# EXAMPLE 1.5-1: Thermal Protection System for Atmospheric Entry

The kinetic energy associated with the atmospheric entry of a space vehicle results in extremely large heat fluxes, large enough to completely vaporize the vehicle if it were not adequately protected. The outer structure of the vehicle is called its aeroshell and the outer layer of the material on the aeroshell is called the Thermal Protection System (or TPS). The heat flux experienced by the aeroshell can reach 100 W/cm$^2$, albeit for only a short period of time.

Consider a TPS consisting of a non-metallic ablative layer with thickness, $th_{ab}$ = 5 cm that is bonded to a layer of steel with thickness, $th_s$ = 1 cm, as shown in Figure 1. The outer edge of the ablative heat shield ($x$ = 0) reaches the material's melting temperature ($T_m$ = 755 K) under the influence of the heat flux. The melting limits the temperature that is reached at the outer surface of the shield and protects the internal air until the shield is consumed. In this problem we will assume that the shield is consumed very slowly so that a quasi-steady temperature distribution is set up in the ablative shield. The latent heat of fusion of the ablative shield is $\Delta i_{fus,ab}$ = 200 kJ/kg and its density is $\rho_{ab}$ = 1200 kg/m$^3$.

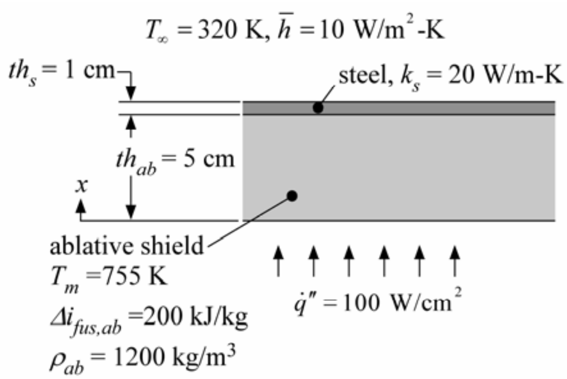


**Figure 1: A Thermal Protection System**

The thermal conductivity of the ablative shield is highly temperature-dependent; thermal conductivity values at several temperatures are provided in Table 1.

**Table 1: Thermal conductivity of ablative shield material in the solid phase**

| Temperature | Thermal conductivity |
|---|---|
| 300 K | 0.10 W/m-K |
| 350 K | 0.15 W/m-K |
| 400 K | 0.19 W/m-K |
| 450 K | 0.21 W/m-K |
| 500 K | 0.22 W/m-K |
| 550 K | 0.24 W/m-K |
| 600 K | 0.28 W/m-K |
| 650 K | 0.33 W/m-K |
| 700 K | 0.38 W/m-K |
| 755 K | 0.45 W/m-K |

The thermal conductivity of the steel may be assumed to be constant at $k_s$ = 20 W/m-K. The internal surface of the steel is exposed to air at $T_\infty$ = 320 K with average heat transfer coefficient, $\overline{h}$ = 10 W/m$^2$-K.

Assume that the TPS reaches a quasi-steady-state under the influence of a heat flux $\dot{q}'' = 100$ W/cm$^2$ and that the surface temperature of the ablation shield reaches its melting point.

a.) Develop a numerical model using MATLAB that can determine the heat flux that is transferred to the air and the rate that the ablative shield is being consumed.

The solution will be developed as a MATLAB function called Ablative_shield; the input to the function will be the number of nodes to use in the solution while the outputs will include the position of the nodes and the predicted temperature at each node as well as the two quantities specifically requested, the heat flux incident on the air and the rate of shield ablation. Select New and M-File from the File menu and save the M-file as Ablative_shield (the .m extension is added automatically). The first line of the function establishes the input/output protocol:

```
function[x,T,q_flux_in_Wcm2,dthabdt_cms]=Ablative_shield(N)

    %EXAMPLE 1.5-1: Thermal Protection System for Atmospheric Entry
    %
    % Inputs:
    % N: number of nodes in solution (-)
    %
    % Outputs:
    % x: position of nodes (m)
    % T: temperatures at nodes (K)
    % q_flux_in_Wcm2: heat flux to air (W/cm^2)
    % dthabdt_cms: rate of shield consumption (cm/s)
```

The next section of the code establishes the remaining input parameters (i.e., those not provided as arguments to the function); note that each input is converted immediately to SI units.

```
    th_ab=0.05;                 %ablation shield thickness (m)
    th_s=0.01;                  %steel thickness (m)
    k_s=20;                     %steel conductivity (W/m-K)
    q_flux=100*100^2;           %heat flux (W/m^2)
    T_m=755;                    %melting temperature (K)
    DELTAi_fus_ab=200e3;        %latent heat of fusion (J/kg)
    h_bar=10;                   %heat transfer coefficient (W/m^2-K)
    T_infinity=320;             %internal air temperature (K)
    rho_ab=1200;                %density (kg/m^3)
    A_c=1;                      %per unit area of wall (m^2)
```

In order to solve this problem, it will be necessary to create a function that returns the conductivity of the ablative shield. The easiest way to do this is to enter the data from Table 1 into a sub-function and interpolate between the data points.

```
function[k]=k_ab(T)

    %data
    Td=[300,350,400,450,500,550,600,650,700,755];
    kd=[0.1,0.15,0.19,0.21,0.22,0.24,0.28,0.33,0.38,0.45];
    k=interp1(Td,kd,T,'spline'); %interpolate data to obtain conductivity
```

```
end
```

The interp1 function in MATLAB is used for the interpolation. The interp1 function requires three arguments; the first two are the vectors of the independent and dependent data, respectively, and the third is the value of the independent variable at which you want to find the dependent variable. An optional 4th argument specifies the type of interpolation to use. To obtain more detailed help for this (or any) MATLAB function, use the help command:

```
>> help interp1
 INTERP1 1-D interpolation (table lookup)
    YI = INTERP1(X,Y,XI) interpolates to find YI, the values of the
    underlying function Y at the points in the array XI. X must be a
    vector of length N.
    If Y is a vector, then ...
```

The numerical model of the TPS will consider the ablative material; the steel will be considered as part of the thermal resistance between the inner surface of the shield and the air and therefore will affect the boundary condition at $x = th_{ab}$. There is no reason to treat the steel with the numerical model since the steel has, by assumption, constant properties and is at steady-state; therefore, the analytical solution derived in Section 1.2.3 for the resistance of a plane wall holds exactly.

The nodes are distributed uniformly from $x = 0$ to $x = th_{ab}$, where $x = 0$ corresponds to the outer surface of the shield, as shown in Figure 2.

$$x_i = (i-1)\frac{t_{ab}}{(N-1)} \quad \text{for } i = 1...N$$

The space between adjacent nodes ($\Delta x$) is:

$$\Delta x = \frac{t_{ab}}{(N-1)}$$

The nodes are setup in the MATLAB code according to:

```
    %setup nodes
    DELTAx=th_ab/(N-1);        %distance between nodes
    for i=1:N
        x(i,1)=th_ab*(i-1)/(N-1);   %position of each node
    end
```

An internal control volume, shown in Figure 2, experiences only conduction; therefore, a steady-state energy balance on the control volume is:

$$\dot{q}_{top} + \dot{q}_{bottom} = 0 \tag{1}$$

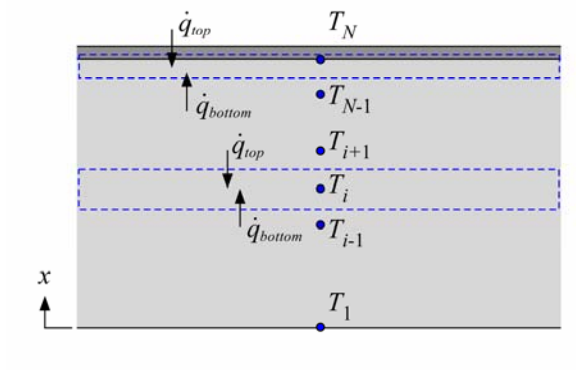


**Figure 2: Distribution of nodes and control volumes.**

The conductivity used to approximate the conduction heat transfer rates in Eq. (1) must be evaluated at the temperature of the boundaries, i.e., the average of the temperatures of the nodes involved in the conduction process, as discussed previously in Section 1.4.3. With this understanding, these rate equations become:

$$\dot{q}_{top} = k_{T=(T_{i+1}+T_i)/2} \frac{A_c}{\Delta x}\left(T_{i+1} - T_i\right) \tag{2}$$

$$\dot{q}_{bottom} = k_{ab,T=(T_{i-1}+T_i)/2} \frac{A_c}{\Delta x}\left(T_{i-1} - T_i\right) \tag{3}$$

where $A_c$ is the cross-sectional area. Substituting Eqs. (2) and (3) into Eq. (1) leads to:

$$k_{ab,T=(T_{i+1}+T_i)/2} \frac{A_c}{\Delta x}\left(T_{i+1} - T_i\right) + k_{ab,T=(T_{i-1}+T_i)/2} \frac{A_c}{\Delta x}\left(T_{i-1} - T_i\right) = 0 \text{ for } i = 2..(N-1) \tag{4}$$

The node on the outer surface (i.e., node 1) has a specified temperature, the melting temperature of the ablative material:

$$T_1 = T_m \tag{5}$$

The energy balance for the node on the inner surface (i.e., node $N$) is also shown in Figure 2:

$$\dot{q}_{top} + \dot{q}_{bottom} = 0 \tag{6}$$

where

$$\dot{q}_{bottom} = k_{ab,T=(T_{N-1}+T_N)/2} \frac{A_c}{\Delta x}\left(T_{N-1} - T_N\right) \tag{7}$$

$$\dot{q}_{top} = \frac{\left(T_{\infty} - T_N\right)}{R_{cond,s} + R_{conv}} \tag{8}$$

where $R_s$ and $R_{conv}$ are the thermal resistances associated with conduction through the steel and convection from the internal surface of the steel to the air.

$$R_{cond,s} = \frac{th_s}{k_s\, A_c}$$

$$R_{conv} = \frac{1}{\bar{h}\, A_c}$$

```
R_cond_s=th_s/(A_c*k_s);  %conduction resistance of steel (K/W)
R_conv=1/(A_c*h_bar);       %convection resistance (K/W)
```

Substituting Eqs. (7) and (8) into Eq. (6) leads to:

$$\frac{\left(T_{\infty} - T_N\right)}{R_{cond,s} + R_{conv}} + k_{ab,T=\left(T_{N-1}+T_N\right)/2} \frac{A_c}{\Delta x}\left(T_{N-1} - T_N\right) = 0 \tag{9}$$

Note that Eqs. (4), (5), and (9) are a complete set of equations in the unknown temperatures $T_i$ for $i$ = 1..$N$; however, these equations cannot be written as a linear combination of the unknown temperatures because the conductivity of the ablative shield depends on temperature. In order to apply successive substitution, the conductivity will be evaluated using guess values for these temperatures ($\hat{T}$). A linear variation in temperature from $T_m$ to $T_\infty$ is used as the guess values to start the process:

$$\hat{T}_i = T_m + \left(T_{\infty} - T_m\right)\frac{\left(i-1\right)}{\left(N-1\right)} \quad \text{for } i = 1..N$$

```
%initial guess for temperature distribution
for i=1:N
    Tg(i,1)=T_m+(T_infinity-T_m)*(i-1)/(N-1); %linear from melting to air (K)
end
```

The matrix A and vector b are initialized according to:

```
%setup matrices
A=spalloc(N,N,3*N);
b=zeros(N,1);
```

Equation (5) is rewritten to make it clear what the coefficient and the constants are:

$$T_1 \underbrace{[1]}_{A_{1,1}} = \underbrace{T_m}_{b_1}$$

```
%node 1
A(1,1)=1;
b(1,1)=T_m;
```

Equation (4) is rewritten, using the guess temperatures to compute the conductivity of the ablative shield and also to clearly identify the coefficients and constants:

$$T_i \underbrace{\left[ -k_{ab,T=\left(\hat{T}_{i+1}+\hat{T}_i\right)/2} \frac{A_c}{\Delta x} - k_{ab,T=\left(\hat{T}_{i-1}+\hat{T}_i\right)/2} \frac{A_c}{\Delta x} \right]}_{A_{i,i}} + T_{i+1} \underbrace{\left[ k_{ab,T=\left(\hat{T}_{i+1}+\hat{T}_i\right)/2} \frac{A_c}{\Delta x} \right]}_{A_{i,i+1}}$$

$$+ T_{i-1} \underbrace{\left[ k_{ab,T=\left(\hat{T}_{i-1}+\hat{T}_i\right)/2} \frac{A_c}{\Delta x} \right]}_{A_{i,i-1}} = 0 \quad \text{for } i = 2..\left(N-1\right)$$

```
%internal nodes
for i=2:(N-1)
    A(i,i)=-k_ab((Tg(i)+Tg(i+1))/2)*A_c/DELTAx-k_ab((Tg(i)+Tg(i-1))/2)*A_c/DELTAx;
    A(i,i+1)=k_ab((Tg(i)+Tg(i+1))/2)*A_c/DELTAx;
    A(i,i-1)=k_ab((Tg(i)+Tg(i-1))/2)*A_c/DELTAx;
end
```

Equation (9) is rewritten:

$$T_N \underbrace{\left[ -\frac{1}{R_{cond,s} + R_{conv}} - k_{ab,T=\left(\hat{T}_{N-1}+\hat{T}_N\right)/2} \frac{A_c}{\Delta x} \right]}_{A_{N,N}} + T_{N-1} \underbrace{\left[ k_{ab,T=\left(\hat{T}_{N-1}+\hat{T}_N\right)/2} \frac{A_c}{\Delta x} \right]}_{A_{N,N-1}} = \underbrace{-\frac{T_\infty}{R_{cond,s} + R_{conv}}}_{b_N}$$

```
%node N
A(N,N)=-k_ab((Tg(N)+Tg(N-1))/2)*A_c/DELTAx-1/(R_cond_s+R_conv);
A(N,N-1)=k_ab((Tg(N)+Tg(N-1))/2)*A_c/DELTAx;
b(N,1)=-T_infinity/(R_cond_s+R_conv);
```

The matrix equation is solved:

```
X=A\b;    %solve matrix equation
T=X;
```

The solution is not complete, it is necessary to iterate until the solution ($T$) matches the assumed temperature ($\hat{T}$). This is accomplished by placing the commands that setup and solve the matrix equation within a while loop that terminates when the rms error ($err$) is below some tolerance ($tol$). The rms error is computed according to:

$$err = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(T_i - \hat{T}_i\right)^2}$$

```
err=sqrt(sum((T-Tg).^2)/N) %compute rms error
```

Note that the sum command computes the sum of all of the elements in the vector provided to it and the use of .^2 indicates that each element in the vector should be squared (as opposed to ^2 which would multiply the vector by itself). Also notice that the error computation is not terminated with a semicolon so that the value of the rms error will be reported after each iteration.

To start the iteration process, set the error to a very large value (larger than *tol*) in order to ensure that the while loop executes at least once. After the solution has been obtained, compute the rms error and reset the vector `Tg` to the vector `T`. The result is below with the new lines are highlighted in bold.

```
err=999;      %initial value of error (K), must be larger than tol
tol = 0.01; %tolerance for convergence (K)
while(err>tol)
    %node 1
    A(1,1)=1;
    b(1,1)=T_m;

    %internal nodes
    for i=2:(N-1)
        A(i,i)=-k_ab((Tg(i)+Tg(i+1))/2)*A_c/DELTAx-k_ab((Tg(i)+Tg(i-1))/2)*A_c/DELTAx;
        A(i,i+1)=k_ab((Tg(i)+Tg(i+1))/2)*A_c/DELTAx;
        A(i,i-1)=k_ab((Tg(i)+Tg(i-1))/2)*A_c/DELTAx;
    end

    %node N
    A(N,N)=-k_ab((Tg(N)+Tg(N-1))/2)*A_c/DELTAx-1/(R_cond_s+R_conv);
    A(N,N-1)=k_ab((Tg(N)+Tg(N-1))/2)*A_c/DELTAx;
    b(N,1)=-T_infinity/(R_cond_s+R_conv);

    X=A\b;     %solve matrix equation
    T=X;

    err=sqrt(sum((T-Tg).^2)/N) %compute rms error
    Tg=T;  %reset guess values used to setup A and b
end
```

The heat flux to the air ($\dot{q}''_{in}$) can be computed.

$$\dot{q}''_{in} = \frac{\left(T_N - T_\infty\right)}{A_c\left(R_s + R_{conv}\right)}$$

The rate at which the ablative shield is consumed can be computed based on an energy balance at the outer surface; the heat flux related to re-entry either consumes the shield or is transferred to the air. Note that this is actually a simplification of the problem; the problem is a moving boundary problem and this solution is valid only in the limit that the energy carried by the motion of interface is small relative to the energy removed by its vaporization.

$$\dot{q}'' = \rho_{ab}\,\Delta i_{fus,ab}\,\frac{dth_{ab}}{dt} + \dot{q}''_{in}$$

or

$$\frac{dth_{ab}}{dt} = \frac{\dot{q}'' - \dot{q}''_{in}}{\rho_{ab}\,\Delta i_{fus,ab}}$$

These calculations are provided by adding the following lines to the Ablative_shield function:

```
    q_flux_in=(T(N)-T_infinity)/(R_cond_s+R_conv)/A_c;          %heat flux to air (W/m^2)
    q_flux_in_Wcm2=q_flux_in/100^2;                             %heat flux to air (W/cm^2)
    dthabdt=(q_flux-q_flux_in)/(DELTAi_fus_ab*rho_ab);          %rate of shield consumption (m/s)
    dthabdt_cms=dthabdt*100;                                    %rate of shield consumption (cm/s)
end
```

Calling the function Ablative_shield from the workspace leads to:

```
>> [x,T,q_flux_in_Wcm2,dthabdt_cms]=Ablative_shield(100);
err =
 105.7742
err =
   9.0059
err =
   0.8979
err =
   0.1798
err =
   0.0238
err =
   0.0035
```

Figure 3 shows the temperature as a function of position within the shield. The temperature gradient agrees with intuition; the temperature gradient is smaller where the conductivity is largest (i.e., at higher temperatures) which is consistent with a constant heat flow through the wall. It is important to verify that the number of nodes used in the solution is adequate. Figure 4 shows the heat flux to the air as a function of the number of nodes in the solution and indicates that at least 20 nodes are required. The heat flux to the air at the inner surface of the TPS is 0.166 $W/cm^2$, nearly three orders of magnitude less than the heat flux at the outer surface. The TPS is being consumed at a rate of 0.416 cm/s suggesting that the atmospheric entry cannot last more than ten seconds without consuming the entire shield. The thermal analysis of this problem does not consider the loss of ablative material with time and it is therefore a very simplified model of the TPS.

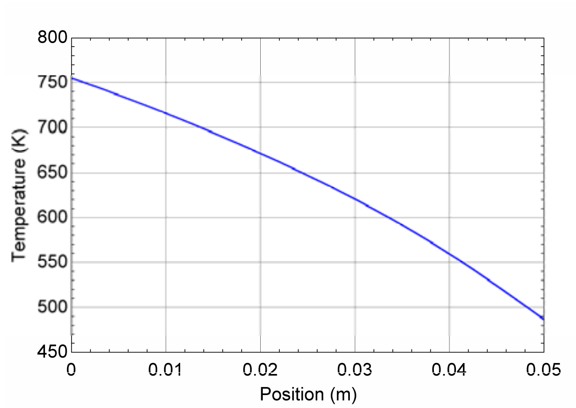


**Figure 3: Temperature as a function of position within the ablative shield.**

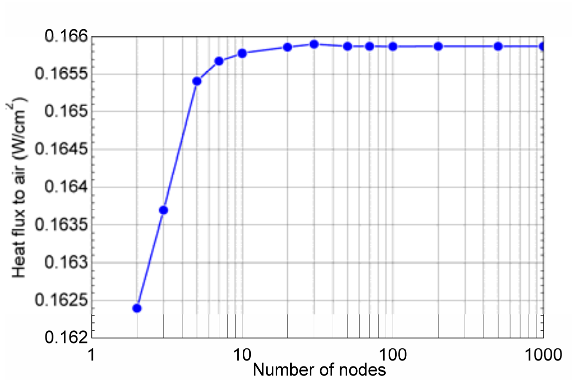


**Figure 4**: **Heat flux to the air as a function of the number of nodes.**