The Textual Standard File Format (TSF)

Yves Chartier
Section informatique
Recherche en Prévision Numérique
Atmospheric Environment Service
Environment Canada

Last Revision: Nov. 5, 1993

Introduction

This document gives an overview of the Textual Standard File Data Format (from now on, TSF), which has been defined by the author and Michel Valin in the summer of 1993. This file format is used to store gridded data from numerical weather prediction models, objective analyses and geophysical fields. As its name implies, it is designed to provide full portability of data among various platforms.

TSF files can be considered as an extension of RPN standard files[1], in the sense that they contain the same information. However, the structure of the files is totally different. TSF files are formatted sequential files, whereas RPN standard files are unformatted random access files.

Historical perspective

The software used with RPN standard files has always been difficult to port. One important reason is that I/O with random access files comes with flavors that differ from vendor to vendor. Another reason is that the RPN standard file software uses sophisticated compression algorithms coded with FORTRAN bit manipulation intrinsics, which are not part of the standard language and differ also from vendor to vendor.

The advent of the MC2 project (Mesoscale Compressible Community) arose the need to facilitate the exportation of the data. TSF provides some of the answers to this need.

General structure

A TSF file is a sequential ASCII file. It is a file that can be inspected with a text editor, and that can be manipulated with standard UNIX facilities (more, cat, head, tail, grep, awk, sed). A TSF file may contain 1 or more data records. A TSF record can be divided in two parts: the **info** part and the **data** part.

The TSF info part

The info part is a FORTRAN namelist containing the equivalent of most RPN standard file attributes that can be accessed through the FORTRAN function FSTPRM, plus a few other items like the physical units in which the variable is stored.

The TSF file attributes can be roughly divided in 5 categories, which are:

Field identification attributes
- variable name
- nature of field (analysis, forecast, climatological field...)
- stamp (model identification)
- physical units

Time attributes
- date of original analysis
- length of time step used in model integration (0 if analysis)
- time step number
- forecast hour (e.g. 12 hour forecast)
- end forecast hour, if the field represents a variation over a given period

[1] See "An introduction to RPN standard files", from the author

vertical coordinate
type of geographical projection
parameters describing the geographical projection

Internal representation attributes

data type (real, integer)
arithmetic base used to represent the data (10 or 90)
encoding format (FORTRAN FORMAT used to write the data (base 10))
compression information (number of digits used per number (base 90)
min and max values of the field (base 90)

The following is a list of the attributes found in TSF files, classified by category.

Field identification attributes

Data element	Data type	Name	Range
Variable name	CHARACTER*48	VARIABLE	
Type of field	CHARACTER*16	NATURE	
Stamp	CHARACTER*16	STAMP	
Physical units	CHARACTER*16	UNITS	

Time attributes

Data element	Data type	Name	Range
Date of orig. analysis	INTEGER	DATE	YYYYMODD.HHMMSS
Forecast hour	INTEGER	TIME	INTEGER*4
End forecast hour	INTEGER	TIME2	INTEGER*4
Length of time step	INTEGER	TIMESTEP	INTEGER*4
Time step number	INTEGER	STEPNO	INTEGER*4

Spatial attributes

Data element	Data type	Name	Range
Vertical level	REAL	LEVEL	REAL*4
Second vertical level	REAL	LEVEL2	REAL*4
Vertical coordinate	CHARACTER*16	VERTCOORD	CHARACTER*16
# of points along X	INTEGER	NI	INTEGER*4
# of points along Y	INTEGER	NJ	INTEGER*4
# of points along Z	INTEGER	NK	INTEGER*4
Type of geographical projection	CHARACTER*16	MAPPROJ	
X position of the pole	REAL	XPOLE	REAL*4
Y position of the pole	REAL	YPOLE	REAL*4
Grid Length	REAL	MESHPS	REAL*4
Grid Rotation	REAL	MAPROT	REAL*4
Lat. of sw corner of grid	REAL	XPOLE	REAL*4
Lon. of sw corner of grid	REAL	YPOLE	REAL*4
Grid Length (deg. lat)	REAL	MESHPS	REAL*4
Grid Length (deg. lon)	REAL	MAPROT	REAL*4

Internal representation attributes

Data element	Data type	Name	Range
base used to encode data	INTEGER	BASE	10 or 90
format (base 10)	CHARACTER*16	FORM	
# of digits (base 90)	INTEGER	DIGITS	1-4
Min. value of the field	INTEGER	MIN	REAL
Max. value of the field	INTEGER	MAX	REAL

Original RPN standard file attributes

Data element	Data type	Name	Range
Data type	INTEGER	DATYP	0-5
# of bits	INTEGER	NBITS	1-32
IP1	INTEGER	IPDESC1	INTEGER*4
IP2	INTEGER	IPDESC2	INTEGER*4
IP3	INTEGER	IPDESC3	INTEGER*4
IG1	INTEGER	MAPDESC1	INTEGER*4
IG2	INTEGER	MAPDESC2	INTEGER*4
IG3	INTEGER	MAPDESC3	INTEGER*4
IG4	INTEGER	MAPDESC4	INTEGER*4

Detailed description of TSF file attributes

Variable name (VARIABLE):
VARIABLE is a 48 character symbol representing a meteorological parameter. The first two characters are normally the same as the NOMVAR attribute used in RPN standard files. Some examples:

```
VARIABLE = 'PN (Sea level pressure)',
VARIABLE = 'GZ (Height)'
VARIABLE = 'TT (Air temperature)'
VARIABLE = 'HR (Relative humidity)'
```

Nature of field (NATURE):
NATURE is a 16 character symbol representing the origin of the data. The first character is normally the same as the TYPVAR attributes used in RPN standard files.
Some examples:

```
NATURE = 'A (Analysis)'
NATURE = 'C (Climatology)'
NATURE = 'P (Forecast)'
```

Label (STAMP):
This is a 16 character symbol whose contents is normally left to the discretion of the user. It can be used to identify a numerical model, or the code of an experiment. For example:

```
STAMP = 'MC2_V3.0'
```

Physical units (UNITS):
This 16-character symbols defines the units in which the field is stored. This attribute is an extension of the RPN standard file format.

Vertical level (LEVEL):
This is a real symbol that represents a level in a given vertical coordinate.

Second vertical level (LEVEL2):
In most situations the value of this integer attribute will be the same as LEVEL. It happens however that one may want to store the vertical variation of a field instead of its ponctual value (ex. the 1000-500mb height thickness). In this case only then will LEVEL2 not the same as LEVEL.

Vertical coordinate (VERTCOORD):
This a 16 character symbol that represents the vertical coordinate on which the field is defined. This attribute is an extension of the RPN standard file format. Some examples:

```
VERTCOORD = 'PRESSURE'
VERTCOORD = 'SIGMA'
VERTCOORD = 'GAL-CHEN'
```

Forecast hour (TIME):
This integer attribute normally represents the time elapsed since the date of origin (in hours). Its value should normally be rounded to the nearest hour as given by the FORTRAN formula:

TIME = ((STEPNO * TIMESTEP+1800)/3600).

End forecast hour (TIME2):
In most situations the value of this integer attribute will be the same as TIME. It happens however that one may want to store the temporal variation of a field instead of its instantaneous value (ex. the accumulation of precipitation during a period of 6 hours). In this case only then will TIME2 ≠ TIME.

Length of time step (TIMESTEP):
This is the length of a time step used during a model integration, in seconds.

Time step number (STEPNO):
This is the time step number at which the field was written during an integration. The number of the initial time step is 0.

Date of original analysis (DATE):
This 16 character attribute represents the date of origin of a field. The encoding format is 'YYYYMODD.HHMMSS' where

YYYY:	year (0000-9999)
MO:	month (01-12)
DD:	day (01-31)
HH:	hour (00-23)
MM:	minute (00-59)
SS:	second (00-59)

Dimension of grid along the X, Y and Z axes (NI, NJ, NK):
This is the physical dimension of the grid along each spatial axis. On a geographical map, NI lies along the horizontal or X axis, NJ along the vertical or Y axis, and NK would point out of the map or along the Z axis.

Type of geographical projection (MAPPROJ) and grid parameters (XPOLE, YPOLE, MESHPS, MAPROT, SWLAT,SWLON,MESHLAT,MESHLON):
This 16 character symbol represents the geographical projection over which the current field is defined. The first character of the field is the one-letter code used in RPN standard files. The projections currently supported by TSF are 'N', 'S' and 'L'. 'N' and 'S' grids are polar stereographic grids (North and South respectively). 'L' grids are lat-lon grids. Other grid types supported by RPN (such as 'G (Gaussian), 'A (Global latlon)' and 'B (Global latlon)') can also be represented but the values of the grid descriptors are irrelevant.

For '**N**' and '**S**' grids the following REAL parameters are defined:

XPOLE:	X position of the pole on the grid (in grid points)
YPOLE:	Y position of the pole on the grid (in grid points)
MESHPS:	grid length valid at 60° North (in meters)
MAPROT:	angle between the Greenwich meridian and the X axis

For '**L**' grids the following REAL parameters are defined:

SWLAT:	latitude of the lower left corner of the grid
SWLON:	longitude of the lower left corner of the grid
MESHLAT:	latitudinal grid length (in degrees)
MESHLON:	longitudinal grid length (in degrees)

For 'N' and 'S' grids the values of SWLAT, SWLON, MESHLAT and MESHLON are irrelevant, and so are XPOLE, YPOLE, MESHPS and MAPROT for 'L' grids.

Arithmetic base used to represent the data (BASE):
This attribute indicates whether the data part is represented in plain format (base 10) or in compressed format (base 90). A detailed description of the two representations is given in the next section (The TSF data part).

Number of digits per value (DIGITS):
This attribute gives the number of digits kept per value, and may vary from 1 to 4. This number indicates the relative precision of the data. When 1 digit per value is used, the relative error on a data item is about 1.1% (1/90), and decreases to 0.0000015% (1/65610000) when 4 digits are used.

Encoding format (FORM):
This attribute is the FORTRAN format that was used to write the data to disk. It may take any valid FORTRAN format. Default value is '(5g14.5)'. This attribute is passed directly to the FORTRAN READ or WRITE command in the following manner:

```
write(iun,form)(((fld(i,j,k),i=1,ni),j=1,nj),k=1,nk)
```

Minimum value of the field (MIN):
This attribute gives the minimum value of the field. It is used to encode and decode a field compressed in base 90.

Maximum value of the field (MAX):
This attribute gives the maximum value of the field. It is also used to encode and decode a field compressed in base 90.

Original RPN standard file attributes (DATYP, NBITS, IP1DESC1, IPDESC2, IPDESC3, MAPDESC1, MAPDESC2, MAPDESC3, MAPDESC4):
These integers attributes are kept to maintain consistency with the original RPN standard file attributes. The user does not need to be concerned about the value of these arguments.

The TSF data part

The data part, as its name implies, contains the values of the field described by the info part. That data may appear either in plain (base 10) or compressed format (base 90). The data encoded in plain format can be interpreted directly, since it appears as a stream of numbers. The data is written to disk using the following command:

```
write(iun,form)(((fld(i,j,k),i=1,ni),j=1,nj),k=1,nk)
```

where **form** is a valid FORTRAN format.

An example of a 120x60 field written with a "(5g14.5)" FORMAT can be found in Appendix A.

Although very convenient, the plain format is very costly in terms of storage (disk space). The default format uses 14 digits (i.e. bytes) per value; a compressed format such as the one used in RPN standard files, only 2 bytes. The plain format takes about 7 times more storage space than necessary.

One way to compact the data while preserving pure ASCII encoding is to use an arithmetic base higher than 10, composed of ASCII printable characters. The base 90 offers a good compromise. In base 90, the range of allowable values per digit goes from '!' (0 - ascii code 33) to 'z' (89 - ascii code 122). TSF supports numbers composed of up to 4 digits, from 0 '!!!!' to 'zzzz' (65,609,999).

The original values of a field are not directly translated into base 90; only the variation of the values over a range given by (max - min). For example suppose that we encode a surface temperature field, with temperatures ranging between -50°C and +50 °C. In a 2 digits base 90 representation, -50 would be encoded '!!', +50, 'zz', +25, 'dM' (+25 - -50)/(+50 - -50)*8099. The round-off error done on any value in that range is less than 0.02 °C.

It is worth to mention that using the UNIX utility 'compress' on a TSF file encoded in base 90 does not yield much reduction (only 10 to 15 %) in file size, and consumes a lot of CPU time.

An example of a 120x60 field written in base 90 can be found in Appendix A..

The TSF library

The creation and processing of TSF files can be done either by calling a set of FORTRAN integer functions (the TSF library) or by using utilities such as TSFCAT. The TSF library contains a set of 45 FORTRAN routines, of which 6 are of interest to the user.

write records
- TSFECR (writes a TSF record)

read and query records
- TSFLIRP (reads the info part of a TSF record)
- TSFLIRFD (reads the data part of a TSF record)
- TSFLIR (reads the info and data part of a TSF record)

print diagnostics
- TSFVERB (prints a diagnostic message when reading or writing a record)
- TSFQUIET (does not print a diagnostic message when reading or writing a record)

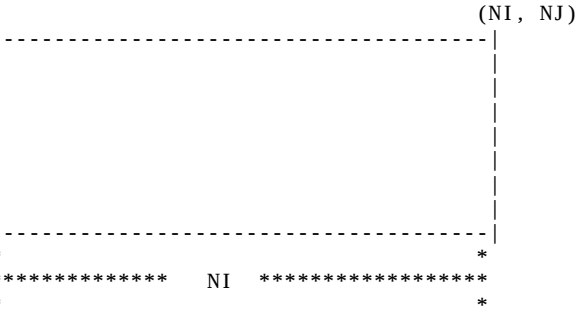
A basic example: creating a TSF file from ASCII data

Th example discussed in this section should be accessible on-line, in the directory $ARMNLIB/demo/tsf. To start working on your own, you can make a copy of this directory in your own HOME directory. Here is a suggested list of commands:

```
% cd
% mkdir tsf
% cp $ARMNLIB/demo/tsf/* tsf
```

The following example covers the conversion of data stored in ASCII format into the TSF format. The file contains climatological monthly surface temperatures on a 120x60 grid, covering the globe and defined on a lat-lon projection. The data is stored in file "ts.asc" and will be converted into TSF file "ts.tsf".

The grid has the following structure: point (1,1) represents the southwest corner of the globe (-88.5 lat, 0 lon), while point (ni,nj) represents the northeast corner (88.5 lat, 357 lon). There is one grid point every 3 degrees of latitude and longitude.



The input file has the following format: the first line contains the month of the year, followed by field values encoded in the following order: i, j, fld(i,j), fld(i+1,j), fld(i+2,j), fld(i+3,j), fld(i+4,j). Here are the first ten lines of the file.

```
          1
  1   1 -28.6152 -28.6816 -28.6015 -28.6230 -28.8769
  6   1 -28.8496 -28.7461 -28.6621 -28.5722 -28.7051
 11   1 -28.5605 -28.5742 -28.4902 -28.2500 -28.3613
 16   1 -28.1113 -28.2285 -28.2441 -28.1621 -28.2715
 21   1 -28.2070 -28.1133 -28.0410 -27.9941 -28.0879
 26   1 -28.1465 -28.2676 -28.3437 -28.3730 -28.4004
 31   1 -28.4394 -28.5703 -28.7109 -28.8008 -28.6797
 36   1 -28.5703 -28.5156 -28.4394 -28.3730 -28.4785
 41   1 -28.5664 -28.5410 -28.6367 -28.7871 -28.5468
```

The FORTRAN code is stored in the file "ex1tsf.f". A copy of the program appears in Appendix A. The "ex1tsf" program uses 2 routines from the TSF library. Here is an overview of what is done by the program.

Declare variables
Initialize proper TSF attributes
Read the data contained in the ASCII file
Write records into the TSF file (TSFECR)

Let's look more closely at the different parts of the program.

Declare variables

In the first part we use the attributes names that were suggested in the "Detailed structure" section.

```
-------------------------------------------------
integer ni,nj,nk,timestep,stepno,time,time2,base,digits
integer extra(1),nextra
character*48 variable
character*16 nature,stamp,vertcoord,mapproj,form
character*16 date,units
real level,level2,xpole,ypole,meshps,maprot,
$     meshlat,meshlon,swlat,swlon,min,max

integer tsfecr
external tsfecr
-------------------------------------------------
```

We continue with the other variables. "ier" contains the return status of the functions used in the examples. "iun" contains the logical FORTRAN unit. "fld" will contain the values of surface temperature for each month.

```
-------------------------------------------------
integer ier
integer i,j,n,ii,jj,iun
integer month
real fld(120, 60)
-------------------------------------------------
```

Initialize proper TSF attributes

```
------------------------------------------------
variable = 'TS (Surface Temperature)'
nature   = 'C (Climatology)'
stamp    = 'TSF_DEMO'
units    = 'DEGREES CELSIUS'

date = '19930101.000000'
time  = 0
time2 = 0
timestep = 0
stepno = 0

level = 0.0
level2= 0.0
vertcoord = 'SURFACE'

ni = 120
nj = 60
nk = 1

mapproj = 'L(latlon)'
form  = '*               '
xpole = 0.0
ypole = 0.0
meshps = 0.0
maprot = 0.0

swlat = -88.5
swlon = 0.0
meshlon = 3.0
meshlat = 3.0

base = 90
digits = 2
------------------------------------------------
```

Read the data contained in the ASCII file

We assume here that we are reading data from the console (through standard UNIX redirection) and that we know exactly the format of the input data.

```
------------------------------------------------
  read(5, *) month
   do 200 j=1,120*60/5
      read(5,*) ii,jj,fld(ii,jj),fld(ii+1,jj),fld(ii+2,jj),
*               fld(ii+3,jj),fld(ii+4,jj)
200       continue
------------------------------------------------
```

Write records in the standard file (TSFECR)

We set the date so that each climatological average has the 1st of each month as date of origin. We then call integer function TSFECR with the attributes that were set at the beginning of the program.

```
------------------------------------------------
 ier = tsfecr(fld,iun,variable,nature,stamp,
$       ni,nj,nk,ni,nj,nk,level,level2,vertcoord,
$       date,timestep,stepno,time,time2,
$       mapproj,xpole,ypole,meshps,maprot,
$       meshlat,meshlon,swlat,swlon,
$       min,max,base,form,digits,units,extra,nextra)
------------------------------------------------
```

Just before the end of the loop increase the date of validity by 1 month

```
-------------------------------------------------
    call incdatss(date,date,'00000100.000000')
-------------------------------------------------
```

To compile the program, the environment variable $TSFLIB has to be defined.
For sites where the RMNLIB library is installed, type

```
% setenv TSFLIB $ARMNLIB/lib/rmnxlib.a
```

For sites where the MC2 model is installed, type

```
% setenv TSFLIB $mc2/lib/$ARCH/tools_v3.0.a
```

Finally, to compile the program, type

```
% f77 ex1tsf.f util.f -o ex1tsf $TSFLIB
```

To execute the program, type

```
% ex1tsf < ts.asc
```

The program should then execute, producing the following output:

```
*****WRITE*************************************************************************************************************************
Variable=TS (Surface Temperature)
Nature  =C (Climatology)   |Stamp   =TSF_DEMO          |Units   =DEGREES CELSIUS   |
Vertcord=SURFACE           |Level   =0.0000E+00        |Level2  =0.0000E+00        |                           |
NI= 120  NJ=  60  NK=  1   |Min     = -48.256          |Max     =  31.516          |                           |
date    =19930101.000000   |timestep=     0          |stepno  =     0            |time    =    0             |time2   =    0
Mapproj =L(latlon)         |meshlat =   3.00           |meshlon =   3.00           |swlat   = -88.50           |swlon   =   0.00
Base    =90                |Format  =*                 |digits  =2                 |                           |
*****WRITE*************************************************************************************************************************
Variable=TS (Surface Temperature)
Nature  =C (Climatology)   |Stamp   =TSF_DEMO          |Units   =DEGREES CELSIUS   |
Vertcord=SURFACE           |Level   =0.0000E+00        |Level2  =0.0000E+00        |                           |
NI= 120  NJ=  60  NK=  1   |Min     = -49.159          |Max     =  30.910          |                           |
date    =19930201.000000   |timestep=     0          |stepno  =     0            |time    =    0             |time2   =    0
Mapproj =L(latlon)         |meshlat =   3.00           |meshlon =   3.00           |swlat   = -88.50           |swlon   =   0.00
Base    =90                |Format  =*                 |digits  =2                 |                           |
*****WRITE*************************************************************************************************************************
Variable=TS (Surface Temperature)
Nature  =C (Climatology)   |Stamp   =TSF_DEMO          |Units   =DEGREES CELSIUS   |
Vertcord=SURFACE           |Level   =0.0000E+00        |Level2  =0.0000E+00        |                           |
NI= 120  NJ=  60  NK=  1   |Min     = -63.803          |Max     =  30.920          |                           |
date    =19930301.000000   |timestep=     0          |stepno  =     0            |time    =    0             |time2   =    0
Mapproj =L(latlon)         |meshlat =   3.00           |meshlon =   3.00           |swlat   = -88.50           |swlon   =   0.00
Base    =90                |Format  =*                 |digits  =2                 |                           |

( ... )

*****WRITE*************************************************************************************************************************
Variable=TS (Surface Temperature)
Nature  =C (Climatology)   |Stamp   =TSF_DEMO          |Units   =DEGREES CELSIUS   |
Vertcord=SURFACE           |Level   =0.0000E+00        |Level2  =0.0000E+00        |                           |
NI= 120  NJ=  60  NK=  1   |Min     = -49.999          |Max     =  30.960          |                           |
date    =19931101.000000   |timestep=     0          |stepno  =     0            |time    =    0             |time2   =    0
Mapproj =L(latlon)         |meshlat =   3.00           |meshlon =   3.00           |swlat   = -88.50           |swlon   =   0.00
Base    =90                |Format  =*                 |digits  =2                 |                           |
*****WRITE*************************************************************************************************************************
Variable=TS (Surface Temperature)
Nature  =C (Climatology)   |Stamp   =TSF_DEMO          |Units   =DEGREES CELSIUS   |
Vertcord=SURFACE           |Level   =0.0000E+00        |Level2  =0.0000E+00        |                           |
NI= 120  NJ=  60  NK=  1   |Min     = -45.602          |Max     =  31.209          |                           |
date    =19931201.000000   |timestep=     0          |stepno  =     0            |time    =    0             |time2   =    0
Mapproj =L(latlon)         |meshlat =   3.00           |meshlon =   3.00           |swlat   = -88.50           |swlon   =   0.00
Base    =90                |Format  =*                 |digits  =2                 |                           |
```

The TSF utilities

Three programs come with the TSF library: **tsfcat**, **fst2tsf** and **tsf2fst**. **tsfcat** is an utility used to compress, decompress or recode a TSF file. **fst2tsf** is used to convert RPN standard files to the TSF format. **tsf2fst** is used to convert TSF files to the RPN standard file format. **tsfcat** is designed to be used on most UNIX systems. **tsf2fst** and **fst2tsf** can only be used on platforms where the RPN standard file library is supported.

The user is invited to consult the man pages relevant to these utilities. We will however take some time to discuss the usage of tsfcat.

By default, tsfcat decompresses a TSF file, and prints the values of the data part in the default (5g14.5) FORTRAN format.

```
START_DATA
   -28.6158      -28.6847      -28.6059      -28.6256      -28.8817
   -28.8522      -28.7438      -28.6650      -28.5764      -28.7044
   -28.5567      -28.5764      -28.4878      -28.2514      -28.3597
   -28.1135      -28.2317      -28.2415      -28.1627      -28.2711
   -28.2021      -28.1135      -28.0445      -27.9953      -28.0839
   -28.1430      -28.2711      -28.3400      -28.3696      -28.3991
   -28.4385      -28.5665      -28.7143      -28.8029      -28.6749
   -28.5665      -28.5173      -28.4385      -28.3696      -28.4779
   -28.5665      -28.5370      -28.6355      -28.7832      -28.5469
   -28.1529      -28.2514      -28.2809      -28.1627      -28.1824
( ... )
   -36.1015      -36.1212      -36.1409      -36.1507      -36.1704
   -36.1901      -36.2098      -36.2295      -36.2492      -36.2590
   -36.2787      -36.2787      -36.2787      -36.2787      -36.2689
   -36.2492      -36.2295      -36.2098      -36.1704      -36.1409
   -36.1212      -36.0522      -35.9340      -35.6976      -35.5006
   -35.3233      -35.2347      -35.1953      -35.2839      -35.3627
   -35.4514      -35.3923      -35.3233      -35.2347      -35.0574
   -34.8309      -34.6930      -34.5551      -34.4369      -34.4566
   -34.5452      -34.6930      -34.8112      -34.8506      -34.7915
   -34.6930      -34.6536      -34.5846      -34.4369      -34.3876
END_DATA
```

The format used to write the data in plain format can also be modified using the **-format** option.

```
tsfcat -format "'(25f5.0)'" < test.tsf
```

```
START_DATA
 -29. -29. -29. -29. -29. -29. -29. -29. -29. -29. -29. -28. -28. -28. -28. -28. -28. -28. -28. -28. -28. -28. -28. -28. -28.
 -28. -28. -28. -28. -28. -28. -29. -29. -29. -29. -29. -29. -28. -28. -28. -29. -29. -29. -29. -29. -28. -28. -28. -28. -28.
 -28. -28. -28. -28. -28. -29. -30. -30. -30. -30. -30. -30. -31. -31. -31. -32. -32. -32. -32. -31. -31. -31. -31. -31. -31.
 -31. -31. -31. -30. -30. -30. -30. -29. -29. -29. -29. -29. -29. -29. -30. -30. -31. -32. -33. -33. -33. -33. -33. -32. -32.
 -32. -31. -31. -31. -31. -31. -30. -30. -30. -29. -29. -29. -28. -28. -28. -27. -28. -28. -28. -28. -29. -29. -29. -29. -30.
 -30. -30. -30. -29. -29. -29. -30. -30. -30. -30. -30. -31. -31. -31. -31. -31. -31. -31. -31. -31. -31. -31. -31. -31. -32.
 -32. -32. -32. -32. -32. -31. -31. -30. -30. -30. -30. -29. -29. -29. -28. -27. -27. -27. -27. -26. -26. -25. -23. -23. -24.
 -25. -25. -28. -30. -29. -29. -27. -29. -29. -28. -27. -22. -20. -22. -20. -20. -21. -21. -22. -23. -23. -23. -24. -24. -26.
 -25. -25. -24. -24. -24. -24. -24. -24. -24. -25. -24. -24. -23. -23. -23. -24. -26. -27. -28. -29. -28. -27. -26. -26. -26.
 -27. -27. -26. -26. -26. -26. -26. -27. -26. -27. -26. -27. -28. -29. -29. -30. -30. -31. -30. -31. -31. -31. -32. -32. -32.

( ... )

 -30. -29. -29. -29. -29. -29. -27. -26. -24. -24. -31. -31. -30. -30. -30. -30. -29. -29. -29. -29. -29. -29. -29. -30. -30.
 -30. -30. -30. -31. -31. -31. -32. -32. -33. -33. -33. -34. -34. -34. -34. -34. -35. -35. -35. -35. -36. -36. -36. -36. -36.
 -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36.
 -35. -35. -35. -35. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36.
 -36. -36. -35. -35. -34. -33. -33. -33. -33. -34. -34. -34. -33. -33. -33. -32. -32. -31. -31. -31. -31. -32. -32. -32. -32.
 -32. -31. -31. -31. -31. -34. -34. -34. -34. -34. -34. -34. -34. -34. -34. -34. -34. -34. -34. -34. -34. -34. -34. -34. -34.
 -35. -35. -35. -35. -35. -35. -35. -35. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36.
 -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36.
 -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36. -36.
 -35. -35. -35. -35. -35. -35. -35. -35. -35. -35. -35. -35. -35. -34. -34. -35. -35. -35. -35. -35. -35. -35. -35. -34. -34.
END_DATA
```

tsfcat can also be used to compress a TSF file written in plain format.

```
tsfcat -c -digits 1 < ts.tsf
```

```
START_DATA
77777777777777777777788877777777777777777777777778877655555544334344554445555566677766654322223333444445566677788887776666566666555555444544444444444334455666677899999;=<;;867786689>@>AA?@?===<<:;;<<<<<<=;<<===<:98789::989999::99998777554544444433221 00/.....-.//011112333334556778 8;<=>>>CEHGHJLLLLKKKJJIHHFFFFFEEEEEEDBA@><<<===>@???ABCCCCBBDDC@AAA@><<;97664433333333221101000/0000122222222223344556679:<>???GLOVULLLLLLLLLLLKKJIHGHHGFDDDCBA@?>?=>>>@CEEEFFFGGFGFFGKJKKLHEB><97543433444554443555555458:87776654221133344456 79:<===>CGJSKVUUUUUUUVVSLLKJMIECBBACCCEEFEFFGFFFHHHHDEFHKNLLKLLLKTROJFDA;84399666778 99;;:::;::::;<=CHDB@=<:8665567666667889:<=>>@CGILNRUUUUUUKLLLLLLLLLLKKSKONVTVLQPNJIJIPRRLPHBFKLLLLLLLLLVUUVSLID=9UTSRUQRTUTSQCOIGFEBCCFKNKPMIE@>=<===<<<( ... )CE@<0-*+238>EIKOWXYZWXYZ[\[Z[[ZZYXWWVQHFHFECA?=;:988777666555554434333233345677889:;;:::;:::99876555334443232435456 49<@BC?5,)'%%)+.9?DFHKKRVWXXXNLLLKKKKKJHHHGFDBBA?><;;::998777877776655544333334444556777678887766543322221111221213236 8;;:70+*('&')+-09>ABEGILFEUUDGFGFHHHHGGFFEEDCCBA@?>=<;;:878876554332222111112222222333444444443322111100000000000--12214389630.-+++-./26889<>ACE==>>??@@?????>>>>===<;;:99877665554332211100000000000000000011111112211100000/////////////010000*+-1045556553344566776 9:<=555666666776655555443332211111000/////////.....////////.....//////011222111122344554433344445011111111111111111100000000/////////////.....////////////////.....///////00/////00000000000000000END_DATA
```