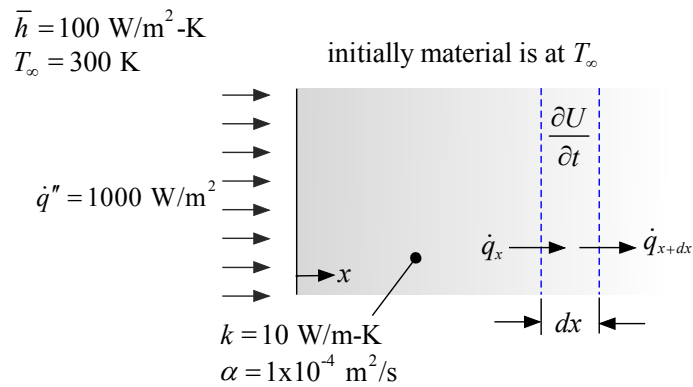


### 3.4.7 Numerical Inverse Laplace Transform

The inverse Laplace transform has proven to be the most difficult step in applying the Laplace transform technique. There are a limited number of transforms and inverse transforms available in most tables and Maple's ability to provide inverse Laplace transforms automatically is limited. The lack of inverse transforms reduces the utility of the Laplace transform method. However, algorithms exist for numerically providing inverse Laplace transforms and they are powerful tools that can be used extend the applicability of the Laplace transform technique. One algorithm is described by de Hoog et al. (1982) and it has been implemented in MATLAB by Hollenbeck (1998). The MATLAB function `invlap.m` can be downloaded from the web site associated with this book ([www.cambridge.org/nellisandklein](http://www.cambridge.org/nellisandklein)). The use of this function is illustrated in this section

Figure E29-1 illustrates a semi-infinite body that is cooled at its surface ( $x = 0$ ) by a fluid with temperature  $T_\infty = 300$  K with heat transfer coefficient  $\bar{h} = 100$  W/m<sup>2</sup>-K. At time  $t = 0$ , the material is in equilibrium with the fluid when a heat flux is applied to the surface,  $\dot{q}'' = 1000$  W/m<sup>2</sup>.



**Figure E29-1: Semi-infinite body exposed to a surface heat flux.**

The material has conductivity  $k = 10$  W/m-K and thermal diffusivity  $\alpha = 1 \times 10^{-4}$  m<sup>2</sup>/s. The governing differential equation for the semi-infinite solid is derived by focusing on a differential control volume (see Figure E29-1). The energy balance suggested by Figure E29-1 is:

$$\dot{q}_x = \dot{q}_{x+dx} + \frac{\partial U}{\partial t} \quad (\text{E29-1})$$

expanding the  $x+dx$  term and simplifying leads to:

$$0 = \frac{\partial \dot{q}_x}{\partial x} dx + \frac{\partial U}{\partial t} \quad (\text{E29-2})$$

The conduction term is evaluated using Fourier's law:

$$\dot{q}_x = -k A_c \frac{\partial T}{\partial x} \quad (\text{E29-3})$$

where  $A_c$  is the cross-sectional area of the material perpendicular to the  $x$ -direction. The time rate of change of the internal energy of the material in the control volume is:

$$\frac{\partial U}{\partial t} = \rho c A_c dx \frac{\partial T}{\partial t} \quad (\text{E29-4})$$

Substituting Eqs. (E29-3) and (E29-4) into Eq. (E29-2) leads to:

$$0 = \frac{\partial}{\partial x} \left[ -k A_c \frac{\partial T}{\partial x} \right] dx + \rho c A_c dx \frac{\partial T}{\partial t} \quad (\text{E29-5})$$

or

$$\frac{\partial^2 T}{\partial x^2} - \frac{1}{\alpha} \frac{\partial T}{\partial t} = 0 \quad (\text{E29-6})$$

The initial condition for the problem is:

$$T_{t=0} = T_\infty \quad (\text{E29-7})$$

An interface energy balance at the surface leads to one boundary condition:

$$\dot{q}'' = -k \left. \frac{\partial T}{\partial x} \right|_{x=0} + \bar{h} (T_{x=0} - T_\infty) \quad (\text{E29-8})$$

The temperature far from the surface must remain at the initial temperature:

$$T_{x \rightarrow \infty} = T_\infty \quad (\text{E29-9})$$

To solve this problem using the Laplace transform it is necessary to transform the governing differential equation, Eq. (E29-6):

$$\frac{d^2 \hat{T}}{dx^2} - \frac{1}{\alpha} (s \hat{T} - T_{t=0}) = 0 \quad (\text{E29-10})$$

Substituting Eq. (E29-7) into Eq. (E29-10) and rearranging:

$$\frac{d^2 \hat{T}}{dx^2} - \frac{s}{\alpha} \hat{T} = -\frac{T_\infty}{\alpha} \quad (\text{E29-11})$$

The spatial boundary conditions, Eqs. (E29-7) and (E29-8), are transformed to the  $s$  domain in order to provide the boundary conditions for the ordinary differential equation in the  $s$  domain, Eq. (E29-11):

$$\widehat{T}_{x \rightarrow \infty} = \frac{T_{\infty}}{s} \quad (\text{E29-12})$$

$$\frac{\dot{q}''}{s} = -k \left. \frac{d\widehat{T}}{dx} \right|_{x=0} + \bar{h} \left( \widehat{T}_{x=0} - \frac{T_{\infty}}{s} \right) \quad (\text{E29-13})$$

The solution to the ordinary differential equation, Eq. (E29-11), is:

$$\widehat{T} = C_1 \exp\left(\sqrt{\frac{s}{\alpha}}x\right) + C_2 \exp\left(-\sqrt{\frac{s}{\alpha}}x\right) + \frac{T_{\infty}}{s} \quad (\text{E29-14})$$

The boundary condition at  $x \rightarrow \infty$  leads to:

$$\widehat{T}_{x \rightarrow \infty} = C_1 \exp\left(\sqrt{\frac{s}{\alpha}}\infty\right) + C_2 \exp\left(-\sqrt{\frac{s}{\alpha}}\infty\right) + \frac{T_{\infty}}{s} = \frac{T_{\infty}}{s} \quad (\text{E29-15})$$

which can only be true if  $C_1 = 0$ ; therefore:

$$\widehat{T} = C_2 \exp\left(-\sqrt{\frac{s}{\alpha}}x\right) + \frac{T_{\infty}}{s} \quad (\text{E29-16})$$

The boundary condition at  $x = 0$  leads to:

$$\frac{\dot{q}''}{s} = k C_2 \sqrt{\frac{s}{\alpha}} + \bar{h} \left( C_2 + \frac{T_{\infty}}{s} - \frac{T_{\infty}}{s} \right) \quad (\text{E29-17})$$

Equation (E29-17) is solved for  $C_2$ :

$$C_2 = \frac{\dot{q}''}{s \left( k \sqrt{\frac{s}{\alpha}} + \bar{h} \right)} \quad (\text{E29-18})$$

The solution in the  $s$  domain is therefore:

$$\hat{T} = \frac{\dot{q}''}{s \left( k \sqrt{\frac{s}{\alpha}} + \bar{h} \right)} \exp \left( -\sqrt{\frac{s}{\alpha}} x \right) + \frac{T_\infty}{s} \quad (\text{E29-19})$$

Equation (E29-19) can be transformed back to the time domain using Table 3-3:

$$T = \frac{\dot{q}''}{h} \left[ \operatorname{erfc} \left( \frac{x}{2\sqrt{\alpha t}} \right) - \exp \left( \frac{\bar{h} x}{k} \right) \exp \left( \frac{\bar{h}^2 \alpha t}{k^2} \right) \operatorname{erfc} \left( \frac{\bar{h} \sqrt{\alpha t}}{k} + \frac{x}{2\sqrt{\alpha t}} \right) \right] + T_\infty \quad (\text{E29-20})$$

Equation (E29-20) is programmed in EES

```

$UnitSystem SI MASS RAD PA K J
$Tabstops 0.2 0.4 0.6 0.8 3.5

"Inputs"
T_infinity=300 [K]                "ambient temperature"
h_bar=100 [W/m^2-K]              "heat transfer coefficient"
k=10 [W/m-K]                    "conductivity"
alpha=1e-4 [m^2/s]              "thermal diffusivity"
qf=1000 [W/m^2]                 "surface heat flux"

x=0.01 [m]                       "position"
time=1.0 [s]                    "time"
T=(qf/h_bar)*(erfc(x/(2*sqrt(alpha*time))))-exp(h_bar*x/k)*exp(h_bar^2*alpha*time/k^2)&
*erfc(h_bar*sqrt(alpha*time)/k+x/(2*sqrt(alpha*time))))+T_infinity
    
```

and used to generate Figures E29-2 and E29-3, which show the temperature as a function of position at several values of time and the temperature as a function of time at several values of position, respectively.

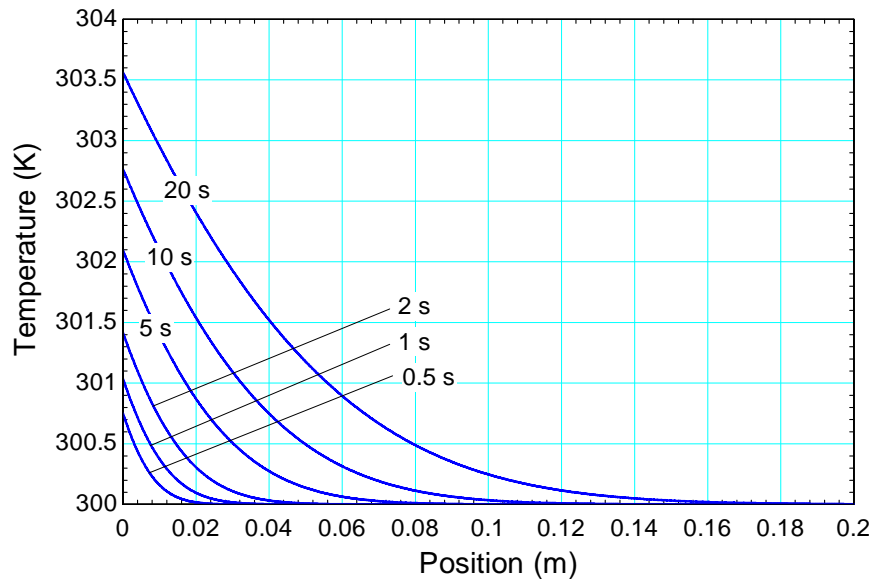


Figure E29-2: Temperature as a function of position at various values of time.

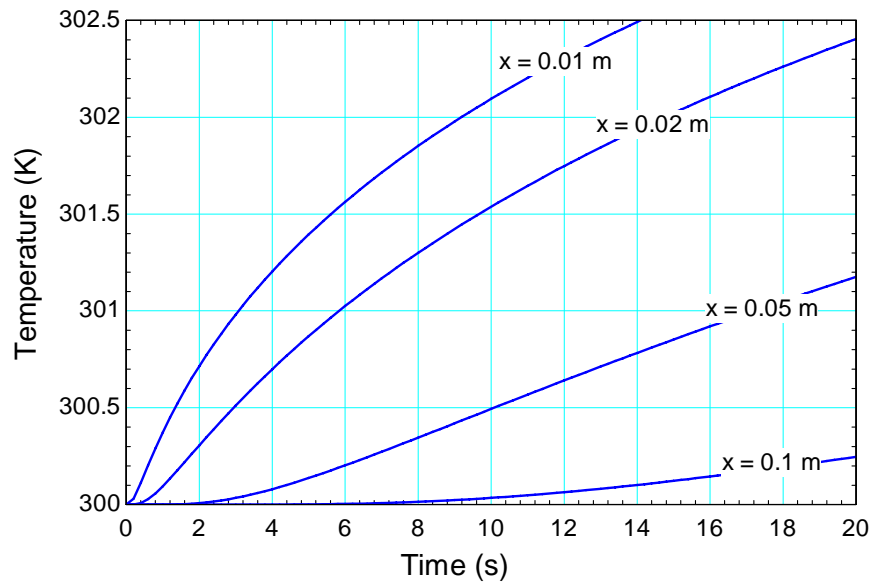


Figure E29-3: Temperature as a function of time at various positions.

The solution can also be obtained by doing a numerical inverse Laplace transform using the MATLAB function `invlap.m`, which can be downloaded from [www.cambridge.org/nellisandklein](http://www.cambridge.org/nellisandklein). The first step is to setup a function that implements the solution in the  $s$  domain. The function should take as arguments a vector of values of  $s$  followed by any other required parameters and return a vector of the values of the solution at each value of  $s$ .

```
function[F]=S3p4p7(s,x,alpha,h_bar,k,T_infinity,qf)
%solution to semi-infinite body subjected to a heat flux and convection
%in the s-domain
%
%Inputs:
% s - vector of values of the s parameter (1/s)
% x - distance from surface of wall (m)
% alpha - thermal diffusivity (m^2/s)
% h_bar - heat transfer coefficient (W/m^2-K)
% k - thermal conductivity (W/m-K)
% T_infinity - ambient temperature (K)
% qf - heat flux (W/m^2)
%
%Output:
% F - solution in the s-domain at each value of s (K-s)

F=qf*exp(-sqrt(s./alpha)*x)./(s.*(k*sqrt(s./alpha)+h_bar))+T_infinity./s;
end
```

Note the use of the `.` operator which ensures that the operations are applied to each element of a vector. We can quickly test that the function does return a column vector of solutions by applying it at the command line:

```
>> [F]=S3p4p7([1;2;3],0,1,1,1,1,1)
```

```
F =
```

```
1.5000
0.7071
0.4553
```

The next step is to obtain the solution in the time domain. A script, S3p4p7\_run, is created in order to call the invlap.m function. The calling protocol for the invlap.m function is:

```
f = invlap(F, t, tol1, tol2, P1,P2,P3,P4,P5,P6,P7,P8,P9);
```

where F is the name of the function that implements the solution in the  $s$  domain (e.g., 'S3p4p7'), t is a vector of times where the solution is desired, tol1 and tol2 are numerical parameters (with default values 0 and  $1 \times 10^{-9}$ , respectively), and the parameters P1, P2, etc. are additional input parameters to provide the function F (e.g., x, alpha, etc.).

The inputs are defined in the script S3p4p7\_run:

```
T_infinity=300;           % average ambient temperature (K)
h_bar = 100;             % heat transfer coefficient (W/m^2-K)
k = 10;                 % conductivity (W/m-K)
alpha = 1e-4;           % thermal diffusivity (m^2/s)
qf=1000;                % heat flux (W/m^2)
```

The values of the numerical parameters alpha and tol are kept at their default values:

```
tol1=0;                 % default for largest pole for function
tol2=1e-9;              % default for numerical tolerance of approaching pole
```

A position and a vector of times are defined:

```
%invert Laplace transform
x=0.01;                 % position (m)
t=linspace(0.01,24,101)'; % times to evaluate solution
```

The solution is obtained:

```
T=invlap('S3p4p7',t,tol1,tol2,x,alpha,h_bar,k,T_infinity,qf);
```

and plotted from the command line:

```
>> S3p4p7_run
>> plot(t,T)
```

leading to Figure E29-4.

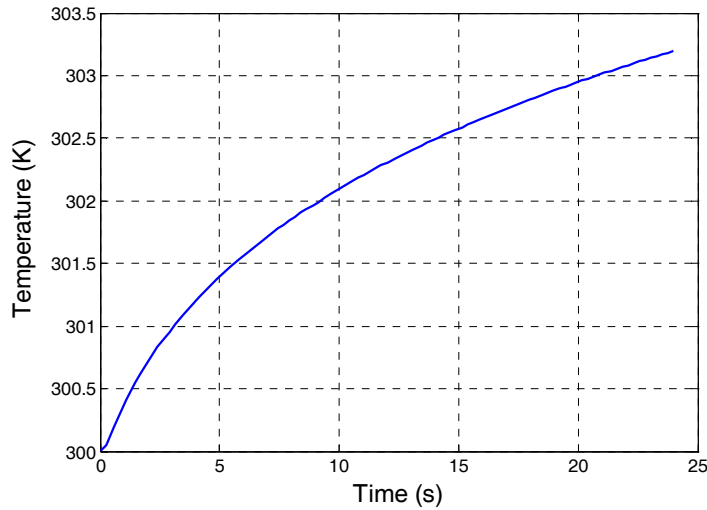


Figure E29-4: Temperature as a function time at  $x = 0.01$  m.

Figure E29-5 illustrates the solution obtained numerically in MATLAB overlaid onto the solution obtained analytically, previously shown in Figure E29-3.

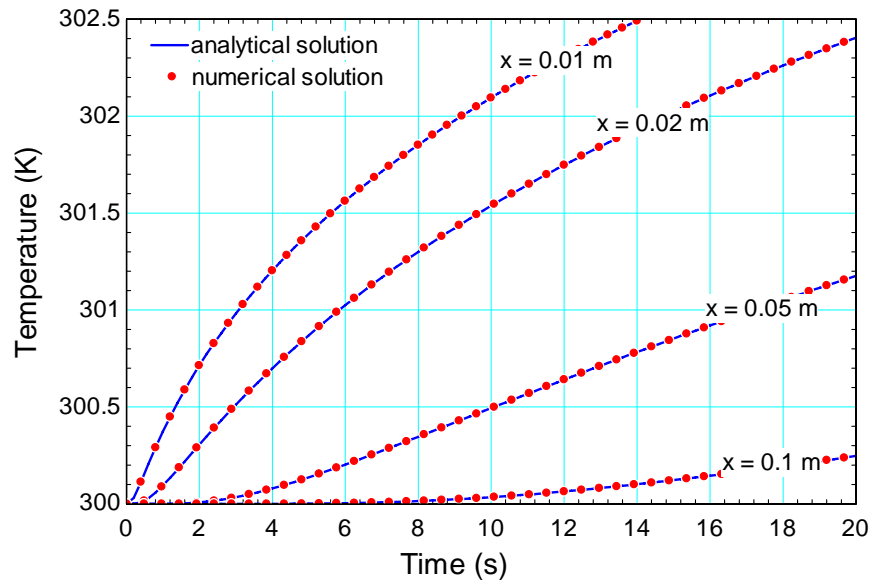
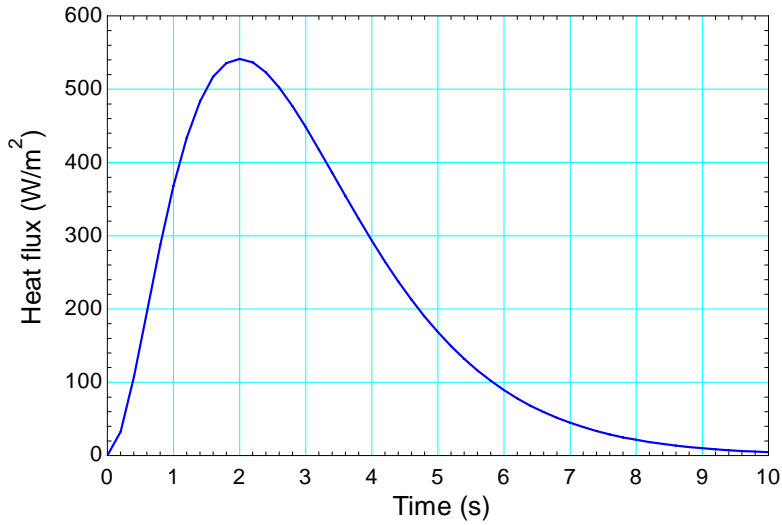


Figure E29-5: Temperature as a function of time for various values of position, predicted analytically and numerically.

Because it is not necessary to carry out the inverse Laplace transform explicitly, it is possible to solve much more difficult problems using the `invlap.m` function. For example, the heat flux at the surface of the wall shown in Figure E29-1 can be allowed to vary with time according to:

$$\dot{q}'' = \dot{q}''_{max} \left( \frac{t}{t_d} \right)^2 \exp\left( -\frac{t}{t_d} \right) \quad (\text{E29-21})$$

Equation (E29-21) is shown in Figure E29-6 with  $\dot{q}''_{max} = 1000 \text{ W/m}^2$  and  $t_d = 1 \text{ s}$ ; the heat flux is short duration pulse that decays to zero after some time.



**Figure E29-6: Surface heat flux as a function of time.**

The problem in the time domain remains the same, except that the boundary condition at  $x = 0$  is:

$$\dot{q}''_{max} \left( \frac{t}{t_d} \right)^2 \exp\left( -\frac{t}{t_d} \right) = -k \left. \frac{\partial T}{\partial x} \right|_{x=0} + \bar{h} (T_{x=0} - T_{\infty}) \quad (\text{E29-22})$$

Taking the Laplace transform of the partial differential equation, Eq. (E29-6), leads to:

$$\frac{d^2 \hat{T}}{dx^2} - \frac{s}{\alpha} \hat{T} = -\frac{T_{\infty}}{\alpha} \quad (\text{E29-23})$$

which is solved by:

$$\hat{T} = C_1 \exp\left( \sqrt{\frac{s}{\alpha}} x \right) + C_2 \exp\left( -\sqrt{\frac{s}{\alpha}} x \right) + \frac{T_{\infty}}{s} \quad (\text{E29-24})$$

Transforming the boundary condition at  $x \rightarrow \infty$ , Eq. (E29-9), leads to:

$$\hat{T}_{x \rightarrow \infty} = \frac{T_{\infty}}{s} \quad (\text{E29-25})$$

Substituting Eq. (E29-24) into Eq. (E29-25) leads to:

$$\hat{T} = C_2 \exp\left( -\sqrt{\frac{s}{\alpha}} x \right) + \frac{T_{\infty}}{s} \quad (\text{E29-26})$$

which is the same solution obtained previously, Eq. (E29-16). The transformation of the new boundary condition at  $x = 0$ , Eq. (E29-22), can be accomplished using Table 3-3 or Maple:

```
> restart;
> with(intrinsics);
  [addtable, fourier, fouriercos, fouriersin, hankel, hilbert, invfourier, invhilbert, invlaplace,
   invmellin, laplace, mellin, savetable]
> laplace(qfmax*(t/t_d)^2*exp(-t/t_d),t,s);
      2 qfmax t_d
      (s t_d + 1)^3
```

which leads to:

$$\frac{2\dot{q}''t_d}{(st_d+1)^3} = -k \left. \frac{dT}{dx} \right|_{x=0} + \bar{h} \left( \hat{T}_{x=0} - \frac{T_\infty}{s} \right) \quad (\text{E29-27})$$

Substituting Eq. (E29-26) into Eq. (E29-27) leads to:

$$\frac{2\dot{q}''t_d}{(st_d+1)^3} = C_1 \left( k \sqrt{\frac{s}{\alpha}} + \bar{h} \right) \quad (\text{E29-28})$$

or

$$C_1 = \frac{2\dot{q}''t_d}{(st_d+1)^3 \left( k \sqrt{\frac{s}{\alpha}} + \bar{h} \right)} \quad (\text{E29-29})$$

Equation (E29-29) and (E29-26) together represent the solution to the problem in the  $s$  domain. It is not easy (or perhaps not even possible) to carry out the inverse transform associated with this solution. Instead, the inverse Laplace transform is accomplished numerically. A function called S3p4p7\_pulse.m is set up to implement the solution in the  $s$  domain:

```
function[F]=S3p4p7_pulse(s,x,alpha,h_bar,k,T_infinity,qfmax,t_d)
%solution to semi-infinite body subjected to a heat flux pulse and
%convection in the s-domain
%
%Inputs:
% s - s parameter (1/s)
% x - distance from surface of wall (m)
% alpha - thermal diffusivity (m^2/s)
% h_bar - heat transfer coefficient (W/m^2-K)
% k - thermal conductivity (W/m-K)
% T_infinity - ambient temperature (K)
% qfmax - related to maximum heat flux during pulse (W/m^2)
```

```

% t_d - related to the duration of the pulse (s)

C2=2*qfmax*t_d./((s*t_d+1).^3.*(k*sqrt(s/alpha)+h_bar));
F=C2.*exp(-sqrt(s/alpha)*x)+T_infinity./s;
end

```

and a script S3p4p7\_pulse\_run.m is set up to call the invlap.m function and obtain the solution in the time domain.

```

t=linspace(0.01,24,101)'; % times to evaluate solution
T_infinity=300; % average ambient temperature (K)
h_bar = 100; % heat transfer coefficient (W/m^2-K)
k = 10; % conductivity (W/m-K)
alpha = 1e-4; % thermal diffusivity (m^2/s)
qfmax=1000; % heat flux (W/m^2)
t_d=1; % pulse duration (s)

tol1=0; % default for largest pole for function
tol2=1e-9; % default for numerical tolerance of approaching pole

%invertLaplace transform
x=0.01; % position (m)
T=invlap('S3p4p7_pulse',t,tol1,tol2,x,alpha,h_bar,k,T_infinity,qfmax,t_d);

```

Figure E29-7 shows the temperature as a function of time at various values of position.

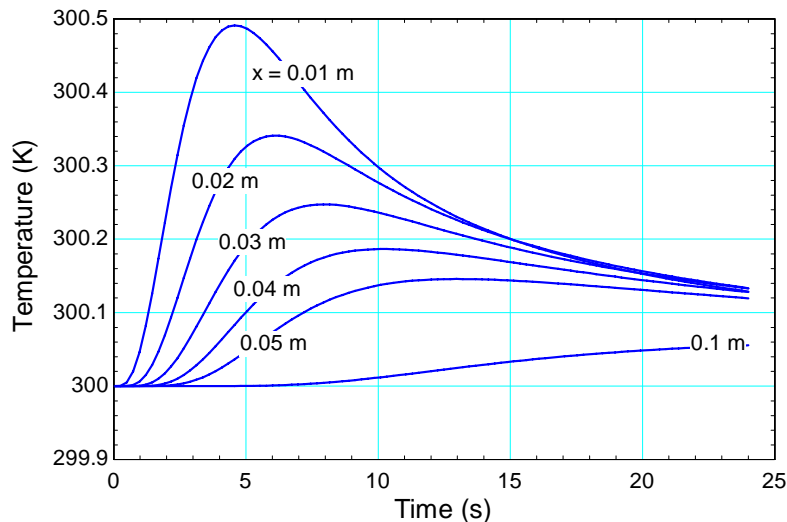


Figure E29-7: Temperature as a function of time at various values of position.

## References

de Hoog, F.R., J.H. Knight, and A.N. Stokes, "An improved method for numerical inversion of Laplace transforms," *S.I.A.M. J. Sci. and Stat. Comp.*, Vol. 3, pp. 357-366, (1982).

Hollenbeck, K.J., INVLAP.m: A MATLAB function for numerical inversion of Laplace transforms by the de Hoog algorithm, <http://www.isva.dtu.dk/staff/karl/invlap.html>, (1998).