高等学校教材

控制系统计算机辅助分析

上海电力学院 杨平 主编

高 等 学 校 教 材

控制系统计算机辅助分析

上海电力学院 杨平 主编

水利电力出版社

内 容 提 要

本书讲述了控制系统计算机辅助分析与设计的基本概念和算法。全书共分九章，较全面地介绍了单输入单输出控制系统和多变量控制系统的模型转换、特性分析和控制器设计三方面的计算机辅助分析与设计的原理和方法，还介绍了系统模型辨识的计算机辅助设计方法。

本书针对高等学校有关自动控制专业的本科生的选修课编写，也适用于相关专业的研究人员和工程技术人员自学或参考。

高等学校教材

控制系统计算机辅助分析

上海电力学院 杨平 主编

*

水利电力出版社出版

（北京三里河路 6 号）

新华书店北京发行所发行·各地新华书店经售

北京市地矿局印刷厂印刷

*

787×1092 毫米 16 开本 8.5 印张 187 千字

1995 年 11 月第一版 1995 年 11 月北京第一次印刷

印数 0001—1770 册

ISBN7-120-02388-8/TP·92

定价 6.80 元

前　言

控制系统计算机辅助设计是一门新兴的边缘学科。它所涉及的学科之多、内容之广，常使得作者觉得要在很有限的篇幅内给予读者一个较全面较深入的阐述是非常困难的。所以在编写过程中，作者一再提醒自己：这是一本本科大学生的选修教材，目的是提供给他们一些关于控制系统计算机辅助设计（CSCAD）的基本知识和原理，使得他们在自动控制理论及系统的学习中，在课程设计和毕业设计的过程中不对 CSCAD 感到茫然并且能应用所学知识解决一些问题或能进一步深入下去。在选修这门课程之前，假定学生已先修过有关计算机原理、计算机高级语言、数值计算方法、自动控制原理和自动控制系统的相应课程。

针对篇幅小而内容多的矛盾，本书的编写原则定为放弃复杂的公式推导和证明，直接给出算法公式和程序框图；只讨论 CSCAD 的基本算法原理而不论述 CSCAD 的系统设计和软硬件实现问题。

本书的书名虽定为“控制系统计算机辅助分析”，但由于分析和设计过程常常是连在一起的，所以书中没有加以刻意地区分。在更多的情况下，按照普通的说法一概称为控制系统计算机辅助设计。

本书的第一章阐述 CSCAD 的基本概念和任务；第二章阐述 CSCAD 中常用的数值算法；第三、四、五章阐述单输入单输出控制系统的模型转换、特性分析和控制器设计的计算机辅助设计（CAD）原理和算法；与之相应的第七、八、九章阐述的是多变量系统的模型转换、特性分析和控制器设计的 CAD 问题；第六章介绍系统模型辨识的 CAD 方法。

全书由上海电力学院自动控制系杨平主编，书中第六章和第七、八、九章分别由华北电力学院动力系的张栾英和吕丽霞编写。书稿经华北电力学院动力系孟希哲全面审阅并提出改进意见，在此表示感谢。

编　者

1993 年

目　　录

第一章 绪　　论

第一节 引　　言

二三十年前，控制工程师或控制理论研究者的工作台上只能看到铅笔、图纸、滑动计算尺，最多有个模拟计算机。而今一种以计算机为主机的控制系统计算机辅助设计系统(Control System Computer-Aided Design，CSCAD)已成为大多数工程师或控制理论研究者的必不可少的工具，同时它也成为有关控制课程的有效的教学辅助手段。它既可作为教师的演示台，又可作为学生的实验台。有了 CSCAD，人们再也不用为逐点描绘一条奈奎斯特(Nyguist)曲线而费心费力，也不用为求解一个李卡蒂方程而头痛。高速的数字电子计算机及其现代化的外围设备可以使得一个复杂的、大计算量的控制系统分析、综合或设计任务在瞬间完成。借助于这样一个有效的工具，控制工程师的直觉判断和丰富的设计经验得以充分地发挥；控制研究者的新设计方案得以迅速地落实；学生学习控制理论的进程可以大大地加快。

经典的控制理论主要是单变量控制系统的频域分析和设计，当扩展到多变量系统时，大量的计算和绘图就成为一个大障碍。正是在这种情况下，控制系统的计算机辅助设计应运而生。英国的卢森布洛克(Rosenbrock)为此写了专著，书名为《控制系统计算机辅助设计》。这本书专门论述了多变量控制系统的计算机辅助频域设计问题。而今天的"控制系统计算机辅助设计"所包涵的内容已经大大拓宽了，并还在进一步拓宽。频域的或时域的，连续的或离散的，确定性的或随机性的，最优的或自适应的，专家系统的或神经网络的，各种控制理论都能与计算机辅助设计(CAD)联系起来。另一方面，计算机学科的新发展也使得计算机辅助设计的方法和手段不断地充实和更新。新的计算机硬件和软件技术正在不断地引入到控制系统计算机辅助设计的领域来。

第二节　控制系统计算机辅助设计的基本概念

一个控制系统计算机辅助设计(CSCAD)系统的硬件部分就是一个计算机的硬件系统。它包括主计算机、输入设备(如键盘)、输出设备(如显示器和打印机)及存储设备(如软、硬盘驱动器)。这套设备可以是一台个人计算机(如 IBM PC/XT)，也可以是一个计算机工作站(如 Svn Station)，或者是一个计算机网络系统(如 VAX)。不同的计算机硬件系统应配相应的 CSCAD 软件，因为不同的硬件设备对 CSCAD 的功能实现有着直接的关系。高分辨率的彩色显示器能比单色显示器表现出更细致、更逼真的图形。基于计算机工作站的 CSCAD 系统可比基于个人计算机的 CSCAD 系统实现更强的功能。

CSCAD 系统的软件结构可分为两种，一种形式较为简单，即只利用某一种标准的计算机语言，如用 FORTRAN 或 BASIC 语言编成的软件包。运行这种软件包时，人机交互的形式是问答式的对话。当有多项选择时，常采用菜单驱动方式。这种具有问答式的人机交互形式的软件包便于初学者使用，能较好地完成预定的任务，但是对于有经验的用户来说就太缺少灵活性了。预定的工作程序对于多变的工作任务来说常常是低效率的，于是一种较高级的软件结构发展起来，那就是在已有的用某种标准计算机语言编写的子程序库上，再加一个“语言翻译器”，它能够把一种专门的 CAD 语言翻译成可执行的语言。这样就使得命令式的人机交互形式建立起来。用户可用专门的 CAD 语言向系统下命令。即可以下一道命令，当即执行；也可以下一串命令，批量处理。专用的 CAD 语言的设计目前也是一个研究课题。它讲究的是语言的简明性、自然性和可扩展性。例如，基于 MATLAB 的 CTRL-C 软件包就使用着较自然的矩阵运算语言，比如

$$a=[1\ 2;\ 3\ 4] \quad \text{意即} \quad a=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$$

$$c=a*b \qquad \text{意即 } c \text{ 是 } a \text{ 矩阵乘 } b \text{ 矩阵的积矩阵}$$

这样的语言对于懂得矩阵运算的用户来说，非常简单，不用记，一看就会用了。此外，CTRL-C 软件还有统一的文件处理、直接的运算和易于扩展命令词汇的特点。正是由于这些特点使它成为一种较受欢迎的软件。一般说来，一个优秀的软件包可使用户一用就爱用，并产生开发它的愿望。最理想的是达到使用户自然纯熟地使用这个工具去全心致志地完成他的任务而感觉不到工具形式的存在。

CSCAD 的数据结构应当是多种多样的，以便灵活地处理整数、实数、复数、数组、矩阵、多项式和多项式矩阵。但是具有多种数据结构的软件一般较难设计。目前出现的大多数软件包只是运行在整数和实数、数组和矩阵结构上。不过这种有些单调的数据结构已能满足基本的 CSCAD 的需要。因为复数可看成一对实数，而多项式可用一个系数数组来表示。

CSCAD 的编程语言，目前用得最多的仍是 FORTRAN 语言。这不但是因为应用 FORTRAN 语言的历史悠久，也因为有许多数值算法程序库都是用 FORTRAN 语言编写的，如 Eispack 和 Linpack 或 Slice 和 Nag。然而人们也意识到 FORTRAN 并不是最好的编程语言，并且开始尝试用其它的语言，如 PASCAL、Ada、C 语言等。

在命令驱动的 CSCAD 软件包中的“语言翻译器”也可用标准的编程语言来设计，如用 APL、LISP、LOGO 或 FORTH 语言。这些编程语言与传统的 FORTRAN 语言的差别在于它的环境具有可扩展性。即用户可创造语言中的新词，新词一旦创立就可用作一个永久性的词。

第三节　CSCAD 的任务

一个 CSCAD 系统一般可用来协助控制工程师完成下列任务：

(1) 受控过程的模型化：数据滤波、时间序列分析、相关分析、模型参数辨识、模型验证、模型简化。

(2) 系统的特性分析：可控性分析、可观测性分析、稳定性分析、频域特性分析以及领域特性图的绘制(如伯德图、奈奎斯特图、尼柯尔斯图)、根轨迹图绘制。

(3) 控制器的设计：校正装置设计、PID 调节器参数的整定、极点配置状态反馈控制器和观测器的设计、最优二次型调节器的设计、卡尔曼滤波器的设计。

(4) 系统仿真和性能检验：线性系统仿真、非线性系统仿真、控制响应图示、性能指标计算。

CSCAD 软件包含有完成上述任务的专用程序。而在这些专用程序中又包含有许多相同的部分，如常用算法的数值计算、图形的显示和数据的输入输出。其中常用的算法有：多项式运算、矩阵运算、多项式矩阵运算、矩阵指数 e^{At} 的计算、求解李雅普诺夫方程、求解李卡蒂方程、最优化算法、傅里叶变换。

第二章 基 础 算 法

第一节 多 项 式 运 算

多项式的运算在控制系统设计过程中颇为常见，如在求系统的特征根和做传递函数的综合时均要用到。

多项式的运算包括两个多项式的相加、相减、相乘和相除以及一个多项式的求根。其中相加、相减和相乘的运算较为简单；相除运算则分正幂指数和负幂指数两种情况；至于多项式求根运算则有多种数值近似计算方法，这里只给出一种实用的算法。

为了以下叙述方便，先定义四个多项式为

$$A(s) = a_n s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0 = \sum_{i=0}^{n} a_i s^i$$

$$B(s) = b_m s^m + b_{m-1}s^{m-1} + \cdots + b_1 s + b_0 = \sum_{i=0}^{m} b_i s^i$$

$$C(s) = c_l s^l + c_{l-1}s^{l-1} + \cdots + c_1 s + c_0 = \sum_{i=0}^{l} c_i s^i$$

$$D(s) = d_k s^k + d_{k-1}s^{k-1} + \cdots + d_1 s + d_0 = \sum_{i=0}^{k} d_i s^i$$

这些都是正幂指数多项式，实际中也常见负幂指数多项式，为此也定义四个类似的多项式为

$$A(z^{-1}) = \sum_{i=0}^{n} a_i z^{-i}$$

$$B(z^{-1}) = \sum_{i=0}^{m} b_i z^{-i}$$

$$C(z^{-1}) = \sum_{i=0}^{l} c_i z^{-i}$$

$$D(z^{-1}) = \sum_{i=0}^{k} d_i z^{-i}$$

一、多项式相加减

设

$$C(s) = A(s) \pm B(s)$$

则可推得

$$C(s) = \begin{cases} \sum_{i=0}^{n} (a_i \pm b_i) s^i & n = m \\ \sum_{i=0}^{m} (a_i \pm b_i) s^i + \sum_{i=m+1}^{n} a_i s^i & n > m \\ \sum_{i=0}^{n} (a_i \pm b_i) s^i \pm \sum_{i=n+1}^{m} b_i s^i & m > n \end{cases}$$

于是可整理得系数 C_i 的计算公式

$$c_i=\left\{\begin{array}{ll} a_i \pm b_i & n=m \\ \left\{\begin{array}{ll} a_i & i>m \\ a_i \pm b_i & i \leqslant m \end{array}\right. & n>m \\ \left\{\begin{array}{ll} \pm b_i & i>n \\ a_i \pm b_i & i \leqslant n \end{array}\right. & n<m \end{array}\right\} \begin{array}{l} i=0,1,2,\cdots,l \\ l=\max\{n,m\} \end{array}$$

根据以上公式就可编写计算机程序。实际上，当序列 a_i 和序列 b_i 不一样长时，若将短的序列用零补齐，则可采用很简单的计算公式：

$$c_i=a_i \pm b_i \quad i=0,1,\cdots,l \quad l=\max\{n,m\}$$

以上针对 $C(s)=A(s)\pm B(s)$ 的 c_i 计算公式也可证明同样适用于 $C(z^{-1})=A(z^{-1})\pm B(z^{-1})$ 的情况。

二、多项式相乘

设

$$C(s)=A(s)\cdot B(s)$$

即

$$\sum_{i=0}^{l} c_i s^i=\sum_{i=0}^{n} a_i s^i \sum_{j=0}^{m} b_i s^i$$

则可归纳出系数 c_k 的计算公式：

$$c_k=\sum_{i,j/i+j=k} a_i b_j \qquad k=0,1,\cdots l \quad l=m+n$$

据此公式可编出如下的 BASIC 程序：

```
10  FOR I=0 TO N
20  FOR J=0 TO M
30  K=I+J
40  C(K)=C(K)+A(I)*B(J)
50  NEXT J : NEXT I
```

显然以上的计算公式和计算程序对于 $C(z^{-1})=A(z^{-1})\cdot B(z^{-1})$ 也是有效的。

三、多项式相除

一般说来，两个多项式相除运算有两种情况需要区别。对于两个正幂指数多项式相除，其商式 $C(s)$ 和余式 $D(s)$ 是唯一的。而对于负幂指数多项式相除，其商式 $C(z^{-1})$ 和余式 $D(z^{-1})$ 则将随人为设定的 $C(z^{-1})$ 的次数 l 的改变而变化。

1. 正幂指数多项式相除

设

$$A(s)/B(s)=C(s)+D(s)/B(s)$$

或写成

$$A(s)=B(s)C(s)+D(s)$$

则可推得 $C(s)$ 和 $D(s)$ 的系数计算公式为

$$\left.\begin{array}{ll} c_i=a_{i+m}/b_m & \\ a_{i-j}\leftarrow{}^{❶}\,a_{i+j}-c_ib_i & j=m-1,m-2,\cdots,0 \\ d_k=a_k & k=m-1,m-2,\cdots,1,0 \end{array}\right\} i=n-m,n-m-1,\cdots,1,0$$

由以上公式可编写出如下的 BASIC 程序：

```
10  FOR I=N-M TO  0  STEP-1
20  C(I)=A(I+M)/B(M)
30  FOR J=M TO 0 STEP-1
40  A(I+J)=A(I+J)-C(I)*B(J)
50  NEXT J : NEXT I
60  FOR I=0 TO M-1
70  D(I)=A(I):NEXT I
```

2. 负幂指数多项式相除

设

$$A(z^{-1})/B(z^{-1}) = C(z^{-1}) + D(z^{-1})/B(z^{-1})$$

则可推得 $C(z^{-1})$和 $D(z^{-1})$的系数计算公式：

$$\left.\begin{array}{ll} c_i = a_i/b_0 & \\ a_{i+j}\leftarrow a_{i+j} - c_ib_j & j = 1,2,\cdots,m \\ d_k = a_k & k = l+1,l+2,\cdots,l+m \end{array}\right\} i = 0,1,\cdots,l$$

式中 l 是人为设定的。当 l 增加时 c_i 的项数随之增加。当 l 变化时，$\{d_k\}$序列全都改变。当 l 很大或趋于无穷时，余式将不再考虑，这时就是长除运算情况。

四、多项式求根

对于代数多项式方程求根已有多种算法。例如，二分法、迭代法、牛顿法、线性插值法、劈因子法等等。在控制系统 CAD 中常用的是劈因子法。因为用这种方法既可方便地求复根，又能避免复数运算。以下就只介绍这种算法。

设所考虑的多项式表示为

$$F(s) = \sum_{i=0}^{n} a_i s^{n-i} = a_0 s^n + a_1 s^{n-1} + \cdots + a_{n-1}s + a_n$$

再设一个二次因子式

$$P(s) = s^2 - us - v$$

则当用 $P(s)$除 $F(s)$时，就有

$$F(s) = P(s)Q(s) + R(s) \qquad (2\text{-}1)$$

式中 $Q(s)$为商，$R(s)$为余项。若 $R(s)=0$则 $P(s)$就是 $F(s)$的一个因式，$P(s)=0$的两个根

$$s_{1,2} = \frac{u \pm \sqrt{u^2 - 4v}}{2}$$

就是 $F(s)$的根。若 $R(s)\neq 0$，则可调整 $P(s)$中的系数 u 和 v，使 $R(s)$趋于0，从而得到 $F(s)$的两个根。如果 $F(s)$有两个以上的根，则可将 $R(s)$趋于零时的 $Q(s)$看成新的 $F(s)$，重复用以

---

❶ 符号←表示右边计算出的内容赋值给左边的变量。

上方法劈得所有因子。如果 $F(s)$ 只有一个根或者最后的 $Q(s)$ 只有一个根，则直接从 $F(s)$ 或 $Q(s)$ 的一次因式中解得这个单根。以上就是劈因子算法原理。

为了推算方便，特设

$$Q(s)=\sum_{i=0}^{n-2}b_i s^{n-2-i}=b_0 s^{n-2}+b_1 s^{n-3}+\cdots+b_{n-3}s+b_{n-2}$$

$$R(s)=b_{n-1}(s-u)+b_n$$

将 $F(s)$、$P(s)$、$Q(s)$ 和 $R(s)$ 的定义式都代入式(2-1)中并展开，然后比较等式两边的同类项，可得如下系数关系：

$$\begin{aligned}
b_0&=a_0\\
b_1&=a_1+ub_0\\
b_2&=a_2+ub_1+vb_0\\
b_i&=a_i+ub_{i-1}+vb_{i-2}\qquad i=3,\cdots,n
\end{aligned}$$

为使 $R(s)$ 趋于0，须使 b_{n-1} 和 b_n 为零。考虑 u 和 v 的调整：

$$\begin{aligned}
u^*&=u+\Delta u\\
v^*&=v+\Delta v
\end{aligned}$$

相应的 b_n 和 b_{n-1} 显然是 u^* 和 v^* 的函数：$b_{n-1}(u^*,v^*)$，$b_n(u^*,v^*)$。将这两个函数在 (u,v) 点按泰勒公式展开，则有

$$b_{n-1}(u^*,v^*)=b_{n-1}(u,v)+\frac{\partial b_{n-1}}{\partial u}\Delta u+\frac{\partial b_{n-1}}{\partial v}\Delta v$$

$$b_n(u^*,v^*)=b_n(u,v)+\frac{\partial b_n}{\partial u}\Delta u+\frac{\partial b_n}{\partial v}\Delta v$$

令上两式为零，则可得到关于 Δu 和 Δv 的线性方程组：

$$\left.\begin{aligned}
\frac{\partial b_{n-1}}{\partial u}\Delta u+\frac{\partial b_{n-1}}{\partial v}\Delta v&=-b_{n-1}\\
\frac{\partial b_n}{\partial u}\Delta u+\frac{\partial b_n}{\partial v}\Delta v&=-b_n
\end{aligned}\right\}\qquad(2\text{-}2)$$

现在的问题是如何计算式(2-2)中的四个偏导数。由于 $\{b_i\}$ 序列中各元素相互关联，故需先算出关于 b_i 的偏导数：

$$\begin{aligned}
&\frac{\partial b_0}{\partial u}=0 && \frac{\partial b_0}{\partial v}=0\\
&\frac{\partial b_1}{\partial u}=b_0 && \frac{\partial b_1}{\partial v}=u\frac{\partial b_0}{\partial v}=0\\
&\frac{\partial b_2}{\partial u}=b_1+ub_0 && \frac{\partial b_2}{\partial v}=b_0+u\frac{\partial b_1}{\partial v}=b_0\\
&\frac{\partial b_3}{\partial u}=b_2+u\frac{\partial b_2}{\partial u}+v\frac{\partial b_1}{\partial v} && \frac{\partial b_3}{\partial v}=b_1+v\frac{\partial b_1}{\partial v}+u\frac{\partial b_2}{\partial v}=b_1+ub_0\\
&\quad\vdots && \quad\vdots
\end{aligned}$$

由观察可知，上述推算很有规律。若设

$$c_i=\frac{\partial b_i}{\partial u}$$

则有

$$\frac{\partial b_i}{\partial v} = c_{i-1}$$

并有递推关系

$$c_i = b_{i-1} + uc_{i-1} + vc_{i-2} \quad i = 0,1,2,\cdots,n \text{ (设 } c_{-1} = c_{-2} = 0)$$

于是关于 Δu 和 Δv 的线性方程组可改写成：

$$\left.\begin{aligned} c_{n-1}\Delta u + c_{n-2}\Delta v &= -b_{n-1} \\ c_n\Delta u + c_{n-1}\Delta v &= -b_n \end{aligned}\right\} \tag{2-3}$$

从而解得

$$\Delta = c_{n-1}^2 - c_{n-2}c_n$$
$$\Delta u = (c_{n-2}b_n - c_{n-1}b_{n-1})/\Delta$$
$$\Delta v = (c_n b_{n-1} - c_{n-1}b_n)/\Delta$$

然后可用 Δu、Δv 进行$\{b_i\}$序列的修正。当$|\Delta u|$和$|\Delta v|$很小时，可认为 $R(s)$已趋于零。于是两个因子被求得。

至此，可归纳出劈因子算法如下：

(1) 给定 $\varepsilon>0$，$u\leftarrow 0$，$v\leftarrow 0$，$N\leftarrow n$，$k\leftarrow 1$。

(2) 计算$\{b_i\}$：$b_0=a_0$

$b_1=a_1+ub_0$

$b_i=a_i+ub_{i-1}+vb_{i-2} \quad i=2,\cdots,N$

(3) 计算$\{c_i\}$：$c_i=b_i+uc_{i-1}+vc_{i-2} \quad i=0,1,\cdots,N$

(4) 计算 Δu 和 Δv：$\Delta=c_{N-1}^2-c_{N-2}c_N$

$\Delta u=(c_{N-2}b_N-c_{N-2}b_{N-1})/\Delta$

$\Delta v=(c_{N-2}b_{N-1}-c_{N-1}b_N)/\Delta$

(5) 若$\|\Delta u\|\leqslant\varepsilon$ 和$\|\Delta v\|\leqslant\varepsilon$，则转步骤(7)

(6) $u\leftarrow u+\Delta u$，$v\leftarrow v+\Delta v$，转(2)

(7) $S(k)\leftarrow(u+\sqrt{u^2-4v})/2$

$S(k+1)\leftarrow(u-\sqrt{u^2-4v})/2$

$k\leftarrow k+2$；$a_i\leftarrow b_i$ $i=0,1,\cdots,N-2$；$N\leftarrow N-2$

当 $N\geqslant 2$，则转(2)。

(8) 劈因子计算结束。

第二节 矩 阵 运 算

矩阵运算在CSCAD中最为常见，特别是对于基于状态空间法的控制系统设计。矩阵运算多种多样，有的极为简单，有的非常复杂。以下叙述的几种矩阵运算算法是按照高效、实用的原则选出的。它们包括矩阵的加、减、乘和转置、矩阵求逆、矩阵行列式、特征多项式、特征值与特征向量、奇异值分解以及矩阵的秩运算。矩阵的特征多项式运算将在状态方程转化为

传递函数的模型转换过程中用到。矩阵的特征值和特征向量运算常用于最优控制系统设计。矩阵的奇异值分解可用于系统辨识中最小二乘求解以及系统的可控性和可观测性分析。

一、矩阵的加、减、乘及转置

1. 两个矩阵相加减

设矩阵

$$\boldsymbol{A}=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}_{m\times n} \qquad \boldsymbol{B}=[b_{ij}]_{m\times n} \qquad \boldsymbol{C}=[c_{ij}]_{m\times n}$$

则对于 $\boldsymbol{C}=\boldsymbol{A}\pm\boldsymbol{B}$，有计算公式

$$c_{ij}=a_{ij}\pm b_{ij} \qquad i=1,2,\cdots,m;\ j=1,2,\cdots,n$$

2. 两个矩阵相乘

设矩阵

$$\boldsymbol{A}=[a_{ij}]_{m\times r} \qquad \boldsymbol{B}=[b_{ij}]_{n\times l} \qquad \boldsymbol{C}=[c_{ij}]_{m\times l}$$

则对于 $\boldsymbol{C}=\boldsymbol{A}\cdot\boldsymbol{B}$，有计算公式

$$c_{ij}=\sum_{k=1}^{n}a_{ik}b_{kj} \qquad i=1,2,\cdots,m \qquad j=1,2,\cdots,n$$

3. 矩阵的转置

设矩阵

$$\boldsymbol{A}=[a_{ij}]_{m\times n} \qquad \boldsymbol{B}=[b_{ij}]_{n\times m}$$

则对于 $\boldsymbol{B}=\boldsymbol{A}^{T}$ 运算，有计算公式

$$b_{ij}=a_{ji} \qquad i=1,2,\cdots,n \quad j=1,2,\cdots,m$$

二、矩阵求逆

矩阵求逆计算常用高斯-约当方法。为了保证数值计算的稳定性，需要配合有选取主元素的措施，故有列主元、行主元和全主元高斯-约当法之分。在 CSCAD 中，最常见的是列主元高斯-约当算法。下面就介绍这种算法。

设初始矩阵置于 $n\times n$ 数组 $\boldsymbol{A}$，则 $\boldsymbol{A}^{-1}$ 的计算步骤如下所述。$\boldsymbol{A}^{-1}$ 的结果也放在 $\boldsymbol{A}$ 数组中。

(1) $k\leftarrow 1$

(2) 按列选主元：$|a_{i_k,k}|=\max\limits_{k\le i\le n}|a_{i_k}|$

$c_0\leftarrow a_{i_k,k}$

(3) 若 $c_0=0$，或 $|c_0|<\varepsilon$，则指示 $\boldsymbol{A}$ 阵为奇异阵并结束。

(4) 若 $i_k=k$，则转(5)，否则两行交换：

$$a_{kj}\longleftrightarrow a_{i_k,j} \qquad j=1,2,\cdots,n$$

(5) $h\leftarrow 1/c_0$；　$a_{kk}\leftarrow h$

$a_{i_k}=-a_{i_k}\cdot h$　$i=1,2,\cdots,n$　且 $i\neq k$