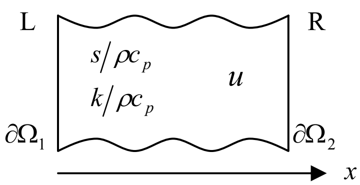
Convection-Diffusion Equation

The convection-diffusion equation is a wonderful model for numerous physical phenomena as a 1-d approximation to the First Law of Thermodynamics. For the one-dimensional fluid transport of temperature (energy) with both Dirichlet and Robin boundary conditions:



u = fluid velocity
s = source
ρ = fluid density
k = thermal conductivity

c_p = specific heat
h = convective heat transfer coef.
T_r = reference temperature

$$
\begin{aligned}
\mathsf{L}(T(x,t)) &= \frac{\partial T(x,t)}{\partial t} + u\frac{\partial T(x,t)}{\partial x} - \frac{\partial}{\partial x}\left(\frac{k}{\rho c_p}\frac{\partial T(x,t)}{\partial x}\right) - \frac{s}{\rho c_p} = 0 && on\ \Omega \\
\mathsf{l}(T(x_L,t)) &= k\frac{\partial T(x,t)}{\partial n} + h\big(T(x,t) - T_r\big) = 0 && on\ \partial\Omega_1 \\
T(x_R,t) &= T_R && on\ \partial\Omega_2 \\
T(x,t_o) &= T_o(x) && on\ \Omega, \partial\Omega
\end{aligned}
\tag{1}
$$

We begin, as usual, by assuming a series expansion approximation for the unknown solution of temperature T(x,t).

$$
T(x,t) \approx T^N(x,t) = \sum_{\alpha=1}^{N} \Psi_\alpha(x)\, T(t)_\alpha = \Psi_\alpha(x)\, T(t)_\alpha \quad \text{for} \quad 1 \le \alpha \le N \tag{2}
$$

Forming the GWS^N and expanding:

$$
GWS^N = \int_\Omega \Psi_\beta \frac{\partial T^N}{\partial t} dx + \int_\Omega \Psi_\beta u \frac{\partial T^N}{\partial x} - \int_\Omega \Psi_\beta \frac{\partial}{\partial x}\left(\frac{k}{\rho c_p}\frac{\partial T^N}{\partial x}\right) dx - \int_\Omega \Psi_\beta \frac{s}{\rho c_p} dx = 0 \quad for \quad 1 \le \alpha \le N \tag{3}
$$

Integrating the third term by parts

$$
GWS^N = \int_\Omega \Psi_\beta \frac{\partial T^N}{\partial t} dx + \int_\Omega \Psi_\beta u \frac{\partial T^N}{\partial x} dx + \int_\Omega \frac{d\Psi_\beta}{dx}\frac{k}{\rho c_p}\frac{\partial T^N}{\partial x} dx - \int_\Omega \Psi_\beta \frac{s}{\rho c_p} dx - \Psi_\beta \frac{k}{\rho c_p}\frac{\partial T^N}{\partial x}\bigg|_{\partial\Omega} = 0 \tag{4}
$$

Substituting the series approximation and rearranging

$$
GWS^N = \int_\Omega \Psi_\beta \Psi_\alpha dx \frac{dT_\alpha}{dt} + \int_\Omega u \Psi_\beta \frac{d\Psi_\alpha}{dx} dx T_\alpha + \int_\Omega \frac{k}{\rho c_p}\frac{d\Psi_\beta}{dx}\frac{d\Psi_\alpha}{dx} dx T_\alpha - \int_\Omega \Psi_\beta \frac{s}{\rho c_p} dx - \Psi_\beta \frac{k}{\rho c_p}\frac{\partial T^N}{\partial x}\bigg|_{\partial\Omega} = 0 \tag{5}
$$

Substituting the Robin boundary condition, the GWS^N becomes:

$$
\begin{aligned}
GWS^N &= \int_\Omega \Psi_\beta \Psi_\alpha dx \frac{dT_\alpha}{dt} + \int_\Omega u \Psi_\beta \frac{d\Psi_\alpha}{dx} dx T_\alpha + \int_\Omega \frac{k}{\rho c_p} \frac{d\Psi_\beta}{dx} \frac{d\Psi_\alpha}{dx} dx T_\alpha - \int_\Omega \Psi_\beta \frac{s}{\rho c_p} dx \\
&\quad + \Psi_\beta \frac{h}{\rho c_p}\left(T^N - T_r\right)\Big|_{\partial\Omega_1} - \Psi_\beta \frac{k}{\rho c_p} \frac{\partial T^N}{\partial x}\Big|_{\partial\Omega_2} \\
&= 0 \qquad \text{for} \quad 1 \le \alpha, \beta \le \mathrm{N}
\end{aligned} \tag{6}
$$

Forming the discrete GWS^h:

$$
\begin{aligned}
GWS^h &= S_e\Bigg(\int_{\Omega_e} \{N_k\}\{N_k\}^T d\bar{x} \frac{d\{T\}_e}{dt} + \int_{\Omega_e} u_e \{N_k\} \frac{d\{N_k\}^T}{dx} d\bar{x} \{T\}_e + \int_{\Omega_e} \left(\frac{k}{\rho c_p}\right)_e \frac{d\{N_k\}}{dx} \frac{d\{N_k\}^T}{dx} d\bar{x} \{T\}_e \\
&\quad - \int_{\Omega_e} \{N_k\}\{N_k\}^T d\bar{x} \{S\}_e + \left(\frac{h}{\rho c_p}\right)_e \{N_k\}\{N_k\}\{T\}_e\Big|_{\partial\Omega_1} - \left(\frac{hT_r}{\rho c_p}\right)_e \{N_k\}\Big|_{\partial\Omega_1} \Bigg) \\
&= 0
\end{aligned} \tag{7}
$$

Identifying integrated terms :

$$
\begin{aligned}
[\mathrm{MASS}]_e &\equiv \int_{\Omega_e} \{N_k\}\{N_k\}^T d\bar{x} \\
&= (\ )(\ )_e \{\ \}_e^T (1)[\mathrm{A200k}]\{\ \}_e
\end{aligned}
$$

$$
\begin{aligned}
[\mathrm{CONVU}]_e &\equiv \int_{\Omega_e} u_e \{N_k\} \frac{d\{N_k\}^T}{dx} d\bar{x} \\
&= (\mathrm{u})(\ )_e \{\ \}_e^T (0)[\mathrm{A201k}]\{\ \}_e
\end{aligned}
$$

$$
\begin{aligned}
[\mathrm{DIFFA}]_e &\equiv \int_{\Omega_e} \left(\frac{k}{\rho c_p}\right)_e \frac{d\{N_k\}}{dx} \frac{d\{N_k\}^T}{dx} d\bar{x} \\
&= (\mathrm{krcp})(\ )_e \{\ \}_e^T (-1)[\mathrm{A211k}]\{\ \}_e
\end{aligned}
$$

$$
\begin{aligned}
\{\mathrm{SRCS}\}_e &\equiv -\int_{\Omega_e} \{N_k\}\{N_k\}^T d\bar{x} \{S\}_e \\
&= (-1)(\ )_e \{\ \}_e^T (1)[\mathrm{A200k}]\{S\}_e
\end{aligned}
$$

Identifying boundary terms:

$$[\text{HBC}]_e \equiv \left(\frac{h}{\rho c_p}\right)_e \{N_k\}\{N_k\}\Big|_{\partial\Omega_1}$$

$$\{\text{HTR}\}_e \equiv -\left(\frac{hT_r}{\rho c_p}\right)_e \{N_k\}\Big|_{\partial\Omega_1}$$

Assembling:

$$\boxed{GWS^h = [\text{MASS}]\frac{d\{Q\}}{dt} + [\text{CONVU}]\{Q\} + [\text{DIFFA}]\{Q\} + [\text{HBC}]\{Q\} + \{\text{SRCS}\} + \{\text{HTR}\} = \{0\}} \qquad (8)$$

where we now have a first order ODE *in time* along with our usual spatially discrete form.

Whipping out our Theta-Taylor Series to handle the time integration

$$\boxed{\Theta\text{TS} = [\text{MASS}]\{\Delta Q\}_{n,n+1} + \Delta t\left(\Theta\{\text{RES}\}_{n+1} + (1-\Theta)\{\text{RES}\}_n\right) = \{0\}} \qquad (9)$$

coupled with the Newton Iteration Algorithm (NIA):

$$\boxed{\begin{gathered}\{Q\}_{n+1}^{p+1} = \{Q\}_{n+1}^{p} + \{\delta Q\}_{n+1}^{p+1} \\ [\text{JAC}]_{n,n+1}^{p}\{\delta Q\}_{n+1}^{p+1} = -\{\text{FQ}\}_{n,n+1}^{p} \\ \{FQ\}_{n,n+1}^{p} = [\text{MASS}]\{\Delta Q\}_{n,n+1}^{p} + \Delta t\left(\Theta\{\text{RES}\}_{n+1}^{p} + (1-\Theta)\{\text{RES}\}_n\right) \\ [\text{JAC}]_{n+1}^{p} = \text{S}_e\left(\frac{\partial\{\text{FQ}\}_{n,n+1}}{\partial\{Q\}_{n+1}}\right)_e^p\end{gathered}} \qquad (10)$$

The final step is to evaluate the terms in the ΘTS (9) and NIA (10). Recalling the term $\{\text{RES}\}_{n,n+1}$ to be the spatial residual of the GWS[h] (2), it can be expressed syntactically as

$$\{\text{RES}\}_{n,n+1} = [\text{CONVU}]\{Q\}_{n,n+1} + [\text{DIFFA}]\{Q\}_{n,n+1} + [\text{HBC}]\{Q\}_{n,n+1} + \{\text{SRCS}\} + \{\text{HTR}\} = \{0\}$$

For the ΘTS and ultimately the NIA, we need to evaluate $[\text{MASS}]\{\Delta Q\}$ and $\{\text{RES}\}$ at t_n and t_{n+1}, hence using $\{Q\}_n$ and $\{Q\}_{n+1}$. The associated matlab syntax thus becomes, for constant coefficients and linear basis functions,

```
RES_MASS =  asres1d([],   [],[], 1,A200L,DELTA_Q);

RES_n =     asres1d(u,    [],[], 0,A201L,Q_n) + ...
            asres1d(krcp,[],[],-1,A211L,Q_n) + ...
            asres1d([],   [],[], 1,A200L,S)   + ...
            HBC_n + HTR;

RES_np1 =   asres1d(u,    [],[], 0,A201L,Q_np1) + ...
            asres1d(krcp,[],[],-1,A211L,Q_np1) + ...
            asres1d([],   [],[], 1,A200L,S)   + ...
            HBC_np1 + HTR;
```

where HBC and TR are the kronecker delta terms to handle the boundary conditions, effectively

$$[\text{HBC}]\{Q\}_{n,n+1} = \frac{h}{\rho c_p}\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & 0 \end{bmatrix}\begin{Bmatrix} Q_1 \\ \vdots \\ \vdots \\ Q_N \end{Bmatrix}_{n,n+1} \qquad \textit{and} \qquad \{\text{HTR}\} = \frac{hT_r}{\rho c_p}\begin{Bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{Bmatrix}$$

Note that `HBC` and H`TR` need to be evaluated *before* the RES lines as does `DELTA_Q=Q_np1-Qn`.

Having obtained all the terms in the ΘTS, we can now form the total residual FQ from (10):

```
FQ = RES_MASS + DELTA_t*(Theta*RES_np1 + (1-Theta)*RES_n);
```

The final step is to form $[\text{JAC}]_{n+1}$ from (10):

$$[\text{JAC}]_{n+1} = \frac{\partial\{\text{FQ}\}_{n,n+1}}{\partial\{Q\}_{n+1}} = \frac{\partial}{\partial\{Q\}_{n+1}}\left([\text{MASS}]\{\Delta Q\}_{n,n+1} + \Delta t\left(\Theta\{\text{RES}\}_{n+1} + (1-\Theta)\{\text{RES}\}_n\right)\right)$$

Noting that $\{\text{RES}\}_n$ is not a function of $\{Q\}_{n+1}$, the last term in $\{\text{FQ}\}_{n,n+1}$ readily goes to zero. Additionally, both $\{\text{SRCS}\}$ and $\{\text{HTR}\}$ are not functions of $\{Q\}_{n+1}$ and also go to zero. Finally, the $\{Q\}_n$ component of the mass term will also go to zero. Substituting in the syntactical form of the remaining terms and expanding:

$$[\text{JAC}]_{n+1} = [\text{MASS}]\frac{\partial\{Q\}_{n+1}}{\partial\{Q\}_{n+1}} + \Delta t\Theta\left([\text{CONVU}]\frac{\partial\{Q\}_{n+1}}{\partial\{Q\}_{n+1}} + [\text{DIFFA}]\frac{\partial\{Q\}_{n+1}}{\partial\{Q\}_{n+1}} + [\text{HBC}]\frac{\partial\{Q\}_{n+1}}{\partial\{Q\}_{n+1}}\right)$$

yielding

$$[\mathrm{JAC}]_{n+1} = [\mathrm{MASS}] + \Delta t\Theta([\mathrm{CONVU}] + [\mathrm{DIFFA}] + [\mathrm{HBC}])$$

The associated matlab syntax thus becomes, for constant coefficients,

```
JAC_MASS =  asjac1d([],  [],[], 1,A200L,[]);

JAC_np1 =   asjac1d(u,   [],[], 0,A201L,[]) + ...
            asjac1d(krcp,[],[],-1,A211L,[]) + ...
            HBC;

JAC = JAC_MASS + DELTA_t*Theta*JAC_np1;
```

Note that, for ***linear*** problem statements, you can readily obtain JAC from RES by differentiating the 6th data entry with respect to Q_np1.

Now iterate away until $\{Q\}_{n+1}$ has converged and then march off to the final time. Happy stepping!