ERDC/EL TR-06-2

Environmental Laboratory

US Army Corps of Engineers®
Engineer Research and Development Center

Countermine Phenomenology Program

1-D Thermal Modeling of Layered Materials in Outdoor Environments

John O. Curtis

February 2006

Countermine Phenomenology Program

ERDC/EL TR-06-2
February 2006

1-D Thermal Modeling of Layered Materials in Outdoor Environments

John O. Curtis

Environmental Laboratory
U.S. Army Engineer Research and Development Center
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

Final report

Approved for public release; distribution is unlimited

Prepared for U.S. Army Corps of Engineers
Washington, DC 20314-1000

ABSTRACT: This report describes a physics-based, but relatively simple, thermal infrared computational tool that can be used in a personal computer (PC) environment to predict temperature profiles of layered media. The tool is a one-dimensional finite difference simulation code (written in FORTRAN) that is executed through a graphical user interface. Its current utility is in helping to design field tests of airborne mine detection sensor systems and to analyze the results of those tests to better understand the performance of the sensor in different environmental conditions. The tool does not address the separate issues of two- and three-dimensional effects, sensor hardware performance, sensor flight paths, or targeting algorithms.

Contents

List of Figures

Preface

This report was prepared as part of the U.S. Army Engineer Research and Development Center (ERDC) Countermine Phenomenology Program, which supports the U.S. Army Research, Development, and Engineering Command (RDECOM) Communications and Electronics Research, Development, and Engineering Center (CERDEC) Night Vision and Electronic Sensors Directorate (NVESD) mine detection sensor test and development programs. The NVESD Technical Monitor for this effort was Dr. Tom Broach, of the NVESD Countermine Division, located at Fort Belvoir, VA. Dr. Larry Lynch, ERDC, Geotechnical and Structures Laboratory, was the manager of the Countermine and Phenomenology Program.

Dr. John Curtis, Environmental Systems Branch (EE-C), Ecosystem Evaluation and Engineering Division (EE), Environmental Laboratory (EL), ERDC, in Vicksburg, MS, conducted this study under the direct supervision of Dr. Rose Kress, Chief of EE-C and Mr. Bruce Sabol, Acting Chief of EE-C and the general supervision of Dr. Dave Tazik, Chief of EE. Dr. Ed Theriot was Director of EL, and Dr. Beth Fleming was Acting Director of EL.

Commander and Executive Director of ERDC was COL James R. Rowan, EN. Director was Dr. James R. Houston.

1 Introduction

Background

The U.S. Army Engineer Research and Development Center (ERDC) in Vicksburg, MS, has developed a Countermine Phenomenology Program (CPP) to support the U.S. Army Research, Development, and Engineering Command (RDECOM) Communications and Electronics Research, Development, and Engineering Center (CERDEC) Night Vision and Electronic Sensors Directorate (NVESD) mine detection sensor test and development efforts. One of the issues being addressed by the CPP is that airborne sensors that attempt to identify ground targets often suffer from unexpected high false alarm rates. ERDC researchers have demonstrated in numerous earlier studies that environmental factors often generate target-like signatures. This is true in both the thermal infrared and radar portions of the electromagnetic spectrum.

Modeling of ground target signatures has historically focused on just the targets themselves, or targets embedded in statistically noisy backgrounds. Little, if any, effort has been made to include realistic natural terrain background signatures in the design and analysis of target detection sensors and algorithms. ERDC believes that physics-based terrain element models need to be included in computational platforms that are capable of modeling the complete sensor detection process. This includes target signatures, background (natural terrain) signatures, atmospheric attenuation of those signatures, sensor hardware and flight path, and targeting algorithms.

A fundamental knowledge of the character of natural terrain and the dynamic processes that alter the properties of the terrain (predominantly season, time-of-day, and weather) are key to the success of the CPP. These models will provide a significant improvement over the current method of treating natural environments as statistical clutter. Instead, the specific geometric and material properties of the terrain can be considered and exploited by the sensor system and algorithm developers.

Objective

The primary objective of this study is to assemble a physics-based, but relatively simple, thermal infrared computational tool that can be used in a personal computer (PC) environment to help design field tests of airborne mine detection sensor systems and to analyze the results of those tests to better understand the performance of the sensor in different environmental conditions. In particular, this study focuses on a one-dimensional (1-D) layered media simulation code that will yield first-order understandings of target and background signatures. It does not address the separate issues of two- and three-dimensional effects, sensor hardware performance, sensor flight paths, or targeting algorithms. Those are left for a much more sophisticated computational platform that is currently being developed at ERDC.

The TSTM/VEGIE Thermal Model

TSTM

In 1981, two reports were published at ERDC, known then as the Waterways Experiment Station (WES), which dealt with a one-dimensional thermal model for predicting surface temperatures of natural terrain elements. The first of these, entitled "Thermal Modeling of Terrain Surface Elements" (Balick et al. 1981a), described a code named the Terrain Surface Temperature Model (TSTM), that simulated non-vegetation-covered surfaces such as bare ground or concrete slabs and their response to variable weather conditions. The basic assumption of the TSTM model was that each of the layers forming the structure, as well as the environment above the structure, was horizontally uniform. In other words, the only significant heat fluxes would be vertical. Under these conditions, physical temperatures within the structure can be found by solving the one-dimensional heat flow equation:

$$\frac{dT(z,t)}{dt} = \alpha(z)\frac{\partial^2 T(z,t)}{\partial z^2} \tag{1}$$

where T is the physical temperature of some point at a depth z below the surface at time t. The thermal diffusivity of the material at that depth $\alpha(z)$ is defined as the ratio of the thermal conductivity of the material to the product of the mass density and specific heat of the material:

$$\alpha(z) = \frac{\kappa(z)}{\rho(z)c(z)} \tag{2}$$

Clearly, thermal diffusivity of a material measures the rate at which a change in temperature spreads through that material (Jumikis 1977).

A TSTM simulation is driven by air temperature, solar heat flux, and wind speed variations throughout the course of a day. The surface temperature is controlled by an energy balance that will be discussed in a later section.

VEGIE

The second WES report dealt with a modification of TSTM that is described in its title: "Inclusion of a Simple Vegetation Layer in Terrain Temperature Models for Thermal Infrared (IR) Signature Prediction," (Balick et al. 1981b). VEGIE, as the new model was named, simply added a layer of vegetative material to the bare material through the inclusion of several new input parameters including the foliage cover fraction, an index that characterized the state of the vegetation, the graybody emissivity and solar absorptivity of the foliage, and the foliage height. The same kind of energy balance performed in TSTM is required at both the vegetation surface and the ground surface.

Previous Applications

TSTM/VEGIE and its many variants have been used extensively in numerous ERDC applications. Early attention focused on simulations to support the ERDC mission of fixed facility camouflage, and publications include the two already referenced. Later the code was adapted to another major ERDC research effort, the Smart Weapon Operability Enhancement (SWOE) Program, and used to generate thermal IR images of targets in natural background settings (Welsh 1994).

Code Modifications

In its original form, the TSTM/VEGIE model could be executed only on a mainframe computer. Variants of the model, used in a number of unpublished ERDC studies, were later installed on workstations and, finally, on PCs. However, none of these model variants could be called “user-friendly.” They were developed for single users and for specialized applications. One goal of this project, then, was to deliver a “user-friendly” version of the TSTM/VEGIE model that operates in a PC environment and is readily transportable from one platform to another. Data input and execution of the code were simplified through the development of a graphical user interface (GUI). Simulation results can now be readily viewed as Excel charts generated at the same time that the simulation takes place. No separate data analysis needs to be performed.

Another limitation of the original TSTM/VEGIE model was that it simulated only one diurnal cycle, utilizing an input data file that was manually created by the user. Therefore, another goal of this study was to conduct multiple-day simulations using input data files that are primarily derived from field weather station micrologger digital files. A detailed description of how to generate those files follows in a later section.

Existing TSTM/VEGIE model variants were limited to constant value thermal properties for each soil layer and constant value optical properties for the surface layer. However, thermal properties in real materials are not constant values. Among other things, they are certainly a function of moisture content (Ochsner et al. 2001). Clearly, over a period of many weeks, soil moisture conditions can change dramatically. It makes no sense to use single-valued thermal properties of soils to conduct a meaningful simulation of conditions at a test site if soil conditions change significantly during that time. Therefore, this new version of TSTM/VEGIE must allow for moisture-dependent thermal properties.

2 Basic Principles of the 1-D Thermal Model

Model Geometry

Figure 1 is a visual representation of the TSTM model mode of operation. Although currently limited to six layers by the dimension statements of the code, theoretically any number of layers of material can be represented by a grid of nodes (equally spaced within each layer) whose spacings and properties are used to solve the finite difference form of Equation 1. The energy balance at the surface and the technique used to solve for temperatures at layer interfaces are discussed in the following sections.

Energy Budget Terms at the Air Interface

The surface temperature of the simulated material is found at each increment of time by an energy balance that can be written in equation form as:

$$S + I - H - E - X + G = 0.0 \tag{2}$$

or

$$D - X + G = 0.0 \tag{3}$$

where

S = net direct short-wave solar radiative flux density, or insolation, received at the air/solid interface

I = net long-wave irradiance (energy flux density impinging on the air/solid interface) from the sky and clouds

H = sensible heat exchange at the surface (primarily convective)

E = latent heat exchange at the surface (primarily evaporative)

X = graybody emittance (energy flux density radiating from the air/solid interface) due to the physical temperature of the surface

G = energy flux density into the solid surface due to conduction

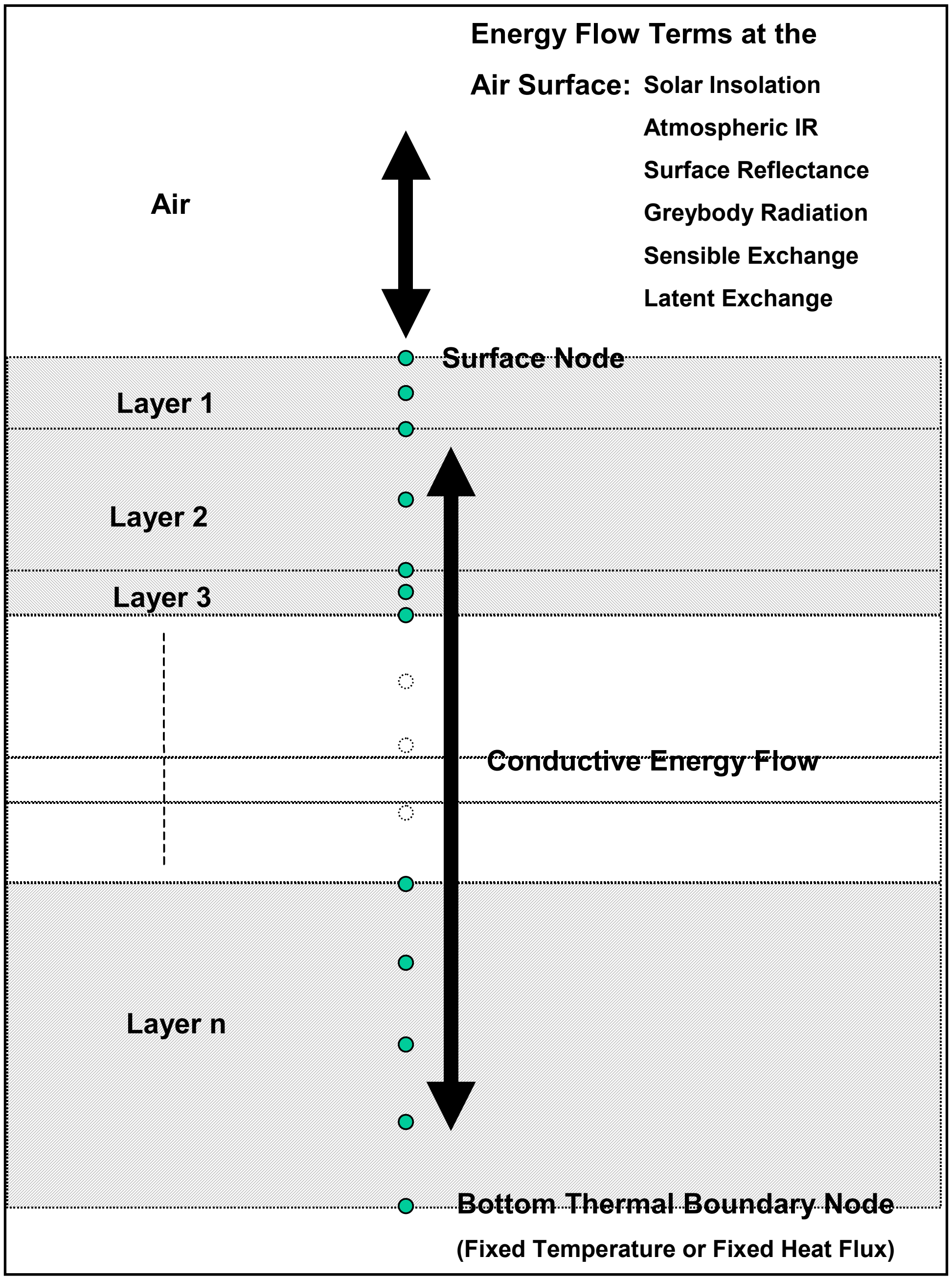

Figure 1. Geometry of the TSTM model

“Short-wave” and “long-wave” are terms used by atmospheric scientists for radiation energy in the 0.15-3.0 micron and 3.0-100 micron wavelength regions, respectively, of the electromagnetic spectrum (Oke 1987). *I*, *H*, and *E* are all calculated using empirical relationships described in the original TSTM report (Balick et al. 1981a). When short-wave insolation *S* is not available as measured data, another empirical relationship referenced in the same report can be used to compute idealized data. This latter technique utilizes the day of the year and the latitude of the test site as controlling factors.

The graybody emittance X is calculated at each time-step within the simulation using the simple relationship:

$$X = \varepsilon_S \sigma \left(T_S \right)^4 \tag{4}$$

where

ε_s = emissivity of the surface

σ = Stephan-Boltzman constant

T_s = current surface temperature predicted by the model

All that remains to define, prior to discussing the numerical solution technique used for these simulations, is the conductive energy flux density G:

$$G = k\left(\frac{\partial T}{\partial z} \right) \quad k\frac{\left(T_1 - T_S \right)}{\Delta z} \tag{5}$$

where

k = thermal conductivity of the surface layer

T_1 = temperature of the first node below the surface

Δz = spacing between the surface node and the first node below the surface

Iterative Finite Difference Solution for Surface Temperatures

Combining Equations 3, 4, and 5, and rearranging terms, one finds that the surface temperature is the root of the following equation:

$$F\left(T_S \right) = T_S^4 + \frac{k}{\varepsilon_S \sigma \Delta z}\left(T_S - T_1 \right) - \frac{D}{\varepsilon_S \sigma} = 0.0 \tag{6}$$

Newton's method is used to find that root:

$$F\left(T_S \right)_{new} = F\left(T_S \right)_{old} + \left(\frac{\partial F\left(T_S \right)}{\partial T_S} \right)_{old} dT_S \tag{7}$$

which provides an estimate for a new surface temperature (by setting $F(T_S)_{new} = 0$):

$$\left(T_S\right)_{new} = \left(T_S\right)_{old} - \frac{F\left(T_S\right)_{old}}{\left(\frac{\partial F\left(T_S\right)}{\partial T_S}\right)_{old}} \tag{8}$$

The partial derivative of $F(T_S)$ comes directly from Equation 6, in which the partial of D with respect to T_S is approximated using the previous and current estimates of the surface temperature.

Finite Difference Solution for Energy Flow Within Layers

For material within each layer of the simulated structure, a central difference form of Equation 1 is used to calculate a new value of temperature at each node n:

$$(T_n)_{new} = (T_n)_{old} + \frac{\alpha \Delta t}{\Delta z^2}\left[(T_{n+1})_{old} - 2(T_n)_{old} + (T_{n-1})_{old}\right] \tag{9}$$

where the "n+1" and the "n-1" subscripts refer to the nodes immediately below and above the node of interest, respectively.

Finite Difference Solution for Energy Flow Through Layer Interfaces

The developers of TSTM combined a truncated Taylor series for the temperature of the node adjacent to each side of an interface and the 1-D heat flow equation (Equation 1) to derive a difference expression for the partial derivative of temperature with respect to depth at that interface node looking at the interface from each of the layer materials. Each expression included an estimate for the new interface temperature. Those derivatives multiplied by the thermal conductivity of each layer resulted in two expressions for the conductive heat flux through the interface, one for each of the two adjoining materials. Assuming continuity of the heat flux through the interface, setting the two expressions equal to each other resulted in a lengthy finite difference expression for the interface temperature at the end of the time-step. Details of this derivation can be found in the original TSTM report (Balick et al. 1981a).

Bottom Boundary Conditions

For all of the simulations conducted during this study, a fixed temperature was chosen as the bottom boundary condition. Selecting either of the other two boundary condition options (a constant heat flux or a constant heat flux combined with a constant temperature radiating surface) results in a finite difference expression that must be evaluated at each time increment. Details can be found in the original TSTM report (Balick et al. 1981a).

3 Executing the 1-D Thermal Code in a PC Environment

Hardware and Software Requirements

TSTM/VEGIE is a Fortran code of reasonable size by today's standards. On the author's PC, the source code occupies 45 kilobytes (KB) of disk space, while the executable code occupies only 105 KB of space. In addition to the source code, a user will need a Fortran compiler to facilitate any necessary changes to the code and an Excel spreadsheet software package to execute the GUI and provide visual simulation results in chart and tabular form. The final element of the simulation tools is the input data file, which is described in detail in the next section.

Included on the CD that accompanies this report is a folder labeled "MinGW" that contains the freeware Fortran 77 compiler and other necessary files that were used by the author to conduct the simulations that follow in the next chapter. The "tstm_files" folder contains the source code, named "tstmforgui.f" (listed in Appendix A), as well as an example input data file ("flw2004 soil.csv," listed in Appendix B) and the resulting output data file ("fort.4").

If it is necessary to make changes to the source code, the user is advised to save a copy of the original source code in a safe place before proceeding. Once that is done, the source code can be opened in any word processing window and the necessary changes made and saved. Then the code must be recompiled. That requires operating in a disk operating system (DOS) command mode. The author's Window's-based PC has a command prompt that opens a DOS window. Once there, the directory needs to be changed to that containing the source code; i.e., by entering the command:

```
cd\tstm_files
```

The source code is compiled and stored as an executable file, named "tstmforgui.exe," in the same folder by entering the command:

```
g77 tstmforgui.f -o tstmforgui
```

Simulation run times will depend upon the PC's speed and memory capabilities as well as the number of nodes simulating the structure and the number of time increments for the simulation. The author's PC has a 2.536-GHz Pentium®4 central processing unit (CPU) and 768 megabytes (MB) of random access memory (RAM). The longest simulation, for which results are shown in the next chapter, included 110 nodes to simulate a land mine over soil and utilized a time increment of 0.0008 minutes. The simulation covered 64 days of weather data and (including 3 days of iterations to achieve simulation stability, took about 24 minutes and 50 seconds of CPU time while occupying 1.7 MB of memory (determined by the size of the code and the size of the specified arrays). Using these numbers to gauge the length of other simulations, one could say that simulation run times should be on the order of 0.11 microseconds/time increment/node.

Input Data File Creation

Appendix B contains a partial listing of an input data file used to perform one of the simulations described in the next chapter. Line numbers printed on those pages are not part of the data file; they have been added to facilitate the writing of this section.

Input data begins with a single-line description of the simulation. How it reads is the user's choice, but in most cases it will be a description of the test site being simulated. In this case, line 1 reads: "midwestern.test.site.2004." The dots between words facilitate handling of the title within the Excel spreadsheet. The number "11556" was added by a previous execution of the GUI and is not necessary for conducting a simulation.

The bulk of the input file comes from weather station data measured at the test site. Lines 2-27 on page B1 represent only a small portion of field measurement data, the first few lines and the last few lines. This particular file actually contains 11,656 lines of field data. To complete the input data file, the user needs to add a line at the end (line 28) that contains the word "End." It is used by the code to delineate the number of entries and to free the user from counting all of the lines of input data. A few columns of data have to be derived by the user. They will be described in the following paragraphs.

The entries shown on lines 29-46 of Appendix B are not needed to perform a simulation. They represent test site characterization data that has entered into the GUI prior to executing TSTM/VEGIE. These lines were then appended to the original input data file through execution of the GUI. Their only function is to populate data boxes on the GUI when a file is accessed that has been used before. This precludes the necessity of entering all of the GUI data entries by hand each time a new simulation is performed. As the listing on page B1 shows, data file entries must be comma separated (the GUI looks for a comma-separated variable (.csv) input file).

As noted above, lines 2 through 27 in Appendix B are a partial listing of the field measurement data required to execute this code. Most of these data can be collected at a test site weather station and recorded on a field micrologger. These data will ordinarily be delivered to the user as a spreadsheet or database file (e.g., Excel). It is relatively easy to delete unnecessary columns of data and to add other columns of required data while still in the spreadsheet format.

Field measurement (and complementary) data shown in lines 2 through 27 include the following parameters for each line of data:

Column 1 Julian day on which the following data were collected

Column 2 Hour of the day (24-hour clock) on which the following data were collected

Column 3 Air temperature (deg C) at a known height above the ground

Column 4 Relative humidity (percent)

Column 5 Barometric pressure (millibars). The -6999 entries on lines 21-27 dictate to the code that the pressure gauge failed and that a value of 1000 mbars is to be used for that point in time. Any number less than zero would trigger this event.

Column 6 Solar insolation (W/m^2). This is the downwelling radiation measured at the weather station by a pyranometer that typically covers the visible and near-infrared portions of the electromagnetic spectrum (400 to 1100 nanometers of wavelength). If these data are not available, the code can be directed to calculate solar insolation on a surface based on the latitude of the test site, time of year, and surface orientation.

Column 7 Wind speed (m/s) at a known height above the ground

Column 8 Cloud type (an integer number ranging from 1 to 8). The cloud type index identifies different cloud genera (such as cirrus, stratus, etc.) and triggers correction factors used when the simulation code is directed to generate insolation values (Balick et al. 1981a). In lieu of real data, clear sky conditions are generally identified by cloud types 1 or 2. There is a cloud correction factor for the long-wave irradiance term in the surface energy balance that is also controlled by the cloud type and the percent of cloud cover (to follow). Cloud type must be entered by hand.

Column 9 Cloud cover (percent). This column of data is also entered by hand and could be significant if insolation is being computed. Otherwise, its effect can be negated by setting all values in this column to zero.

Column 10 Saturation factor. This is a decimal number ranging from 0.0 to 1.0 that triggers the latent heat exchange calculation. The original documentation for the TSTM code identified this term as the relative saturation of the top surface material but used it only as a weighting factor for controlling the impact of evaporative cooling on the simulation. Furthermore, the original code used the wind speed indicator height above the ground as a factor in both the empirical latent heat exchange and sensible heat exchange functions, but the source for those functions (Oke 1987) specified that the log height (z/ln(z)) of the wind speed indicator should be used in the formulation, because of an assumed exponential wind speed profile. In other words, this author had some concern about the physics behind the use of this saturation factor. As a result of this concern, log height has been inserted into the sensible and latent heat exchange relationships. Furthermore, the numbers in this data file column are now reasonable approximations to the actual near surface degree of saturation. If near-surface volumetric moisture content data are available (Column 11), then saturation factor values are computed as the ratio of that moisture content to an assumed porosity of the soil (0.40).

Column 11 Volumetric soil moisture no. 1. Because soil thermal properties are dependent upon moisture content, an attempt was made in this code to track changes in moisture content as a function of depth in the soil. This number represents the moisture content recorded at the shallowest depth on the test site (if those data were collected).

Column 12 Volumetric soil moisture no. 2. This is the moisture content in the soil at the deepest depth recorded on the test site. Along with a fixed moisture content boundary value at a third depth, the numbers in columns 11 and 12 were used to compute a soil moisture profile at every time step in the simulation.

Column 13 Physical temperature no. 1. A thermistor or thermocouple temperature measurement made at the same depth in the soil. This number is not used in the simulation, but could be compared to the simulation results as a measure of goodness.

Column 14 Physical temperature no. 2. Another temperature measurement at another depth. For the simulations reported in this study, the two temperature measurement depths in this data file corresponded to the volumetric moisture measurement depths.

Column 15 Radiometric temperature no. 1. This column and the next contain surface temperatures measured with a staring radiometer (see the report cover photograph) that can be compared to the surface temperature predictions made by the TSTM/VEGIE code. For this study, the temperatures in column 15 are those of the land mine shown in the cover photograph.

Column 16 Radiometric temperature no. 2. A second surface temperature that could also be used to verify the simulations. For this study, these temperatures are of the bare soil shown in the cover photograph.

Volumetric soil moisture data are not critical if soil thermal properties that are independent of moisture values are going to be used in the simulation. Physical and radiometric temperatures are also necessary only if one wishes to validate the model simulations at a given test site. With sufficient experience, a model user can generate sensible, if not accurate, results for any test site without the burden of validating the simulations against real data. Naturally, the optimum situation is one of validated simulations.

Model Execution Using the Graphical User Interface

Within the “tstm_files” folder found on the enclosed CD, there exists an Excel spreadsheet called “TSTM simulation results template.xls.” It contains all of the charts that are used to display results for any simulation. To perform a TSTM/VEGIE simulation, the user must open the Excel spreadsheet on his PC, pull down the “Tools” menu, select “Macro,” then select “Macros” from the submenu, and then click on “TSTM” in the Macro window that appears. This action executes a Visual Basic (VBA) program that displays the GUI (Figure 2) which, in turn, controls the simulation and display of results. A listing of the TSTM macro can be found in Appendix C.

While the weather data and temperature and moisture measurements described in the previous section form the bulk of the input data file, the user must still provide other parameter values that control the flow of simulation output as well as define thermal and optical properties of the layered structure being simulated. Referring to the highlighted text and buttons shown on Figure 2, the following procedure should be followed to perform a TSTM/VEGIE simulation.

Select Input File: Clicking on this button produces a window that helps the user select the .csv file that contains the site description, the weather data and moistures and temperatures described above, and ends with a line containing the word “End.” If a given input data file has not been previously accessed, then additional data must be entered into the GUI input boxes by hand. If, on the other hand, the chosen data file has been previously accessed, then it is possible that the additional data have already been appended to the input data file (lines 29-46 in Appendix B). Those data will be used to automatically populate most of the GUI input data boxes as soon as the input data file is selected. In either case, the macro scans the data file for the range of days, counts the number of input data file line entries, and displays those results in the appropriate input boxes.

Single-day simulation control: In this area of the GUI user form, the user is allowed to choose whether he/she wants to do a single-day simulation or a simulation for all of the days listed in the input data file. For the example shown, a single-day simulation was chosen, and that day was 210. Even if a multi-day simulation was selected, the user still has the option of specifying a single day for which results will be displayed.

Surface properties: Several parameters are required to properly specify air-soil interface conditions. These include a solar insolation flag, optical properties, and flags that control how latent heat and sensible heat calculations will be carried out within the simulation. For this example, the flag indicating that solar insolation values would be read from the input data file was chosen. If the user had chosen to let TSTM/VEGIE calculate solar insolation values, then he/she would have to enter the surface element slope, azimuth, and latitude of the test site into the appropriate boxes. A slope of zero degrees is horizontal. The azimuth angle of the surface, in degrees, only has meaning if the slope of the surface is not horizontal. An azimuth angle of zero degrees means that the projection of the surface normal unit vector onto the local horizontal plane at that location points south. Positive angles are clockwise from south. The test site latitude is also expressed in degrees.

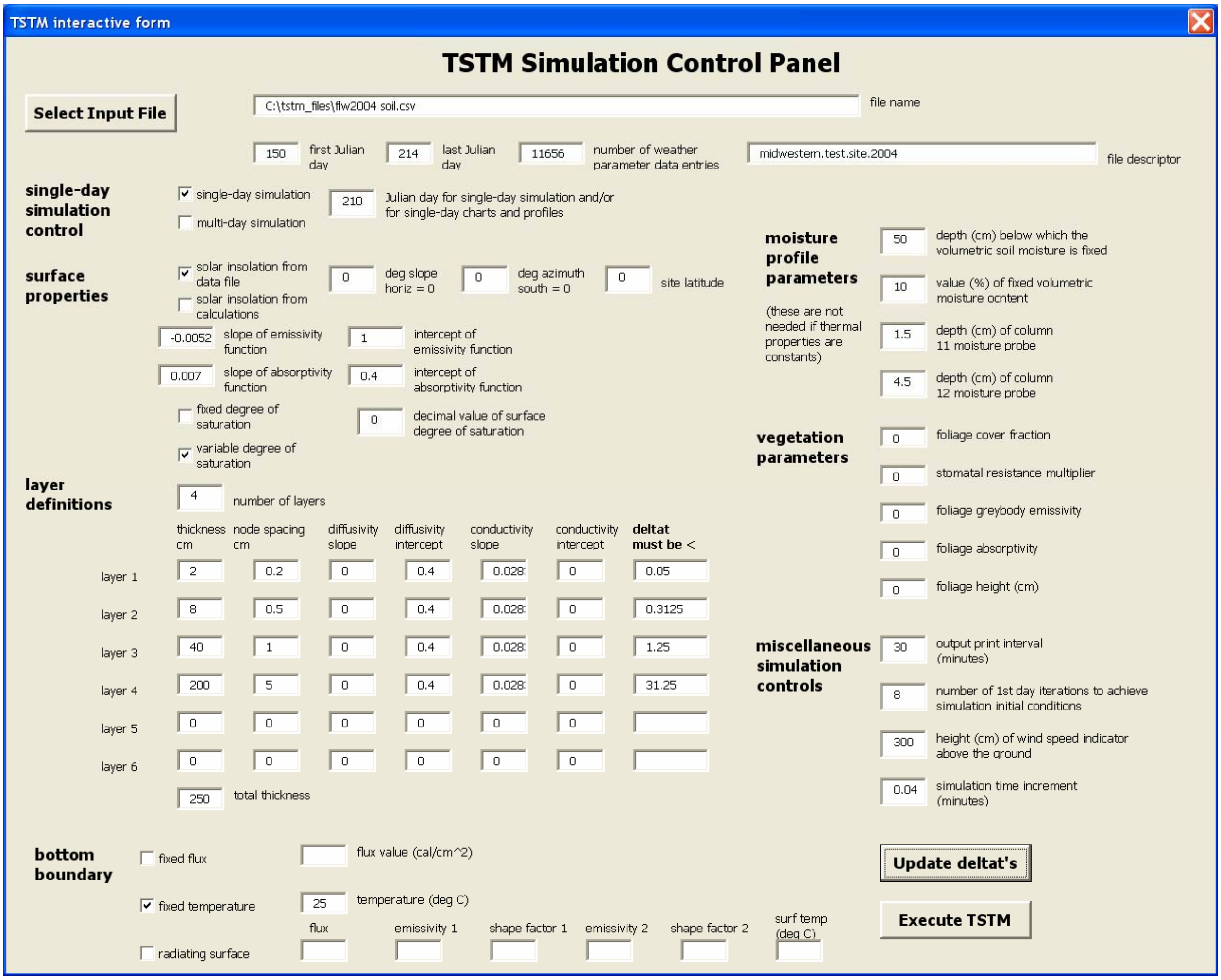

Figure 2. The TSTM/VEGIE Graphical User Interface

Two numbers are required in this section that represent the slope and intercept of a linear equation that defines the long-wave emissivity of the surface material as a function of surface volumetric moisture content. One method of determining the relationship between optical properties and moisture content is described in Chapter 4. If one does not know how long-wave emissivity varies as a function of moisture content, then he/she can choose emissivity to be a fixed value, in which case the slope should be set equal to 0.0.

Two additional numbers are required that represent the slope and intercept of a linear equation that defines the shortwave absorptivity of the surface material as a function of surface volumetric moisture content. The same instructions hold for constant values as for the emissivity numbers.

The final option given to the user in the "surface properties" section is to set the flag for allowing the surface to have either a fixed degree of saturation or a variable degree of saturation. Degree of saturation is a parameter used in the calculation of latent heat transfer. The variable condition was described earlier as being related to the availability of near-surface volumetric moisture data (column 10 in the input data file). If a constant value for degree of saturation is chosen, then that value will be used for all of the simulation time-steps.

Layer definitions: This is the section of the GUI where the user can define up to six layers of solid material for which the simulation is being performed. In addition to the number of layers chosen, one must enter six numbers that define each layer geometry and the thermal properties of each layer. In this example, there were four material layers. The first number is the layer thickness, in centimeters. The next is the node spacing for that layer, in centimeters. They are followed by the slope and intercept of the linear relationship between thermal diffusivity (units of square centimeters per minute) and volumetric moisture content. The last two numbers are the slope and intercept of the linear relationship between thermal conductivity (units of calories per centimeter-minutes degrees Celsius) and volumetric moisture content. As with the optical properties, constant thermal properties can be defined by setting the slope values to 0.0. The "deltat" entry will be discussed below.

Bottom boundary: One of three conditions may be chosen for the bottom boundary of the material column being simulated. One is a constant heat flux that can be specified by the user. The second, which is the one most commonly used, is to specify a constant temperature boundary. For the example shown in Figure 2, the bottom boundary temperature was fixed at 25 °C. This is a very reasonable condition in soils over a time period of a few days, or even weeks. However, temperatures can vary at depths of 2 or 3 m when viewed on a seasonal basis. The final bottom boundary condition is that of a fixed flux lower boundary that faces a fixed-temperature radiating surface beneath the bottom boundary. One could imagine that such a condition might represent a layered medium over a cavity. The input values required for this third boundary condition include the bottom boundary flux, the temperature of the radiating surface, the emissivities of both surfaces, and two shape factors, which are related to the emitting and absorbing efficiencies of those surfaces.

Moisture profile parameters: Moving to the upper right area of the control panel, the user is next asked to define parameters that describe how moisture conditions in the materials vary as a function of depth. Those moisture values will be used by the code to calculate moisture-dependent thermal properties for each material layer. If the material is a man-made solid, then there will be no moisture variability, and these parameters are not needed. Furthermore, if the material is porous, but the user chooses to set thermal properties to constant values for the entire simulation, then these parameters are not needed.

If volumetric moisture data are available in the data input file (columns 11 and, possibly, 12), then the following parameters may be used to help track a realistic moisture profile throughout the layered media. This would be a common need for simulating soils. The parameters include a depth below which the moisture content is fixed, the value of that fixed moisture content, and the depths of the volumetric moisture meters whose data are listed in columns 11 and 12 of the input data file.

Vegetation parameters: There are five parameters required to define surface vegetation conditions. If the first number (or only number) is zero, then the vegetation contributions will be skipped. The first parameter is a number between the values of 0.0 and 1.0 that defines the foliage cover fraction and that can be roughly related to the leaf area index. The second number is a multiplier of the stomatal resistance function for stressed plants. A third parameter is the graybody emissivity of the foliage, while the fourth number is the shortwave absorptivity of the foliage. The last parameter is the foliage height, in centimeters. A simulation using the foliage cover option was not conducted for this study. The reader is referred to the original VEGIE report (Balick et al. 1981b) for a more thorough discussion of this option.

Miscellaneous simulation controls: The final four parameters that can be specified by the user are found in this area of the GUI, the first of which is the interval at which the user wants to see simulation output results sent to the output file. For the simulation depicted by Figure 2, the user wanted results displayed at half-hour intervals.

The second number specifies how many iterations on the first day of simulation will be performed to achieve something of a steady-state environment. While surface temperatures are very much controlled by the energy balance at the surface, several iterations on the first day's simulation might be required to achieve a repeatable set of temperature-depth profiles for that day. Typically, only a few iterations are required to achieve stability.

Another parameter specified in this area of the GUI is the height above the ground, in centimeters, at which the wind speed measurements were made at the test site weather station. It is the number that is used in both the latent and sensible heat exchange calculations.

The final parameter is the time increment for this simulation, in minutes. Since TSTM/VEGIE functions as an explicit finite difference code, a *stability condition* exists *for the time increment* that must be satisfied for all material layers. Within each layer, Equation 9 controls the calculation of the next time increment. As long as the coefficient of the bracketed term is less than ½, the calculation will not violate the second law of thermodynamics and the results will be stable (Holman 1968). While this is not an airtight proof, consider the following conditions. Let the temperature of the two nodes surrounding the center node in the finite difference calculation be equal and less than that of the center node. While the new temperature of the center node (at the end of the time increment) should be less than its old temperature (at the beginning of the time increment), it should not be less than that of the surrounding nodes. In mathematical notation, let

$$(T_{n-1})_{old} = (T_{n+1})_{old}$$

and

$$(T_n)_{old} = (T_{n+1})_{old} + \Delta T$$

For stability,

$$(T_n)_{new} > (T_n)_{old} - \Delta T$$

or

$$(T_n)_{new} - (T_n)_{old} > -\Delta T$$

If one defines

$$M = \frac{\alpha \Delta t}{(\Delta z)^2}$$

then from Equation 9,

$$M\left[2(T_{n+1})_{old} - 2\left\{(T_{n+1})_{old} + \Delta T\right\}\right] > -\Delta T$$

or,

$$M < \frac{1}{2}$$

In other words, the stability condition that must be met for each layer of material is:

$$\Delta t < \frac{(\Delta z)^2}{2\alpha} \qquad (10)$$

The GUI is set up to calculate a limiting time increment for each layer according to Equation 10. If thermal diffusivity values are defined as being dependent on volumetric moisture, then a value of diffusivity at a moisture content of 40 percent is taken as an upper bound value (assumes diffusivity increases with moisture content). To display the maximum time increments allowed for each layer, the user simply presses the “Update deltat’s” button on the lower right corner of the GUI screen. The numbers shown in the right-hand column of the “layer definitions” area of the screen are then displayed. For the example shown in Figure 2, the simulation time increment was controlled by the properties of the top soil layer, which required an increment of less than 0.05 minute.

Execute TSTM: When the user is satisfied that all simulation parameters have been properly defined, he/she may proceed with the simulation by pressing the "Execute TSTM" button at the lower-right corner of the GUI screen. What will happen immediately is that a small message window will pop up on the display screen that will say "Wait for TSTM to finish executing!" Since there is currently no way to monitor the progress of the simulation (it is proceeding through a macro shell command), the user must watch the color intensity of the message box (it will brighten when the code finishes execution) or watch the taskbar buttons across the bottom of the screen (there will be one for "tstmforgui.exe" that will disappear when the code finishes). The message box forces the macro behind the GUI to pause while the Fortran code executes. Once the simulation has finished, the user must press the "OK" button on the message box.

The next thing that will happen is another message box will appear asking: "Which column contains the measured surface temperature?" The spreadsheet page containing the simulation output values will be in the background. If the input data file contained a column of measured surface temperature data, then the charts generated by this macro can include the measured data as well as the simulated data. For the input data file described earlier, there were two columns of data containing surface measurements. Column 13 (or column "m" as seen in the background spreadsheet page) contained the mine surface measurement, and column 14 contained the soil surface measurement. If no measured surface data exists, then the user should select a column that has no data.

A second message box will then appear asking: "Which column contains the difference between measured and simulated temperatures?" Again, if such data do not exist, then the user can avoid plotting useless data by naming a column without data. For the simulation for which results will be shown in the next chapter, measured and simulated temperature difference results were displayed in column "o" for the mine surface and column "p" for the soil surface. Answering this final question and clicking the "OK" button completes the TSTM/VEGIE simulation. As the macro is currently written (Appendix C) two charts representing simulation results will be sent to the printer. One is the single-day simulation result for surface temperature compared to the measured data and the other is the set of 2-hr snapshots of temperature profiles as a function of depth for the chosen day. Example charts may be viewed in the next chapter.

Rules of Thumb for Selecting Surface Material Properties

This model can be executed in one of two ways. First of all, one can select material properties for all of the layers of material based on published data and previous experience with the model and simply predict surface temperatures for whatever the input weather parameters may be.

The second method (which is much more realistic for natural materials with thermal properties that are expected to vary with volumetric moisture content) is to select weather extremes for which iterative single-day simulations will be performed to establish the optimum set of properties for each set of weather conditions. For example, one can choose a day for which site soils would be quite dry near the surface. While allowing the code to use moisture-depth profiles specified by the user (see previous section), the user can determine the optimum values of constant thermal and optical properties that give the best comparison to measured data. Those values can then be assigned to an average soil moisture for that day. The user can then select a wet-soil day and repeat the process. Finally an intermediate soil moisture day can be simulated in the same trial and error manner. The user then will have a crude relationship between each property and soil moisture which, in turn, becomes part of the input data file for a complete multiple-day simulation for that test site. An example of this process is shown in the next chapter.

The following table provides a useful summary of how property value changes for these iterative simulations will change the resulting predicted daytime temperatures of the surface. While the effects of long-wave emissivity and short-wave absorptivity are very sensible, the impact of changes in thermal diffusivity and thermal conductivity are less intuitive. One way to rationalize their effects is to combine the defining relationships for specific heat and thermal diffusivity in the following way. Consider first the relationship that defines how much heat energy Q is required to raise the temperature of a lump of material (mass m and volume V) by an amount labeled ΔT (Ohanian 1985):

$$Q = m\,c\,\Delta T = \rho V\,c\,\Delta T \tag{11}$$

When combined with Equation 2, which defines thermal diffusivity, one can easily show that

$$\Delta T = \frac{\alpha Q}{\kappa V} \tag{12}$$

In other words, for a given amount of heat energy flowing into a fixed volume of material, an increase in thermal diffusivity will result in an increase in material temperature. The inverse is true for an increase in thermal conductivity.

Table 1
Rules of Thumb for Selecting Surface Material Properties

Material Property	Physical Description	Effect on Predicted Daytime Temperatures Due to an Increase in the Property Value
Long-wave Emissivity	A measure of the rate at which a surface can radiate IR energy to its surroundings	Decrease
Short-wave Absorptivity	The fraction of incoming solar radiation that is absorbed by the surface material	Increase
Thermal Diffusivity	The ratio of thermal conductivity to volumetric heat capacity. A measure of the speed with which temperature spreads throughout the material.	Increase
Thermal Conductivity	A measure of the rate of heat energy flow through the material to a lower temperature reservoir.	Decrease

4 Example Simulations

Weather data, volumetric soil moisture values, and radiometric surface temperature measurements were collected at a midwestern test site during the summer of 2004. These data covered a time period of 64 days at a rate of one set of readings every hour for the first 56 days and one set of readings every minute for the next 8 days.

Determining Moisture-Dependent Soil Properties

Laboratory data clearly show that soil thermal properties are a strong function of moisture content (Ochsner et al. 2001), with a general trend of increasing thermal conductivity, volumetric heat capacity, and thermal diffusivity with increasing volumetric moisture content. Those same measurements also show that the spread in the data is large enough that simple model fits cannot be used for predictive purposes. In fact, other sources argue that thermal diffusivity decreases with increasing moisture content at higher moisture values, because the volumetric heat capacity increases faster than the thermal conductivity (Jumikis 1977).

If a realistic simulation of a layered test site soil is going to be performed over a period of several weeks, during which multiple rain events followed by drying periods occur, then some accounting for a change in thermal (and possibly optical) properties with changing soil moisture content must be made. The approach taken by the author to deal with this dilemma is to conduct at least three single-day simulations for different soil moisture conditions and adjust the surface material properties to best match measured data. Simple model fits to those thermal (and optical) properties plotted against soil moisture content can then be used to define the input data file properties for a multi-week simulation in which the soil moisture conditions were highly variable.

For the following simulations of surface temperatures at a midwestern U.S. test site, single-day calculations were done for day 193 (average measured volumetric moisture content equal to 5.5 percent, peak measured surface temperature equal to 54 deg C), day 178 (7.0 percent, 44 deg C), and day 212 (15.0 percent, 28 deg C). Trial-and-error simulations were performed for each of these days, resulting in a different set of values for both the thermal properties and the optical properties of the surface soil. Those values and the corresponding regression model fits are shown on Figure 3.

It is important to remember that these relationships between thermal and optical properties and volumetric soil moisture content hold for this site and this type of soil. At another test site, or even at another location within this particular test site, a similar single-day simulation exercise might result in a much different set of thermal and optical parameters. In other words, *thermal and optical properties for soils are very likely to be site-dependent*.

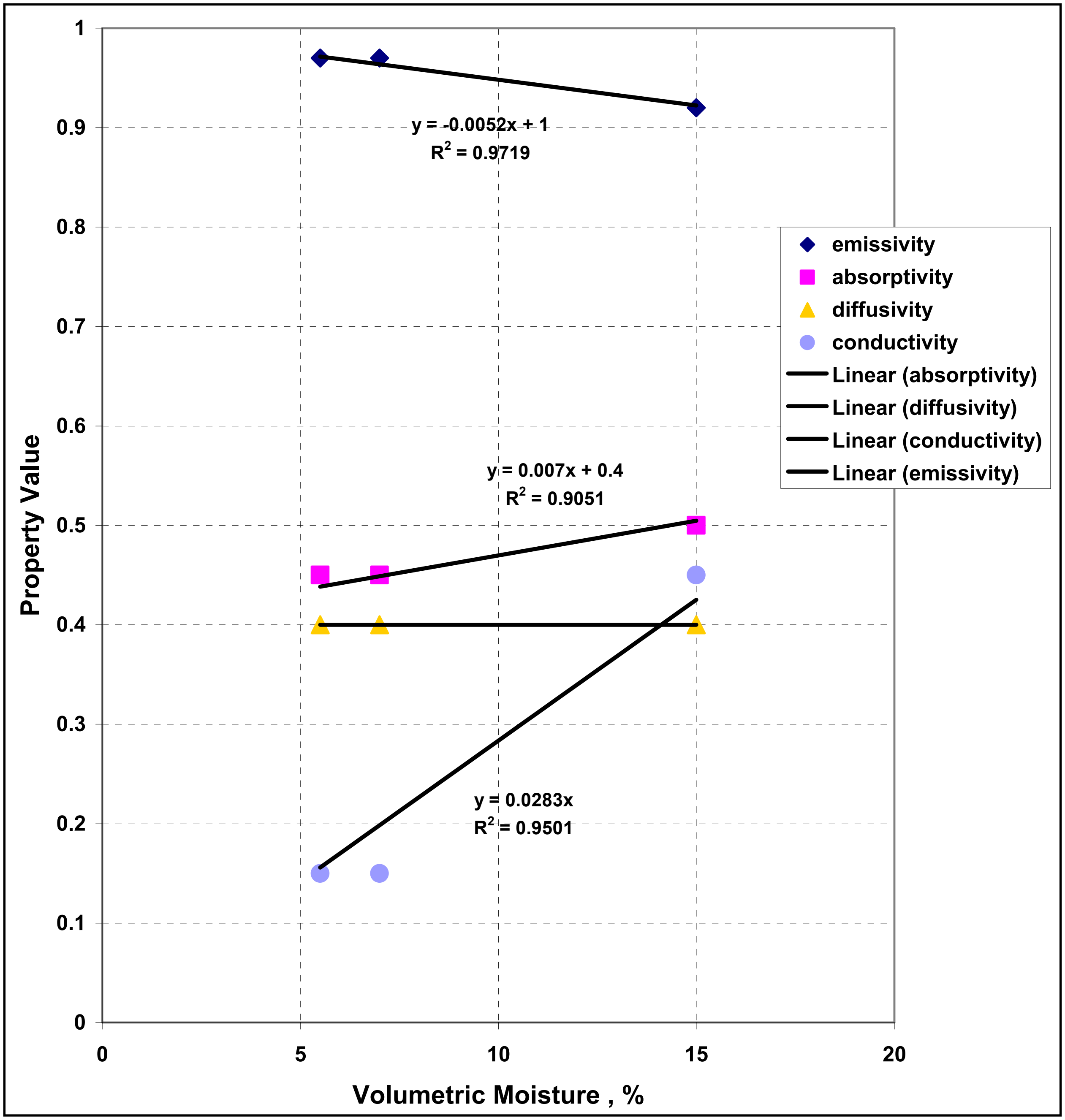

Figure 3. Test site thermal and optical properties of soil as a function of volumetric moisture content

A Simulation of Soil Surface Temperatures

One goal of this study is to demonstrate that the modified TSTM/VEGIE code can do a reasonably good job of predicting surface temperatures under a variety of weather conditions. Weather data, surface temperature data, and soil moisture data collected at a midwestern U.S. test site during the summer of 2004 were used as input and validation data for two 64-day simulations of a bare soil surface and a metallic land mine surface. In a later section, these simulation results will be used to calculate a predicted thermal contrast between the mine surface and the soil surface.

All of the following charts are self-explanatory, but some commentary may be useful in interpreting the results. Simulation results for any single day could have been chosen for display. In this case, day 193 (one of the hotter, drier days) was chosen, and those results are shown in Figures 4 and 5. Obviously, one can never expect perfect simulation results; however, predicted surface temperatures compare very well with measured data for this day. Predicted results appear to lag the daytime data in a manner that results in predicted morning temperatures that are as much as 4 °C lower than measured data and early evening temperatures that are as much as 5 °C higher than the measured values. The material properties could have been adjusted for this one day to give much smaller differences between predictions and measurements, but that would violate the spirit of this exercise, which was to demonstrate physically sound simulations over a variety of weather conditions using one set of material property definitions.

Figure 6 compares the predicted surface temperature with the measured values for all 64 days of the simulation. In general, the results are quite reasonable, except for extremely wet and overcast conditions. One of the model input parameters that was not measured and used properly was that of percent cloud cover. Those data would have had an impact on simulation results through the long-wave irradiance term in the surface energy budget equation (Equation 2). In addition, there is still some question as to whether or not the latent heat exchange and sensible heat exchange formulations are being used properly. That question remains to be answered through future research.

Figure 7 summarizes the differences between the predicted surface temperatures and the measured data. Although the results appear somewhat noisy, note that the average difference is only 1.1 °C and that the standard deviation of the differences over the entire 64-day period is only 3.7 °C.

The final results shown (Figure 8) for this simulation are the soil temperature profiles for the even hours of day 193. Note that, even though three different layers of soil with the same thermal properties but different node spacings were used for this simulation, the resulting soil temperature profiles are very smooth and clearly show physically correct results such as the thermal inertia of the underlying soil (there is a time lag for temperature change beneath the surface). It would also appear that the assumption of a fixed bottom boundary temperature of 25 °C is not unreasonable, although in hindsight, a fixed temperature of 30 °C, or so, might have been a better choice. Further note that the zone of active temperature fluctuations is limited to a depth of about 40 cm.

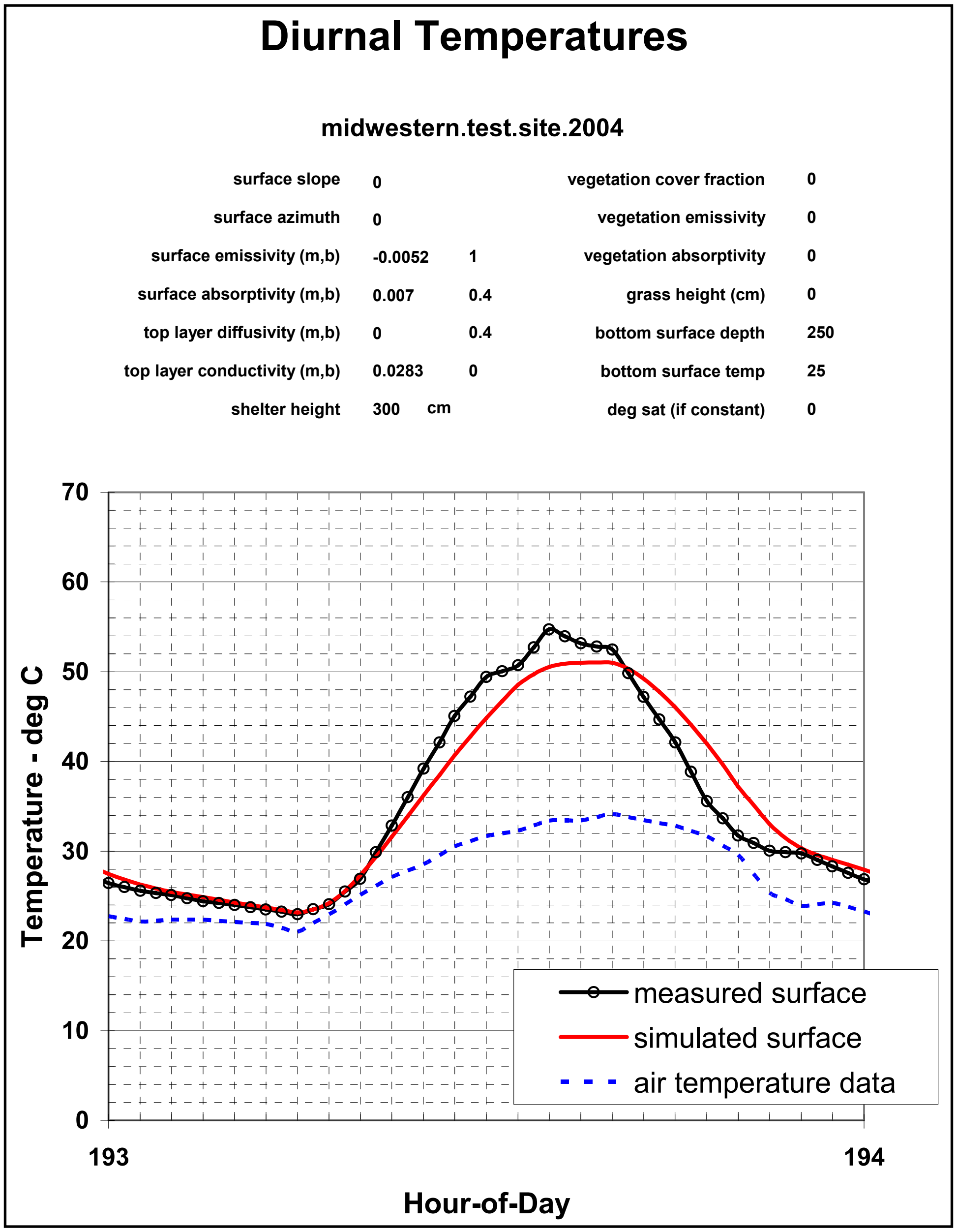

Figure 4. Predicted and measured soil temperatures for the midwestern U.S. test site (day 193)

A Simulation of Land Mine Surface Temperatures

A second 64-day simulation was performed for a metallic landmine sitting on soil at this test site (see report cover photograph). The geometry of this mine was approximated by a three-layer structure consisting of a 0.33-cm-thick carbon steel jacket filled with dense concrete. The overall thickness was taken to be about 10 cm. The latent heat exchange term in Equation 2 was turned off for this simulation by setting the saturation factor value at zero for all time-steps. Under those conditions, thermal and optical properties for the painted steel and thermal properties for the concrete were chosen to best predict the measured surface temperatures for days 193 and 212 (extremes in weather conditions). Results are shown in the following charts.

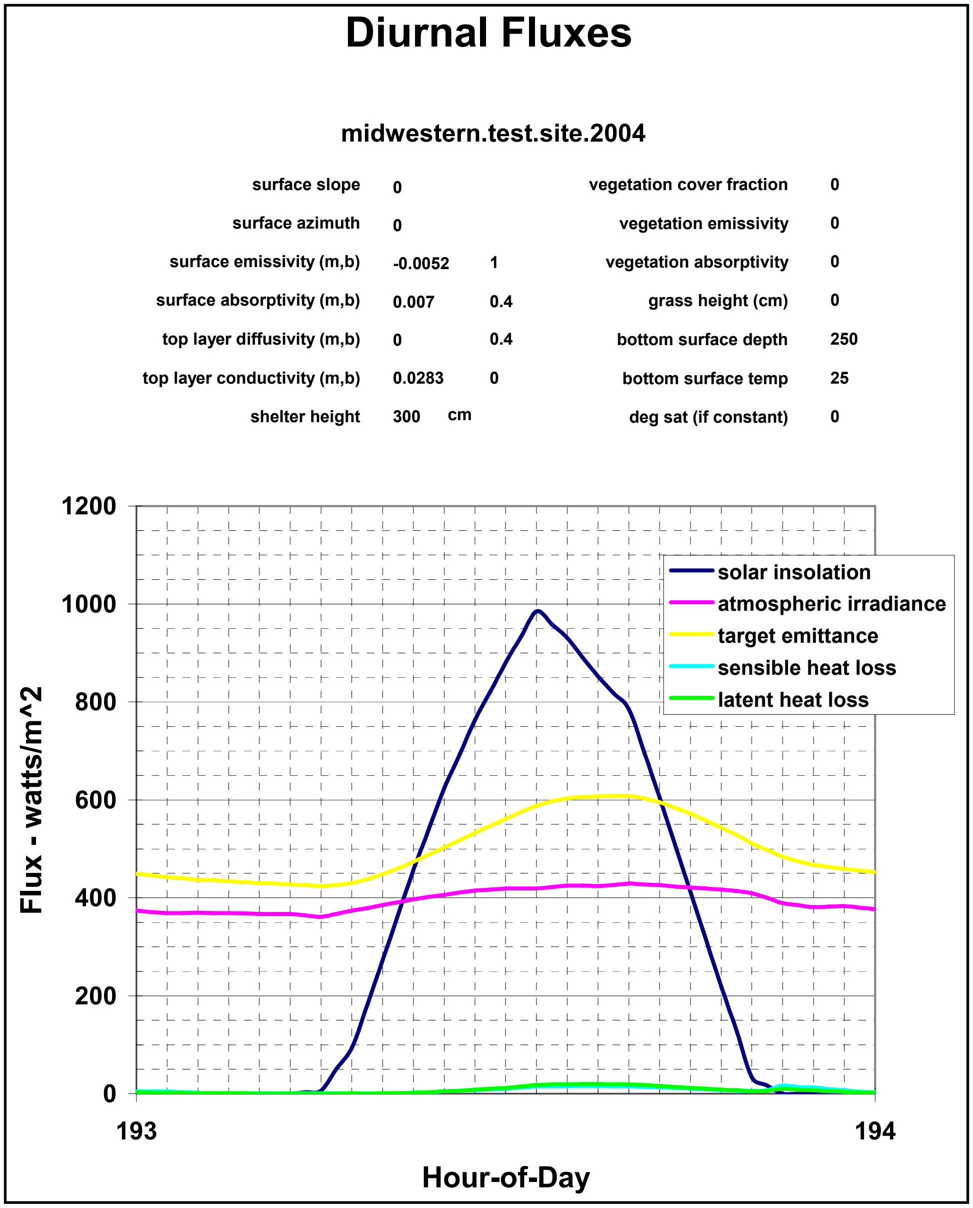

Figure 5. Predicted and measured energy fluxes for the midwestern U.S. test site soil simulation (day 193)

It appears that the land mine simulation was about as good as the soil simulation, with an average difference of -1.1 °C and a standard deviation of 4.9 °C. The most noteworthy result is that of the temperature profiles shown in Figure 13. The profiles within the underlying soil (below 10 cm) have a much different character than those generated for the soil simulation (Figure 8). The temperature of the soil in contact with the mine is forced to track the temperature of the bottom of the mine. The thermal inertia displayed in Figure 8 is not as evident on this chart. What the bottom boundary temperature should be is certainly left to speculation, and the zone of influence extends a little farther into the underlying soil.

Landmine-Soil Thermal Contrasts

Of paramount importance to the airborne sensor community is the temperature contrast between man-made targets, such as the land mine, and background materials, such as the soil. Thermal infrared sensors that image the apparent temperatures of objects within their fields of view can easily detect large differences between targets and backgrounds. During the daylight hours, those contrasts can be large and positive (the target is hotter than the background). On the other hand, man-made materials can be cooler than their natural surroundings at night, resulting in a negative thermal contrast between the two.

Figure 13 shows both the predicted and measured thermal contrasts between the land mine and the surrounding soil for only 2 days during the 64-day simulation. In both cases, the positive contrasts are much greater than the negative contrasts, although the simulations produce greater extremes. What is most interesting occurs at the "cross-over times." Those are the times of day when the contrast polarity switches from positive to negative, or vice-versa. While most IR sensors have the ability to detect fairly small differences in temperature between two objects, such differences can easily be washed out in the array of contrasts at points surrounding the target. Therefore, one wants to avoid using such a sensor to search for targets at times near the cross-overs. For days 193 and 194, the morning cross-over occurs between 8:00 AM and 9:00 AM, while the evening crossovers take place between 8:00 PM and 9:00 PM. For another season of the year, these cross-over times will probably occur at different times. It is noteworthy that the predicted cross-over times take place within an hour of the measured results.

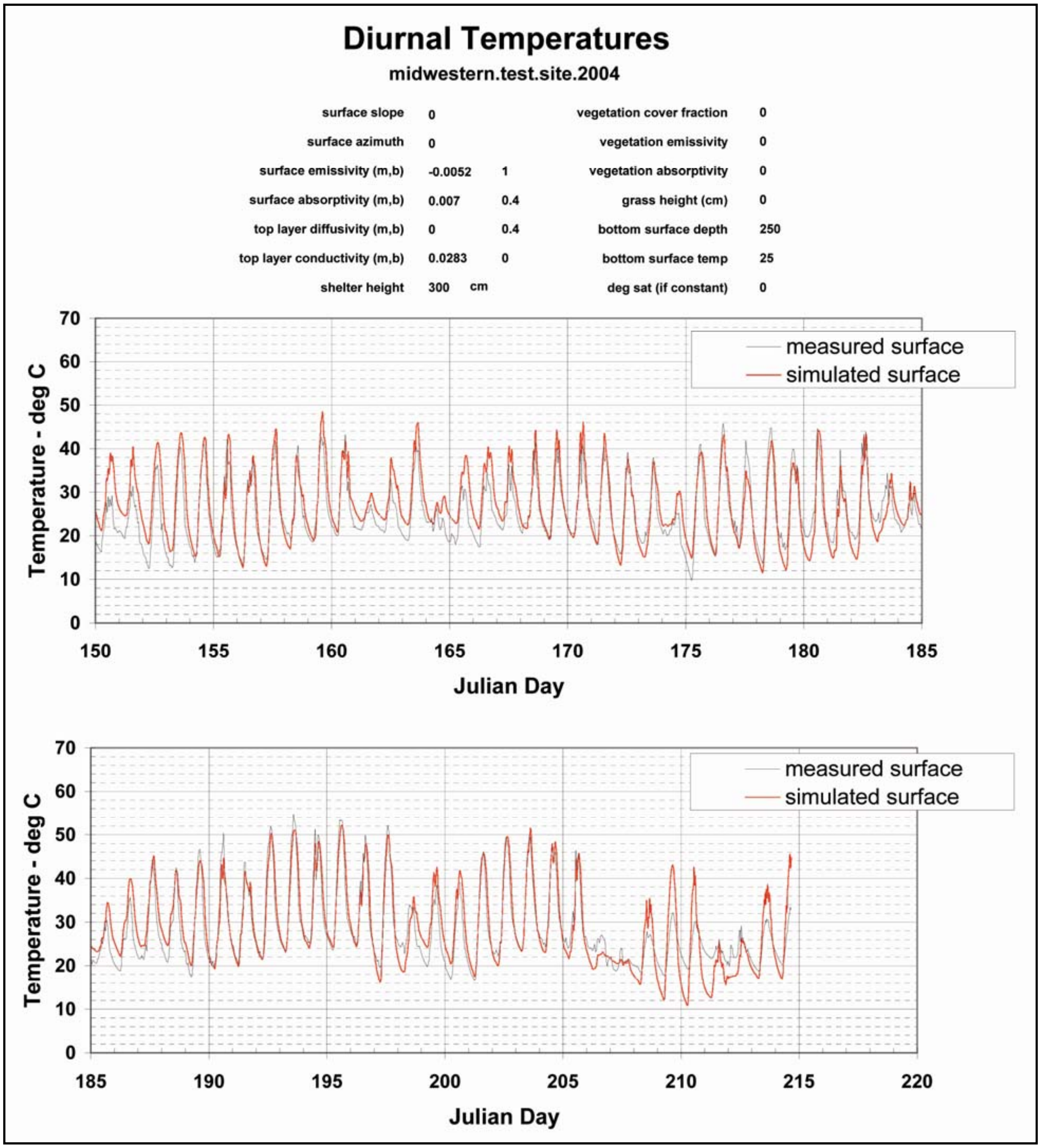

Figure 6. Predicted and measured soil temperatures for the midwestern U.S. test site (all days)

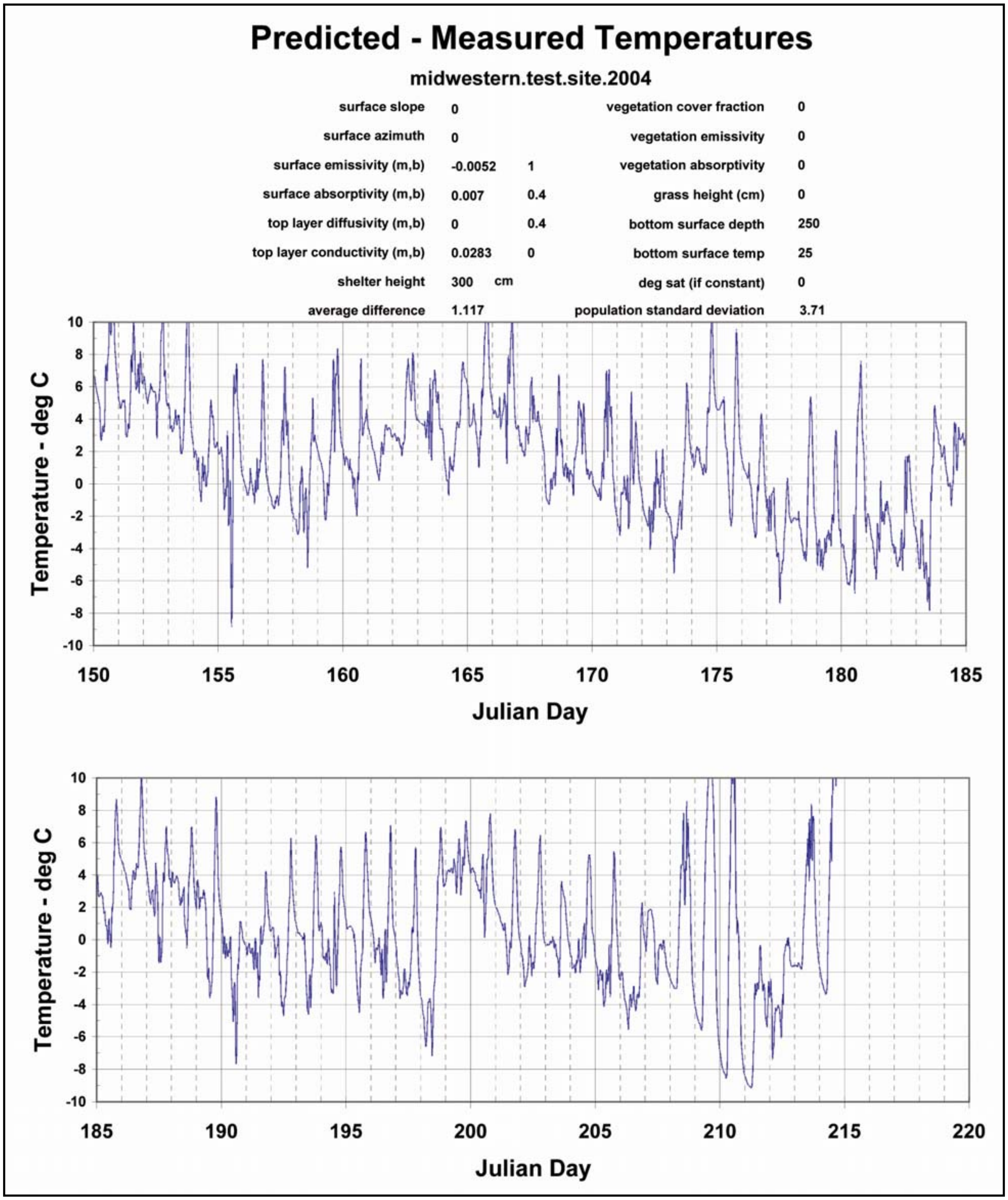

Figure 7. Differences between predicted and measured soil temperatures for the midwestern U.S. test site (all days)

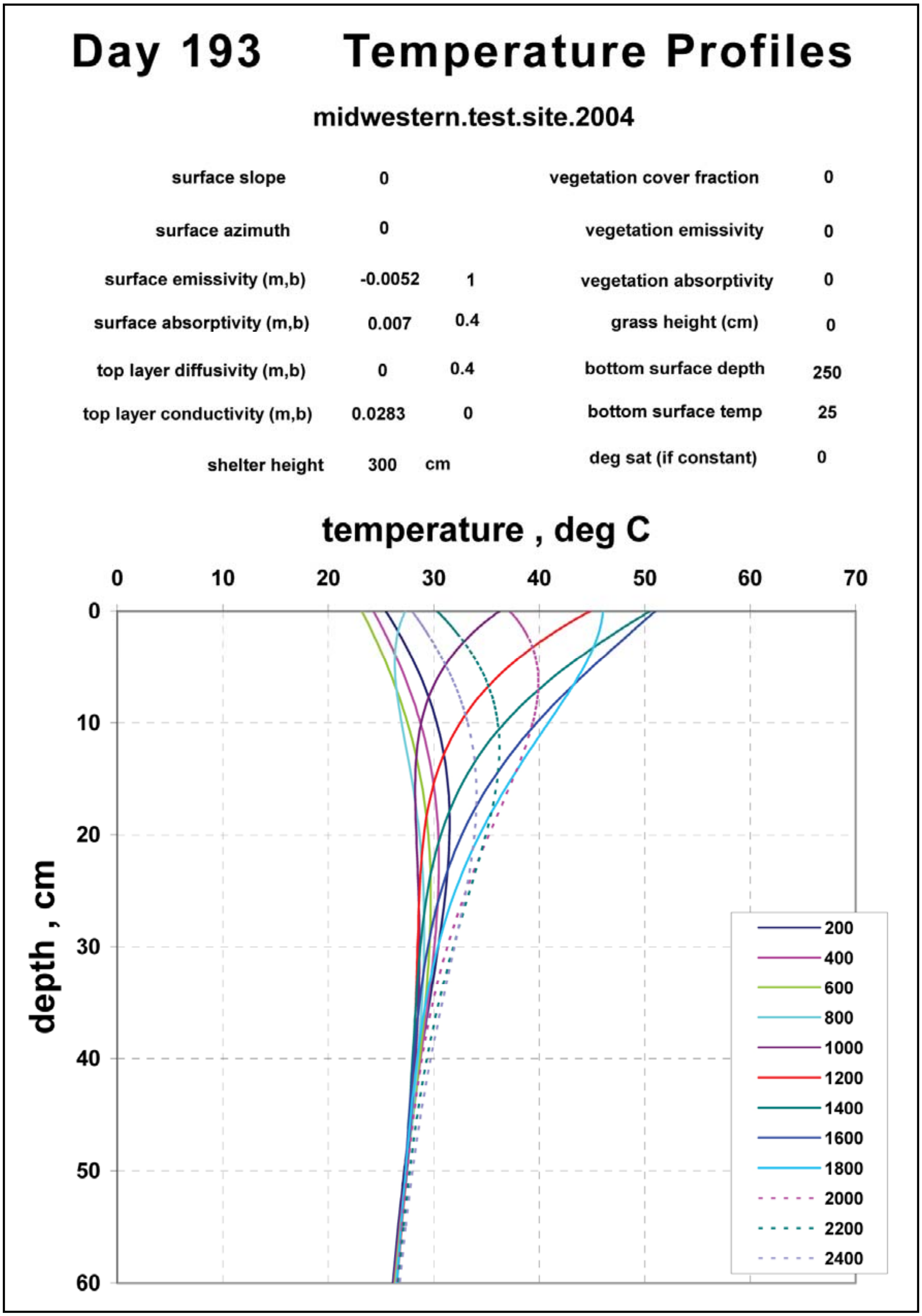

Figure 8. Predicted soil temperature profiles for the midwestern U.S. test site (day 193)

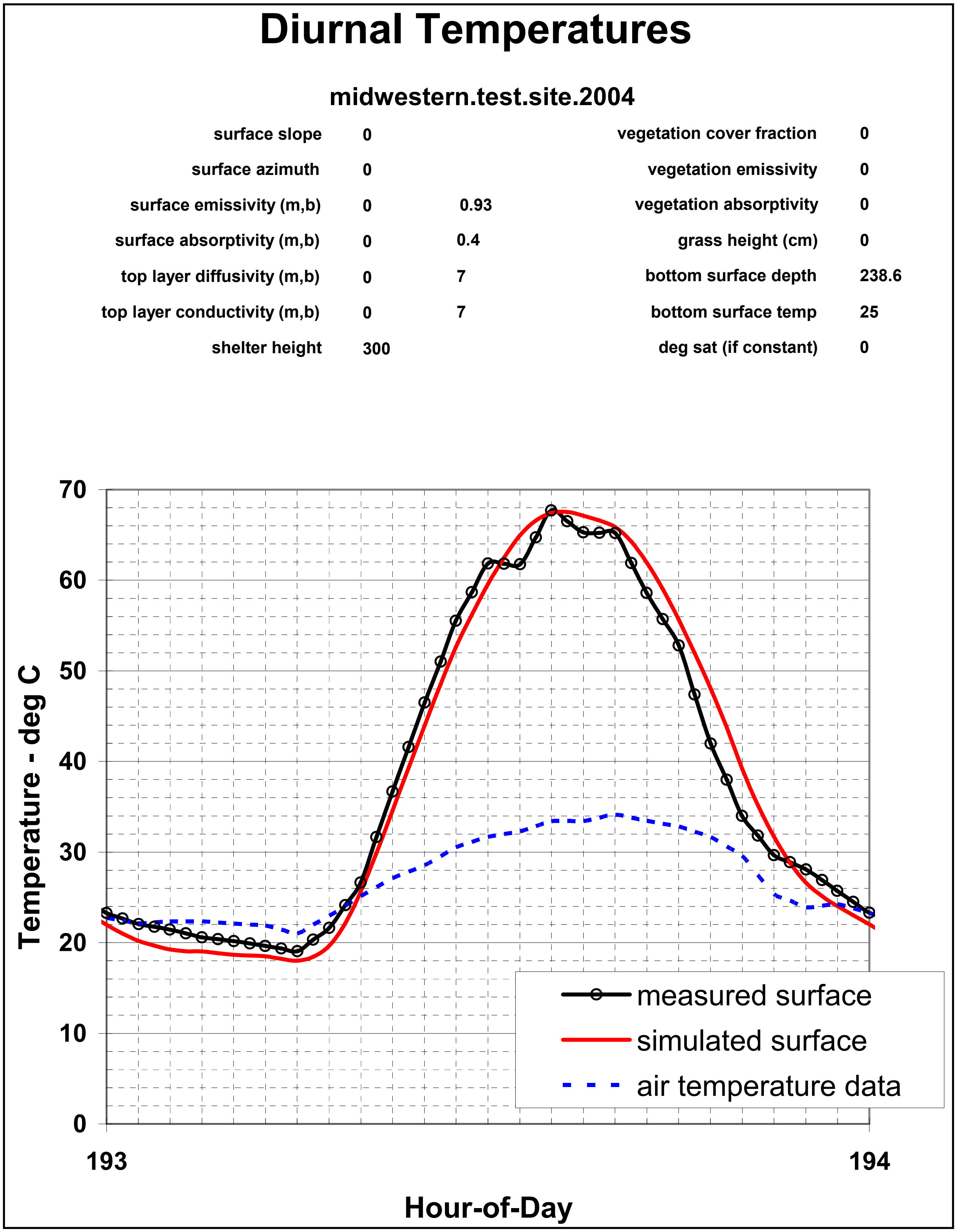

Figure 9. Predicted and measured land mine temperatures for the midwestern U.S. test site (day 193)

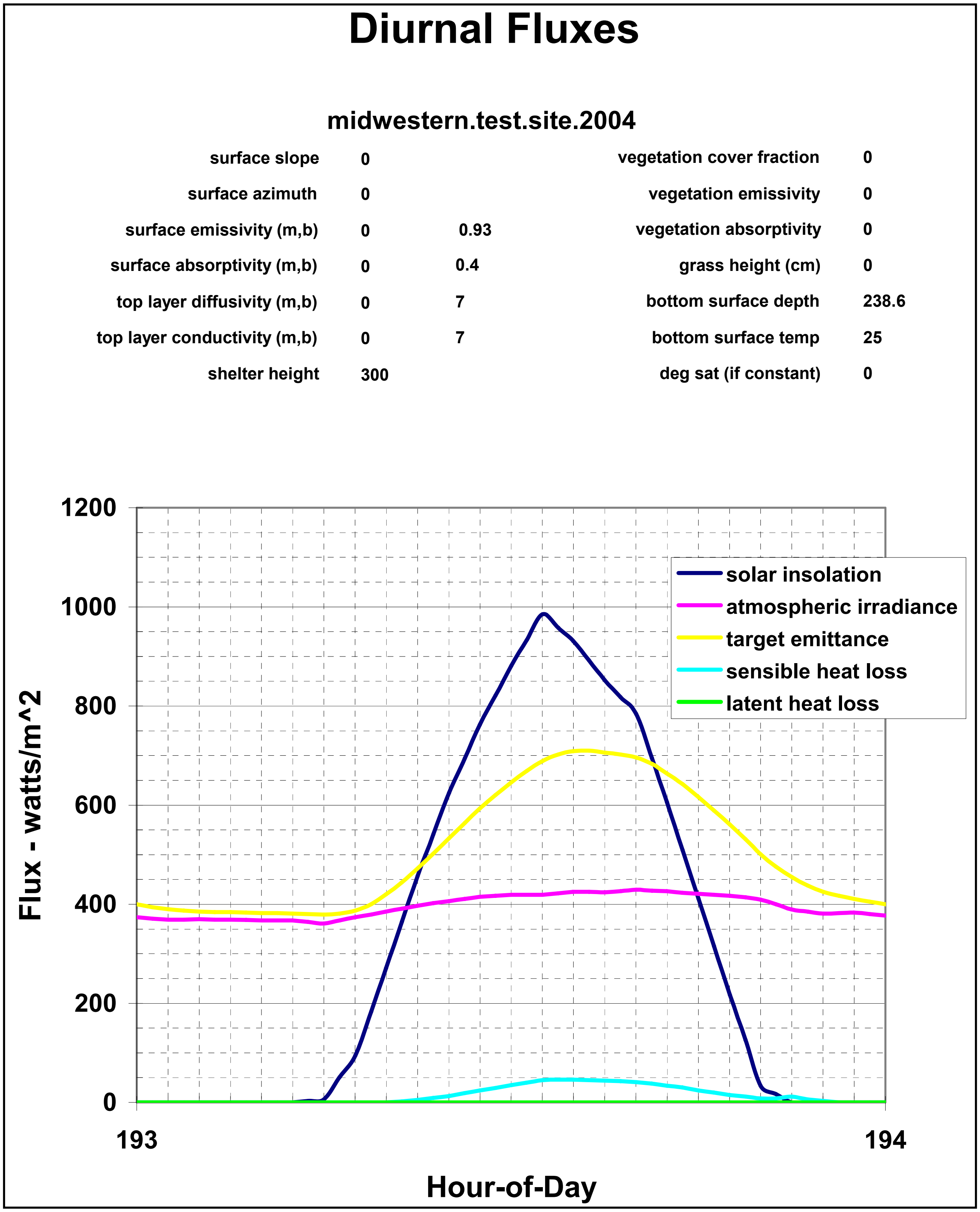

Figure 10. Predicted and measured energy fluxes for the midwestern U.S. test site land mine simulation (day 193)

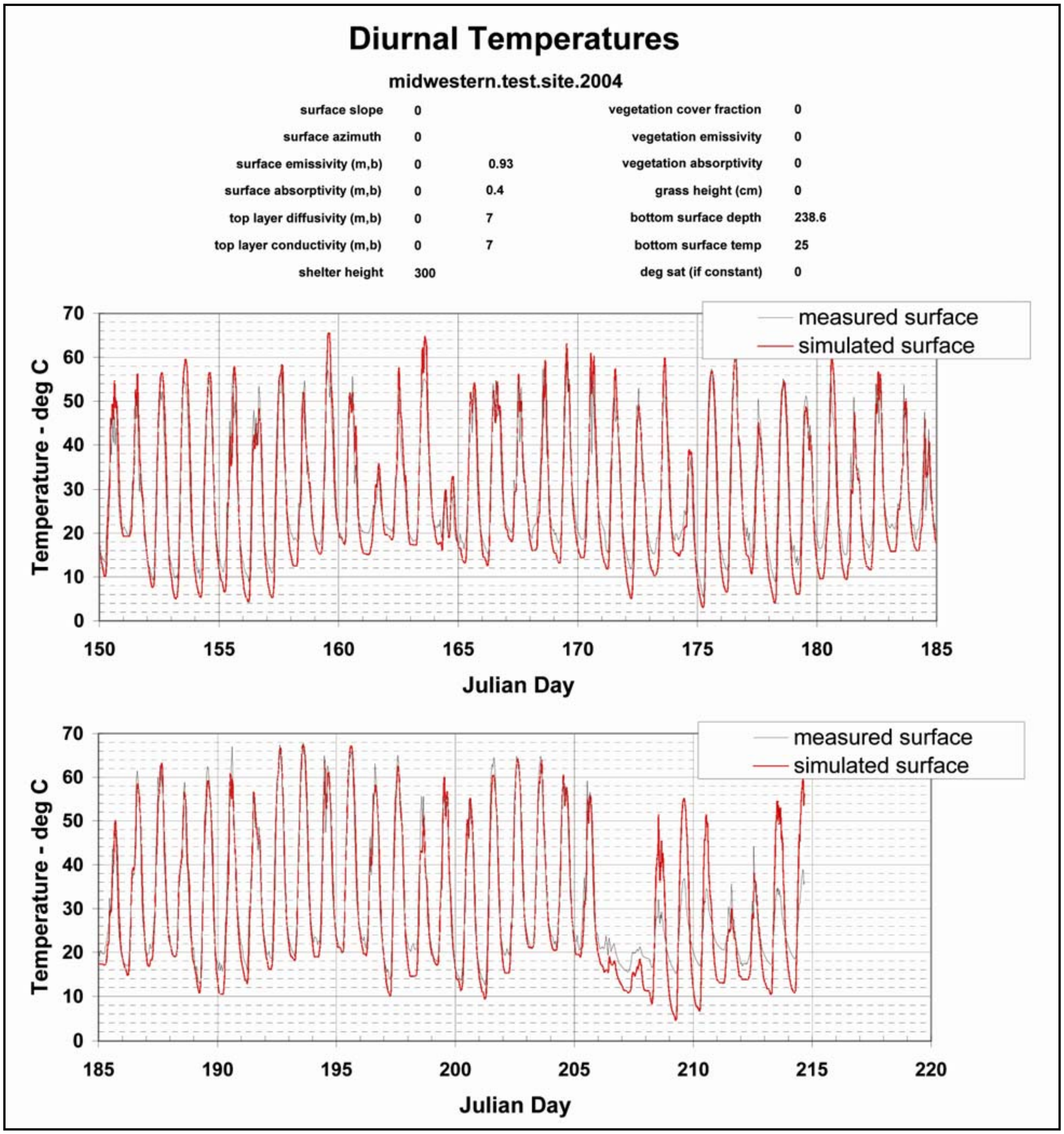

Figure 11. Predicted and measured land mine temperatures for the midwestern U.S. test site (all days)

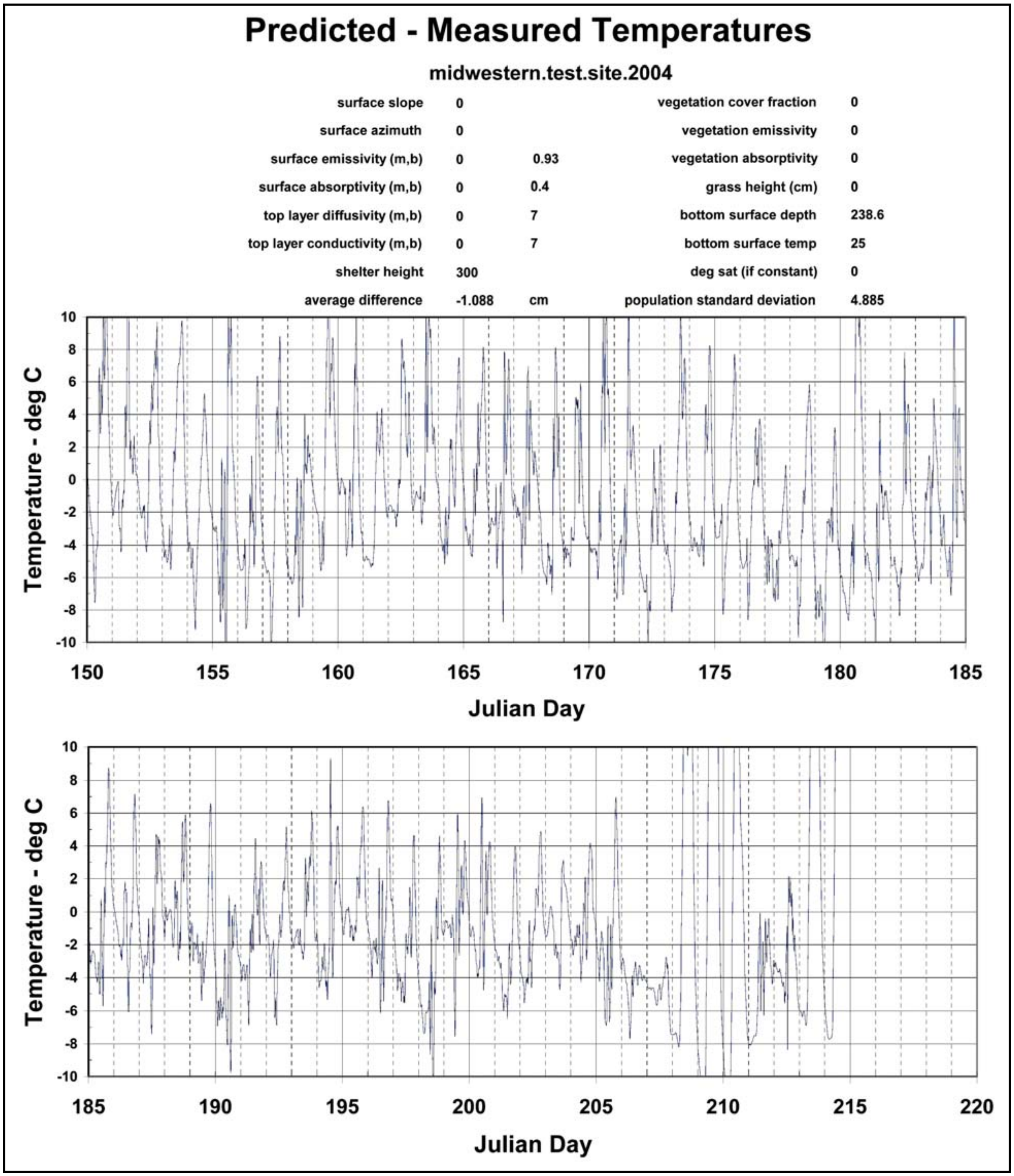

Figure 12. Differences between predicted and measured land mine temperatures for the midwestern U.S. test site (all days) (Continued)

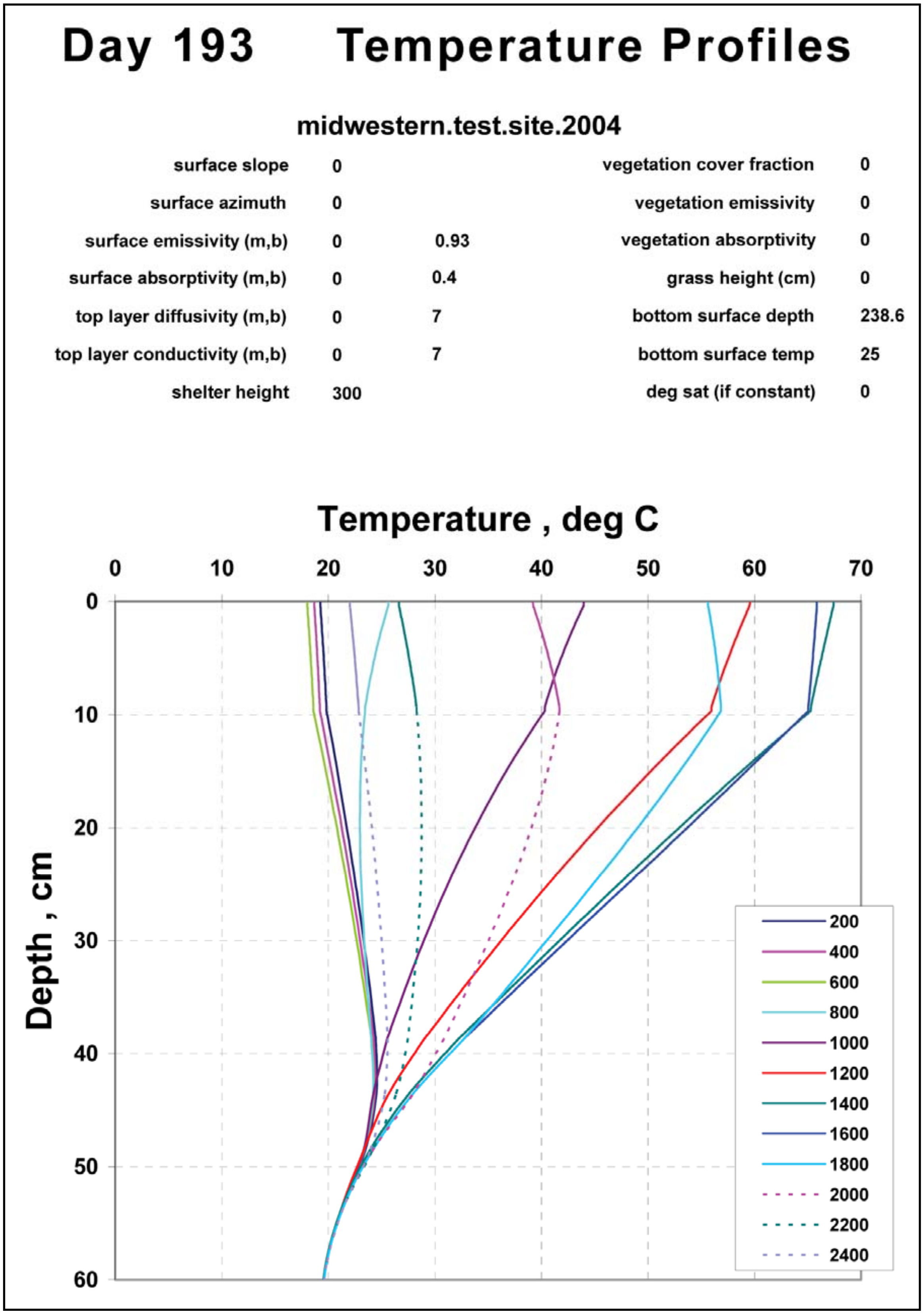

Figure 13. Predicted land mine (over soil) temperature profiles for the midwestern U.S. test site (day 193)

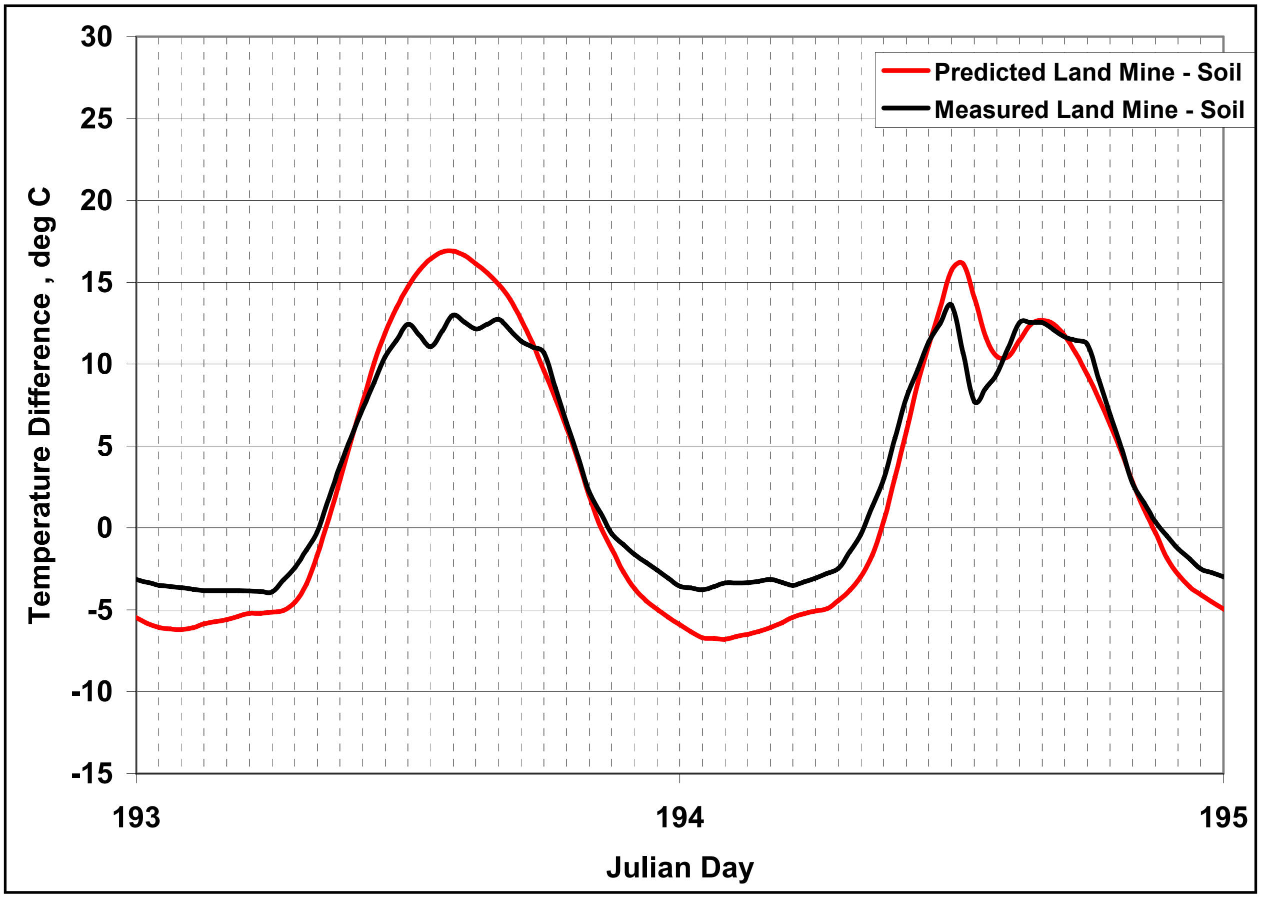

Figure 14. Predicted and measured land mine/soil thermal contrasts (days 193-194) for the midwestern U.S. test site

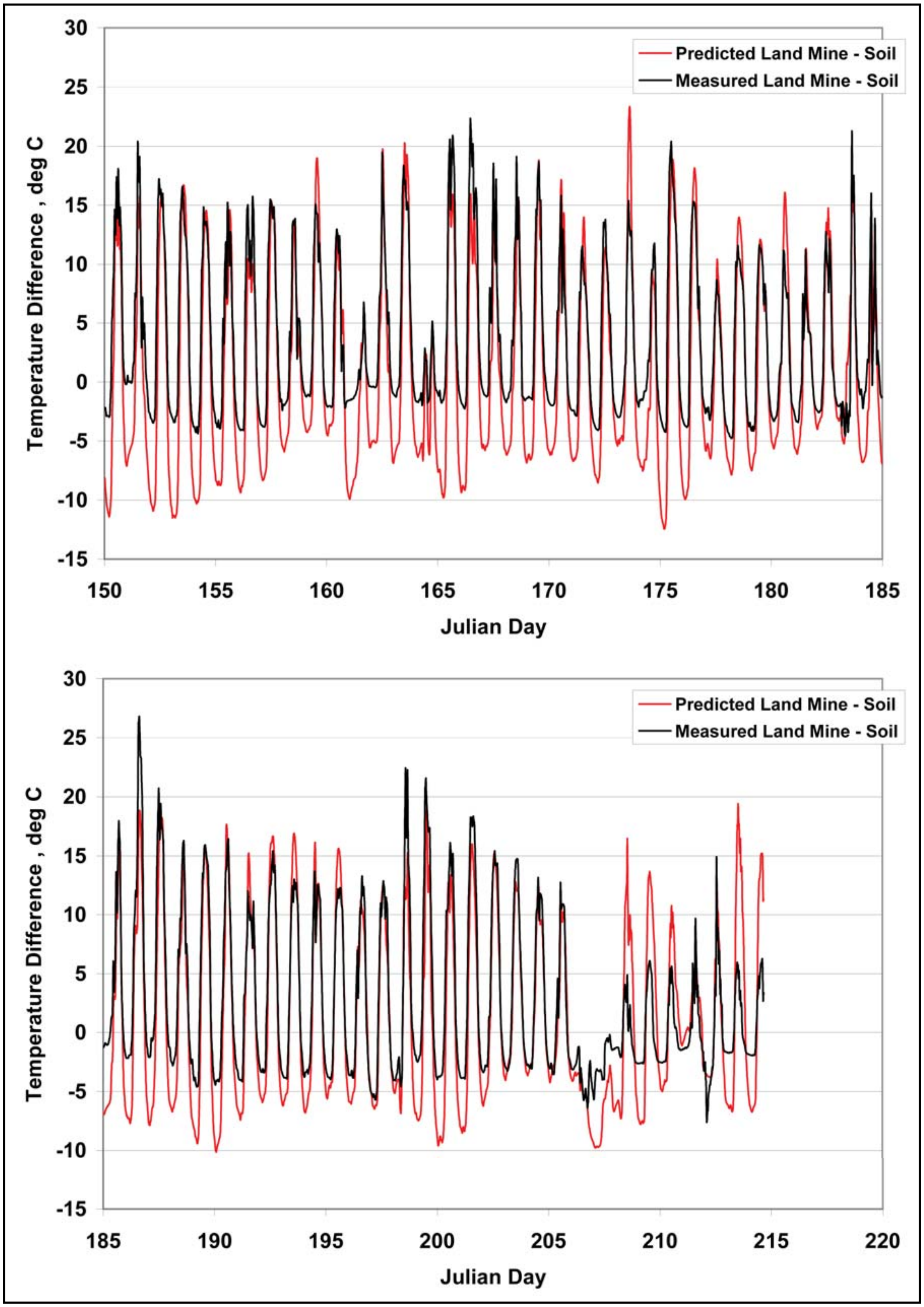

Figure 15. Predicted and measured land mine/soil thermal contrasts (all days) for the midwestern U.S test site

5 Summary and Recommendations

Summary

An existing 1-D finite difference surface temperature prediction model was modified to perform multiple-day simulations of natural terrain and man-made surfaces using weather station data as the primary input. The original code was modified in a number of ways, including the ability to exercise the code on a PC, to input thermal and optical material properties that vary with moisture content, and to utilize more realistic latent and sensible heat exchange models. An Excel spreadsheet was developed to help display simulation results in meaningful ways.

The modified code was exercised against 64 days of weather, soil moisture, and surface temperature data collected at a midwestern U.S. test site. Comparisons of measured data with predicted results for bare soil and a metallic land mine were quite favorable. Of particular interest were the daily displays of thermal contrast between the land mine and the bare soil. While the predicted magnitudes of peak thermal contrasts exceeded the measured values, crossover times (times of day when the thermal contrast goes to zero) for both the real data and the simulations were within an hour of each other.

It is anticipated that this code can make reasonable, physics-based predictions for man-made target surface temperatures as well as those of background materials for any test site in the world. Such simulations can be used to anticipate times of day during which airborne sensors can be expected to have difficulty in detecting targets against natural backgrounds. As this code only computes physical temperatures of the surfaces, it cannot be expected to predict sensor performance. That would be a function of the sensitivity of the sensor detectors, the angular resolution of the sensor, and the algorithms that might be used to process measured data.

Furthermore, it must also be recognized that this code cannot simulate two- and three-dimensional effects, which certainly must play a significant part in determining the temperature distribution around a finite three-dimensional target in a large heterogeneous background. Such studies will require a computational test bed that operates on a much larger computer system than the author's PC.

Recommendations

Several improvements could be made to the code and analysis procedures that would create a more user-friendly environment in which to perform simulations. For example, as the original authors of the TSTM/VEGIE simulation code suggested, one could make the code be implicit in nature, in which the temperatures on the right side of Equation 9 would be written in terms of values at the end of the time-step. The resulting iterative solution would ensure time-step stability.

Another improvement would be to develop an optimization routine for determining material properties. The current process for finding an optimum set of surface material thermal and optical properties is best performed by varying each of the four property values in question (long-wave emissivity, short-wave absorptivity, thermal diffusivity, and thermal conductivity) one at a time and comparing the predicted surface temperatures to measured values. It requires a lot of analyst judgment and the use of the rules of thumb shown in Table 1 in the text. It should be possible to automate that search for an optimum set of values, perhaps using the standard deviation of the difference between the predicted temperatures and measured temperatures as an optimization metric.

References

Balick, L. K., Link, L. E., Scoggins, R. K, and Solomon, J. L. (1981a). "Thermal modeling of terrain surface elements," Technical Report EL-81-2, U.S. Army Engineer Waterways Experiment Station, Vicksburg, MS.

Balick, L. K., Scoggins, R. K., and Link, L. E. (1981b). "Inclusion of a simple vegetation layer in terrain temperature models for thermal infrared (IR) signature prediction," Miscellaneous Paper EL-81-4, U.S. Army Engineer Waterways Experiment Station, Vicksburg, MS.

Holman, J. P. (1968). *Heat Transfer*, 2nd ed., McGraw-Hill Book Company, New York.

Jumikis, A. R. (1977). *Thermal Geotechnics*, Rutgers University Press, New Brunswick, NJ.

Ochsner, T. E., Horton, R., and Ren, T. (2001). "A new perspective on soil thermal properties," *Soil Science Society of America Journal* 65, 1641-1647.

Ohanian, H. C. (1985). *Physics*, W.W. Norton and Company, New York.

Oke, T. R. (1987). *Boundary Layer Climates*, 2nd ed., Routledge, New York.

Welsh, J. P. (1994). "Smart weapons operability enhancement (SWOE) joint test and evaluation (JT&E) program, Final report," SWOE Report 94-10, U.S. Army Engineer Cold Regions Research and Engineering Laboratory, Hanover, NH.

Appendix A
Code Listing

```
C
C     TSTM:VEGIE----- BARE OR VEGETATED SURFACE TEMPERATURE MODEL
C---------------------------------------------------------------
C
C     THE FUNCTION OF THIS PROGRAM IS TO PREDICT THE PHYSICAL
C     TEMPERATURES OF SURFACES EXPOSED TO VARIOUS WEATHER
C     CONDITIONS                                        -
C
C        IT IS A 1-D FINITE DIFFERENCE CODE CURRENTLY LIMITED TO
C        6 LAYERS OF MATERIAL AND 150 NODES
C
C---------------------------------------------------------------
C
C        MAJOR REVISIONS IN SEPT-DEC 2004 BY JOHN CURTIS
C        REVISIONS INCLUDE MET STATION WEATHER FILES AS INPUT FOR
C        MULTIPLE-DAY SIMULATIONS AS WELL AS UTILIZING MOISTURE-
C        DEPENDENT SOIL THERMAL PROPERTIES AND OPTICAL PROPERTIES
C
C        MORE REVISIONS IN JUNE 2005 TO MAKE INPUT COMPATIBLE WITH
C        A GRAPHICAL USER INTERFACE DEVELOPED BY CURTIS
C
C---------------------------------------------------------------
C     PARAMETER (ND=30)
      DIMENSION THK(6),DEPTH(150),PROF(2,150),
     & XXX(30),YYY(30),LNUM(150)
      DIMENSION JD(15000),DT(15000),ATEMP(15000),RELHUM(15000),
     & BPRESS(15000),SOLAR(15000),WINDSP(15000),CLDTYPE(15000),
     & CLDCOV(15000),DEGSAT(15000),VMOIS1(15000),VMOIS2(15000),
     & STEMP1(15000),STEMP2(15000),RTEMP1(15000),RTEMP2(15000)
C
C      ARRAY FOR HOURLY TEMPERATURE PROFILES FOR DAY NSNGLDAY
C
      DIMENSION TEMPPROF(151,26)
C
      DIMENSION ALPHM(6),ALPHB(6),SFRQ(6),FKM(6),FKB(6)
      DIMENSION TITLE(7),CLABEL(17)
      DIMENSION CLR(8),NX(6),ATF(2),FEB(2)
      DIMENSION STOR(8,150),RR(6),INTR(7)
      REAL KTEMPG,KTEMPA,KTEMPT,LAT,ACL(8),BCL(8),M,KSQ
      CHARACTER DATE*8,HEADER*72,AN*1,CLABEL*10
```

```
              CHARACTER*30 FNAME
          DATA CLR/0.04,0.08,0.17,0.20,0.22,0.24,0.24,0.25/
          DATA  ACL/82.2,87.1,52.5,39.0,34.7,23.8,11.2,15.4/
          DATA  BCL/.079,.148,.112,.063,.104,.159,-.167,.028/
         DATA SIGMA,PI,AC,BC/8.12E-11,3.141593,
       &      17.269,35.86/
          DATA CC/0.261/
          DATA LAST,G,KSQ,CP/24,980.0,0.16,0.24/
C-----------------------------------------------------------
C        FORMAT STATEMENTS
C
 90    FORMAT(' botm bndry index=',I3)
 92    FORMAT(' botm bndry temp=',F6.1,' deg_C')
 95    FORMAT(' botm bndry heat flux=',F6.1,'cal/cm**2-min')
 97    FORMAT(5F8.2)
 120   FORMAT(A8,F6.1)
 139   FORMAT(1H\\)
 140   FORMAT(F5.1,F10.1,F6.1,F12.1,F12.1,F13.1)
 145   FORMAT(F6.1,F7.1)
 150   FORMAT(F12.1,I12,F11.2)
 160   FORMAT(3F8.2)
 170   FORMAT(F9.4,F12.1)
 180   FORMAT(I7,F6.1,F11.4,F8.2)
 190   FORMAT(4X,F5.2,F6.1,2X,I8,I9,I11,I9,I9,I8)
 195   FORMAT(4X,F5.2,1H;,F6.1,2X,1H;,I8,
     &  1H;,I9,1H;,I11,1H;,I9,1H;,I9,1H;,I8)
 200   FORMAT(4F10.4)
 210   FORMAT(I4,F9.2,F8.2,4F10.4)
 220   FORMAT(1H1)
 230   FORMAT(A72)
 235     FORMAT(' THIS.IS.NOT.A.SINGLE-DAY.SIMULATION'/I5,F5.2/)
 236     FORMAT(' THIS.IS.A.SINGLE-DAY.SIMULATION.FOR.DAY'/I5,F5.2/)
 240   FORMAT(4X,F7.2,3X,F5.1,5X,F6.2,10X,F6.2,9X,F7.2)
 250   FORMAT(6X,F6.1,13X,F4.1,12X,F5.1)
 260   FORMAT(9X,F8.1,F11.2,F9.2,F9.2,F8.2,F10.2,F8.2)
 310  FORMAT(1H;,'tot grybdy efectv ground foliage solar')
 320  FORMAT(1H;,'radnce temp temp temp insol')
 330  FORMAT('hr (W/m**2) (C) (C) (C) (W/m**2)')
 340  FORMAT(9X,'-----refl-nrefl----refl----nrefl',30(1H-))
 270  FORMAT(3X,F5.2,5X,I4,2X,I4,3X,F6.1,2X,F6.1,3X,F6.1,3X,F6.1,7X,I4)
 275  FORMAT(3X,F5.2,1H;,5X,1H;,I4,2X,1H;,
     & I4,3X,1H;,F6.1,2X,1H;,1H;,F6.1,3X,1H;,F6.1,3X,1H;,F6.1,7X,
     & 1H;,I4)
 350  FORMAT(' . . . . . . . . . sensbl latent')
 360  FORMAT(' jday hr jd&hr air surf grybdy solar surf
     &atms_ir heat heat rad1 rad2 surf-r1 surf-r2')
 370   FORMAT(' . . . temp temp radnce insol absorp emissn loss loss
     & temp temp temp temp')
 380  FORMAT(' . . . deg_C . ',23(1H-),'(W/m**2)',24(1H-))
  400  FORMAT(2H0 ,F5.2,4(3X,F5.1,' ',F5.2))
  410  FORMAT(10X,F5.1,1X,F5.2,3X,F5.1,1X,F5.2,3X,F5.1,1X,F5.2,
     & 3X,F5.1,1X,F5.2)
 2610 FORMAT(I4,4F8.3,6I6,4F7.2)
C
C       STATEMENT FUNCTIONS FOR USE IN VEGETATION SECTION
C
```

```
      E(T)=RH*6.108*EXP(AC*(T-273.15)/(T-BC))
      ESAT(T)=6.108*EXP(AC*(T-273.15)/(T-BC))
      Q(T)=0.622/(PRESS/E(T)-.378)
      QSAT(T)=0.622/(PRESS/ESAT(T)-.378)
C
C        FUNCTION STATEMENT FOR ALL LINEAR INTERPOLATION NEEDS
C        THIS INCLUDES WEATHER PARAMETERS AS A FUNCTION OF TIME AND
C        MOISTURE CONTENT AS A FUNCTION OF DEPTH
C
         YVALUE(XVALUE,Y1,Y2)=Y1+(XVALUE-X1)*(Y2-Y1)/(X2-X1)
C
         OPEN(2,STATUS='UNKNOWN')
         OPEN(4,STATUS='UNKNOWN')
C------------------------------------------------------------
C
C  ************   DATA INPUT   **********
C
C------------------------------------------------------------
C
C      INITIALIZE-VARIABLES-AND-CONSTANTS
C
      BB=-2.4E-4
      IBUG=0
      IEOF=0
      DO 100 I=1,6
      THK(I)=0.
      SFRQ(I)=0.
      ALPHM(I)=0.
      ALPHB(I)=0.
      FKM(I)=0.
  100 FKB(I)=0.
C
      DO 101 I=1,26
      DO 101 J=1,151
  101 TEMPPROF(J,I)=0
      JPROF=2
C
         DO 102 I=1,150
         DEPTH(I)=0.
 102     LNUM(I)=0
C
      SIGF=0.
      STATE=0.
      EPF=0.
      FOLA=0.
      HFOL=0.
C-----------------------------------------------------------
C
C        INPUT SIMULATION TITLE AND NO. OF FIELD WEATHER STATION
C        DATA LINES
C
      READ(2,*)HEADER,NLDATA
C-----------------------------------------------------------
C
C      INPUT FIELD WEATHER STATION DATA
C
```

```
C         INPUT DATA LINES CONTAIN: JULIAN DAY, 24-HR CLOCK TIME,
C         AIR TEMP, RELATIVE HUMIDITY, BAR PRESSURE,
C         SOLAR LOADING, WIND SPEED, CLOUD INDEX, CLOUD COVER PERCENT,
C         SURFACE DEGREE OF SATURATION (DECIMAL), MOISTURE CONTENT
(SHALLOW),
C         MOISTURE CONTENT (DEEP), SOIL TEMPERATURE (SHALLOW),
C         SOIL TEMPERATURE (DEEP), RADIOMETRIC TEMPERATURE 1,
C         RADIOMETRIC TEMPERATURE 2
C
C         IF NSOLAR=1 (A FLAG TO BE READ LATER), THEN THE CODE WILL
C         GENERATE SOLAR INSOLATION VALUES FOR THE ENTIRE INPUT FILE
C
          DO 800 I=1,NLDATA
          READ(2,*)JD(I),HR24,ATEMP(I),RELHUM(I),BPRESS(I),SOLAR(I),
     & WINDSP(I),CLDTYPE(I),CLDCOV(I),DEGSAT(I),VMOIS1(I),VMOIS2(I),
     & STEMP1(I),STEMP2(I),RTEMP1(I),RTEMP2(I)
C
C         A CORRECTION TO PREVENT THE RICHARDSON # FROM BLOWING UP
          IF(WINDSP(I).LT..1) WINDSP(I)=.1
C
C         CALCULATE A JULIAN DAY DECIMAL TIME
          DT(I)=INT(HR24/100.)+((HR24-INT(HR24/100.)*100.))/60.
C
C         CONVERSION OF TEMPERATURE TO DEG KELVIN
          ATEMP(I)=ATEMP(I)+273.15
C
C         CONVERSION OF RELATIVE HUMIDITY TO A DECIMAL VALUE
          RELHUM(I)=RELHUM(I)/100.
C
C         CONVERSION OF WIND SPEED TO CM/S
          WINDSP(I)=WINDSP(I)*100.
C
C         CONVERSION OF SOLAR LOADING TO (SMALL CAL)/(MIN-CM^2)
          SOLAR(I)=SOLAR(I)/697.6
C
C         CORRECTION FOR A BAD PRESSURE GUAGE
          IF(BPRESS(I).LT.0.) BPRESS(I)=1000.
 800      CONTINUE
C
C         SKIP THE LINE OF DATA CONTAINING "END"
          READ(2,*)DATE
C-----------------------------------------------------------
C
C         READ SINGLE-DAY SIMULATION PARAMETERS
C
C         NSINGLE =0 IF DOING A MULTIPLE-DAY SIMULATION
C         NSINGLE =1 IF DOING A SINGLE-DAY SIMULATION
C         NSNGLDAY = THE JULIAN DAY CHOSEN FOR A SINGLE-DAY SIMULATION
C
          READ(2,*) NSINGLE,NSNGLDAY
          NFIRST=1
C
C         IDENTIFY 1ST LINE OF DATA FOR A SINGLE-DAY SIMULATION
C
          IF(NSINGLE.EQ.0) GO TO 806
          DO 805 I=1,NLDATA
```

```
            IF(JD(I).NE.NSNGLDAY) GO TO 805
            NFIRST=I
            GO TO 806
 805    CONTINUE
C------------------------------------------------------------
C
C       INPUT SURFACE SLOPE INFO AND SOLAR CALCULATION FLAG
C
C        SFC SLOPE   SFC AZIMUTH  LATITUDE
C        DEG-HORIZ=0 DEG S=0      DEG
C
 806   READ(2,*)NSOLAR,SLOPE,SURFAC,LAT
      SLOPE=SLOPE*PI/180.0
      SURFAC=SURFAC*PI/180.
C
C     COMPUTE SOLAR INSOLATION, IF NECESSARY
C
            IF(NSOLAR.EQ.0) GO TO 88888
C
C       CALCULATE-INSOLATION-ON-SLOPE-SURFACE
C
            DO 8888 I=1,NLDATA
C
C       SOLVE-SOLAR-ZENITH
C
      TYME=DT(I)
            DAY=JD(I)
            NCLOUD=CLDTYPE(I)
            PRESS=BPRESS(I)
      T0=2.0*PI*(DAY-1.0)/365.0
      DECL=0.006918-0.399912*COS(T0)+0.070257*SIN(T0)
     &      -0.006758*COS(2.0*T0)+0.000907*SIN(2.0*T0)
     &      -0.002697*COS(3.0*T0)+0.001480*SIN(3.0*T0)
      ELF=(LAT/180*PI)
      TIMER=(TYME/12*PI)+PI
      IF(TIMER.GT.2.*PI)TIMER=TIMER-2.*PI
      AA=COS(DECL)*COS(ELF)*COS(TIMER)
      BB=SIN(DECL)*SIN(ELF)
      C=AA+BB
      Z=ACOS(C)
C
C       SOLVE-SOLAR-AZIMUTH
C
      XNUM=-COS(DECL)*SIN(TIMER)
      XDNOM=COS(ELF)*SIN(DECL)-SIN(ELF)*COS(TIMER)
      SAZ=ATAN(XNUM/XDNOM)
      IF(.NOT.(XNUM.LT.0.0.AND.XDNOM.GT.0.0)) GO TO 99944
      SAZ=SAZ+PI
      GO TO 99943
99944  IF(.NOT.(XNUM.GT.0.0.AND.XDNOM.GT.0.0)) GO TO 99943
      SAZ=SAZ-PI
99943  CONTINUE
C
C       CALCULATE-SLOPE-ATMOS-ATTEM-AND-CLOUD-ADJUSTMENTS
C
      SICF=COS(Z)*COS(SLOPE)+SIN(Z)*SIN(SLOPE)
```

```
     & *COS(SAZ-SURFAC)
      IF(.NOT.(SICF.LT.0.0.OR.COS(Z).LE.0.0)) GO TO 99941
      SUN=0.0
      GO TO 99942
99941 M=1/COS(Z)
      IF(.NOT.(M.GE.0.0)) GO TO 99939
      TAL=0.02023
      IF(DAY.GE.92.0 .AND. DAY.LE.152.0)TAL=-0.02290
         TA=ATEMP(I)
         RH=RELHUM(I)
      TD=5352.2/(21.4-ALOG(RH*ESAT(TA)))
      WATER=EXP(0.07074*(TD-273.15)+TAL)
      AB=0.271*(WATER*M)**0.303
      A0=0.085-0.247*ALOG10(PRESS/1000.*1./M)
      ARG1=((1.-AB)*0.349+(1.-A0)/(1.-A0*0.2)*0.651)
      GO TO 99940
99939 ARG1=1.0
99940 QP=2.0*ARG1
      QO=QP*SICF
      IF(.NOT.(NCLOUD.EQ.0)) GO TO 99937
      SUN=QO
      GO TO 99938
99937 CLOUD=CLDCOV(I)
      ARG2=-(BCL(NCLOUD)-.059)*M
      CTF=(ACL(NCLOUD)/94.4)*EXP(ARG2)
      SUN=QO-((CLOUD*CLOUD)*(QO-QO*CTF))
99938 CONTINUE
99942 SOLAR(I)=SUN
 8888  CONTINUE
88888  CONTINUE
C-----------------------------------------------------------
C
C      INPUT SURFACE OPTICAL PROPERTIES
C
C       SLOPE AND INTERCEPT OF EMISSIVITY EQUATION
C          SLOPE AND INTERCEPT OF ABSORBTIVITY EQUATION
C
C       IF SOIL IS THE TOP SURFACE, THESE MAY BE FUNCTIONS
C       OF SOIL VOLUMETRIC MOISTURE
C
C       IF ANOTHER MATERIAL IS THE TOP SURFACE,
C            THE SLOPE MAY BE SET TO
C       ZERO TO YIELD A CONSTANT VALUE OF PARAMETERS
C
       READ(2,*)EPSNM,EPSNB
           READ(2,*)SMALLAM,SMALLAB
C-----------------------------------------------------------
C
C        INPUT SURFACE DEGREE OF SATURATION FLAG AND VALUE
C
C        NSATFLAG =0 IF SOIL MOISTURE DATA ARE IN THE INPUT FILE
C        NSATFLAG =1 IF THE SURFACE DEGREE OF SAT VALUE IS FIXED
C        SATVAL = FIXED SURFACE DEGREE OF SAT VALUE (DECIMAL)
C
           READ(2,*)NSATFLAG,SATVAL
C
```

```
C         IF NSATFLAG IS NOT ZERO, THEN SET THE DEGSAT VALUE
C
          IF(NSATFLAG.EQ.0) GO TO 99985
          DO 99984 I=1,NLDATA
99984  DEGSAT(I)=SATVAL
C-----------------------------------------------------------
C
C       INPUT-VEGETATION-PARAMETERS
C
99985   IVEG=0
       READ(2,*)SIGF,STATE,EPF,FOLA,HFOL
       IF(SIGF.LE.0.0)GO TO 99969
       TF=ATEMP(1)
       IVEG=1
       EP1=EPF+EPSN-EPF*EPSN
       Z0=0.131*HFOL**0.997
       CH0=KSQ/(ALOG(ZASH/Z0)**2)
       ZDSP=0.701*HFOL**0.979
       CHH=KSQ/(ALOG((ZASH-ZDSP)/Z0)**2)
       CHG=(1.-SIGF)*CH0+SIGF*CHH
       DELTMP=1.
       QAF=QSAT(TF)
C-----------------------------------------------------------
C
C       INPUT LAYER SPECIFICATIONS
C
C        THICKNESS  VERT. GRID   THERMAL DIFF    HEAT COND
C         CM       SPACE-CM      CM**2/MIN      CAL/MIN-CM-K
C
99969  CONTINUE
          TOTTHICK=0.
          READ(2,*)NOMATL
       DO 99960 J4=1,6
       READ(2,*)THK(J4),SFRQ(J4),ALPHM(J4),ALPHB(J4),FKM(J4),FKB(J4)
          TOTTHICK=TOTTHICK+THK(J4)
99960  CONTINUE
C-----------------------------------------------------------
C
C      INPUT BOTTOM BOUNDARY DATA
C
C      IF IFLUXY LT 0, THERE IS A FIXED HEAT FLUX THROUGH THE
C         BOTTOM BOUNDARY
C      IF LFLUXY=0, THE BOTTOM BOUNDARY HAS A FIXED TEMPERATURE
C      IF IFLUXY GT 0, THERE IS AIRSPACE BENEATH BOTTOM
C
       READ(2,*)LFLUXY
       IF(.NOT.(LFLUXY.EQ.0)) GO TO 99962
       READ(2,*)DPRM0
       TB=0.
       DPRM0=DPRM0+273.15
       GO TO 99963
99962  IF(.NOT.(LFLUXY.LT.0)) GO TO 99961
C
       READ(2,*)DPRM1
       TB=FKM(NOMATL)*VMF+FKB(NOMATL)
       BEP=0.0
```

```
      BEP=0.
      BK=0.
      REP=0.
      TR=0.0
      FACTD=0.
      FACTE=0.
      RK=0.
      GO TO 99963
99961 READ(2,*)DPRM1,BEP,BK,REP,RK,TR
      TB=FKM(NOMATL)*VMF+FKB(NOMATL)
      TR=TR+273.15
      FACTD=SIGMA*RK*REP*TR**4
      FACTE=SIGMA*BK*BEP
99963 CONTINUE
C-----------------------------------------------------------
C
C        INPUT MOISTURE PROFILE PARAMETERS.  THESE STATEMENTS ARE
C        USED TO SET THE BOTTOM MOISTURE CONDITION AND THE
C        DEPTHS TO BE USED TO ESTABLISH A MOISTURE PROFILE AT
C        EACH POINT IN TIME
C
C        THE FIRST DATA ENTRY IS THE DEPTH (CM) BELOW WHICH THE
C        MOISTURE CONTENT IS TAKEN TO BE FIXED; THE SECOND DATA
C        ENTRY IS THAT MOISTURE LEVEL (%)
C
C        THE NEXT LINE OF DATA CONTAINS TWO DEPTHS FOR WHICH
C        MOISTURE CONTENT VALUES ARE CONTAINED IN THE INPUT FILE
C
C        IF THERE IS NO MOISTURE DATA, THE RELATIONSHIPS FOR
C        MATERIAL PROPERTIES WILL REFLECT THAT.  THE SLOPE WILL BE
C        GIVEN AS ZERO, AND THE INTERCEPT WILL BE THE CONSTANT
C        PROPERTY VALUE THAT WILL BE USED BY THE PROGRAM
C
         READ(2,*) ZMF,VMF
         READ(2,*) ZM1,ZM2
C-----------------------------------------------------------
C
C      INPUT SIMULATION CONTROLS
C
C        TIME STEP (MIN), PRINT FREQ (MIN), NO. OF 1ST DAY
C        ITERATIONS, WIND SPEED INDICATOR HEIGHT (CM)
C
      READ(2,*)TFRQ,TPRNT,REPDAY,ZASH
         IPRNT=TPRNT/TFRQ
         IPPRNT=60/TFRQ
         NPRNT=1
         NPPRNT=1
C-----------------------------------------------------------
C
C        SET INITIAL TOP SURFACE PARAMETERS
C
         X1=ZM1
         X2=ZM2
         VMSURF=YVALUE(0.,VMOIS1(1),VMOIS2(1))
         IF(NSATFLAG.NE.0) VMSURF=SATVAL*0.4
         EPSN=EPSNM*VMSURF+EPSNB
```

```
            SMALLA=SMALLAM*VMSURF+SMALLAB
        FACTA=SIGMA*EPSN
C---------------------------------------------------------
C
C        *************  PRINT-INPUT-DATA  *********************
C
C---------------------------------------------------------
C

       WRITE(4,*)' '
       WRITE(4,230)HEADER
       WRITE(4,*)' '
            IF(NSINGLE.EQ.0) WRITE(4,235)NSNGLDAY,SATVAL
            IF(NSINGLE.NE.0) WRITE(4,236)NSNGLDAY,SATVAL
            WRITE(4,*)' '
       WRITE(4,*)' SHLT_HT_CM'
       WRITE(4,150) ZASH
       WRITE(4,*)' '
       WRITE(4,*)' SURFACE-ORIENTATION-SPECIFICATIONS'
       WRITE(4,*)' sfc_slp sfc_az latitude'
       WRITE(4,*)' deg-hor=0 deg_S=0 deg'
       WRITE(4,160)SLOPE*180/PI,SURFAC*180.0/PI,LAT
       WRITE(4,*)' '
       WRITE(4,*)' HEAT-FLOW-CACULATION-CONTROLS'
       WRITE(4,*)' no_of no_24 time_stp prn_freq'
       WRITE(4,*)' layers hr_reps min min'
       WRITE(4,*)' <=6'
       WRITE(4,180)NOMATL,REPDAY,TFRQ,TPRNT
       WRITE(4,*)' '
       WRITE(4,*)' TOP-SURFACE-CONSTANTS'
       WRITE(4,*)' emiss-m emiss-b absrb-m absrb-b'
       WRITE(4,200)EPSNM,EPSNB,SMALLAM,SMALLAB
       WRITE(4,*)' '
       WRITE(4,*)' MATERIAL-LAYER-SPECIFICATIONS'
      WRITE(4,*)' layer thknss node-sp diff-m diff-b cond-m cond-b'
       WRITE(4,*)' no. cm cm . cm^2/min . cal/min-cm-deg-K'
C     FOLLOWING ADDED ON 21 Aug 2004 TO HELP MAKE ALL OUTPUT FILES
C          THE SAME SIZE
       DO 99956 J4=1,6
       WRITE(4,210)J4,THK(J4),SFRQ(J4),
     &    ALPHM(J4),ALPHB(J4),FKM(J4),FKB(J4)
99956  CONTINUE
       WRITE(4,*) TOTTHICK
       WRITE(4,*)' BOTTOM_BOUNDARY_THERMAL_CONDITIONS'
       IF(LFLUXY.NE.0) GO TO 99958
       WRITE(4,90)LFLUXY
       WRITE(4,92)DPRM0-273.15
       WRITE(4,*)' '
       WRITE(4,*)' '
       GO TO 99959
99958  IF(LFLUXY.GT.0) GO TO 99957
       WRITE(4,90)LFLUXY
       WRITE(4,95)DPRM1
       WRITE(4,*)' '
            WRITE(4,*)' '
       GO TO 99959
```

```
99957  WRITE(4,90)LFLUXY
      WRITE(4,95)DPRM1
     WRITE(4,*)' ---BOTTOM SURFACE-----',2H;;,'SURF_BENTH AIRSP_TEMP'
     WRITE(4,*)' EMISS GEO_SHAPE EMISS GEO_SHAPE DEG_C'
      WRITE(4,*)1H;,'FACT(0.-1.)',1H;,'FACT(0.-1.)'
      WRITE(4,97)BEP,BK,REP,RK,TR-273.15
      WRITE(4,*)' '
99959  CONTINUE
            WRITE(4,*)'FIXED_SOIL_MOISTURE_BOUNDARY_CONDITIONS'
            WRITE(4,*)'depth to fixed moisture (cm)',ZMF
            WRITE(4,*)'fixed volumetric moisture value (%)',VMF
            WRITE(4,*)' '
            WRITE(4,*)'DEPTHS_OF_TWO_MEASURED_VOLUMETRIC_SOIL_MOISTURES_(cm
)'
            WRITE(4,*)ZM1,ZM2
            WRITE(4,*)' '
      CALL FLUSH()
      WRITE(4,*)' VEGETATION_PARAMETERS'
      WRITE(4,*)'  covrg state emiss absorb fol_ht'
      WRITE(4,*)'(0.0-1.0) . (0.0-1.0) (0.0-1.0) (cm)'
      WRITE(4,240)SIGF,STATE,EPF,FOLA,HFOL
      WRITE(4,*)' '
      WRITE(4,*)' '
      WRITE(4,*)' '
      CALL FLUSH()
C--------------------------------------------------------------
C
C     WRITE COLUMN HEADINGS TO OUTPUT FILE
C
      IF(IVEG.GT.0)  GO TO 1420
      WRITE(4,350)
      WRITE(4,360)
      WRITE(4,370)
      WRITE(4,380)
      CALL FLUSH()
      GO TO 1425
 1420  WRITE(4,310)
      WRITE(4,320)
      WRITE(4,330)
      WRITE(4,340)
      CALL FLUSH()
CC------------------------------------------------------------
C
C       SET-UP-INITIAL-CONDITIONS
C
 1425    NDTS=NFIRST
            NSTEP=1
            TIME=DT(1)
      DIST=0.
      IFLAG=0
      DELT=TFRQ/60.
            TEMPPROF(1,1)=NSNGLDAY
C
C         IX=LAYER NUMBER; IY=DEPTH SUBSCRIPT (1 AT SURFACE;
C         JMAX AT BOT BNDARY)
C
```

```
      IX=1
      IY=1
      GO TO 99913
99914  IF(IX.GT.NOMATL) GO TO 99912
99913  INTR(IX)=IY
      IF (SFRQ(IX).LE.0.) SFRQ(IX)=THK(IX)/10.
      NX(IX)=MAX1(THK(IX)/SFRQ(IX)+.9,1.1)
      RR(IX)=60.0*DELT/(SFRQ(IX)*SFRQ(IX))
      LAYERS=0
      GO TO 99910
99911  IF(LAYERS.GT.NX(IX)) GO TO 99909
99910  DEPTH(IY)=DIST
         TEMPPROF(IY+1,1)=DEPTH(IY)
         LNUM(IY)=IX
C
C        CALCULATE MOISTURE FOR THIS DEPTH
C
         IF(DEPTH(IY).LT.ZMF) GO TO 2100
         VOLMOIS=VMF
         GO TO 99908
 2100 IF(DEPTH(IY).LT.ZM2) GO TO 2050
         X1=ZM2
         X2=ZMF
         VOLMOIS=YVALUE(DEPTH(IY),VMOIS2(1),VMF)
         GO TO 99908
 2050 X1=ZM1
         X2=ZM2
         VOLMOIS=YVALUE(DEPTH(IY),VMOIS1(1),VMOIS2(1))
99908   CONTINUE
C
C        RETRIEVE INITIAL MATERIAL PROPERTY VALUES AT EACH DEPTH
C
         VALK=FKM(IX)*VOLMOIS+FKB(IX)
         VALALPH=ALPHM(IX)*VOLMOIS+ALPHB(IX)
      STOR(6,IY)=VALK
      STOR(7,IY)=VALK/VALALPH
         STOR(8,IY)=VALALPH
      STOR(4,IY)=0.
      STOR(2,IY)=STOR(6,IY)
      STOR(3,IY)=STOR(7,IY)
      IY=IY+1
      DIST=DIST+SFRQ(IX)
      LAYERS=LAYERS+1
      GO TO 99911
99909  IX=IX+1
      DIST=DIST-SFRQ(IX-1)
      GO TO 99914
99912  JMAX=IY-1
      INTR(IX)=JMAX
C
C        SET INITIAL TEMPERATURE PROFILE AS A LINEAR FIT
C        BETWEEN THE INITIAL AIR TEMPERATURE AND EITHER
C        A FIXED BOTTOM BOUNDARY TEMPERATURE OR A VALUE
C        OF 10 DEG C (283.15 K)
C
         YYY(1)=ATEMP(1)
```

```
            YYY(2)=283.15
            IF(LFLUXY.EQ.0) YYY(2)=DPRM0
            DO 2000 I=1,JMAX
            X1=0.
            X2=TOTTHICK
            STOR(1,I)=YVALUE(DEPTH(I),YYY(1),YYY(2))
            STOR(5,I)=STOR(1,I)
 2000 CONTINUE
C------------------------------------------------------------
C
C   ******    THIS IS THE SIMULATION CONTROL LOOP    ******
C
C------------------------------------------------------------
c
C     RUN-HEAT-FLOW-PROGRAM
C
99919  CONTINUE
       ASSIGN 99917 TO I99918
       GO TO 99918
C
C     WRITE SIMULATION RESULTS FOR THIS TIME INCREMENT
C         TO THE OUTPUT FILE (IF PRINT PARAMETERS ARE MET)
C
99917  ASSIGN 99915 TO I99916
       GO TO 99916
99915  CONTINUE
            CALL FLUSH()
C
C         GO TO THE NEXT TIME STEP
C
       GO TO 99919
C
C    TERMINATE THE SIMULATION AND STORE TEMPERATURE PROFILES
C
99980   CONTINUE
        WRITE(4,*)'NORMAL TERMINATION'
        CALL FLUSH()
C
C      OUTPUT THE TEMPERATURE PROFILES
C
        DO 104 I=1,JMAX+1
        WRITE(4,103) (TEMPPROF(I,J),J=1,26)
  104   CONTINUE
  103   FORMAT(26F8.3)
        CALL FLUSH()
        STOP
C------------------------------------------------------------
C
C    ************  DO THE CALCULATIONS  **************
C
C------------------------------------------------------------
99918  CONTINUE
C
            NSTEP=NSTEP+1
       TIME=NSTEP*DELT
            IF(TIME.LE.24.) GO TO 940
```

```
      TIME=DELT
      NSTEP=1
      NDTS=NDTS+1
      IF(REPDAY.EQ.0) GO TO 940
C
C     ITERATION ON THE FIRST DAY
C
      NDTS=NFIRST
      REPDAY=REPDAY-1
  940 IF(NDTS.GE.NLDATA) GO TO 99980
      IF(TIME.LE.DT(NDTS+1)) GO TO 938
      IF(DT(NDTS+1).LT.DT(NDTS)) GO TO 938
C     PREVIOUS LINE A CHECK FOR MIDNIGHT BEING LABELED 0000 HRS
      NDTS=NDTS+1
      GO TO 940
  938 CONTINUE
C
C     CHECK FOR THE END OF A SINGLE-DAY SIMULATION
C
      IF(NSINGLE.NE.0.AND.JD(NDTS).NE.NSNGLDAY) GO TO 99980
C
C     CALCULATE ALL TIME-BASED INTERPOLATED PARAMETER VALUES
C
      X1=DT(NDTS)
      X2=DT(NDTS+1)
      IF(DT(NDTS+1).LT.DT(NDTS)) X2=DT(NDTS+1)+24.
C     PREVIOUS LINE A CORRECTION FOR MIDNIGHT BEING LABELED 0000 HRS
      AT= YVALUE(TIME,ATEMP(NDTS),ATEMP(NDTS+1))
      TA=AT
      CTEMA=AT
      RH= YVALUE(TIME,RELHUM(NDTS),RELHUM(NDTS+1))
      PRESS=YVALUE(TIME,BPRESS(NDTS),BPRESS(NDTS+1))
      FACTH=(1000/PRESS)**.286
C     FACTH IS A FACTOR USED IN CALCULATING THE CONVECTION TERM
      SOL= YVALUE(TIME,SOLAR(NDTS),SOLAR(NDTS+1))
      BTERM=SOL
      SPEED=YVALUE(TIME,WINDSP(NDTS),WINDSP(NDTS+1))
      UA=SPEED
      CLOUD=YVALUE(TIME,CLDCOV(NDTS),CLDCOV(NDTS+1))
      WET=YVALUE(TIME,DEGSAT(NDTS),DEGSAT(NDTS+1))
      VM1=YVALUE(TIME,VMOIS1(NDTS),VMOIS1(NDTS+1))
      VM2=YVALUE(TIME,VMOIS2(NDTS),VMOIS2(NDTS+1))
      ST1=YVALUE(TIME,STEMP1(NDTS),STEMP1(NDTS+1))
      ST2=YVALUE(TIME,STEMP2(NDTS),STEMP2(NDTS+1))
      RT1=YVALUE(TIME,RTEMP1(NDTS),RTEMP1(NDTS+1))
      RT2=YVALUE(TIME,RTEMP2(NDTS),RTEMP2(NDTS+1))
C
C     CALCULATE NEW MOISTURE AND THERMAL PROPERTIES PROFILES
C
      DO 2400 IPROF=1,JMAX
      IX=LNUM(IPROF)
      IF(DEPTH(IPROF).LT.ZMF) GO TO 2330
      VOLMOIS=VMF
      GO TO 2350
 2330 IF(DEPTH(IPROF).LT.ZM2) GO TO 2340
      X1=ZM2
```

```
          X2=ZMF
          VOLMOIS=YVALUE(DEPTH(IPROF),VM2,VMF)
          GO TO 2350
 2340   X1=ZM1
          X2=ZM2
          VOLMOIS=YVALUE(DEPTH(IPROF),VM1,VM2)
 2350    CONTINUE
          VALK=FKM(IX)*VOLMOIS+FKB(IX)
          VALALPH=ALPHM(IX)*VOLMOIS+ALPHB(IX)
          STOR(1,IPROF)=STOR(5,IPROF)
       STOR(6,IPROF)=VALK
       STOR(7,IPROF)=VALK/VALALPH
          STOR(8,IPROF)=VALALPH
 2400 CONTINUE
C
C         SET TOP SURFACE PARAMETERS FOR THIS TIME INCREMENT
C
          X1=ZM1
          X2=ZM2
          VMSURF=YVALUE(0.,VM1,VM2)
          IF(NSATFLAG.NE.0) VMSURF=SATVAL*0.4
          EPSN=EPSNM*VMSURF+EPSNB
          SMALLA=SMALLAM*VMSURF+SMALLAB
      FACTA=SIGMA*EPSN
C
C         COUNTERS FOR PRINT OUTPUT AND PROFILE OUTPUT
C
          NPRNT=NPRNT+1
          IF(JD(NDTS).EQ.NSNGLDAY) NPPRNT=NPPRNT+1
99907  ZZA=STOR(5,1)
      ZZB=STOR(5,JMAX)
      TEML=ZZA
      TEMR=ZZB
C
C     CALCULATE-BOUNDARY-CONDITIONS
C
      IF(IVEG.EQ.0)GO TO 930
      ASSIGN 99905 TO I99800
      GO TO 99800
 930   ASSIGN 99905 TO I99906
      GO TO 99906
C
C     CALCULATE-UPPER-BOUNDARY-VALUES
C
99905  IF(IVEG.EQ.0) GO TO 900
      ASSIGN 99903 TO I99797
      GO TO 99797
 900   ASSIGN 99903 TO I99904
      GO TO 99904
C
99903  IX=1
      J=2
      IMATL=NOMATL
      IF(NOMATL.NE.1) GO TO 99896
      IZ=NX(IX)-1
      IF(IZ.LE.0) GO TO 99902
```

```
C
C     CALCULATE-INSIDE-MATERIAL-VALUES WHEN THERE IS
C         ONLY A SINGLE LAYER OF MATERIAL
C
      ASSIGN 99902 TO I99899
      GO TO 99899
C
99896  IF(IMATL.EQ.1) GO TO 99893
      IZ=NX(IX)-1
      IF(IZ.LE.0) GO TO 99892
C
C     CALCULATE-INSIDE MATERIAL-VALUES WHEN THERE IS
C         MORE THAN ONE LAYER OF MATERIAL
C
      ASSIGN 99892 TO I99899
      GO TO 99899
C
C     CALCULATE-INTERFACE-VALUES
C
99892  ASSIGN 99894 TO I99890
      GO TO 99890
C
C     CALCULATE-INSIDE-MATERIAL-VALUES FOR THE LAST
C         LAYER OF MATERIAL
C
99893  IZ=NX(IX)-1
      IF(IZ.LE.0) GO TO 99902
      ASSIGN 99902 TO I99899
      GO TO 99899
99894  IMATL=IMATL-1
      GO TO 99896
C
C     CALCULATE-LOWER-BOUNDARY-VALUES
C
99902  ASSIGN 99883 TO I99886
      GO TO 99886
C
99883  GO TO I99918
C------------------------------------------------------------
C
C    ******          END OF CALCULATIONS FOR THIS TIME STEP  ******
C
C------------------------------------------------------------
99906  CONTINUE
C
C       CALCULATE-BOUNDARY-CONDITIONS
C
         X1=ZM1
         X2=ZM2
         VMSURF=YVALUE(0.,VM1,VM2)
         IF(NSATFLAG.NE.0) VMSURF=SATVAL*0.4
      B = -FKM(1)*VMSURF-FKB(1)
      T=TIME
      IF(BTERM.GT.0.0)BTERM=BTERM*SMALLA
C
C     CALCULATE BOTTOM BOUNDARY HEAT TERMS (APRM,DPRM,BPRM)
```

```
C
          ASSIGN 99880 TO I99881
          GO TO 99881
C
C         CALCULATE THE ATMOSPHERIC IR EMISSION (ATERM)
C
99880     ASSIGN 99878 TO I99879
          GO TO 99879
C
C         CALCULATE CONVECTION (HTERM)
C
99878     ASSIGN 99876 TO I99877
          GO TO 99877
C
C         CALCULATE EVAPORATIVE HEAT LOSS (DTERM)
C
99876     ASSIGN 99874 TO I99875
          GO TO 99875
C
99874     D= ATERM + BTERM - HTERM - DTERM
          GO TO I99906
C
C-----------------------------------------------------------
C         BOTTOM BOUNDARY HEAT TERMS
C
99881  BPRM=TB
     IF(.NOT.(TB.EQ.0.0)) GO TO 99872
     APRM=1.0
     DPRM=DPRM0
     GO TO 99873
99872  APRM=FACTE*TEMR*TEMR*TEMR
     DPRM=DPRM1+FACTD
99873  GO TO I99881
C
C-----------------------------------------------------------
C       ATMOSPHERIC-INFRARED-EMISSION-ATERM
C
99879  TAK=TA
     TAC=(TAK-273.15)
     EA=6.108*RH*EXP((AC*TAC)/(TAK-BC))
     ALPHI=(0.61+0.05*SQRT(EA))*(1.0+(CLR(NCLOUD)*(CLOUD**2)))
     DOWNIR=0.8132E-10*TAK**4*ALPHI
     ATERM=DOWNIR
          GO TO I99879
C
C-----------------------------------------------------------
C     CALCULATE-CONVECTION-HTERM
C
C         TA IS THE AIR TEMPERATURE
C         TEML IS THE SURFACE TEMPERATURE
C
99877  TAK=TA
     TSK=TEML
     RHOA=-0.001*0.348*PRESS/TAK
 1200  THETAZ=TAK*FACTH
     THETAS=TSK*FACTH
```

```
      DTHETA=(THETAZ-THETAS)/ZASH
      DU=SPEED/ZASH
      THETAV=(THETAZ+THETAS)/2.0
      RI=G*DTHETA/(THETAV*DU**2)
      COE1=15.0
      COE2=1.175
      EX=.75
      IF(TSK.GT.TAK)GO TO 31
      IF(RI.GT.0.2)RI=.19999
      COE1=5.0
      COE2=1.0
      EX=2.0
  31  HTER=RHOA*KSQ*(ZASH/ALOG(ZASH))**2*DU
     & *(COE2*(1.0-COE1*RI)**EX)
C
C         JOHN CURTIS REPLACED ZASH IN 31
C         WITH THE LOGARITHMIC HEIGHT (ZASH/ALOG(ZASH))
C         BASED ON A REVIEW OF OKE'S FORMULATION.
C
      HTERM=HTER*CP*DTHETA
99864  GO TO I99877
C
C---------------------------------------------------------
C     CALCULATE-EVAPORATIVE-HEAT-LOSS-DTERM
C
C
99875   CONTINUE
      IF(.NOT.(TEML.GT.TA)) GO TO 99860
          CTEMA=TA
      KTEMPA=CTEMA
      CTEMA=CTEMA-273.15
      KTEMPG=TEML
      ES=EXP((AC*(KTEMPG-273.15))/(KTEMPG-BC))*6.1071
      EA=EXP((AC*CTEMA)/(KTEMPA-BC))*6.1071*RH
      DG=0.622/PRESS*(EA-ES)*WET/ZASH
      XL=597.3-0.566*(CTEMA+KTEMPG-273.15)/2.0
      DTERM=HTER*XL*DG
      GO TO 99861
99860  DTERM=0.0
99861   GO TO I99875
C---------------------------------------------------------

99904  CONTINUE
C      CALCULATE-UPPER-BOUNDARY-VALUES
C
C         T1 IS AN ESTIMATE FOR THE TEMPERATURE OF THE FIRST NODE BELOW
C         THE SURFACE AT THE END OF THIS TIME INCREMENT; FOUND USING
C         THE 1-D HEAT FLOW EQUATION
C
      T1=STOR(8,1)*RR(1)*(STOR(1,3)-2.*STOR(1,2)+STOR(1,1))+STOR(1,2)
      III=0
  830  III=III+1
C
C         T2 IS F(Ts)/(PARTIAL OF F WRT Ts), WHICH IS THE CHANGE IN Ts
C         FROM THE NEWTON METHOD FOR SOLVING F(Ts)=0
C
```

```
C         F(Ts) IS EQUATION (4) IN THE ORIGINAL REPORT
C
      T2=STOR(5,1)**4*FACTA*SFRQ(1)+STOR(6,1)*STOR(5,1)
     &   -(STOR(6,1)*T1+D*SFRQ(1))
      T2=T2/(4.*FACTA*SFRQ(1)*STOR(5,1)**3+STOR(6,1)-SFRQ(1)*DDDT)
      STOR(5,1)=STOR(5,1)-T2
      TEML=STOR(5,1)
      ASSIGN 825 TO I99877
C
C         GET HTERM
C
      GO TO 99877
  825  ASSIGN 810 TO I99875
C
C         GET DTERM
C
      GO TO 99875
  810  DNEW=ATERM+BTERM-HTERM-DTERM
      IF(ABS(T2).LT.0.005 .OR. III.GT.5)GO TO I99904
      DDDT=-(DNEW-D)/T2
      D=DNEW
      GO TO 830
C
C-----------------------------------------------------------
99899  CONTINUE
C       CALCULATE-INSIDE-MATERIAL-VALUES
C
      GO TO 99856
99857  IF(IZ.LE.0) GO TO 99855
99856  CONTINUE
      STOR(5,J)=STOR(1,J)+STOR(8,IX)*RR(IX)*(STOR(1,J-1)-2.*STOR(1,J)
     &      +STOR(1,J+1))
C         WRITE(4,*)J,IZ,IX,STOR(1,J),STOR(5,J),STOR(8,IX),RR(IX)
      J=J+1
      IZ=IZ-1
      GO TO 99857
99855  GO TO I99899
C-----------------------------------------------------------
C       CALCULATE-INTERFACE-VALUES
C
99890  CONTINUE
      BCOEF=STOR(6,J-1)/SFRQ(IX)
      DCOEF=STOR(6,J+1)/SFRQ(IX+1)
      CCOEF=BCOEF+DCOEF
      ACOEF=BCOEF/(2.*STOR(8,IX)*RR(IX))+DCOEF/(2.*STOR(8,IX+1)
     &*RR(IX+1))
      STOR(5,J)=STOR(1,J)+(BCOEF*STOR(1,J-1)-CCOEF*STOR(1,J)+DCOEF*
     &      STOR(1,J+2))/ACOEF
      STOR(5,J+1)=STOR(5,J)
C         WRITE(4,*)J,IZ,IX,STOR(1,J),STOR(5,J),STOR(8,IX),RR(IX)
      IX=IX+1
      J=J+2
      GO TO I99890
C-----------------------------------------------------------
C       CALCULATE-LOWER-BOUNDARY-VALUES
99886  IF(LFLUXY.EQ.0) GO TO 880
```

```
        I=1
  870  CONTINUE
        F2=4.0*FACTE*STOR(5,J)**3 - BPRM
CCC       F2=4.*APRM-BPRM
        IF(F2.EQ.0)F2=.000001
        F2= -(FACTE*SFRQ(IX)*STOR(5,J)**4-BPRM*STOR(5,J)
     &              +BPRM*STOR(5,J-1)-DPRM*SFRQ(IX))/F2
        STOR(5,J)=STOR(5,J) + F2
        PRINT *,I,F2
        I=I+1
        IF(I.LE.3) GO TO 870
 880   IF(LFLUXY.EQ.0) STOR(5,J)=STOR(5,J)
        GO TO I99886
C-----------------------------------------------------------
99916  CONTINUE
C
C        PRINT-OUTPUT
C
         DO 99842 JKX=1,NOMATL+1
         IJ=INTR(JKX)
         TITLE(JKX)=(STOR(5,IJ)-273.15)
99842   CONTINUE
            STEMP=STOR(5,1)-273.15
         PRINTHR=AMOD(TIME,24.0)
         IF(PRINTHR.EQ.0.)PRINTHR=24
         IF(IVEG.EQ.1) GO TO 1110
         IGBR=5.67E-8*EPSN*STOR(5,1)**4
         ISOL=BTERM/SMALLA*697.6+0.5
         IABSOR=ISOL*SMALLA
         IATERM=ATERM*697.6
         IHTERM=HTERM*697.6+0.5
         IDTERM=DTERM*697.6+0.5
            IF(NPRNT.LT.IPRNT) GO TO 99844
            IF(REPDAY.GT.0) GO TO 99843
         WRITE(4,2610)JD(NDTS),PRINTHR,JD(NDTS)+PRINTHR/24.,
     &TA-273.15,STEMP,IGBR,
     & ISOL,IABSOR,IATERM,IHTERM,IDTERM,RT1,RT2,STEMP-RT1,STEMP-RT2
            CALL FLUSH()
c           WRITE(4,*)TAK,TSK,PRESS,RHOA,FACTH,THETAZ,THETAS,DTHETA,DU,
c     & RI,THETAV,KSQ,ZASH
99843   NPRNT=0
99844   CONTINUE
            IF(JD(NDTS).NE.NSNGLDAY) GO TO I99916
            IF(NPPRNT.LT.IPPRNT) GO TO I99916
            IF(REPDAY.GT.0) GO TO 7778
C
C    CAPTURE TEMPERATURE VS DEPTH PROFILES AT HOURLY INTERVALS
C
         TEMPPROF(1,JPROF)=PRINTHR
         DO 7777 IPROF=1,JMAX
         TEMPPROF(IPROF+1,JPROF)=STOR(1,IPROF)-273.15
 7777   CONTINUE
         JPROF=JPROF+1
 7778    NPPRNT=0
            GO TO I99916
C
```

```
 1110   ASSIGN 1400 TO I1410
        GO TO 1410
 1400  CONTINUE
        WRITE(4,270)PRINTHR,ISURFG+IREFRA,ISURFG,TEFFR-273.15,
      &   TEFF-273.15,TEML-273.15,TF-273.15,ISOL
         CALL FLUSH()
        GO TO I99916
C
C-----------------------------------------------------------
99800  CONTINUE
C         CALCULATE-BOUNDARY-CONDITIONS-WITH-VEG
         T=TIME
C
C       ATMOSPHERIC-INFRARED-EMISSION-ATERM
         ASSIGN 980 TO I99879
         GO TO 99879
C
 980     CONTINUE
         IF(UA.LT.10.0)UA=10.0
         UAF=0.83*SIGF*UA*SQRT(CHH)+(1.-SIGF)*UA
         DELTMP=5.
         CF=0.01*(1.+30.0/UAF)
         DU=(UA-UAF)/ZASH
         RS=1/(.05+.0021*(SOL*697.6))
         RC=RS*STATE/(7.0*SIGF)
         ATF(1)=TF
         ASSIGN 1210 TO I950
         GO TO 950
1210    CONTINUE
         FEB(1)=FENB
         NDEX=0
1240    TF=TF+DELTMP
         NDEX=NDEX+1
         ASSIGN 1220 TO I950
         GO TO 950
1220    CONTINUE
         FEB(2)=FENB
         IF(FEB(1)*FEB(2).LT.0.0) GO TO 1230
         IF(ABS(FEB(2)).GT.ABS(FEB(1)))DELTMP=-5.
         IF(NDEX.LT.100)GO TO 1240
         WRITE(4,*)'FOLIAGE ENERGY BUDGET HAS NOT CROSSED X-AXIS'
         WRITE(4,*)'AFTER 100 SEARCH STEPS. CHECK INPUT DATA.'
         CALL FLUSH()
         STOP
1230    CONTINUE
         ATF(2)=TF
1270    SLOPE1=(FEB(2)-FEB(1))/(ATF(2)-ATF(1))
         BINT=FEB(1)-SLOPE1*ATF(1)
         TF0=-BINT/SLOPE1
         IF(ABS(TF-TF0).LE.0.001)GO TO 1260
         TF=TF0
         ASSIGN 1250 TO I950
         GO TO 950
1250    CONTINUE
         IF(FENB*FEB(2).GT.0.0)IP=2
         IF(FENB*FEB(1).GT.0.0)IP=1
```

```
      ATF(IP)=TF
      FEB(IP)=FENB
      GO TO 1270
1260  GO TO I99800
C-----------------------------------------------------------
C     CALCULATE-UPPER-BOUNDARY-VALUES-FOR-FOLAGE
99797 CONTINUE
      DELTMP=5.
      ATF(1)=TEML
      ASSIGN 1310 TO I1300
      GO TO 1300
1310  CONTINUE
      FEB(1)=FENB
      NDEX=0
1340  TEML=TEML+DELTMP
      NDEX=NDEX+1
      ASSIGN 1320 TO I1300
      GO TO 1300
1320  CONTINUE
      FEB(2)=FENB
      IF(FEB(1)*FEB(2).LT.0.0) GO TO 1330
      IF(ABS(FEB(2)).GT.ABS(FEB(1)))DELTMP=-5.
      IF(NDEX.LT.100)GO TO 1340
      WRITE(4,*)'GROUND ENERGY BUDGET HAS NOT CROSSED X-AXIS'
      WRITE(4,*)'AFTER 100 SEARCH STEPS. CHECK INPUT DATA.'
      CALL FLUSH()
      STOP
1330  CONTINUE
      ATF(2)=TEML
1370  SLOPE1=(FEB(2)-FEB(1))/(ATF(2)-ATF(1))
      BINT=FEB(1)-SLOPE1*ATF(1)
      TF0=-BINT/SLOPE1
      IF(ABS(TEML-TF0).LE.0.001)GO TO 1360
      TEML=TF0
      ASSIGN 1350 TO I1300
      GO TO 1300
1350  CONTINUE
      IF(FENB*FEB(2).GT.0.0)IP=2
      IF(FENB*FEB(1).GT.0.0)IP=1
      ATF(IP)=TEML
      FEB(IP)=FENB
      GO TO 1370
1360  STOR(5,1)=TEML
      GO TO I99797
C-----------------------------------------------------------
C     CALCULATE-ENERGY-BUDGET
 950  TAF=(1.-SIGF)*TA+SIGF*(0.3*TA+0.6*TF+0.1*TEML)
      DTHETA=(TA-TF)*FACTH/ZASH
      THETAV=(TA+TF)*FACTH/2.0
      RI=G*DTHETA/(THETAV*DU**2)
      RHOAF=-0.001*.348*PRESS/((TF+TA)/2.)
      COE1=15.
      COE2=1.175
      EX=.75
      IF(RI.LE.0.)GO TO 1280
      IF(RI.GT.0.2)RI=0.199
```

```
        COE1=5
        COE2=1.
        EX=2.0
1280    CONTINUE
        HTER=RHOAF*KSQ*ZASH**2*DU
C    &   *COE2*(1.-COE1*RI)**EX
C       HSF=1.1*7.*SIGF*CP*CF*UAF*(TF-TAF)*60.
        HSF=HTER*CP*DTHETA*60.
C       XL=597.3-0.566*TAF
        RA=(ALOG((ZASH-ZDSP)/Z0)*COE2*((1.-COE1*RI)**EX))**2
     &   /(.16*UA)
       RDP=RA/(RS+RA)
       QF=RDP*QSAT(TF)+(1.-RDP)*QAF
       QAF=(1.-SIGF)*Q(TA)+SIGF*(Q(TA)*0.3+QF*0.6+QG*0.1)
       EF=-(RHOAF*CP/0.66)*(ESAT(TF)-E(TA))/(RA+RC)*60.
       IF(EF.LT.0.0)EF=0.0
       SHRW=FOLA*SOL
       XLNGW=EPF*ATERM
       TG4=EPF*EPSN/EP1*SIGMA*TEML**4
       TF4=(EP1+EPSN)/EP1*EPF*SIGMA*TF**4
       FENB=SIGF*(SHRW+XLNGW+TG4-TF4)-HSF-EF
       GO TO I950
C-----------------------------------------------------------
C         CALCULATE-ENERGY-BUDGET-FOR-GROUND
 1300   CONTINUE
       T1=STOR(8,1)*RR(1)*(STOR(1,3)-2.*STOR(1,2)+STOR(1,1))
     &  +STOR(1,2)
       TF4=SIGMA*TF**4
       TG4=SIGMA*TEML**4
       QG=WET*QSAT(TEML)+(1.-WET)*QAF
       RHOAG=0.001*0.348*PRESS/TAF
        XL1=597.3-0.566*(TAF+TEML-2.0*273.15)/2.
        SG=(1.-SIGF)*SOL
        RLU=(1.-SIGF)*(EPSN*TG4+(1.-EPSN)*ATERM)
     &   +SIGF*(EPSN*TG4+(1.-EPSN)*EPF*TF4)/EP1
        RLD=(1.-SIGF)*ATERM+SIGF*(EPF*TF4+(1.-EPF)*EPSN*TG4)/EP1
        HSG=RHOAG*CP*CHG*UAF*(TEML-TAF)*60.
        ELG=RHOAG*CHG*UAF*(QG-QAF)*60.
        FENB=SMALLA*SG-RLU+RLD-HSG-ELG*XL1+(T1-TEML)/SFRQ(1)*STOR(6,1)
        GO TO I1300
C----------------------------------------------------------
C       CALCULATE-RADIANCE-VALUES
 1410    CONTINUE
        REFRAD=((1.-SIGF)*(1-EPSN)+SIGF*(1-EPF))*DOWNIR*697.6
        FOLGB=EPF*5.67E-8*TF**4
        GRNDGB=EPSN*5.67E-8*TEML**4
        SURFGB=SIGF*FOLGB+(1.-SIGF)*GRNDGB
        EEF=SIGF*EPF+(1.-SIGF)*EPSN
        TEFF=(SURFGB/5.67E-8)**.25
        ISURFG=SURFGB+.5
        TEFFR=((SURFGB+REFRAD)/(5.67E-8))**.25
        IREFRA=REFRAD+0.5
        ISOL=SOL*697.6+0.5
        GO TO I1410
C----------------------------------------------------------
C
```

```
C        *********  VARIABLE   DEFINITIONS  **********
C
C-----------------------------------------------------------
C    ALPH(IX)   THERMAL DIFFUSIVITY OF LAYER IX IN CM**2/MIN
C
C    APRM       FACTE*TEMP**3 IN CAL/CM**2-MIN-C
C
C    ATERM      ENERGY CONTRIBUTED BY ATMOSPHERIC IR EMISSION
C               CAL CM**2-MIN
C
C    B          HEAT CONDUCTIVITY OF SURFACE CAL/CM**2-MIN-C
C
C    BBB(J,I)   Y INTERCEPT OF LINEAR EQUATION, USED
C               FOR TABLE INTERPOLATION.
C
C    BK         BOTTOM SURFACE GEOMETRIC SHAPE IN FRACTION(0.0-1.0)
C
C    BPRM       HEAT CONDUCTIVITY OF BOTTOM BOUNDARY LAYER
C
C    BTERM      ENERGY CONTRIBUTED BY INSOLATION AFTER ADJUSTMENT USING
C               SURFACE ABSORPTIVITY. IN CAL/CM**2-MIN
C
C    CLOUD      CLOUD COVER IN FRACTION OF 0.1-1.0
C
C    DAY        JULIAN DAY USED IN SOLVING INSOLATION
C
C    DECL       SOLAR DECLINATION ANGLE
C
C    DELT       TIME STEP IN HOURS
C
C    DIST       DEPTH IN CM OF INITIAL SOIL PROFILE AT WHICH
C               CORRESPONDING SOIL TEMPERATURE IN DEGREE C IS
C               INTERPOLATED.(TABLE 5)
C
C    DPRM       HEAT FLUX IN CAL/CM**2-MIN AT BOTTOM BOUDARY OR
C               TEMPERATURE IN RANKINS AT BOTTOM BOUNDARY.
C
C    DPRM0      TEMPERATURE OF BOTTOM MATERIAL IN
C               DEGREE CELSIUS.USED WHEN LFLUXY=0
C
C    DPRM1      HEAT FLUX OF BENEATH BOTTOM MATERIAL,
C               IN CAL/CM**2-MIN, USED WHEN LFLUXY NOT
C               EQUAL 0
C    DTERM      ENERGY LOSS DUE TO EVAPORATION
C
C    DUST       ATMOSPERIC DUST IN POUNDS/CUBIC CENTIMETERS
C               (LBS/CC)USED IN SOLVING INSOLATION.
C
C    ELF        LATITUDE IN RADIANS
C
C    EPSN       EMISSIVITY  OF SURFACE MATERIAL
C
C    FACTA      SIGMA*EPSN
C
C    FACTD      FACTD=SIGMA*BK*BEP*TR**4  USED IN BOTTOM BOUNDARY
C               CALCULATION WHEN THERE IS AIRSPACE BENEATH THE BOTTOM
```

```
C
C  FACTE      FACTE=SIGMA*BK*BEP  USED IN BOTTOM BOUNDARY CALCULATION
C             WHEN THERE IS AIRSPACE BENEATH THE BOTTOM
C
C  FACTH      USED IN SOLVING CONVECTION TERM (HTERM)
C             (1000.0/PRESS)**0.286
C
C  FK(IX)     HEAT CONDUCTIVITY OF LAYER IX IN CAL/MIN-CM-K
C
C  FMM(J,I)   SLOPE OF LINEAR EQUATION,USED FOR TABLE INTERPOLATION
C
C  HEADER     72 CHARCTER INPUT VARIABLE USED TO PRINT
C             COMMENTS ON OUTPUT.
C
C  HTERM      ENERGY LOSS OF GAIN DUE TO CONVECTION CAL/CM**2-MIN
C
C  IEOF       SET FROM 0 TO 1 WHEN AN EOF IS ENCOUNTERED. USED TO
C             TERMINATE PROGRAM
C
C  IMATL      BACKWARD COUNTER OF LAYERS. STARTING WITH THE NUMBER
C             OF LAYERS.
C
C  INTR(IX)   BEGINNING SUB-LAYER DEPTH NUMBER FOR LAYER NUMBER IX
C
C  IPRNT      BACKWARD COUNTER SET=NPRNT. WHEN EQUAL TO 1 OUTPUT IS
C             PRINTED.
C
C  ITIME      BACKWARD COUNTER INITIALIZE AS TOTAL TIME STEPS IN HOUR
C
C  IX         LAYER NUMBER STARTING WITH TOP LAYER
C
C  IY         SUB-LAYER DEPTH NUMBER
C
C  JMAX       THE TOTAL NUMBER OF SUB-LAYERS
C
C  LAT        LATITUDE USED IN SOLVING INSOLATION
C!
C  LFLUXY     INPUT BOTTOM BOUNDARY DATA CONTROL SWITCH. IF=0,THERE
C             IS NO HEAT FLUX THROUGH BOTTOM OF MATERIAL,IF NEGATIVE
C             THERE IS NO AIR SPACE BENEATH BOTTOM MATERIAL,IF POSIT-
C             IVE THERE IS AIR SPACE BENEATH BOTTOM MATERIAL.
C
C  LN         DUMMY VARIABLE TO READ LINE NUMBER FROM INPUT FILE
C
C  M          SECANT OF SOLAR ZENITH ANGLE IN RADIANS
C
C             INTERPOLATION MODULE.
C
C  NCLOUD     CLOUD TYPE INDEX NUMBER (1-9) USED IN
C             SOLVING INSOLATION,INFRARED EMISSION.
C
C  NOMATL     NUMBER OF MATERIAL LAYERS USED IN SOLVING HEAT FLOW
C
C  NPRNT      NUMBER OF TIMES OUTPUT TIME PRINT FREQUENCY IS DIVISIBL
C             BY TIME STEPS. USED TO DETERMINED WHEN TO PRINT OUTPUT.
C
```

```
C   NTABL     TABLE NUMBER
C
C   NX(IX)    NUMBER OF SUBLAYER OF EACH LAYER,NX(IX)=THK(IX)/SFRQ(IX
C
C   PRESS     ATMOSPHERIC PRESSURE IN MILLIBAR(MB)
C           USED IN SOLVING INSOLATION
C
C   REP       EMISSIVITY BENEATH AIRSPACE
C
C   RH        RELATIVE HUMIDITY
C
C   RHOC(IX)   FK(IX)/ALPH(IX) IN CAL/CM**2-K
C
C   RI        RICHARDSON INDEX NUMBER USE IN SOLVING CONVECTION
C           ENERGY LOSS.
C
C   RK        SURFACE BENEATH AIRSPACE GE0METRIC
C           SHAPE IN FRACTION (0.0 - 1.0)
C
C   RR(IX)    RR(IX)=DELT/SFREQ**2.(PART OF HEAT FLOW EQUATION)
C
C
C   SAZ       SOLAR AZIMUTH IN RADIANS. SAZ=ATAN(-COS(DECL)*SIN(TIMER
C           (COS(ELF*SIN(DECL)-SIN(ELF)*COS(TIMER)))
C
C   SFRQ(IX)   VERTICAL GRID SPACING IN CM IN EACH LAYER IX IN CM**2/M
C
C   SICF      INSOLATION ADJUSTMENT DUE TO ZENITH ANGLE,SURFACE SLOPE
C           AND SURFACE ASPECT ANGLE. SICF=COS(Z)*COS(SLOPE)+SIN(Z)
C           SIN(SLOPE)*COS(SAZ-SURFAC)
C
C      SIGMA          STEFAN-BOLTZMANN CONSTANT
C                     5.67E-8 W/(m**2-K**4), OR
C                     8.12E-11 cal/(min-cm**2-K**4)
C
C   SLOPE     SURFACE SLOPE IN DEGREES WITH HORIZONTAL=0 DEGREE,
C           USED IN SOLVING INSOLATION
C
C   SMALLA    ABSORBTIVITY OF SURFACE MATERIAL
C
C   SPEED     WIND SPEED IN CM/SEC
C
C   STOR(1,IY) ESTIMATE SUB-LAYER TEMPERATURE IN DEGREE RANKINE
C
C   STOR(2,IY) FK;HEAT CONDUCTIVITY OF SUB-LAYER IY IN CAL/MIN-CM-K
C
C   STOR(3,IY) RHOC,FK/ALPH IN CAL/CM**2-K
C
C   STOR(4,IY) CONSTANT DIMENSIONLESS.
C
C   STOR(5,IY) INITIAL SOIL TEMPERATURE IN DEGREE RANKINS
C           OF INITIAL SOIL PROFILE
C
C   STOR(6,IY) SAME AS STOR(2,IY)
C
C   STOR(7,IY) SAME AS STOR(3,IY)
```

```
C
C  SUN       CALCULATED INSOLATION VALUE.
C
C  SURFAC   SURFACE AZIMUTH IN DEGREE WITH SOUTH =0 DEGREE,USED
C           IN SOLVING INSOLATION
C
C  T         SAME AS TIME
C
C  TA        AIR TEMPERATURE IN DEGREE RANKINE
C
C  TAC       AIR TEMPERATURE IN DEGREE CELSIUS
C
C  TAK       AIR TEMPERATURE IN DEGREE KELVIN
C
C  TB        THERMAL CONDUCTIVITY OF BOTTOM MATERIAL
C           CAL/CM**2-DEG C-MIN
C
C  TFRQ      TIME STEP IN MINUTES USED IN SOLVING HEAT FLOW
C
C  THK(IX)   LAYER THICKNESS IN CM OF LAYER IX
C
C  TIME      TIME IN HOURS IN WHICH MATERIAL TEMPERATURES
C           ARE ESTIMATED
C
C  TIMER     SUN'S HOUR ANGLE IN RADIANS
C
C  TOTTIM    TOTAL NUMBER OF 24 HOUR REPETITIONS USED IN SOLVING
C           HEAT FLOW
C
C  TPRNT     OUTPUT TIME PRINT FREQUENCY IN MINUTES
C
C  TR        TEMPERATURE OF AIRSPACE BENEATH BOTTOM MATERIAL.
C
C  TSK       MATERIAL SUB-LAYER TEMPERATURE IN DEGREES KELVIN
C
C  TYME      TIME IN HOURS USE INSOLATION CALCULATION
C
C     VMSURF         MOISTURE CONTENT OF SURFACE MATERIAL (DECIMAL)
C
C  WATER     THE AMOUNT OF PRECIPIPAL WATER IN MILLIMETERS
C           (MM) USED IN SOLVING INSOLATION.
C
C  WET       DEGREE OF SATURATION OF SURFACE MATERIAL (DECIMAL)
C
C  XXX(J)    DEPTH (IN CENTIMETERS) FOR
C           INITIAL TEMPERATURE PROFILE
C
C  YYY(J)     INITIAL TEMPERATURE PROFILE VALUES, DEG C
C
C  Z         SOLAR ZENITH ANGLE. Z=SIN(DECL)*SIN(ELF)+COS(DECL)*
C           COS(ELF)*COS(TIMER)
C
C  ZASH      HEIGHT OF WIND SPEED INDICATOR (CM)
C
C  ZZA       SURFACE TEMPERATURE OF MATERIAL IN DEGREE RANKINE
C
```

```
C    ZZB        BOTTOM LAYER TEMPERATURE OF MATERIAL IN DEGREE RANKINE
C
C         **** NEW PARAMETERS UTILIZED FOR MULTIPLE-DAY SIMULATIONS: ****
C
C         ATEMP             AIR TEMPERATURE , DEG C
C
C         BPRESS                    BAROMETRIC PRESSURE , MILLIBARS
C
C         CLDCOV                    CLOUD COVER , PERCENT
C
C         CLDTYPE                   CLOUD INDEX
C
C         DEGSAT                    DEGREE OF SATURATION , DECIMAL
C
C         DEPTH             A VECTOR OF Z VALUES FOR ALL OF THE NODES
C
C         DT                TIME OF DAY IN DECIMAL HOURS
C
C         IPPRNT                    NUMBER OF TIME STEPS BETWEEN HOURLY
TEMPERATURE
C                           PROFILE OUTPUT ON SELECTED DAY: NSNGLDAY
C
C         IPRNT             NUMBER OF TIME STEPS BETWEEN SIMULATION OUTPUTS
C                           = TPRNT/TFRQ
C
C         JD                JULIAN DAY
C
C         LNUM              A VECTOR OF LAYER NUMBERS FOR ALL OF THE NODES
C
C         NDTS              INPUT DATA TIME SUBSCRIPT; VALUE OF 24 MEANS THAT THE
C                           CURRENT CALCULATION FALLS BETWEEN THE 24TH AND 25TH
C                           DATA STRINGS; USED AS SUBSCRIPT FOR
C                           DATA INTERPOLATIONS
C
C         NFIRST                    THE INPUT DATA LINE NUMBER CONTAINING THE FIRST
C                           LINE OF DATA FOR THE SIMULATION.  EQUALS 1 IF THE
C                           ENTIRE INPUT DATA FILE IS GOING TO BE USED.
C
C         NLDATA                    NUMBER OF LINES OF WEATHER DATA
C
C         NPPRNT                    ACCUMULATING NUMBER OF TIME STEPS BETWEEN
C                           PROFILE OUTPUTS
C
C         NPRNT             ACCUMULATING NUMBER OF TIME STEPS BETWEEN
C                           SIMULATION OUTPUTS
C                           RESET TO 1 AFTER EACH SIMULATION OUTPUT
C
C         NSINGLE                   A FLAG FOR PERFORMING SINGLE-DAY SIMULATIONS
C                           =1 (OR NOT 0) IF DOING A SINGLE-DAY SIMULATION
C                           =0 FOR A MULTIDAY SIMULATION
C
C         NSNGLDAY          THE JULIAN NUMBER FOR THE DAY CHOSEN FOR A
C                           SINGLE-DAY SIMULATION AND/OR 1-HR TEMPERATURE
C                           PROFILES
C
C         RELHUM                    RELATIVE HUMIDITY , PERCENT
```

```
C
C         RTEMP1                 RADIOMETRIC TEMPERATURE OF SOME SURFACE
OBJECT , DEG C
C
C         RTEMP2                 RADIOMETRIC TEMPERATURE OF 2ND SURFACE OBJECT
,         DEG C
C
C         SOLAR         SOLAR LOADING , W/M^2
C
C         STEMP1                 SOIL TEMPERATURE AT SHALLOWEST DEPTH , DEG C
C
C         STEMP2                 SOIL TEMPERATURE AT NEXT SHALLOWEST DEPTH , DEG
C
C
C         STOR(8,I)     VECTOR OF DIFFUSIVITY VALUES FOR EACH NODE
C
C         VMOIS1                 VOLUMETRIC MOISTURE CONTENT AT SHALLOWEST
DEPTH , %
C
C         VMOIS2                 VOLUMETRIC MOISTURE CONTENT AT NEXT
SHALLOWEST DEPTH , %
C
C         WINDSP                 WIND SPEED , M/S
C
      END
```

Appendix B
Example Input File

```
midwestern.test.site.2004,11656,,,,,,,,,,,,,,
150,0,17.29,95.5,973,0,0.06,1,0,0.32,12.8,16,20.55,23,16.59,19.01
150,100,17.44,94,973,0,0.183,1,0,0.32,12.8,15.8,19.86,22.28,16.12,18.28
150,200,16.08,96.3,972,0,0.016,1,0,0.318,12.7,15.7,19.2,21.66,14.84,17.71
150,300,15.77,96.9,972,0,0.051,1,0,0.318,12.7,15.6,18.72,21.09,14.54,17.35
150,400,14.89,98.6,972,0,0.002,1,0,0.318,12.7,15.5,18.19,20.58,14.04,16.89
150,500,14.1,98.4,972,0,0,1,0,0.315,12.6,15.4,17.7,20.1,13.52,16.49
150,600,13.94,99.2,972,9.27,0,1,0,0.315,12.6,15.3,17.29,19.65,13.42,16.22
150,700,16.67,99.4,972,88,0.154,1,0,0.318,12.7,15.3,18.86,19.75,18.73,19.07
150,800,20.26,88.8,972,244.9,0.584,1,0,0.32,12.8,15.3,21.64,20.72,25.66,20.86
150,900,21.25,87.9,972,295.8,1.256,1,0,0.323,12.9,15.4,24.4,22.53,28.69,21.91
150,1000,22.49,80.8,972,367.3,1.679,1,0,0.323,12.9,15.5,25.57,23.51,33.46,23.92
150,1100,24.27,73.2,971,931,1.654,1,0,0.323,12.9,15.6,28.24,24.46,42.1,27.51
150,1200,23.79,70.8,971,489.2,1.854,1,0,0.32,12.8,15.7,29.27,25.9,37.83,25.31
………
………
………
………
………
………
214,1619,32.13,48.09,-6999,710,0.923,1,0,0.19,7.59,14.35,19.45,22.39,35.93,32.91
214,1620,32.17,44.25,-6999,702,0.882,1,0,0.19,7.58,14.35,19.44,22.39,35.86,32.91
214,1621,32.18,48.19,-6999,694.3,0.48,1,0,0.19,7.58,14.35,19.44,22.39,35.83,32.92
214,1622,32.25,46.1,-6999,686.7,0.474,1,0,0.189,7.57,14.35,19.44,22.39,35.79,32.91
214,1623,32.35,46.47,-6999,682,0.776,1,0,0.189,7.57,14.34,19.44,22.39,35.77,32.9
214,1624,32.45,45.86,-6999,674.2,0.784,1,0,0.189,7.56,14.34,19.43,22.38,35.73,32.88
214,1625,32.3,48.61,-6999,669.9,0.982,1,0,0.189,7.56,14.33,19.43,22.38,35.7,32.9
End,,,,,,,,,,,,,,,
1,210,0,0,0,0,,,,,,,,,,
0,0,0,0,0,0,,,,,,,,,,
-0.0052,1,0,0,0,0,,,,,,,,,,
0.007,0.4,0,0,0,0,,,,,,,,,,
0,0,0,0,0,0,,,,,,,,,,
0,0,0,0,0,0,,,,,,,,,,
4,0,0,0,0,0,
2,0.2,0,0.4,0.0283,0,
8,0.5,0,0.4,0.0283,0,
40,1,0,0.4,0.0283,0,
200,5,0,0.4,0.0283,0,
0,0,0,0,0,0,
```

```
0,0,0,0,0,0,284020000
0,0,0,0,0,0,
25,0,0,0,0,0,
50,10,0,0,0,0,
1.5,4.5,0,0,0,0,
0.04,30,8,300,0,0,
```

Appendix C
Graphical User Interface Macro

```
Public ndlines As Integer
Public numlines As Integer
Public nsingle As Integer
Public ndos As Integer
Public nbbflag As Integer
Public nsolarflag As Integer
Public julian1 As Integer
Public julian2 As Integer
Public filedesc As String
Public infilename As String
Public status As String

Private Sub CheckBox1_Click()
If CheckBox1.Value = True Then nbbflag = -1
End Sub

Private Sub CheckBox2_Click()
If CheckBox2.Value = True Then nbbflag = 0
End Sub

Private Sub CheckBox3_Click()
If CheckBox3.Value = True Then nbbflag = 1
End Sub

Private Sub CheckBox4_Click()
If CheckBox4.Value = True Then nsolarflag = 0
End Sub

Private Sub CheckBox5_Click()
If CheckBox5.Value = True Then nsolarflag = 1
End Sub



Private Sub fixeddosbox_Click()
ndos = 0
If fixeddosbox.Value = True Then ndos = 1
End Sub

Private Sub lastjulianbox_Change()
```

```
End Sub
Private Sub multidaybox_Click()
nsingle = 1
If multidaybox.Value = True Then nsingle = 0
End Sub
Private Sub singledaybox_Click()
nsingle = 0
If singledaybox.Value = True Then nsingle = 1
End Sub
Private Sub updatebutton_Click()
dt1box.Text = space1box.Text ^ 2 / (2 * (diffslp1box.Text * 40 + diffint1box.Text))
If nlayerbox.Text > 1 Then dt2box.Text = space2box.Text ^ 2 / (2 * (diffslp2box.Text * 40 +
diffint2box.Text))
If nlayerbox.Text > 2 Then dt3box.Text = space3box.Text ^ 2 / (2 * (diffslp3box.Text * 40 +
diffint3box.Text))
If nlayerbox.Text > 3 Then dt4box.Text = space4box.Text ^ 2 / (2 * (diffslp4box.Text * 40 +
diffint4box.Text))
If nlayerbox.Text > 4 Then dt5box.Text = space5box.Text ^ 2 / (2 * (diffslp5box.Text * 40 +
diffint5box.Text))
If nlayerbox.Text > 5 Then dt6box.Text = space6box.Text ^ 2 / (2 * (diffslp6box.Text * 40 +
diffint6box.Text))
totalthickbox.Text = Val(thick1box.Text) + Val(thick2box.Text) + Val(thick3box.Text) +
Val(thick4box.Text) + Val(thick5box.Text) + Val(thick6box.Text)
End Sub
Private Sub UserForm_Click()
End Sub
Private Sub variabledosbox_Click()
ndos = 1
If variabledosbox.Value = True Then ndos = 0
End Sub
Private Sub inputfilebutton_Click()
'
' open a window for selecting the input file
'
Dim irow As Integer
Dim th1, th2, th3, th4, th5, th6
infile = Application.GetOpenFilename(filefilter:="csv files(*.csv),*.csv", Title:="Input Files")
Workbooks.OpenText Filename:=infile, DataType:=xlDelimited, Comma:=True
slashnum = InStrRev(infile, "\") + 1
infilename = Mid(infile, slashnum)
ndlines = Range("a1:aa20000").Find(what:="End").Row - 2
numlines = ndlines + 2
filedesc = Cells(1, 1)
julian1 = Cells(2, 1)
```

```
julian2 = Cells(ndlines + 1, 1)
datalinesbox.Text = ndlines
Cells(1, 2) = ndlines
firstjulianbox.Text = julian1
lastjulianbox.Text = julian2
filenamebox.Text = infile
filedescbox.Text = filedesc
'
' populate the userform with zeros when cells are blank
'
For i = 1 To 18
For j = 1 To 6
If IsEmpty(Cells(numlines + i, j)) Then Cells(numlines + i, j) = 0
Next j
Next i
'
If Cells(numlines + 1, 1) = 1 Then singledaybox.Value = True Else singledaybox.Value = False
If Cells(numlines + 1, 1) = 0 Then multidaybox.Value = True Else multidaybox.Value = False
nsngldaybox.Text = Cells(numlines + 1, 2)
If Cells(numlines + 2, 1) = 0 Then CheckBox4.Value = True Else CheckBox4.Value = False
If Cells(numlines + 2, 1) = 1 Then CheckBox5.Value = True Else CheckBox5.Value = False
surfslopebox.Text = Cells(numlines + 2, 2)
surfazbox.Text = Cells(numlines + 2, 3)
sitelatbox.Text = Cells(numlines + 2, 4)
emissslopebox.Text = Cells(numlines + 3, 1)
emissintercbox.Text = Cells(numlines + 3, 2)
absslopebox.Text = Cells(numlines + 4, 1)
absintercbox.Text = Cells(numlines + 4, 2)
If Cells(numlines + 5, 1) = 0 Then variabledosbox.Value = True Else variabledosbox.Value =
False
If Cells(numlines + 5, 1) = 1 Then fixeddosbox.Value = True Else fixeddosbox.Value = False
surfsatbox.Text = Cells(numlines + 5, 2)
folcoverbox.Text = Cells(numlines + 6, 1)
stomatresisbox.Text = Cells(numlines + 6, 2)
folemissbox.Text = Cells(numlines + 6, 3)
folabsbox.Text = Cells(numlines + 6, 4)
folheightbox.Text = Cells(numlines + 6, 5)
nlayerbox.Text = Cells(numlines + 7, 1)
thick1box.Text = Cells(numlines + 8, 1)
th1 = Cells(numlines + 8, 1)
space1box.Text = Cells(numlines + 8, 2)
diffslp1box.Text = Cells(numlines + 8, 3)
diffint1box.Text = Cells(numlines + 8, 4)
condslp1box.Text = Cells(numlines + 8, 5)
condint1box.Text = Cells(numlines + 8, 6)
thick2box.Text = Cells(numlines + 9, 1)
th2 = Cells(numlines + 9, 1)
space2box.Text = Cells(numlines + 9, 2)
diffslp2box.Text = Cells(numlines + 9, 3)
diffint2box.Text = Cells(numlines + 9, 4)
condslp2box.Text = Cells(numlines + 9, 5)
condint2box.Text = Cells(numlines + 9, 6)
thick3box.Text = Cells(numlines + 10, 1)
th3 = Cells(numlines + 10, 1)
space3box.Text = Cells(numlines + 10, 2)
diffslp3box.Text = Cells(numlines + 10, 3)
```

```
diffint3box.Text = Cells(numlines + 10, 4)
condslp3box.Text = Cells(numlines + 10, 5)
condint3box.Text = Cells(numlines + 10, 6)
thick4box.Text = Cells(numlines + 11, 1)
th4 = Cells(numlines + 11, 1)
space4box.Text = Cells(numlines + 11, 2)
diffslp4box.Text = Cells(numlines + 11, 3)
diffint4box.Text = Cells(numlines + 11, 4)
condslp4box.Text = Cells(numlines + 11, 5)
condint4box.Text = Cells(numlines + 11, 6)
thick5box.Text = Cells(numlines + 12, 1)
th5 = Cells(numlines + 12, 1)
space5box.Text = Cells(numlines + 12, 2)
diffslp5box.Text = Cells(numlines + 12, 3)
diffint5box.Text = Cells(numlines + 12, 4)
condslp5box.Text = Cells(numlines + 12, 5)
condint5box.Text = Cells(numlines + 12, 6)
thick6box.Text = Cells(numlines + 13, 1)
th6 = Cells(numlines + 13, 1)
space6box.Text = Cells(numlines + 13, 2)
diffslp6box.Text = Cells(numlines + 13, 3)
diffint6box.Text = Cells(numlines + 13, 4)
condslp6box.Text = Cells(numlines + 13, 5)
condint6box.Text = Cells(numlines + 13, 6)
totalthickbox.Text = th1 + th2 + th3 + th4 + th5 + th6

If Cells(numlines + 14, 1) = -1 Then
CheckBox1.Value = True
bbfluxbox.Text = Cells(numlines + 15, 1)
Else: CheckBox1.Value = False
End If
If Cells(numlines + 14, 1) = 0 Then
CheckBox2.Value = True
bbtempbox.Text = Cells(numlines + 15, 1)
Else: CheckBox2.Value = False
End If
If Cells(numlines + 14, 1) = 1 Then
CheckBox3.Value = True
rbbfluxbox.Text = Cells(numlines + 15, 1)
rbbemiss1box.Text = Cells(numlines + 15, 2)
rbbsf1box.Text = Cells(numlines + 15, 3)
rbbemiss2box.Text = Cells(numlines + 15, 4)
rbbsf2box.Text = Cells(numlines + 15, 5)
rbbtempbox.Text = Cells(numlines + 15, 6)
Else: CheckBox3.Value = False
End If
depthfmbox.Text = Cells(numlines + 16, 1)
fixedmoistbox.Text = Cells(numlines + 16, 2)
col11depthbox.Text = Cells(numlines + 17, 1)
col12depthbox.Text = Cells(numlines + 17, 2)
timeincbox.Text = Cells(numlines + 18, 1)
outputintbox.Text = Cells(numlines + 18, 2)
iterationsbox.Text = Cells(numlines + 18, 3)
windheightbox.Text = Cells(numlines + 18, 4)
End Sub
```

```
Private Sub exectstmbutton_Click()
'
' load all of the chosenrun parameters into the end of
' the input file, save that file, and execute the fortran code
'
Cells(numlines + 1, 1) = nsingle
Cells(numlines + 1, 2) = nsngldaybox.Text
Cells(numlines + 2, 1) = nsolarflag
Cells(numlines + 2, 2) = surfslopebox.Text
Cells(numlines + 2, 3) = surfazbox.Text
Cells(numlines + 2, 4) = sitelatbox.Text
Cells(numlines + 3, 1) = emissslopebox.Text
Cells(numlines + 3, 2) = emissintercbox.Text
Cells(numlines + 4, 1) = absslopebox.Text
Cells(numlines + 4, 2) = absintercbox.Text
Cells(numlines + 5, 1) = ndos
Cells(numlines + 5, 2) = surfsatbox.Text
Cells(numlines + 6, 1) = folcoverbox.Text
Cells(numlines + 6, 2) = stomatresisbox.Text
Cells(numlines + 6, 3) = folemissbox.Text
Cells(numlines + 6, 4) = folabsbox.Text
Cells(numlines + 6, 5) = folheightbox.Text
Cells(numlines + 7, 1) = nlayerbox.Text
Cells(numlines + 8, 1) = thick1box.Text
Cells(numlines + 8, 2) = space1box.Text
Cells(numlines + 8, 3) = diffslp1box.Text
Cells(numlines + 8, 4) = diffint1box.Text
Cells(numlines + 8, 5) = condslp1box.Text
Cells(numlines + 8, 6) = condint1box.Text
Cells(numlines + 9, 1) = thick2box.Text
Cells(numlines + 9, 2) = space2box.Text
Cells(numlines + 9, 3) = diffslp2box.Text
Cells(numlines + 9, 4) = diffint2box.Text
Cells(numlines + 9, 5) = condslp2box.Text
Cells(numlines + 9, 6) = condint2box.Text
Cells(numlines + 10, 1) = thick3box.Text
Cells(numlines + 10, 2) = space3box.Text
Cells(numlines + 10, 3) = diffslp3box.Text
Cells(numlines + 10, 4) = diffint3box.Text
Cells(numlines + 10, 5) = condslp3box.Text
Cells(numlines + 10, 6) = condint3box.Text
Cells(numlines + 11, 1) = thick4box.Text
Cells(numlines + 11, 2) = space4box.Text
Cells(numlines + 11, 3) = diffslp4box.Text
Cells(numlines + 11, 4) = diffint4box.Text
Cells(numlines + 11, 5) = condslp4box.Text
Cells(numlines + 11, 6) = condint4box.Text
Cells(numlines + 12, 1) = thick5box.Text
Cells(numlines + 12, 2) = space5box.Text
Cells(numlines + 12, 3) = diffslp5box.Text
Cells(numlines + 12, 4) = diffint5box.Text
Cells(numlines + 12, 5) = condslp5box.Text
Cells(numlines + 12, 6) = condint5box.Text
Cells(numlines + 13, 1) = thick6box.Text
Cells(numlines + 13, 2) = space6box.Text
Cells(numlines + 13, 3) = diffslp6box.Text
```

```
Cells(numlines + 13, 4) = diffint6box.Text
Cells(numlines + 13, 5) = condslp6box.Text
Cells(numlines + 13, 6) = condint6box.Text
Cells(numlines + 13, 7) = totalthickbox.Text
Cells(numlines + 14, 1) = nbbflag
Cells(numlines + 15, 1) = rbbfluxbox.Text
Cells(numlines + 15, 2) = rbbemiss1box.Text
Cells(numlines + 15, 3) = rbbsf1box.Text
Cells(numlines + 15, 4) = rbbemiss2box.Text
Cells(numlines + 15, 5) = rbbsf2box.Text
Cells(numlines + 15, 6) = rbbtempbox.Text
If nbbflag = -1 Then Cells(numlines + 15, 1) = bbfluxbox.Text
If nbbflag = 0 Then Cells(numlines + 15, 1) = bbtempbox.Text
Cells(numlines + 16, 1) = depthfmbox.Text
Cells(numlines + 16, 2) = fixedmoistbox.Text
Cells(numlines + 17, 1) = col11depthbox.Text
Cells(numlines + 17, 2) = col12depthbox.Text
Cells(numlines + 18, 1) = timeincbox.Text
Cells(numlines + 18, 2) = outputintbox.Text
Cells(numlines + 18, 3) = iterationsbox.Text
Cells(numlines + 18, 4) = windheightbox.Text
'
'  save a new input file for TSTM as "fort.2"
'
ActiveWorkbook.SaveCopyAs Filename:="fort.2"
'
'   execute TSTM
'
Shell ("c:\tstm_files\tstmforgui.exe")
'
'  wait for the fortran code to finish executing before displaying results
'
 msg = MsgBox("Wait for TSTM to finish executing!", vbOKOnly)
'
'   hide the TSTM userform
'
    Executing_TSTM.Hide
'
'   display output
'
'    begin by importing the results of TSTM simulation
'    found in "fort.4" into this Excel file
'
    Windows("TSTM simulation results template.xls").Activate
    Sheets("simulation output data").Select
    Range(Cells(1, 1), Cells(3400, 18)).Clear
    Workbooks.OpenText Filename:= _
        "fort.4", Origin:= _
        xlWindows, StartRow:=1, DataType:=xlDelimited, TextQualifier:= _
        xlDoubleQuote, ConsecutiveDelimiter:=True, Tab:=True, Semicolon:=False, _
        Comma:=False, Space:=True, Other:=False, FieldInfo:=Array(Array(1, 1), _
        Array(2, 1))
'    transfer the "fort.4" sheet to the output data sheet of this file
    Range("1:3400").Select
    Selection.Copy
    Windows("TSTM simulation results template.xls").Activate
```

```
    Sheets("simulation output data").Select
    Range("A1").Select
    ActiveSheet.Paste
'   copy the profiles data to another data sheet
'
'   determine which lines to cut and paste by counting
'   output lines until reaching "NORMAL TERMINATION"
'
    irow = Range("a1:aa3400").Find(what:="NORMAL").Row

    Range(Cells(irow + 1, 2), Cells(3400, 28)).Select
    Selection.Copy
    Sheets("profiles data").Select
    Range("a2").Select
    ActiveSheet.Paste
    Sheets("simulation output data").Select
    Range(Cells(irow + 1, 1), Cells(3400, 28)).Clear
    Range("k50").Select

    copycoln = InputBox("Which column contains the measured surface temperature?", vbOKOnly)
    colname = copycoln & ":" & copycoln
    Columns(colname).Select
    Selection.Copy
    Columns("r:r").Select
    ActiveSheet.Paste
    Range("k50").Select
    copycoln = InputBox("Which column contains the difference between measured and simulated
temperatures?", vbOKOnly)
    colname = copycoln & ":" & copycoln
    Columns(colname).Select
    Selection.Copy
    Columns("s:s").Select
    ActiveSheet.Paste
'
'  calculate the average and population standard deviation of column "s"
'
    Range("t59").Select
    ActiveCell.FormulaR1C1 = "=average(r[0]c[-1]:r[3400]c[-1])"
    Selection.NumberFormat = "0.000"
    Range("t60").Select
    ActiveCell.FormulaR1C1 = "=stdevp(r[-1]c[-1]:r[3399]c[-1])"
    Selection.NumberFormat = "0.000"
'   print the selected day temperature prediction chart
    Sheets("temperature chart").Select
    nsd = nsngldaybox.Text
        With ActiveChart.Axes(xlCategory)
        .MinimumScale = nsd
        .MaximumScale = nsd + 1
        .MinorUnit = 0.04166666
        .MajorUnit = 1
        .Crosses = xlCustom
        .CrossesAt = 0
        .ReversePlotOrder = False
        .ScaleType = xlLinear
        .DisplayUnit = xlNone
        End With
```

```
    Application.CutCopyMode = False
    ActiveWindow.SelectedSheets.PrintOut Copies:=1, Collate:=True
'   print the temperature prediction charts for all days
    Sheets("temperature chart (2)").Select
    ndaybegin = firstjulianbox.Text
    ndaylast = lastjulianbox.Text
    ndayhalf = ndaybegin + (ndaylast - ndaybegin) / 2 + 1
    nofiveday = Int((ndayhalf - ndaybegin) / 5) + 1
    ndayhalf = ndaybegin + 5 * nofiveday
    ndayend = ndaybegin + 10 * nofiveday
    With ActiveChart.Axes(xlCategory)
       .MinimumScale = ndaybegin
       .MaximumScale = ndayhalf
       .MinorUnit = 1
       .MajorUnit = 5
       .Crosses = xlCustom
       .CrossesAt = 0
       .ReversePlotOrder = False
       .ScaleType = xlLinear
       .DisplayUnit = xlNone
       End With
'   Application.CutCopyMode = False
'   ActiveWindow.SelectedSheets.PrintOut Copies:=1, Collate:=True
    Sheets("temperature chart (3)").Select
    With ActiveChart.Axes(xlCategory)
       .MinimumScale = ndayhalf
       .MaximumScale = ndayend
       .MinorUnit = 1
       .MajorUnit = 5
       .Crosses = xlCustom
       .CrossesAt = 0
       .ReversePlotOrder = False
       .ScaleType = xlLinear
       .DisplayUnit = xlNone
       End With
'   print the temperature profiles chart for the selected day
    Sheets("profiles chart").Select
    Application.CutCopyMode = False
    ActiveWindow.SelectedSheets.PrintOut Copies:=1, Collate:=True
    Sheets("pred-meas").Select
      With ActiveChart.Axes(xlCategory)
       .MinimumScale = nsd
       .MaximumScale = nsd + 1
       .MinorUnit = 0.04166666
       .MajorUnit = 1
       .Crosses = xlCustom
       .CrossesAt = 0
       .ReversePlotOrder = False
       .ScaleType = xlLinear
       .DisplayUnit = xlNone
       End With
'   print the predicted-measured temperature charts for all days
    Sheets("pred-meas (2)").Select
    With ActiveChart.Axes(xlCategory)
       .MinimumScale = ndaybegin
       .MaximumScale = ndayhalf
```

```
        .MinorUnit = 1
        .MajorUnit = 5
        .Crosses = xlCustom
        .CrossesAt = 0
        .ReversePlotOrder = False
        .ScaleType = xlLinear
        .DisplayUnit = xlNone
        End With
    Sheets("pred-meas (3)").Select
    With ActiveChart.Axes(xlCategory)
        .MinimumScale = ndayhalf
        .MaximumScale = ndayend
        .MinorUnit = 1
        .MajorUnit = 5
        .Crosses = xlCustom
        .CrossesAt = 0
        .ReversePlotOrder = False
        .ScaleType = xlLinear
        .DisplayUnit = xlNone
        End With
'    Application.CutCopyMode = False
'    ActiveWindow.SelectedSheets.PrintOut Copies:=1, Collate:=True
'    print the selected day flux prediction chart
    Sheets("flux chart").Select
   With ActiveChart.Axes(xlCategory)
        .MinimumScale = nsd
        .MaximumScale = nsd + 1
        .MinorUnit = 0.04166666
        .MajorUnit = 1
        .Crosses = xlCustom
        .CrossesAt = 0
        .ReversePlotOrder = False
        .ScaleType = xlLinear
        .DisplayUnit = xlNone
        End With
'    Application.CutCopyMode = False
'    ActiveWindow.SelectedSheets.PrintOut Copies:=1, Collate:=True
'   print the flux charts for all days
    Sheets("flux chart (2)").Select
    With ActiveChart.Axes(xlCategory)
        .MinimumScale = ndaybegin
        .MaximumScale = ndayhalf
        .MinorUnit = 1
        .MajorUnit = 5
        .Crosses = xlCustom
        .CrossesAt = 0
        .ReversePlotOrder = False
        .ScaleType = xlLinear
        .DisplayUnit = xlNone
        End With
    Sheets("flux chart (3)").Select
    With ActiveChart.Axes(xlCategory)
        .MinimumScale = ndayhalf
        .MaximumScale = ndayend
        .MinorUnit = 1
        .MajorUnit = 5
```

```
        .Crosses = xlCustom
        .CrossesAt = 0
        .ReversePlotOrder = False
        .ScaleType = xlLinear
        .DisplayUnit = xlNone
        End With
'   return to the 1st day temperature chart
    Sheets("temperature chart").Select
'
'    pause to look at single-day temperature results before continuing
'
'
'   choose to save the excel output file and the input file under new names
'
msg = MsgBox("Do you want to save the output file under a new name?", vbYesNo)
    If msg = 6 Then
    filesavename = Application.GetSaveAsFilename(filefilter:="Excel files(*.xls),*.xls", Title:="Save
Output File Under a New Name")
    If filesavename <> False Then ActiveWorkbook.SaveCopyAs Filename:=filesavename
    Else
    End If
    Windows(infilename).Activate
msg = MsgBox("Do you want to save the .csv input file under a new name?", vbYesNo)
    If msg = 6 Then
    filesavename = Application.GetSaveAsFilename(filefilter:="csv files(*.csv),*.csv", Title:="Save
Modified Input File (csv format)")
    If filesavename <> False Then ActiveWorkbook.SaveCopyAs Filename:=filesavename
    Else
    End If
    Windows("TSTM simulation results template.xls").Activate
'
'
msg = MsgBox("Do you want to do another simulation?", vbYesNo)
    If msg = 6 Then
        Windows("fort.4").Activate    '    close the file called "fort.4"
        ActiveWorkbook.Close
        Windows(infilename).Activate
        Executing_TSTM.Show    '   reveal the TSTM Graphical User Interface
        Else
    End If
'
'   close all files except the output template file
'
        Windows(infilename).Activate
        ActiveWorkbook.Close
        Windows("fort.4").Activate
        ActiveWorkbook.Close
End Sub
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
February 2006	Final report	

4. TITLE AND SUBTITLE

1-D Thermal Modeling of Layered Materials in Outdoor Environments

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

John O. Curtis

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Environmental Laboratory, U.S. Army Engineer Research and Development Center,
3909 Halls Ferry Road, Vicksburg, MS 39180-6199

8. PERFORMING ORGANIZATION REPORT NUMBER

ERDC/EL TR-06-2

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Army Night Vision and Electronic Sensors Directorate (NVESD),
10221 Burbeck Road, Fort Belvoir, VA 22060-5806

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This report describes a physics-based, but relatively simple, thermal infrared computational tool that can be used in a personal computer (PC) environment to predict temperature profiles of layered media. The tool is a one-dimensional finite difference simulation code (written in FORTRAN) that is executed through a graphical user interface. Its current utility is in helping to design field tests of airborne mine detection sensor systems and to analyze the results of those tests to better understand the performance of the sensor in different environmental conditions. The tool does not address the separate issues of two- and three-dimensional effects, sensor hardware performance, sensor flight paths, or targeting algorithms.

15. SUBJECT TERMS

	Heat flow	Terrain surface
Finite difference code	One-dimensional thermal model	Thermal signatures
Graphical user interface	Temperature model	

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(include area code)*
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED		92	