AJChE 2005, Vol. 5, No. 2, 125–129

Utilization of Mathematical Software Packages in Chemical Engineering Research

Ang Wee Lee
Nayef Mohamed Ghasem
Mohamed Azlan Hussain
Department of Chemical Engineering
University of Malaya
50603 Kuala Lumpur, MALAYSIA
Email: nayef@um.edu.my

Using Fortran taken as the starting point, we are now on the sixth decade of high-level programming applications. Among the programming languages available, computer algebra systems (CAS) appear to be a good choice in chemical engineering as they can be applied easily. Until the emergence of CAS, the assistance from a specialized group for large-scale programming is justified. Nowadays, it is more effective for the modern chemical engineer to rely on his/her own programming ability for problem solving. In the present paper, the abilities of Polymath, Maple, Matlab, Mathcad, and Mathematica in handling differential equations are illustrated for differential-algebraic equations, large system of nonlinear differential equations, and partial differential equations. The programming of solutions with these CAS are presented, contrasted, and discussed in relation to chemical engineering problems.

Keywords: Computer algebra systems (CAS), computer simulation, Mathcad, Mathematica, Mathlab, *and* numerical methods.

INTRODUCTION

"In the beginning, there was Fortran." The introduction of the Fortran language by IBM as the first high-level programming language in 1954 has economized the solution of chemical engineering problems previously impossible. Following Fortran, C (1972), Visual Basic (1991), Java (1995), and gPROMS (1997), among others, were introduced to chemical engineers as general modeling tools. Fortress, a successor to Fortran which supports mathematical notations, is currently in research phase at Sun Microsystems.

The 80s saw the proliferation of programming tools dedicated to mathematics, known as *computer algebra systems* (CAS). As a hybrid tool that is capable of both symbolic and numerical computations, CAS is an ideal alternative for solving engineering problems, which are mathematical in essence. The earlier languages, although powerful, are less user-friendly and can be difficult for beginners, since they normally require the user to write a program from the first principle, which is time-consuming. These disadvantages are minimized in the case of CAS, since the access to a multitude of advanced numerical algorithms and built-in mathematical functions is relatively easier as these capabilities can be interactively invoked in CAS. It is well known that usually the chemical engineer must

rely on his own mathematical and programming ability for the solution of the physical problem, applied mathematicians and computer scientists are not trained to comprehend the fundamental principles and laws involved. Therefore, assistance from this specialized group for solving medium- to even large-scale problems is unnecessary if CAS can be skillfully used.

Given the number of packages available, the selection of CAS can be difficult and subject to personal taste. Steinhaus (1997) tabulated the functionality of nine CAS, mainly from the perspective of mathematics and data analysis, and concluded that Gauss excels in mathematical functionality and Matlab excels in graphical and programming environments. Shacham et al. (1998) compared four CAS for solving nonlinear algebraic equations and recommended Mathematica/ Polymath for their performance and Polymath for its user-friendly features. Cutlip and Shacham (2003), as their latest attempt to demonstrate the usability of mathematical packages in chemical engineering courses, compiled 24 representative problems, with the solutions separately programmed and available from different authors using Excel, Matlab, and Polymath. This compilation serves an excellent source for beginners in learning CAS for use in chemical engineering.

As the CAS are constantly being updated, we intend to provide an up-to-date qualitative evaluation of the leading packages in solving differential equations, where they are contrasted in terms of user-friendliness and performance according to a set of objective criteria. For measuring user-friendliness, we considered the method of equation input and manipulation of output, together with the criteria stated by Shacham (1998), such as online debugging and error detection, in the programming environment. For measuring performance, we considered the ability of in-built algorithms to solve the mathematical statements as they are formulated. Solution is not available from a particular CAS if additional module or further algebraic manipulations on the equations are required, i.e. not directly solvable. The packages considered in the present paper are Polymath 6, Maple 9, Matlab 7, Mathcad 12, and Mathematica 5. The problems considered are selected to represent the simplest variants of differential equations from familiar chemical engineering situations.

DESCRIPTION OF THE PROBLEMS

Differential-algebraic equations

Vessel with fluid flow is a system commonly considered in introductory process modeling. For conical vessel geometry (Luyben 1990), the model consists of the following equations:

on mass holdup: $m = \frac{\pi}{3}\rho r^2 h$

vessel geometry: $\frac{r}{h} = \frac{R}{H}$

and mass conservation: $\frac{dm}{dt} = \rho(F - kh^{1/2})$

These equations form a system of nonlinear differential-algebraic equations.

Large system of differential equations

Liu and Amundson (1962) considered an isothermal batch addition polymerization for calculation of molecular weight distribution, in which the conservation equations on monomer M_1 and the live polymers P_r are

$$\frac{dM_1}{dt} = k_i M_1 - (k_p + k_t) M_1 \sum_{r=1}^{DP} P_r$$

and

$$\frac{dP_r}{dt} = \begin{cases} k_i M_1 - (k_p + k_t) M_1 P_r, & r=1 \\ k_p M_1 P_{r-1} - (k_p + k_t) M_1 P_r, & r>1 \end{cases}$$

In general, the number of differential equations to be solved is tremendous when the degree of polymerization, *DP*, is large.

Partial differential equation

Partial differential equations governing a chemical engineering system usually defy analytical treatment except in cases of highly idealized situations. For instance, a general energy balance

$$\frac{\partial u}{\partial t} + \mathbf{v} \cdot \nabla u = \alpha \nabla^2 u + \lambda q$$

applied to unidimensional object with conduction-only heat transfer give rise to

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2},$$

which is a parabolic equation where closed-form solution is possible.

SOLUTION OF THE DIFFERENTIAL EQUATIONS

The programs for solving differential equations are given, as they appeared in the programming environment in the CAS, in Table 1. For the first problem, it is interesting to note that the set of equations formulated is solvable only when the algebraic equations are modified to explicit form. Even though algorithm for solving differential equations with implicit algebraic equations is available in Maple, Mathcad, and Mathematica, they failed to solve the unmodified mathematical statements; this poses a problem when the implicit equations involved cannot be transformed to explicit statements.

For the second problem, solution is available from all packages except Polymath and Mathcad, the reason is that Polymath restrict to number of simultaneous equations to 300. Even when this limitation is relaxed, program control structure necessary for effective solution, such as if- and for-statements, is not found in Polymath. For Mathcad, the solution is only possible if the equations are separately entered, which is laborious and not feasible. As for the last problem, algorithms for solving partial differential equations are available in Maple, Matlab, Mathcad, and Mathematica. The simple heat equation can be sufficiently handled by these packages.

DISCUSSION

Interfacing with the Program

The input of the mathematical equations into the CAS is most easily achieved in Polymath, as it provides an interactive window for this purpose. Another aspect to consider is the support of mathematical notations and symbols, e.g. *p*, subscripts, etc. This is available in Maple, Mathcad, and Mathematica. In Polymath and Matlab, only alphanumerical symbols are supported. However, it is also clear from Table 1, that the equations in Mathcad bear the greatest resemblance to natural mathematical notations, which enhance the readability of the program.

In the programming environment, the CAS can be divided into command-based and semicommand-based environment, the later being more user-friendly. All of the CAS except Matlab can be considered as belonging to the latter class. Furthermore, Polymath and Mathcad support online debugging and error detection, while in other CAS, the errors or bugs in the programs are printed only after the execution of these program. In Maple, Mathcad, and Mathematica, word-processing, numerical-processing and graphing capabilities are available in one window. The user can perform calculations, graphing, text editing in one document with active contents in it.

Complexity of the codes and syntaxes

The complexity of the codes and syntaxes is another useful measure in user-friendliness. As different solvers will require a different syntax, it is desirable if all types of differential equations can be handled by one solver. In this aspect, Mathematica's NDSolve is favored over other solvers, as it can be used to handle initial-boundary-value problems, differential-algebraic equations, and partial differential equations. NDSolve is followed closed by dsolve/pdsolve in Maple and odesolve/pdesolve in Mathcad. On the other hand, in Matlab, the solvers odeXX, dde, bvp4c, pdepe are required according to type of equations involved. Furthermore, as illustrated in Table 1, the programming effort required in Matlab is generally more than other CAS. Solving even the simplest type of partial differential equation is not so straight-forward in Matlab as compared to Maple, Mathcad, and Mathematica. The need to transform the original equation to conform to the required format in Matlab, which is:

$$c(x,t,u,\tfrac{\partial u}{\partial x})\frac{\partial u}{\partial t} = x^{-m}\frac{\partial}{\partial x}\left(x^m f(x,t,u,\tfrac{\partial u}{\partial x})\right) + s(x,t,u,\tfrac{\partial u}{\partial x})$$

Table 1. Solvability of the Problems in CAS

	Problem 1	Problem 2	Problem 3
(a) Polymath	h = (m/(3.14159 / 3 * rho * (R / H) ^ 2)) ^ (1 / 3) d(m)/d(t) = rho * (F – k * h ^ 0.5) t(0) = 0 t(f) = 500 m(0) = 0	number of differential equations exceeded supported limit AND the hundreds of differential equations are to be defined separately	additional user intervention required, e.g. spatial discretization

(b) Maple	Parameters := {R=1, H=2, rho =1, F=1, k=0.5}:
	OdeM := diff(m(t),t) = rho*(F-k*(m(t)/(Pi/3)/(R/H)^2)^(0.5/3)):
	Dfeq := subs(Parameters, OdeM):
	Sol:= dsolve({dfeq,m(0)=0}, M(t), numeric, method=gear);
		Parameters := {ki=1e-4,kp=10,kt=5e-3}:
		OdeP[1] := diff(P[1](t),t)
		= ki * M1(t) – (kp +kt) * M1(t) * P[1](t):
		OdeM1 := diff(M1(t),t)
		= -ki * M1(t) – (kp + kt) * M1(t) * sum(P[r](t),r=1..DP):
		for r from 2 to DP do
		OdeP[i] := diff(P[i](t),t) = kp * M1(t) * P[i-1](t) – (kp + kt) * M1(t) * P[i](t)
		od:
		Diffeqns := subs(Parameters,{OdeM1,seq(OdeP[i],i=1..DP)}):
		ic := {M1(0)=1,op({seq(P[r](0) = 0, r=1..DP)})}:
		sol:=dsolve({op(Diffeqns),op(ic)}, {M1(t), seq(P[r](t),r=1..DP)}, type=numeric, method=rkf45);
			sol:=pdsolve(
			diff(u(x,t),t)=16*diff(u(x,t),x,x),
			{u(x,0)=piecewise(x<2,0,x<8,15,x>8,0), D[1](u)(0,t)=0, D[1](u)(10,t)=0}
			):
			sol:-plot3d(t=0..1,x=0..10)

(c) Matlab	function problemone
	T = 500;
	m0 = 0;
	sol = ode45(@flow, [0 T], m0);
	function dx = flow(t,m)
	R = 1; H = 2; rho = 1;
	F = 1; k = 0.5
	h = (m / (pi/3) / (R/H)^2)^(1/3);
	dm = rho * (F – k * h^(1/2));
		function problemtwo
		DP = 500;
		x0 = zeros(DP+1,1);
		sol = ode45(@poly, [0 T], x0)
		function dx = poly(t,x)
		DP = 500;
		M1 = x(DP+1);
		P = sum(x) – M1;
		dx(1) = ki * M1 – (kp+kt) * M1 * x(1);
		dx(DP+1) = -ki * M1 – (kp + kt) * M1 * P
		for r = 2:DP
		Dx(r) = kp*x(r-1)*M1 – (kp+kt)*M1*x(r)
		end
		dx = dx';
			function problemthree
			x = linspace(0,X,100);
			t = linspace(0,T,100);
			sol = pdepe(0,@pde,@ic,@bc,x,t);
			surf(x,t,sol(:,:,1));
			function [c,f,s] = pde(x,t,u,DuDx)
			c = alpha; f = DuDx; s = 0;
			function u0 = ic(x)
			if x < 2 u0 = 0;
			else if x < 8 u0 = 15;
			else u0 = 0; end end
			function [pl, ql, pr, qr] = bc(xl,ul,xr,ur,t)
			pl = 0; ql = 1; pr = 0; qr = 1;

(d) Mathcad	Given R := 1 H := 2 ρ := 1 F := 1 k := 0.5 T := 500 m(0) = 0 $h_{cubed}(0) = 0$ $m(t) = (\pi/3)(R/H)^2 h_{cubed}(t)$ $m'(t) = \rho(F - k\, h_{cubed}(t)^{0.5/3})$ $\begin{pmatrix} m \\ h_{cubed} \end{pmatrix} :=$ $odesolve\left(\begin{pmatrix} m \\ h_{cubed} \end{pmatrix}, t, T\right)$	the hundreds of differential equations are to be defined separately	Given α := 16 X := 10 T := 1 $u_t(x,t) = \alpha\, u_{xx}(x,t)$ u(x,0) = 15 if 2 < x < 8 0 otherwise u(0,t) = 0 u(X,t) = 0 $u := pdesolve(u, x, \begin{pmatrix} 0 \\ X \end{pmatrix}, t, \begin{pmatrix} 0 \\ T \end{pmatrix})$ u_{mesh} := CreateMesh(u,0,X,0,T)

(e) Mathematica	R = 1; H = 2; r = 1; F = 1;
	k = 0.5; T = 500;
	algeq = h[t] -> (m[t]/(π/3)(R/H)^2)^1/3;
	sol = NDSolve[{ m'[t]== r(F-k h[t]^1/2)/.algeq, m[0]==0},m, {t,0,T}];
		k_i=0.0001; k_p = 10; k_t = 0.005; DP = 500;
		sol = NDSolve[{
		Join[$M_1'[t] == -k_i M_1[t] - (k_p + k_t) M_1[t] \sum_{r=1}^{DP} P_r[t]$,
		$P_1'[t] == k_i M_1[t] - (k_p + k_t) M_1[t] P_1[t]$,
		Table[
		$P_r'[t] == k_p P_{r-1}[t] M_1[t] - (k_p + k_t) M_1[t] P_r[t]$,
		{r,2,DP}],
		$M_1[0]$==0, Table[$P_r[0]$ == 0, {r, 1, DP}],
		Join[{M_1},Table[P_r,{r,1,DP}]], {t,0,500}];
			sol = NDSolve[{
			∂_t u[x,t] == α $\partial_{x,x}$ u[x,t],
			u[x,0]== If[x<2,0,If[x<8,15,0]],
			Derivative[1,0][u][0,t] == 0,
			Derivative[1,0][u][X,t] == 0},
			U, {x,0,X}, {t,0,T}];
			u_{mesh} = Evaluate[u[x,t]/.sol[[1]]];
			Plot3D[u_{mesh},{x,0,X},{t,0,T}];

is inconvenient, especially when large system is involved.

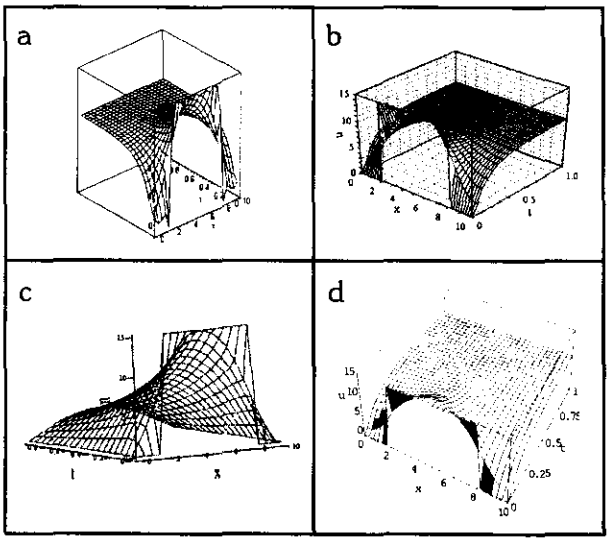

Figure 1. Plots of u(x, t) in (a) Maple, (b) Matlab, (c) Mathcad, (d) Mathematica

Evaluation and Manipulation of the Solution

All the software packages mentioned in this paper can be used to generate the graphical output of the solution. Maple, Matlab and Mathcad generate dynamic graphics that can be interactively manipulated, e.g., rotated, while Polymath and Mathematica produce static graphics. Interactive plot editing is available in all programs except Mathematica, which require command-line based control.

All of the CAS produce good-quality and professional-looking two- and three-dimensional graphics, as an illustration, the surface plot for $u(x,t)$ is shown below.

To evaluate the solution to differential equations at specific points, all of the CAS except Polymath support functional evaluation; therefore the need for interpolation does no arise. For instance, the calculation of mass holdup at $t = t0$ is achieved by deval(sol,t0), m[t]/.sol/.t->t0, m(t0) in Matlab, Mathematica, and Mathcad respectively. Additionally, in Maple, Mathematica and Mathcad, the solution can be treated as a functional entity that can be subjected to other mathematical operations, such as differentiation and equation solving.

CONCLUSIONS

As each of the CAS are programmed with different viewpoints and targeted at different niches, they are strong only in attacking certain problems. It is very desirable that a chemical engineer can choose the appropriate software in helping him to do numerical calculation, depending on the complexity and the result or output required.

Table 2. Comparative Features of Different CAS

Feature	Polymath	Maple	Matlab	Mathcad	Mathematica
Programming Environment	Semi Command-based	Semi Command-based	Command-based	Semi Command-based	Semi Command-based
Interactive Input of Diff. Equations	Yes	No	No	No	No
Mathematical Symbol Input	No	Yes	No	Yes	Yes
Online Debugging & Error Detection	Yes	No	No	Yes	No
ODE Algorithm	Numerical	Symbolic & Numerical	Symbolic & Numerical	Symbolic & Numerical	Symbolic & Numerical
PDE Algorithm	No	Symbolic & Numerical	Numerical & Limited	Numerical & Limited	Numerical & Limited
Graphical Output & Editing	Static & Interactive	Dynamic & Interactive	Dynamic & Interactive	Dynamic & Interactive	Static & Non-interactive
Specific Evaluation of Output	No	Yes	Yes	Yes	Yes

In terms of user-friendliness, Polymath and Mathcad are recommended; however, their power in handling medium- to large-scale problems is often sacrificed. In contrast, the performances and computing powers of Maple, Matlab, and Mathematica are comparable to each other. However, all the packages failed to directly solve the simple differential algebraic equations in their original form.

REFERENCES

Cutlip, M. B., and Shacham, M. (2003). "Integration of numerical problem solving into the Chemical Engineering Curriculum," Proceedings of the American Society for Engineering Education Annual Conference and Exposition, Session 2793.

Liu, S. L., and Amundson, N. R. (1962). "Calculation of molecular weight distribution in polymerization," *Chem. Eng. Sci. 17,* 797–802.

Luyben, W. L. (1990). *Process modeling, Simulation and control for chemical engineers,* 2nd ed., McGraw-Hill, New York.

Shacham, M., Brauner, N., and Pozin, M.(1998). "Comparing software for interactive solution of systems of nonlinear algebraic equations," *Comp. Chem. Eng., 22,* 323–31.

Steinhaus, S. (1997). *Comparison of mathematical programs for data analysis.* Available at: http://www.uni-frankfurt.de/~stst/ncrunch.html (July 9, 2005).