扩展CSS仿真程序及其应用

马纪虎　王　刚　马忠芳　吴柯玲

（中国科学院沈阳自动化研究所）

〔提要〕 本文介绍了一个连续－离散复合仿真程序。该程序的连续部分基本取自美国的 CSS 程序。经作者扩充后成为复合仿真程序。扩充的模块包括采样系统仿真模块、两点边值问题求解以及参量最优化等。作者提出的单纯形搜索法改进策略可以显著加快参量最优化过程的收敛速度。

一　前　言

连续系统仿真程序CSS(Continuous Systems Simulation Program)原是美国宾州大学 (University of Pennsylvania) 使用的程序〔1〕。上海交大32室将其在国产DJS—130机上通过。我们在上海交大工作基础上进一步对全部程序加以分析、订正。并开发出全部功能。在此后，我们又将原程序进行了实质性扩充，下面给出我们的工作结果。

二　CSS 源程序分析

CSS 是一种模块化的仿真程序，只要按照手册的简单约定给出仿真图，然后输入相应数据，就可以对各种复杂的系统进行仿真。

对任何连续系统，如果要用 CSS 仿真的话，首先要经过在模拟机上排题的类似过程，即化成基本 CSS 模块组成的仿真图。从数学上说，就是将n 阶非线性微分方程化为n 维一阶微分方程组，其状态方程表示是：

$$\begin{cases}\dot{\overline{Y}}(t)=\overline{f}\,(t,\ \overline{Y}(t),\ \overline{U}(t)) \\ \overline{Y}(t_0)=\overline{Y}_0\end{cases} \qquad (1)$$

CSS 用定步长四阶 Runge-Kutta 方法对（1）式在间距为 h 的离散序列点上求数值解。

四阶Runge-Kutta公式是：

$$\begin{cases}\overline{K_1}=\overline{f}[t^{(r)},\overline{Y}^{(r)}] \\ \overline{K_2}=\overline{f}[t^{(r)}+h/2,\overline{Y}^{(r)}+h/2\,\overline{K_1}] \\ \overline{K_3}=\overline{f}[t^{(r)}+h/2,\overline{Y}^{(r)}+h/2\,\overline{K_2}] \\ \overline{K_4}=\overline{f}[t^{(r)}+h,\overline{Y}^{(r)}+h\overline{K_3}] \\ \overline{Y}^{(r+1)}=\overline{Y}^{(r)}+h(\overline{K_1}+2\overline{K_2}+2\,\overline{K_3} \\ \qquad +\overline{K4})/6\end{cases} \qquad (2)$$

（2）式中除步长h外，都是矢量。

程序根据$t=t^{(r)}$ 时刻$\overline{Y}^{(r)}$值，即n个积分块的输出，求出n个导数值$\overline{Y}^{(r)}$，即为$\overline{K}$。然后顺次算出$\overline{K_2}$，$\overline{K_3}$，$\overline{K_4}$ 就可以求出状态变量在$t^{(r+1)}$时刻的数值$\overline{Y}^{(r+1)}$。

上面所说的求 n 个导数的过程用矢量符号 $\overline{f}$ 表示，在实现时用CSS功能块可以任意组合连接。由于CSS功能块种类繁多，CSS 适应用的系统就可以十分广泛了，

关于CSS的详细功能可参看〔2〕、〔4〕、〔5〕。

三　对 CSS 源程序的订正与改进

上节中曾引证标准四阶 Runge-Kutta 公式，但实际清单分析结果表明，CSS中的算法并不严格符合公式（2），而是按照下式计算的：

$$\begin{cases}\overline{K_1}=\overline{f}(t^{(r)},\overline{Y}^{(r)}) \\ \overline{K_2}=\overline{f}(t^{(r)}+h/4,\overline{Y}^{(r)}+h/2\overline{K_1}) \\ \overline{K_3}=\overline{f}(t^{(r)}+h/2,\overline{Y}^{(r)}+h/2\,\overline{K_2}) \\ \overline{K_4}=\overline{f}(t^{(r)}+3h/4,\overline{Y}^{(r)}+h\overline{K_3}) \\ \overline{Y}^{(r+1)}=\overline{Y}^{(r)}+h(\overline{K_1}+2\overline{K_2}+ \\ \qquad +2\overline{K_3}+\overline{K_4})/6\end{cases} \qquad (3)$$

比较（3）式与（2）式，可知原CSS在计算四个系数时，自变量的选取与标准Runge-Kutta 公式有区别。我们试将上述算法改为标准算法后，对一个有解析解的$u(t)$不为恒值的一阶系统进行仿真，将改进前后结果对比如下：

时间t（秒）	0.1	0.2	0.3	0.4	0.5
原CSS解	0.908465	0.835156	0.77834	0.736447	0.708057
改进CSS解	0.909675	0.831462	0.781637	0.740641	0.713062
解析解	0.909675	0.83746	0.781637	0.740638	0.713058
原误差	-1.21×10^{-3}	-2.3×10^{-3}	-3.3×10^{-3}	-4×10^{-3}	-5×10^{-3}
改进后 i	0×10^{-6}	2×10^{-6}	4×10^{-6}	3×10^{-6}	4×10^{-6}

上述结果表明，对CSS算法进行改进不仅从数学严格性上讲是必要的。而且使程序中的时间变量可以直接引用，对随动系统和变参数系统的仿真提供了便利。

此外，对源程序还有若干处订正，可参阅〔2〕。

四 对CSS的扩展

CSS程序包含的连续仿真功能是很全的，大致包含了CSSL标准的全部功能。CSSL是1967年制定的标准，至今仍是各国连续系统仿真语言文本的主要依据。但是CSSL的背景毕竟是六十年代初期系统仿真的要求，在当时，计算机控制系统的研究、参量最优化等远没有今天这样重要。进入七十年代后，仿真语言向复合语言方向发展〔8〕。因此，我们对CSS的扩展重点放在下述几种离散功能上：

——采样系统的仿真

——参量最优化功能

——求解两点边值问题模块

经过这些扩充后，程序有了实质性变化，成为连续—离散复合仿真语言。

1．关于计算机控制系统的仿真

典型的单变量计算机控制系统结构图示于图1。

图1中数字计算机的行为在Z域中用脉冲传递函数表示：

$$D(Z)=\frac{a_0+a_1Z^{-1}+a_2Z^{-2}+\cdots+a_8Z^{-8}}{1+b_1Z^{-1}+b_2Z^{-2}+\cdots+b_8Z^{-8}}=\frac{Y(Z)}{U(Z)}$$

在时域中用差分方程描述：

$$Y^{(i)}=\sum_{k=0}^{8}a_kU^{(i-k)}-\sum_{k=1}^{8}b_kY^{(i-k)}$$

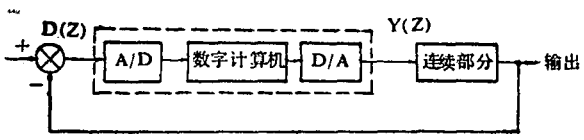


图 1 计算机控制系统典型结构

显然，处理这一差分方程的仿真方法应与连续部分完全不同。我们参照SIMMON〔8〕中连续子系统与离散子系统的仿真连接原理设计了计算机系统仿真模块。与SIMMON比较，本模块不再需要用户考察程序细节，而且将SIMMON的一阶差分方程计算能力扩展为八阶。

关于本模块的详细设计思想可参阅〔3〕。

2．参量最优化功能

在系统仿真研究中，求得最优参量经常是仿真实验的目的。虽然近十多年来，最优化理论及方法有了相当的发展。但是现有的仿真语言标准尚没有把参量最优化作为必不可少的组成部分。因此实际的局面往往是由人判断仿真结果，然后根据经验重新设置参量，进行多趟仿真。用这种原始方法寻找最优参量当然是很

困难的。L.G.Birta 试图改变这种局面，创立了与CSSL 衔接的参量最优化模块[12]。我们采取类似的思路，同时改进了搜索策略，使收敛速度大大加快。

本模块中包含单纯形法（SIMPLX）、最速下降法及共轭斜量法等三种方法，以适应不同问题的需要。

最速下降法是要在负斜量方向上作一维搜索

$$J(a)=\min_{a} J(\overline{p}-a\Delta J(\overline{p}))$$

其中$\overline{p}$为参量矢量。

一维搜索不成问题，关键在于如何求斜量$\Delta J(\overline{p})$，在仿真处理的问题中，导数的显式表达式实际总是不存在的。但是在仿真精度要求之内差分近似可以满足要求：

$$\frac{\partial J}{\partial p_k}\Big|_{\overline{p}}=\frac{J(\overline{p}+\varepsilon e_k)-J(\overline{p})}{\varepsilon}$$

其中$\partial J/\partial p_k$是$\Delta J(\overline{p})$的第K个分量，e^k是$n\times n$单位矩阵的第K列，ε是一个小正数。

单纯形法在各种文献中讨论得已经很多了。我们的经验表明，对很多问题单纯形法都是十分有效的搜索方法。因此，我们把精力集中于单纯形搜索策略的进一步改进。

以往的文献，都致力于如何减少参量最优化过程中的迭代次数，在这方面已没有什么显著改进的余地了。但我们发现，当搜索模块与仿真程序连接时，如果把仿真程序中的积分步长也受最优化模块控制，将仿真步长的调整作为最优化搜索策略的一部分，那么虽然迭代次数并不减少，但总的收敛时间却大大减少了。图2是这种搜索策略的示意框图。

例2取自文献[12]，由例可以看出，经我们改进后的程序得到同样精确的最优点，但时间却大大缩短了。

3．两点边值问题求解块。

采用Pontryagin's原理可将最优控制问题化为两点边值问题。由于CSS本身为用数值方法解任何一种非线性微分方程初值问题提供了

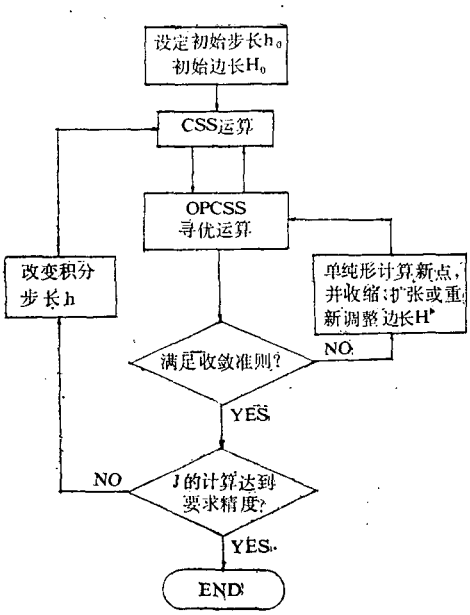


图 2 改进的单纯形法搜索策略

便利，当增加两点边值方法模块后，就可以求解相当大范围内的两点边值问题。两点边值问题用试射法求解时可以表示为：

$$\begin{cases}Y_i=f_i(t,\overline{y},\overline{p},) & i=1\cdots n\\ Y_i(t=0)=Y_{i0} & i=1\cdots m\\ Y_i(t=t_k)=Y_{ik} & i=1\cdots j\end{cases}$$

其中$\overline{P}$为待定j维参数矢量，矢量$\overline{P}$的分量的一部或全部可以是待定的初值。

我们采用牛顿迭代法逐步改进$\overline{P}$的估计值：

$$\overline{P}^{(r+1)}=\overline{P}^{(r)}-\frac{\partial\overline{F}}{\partial p}\Big|_{\overline{p}^{(r)}}\overline{F}$$

其中$\overline{F}$为终点误差矢量，$\partial\overline{F}/\partial P$为Jacobian矩阵，采用一阶差分近似法求出。

例3给出了一个一维两点边值问题的例子，该例取自[9]，两种仿真工具取得了完全一致的结果。但是对用户来讲，使用扩展CSS要简单方便得多。

4．连续仿真功能的扩充

原CSS与标准CSSL相比，缺少纯时延环节、常用一阶、二阶传递函数环节。我们均按照CSSL标准[7]加以补充。此外我们还增加了

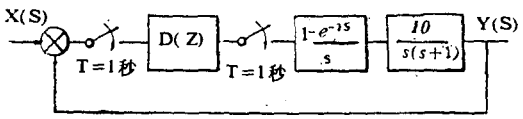
任意n阶微分方程(或传递函数)输入块，用户只要将方程系数输入，程序就可以自动排题后进行仿真计算，便利了用户。

五 应用举例

本节给出三个典型例子，说明扩充CSS的实际应用。本节例子都注明出处，以便读者能与解析解或其它仿真方法得出的解互相对照。当然，在实际问题中可用扩充CSS功能的组合对复杂得多的系统进行仿真。

例1.对图3所示计算机控制系统进行仿真验证。图中$D(Z)$是按最少拍原则设计的[6]。

$$D(Z)=\frac{0.543-0.47132Z^{-1}+(9.99912E-2)Z^{-2}}{1-0.282Z^{-1}-0.718Z^{-2}}$$



其中X(S)为单位速度输入
化为扩充CSS仿真图：

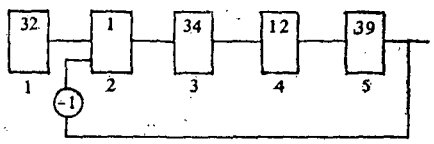


图3 最少拍系统

程序运行后，机器向用户提问$D(Z)$的系数。用户回答后，开始运算，最后得出与[6]中分析解完全一致的响应曲线。结果见图4。

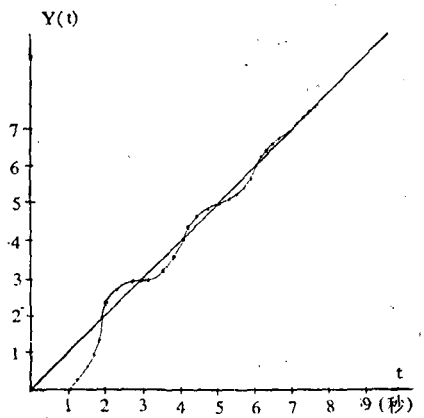


图.4 最少拍系统仿真结果

例2.求图5所示纯滞后对象用比例积分调节器调节时的最优参量a_1及a_2。设输入量Y为0。扰动量Y_L为单位阶跃[10][11][12]。

参量最优化的目的是要使评价函数J取极小值：

$$J(a_1,a_2)=\int_0^{t_f}|y_2(t)|dt;t_f=70$$

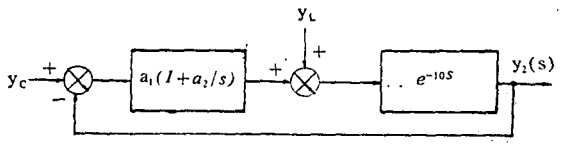


图5 纯滞后过程的比例积分(P+I)调节

图6是用参量最优化模块对图5所示系统的搜索过程，图中曲线B为原文献[12]给出的收敛过程，曲线A为用我们的搜索策略得到的收敛过程。曲线A分为两段，第一段仿真步长为10，因此以极短的时间就完成了13次迭代，但这时的动态计算是不准确的，只是接近了极值点并没有真正达到极值点。因此改变积分步长后，评价函数又上跳了一小段，然后才收敛到真正的极值点。

例3.求解通过边界点$R(x=0)=1$，$R(x=1)=0.6$的曲线$R(X)$，使之绕横轴旋转时所得曲面之表面积取极小值[9]。

根据古典变分法，求得$R(X)$需满足下述方程组及边界条件：

$$\begin{cases}U=-\lambda\sqrt{4\pi^2R^2-\lambda^2}\\ \dfrac{dR}{dx}=U\\ \dfrac{d\lambda}{dx}=-2\pi\sqrt{1+U^2}\\ R(x=0)=1;\quad R(x=1)=0.\end{cases}$$

用CSS可以方便地排出U、R、λ的仿真图，用扩充两点边值块可以自动控制多次迭代过程。这里的待定参量只有一个，就是λ的初值λ_0。图7给出仿真结果。图中曲线Ⅰ—Ⅳ为R的计算曲线，每次迭代由于λ不同，R曲线形状不同，但都满足R的方程。经四次迭代后，$R(x)$即满足边值条件(1,0.6)。

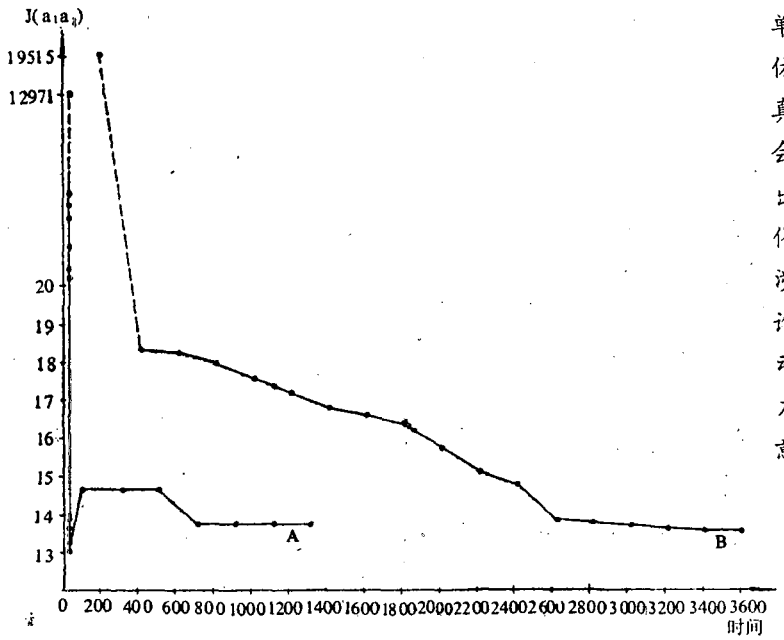


图 6 例2的收敛过程（图中纵座标为目标函数J，横座标为时间）

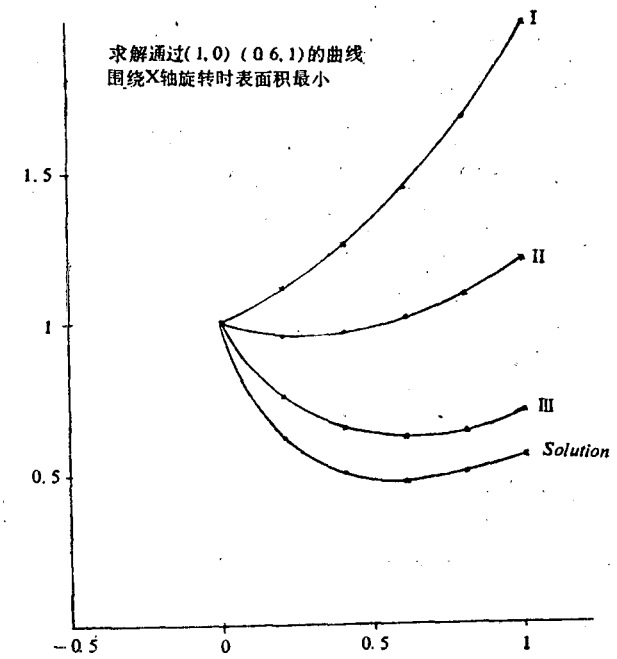


图 7 两点边值问题仿真结果

六 后 记

本文的工作是在科学院学部委员、沈阳自动化所顾问张钟俊教授指导下完成的。此外交大32室曾为我们提供原始CSS清单与纸带；在八一年全国第二次仿真会议及八二年全国“数字仿真及计算机辅助设计”专题讨论会上，许多同志曾对本文初稿提出过宝贵意见及建议；沈阳自动化所徐书新同志曾与作者就文中涉及的数值方法问题进行多次讨论使作者得到很多启发；沈阳自动化所六室机房同志给与我们全力支持，作者仅在此一并致以谢意。

参考文献

〔1〕上海交大外籍专家讲学讲稿(27)：仿真理论和技术，上海交大科技情报室，1981.7。

〔2〕马纪虎、马忠芳，CSS仿真程序分析及应用，全国第三次仿真会议资料，1981.9.北京。

〔3〕马纪虎，关于采样系统仿真的若干问题，“全国数字仿真及控制系统计算机辅助设计”专题讨论会资料1982.11,上海。

〔4〕扩展CSS程序使用手册，沈阳自动所资料，1982,7。

〔5〕孙国基，连续系统仿真语言CSS，全国第二次仿真学习班教材，1982.7。

〔6〕刘植祯、郭木河、何克忠，计算机控制，清华大学出版社，1982。

〔7〕Robert E. Stephenson, Computer Simulation for Engineers,Harcourt Brace Jovanovich,Inc.

〔8〕Tuncer I. Oren, Software for Simulation of Combined Continuous and Discrete Systems, Simulation,Vol.28.,No2,1977.

〔9〕Elmgvist,H, Simnon, An Interactive Program for Simulating Non-linear Systems,Detp. of Automatic Control,Lund Institute of Technology Lund,Sweden,1972.

〔10〕Harriott,P., Process Control Mcgraw-Hill, New York,1964,p.187.

〔11〕John R.Roach and Ee-Pan Chow,An Interactive Digital Simulation and Optimization Package:FORTRAN Programs MINISIM and OPTSIM,Simulation,Vol.25,No.4, 1975。

〔12〕Louis G.Birta,A.,Parameter Optimization Module for CSSL-based Simulation Software, Simulation,Vol.28,No.4,April, 1977.

ABSTRACTS

The Technology of Digital Simulation and Its Application

Xiong Guangleng Sun Guoji

In this paper the status quo of digital simulation in our country is surveyed. And the digital simulation software and its application are described.

Finally, some constructive suggestions as regards this technique in China are given. (p.1)

An Expanded CSS Program and Its Application

Ma Jihu et al.

This paper presents an expanded simulation software package which combines continuous system simulation (CSS) program with discrete systems simulation modules. The enhanced package includes modules for computer control systems, the algorithm for the solution of the two point boundary value problem, and the algorithms for parameter optimization. A new SIMPLX strategy is proposed which speeds up the process of optimization considerably. (p.10)

Continuous System Simulation Language ECSL

Sun Guoji at al.

ECSL is an equation-oriented continuous system simulation language which can be used to simulate the linear and nonlinear systems of the order of 200. Using this language, the source program written in the state equations of the system is explicit and comprehensible. ECSL is compatible with FORTRAN and even more powerful with the system program coded in ANSI FORTRAN, this language is essentially machine-independent.

This paper gives a briefing on the features and rules of ECSL as well as the basic construction of its system program. (p.15)

Continuous System Simulation Language CSSL-F2

Bi Jiancheng et al.

CSSL-F2 (Continuous System Simulaion Language in FORTRAN, Version 2) is a machine-independent language written entirely in FORTRAN for continuous system simulation. The language has been developed by authors on their own in accor-