ZASTOSOWANIA MATEMATYKI
APPLICATIONES MATHEMATICAE
VIII (1965)

С. ПАШКОВСКИ (Вроцлав)

АЛГОРИТМ ВЫЧИСЛЕНИЯ КОЭФФИЦИЕНТОВ ПРИБЛИЖАЮЩЕГО МНОГОЧЛЕНА

В вычислениях на автоматической цифровой вычислительной машине непрерывные функции обычно заменяются многочленами, так как их значения определяются исключительно просто. В этой работе мы предлагаем новый алгоритм вычисления коэффициентов многочлена приближающего данную функцию с данной ошибкой(1). Алгоритм связан с разложением функции в ряд по многочленам Чебышева.

1. Известные методы аппроксимации. Ошибкой равномерной аппроксимации функции $f(t)$ многочленом $w(t)$ в сегменте $\langle a, b\rangle$ называется число

$$\delta(f, w) = \max_{a\leqslant t\leqslant b} |f(t)-w(t)|.$$

Часто необходимо решить следующую аппроксимационную задачу $\mathscr{A}$:

Для данной функции $f(t)$, сегмента $\langle a, b\rangle$ и числа $\varepsilon > 0$ найти многочлен $w(t)$ такой, что $\delta(f, w) \leqslant \varepsilon$.

Решение этой задачи неоднозначно и среди многочленов удовлетворяющих неравенству $\delta(f, w) \leqslant \varepsilon$ следует выбрать многочлен низкой степени, значения которого вычисляются с помощью возможно малого числа действий.

В классе многочленов степени не превышающей данного числа $n \geqslant 0$ можно выделить тот многочлен, для которого ошибка $\delta(f, w)$ минимальна. Он называется n-м многочленом наилучшего приближения для функции $f(t)$ в сегменте $\langle a, b\rangle$. Наилучшее — в вышеуказанном смысле — решение задачи $\mathscr{A}$ состояло бы в выборе многочлена наилучшего приближения $w_n(t)$ наименьшей степени n, для которой еще выполняется неравенство $\delta(f, w_n) \leqslant \varepsilon$. Другие многочлены степени n и все многочлены низших степеней аппроксимируют функцию $f(t)$ с большей ошибкой. Однако нахождение многочлена наилучшего

(1) Содержание работы было доложено на семинаре Кафедры вычислительной математики Вроцлавского университета.

приближения требует настолько сложных вычислений (см., например, [2], § 15), что часто вместо него мы ищем — применяя более удобные методы — многочлены, которые в отношении ошибки хотя и не наилучши, но довольно хороши.

Теперь мы кратко изложим известный метод вычисления многочлена хорошо приближающего данную функцию $f(t)$.

Мы предположим, что $\langle a, b\rangle = \langle -1, 1\rangle$ ([2], § 13.2). Это не снижает общности рассуждений, так как любой другой сегмент можно свести к $\langle -1, 1\rangle$ путем линейного преобразования. Существенно только второе предположение (впрочем, верное для чаще всего применяемых функций) — что функция $f(t)$ разлагается в сегменте $\langle -1, 1\rangle$ в степенной ряд

$$\sum_{k=0}^{\infty} a_k t^k \tag{1}$$

и в ряд по многочленам Чебышева

$$\sum_{k=0}^{\infty} a_k T_k(t). \tag{2}$$

Многочлены Чебышева $T_k(t)$ $(k = 0, 1, \ldots)$ определяются формулами

$$\begin{gathered} T_0(t) = 1, \quad T_1(t) = t, \\ T_k(t) = k\sum_{j=0}^{[k/2]} \frac{(-1)^j}{2(k-j)}\binom{k-j}{j}(2t)^{k-2j} \quad (k>1). \end{gathered} \tag{3}$$

Они удовлетворяют также рекуррентному соотношению

$$T_k(t) = 2tT_{k-1}(t) - T_{k-2}(t) \quad (k > 1). \tag{4}$$

Коэффициенты ряда (2) определены формулами

$$a_k = \frac{2q_k}{\pi}\int_{-1}^{1} f(t)T_k(t)(1-t^2)^{-1/2}dt, \tag{5}$$

где $q_0 = \frac{1}{2}$, $q_k = 1$ для $k > 0$.

Ряд (2) применяется в аппроксимации потому, что его n-я частичная сумма

$$s_n(t) = \sum_{k=0}^{n} a_k T_k(t)$$

аппроксимирует функцию $f(t)$ с ошибкой

$$\delta(f, s_n) \leqslant \sum_{k=n+1}^{\infty} |a_k|,$$

которая обычно немного больше ошибки аппроксимации n-м многочленом наилучшего приближения ([2], теоремы 5:6 и 13:3).

Вычисление многочлена $s_n(t)$ приближающего функцию $f(t)$, разложенную в ряд (1), с ошибкой не превышающей ε разделяется на три этапа:

(a) вычисление коэффициентов a_k ряда (2),

(b) определение наименьшего n такого, что

$$\sum_{k=n+1}^{\infty} |a_k| \leqslant \varepsilon, \tag{6}$$

(c) вычисление коэффициентов многочлена $s_n(t)$ при $t^0, t^1, \ldots, t^n$.

Этап (b) конечно очень прост, по крайней мере тогда, когда последовательность $\{a_k\}$ быстро убывает к нулю, что позволяет в (6) заменить бесконечную сумму ее несколькими начальными слагаемыми. Значительно более трудными кажутся этапы (a) и (c).

Для немногочисленных функций (например синуса и косинуса) коэффициенты a_k протабулированы (напр. в [1]) или выражаются формулами, которые проще (5). Для других функций до сих пор применялись в этапе (a) формулы

$$a_k = q_k \sum_{j=0}^{\infty} \frac{1}{2^{2j+k-1}} \binom{2j+k}{j} \alpha_{2j+k} \qquad (k = 0, 1, \ldots)$$

связывающие ряд (2) со степенным рядом (1). Эти формулы будут довольно простыми, если раньше вычислить константы

$$\frac{q_k}{2^{2j+k-1}} \binom{2j+k}{j}.$$

Однако тогда недостатком этих формул будет необходимость хранения в памяти цифровой машины обширной таблицы констант вычисленных с большой точностью.

В этапе (c) применялся до сих пор метод вытекающий непосредственно из формул (3). Он имеет тот же недостаток.

2. Новый алгоритм вычисления $s_n(t)$. Теперь мы дадим обоснование более простым алгоритмам выполнения этапов (a) и (c) задачи $\mathscr{A}$.

Рассмотрим сначала этап (с) на примере вычисления коэффициентов многочлена

$$s_4(t) = 26T_0(t)-19T_1(t)+7T_2(t)-11T_3(t)+T_4(t). \tag{7}$$

Из формулы (4) для $k=4$ следует, что

$$\begin{aligned} s_4(t) &= 26T_0(t)-19T_1(t)+7T_2(t)-11T_3(t)+2tT_3(t)-T_2(t) = \\ &= 26T_0(t)-19T_1(t)+6T_2(t)-11T_3(t)+2tT_3(t). \end{aligned}$$

Рекуррентное соотношение (4) мы применим еще два раза — для $k=3$, только к слагаемому $-11T_3(t)$:

$$s_4(t) = 26T_0(t)-8T_1(t)+6T_2(t)+t\big(-22T_2(t)+2T_3(t)\big)$$

и для $k=2$, к слагаемому $6T_2(t)$:

$$s_4(t) = 20T_0(t)-8T_1(t)+t\big(12T_1(t)-22T_2(t)+2T_3(t)\big).$$

Так как $T_0(t)=1$, $T_1(t)=tT_0(t)$, то

$$s_4(t) = 20+t\big(-8T_0(t)+12T_1(t)-22T_2(t)+2T_3(t)\big).$$

Итак, мы знаем, что число 20 является свободным членом многочлена $s_4(t)$. Если с только что полученным многочленом $-8T_0(t)+12T_1(t)-22T_2(t)+2T_3(t)$ мы поступим так, как с $s_4(t)$:

$$\begin{aligned} -8T_0(t)+12T_1(t)-22T_2(t)+2T_3(t) &= \\ &= -8T_0(t)+10T_1(t)-22T_2(t)+4tT_2(t) = \\ &= 14T_0(t)+10T_1(t)+t\big(-44T_1(t)+4T_2(t)\big) = \\ &= 14+t\big(10T_0(t)-44T_1(t)+4T_2(t)\big), \end{aligned}$$

то получим свободный член 14 этого многочлена, т. е. коэффициент при t в многочлене $s_4(t)$. На конец, преобразование полученного трехчлена: дает коэффициенты 6, -44, 8 многочлена $s_4(t)$, соответственно при t^2, t^3, t^4.

Легко заметить, что вычисляя коэффициенты многочлена $s_4(t)$ мы выполняем только вычитания и удвоения. Алгоритм этих вычислений будет более понятным, если рассмотреть таблицу

$$\begin{array}{ccccccccccccc} 26 & & 26 & & 26 & \longrightarrow & \mathbf{20} & & & & & & \\ -19 & & -19 & \longrightarrow & -8 & & -8 & & -8 & \longrightarrow & \mathbf{14} & & \\ 7 & \longrightarrow & 6 & & 6 & \Longrightarrow & 12 & \longrightarrow & 10 & & 10 & \longrightarrow & \mathbf{6} \\ -11 & & -11 & \Longrightarrow & -22 & & -22 & & -22 & \Longrightarrow & -44 & & \mathbf{-44} \\ 1 & \Longrightarrow & 2 & & 2 & & 2 & \Longrightarrow & 4 & & 4 & \Longrightarrow & \mathbf{8} \end{array} \tag{8}$$

первый столбец которой образован из данных чисел $a_0, a_1, \ldots, a_4$. Любой другой столбец получается применением к предыдущему столбцу формулы (4). Двойная стрелка ведет от удваиваемого числа к результату этого действия. Две обычные стрелки ведут соответственно от уменьшаемого (горизонтальная стрелка) и вычитаемого (косая) к их разности. Найденные коэффициенты многочлена напечатаны полужирным шрифтом.

Общий алгоритм выполнения этапа (с), легко вытекающий из рассмотренного примера, особенно просто выражается на алгоритмическом языке АЛГОЛ 60 ([3]) в виде описания процедуры. В этом описании мы считаем все числа из одной строки таблицы типа (8) значениями одной переменной, которую называем так же, как первое из этих чисел. Процедуре мы дадим название *CP*:

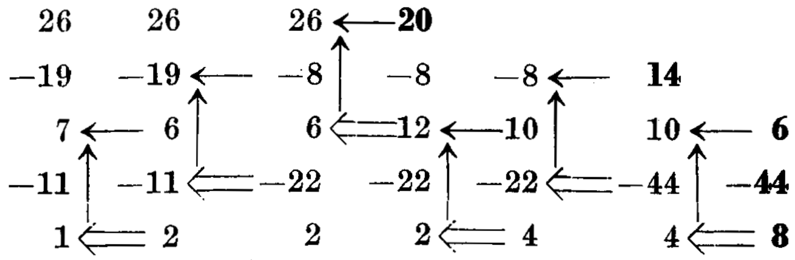
```
procedure CP(a, n);
  comment Для данного целого числа n ⩾ 0 процедура CP преобра-
    зовывает данные коэффициенты a[k] многочлена при k-м
    многочлене Чебышева на так же обозначенные коэффициенты
    этого многочлена при t↑k (k = 0, 1, ..., n);
  value n;
  integer n;
  array a;
  begin
    integer j, k;
    for j := 2 step 1 until n do
      for k := n step −1 until j do
        begin
          a[k−2] := a[k−2]−a[k];
          a[k] :=2×a[k]
        end k, j
  end CP
```

Процедура *CP* требует выполнения $\frac{1}{2}n(n-1)$ вычитаний и столько же удвоений.

Этап (а) вычислений является в некотором смысле обращением этапа (с), так как состоит по существу в выражении многочлена (частичной суммы ряда (1)) в виде линейной комбинации многочленов Чебышева. Поэтому, если выполняемые в процедуре *CP* действия заменить обратными к ним (удвоение — делением пополам, вычитание — сложением) и порядок действий тоже заменить обратным, то получится алгоритм выполнения этапа (a):

```
procedure PC(a, n);
  comment Для данного целого числа n ⩾ 0 процедура PC пре-
    образовывает данные коэффициенты a[k] многочлена при
    t ↑ k на так же обозначенные коэффициенты этого много-
    члена при k-м многочлене Чебышева (k = 0, 1, ..., n);
  value n;
  integer n;
  array a;
  begin
    integer j, k;
    for j := n step −1 until 2 do
      for k := j step 1 until n do
        begin
          a[k] := .5×a[k];
          a[k−2] := a[k−2]+a[k]
        end k, j
  end PC
```

Корректность этой процедуры легко проверить рассмотрев пример преобразования многочлена $20+14t+6t^2-44t^3+8t^4$ к виду (7), которое можно описать снова с помощью таблицы (8). Достаточно изменить в ней направление горизонтальных стрелок, косые заменить вертикальными, а числа напечатанные полужирным шрифтом считать данными:



Двойная стрелка теперь ведет от числа деленного пополам к результату этого действия, а обычные стрелки — от слагаемых к их сумме.

Процедура *PC* требует выполнения $\frac{1}{2}n(n-1)$ сложений и столько же делений пополам. Как и в предыдущей процедуре, число действий в два раза больше, чем в случае преобразования многочлена традиционными методами. Этот недостаток не слишком значителен, так как удвоение или деление пополам проще (особенно в вычислениях с удвоенной точностью, которые в аппроксимации часто необходимы) умножения в общем случае. Кроме того, в процедурах *CP* и *PC* нет констант, которые надо хранить в памяти.

С помощью процедур *CP* и *PC* легко описать процедуру *Aprox*, решающую всю задачу $\mathscr{A}$:

```
procedure Aprox (alfa, p, eps, q);
  comment Данными процедуры Aprox является целое число
    p ≥ 0, вещественное число eps > 0 и массив коэффициентов
    alfa[k] при t↑k (k = 0, 1, ..., p) в разложении аппроксими-
    руемой функции в степенной ряд. Процедура Aprox преобра-
    зовывает массив alfa на коэффициенты этого ряда при после-
    довательных многочленах Чебышева, находит наименьшую сте-
    пень q частичной суммы нового ряда, приближающей его
    с ошибкой ≤ eps и вычисляет коэффициенты alfa[k] этой
    суммы при t↑k (k = 0, 1, ..., q). Степень p должна превы-
    шать ожидаемое q по крайней мере на 10-15, чтобы отбро-
    шенные коэффициенты степенного ряда не исказили оценки
    ошибки аппроксимации и коэффициентов приближающего
    многочлена;
  value p, eps;
  real eps; p, q;
  array alfa;
  begin
    real E;
    PC(alfa, p);
    q := p;
    E := 0;
    for E := E + abs(alfa[q]) while E ≤ eps ∧ q > 0 do
        q := q-1;
    CP(alfa, q)
  end Aprox
```

На конец мы отметим, что

1° процедуры *PC* и *CP* упрощаются очевидным способом для функции $f(t)$ четной ($a_1 = a_3 = \ldots = 0$) или нечетной ($a_0 = a_2 = \ldots = 0$); в ряде Чебышева для такой функции и в ее частичной сумме коэффициенты с этими индексами тоже исчезают,

2° процедуры аналогичные к *PC* и *CP* можно построить не только для многочленов Чебышева, но и для других ортогональных многочленов, благодаря известным рекуррентным соотношениям.

Цитированые работы

[1] C. W. Clenshaw, *Chebyshev series for mathematical functions*, London 1962.

[2] S. Paszkowski, *The theory of uniform approximation, I. Non-asymptotic theoretical problems*, Warszawa 1962.

[3] — *Język algorytmiczny* ALGOL 60 (в печати).

КАФЕДРА ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ ВРОЦЛАВСКОГО УНИВЕРСИТЕТА

Поступило в Редакцию 15. 9. 1964

S. PASZKOWSKI (Wrocław)

ALGORYTM OBLICZENIA WSPÓŁCZYNNIKÓW WIELOMIANU APROKSYMUJĄCEGO

STRESZCZENIE

Dana jest funkcja ciągła $f(t)$ w przedziale $\langle -1, 1\rangle$ o znanym rozwinięciu (1) w szereg potęgowy, rozwijalna również w szereg (2) względem wielomianów Czebyszewa (3). Wiadomo (zob. np. [2], § 13.2), że wielomian $w(t)$ aproksymujący tę funkcję z błędem

$$\max_{-1\leqslant t\leqslant 1} |f(t)-w(t)|$$

niewiększym od danej liczby $\varepsilon > 0$ można znaleźć w następujący sposób:

(a) obliczyć współczynniki a_k szeregu (2),

(b) znaleźć najmniejsze n, dla którego jest spełniona nierówność (6),

(c) obliczyć współczynniki wielomianu

$$\sum_{k=0}^{n} a_k T_k(t)$$

przy $t^0, t^1, \ldots, t^n$.

W pracy podaje się nowe algorytmy wykonania etapów (a) i (c), wyrażone w języku algorytmicznym ALGOL 60 w postaci opisów procedur *CP*, *PC* i *Aprox*.

Procedura *CP* dla danej liczby całkowitej $n \geqslant 0$ przekształca dane współczynniki $a[k]$ wielomianu przy k-tym wielomianie Czebyszewa na tak samo oznaczone współczynniki tegoż wielomianu przy t^k $(k = 0, 1, \ldots, n)$. Procedura ta realizuje etap (c) za pomocą $\frac{1}{2}n(n-1)$ odejmowań i tyluż podwajań.

Procedura *PC* dla danej liczby całkowitej $n \geqslant 0$ przekształca dane współczynniki $a[k]$ wielomianu przy t^k na tak samo oznaczone współczynniki tegoż wielomianu przy k-tym wielomianie Czebyszewa $(k = 0, 1, \ldots, n)$. Procedura ta wykonuje etap (a) (dla szeregu (1) uciętego na składniku $\alpha_n t^n$, ze zmianą litery α na a) za pomocą $\frac{1}{2}n(n-1)$ dodawań i tyluż połowień.

W procedurze *Aprox* danymi są — liczba całkowita $p \geqslant 0$, rzeczywista $eps > 0$ i tablica współczynników $alfa[k]$ przy t^k $(k = 0, 1, \ldots, p)$ w rozwinięciu funkcji aproksymowanej w szereg potęgowy. Procedura *Aprox* przekształca tablicę *alfa* na współczynniki tegoż szeregu przy kolejnych wielomianach Czebyszewa, znajduje najkrótszą sumę częściową nowego szeregu, aproksymującą go z błędem $\leqslant eps$, stopień q tej sumy i jej współczynniki $alfa[k]$ przy t^k $(k = 0, 1, \ldots, q)$. Stopień p powinien być większy od oczekiwanego q co najmniej o 10-15, aby odrzucone współczynniki szeregu potęgowego nie zniekształciły oszacowania błędu aproksymacji i współczynników wielomianu aproksymującego. Procedura *Aprox* korzysta oczywiście z dwóch poprzednich.

S. PASZKOWSKI (Wrocław)

AN ALGORITHM FOR CALCULATING THE COEFFICIENTS OF AN APPROXIMATING POLYNOMIAL

SUMMARY

We are given a continuous function $f(t)$ in the interval $\langle -1, 1\rangle$ with a known expansion (1) in a power series; it can also be expanded in series (2) with respect to Chebyshev polynomials (3). It is known (cf. [2], § 13.2) that the polynomial $w(t)$ approximating this function with an error

$$\max_{-1\leqslant t\leqslant 1} |f(t)-w(t)|$$

not greater than a given number $\varepsilon > 0$ can be found in the following way:

(a) by calculating the coefficient a_k of series (2),

(b) by finding the least n for which inequality (6) is satisfied,

(c) by finding the coefficients of the polynomials

$$\sum_{k=0}^{n} a_k T_k(t)$$

for $t^0, t^1, \ldots, t^n$.

The paper presents new algorithms for the realization of stages (a) and (c) expressed in the algorithmic language ALGOL 60 in the form of procedure declarations *CP*, *PC* and *Aprox*.

Procedure *CP* for a given integer $n \geqslant 0$ transforms given coefficients $a[k]$ of the polynomial at the kth Chebyshev polynomial into the identically marked coefficients of that polynomial for t^k $(k = 0, 1, \ldots, n)$. This procedure realizes stage (c) by means of $\frac{1}{2}n(n-1)$ subtractions and as many doublings.

Procedure *PC* for a given integer $n \geqslant 0$ transforms given coefficients $a[k]$ of the polynomial for t^k into the identically marked coefficients of that polynomial at the kth Chebyshev polynomial $(k = 0, 1, \ldots, n)$. This procedure realises stage (a) (for series (1) cut off at the component $a_n t^n$ with a change of letter α into a) by means of $\frac{1}{2}n(n-1)$ summations and as many divisions in half.

In procedure *Aprox* we are given an integer $p \geqslant 0$, a real number $eps > 0$ and an array of coefficients $alfa[k]$ for t^k $(k = 0, 1, \ldots, p)$ in the expansion of the function which is being approximated in a power series. Procedure *Aprox* transforms the *alfa* array into the coefficients of that series at the successive Chebyshev polynomials, gives the shortest partial sum of the new series approximating it with an error $< eps$, and gives the degree q of that sum and its coefficients $alfa[k]$ for t^k $(k = 0, 1, \ldots, q)$. The degree p should be greater than the expected q by at least 10-15 lest the rejected coefficients of the power series should distort the estimate of the approximation error and the coefficients of the approximating function. Procedure *Aprox* makes use of course of the preceding two.