# 基于 C#的主成分分析算法及其与 GIS 接口*

饶 元[1], 元昌安[2], 邓 松[1], 杨乐婵[1], 廖剑平[1]
(1. 广西师范学院 资源环境学院, 广西 南宁 530001; 2. 广西师范学院 信息技术系, 广西 南宁 530001)

**摘 要**: 分析了传统方法开发的组件存在"DLL Hell"问题和传统 GIS 软件在面对大数据集时效率低的不足, 指出使用 C#语言开发的组件可以有效地解决这些问题。并以开发主成分分析模块为例, 介绍了主成分分析主要计算方法, 给出了基于 C#的算法, 最后给出了基于 C#的主成分分析模块实现及其与 GIS 软件的接口技术。

**关键词**: C#; GIS; 主成分分析; 算法; 接口
**中图法分类号**: TP301.6 **文献标识码**: A **文章编号**: 1001-3695(2006)07-0038-03

# Principal Component Analysis Algorithm Based on C# and Interface Technology with GIS

RAO Yuan[1], YUAN Chang-an[2], DENG Song[1], YANG Le-chan[1], LIAO Jian-ping[1]
*(1. College of Resources & Environment Science, Guangxi Teachers Education University, Nanning Guangxi 530001, China; 2. Dept. of Information & Technology, Guangxi Teachers Education University, Nanning Guangxi 530001, China)*

**Abstract:** Analysizes the "DLL Hell" problem that traditional component development method results in and the low efficiency problem that traditional GIS software deals with a great deal data and introduces an efficient method solving these problems by means of C#. Gives an example based on principal component analysis by introducing the computation procedure, the algorithm based on C#, the module based on C# and the interface technology between the module and GIS software.

**Key words:** C#; GIS; Principal Component Analysis; Algorithm; Interface

当前热门的开发平台之一——.NET 平台是以.NET Framework 为基础架构。.NET Framework 包含了通用语言运行时(Common Language Runtime, CLR)、通用类型系统(Common Type System, CTS)和多种类库以及程序编译时使用的独立于具体机器、操作系统和 CPU 的微软中间语言(Microsoft Intermediate Language, MSIL), 使得.NET 平台下开发的应用程序具有移植性好、通用性强等众多优点[1]。

C#是 Microsoft 专门为使用.NET 平台而创建的开发语言。作为 Visual Studio.NET 的重要成员之一, C#本身就是一种完全面向对象的语言, 它继承了 C/C++的优点, 并解决了它们存在的一些问题, 如处理大数据集效率低、使用不方便等。与 C++类似, 使用 C#不仅能开发应用程序, 也可以根据需要开发用户组件(类库、DLL 库等), 并能方便地与 COM(COM+)组件、相同或不同开发工具开发的应用程序实现接口[2]。C#还具有语法结构简洁、有完整的安全性与异常处理机制、无用存储单元收集等优点。C#语言的诸多优点使其无愧成为当前程序员的新宠儿。

借助于.NET 提供的强大的.NET 装配件功能, 用 C#开发的组件中包含了组件完全自我描述的元信息, 不再需要其他信息就可以正常使用组件, 省去了以往需要修改或访问注册表的麻烦, 避免共享 DLL 的不同版本时安装由另一程序使用的 DLL 的不同版本而造成原来程序不能正常运行, 有效地解决了"DLL Hell", 使得 C#开发的组件升级和维护更加方便[1]。

主成分分析是将多个指标转换为少数几个互不相关的综合指标的一种多元统计分析方法, 它是地学分析领域最常用的数学工具之一, 被广泛应用于地学数据的分类、趋势分析、综合评价等研究方向[3]。主成分分析模块是 GIS 软件中的不可缺少的基础模块之一, 而传统 GIS 软件中的主成分分析模块在面对大数据集时往往效率很低。

## 1 主成分分析的基本原理和计算步骤[4]

设有 $n$ 个样本, 每个样本由 $p$ 个指标 $i_1, i_2, \cdots, i_p$ 描述, 可得原始数据矩阵为

$$Input_{n\times p} = \begin{bmatrix} i_{11} & i_{12} & \cdots & i_{1p} \\ i_{21} & i_{22} & \cdots & i_{2p} \\ \cdots & \cdots & \cdots & \cdots \\ i_{n1} & i_{n2} & \cdots & i_{np} \end{bmatrix} \tag{1}$$

针对原始数据矩阵 $Input_{n\times p}$, 实现主成分分析的基本步骤可以描述为:

(1) 矩阵 $Input_{n\times p}$ 中的原始数据进行标准化处理。

将原始数据矩阵 $Input_{n\times p}$ 进行标准化, 得到标准化后的矩阵 $Standard_{n\times p}$:

$$Standard_{n\times p} = (Standard_{ij})_{n\times p} \tag{2}$$

$$Standard_{ij} = \frac{x_{ij} - \bar{x}_j}{S_j} \tag{3}$$

其中, $\bar{x}_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}$ (4)

收稿日期: 2005-03-24; 修返日期: 2005-06-29
基金项目: 国家"973"计划资助项目(2002CB111504); 广西自然科学基金资助项目(0339039)

$$S_j = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2} \tag{5}$$

(2) 建立标准化后的 $p$ 个指标相关系数矩阵 $CorMatrix_{p\times p}$。

$$CorMatrix_{p\times p} = (c_{ij}) \tag{6}$$

$$c_{ij} = \frac{S_{ij}}{S_{ii}\quad S_{jj}} \tag{7}$$

其中, $S_{ij} = \frac{1}{n-1}\sum_{l=1}^{n}(Standard_{li} - \overline{Standard_i})(Standard_{lj} - \overline{Standard_j})$ (8)

(3) 求相关系数矩阵 $CorMatrix_{p\times p}$的特征值、特征向量, 并对特征值、特征向量按照特征值的大小进行排序。

排序后特征值、特征向量为 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p > 0$;

相应的标准化特征向量矩阵 $Vector_{p\times p}$, 正交特征向量 $V_i = (v_{1i}, v_{2i}, \cdots, v_{pi})^T$, $i = 1, 2, \cdots, p$。

(4) 计算各主成分的方差贡献率 $a_k$ 和累积方差贡献率 $a(k)$, 确定主成分个数; 再求出各样本的主成分值矩阵, 并以每个主成分的方差贡献率为权数计算各样本的综合评价值。

$$a_k = \frac{\lambda_k}{\sum_{i=1}^{p}\lambda_i} \tag{9}$$

$$a(k) = \frac{\sum_{i=1}^{k}\lambda_i}{\sum_{i=1}^{p}\lambda_i} \tag{10}$$

其中, 方差贡献率 $a_k$ 表示第 $k$ 个主成分提取原始 $p$ 个指标的信息量, 累计方差贡献率 $a(k)$ 表示前 $k$ 个主成分保留的原始信息量。

确定主成分个数有两种方法: ①使前 $k$ 个主成分的累积方差贡献率达到一定的要求, 如可取 $a(k) \geq 98\%$, 这是实际中常用的一种方法。②平均数原则, 选用特征值大于 1 的所有指标。

样本的主成分值矩阵为

$$F = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1k} \\ f_{21} & f_{22} & \cdots & f_{2k} \\ \cdots & \cdots & \cdots & \cdots \\ f_{n1} & f_{n2} & \cdots & f_{nk} \end{bmatrix} = Standard_{n\times p} \times Vector_{p\times k} \tag{11}$$

样本综合评价值的矩阵为

$$EvaluateValue_{n\times 1} = (e_{ij})_{n\times 1} \tag{12}$$

其中, $e_i = a_1 f_{i1} + a_2 f_{i2} + \cdots + a_k f_{ik}$, $i = 1, 2, \cdots, n$。

## 2　基于 C#的主成分分析算法

算法中用二维数组存放矩阵, 为方便起见, 对矩阵及相应数组统一用数组名称。

### 2.1　求原始样本矩阵的标准化矩阵、相关系数矩阵

定义类 CalStandardCorMatrix, 实现原始样本矩阵的标准化, 标准化后矩阵的相关系数矩阵的求解, 即式(1)、式(2)。

输入: float[ n, p] Input 原始样本数据矩阵。

输出: float[ n, p] Standard, float[ p, p] CorMatrix 标准化后的矩阵和其相关系数矩阵。

```
class CalStandardCorMatrix
{ float [ n, p] Input;
/* 存放原始样本数据 Input_{n×p}矩阵二维数组* /
float [ n, p] Standard;
/* 存放标准化后的矩阵 Standard_{n×p}的二维数组* /
float [ p, p] CorMatrix;
/* 存放相关系数矩阵 CorMatrix_{p×p}的二维数组* /
CalStandardCorMatrix( float [ n, p] Input) { …}
/* 构造函数接收原始样本矩阵* /
float Standlize( float [ n, p] Input, out float[ n, p] Standard)
/* 原始样本矩阵的标准化* /
{ …} /* 通过计算式(4)、式(5) 求得式(3), 即得式(2) * /
float calCorrelateMatrix( float [ n, p] Standard, out float[ p, p] CorMatrix) /* 求相关系数矩阵 CorMatrix_{p×p}* /
{
求矩阵 Standard_{n×p}列平均值并存入数组 Ave[ i] ( i = 1, 2, …, p) ;
Rsum[ i, j] + = ( Standard[ l, i] - Ave[ i] )*( Standard[ l, j] - Ave[ j] ) ;
Cov[ i, j] = Rsum[ i, j] /( n - 1) ; ( l = 1, 2, …, n) /* 求标准化
矩阵 Standard_{n×p}的协方差矩阵 Cov[ p, p] , 即式(8) * /
CorMatrix[ i, j] = Cov[ i, j] / Sqrt( Cov[ i, i] * Cov[ j, j] ) ; ( i, j = 1, 2, …, p) /* 求 Standard_{n×p}的相关系数矩阵 CorMatrix_{p×p}, 即式(6) * /
}
}
```

### 2.2　求相关系数矩阵的特征值、特征向量

定义类 CalEigenValueVector, 实现相关系数矩阵特征值、特征向量的求解。

输入: float[ p, p] CorMatrix 相关系数矩阵。

输出: float[ p] EigenValue, float [ p, p] EigenVector 相关系数矩阵的特征值和特征向量(已按特征值大小排序)。

```
class CalEigenValueVector
{
float[ p, p] Eigenvalue; /* 对角线元素存放特征值的二维数组* /
float[ p] EigenValue; /* 存放原始矩阵特征值的二维数组* /
float[ p, p] EigenVector; /* 存放特征向量矩阵 A_{p×p}的二维数组* /
float calEigenValueVector( ref float[ p, p] CorMatrix, out float[ p] EigenValue;
out float[ p, p] EigenVector)
/* 利用雅可比法计算矩阵 CorMatrix_{p×p}的特征值、特征向量* /
{
Eigenvalue[ i, j] = CorMatrix[ i, j] , ( i, j = 1, 2, …, p) ;
int l = 0; /* 定义计数变量* /
while( l < ( p* ( p - 1) ) /2) /* 判断上三角元素是否全部满足设定值, 满足跳出循环, 否则继续循环* /
{ l = 0;
求在 Eigenvalue[ p, p] 矩阵上三角元素中最大值的位置 pos1, pos2
根据 pos1, pos2 进行一轮特征值、特征向量的计算
if( abs( Eigenvalue[ i, j] ) < 10^-10) , ( i = 0, 1, …, p, j = i + 1, …, p)
/* 判断上三角元素是否满足条件* /
{ l++; } /* 满足计数器 l 加 1* /
}
EigenValue[ i] = Eigenvalue[ i, i] , ( i = 1, 2, …, p) ;
/* 将特征值放入一维数组中* /
将特征值、特征向量按照特征值的大小进行排序得到 EigenValue[ p] 和 EigenVector[ p, p]
}
}
```

### 2.3　求各样本综合评价值

定义类 CalEvaluateValue, 实现各样本综合评价值的求解, 即式(11)。

输入: float [ p] EigenValue, float[ p, p] EigenVector, float[ n, p] Standard 相关系数矩阵的特征值和特征向量(已按特征值大小排序)及原始矩阵的标准化矩阵。

输出: float[ n, 1] EvaluateValue 各样本的综合评价值。

```
class CalEvaluateValue
{
int k = 0; /* 存放提取的主成分的个数* /
float [ n, 1] Value; /* 存放最终分析结果* /
float calTotalEvaluate( ref float [ p] EigenValue, ref float[ p, p] Standard, out float [ n, 1] Value, out int k)
{
计算每个指标(共 p 个)的 a_k(即式(9)存入 GX[ p] 和 a( k) (即式10)存入 totalGX[ p]
确定主成分个数 k
F[ i, j] + = Standard[ i, l] * Eigenvector[ l, j] , ( i, j = 0, 1, …, p; l = 0, 1, …, k) ; /* 将 Standard[ n, p] 右乘 Eigenvector[ p, k] 求得各样本的主成分值并存入 F[ n, k] , 即式(11) * /
EvaluateValue[ i, 0] + = GX[ j] * F[ i, j] ; ( i = 0, 1, …, n; j = 0, 1,
```

```
…, k) /* 将 GX[ k] 右乘 F[ i, j] 求出各样本的综合评价值并存入 EvaluateValue[ n, 1] 中, 即式( 12) * /
    } }
```

### 2.4 主成分分析类

定义主成分分析类 PriComponentAnalysis 实现主成分分析的整个计算。

```
class PriComponentAnalysis
{
public int n, p; /* 定义表示原始样本的样本数和指标个数变量* /
public float [ n, p] Input; /* 存放原始样本数据矩阵的二维数组* /
CalStandardCorMatrix myStandCorMatrix;
/* 定义类 CalStandardCorMatrix 类的对象* /
CalEigenValueVector myEigenValueVector;
/* 定义 CalEigenValueVector 类的对象* /
CalEvaluateValue myEvaluateValue;
/* 定义 CalEvaluateValue 类的对象* /
public void PriComAnalysis( )
{
myStandCorMatrix = new CalStandardCorMatrix( float [ n, p] Input) ;
myStandCorMatrix. Standlize( float[ , ] Input, out float[ , ] Standard) ;
/* 调用方法将原始样本矩阵标准化* /
myStandCorMatrix. calCorrelateMatrix( float [ , ] Standard, out float[ , ] CorMatrix) ; /* 调用方法求相关系数矩阵* /
myEigenValueVector = new CalEigenValueVector( ) ;
myEigenValueVector. calEigenValueVector( ref myStandCorMatrix. CorMatrix, out float [ p] EigenValue;
out float [ p, p] EigenVector)
/* 调用相应方法计算矩阵 CorMatrix_{p×p}的特征值、特征向量* /
myEvaluateValue = new CalEvaluateValue( ) ;
myEvaluateValue. calTotalEvaluate( ref myEigenValueVector. EigenValue, ref myStandCorMatrix. Standard, out float [ n, 1] Value, out int k)
/* 调用相应方法计算样本的综合评价值* /
}
public float GetStandradMatrix( )
/* 获得原始样本矩阵的标准化矩阵* /
{ return myStandCorMatrix. Standard; }
public float GetCorMatrix( )
/* 获得标准化矩阵的相关系数矩阵* /
{ return myStandCorMatrix. CorMatrix; }
public float GetEigenValue( )
/* 获得相关系数矩阵的特征值* /
{ return myEigenValueVector. EigenValue; }
public float GetEigenVector( )
/* 获得相关系数矩阵的特征向量* /
{ return myEigenValueVector. EigenVector; }
public float GetEvaluateValue( ) /* 获得各样本的综合评价值* /
{ return myEvaluateValue. EvaluateValue; }
public int GetIndexNum( ) /* 获得选取的指标个数* /
{ return myEvaluateValue. k; }
}
```

整个算法如上所述, 全部用 C#实现, 实现平台为 Visual Studio. NET 2003, Windows XP 操作系统, 128MB 内存。主成分分析类 PriComponentAnalysis 的接口字段、方法用修饰符 Public, 将其暴露给接口程序, 接口程序只需将类 PriComponentAnalysis实例化, 按照二维数组格式向本算法提供原始样本数据, 然后调用方法 PriComAnalysis( ) , 即可完成整个主成分分析计算。分析结果可以用指定的相应方法获得, 如调用 GetEvaluateValue( ) 方法即获得各样本的综合评价值, 完全实现了主成分分析算法的封装, 便于维护和升级。

## 3 基于 C#的主成分分析模块框架及其与 GIS 软件接口技术

### 3.1 主成分分析模块实现

在用 C#实现主成分分析算法的基础上, 开发了基于 C#的主成分分析模块。实现框架如图 1 所示。本模块数据输入可提供从文件、数据库和数据仓库读取, 实现了获取数据的多样化; 输出部分除提供主要功能各样本的综合评价值外, 还提供获取原始样本数据标准化结果、相关系数矩阵、选取的主成分个数等五个开放的接口, 可根据需要选用。

本模块除可以从数据仓库、数据库读取数据外, 还可以读取常用的* . xls, * . txt, 自定义的* . pca 等文件数据。下面以读取* . txt 文件为例介绍相应的代码:

```
public void LoadDataFromTxt( string strName)
{ PriComponentAnalysis mypca = new PriComponentAnalysis( ) ;
 /* 创建 PriComponentAnalysis 实例* /
 string str, [ ] tokens; /* 定义字符串变量, 字符串数组* /
 try{
 StreamReader fileread = File. OpenText( strName) ;
 /* 打开指定的文本文件* /
}
  catch( Exception e)
  { … } /* 打开文件时发生错误的异常处理* /
  str = fileread. ReadLine( ) ;
  tokens = str. Split( " " . ToCharArray( ) ) ;
  mypca. p = tokens. Length; /* 获得指标个数* /
  while( str! = null) /* 读取到文件结束停止读取* /
  {
    for( int j = 0; j < mypca. p; j ++ )
    { mypca. Input[ mypca. n, j] = Convert. ToSingle( tokens[ j] ) ; }
    str = ts. ReadLine( ) ;
    mypca. n ++ ; /* 获取样本个数* /
  }
}
```

该方法非常方便地实现了文本文件的读取, 读取过程中对可能出错的打开文件阶段进行了异常处理, 避免因为读取文件的错误导致整个主成分模块, 甚至调用该模块的母程序的异常, 保证了主成分分析模块组件的鲁棒性。

### 3.2 与 GIS 软件的接口技术( 以 MapInfo 开发的 GIS 软件为例)

主成分分析模块与基于 MapInfo 平台开发的 GIS 软件之间的集成可通过两种方式实现, 如图 2 所示。

( 1) 编写出应用程序并编译为 PriComAnalysis. exe, 然后在 GIS 软件中使用运行外部程序的方式: RUN PROGRAM " PRICOMANALYSIS. EXE" 。

( 2) 使用 OLE/DDE 技术使 C#开发的主成分分析模块组件与 GIS 软件集成。

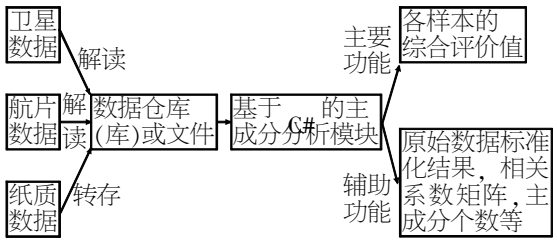


图 1 主成分分析模块实现框架图

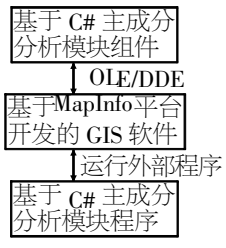


图 2 主成分分析模块与 GIS 软件的集成模式

在. NET Framework 环境下用 C#开发主成分分析模块具有移植性好、升级和维护方便等优点。在实现与 GIS 软件集成时, 既可以将地学分析中的具体分析模块与主成分分析集成为一个完整的模块, 也可以将主成分分析模块作为一个共享模块。应用程序安装时将其复制到计算机中所有应用程序都能访问到的位置, 这个位置称为全局装配件高速缓冲存储器( Global Assembly Cache, GAC) 。一般位置为: 系统盘符: \WINDOWS \system32 目录。由于此模块中包含了完全自我描述的元信息, 不再需要其他信息就可以正常使用, ( 下转第 54 页)

象引用关系图)、类系结构图(类派生/包含/引用关系图)、系统流程图(函数引用关系图)、函数流程图(独立代码块执行关系图)。

在组织结构图中,用折角的矩形表示源文件,矩形表示符号,带箭头的连接线表示关系;在函数耦合图中,用折角的矩形表示函数,矩形表示全局变量和全局对象,带箭头的连接线表示关系;类系结构图直接采用 OMT 表示法进行描述;在系统流程图和函数流程图中,用矩形表示函数和独立代码块,带箭头的连接线表示执行关系并在连接线上标注运行条件。

模型文档采用 Microsoft Office 应用程序对象模型来生成,其中表格文档采用 Microsoft Word 对象模型生成,关系图采用 Microsoft Visio 对象模型生成。在自动生成的关系图中,图元被均匀分布到页面上,用户可以在生成文档上对布局进行重新调整以达到更理想的表达效果。

### 2.4 深度分析

通过源代码分析所获得的基本模型信息能够反映软件设计的基本结构框架,在基本模型信息的基础上做进一步的深度分析可获得更深层次的模型信息。例如,通过对独立代码块间执行关系的分析可以确定代码段的所有可能运行流程以及相应的运行条件,这是源代码测试数据设计的重要依据。通过对软件模型中的结构分析、流程分析和数据流分析,还可以帮助设计者找出代码设计中的冗余结构,从而对代码设计做出优化。另外,在处理实际开发项目时,通常会涉及到许多具体的设计模式,要更有效地对这些具体设计模式做出分析和处理,也需要对基本模型信息做进一步的建模和分析。

## 3 小结

逆向建模与编译都是对源代码进行分析和处理的过程,可以采用相同的设计技术,但两者处理的深度、细度和目标不同。

虽然目前很多开发环境已提供了逆向建模的功能,但所生成的模型文档还比较简单,远远不能满足应用需求。基于源代码分析的逆向建模可以提取出源代码中的各种对象信息、分析对象之间的相互关系、建立模型数据库并生成模型描述文档,从而更准确、更直观、更深入地表现软件的设计结构和设计框架,帮助阅读者更直观、更方便地阅读、理解和分析源代码,通过进一步的深度分析和建模还可帮助设计人员制定软件的测试方案和测试数据,帮助其对软件的设计进行优化。

在基于 Web Sphere 的化工数据库搜索引擎开发项目中,即采用逆向建模的方法生成了对 Web Sphere 源代码的结构模型描述和流程描述,为分析、理解和改进源代码提供了极大的方便,从而提高了项目的开发效率。需要注意的是,逆向建模可以作为编写模型文档的辅助手段,但并不能作为编写模型文档的替代手段。在模型文档中包含着很多设计思想、设计逻辑的表述,而逆向建模过程只能对代码进行静态分析,并不能完全、自动地解释与表现其中的思想和逻辑。

参考文献:


[1] aniel Jackson. Alloy: A New Technology for Software Modeling [C]. Tools and Algorithms for the Construction and Analysis of Systems, Grenoble: Springer-Verlag GmbH, 2002. 20-21.

[2] Gill J Y, Kent S. Three Dimensional Software Modeling[C]. Proceedings of the 1998 International Conference on Software Engineering, 1998.

[3] F J Argul, M Castro, A Delgado. Software Modeling of Grid-connected Photovoltaic Buildings[J]. Progress in Photo Voltaics, 2003, 11: 255-273.

[4] 麻志毅,孟祥文. 面向对象的软件建模工具的研制[C]. 北京: 2001 全国软件技术研讨会, 2001.

[5] Balmas F. Using Dependence Graphs as a Support to Document Programs[C]. Proceedings of the 2nd IEEE International Workshop on Source Code Analysis and Manipulation, 2002.

[6] 刘亚军,高建国. 一种表单驱动的逆向建模方法[J]. 小型微型计算机系统, 2002, 23(6): 747-750.

[7] Giuseppe Antonio Di Lucca, Anna Rita Fasolino, Porfirio Tramontana, *et al*. Recovering a Business Object Model from Web Applications[C]. Dallas: The 27th Annual International Computer Software and Applications Conference(COMPSAC 2003), 2003.

[8] Giuseppe Antonio Di Lucca, Anna Rita Fasolino, Porfirio Tramontana, *et al*. Abstracting Business Level UML Diagrams from Web Applications[C]. Amsterdam: The 5th IEEE International Workshop on Web Site Evolution(WSE 2003), 2003.

[9] Pinzger M, Oberleitner J, Gall H. Analyzing and Understanding Architectural Characteristics of COM + Components[C]. The 11th IEEE International Workshop on Program Comprehension, 2003.

[10] Krzysztof Ciebiera, Piotr Sankowski. Improving Web Sites by Automatic Source Code Analysis and Modifications[C]. Munich: The 4th Web Engineering, 2004.


作者简介:


彭四伟(1970-),男,副教授,在读博士生,研究方向为过程工业监测、编译技术、并行计算;朱群雄,男,教授,博士,研究方向为计算机应用技术、控制理论与控制工程、信息工程。


(上接第 40 页)即使以后主成分分析模块升级,安装过程中仅需简单覆盖原先文件即可,给安装带来了很大方便。

## 4 总结

以开发主成分分析模块为例,介绍了主成分分析的主要计算步骤,并给出了其计算机实现的详细算法,最后介绍了基于 C#的主成分分析模块实现及其与 GIS 软件的接口技术。本文所介绍的主成分分析模块开发技术通用性强,可推广应用于使用C#开发其他多元分析方法计算软件。

参考文献:


[1] arli Watson, Marco Bellinaso, *et al*. C#入门经典[M]. 康博. 北京: 清华大学出版社, 2003. 1-5, 577-579.

[2] Joseph Mayo. C#技术内幕[M]. 王启丁,高镜文,刘明,等. 北京: 机械工业出版社, 2003. 490.

[3] 元昌安,唐常杰,等. 复共线性空间数据回归模型挖掘算法及其实现[J]. 四川大学学报(自然科学版), 2004, 41(1): 66-70.

[4] 袁志发,周静芋. 多元统计分析[M]. 北京: 科学出版社, 2003. 188-201.


作者简介:


饶元(1982-),男,安徽庐江人,硕士研究生,研究方向为数据挖掘;元昌安(1964-),男,安徽肥东人,副教授,硕导,主要研究方向为数据库与知识工程;邓松(1980-),男,安徽肥东人,硕士研究生,研究方向为数据挖掘;杨乐婵(1980-),女,广西梧州人,硕士研究生,研究方向为数据挖掘;廖剑平(1979-),女,广西桂林人,硕士研究生,研究方向为数据挖掘。