

QC
807.5
.U6
W6
no.229
c.2

NOAA Technical Memorandum ERL WPL-229



**RTE: A UNIX LIBRARY WITH ON-LINE DOCUMENTATION
AND SAMPLE PROGRAMS FOR MICROWAVE RADIATIVE
TRANSFER CALCULATIONS**

J. C. Reynolds
J. A. Schroeder

Wave Propagation Laboratory
Boulder, Colorado
March 1993

noaa

NATIONAL OCEANIC AND
ATMOSPHERIC ADMINISTRATION

Environmental Research
Laboratories

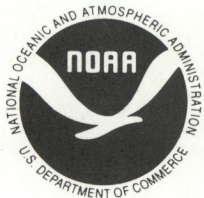
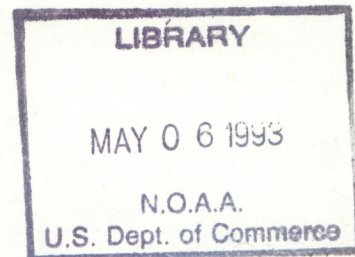
QC
807.5
.U6
W6
No. 229
C.2

NOAA Technical Memorandum ERL WPL-229

**RTE: A UNIX LIBRARY WITH ON-LINE DOCUMENTATION
AND SAMPLE PROGRAMS FOR MICROWAVE RADIATIVE
TRANSFER CALCULATIONS**

J. C. Reynolds
J. A. Schroeder

Wave Propagation Laboratory
Boulder, Colorado
March 1993



**UNITED STATES
DEPARTMENT OF COMMERCE**

NATIONAL OCEANIC AND
ATMOSPHERIC ADMINISTRATION

Environmental Research
Laboratories

NOTICE

Mention of a commercial company or product does not constitute an endorsement by the NOAA Environmental Research Laboratories. Use of information from this publication concerning proprietary products or the tests of such products for publicity or advertising purposes is not authorized.

For sale by the National Technical Information Service, 5285 Port Royal Road
Springfield, VA 22061

CONTENTS

	PAGE
ABSTRACT	1
1. INTRODUCTION	1
2. SCIENTIFIC OVERVIEW	3
3. USER-CALLABLE SUBROUTINES	4
3.1. ViewConst_RTE	6
3.2. ModifyProf_RTE	7
3.3. CldModel91_RTE	10
3.4. TbZenith_RTE	13
3.5. TbScan_RTE	15
3.6. TbCloud_RTE	17
3.7. Radiance_RTE	19
3.8. Weight_RTE	22
4. USE OF RTE LIBRARY	24
4.1. Computing Clear-Sky Brightness Temperatures	24
4.2. Computing Cloudy-sky Brightness Temperatures	25
4.3. Computing Weighting Functions	26
5. INSTALLATION AND TESTING	27
5.1. Creating The RTE Directory Structure	29
5.2. Machine Dependencies	29
5.3. Creating The RTE Library	29
5.4. Installing and Running Sample Programs	30
5.5. Installing The Unix On-line Manual Pages	30
5.6. Updating RTE	31
ACKNOWLEDGMENTS	31

CONTENTS

	PAGE
REFERENCES	31
APPENDIX A: Program TbClear	32
B: Program TbModel	47
C: Program WgtTest	65

RTE: A UNIX LIBRARY WITH ON-LINE DOCUMENTATION AND SAMPLE
PROGRAMS FOR MICROWAVE RADIATIVE TRANSFER CALCULATIONS

J.C. Reynolds and J.A. Schroeder

NOAA Wave Propagation Laboratory
R/E/WP5, 325 Broadway
Boulder, CO 80303

ABSTRACT. The FORTRAN library that the NOAA Wave Propagation Laboratory (WPL) developed to perform radiative transfer calculations for an upward-looking microwave radiometer is described. Although the theory and algorithms have been used for many years in WPL radiometer research, the **Radiative Transfer Equation (RTE)** software has combined them into a toolbox that is portable, readable, application independent, and easy to update. RTE has been optimized for the UNIX environment. However, the FORTRAN source code can be compiled on any platform that provides a Standard FORTRAN 77 compiler. RTE allows a user to do cloud modeling, calibrate radiometers, simulate hypothetical radiometer systems, develop retrieval techniques, and compute weighting functions. The radiative transfer model used is valid for channel frequencies below 1000 GHz in clear conditions and for frequencies below 100 GHz when clouds are present.

1. INTRODUCTION

The **Radiative Transfer Equation (RTE)** software allows the user to perform radiative transfer calculations for a given atmospheric profile. These calculations simulate the downward propagation of radiation observed by an upward-looking microwave radiometer. The FORTRAN source code was constructed from algorithms described by Schroeder and Westwater (1991, 1992). We modified the existing structure to improve modularity and take advantage of modern programming techniques. RTE has been designed for the UNIX environment and consists of three components:

- A FORTRAN library
- On-line manual pages
- Sample programs

Our discussion of RTE assumes a UNIX platform. However, the software can be ported to other platforms by using the following guidelines:

- The FORTRAN source code can be compiled on any platform that supports a Standard FORTRAN 77 Compiler.
- The library can be created with the archive tool supported by the platform.
- The on-line manual pages can be executed on any platform that supports the NROFF text formatter.

RTE's FORTRAN library allows many users to link with the same source code, thus eliminating the need to maintain multiple copies of the software. This technique also helps ensure that all users are operating with the same software version. The library includes a toolbox of special-purpose, user-callable subroutines that are application independent, are unconstrained by I/O formats, and have extensive error-handling features. Currently, RTE includes eight user-callable subroutines but it can easily accommodate additional subroutines to meet future requirements. These eight subroutines are summarized below and are explained in detail in separate subsections of section 3:

- (1) ViewConst_RTE - returns values of physical constants that RTE uses in its radiative transfer calculations
- (2) ModifyProf_RTE - modifies input vertical profiles to better approximate a continuous atmosphere
- (3) CldModel91_RTE - models cloud liquid and ice density profiles using the algorithms of Decker et al. (1978)
- (4) TbZenith_RTE - computes brightness temperatures for a fixed zenith-pointing radiometer antenna; ignores clouds
- (5) TbScan_RTE - computes brightness temperatures for an elevation-angle-scanning radiometer antenna; ignores clouds
- (6) TbCloud_RTE - computes brightness temperatures for an elevation-angle-scanning radiometer antenna; includes the effects of clouds
- (7) Radiance_RTE - returns selected portions of the radiative transfer equation required for other calculations, such as weighting functions

- (8) Weight_RTE - computes weighting functions for a given channel frequency and antenna orientation

RTE's on-line manual pages allow the user to obtain an on-line description of the eight user-callable subroutines. These manual pages provide the same information found in sections 3.1 through 3.8 of this document. There is a general manual entry that gives an overview of the RTE library. This entry can be accessed with the following command: **prompt> man RTE**. Each user-callable subroutine also has its own on-line entry that can be accessed with the man command and the subroutine name. For example: **prompt> man ViewConst_RTE**.

RTE's sample programs (TbClear, TbModel, and WgtTest) demonstrate how to use the library. They can also be used for testing the library after it has been installed. The sample program TbModel produces output similar to the program TBMODEL described by Schroeder and Westwater (1991).

This Technical Memorandum provides a scientific overview, a detailed description of the eight user-callable subroutines, some example FORTRAN sequences that illustrate typical RTE uses, and an installation and testing section. Appendices A, B, and C provide the source code, test data, and example output, for the three sample programs.

2. SCIENTIFIC OVERVIEW

RTE allows the user to model the downward propagation of radiation through the atmosphere in the microwave and millimeter-wave spectral region. Input vertical profiles of atmospheric pressure, temperature, humidity, and cloud water density define the atmospheric conditions. Currently, the library is designed to simulate the radiation observed by an upward-looking microwave radiometer with a specified channel frequency and antenna orientation. The theory and numerical algorithms employed were described by Schroeder and Westwater (1991, 1992). We repeat algorithm discussions when needed to explain subroutine usage. The following is a summary of important restrictions and assumptions:

- All calculations are currently done in single precision. Some machines may have problems with antenna positions represented by low elevation angles.
- We assume that the input profiles represent a vertical cross section of a horizontally stratified atmosphere. Each successive level must be higher in the atmosphere than the previous one (no radiosonde bobbing).

- We assume that discrete radiation sources (e.g., sun or moon) are outside the radiometer field of view.
- Clear-sky radiative transfer calculations that are based on the Liebe & Layton (1987) absorption models, are valid for frequencies up to 1000 GHz. Water vapor and oxygen are the only clear-sky absorbing constituents modeled. We neglect dispersion around the 22.235 GHz water vapor absorption line and in the 60 GHz oxygen complex.
- Cloud radiative transfer calculations are based on the Westwater (1974) cloud absorption model, that assumes the Rayleigh approximation. As a result, these calculations are only valid for non-precipitating clouds with particle radii less than about 100 μm for frequencies less than 100 GHz.

3. USER-CALLABLE SUBROUTINES

This section describes the functionality of the eight special-purpose subroutines currently furnished by RTE. The extension `_RTE` identifies these subroutines as user-callable. These subroutines are application independent, are unconstrained by I/O formats, and have extensive error-handling features. We first discuss the attributes common to all the user-callable subroutines. Then in sections 3.1 through 3.8 we provide the following for each user-callable subroutine:

- A description of the purpose of the subroutine
- The call format that presents the order of the formal arguments
- The declaration statements that present the type and dimension of each formal argument
- A description of each formal input argument
- A description of each formal output argument
- A discussion of the algorithms implemented by the subroutine
- A list of possible error conditions

The user's calling program must employ the same array dimensions as defined by the RTE subroutine. The following table provides the naming scheme and current values for the array dimension parameters:

<u>Parameter name</u>	<u>Current value</u>	<u>Definition</u>
maxl	2000	maximum number of profile levels
maxf	50	maximum number of channel frequencies
maxa	50	maximum number of antenna angles
maxc	10	maximum number of cloud layers

RTE has taken into account the need to control array dimensions for different platforms and scientific uses. A software manager can easily change these values in one central RTE file and rebuild the library (see section 5.6). The user can access the current dimension values through a call to ViewConst_RTE (see section 3.1).

The user's input values are never modified by the RTE user-callable subroutines. All input arrays are copied into working arrays before the computations begin. When the purpose of the RTE subroutine is to modify an input value, the result is returned in a separate output variable.

If an RTE computation is disrupted, an error message is written to the standard error file (unit=0). Each of the user-callable subroutines also provides a return status flag, that verifies the validity of the other returned values:

- status flag = 0: The computations were completed without disruption. All the returned values are valid.
- status flag > 0: A computation was disrupted but the process was able to proceed. Affected returned values are flagged with -999.0. All other returned values are valid.
- status flag < 0: A computation was disrupted and the process could not proceed. None of the returned values are valid or flagged.

The user-callable subroutines perform array boundary checking during all phases of the process. Array indices are checked against the corresponding array dimensions. When an index exceeds its dimension value, an error message is written to the standard error file, the status flag is set to -1, and the subroutine returns.

Input profiles of atmospheric height, pressure, temperature, and relative humidity are checked for their validity based on the following boundaries:

0.0 km MSL $\leq$ height $\leq$ 100.0 km MSL
 0.0 mb $\leq$ pressure $\leq$ 1100.0 mb
 0.0 K $\leq$ temperature $\leq$ 450.0 K
 0.0 $\leq$ relative humidity $\leq$ 1.0

If one of these boundaries is exceeded, an error message is written to the standard error file, the status flag is set to -1, and the subroutine returns. If these boundaries are insufficient for your scientific conditions, a software manager can change them in one central RTE file and rebuild the library (see section 5.6).

The radiative transfer algorithms require a set of input profiles that reach the 50-mb pressure level. If this criterion is not met, an error message is written to the standard error file, the status flag is set to -1, and the subroutine returns. Note that subroutine ModifyProf_RTE (see section 3.2) can be called to extrapolate an input profile up to 50 mb.

3.1. ViewConst_RTE

Subroutine ViewConst_RTE provides the user with values of physical constants, conversion factors, and array dimensions that RTE uses in its radiative transfer calculations.

CALL FORMAT:

```
call ViewConst_RTE (pfrac, tbg, c, hoverk, re, g, epsilon,
                   rdry, rvap, expmax, ghz2hz, db2np,
                   deg2rad, press50, temp50,
                   maxl, maxf, maxa, maxc, status)
```

VARIABLE DECLARATIONS:

```
integer maxl, maxf, maxa, maxc, status
real      pfrac, tbg, c, hoverk, re, g, epsilon, rdry, rvap,
          expmax, ghz2hz, db2np, deg2rad, press50(3), temp50(3)
```

INPUT VARIABLE DESCRIPTIONS:

None

OUTPUT VARIABLE DESCRIPTIONS:

pfrac	- fraction of ambient pressure used for vapor pressure extrapolation; real scalar
tbg	- cosmic background brightness temperature (K); real scalar
c	- speed of light in a vacuum (cm s ⁻¹); real scalar
hoverk	- [Planck constant (J s)] / [Boltzmann constant (J K ⁻¹)]; real scalar

re - mean earth radius (km); real scalar
 g - general gravitational constant (m s^{-2}); real scalar
 epsilon - ratio of molecular weights: (water vapor)/(dry air); real scalar
 rdry - gas constant for dry air ($\text{m s}^{-2} \text{ km K}^{-1}$); real scalar
 rvap - gas constant for water vapor ($\text{mb m}^3 \text{ g}^{-1} \text{ K}^{-1}$); real scalar
 expmax - maximum absolute value for exponential function argument; computer/compiler-dependent; real scalar
 ghz2hz - conversion factor: gigahertz to hertz; real scalar
 db2np - conversion factor: decibels to nepers; real scalar
 deg2rad - conversion factor: degrees to radians; real scalar
 press50 - pressure levels for extrapolating short input profiles up to 50 mb; real vector of length 3
 temp50 - default temperatures at pressure levels defined by press50 ($^{\circ}\text{C}$); real vector of length 3
 maxl - parameter defining maximum number of profile levels; integer scalar
 maxf - parameter defining maximum number of channel frequencies; integer scalar
 maxa - parameter defining maximum number of elevation angles for antenna position; integer scalar
 maxc - parameter defining maximum number of cloud layers; integer scalar
 status - status flag; integer scalar
 status = 0: implies execution complete (all valid values returned)
 status > 0: implies execution complete (valid and flagged values returned)
 status < 0: implies execution incomplete (no valid or flagged values returned)

DISCUSSION:

The physical constants and conversion factors are those described in sections 4.2.4 and 4.2.5 of Schroeder and Westwater (1991).

ERROR CONDITIONS:

- None; status = 0

3.2. ModifyProf_RTE

Subroutine ModifyProf_RTE provides a tool for modifying the input vertical profiles by extrapolation, interpolation, and/or truncation to represent better the atmosphere being observed. This subroutine returns a set of profiles modified to the caller's specifications as determined by the input argument list flags (extrap50, extrap1, zstart, maxdp).

CALL FORMAT:

```
call ModifyProf_RTE (nlX, zX, pX, tkX, rhX, extrap50, extrap1,  
                    zstart, maxdp, raoblat, ice,  
                    nl, z, p, tk, rh, status)
```

VARIABLE DECLARATIONS:

```
parameter (maxl=2000)  
integer    nlX, extrap50, extrap1, ice, nl, status  
real       zX(maxl), pX(maxl), tkX(maxl), rhX(maxl),  
           zstart, maxdp, raoblat,  
           z(maxl), p(maxl), tk(maxl), rh(maxl),
```

INPUT VARIABLE DESCRIPTIONS:

```
nlX      - number of levels in each input profile ( $\leq$  maxl);  
           integer scalar  
zX       - height profile (km MSL); real vector of length maxl  
pX       - pressure profile (mb); real vector of length maxl  
tkX      - temperature profile (K); real vector of length maxl  
rhX      - relative humidity profile (fraction); real vector  
           of length maxl  
extrap50 - flag to extrapolate short profiles up to 50 mb;  
           integer scalar  
           extrap50 > 0: extrapolate  
           extrap50  $\leq$  0: do not extrapolate  
extrap1  - flag to extrapolate profiles from 50 to 0.1 mb;  
           integer scalar  
           extrap1 > 0: extrapolate  
           extrap1  $\leq$  0: do not extrapolate  
zstart   - flag for simulated antenna height (km MSL); real  
           scalar  
           zstart > zX(1): antenna height is simulated at  
                           zstart  
           zstart  $\leq$  zX(1): antenna height is simulated at  
                           zX(1)  
maxdp    - flag for maximum acceptable pressure difference  
           between adjacent profile levels (mb); real scalar  
           maxdp > 0: Insert level(s) where pressure  
                       profile levels are separated by more  
                       than maxdp (mb)  
           maxdp = 0: Insert more levels near the antenna  
                       and fewer aloft using a traditional  
                       algorithm  
           maxdp < 0: do not insert any levels into the  
                       profiles  
raoblat  - latitude associated with input profiles (degrees),  
           used to select coefficients for temperature  
           extrapolation; real scalar
```


ice - saturation vapor pressure (e_s) switch; integer scalar
 ice = 0: Compute e_s over water only
 ice = 1: Compute e_s over water (temp. > -10 °C)
 Compute e_s over ice (temp. ≤ -10 °C)

OUTPUT VARIABLE DESCRIPTIONS:

nl - number of levels in each output profile (≤ maxl); integer scalar
 z - height profile (km MSL); real vector of length maxl
 p - pressure profile (mb); real vector of length maxl
 tk - temperature profile (K); real vector of length maxl
 rh - relative humidity profile (fraction); real vector of length maxl
 status - status flag; integer scalar
 status = 0: implies execution complete (all valid values returned)
 status > 0: implies execution complete (valid and flagged values returned)
 status < 0: implies execution incomplete (no valid or flagged values returned)

DISCUSSION:

The profile modification algorithms are those described in section 3.1 of Schroeder and Westwater (1991).

If the *extrap50* flag is set, the short input profiles are extrapolated up to 50 mb by adding three new levels. The user's input profiles should reach the 300-mb pressure level, since the default pressure levels are currently defined at 200, 100, and 50 mb. The user can call *ViewConst_RTE* to access these default pressure levels and associated temperatures (see section 3.1: variables *press50* and *temp50*). The humidity is extrapolated assuming a constant mixing ratio defined as a constant fraction of ambient pressure (see section 3.1: variable *pfrac*). Also computed are the geopotential heights at the extrapolated levels.

If the *extrap1* flag is set, the input profiles are extrapolated up to 0.1 mb. The input profiles must reach the 50-mb pressure level. New temperatures are estimated at 15 levels above 50 mb from temperatures at 700, 500, 300, 200, 100, and 50 mb, using the regression coefficients of Crosby et al. (1973). The humidity is extrapolated assuming a constant mixing ratio defined as a constant fraction of ambient pressure (see section 3.1: variable *pfrac*). Also computed are the geopotential heights at the extrapolated levels.

The *zstart* flag represents the simulated antenna height. All profile levels below this height are discarded.

The *maxdp* flag represents the maximum acceptable pressure difference (mb) between adjacent profile levels. For example, if *maxdp*=10, additional levels will be inserted between levels separated by more than 10 mb. Setting *maxdp*=0 selects an algorithm that relaxes the difference in pressure (dp) constraint with height. The algorithm enforces a dp of 5 mb in the first 50 mb above the simulated antenna height *zstart*; dp increases by 5 mb for every 50-mb decrease in pressure. For example, between 100 and 150 mb above *zstart*, the pressure difference between levels cannot exceed 15 mb. When it does, level(s) are inserted until it does not.

ERROR CONDITIONS:

- None; status = 0
- An array index exceeds its array dimension (see section 3); status < 0
- Input values of atmospheric height, pressure, temperature, or relative humidity exceed their boundaries (see section 3); status < 0
- Could not find the temperature at interpolated pressure levels; status < 0

3.3. CldModel91_RTE

The subroutine *CldModel91_RTE* provides a tool for modeling profiles of cloud liquid and ice density from the input vertical profiles of temperature and humidity. The input vertical profiles must reach the 50-mb pressure level. The simulated antenna height is assumed to be at the first profile level. The user can call subroutine *ModifyProf_RTE* first to insure that the input profiles meet these criteria (see section 3.2).

CALL FORMAT:

```
call CldModel91_RTE ( nlX, zX, pX, tkX, rhX, nang, angles,
                    ice, cldrh, cldthick, cldmax,
                    nl, z, p, tk, rh, denliq, denice,
                    denfrac, nlay, cldtop, cldbase, status )
```

VARIABLE DECLARATIONS:

```
parameter (maxl=2000, maxa=50, maxc=10)
integer    nlX, nang, ice, cldmax, nl, nlay, status
```



```

real      zX(maxl), pX(maxl), tkX(maxl), rhX(maxl),
          angles(maxa), cldrh, cldthick,
          z(maxl), p(maxl), tk(maxl), rh(maxl),
          denliq(maxl), denice(maxl), denfrac(3),
          cldbase(maxc), cldtop(maxc)

```

INPUT VARIABLE DESCRIPTIONS:

```

nlX      - number of levels in each input profile; integer
          scalar ( $\leq$  maxl)
zX       - height profile (km MSL); real vector of length maxl
pX       - pressure profile (mb); real vector of length maxl
tkX      - temperature profile (K); real vector of length maxl
rhX      - relative humidity profile (fraction); real vector
          of length maxl
nang     - number of elevation angles; integer scalar
          ( $\leq$  maxa)
angles   - elevation angles of antenna (deg); real vector of
          length maxa
ice      - saturation vapor pressure ( $e_s$ ) switch; integer
          scalar
          ice = 0: Compute  $e_s$  over water only
          ice = 1: Compute  $e_s$  over water (temp.  $> -10$  °C)
               Compute  $e_s$  over ice (temp.  $\leq -10$  °C)
cldrh    - threshold relative humidity (fraction) used to
          define a cloud; real scalar
cldthick - minimum acceptable cloud thickness (km); detected
          cloud layers thinner than cldthick are ignored;
          real scalar
cldmax   - maximum number of cloud layers permitted; integer
          scalar ( $\leq$  maxc)

```

OUTPUT VARIABLE DESCRIPTIONS:

```

nl      - number of levels in each output profile; integer
          scalar ( $\leq$  maxl)
z       - height profile (km MSL); real vector of length
          maxl
p       - pressure profile (mb); real vector of length maxl
tk      - temperature profile (K); real vector of length
          maxl
rh      - relative humidity profile (fraction); real vector
          of length maxl
denliq  - liquid density profile ( $\text{g m}^{-3}$ ); real vector of
          length maxl
denice  - ice density profile ( $\text{g m}^{-3}$ ); real vector of length
          maxl

```


denfrac - cloud water density fraction; real vector of length 3
 nlay - number of cloud layers identified; integer scalar ($\leq$ cldmax)
 cldtop - cloud top heights (km MSL); real vector of length maxc
 cldbase - cloud base heights (km MSL); real vector of length maxc
 status - status flag; integer scalar
 status = 0: implies execution complete (all valid values returned)
 status > 0: implies execution complete (valid and flagged values returned)
 status < 0: implies execution incomplete (no valid or flagged values returned)

DISCUSSION:

This subroutine uses the cloud model algorithms of Decker et al. (1978), whose implementation is described by Schroeder and Westwater (1991).

For simplicity, this cloud model assumes the Rayleigh approximation, under which scattering is negligible relative to absorption, and absorption is independent of cloud particle size distribution. These assumptions restrict the model to non-precipitating clouds with particle radii less than about 100 μm for frequencies less than 100 GHz. Therefore, the algorithms are not adequate for modeling rain, large droplets, or large ice particles in clouds. The validity of the Rayleigh approximation is discussed by van de Hulst (1981) and Ulaby et al. (1981).

The user controls the cloud water density profiles through the input variables *cldrh*, *cldthick*, and *cldmax*. A cloud exists where the relative humidity equals or exceeds the *cldrh* value. The relative humidity profile is searched from the bottom upward for clouds. Cloud bases and tops are defined at levels where the *cldrh* threshold is crossed when entering or leaving a cloud. If the input profiles have no level precisely at a cloud base or top, a new level is added to each profile (by interpolation) at the cloud boundary. If the resulting cloud is thinner than *cldthick*, the cloud is discarded. The search for clouds continues until the specified maximum number of cloud layers, *cldmax*, are found or the top of the profile is reached.

The cloud water density (ρ) associated with each identified cloud is determined from the cloud thickness using the densest model of Decker et al. (1978), where $0.2 \text{ g m}^{-3} \leq \rho \leq 0.8 \text{ g m}^{-3}$. The same ρ value is assigned to each profile level in a given cloud. The input temperature profile is used

to divide the cloud water density between the ice and liquid, allowing the ice fraction to increase exponentially from 0.0 at 0 °C to 1.0 at -30 °C. The ice density profile, *denice*, is constructed by applying the appropriate ice fraction to the assigned cloud water density at each profile level. The liquid density profile, *denliq*, is the difference between the ρ and *denice* values at each level. Thus, *denliq* and *denice* represent only the densest model of Decker et al. (1978). Clear profile levels are set to zero. All three of their models can be obtained by multiplying *denliq* and *denice* by the fractions returned in *denfrac* (i.e. 0.25, 0.50, 1.00).

ERROR CONDITIONS:

- None; status = 0
- An array index exceeds its array dimension (see section 3); status < 0
- Input values of atmospheric height, pressure, temperature, or relative humidity exceed their boundaries (see section 3); status < 0
- Input profiles do not reach the 50-mb pressure level; status < 0

3.4. TbZenith_RTE

Subroutine TbZenith_RTE returns clear-sky brightness temperature, mean radiating temperature, and integrated absorption at specified channel frequencies of an upward-looking microwave radiometer. The antenna is fixed in the zenith direction. The input vertical profiles must reach the 50-mb pressure level. The simulated antenna height is assumed to be at the first profile level. The user can call subroutine ModifyProf_RTE first to insure that the input profiles meet these criteria (see section 3.2).

CALL FORMAT:

```
call TbZenith_RTE (nl, z, p, tk, rh, nfrq, frq, ice
                  tbttotal, tmr, tauclr, status)
```

VARIABLE DECLARATIONS:

```
parameter (maxl=2000, maxf=50)
integer    nl, nfrq, ice, status
real       z(maxl), p(maxl), tk(maxl), rh(maxl),
           frq(maxf), tbttotal(maxf), tmr(maxf), tauclr(maxf)
```


INPUT VARIABLE DESCRIPTIONS:

nl - number of levels in each profile ($\leq$ maxl); integer scalar
z - height profile (km MSL); real vector of length maxl
p - pressure profile (mb); real vector of length maxl
tk - temperature profile (K); real vector of length maxl
rh - relative humidity profile (fraction); real vector of length maxl
nfrq - number of channel frequencies ($\leq$ maxf); integer scalar
frq - channel frequencies (GHz); real vector of length maxf
ice - saturation vapor pressure (e_s) switch; integer scalar
ice = 0: Compute e_s over water only
ice = 1: Compute e_s over water (temp. > -10 °C)
Compute e_s over ice (temp. ≤ -10 °C)

OUTPUT VARIABLE DESCRIPTIONS:

tbttotal - brightness temperature (K) including cosmic background; real vector of length maxf
tmr - mean radiating temperature of the atmosphere (K); real vector of length maxf
tauc1r - total absorption (N_p) integrated over the ray path; real vector of length maxf
status - status flag; integer scalar
status = 0: implies execution complete (all valid values returned)
status > 0 : implies execution complete (valid and flagged values returned)
status < 0 : implies execution incomplete (no valid or flagged values returned)

DISCUSSION:

This subroutine uses the algorithms described by Schroeder and Westwater (1991).

ERROR CONDITIONS:

- None; status = 0
- An array index exceeds its array dimension (see section 3); status < 0
- Input values of atmospheric height, pressure, temperature, or relative humidity exceed their boundaries (see section 3); status < 0

- Input profiles do not reach the 50-mb pressure level; status < 0
- Negative value encountered during exponential integration; status < 0

3.5. TbScan_RTE

Subroutine TbScan_RTE returns clear-sky brightness temperature, mean radiating temperature, and integrated absorption at specified channel frequencies of an upward-looking microwave radiometer. Antenna orientation is defined by the specified elevation angles. The input vertical profiles must reach the 50-mb pressure level. The simulated antenna height is assumed to be at the first profile level. The user can call subroutine ModifyProf_RTE first to insure that the input profiles meet these criteria (see section 3.2).

CALL FORMAT:

```
call TbScan_RTE (nl, z, p, tk, rh, nfrq, frq,
                 nang, angles, ice,
                 tbttotal, tmr, tauclr, status)
```

VARIABLE DECLARATIONS:

```
parameter (maxl=2000, maxf=50, maxa=50)
integer    nl, nfrq, nang, ice, status
real       z(maxl), p(maxl), tk(maxl), rh(maxl),
           frq(maxf), angles(maxa), tbttotal(maxf,maxa),
           tmr(maxf,maxa), tauclr(maxf,maxa)
```

INPUT VARIABLE DESCRIPTIONS:

nl	- number of levels in each profile ($\leq$ maxl); integer scalar
z	- height profile (km MSL); real vector of length maxl
p	- pressure profile (mb); real vector of length maxl
tk	- temperature profile (K); real vector of length maxl
rh	- relative humidity profile (fraction); real vector of length maxl
nfrq	- number of channel frequencies ($\leq$ maxf); integer scalar
frq	- channel frequencies (GHz); real vector of length maxf
nang	- number of elevation angles ($\leq$ maxa); integer scalar
angles	- elevation angles of antenna (deg); real vector of length maxa

ice - saturation vapor pressure (e_s) switch; integer scalar
 ice = 0: Compute e_s over water only
 ice = 1: Compute e_s over water (temp. > -10 °C)
 Compute e_s over ice (temp. ≤ -10 °C)

OUTPUT VARIABLE DESCRIPTIONS:

tbtotal - brightness temperature (K) including cosmic background; real 2-dimensional matrix (maxf by maxa)
 tmr - mean radiating temperature of the atmosphere (K); real 2-dimensional matrix (maxf by maxa)
 tauclr - total absorption (N_p) integrated along each ray path; real 2-dimensional matrix (maxf by maxa)
 status - status flag; integer scalar
 status = 0: implies execution complete (all valid values returned)
 status > 0: implies execution complete (valid and flagged values returned)
 status < 0: implies execution incomplete (no valid or flagged values returned)

DISCUSSION:

This subroutine uses the algorithms described by Schroeder and Westwater (1991).

ERROR CONDITIONS:

- None; status = 0
- An array index exceeds its array dimension (see section 3); status < 0
- Input values of atmospheric height, pressure, temperature, or relative humidity exceed their boundaries (see section 3); status < 0
- Input profiles do not reach the 50-mb pressure level; status < 0
- Refractive index drops below 1.0 during raytracing algorithm; status > 0
- Ducting conditions encountered during raytracing algorithm; status > 0
- Negative value encountered during exponential integration (see section 5.2); status > 0

3.6. TbCloud_RTE

The subroutine TbCloud_RTE returns brightness temperature, mean radiating temperature, and integrated absorption for clear or cloudy conditions at specified channel frequencies of an upward-looking microwave radiometer. Antenna orientation is defined by the specified elevation angles. Also returned are the additional integrated quantities that were output by program TBMODEL as described in section 4.1.4 of Schroeder and Westwater (1991). The input vertical profiles must reach the 50-mb pressure level. The simulated antenna height is assumed to be at the first profile level. The user can call subroutine ModifyProf_RTE first to insure that the input atmospheric profiles of height, pressure, temperature, and relative humidity meet these criteria (see section 3.2). The cloud liquid and ice density profiles required as input can be constructed by calling CldModel91_RTE (see section 3.3).

CALL FORMAT:

```
call TbCloud_RTE (nl, z, p, tk, rh, denliq, denice,
                  nfrq, frq, nang, angles,
                  nlay, cldbase, clddtop, ice,
                  tbttotal, tbatm, tmr, tmrcld,
                  tauwet, taudry, tauhq, tauice,
                  srho, swet, sdry, slhq, sice, status)
```

VARIABLE DECLARATIONS:

```
parameter (maxl=2000, maxf=50, maxa=50, maxc=10)
integer    nl, nfrq, nang, nlay, ice, status
real       z(maxl), p(maxl), tk(maxl), rh(maxl),
           denliq(maxl), denice(maxl), frq(maxf),
           angles(maxa), cldbase(maxc), clddtop(maxc),
           tbttotal(maxf,maxa), tbatm(maxf,maxa),
           tmr(maxf,maxa), tmrcld(maxf,maxa),
           tauwet(maxf,maxa), taudry(maxf,maxa),
           tauhq(maxf,maxa), tauice(maxf,maxa),
           srho(maxa), swet(maxa), sdry(maxa),
           slhq(maxa), sice(maxa)
```

INPUT VARIABLE DESCRIPTIONS:

nl	- number of levels in each profile ($\leq$ maxl); integer scalar
z	- height profile (km MSL); real vector of length maxl
p	- pressure profile (mb); real vector of length maxl
tk	- temperature profile (K); real vector of length maxl
rh	- relative humidity profile (fraction); real vector of length maxl
denliq	- cloud liquid density profile (g m^{-3}); real vector of length maxl

denice - cloud ice density profile (g m^{-3}); real vector of length maxl
 nfrq - number of channel frequencies ($\leq$ maxf); integer scalar
 frq - channel frequencies (GHz); real vector of length maxf
 nang - number of elevation angles ($\leq$ maxa); integer scalar
 angles - elevation angles of antenna(deg); real vector of length maxa
 nlay - number of cloud layers identified; integer scalar ($\leq$ maxc)
 cldbase - cloud base heights (km MSL); real vector of length maxc
 cldtop - cloud top heights (km MSL); real vector of length maxc
 ice - saturation vapor pressure (e_s) switch; integer scalar
 ice = 0: Compute e_s over water only
 ice = 1: Compute e_s over water (temp. > -10 °C)
 Compute e_s over ice (temp. ≤ -10 °C)

OUTPUT VARIABLE DESCRIPTIONS:

tbttotal - brightness temperature (K) including cosmic background; real 2-dimensional matrix (maxf by maxa)
 tbatm - atmospheric brightness temperature (K), no cosmic background; real 2-dimensional matrix (maxf by maxa)
 tmr - mean radiating temperature of the atmosphere (K); real 2-dimensional matrix (maxf by maxa)
 tmrcld - mean radiating temperature (K) of the lowest cloud layer; real 2-dimensional matrix (maxf by maxa)
 tauwet - water vapor absorption integrated along each ray path (N_p); real 2-dimensional matrix (maxf by maxa)
 taudry - dry air absorption integrated along each ray path (N_p); real 2-dimensional matrix (maxf by maxa)
 taucliq - cloud liquid absorption integrated along each ray path (N_p); real 2-dimensional matrix (maxf by maxa)
 tauice - cloud ice absorption integrated along each ray path (N_p); real 2-dimensional matrix (maxf by maxa)
 srho - water vapor density integrated along each ray path (cm); real vector of length maxa
 swet - wet refractivity integrated along each ray path (cm); real vector of length maxa
 sdry - dry refractivity integrated along each ray path (cm); real vector of length maxa
 sliq - cloud liquid density integrated along each ray path (cm); real vector of length maxa
 sice - cloud ice density integrated along each ray path (cm); real vector of length maxa

status - status flag; integer scalar
status = 0: implies execution complete
(all valid values returned)
status > 0: implies execution complete
(valid and flagged values returned)
status < 0: implies execution incomplete
(no valid or flagged values returned)

DISCUSSION:

This subroutine uses the algorithms described by Schroeder and Westwater (1991).

ERROR CONDITIONS:

- None; status = 0
- An array index exceeds its array dimension (see section 3); status < 0
- Input values of atmospheric height, pressure, temperature, or relative humidity exceed their boundaries (see section 3); status < 0
- Input profiles do not reach the 50-mb pressure level; status < 0
- Refractive index drops below 1.0 during raytracing algorithm; status > 0
- Ducting conditions encountered during raytracing algorithm; status > 0
- Negative value encountered during exponential integration; status > 0
- Absorption too large to exponentiate; machine-dependent value (see section 5.2); status > 0

3.7. Radiance_RTE

Subroutine Radiance_RTE computes the modified Planck radiance (Planck function without the constant $2h\nu^3/c^2$) and intermediate quantities for an upward-looking microwave radiometer with a specified channel frequency and antenna orientation. The input vertical profiles must reach the 50-mb pressure level. The simulated antenna height is assumed to be at the first profile level. The user can call subroutine ModifyProf_RTE first to insure that the input atmospheric

profiles of height, pressure, temperature, and relative humidity meet these criteria (see section 3.2). The cloud liquid and ice density profiles required as input can be constructed by calling subroutine CldModel91_RTE (see section 3.3).

CALL FORMAT:

```
call Radiance_RTE (nl, z, p, tk, rh, denliq, denice,
                  frq, angle, ice,
                  hvk, bakgrnd, boft, boftatm, boftb, alpha,
                  ds, tauprof, status)
```

VARIABLE DECLARATIONS:

```
parameter (maxl=2000)
integer    nl, status
real       z(maxl), p(maxl), tk(maxl), rh(maxl), denliq(maxl),
           denice(maxl), frq, angle,
           hvk, bakgrnd, boft(maxl), boftatm(maxl),
           boftb, alpha(maxl), ds(maxl), tauprof(maxl)
```

INPUT VARIABLE DESCRIPTIONS:

nl	- number of levels in each profile ($\leq$ maxl); integer scalar
z	- height profile (km MSL); real vector of length maxl
p	- pressure profile (mb); real vector of length maxl
tk	- temperature profile (K); real vector of length maxl
rh	- relative humidity profile (fraction); real vector of length maxl
denliq	- cloud liquid density profile (g m^{-3}); real vector of length maxl
denice	- cloud ice density profile (g m^{-3}); real vector of length maxl
frq	- channel frequency (GHz); real scalar
angle	- elevation angle of antenna (deg); real scalar
ice	- saturation vapor pressure (e_s) switch; integer scalar
	ice = 0: Compute e_s over water only
	ice = 1: Compute e_s over water (temp. > -10 °C)
	Compute e_s over ice (temp. ≤ -10 °C)

OUTPUT VARIABLE DESCRIPTIONS:

hvk	- (Planck constant * frequency) / Boltzmann constant; real scalar
bakgrnd	- cosmic background term of radiative transfer equation; real scalar
boft	- modified Planck function profile from input temperature profile; real vector of length maxl
boftatm	- atmospheric radiance profile; boftatm(i) = integral from 0 to i; real vector of length maxl

boftb - total modified radiance including cosmic background; real scalar

alpha - total absorption profile ($N_p \text{ km}^{-1}$); real vector of length maxl

ds - refracted path length profile (km); real vector of length maxl

tauprof - absorption integral profile (N_p); tauprof(i) = integral from 0 to i; real vector of length maxl

status - status flag; integer scalar

status = 0: implies execution complete
(all valid values returned)

status > 0: implies execution complete
(valid and flagged values returned)

status < 0: implies execution incomplete
(no valid or flagged values returned)

DISCUSSION:

This subroutine uses the algorithms described in section 2.1 of Schroeder and Westwater (1991).

This subroutine returns pieces of the radiative transfer equation that are useful for calculations other than brightness temperature, such as weighting functions (see section 3.8). The so-called "modified Planck function" is the Planck function $B(T)$ with the constant term $2h\nu^3/c^2$ removed.

ERROR CONDITIONS:

- None; status = 0
- An array index exceeds its array dimension (see section 3); status < 0
- Input values of atmospheric height, pressure, temperature, or relative humidity exceed their boundaries (see section 3); status < 0
- Input profiles do not reach the 50-mb pressure level; status < 0
- Refractive index drops below 1.0 during raytracing algorithm; status < 0
- Ducting conditions encountered during raytracing algorithm; status < 0
- Negative value encountered during exponential integration; status < 0

3.8. Weight_RTE

Subroutine Weight_RTE computes a weighting function profile for a specified atmospheric parameter (temperature, pressure, water vapor pressure, dry air density, or cloud liquid density). Computations apply to an upward-looking microwave radiometer with a specified channel frequency and antenna orientation. The input vertical profiles must reach the 50-mb pressure level. The simulated antenna height is assumed to be at the first profile level. The user can call subroutine ModifyProf_RTE first to insure that the input atmospheric profiles of height, pressure, temperature, and relative humidity meet these criteria (see section 3.2). The cloud liquid and ice density profiles required as input can be constructed by calling subroutine CldModel91_RTE (see section 3.3). The radiative transfer quantities required as input can be obtained by calling subroutine Radiance_RTE (see section 3.7).

CALL FORMAT:

```
call Weight_RTE (nl, z, p, tk, rh, denliq, denice,  
                frq, angle, ice, bakgrnd, boft, boftatm,  
                boftb, alpha, ds, tauprof, Xflag,  
                Wx, status)
```

VARIABLE DECLARATIONS:

```
parameter (maxl=2000)  
integer    nl, Xflag, status  
real       z(maxl), p(maxl), tk(maxl), rh(maxl), denliq(maxl),  
           denice(maxl), frq, angle, bakgrnd, boft(maxl),  
           boftatm(maxl), boftb, alpha(maxl), ds(maxl),  
           tauprof(maxl), Wx(maxl)
```

INPUT VARIABLE DESCRIPTIONS:

nl	- number of levels in each profile ($\leq$ maxl); integer scalar
z	- height profile (km MSL); real vector of length maxl
p	- pressure profile (mb); real vector of length maxl
tk	- temperature profile (K); real vector of length maxl
rh	- relative humidity profile (fraction); real vector of length maxl
denliq	- cloud liquid density profile (g m^{-3}); real vector of length maxl
denice	- cloud ice density profile (g m^{-3}); real vector of length maxl
frq	- channel frequency (GHz); real scalar
angle	- elevation angle of antenna (deg); real scalar

ice - saturation vapor pressure (e_s) switch; integer scalar
 ice = 0: Compute e_s over water only
 ice = 1: Compute e_s over water (temp. > -10 °C)
 Compute e_s over ice (temp. ≤ -10 °C)

bakgrnd - cosmic background term of radiative transfer equation; real scalar

boft - modified Planck function profile from input temperature profile; real vector of length maxl

boftatm - atmospheric radiance profile; boftatm(i) = integral from 0 to i; real vector of length maxl

boftb - total modified radiance including cosmic background; real scalar

alpha - total absorption profile ($N_p \text{ km}^{-1}$); real vector of length maxl

ds - refracted path length profile (km); real vector of length maxl

tauprof - absorption integral profile (N_p); tauprof(i) = integral from 0 to i; real vector of length maxl

Xflag - flag identifying the atmospheric parameter as follows:
 (1) temperature
 (2) pressure
 (3) water vapor density
 (4) dry air density
 (5) cloud liquid density
 integer scalar

OUTPUT VARIABLE DESCRIPTIONS:

Wx - weighting function profile; units depend on Xflag as follows:
 (1) km^{-1}
 (2) $\text{K km}^{-1} \text{ mb}^{-1}$
 (3) $\text{K km}^{-1} \text{ g m}^{-3}$
 (4) $\text{K km}^{-1} \text{ g m}^{-3}$
 (5) $\text{K km}^{-1} \text{ g m}^{-3}$
 real vector of length maxl

status - status flag; integer scalar
 status = 0: implies execution complete (all valid values returned)
 status > 0: implies execution complete (valid and flagged values returned)
 status < 0: implies execution incomplete (no valid or flagged values returned)

DISCUSSION:

The theory and numerical algorithms that RTE uses to compute weighting functions were given by Schroeder and Westwater (1992).

In this subroutine, a weighting function for a particular atmospheric parameter is defined as the change in brightness temperature that results from perturbing the parameter by a small amount at a single profile level over a layer 1 km thick. By successively perturbing each profile level, this subroutine computes a weighting function profile.

ERROR CONDITIONS:

- None; status = 0
- An array index exceeds its array dimension (see section 3); status < 0
- Input values of atmospheric height, pressure, temperature, or relative humidity exceed their boundaries (see section 3); status < 0
- Input profiles do not reach the 50-mb pressure level; status < 0

4. USE OF RTE LIBRARY

In this section we provide example program sequences of RTE subroutine calls. These examples illustrate typical RTE uses. For clarity, we present only the name of the subroutines, and omit all declarations, parameters, and error checking. We use ! to represent comments, we use ... to represent arbitrary sequences of other statements, and we use do:endloop to represent iterative looping. For a detailed description of the RTE subroutines see section 3. Note that the calls to subroutine ModifyProf_RTE are optional, as long as the input profiles satisfy the conditions of the other RTE subroutines. Also note that calls to subroutine CldModel91_RTE can be replaced with the user's own subroutine that computes cloud water density profiles.

4.1. Computing Clear-Sky Brightness Temperatures

We demonstrate here three typical sequences that can be used to compute brightness temperatures for clear-sky conditions. Each of these methods will perform computations for multiple-channel frequencies of an upward-looking microwave radiometer. The examples progress from the simplest to the most complex.

The following example computes brightness temperature, mean radiating temperature, and integrated absorption for a fixed zenith-pointing antenna:


```

...      ! input data
call ModifyProf_RTE ! modify input profiles (optional)
call TbZenith_RTE   ! compute brightness temperatures
...      ! output data products

```

The next sequence can be used to compute brightness temperature, mean radiating temperature, and integrated absorption for an elevation-angle-scanning antenna. This example can be used for both zenith and off-zenith antenna positions:

```

...      ! input data
call ModifyProf_RTE ! modify input profiles (optional)
call TbScan_RTE     ! compute brightness temperatures
...      ! output data products

```

The last sequence produces brightness temperature, mean radiating temperature, and integrated absorption, as well as other integrated quantities that the previous examples do not. This example works for an elevation-angle-scanning antenna with both zenith and off-zenith antenna positions. It also assumes that the input cloud liquid and ice density profiles and the input number of cloud layers are set to zero:

```

...      ! input data
call ModifyProf_RTE ! modify input profiles (optional)
call TbCloud_RTE    ! compute brightness temperatures
...      ! output data products

```

4.2. Computing Cloudy-Sky Brightness Temperatures

We demonstrate here some possible sequences for computing cloudy-sky brightness temperatures. Each of these methods will perform computations for multiple-channel frequencies of an elevation-angle-scanning antenna of an upward-looking microwave radiometer. These algorithms apply to both zenith and off-zenith antenna positions. The examples progress from the simplest to the most complex.

The first example shows how to compute brightness temperatures and other related values from one of the cloud models returned by subroutine CldModel91_RTE:

```

...      ! input data
call ModifyProf_RTE ! modify input profiles (optional)
call CldModel91_RTE ! construct cloud density profiles
...      ! assign cloud density model
call TbCloud_RTE    ! compute brightness temperatures
...      ! output data products

```


The next example shows how to insert an alternate cloud density modeling routine into the previous cloudy-sky computation:

```

...           ! input data
call ModifyProf_RTE ! modify input profiles (optional)
...           ! construct cloud density profiles
call TbCloud_RTE   ! compute brightness temperatures
...           ! output data products

```

The last example depicts the same sequence of computations as the program TBMODEL described by Schroeder and Westwater (1991). With this set of calls we compute brightness temperatures (Tb) and other related values for one clear-sky and three cloud density models. This structure is also implemented in the sample program TbModel provided in appendix B:

```

...           ! input data
call ModifyProf_RTE ! modify input profiles (optional)
call CldModel91_RTE ! construct cloud density profiles
...           ! assign clear-sky model
call TbCloud_RTE   ! compute Tb for clear-sky
...           ! output data products
do m = 1, 3        ! loop of each cloud model
...           ! assign cloud density model
    call TbCloud_RTE ! compute Tb for cloudy-sky
...           ! output data products
endloop

```

4.3. Computing Weighting Functions

This section illustrates some possible sequences for computing weighting functions for an upward-looking microwave radiometer. These algorithms apply to both zenith and off-zenith antenna positions. The examples progress from the simplest to the most complex.

The first sequence can be used to compute a weighting function profile given clear-sky conditions. This example assumes a single channel frequency, antenna position, and weighting parameter:

```

...           ! input data
call Radiance_RTE ! obtain radiative transfer values
call Weight_RTE   ! compute weighting function
...           ! output data products

```


The following set of RTE calls describes the sequence WPL currently uses to compute weighting functions. Here we compute weighting functions for multiple channel frequencies, antenna positions, and weighting parameters. We assume clear-sky conditions. This structure is also implemented in the sample program WgtTest provided in appendix C:

```

...                               ! input data
do j = 1, nfrq                     ! handle each channel frequency
  do k = 1, nang                     ! handle each antenna position
    call Radiance_RTE               ! obtain radiative transfer values
    ...                             ! output data products
    do l = 1, nx                     ! handle each weighting parameter
      call Weight_RTE               ! compute weighting function
    endloop
  ...                               ! output data products
endloop
endloop

```

The next sequence of RTE calls illustrates how to incorporate cloud modeling into the computation of weighting functions. Here we compute weighting functions for multiple channel frequencies, antenna positions, and weighting parameters. We use only one cloud density model returned by the subroutine CldModel91_RTE. Note that the call to subroutine ModifyProf_RTE is optional and the call to subroutine CldModel91_RTE can be replaced with the user's own routine.

```

...                               ! input data
call ModifyProf_RTE                ! modify input profiles (optional)
call CldModel91_RTE                ! construct cloud density profiles
...                               ! assign cloud density model
do j = 1, nfrq                     ! handle each channel frequency
  do k = 1, nang                     ! handle each antenna position
    call Radiance_RTE               ! obtain radiative transfer values
    do l = 1, nx                     ! handle each weighting parameter
      call Weight_RTE               ! compute weighting function
    endloop
  endloop
endloop
endloop

```

5. INSTALLATION AND TESTING

This section is important to the system manager who will be installing the RTE software. It describes the step-by-step installation procedure for a System V/Release 3 UNIX platform. A UNIX tar file, *rte.tar*, and an informational postscript file, *readme.ps*, can be obtained by writing to

Jannelle Reynolds, R/E/WP5
NOAA/Wave Propagation Laboratory
325 Broadway
Boulder Co, 80303

email: jcr@wpl.erl.gov

The tar file contains the library source code, on-line manual pages, and sample programs. The readme file is a postscript formatted file that contains the same installation and testing procedure described in this document. Below is a brief summary of the procedure we use to manage the RTE software. We also include some brief suggestions for installation on other platforms.

FILE GENERATION:

RTE uses the *make* utility to generate and update the library and the sample-program executables. The *make* utility reads its information from the various *makefiles*. This utility can be replaced with a batch process supported by your platform.

BATCH PROCESSING:

RTE uses c-shell scripts to perform its batch processing. These scripts enable the testing sequence for each sample program to be initiated with only a few commands.

COMPILING THE SOURCE CODE:

RTE uses the *f77* command to compile all source code. This command can be replaced with the Standard FORTRAN 77 Compiler supported by your platform. This change will need to be made in all *makefiles*.

CREATING THE ARCHIVE LIBRARY:

RTE uses the *ar* command to create the archived library named *libRTE.a*. We do not use the *ranlib* function. RTE's sample programs use the *libRTE.a* file to link with the library source code. This command can be replaced with the archive tool supported by your platform.

INSTALLING ON-LINE MANUAL PAGES:

RTE uses the *NROFF* text formatter to format the manual pages. The *man* utility is used to view the manual pages on-line. If the *NROFF* text formatter is not supported, modify the manual pages for your own formatter and add them to your platform's on-line help utility.

5.1. Creating The RTE Directory Structure

Before unpacking the tar file, verify that sufficient disk space is available. The total amount of space required is approximately 2.0 Mbytes. The following tree shows the directory structure supplied in the tar file, and the approximate size of each subdirectory:

RTE (1800 Kbytes)		
src.d (310 Kbytes)	test.d (1440 Kbytes)	man.d (50 Kbytes)

The following command will unpack the tar file and create the RTE directory structure:

```
prompt> tar xvf rte.tar
```

5.2. Machine Dependencies

The file RTE/src.d/InitCon_xxx.f contains machine-dependent variables. The following value must be modified so it can perform correctly on your machine:

expmax: maximum absolute value for exponential function
 argument used to prevent underflow and overflow
 in the RTE calculations.

5.3. Creating The RTE Library

The RTE library is created in the file RTE/src.d/libRTE.a. This file will need to be copied into /usr/local/lib/libRTE.a to allow for global access. All users can then link with the RTE library by using the -lRTE option at compile time. For more information see the manual pages on the f77 command.

The following commands will compile and build the RTE library:

```
prompt> cd RTE/src.d
prompt> make
```


5.4. Installing and Running Sample Programs

Three sample programs have been provided in the RTE directory structure for testing the library on your machine. The following commands will activate these programs:

```
prompt> cd RTE/test.d/tbclear.d/  
prompt> make  
prompt> tbclear.scr
```

```
prompt> cd RTE/test.d/tbmodel.d/  
prompt> make  
prompt> tbmodel.scr
```

```
prompt> cd RTE/test.d/wgttest.d/  
prompt> make  
prompt> wgttest.scr
```

Compare the results in files `tbclear.result`, `tbmodel.result`, and `wgttest.result` with those in appendices A, B, and C, respectively.

5.5. Installing The UNIX On-line Manual Pages

The on-line manual pages are provided in the NROFF format. They can be installed with the following commands:

```
prompt> cd RTE/man.d  
prompt> cp *.1 /usr/man/local/
```

The users can access these on-line manual pages with the following commands:

```
prompt> man RTE  
prompt> man ViewConst_RTE  
prompt> man ModifyProf_RTE  
prompt> man CldModel91_RTE  
prompt> man TbZenith_RTE  
prompt> man TbScan_RTE  
prompt> man TbCloud_RTE  
prompt> man Radiance_RTE  
prompt> man Weight_RTE
```

NOTE: The `man` command is not case sensitive.

5.6. Updating RTE

An informational file, RTE/src.d/README, has been included with the RTE source code. This file contains programming techniques and suggestions that will be useful when updating the existing library source code or adding additional user-callable subroutines.

ACKNOWLEDGMENTS

We thank B.B. Stankov and E.R. Westwater for their patience and cooperation throughout this project.

REFERENCES

- Crosby, D.S., H.E. Fleming, and D.Q. Wark, 1973. Covariance Matrices And Means Of Atmospheric Planck Function Profiles For Application To Temperature Sounding From Satellite Measurements. J. Atmos. Sci., 30, 141-144.
- Decker, M.T., E.R. Westwater, and F.O. Guiraud, 1978. Experimental Evaluation Of Ground-Based Microwave Radiometric Remote Sensing Of Atmospheric Temperature And Water Vapor Profiles. J. Appl. Meteorol., 17, 1788-1795.
- Hulst, H.C. van de, 1981. Light Scattering By Small Particles. Dover Publications, New York, 470 pp.
- Liebe, H.J., and D.H. Layton, 1987. Millimeter-wave Properties Of The Atmosphere. National Telecommunications And Information Administration, Boulder, Co, 74 pp.
- Schroeder, J.A., and E.R. Westwater, 1991. Users' Guide To WPL Microwave Radiative Transfer Software. NOAA Wave Propagation Laboratory, Boulder, Co, 84 pp.
- Schroeder, J.A., and E.R. Westwater, 1992. Guide To Passive Microwave Weighting Function Calculations. NOAA Wave Propagation Laboratory, Boulder, Co, 18 pp.
- Ulaby, F.T., R.K. Moore, and A.K. Fung, 1981. Microwave Remote Sensing, Active And Passive, VOLUME 1. Addison-Wesley, Reading, MA, 456 pp.
- Westwater, E.R., 1974. Microwave Emission From Clouds. NOAA Wave Propagation Laboratory, Boulder, CO, 43 pp.

APPENDIX A: PROGRAM TbClear

	PAGE
SOURCE CODE	33
SAMPLE USER SPECIFICATIONS	42
SAMPLE DATA	43
SAMPLE RESULTS	44

TbClear: SOURCE CODE

```
C.....
c Program TbClear computes brightness temperatures for clear-sky
c conditions only. This program uses the same specFILE format as
c needed for cloudy conditions. Therefore, some of the user
c specifications may not apply.
C
c USAGE:  prompt> tbclear.exe specFILE dataFILE resultFILE
C
C
C SUBROUTINES CALLED FROM HERE:
C
c ReadTbSpec
c   USER routine which inputs user specifications;
c   user can modify this routine;
c   user can use some other means for getting these values
c ReadData
c   USER routine which reads Decker formatted radiosonde data
c   with headers
c   user can modify this routine;
c   user can substitute a different read routine here;
c ModifyProf_RTE
c   RTE routine which modifies profiles to the user specifications;
c   user can eliminate if raw profiles meet criteria of the other
c   RTE subroutines
c TbZenith_RTE
c   RTE routine which computes brightness temperatures for
c   zenith-pointing antenna only
c TbScan_RTE
c   RTE routine which computes brightness temperatures for
c   elevation angle-scanning antenna
C .....

      program TbClear

c      ... force all variables to be declared ...
      implicit none

c      ... parameter variables ...
      integer SIZE, MAXA, MAXF, errUNIT
      parameter (SIZE=2000, MAXA=50, MAXF=50)
      parameter (errUNIT = 6)

c      ... local variables ...
      real hum, raoblat,z(SIZE),p(SIZE),tk(SIZE),
&         dew(SIZE),rh(SIZE),angles(MAXA),frq(MAXF),
&         zstart,maxdp,cldrh,cldthick,pmax,
&         tbttotal_1d(MAXF), tauclr_1d(MAXF),tmr_1d(MAXF),
&         tbttotal_2d(MAXF,MAXA),tauclr_2d(MAXF,MAXA),
&         tmr_2d(MAXF,MAXA)
```


TbClear: SOURCE CODE

```
integer i,j,k, date(4),time(3),
&      dataUNIT,specUNIT,outUNIT,cldcnt,
&      nl,nang,nfrq,extrap50,extrap1,ice,
&      status,printprof,nlay,nmodels,iargc
character title*70,specFILE*15,dataFILE*40,outFILE*15

c      ... command line arguments ...
      if (iargc() .ne. 3) then
&        write(errUNIT,*) 'USAGE:  prompt> tbclear.exe ',
&                          'specFILE dataFILE resultFILE'
        goto 333
      endif
      call getarg(1,specFILE)
      call getarg(2,dataFILE)
      call getarg(3,outFILE)

c      ... initialize ...
      specUNIT = 55
      dataUNIT = 66
      outUNIT = 77
      nlay = 0
      nmodels = 0
      status = 0

c      ... open I/O files ...
      open(specUNIT, file=specFILE, status='OLD', err=995)
      open(dataUNIT, file=dataFILE, status='OLD', err=996)
      open(outUNIT, file=outFILE, status='UNKNOWN', err=997)

c      ... read user specs ...
      call ReadTbSpecs(specUNIT,title,printprof,nfrq,frq,
&      nang,angles,extrap50,extrap1,zstart,maxdp,
&      raoblat,pmax,ice,cldrh,cldcnt,cldthick,status)
      if (status .lt. 0) then
&        write (errUNIT, 2000) specFILE
        goto 333
      endif

c      while (not EOF) loop
22      continue

c      ... read raw data ...
      call ReadData(dataUNIT,date,time,nl,p,z,tk,dew,status)
      if (status .lt. 0) then
&        write (errUNIT, 2000) dataFILE
        goto 333
      else if (status .gt. 0) then
&        goto 333
      endif
```


TbClear: SOURCE CODE

```
c      ... check if profiles reach necessary pressure height ...
      if (p(nl) .gt. pmax) then
        write (errUNIT, 4000) p(nl), pmax
        goto 333
      endif

c      ... Convert units for input profile ...
      do 13 i = 1, nl
        rh(i) = .01 * (hum(tk(i), (tk(i)-dew(i))))
        tk(i) = tk(i) + 273.16
        z(i) = .001 * z(i)
13    continue

c      ... Modify profile to user specifications ...
      call ModifyProf_RTE (nl, z, p, tk, rh, extrap50, extrap1,
&        zstart, maxdp, raoblat, ice, nl, z, p, tk, rh, status)
      if (status .lt. 0) goto 333

c      ***** output *****
c      ... write user specs ...
      write (outUNIT, 6600) title, pmax, maxdp, ice,
&        cldcnt, cldrh, cldthick

c      ... write header information ...
      write (outUNIT, 1100) date(3), date(4), date(2), time(1),
&        time(2), raoblat, z(1), nlay, printprof,
&        nl, nmodels, nang, nfrq

c      ... write modified profiles ....
      write(outUNIT, *) '      height      press
&        'temp', '      rh'
      do 300 i = 1, nl
        write(outUNIT, *) z(i), p(i), tk(i), rh(i)
300    continue
      write(outUNIT, *)
c      *****

c      ... compute RTE values for a fixed zenith-pointing antenna ...
      call TbZenith_RTE (nl, z, p, tk, rh, nfrq, frq, ice,
&        tbttotal_ld, tmr_ld, tauclr_ld, status)
      if (status .lt. 0) goto 333
```


TbClear: SOURCE CODE

```
c      ***** output *****
write(outUNIT, *) 'TbZenith_RTE:'
write(outUNIT, *) '      angle      freq      tbttotal',
&      '      tmr      tauclr'
do 70 j = 1, nfrq
    write(outUNIT, *) 90.0, frq(j), tbttotal_1d(j),
&      tmr_1d(j), tauclr_1d(j)
70    continue
write(outUNIT, *)
c      *****

c      ... compute RTE values for multiple angles ...
call TbScan_RTE (nl,z,p,tk,rh,nfrq,frq,nang,angles,ice,
1      tbttotal_2d,tmr_2d,taucclr_2d,status)
    if (status .lt. 0) goto 333

c      ***** output *****
write(outUNIT, *) 'TbScan_RTE:'
write(outUNIT, *) '      angle      freq      tbttotal',
&      '      tmr      tauclr'
do 100 k = 1, nang
    do 200 j = 1, nfrq
        write(outUNIT, *) angles(k), frq(j), tbttotal_2d(j,k),
&      tmr_2d(j,k), tauclr_2d(j,k)
200    continue
100    continue
write(outUNIT, *)
c      *****

c      goto 22
c      end loop

c      ... close I/O files ...
333    continue
close(specUNIT)
close(dataUNIT)
close(outUNIT)
stop

995    continue
write(errUNIT,1000) specFILE
goto 333

996    continue
write(errUNIT,1000) dataFILE
goto 333
```


TbClear: SOURCE CODE

```
997  continue
    write(errUNIT,1000) outFile
    goto 333

c    ... local formats ...
1000    format('MAIN:  ERROR OPENING FILE ',a)
2000    format('MAIN:  ERROR READING FILE ',a)
4000    format('MAIN:  profile terminates at',f7.1,' mb. user ',
&        'requested termination at or above', f7.1, ' mb.')
1100    format (3x,3i3,3x,i2,f10.2,f10.3,2i6,i7,3i6)
6600    format (1x,'Title: ',a70, '//,
0        1x,'The results in this file reflect the following user ',
1        'specifications:', '//,
2        1x,'Discard profiles that terminate below this pressure ',
3        'level (mb):',f13.0, '//,
4        1x,'Insert level if pressure difference between levels ',
5        'exceeds (mb):',f12.0, '//,
6        1x,'(zero allows the pressure difference to increase ',
7        'with height.)', '//,
8        1x,'Calculate saturation vapor pressure over water (0) ',
9        'or water/ice (1):', i7, '//,
0        1x,'Maximum number of cloud layers to model per ',
1        'profile:', i23, '//,
2        1x,'Relative humidity threshold for cloud detection ',
3        '(fraction):', f16.2, '//,
4        1x,'Minimum acceptable cloud thickness (km):', f35.3, '/')

    end

c .....
    real function hum(t,td)
        real t, td, esat

        hum= 100.*(esat(td)/esat(t))

        return
    end
```


TbClear: SOURCE CODE

```
C .....
  real function esat(t)
    real t, tk, p1, p2, c1

    tk = t+273.16
    p1 = 11.344-0.0303998*tk
    p2 = 3.49149-1302.8844/tk
    c1 = 23.832241-5.02808*alog10(tk)

    esat = 10.0**(c1-1.3816e-7*
&          10.0**p1+8.1328e-3*
&          10.0**p2-2949.076/tk)
    return
  end

C .....
C Subroutine ReadTbSpecs reads the user specifications from an
C open file corresponding to inUNIT.
C
C   ierror - status of execution
C           ierror < 0: implies execution incomplete
C           ierror = 0: implies execution complete
C .....

  subroutine ReadTbSpecs(inUNIT,title,printprof,nfrq,frq,
&    nang,angles,extrap50,extrap1,zstart,maxdp,raoblat,
&    pmax,ice,cldrh,cldmax,cldthick,ierror)

C   ... force all variables to be declared ...
  implicit none

C   ... parameter variables ...
  integer maxf, maxa
  parameter (maxf=10, maxa=10)

C   ... argument list output variables ...
  character title*70
  real frq(maxf),angles(maxa),zstart,maxdp,raoblat,
&    cldrh,cldthick,pmax
  integer inUNIT,printprof,nfrq,nang,extrap50,extrap1,
&    ice,cldmax,ierror

C   ... local variables ...
  integer i, j, k
```


TbClear: SOURCE CODE

```
c    ... input title ...
    read (inUNIT,'(a)',end=999,err=999)
    read (inUNIT,'(a)',end=999,err=999) title

c    ... input switch to write modified profiles (1=yes; 0=no).
    read (inUNIT,1200,end=999,err=999)
    read (inUNIT,*,end=999,err=999) printprof

c    ... input number of frequencies ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) nfrq

c    ... input channel frequencies ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) (frq(i),i=1,nfrq)

c    input elevation angles of interest (degrees, maximum of 10).
c    ... input number of angles ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) nang

c    ... input elevation angles ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) (angles(i),i=1,nang)

c    ... input switch for extrapolation up to 50.0 mb. (1=yes, 0=no).
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) extrapol50

c    ... input switch for extrapolation up to 0.10 mb. (1=yes, 0=no).
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) extrapol1

c    ... input simulated antenna elevation (km MSL) ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) zstart

c    ... input latitude of radiosonde launch site (degrees) ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) raoblat

c    ... input maximum acceptable pressure at highest level of
&    profile (mb) ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) pmax
```


TbClear: SOURCE CODE

```
c    ... input max acceptable press. difference between adjacent
&    levels ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) maxdp

c    ... input switch to select saturation vapor pressure
&    calculation ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) ice

c    ... input maximum number of cloud layers to model ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) cldmax

c    ... input relative humidity threshold for cloud detection
&    (fraction) ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) cldrh

c    ... input minimum acceptable cloud layer thickness (km) ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) cldthick

c    ... exit gracefully, no errors ...
888  continue
    ierror = 0
    return

c    ... exit with error ...
999  continue
    ierror = -1
    return

c    ... local formats ...
1100 format (//)
1200 format (//)

    end
```


TbClear: SOURCE CODE

```
c ... jcr .....
c Subroutine ReadData reads Decker formatted radiosonde data with
c headers. The data is read from an open file corresponding to inUNIT
c
c   ierror    - status of execution
c               ierror < 0: implies execution incomplete
c               ierror = 0: implies execution complete
c .....
c       subroutine ReadData(iunit, date, time, npts,
c           &               press, hgt, temp, dew, ierror)

c   ... force all variables to be declared ...
c   implicit none

c   ... argument list input variables ...
c   integer iunit

c   ... argument list output variables ...
c   integer date(4), time(3), npts, ierror
c   real press(*), hgt(*), temp(*), dew(*)

c   ... local variables ...
c   integer i

c   ... local formats ...
200  format(1x, 3(i2), 1x, 2(i2), 6x, i4)

c   ... read header information ...
c       read(iunit,200,err=999,end=888) date(2),date(3),date(4),
c       &               time(1),time(2),npts
c   ... read data ...
c       read(iunit,*,err=999,end=999)
c       &       (press(i),hgt(i),temp(i),dew(i),i=1,npts)

c   ... exit, no error, good read ...
777  continue
c       ierror = 0
c       return

c   ... exit no read, EOF...
888  continue
c       ierror = 1
c       return

c   ... exit with error ...
999  continue
c       ierror = -1
c       return
c   end
```


TbClear: SAMPLE USER SPECIFICATIONS

Specify Title:

Test RTE using TbClear

Specify whether (1) or not (0) to output modified profiles
(z,p,tk,rh,denice,denliq):

1

Specify number of frequencies ($\leq$ maxf):

2

Specify frequencies (GHz):

20.6 31.65

Specify number of elevation angles ($\leq$ maxa):

3

Specify elevation angles (degrees):

90.0 19.47122 9.5941

Specify whether (1) or not (0) to extrapolate up to 50.0 mb.

1

Specify whether (1) or not (0) to extrapolate up to 0.1 mb.

1

Specify simulated antenna height (km MSL):

0.0

Specify latitude of radiosonde launch site (degrees):

39.0

Specify maximum acceptable pressure at highest level of input profile (mb):

300.0

Specify maximum acceptable pressure difference between adjacent levels (mb):

0.0

Select saturation vapor pressure calculation over WATER (0) or WATER/ICE

(1):

0

Specify the maximum number of cloud layers ($\leq$ maxc):

4

Specify relative humidity threshold for cloud detection (fraction):

0.90

Specify the minimum acceptable cloud layer thickness (km):

0.001

TbClear: SAMPLE DATA

900313 1100 72469 44
829.9 1610 -1.0 0.4 826.8 1640 -1.1 4.7 817.9 1727 -1.9 1.4
750.0 2410 -6.8 1.3 706.4 2876 -9.8 1.4 700.0 2946 -9.5 1.3
690.1 3056 -9.4 1.3 680.8 3161 -9.8 1.3 659.6 3406 -8.3 1.3
629.7 3765 -10.0 1.4 525.7 5130 -20.6 3.0 500.0 5499 -23.3 5.7
452.5 6221 -29.4 4.5 436.1 6483 -32.2 16.6 400.0 7086 -37.1 15.8
372.4 7578 -39.8 16.7 300.0 9022 -50.1 30.0 250.0 10205 -53.3 30.0
244.2 10356 -54.0 30.0 214.7 11190 -49.9 30.0 203.0 11556 -50.6 30.0
200.0 11653 -49.3 30.0 189.8 11996 -50.2 30.0 175.2 12521 -47.4 30.0
160.5 13102 -46.5 30.0 150.0 13550 -47.9 30.0 137.1 14140 -50.3 30.0
121.9 14905 -51.4 30.0 113.5 15366 -54.2 30.0 100.0 16178 -54.1 30.0
83.6 17320 -56.8 30.0 79.6 17632 -54.5 30.0 75.3 17986 -56.5 30.0
70.0 18451 -54.8 30.0 55.9 19869 -61.0 30.0 50.0 20567 -58.1 30.0
43.5 21437 -61.5 30.0 41.6 21714 -61.0 30.0 40.1 21943 -58.8 30.0
30.0 23762 -59.5 30.0 24.3 25079 -59.9 30.0 20.0 26301 -58.1 30.0
16.4 27554 -56.8 30.0 16.3 27593 -56.8 30.0

TbClear: SAMPLE RESULTS

Title: RUN TbClear USING tb.0.spec.

The results in this file reflect the following user specifications:

Discard profiles that terminate below this pressure level (mb): 300.
Insert level if pressure difference between levels exceeds (mb): 0.
(zero allows the pressure difference to increase with height.)
Calculate saturation vapor pressure over water (0) or water/ice (1): 0
Maximum number of cloud layers to model per profile: 4
Relative humidity threshold for cloud detection (fraction): 0.900
Minimum acceptable cloud thickness (km): 0.001

3	13	90	11	0	39.00	1.610	0	1	71	0	3	2
height			press			temp			rh			
1.610000			829.9000			272.1600			0.9710695			
1.640000			826.8000			272.0600			0.7036579			
1.683379			822.3500			271.6611			0.8021936			
1.727000			817.9000			271.2600			0.9012780			
1.821021			808.2000			270.5855			0.9017023			
1.916178			798.5000			269.9028			0.9021319			
2.012499			788.8000			269.2118			0.9025666			
2.110011			779.1000			268.5122			0.9030067			
2.208745			769.4000			267.8038			0.9034524			
2.308730			759.7000			267.0865			0.9039037			
2.410000			750.0000			266.3600			0.9043608			
2.562253			735.4667			265.3799			0.9012486			
2.717544			720.9333			264.3801			0.8980743			
2.876000			706.4000			263.3600			0.8948354			
2.946000			700.0000			263.6600			0.9022502			
3.056000			690.1000			263.7600			0.9023315			
3.161000			680.8000			263.3600			0.9020087			
3.282533			670.2000			264.1041			0.9025946			
3.406000			659.6000			264.8600			0.9031899			
3.583417			644.6500			264.0199			0.8989805			
3.765000			629.7000			263.1600			0.8946723			
4.193194			595.0333			259.8348			0.8550419			
4.647100			560.3666			256.3100			0.8130317			
5.130000			525.7000			252.5600			0.7683381			
5.499000			500.0000			249.8600			0.5948852			
5.850993			476.2500			246.8861			0.6215857			
6.221000			452.5000			243.7600			0.6496526			
6.483000			436.1000			240.9600			0.1766467			
7.086000			400.0000			236.0600			0.1782228			
7.578001			372.4000			233.3600			0.1522539			
8.261056			336.2000			228.4878			8.8169783E-02			
9.022000			300.0000			223.0600			1.6778065E-02			

TbClear: SAMPLE RESULTS

10.20500	250.0000	219.8600	1.4307737E-02
10.35600	244.2000	219.1600	1.3796306E-02
11.19000	214.7000	223.2600	1.6939625E-02
11.55600	203.0000	222.5600	1.6377719E-02
11.65300	200.0000	223.8600	1.7429629E-02
11.99600	189.8000	222.9600	1.6697573E-02
12.52100	175.2000	225.7600	1.9031974E-02
13.10200	160.5000	226.6600	1.9817704E-02
13.55000	150.0000	225.2600	1.8602785E-02
14.14000	137.1000	222.8600	1.6617248E-02
14.90500	121.9000	221.7600	1.5748283E-02
15.36600	113.5000	218.9600	1.3652073E-02
16.17800	100.0000	219.0600	1.3724091E-02
17.32000	83.60000	216.3600	1.1853845E-02
17.63200	79.60000	218.6600	1.3437333E-02
17.98600	75.30000	216.6600	1.2054048E-02
18.45100	70.00000	218.3600	1.3224459E-02
19.86900	55.90000	212.1600	9.2476234E-03
20.56700	50.00000	215.0600	1.1007915E-02
21.43700	43.50000	211.6600	8.9617064E-03
21.71400	41.60000	212.1600	9.2476234E-03
21.94300	40.10000	214.3600	1.0566950E-02
23.76200	30.00000	213.6600	1.0136342E-02
25.07900	24.30000	213.2600	9.8948013E-03
26.30100	20.00000	215.0600	1.1007915E-02
27.55400	16.40000	216.3600	1.1853845E-02
27.59300	16.30000	216.3600	1.1853845E-02
28.12654	15.00000	222.2727	4.2025759E-04
30.78637	10.00000	225.9451	1.8464633E-04
33.17481	7.000000	231.6162	6.9922979E-05
35.48934	5.000000	238.4235	2.4953777E-05
37.06348	4.000000	243.5894	1.2138637E-05
39.14293	3.000000	250.3215	4.9280125E-06
42.16786	2.000000	259.4871	1.5084656E-06
44.36541	1.500000	262.4385	8.9201410E-07
47.50459	1.000000	266.5551	4.3111501E-07
52.88485	0.5000000	263.7892	2.6722543E-07
59.80841	0.2000000	252.5951	2.6924883E-07
64.83220	0.1000000	242.7029	3.2995172E-07

TbZenith_RTE:

angle	freq	tbttotal	tmr	tauc1r
90.00000	20.60000	13.51007	260.4313	4.2564034E-02
90.00000	31.65000	10.77985	255.6985	3.2057259E-02

TbScan_RTE:

angle	freq	tbttotal	tmr	tauc1r
90.00000	20.60000	13.51007	260.4313	4.2564034E-02
90.00000	31.65000	10.77985	255.6985	3.2057259E-02

TbClear: SAMPLE RESULTS

19.47122	20.60000	33.62573	260.6841	0.1273916
19.47122	31.65000	25.95379	256.0100	9.5849395E-02
9.594100	20.60000	60.50455	261.1011	0.2529162
9.594100	31.65000	46.66248	256.5429	0.1897235

APPENDIX B: PROGRAM TbModel

	PAGE
SOURCE CODE	48
SAMPLE USER SPECIFICATIONS	58
SAMPLE DATA	59
SUMMARY OF SAMPLE RESULTS	60
SAMPLE RESULTS	62

TbModel: SOURCE CODE

```
C .....
C Program which computes brightness temperatures for clear-sky and
C cloudy-sky conditions.
C
C USAGE:  prompt> tbmodel.exe specFILE dataFILE resultFILE
C
C SUBROUTINES CALLED FROM HERE:
C
C ReadTbSpec
C   USER subroutine which inputs user specifications;
C   user can modify this routine;
C   user can use some other means for getting these values
C ReadDecMet
C   USER subroutine which reads Decker formatted radiosonde data
C   user can modify this routine;
C   user can substitute a different read routine here;
C ModifyProf_RTE
C   RTE subroutine which modifies profiles to the user specifications;
C   user can eliminate if raw profiles meet criteria of the other
C   RTE subroutines
C CldModel91_RTE
C   RTE subroutine which performs cloud modeling;
C   (modifies profiles where necessary)
C   user can eliminate for clear sky applications;
C   users own cloud modeling routine can be substituted here
C TbCloud_RTE
C   RTE subroutine which computes brightness temperatures and other
C   related values given clear or cloudy conditions for an
C   elevation angle-scanning antenna
C .....

      program tbmodel

C   ... force all variables to be declared ...
      implicit none

C   ... parameter variables ...
      integer MAXL, MAXA, MAXF, MAXC, errUNIT
      parameter (MAXL=2000, MAXA=50, MAXF=50, MAXC=10)
      parameter (errUNIT = 0)

C   ... local variables ...
      real
      &      cldbase(MAXC), cldtop(MAXC),
      &      denfrac(3), denliq(MAXL), denice(MAXL),
      &      Mdenliq(MAXL), Mdenice(MAXL),
      &      tbttotal(MAXF, MAXA), tbatm(MAXF, MAXA),
      &      tmr(MAXF, MAXA), tmrcld(MAXF, MAXA),
```


TbModel: SOURCE CODE

```
&      tauwet (MAXF,MAXA),taudry (MAXF,MAXA);
&      tauhq (MAXF,MAXA),tauice (MAXF,MAXA),
&      srho (MAXA),swet (MAXA),sdry (MAXA),slhq (MAXA),
&      sice (MAXA)

      real hum, raoblat,raobhgt,z (MAXL),p (MAXL),tk (MAXL),
&      dew (MAXL),rh (MAXL),angles (MAXA),frq (MAXF),
&      zstart,pmax,maxdp,cldrh,cldthick

      integer i,j,k,iang,ifrq,imod,date(4),time(3),iunit,
&      dataUNIT,specUNIT,outUNIT,cldmax,
&      nl,nang,nfrq,extrap50,extrapl,ice,
&      nlay,nmodels,status,printprof,
&      iargc

      character title*70,specFILE*15,dataFILE*40,outFILE*15

C      ... command line arguments ...
      if (iargc() .ne. 3) then
&         write(errUNIT,*) 'USAGE:  prompt> tbmodel.exe ',
&         'specFILE dataFILE resultFILE'
         goto 333
      endif
      call getarg(1, specFILE)
      call getarg(2, dataFILE)
      call getarg(3, outFILE)

C      ... initialize ...
      status = 0
      specUNIT = 55
C      specFILE = 'tb.spec'
      dataUNIT = 66
C      dataFILE = 'dec_met.dat'
      outUNIT = 77
C      outFILE = 'results.dat'

C      ... open I/O files ...
      open(specUNIT, file=specFILE, status='OLD', err=995)
      open(dataUNIT, file=dataFILE, status='OLD', err=996)
      open(outUNIT, file=outFILE, status='UNKNOWN', err=998)

C      ... read user specs ...
      call ReadTbSpecs(specUNIT,title,printprof,nfrq,frq,
&         nang,angles,extrap50,extrapl,zstart,maxdp,
&         raoblat,pmax,ice,cldrh,cldmax,cldthick,status)
```


TbModel: SOURCE CODE

```

    if (status .eq. -1) then
        write (errUNIT, 2000) specFILE
        goto 333
    endif

c      ***** output *****
c      ... write user title
        write (outUNIT,6600) title

c      ... write user specs ...
        write (outUNIT,8800) pmax,maxdp,ice,
&          cldmax,cldrh,cldthick

c      ... write format summary ...

c      *****

c      while (not EOF) loop
22      continue

c      ... read raw data ...
        call readdecmet(dataUNIT,date,time,nl,p,z,tk,dew,status)
        if (status .eq. -1) then
            write (errUNIT, 2000) dataFILE
            goto 333
        else if (status .eq. 1) then
            goto 333
        endif

c      ... check if profiles reach necessary pressure height ...
        if (p(nl) .gt. pmax) then
            write (errUNIT, 4000) p(nl), pmax
            goto 333
        endif

c      ... Convert units for input profiles ...
        do 13 i = 1, nl
            rh(i) = .01 * (hum(tk(i),(tk(i)-dew(i))))
            tk(i) = tk(i) + 273.16
            z(i) = .001 * z(i)
13      continue
        raobhgt = z(1)

c      ... Modify profile to user specifications ...
        call ModifyProf_RTE (nl,z,p,tk,rh,extrap50,extrap1,
&          zstart,maxdp,raoblat,ice,nl,z,p,tk,rh,status)
        if (status .lt. 0) goto 333
```


TbModel: SOURCE CODE

```
C      ... perform cloud modeling (also modifies profiles) ...
      call CldModel91_RTE (nl,z,p,tk,rh,nang,angles,
2         ice,cldrh,cldthick,cldmax,
4         nl,z,p,tk,rh,denliq,denice,denfrac,
3         nlay,cldtop,cldbase,status)
      if (status .lt. 0) goto 333
      if (nlay .le. 0) then
        nmodels = 0
      else
        nmodels = 3
      endif

C      ***** output *****
C      ... write header information ...
      write (outUNIT,1100) date(3),date(4),date(2),
&         time(1),time(2),raoblat,raobhgt,nlay,printprof,
&         nl,nmodels,nang,nfrq

C      ... write cloud layers ...
&      if (nlay .gt. 0)
&         write (outUNIT,2200) (cldbase(i),cldtop(i),i=1,nlay)

C      ... write modified profiles ...
      if (printprof .eq. 1) then
        write(outUNIT, *)
&         '          height      press      temp      rh
&         'denliq      denice'
        write(outUNIT,3300) (z(i),p(i),
&         tk(i),rh(i),denliq(i),denice(i), i=1,nl)
      endif

C      *****
C      ... calculate RTE values for clear conditions ...
      do 74 i = 1, nl
        Mdenliq(i) = 0.0
        Mdenice(i) = 0.0
74      continue

      call TbCloud_RTE (nl,z,p,tk,rh,Mdenliq,Mdenice,
&         nfrq,frq,nang,angles,0,cldbase,cldtop,ice,
&         tbttotal,tbatm,tmr,tmrcld,
&         tauwet,taudry,tauliq,tauice,
&         srho,swet,sdry,sliq,sice,status)
      if (status .lt. 0) goto 333
```


TbModel: SOURCE CODE

```
c      ***** output *****
c      ... write TBMODEL values for clear conditions ...
      do 212 iang = 1, nang
        write(outUNIT,4400) angles(iang),srho(iang),
&          swet(iang),sdry(iang)
      do 223 ifrq = 1, nfrq
        write(outUNIT,5500) frq(ifrq),tbttotal(ifrq,iang),
&          tbatm(ifrq,iang),tmr(ifrq,iang),
&          tmrcld(ifrq,iang),tauwet(ifrq,iang),
&          taudry(ifrq,iang)
223      continue
212      continue
c      *****
c      ... handle each cloud model ...
      do 200 imod = 1, nmodels
c      ... store model specific values ...
      do 100 i = 1, nl
        Mdenliq(i) = denliq(i) * denfrac(imod)
        Mdenice(i) = denice(i) * denfrac(imod)
100      continue
c      ... calculate RTE values for current model ...
      call TbCloud_RTE (nl,z,p,tk,rh,Mdenliq,Mdenice,
&          nfrq,frq,nang,angles,nlay,cldbase,cldtop,ice,
&          tbttotal,tbatm,tmr,tmrcld,
&          tauwet,taudry,tauliq,tauice,
&          srho,swet,sdry,sliq,sice,status)
      if (status .lt. 0) goto 333
c      ***** output *****
c      ... write TBMODEL values for current model ...
      do 211 iang = 1, nang
        write(outUNIT,4400) angles(iang),sliq(iang),
&          sice(iang),denfrac(imod)
      do 222 ifrq = 1, nfrq
        write(outUNIT,5500) frq(ifrq),tbttotal(ifrq,iang),
&          tbatm(ifrq,iang),tmr(ifrq,iang),
&          tmrcld(ifrq,iang),tauliq(ifrq,iang),
&          tauice(ifrq,iang)
222      continue
211      continue
c      *****
200      continue
      write(outUNIT, *)
      goto 22
c      end loop
```


TbModel: SOURCE CODE

```
c    ... close I/O files ...
333  continue
      close(specUNIT)
      close(dataUNIT)
      close(outUNIT)

      stop

995  continue
      write(errUNIT,1000) specFILE
      goto 333

996  continue
      write(errUNIT,1000) dataFILE
      goto 333

998  continue
      write(errUNIT,1000) outFile
      goto 333

c    ... local formats ...
1000  format('MAIN:  ERROR OPENING FILE ',a)
2000  format('MAIN:  ERROR READING FILE ',a)
3000  format('MAIN:  END OF FILE ',a)
4000  format('MAIN:  profile terminates at',f7.1,' mb. user ',
&      'requested termination at or above', f7.1, ' mb.')
```

```
1100  format (3x,3i3,3x,2i2,f10.2,f10.3,2i6,i7,3i6)
2200  format (2x,2f8.3,3(4x,2f8.3))
3300  format (5x,f10.3,2f10.2,3e11.3)
4400  format (10x,4e13.5)
5500  format (3x,f8.3,4f10.3,2e13.5)
6600  format (1x,'Title: ',a70,/)

8800  format (1x,'The results in this file refl',
1      'ect the following user specifications:',/,/,1x,'Discard profi',
2      'les that terminate below this pressure level (mb):',f12.1,/,/,1x,
3      'Insert level if pressure difference between levels exceeds (',
4      'mb):',f11.1,/,/,1x,'(zero allows the pressure difference to in',
5      'crease with height.)',/,/,1x,'Calculate saturation vapor press',
6      'ure over water (0) or water/ice (1):',i7,/,/,1x,'Maximum numb',
7      'er of cloud layers to model per profile:',i23,/,/,1x,'Relative',
8      ' humidity threshold for cloud detection (fraction):',f16.2,/,/,
9      'Minimum acceptable cloud thickness (km):',f35.3,/,/)
```

end

TbModel: SOURCE CODE

```
C .....
  real function hum(t,td)
    real t, td, esat

    hum= 100.*(esat(td)/esat(t))
    return

  end

C .....
  real function esat(t)
    real t, tk, p1, p2, c1

    tk = t+273.16
    p1 = 11.344-0.0303998*tk
    p2 = 3.49149-1302.8844/tk
    c1 = 23.832241-5.02808*alog10(tk)

    esat = 10.0**(c1-1.3816e-7*
&          10.0**p1+8.1328e-3*
&          10.0**p2-2949.076/tk)
    return
  end

C .....
C Subroutine ReadTbSpecs reads the user specifications from an
C open file corresponding to inUNIT.
C
C   ierror - status of execution
C           ierror < 0: implies execution incomplete
C           ierror = 0: implies execution complete
C .....

  subroutine ReadTbSpecs(inUNIT,title,printprof,nfrq,frq,
&    nang,angles,extrap50,extrap1,zstart,maxdp,raoblat,
&    pmax,ice,cldrh,cldmax,cldthick,ierror)

C   ... force all variables to be declared ...
  implicit none

C   ... parameter variables ...
  integer maxf, maxa
  parameter (maxf=10, maxa=10)
```


TbModel: SOURCE CODE

```
C    ... argument list output variables ...
    character title*70
    real frq(maxf),angles(maxa),zstart,maxdp,raoblat,
&      cldrh,cldthick,pmax
    integer inUNIT,printprof,nfrq,nang,extrap50,extrap1,
&      ice,cldmax,ierror

C    ... local variables ...
    integer i, j, k

C    ... input title ...
    read (inUNIT,'(a)',end=999,err=999)
    read (inUNIT,'(a)',end=999,err=999) title

C    ... input switch to write modified profiles (1=yes; 0=no).
    read (inUNIT,1200,end=999,err=999)
    read (inUNIT,*,end=999,err=999) printprof

C    ... input number of frequencies ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) nfrq

C    ... input channel frequencies ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) (frq(i),i=1,nfrq)

C    input elevation angles of interest (degrees, maximum of 10).
C    ... input number of angles ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) nang

C    ... input elevation angles ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) (angles(i),i=1,nang)

C    ... input switch for extrapolation up to 50.0 mb. (1=yes, 0=no).
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) extrap50

C    ... input switch for extrapolation up to 0.10 mb. (1=yes, 0=no).
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) extrap1

C    ... input simulated antenna elevation (km MSL) ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) zstart
```


TbModel: SOURCE CODE

```
c    ... input latitude of radiosonde launch site (degrees) ...
      read (inUNIT,1100,end=999,err=999)
      read (inUNIT,*,end=999,err=999) raoblat

c    ... input maximum acceptable pressure at highest level of
&    profile (mb) ...
      read (inUNIT,1100,end=999,err=999)
      read (inUNIT,*,end=999,err=999) pmax

c    ... input max acceptable press. difference between adjacent
&    levels ...
      read (inUNIT,1100,end=999,err=999)
      read (inUNIT,*,end=999,err=999) maxdp

c    ... input switch to select saturation vapor pressure
&    calculation ...
      read (inUNIT,1100,end=999,err=999)
      read (inUNIT,*,end=999,err=999) ice

c    ... input maximum number of cloud layers to model ...
      read (inUNIT,1100,end=999,err=999)
      read (inUNIT,*,end=999,err=999) cldmax

c    ... input relative humidity threshold for cloud detection
&    (fraction) ...
      read (inUNIT,1100,end=999,err=999)
      read (inUNIT,*,end=999,err=999) cldrh

c    ... input minimum acceptable cloud layer thickness (km) ...
      read (inUNIT,1100,end=999,err=999)
      read (inUNIT,*,end=999,err=999) cldthick

c    ... exit gracefully, no errors ...
888  continue
      ierror = 0
      return

c    ... exit with error ...
999  continue
      ierror = -1
      return

c    ... local formats ...
1100  format (//)
1200  format (//)

      end
```


TbModel: SOURCE CODE

```
c ... jcr .....
c Subroutine ReadData reads Decker formatted radiosonde data with
c headers. The data is read from an open file corresponding to inUNIT
c
c   ierror    - status of execution
c               ierror < 0: implies execution incomplete
c               ierror = 0: implies execution complete
c .....
c   subroutine ReadData(iunit, date, time, npts,
c   &               press, hgt, temp, dew, ierror)

c   ... force all variables to be declared ...
c   implicit none

c   ... argument list input variables ...
c   integer iunit
c   ... argument list output variables ...
c   integer date(4), time(3), npts, ierror
c   real press(*), hgt(*), temp(*), dew(*)

c   ... local variables ...
c   integer i

c   ... local formats ...
200  format(1x, 3(i2), 1x, 2(i2), 6x, i4)

c   ... read header information ...
c   read(iunit,200,err=999,end=888) date(2),date(3),date(4),
c   &               time(1),time(2),npts
c   ... read data ...
c   read(iunit,*,err=999,end=999)
c   &               (press(i),hgt(i),temp(i),dew(i),i=1,npts)

c   ... exit, no error, good read ...
777  continue
c   ierror = 0
c   return

c   ... exit no read, EOF...
888  continue
c   ierror = 1
c   return

c   ... exit with error ...
999  continue
c   ierror = -1
c   return

end
```


TbModel: SAMPLE USER SPECIFICATIONS

Specify Title:

Test RTE using TbModel

Specify whether (1) or not (0) to output modified profiles
(z,p,tk,rh,denice,denliq):

1

Specify number of frequencies ($\leq$ maxf):

2

Specify frequencies (GHz):

20.6 31.65

Specify number of elevation angles ($\leq$ maxa):

3

Specify elevation angles (degrees):

90.0 19.47122 9.5941

Specify whether (1) or not (0) to extrapolate up to 50.0 mb.

1

Specify whether (1) or not (0) to extrapolate up to 0.1 mb.

1

Specify simulated antenna height (km MSL):

0.0

Specify latitude of radiosonde launch site (degrees):

39.0

Specify maximum acceptable pressure at highest level of input profile (mb):

300.0

Specify maximum acceptable pressure difference between adjacent levels (mb):

0.0

Select saturation vapor pressure calculation over WATER (0) or WATER/ICE (1):

0

Specify the maximum number of cloud layers ($\leq$ maxc):

4

Specify relative humidity threshold for cloud detection (fraction):

0.90

Specify the minimum acceptable cloud layer thickness (km):

0.001

TbModel: SAMPLE DATA

```
-----
900313 1100 72469 44
829.9 1610 -1.0 0.4 826.8 1640 -1.1 4.7 817.9 1727 -1.9 1.4
750.0 2410 -6.8 1.3 706.4 2876 -9.8 1.4 700.0 2946 -9.5 1.3
690.1 3056 -9.4 1.3 680.8 3161 -9.8 1.3 659.6 3406 -8.3 1.3
629.7 3765 -10.0 1.4 525.7 5130 -20.6 3.0 500.0 5499 -23.3 5.7
452.5 6221 -29.4 4.5 436.1 6483 -32.2 16.6 400.0 7086 -37.1 15.8
372.4 7578 -39.8 16.7 300.0 9022 -50.1 30.0 250.0 10205 -53.3 30.0
244.2 10356 -54.0 30.0 214.7 11190 -49.9 30.0 203.0 11556 -50.6 30.0
200.0 11653 -49.3 30.0 189.8 11996 -50.2 30.0 175.2 12521 -47.4 30.0
160.5 13102 -46.5 30.0 150.0 13550 -47.9 30.0 137.1 14140 -50.3 30.0
121.9 14905 -51.4 30.0 113.5 15366 -54.2 30.0 100.0 16178 -54.1 30.0
83.6 17320 -56.8 30.0 79.6 17632 -54.5 30.0 75.3 17986 -56.5 30.0
70.0 18451 -54.8 30.0 55.9 19869 -61.0 30.0 50.0 20567 -58.1 30.0
43.5 21437 -61.5 30.0 41.6 21714 -61.0 30.0 40.1 21943 -58.8 30.0
30.0 23762 -59.5 30.0 24.3 25079 -59.9 30.0 20.0 26301 -58.1 30.0
16.4 27554 -56.8 30.0 16.3 27593 -56.8 30.0
-----
```


TbModel: SUMMARY OF SAMPLE RESULTS

USER SPEC BLOCK:

12 LINES CONTAINING USER SPECIFICATIONS

HEADER BLOCK:

1 LINE CONTAINING:

DATE OF DATA AQUISITION (YY MM DD)
TIME OF DATA AQUISITION (HH MM SS)
LATITUDE OF SITE LOCATION (DEGREES)
ELEVATION OF SITE LOCATION (KM MSL)
NUMBER OF PROFILE LEVELS
NUMBER OF MODELS (CLEAR AND CLOUDY)
NUMBER OF CHANNEL FREQUENCIES
NUMBER OF ELEVATION ANGLES (ANTENNA POSITIONS)
NUMBER OF CLOUD LAYERS IDENTIFIED
OUTPUT SWITCH (1=YES; 0=NO)

OPTIONAL CLOUD LAYER BLOCK (ABSENT IF NUMBER OF CLOUD LAYERS = 0):

1 LINE PER CLOUD LAYER CONTAINING:

CLOUD BASE (KM MSL)
CLOUD TOP (KM MSL)

OPTIONAL ATMOSPHERIC PROFILE BLOCK (ABSENT IF OUTPUT SWITCH = 0):

1 LINE OF USER FRIENDLY COMMENTS

1 LINE PER PROFILE LEVEL, CONTAINING:

HEIGHT (KM MSL)
PRESSURE (MB)
TEMPERATURE (K)
RELATIVE HUMIDITY (FRACTION)
CLOUD LIQUID DENSITY (G/M3)
CLOUD ICE DENSITY (G/M3)

CLEAR-SKY RESULT BLOCK (REPEATS FOR EACH ELEVATION ANGLE):

1 LINE OF FREQUENCY-INDEPENDENT QUANTITIES, CONTAINING:

ELEVATION ANGLE (DEGREES)
PATH-INTEGRATED WATER VAPOR (CM)
EXCESS PATH LENGTH DUE TO WATER VAPOR (CM)
EXCESS PATH LENGTH DUE TO DRY AIR (CM)

1 LINE PER FREQUENCY, CONTAINING:

CHANNEL FREQUENCY (GHZ)
BRIGHTNESS TEMPERATURE WITH COSMIC BACKGROUND TERM (K)
BRIGHTNESS TEMPERATURE WITHOUT COSMIC BACKGROUND TERM (K)
MEAN RADIATING TEMPERATURE OF THE ATMOSPHERE (K)
0. (MEAN RADIATING TEMPERATURE OF NONEXISTENT CLOUD)
PATH-INTEGRATED WATER VAPOR ABSORPTION (NP)
PATH-INTEGRATED DRY AIR ABSORPTION (NP)

CLOUDY-SKY RESULT BLOCK (REPEATS FOR EACH MODEL-1 AND ELEVATION ANGLE):

1 LINE OF FREQUENCY-INDEPENDENT QUANTITIES, CONTAINING:

ELEVATION ANGLE (DEGREES)

TbModel: SUMMARY OF SAMPLE RESULTS

PATH-INTEGRATED CLOUD LIQUID DENSITY (CM)
PATH-INTEGRATED CLOUD ICE DENSITY (CM)
CLOUD DENSITY FRACTION (MULTIPLIER FOR CLOUD DENSITY PROFILES)
1 LINE PER FREQUENCY, CONTAINING:
CHANNEL FREQUENCY (GHZ)
BRIGHTNESS TEMPERATURE WITH COSMIC BACKGROUND TERM (K)
BRIGHTNESS TEMPERATURE WITHOUT COSMIC BACKGROUND TERM (K)
MEAN RADIATING TEMPERATURE OF THE ATMOSPHERE (K)
MEAN RADIATING TEMPERATURE OF LOWEST CLOUD LAYER IDENTIFIED (K)
PATH-INTEGRATED CLOUD LIQUID ABSORPTION (NP)
PATH-INTEGRATED CLOUD ICE ABSORPTION (NP)

TbModel: SAMPLE RESULTS

Title: TEST TbModel

The results in this file reflect the following user specifications:

Discard profiles that terminate below this pressure level (mb): 300.0
Insert level if pressure difference between levels exceeds (mb): 0.0
(zero allows the pressure difference to increase with height.)
Calculate saturation vapor pressure over water (0) or water/ice (1): 0
Maximum number of cloud layers to model per profile: 4
Relative humidity threshold for cloud detection (fraction): 0.90
Minimum acceptable cloud thickness (km): 0.001

3	13	90	11	0	39.00	1.610	3	1	76	3	3	2
1.610	1.618				1.726	2.623	2.925	3.540				
height	press	temp	rh	denliq	denice							
1.610	829.90	272.16	0.971E+00	0.200E+00	0.247E-06							
1.618	829.08	272.13	0.900E+00	0.200E+00	0.274E-06							
1.640	826.80	272.06	0.704E+00	0.000E+00	0.000E+00							
1.683	822.35	271.66	0.802E+00	0.000E+00	0.000E+00							
1.726	817.96	271.27	0.900E+00	0.800E+00	0.127E-04							
1.727	817.90	271.26	0.901E+00	0.800E+00	0.129E-04							
1.821	808.20	270.59	0.902E+00	0.800E+00	0.434E-04							
1.916	798.50	269.90	0.902E+00	0.800E+00	0.111E-03							
2.012	788.80	269.21	0.903E+00	0.800E+00	0.240E-03							
2.110	779.10	268.51	0.903E+00	0.800E+00	0.461E-03							
2.209	769.40	267.80	0.903E+00	0.799E+00	0.813E-03							
2.309	759.70	267.09	0.904E+00	0.799E+00	0.134E-02							
2.410	750.00	266.36	0.904E+00	0.798E+00	0.211E-02							
2.562	735.47	265.38	0.901E+00	0.796E+00	0.362E-02							
2.623	729.72	264.99	0.900E+00	0.796E+00	0.441E-02							
2.718	720.93	264.38	0.898E+00	0.000E+00	0.000E+00							
2.876	706.40	263.36	0.895E+00	0.000E+00	0.000E+00							
2.925	701.94	263.57	0.900E+00	0.792E+00	0.836E-02							
2.946	700.00	263.66	0.902E+00	0.792E+00	0.804E-02							
3.056	690.10	263.76	0.902E+00	0.792E+00	0.771E-02							
3.161	680.80	263.36	0.902E+00	0.791E+00	0.911E-02							
3.283	670.20	264.10	0.903E+00	0.793E+00	0.664E-02							
3.406	659.60	264.86	0.903E+00	0.795E+00	0.469E-02							
3.540	648.24	264.22	0.900E+00	0.794E+00	0.630E-02							
3.583	644.65	264.02	0.899E+00	0.000E+00	0.000E+00							
3.765	629.70	263.16	0.895E+00	0.000E+00	0.000E+00							
4.193	595.03	259.83	0.855E+00	0.000E+00	0.000E+00							
4.647	560.37	256.31	0.813E+00	0.000E+00	0.000E+00							
5.130	525.70	252.56	0.768E+00	0.000E+00	0.000E+00							
5.499	500.00	249.86	0.595E+00	0.000E+00	0.000E+00							
5.851	476.25	246.89	0.622E+00	0.000E+00	0.000E+00							
6.221	452.50	243.76	0.650E+00	0.000E+00	0.000E+00							
6.483	436.10	240.96	0.177E+00	0.000E+00	0.000E+00							
7.086	400.00	236.06	0.178E+00	0.000E+00	0.000E+00							

TbModel: SAMPLE RESULTS

7.578	372.40	233.36	0.152E+00	0.000E+00	0.000E+00
8.261	336.20	228.49	0.882E-01	0.000E+00	0.000E+00
9.022	300.00	223.06	0.168E-01	0.000E+00	0.000E+00
10.205	250.00	219.86	0.143E-01	0.000E+00	0.000E+00
10.356	244.20	219.16	0.138E-01	0.000E+00	0.000E+00
11.190	214.70	223.26	0.169E-01	0.000E+00	0.000E+00
11.556	203.00	222.56	0.164E-01	0.000E+00	0.000E+00
11.653	200.00	223.86	0.174E-01	0.000E+00	0.000E+00
11.996	189.80	222.96	0.167E-01	0.000E+00	0.000E+00
12.521	175.20	225.76	0.190E-01	0.000E+00	0.000E+00
13.102	160.50	226.66	0.198E-01	0.000E+00	0.000E+00
13.550	150.00	225.26	0.186E-01	0.000E+00	0.000E+00
14.140	137.10	222.86	0.166E-01	0.000E+00	0.000E+00
14.905	121.90	221.76	0.157E-01	0.000E+00	0.000E+00
15.366	113.50	218.96	0.137E-01	0.000E+00	0.000E+00
16.178	100.00	219.06	0.137E-01	0.000E+00	0.000E+00
17.320	83.60	216.36	0.119E-01	0.000E+00	0.000E+00
17.632	79.60	218.66	0.134E-01	0.000E+00	0.000E+00
17.986	75.30	216.66	0.121E-01	0.000E+00	0.000E+00
18.451	70.00	218.36	0.132E-01	0.000E+00	0.000E+00
19.869	55.90	212.16	0.925E-02	0.000E+00	0.000E+00
20.567	50.00	215.06	0.110E-01	0.000E+00	0.000E+00
21.437	43.50	211.66	0.896E-02	0.000E+00	0.000E+00
21.714	41.60	212.16	0.925E-02	0.000E+00	0.000E+00
21.943	40.10	214.36	0.106E-01	0.000E+00	0.000E+00
23.762	30.00	213.66	0.101E-01	0.000E+00	0.000E+00
25.079	24.30	213.26	0.989E-02	0.000E+00	0.000E+00
26.301	20.00	215.06	0.110E-01	0.000E+00	0.000E+00
27.554	16.40	216.36	0.119E-01	0.000E+00	0.000E+00
27.593	16.30	216.36	0.119E-01	0.000E+00	0.000E+00
28.127	15.00	222.27	0.420E-03	0.000E+00	0.000E+00
30.786	10.00	225.95	0.185E-03	0.000E+00	0.000E+00
33.175	7.00	231.62	0.699E-04	0.000E+00	0.000E+00
35.489	5.00	238.42	0.250E-04	0.000E+00	0.000E+00
37.063	4.00	243.59	0.121E-04	0.000E+00	0.000E+00
39.143	3.00	250.32	0.493E-05	0.000E+00	0.000E+00
42.168	2.00	259.49	0.151E-05	0.000E+00	0.000E+00
44.365	1.50	262.44	0.892E-06	0.000E+00	0.000E+00
47.505	1.00	266.56	0.431E-06	0.000E+00	0.000E+00
52.885	0.50	263.79	0.267E-06	0.000E+00	0.000E+00
59.808	0.20	252.60	0.269E-06	0.000E+00	0.000E+00
64.832	0.10	242.70	0.330E-06	0.000E+00	0.000E+00
0.90000E+02 0.82178E+00 0.56894E+01 0.18846E+03					
20.600	13.502	11.311	260.448	0.000	0.33345E-01 0.91837E-02
31.650	10.773	8.774	255.704	0.000	0.13161E-01 0.18869E-01
0.19471E+02 0.24611E+01 0.17038E+02 0.56082E+03					
20.600	33.604	31.591	260.700	0.000	0.99861E-01 0.27424E-01
31.650	25.936	24.063	256.015	0.000	0.39423E-01 0.56347E-01
0.95941E+01 0.48953E+01 0.33887E+02 0.10950E+04					

TbModel: SAMPLE RESULTS

20.600	60.466	58.691	261.117	0.000	0.19863E+00	0.54076E-01
31.650	46.630	44.926	256.548	0.000	0.78462E-01	0.11111E+00
	0.90000E+02	0.30157E-01	0.13518E-03	0.25000E+00		
20.600	20.333	18.201	262.807	272.138	0.27377E-01	0.16668E-05
31.650	26.022	24.145	262.830	272.146	0.61469E-01	0.25609E-05
	0.19471E+02	0.90376E-01	0.40493E-03	0.25000E+00		
20.600	51.948	50.094	263.106	272.138	0.82044E-01	0.49928E-05
31.650	66.415	64.858	263.245	272.146	0.18421E+00	0.76709E-05
	0.95941E+01	0.18016E+00	0.80597E-03	0.25000E+00		
20.600	91.573	90.065	263.565	272.138	0.16353E+00	0.99376E-05
31.650	114.266	113.085	263.863	272.146	0.36718E+00	0.15268E-04
	0.90000E+02	0.60313E-01	0.27036E-03	0.50000E+00		
20.600	26.979	24.905	263.857	272.138	0.54754E-01	0.33336E-05
31.650	40.359	38.594	264.337	272.146	0.12294E+00	0.51217E-05
	0.19471E+02	0.18075E+00	0.80986E-03	0.50000E+00		
20.600	68.855	67.147	264.192	272.138	0.16409E+00	0.99856E-05
31.650	100.117	98.823	264.823	272.146	0.36842E+00	0.15342E-04
	0.95941E+01	0.36032E+00	0.16119E-02	0.50000E+00		
20.600	118.014	116.734	264.688	272.138	0.32706E+00	0.19875E-04
31.650	161.238	160.421	265.514	272.146	0.73435E+00	0.30536E-04
	0.90000E+02	0.12063E+00	0.54072E-03	0.10000E+01		
20.600	39.740	37.777	264.831	272.138	0.10951E+00	0.66671E-05
31.650	66.529	64.969	265.399	272.146	0.24588E+00	0.10243E-04
	0.19471E+02	0.36150E+00	0.16197E-02	0.10000E+01		
20.600	98.800	97.351	265.236	272.138	0.32817E+00	0.19971E-04
31.650	151.561	150.665	266.032	272.146	0.73684E+00	0.30684E-04
	0.95941E+01	0.72064E+00	0.32239E-02	0.10000E+01		
20.600	159.607	158.684	265.820	272.147	0.65413E+00	0.39750E-04
31.650	216.591	216.201	266.886	272.146	0.14687E+01	0.61073E-04

APPENDIX C: PROGRAM WgtTest

	PAGE
SOURCE CODE	66
SAMPLE USER SPECIFICATIONS	80
SAMPLE DATA	82
SAMPLE RESULTS	84

WgtTest: SOURCE CODE

program WgtTest

```
C .....
C Program which uses the RTE library, to compute weighting functions
C for upward-looking microwave radiometers with specified channel
C frequencies and antenna orientations from input vertical profiles
C of pressure, temperature, vapor pressure, cloud liquid density, and
C cloud ice density. This software can compute weighting functions
C for any or all of the following parameters: atmospheric
C temperature, atmospheric pressure, atmospheric water vapor density,
C cloud dry air density, and cloud liquid water density. The algorithms
C follow the theory presented in (Schroeder and Westwater, 1992: Guide To
C Passive Microwave c Weighting Function Calculations, NOAA Tech. Memo.
C ERL WPL-225.)
C
C This program uses the same specFILE format as needed for cloudy conditions.
C Therefore, some of the user specifications may not apply.
C
C USAGE:  prompt> tbclear.exe specFILE dataFILE resultFILE
C
C SUBROUTINES CALLED FROM HERE:
C
C ReadWgtSpec
C   USER routine which inputs user specifications;
C   user can modify this routine;
C   user can use some other means for getting these values
C ReadMeanRaob
C   USER routine which inputs 10-year mean profiles;
C   user can modify this routine;
C   user can use different averaging periods (ie monthly, yearly ...);
C   user can substitute different read routines here;
C Radiance_RTE
C   RTE routine which computes Modified Planck Radiance;
C   required for weighting function calculations
C Weight_RTE
C   RTE routine which computes weighting function profile
C .....
C
C   ... force all variables to be declared ...
C   implicit none
C
C   ... parameter variables ...
C   integer maxl, maxa, maxf, maxc, maxw, maxm, errUNIT
C   parameter (maxl=2000,maxa=50,maxf=50,maxc=10,maxw=5,maxm=3)
C   parameter (errUNIT = 0)
```


WgtTest: SOURCE CODE

```
C    ... local variables ...
      real z(maxl),p(maxl),tk(maxl),rh(maxl),
&      denliq(maxl),denice(maxl),
&      boft(maxl),
&      tauprof(maxl),boftatm(maxl),alpha(maxl),ds(maxl),
&      dummy(maxl),wx(maxl,maxw),cldbase(maxc),cldtop(maxc),
&      frq(maxf),angles(maxa),denfrac
      real raoblat, raobelev, pmax, maxdp, cldmax, cldrh,
&      cldthick, hvk, bakgrnd, boftb, zstart
      integer xflag(maxw),specUNIT, dataUNIT, outUNIT, status
      integer iargc, nx, nl, nlp, ice, nlay, printprof, nmodels,
&      nang, nfrq, i, j, k, l, m, date(4), time(3),
&      extrap50, extrap1
      character title*70
      character*20 specfile, datafile, outfile

C    ... command line arguments ...
      if (iargc() .ne. 3) then
        write (errUNIT,*) 'USAGE:  prompt> wgttest.exe ',
&          'specFILE dataFILE resultFILE'
        goto 999
      endif
      call getarg(1,specFILE)
      call getarg(2,dataFILE)
      call getarg(3,outFILE)

C    ... initialize ...
      specUNIT = 55
      dataUNIT = 66
      outUNIT = 77
      status = 0
      nlay = 0

C    ... open I/O files ...
      open(specUNIT, file=specFILE, status='OLD', err=995)
      open(dataUNIT, file=dataFILE, status='OLD', err=996)
      open(outUNIT, file=outFILE, status='UNKNOWN', err=997)

C    ... read user specifications ...
      call ReadWgtSpecs(specUNIT,title,printprof,nx,xflag,nfrq,frq,
&      nang,angles,extrap50,extrap1,zstart,maxdp,raoblat,
&      raobelev,pmax,ice,cldrh,cldmax,cldthick,status)
      if (status .lt. 0) then
        write (errUNIT, 2220) specFILE
        goto 999.
      endif
```


WgtTest: SOURCE CODE

```
c    ... get profile data ...
      call ReadMeanRaob (dataUNIT,ice,raobelev,
&      date,time,nl,z,p,tk,rh,status)
      if (status .lt. 0) then
        write (errUNIT, 2220) dataFILE
        goto 999
      endif

c    ... check if profiles reach user specified pressure height ...
      if (p(nl) .gt. pmax) then
        write (errUNIT, 4000) p(nl), pmax
      endif

c    ... check if profiles reach necessary pressure height ...
      if (p(nl) .gt. 50.0) then
        write (errUNIT, 5000) p(nl), 50.0
        goto 999
      endif

c    ... determine number of models ...
c    ... compute clear sky density fraction ...
      nmodels = 1
      denfrac = 0.0

c    ... compute clear sky cloud density for liquid and ice ...
c    ... compute number of mandatory levels ...
      nlp = 0
      do 27 i = 1, nl
        denliq(i) = 0.0
        denice(i) = 0.0
        if ((mod(p(i),100) .eq. 0) .or.
&          (p(i) .eq. 50.0)) nlp = nlp + 1
27      continue

c    ***** WRITE RESULTS *****
c    ... write header information ...
      write (outUNIT,1100) date(2),date(3),date(4),
&      time(1),time(2),time(3),raoblat,raobelev,
&      nlp,nmodels,nfrq,nang,nlay,printprof
```


WgtTest: SOURCE CODE

```
c      ... write optional modified profiles ...
      if (printprof .eq. 1) then
        write(outUNIT, 1111)
&        'HEIGHT', 'PRESS', 'TEMP', 'RH', 'DENLIQ', 'DENICE'
      do 28 i = 1, nl
        if ((mod(p(i),100) .eq. 0) .or.
&          (p(i) .eq. 50.0)) then
&          write(outUNIT,1000) z(i),p(i),
&          tk(i),rh(i),denliq(i),denice(i)
        endif
28      continue
      endif
c      *****

c      ... RTE computations for each frequency and angle ...
      do 444 j = 1,nfrq
        do 333 k = 1,nang

          call Radiance_RTE (nl,z,p,tk,rh,denliq,denice,
&          frq(j),angles(k),ice,
&          hvk,bakgrnd,boft,boftatm,boftb,
&          alpha,ds,tauprof,status)
          if (status .lt. 0) goto 999

          do 222 l = 1,nx

            call Weight_RTE (nl,z,p,tk,rh,denliq,denice,
&            frq(j),angles(k),ice,bakgrnd,boft,boftatm,
&            boftb,alpha,ds,tauprof,xflag(1),dummy,status)
            if (status .lt. 0) goto 999

            do 112 i = 1, nl
              wx(i,1) = dummy(i)
112          continue

222        continue
```


WgtTest: SOURCE CODE

```
C          ***** WRITE RESULTS *****
          write (outUNIT,2000) denfrac, hvk, bakgrnd, boftb,
&          frq(j), angles(k), nlp, nx, (xflag(1), 1=1,nx)
          if (printprof .eq. 1) then
&              write (outUNIT,8888)
&              'HEIGHT', 'PRESS', 'DS', 'BOFT',
&              'BOFTATM', 'ALPHA', 'TAUPROF'
&              do 11 i = 1,nl
&                  if ((mod(p(i),100) .eq. 0) .or.
&                      (p(i) .eq. 50.0)) then
&                      write (outUNIT,8000) z(i), p(i), ds(i),
&                      boft(i),boftatm(i),alpha(i),tauprof(i)
&                  endif
11          continue
          endif
          write (outUNIT,3333)
&          'HEIGHT', 'WEIGHTING FUNCTIONS:', xflag
&          do 111 i = 1, nl
&              if ((mod(p(i),100) .eq. 0) .or.
&                  (p(i) .eq. 50.0)) then
&                  write (outUNIT,3000) z(i), (wx(i,1), 1=1,nx)
&              endif
111         continue
C          *****
333         continue
444         continue

C      ... close I/O files ...
999     continue
        close(specUNIT)
        close(dataUNIT)
        close(outUNIT)

        stop

C      ... error messages ...
995     continue
        write(errUNIT,1110) specFILE
        goto 999

996     continue
        write(errUNIT,1110) dataFILE
        goto 999

997     continue
        write(errUNIT,1110) outFile
        goto 999
```


WgtTest: SOURCE CODE

```
c    ... local formats ...
1111 format (a8,1x, a8,1x, a8, 3(1x,a13))
1000    format (f8.3, 1x, f8.2, 1x, f8.2, 3(1x,e13.5))

8888 format (5(a8,1x), a13, 1x, a13)
8000 format (5(f8.3,1x),e13.5, 1x, e13.5)

2000    format (f5.3,1x,3(f8.4,1x),f7.3,1x,f7.3,1x,i4,1x,
&        i2,<nx>(1x,i1))

3000    format (f8.3,<nx>(1x,e13.5))
3333    format (a8,3x,a,<nx>(1x,i2))

4000    format('MAIN:  profile terminates at',f7.1,' mb. user ',
&        'requested termination at or above', f7.1, ' mb.')

5000    format('MAIN:  profile terminates at',f7.1,' mb. ',
&        'requires termination at or above', f7.1, ' mb.')

1110    format('MAIN:  ERROR OPENING FILE ',a)
2220    format('MAIN:  ERROR READING FILE ',a)

1100    format (3(i2.2,1x),1x,3(i2.2,1x),1x,f10.2,1x,
&        f10.3,1x,5(i5,1x,),1x,i1)
2200    format (2f8.3,3(4x,2f8.3))
3300    format (f10.3,2f10.2,3e11.3)

6600    format (1x,'Title: ',a70,/)

8800    format (1x,'The results in this file refl',
1    'ect the following user specifications:',//,1x,'Discard profi',
2    'les that terminate below this pressure level (mb):',f12.1,/,1x,
3    'Insert level if pressure difference between levels exceeds (',
4    'mb):',f11.1,/,1x,'(zero allows the pressure difference to in',
5    'crease with height.)',/,1x,'Calculate saturation vapor press',
6    'ure over water (0) or water/ice (1):',i7,/,1x,'Maximum numb',
7    'er of cloud layers to identify:',i32,/,1x,'Relative',
8    ' humidity threshold for cloud detection (fraction):',f16.2,/,
9    'Minimum acceptable cloud thickness (km):',f35.3,/)

end
```


WgtTest: SOURCE CODE

```
C .....
c Subroutine ReadWgtSpecs reads user specifications from inUNIT
c corresponding to an open ascii file.
C .....

      subroutine ReadWgtSpecs(inUNIT,title,printprof,nx,xflag,nfrq,frq,
&      nang,angles,extrap50,extrap1,zstart,maxdp,raoblat,raobelev,
&      pmax,ice,cldrh,cldmax,cldthick,status)

c    ... force all variables to be declared ...
      implicit none

c    ... parameter variables ...
      integer maxf, maxa, maxw
      parameter (maxf=10, maxa=10, maxw=5)

c    ... argument list output variables ...
      character title*70
      real frq(maxf),angles(maxa),zstart,maxdp,raoblat,raobelev,
&      cldrh,cldthick,pmax
      integer inUNIT,printprof,nx,xflag(maxw),nfrq,nang,
&      extrap50,extrap1,ice,cldmax,status

c    ... local variables ...
      integer i, j, k

c    ... input title ...
      read (inUNIT,'(a)',end=999,err=999)
      read (inUNIT,'(a)',end=999,err=999) title

c    ... input switch to write modified profiles (1=yes; 0=no).
      read (inUNIT,1200,end=999,err=999)
      read (inUNIT,*,end=999,err=999) printprof

c    ... input the number of weighting function parameters ...
      read (inUNIT,1100,end=999,err=999)
      read (inUNIT,*,end=999,err=999) nx

c    ... input the weighting function parameters ...
c      1 : temperature
c      2 : pressure
c      3 : water vapor density
c      4 : dry air density
c      5 : cloud liquid density
      read (inUNIT,1300,end=999,err=999)
      read (inUNIT,*,end=999,err=999) (xflag(i),i=1,nx)
```


WgtTest: SOURCE CODE

```
c    ... input number of frequencies ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) nfrq

c    ... input channel frequencies ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) (frq(i),i=1,nfrq)

c    input elevation angles of interest (degrees, maximum of 10).
c    ... input number of angles ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) nang

c    ... input elevation angles ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) (angles(i),i=1,nang)

c    ... input switch for extrapolation up to 50.0 mb. (1=yes, 0=no).
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) extrap50

c    ... input switch for extrapolation up to 0.10 mb. (1=yes, 0=no).
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) extrap1

c    ... input simulated antenna elevation (km MSL) ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) zstart

c    ... input latitude of radiosonde launch site (degrees) ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) raoblat

c    ... input elevation of radiosonde launch site (degrees) ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) raobelev

c    ... input maximum acceptable pressure at highest level of profile (mb)
...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) pmax

c    ... input max acceptable press. difference between adjacent levels ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) maxdp

c    ... input switch to select saturation vapor pressure calculation ...
    read (inUNIT,1100,end=999,err=999)
    read (inUNIT,*,end=999,err=999) ice
```


WgtTest: SOURCE CODE

```
c      ... input maximum number of cloud layers to model ...
      read (inUNIT,1100,end=999,err=999)
      read (inUNIT,*,end=999,err=999) cldmax

c      ... input relative humidity threshold for cloud detection (fraction)
...
      read (inUNIT,1100,end=999,err=999)
      read (inUNIT,*,end=999,err=999) cldrh

c      ... input minimum acceptable cloud layer thickness (km) ...
      read (inUNIT,1100,end=999,err=999)
      read (inUNIT,*,end=999,err=999) cldthick

c      ... exit gracefully, no errors ...
888  continue
      status = 0
      return

c      ... exit with error ...
999  continue
      status = -1
      return

c      ... local formats ...
1100 format (//)
1200 format (//)
1300 format (//////)

      end
```


WgtTest: SOURCE CODE

```
      subroutine ReadMeanRaob (inUNIT,ice,raobelev,date,time,
&                               nl,z,p,tk,rh,ierror)

C .....
C Subroutine ReadMeanRaob reads 10-yr mean profile, such as those
C keypunched by Ruth Rodriguez, from the unit number corresponding to
C an open ascii file. This subroutine can be modified to fit users
C own data format or a different averaging period.
C
C Inputs:
C   inUNIT = unit number corresponding to an open ascii file
C
C Outputs:
C   date = date of radiosone launch; integer array
C   time = time of radiosone launch; integer array
C   ice  = saturation vapor pressure switch; integer scaler
C       ice = 0: compute (es) over water only
C       ice = 1: compute (es) over water (temperature > -10 C)
C               compute (es) over ice (temperature <= -10 C)
C   raobelev = Elevation height of radiosonde launch (km MSL)
C   nl = number of profile levels; integer scaler
C   z  = geopotential height profile (km msl); real array
C   p  = pressure profile (mb); real array
C   tk = temperature profile (k); real array
C   rh = relative humidity (fractional); real array
C   ierror = error flag (0=none, -1=error)
C .....

C   ... force all variables to be declared ...
      implicit none

C   ... parameter variables ...
      integer maxl, errUNIT
      parameter (maxl=2000, errUNIT=0)

C   ... argument list variables ...
      integer inUNIT, date(4), time(3), nl, ice, ierror
      real raobelev, z(maxl), p(maxl), tk(maxl), rh(maxl)

C   ... local variables ...
      integer i
      real psfc, pdif, t, rho, rvap, es, e(maxl), zz

C   ... user functions ...
      real fGeoHgt
```


WgtTest: SOURCE CODE

```
c    ... read header information ...
    read (inUNIT,3000,end=999) date, time, psfc, nl
3000 format (4(i2.2), 1x, 3(i2.2), 1x, f7.2, 1x, i4)

c    ... read raw data (p, t, rho) and compute (tk, rh, z,) ...
    do 11 i = 1, nl
        read (inUNIT,*,end=999) pdif,t,rho
        p(i) = psfc - pdif
        if (p(i) .le. 0.0) go to 998
        tk(i) = t + 273.16
        call SatVap(tk(i), ice, rvap, es)
        if (rho .le. 0.0) then
            e(i) = p(i) * 4.e-6
        else
            e(i) = rho * rvap * tk(i)
        endif
        rh(i) = e(i) / es
        if (i .eq. 1) then
            z(i) = raobelev
        else
            zz = fGeoHgt(p(i-1),p(i),tk(i-1),tk(i),e(i-1),e(i))
            z(i) = z(i-1) + zz
        endif
11    continue

    return

998    continue
    write (errUNIT,1000) i
    ierror = -1
    return

999    continue
    write (errUNIT,2000) inUNIT
    ierror = -1
    return

1000    format (1x,'ERROR from ReadMean:  pressure difference ',
&           'exceeds psfc at level',i3,'.')
2000    format (1x,'ERROR from ReadMean:  invalid eof found on ',
&           'unit',i3,'.')

    end
```


WgtTest: SOURCE CODE

```
C .....
C Subroutine SatVap computes saturation vapor pressure and water
C vapor gas constant Saturation vapor pressure (es,in mb) is computed
C over water or ice at temperature tk (kelvins), using the Goff-Gratch
C formulation (List,1963).
C
C input arguments:
C   tk   = temperature (K)
C   ice  = switch to calculate saturation vapor pressure over
C         water only (0) or water and ice, depending on tk (1)
C output arguments:
C   rvap = gas constant for water vapor (mb*(m**3)/g/k)
C   es   = saturation vapor pressure
C
C example uses:
C   e = rho * rvap * tk
C   rh = e / es
C
C   e = rh * es
C   rho = e / (rvap * tk)
C .....
      subroutine SatVap (tk,ice,rvap,es)
C
C   ... force all variables to be declared ...
      implicit none
C
C   ... argument list variables ...
      real tk, rvap, es
      integer ice
C
C   ... local variables ...
      real y
C
C   ... compute gas constant for water vapor (mb*(m**3)/g/k) ...
      rvap = 0.01 * 8.314510 / 18.01528
C
C   ... compute saturation vapor pressure over water ...
      if ((tk.gt.263.16) .or. (ice.eq.0)) then
        y = 373.16 / tk
        es = -7.90298 * (y-1.) + 5.02808 * alog10 (y) -
1         1.3816e-7 * (10 ** (11.344 * (1. - (1./ y)))) - 1.) +
2         8.1328e-3 * (10.** (-3.49149 * (y - 1.)) - 1.) +
3         alog10 (1013.246)
```


WgtTest: SOURCE CODE

```
c ... compute saturation vapor pressure over ice ...
else
  y = 273.16 / tk
  es = -9.09718 * (y - 1.) - 3.56654 * alog10 (y) +
1    .876793 * (1.- (1./ y)) + alog10 (6.1071)
end if

es = 10. ** es

return
end
```

```
c .....
c Function fGeoHgt return Geopotential Height at current level in (km).
c The algorithm follows eq.(6.5) on p.79 of Introduction to Theoretical
c Meteorology by S.L. Hess (1959). Mixing ratio and virtual temperature
c are defined on pp.59 and 60, respectively.
```

```
c
c inputs passed as arguments:
```

```
c   p1   = atmospheric pressure at previous level (mb)
c   p2   = atmospheric pressure at current level (mb)
c   tk1  = atmospheric temperature at previous level (K)
c   tk2  = atmospheric temperature at current level (K)
c   e1   = atmospheric vapor pressure at previous level (mb)
c   e2   = atmospheric vapor pressure at current level (mb)
```

```
c
c .....
```

```
real function fGeoHgt (p1, p2, tk1, tk2, e1, e2)
```

```
c ... force all variables to be declared ...
implicit none
```

```
c ... argument list variables ...
real p1, p2, tk1, tk2, e1, e2
```

```
c ... local variables ...
real roverg, w, tvlower, tvupper, tvbar, z,
&      r, mdry, mvap, epsilon, g, rdry
```

```
c ... initialization ...
data r, mdry, mvap / 8.314510, 28.96415, 18.01528 /
epsilon = mvap / mdry
g = 9.80665
rdry = r / mdry
roverg = rdry / g
```


WgtTest: SOURCE CODE

```
c      Compute mixing ratio and virtual temperature
c      at the lower boundary of the first layer.

      w = epsilon * e1 / (p1 - e1)
      tvlower = tk1 * (1.+ (w / epsilon)) / (1.+ w)

c      Compute geopotential height profile.

      w = epsilon * e2 / (p2 - e2)
      tvupper = tk2 * (1.+ (w / epsilon)) / (1.+ w)
      tvbar = sqrt (tvlower * tvupper)
      z = roverg * tvbar * alog (p1 / p2)
fGeoHgt = z

      return
      end
```


WgtTest: SAMPLE USER SPECIFICATIONS

Specify Title:

Test Weighting Function Program for clear sky conditions.

Specify whether (1) or not (0) to output modified profiles
(z,p,tk,e,denice,denliq):

1

Specify number of Weighting Function Parameters ($\leq$ maxw):

5

Specify Weighting Function Parameters:

- 1) TEMPERATURE
- 2) PRESSURE
- 3) WATER VAPOR DENSITY
- 4) DRY AIR DENSITY
- 5) CLOUD LIQUID DENSITY

1 2 3 4 5

Specify number of frequencies ($\leq$ maxf):

2

Specify frequencies (GHz):

20.6 90.00

Specify number of antenna positions ($\leq$ maxa):

2

Specify antenna positions [ie. elevation angles (degrees)]:

90.0 15.0

Specify whether (1) or not (0) to extrapolate up to 50.0 mb.

-1

Specify whether (1) or not (0) to extrapolate up to 0.1 mb.

-1

Specify simulated antenna height (km MSL):

0.0

Specify latitude of radiosonde launch site (degrees):

39.0

Specify elevation of radiosonde launch site (km MSL):

1.610

Specify maximum acceptable pressure at highest level of input profile (mb):

300.0

WgtTest: SAMPLE USER SPECIFICATIONS

Specify maximum acceptable pressure difference between adjacent levels (mb):
-1.0

Select saturation vapor pressure calculation over WATER (0) or WATER/ICE (1):

0

Specify the maximum number of cloud layers ($\leq$ maxc):

0

Specify relative humidity threshold for cloud detection (fraction):

0.90

Specify the minimum acceptable cloud layer thickness (km):

0.001

WgtTest: SAMPLE DATA

19920731 000000 840.00 84
0 .210E+02 .777E+01
10 .208E+02 .723E+01
20 .206E+02 .685E+01
30 .201E+02 .656E+01
40 .196E+02 .630E+01
50 .190E+02 .603E+01
60 .183E+02 .580E+01
70 .175E+02 .557E+01
80 .168E+02 .534E+01
90 .160E+02 .512E+01
100 .151E+02 .495E+01
110 .143E+02 .478E+01
120 .134E+02 .461E+01
130 .124E+02 .443E+01
140 .115E+02 .427E+01
150 .106E+02 .414E+01
160 .967E+01 .401E+01
170 .872E+01 .387E+01
180 .777E+01 .373E+01
190 .678E+01 .359E+01
200 .575E+01 .346E+01
210 .471E+01 .332E+01
220 .367E+01 .320E+01
230 .262E+01 .308E+01
240 .155E+01 .295E+01
250 .449E+00 .282E+01
260 -.651E+00 .270E+01
270 -.175E+01 .257E+01
280 -.284E+01 .245E+01
290 -.393E+01 .232E+01
300 -.503E+01 .219E+01
310 -.613E+01 .205E+01
320 -.721E+01 .190E+01
330 -.828E+01 .174E+01
340 -.934E+01 .156E+01
350 -.104E+02 .139E+01
360 -.114E+02 .122E+01
370 -.125E+02 .107E+01
380 -.135E+02 .935E+00
390 -.146E+02 .814E+00
400 -.158E+02 .701E+00
410 -.169E+02 .604E+00
420 -.181E+02 .516E+00
430 -.193E+02 .444E+00
440 -.206E+02 .381E+00
450 -.219E+02 .323E+00
460 -.233E+02 .276E+00

WgtTest: SAMPLE DATA

470	-.247E+02	.239E+00
480	-.262E+02	.202E+00
490	-.277E+02	.171E+00
500	-.293E+02	.143E+00
510	-.310E+02	.119E+00
520	-.327E+02	.978E-01
530	-.344E+02	.802E-01
540	-.362E+02	.641E-01
550	-.380E+02	.507E-01
560	-.398E+02	.392E-01
570	-.417E+02	.286E-01
580	-.437E+02	.187E-01
590	-.456E+02	.107E-01
600	-.474E+02	.813E-02
610	-.493E+02	.648E-02
620	-.512E+02	.517E-02
630	-.530E+02	.414E-02
640	-.547E+02	.331E-02
650	-.563E+02	.264E-02
660	-.579E+02	.212E-02
670	-.593E+02	.173E-02
680	-.606E+02	.141E-02
690	-.619E+02	.113E-02
700	-.632E+02	.884E-03
710	-.644E+02	.689E-03
720	-.652E+02	.539E-03
730	-.654E+02	.431E-03
740	-.650E+02	.343E-03
750	-.638E+02	.00
760	-.621E+02	.00
770	-.601E+02	.00
780	-.579E+02	.00
790	-.557E+02	.00
800	-.532E+02	.00
810	-.502E+02	.00
820	-.451E+02	.00
830	-.326E+02	.00

WgtTest: SAMPLE RESULTS

92 07 31	00 00 00	39.00	1.610	9	1	2	2	0	1
HEIGHT	PRESS	TEMP	RH	DENLIQ		DENICE			
2.031	800.00	292.76	0.37335E+00	0.00000E+00		0.00000E+00			
3.164	700.00	284.66	0.41365E+00	0.00000E+00		0.00000E+00			
4.430	600.00	274.71	0.54753E+00	0.00000E+00		0.00000E+00			
5.869	500.00	263.82	0.62989E+00	0.00000E+00		0.00000E+00			
7.557	400.00	252.56	0.37300E+00	0.00000E+00		0.00000E+00			
9.619	300.00	236.96	0.25166E+00	0.00000E+00		0.00000E+00			
12.313	200.00	218.46	0.91205E-01	0.00000E+00		0.00000E+00			
16.597	100.00	208.16	0.33450E-01	0.00000E+00		0.00000E+00			
20.916	50.00	217.46	0.61690E-02	0.00000E+00		0.00000E+00			
0.000	0.9886	2.1294	24.3471	20.600	90.000	9	5	1	2 3 4 5
HEIGHT	PRESS	DS	BOFT	BOFTATM	ALPHA		TAUPROF		
2.031	800.000	0.107	295.623	3.505	0.25956E-01		0.11890E-01		
3.164	700.000	0.119	287.429	10.532	0.18349E-01		0.36548E-01		
4.430	600.000	0.133	277.364	15.883	0.13085E-01		0.56370E-01		
5.869	500.000	0.153	266.356	19.651	0.71467E-02		0.71124E-01		
7.557	400.000	0.183	254.965	21.323	0.20480E-02		0.78002E-01		
9.619	300.000	0.228	239.182	21.869	0.60696E-03		0.80376E-01		
12.313	200.000	0.313	220.469	22.074	0.22324E-03		0.81337E-01		
16.597	100.000	0.580	210.051	22.182	0.61526E-04		0.81889E-01		
20.916	50.000	1.155	219.459	22.209	0.13714E-04		0.82026E-01		
HEIGHT	WEIGHTING FUNCTIONS: 1 2 3 4 5								
2.031	-0.11603E-01	-0.83814E-03	0.78101E+00		0.63456E-02		0.13831E+02		
3.164	-0.81482E-02	0.47107E-04	0.80596E+00		0.53593E-02		0.15805E+02		
4.430	-0.59044E-02	0.71099E-03	0.82419E+00		0.46410E-02		0.18845E+02		
5.869	-0.33287E-02	0.11210E-02	0.81879E+00		0.32167E-02		0.23175E+02		
7.557	-0.13276E-02	0.91476E-03	0.77193E+00		0.12859E-02		0.29062E+02		
9.619	-0.67684E-03	0.63429E-03	0.68742E+00		0.54086E-03		0.39795E+02		
12.313	-0.33619E-03	0.43221E-03	0.53676E+00		0.27665E-03		0.51949E+02		
16.597	-0.90844E-04	0.23181E-03	0.30024E+00		0.13909E-03		0.51756E+02		
20.916	-0.21192E-04	0.10790E-03	0.15464E+00		0.67724E-04		0.52273E+02		
0.000	0.9886	1.6857	78.2885	20.600	15.000	9	5	1	2 3 4 5
HEIGHT	PRESS	DS	BOFT	BOFTATM	ALPHA		TAUPROF		
2.031	800.000	0.413	295.623	13.309	0.25956E-01		0.45925E-01		
3.164	700.000	0.458	287.429	38.617	0.18349E-01		0.14104E+00		
4.430	600.000	0.513	277.364	56.657	0.13085E-01		0.21732E+00		
5.869	500.000	0.588	266.356	68.720	0.71467E-02		0.27396E+00		
7.557	400.000	0.700	254.965	73.895	0.20480E-02		0.30029E+00		
9.619	300.000	0.869	239.182	75.556	0.60696E-03		0.30934E+00		
12.313	200.000	1.186	220.469	76.173	0.22324E-03		0.31300E+00		
16.597	100.000	2.178	210.051	76.499	0.61526E-04		0.31507E+00		
20.916	50.000	4.298	219.459	76.578	0.13714E-04		0.31559E+00		
HEIGHT	WEIGHTING FUNCTIONS: 1 2 3 4 5								
2.031	-0.19528E-01	-0.25929E-02	0.24162E+01		0.19631E-01		0.42787E+02		
3.164	-0.17984E-01	0.14472E-03	0.24761E+01		0.16465E-01		0.48556E+02		
4.430	-0.15327E-01	0.21711E-02	0.25168E+01		0.14172E-01		0.57545E+02		
5.869	-0.95149E-02	0.34069E-02	0.24884E+01		0.97758E-02		0.70432E+02		

WqtTest: SAMPLE RESULTS

7.557	-0.39570E-02	0.27688E-02	0.23364E+01	0.38920E-02	0.87964E+02
9.619	-0.20319E-02	0.19111E-02	0.20712E+01	0.16296E-02	0.11990E+03
12.313	-0.10061E-02	0.12949E-02	0.16082E+01	0.82887E-03	0.15564E+03
16.597	-0.26978E-03	0.68861E-03	0.89190E+00	0.41317E-03	0.15375E+03
20.916	-0.62433E-04	0.31788E-03	0.45559E+00	0.19953E-03	0.15401E+03
0.000	4.3193	0.2244	9.5461	90.000	90.000
9	5	1	2	3	4
5					
HEIGHT	PRESS	DS	BOFT	BOFTATM	ALPHA
2.031	800.000	0.107	67.281	1.701	0.53977E-01
3.164	700.000	0.119	65.406	4.780	0.34515E-01
4.430	600.000	0.133	63.102	6.890	0.22861E-01
5.869	500.000	0.153	60.581	8.259	0.11643E-01
7.557	400.000	0.183	57.974	8.844	0.36401E-02
9.619	300.000	0.228	54.362	9.080	0.15507E-02
12.313	200.000	0.313	50.079	9.212	0.77448E-03
16.597	100.000	0.580	47.695	9.295	0.22827E-03
20.916	50.000	1.155	49.848	9.316	0.48731E-04
0.000	4.3193	0.2244	9.5461	90.000	90.000
9	5	1	2	3	4
5					
HEIGHT	PRESS	DS	BOFT	BOFTATM	ALPHA
2.031	800.000	0.107	67.281	1.701	0.53977E-01
3.164	700.000	0.119	65.406	4.780	0.34515E-01
4.430	600.000	0.133	63.102	6.890	0.22861E-01
5.869	500.000	0.153	60.581	8.259	0.11643E-01
7.557	400.000	0.183	57.974	8.844	0.36401E-02
9.619	300.000	0.228	54.362	9.080	0.15507E-02
12.313	200.000	0.313	50.079	9.212	0.77448E-03
16.597	100.000	0.580	47.695	9.295	0.22827E-03
20.916	50.000	1.155	49.848	9.316	0.48731E-04
0.000	4.3193	0.2244	9.5461	90.000	90.000
9	5	1	2	3	4
5					
HEIGHT	PRESS	DS	BOFT	BOFTATM	ALPHA
2.031	800.000	0.107	67.281	1.701	0.53977E-01
3.164	700.000	0.119	65.406	4.780	0.34515E-01
4.430	600.000	0.133	63.102	6.890	0.22861E-01
5.869	500.000	0.153	60.581	8.259	0.11643E-01
7.557	400.000	0.183	57.974	8.844	0.36401E-02
9.619	300.000	0.228	54.362	9.080	0.15507E-02
12.313	200.000	0.313	50.079	9.212	0.77448E-03
16.597	100.000	0.580	47.695	9.295	0.22827E-03
20.916	50.000	1.155	49.848	9.316	0.48731E-04
0.000	4.3193	0.2244	9.5461	90.000	90.000
9	5	1	2	3	4
5					
HEIGHT	PRESS	DS	BOFT	BOFTATM	ALPHA
2.031	800.000	0.107	67.281	1.701	0.53977E-01
3.164	700.000	0.119	65.406	4.780	0.34515E-01
4.430	600.000	0.133	63.102	6.890	0.22861E-01
5.869	500.000	0.153	60.581	8.259	0.11643E-01
7.557	400.000	0.183	57.974	8.844	0.36401E-02
9.619	300.000	0.228	54.362	9.080	0.15507E-02
12.313	200.000	0.313	50.079	9.212	0.77448E-03
16.597	100.000	0.580	47.695	9.295	0.22827E-03
20.916	50.000	1.155	49.848	9.316	0.48731E-04
0.000	4.3193	0.2244	9.5461	90.000	90.000
9	5	1	2	3	4
5					
HEIGHT	PRESS	DS	BOFT	BOFTATM	ALPHA
2.031	800.000	0.107	67.281	1.701	0.53977E

*U.S. GOVERNMENT PRINTING OFFICE: 1993-774-025/89003