12. Обработка результатов эксперимента. Интерполяция функций

Данная глава посвящена решению часто встречающихся на практике задач по обработке реальных количественных экспериментальных данных, полученных в результате всевозможных научных опытов, технических испытаний методом интерполяции. Описаны численные методы интерполирования, а также рассмотрено решение задач интерполирования в Octave.

12.1 Постановка задачи

Напомним читателю задачу интерполирования. На отрезке [a,b] заданы n+1 точки x_0, x_1, x_2, …, x_n $(a=x_0, b=x_n)$, называемые узлами интерполяции, и значения неизвестной функции $f(x)$ в этих точках

$$f(x_0)=y_{0,}f(x_1)=y_{1,}f(x_2)=y_{2,}\ldots,f(x_n)=y_n \qquad (12.1)$$

Требуется построить интерполирующую функцию $F(x)$, которая в узлах интерполяции принимает те же значения, что и f(x):

$$F(x_0)=y_{0,}F(x_1)=y_{1,}F(x_2)=y_{2,}\ldots,F(x_n)=y_n \qquad (12.2)$$

В общей постановке задача может не иметь однозначного решений или совсем не иметь решений. Задача становится однозначной, если функцию $F(x)$ будет искать в виде полинома $F(x)=P_n(x)$ степени n, удовлетворяющий условиям (12.2).

Полученную интерполяционную формулу $y=F(x)$ зачастую используют для приближенного нахождения значений данной функции $f(x)$ в точках x, отличных от узлов интерполирования. Такая операция называется интерполированием функции f(x). При этом различают *интерполирование* в узком смысле, когда $x\in[x0, xn]$, и *экстраполирование*, когда x $x\notin[x0, xn]$.

Рассмотрим некоторые наиболее часто используемые интерполяционные полиномы.

12.1.1 Канонический полином

Будем искать интерполирующую функцию $F(x)$ в виде канонического полинома степени *n*.

$$F(x)=P_n(x)=a_0+a_1x+a_2x^2+\ldots a_nx^n \qquad (12.3)$$

Выбор многочлена степени *n* основан на том факте, что через *n+1* точку проходит единственная кривая степени *n*. Подставив (12.3) в (12.1), получим систему линейных алгебраических уравнений (12.4).

$$\begin{cases} a_0+a_1x_0+a_2x_0^2+\dots a_nx_0^n=y_0 \\ a_0+a_1x_1+a_2x_1^2+\dots a_nx_1^n=y_1 \\ a_0+a_1x_2+a_2x_2^2+\dots a_nx_2^n=y_2 \\ \dots \\ a_0+a_1x_n+a_2x_n^2+\dots a_nx_n^n=y_n \end{cases} \qquad (12.4)$$

Решая эту систему линейных алгебраических уравнений, найдём коэффициенты интерполяционного полинома a_0, a_1, ..., a_n.

12.1.2 Полином Ньютона

И. Ньютон предложил интерполирующую функцию записать в виде следующего полинома n-й степени:

$$F(t)=A_0+A_1(t-x_0)+A_2(t-x_0)(t-x_1)+\dots+A_n(t-x_0)(t-x_1)\dots(t-x_{n-1}) \quad (12.5)$$

Подставим $F(x_0)=y_0$ в (12.5) и вычислим значение коэффициента A_0 $A_0=y_0$.

Подставим $F(x_1)=y_1$ в (12.5), после чего получим соотношение для вычисления A_1:

$F(x_1)=A_0+A_1(x_1-x_0)=y_1$.

Отсюда коэффициент A_1 рассчитывается по формуле:

$$A_1=\frac{y_0-y_1}{x_0-x_1}=y_{01}$$

где y_{01} – разделённая разность первого порядка, которая стремится к первой производной функции при $x_1\to x_0$. По аналогии вводятся и другие разделённые разности первого порядка:

$$y_{02}=\frac{y_0-y_2}{x_0-x_2},\ y_{03}=\frac{y_0-y_3}{x_0-x_3},\dots,y_{0\mathrm{n}}=\frac{y_0-y_n}{x_0-x_n}$$

Подставим соотношение $F(x_2)=y_2$ в (12.5), в результате чего получим

$A_0+A_1(x_2-x_0)+A_2(x_2-x_0)(x_2-x_1)=y_2$,

$y_0+y_{01}(x_2-x_0)+A_2(x_2-x_0)(x_2-x_1)=y_2$.

Отсюда A_2 вычисляется по формуле $A_2=y_{012}=\dfrac{y_{01}-y_{02}}{x_1-x_2}$

здесь y_{012} – разделённая разность второго порядка, эта величина стремится ко второй производной при $x_1\to x_2$. Аналогично вводятся $y_{013}=\dfrac{y_{01}-y_{03}}{x_0-x_3},\ y_{014}=\dfrac{y_{01}-y_{04}}{x_1-x_4},\dots,y_{01\mathrm{n}}=\dfrac{y_{01}-y_{0\mathrm{n}}}{x_1-x_n}$.

Подставим $F(x_3)=y_3$ в (12.5), после чего получим $A_3=y_{0123}=\dfrac{y_{012}-y_{013}}{x_2-x_3}$.

Аналогично можно ввести коэффициенты $y_{0124}=\dfrac{y_{012}-y_{014}}{x_2-x_4},\dots,y_{012\mathrm{n}}=\dfrac{y_{012}-y_{01\mathrm{n}}}{x_2-x_n}$.

Этот процесс будем продолжать до тех пор, пока не вычислим $A_n=y_{012\dots\mathrm{n}}=\dfrac{y_{012\dots\mathrm{n}-1}-y_{01\dots\mathrm{n}-2\mathrm{n}}}{x_{n-1}-x_n}$.

Полученные результаты запишем в табл. 2.1.

Таблица *12.1:* Таблица разделенных разностей полинома Ньютона

x	f(x)	1	2	3	4	…	n
x_0	y_0						
x_1	y_1	y_{01}					
x_2	y_2	y_{02}	y_{012}				
x_3	y_3	y_{03}	y_{013}	y_{0123}			
x_4	y_4	y_{04}	y_{014}	y_{0124}	y_{01234}		
…	…	…	…	…	…	…	…
x_n	y_n	y_{0n}	y_{01n}	y_{012n}	y_{0123n}	…	$y_{012..n}$

В вычислении по формуле (12.5) будут участвовать только диагональные элементы таблицы (т.е. коэффициенты A_i), а все остальные элементы таблицы являются промежуточными и нужны для вычисления диагональных элементов.

12.1.3 Полином Лагранжа

Еще одно представление интерполяционного полинома степени n предложил Лагранж:

$$F(t)=\sum_{i=0}^{n} y_i \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{t-x_j}{x_i-x_j} \quad . \tag{12.6}$$

Напомним читателю, что рассмотренные три способа построения полинома – это три различных формы записи одной и той же функции.

Совет. Полином Лагранжа лучше использовать, если необходимо вычислить значение в небольшом количестве точек. Для расчёта во многих точках рационально использовать полином Ньютона, в котором, можно один раз вычислить значения коэффициентов A_i , после чего можно рассчитать ожидаемое значение в точках по формуле (12.5). При использовании канонического полинома приходится решать систему линейных алгебраических уравнений (12.4), поэтому он используется значительно реже.

12.1.4 Реализация интерполяционного полинома **n**-й степени в Octave

Построить интерполяционный полином n-й степени в Octave можно одним из следующих способов:

1. Средствами языка программирования реализовать один из рассмотренных алгоритмов построения полинома: канонический (см. листинг 12.1), полином Ньютона (см. листинг 12.2), полином Лагранжа (см. листинг 12.3), после чего посчитать значения в нужных точках.
2. Воспользоваться функцией `polyfit(x,y,k)` (в этом случае `k=length(x)-1`) для вычисления коэффициентов полинома, после чего с помощью функции `polyval(A,t)` вычислить значение полинома в необходимых точках.

```
%x — массив абсцисс экспериментальных точек, y — массив
% ординат экспериментальных точек, t — точка в которой
%требуется найти значение.
function s=kanon(x,y,t)
% Вычисление количества точек в массивах x и y
n=length(x);
%Формирование коэффициентов системы уравнений (12.4)
```

```
for i=1:n
for j=1:n
A(i,j)=x(i).^(j-1);
end
end
%Решение системы уравнений (12.4)
a=A^(-1)*y';
%Вычисление значения полинома в точке t по формуле (12.3)
s=0;
for i=1:n
s=s+a(i)*t^(i-1);
end
end
```

Листинг 12.1. Функция kanon для вычисления канонического полинома в точке *t* по экспериментальным значениям (x_i, y_i)

```
%x — массив абсцисс экспериментальных точек, y — массив
% ординат экспериментальных точек, t — точка в которой
%требуется найти значение.
function s=newton(x,y,t)
% Вычисление количества точек в массивах x и y
n=length(x);
% Запись в первый столбец матрицы разделенных разностей
% вектора y
for i=1:n
C(i,1)=y(i);
end
% Формирование матрицы разделенных разностей
for i=2:n
for j=2:n
if (i<j)
C(i,j)=0;
else
C(i,j)=(C(i,j-1)-C(j-1,j-1))/(x(i)-x(j-1));
end
end
end
% Формирование массива коэффициентов полинома Ньютона
for i=1:n
A(i)=C(i,i);
end
% Расчет значения полинома в точке t по формуле (12.5)
s=0;
for i=1:n
p=1;
for j=1:i-1
p=p*(t-x(j));
end
s=s+A(i)*p;
end
end
```

Листинг 12.2. Функция `newton` для вычисления полинома Ньютона в точке *t* по экспериментальным значениям (x_i, y_i)

```
%x — массив абсцисс экспериментальных точек, y — массив
% ординат экспериментальных точек, t — точка в которой
%требуется найти значение.
function s=lagrang(x,y,t)
% Вычисление количества точек в массивах x и y
n=length(x);
s=0;
% Расчет суммы произведений по формуле (12.6)
% для вычисления значения полинома Лагранжа в точке t
for i=1:n
p=1;
for j=1:n
if (j~=i)
p=p*(t-x(j))/(x(i)-x(j));
end
end
s=s+y(i)*p;
end
end
```

Листинг 12.3. Функция `lagrang` для вычисления полинома Лагранжа в точке *t* по экспериментальным значениям (x_i, y_i)

ПРИМЕР 12.1.

В результате эксперимента получена табличная зависимость $y(x)$ (см. табл. 12.2). Построить интерполяционный полином. Вычислить ожидаемое значение в точках 0.5, 0.6 и 0.7, построить график зависимости.

Таблица *12.2:*

x	0.43	0.48	0.55	0.62	0.7	0.75
y	1.63597	1.73234	1.87686	2.03345	2.22846	2.35973

Решение с подробными комментариями представлено на листинге

```
function s=kanon(x,y,t)
% Вычисление количества точек в массивах x и y
n=length(x);
%Формирование коэффициентов системы уравнений (12.4)
for i=1:n
for j=1:n
A(i,j)=x(i).^(j-1);
end
end
%Решение системы уравнений (12.4)
a=A^(-1)*y';
%Вычисление значения полинома в точке t по формуле (12.3)
s=0;
for i=1:n
s=s+a(i)*t^(i-1);
```

```
end
end
function s=newton(x,y,t)
% Вычисление количества точек в массивах x и y
n=length(x);
% Запись в первый столбец матрицы разделенных разностей
% вектора y
for i=1:n
C(i,1)=y(i);
end
% Формирование матрицы разделенных разностей
for i=2:n
for j=2:n
if (i<j)
C(i,j)=0;
else
C(i,j)=(C(i,j-1)-C(j-1,j-1))/(x(i)-x(j-1));
end
end
end
% Формирование массива коэффициентов полинома Ньютона
for i=1:n
A(i)=C(i,i);
end
% Расчет значения полинома в точке t по формуле (12.5)
s=0;
for i=1:n
p=1;
for j=1:i-1
p=p*(t-x(j));
end
s=s+A(i)*p;
end
end
function s=lagrang(x,y,t)
% Вычисление количества точек в массивах x и y
n=length(x);
s=0;
% Расчет суммы произведений по формуле (12.6)
% для вычисления значения полинома Лагранжа в точке t
for i=1:n
p=1;
for j=1:n
if (j~=i)
p=p*(t-x(j))/(x(i)-x(j));
end
end
s=s+y(i)*p;
end
end
%x — массив абсцисс экспериментальных точек, y — массив
```

```
%ординат экспериментальных точек задачи 12.1.
x=[0.43 0.48 0.55 0.62 0.7 0.75];
y=[1.63597 1.73234 1.87686 2.03345 2.22846 2.35973];
r=[0.5 0.6 0.7]
for i=1:3
%Вычисление ожидаемого значения интерполяционного полинома
% Ньютона в точках r=[0.5 0.6 0.7]
rsn(i)=newton(x,y,r(i));
%Вычисление ожидаемого значения канонического
%интерполяционного полинома в точках r=[0.5 0.6 0.7]
rsk(i)=kanon(x,y,r(i));
%Вычисление ожидаемого значения интерполяционного полинома
% Лагранжа в точках r=[0.5 0.6 0.7]
rsl(i)=lagrang(x,y,r(i));
end
rsn
rsk
rsl
%Вычисление ожидаемого значения интерполяционного полинома в
% точках r=[0.5 0.6 0.7] с помощью функции polyfit
A=polyfit(x,y,length(x)-1)
rsp=polyval(A,r)
%Вычисление точек для построения графика интерполяционного
%полинома.
x1=0.43:0.01:0.75;
y1=polyval(A,x1);
%Построение графика.
plot(x,y,'*b;experment;',x1,y1,'-r;interpolation;',...
r,rsp,'pb;f(r);');
grid();
```

Листинг 12.4. Решение задачи 12.1

Результаты представлены ниже

```
>>>r =
0.50000 0.60000 0.70000
>>>rsn =
1.7725 1.9874 2.2285
>>>rsk =
1.7725 1.9874 2.2285
>>>rsl =
1.7725 1.9874 2.2285
>>>A =
0.44180 -1.17180 1.70415 -0.18866 1.38721 0.97243
>>>rsp =
1.7725 1.9874 2.2285
```

Как и следовало ожидать, все четыре используемых метода построения интерполяционного полинома пятой степени дали одни и те же значения.

На рис. 12.1 представлено графическое решение задачи 12.1. Подобным образом можно подбирать коэффициенты интерполяционного полинома и для других задач.

Полиномиальная интерполяция не всегда дает удовлетворительные результаты при аппроксимации зависимостей. При представлении полиномами возможна большая погрешность на концах этих кривых. Несмотря на выполнение условий в узлах, интерполяционная функция может иметь значительное отклонение от аппроксимируемой кривой между узлами. При этом повышение степени интерполяционного полинома приводит не к уменьшению, а к увеличению погрешности. Решение этой проблемы предложено теорией сплайн-интерполяции (от английского слова spline - рейка, линейка).

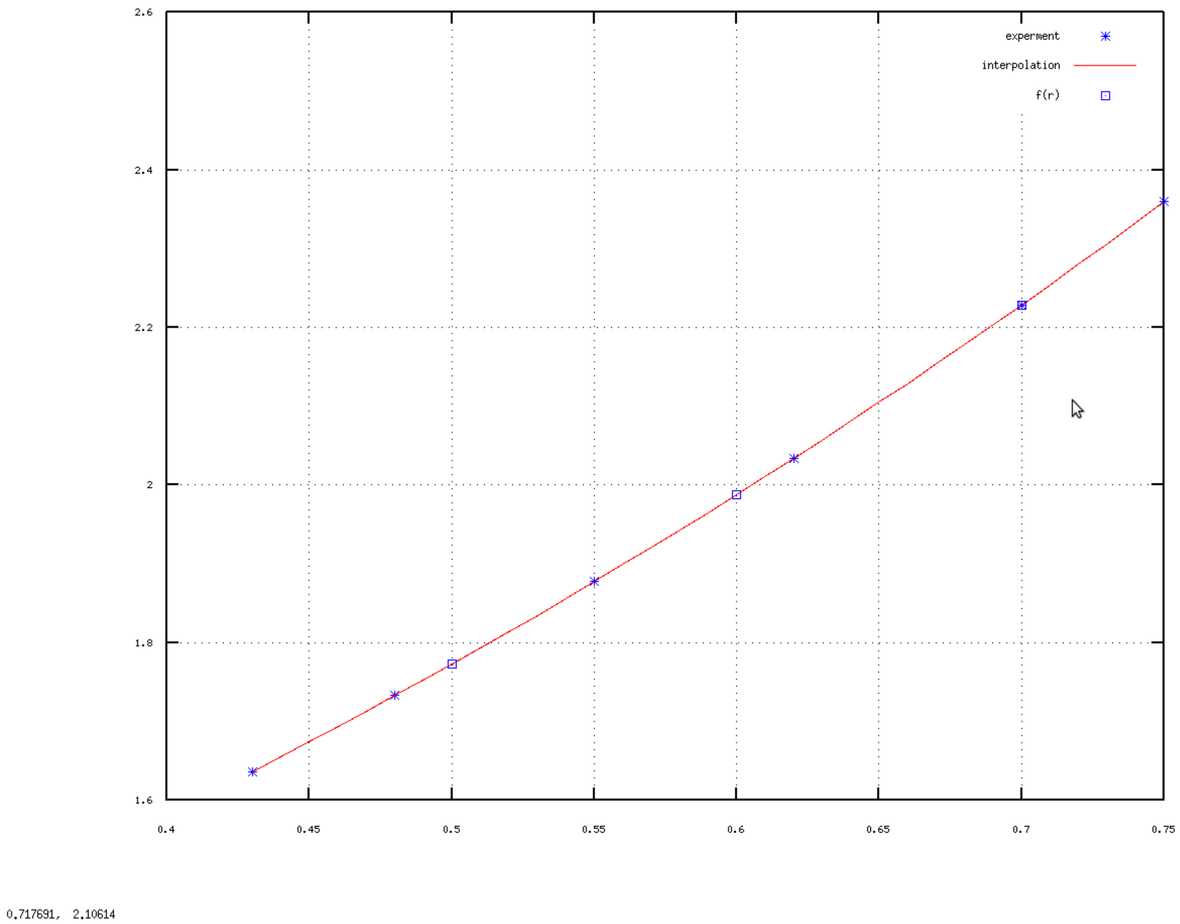

Рисунок *12.1*

12.2 Интерполяция сплайнами

Рассмотрим один из наиболее распространенных вариантов интерполяции кубическими сплайнами. Было установлено [], что недеформируемая линейка между соседними углами проходит по линии, удовлетворяющей уравнению

$$\varphi^{IV}(x)=0 \quad . \tag{12.7}$$

Функцию $\varphi(x)$ будем использовать для интерполяции зависимости $y(x)$, заданной на интервале (a,b) в узлах $a=x_0, x_1 \ldots, b=x_n$ значениями $y_{0,} y_{1,} \ldots y_n$.

Кубическим сплайном, интерполирующим на отрезке [a,b] данную функцию y(x), называется функция [9]

$$g_k(s)=a_k+b_k(s-x_k)+c_k(s-x_k)^2+d_k(s-x_k)^{3,} s\in[x_{k-1}, x_k], k=1,2,\ldots,n, \quad .(12.8)$$

удовлетворяющая следующим условиям:

- $g_k(x_k)=y_k; g_k(x_{k-1})=y_{k-1}$ (условие интерполяции в узлах сплайна);
- функция $g(x)$ дважды непрерывно дифференцируема на интервале $[a,b]$;
- на концах интервала функция g должна удовлетворять следующим соотношениям

$g_1''(a)=g_k''(b)=0$.

Для построения интерполяционного сплайна необходимо найти 4n коэффициентов a_k , b_k , c_k , d_k (k=1,2,…,n).

Из определения сплайна получаем n+1 соотношение (12.9)

$$g_1(x_0)=y_0,\ g_k(x_k)=y_k,\ k=1,2,\ldots,n \quad . \tag{12.9}$$

Из условий гладкой стыковки звеньев сплайна (во внутренних узловых точках совпадают значения двух соседних звеньев сплайна[1], их первые и вторые производные) получаем еще ряд соотношений (12.10 – 12.11) [9]:

$$\begin{array}{l} g_{k-1}(x_{k-1})=g_k(x_k) \\ g_{k-1}'(x_{k-1})=g_k'(x_k), \quad k=2,3,\ldots,n \quad , \\ g_{k-1}''(x_{k-1})=g_k''(x_k) \end{array} \tag{12.10}$$

$$g_1''(x_0)=0,\ g_n''(x_n)=0 \quad . \tag{12.11}$$

Соотношения (12.9) – (12.11) образуют 4n соотношений для нахождения коэффициентов сплайна. Подставляя выражения функций (12.8) и их производных (2.12)

$$\begin{array}{l} g_k'(s)=b_k+2\mathrm{c}_k(s-x_k)+3\mathrm{d}_k(s-x_k)^2, \\ g_k''(s)=2\mathrm{c}_k+6\mathrm{d}_k(s-x_k) \end{array} \tag{12.12}$$

в соотношения (12.9) – (12.11) и принимая во внимание соотношение

$$h_k=x_k-x_{k-1}, \quad k=1,2,\ldots,n \quad . \tag{12.13}$$

получим следующую систему уравнений (12.14) – (12.20)

$$a_1-b_1h_1+c_1h_1^2-d_1h_1^3=y_0 \quad , \tag{12.14}$$

$$a_k=y_k, \quad k=1,2,\ldots,n \quad , \tag{12.15}$$

$$a_{k-1}=a_k-b_kh_k+c_kh_k^2-d_kh_k^3, \quad k=2,3,\ldots,n \quad , \qquad 12.16)$$

$$b_{k-1}=b_k-2\mathrm{c}_kh_k+3\mathrm{d}_kh_k^2, \quad k=2,3,\ldots,n \quad , \tag{12.17}$$

$$c_{k-1}=c_k-3\mathrm{d}_kh_k, \quad k=2,3,\ldots,n \quad , \tag{12.18}$$

$$c_1-3\mathrm{d}_1h_1=0 \quad , \tag{12.19}$$

$$c_n=0 \quad . \tag{12.20}$$

Задача интерполяции свелась к решению системы (12.14-12.20). Из соотношения (12.15) следует, что все коэффициенты $a_k=y_k,\ k=1,2,\ldots,n$. Подставив соотношения (12.14), (12.15) в (12.16) и используя фиктивный коэффициент $c_0=0$, получим соотношение между b_k , c_k и d_k $b_kh_k-c_kh_k^2+d_kh_k^3=y_k-y_{k-1}$

Отсюда коэффициенты b_k вычисляются по формуле

$$b_k=\frac{y_k-y_{k-1}}{h_k}+c_kh_k-d_kh_k^{2,} \quad k=1,2,\ldots,n \quad . \tag{12.21}$$

---

1 Звеном сплайна называется функция $g_i(x)$ на интервале $[x_{i-1},x_i]$

Из (12.18) и (12.19) выразим d_k через c_k (с учётом коэффициента $c_0=0$)

$$d_k=\frac{c_k-c_{k-1}}{3\mathrm{h}_k},\quad k=1,2,\ldots,n \quad . \tag{12.22}$$

Подставим (12.22) в (12.21)

$$b_k=\frac{y_k-y_{k-1}}{h_k}+\frac{2}{3}c_k h_k+\frac{1}{3}h_k c_{k-1},\quad k=1,2,\ldots,n \quad . \tag{12.23}$$

Введем обозначение

$$l_k=\frac{y_k-y_{k-1}}{h_k},\quad k=1,2,\ldots,n \quad , \tag{12.24}$$

после чего соотношение (12.23) примет вид:

$$b_k=l_k+\frac{2}{3}c_k h_k+\frac{1}{3}h_k c_{k-1},\quad k=1,2,\ldots,n \quad . \tag{12.25}$$

Подставим (12.25) и (12.22) в соотношение (12.17), получим систему относительно c_k

$$b_{k-1}c_{k-2}+2(h_{k-1}+h_k)c_k=3(l_k-l_{k-1}),\quad k=2,3,\ldots,n \quad . \tag{12.26}$$

$$c_0=0,\quad c_n=0 \quad . \tag{12.27}$$

Систему (12.26) можно решить, используя метод прогонки (http://ru.wikipedia.org/wiki/Метод_прогонки). Этот метод сводится к нахождению прогоночных коэффициентов по формулам прямой прогонки.

$$\delta_1=-\frac{1}{2}\frac{h_2}{h_1+h_2},\quad \lambda_1=\frac{3}{2}\frac{l_2-l_1}{h_1+h_2} \quad , \tag{12.28}$$

$$\begin{matrix}\delta_{k-1}=-\dfrac{h_k}{2\mathrm{h}_{k-1}+2\mathrm{h}_k+h_{k-1}\delta_{k-2}}, \\ \lambda_{k-1}=\dfrac{3\mathrm{l}_k-3\mathrm{l}_{k-1}-h_{k-1}\lambda_{k-2}}{2\mathrm{h}_{k-1}+2\mathrm{h}_k+h_{k-1}\delta_{k-2}}\end{matrix}\quad k=3,4,\ldots,n \quad , \tag{12.29}$$

а затем к нахождению искомых коэффициентов c_k по формулам обратной прогонки

$$c_{k-1}=\delta_{k-1}c_k+\lambda_{k-1},\quad k=n,,n-1,\ldots,2. \quad , \tag{12.30}$$

После нахождения коэффициентов с по формуле (12.30), находим *b* и *d* по формулам (12.22), (12.25).

Таким образом, алгоритм расчёта коэффициентов интерполяционного сплайна можно свести к следующим шагам.

Шаг 1. Ввод значений табличной зависимости $y(x)$, массивов *x* и *y*.

Шаг 2. Расчёт элементов массивов *h* и *l* по формулам (12.13) и (12.25).

Шаг 3. Расчёт массивов прогоночных коэффициентов δ и λ по формулам (12.28), (12.29).

Шаг 4. Расчёт массивов коэффициентов c по формуле (12.30).

Шаг 5. Расчёт массивов коэффициентов b по формуле (12.25).

Шаг 6. Расчёт массивов коэффициентов d по формуле (12.22).

После этого в формулу (12.8) можно подставлять любую точку *s* и вычислять ожидаемое значение.

Расчёт коэффициентов кубического сплайна очень громоздкий и зачастую на практике вместо кубического сплайна используется *линейная интерполяция* (*линейный сплайн*).

Использование линейного сплайна оправдано в случае, если необходимо просто вычислить значение в определённых точках и нет требования непрерывности производных интерполяционной функции.

В случае линейной интерполяции в качестве сплайна выступает линейная функция

$$f_k(s)=a_k+b_k s, s\in[x_{k-1}, x_k], k=1,2,\ldots,n, \quad , \qquad (12.31)$$

удовлетворяющая условию интерполяции в узлах сплайна $f_k(x_k)=y_k; f_k(x_{k-1})=y_{k-1}$. Коэффициенты a и b в этом случае рассчитываются по формулам (2.33), которые получаются из уравнения прямой, проходящей через две точки с координатами (x_{k-1}, y_{k-1}) , (x_k, y_k) .

$$a_k=y_{k-1}-\frac{y_k-y_{k-1}}{x_k-x_{k-1}}x_{k-1}, b_k=\frac{y_k-y_{k-1}}{x_k-x_{k-1}} \quad , \qquad (12.32)$$

Найдя коэффициенты линейного сплайна, можно рассчитать значения в любой точке интервала $[x_0, x_n]$. Линейная интерполяция дает достаточно хорошие результаты при практическом счёте внутри интервала $[x_0, x_n]$, когда от получаемой функции не требуют дополнительных свойств (дифференцируемости и т.д.).

Рассмотрим реализацию сплайн-интерполяции в Octave. Это можно сделать запрограммировав рассмотренные выше методы сплайн-интерполирования в Octave или воспользовавшись функцией `interp1`

`interp1(x, y, xi, '<метод>')`,

где `x` – массив абсцисс экспериментальных точек, `y` – массив ординат экспериментальных точек, `xi` – точки, в которых необходимо вычислить значение с помощью сплайна, `method` – определяет метод построения сплайна, для реализации сплайн-интерполяции параметр `method` может принимать одно из следующих значений: `'linear'` – линейная интерполяция, `'spline'` – кубический сплайн.

Рассмотрим несколько практических задач.

ПРИМЕР 12.2.

В результате опыта холостого хода определена зависимость потребляемой из сети мощности (P_0 , Вт) от входного напряжения (U_1 , В) для асинхронного двигателя MTH111-6 (см. табл. 12.3). Построить график интерполяционной зависимости. Вычислить ожидаемое значение мощности при $U_1=145, 155, 175\,B$.

Таблица *12.3:*

Напряжение U_1 , В	132	140	150	162	170	180
Мощность P_0 , Вт	330	350	385	425	450	485

Реализуем рассмотренный ранее алгоритм кубического сплайна. Решение с комментариями представлено на листинге 12.5.

```
function [b, c,d]=koef_spline(x,y)
% Функция предназначена для вычисления коэффициентов сплайна,
% здесь x,y - массивы абсцисс и ординат экспериментальных
% точек, b, c,d - коэффициенты сплайна, рассчитываемые по
% формулам (12.21), (12.23), (12.27), (12.30)
n=length(x);
for k=2:n
```

```
h(k)=x(k)-x(k-1);
end
for k=2:n
l(k)=(y(k)-y(k-1))/h(k);
end
delt(2)=-h(3)/(2*(h(3)+h(2)));
lyam(2)=1.5*(l(3)-l(2))/(h(3)+h(2));
for k=4:n
delt(k-1)=-h(k)/(2*(h(k-1)+h(k))+h(k-1)*delt(k-2));
lyam(k-1)=(3*(l(k)-l(k-1))-h(k-1)*delt(k-2))/(2*(h(k-1)+...
h(k))+h(k-1)*delt(k-2));
end
c(n)=0;
for k=n:-1:3
c(k-1)=delt(k-1)*c(k)+delt(k-1);
end
for k=2:n
d(k)=(c(k)-c(k-1))/3/h(k);
b(k)=l(k)+(2*c(k)*h(k)+h(k)*c(k-1))/3;
end
end
function z=my_spline(x,y,t)
% Вычисление значение кубического сплайна в точке t,
% здесь x,y - массивы абсцисс и ординат экспериментальных
% точек
[b,c,d]=koef_spline(x,y);
n=length(x);
a=y;
%j — номер интервала, которому принадлежит точка t.
if t>x(n-1)
      j=n;
else
      for i=2:n-1
           if t<=x(i)
                j=i;
                break
      end
 end
end
z=a(j)+b(j)*(t-x(j))+c(j)*(t-x(j))^2+d(j)*(t-x(j))^3;
end
%Экспериментальные точки.
U1=[132 140 150 162 170 180];
P0=[330 350 385 425 450 485];
%Точки, в которых надо посчитать ожидаемое значение сплайна.
x=[145 155 175];
for i=1:3
%Расчёт ожидаемого значения с помощью функции my_spline.
y(i)=(U1,P0,x(i));
end
%Вычисление значений для построения графика слайна.
```

```
U2=132:1:180;
for i=1:length(U2)
P2(i)=my_spline(U1,P0,U2(i));
end
x
y
%Построение графика.
plot(U1,P0,'*b;experiment;',U2,P2,...
'-r;spline;',x,y,'pb;points;');
grid on;
```

Листинг 12.5. Решение задачи12.2

Ниже приведены результаты работы программы, а на рис. 12.2 можно увидеть графическое решение задачи.

```
>>>x =
145 155 175
>>>y =
373.19 408.77 471.15
```

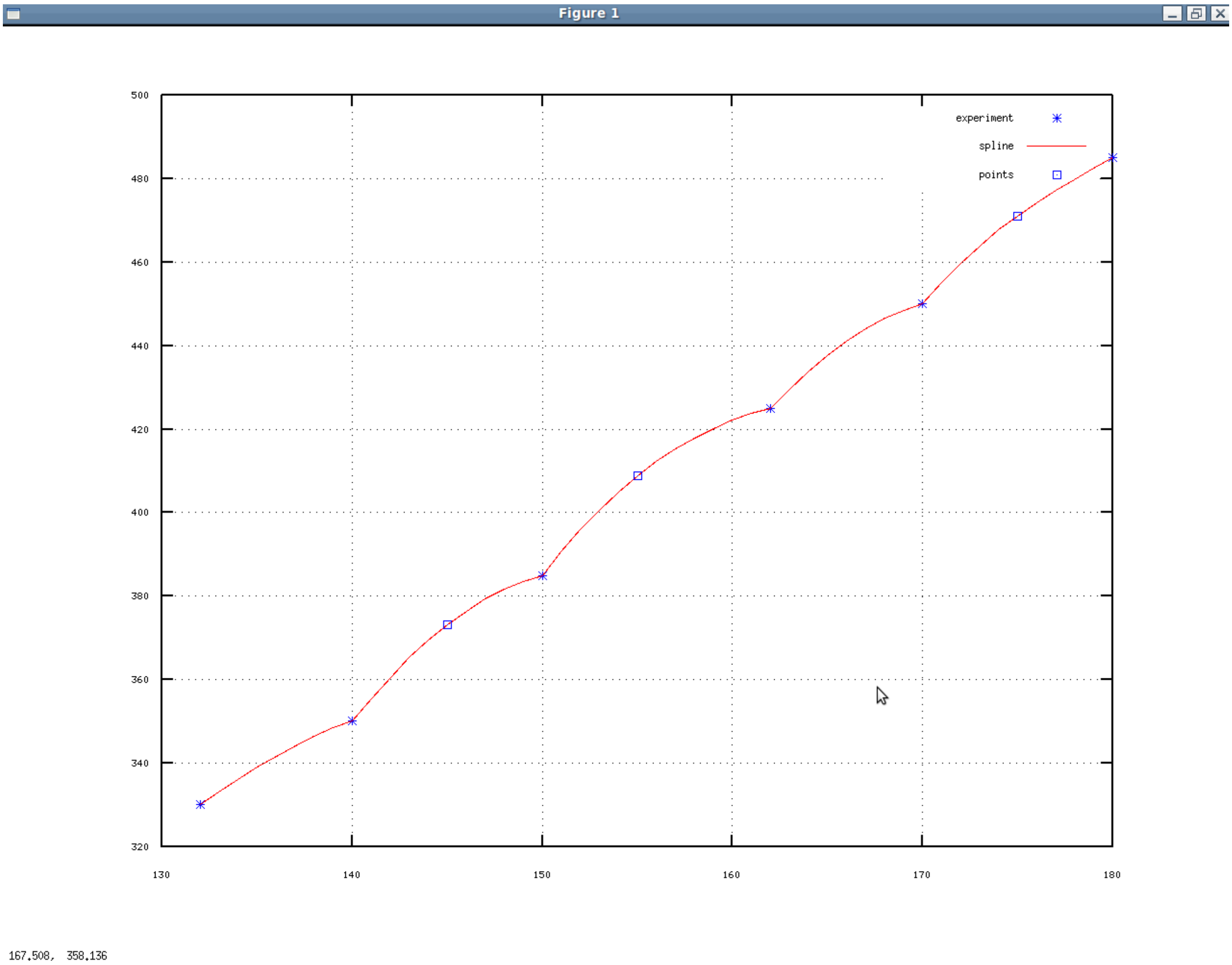

Рисунок 12.2.

В следующей задаче воспользуемся встроенными функциями Octave.

ПРИМЕР 12.3.

В результате опыта холостого хода определена зависимость $u(x)$ (см. табл. 12.4). Построить график интерполяционной зависимости. Вычислить ожидаемое значение мощности при $x=0.308, 0.325, 0.312$.

Таблица *12.4:*

x	0.298	0.303	0.31	0.317	0.323	0.33
u	3.25578	3.17639	3.1218	3.04819	2.98755	2.9195

Для решения задачи воспользуемся функцией interp1. На листинге 12.6 представлено решение задачи 12.3.

```
%Экспериментальные точки
x=[0.298 0.303 0.31 0.317 0.323 0.33];
u=[3.25578 3.17639 3.1218 3.04819 2.98755 2.9195];
%Точки, в которых надо посчитать ожидаемое значение.
X1=[0.308 0.312 0.325];
%Расчёт значений в точках 0.308, 0.312, 0.325 с помощью
% кубического сплайна.
u1s=interp1(x,u,x1,'spline')
%Расчёт значений в точках 0.308, 0.312, 0.325 с помощью
% линейного сплайна.
u1l=interp1(x,u,x1,'linear')
%Вычисление значений для построения графиков линейного и
% кубического слайнов.
xi=0.298:0.002:0.33;
uxis=interp1(x,u,xi,'spline');
uxil=interp1(x,u,xi,'linear');
%Построение графика
plot(x,u,'*b;experimentl;',xi,uxil,'-r;linear spline;',...
xi,uxis,'-b;cubic spline;',x1,u1s,...
'pr;points (cubic spline);',x1,u1l,...
'<b;points (linear spline);');
axis([0.29,0.34,2.8,3.3]);
grid on;
```

Листинг 12.6. Решение задачи 12.3.

Ниже приведены результаты работы программы, а на рис. 12.3 – графическое решение задачи.

```
>>>u1s =

3.1370 3.1031 2.9685

>>>u1l =

3.1374 3.1008 2.9681
```

Как и при решении задачи аппроксимации (подбора кривой методом наименьших квадратов), при построении интерполяционных зависимостей, необходимо первоначально поставить и решить задачу с точки зрения математики, и только потом использовать функции пакета Octave.

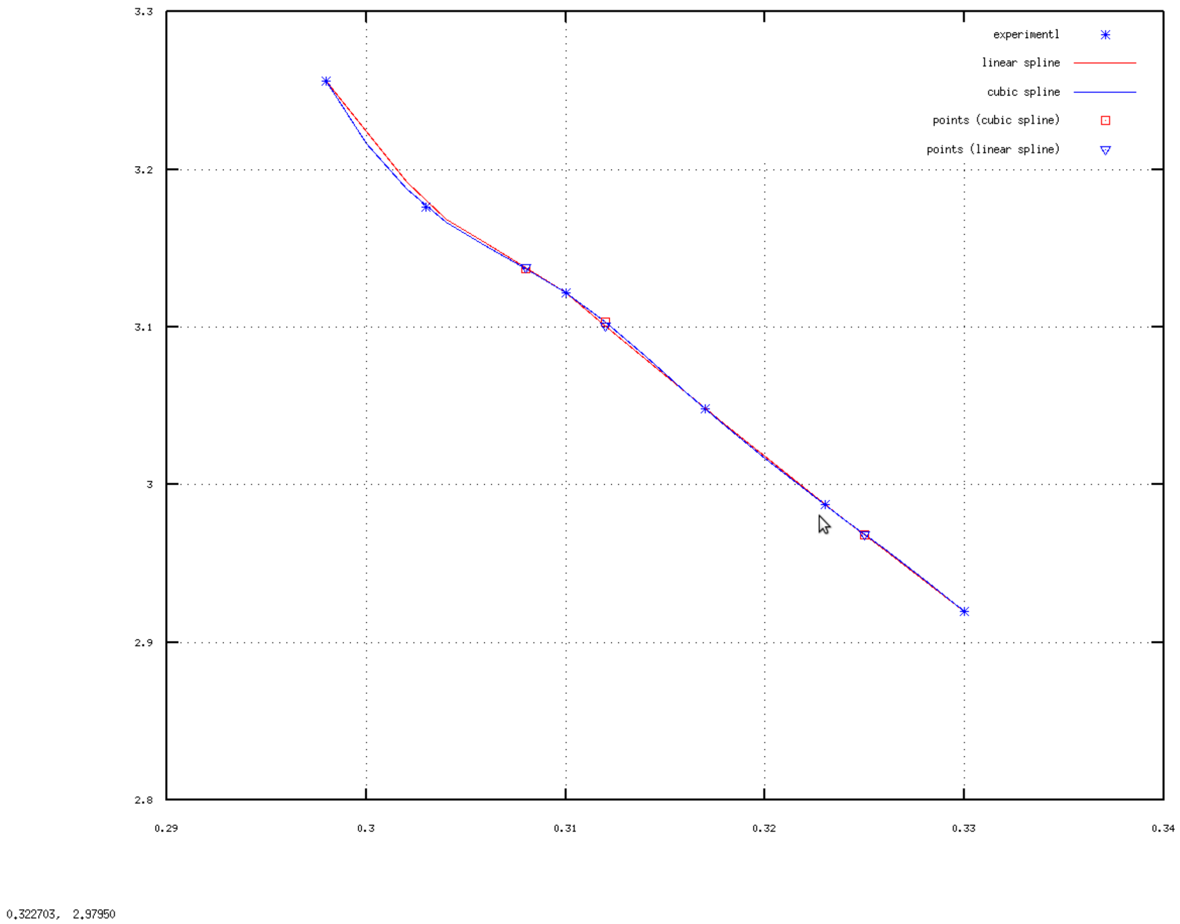

Рисунок *12.3*

Этой задачей мы завершаем краткое введение в Octave. В книге были рассмотрены только некоторые функции и методы решения математических и инженерных задач. Далее читатель может самостоятельно продолжить изучение Octave. Официальная страница справки http://www.gnu.org/software/octave/doc/interpreter/. Следует помнить, что существует огромное количество расширений к пакету, описание пакетов расширений приведено на странице http://octave.sourceforge.net.