

清华大学

计算机系列教材

徐士良 马尔妮 编著

常用算法程序集

(C/C++描述) (第五版)



清华大学出版社

013034441

TB115
87-5

清华大学 计算机系列教材

徐士良 马尔妮 编著

常用算法程序集

(C/C++描述) (第五版)



TB115
87-5



北航

C1641769

清华大学出版社
北京

133900810

内 容 简 介

本书是针对工程中常用的行之有效的算法而编写的,主要内容包括多项式的计算、复数运算、随机数的产生、矩阵运算、矩阵特征值与特征向量的计算、线性代数方程组的求解、非线性方程与方程组的求解、插值与逼近、数值积分、常微分方程组的求解、数据处理、极值问题的求解、数学变换与滤波、特殊函数的计算、排序、查找等。

书中所有的算法程序均用 C/C++ 描述,可从清华大学出版社网站(www.tup.com.cn)下载。

本书可供广大科研人员、工程技术人员及管理工作者阅读使用,也可作为高等院校师生的参考书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

常用算法程序集: C/C++ 描述/徐士良,马尔妮编著.--5 版.--北京:清华大学出版社,2013.5

清华大学计算机系列教材

ISBN 978-7-302-30343-5

I. ①常… II. ①徐… ②马… III. ①工程计算程序—程序设计—高等学校—教材 ②C 语言—程序设计—高等学校—教材 IV. ①TP319 ②TP312

中国版本图书馆 CIP 数据核字(2012)第 240828 号

责任编辑:白立军 徐跃进

封面设计:何凤霞

责任校对:李建庄

责任印制:何 芊

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社 总 机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课件下载: <http://www.tup.com.cn>, 010-62795954

印 刷 者:三河市君旺印装厂

装 订 者:三河市新茂装订有限公司

经 销:全国新华书店

开 本:185mm×260mm

印 张:37.75

字 数:939 千字

版 次:1989 年 11 月第 1 版 2013 年 4 月第 5 版

印 次:2013 年 4 月第 1 次印刷

印 数:1~3000

定 价:69.00 元

产品编号:049296-01

序

“清华大学计算机系列教材”已经出版发行了 30 余种,包括计算机科学与技术专业的基础数学、专业技术基础和专业等课程的教材,覆盖了计算机科学与技术专业本科生和研究生的主要教学内容。这是一批至今发行数量很大并赢得广大读者赞誉的书籍,是近年来出版的大学计算机专业教材中影响比较大的一批精品。

本系列教材的作者都是我熟悉的教授与同事,他们长期在第一线担任相关课程的教学工作,是一批很受本科生和研究生欢迎的任课教师。编写高质量的计算机专业本科生(和研究生)教材,不仅需要作者具备丰富的教学经验和科研实践,还需要对相关领域科技发展前沿的正确把握和了解。正因为本系列教材的作者们具备了这些条件,才有了这批高质量优秀教材的产生。可以说,教材是他们长期辛勤工作的结晶。本系列教材出版发行以来,从其发行的数量、读者的反映、已经获得的国家级与省部级的奖励,以及在各个高等院校教学中所发挥的作用上,都可以看出本系列教材所产生的社会影响与效益。

计算机学科发展异常迅速,内容更新很快。作为教材,一方面要反映本领域基础性、普遍性的知识,保持内容的相对稳定性;另一方面,又需要跟踪科技的发展,及时地调整和更新内容。本系列教材都能按照自身的需要及时地做到这一点。如王爱英教授等编著的《计算机组成与结构》、戴梅萼教授等编著的《微型计算机技术及应用》都已经出版了第四版,严蔚敏教授的《数据结构》也出版了三版,使教材既保持了稳定性,又达到了先进性的要求。

本系列教材内容丰富,体系结构严谨,概念清晰,易学易懂,符合学生的认知规律,适合于教学与自学,深受广大读者的欢迎。系列教材中多数配有丰富的习题集、习题解答、上机及实验指导和电子教案,便于学生理论联系实际地学习相关课程。

随着我国进一步的开放,我们需要扩大国际交流,加强学习国外的先进经验。在大学教材建设上,我们也应该注意学习和引进国外的先进教材。但是,“清华大学计算机系列教材”的出版发行实践以及它所取得的效果告诉我们,在当前形势下,编写符合国情的具有自主版权的高质量教材仍具有重大意义和价值。它与国外原版教材不仅不矛盾,而且是相辅相成的。本系列教材的出版还表明,针对某一学科培养的要求,在教育部等上级部门的指导下,有计划地组织任课教师编写系列教材,还能促进对该学科科学、合理的教学体系和内容的研究。

我希望今后有更多、更好的我国优秀教材出版。

清华大学计算机系教授,中国科学院院士

张钹

前 言

本书是针对工程中常用的行之有效的算法而编写的,并且根据算法的分类以及使用特点做了精心的组织和安排。本书具有以下几个特点:

(1) 书中除收集了传统的算法外,还根据作者工作的经验和近年来数值计算的发展,选取了一些新的、实用的算法。可以说,书中各章几乎都有一些新的算法。

(2) 书中所有的算法程序都经过认真的调试(在 Visual C++ 6.0 环境下)。

(3) 书中收集的算法都是行之有效的,基本可以满足解决工程中各种实际问题的需要。

书中所有程序均采用面向过程的 C 语言描述。但对于有些问题,为了便于读者直接使用,在用 C 语言描述的基础上,还采用了面向对象的 C++ 语言描述,将若干同类算法封装在一个类中。例如,在本书中,将复数运算封装成一个类,将实系数多项式运算封装成一个类,将复系数多项式运算封装成一个类等。

限于作者水平,书中难免有错误之处,恳请读者批评指正。

作 者

2013 年 2 月

目 录

第 1 章 多项式计算	1
1.1 一维多项式求值	1
1.2 一维多项式多组求值	2
1.3 二维多项式求值	5
1.4 复系数多项式求值	6
1.5 多项式相乘	8
1.6 复系数多项式相乘	9
1.7 多项式相除	11
1.8 复系数多项式相除	13
1.9 实系数多项式类	16
1.10 复系数多项式类	21
第 2 章 复数运算	29
2.1 复数乘法	29
2.2 复数除法	30
2.3 复数乘幂	31
2.4 复数的 n 次方根	32
2.5 复数指数	34
2.6 复数对数	35
2.7 复数正弦	36
2.8 复数余弦	37
2.9 复数类	39
第 3 章 随机数的产生	44
3.1 产生 $0\sim 1$ 之间均匀分布的一个随机数	44
3.2 产生 $0\sim 1$ 之间均匀分布的随机数序列	45
3.3 产生任意区间内均匀分布的一个随机整数	46
3.4 产生任意区间内均匀分布的随机整数序列	48
3.5 产生任意均值与方差的正态分布的一个随机数	49
3.6 产生任意均值与方差的正态分布的随机数序列	50
第 4 章 矩阵运算	53
4.1 实矩阵相乘	53
4.2 复矩阵相乘	54
4.3 一般实矩阵求逆	57
4.4 一般复矩阵求逆	60
4.5 对称正定矩阵的求逆	65

4.6	托伯利兹矩阵求逆的特兰特方法	68
4.7	求一般行列式的值	72
4.8	求矩阵的秩	74
4.9	对称正定矩阵的乔里斯基分解与行列式求值	76
4.10	矩阵的三角分解	78
4.11	一般实矩阵的 QR 分解	81
4.12	一般实矩阵的奇异值分解	85
4.13	求广义逆的奇异值分解法	98
第 5 章	矩阵特征值与特征向量的计算	102
5.1	约化对称矩阵为对称三对角阵的豪斯荷尔德变换法	102
5.2	求对称三对角阵的全部特征值与特征向量	106
5.3	约化一般实矩阵为赫申伯格矩阵的初等相似变换法	110
5.4	求赫申伯格矩阵全部特征值的 QR 方法	112
5.5	求实对称矩阵特征值与特征向量的雅可比法	118
5.6	求实对称矩阵特征值与特征向量的雅可比过关法	124
第 6 章	线性代数方程组的求解	128
6.1	求解实系数方程组的全选主元高斯消去法	128
6.2	求解实系数方程组的全选主元高斯-约当消去法	131
6.3	求解复系数方程组的全选主元高斯消去法	134
6.4	求解复系数方程组的全选主元高斯-约当消去法	137
6.5	求解三对角线方程组的追赶法	141
6.6	求解一般带型方程组	144
6.7	求解对称方程组的分解法	148
6.8	求解对称正定方程组的平方根法	152
6.9	求解托伯利兹方程组的列文逊方法	155
6.10	高斯-赛德尔迭代法	159
6.11	求解对称正定方程组的共轭梯度法	161
6.12	求解线性最小二乘问题的豪斯荷尔德变换法	164
6.13	求解线性最小二乘问题的广义逆法	167
6.14	求解病态方程组	169
第 7 章	非线性方程与方程组的求解	172
7.1	求非线性方程实根的对分法	172
7.2	求非线性方程一个实根的牛顿法	174
7.3	求非线性方程一个实根的埃特金迭代法	176
7.4	求非线性方程一个实根的试位法	178
7.5	求非线性方程一个实根的连分式法	180
7.6	求实系数代数方程全部根的 QR 方法	184
7.7	求实系数代数方程全部根的牛顿下山法	186
7.8	求复系数代数方程全部根的牛顿下山法	191

7.9	求非线性方程组一组实根的梯度法	196
7.10	求非线性方程组一组实根的拟牛顿法	199
7.11	求非线性方程组最小二乘解的广义逆法	204
7.12	求非线性方程一个实根的蒙特卡罗法	210
7.13	求实函数或复函数方程一个复根的蒙特卡罗法	212
7.14	求非线性方程组一组实根的蒙特卡罗法	215
第8章	插值与逼近	218
8.1	一元全区间插值	218
8.2	一元三点插值	220
8.3	连分式插值	222
8.4	埃尔米特插值	224
8.5	埃特金逐步插值	226
8.6	光滑插值	228
8.7	第一种边界条件的三次样条函数插值、微商与积分	233
8.8	第二种边界条件的三次样条函数插值、微商与积分	238
8.9	第三种边界条件的三次样条函数插值、微商与积分	242
8.10	二元三点插值	248
8.11	二元全区间插值	250
8.12	最小二乘曲线拟合	253
8.13	切比雪夫曲线拟合	258
8.14	最佳一致逼近的里米兹方法	262
8.15	矩形域的最小二乘曲面拟合	266
第9章	数值积分	274
9.1	变步长梯形求积法	274
9.2	变步长辛卜生求积法	276
9.3	自适应梯形求积法	278
9.4	龙贝格求积法	280
9.5	计算一维积分的连分式法	283
9.6	高振荡函数求积法	286
9.7	勒让德-高斯求积法	289
9.8	拉盖尔-高斯求积法	292
9.9	埃尔米特-高斯求积法	294
9.10	切比雪夫求积法	296
9.11	计算一维积分的蒙特卡罗法	298
9.12	变步长辛卜生二重积分法	300
9.13	计算多重积分的高斯方法	303
9.14	计算二重积分的连分式法	307
9.15	计算多重积分的蒙特卡罗法	311

第 10 章 常微分方程组的求解	313
10.1 全区间积分的定步长欧拉方法	313
10.2 积分一步的变步长欧拉方法	316
10.3 全区间积分的维梯方法	319
10.4 全区间积分的定步长龙格-库塔方法	322
10.5 积分一步的变步长龙格-库塔方法	325
10.6 积分一步的变步长基尔方法	328
10.7 全区间积分的变步长默森方法	333
10.8 积分一步的连分式法	337
10.9 全区间积分的双边法	342
10.10 全区间积分的阿当姆斯预报校正法	347
10.11 全区间积分的哈明方法	352
10.12 积分一步的特雷纳方法	357
10.13 积分刚性方程组的吉尔方法	362
10.14 求解二阶微分方程边值问题的差分法	377
第 11 章 数据处理	382
11.1 随机样本分析	382
11.2 一元线性回归分析	385
11.3 多元线性回归分析	388
11.4 逐步回归分析	392
11.5 半对数数据相关	403
11.6 对数数据相关	405
第 12 章 极值问题的求解	409
12.1 一维极值连分式法	409
12.2 n 维极值连分式法	412
12.3 不等式约束线性规划问题	416
12.4 求 n 维极值的单单调优法	420
12.5 求约束条件下 n 维极值的复形调优法	426
第 13 章 数学变换与滤波	434
13.1 傅里叶级数逼近	434
13.2 快速傅里叶变换	437
13.3 快速沃什变换	443
13.4 五点三次平滑	446
13.5 离散随机线性系统的卡尔曼滤波	447
13.6 α - β - γ 滤波	454
第 14 章 特殊函数的计算	459
14.1 伽马函数	459
14.2 不完全伽马函数	461
14.3 误差函数	464

14.4	第一类整数阶贝塞耳函数	465
14.5	第二类整数阶贝塞耳函数	470
14.6	变型第一类整数阶贝塞耳函数	475
14.7	变型第二类整数阶贝塞耳函数	479
14.8	不完全贝塔函数	482
14.9	正态分布函数	486
14.10	t -分布函数	487
14.11	χ^2 -分布函数	489
14.12	F -分布函数	490
14.13	正弦积分	492
14.14	余弦积分	493
14.15	指数积分	495
14.16	第一类椭圆积分	497
14.17	第二类椭圆积分	500
第 15 章	排序	503
15.1	冒泡排序	503
15.2	快速排序	509
15.3	希尔排序	518
15.4	堆排序	525
15.5	结构排序	533
15.6	磁盘文件排序	546
15.7	拓扑分类	550
第 16 章	查找	554
16.1	结构体数组的顺序查找	554
16.2	磁盘随机文本文件的顺序查找	562
16.3	有序数组的对分查找	565
16.4	按关键字成员有序的结构体数组的对分查找	572
16.5	按关键字有序的磁盘随机文本文件的对分查找	583
16.6	磁盘随机文本文件的字符串匹配	587
参考文献		591

第 1 章 多项式计算

1.1 一维多项式求值

【功能】 计算多项式

$$p(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1x + a_0$$

在指定点 x 处的函数值。

【方法说明】 首先将多项式表述成如下嵌套形式：

$$p(x) = (\cdots((a_{n-1}x + a_{n-2})x + a_{n-3})x + \cdots + a_1)x + a_0$$

然后从里往外一层一层地进行计算。其递推计算公式如下：

$$u = a_{n-1}$$

$$u = u * x + a_i, \quad i = n-2, \cdots, 1, 0$$

最后得到的 u 即是多项式值 $p(x)$ 。

【函数语句与形参说明】

```
double plyv(a, n, x)
```

形参与函数类型	参数意义
double a[n]	存放 $n-1$ 次多项式的 n 个系数
int n	多项式的项数
double x	指定的自变量值
double plyv()	返回多项式值 $p(x)$

【函数程序】 (文件名: lplyv.c)

```
double plyv(a, n, x)
int n;
double x, a[];
{ int i;
  double u;
  u=a[n-1];
  for (i=n-2; i>=0; i--)
    u=u * x+a[i];
  return (u);
}
```

【例】 计算多项式

$$p(x) = 2x^6 - 5x^5 + 3x^4 + x^3 - 7x^2 + 7x - 20$$

在 $x = \pm 0.9, \pm 1.1, \pm 1.3$ 处的函数值。

主函数程序(文件名: lplyv0.c)如下:

```
#include "stdio.h"
#include "lplyv.c"
main()
{ int i;
  double a[7]={-20.0,7.0,-7.0,1.0,3.0,-5.0,2.0};
  double x[6]={0.9,-0.9,1.1,-1.1,1.3,-1.3};
  printf("\n");
  for (i=0; i<=5; i++)
    printf("x(%d)=%5.2lf  p(%d)=%13.7e\n",
           i,x[i],i,plyv(a,7,x[i]));
  printf("\n");
}
```

运行结果为

```
x(0)= 0.90  p(0)=-1.8562268e+001
x(1)=-0.90  p(1)=-2.6715368e+001
x(2)= 1.10  p(2)=-1.9556128e+001
x(3)=-1.10  p(3)=-2.1513028e+001
x(4)= 1.30  p(4)=-2.0875732e+001
x(5)=-1.30  p(5)=-6.3404320e+000
```

1.2 一维多项式多组求值

【功能】 利用系数预处理法对多项式

$$p(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1x + a_0$$

同时对多个 x 值进行求值,其中 $n=2^k (k \geq 1)$ 。

【方法说明】 首先将多项式变为首一多项式,即

$$T(x) = p(x)/a_{n-1} = x^{n-1} + t_{n-2}x^{n-2} + \cdots + t_1x + t_0$$

其中 $t_k = a_k/a_{n-1} (k = n-2, \cdots, 1, 0)$ 。

然后对首一多项式 $T(x)$ 的系数进行预处理。预处理过程如下:

将 $T(x)$ 分解为

$$T(x) = (x^j + b)q(x) + r(x)$$

其中 $j=2^{k-1}$, $q(x)$ 与 $r(x)$ 均为 $2^{k-1}-1$ 次的首一多项式。 b , $q(x)$ 与 $r(x)$ 按以下规则确定:

多项式中的中项系数减 1 即为 b , 即

$$b = t_{2^{k-1}-1} - 1$$

多项式左半部分除以 x^j 即为 $q(x)$, 即

$$q(x) = x^{s-1} + q_{s-2}x^{s-2} + \cdots + q_1x + q_0$$

其中 $s=2^{k-1}$, $q_i = t_{i+s} (i = s-2, \cdots, 1, 0)$ 。

多项式右半部分减去 b 与 $q(x)$ 的乘积即为 $r(x)$, 即

$$r(x) = x^{s-1} + r_{s-2}x^{s-2} + \cdots + r_1x + r_0$$

其中 $r_i = t_i - bq_i (i = s-2, \dots, 1, 0)$ 。

由于 $q(x)$ 与 $r(x)$ 还是首一多项式, 因此, 它们也可以按照上述方法分别进行分解。这个过程一直做到 $q(x)$ 与 $r(x)$ 的次数为 1 为止。在分解过程中, 每次分解后的系数仍存放在 $T(x)$ 各系数的存储单元中, 最后得到 $T(x)$ 经分解处理后的系数。

首一多项式 $T(x)$ 的系数经预处理后, 就可以用这些预处理后的系数对不同的 x 求函数值。这种方法特别适用于对多个 x 进行求值, 减少求值中的乘法次数。

如果原来的多项式不满足 $n = 2^k$ 这个条件, 则可以添加系数为 0 的各项及系数为 1 的最高次项。

【函数语句与形参说明】

```
void plys(a, n, x, m, p)
```

形参与函数类型	参数意义
Double a[n]	存放 $n-1$ 次多项式的 n 个系数, 返回时其值将被改变
int n	多项式的项数
double x[m]	存放给定的 m 个自变量值
int m	给定自变量的个数
double p[m]	返回时存放与给定 m 个自变量值对应的多项式值
void plys()	过程

【函数程序】 (文件名: lplys.c)

```
#include "math.h"
#include "stdlib.h"
void plys(a, n, x, m, p)
int n, m;
double a[], x[], p[];
{ int i, j, mm, nn, ll, t, s, kk, k;
  double *b, y, z;
  b = malloc(2 * n * sizeof(double));
  y = a[n-1];
  for (i=0; i<=n-1; i++) b[i] = a[i]/y;
  k = (int) (log(n-0.5)/log(2.0))+1;
  nn = 1;
  for (i=0; i<=k-1; i++) nn = 2 * nn;
  for (i=n; i<nn-1; i++) b[i] = 0.0;
  b[nn-1] = 1.0;
  t = nn; s = 1;
  for (i=1; i<=k-1; i++)
  { t = t/2; mm = -t;
    for (j=1; j<=s; j++)
    { mm = mm+t+t; b[mm-1] = b[mm-1]-1.0;
      for (kk=2; kk<=t; kk++)
```

```

        b[mm-kk]=b[mm-kk]-b[mm-1]*b[mm+t-kk];
    }
    s=s+s;
}
for (kk=1; kk<=m; kk++)
{ for (i=0; i<=(nn-2)/2; i++)
    a[i]=x[kk-1]+b[2*i];
    mm=1; z=x[kk-1];
    for (i=1; i<=k-1; i++)
        { mm=mm+mm; ll=mm+mm; z=z*z;
          for (j=0; j<=nn-1; j=j+ll)
              a[j/2]=a[j/2]+a[(j+mm)/2]*(z+b[j+mm-1]);
        }
    z=z*z/x[kk-1];
    if (nn!=n) a[0]=a[0]-z;
    p[kk-1]=a[0]*y;
}
return;
}

```

【例】 利用系数预处理法计算多项式

$$p(x) = 2x^6 - 5x^5 + 3x^4 + x^3 - 7x^2 + 7x - 20$$

在 $x = \pm 0.9, \pm 1.1, \pm 1.3$ 处的函数值。

主函数程序(文件名: lplys0.c)如下:

```

#include "stdio.h"
#include "lplys.c"
main()
{ int i;
  double p[6];
  double a[7]={-20.0,7.0,-7.0,1.0,3.0,-5.0,2.0};
  double x[6]={0.9,-0.9,1.1,-1.1,1.3,-1.3};
  plys(a,7,x,6,p);
  printf("\n");
  for (i=0; i<=5; i++)
      printf("x(%d)=%5.2lf  p(%d)=%13.7e\n",
            i,x[i],i,p[i]);
  printf("\n");
}

```

运行结果为

```

x(0)= 0.90    p(0)=-1.8562268e+001
x(1)=-0.90   p(1)=-2.6715368e+001
x(2)= 1.10   p(2)=-1.9556128e+001
x(3)=-1.10   p(3)=-2.1513028e+001
x(4)= 1.30   p(4)=-2.0875732e+001

```

x(5)=-1.30 p(5)=-6.3404320e+000

1.3 二维多项式求值

【功能】 计算二维多项式

$$p(x,y) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} a_{ij} x^i y^j$$

在给定点 (x,y) 处的函数值。

【方法说明】 将二维多项式变形如下：

$$p(x,y) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} a_{ij} x^i y^j = \sum_{i=0}^{m-1} \left[\sum_{j=0}^{n-1} (a_{ij} x^i) y^j \right]$$

令

$$s_i = \sum_{j=0}^{n-1} (a_{ij} x^i) y^j, \quad i = 0, 1, \dots, m-1$$

则计算 s_i 的递推公式如下：

$$u = a_{i,n-1} x^i$$

$$u = u \times y + a_{ij} x^i, \quad j = n-2, \dots, 1, 0$$

其中最后的 u 即为 s_i 。

最后将所有的 $s_i (i=0, 1, \dots, m-1)$ 累加, 即

$$p(x,y) = \sum_{i=0}^{m-1} s_i$$

【函数语句与形参说明】

double bply(a, m, n, x, y)

形参与函数类型	参数意义
double a[m][n]	存放二维多项式的系数
int m	自变量 x 的最高次数为 $m-1$
int n	自变量 y 的最高次数为 $n-1$
double x, y	给定的一对自变量值
double bply()	返回函数值 $p(x,y)$

【函数程序】 (文件名: 1bply.c)

```
double bply(a, m, n, x, y)
int m, n;
double a[], x, y;
{ int i, j;
  double u, s, xx;
  u=0.0; xx=1.0;
  for (i=0; i<=m-1; i++)
    { s=a[i * n+n-1] * xx;
```

```

    for (j=n-2; j>=0; j--)
        s=s*y+a[i*n+j]*xx;
    u=u+s; xx=xx*x;
}
return (u);
}

```

【例】 计算二维多项式

$$p(x, y) = \sum_{i=0}^3 \sum_{j=0}^4 a_{ij} x^i y^j$$

在给定点(0.6, -1.3)处的函数值,其中系数矩阵为

$$A = \begin{bmatrix} 1.0 & 2.0 & 3.0 & 4.0 & 5.0 \\ 6.0 & 7.0 & 8.0 & 9.0 & 10.0 \\ 11.0 & 12.0 & 13.0 & 14.0 & 15.0 \\ 16.0 & 17.0 & 18.0 & 19.0 & 20.0 \end{bmatrix}$$

主函数程序(文件名: lbply0.c)如下:

```

#include "stdio.h"
#include "lbply.c"
main()
{ double z;
  double a[4][5]={{1.0,2.0,3.0,4.0,5.0},
                  {6.0,7.0,8.0,9.0,10.0},
                  {11.0,12.0,13.0,14.0,15.0},
                  {16.0,17.0,18.0,19.0,20.0}};

  printf("\n");
  z=bply(a,4,5,0.6,-1.3);
  printf(" p(0.60,-1.30)=%13.7e\n",z);
  printf("\n");
}

```

运行结果为

p(0.60,-1.30)=3.9665544e+001

1.4 复系数多项式求值

【功能】 计算复系数多项式

$$p(z) = a_{n-1}z^{n-1} + a_{n-2}z^{n-2} + \cdots + a_1z + a_0$$

在给复数 z 时的函数值。

【方法说明】 同 1.1 节,只是改成复数运算。有关复数乘法的方法说明见 2.1 节。

【函数语句与形参说明】

```
void cply(ar,ai,n,x,y,u,v)
```


形参与函数类型	参数意义
double ar[n],ai[n]	分别存放多项式系数的实部与虚部
int n	多项式的项数,其最高次数为 $n-1$
double x,y	给定复数 z 的实部与虚部
double *u,*v	指向返回多项式值 $p(z)$ 的实部与虚部
void cply()	过程

【函数程序】 (文件名: 1cply.c)

```

void cply(ar,ai,n,x,y,u,v)
int n;
double x,y,ar[],ai[],*u,*v;
{ int i;
  double p,q,s,t;
  void cmul(double,double,double,double,double *,double *);
  s=ar[n-1]; t=ai[n-1];
  for (i=n-2; i>=0; i--)
    { cmul(s,t,x,y,&p,&q);
      s=p+ar[i]; t=q+ai[i];
    }
  *u=s; *v=t;
  return;
}

static void cmul(a,b,c,d,e,f)
double a,b,c,d,*e,*f;
{ double p,q,s;
  p=a*c; q=b*d; s=(a+b)*(c+d);
  *e=p-q; *f=s-p-q;
  return;
}

```

【例】 计算复系数多项式

$$p(z) = (2+j2)z^3 + (1+j)z^2 + (2+j)z + (2+j)$$

当 $z=1+j$ 时的函数值。

主函数程序(文件名: 1cply0.c)如下:

```

#include "stdio.h"
#include "1cply.c"
main()
{ double x,y,u,v;
  double ar[4]={2.0,2.0,1.0,2.0};
  double ai[4]={1.0,1.0,1.0,2.0};
  printf("\n");
}

```