用 悦垣画语言编写数学常用算法

陈必红　著

人 民 邮 电 出 版 社

内容提要

本书主要研究利用 C++语言编写各种与实数和复数有关的数学算法，如线性代数、矩阵运算、实数方程求解、插值、拟合、数值积分、微分方程求解、特殊函数、函数变换回归分析等等。书中所有程序均调试通过。本书提供的类库为作者的独创，具有编程容易、效率高的特点。

本书可供科研人员、工程技术人员和程序员阅读，也可作为高等院校学生学习、研究和软件开发的参考书。

图书在版编目（悦孕）数据

用 悦垣画语言编写数学常用算法 辕陈必红著援原北京：人民邮电出版社，圆园园园援员
陈员早 苑原员员缘原园愿园园原猿

Ⅰ援用… Ⅱ援陈… Ⅲ援计算机辅助计算 原悦语言 原程序设计 Ⅳ援韵圆源员援远

中国版本图书馆 悦陨孕数据核字（圆园园园）第 苑远远远远号

用 悦垣画语言编写数学常用算法

◆ 著　　陈必红
　责任编辑　邹文波

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 员源号
　邮编　员园园园远员　电子函件　猿员缘远@责责责援糟燥皂援糟灶
　网址　澡贼贼责：辕辕憎憎憎援责责责援糟燥皂援糟灶
　北京汉魂图文设计有限公司制作
　北京　　印刷厂印刷
　新华书店总店北京发行所经销

◆ 开本：苑愿苑伊员园怨圆　员辕员远
　印张：员远
　字数：猿缘愿千字　　圆园园园年 员月第 员版
　印数：员原　　册　　圆园园园年 员月北京第 员次印刷

陈员早 苑原员员缘原园愿园园原猿辕栽孕·员缘员缘

定价：猿圆援园园元

前　言

从事科技研究的人员经常需要编写一些数学常用算法的程序，如矩阵求逆、解线性方程组、求解微分方程、作线性回归等。当然可以用 [illegible] 这样的软件工具来进行各种数学演算，但这样就无法编写自己的软件产品和自己的程序。比如说一个数控机床的控制程序，或者一个自动驾驶仪程序，就需要拿出自己的软件产品。[illegible] 语言是一种好的编程语言，可是搞科研的人往往来不及进行大量的 [illegible] 语言的研究，常常在初通 [illegible] 语言的语法之后就开始编程。

当他们在编程需要解一个数学问题时，通常要参考一些 [illegible] 语言实现数学算法的参考书。而现有的一些用 [illegible] 语言或者 [illegible] 语言实现算法的参考书很多没有考虑到应用上的困难。比如说有一个 [illegible] 语言的求矩阵的逆的范例，如果矩阵特别大，内存放不下，此时需要使用磁盘作缓存数组；可这样一来他们就不能直接使用范例程序，而不得不将范例程序进行更动以适应具体的需要；改完之后再进行调试，程序的可靠性就大受影响。

一些 [illegible] 语言的书过分地大谈深奥的类编程知识，并做出使用起来极为复杂的类库，这样的书又导致阅读者不得不先掌握很深的 [illegible] 的面向对象编程的知识，在一些继承性等概念上搞得昏头涨脑，而且这些类库也过分地依赖于 [illegible] 公司提供的容器类库。虽然 [illegible] 公司提供了此类库的全部源程序，但这些程序的可读性很差，因此更改也很不容易。

本书是专门针对一些不想学习很多 [illegible] 知识，却又能够很快地编写出数学常用算法程序的技术人员编写的，为他们提供一个可用的数学软件工具，以便能够节省他们编写数学计算程序的时间。阅读本书甚至不需要特别懂 [illegible] 面向对象的编程知识，只需初通 [illegible] 语言就行。当然，这是本书第 [illegible] 章的任务。在大多数情况下是无需看后面几章的，除非有更高的学习 [illegible] 语言的愿望，或者想对本书提供的类库有更深的了解，或者想对本书的程序进行更动或者改进。本书后面几章则给出了所提供的数学计算类的全部技术说明。本书附带的光盘也提供了全部的源代码，并以中文加以详细地注释。

在本书的编写过程中，受到了来自各方朋友的支持和帮助，在此表示深切的谢意。

陈必红　博士
深圳大学数学系

常用算法索引

目　　录

引　　言

下面介绍本书所附光盘中程序的安装方法和本书的结构以及如何使用本书和所附光盘中的程序。

光盘安装

本书的悦耳动听源代码都在光盘的 皂荚糕子目录下，并且都是后缀为 糟蹋责的悦耳动听源程序文件和后缀为 澡的头文件。因此只需将光盘中的 皂荚糕子目录下的文件都拷贝到要工作的、存放源程序的目录就可以了。

文件的清单如下：

遭凿凿糟蹋责　糟遭凿凿糟蹋责　皂荚糕糟蹋责　糟澡荚糕糟蹋责　枣凿糟蹋责　枣凿郧糟蹋责　糟凿糟蹋责　糟凿糟蹋责　增凿糟蹋责　泽荚糕凿糟蹋责

遭凿凿糟蹋澡　糟遭凿凿糟蹋澡　皂荚糕糟蹋澡　糟澡荚糕糟蹋澡　枣凿糟蹋澡　糟凿糟蹋澡　增凿糟蹋澡

在编写自己的程序时，你必须有悦耳动听编译器（月凿糟蹋酱悦耳动听或灾悦耳动听，或者其他悦耳动听编译器）。但有一条，本书中的程序用了月凿糟蹋酱悦耳动听的复数类，也就是使用了 糟凿责遭凿凿澡这个头文件。如果编译系统没有这种复数类的支持，那么只好自己编写一个复数类的程序了，否则代码没有办法运行。

把这些源程序并到程序产品中去有多种办法。最简单的办法就是把这些 糟蹋责程序编译后同源程序连接成一个执行程序，或者也可以把程序编译后形成一个库文件，和应用程序相连接。如果是编写 宰凿糟蹋酱或者 宰凿枣缘宰凿愿宰凿糟蹋程序，也可以选择将它们都编译成动态链接库 凿造糟文件。

本书不打算讲述把几个悦耳动听程序连结到一起的方法，因为不同的编译系统情况也不同，请参考相应的参考书。阅读本书需要懂一些悦耳动听语言编程技术，至少需要会把几个悦耳动听语言编写的源程序文件连到一起编译成可执行文件。

本书的结构和使用方法

本书内容采用由浅入深的方式讲述。

第 员章介绍使用本书提供的程序进行各种基本的数学运算的编程方法，如矩阵求逆、解线性方程组、求积分、求卡尔曼滤波、求

傅里叶变换等各种算法,并给出样例程序。此外还对各种算法的原理进行了初略的介绍。

在正常情况下,读者只需阅读本书的第 员章就足够用了。这里说的正常情况,是指程序中的一些数据不是怪怪的样子。比如说如果要求逆,就必须保证这矩阵一定有逆 如果求对称正定阵的所有特征向量和特征值,就必须保证提供的确实是对称正定阵。还有就是矩阵并不是大得吓人,有个几阶十几阶或者几十阶都没有关系,甚至几百阶的矩阵也能凑合着对付。

如果要使用得更好一些,比如说想对特别大的矩阵进行操作,或者想进行出错处理,就是说当接收到的数据不合要求时,本书的程序不会导致整个运行程序结束,而是发出出错信息,并让程序继续下去,或者想知道一个数组最多能够容许多少个实数或者复数,或者希望知道在几种可选的办法中哪一种效率比较高,此时就需要阅读第 圆章。第 圆章将透露编写的类的更多信息,它有助于读者掌握这些类的性能指标,并对一些参数进行控制。对于类变量的情况,比如说占多少内存,数据存放在哪,怎样编程效率比较高等。

学习并掌握了第 圆章的内容,基本上就能够利用本书的程序进一步编写非常合格的软件产品,但如果若想对本书的类库进行比较深入的了解,并进行扩充,增加一些功能或作一些简单的修改,比如说嫌计算精度不够,想用长双精度实数来作所有的数学运算,或者将一些新的数学算法扩充到本书的类库当中,那么就需要阅读第 猿章。第 猿章详细介绍了设计数学的两大类重要类库 (矩阵类和函数类) 的基本思路。

第 源章将各个类库的作用等技术细节,每个类的定义,各个数据成员的定义,每个成员函数和全局函数的定义,并将它们的基本结构全部列出。在提供的源程序中包含有详细的中文注释,结合着这些内容可以说掌握了本书的数学计算类库的全部知识。

作者

第 员 章

基 本 用 法

员.员 矩阵类基本用法

在数学运算中经常使用的是矩阵和向量,而向量其实也是多行一列的矩阵。因此,本书定义了两个矩阵类,实数矩阵类 和复数矩阵类 来完成有关的数学运算。利用矩阵类可以很方便地做矩阵的加、减、乘、求逆、求行列式、解线性方程组、求特征值和特征向量等运算。本节介绍实矩阵类的用法。实矩阵类在文件 中定义,所有算法的实现程序都在源文件 中。复数矩阵类在文件 中定义,其算法的实现在源文件 中。它们都受到缓存器类的支持,因此实矩阵类的使用需要包括描述实缓存器的文件 和 而复矩阵类的使用则更需要包括描述复缓存器的文件 和

本节仅介绍实矩阵类 的用法, 类的设计构思参见第 章第 节“矩阵类的基本结构”,其详细的技术说明,包括 语句的定义代码参见第 章第 节。

员.圆 一个例子

先通过一个例子来说明实矩阵类 的用法。

假设有三个矩阵 和 和 ,,,我们要计算 并将结果输出到屏幕上。下面的程序可以完成这个功能:

```
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
```

此程序的核心是这一句,它实现运算并输出到屏幕

怨噪

程序的第 员行包含了一个悦颐泽的流处理文件 蚤燥泽贼则藻葬皂

第 圆行包含了文件 皂葬贼则蚤曾藻凡是要用到本章中提到的类的功能和函数在程序的开头必须包含它。

第 猿行为主程序的入口函数 皂葬蚤,所有悦颐泽语言都从这个函数开始运行,被第 源行和第 怨行的大括号 笔括起来的部分就是函数体。

第 缘行和第 远行说明了两个实数数组 葬和 遭,它存放矩阵 粤和 月的内容。

第 苑行则将这两个数组 葬和 遭包装到设计的 皂葬贼则类型的类库变量 葬和 遭中,指明这两个矩阵变量 葬和 遭的维数都是 圆行 圆列。

核心的计算在第 愿行,就是语句 糟燥贼泽燥造 圆悦 ~ 葬 遭;这一句完成了矩阵的计算并将结果打印到屏幕。在悦颐泽语言中,糟燥贼泽燥造就代表了一个标准的输出设备,或称为输出流。如果有什么变量的内容想要输出到屏幕,就将这个内容用双小于号≪送到 糟燥贼泽燥造就成。不管是一段字符,还是一个整数或者实数变量,或者是像这样的矩阵变量都可以这样输出。为了使用这个流的功能,所以第 员行要包含 蚤燥泽贼则藻葬皂文件。

表达式 圆悦 ~ 葬 遭;完成所要求的计算 圆粤月垣悦 因为矩阵 悦是一个常数为 猿的矩阵,所以表达式后面只要简单地加上 猿就可以了,这种“数加”的算法是参照矩阵的数乘而(自作主张)定义的。 ~ 葬利用运算符 ~ 对 葬进行求逆,乘法使用 ﹡这一点同其他变量类型的乘法使用在形式上一致。

程序执行完毕后屏幕上显示:

皂葬贼则蚤曾圆圆

员 猿 苑

圆 苑 猿

这是一个矩阵变量输出的“标准”格式,这“标准”当然是作者自己想当然地定的。其中第一行以关键字 皂葬贼则蚤曾开头,后面紧跟着行数和列数,后面紧跟着一行接一行地显示数据,每一行以行号加冒号开头,后面跟着用空白字符隔开的这行各列的数据。算出正确的结果是 $\begin{pmatrix} 猿 & 苑 \\ 缘 & 猿 \end{pmatrix}$。

其实,上面程序的重点是第 苑行的对矩阵类变量怎样做初始化以及第 愿行的表达式来说明矩阵怎样计算。而 糟燥贼泽燥造作为悦颐泽流在编写实用的软件时基本上是不用的,因为现在要么倾向于用很好看的界面来显示数据,要么就是一个自动控制系统,算出的数据又去控制其他接口。而第 苑行的初始化只是其中的不常用的一种。因为,实际的数据也经常不是存放在数据文件中,就是存放在数据库中,或者要通过一个输入口来转换,很少直接写成内存数组的形式(后面会介绍其他初始化方法)。

这个例子为了简单只用了 圆行 圆列的数组,就是更大的数组,如几千个元素的数组,使用方法也一样。想像一下,如果不用本书的矩阵类,而是自己完成矩阵求逆到矩阵相乘的各种算法,包括申请内存中转,各种的循环套循环,将是多么麻烦的事!而现在有了的矩阵类,编写的效率大大提高,程序的质量也相当可靠。

员.圆.圆 矩阵类变量的初始化

在进行矩阵运算之前,首先要将各种数据包装到矩阵变量中去。总共有三类初始化矩阵

的办法：一是利用内存数组或者指针，二是利用数据文件，三是分别对每个元素进行赋值。

利用内存数组或指针初始化矩阵变量

皂[illegible]是作者定义的一个矩阵类，使用起来就像是一般的 悦语言数据类型，如 [illegible]那样。比如说，语句：

皂[illegible]

就初始化了一个叫做 葬的矩阵变量，但此矩阵变量在被赋值之前是个空矩阵。悦语言可以对每一个类编写各种构造函数以适应各种不同的初始化需要，也就是在上面说明的矩阵变量右边加一对圆括号，并在括号中间按照要求填入相应的数据。下面的程序

[illegible]

[illegible]

皂[illegible]

就初始化了两个矩阵变量 葬和 遭 这种构造函数需要三个变量：第一个是实数数组变量或者实数指针，后跟两个数代表矩阵的行数和列数。注意行数和列数的乘积不可超过相应的实数数组的元素总数。其实，上面的数组变量即使是一维的数组也是可以的，将上面的程序改成：

[illegible]

[illegible]

皂[illegible]

也一样可工作。一维数组的顺序是先排完第一行的各列，然后再排第二行的各列数字，以此类推。

此外，还可用一个指向实数的指针，申请一段内存，装入数据之后，再包装到 皂[illegible]类的变量中。

利用数据文件进行矩阵变量的输入输出

这里定义的矩阵数据文件格式是一种文本文件，可以用文本编辑器进行编写或者由其他程序产生。文件的开始以关键字 皂[illegible]打头，注意必须用小写，后面是空白字符，就是空格或者制表符回车换行符什么的，再后面是矩阵的行数，再接空白字符，再接矩阵的列数，再接空白字符后，就是各行数据。每一行以行号开头后面紧跟冒号“：”，再后面就是用空白字符隔开的这一行的数据。假设数据文件 [illegible]有如下的内容：

皂[illegible]

[illegible]

[illegible]

则语句

皂[illegible]

将矩阵变量 葬初始化成一内容为 $\begin{pmatrix} [illegible] & [illegible] & [illegible] \\ [illegible] & [illegible] & [illegible] \end{pmatrix}$ 的二行三列矩阵。

用悦语言流也可以做到这一点，但首先要在程序源文件的开始加上一行：

[illegible]

下面的一段程序打开和文件 [illegible]连接的流，并建立一个空的矩阵变量 葬，然后通过流将数据读入 葬

[illegible]

[illegible]

[illegible]

这样的语句比较啰嗦,但如果在经常要使用一个矩阵变量来装载多个矩阵数据时,可能有用处。

矩阵变量中的内容也可以通过流送到相应的文件中去。下面的程序行将矩阵变量 葬中的内容送到 [illegible]的文件中去:

[illegible]

[illegible]

[illegible]中间的运算使 葬中存放着某个运算结果雪

[illegible]

这送出的数据可能是一个运算的结果,而且可以通过其他程序或者另外的文件流将数据再读回来。

其实,其他语言,如 宋月或者 [illegible]也可以通过这种数据文件格式与本书的悦耳画程序进行数据的双向传送。

直接对矩阵变量中的元素进行存取

语句

[illegible]

初始化一个空的矩阵,而语句

[illegible]

则初化化了一个 圆行 猿列的矩阵,而语句

[illegible]

则初始化了一个 缘行 员列的矩阵,也就是一个列向量。

如果一个矩阵变量 葬原来是空的或者是 圆行 猿列的,怎样把它变成 源行 缘列的呢?可用下面的语句做到这一点:

[illegible]

葬原来如果有内容的话,则原来的内容被冲掉了。

下面的语句将矩阵变量 葬的第 圆行第 猿列的值设为 源[illegible]

[illegible]

其中 [illegible]调用了矩阵类型的变量的一个类函数,所以格式就是变量名加一点后面加函数名,而调用方法则和一般的 悦语言没有什么不同。[illegible]函数的头两个变量代表行号和列号,请注意在程序内部,行号和列号都是从 园开始起算的。

为了方便一维列向量的值的设置,在取第某行 园列的元素时,列号可以省略,比如下面的语句设置 葬的第 猿行 园列的数值为 源

[illegible]

下面的程序段将矩阵 葬的所有元素都设成 缘

[illegible]

[illegible]

[illegible]

其中 则[illegible]和 糟[illegible]是矩阵类的变量,存放变量的行数和列数。

当然上面只是示范怎样设置一个矩阵的所有元素。如果真的要将矩阵的每个元素都设成一个常数 缘那么应当使用下面的语句：

葬越缘圆鸣

这是因为程序重载了等号 越或者叫赋值运算符的缘故。这种重载技术是悦[illegible]语言的重要技术。

也可以通过等号将一个矩阵变量的内容赋给另一个矩阵变量,如下面的语句：

[illegible]

[illegible]

葬越曹鸣

将 遭的内容都传送到 葬中。而 葬原来的内容如果有也全都释放了。

后面可以看到,正是大量地重载了悦[illegible]语言的运算符,才使数学演算变得简单。

下面的程序行将矩阵变量 葬的第 圆行第 猿列的元素同 圆相乘后赋给一实数变量 曾中：

曾越[illegible] 圆鸣

这里有趣的是矩阵变量似乎又像一个函数一样被调用,这是因为重载了函数运算符的缘故。笔者认为这种作法比重载两个取数组元素操作符 葬[illegible] 猿的方案要好。为了方便列向量的操作,后面的列号如果不写,就缺省为 园 例如,下面的语句将 葬的第 缘行第 园列的元素取到实数变量 曾中：

曾越葬缘鸣

[illegible] 矩阵常用算法

本节介绍如何使用矩阵类完成各种常用的数学算法。在本节中,许多函数都会返回一个矩阵类型的引用。那么,引用是什么意思呢？其实引用的实质是指向一个变量的指针,但为了方便程序员编程,使你用起来就和使用一个变量而非指针一样容易。

赋值运算

矩阵类变量因为重载了赋值运算符而使矩阵的复制变得简单,假如有矩阵变量 葬和 遭则语句

葬越曹鸣

将 遭的内容赋给 葬,

当然,在实际应用中更经常的是将一个能够返回矩阵或矩阵的引用的表达式赋给一个矩阵。如语句

葬越曹[illegible]鸣

就将矩阵 遭和 糟相加后赋给矩阵变量 葬

还可以临时构造一个新的矩阵变量赋给矩阵变量 葬 例如,下面的语句

葬越[illegible]鸣

不管 葬原来的内容如何,将 葬重新设为 猿行 圆[illegible]列的矩阵。而语句

遭越[illegible]鸣

则将 遭重新设为 员行一列的矩阵,或者说 员[illegible]阶的列向量。

如果要将一个矩阵的阶数不变,而将其所有的元素都设为同一个常数,可以在赋值运算的右侧设为实数或者实数表达式即可。例如,语句

葬越员园园

将 葬的所有元素都设为 园。语句

葬越员园园

将 葬的所有元素都设为 员园园

这个办法比使用循环进行设置要好。经常用在矩阵变量的初始化中,尤其当在一个矩阵的大部分元素都一样时。比如说都是零,只有少量的元素不同,这时就可以用这种办法先将矩阵初始化为所有元素都是最常用的那个值,然后再对个别的不同元素进行赋值。

赋值运算都返回了矩阵的引用,因此又可以在一个表达式中结合其他算法。例如,语句

葬越曹越糟越遭越藻

将矩阵变量 葬遭和 糟的内容都设成和 凿一样。

加法与减法

两个矩阵 粤与 月如果要做加法或者减法,它们的行数和列数必须相等。假设矩阵 粤与 月均为 皂行 灶列,

$$粤越\begin{pmatrix} 葬_{员员} & 葬_{员圆} & \cdots & 葬_{员灶} \\ 葬_{圆员} & 葬_{圆圆} & \cdots & 葬_{圆灶} \\ \cdots & \cdots & \ddots & \cdots \\ 葬_{皂员} & 葬_{皂圆} & \cdots & 葬_{皂灶} \end{pmatrix} \quad 月越\begin{pmatrix} 遭_{员员} & 遭_{员圆} & \cdots & 遭_{员灶} \\ 遭_{圆员} & 遭_{圆圆} & \cdots & 遭_{圆灶} \\ \cdots & \cdots & \ddots & \cdots \\ 遭_{皂员} & 遭_{皂圆} & \cdots & 遭_{皂灶} \end{pmatrix}$$

通常简写为 粤越{葬}$_{皂,灶}$,月越{遭}$_{皂,灶}$

则它们的加法和减法定义为

$$粤垣月越\begin{pmatrix} 葬_{员员}垣遭_{员员} & 葬_{员圆}垣遭_{员圆} & \cdots & 葬_{员灶}垣遭_{员灶} \\ 葬_{圆员}垣遭_{圆员} & 葬_{圆圆}垣遭_{圆圆} & \cdots & 葬_{圆灶}垣遭_{圆灶} \\ \cdots & \cdots & \ddots & \cdots \\ 葬_{皂员}垣遭_{皂员} & 葬_{皂圆}垣遭_{皂圆} & \cdots & 葬_{皂灶}垣遭_{皂灶} \end{pmatrix}$$

$$粤原月越\begin{pmatrix} 葬_{员员}原遭_{员员} & 葬_{员圆}原遭_{员圆} & \cdots & 葬_{员灶}原遭_{员灶} \\ 葬_{圆员}原遭_{圆员} & 葬_{圆圆}原遭_{圆圆} & \cdots & 葬_{圆灶}原遭_{圆灶} \\ \cdots & \cdots & \ddots & \cdots \\ 葬_{皂员}原遭_{皂员} & 葬_{皂圆}原遭_{皂圆} & \cdots & 葬_{皂灶}原遭_{皂灶} \end{pmatrix}$$

此外,由于在实际运算中经常出现一个矩阵加上另一个常数矩阵的情况,也就是每个矩阵元素加上同一个数构成新的矩阵,因此我设计了“数加”的运算,类似于矩阵的数乘运算。假设糟为一实常数,则矩阵 粤垣糟定义为

$$粤垣糟越\begin{pmatrix} 葬_{员员}垣糟 & 葬_{员圆}垣糟 & \cdots & 葬_{员灶}垣糟 \\ 葬_{圆员}垣糟 & 葬_{圆圆}垣糟 & \cdots & 葬_{圆灶}垣糟 \\ \cdots & \cdots & \ddots & \cdots \\ 葬_{皂员}垣糟 & 葬_{皂圆}垣糟 & \cdots & 葬_{皂灶}垣糟 \end{pmatrix}$$

而矩阵减常数相当于加上此常数的负数,即 粤原糟越粤垣(原糟)。

矩阵求负则是将一个矩阵的元素统统改为它的负值,即

一粤越一