Enhancement of Heat Transfer Teaching and Learning using MATLAB as a Computing Tool

Paper # 173 **(this paper has not been reviewed for technical content)**

SUZANA YUSUP and NOORYUSMIZA YUSOFF*
Department of Chemical Engineering
Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh, Perak, Malaysia

ABSTRACT

An important aspect of chemical engineering curriculum is to teach multidimensional transient conduction, convection and radiation processes of heat transfer. To deliver better illustration of these concepts, which involve partial differential equations (PDE's), a computer simulation tool is required. This tool acts as an additional teaching aid for visualizing the complex transport processes, and therefore is helpful in facilitating and enhancing learning development. The objective of this paper is to illustrate how programming, modelling and simulation of a transient PDE problem using a mathematical software package, i.e., MATLAB, can expose the students to conceptual and practical aspects of heat transfer. A case study was selected whereby the system is modelled by applying heat balance across a cylindrical tube wall and the resulting parabolic PDE is solved via explicit finite difference method. Two MATLAB programs were developed to calculate and display the results for 2D transient temperature profiles inside the tube wall. In addition, the case study is also simulated using PDE Toolbox (`pdetool`). The visualization of temperatures profiles across the cylindrical tube wall was possible using both approaches.

Key word: MATLAB, Heat Conduction, Partial Differential Equation

INTRODUCTION

Programming, modelling and simulation are useful to illustrate the solutions of complex engineering problems. Among the available commercial packages to achieve the objective are Aspen+, HYSYS, CFD and MATLAB. J.L. Sinclair (1998) believed that incorporating computational software such as CFD package into teaching of undergraduate transport courses is an effective way to expose students to conceptual and practical examples of fluid-particle flow. This way, the students can better visualize the flow behaviour, and explore the effect of changes in system geometry, properties and operating conditions. Other researchers like D.J. Brown et. al (1997) also applied CFD to simulate the temperature profiles inside furnace tube wall.

The importance of computational techniques in many areas of chemical engineering is indubitable. It acts as an educational tool in design, modelling and simulation of chemical processes. H.O. Kassim and R.G. Cadbury (1996) discussed the role of computer in chemical engineering education. One of the ways to increase the students' interests in programming is by integrating computational modelling with underlying key elements of the theory, on which the model is based. They believed that programming approach could inculcate a logical and disciplined approach to design problems.

Furthermore, it is a common to acknowledge that real-life processes often deviate from ideality. Awareness and ability to apply programming skills to solve transient heat transfer processes would be of great advantage for the undergraduate students to tackle the real-life complexity when they later work as engineers. With this knowledge, it is hope that these future graduates will not only be the users but also creators of technology.

Hence this paper tries to illustrate the linkage between theoretical concepts of multidimensional transient heat conduction and computer simulation using MATLAB. The simulation tool acts as an additional teaching aid for better illustration of the complex heat transfer process. This approach was introduced as one of the topics in the Final Year Project course whereby the students were given the opportunity to integrate their theoretical knowledge in heat transfer, numerical method and computing skills gained *a priori*. A case study of predicting temperature profiles inside a cylindrical tube wall is selected. The system is modelled by applying heat balance across the tube wall. The resulting parabolic partial differential equation (PDE) is solved using finite difference method and PDE Toolbox (`pdetool`).

MATHEMATICAL FORMULATION

The estimation of temperature profiles inside a cylindrical tube wall is performed by developing a numerical programming using MATLAB and applying its PDE Toolbox (`pdetool`) Graphical User Interface (GUI). The system in concern is a cylindrical tube wall with differential thickness Δr and length Δz. The wall is exposed to uniform thermal radiation at the outside. Inside the tube, a non-compressible fluid at certain temperature is flowing steadily and hence energy is convected from the tube wall to the process fluid flowing inside it. In this case, it is widely known that the temperature profiles inside the solid wall vary in a temporal sense and two spatial dimensions (radial r and longitudinal / axial z). Figure 1 shows a diagram of the cylindrical tube and its differential volume element (DVE) with the applicable fluxes entering and leaving the system.

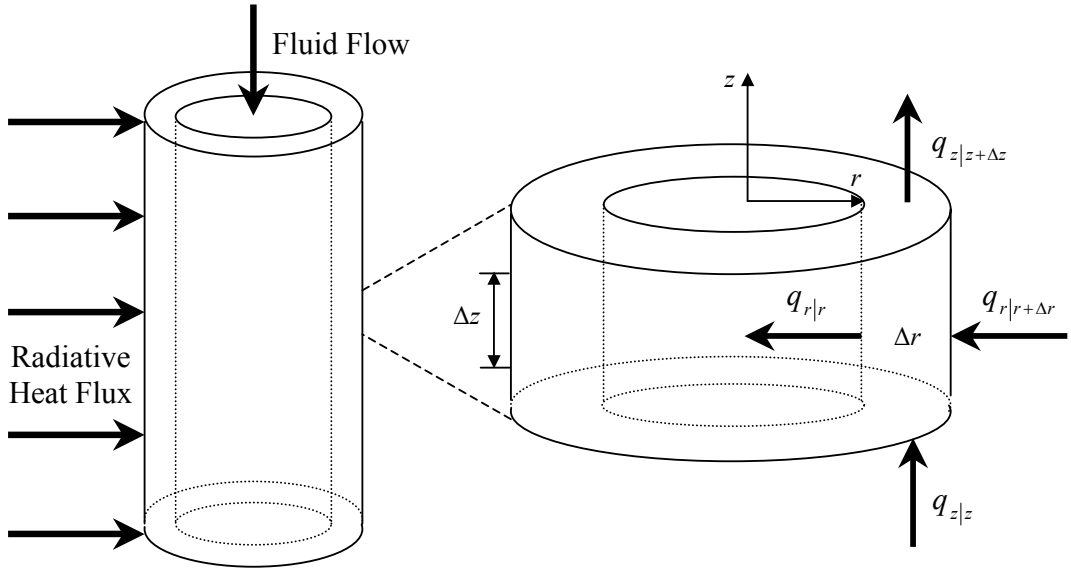


Figure 1: Cylindrical tube with its differential volume element (DVE).

In this work, the thermal parameters (heat conductivity k; thermal capacity C) and the tube's density ρ are assumed homogeneous and isotropic resulting in a constant value of the thermal diffusivity $\alpha = k/\rho C$. The resulting parabolic PDE of the temperature T with respect to r, z, and t is given in Equation (1).

$$\frac{\partial T}{\partial t} = \alpha\left[\frac{\partial^2 T}{\partial r^2} + \frac{1}{r}\frac{\partial T}{\partial r} + \frac{\partial^2 T}{\partial z^2}\right] \tag{1}$$

where:

T = temperature (^{o}C), t = time (s), α = thermal diffusivity (m^2/s), r = radius (m), z = longitudinal / axial length (m).

The initial condition of the cylindrical tube wall is 10 ^{o}C and boundary conditions are prescribed in Tables 1 and 2:

TABLE 1: Boundary conditions of the concentric cylinder (top view)

Boundary	**Type**	**Conditions**	
		Lower-z	**Upper-z**
Outer	Uniform radiative heat flux (Neumann)	$q_o = 2000$	$q_o = 2000$
Inner	Convective heat flux (Neumann)	$-k\frac{\partial T}{\partial r} = h(T-10)$	$-k\frac{\partial T}{\partial r} = h(T-60)$

TABLE 2: Boundary conditions of the solid half of the cylinder (side view)

Boundary	**Type**	**Condition**
Left	Inlet temperature (Dirichlet)	T = 10 ^{o}C
Right	Outlet temperature (Dirichlet)	T = 60 ^{o}C
Top	Uniform radiative heat flux (Neumann)	$q_o = 2000$ W/m^2
Bottom	Convective heat flux (Neumann)	$-k\frac{\partial T}{\partial r} = h(T-10)$

METHODOLOGIES

The transient temperature profiles inside the tube wall are initially solved using explicit finite difference method. MATLAB programming capability is utilised to obtain the insights of the problem. Then the `pdetool` GUI is applied to enhance appreciation of solving the parabolic PDE in a graphical manner.

Finite Difference Method

The main advantage of applying this method is that the differential Equation (1) can be expressed algebraically in terms of finite differences as shown in Equation (2). The first-order and second-order derivative terms are approximated in terms of central differences around the point (i, j, n), using the counter i for the r-direction, j for the z-direction and n for t-dimension (Chapra and Canale, 1998).

$$T_{i,j,n+1} = \left(\frac{\alpha\Delta t}{\Delta r^2}\right)\left(T_{i+1,j,n} + T_{i-1,j,n}\right) + \left(\frac{\alpha\Delta t}{2r\Delta r}\right)\left(T_{i+1,j,n} - T_{i-1,j,n}\right) + \left(\frac{\alpha\Delta t}{\Delta z^2}\right)\left(T_{i,j+1,n} + T_{i,j-1,n}\right) + \left(1 - \frac{2\alpha\Delta t}{\Delta r^2} - \frac{2\alpha\Delta t}{\Delta z^2}\right)\left(T_{i,j,n}\right) \tag{2}$$

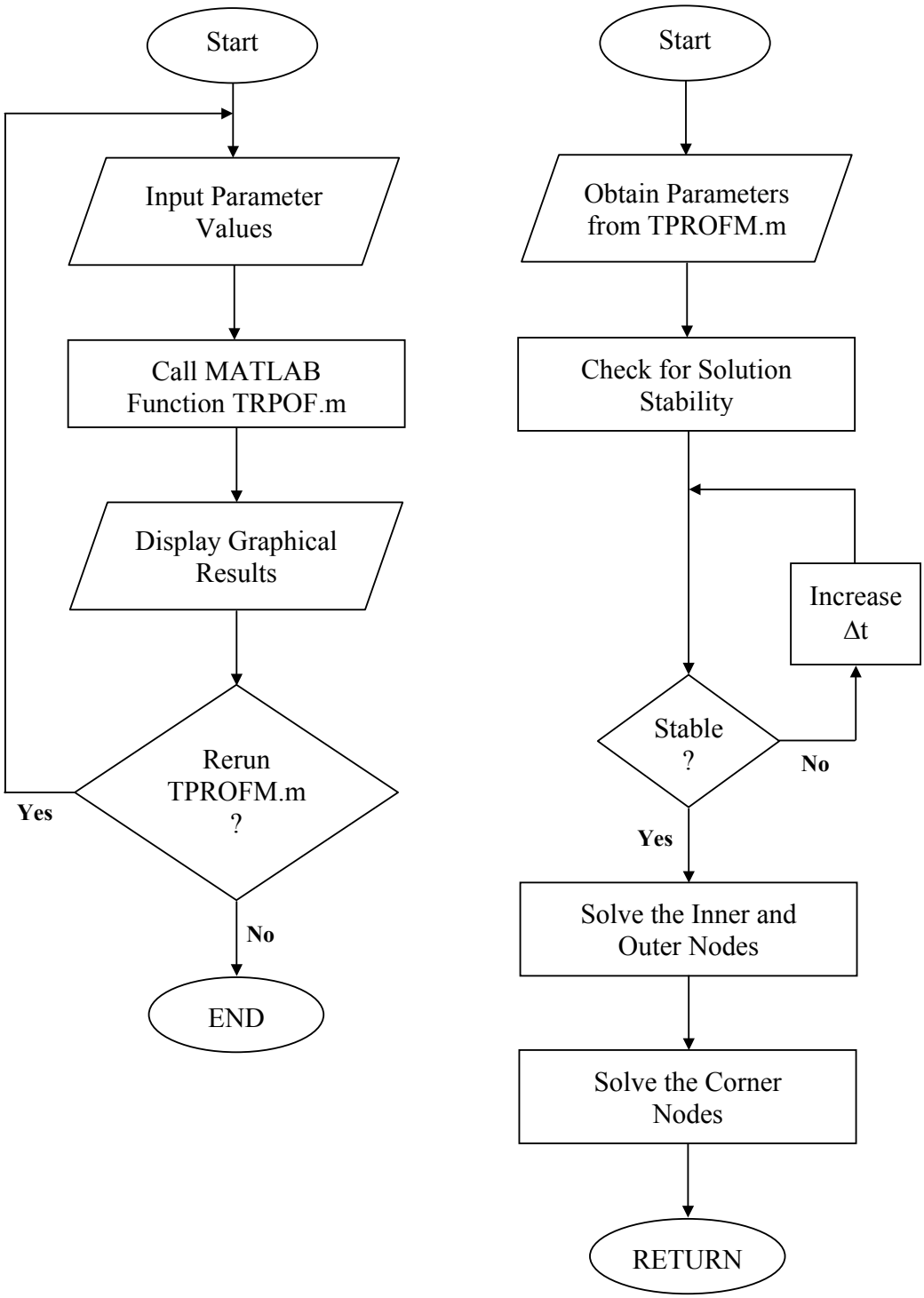
with the stability condition given below,

$$\frac{\alpha\Delta t}{\Delta r^2 + \Delta z^2} \leq \frac{1}{8} \tag{3}$$

Two MATLAB programs were developed to calculate and display the result for two-dimensional (2D) transient temperature profiles inside the tube wall. The first is the main program MTPROF.m, which is developed to prompt the user to insert relevant parameters according to a specific tube. More specifically, the user needs to insert the type of tube material, the tube dimensions (inner radius, thickness and length) and the step sizes (Δr, Δz, Δt) in all directions. Other required information is the total radiative heat flux, fluid convective heat transfer coefficient, and the inlet and outlet temperatures of the flowing fluid. The flow diagram showing basic structure of the main program is given in FIGURE 2.a**.**

Upon acquiring all the required information, the program will call MATLAB function program TPROF.m to solve for the temperature variation inside the tube wall. The temperature values for all nodes inside the system are iteratively calculated using finite difference equations formulated in Equation (2). Finally, the results are displayed graphically in terms of 3D transient temperature profiles and also the temperature variation at maximum time from the top view. The flow diagram for this function program is given in FIGURE 2.b.

In another approach using the `pdetool` GUI, similar mathematical formulation and boundary conditions are applied. This approach is perhaps easier to grasp and can be taught as part of the heat transfer curriculum. However, care should be taken to balance the skill of solving parabolic PDE numerically with that of using this 'black-box'. The next section discusses the corresponding `pdetool` GUI notations used in the simulation.



(a) MTPROF.m (b) TPROF.m

Figure 2: The flow diagrams of (a) the main program MTPROF.m, and (b) the temperature profile function TPROF.m.

MATLAB's Partial Differential Equation Toolbox (`pdetool`)

One of the important advantages of using MATLAB in engineering applications is the availability of built-in functions called toolboxes. These toolboxes range from Control System Toolbox to Wavelet Toolbox comprising not only subroutines but also graphical user interface (GUI) that are useful to simplify or even expedite computations. In this work, the `pdetool` GUI is applied in solving transient heat transfer problem and the subsequent discussion highlights its salient features.

The `pdetool` GUI can be started by typing `pdetool` at MATLAB's command prompt as follows:

```
>> pdetool
```

A new window (FIGURE 3.a) will then pop out and we can start by first saving our case through **File-Save As**. The next step is to model the PDE problem in the *Generic Scalar* mode. Note that the current MATLAB PDE Toolbox (Version 1.0.4) cannot handle 3D graphics. As a result, the transient heat transfer problem in this work is modelled in the 2D spatial domain by separately simulating the top and side views of the cylinder. To assist drawing, the axes are rescaled by selecting **Options-Axis Limits** (FIGURE 3.b) and entering [-5.25 5.25] and [-3.5 3.5] for the x- and y-axis ranges, respectively. In doing so, the *Auto* rescaling feature will be automatically unchecked. Then open the Grid Spacing dialog box (FIGURE 3.c) from the **Options** menu to ensure the suitability of the *x*- and *y-axis linear spacing*. If not, uncheck the *Auto* button and change the *axis linear spacing* accordingly, as performed in this work for the side view. For convenience, turn on the grid and apply snap-to-grid feature by respectively selecting **Options-Grid** and **Options-Snap**.

The top view (FIGURE 3.d) is modelled as two circles with the respective inner and outer radii R_i = 2 m and R_o = 3 m. The cross-sectional area of the smaller circle is subtracted from the larger one to obtain the solid domain (shaded area), representing the top view of the cylinder with a thickness of 1 m. In `pdetool`, this procedure is performed by setting C1 – C2 (the larger minus the smaller circles) in the formula box. In another case, the solid half of the cylinder (side view; FIGURE 6) is modelled as a rectangular slab having a width W of 1 m and a length L of 5 m. The procedure to draw the rectangle is similar to that explained above.

Now leave the draw mode and enter the boundary mode by pressing the ∂Ω button or selecting **Boundary-Boundary Mode**. The boundary conditions may be entered by double-clicking each boundary and entering parameter values in the Boundary Condition dialog box as illustrated in FIGURE 3.e. In this work, similar inner and outer boundaries are selected entirely by pressing the *Shift* button and choosing **Boundary-Specify Boundary Conditions** for each boundary. Next, open the PDE Specification dialog box (FIGURE 3.f) by pressing PDE button or selecting **PDE-PDE Specification** and enter the coefficients. The general parabolic PDE that `pdetool` solves is:

$$d\frac{\partial u}{\partial t} - \nabla \cdot (c\nabla u) + au = f \tag{4}$$

with the initial value $u_o = u(t_o)$; in this case, u denotes the temperature T and the coefficients d, c, a and f are explained below.

Prior to solving the PDE problem using `pdetool`, the governing Equation (1) and all boundary conditions (TABLES 1 and 2) must be transformed into the typical `pdetool` convention. The radial (r) and longitudinal (z) directions of the cylindrical coordinate are represented by y- and x-axes, respectively in the Cartesian coordinate. In essence, the governing equation and all Neumann boundary conditions are multiplied by the coordinate r or in this case, coordinate y as summarized in TABLE 3. It is worth to note that the parameter r used in the `pdetool` differs from the coordinate r mentioned above. Other `pdetool` parameters (a, c, d, f, g, h, q, r) should also be distinguished accordingly. The normal vector n (`pdetool` dialog box) or $\vec{n}$ (mathematical notation) is included to show its significance in determining the direction of the heat flow. The initial value of 10 ^{o}C and the time array of [0:100:3600] can be entered in the Solve Parameters dialog box (FIGURE 3.g), which is accessed by selecting **Solve-Parameters**. In this work, the maximum time limit is varied from 3600 sec (1 hr) to 18000 sec (5 hr) to obtain 5 frames of maximum temperature profiles.

TABLE 3: Governing equation and boundary conditions in the transformed coordinate.

a. *Governing Equation*

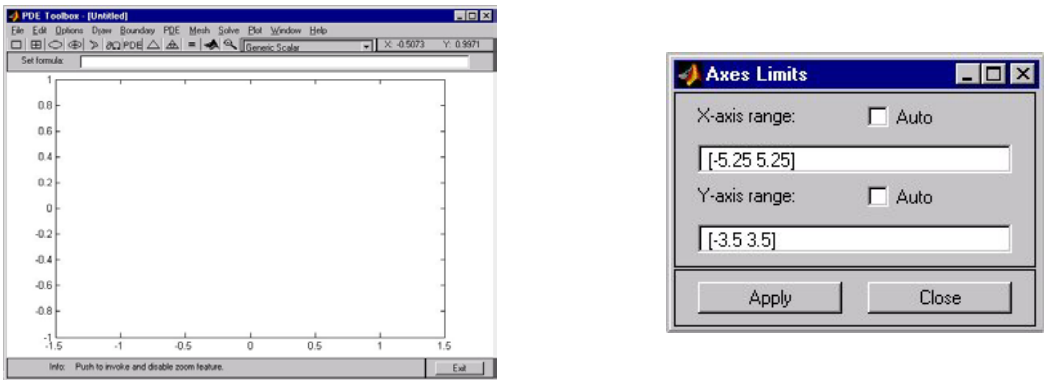
	Convention	**Description**
	`pdetool`	`d*u' - div(c*grad(u)) + a*u = f` where `d`=7830*434*y, `c`=64*y, `a`=0, `f`=0
	Mathematical Notation	$\rho Cy\frac{\partial T}{\partial t}-\frac{\partial}{\partial x}\left(ky\frac{\partial T}{\partial x}\right)-\frac{\partial}{\partial y}\left(ky\frac{\partial T}{\partial y}\right)+aT=fy$

b. *Top View Concentric Cylinder's Boundary Conditions*

Boundary Convention			**Description**
Outer	Lower and Upper-z	`pdetool`	`n*c*grad(u) + q*u = g;` where `q`=0, `g`=2000*y
		Mathematical Notation	$\vec{n}\cdot ky\frac{\partial T}{\partial y}+qyT=gy$
Inner	Lower-z	`pdetool`	`n*c*grad(u) + q*u = g;` where `q`=12*y, `g`=12*y*10
		Mathematical Notation	$\vec{n}\cdot ky\frac{\partial T}{\partial y}+qyT=gy$
	Upper-z	`pdetool`	`n*c*grad(u) + q*u = g;` where `q`=12*y, `g`=12*y*60
		Mathematical Notation	$\vec{n}\cdot ky\frac{\partial T}{\partial y}+qyT=gy$

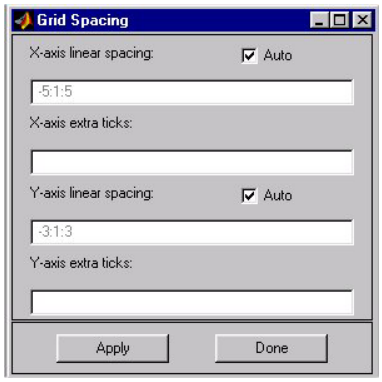
c. *Side View of the Solid Half of the Cylinder's Boundary Conditions*			
Boundary Convention			**Description**
	Left and Right	`pdetool`	`h*u = r;` where `h`=1, `r`=10 (left) and `r`=60 (right)
		Mathematical Notation	$hT = r$
	Top	`pdetool`	`n*c*grad(u) + q*u = g;` where `q`=0, `g`=2000*y
		Mathematical Notation	$\rho_h \cdot ky \frac{\partial T}{\partial y} + qyT = gy$
	Bottom	`pdetool`	`n*c*grad(u) + q*u = g;` where `q`=12*y, `g`=12*y*35
		Mathematical Notation	$\rho_h \cdot ky \frac{\partial T}{\partial y} + qyT = gy$

Then draw the mesh by pressing the ∇ button or selecting **Mesh-Initilize Mesh** (FIGURE 3.h). The mesh can be further refined to increase the computation 'accuracy' by selecting **Mesh-Refine Mesh** menu. To solve the PDE problem, press the = button or select **Solve-Solve PDE**. MATLAB will interpolate the temperature distribution from 0 sec to 3600 sec with an increment of 100 sec, as specified above. By default, the 2D temperature profiles (may be in color, contour, etc.) displayed are those computed at the final time span (FIGURE 3.ij). A more dynamic profile can be visualized by animating the solution. To do this, check the *Animation* check box in the Plot Selection dialog box (FIGURE 3.j), select the *Colormap* 'Jet' and press *Plot* button. The recorded animation is then played in a separate figure window.

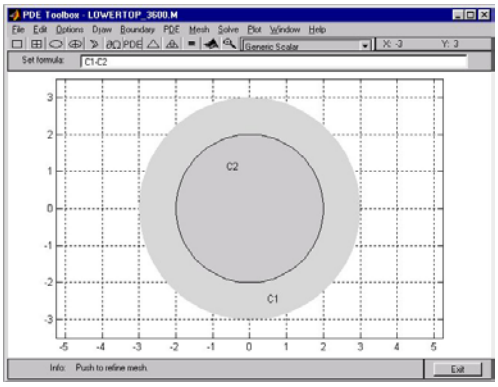


(a) Default `pdetool` Environment (b) Axis Limit

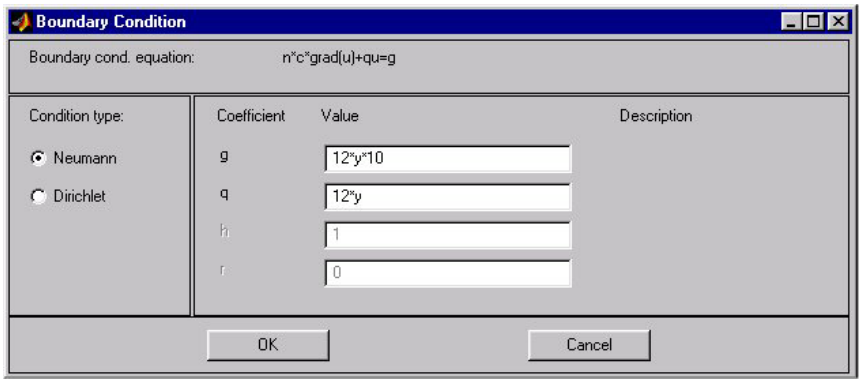
FIGURE 3: Using `pdetool` GUI to solve 2D temperature profiles of a cylindrical tube**.**



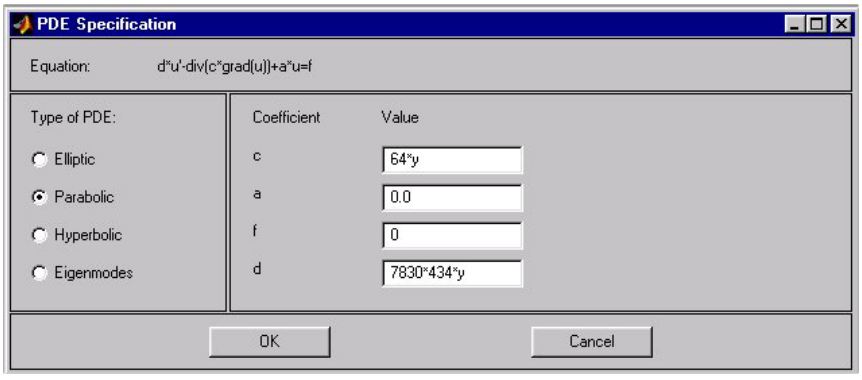
(c) Grid Spacing



(d) Draw Mode

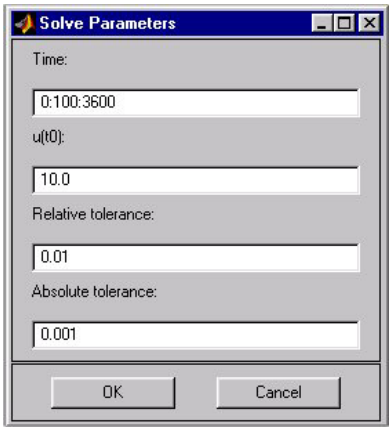


(e) Boundary Condition (inner)

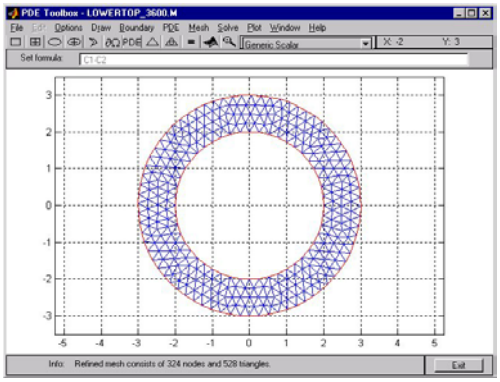


(f) PDE Specification

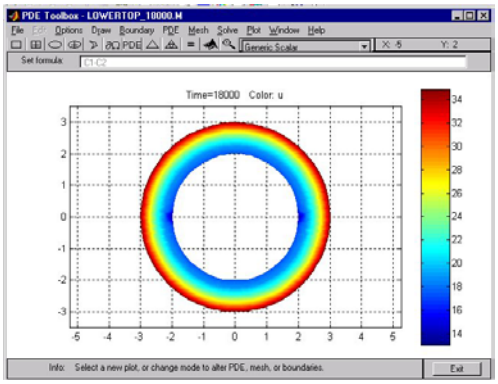
FIGURE 3: Using `pdetool` GUI to solve 2D temperature profiles of a cylindrical tube**.**



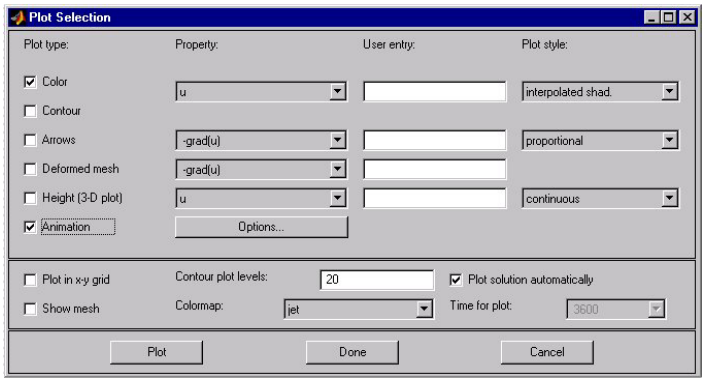
(g) Solve Parameters



(h) Mesh Initialisation



(i) Final Solution (t = 18000 sec)



(j) Plot Selection

FIGURE 3: Using `pdetool` GUI to solve 2D temperature profiles of a cylindrical tube**.**

RESULTS AND DISCUSSION

The graphical outputs of both finite difference programming and `pdetool` GUI approaches are presented in this section. In the former method, the resultant temperature profiles and graphical arrangement can be customised according to our preferences. The latter approach, however, has some limitations although it is easier to operate and more appreciated by average students.

Finite Difference Programming in MATLAB

As discussed earlier in the methodology section, the main program (MTPROF.m) will request several parameters as the inputs to the function program (TPROF.m). The selected parameter values used to simulate the case study are listed in Table 4, and the MATLAB command window asking for the information is subsequently depicted in Figure 4.

The simulation results obtained through MATLAB programming are shown in Figure 5, of which part (a) shows that the temperature profiles varying spatially (r, z) and temporally (t). As expected, the temperature profiles flattened at 0 hr showing the initial temperature of the solid wall. As the outer tube wall is exposed further to external radiation, its temperature increases spatially as indicated by the coloured SURF plot and temporally from 10 ^{o}C at the initial stage to 60 ^{o}C at the maximum duration of 10 hr. The temperature profiles depict a downward trend along the tube wall and reach a minimum value at the inner layer of the wall. This phenomenon occurs due to low inlet fluid temperature (10 ^{o}C) flowing in the inside tube and higher heat transfer into rather than out of the tube wall. To obtain a clearer picture, the maximum temperature profiles of the top view are displayed in Figure 5.b.

Table 4: Parameter values for MTPROF.m applied in this work.

Parameter	Value
Tube Material	1. Carbon Steel AISI 1010
Tube Inner Radius	2 m
Tube Thickness	1 m
Tube Length	5 m
Maximum Time	10 hr
Division in z	10
Division in t	30
Radiation Heat Flux	2000 W/m^2
Convective Heat Transfer Coefficient, h	12 W/(m^2· ^{o}C)
Process Fluid Inlet Temperature, T_i	10 ^{o}C
Process Fluid Outlet Temperature, T_o	60 ^{o}C

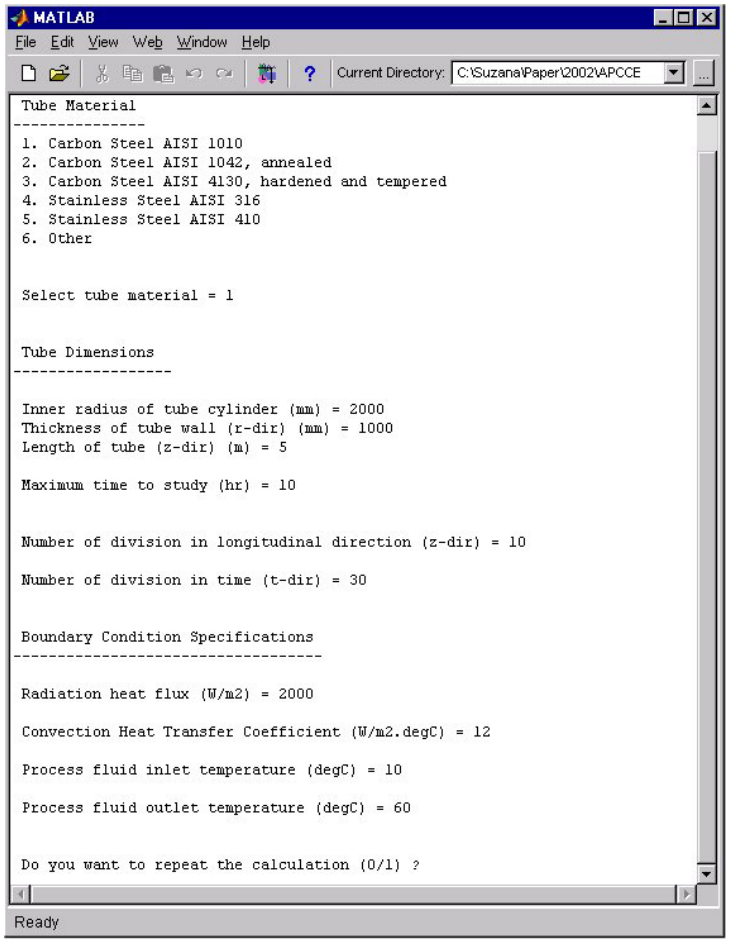
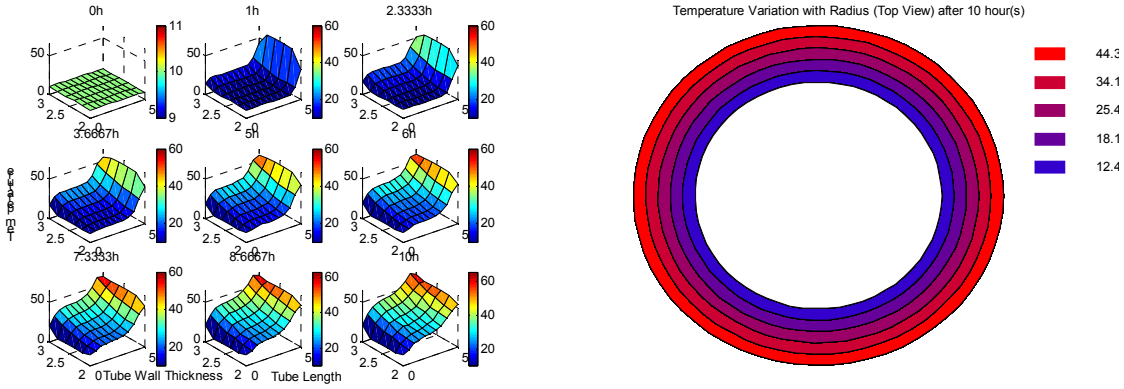


Figure 4: Sample input values from the main program, MTPROF.m.



(a) (b)

Figure 5: The transient temperature profiles of the: (a) 3D views with increasing time from 0 to 5 hours, and (b) top view at the maximum time.

Using `pdetool` GUI

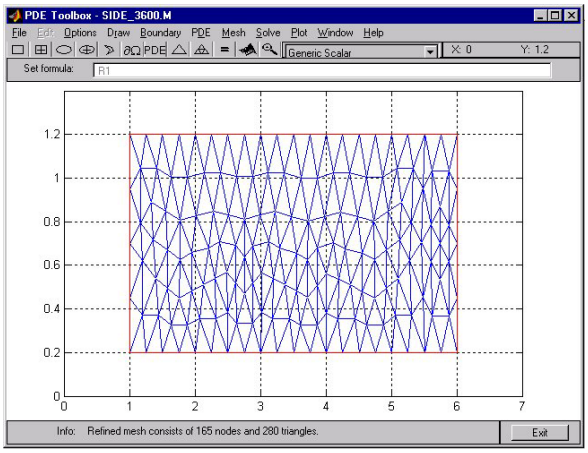
In many situations, students perceive computer programming as troublesome and tedious. This phenomenon lies on the fact that programming may be difficult to the novice. A missing comma can cause a major headache for the debugger. Hence it is useful to initially attract the students' interests by showing the simpler solutions to any engineering problems. The application of `pdetool` GUI is one of the many ways to accomplish this. In this case, PDE can be implicitly solved by plugging in a few parameters in the GUI as explained in the methodology section.

A set of `pdetool` GUI outputs is illustrated in Figure 6. It is worth to note that the graphical results of the `pdetool` GUI may vary depending on the mesh size. The finer the mesh, the smoother the temperature profiles will be but the longer it will take for MATLAB to obtain the solution. Figure 6.a depicts the mesh grid at the second level refinement, which was found optimum (with respect to the balance between accuracy and computing time) for the heat transfer problem posed in this work.

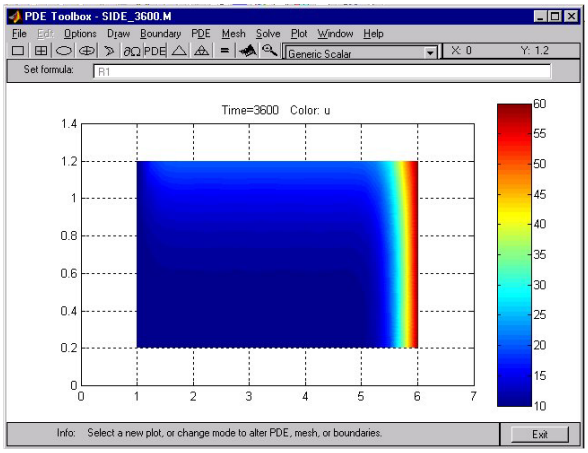
Figures 6.b-f show the resultant temperature profiles of the side view from 1 hr to 5 hr duration. Initially (0 hr), the system is at an initial temperature of 10 ^{o}C, which is the temperature of the incoming cold fluid. After an hour of constant radiation at the outer tube wall, the system temperature starts to gradually increase and shows curvature profiles. These temperature profiles vary according to the prescribed boundary condition specifications, of which the inlet and outlet temperatures are respectively set at 10 ^{o}C and 60 ^{o}C. The outer tube wall is exposed to the uniform radiative heat transfer and the inner side with the convective boundary condition. Consequently, the temperature profiles depict higher values (in terms of colour variation) at the top and right sides of the tube.

As the simulation time increases, indicating more radiative exposure to the outer tube wall, the higher temperature profiles move deeper diagonally to the left side of the wall. This phenomenon is clearly depicted by the change of colour from dark blue to the lighter blue and green at the major parts of the wall. At the end of 5 hr simulation, almost all sectors of the tube wall are heated to temperature greater than 20 ^{o}C. The targeted outlet stream temperature of 60 ^{o}C is achieved after the three-quarter end of the tube.

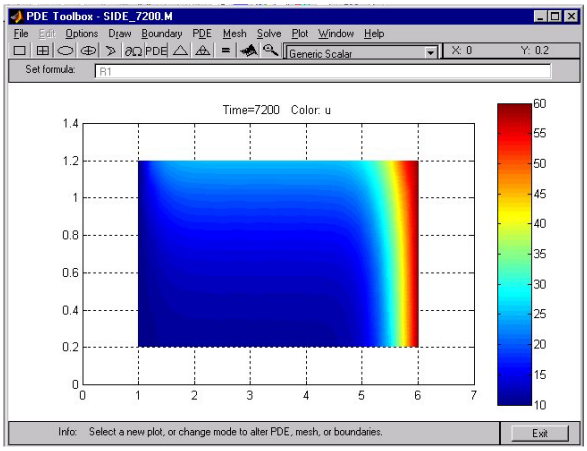
The usefulness of `pdetool` GUI in the enhancement of heat transfer teaching is indeed obvious. Average students that have difficulty in understanding PDE, let alone writing computer programs themselves, will appreciate the graphical outputs of the `pdetool` GUI. As they become more comfortable with the conceptual aspects, the students may be more enthusiastic about learning the mathematics and computer programming of PDE. The MATLAB programming capability, the next step after mastering the `pdetool` GUI, is powerful and suitable for the purpose of this learning process.



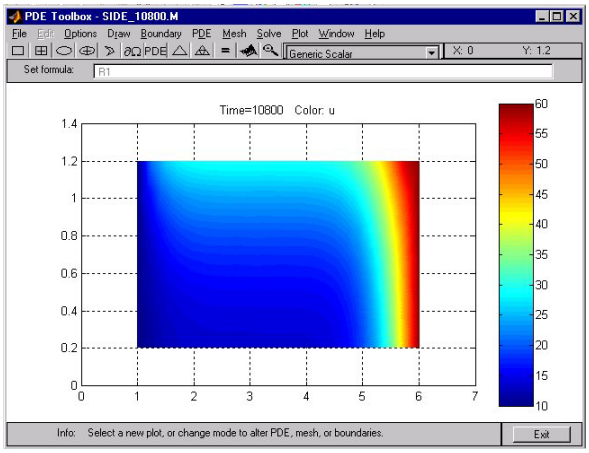
(a) Mesh grid



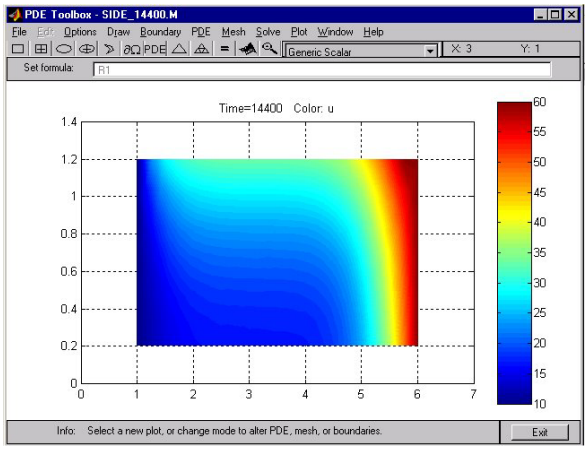
(b) t = 3600 sec (1 hr)



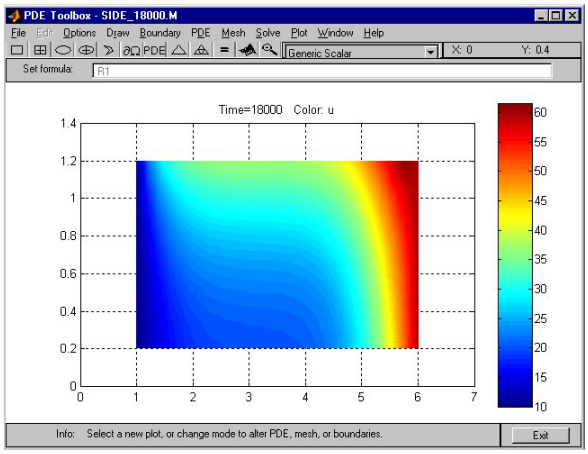
(c) t = 7200 sec (2 hr)



(d) t = 10800 sec (3 hr)



(e) t = 14400 sec (4 hr)



(f) t = 18000 sec (5 hr)

FIGURE 6: Side view of the temperature distribution (color bar in ^{o}C) at various time intervals.

CONCLUSION

Computer programming, modelling and simulation using MATLAB elevate the students' interests and understanding of heat transfer. This approach enables the students to better visualize complex processes of multidimensional transient heat transfer. Once they appreciate the graphical solutions of the `pdetool` GUI, the students can then develop computer programs based on appropriate numerical techniques. In short, the simulation approach acts as an eye opener to the students that problems related to the non-ideality of real life processes can be solved. By applying the developed simulation programs, the students can quickly explore the effect of changes in system properties and operating conditions on the temperature profiles within the cylindrical tube wall. This way, the facilitation and enhancement of the learning process can be easily achieved as compared to the conventional method of drawing on the white (or black) board.

ACKNOWLEDGEMENT

The authors would like to express gratitude to a UTP final year student, Wan Ahmad Akram, for the programming work using MATLAB.

REFERENCES

1. Kassim, H.O and Cadbury, R.G.; Computers and Chemical Engineering. 1996, **20**, S1341-S1346.

2. Sinclair J.L. *Journal of Chemical Engineering Education*, 1998, 108-112.

3. Brown D.J.; Cremer M.A.; Smith P.J.; Waibel R.T. *Fireside Modelling In Cracking Furnaces*, AIChE Ninth Annual Ethylene Producers' Conference, March 1997, Houston, Texas.

4. Chapra S.C.; Canale R.P. *Numerical Methods for Engineers*, McGraw-Hill, 1998.

5. MATLAB Partial Differential Equation Toolbox User's Guide (Release 12). The MathWorks, Inc.