rational algebraically exact differential from with quasielliptic denominator. It is described the algorithm and presented its implementation in the algebraic computer system Maple 9.