TNSolver: An Open Source Thermal Network Solver for Octave or MATLAB®

Bob Cochran
Applied Computational Heat Transfer (CHT)
Seattle, WA
TNSolver@heattransfer.org

Thermal & Fluids Analysis Workshop (TFAWS) 2016
NASA Ames Research Center
Moffett Field, California
August 2, 2016

Outline

TNSolver: Heat Transfer Analysis using Thermal Networks

- Introduction to TNSolver
- Nodes
- Conductors
- Sources
- Material Properties
- Functions
- Results Output
- Example Problem

Introducing TNSolver

- Thermal Network Solver - TNSolver
- Octave program using MATLAB programming language
  - GNU Octave is an open source clone of MATLAB
- Thermal model is described in a text input file
  - Do not use a word processor, use a text editor, such as:
    - Cross-platform: vim/gvim, emacs, Bluefish, among many others
    - Windows: notepad, Notepad++
    - MacOS: TextEdit, Smultron
    - Linux: see cross-platform options
- Simulation results are both returned from the function and written to text output files for post-processing

TNSolver Web Site: http://www.heattransfer.org/TNSolver
TNSolver email: TNSolver@heattransfer.org

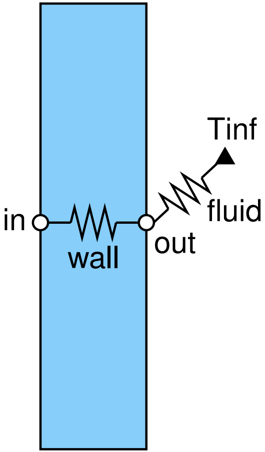
Thermal Network Terminology

- Time dependency
  - Steady state or transient
  - Initial condition is required for transient
- Geometry
  - Control Volume - volume, $V = \int_V dV$
    - Node: ●, $T_{\text{node}} = \int_V T(x_i)dV$
  - Control Volume Surface - area, $A = \int_A dA$
    - Surface Node: ○, $T_{\text{surface node}} = \int_A T(x_i)dA$
- Conductors
  - Conduction
  - Convection
  - Radiation
- Boundary conditions
  - Boundary node: ▲
- Material properties
- Sources/sinks

TNSolver Input Example of Text Input File

Any number and order of command blocks
is allowed in the input file

```
!  Simple Wall Model

Begin Solution Parameters
  type   = steady
End Solution Parameters

Begin Conductors
 wall   conduction in  out   2.3  1.2   1.0 ! k L A
 fluid convection out Tinf  2.3  1.0       ! h A
End Conductors

Begin Boundary Conditions
  fixed_T  21.0   in    ! Inner wall T
  fixed_T  5.0    Tinf  ! Fluid T
End Boundary Conditions
```



! begins a comment (Octave and MATLAB use %)

Nodes

Input file commands for nodes:

```
Begin Nodes

! label   material   volume
   (S)       (S)        (R)

! label   density*specific heat   volume
   (S)            (R)                (R)

End Nodes
```

Node labels are implicitly defined by conductors and only needed here if they have a finite volume.

Conductors

Heat flow rate, Q_{ij}, between nodes in the thermal network:

Conduction:

$$Q_{ij} = \frac{kA}{L}\left(T_i - T_j\right)$$

Convection:

$$Q_{ij} = hA\left(T_i - T_j\right)$$

Radiation:

$$Q_{ij} = \sigma \mathscr{F}_{ij} A_i (T_i^4 - T_j^4)$$

Advection:

$$Q_{ij} = \dot{m} c_P \left(T_i - T_j\right) = \rho u A c_P \left(T_i - T_j\right)$$

Conduction Conductors

Input file commands for conduction conductors:

```
Begin Conductors

! label    type         nd_i  nd_j  parameters
   (S)   conduction     (S)   (S)   (R)  (R)  (R)        ! k, L, A
   (S)   conduction     (S)   (S)   (S)  (R)  (R)        ! material, L, A
   (S)   cylindrical    (S)   (S)   (R)  (R)  (R)  (R)   ! k, ri, ro, L
   (S)   cylindrical    (S)   (S)   (S)  (R)  (R)  (R)   ! material, ri, ro, L
   (S)   spherical      (S)   (S)   (R)  (R)  (R)        ! k, ri, ro
   (S)   spherical      (S)   (S)   (S)  (R)  (R)        ! material, ri, ro

End Conductors
```

Convection Conductors

Requires a convection heat transfer coefficient, h

Three methods of specifying h:

1. h is specified
2. h is evaluated using a correlation
3. h is evaluated using a user written function

Convection Conductor Commands

```
Begin Conductors

! label  type        node i node j  parameters
   (S)  convection     (S)    (S)   (R)  (R)             ! h, A
   (S)  IFCduct        (S)    (S)   (S)  (R)  (R)  (R)   ! material, velocity, Dh, A
   (S)  EFCcyl         (S)    (S)   (S)  (R)  (R)  (R)   ! material, velocity, D, A
   (S)  EFCdiamond     (S)    (S)   (S)  (R)  (R)  (R)   ! material, velocity, D, A
   (S)  EFCimpjet      (S)    (S)   (S)  (R)  (R)  (R)  (R) ! material, velocity,
                                                            ! D, H, r
   (S)  EFCplate       (S)    (S)   (S)  (R)  (R)  (R)  (R) ! material, velocity,
                                                            ! Xbegin, Xend, A
   (S)  EFCsphere      (S)    (S)   (S)  (R)  (R)        ! material, velocity, D
   (S)  INCvenc        (S)    (S)   (S)  (R)  (R)  (R)   ! material, W, H, A
   (S)  ENChcyl        (S)    (S)   (S)  (R)  (R)        ! material, D, A
   (S)  ENChplatedown  (S)    (S)   (S)  (R)  (R)        ! material, L=A/P, A
   (S)  ENChplateup    (S)    (S)   (S)  (R)  (R)        ! material, L=A/P, A
   (S)  ENCiplatedown  (S)    (S)   (S)  (R)  (R)  (R)  (R) ! material, H, L=A/P,
                                                            ! angle, A
   (S)  ENCiplateup    (S)    (S)   (S)  (R)  (R)  (R)  (R) ! material, H, L=A/P,
                                                            ! angle, A
   (S)  ENCsphere      (S)    (S)   (S)  (R)             ! material, D
   (S)  ENCvplate      (S)    (S)   (S)  (R)  (R)        ! material, L, A
   (S)  FCuser         (S)    (S)   (S)  (S)  (R ...)  (R) ! function, material,
                                                            ! parameters, A
   (S)  NCuser         (S)    (S)   (S)  (S)  (R ...)  (R) ! function, material,
                                                            ! parameters, A

End Conductors
```

Forced Convection Correlations Conductors

Geometry	Conductor	Flow Range
Internal Forced Convection (`IFC`)		
pipe/duct	`IFCduct`	$0 \lesssim Re \lesssim 5 \times 10^6$
External Forced Convection (`EFC`)		
cylinder	`EFCcyl`	$0.4 \lesssim Re \lesssim 400,000$
diamond/square	`EFCdiamond`	$6,000 \lesssim Re \lesssim 60,000$
impinging round jet	`EFCimpjet`	$6,000 \lesssim Re \lesssim 60,000$
flat plate	`EFCplate`	$0 \lesssim Re \lesssim 10^8$
sphere	`EFCsphere`	$3.5 \lesssim Re \lesssim 7.6 \times 10^4$

Natural Convection Correlations Conductors

Geometry	Conductor	Flow Range
Internal Natural Convection (`INC`)		
rectangular enclosure	`INCvenc`	$10^3 \lesssim Ra \lesssim 10^{10}$
External Natural Convection (`ENC`)		
horizontal cylinder	`ENChcyl`	$Ra \lesssim 10^{12}$
horizontal plate facing down	`ENChplatedown`	$10^4 \lesssim Ra \lesssim 10^{11}$
horizontal plate facing up	`ENChplateup`	$10^4 \lesssim Ra \lesssim 10^{11}$
inclined plate facing down	`ENCiplatedown`	$Ra \lesssim 10^{11}$
inclined plate facing up	`ENCiplateup`	$Ra \lesssim 10^{11}$
sphere	`ENCsphere`	$Ra \lesssim 10^{11}$
vertical flat plate	`ENCvplate`	$Ra \lesssim 10^{12}$

Enclosure Radiation: The Exchange Factor, $\mathscr{F}$

The exchange factor concept is based on proposing that there is a parameter, $\mathscr{F}_{ij}$, based on surface properties and enclosure geometry, that determines the radiative heat exchange between two surfaces [Hot54, HS67] :

$$Q_{ij} = A_i \mathscr{F}_{ij} \sigma (T_i^4 - T_j^4)$$

$\mathscr{F}$ is known by many names in the literature: script-F, gray body configuration factor, transfer factor and Hottel called it the over-all interchange factor.

For an enclosure with N surfaces, the net heat flow rate, Q_i, for surface i, is:

$$Q_i = \sum_{j=1}^{N} A_i \mathscr{F}_{ij} \left(\sigma T_i^4 - \sigma T_j^4 \right) = \sum_{j=1}^{N} A_i \mathscr{F}_{ij} \left(E_{bi} - E_{bj} \right)$$

Radiation Conductor Commands

```
Begin Conductors

! label   type       node i node j  parameters
   (S)   surfrad       (S)    (S)   (R)  (R)          ! emissivity, A
   (S)   radiation     (S)    (S)   (R)  (R)          ! script-F, A

End Conductors
```

Radiation conductors for an enclosure can be generated using the radiation enclosure block:

```
Begin Radiation Enclosure

! label emiss   area   view factors
   (S)    (R)    (R)    (R ...)

End Radiation Enclosure
```

The radiation conductors are given in the output file

Advection Conductors

- Advective transport of energy is modeled using advection conductors
- Note that an outflow boundary requires the outflow conductor

```
Begin Conductors

! label    type         nd_i  nd_j  parameters
   (S)   advection      (S)    (S)   (S)  (R)  (R)      ! material, velocity, A
   (S)   outflow        (S)    (S)   (S)  (R)  (R)      ! material, velocity, A

End Conductors
```

Source Terms

- Sources are applied to nodes
- Define the node volume in the Nodes command block
- There are currently three types of sources:
  1) per volume, $\dot{q}$, 2) total, Q, 3) thermostat heater

```
Begin Sources

! type   parameter(s)     node(s)
  qdot   (R)               (S ...)  ! q̇, uses node volume: Q = q̇V
  Qsrc   (R)               (S ...)  ! Q
  tstatQ (R)  (S)  (R)  (R) (S ...)  ! Q, thermostat node, Ton, Toff

End Sources
```

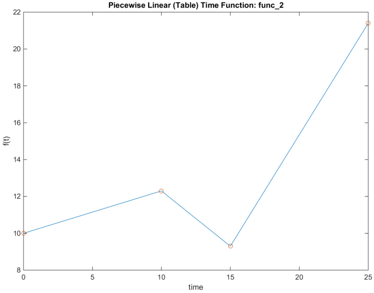
Material Properties

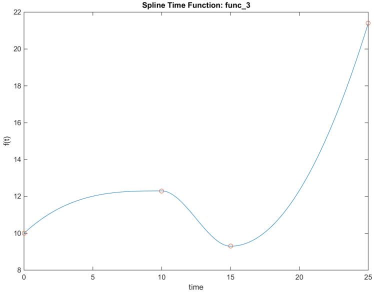
- Built-in material library
  - You can add your own materials to the function
- Define material properties in the input file:

```
Begin Material (S)

  state = (S)          ! {gas|liquid|solid}
  density              ! {= (R)|table|spline|polynomial|ideal gas}
  conductivity         ! {= (R)|table|spline|polynomial}
  specific heat        ! {= (R)|table|spline|polynomial}
  c_p                  ! {= (R)|table|spline|polynomial}
  c_v                  ! {= (R)|table|spline|polynomial}
  viscosity            ! {= (R)|table|spline|polynomial}
  beta                 ! {= (R)|table|spline|polynomial|ideal gas}
  Pr                   ! {= (R)|table|spline|polynomial}
  gas constant = (R)   ! gas constant for use with ideal gas
  reference = (S ...)

End Material (S)
```

Functions

- Time dependent functions can be defined and used for parameters in the input file
- Four types of functions:
  - 1) Constant, 2) Piecewise Linear (table), 3) Piecewise cubic Hermite interpolating polynomial (spline) 4) Polynomial with range



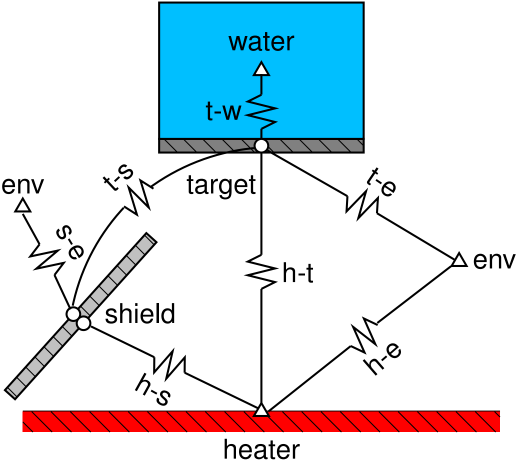


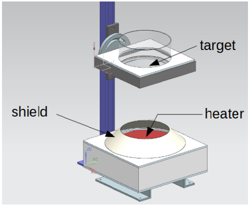
Results Output from TNSolver

- The input file uses .inp extension: `inpfile.inp`
- Function return:
  - `[T, Q, nd, el] = tnsolver('inpfile');`
- Text output is written to `inpfile.out`
- CSV data files are written to `inpfile_cond.csv` and `inpfile_nd.csv`
- Transient data is written to `inpfile_timedata.csv`
- Restart data is written to `inpfile.rst`
- If requested, graphviz dot file is written to `inpfile.gv`
  - Produce PDF image:
    - `dot -Tpdf inpfile.gv -o inpfile.pdf`

TNSolver Thermal Network Model Example

Radiation Heat Transfer Experiment from University of Washington ME 331 Introduction to Heat Transfer Class





Example Input File

TNSolver Input File

```
Begin Solution Parameters

  title = Radiation Heat Transfer Experiment - Black Target
  type                         = steady
  nonlinear convergence        = 1.0e-8
  maximum nonlinear iterations = 50

End Solution Parameters

Begin Conductors

! Conduction through the beaker wall, 0.1" thick pyrex glass
     t-bbin  conduction targ bbin  0.14 0.00254 0.024829 ! k L A
! Convection from beaker to water
! label    type        nd_i   nd_j    mat     L      A
  t-w  ENChplateup   bbin   water  water 0.04445 0.02483
! Convection from target to air
! label    type         nd_i   nd_j    mat     L      A
  t-air ENChplatedown targ  env    air  0.0889 0.0248
! Convection from outer shield to air
! label    type         nd_i   nd_j  mat    L   theta   A
  s-air  ENCiplateup  s_out  env  air 0.0508 48.0 0.056439
! Conduction from inner to outer side of shield
  shield  conduction  s_in s_out  steel  0.001 0.056439

End Conductors
```

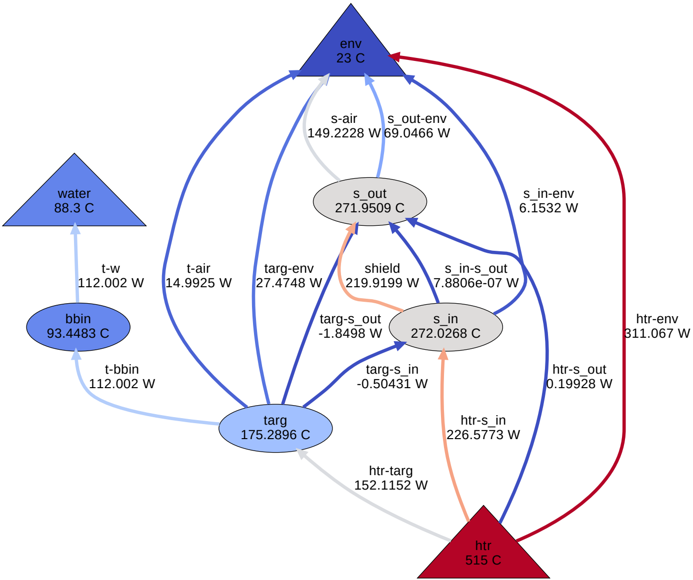
Example Input File (continued)

TNSolver Input File

```
Begin Radiation Enclosure

! surf emiss  A       Fij
 htr   0.92 0.06701 0.0     0.1264  0.68415 0.0    0.1893
 targ  0.95 0.02482 0.34132 0.0     0.00603 0.1031 0.5494
 s_in  0.28 0.05643 0.81231 0.00265 0.12278 0.0    0.0622
 s_out 0.28 0.05643 0.0     0.0453  0.0     0.0    0.9546
 env   1.0  0.1184  0.10711 0.1151  0.02965 0.4547 0.2933

End Radiation Enclosure

Begin Boundary Conditions

!  type        Tb      Node(s)
  fixed_T     23.0      env
  fixed_T     88.3      water
  fixed_T    515.0      htr

End Boundary Conditions
```

Thermal Network Solution

Results Visualized with graphviz dot



Conclusion

- Introduced TNSolver, an open source thermal network solver for Octave and MATLAB
- Provided an overview of the functionality
  - see User Manual for the details
- Example problem showed use of convection correlations and radiation heat transfer in an enclosure

- What is Next in the Development Plan?
- Sensitivity analysis, local and global
- Uncertainty quantification
  - See Dakota at Sandia National Labs
  - See Reliability Engineering at C&R Sinda

Acknowledgments

- Leigh Sedgwick, The Boeing Company (retired)
- John Gray, The Boeing Company
- Professor Ashley Emery, University of Washington
  - and the students of ME 331 Introduction to Heat Transfer, Autumn Quarter 2014 and Autumn Quarter 2015

Appendix

- GNU Octave
  - `http://www.gnu.org/software/octave/`
- Octave Wiki
  - `http://wiki.octave.org`
- Octave-Forge Packages (similar to MATLAB Toolbox packages)
  - `http://octave.sourceforge.net`
- Windows Installation
  - Binaries are at: `https://ftp.gnu.org/gnu/octave/windows/`
  - As of August 1, 2016, the latest version of Octave is 4.0.3
  - Download the octave-4.0.3.zip file and unzip in a Windows folder

Time Integration for Transient Problems

TNSolver Verification

Backward Euler time integration is used in TNSolver.
How does time step affect accuracy?
Utilitize the analytical solution Equation (5.6), p. 282 in [BLID11]:

$$\frac{T - T_\infty}{T_i - T_\infty} = \exp\left[-\left(\frac{hA}{\rho c V}\right)t\right]$$

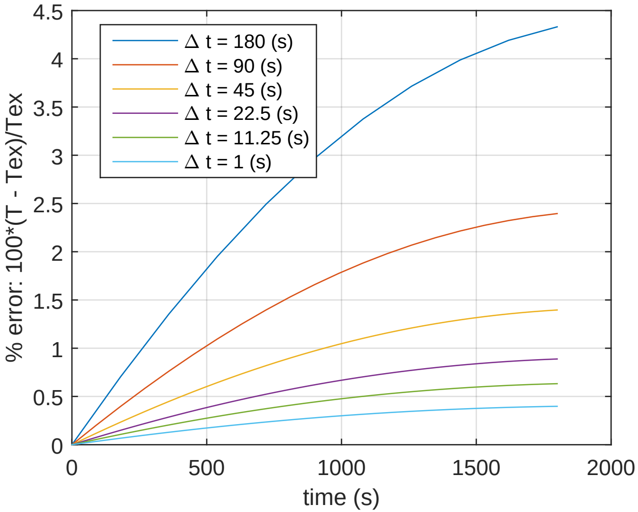
This is provided in the MATLAB function `lumpedmass.m`:

```
[T, Bi] = lumpedmass(time, rho, c, V, h, A, Ti, Tinf, k)
```

Example calculation using:
$D = 0.04931\ m$, $T_i = 100\ C$, $T_\infty = 25\ C$
$\rho = 7850\ kg/m^3$, $c = 500\ J/kg \cdot K$
$h = 25.0\ W/m^2 \cdot K$, $k = 62.0\ W/m \cdot K$

Time Step Error Study

TNSolver Verification



Exchange Factor, $\mathscr{F}$, Properties

Reciprocity:

$$A_i \mathscr{F}_{ij} = A_j \mathscr{F}_{ji}$$

Summation:

$$\sum_{j=1}^{N} \mathscr{F}_{ij} = \epsilon_i$$

Matrix Form of Enclosure Radiation with $\mathscr{F}$

$$Q_i = \sum_{j=1}^{N} A_i \mathscr{F}_{ij} \left(E_{bi} - E_{bj} \right)$$

$$\{Q\} = [A] \left([\epsilon] - [\mathscr{F}] \right) \{E_b\}$$

$$\{q\} = [A]^{-1} \{Q\} = \left([\epsilon] - [\mathscr{F}] \right) \{E_b\}$$

where, for $N = 3$:

$$\{Q\} = \begin{Bmatrix} Q_1 \\ Q_2 \\ Q_3 \end{Bmatrix} \quad [A] = \begin{bmatrix} A_1 & 0 & 0 \\ 0 & A_2 & 0 \\ 0 & 0 & A_3 \end{bmatrix}$$

$$[\mathscr{F}] = \begin{bmatrix} \mathscr{F}_{11} & \mathscr{F}_{12} & \mathscr{F}_{13} \\ \mathscr{F}_{21} & \mathscr{F}_{22} & \mathscr{F}_{23} \\ \mathscr{F}_{31} & \mathscr{F}_{32} & \mathscr{F}_{33} \end{bmatrix} \quad \{E_b\} = \begin{Bmatrix} \sigma T_1^4 \\ \sigma T_2^4 \\ \sigma T_3^4 \end{Bmatrix} \quad [\epsilon] = \begin{bmatrix} \epsilon_1 & 0 & 0 \\ 0 & \epsilon_2 & 0 \\ 0 & 0 & \epsilon_3 \end{bmatrix}$$

Exchange Factors, $\mathscr{F}$, from View Factors, F

The net heat flux using view factors, F_{ij}, is:

$$\{q\} = [\epsilon]([I] - [F][\rho])^{-1}([I] - [F])\{E_b\}$$

The net heat flux using exchange factors, $\mathscr{F}_{ij}$, is:

$$\{q\} = ([\epsilon] - [\mathscr{F}])\,\{E_b\}$$

The two enclosure heat fluxes are equal, so equating gives:

$$([\epsilon] - [\mathscr{F}])\,\{E_b\} = [\epsilon]([I] - [F][\rho])^{-1}([I] - [F])\{E_b\}$$

$$([\epsilon] - [\mathscr{F}]) = [\epsilon]([I] - [F][\rho])^{-1}([I] - [F])$$

$$[\mathscr{F}] = [\epsilon]\left([I] - ([I] - [F][\rho])^{-1}([I] - [F])\right)$$

See [IB63] for an early reference to this method.

Linearization of Radiation Conductors

The temperature is linearized using a two term Taylor series expansion about the previous iteration temperature, T^*:

$$T_i^4 \approx (T_i^*)^4 + (T_i - T_i^*)\,4(T_i^*)^3$$

$$T_i^4 \approx 4(T_i^*)^3 T_i - 3(T_i^*)^4$$

$$T_j^4 \approx \left(T_j^*\right)^4 + \left(T_j - T_j^*\right) 4\left(T_j^*\right)^3$$

$$T_j^4 \approx 4\left(T_j^*\right)^3 T_j - 3\left(T_j^*\right)^4$$

The linearized form of the heat transfer rate is:

$$Q_{ij} = \sigma \mathscr{F}_{ij} A_i \left[4(T_i^*)^3 T_i - 3(T_i^*)^4 - 4\left(T_j^*\right)^3 T_j + 3\left(T_j^*\right)^4\right]$$

$$Q_{ij} = \sigma \mathscr{F}_{ij} A_i \left\{4(T_i^*)^3 T_i - 4\left(T_j^*\right)^3 T_j\right\} - \sigma \mathscr{F}_{ij} A_i \left\{3(T_i^*)^4 - 3\left(T_j^*\right)^4\right\}$$

References I

[BLID11] T.L. Bergman, A.S. Lavine, F.P. Incropera, and D.P. DeWitt.
Introduction to Heat Transfer.
John Wiley & Sons, New York, sixth edition, 2011.

[Hot54] H. C. Hottel.
Radiant-heat transmission.
In *Heat Transmission* [McA54], chapter 4, pages 55–125.

[HS67] H. C. Hottel and A. F. Sarofim.
Radiative Transfer.
McGraw-Hill, New York, 1967.

References II

[IB63] T. Ishimoto and J. T. Bevans.
Method of evaluating script f for radiant exchange within an enclosure.
AIAA Journal, 1(6):1428–1429, 1963.

[LL12] J. H. Lienhard, IV and J. H. Lienhard, V.
A Heat Transfer Textbook.
Phlogiston Press, Cambridge, Massachusetts, fourth edition, 2012.
Available at: `http://ahtt.mit.edu`.

[McA54] W. H. McAdams.
Heat Transmission.
McGraw-Hill, New York, third edition, 1954.