

```

1 !Skew-T Program
2 !Designed to read in an ascii text file holding an atmospheric sounding,
3 !and output an ascii file for easy plotting, such as by Excel.
4 !Copyright © 2007 by Roland Stull
5 !version 13. 4 Feb 2007. Modified 5 Aug 2021.
6
7
8 !===== modules =====
9
10 module soundmod                                !holds sounding arrays
11     character (len=50) :: filename              !holds name of sounding file
12     character (len=60) :: outname               !holds name of output file
13     character (len=100) :: title                !title line for sounding
14     integer :: nlines                          !number of lines in sounding, initialized to 0
15     integer, parameter :: maxlines = 120        !max sounding lines that can be captured
16     real, dimension(maxlines) :: P              !Pressure (kPa)
17     real, dimension(maxlines) :: z              !Height (m)
18     real, dimension(maxlines) :: T              !Temperature (C)
19     real, dimension(maxlines) :: Td             !Dew Point (C)
20     character (len=1) :: tab=achar(9)           !ascii tab character
21 end module soundmod
22
23 !===== main program =====
24 program skewtmain                                !Skew-T plotting main program
25
26 !declare variables
27     implicit none                                !enforce strong typing
28
29 !set-up
30     call welcome                                !welcome the user
31     call getfile                                !read in the sounding file
32     call prepareoutput                          !prepare disk to receive output
33
34 !compute sounding plot data
35     call plotsounding                          !plot the sounding
36     call plotisobar                            !plot the isobars in the background
37     call plotisotherm                         !plot a few isotherms in the background
38     call plotdryadiabat                       !plot a few dry adiabats in background
39     call plotwetadiabat                       !plot a few saturated adiabats in backgr
40     call plotisohume                          !plot a few isohumes in the background
41
42 !finish
43     call cleanup                                !close files and release memory
44
45 end program skewtmain
46
47
48
49 !=====
50 subroutine welcome                                !Welcome the interactive user
51     implicit none                                !enforce strong typing
52     write(*,*)
53     write(*,*) "=====
54     write(*,*) "Welcome to Skew-T ... a Sounding Plotting Program"
55     write(*,*) "=====
56     write(*,*)
57     write(*,*) "   Based on Stull,R., 2018: Practical Meteorology"
58     write(*,*) "   Chapters 1 – 6. Coded by R. Stull, UBC, Feb 2007. Updated Aug 2021."
59     write(*,*)
60     write(*,*) "Expects ascii 'Text List' sounding as input, as from U.Wyoming site:"
61     write(*,*) "   http://weather.uwyo.edu/upperair/sounding.html"
62     write(*,*) "Produces .csv output file, which can be read into"
63     write(*,*) "   Excel as one long Series for plotting."
64     write(*,*)
65 end subroutine welcome
66

```

```

67
68 !=====
69 subroutine getfile                                !read in the sounding file
70     use soundmod                                  !use sounding module
71     implicit none                                !enforce strong typing
72     character (len=100) :: line                  !holds one line from the sounding
73     integer :: ios                                !input/output error status
74     integer :: i                                  !dummy iteration counter
75     real :: PhPa, zm, TC, TdC                    !temporary Pressure, height, Temp, Td variabl
76
77     write(*,*) "...starting getfile"             !debugging output
78
79 !First, ask the user for the file name that holds the sounding ascii data
80     do i = 1,3                                    !give user 3 tries to enter correct file
81         write(*,*)
82         write(*,"(a)",advance="no") "Type in file name for sounding, then hit Enter: "
83         read(*,*) filename
84
85 !Next, open the file
86         open(1,file=filename,status="old",iostat=ios) !connect to sounding file
87         if (ios .ne. 0) then                       !Can't open the file
88             write(*,*) "Sorry, can't find file: ",filename
89             write(*,*) "Don't forget to include the suffix .txt in the file name."
90             if (i < 3) write(*,*) "Try again."
91             cycle                                  !allow user to try again
92         else                                       !Can open the file
93             write(*,*) "Good. Successfully opened: ", filename
94             write(*,*)
95             exit                                  !jump out of do loop
96         endif
97     enddo
98     if (ios .ne. 0) stop "Sorry. Perhaps the file doesn't exist. Bye."
99
100 !At this point, we have a good file.
101
102 !Read title line and skip other header lines in the data file
103     do i = 1,5                                    !for all 5 header lines
104         read(1,"(a)",iostat=ios) line              !ios is newly set here
105         if (ios .ne. 0) exit                       !Cannot read lines
106         if (i .eq. 1) title = line                 !save title
107     enddo
108
109     write(*,*) title                               !Display title
110     write(*,*)
111     write(*,*) " i      P (kPa)      z (m)      T (C)      Td (C)"
112
113 !Read each data line in the sounding file, and echo to screen
114     i = 0                                          !initialize line counter
115     do                                           !for each sounding level
116         read(1,"(F7.1, F7.0, 2F7.1)",iostat=ios) PhPa, zm, TC, TdC !read obs
117         if (ios .ne. 0) exit                       !finished reading lines
118         if ((PhPa < 100.0) .or. (i >= maxlines)) exit !don't care about low P.
119
120         !At this point, a good line has been found, so save in data arrays.
121         i = i + 1                                  !count data lines
122         P(i) = 0.1*PhPa                             !store Pressure, convert kPa
123         z(i) = zm                                    !store heights
124         T(i) = TC                                    !store temperatures
125         Td(i) = TdC                                  !store dew points
126         write(*,"(I3,4F12.2)") i, P(i), z(i), T(i), Td(i) !echo line to screen
127     enddo
128     nlines = i                                       !save line count
129     write(*,*)
130     write(*,*) "Number of sounding lines captured = ",nlines
131
132 !Finished getting the sounding data. Close the open file.

```

```

133     write(*,*)
134     close(1)                                !close the connection to the input file
135 end subroutine getfile
136
137
138 !=====
139 subroutine prepareoutput                    !prepare disk to receive output
140     use soundmod                          !include sounding variables
141     implicit none                        !enforce strong typing
142     integer :: i                          !position of "." in filename
143     integer :: ios                        !error status for file opening
144
145     write(*,*) "...starting prepareoutput" !debugging output
146     write(*,*)
147
148 !Create an output file to hold the results
149     i = index(filename, ".") - 1          !find the character just before "." in the filename
150     if (i .le. 0) i = len_trim(filename) !use whole filename if no "."
151     outname = filename(1:i) // "out.csv"  !create a new output file name
152     write(*,*) "Output will be saved in file: ", outname
153     write(*,*)
154     open(2, file=outname, status="replace", iostat=ios) !open the output file
155     if (ios .ne. 0) then                    !error opening file
156         write(*,*) "Sorry, can't create file: ", outname
157         stop "Bye."
158     endif
159
160 !Write the title (sounding time and place) from the first header line of the input file
161     write(*,*) trim(title)                 !display title on screen
162     write(*,*)
163     write(*,*) " Calculated in addition to the sounding temperature and dew point:"
164     write(*,*) "   Isobars: P (kPa) = 100, 90, 80, 70, ..., 20, 10"
165     write(*,*) "   Isotherms: T (C) = -80, -60, -40, -20, 0, 20"
166     write(*,*) "   Dry Adiabats: Theta (C) = -20, 0, 20, 40, 60, 80"
167     write(*,*) "   Saturated adiabat: ThetaL (C) = 0, 10, 20, 30"
168     write(*,*) "   Isohumes: r (g/kg) = 0.2, 1.0, 5.0, 20.0"
169     write(*,*)
170     write(*,*) "   X(T)      ", tab, "   Y(P)" !column headers
171
172     write(2,*) trim(title)                 !display title on output file
173     write(2,*)
174     write(2,*) " Calculated in addition to the sounding temperature and dew point:"
175     write(2,*) "   Isobars: P (kPa) = 100 90 80 70 ... 20 10"
176     write(2,*) "   Isotherms: T (C) = -80 -60 -40 -20 0 20"
177     write(2,*) "   Dry Adiabats: Theta (C) = -20 0 20 40 60 80"
178     write(2,*) "   Saturated adiabat: ThetaL (C) = 0 10 20 30"
179     write(2,*) "   Isohumes: r (g/kg) = 0.2 1.0 5.0 20.0"
180     write(2,*)
181     write(2,*) "   X(T)      ", ", ", ", " Y(P)" !column headers
182
183     write(*,*)
184 end subroutine prepareoutput
185
186
187 !=====
188 subroutine plotsounding                    !plot the sounding
189     use soundmod                          !include sounding arrays
190     implicit none                        !enforce strong typing
191     integer :: i                          !dummy index
192     real :: xst, yst                     !function names
193     real :: y                             !ordinate value
194     real :: xT, xTd                       !abscissa values for T and Td(C)
195     character (len=40) :: fmt             !format for output
196
197     write(*,*) "...starting plotsounding" !debugging output
198     write(*,*)

```

```

199     write(2,*)                                !skip line in output file
200
201     fmt = "(F12.4, a1, F12.4)"                 !format for output
202
203 !   For the temperature profile
204 do i=1,nlines                                  !for each sounding level
205     y = yst(P(i))                              !find ordinate value
206     xT = xst(T(i),y)
207     if ((xT .ge. -40.) .and. (xT .le. 40.)) then !inside skewT domain
208         write(*,fmt) xT,tab,y                  !Y & x coord for T, show on screen
209         write(2,fmt) xT,"",y                  !Y & x coord fpr T, write in output file
210     endif
211 enddo
212
213 write(*,*)
214 write(2,*)                                     !skip line in output file
215
216 !   For the dew point profile
217 do i=1,nlines                                  !for each sounding level
218     y = yst(P(i))                              !find ordinate value
219     xTd = xst(Td(i),y)
220     if ((xTd .ge. -40.) .and. (xTd .le. 40.)) then !inside skewT domain
221         write(*,fmt) xTd,tab,y                 !Y & x coord for Td, show on screen
222         write(2,fmt) xTd,"",y                 !Y & x coord for Td, write in output file
223     endif
224 enddo
225
226 write(*,*)
227 write(2,*)                                     !skip line in output file
228 end subroutine plotsounding
229
230
231 !=====
232 subroutine plotisobar                          !plot several isobars in background
233 !Because isobars are straight lines, we need to find only the left and right ends
234     use soundmod
235     implicit none                             !enforce strong typing
236     integer :: i                               !dummy height index
237     real :: yst                                !for the function subroutine
238     real, parameter :: xcold = -40.0          !degC cold (left) end of isobar
239     real, parameter :: xhot = 40.0            !degC hot (right) end of isobar
240     real :: PkPa                               !pressure (kPa)
241     real :: y                                  !Y coordinate on skewT
242     character(len=30) :: fmt                  !string holding output format
243
244     write(*,*) "...starting plotisobar"        !debugging output
245     write(*,*)
246     write(*,*) "Isobars:"
247     write(*,*) "      X(T)      ",tab,"      Y(P)" !column headers for output
248     write(*,*)
249
250     fmt = "(F12.4, a1, F12.4)"                 !format for output
251     write(2,*)                                !skip line in output file
252     write(2,*)                                !skip line in output file
253
254     do i = 10, 100, 10                        !for each isobar to be plotted
255         PkPa = real(i)                        !convert pressure to real number
256         y = yst(PkPa)                         !get Y coordinate on skewT
257
258         write(*,"(a,F10.1)") "P (kPa) = ",PkPa !debug output on screen
259
260         write(*,fmt) xcold,tab,y              !write isobar table to screen
261         write(*,fmt) xhot,tab,y
262         write(*,*)
263
264         write(2,fmt) xcold,"",y              !write isobar table to output

```

```

265     write(2,fmt) xhot,"",y
266     write(2,*)
267     enddo
268
269     write(*,*)
270     write(2,*) !skip line in output file
271 end subroutine plotisobar
272
273
274 !=====
275 subroutine plotisotherm !plot a few isotherms in the background
276     use soundmod !include sounding data
277     implicit none !enforce strong typing
278     real :: xst, yst !type declaration for functions
279     real :: x, y !temporary coordinates in skewT
280     real :: TC !temperature (C)
281     real :: PkPa !pressure (kPa)
282     integer :: i !index for pressure height
283     integer :: j !index for temperature
284     character(len=30) :: fmt !string holding output format
285
286     write(*,*) "...starting plotisotherm" !debugging output
287     write(*,*)
288     write(*,*) "Isotherms:"
289     write(*,*) "    X(T)    ",tab,"    Y(P)" !column headers for output
290
291     fmt = "(F12.4, a1, F12.4)" !format for output
292
293     do j = -80, 20, 20 !for each isotherm
294         TC = real(j) !temperature (C)
295         write(*,*)
296         write(*,"(a,F8.1)") "T (C) = ", TC
297         write(2,*)
298
299         do i = 100, 10, -5 !for each pressure height
300             PkPa = real(i) !pressure (kPa)
301             y = yst(PkPa) !y coordinate of skewT
302             x = xst(TC,y) !x coordinate of skewT
303
304             write(*,fmt) x,tab,y !x & y coord, show on screen
305             if ((x .ge. -40.) .and. (x .le. 40.)) then !inside skewT domain
306                 write(2,fmt) x,"",y !x & y coord, write to output
307             endif
308         enddo
309     enddo !end of isotherm loop
310
311     write(*,*)
312 end subroutine plotisotherm
313
314
315 !=====
316 subroutine plotdryadiabat !plot a few dry adiabats in background
317     use soundmod !include sounding data
318     implicit none !enforce strong typing
319     real :: xst, yst !type declaration for functions
320     real :: x, y !temporary coordinates in skewT
321     real :: TC, TK !temperature (C) & (K)
322     real :: thetaC, thetaK !potential temperature (C) & (K)
323     real :: PkPa !pressure (kPa)
324     real, parameter :: RdCp = 0.28571 !Rd/Cp in eq for pot temperature
325     real, parameter :: Po = 100. !reference pressure (kPa)
326     integer :: i !index for pressure height
327     integer :: j !index for potential temperature
328     character(len=30) :: fmt !string holding output format
329
330     write(*,*) "...starting plotdryadiabat" !debugging output

```

```

331 write(*,*)
332 write(*,*) "Dry Adiabats:"
333
334 fmt = "(F12.4, a1, F12.4)" !format for output
335 write(2,*)
336 write(2,*)
337
338 do j = -20, 80, 20 !for each adiabat
339     thetaC = real(j) !potential temperature (C)
340     thetaK = thetaC + 273. !potential temperature (K)
341     write(*,*)
342     write(*,"(a,F8.1)") "Theta (C) = ", thetaC
343     write(*,*) "    P (kPa)    ", tab, "    T (C)"
344     write(2,*)
345
346     do i = 100, 10, -5 !for each pressure height
347         PkPa = real(i) !pressure (kPa)
348         TK = thetaK*( PkPa/Po)**RdCp ) !temperature (K)
349         TC = TK - 273. !temperature (C)
350         write(*,fmt) PkPa, tab, TC !debugging output
351
352         y = yst(PkPa) !y coordinate of skewT
353         x = xst(TC,y) !x coordinate of skewT
354
355         if ((x .ge. -40.) .and. (x .le. 40.)) then !inside skewT domain
356             write(2,fmt) x,"",y !x & y coord, write to output
357         endif
358
359     enddo !end of pressure height loop
360 enddo !end of adiabat loop
361
362 write(2,*)
363 write(*,*)
364 end subroutine plotdryadiabat
365
366
367 !=====
368 subroutine plotwetadiabat !plot a few saturated adiabats in backgr
369     use soundmod !include sounding data
370     implicit none !enforce strong typing
371     integer :: j !thetaL index
372     integer :: i !pressure (height) index
373     real, parameter :: dP = 0.10000 !pressure increment delta P (kPa)
374     real, parameter :: a = 0.28571 !dimensionless const for sat adiabat
375     real, parameter :: b = 1.35E7 !(K^2) constant for sat adiabat
376     real, parameter :: c = 2488.4 !(K) constant for saturated adiabat
377     real, parameter :: LR = 5423. !Lv/Rv (K) in Clausius-Clapeyron eq
378     real, parameter :: eo = 0.611 !(kPa) reference vapor pressure
379     real, parameter :: Toinv = 1./273. !inverse of T (K) const in Clausius Clap eq
380     real, parameter :: eps = 0.622 !(g/g) epsilon = ratio of gas constants
381     real :: TC, TK !temperature of rising air parcel (C) and (K)
382     real :: es !(kPa) saturated vapor pressure
383     real :: rs !(g/g) saturated mixing ratio
384     real :: PkPa !pressure (kPa)
385     real :: dTdP !(K/kPa) change of T with P along sat adiabat
386     real :: num, den !temporary numerator & denominator of dT/dP eq.
387     real :: xst, yst !type declaration for functions
388     real :: x, y !temporary coordinates in skewT
389     character(len=30) :: fmt !string holding output format
390
391     write(*,*) "...starting plotwetadiabat" !debugging output
392     write(*,*)
393     write(2,*) !skip line in output file
394     write(2,*) !skip line in output file
395
396     fmt = "(F12.4, a1, F12.4)" !format for output

```

[illegible]


```

463     do j=1,4                                !for each humidity
464         write(*,*)                          !debugging output
465         write(*,"(a,F8.2)") "r (g/kg) = ",r(j)    !debugging output
466         write(*,*) "      P(kPa) ",tab,"      Td(C)" !debugging output
467         rg = 0.001 * r(j)                    !convert mixing ratio to (g/g)
468
469         do i = 1,5                            !for each pressure
470             lnarg = (rg*PkPa(i)) / (eo*(rg+eps))    !argument of ln in eq (3)
471             sbarg = (Toinv - (RvLv*log(lnarg)))      !argument in square brackets in (3)
472             TdC = (1./sbarg) - 273.                !dew point (C), from eq (3)
473             write(*,fmt) PkPa(i), tab, TdC         !debugging output
474
475             y = yst(PkPa(i))                      !Y coordinate on skew T
476             x = xst(TdC,y)                        !X coordinate on skew T
477
478             if (x .le. 40.) write(2,fmt) x,"",y    !write coord if inside skewT domain
479         enddo
480
481         write(2,*)                                !blank line between each isohume data
482     enddo
483
484     write(*,*)
485 end subroutine plotisohume
486
487
488 !=====
489 subroutine cleanup                                !close files and release memory
490     use soundmod                                  !including sounding info
491     implicit none                                  !enforce strong typing
492     write(*,*) "...starting cleanup"              !debugging output
493     write(*,*)
494
495     write(*,*) title                              !remind user of title
496     write(*,*) "Your sounding output file is: ",outname
497     close(2)                                       !close the output file
498
499     write(*,*)
500     write(*,*) "Bye."
501     write(*,*)
502 end subroutine cleanup
503
504
505 !=====
506 real function yst (P)                             !finds ordinate (y) value on skewT
507     implicit none                                  !enforce strong typing
508     real, intent(in) :: P                        !pressure (kPa)
509     real, parameter :: Po = 100.0                !reference pressure (kPa)
510     yst = log(Po/P)                              !ordinate for skewT - logP
511 end function yst
512
513
514 !=====
515 real function xst (T,y)                           !finds abscissa (x) value on skewT
516     implicit none                                  !enforce strong typing
517     real, intent(in) :: T                        !temperature (C)
518     real, intent(in) :: y                        !ordinate (dimensionless)
519     real, parameter :: K = 35.0                  !reference temperature (C)
520     xst = T + K*y                                !abscissa for skewT - logP
521 end function xst
522
523
524 !===== end ==
525
526

```