UNITED STATES DEPARTMENT OF THE INTERIOR

GEOLOGICAL SURVEY

Forward and inverse cartographic projection procedures

by

Gerald I. Evenden

Open-File Report ~~83-523~~
83-625

This report is preliminary and has not been reviewed for conformity with U.S. Geological Survey editorial standards and stratigraphic nomenclature.

1. Woods Hole, Mass.

1983

Contents Page

Abstract 1

Introduction 2

Procedure usage 4

Implementation and installation 5

Conclusions 7

References 8

Appendix A. - Projection list and 'consts' definition A-1

B. - Macro symbol file B-1

C. - Listing of projection routines C-1

D. - Listing of projection support library D-1

E. - Test execution results E-1

F. - Example usage program F-1

G. - Source code distribution G-1

Abstract

A set of procedures for performing 19 common cartographic transformations between geographic and cartesian coordinate systens are presented along with instruction for usage, installation and testing results. These procedures are written in the RATFOR preprocessor language and effort was made to ensure transportablility of the code as well as consistency and flexibility of usage.

Disclaimer

Although these procedures have been subject to many tests and considerable usage, a warrantee on proper functioning is neither implied nor expressed.

Introduction

The original purpose of developing this software was for a map graphics system which requires a set of procedures which will transform (or "project") geographic coordinates into the cartesian system required by graphic devices. The effort to design a comprehensive and flexible set of projections is a very substantial task which was facilitated by recent efforts of Snyder (1982) in providing mathematical development of many common projections and A. Elassal's (personal communications) set of FORTRAN procedures.

Because the source language for the MAPGEN system is RATFOR (Kernighan and Plauger, 1976) it was necessary, for consistency, to rewrite the original code kindly provided by A. Elassal. Although a great deal of the effort could be performed with global editing functions, additional manual editing was required in order to restructure the code and remove nonstandard FORTRAN constructs. After conversion, cross checks were made between runs of each version. Although the inverse projections are not involved in the MAPGEN requirements, they are included in the conversion as they may prove useful in other applications and provide a convenient cross check (errors in one process will show up when the inverse process does not reproduce the original input). When secondary sources of projection data were available, performance of the projection routines was cross checked with these data.

Currently, 19 projections have been included in this report:

01 - Universal Transverse Mercator (UTM),
03 - Alber's conic equal area,
04 - Lambert conformal conic,
05 - Mercator,
06 - Polar stereographic,
07 - Polyconic,
08 - Equidistant Conic,
09 - Transverse mercator,
10 - Stereographic,
11 - Lambert azimuthal equal-area,
12 - Azimuthal equidistant.
13 - Gnomonic,
14 - Orthographic,
15 - General vertical near-sided perspective,
16 - Sinusoidal,
17 - Equirectangular,
18 - Miller Cylindrical,
19 - Van Der Grinten I, and
20 - Oblique mercator.

Because of the FORTRAN(66) identifier length limitation of six characters, no attempt was made to employ acronymic procedure names related to the projection names. Instead, the arbitrary numbers listed above are employed when referring to a particular projection.

This report presents the programmatic aspects of using the projection procedures and implementation recommendations. The cartographic aspects of usage are inappropriate in this report. Mapping (Greenhood, 1964) is an excellent reference for a layman's overview of cartography, and Snyder (1982) provides a more detailed discussion of the projections.

All of the source code discussed in this paper is also available on magnetic tape media (see Appendix G).

Procedure usage.

Each of the projection procedures has several elements defining attributes of the particular projection (e.g. spheroid radius, false easting and northing, etc.) and, in most cases, some preliminary calculations based on these parameters can be made in order to eliminate redundant processing. To preserve the widest flexibility of usage, these parameters and any preliminary constants are stored in a real array that is passed as an argument in each procedure call. The definition of projection attributes is the responsibility of the external calling program. Appendix A summarizes the initialization requirements of the constants array for each of the projections.

After these attributes have been assigned to the appropriate locations in the constants array, the initialization procedure for the selected projection is executed as follows:

```
integer err, set XX
PREC consts(MAXCONSTS)
     ...
     ...  # constant assignments
     ...
     err = set XX(consts)
      ...
```

The upper case identifiers always denote macros, and in this case, 'XX' would have been previously defined as one of the projection numbers (e.g. define(XX, 01) for UTM). A discussion of the other macros will be made later. Note that the initialization procedure as well as the projection procedures are all integer functions where the returned value will either be 'OK' for successful execution or 'ERR' if an error is detected (both OK and ERR are standard RATFOR macros). In the latter case, a

call is made to 'remark' with a one line description of the error.

Either the forward ('for' identifier prefix) or inverse ('inv' prefix) transformation procedure can now be executed:

```
integer err, for XX, inv XX
PREC consts(MAXCONSTS), lam in, lam out, phi in, phi out,
    x in, x out, y in, y out
      ....
     err = for XX(lam in, phi in, x out, y out, consts)
      ....
     err = inv XX(x in, yin, lam out, phi out, consts)
      ....
```

The 'lam' and 'phi' identifier prefixs are respective longitude and latitude geographic coordinates in radians. Identifiers prefixed with 'x' and 'y' are the cartesian coordinates whose units are defined by the initialization of the 'consts' array. The suffixes 'in' and 'out' denote the input and output value usages.

Implementation and installation

The projection procedures adhere as closely as possible to RATFOR and FORTRAN(66) standards and to structured programming techniques. Depending upon local RATFOR implementations, the sources may require a preliminary pass through the 'macro' tool since the 'define' statements employ many of the operations not often considered part of the RATFOR 'define'. In addition (due to local system requirements), the continuation character for arithmetic statements spanning multiple lines is a '%' character rather than the normal underscore. Since '%' is not used in any other context in the source code, it can be readily corrected by a single global edit replacement.

Source material is divided into three files: 1) Appendix B, containing common macro definitions, 2) Appendix C, containing the projection code, and 3) Appendix D, containing the library of support routines called by the projection programs. Each projections is subarchived into a quartet of files: 'strXX' which contains structure macro definitions of 'consts' and other macros unique to the projection, and 'setXX', 'forXX', and 'invXX'. Any external references made by procedures in Appendix B that are not resolved by the procedures in Appendix C are standard FORTRAN(66) intrinsic or RATFOR tool procedures. All variables and functions are declared as either 'integer' or 'PREC' and all real constants are either coded as 'C' macro arguments (for rational values) or obvious macro names (ie. PI, HALFPI) for irrational values. For readability, the data contained in the 'consts' array are referred to by appropriate symbols and mapped to the actual array location with macros. In addition, all intrinsic functions are defined with macros (e.g., SQRT, LOG, etc.). Macro symbols not expressly defined are standard RATFOR symbols.

To maintain the precision designed into the projection procedures, it is neccesary to employ hardware with at least 40 bits of mantissa precision. For the preponderance of computer hardware systems, the macro 'EXPLET' (Appendix B) must be defined as 'd' to perform appropriate 'double precision' expansion of the macros. Obviously, for machines with large mantissas, single precision or 'real' can be employed.

A program (Appendix F) to initialize, and to forward and inverse project each of the projections, generated at least one

"benchmark" table of transformation values for each projection (Appendix E). Such a program should be written for each new destination machine to verify the performance of the projection software.

In many cases, the accuracy of these projections exceeds precision requirements and a single precision (a mantissa with 24 or 28 bits) version could be employed. However, appropriate alteration of several macros (notably EPSILON10, etc.) may need to be made with single precision selection. Most projections appear to have no problem with precision reduction, however, the accuracy of the implimentation must be carefully verified.

Conclusions

Although this software was originally modified and redesigned for a graphics system, the results can be of general interest because of their modularity and attention to machine independence. This should facilitate the user's efforts to extract, expand, or modify the code for inclusion in other applications tasks. Use of the RATFOR preprocessor also provides a convenient mechanism for translation into other programing languages such as 'C' or Pascal.

References

Greenhood, David, 1964, Mapping: Chicago, U. of Chicago Press, 289 p.

Kernighan, Brian W., and Plauger, P.J., 1976, Software tools: Reading, Mass., Addison-Wesley, 338 p.

Snyder, John P., 1982, Map projections used by the Geological Survey: U.S. Geological Survey Bulletin 1532, 313 p.

The following matrix defines the initialization requirements for the constants ('consts') array employed by each of the projection procedures. All nonblank entries must be initialized before executing the initialization procedure. In addition, all blank entries may be employed by the projection procedures and the actual length of the array required may be longer than that indicated by the table (for required length see MAXCONSTS macro in Appendix B). The precision of the array is determined by the macro PREC (normally 'double precision').

Proj. No.	Constants array index												
	1	2	3	4	5	6	7	8	9	10	11	12	13
01	1	2			13	14							
03	1	2	4	5	6	7	8	9					
04	1	2	4	5	6	7	8	9					
05	1	2			6	11	8	9					
06	1	2			10	11	8	9					
07	1	2					8	9					
08a	1	2					8	9					
08b	1	2					8	9					
09	1	2					8	9					
10	3				6	7							
11	3				6	7	8	9					
12	3				6	7	8	9					
13	3				6	7	8	9					
14	3				6	7	8	9					
15	3		23		6	7	8	9					
16	3				6	7	8	9					
17	3				6	7	8	9					
18	3				6	7	8	9					
19a	1	2	24	15	16	7	8	9					27
19b	1	2	24			7	8	9	17	18	19	20	21

Entry No.	Definition of entry
1	- Semi-major axis of ellipsoid (see note).
2	- Square of spheroid eccentricity (unitless).

3 - Radius of reference sphere (see note).
4 - First standard parallel of conic (radians).
5 - Second standard parallel of conic (radians).
6 - Central meridian of projection (radians).
7 - Central parallel of projection (radians).
8 - False easting (see note).
9 - False northing (see note).
10 - Longitude directed down below pole of map (radians).
11 - Latitude of true scale (radians).
12 - Scale factor at central meridian (unitless).
13 - Longitude of any point in UTM zone (radians).
14 - Latitude of any point in UTM zone (radians).
15 - Azimuth of central line of projection (radians).
Measured clockwise from North.
16 - Longitude of point on central line where 15 is measured (radians).
17 - Longitude of first point on central line of projection (radians).
18 - Latitude of first point on central line of projection (radians).
19 - Longitude of second point on central line of projection (radians).
20 - Latitude of second point on central line of projection (radians).
21 - Option select = 0
22 - Option select <> 0
23 - Height of perspective point above sphere (see note).
24 - Scale factor at center of projection (unitless).

Note: these units must be consistent (normally meters) and determine the units of the projection cartesian coordinates.

Appendix B

The following is a listing of the macro symbols required for RATFOR preprocessing of all projection and support library procedures. Any changes should be made with care and understanding.

```
# symbol file for map projection software
#
#  set precision for routines
#define(EXPLET,      e)  # single precision
define(EXPLET,      d)  # double precision
#
#  precision declaration
define(PREC,[ifelse(EXPLET,d,double precision ,real )])
#
# constants "function"
define(C,     $1 EXPLET 0)
#
#  intrinsic functions
define(SUBEXP,[ifelse(EXPLET,d,d,)])
define(SQRT,  SUBEXP sqrt [ifelse($1,,,($1))]) # square root
define(COS,   SUBEXP cos [ifelse($1,,,($1))]) # cosine
define(SIN,   SUBEXP sin [ifelse($1,,,($1))]) # sine
define(EXP,   SUBEXP exp [ifelse($1,,,($1))]) # e ** arg
define(TAN,   SUBEXP tan [ifelse($1,,,($1))]) # tangent
define(ABS,   SUBEXP abs [ifelse($1,,,($1))]) # absolute
define(SIGN,  SUBEXP sign [ifelse($1,,,($1, $2))]) # sign
define(ATAN,  SUBEXP atan [ifelse($1,,,($1))]) # arc tangent
#                arc tangent(arg1 / arg2)
define(ATAN2, SUBEXP atan2 [ifelse($1,,,($1, $2))])
#                natural logrithm
define(LOG,   ifelse(EXPLET,d,dlog,alog)[ ifelse($1,,,($1))])
#
#  common irrational constants
define(HALFPI,C(1.5707963267948966))
define(FORTPI,C(0.78539816339744833))
define(PI,   C(3.14159265358979323846))
define(TWOPI,C(6.28318530717958647769))
#
# tolerances
define(TOL7,       1 EXPLET -7)
define(TOL5,       1 EXPLET -5)
define(EPSILON10, 1 EXPLET -10)
#
# current maximum required size of 'consts'
define(MAXCONSTS,  25)
#
# other useful constants...
define(RADTODEG, C(57.2957 79513 08232 08768))
define(DEGTORAD, C(.0174 53292 51994 32957 69))
define(ACLARK66, C(6 378 206.4))
```

```
define(ESCLARK66, C(.006 768 658))
define(RSPHERE,   C(6370977))
#
# common structure definitions for "consts"
define(a,      consts( 1)) # semi-major axis of ellipsoid or
                           # radius of sphere of reference
define(es,     consts( 2)) # ellipsoid eccentricity squared
define(lon0,   consts( 5)) # central meridian
define(lat0,   consts( 6)) # central parallel
define(x0,     consts( 7)) # false eastine
define(y0,     consts( 8)) # false northing
```

Appendix C

The following is a listing of the projection procedures archive file. Each projection is subarchived into its 'strXX', 'setXX', 'forXX', and 'invXX' sections for respective unique macros, initialization, forward and inverse projection procedures. It is recommended that the projections be split into their respective files before compilation so that macros of a previous projection cannot adversely affect subsequent projections. Obviously, the projections must be concatenated to the end of the macro file (Appendix B) prior to compilation.

Contents

	Page
prj01 - Universal Transverse Mercator (UTM)	C- 2
prj03 - Alber's conic equal area	C- 4
prj04 - Lambert conformal conic	C- 6
prj05 - Mercator	C- 8
prj06 - Polar stereographic	C-10
prj07 - Polyconic	C-12
prj08 - Equidistant Conic	C-14
prj09 - Transverse mercator	C-16
prj10 - Stereographic	C-19
prj11 - Lambert azimuthal equal-area	C-21
prj12 - Azimuthal equidistant	C-23
prj13 - Gnomonic	C-25
prj14 - Orthographic	C-27
prj15 - General vertical near-sided perspective	C-29
prj16 - Sinusoidal	C-31
prj17 - Equirectangular	C-32
prj18 - Miller Cylindrical	C-33
prj19 - Van Der Grinten I	C-34
prj20 - Oblique mercator	C-37

```
#-h- PRJ01 1555 asc FRI.,  4  FEB., 1983 9:28:3.52
#-h- STR01 270 asc TUE., 11  JAN., 1983 14:31:10.99
# str01 - universal transverse mercator - structure
#    a      - semi-major axis
#    es     - eccentricity squared
#    lon0   - central meridian
#    lat0   - central parallel
# local constants
define(ks0,      consts( 3))
#    x0     - false easting
#    y0     - false northing


#-h- SET01 584 asc TUE., 11  JAN., 1983 14:31:11.94
# set01 - universal transverse mercator - initialization
   integer function set01(consts)
   integer  mod, zone, set09
   PREC  consts(ARB)

      zone = (lon0 * C(15) / HALFPI)
      zone = mod(zone + 30 , 60)
      if (lon0 >= C(0))
         zone = zone + 1
      if (zone < 0)
         zone = zone + 60
      else if (zone == 0)
         zone = 1
      lon0 = (6 * zone - 183) * HALFPI / C(90)
      ks0 = C(0.9996)
      x0 = C(500000)
      if (lat0 < C(0))
         y0 = C(10000000)
      else
         y0 = C(0)
      lat0 = C(0)
      set01 = set09(consts)

   return
   end


#-h- FOR01 247 asc TUE., 11  JAN., 1983 14:31:13.27
# for01 - universal transverse mercator - forward
   integer function for01(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   integer for09

      for01 = for09(lamin, phiin, xout, yout, consts)

   return
   end
```

```
#-h- INV01 246 asc TUE., 11  JAN., 1983 14:31:14.61
#inv01 - universal transverse mercator - inverse
   integer function inv01(xin, yin, lamout, phiout, consts)
   integer inv09
   PREC  consts(ARB), xin, yin, lamout, phiout

      inv01 = inv09(xin, yin, lamout, phiout, consts)

   return
   end
```

```
#-h- PRJ03 2998 asc THU.,  3  FEB., 1983 7:0:4.49
#-h- STR03 799 asc MON., 10  JAN., 1983 11:8:25.83
# str03 - Alber's Equal Area - structure
#    a      - semi-major axis
#    es     - eccentricity squared
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
define(lat1,     consts( 3)) # 1st standard parallel (radian)
define(lat2,     consts( 4)) # 2nd standard parallel (radian)
# local constants
define(e,        consts( 9))
define(ns,       consts(10))
define(c,        consts(11))
define(rh0,      consts(12))
define(TOL7,    1 EXPLET -7)


#-h- SET03 987 asc MON., 10  JAN., 1983 10:58:36.62
# set03 - alber's conical equal area - initialization
   integer function set03(consts)
   PREC  consts(ARB), con, cosphi, ABS, COS,  SIN, SQRT,
          ms1, ms2, msfn0, qs0, qs1, qs2, qsfn0, sinphi

      if (ABS(lat1 + lat2) < EPSILON10) {
         call remark("s03: 1st parallel = - 2nd parallel.")
         set03 = ERR
         }
      else {
         e = SQRT(es)
         sinphi = SIN(lat1)
         con = sinphi
         cosphi = COS(lat1)
         ms1 = msfn0(e, sinphi, cosphi)
         qs1 = qsfn0(e, sinphi, cosphi)
         sinphi = SIN(lat2)
         cosphi = COS(lat2)
         ms2 = msfn0(e, sinphi, cosphi)
         qs2 = qsfn0(e, sinphi, cosphi)
         qs0 = qsfn0(e, SIN(lat0), COS(lat0))
         if (ABS(lat1 - lat2) < EPSILON10)
            ns = con
         else
            ns = (ms1 * ms1 - ms2 * ms2) / (qs2 - qs1)
         c = ms1 * ms1 + ns * qs1
         rh0 = a * SQRT(c - ns * qs0) / ns
         set03 = OK
         }

   return
   end


#-h- FOR03 438 asc MON., 10  JAN., 1983 10:59:13.97
```

```
# for03 - alber's conical equal area - forward
   integer function for03(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, SIN, COS, SQRT, qsfn0, rh, theta

      rh = a * SQRT(c - ns * qsfn0(e, SIN(phiin),
             COS(phiin))) / ns
      theta = ns * adjl0(lamin - lon0)
      xout = x0 + rh * SIN(theta)
      yout = y0 + rh0 - rh * COS(theta)
      for03 = OK

   return
   end


#-h- INV03 880 asc MON., 10  JAN., 1983 10:59:50.03
#inv03 - alber's conical equal area - inverse
   integer function inv03(xin, yin, lamout, phiout, consts)
   PREC xin, yin, lamout, phiout, consts(ARB)
   PREC  adjl0, ABS, ATAN2, COS, LOG, SIGN, SIN, SQRT, phi10,
          qs, rh, theta, x, y
   integer err

      err = OK
      x = xin - x0
      y = rh0 - yin + y0
      rh = SIGN(SQRT(x * x + y * y) , ns)
      if (rh ^= C(0)) {
         qs = SIGN(C(1), ns)
         theta = ATAN2(qs * x, qs * y)
         }
      else
         theta = C(0)
      qs = (c - ( rh * ns / a ) ** 2) / ns
      if (e >= TOL7)
         if ((ABS( C(1) - C(.5) * (C(1) - es) * LOG((C(1) - e) %
                / (C(1) + e)) / e) - ABS(qs)) > TOL7)
            phiout = phi10(e, qs, err)
         else
            phiout = SIGN(HALFPI , qs)
      else
         phiout = phi10(e, qs, err)
      lamout = adjl0(theta / ns + lon0)
      inv03 = err

   return
   end
```

```
#-h- PRJ04 2936 asc THU.,  3  FEB., 1983 7:0:7.91
#-h- STR04 804 asc MON., 10  JAN., 1983 13:56:2.74
# str04 - Lambert Conformal Conic - structure
#    a      - semi-major axis
#    es     - eccentricity squared
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
define(lat1,     consts( 3)) # 1st standard parallel (radian)
define(lat2,     consts( 4)) # 2nd standard parallel (radian)
# local constants
define(e,        consts( 9))
define(ns,       consts(10))
define(f,        consts(11))
define(rh0,      consts(12))


#-h- SET04 923 asc MON., 10  JAN., 1983 13:58:9.22
# set04 - lambert conformal conic - initialization
   integer function set04(consts)
   PREC  consts(ARB), con, cosphi, ABS, COS, SIN, SQRT, LOG,
         ms1, ms2, msfn0, sinphi, ts1, ts2, tsfn0

      if (ABS(lat1 + lat2) < EPSILON10) {
         call remark("s04: 1st parallel = - 2nd parallel.")
         set04 = ERR
         }
      else {
         e = SQRT(es)
         sinphi = SIN(lat1)
         con = sinphi
         cosphi = COS(lat1)
         ms1 = msfn0(e, sinphi, cosphi)
         ts1 = tsfn0(e, lat1, sinphi)
         sinphi = SIN(lat2)
         cosphi = COS(lat2)
         ms2 = msfn0(e, sinphi, cosphi)
         ts2 = tsfn0(e, lat2, sinphi)
         if (ABS(lat1 - lat2) < EPSILON10)
            ns = con
         else
            ns = LOG(ms1 / ms2) / LOG(ts1 / ts2)
         f = ms1 / (ns * ts1 ** ns)
         rh0 = a * f * tsfn0(e, lat0, SIN(lat0)) ** ns
         set04 = OK
         }

   return
   end
```

```
#-h- FOR04 643 asc MON., 10  JAN., 1983 13:56:31.04
# for04 - lambert conformal conic - forward
   integer function for04(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, ABS, COS, SIN, SQRT, rh, theta, tsfn0

      for04 = OK
      if (ABS(ABS(phiin) - HALFPI) <= EPSILON10) {
         if (phiin * ns <= C(0)) {
            for04 = ERR
            call error("f04: lat ^= pi/2.")
            }
         else
            rh = C(0)
         }
      else
         rh = a * f * tsfn0(e, phiin, SIN(phiin)) ** ns
      theta = ns * adjl0(lamin - lon0)
      xout = x0 + rh * SIN(theta)
      yout = y0 + rh0 - rh * COS(theta)

   return
   end


#-h- INV04 644 asc MON., 10  JAN., 1983 13:57:19.30
#inv04 - lambert conformal conic - inverse
   integer function inv04(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adjl0, ATAN2, COS, SIGN, SIN, SQRT, phi20, rh,
         theta, x, y, con
   integer err

      err = OK
      x = xin - x0
      y = rh0 - yin + y0
      rh = SIGN(SQRT(x * x + y * y) , ns)
      if (rh ^= C(0)) {
         con = SIGN(C(1), ns)
         theta = ATAN2(con * x, con * y)
         }
      else
         theta = C(0)
      if (rh == C(0) & ns <= C(0))
         phiout = - HALFPI
      else
         phiout = phi20(e, (rh / (a * f)) ** (C(1) / ns) , err)
      lamout = adjl0(theta / ns + lon0)
      inv04 = err

   return
   end
```

```
#-h- PRJ05 1537 asc WED.,  2  FEB., 1983 11:3:12.77
#-h- STR05 357 asc MON., 10  JAN., 1983 15:22:17.94
# str05 - mercator - structure
#    a      - semi-major axis
#    es     - eccentricity squared
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
#  additional input
define(latts,    consts( 6)) # parallel of true scale (radian)
#
# local usage
define(e,        consts( 9))
define(ap,       consts(10))


#-h- SET05 201 asc MON., 10  JAN., 1983 15:23:11.08
# set05 - mercator
   integer function set05(consts)
   PREC  consts(ARB), COS, SIN, SQRT, msfn0

      e = SQRT(es)
      ap = a * msfn0(es, SIN(latts), COS(latts))
      set05 = OK

   return
   end


#-h- FOR05 455 asc MON., 10  JAN., 1983 15:22:38.67
# for05 - mercator
   integer function for05(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, ABS, LOG, SIN, tsfn0
      if (ABS(ABS(phiin) - HALFPI) <= EPSILON10) {
         call remark("f05: lat ~= pi/2.")
         for05 = ERR
         }
      else {
         xout = x0 + ap * adjl0(lamin - lon0)
         yout = y0 - ap * LOG(tsfn0(e, phiin, SIN(phiin)))
         for05 = OK
         }

   return
   end
```

```
#-h- INV05 316 asc MON., 10  JAN., 1983 15:23:39.99
#inv05 - mercator
   integer function inv05(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adjl0, EXP, phi20
   integer err

      err = OK
      phiout = phi20(e, EXP((y0 - yin) / ap) , err)
      lamout = adjl0(lon0 + (xin - x0) / ap)
      inv05 = err

   return
   end
```

```
#-h- PRJ06 2165 asc FRI.,  4  FEB., 1983 9:28:10.25
#-h- STR06 490 asc SAT., 15  JAN., 1983 14:27:49.54
# str10 - polar stereographic - structure
# input requirements:
#    a      - semi-major axis
#    es     - eccentricity squared
#    lon0   - central meridian
#    x0     - false easting
#    y0     - false northing
define(k0,       consts(03)) # scale factor at pole
define(latts,    consts(06)) # parallel of true scale or pole sign
define(check,    consts( 9)) # ^= 0 then k0 else lat.true scale
#
#  local storage
define(e,        consts(10))
define(fmul,     consts(11))
define(sfac,     consts(12))

#-h- SET06 562 asc SAT., 15  JAN., 1983 14:27:50.67
# set06 - polar stereographic - initialization
   integer function set06(consts)
   PREC  consts(ARB), con, ABS, COS, SIN, e3fn0,
          msfn0, sinphi, tsfn0

      set06 = OK
      e = SQRT(es)
      sfac = SIGN(C(1), latts)
      con = ABS(latts)
      if (check ^= C(0) & ABS(con - HALFPI) < EPSILON10) {
         sinphi = SIN(con)
         fmul = sfac * a * msfn0(e, sinphi, COS(con)) / tsfn0(e,
            con, sinphi)
         }
      else
         fmul = sfac * C(2) * a / e3fn0(e)
      if (check ^= C(0))
         fmul = k0 * fmul

   return
   end

#-h- FOR06 401 asc SAT., 15  JAN., 1983 14:32:51.61
# for06 - polar stereographic - forward
   integer function for06(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adj10, con, ABS, COS, SIN, rh, tsfn0

      for06 = OK
      con = sfac * phiin
      rh = fmul * tsfn0(e, con, SIN(con))
      con = sfac * adj10(lamin - lon0)
      xout = x0 +  rh * SIN(con)
      yout = y0 -  rh * COS(con)
   return
   end
```

```
#-h- INV06 504 asc SAT., 15  JAN., 1983 14:29:40.99
#inv06 - polar stereographic - inverse
   integer function inv06(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   integer err
   PREC  adj10, ABS, ATAN2, SQRT, phi20, rh, x, y

      x = sfac * (xin - x0)
      y = sfac * (yin - y0)
      rh = SQRT(x * x + y * y)
      phiout = sfac * phi20(e, rh / ABS(fmul) , err)
      if (rh == C(0))
         lamout = sfac * lon0
      else
         lamout = adj10(sfac * ATAN2(x , -y) + lon0)
      inv06 = err

   return
   end
```

```
#-h- PRJ07 2161 asc THU.,  3  FEB., 1983 7:0:13.44
#-h- STR07 681 asc MON., 10  JAN., 1983 14:38:11.68
# str07 - Polyconic - structure
#    a      - semi-major axis
#    es     - eccentricity squared
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
# local constants
define(e,        consts( 9))
define(e0,       consts(10))
define(e1,       consts(11))
define(e2,       consts(12))
define(ml0,      consts(13))
define(TOL7,    1 EXPLET -7)


#-h- SET07 280 asc MON., 10  JAN., 1983 14:39:43.74
# set07 - polyconic - initialization
   integer function set07(consts)
   PREC  consts(ARB), SQRT, e0fn0, e1fn0, e2fn0, ml0fn0

      e = SQRT(es)
      e0 = e0fn0(es)
      e1 = e1fn0(es)
      e2 = e2fn0(es)
      ml0 = ml0fn0(e0, e1, e2, lat0)
      set07 = OK

   return
   end
```

```
#-h- FOR07 708 asc MON., 10  JAN., 1983 14:38:32.75
# for07 - polyconic - forward
   integer function for07(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, con, cosphi, ABS, COS, SIN, SQRT, ml,
          mlfn0, ms, msfn0, sinphi

      con = adjl0(lamin - lon0)
      if (ABS(phiin) <= TOL7) {
         xout = x0 + a * con
         yout = y0 - a * ml0
         }
      else {
         sinphi = SIN(phiin)
         cosphi = COS(phiin)
         ml = mlfn0(e0, e1, e2, phiin)
         ms = msfn0(e, sinphi, cosphi)
         con = con * sinphi
         xout = x0 + a * ms * SIN(con) / sinphi
         yout = y0 + a * (ml - ml0 + ms * (C(1) - COS(con)) / sinphi)
         }
      for07 = OK

   return
   end


#-h- INV07 600 asc MON., 10  JAN., 1983 14:39:5.00
#inv07 - polyconic
   integer function inv07(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adjl0, al, b, c, ABS, darsin, SIN, phi40, x, y
   integer err

      err = OK
      x = xin - x0
      y = yin - y0
      al = ml0 + y / a
      if (ABS(al) <= TOL7) {
         lamout = x / a + lon0
         phiout = C(0)
         }
      else {
         b = al * al + (x / a) ** 2
         phiout = phi40(es, e0, e1, e2, al, b, c, err)
         lamout = adjl0(darsin(x * c / a) / SIN(phiout) + lon0)
         }
      inv07 = err

   return
   end
```

```
#-h- PRJ08 2654 asc THU.,  3  FEB., 1983 7:0:16.36
#-h- STR08 996 asc TUE., 11  JAN., 1983 7:56:52.64
# str04 - Equidistant Conic - structure
#    a      - semi-major axis
#    es     - eccentricity squared
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
define(lat1,     consts( 3)) # 1st standard parallel (radian)
define(lat2,     consts( 4)) # 2nd standard parallel (radian)
define(check,    consts( 9)) # mode selection
# local constants
define(e,        consts(10))
define(e0,       consts(11))
define(e1,       consts(12))
define(e2,       consts(13))
define(ns,       consts(14))
define(gl,       consts(15))
define(rh0,      consts(16))


#-h- SET08 876 asc TUE., 11  JAN., 1983 7:59:5.91
# set08 - equidistant conic - initialization
   integer function set08(consts)
   PREC  adj10, consts(ARB), ABS, COS, SIN, e0fn0, e1fn0, e2fn0,
          mlfn0, ml1, ml2, ms1, ms2, msfn0, phi30

      e = SQRT(es)
      e0 = e0fn0(es)
      e1 = e1fn0(es)
      e2 = e2fn0(es)
      if (ABS(lat1 + lat2) < EPSILON10) {
         call remark("s08: 1st parallel ~= - 2nd parallel.")
         set08 = ERR
         }
      else {
         ns = SIN(lat1)
         ms1 = msfn0(e, ns, COS(lat1))
         ml1 = mlfn0(e0, e1, e2, lat1)
         if (check ~= C(0)) {
            ns = SIN(lat2)
            ms2 = msfn0(e, ns, COS(lat2))
            ml2 = mlfn0(e0, e1, e2, lat2)
            if (ABS(lat1 - lat2) >= EPSILON10)
               ns = (ms1 - ms2) / (ml2 - ml1)
            }
         gl = ml1 + ms1 / ns
         rh0 = a * (gl - mlfn0(e0, e1, e2, lat0))
         set08 = OK
         }

   return
   end
```

```
#-h- FOR08 374 asc TUE., 11  JAN., 1983 7:59:7.33
# for08 - equidistant conic - forward
   integer function for08(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, COS, SIN, mlfn0, rh, theta

      rh = a * (gl - mlfn0(e0, e1, e2, phiin))
      theta = ns * adjl0(lamin - lon0)
      xout = x0 + rh * SIN(theta)
      yout = y0 + rh0 - rh * COS(theta)
      for08 = OK

   return
   end


#-h- INV08 533 asc TUE., 11  JAN., 1983 7:59:8.41
#inv08 - equidistant conic
   integer function inv08(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adjl0, ATAN2, SIGN, phi30, rh, theta, x, y, con
   integer err

      err = 0
      x = xin - x0
      y = rh0 - yin + y0
      rh = SIGN(SQRT(x * x + y * y) , ns)
      if (rh ^= C(0)) {
         con = SIGN(C(1), ns)
         theta = ATAN2(con * x, con * y)
         }
      else
         theta = C(0)
      phiout = phi30(gl - rh / a, e0, e1, e2, err)
      lamout = adjl0(lon0 + theta / ns)
      inv08 = err

   return
   end
```

```
#-h- PRJ09 4952 asc FRI.,  4  FEB., 1983 9:28:16.43
#-h- STR09 514 asc TUE., 11  JAN., 1983 13:29:56.01
# str01 - transverse mercator - structure
#    a      - semi-major axis
#    es     - eccentricity squared
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
define(ks0,      consts( 3)) # scale factor at central meridian
# local constants
define(e,        consts( 9))
define(e0,       consts(10))
define(e1,       consts(11))
define(e2,       consts(12))
define(esp,      consts(13))
define(ml0,      consts(14))
define(MAXITER,       6)
define(TOL,    1 EXPLET -5)


#-h- SET09 331 asc TUE., 11  JAN., 1983 13:50:41.95
# set09 - transverse mercator - initialization
   integer function set09(consts)
   PREC  mlfn0, SQRT, consts(ARB), e0fn0, e1fn0, e2fn0

      e = SQRT( es )
      e0 = e0fn0(es)
      e1 = e1fn0(es)
      e2 = e2fn0(es)
      ml0 = a * mlfn0(e0, e1, e2, lat0)
      if (e >= TOL)
         esp = es / (C(1) - es)

   return
   end


#-h- FOR09 1497 asc TUE., 11  JAN., 1983 14:11:45.86
# for09 - transverse mercator
   integer function for09(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, al, als, b, c, cosphi, ABS, COS,  LOG, dlon,
         darcos, SIGN, SIN, SQRT, TAN, ml, mlfn0, n, sinphi,
         t, tq

      for09 = OK
      dlon = adjl0(lamin - lon0)
      sinphi = SIN(phiin)
      cosphi = COS(phiin)
      if (e < TOL) { # spherical earth
         b = cosphi * sinphi
         t = C(1) - b
         if (ABS(t) <= EPSILON10) {
            call remark("f09: projection to inf.")
```

```
            for09 = ERR
            }
         else {
            xout = C(.5) * a * ks0 * LOG((C(1) + b) / t)
            yout = a * ks0 * (SIGN(darcos(cosphi * COS(dlon) / %
               SQRT(C(1) - b * b)), phiin) - lat0)
            }
         }
      else  { # elliptical earth
         al = cosphi * dlon
         als = al * al
         c = esp * cosphi * cosphi
         tq = TAN(phiin)
         t = tq * tq
         n = a / SQRT(C(1) - es * sinphi * sinphi)
         ml = a * mlfn0(e0, e1, e2, phiin)
         xout = ks0 * n * al * (C(1) + als / C(6) * (C(1) - %
            t + c + als / C(20) * (C(5) - C(18) * t + t * t %
            + C(72) * c - C(58) * esp))) + x0
         yout = ks0 * (ml - ml0 + n * tq * (als * (C(.5) + %
            als / C(24) * (C(5) - t + C(9) * c + C(4) * c * c %
            + als / C(30) * (C(61) - C(58) * t + t * t + %
            C(600) * c - C(330) * esp))))) + y0

         }

   return
   end


#-h- INV09 2400 asc TUE., 11  JAN., 1983 14:11:48.19
#inv09 - transverse mercator - inverse
   integer function inv09(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   integer i
   PREC  adjl0, al, als, b, c, con, cosphi, cs, d, ABS,
         ATAN2, COS, EXP, dphi, ds, SIGN, SIN, SQRT, TAN,
         f, g, h, n, phi, r, sinphi, t, tanphi, ts,
         x, y, darsin

      inv09 = OK
      x = xin - x0
      y = yin - y0
      if (e < TOL) { # spherical earth
         f = EXP(x / (a * ks0))
         g = C(.5) * (f - C(1) / f)
         h = COS(lat0 + y / (a * ks0))
         phiout = SIGN(darsin(SQRT((C(1) - h * h) /  %
              (C(1) + g * g))), y)
         if (g < C(0) & h < C(0))
            lamout = lon0
         else
            lamout = adjl0(ATAN2(g, h) + lon0)
         }
      else { # elliptical earth
```

```
         con = (ml0 + y / ks0) / a
         phi = con
         for(i = 1; i <= MAXITER ; i = i + 1) {
            dphi = ((con + e1 * SIN(C(2) * phi) - e2 * %
                   SIN(C(4) * phi)) / e0) - phi
            phi = phi + dphi
            if (ABS(dphi) <= EPSILON10)
               break
            }
         if (i > MAXITER) {
            call remark("i09: convergence failure.")
            inv09 = ERR
            }
         else {
            if (ABS(phi) >= HALFPI) {
               phiout = SIGN(HALFPI , y)
               lamout = lon0
               }
            else {
               sinphi = SIN(phi)
               cosphi = COS(phi)
               tanphi = TAN(phi)
               c = esp * cosphi * cosphi
               cs = c * c
               t = tanphi * tanphi
               ts = t * t
               con = C(1) - es * sinphi * sinphi
               n = a / SQRT(con)
               r = n * (C(1) - es) / con
               d = x / (n * ks0)
               ds = d * d
               phiout = phi - (n * tanphi * ds / r) * (C(.5) - %
                  ds / C(24) * (C(5) + C(3) * t + C(10) * c %
                  -C(4) * cs - C(9) * esp - ds / C(30) * (C(61) %
                  + C(90) * t + C(298) * c + C(45) * ts %
                  - C(252) * esp - C(3) * cs)))
               lamout = adjl0(lon0 + (d * (C(1) - ds / C(6) %
                  * (C(1) + C(2) * t + c - ds / C(20) * (C(5) %
                  - C(2) * c + C(28) * t - C(3) * cs %
                  + C(8) * esp + C(24) * ts))) / cosphi))
               }
            }
         }

return
end
```

```
#-h- PRJ10 2309 asc FRI., 28  JAN., 1983 10:46:50.89
#-h- STR10 280 asc SAT., 15  JAN., 1983 9:24:21.20
# str10 - stereographic - structure
# input requirements:
#    a      - reference sphere radius
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
#
#  local storage
define(sinph0,  consts( 9))
define(cosph0,  consts(10))


#-h- SET10 192 asc SAT., 15  JAN., 1983 9:24:22.24
# set10 - stereographic - initialization
   integer function set10(consts)
   PREC  consts(ARB), COS, SIN

      sinph0 = SIN(lat0)
      cosph0 = COS(lat0)
      set10 = OK

   return
   end


#-h- FOR10 682 asc SAT., 15  JAN., 1983 9:26:21.57
# for10 - stereographic - forward
   integer function for10(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adj10, coslon, cosphi, COS, SIN, g, sinphi, lon

      sinphi = SIN(phiin)
      cosphi = COS(phiin)
      lon = adj10(lamin - lon0)
      coslon = COS(lon)
      g = C(1) + sinph0 * sinphi + cosph0 * cosphi * coslon
      if (g == C(0)) {
         for10 = ERR
         call remark("f10: cannot project.")
         }
      else {
         for10 = OK
         g = a * C(2) / g
         xout = x0 + g * cosphi * SIN(lon)
         yout = y0 + g * (cosph0 * sinphi - sinph0 * cosphi * %
            coslon)
         }

   return
   end
```

```
#-h- INV10 951 asc SAT., 15  JAN., 1983 9:24:24.84
#inv10 - stereographic - inverse
   integer function inv10(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adj10, con, coslon, cosphi, cosz, ABS,
         darsin, ATAN2, COS, SIN, SQRT, rh, sinz, x, y, z

      inv10 = OK
      x = xin - x0
      y = yin - y0
      rh = SQRT(x * x + y * y)
      z = C(2) * ATAN(rh / (C(2) * a))
      sinz = SIN(z)
      cosz = COS(z)
      lamout = lon0
      if (ABS(rh) <= EPSILON10)
         phiout = lat0
      else {
         phiout = darsin(cosz * sinph0 + y * sinz * cosph0 / rh)
         if (ABS(ABS(lat0) - HALFPI) <= EPSILON10)
            lamout = adj10(lon0 + ATAN2(SIGN(x, lat0),
                     SIGN(y, -lat0)))
         else {
            con = cosz - sinph0 * SIN(phiout)
            if (con ^= C(0) | x ^= C(0))
               lamout = adj10(lon0 + ATAN2((x * sinz * cosph0),
                                (con * rh)))
         }
      }

   return
   end
```

```
#-h- PRJ11 2621 asc FRI., 28  JAN., 1983 10:46:57.49
#-h- STR11 295 asc SAT., 15  JAN., 1983 9:30:57.76
# str11 - lambert azimuthal equal-area - structure
# input requirements:
#    a      - reference sphere radius
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
#
#  local storage
define(sinph0,  consts( 9))
define(cosph0,  consts(10))


#-h- SET11 207 asc SAT., 15  JAN., 1983 9:30:58.80
# set11 - lambert azimuthal equal area - initialization
   integer function set11(consts)
   PREC  consts(ARB), COS, SIN

      sinph0 = SIN(lat0)
      cosph0 = COS(lat0)
      set11 = OK

   return
   end


#-h- FOR11 709 asc SAT., 15  JAN., 1983 9:32:15.73
# for11 - lambert azimuthal equal-area - forward
   integer function for11(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, coslon, cosphi, COS, SIN, g, sinphi, lon, SQRT

      sinphi = SIN(phiin)
      cosphi = COS(phiin)
      lon = adjl0(lamin - lon0)
      coslon = COS(lon)
      g = C(1) + sinph0 * sinphi + cosph0 * cosphi * coslon
      if (g == C(0)) {
         for11 = ERR
         call remark("f11: cannot project.")
         }
      else {
         for11 = OK
         g = a * SQRT(C(2) / g)
         xout = x0 + g * cosphi * SIN(lon)
         yout = y0 + g * (cosph0 * sinphi - sinph0 * cosphi * %
            coslon)
         }

   return
   end
```

```
#-h- INV11 1205 asc SAT., 15  JAN., 1983 9:34:42.89
#inv11 - lambert azimuthal equal-area - inverse
   integer function inv11(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adj10, con, coslon, cosphi, cosz, ABS,
         darsin, ATAN2, COS, SIN, SQRT, rh, sinz, x, y, z

      x = xin - x0
      y = yin - y0
      rh = SQRT(x * x + y * y)
      con = rh / (C(2) * a)
      if (con > C(1)) {
         call remark("i11: data error.")
         inv11 = ERR
         }
      else {
         inv11 = OK
         z = C(2) * darsin(con)
         sinz = SIN(z)
         cosz = COS(z)
         lamout = lon0
         if (ABS(rh) <= EPSILON10)
            phiout = lat0
         else {
            phiout = darsin(cosz * sinph0 + y * sinz * %
                     cosph0 / rh)
                         if (ABS(ABS(lat0) - HALFPI) <= EPSILON10)
               lamout = adj10(lon0 + ATAN2(SIGN(x, lat0),
                        SIGN(y, -lat0)))
            else {
               con = cosz - sinph0 * SIN(phiout)
               if (con ^= C(0))
                 lamout = adj10(lon0 + ATAN2((x * sinz * cosph0),
                                (con * rh)))
               }
            }
         }

   return
   end
```

```
#-h- PRJ12 2644 asc FRI., 28  JAN., 1983 10:47:4.30
#-h- STR12 288 asc SAT., 15  JAN., 1983 9:38:31.09
# str12 - azimuthal equidistant - structure
# input requirements:
#    a      - reference sphere radius
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
#
#  local storage
define(sinph0,  consts( 9))
define(cosph0,  consts(10))

#-h- SET12 200 asc SAT., 15  JAN., 1983 9:38:32.10
# set12 - azimuthal equidistant - initialization
   integer function set12(consts)
   PREC  consts(ARB), COS, SIN

      sinph0 = SIN(lat0)
      cosph0 = COS(lat0)
      set12 = OK

   return
   end

#-h- FOR12 809 asc SAT., 15  JAN., 1983 9:38:33.17
# for12 - azimuthal equidistant - forward
   integer function for12(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, coslon, cosphi, COS, SIN, g, sinphi, lon,
          darcos, ABS

      for12 = OK
      sinphi = SIN(phiin)
      cosphi = COS(phiin)
      lon = adjl0(lamin - lon0)
      coslon = COS(lon)
      g = sinph0 * sinphi + cosph0 * cosphi * coslon
      if (DABS(DABS(g) - C(1)) < EPSILON10)
         if (g < C(0)) {
            for12 = ERR
            call remark("f12: cannot project.")
            }
         else
            g = a
      else {
         g = darcos(g)
         g = a * g / SIN(g)
         }
      xout = x0 + g * cosphi * SIN(lon)
      yout = y0 + g * (cosph0 * sinphi - sinph0 * cosphi * %
            coslon)
   return
   end
```

```
#-h- INV12 1142 asc SAT., 15  JAN., 1983 9:38:34.71
#inv12 - azimuthal equidistant - inverse
   integer function inv12(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adjlO, con, coslon, cosphi, cosz, ABS,
         darsin, ATAN2, COS, SIN, SQRT, rh, sinz, x, y, z

      x = xin - x0
      y = yin - y0
      rh = SQRT(x * x + y * y)
      if (rh > PI * a) {
         call remark("i12: data error.")
         inv12 = ERR
         }
      else {
         inv12 = OK
         z = rh / a
         sinz = SIN(z)
         cosz = COS(z)
         lamout = lon0
         if (ABS(rh) <= EPSILON10)
            phiout = lat0
         else {
            phiout = darsin(cosz * sinph0 + y * sinz * %
                  cosph0 / rh)
            con = ABS(lat0) - HALFPI
            if (ABS(con) <= EPSILON10)
               lamout = adjl0(lon0 + ATAN2(SIGN(x, lat0),
                     SIGN(y, -lat0)))
            else {
               con = cosz - sinph0 * SIN(phiout)
               if (con ^= C(0))
                 lamout = adjl0(lon0 + ATAN2((x * sinz * cosph0),
                               (con * rh)))
            }
         }
      }

   return
   end
```

```
#-h- PRJ13 2281 asc FRI., 28  JAN., 1983 10:47:11.12
#-h- STR13 275 asc SAT., 15  JAN., 1983 10:12:50.37
# str13 - Gnomonic - structure
# input requirements:
#    a      - reference sphere radius
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
#
#  local storage
define(sinph0,  consts( 9))
define(cosph0,  consts(10))


#-h- SET13 187 asc SAT., 15  JAN., 1983 10:12:51.33
# set13 - Gnomonic - initialization
   integer function set13(consts)
   PREC  consts(ARB), COS, SIN

      sinph0 = SIN(lat0)
      cosph0 = COS(lat0)
      set13 = OK

   return
   end


#-h- FOR13 663 asc SAT., 15  JAN., 1983 10:15:30.34
# for13 - Gnomonic - forward
   integer function for13(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, coslon, cosphi, COS, SIN, g, sinphi, lon

      sinphi = SIN(phiin)
      cosphi = COS(phiin)
      lon = adjl0(lamin - lon0)
      coslon = COS(lon)
      g = sinph0 * sinphi + cosph0 * cosphi * coslon
      if (g <= C(0)) {
         for13 = ERR
         call remark("f13: cannot project.")
         }
      else {
         for13 = OK
         g = a / g
         xout = x0 + g * cosphi * SIN(lon)
         yout = y0 + g * (cosph0 * sinphi - sinph0 * cosphi * %
            coslon)
         }

   return
   end
```

```
#-h- INV13 948 asc SAT., 15  JAN., 1983 10:12:53.66
#inv13 - gnomonic - inverse
   integer function inv13(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adjl0, con, coslon, cosphi, cosz, ABS, darsin, ATAN,
         darsin, ATAN, ATAN2, COS, SIN, SQRT,rh, sinphi,
         sinz, x, y, z

      inv13 = OK
      x = xin - x0
      y = yin - y0
      rh = SQRT(x * x + y * y)
      z = ATAN(rh / a)
      sinz = SIN(z)
      cosz = COS(z)
      lamout = lon0
      if (ABS(rh) <= EPSILON10)
         phiout = lat0
      else {
         phiout = darsin(cosz * sinph0 + y * sinz * cosph0 / rh)
         if (ABS(ABS(lat0) - HALFPI) <= EPSILON10)
            lamout = adjl0(lon0 + ATAN2(SIGN(x, lat0),
                        SIGN(y, -lat0)))
         else {
            con = cosz - sinph0 * SIN(phiout)
            if (con ^= C(0))
               lamout = adjl0(lon0 + ATAN2((x * sinz * cosph0),
                  (con * rh)))
         }
      }

   return
   end
```

```
#-h- PRJ14 2456 asc FRI., 28  JAN., 1983 10:47:17.26
#-h- STR14 279 asc SAT., 15  JAN., 1983 10:13:15.71
# str14 - orthographic - structure
# input requirements:
#    a      - reference sphere radius
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
#
#  local storage
define(sinph0,  consts( 9))
define(cosph0,  consts(10))


#-h- SET14 191 asc SAT., 15  JAN., 1983 10:13:16.60
# set14 - orthographic - initialization
   integer function set14(consts)
   PREC  consts(ARB), COS, SIN

      sinph0 = SIN(lat0)
      cosph0 = COS(lat0)
      set14 = OK

   return
   end


#-h- FOR14 651 asc SAT., 15  JAN., 1983 10:15:45.84
# for14 - orthographic - forward
   integer function for14(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, coslon, cosphi, COS, SIN, sinphi, lon

      sinphi = SIN(phiin)
      cosphi = COS(phiin)
      lon = adjl0(lamin - lon0)
      coslon = COS(lon)
      if (sinph0 * sinphi + cosph0 * cosphi *
            coslon < - EPSILON10) {
         for14 = ERR
         call remark("f14: cannot project.")
         }
      else {
         for14 = OK
         xout = x0 + a * cosphi * SIN(lon)
         yout = y0 + a * (cosph0 * sinphi - sinph0 * cosphi * %
            coslon)
         }

   return
   end
```

```
#-h- INV14 1126 asc SAT., 15  JAN., 1983 10:19:17.78
#inv14 - orthographic - inverse
   integer function inv14(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adjlO, con, coslon, cosphi, cosz, ABS, darsin, ATAN2,
          COS, SIN, SQRT,rh, sinphi, sinz, x, y, z

      x = xin - xO
      y = yin - yO
      rh = SQRT(x * x + y * y)
      if (rh > a) {
         call remark("i14: data error.")
         inv14 = ERR
         }
      else {
         inv14 = OK
         z = darsin(rh / a)
         sinz = SIN(z)
         cosz = COS(z)
         lamout = lonO
         if (ABS(rh) <= EPSILON1O)
            phiout = latO
         else {
            phiout = darsin(cosz * sinphO + y * sinz * %
                  cosphO / rh)
            if (ABS(ABS(latO) - HALFPI) <= EPSILON1O)
               lamout = adjlO(lonO + ATAN2(SIGN(x, latO),
                     SIGN(y, -latO)))
            else {
               con = cosz - sinphO * SIN(phiout)
               if (con ^= C(O))
                 lamout = adjlO(lonO + ATAN2((x * sinz * cosphO),
                           (con * rh)))
               }
            }
         }

   return
   end
```

```
#-h- PRJ15 2851 asc FRI., 28  JAN., 1983 10:47:24.11
#-h- STR15 361 asc SAT., 15  JAN., 1983 10:13:35.18
# str15 - general vertical near-side perspective - structure
# input requirements:
#    a      - reference sphere radius
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
define(height,  consts( 3))
#
#  local storage
define(sinph0,  consts( 9))
define(cosph0,  consts(10))
define(p,       consts(11))


#-h- SET15 245 asc SAT., 15  JAN., 1983 10:13:36.08
# set15 - general vertical near-sided perspective -initialization
   integer function set15(consts)
   PREC  consts(ARB), COS, SIN

      sinph0 = SIN(lat0)
      cosph0 = COS(lat0)
      p = C(1) + height / a
      set15 = OK

   return
   end


#-h- FOR15 717 asc SAT., 15  JAN., 1983 10:16:1.05
# for15 - general vertical near-side perspective - forward
   integer function for15(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, coslon, cosphi, COS, SIN, g, sinphi, lon

      sinphi = SIN(phiin)
      cosphi = COS(phiin)
      lon = adjl0(lamin - lon0)
      coslon = COS(lon)
      g = sinph0 * sinphi + cosph0 * cosphi * coslon
      if (g < (C(1) / p)) {
         for15 = ERR
         call remark("f15: cannot project.")
         }
      else {
         for15 = OK
         g = a * (p - C(1)) / (p - g)
         xout = x0 + g * cosphi * SIN(lon)
         yout = y0 + g * (cosph0 * sinphi - sinph0 * cosphi * %
            coslon)
         }
   return
   end
```

```
#-h- INV15 1320 asc SAT., 15  JAN., 1983 10:21:18.71
#inv15 - general vertical near-side perspective - inverse
   integer function inv15(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adjlO, com, con, coslon, cosphi, cosz, ABS, darsin,
          ATAN2, COS, SIN, SQRT, r, rh, sinphi, sinz, x, y, z

      x = xin - xO
      y = yin - yO
      rh = SQRT(x * x + y * y)
      r = rh / a
      con = p - C(1)
      com = p + C(1)
      if (r > SQRT(con / com)) {
         call remark("i15: data error.")
         inv15 = ERR
         }
      else {
         inv15 = OK
         sinz = (p - SQRT(C(1) - r * r * com / con)) / %
                (con / r + r / con)
         z = darsin(sinz)
         sinz = SIN(z)
         cosz = COS(z)
         lamout = lonO
         if (ABS(rh) <= EPSILON10)
            phiout = latO
         else {
            phiout = darsin(cosz * sinphO + y * sinz * %
                  cosphO / rh)
            if (ABS(ABS(latO) - HALFPI) <= EPSILON10)
               lamout = adjlO(lonO + ATAN2(SIGN(x, latO),
                     SIGN(y, -latO)))
            else {
               con = cosz - sinphO * SIN(phiout)
               if (con ^= C(0))
                 lamout = adjlO(lonO + ATAN2((x * sinz * cosphO),
                       (con * rh)))
               }
            }
         }

   return
   end
```

```
#-h- PRJ16 1333 asc WED.,  2  FEB., 1983 11:3:29.52
#-h- STR16 180 asc FRI., 14  JAN., 1983 8:15:10.52
# str16 - sinusoidal - structure
#    a      - reference sphere radius
#    lon0   - central meridian
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing


#-h- SET16 129 asc FRI., 14  JAN., 1983 8:15:11.26
# set16 - sinusoidal - initialization
   integer function set16(consts)
   PREC  consts(ARB)

      set16 = OK

   return
   end

#-h- FOR16 277 asc FRI., 14  JAN., 1983 8:15:12.06
# for16 - sinusoidal - forward
   integer function for16(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, COS

      for16 = OK
      xout = x0 + a * adjl0(lamin - lon0) * COS(phiin)
      yout = y0 + a * phiin

   return
   end

#-h- INV16 543 asc FRI., 14  JAN., 1983 8:20:12.23
#inv16 - sinusoidal - inverse
   integer function inv16(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adjl0, ABS, COS

      phiout = (yin - y0) / a
      if (ABS(phiout) > HALFPI) {
         call remark("i16: y out of range.")
         inv16 = ERR
         }
      else {
         inv16 = OK
         if (ABS(ABS(phiout) - HALFPI) <= EPSILON10)
            lamout = lon0
         else
.           lamout = adjl0(lon0 + (xin - x0) / (a * %
                COS(phiout)))
         }

   return
   end
```

```
#-h- PRJ17 1218 asc WED.,  2  FEB., 1983 11:3:31.57
#-h- STR17 170 asc FRI., 14  JAN., 1983 7:58:24.53
# str16 - equirectangular(plate caree) - structure
#    a      - reference sphere radius
#    lon0   - central meridian
#    x0     - false easting
#    y0     - false northing


#-h- SET17 147 asc FRI., 14  JAN., 1983 7:58:25.26
# set17 - equirectangular(plate caree) - initialization
   integer function set17(consts)
   PREC  consts(ARB)

      set17 = OK

   return
   end


#-h- FOR17 275 asc FRI., 14  JAN., 1983 7:58:26.04
# for17 - equirectangular(plate caree) - forward
   integer function for17(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0

      for17 = OK
      xout = x0 + a * adjl0(lamin - lon0)
      yout = y0 + a * phiin

  return
  end


#-h- INV17 422 asc FRI., 14  JAN., 1983 7:58:26.98
#inv17 - equirectangular(plate caree) - inverse
   integer function inv17(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adjl0

      phiout = (yin - y0) / a
      if (ABS(phiout) > HALFPI) {
         call remark("i17: invalid y coord.")
         inv17 = ERR
         }
      else {
         inv17 = OK
         lamout = adjl0(lon0 + (xin - x0) / a)
         }

   return
   end
```

```
#-h- PRJ18 1161 asc WED.,  2  FEB., 1983 11:3:33.44
#-h- STR18 159 asc FRI., 14  JAN., 1983 7:45:29.63
# str18 - miller cylindrical - structure
#    a      - reference sphere radius
#    lon0   - central meridian
#    x0     - false easting
#    y0     - false northing


#-h- SET18 137 asc FRI., 14  JAN., 1983 7:45:30.33
# set18 - miller cylindrical - initialization
   integer function set18(consts)
   PREC  consts(ARB)

      set18 = OK

   return
   end


#-h- FOR18 327 asc FRI., 14  JAN., 1983 7:45:31.13
# for18 - miller cylindrical - forward
   integer function for18(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adj10, LOG, TAN

      for18 = OK
      xout = x0 + a * adj10(lamin - lon0)
      yout = y0 + a * LOG(TAN(FORTPI + phiin / C(2.5))) * %
          C(1.25)

   return
   end


#-h- INV18 334 asc FRI., 14  JAN., 1983 7:45:31.98
#inv18 - miller cylindrical - inverse
   integer function inv18(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adj10, ATAN, EXP

      inv18 = OK
      lamout = adj10(lon0 + (xin - x0) / a)
      phiout = C(2.5) * (ATAN(EXP((yin - y0) / a / C(1.25))) %
          - FORTPI)

   return
   end
```

```
#-h- PRJ19 5126 asc FRI.,  4  FEB., 1983 9:28:33.81
#-h- STR19 189 asc THU., 13  JAN., 1983 15:10:3.91
# str16 - sinusoidal - structure
#    a      - reference sphere radius
#    lon0   - central meridian
#    x0     - false easting
#    y0     - false northing
define(NITER, 55)
define(TOL,  C(.7))


#-h- SET19 136 asc THU., 13  JAN., 1983 15:10:4.81
# set19 - van der grinten I - initialization
   integer function set19(consts)
   PREC  consts(ARB)

      set19 = OK

   return
   end


#-h- FOR19 1188 asc THU., 13  JAN., 1983 15:22:54.98
# for19 - van der grinten I - forward
   integer function for19(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, al, asq, con, costht, ABS, darsin, COS,
       SIGN, SIN, SQRT, TAN, g, gsq, lon, m, msq, sintht, theta

      for19 = OK
      lon = adjl0(lamin - lon0)
      theta = darsin( ABS(phiin / HALFPI) )
      if (ABS(phiin) <= EPSILON10) {
         xout = x0 + a * lon
         yout = y0
         }
      else if (ABS(lon) <= EPSILON10) {
         xout = x0
         yout = y0 + PI * a * TAN(C(.5) * theta)
         }
      else {
         al = C(.5) * ABS(PI / lon - lon / PI)
         asq = al * al
         sintht = SIN(theta)
         costht = COS(theta)
         g = costht / (sintht + costht - C(1))
         gsq = g * g
         m = g * (C(2) / sintht - C(1))
         msq = m * m
         con = SIGN(PI * a * (al * (g - msq) + SQRT(asq * %
            (g - msq) **2 - (msq + asq) * (gsq - msq))) / %
            (msq + asq), lon)
         xout = x0 + con
         con = ABS(con / (PI * a))
         yout = y0 + SIGN(PI * a * SQRT(C(1) - con * con - %
```

```
                    C(2) * al * con) , phiin)
          }

   return
   end


#-h- INV19 3406 asc FRI., 14  JAN., 1983 7:28:20.96
#inv19 - van der grinten I - inverse
   integer function inv19(xin, yin, lamout, phiout, consts)
   PREC  xin, yin, lamout, phiout, consts(ARB)
   PREC  adjlO, al, asq, cmm, cnn, com, con, costht, d, ABS,
         darsin, ATAN, COS, dphi, SIGN, SIN, SQRT, TAN, g, gsq,
         h, j, lon, m, msq, phi, sintht, theta, x, y, y1
   integer i

 i = 0
      inv19 = OK
      x = xin - x0
      y = yin - y0
      con = ABS(y / (PI * a))
      theta = C(2) * ATAN(con)
      if (ABS(x) <= EPSILON10) {
         lamout = lon0
         phiout = HALFPI * SIN(theta)
         }
      else if (ABS(y) <= EPSILON10) {
         lamout = adjlO(lon0 + x / a)
         phiout = C(0)
         }
      else if (SQRT(x * x + y * y) > PI * a) {
         call remark("i19: data error.")
         inv19 = ERR
         }
      else {
         cnn = con * con
         com = ABS(x / (PI * a))
         cmm = com * com
         al = (C(1) - cmm - cnn) / (C(2) * com)
         lamout = adjlO(lon0 + SIGN(PI * ( - al + SQRT(al * al %
             + C(1))) , x))
         phi = theta
         if (con <= TOL)
            for(i = 1; i <= NITER ; i = i + 1) {
               theta = darsin(phi / HALFPI)
               sintht = SIN(theta)
               costht = COS(theta)
               g = costht / (sintht + costht - C(1))
               d = con / sintht - C(1) / (C(1) + costht)
               h = C(2) - sintht
               j = TAN(C(.5) * theta)
               dphi = (cmm + cnn - C(2) * d * g * h - j * j) * %
                  PI * costht / (C(4) * (g * h * (con * costht %
                  / (C(1) - costht) + j) / (C(1) + costht) + d %
                  * g * ((C(1) + C(2) * costht * costht) / %
```

```
                costht + h * (costht - sintht) / (sintht %
                 + costht - C(1))) - %
                j * (j * j + C(1))))
            phi = phi - dphi
            if (ABS(dphi) < EPSILON10)
               break
            }
         else {
            lon = adjl0(lamout - lon0)
            for(i = 1; i <= NITER ; i = i + 1) {
               if (ABS(phi) <= EPSILON10)
                  y1 = C(0)
               else if (ABS(lon) >= EPSILON10)
                  y1 = PI * a * TAN(C(.5) * darsin(ABS(phi %
                        / HALFPI)))
               else {
                  theta = darsin(ABS(phi /HALFPI))
                  al = C(.5) * ABS(PI / lon - lon / PI)
                  asq = al * al
                  sintht = SIN(theta)
                  costht = COS(theta)
                  g = costht / (sintht + costht - C(1))
                  gsq = g * g
                  m = g * (C(2) / sintht - C(1))
                  msq = m * m
                  con = ABS((al * (g - msq) + SQRT(asq * (g - %
                     msq) ** 2 - (msq + asq) * (gsq - msq))) / %
                     (msq + asq))
                  y1 = SIGN(PI * a * SQRT(C(1) - con * con -  %
                           C(2) * al * con) , phi)
                  }
               dphi = ((ABS(y) - y1) / (PI * a - y1)) * %
                    (HALFPI - phi)
               phi = phi + dphi
               if (ABS(dphi) < EPSILON10)
                  break
               }
            }
         phiout = SIGN(phi , y)
         if (i > NITER) {
            call remark("i19: convergence failure.")
            inv19 = ERR
            }
         }
 write(6,9700) i,con
9700 format("i:",i6,1p,e15.5)
  return
  end
```

```
#-h- PRJ20 5032 asc FRI.,  4  FEB., 1983 9:28:39.53
#-h- STR20 877 asc THU., 13  JAN., 1983 13:51:39.68
# str20 - oblique mercator - structure
#    a      - semi-major axis
#    es     - eccentricity squared
#    lat0   - central parallel
#    x0     - false easting
#    y0     - false northing
define(ks0,      consts( 3)) # central scale factoe
define(alpha,    consts( 4)) # azimuth of central line
define(lonc,     consts( 5)) # lon. of azimuth pt.
define(lon1,     consts( 9)) # lon of first point
define(lat1,     consts(10)) # lat of first point
define(lon2,     consts(11)) # lon of second point
define(lat2,     consts(12)) # lat of second point
define(dmode,    consts(13)) # mode selection
# local constants
define(e,        consts(14))
define(gamma,    consts(15))
define(al,       consts(16))
define(bl,       consts(17))
define(el,       consts(18))
define(singam,   consts(19))
define(cosgam,   consts(20))
define(sinalf,   consts(21))
define(cosalf,   consts(22))
define(TOL,    1 EXPLET -7)


#-h- SET20 1797 asc THU., 13  JAN., 1983 14:24:18.51
# set20 - oblique mercator(hotine) - initialization
   integer function set20(consts)
   PREC  adjl0, consts(ARB), com, con, cosph0, d,
         ABS, darsin, ATAN, ATAN2, COS, SIGN, SIN,
         SQRT, TAN, f, h, l, sinph0, tsfn0

      set20 = OK
      e = SQRT(es)
      sinph0 = SIN(lat0)
      cosph0 = COS(lat0)
      con = C(1) - es * sinph0 * sinph0
      com = SQRT(C(1) - es)
      bl = SQRT(C(1) + es * cosph0 ** 4 / (C(1) - es))
      al = a * bl * ks0 * com / con
      d = bl * com / (cosph0 * SQRT(con))
      f = d + SIGN(SQRT(d * d - C(1)) , lat0)
      el = f * tsfn0(e, lat0, sinph0) ** bl
      if (dmode ^= C(0)) {
         gamma = darsin(SIN(alpha) / d)
         lon0 = lonc - darsin((C(.5) * (f - C(1) / f)) * %
            TAN(gamma)) / bl
         con = ABS(lat0)
         if (con <= EPSILON10 | ABS(con - HALFPI) <= EPSILON10) {
            call remark("s20: input data error.")
            set20 = ERR
```

```
            }
         }
      else {
         h = tsfn0(e, lat1, SIN(lat1)) ** bl
         l = tsfn0(e, lat2, SIN(lat2)) ** bl
         f = el / h
         lon0 = C(.5) * (lon1 + lon2) - ATAN((el * el - 1 * h) %
            / (el * el + 1 * h))  * TAN(C(.5) * bl * %
            adjl0(lon1 - lon2)) /  ((1 - h) / (1 + h)) ) / bl
         gamma = ATAN2(SIN(bl * adjl0(lon1 - lon0)) ,
             C(.5) * (f - C(1) / f))
         alpha = darsin(d * SIN(gamma))
         con = ABS(lat1)
         if ((ABS(lat1 - lat2) <= EPSILON10) |
            (con <= EPSILON10 | ABS(con - HALFPI) <= EPSILON10) |
            (ABS(ABS(lat0) - HALFPI) <= EPSILON10)) {
            call remark("s20: data error.")
            set20 = ERR
            }
         }
      singam = SIN(gamma)
      cosgam = COS(gamma)
      sinalf = SIN(alpha)
      cosalf = COS(alpha)

   return
   end


#-h- FOR20 1222 asc THU., 13  JAN., 1983 14:30:43.71
# for20 - oblique mercator(hotine) - forward
   integer function for20(lamin, phiin, xout, yout, consts)
   PREC  lamin, phiin, xout, yout, consts(ARB)
   PREC  adjl0, con, ABS, ATAN, COS, LOG, dlon, SIGN, SIN,
         lon, q, s, sinph0, tsfn0, ul, us, vl, vs

      for20 = OK
      dlon = adjl0(lamin - lon0)
      vl = SIN(bl * dlon)
      if (ABS(ABS(phiin) - HALFPI) <= EPSILON10) {
         ul = singam * SIGN(C(1) , phiin)
         us = al * phiin / bl
         }
      else {
         q = el / tsfn0(e, phiin, SIN(phiin)) ** bl
         s = C(.5) * (q - C(1) / q)
         ul = C(2) * (s * singam - vl * cosgam) / (q + C(1) / q)
         con = COS(bl * dlon)
         if (ABS(con) >= TOL) {
            us = al * ATAN((s * cosgam + vl * singam) / con) / bl
            if (con < C(0))
                us = us + PI * al / bl
            }
         else
            us = al * bl * dlon
```

```
        }
    if (ABS(ABS(ul) - C(1)) <= EPSILON10) {
        call remark("f20: data error.")
        for20 = ERR
        }
    else {
        vs = C(.5) * al * LOG((C(1) - ul) / (C(1) + ul)) / bl
        xout = x0 + vs * cosalf + us * sinalf
        yout = y0 + us * cosalf - vs * sinalf
        }

return
end


#-h- INV20 926 asc THU., 13  JAN., 1983 14:33:53.93
#inv20 - oblique mercator(hotine) - inverse
    integer function inv20(xin, yin, lamout, phiout, consts)
    PREC  xin, yin, lamout, phiout, consts(ARB)
    PREC  adjl0, ABS, ATAN2, COS, LOG, SIGN, SIN, SQRT,
          phi20, q, s, ul, us, vl, vs, x, y
    integer err

        inv20 = OK
        x = xin - x0
        y = yin - y0
        vs = x * cosalf - y * sinalf
        us = y * cosalf + x * sinalf
        q = EXP(- bl * vs / al)
        s = C(.5) * (q - C(1) / q)
        vl = SIN(bl * us / al)
        ul = C(2) * (vl * cosgam + s * singam) / (q + C(1) / q)
        if (ABS(ABS(ul) - C(1)) < EPSILON10) {
            lamout = lon0
            phiout = SIGN(HALFPI , ul)
            }
        else {
            phiout = phi20(e,(el / SQRT((C(1) + ul) / %
                (C(1) - ul))) ** (C(1) / bl) , err)
            lamout = adjl0(lon0 - ATAN2((s * cosgam - vl * singam) ,
                COS(bl * us / al)) / bl)
            inv20 = err
            }

return
end
```

Appendix D

The following is a listing of the support library required by the projection procedures. In practice this can normally be compiled as one file and 'searched' by the linking-loader when creating a run-time module.

Contents

	Page
# sym - supplimentary macros	D-2
# adjl0 - adjust longitude to modulo 180 degrees	D-2
# darcos - arc cosine	D-2
# darsin - arc sine	D-2
# e0fn0 - determine constant e0	D-2
# e1fn0 - determine constant e1	D-3
# e2fn0 - determine constant e2	D-3
# e3fn0 - determine constant e3	D-3
# mlfn0 - determine constant M	D-3
# msfn0 - determine constant small m	D-4
# phi10 - determine latitude angle phi-1	D-4
# phi20 - determine latitude angle phi-2	D-5
# phi30 - determine latitude angle phi-3	D-5
# phi40 - determine latitude angle phi-4	D-6
# qsfn0 - determine small q	D-6
# tsfn0 - determine small t	D-7

```
#-h- SYM 138 asc TUE.,  1  FEB., 1983 11:35:7.16
# sym - supplimentary macros
include prjsym
define(MAXITER,  15) # iteration limit
define(TOL,    1 EXPLET -10)
define(EPSLN, 1 EXPLET -7)


#-h- ADJLO 204 asc TUE.,  1  FEB., 1983 10:58:14.51
# adjlo - adjust longitude to modulo 180 degrees
   PREC  function adjlo(lon)
   PREC  lon, SIGN, ABS

      adjlo = lon
      while( ABS(adjlo) > PI )
         adjlo = adjlo - SIGN(TWOPI, adjlo)

   end


#-h- DARCOS 143 asc TUE.,  1  FEB., 1983 10:59:56.47
# darcos - arc cosine
   PREC  function darcos(v)
   PREC  v, ATAN2, SQRT

      darcos = ATAN2( SQRT( C(1)  - v * v ) , v )

   return
   end


#-h- DARSIN 141 asc TUE.,  1  FEB., 1983 10:59:57.38
# darsin - arc sine
   PREC  function darsin(v)
   PREC  v, ATAN2, SQRT

      darsin = ATAN2( v , SQRT( C(1)  - v * v ) )

   return
   end


#-h- E0FN0 205 asc TUE.,  1  FEB., 1983 10:49:21.89
# e0fn0 - determine constant e0
   PREC  function e0fn0(eccnts)
   PREC  eccnts

      e0fn0 = C(1) - C(.25) * eccnts * (C(1) + eccnts / C(16) %
             * (C(3) + C(1.25) * eccnts))

   return
   end
```

```
#-h- E1FN0 203 asc TUE.,  1  FEB., 1983 10:49:22.85
# e1fn0 - determine constant e1
   PREC  function e1fn0(eccnts)
   PREC  eccnts

      e1fn0 = C(.375) * eccnts * (C(1) + C(.25) * eccnts * %
              (C(1) + C(.46875) * eccnts))

   return
   end


#-h- E2FN0 187 asc TUE.,  1  FEB., 1983 10:49:23.75
# e2fn0 - determine constant e2
   PREC  function e2fn0(eccnts)
   PREC  eccnts

      e2fn0 = C(.05859375) * eccnts * eccnts * (C(1) + %
              C(.75) * eccnts)

   return
   end


#-h- E3FN0 209 asc TUE.,  1  FEB., 1983 10:49:24.97
# e3fn0 - determine constant e3
   PREC  function e3fn0(eccent)
   PREC  eccent, con, com

      con = C(1) + eccent
      com = C(1) - eccent
      e3fn0 = SQRT((con ** con) * (com ** com))

   return
   end


#-h- MLFN0 201 asc TUE.,  1  FEB., 1983 11:8:4.29
# mlfn0 - determine constant M
   PREC  function mlfn0(e0, e1, e2, phi)
   PREC  e0, e1, e2, phi, SIN

      mlfn0 = e0 * phi - e1 * SIN(C(2) * phi) + e2 * %
         SIN(C(4) * phi)

   return
   end
```

```
#-h- MSFN0 221 asc TUE.,  1  FEB., 1983 10:49:39.76
# msfn0 - determine constant small m
   PREC  function msfn0(eccent, sinphi, cosphi)
   PREC  eccent, sinphi, cosphi, con, SQRT

      con = eccent * sinphi
      msfn0 = cosphi / SQRT(C(1) - con * con)

   return
   end


#-h- PHI10 901 asc TUE.,  1  FEB., 1983 10:49:41.28
# phi10 - determine latitude angle phi-1
   PREC  function phi10(eccent, qs, err)
   PREC  eccent, qs, darsin, eccnts, phi, sinpi, cospi,
          con, com, dphi, SIN, COS, LOG
   integer i, err

      phi10 = darsin(C(.5) * qs)
      if (eccent < EPSLN)
         return
      eccnts = eccent * eccent
      phi = phi10
      for (i = 1; i <= MAXITER ; i = i + 1) {
         sinpi = SIN(phi)
         cospi = COS(phi)
         con = eccent * sinpi
         com = C(1) - con * con
         dphi = C(.5) * com * com / cospi * (qs / (C(1) - %
            eccnts) - sinpi / com + C(.5) / eccent * %
            LOG((C(1) - con) / (C(1) + con)))
         phi = phi + dphi
         if (ABS(dphi) <= TOL)
            break
         }
      phi10 = phi
      if (i > MAXITER) {
         err = ERR
         call remark("phi10 - convergence failure.")
         }
      else
         err = OK

   return
   end
```

```
#-h- PHI20 710 asc TUE.,  1  FEB., 1983 10:49:42.88
# phi20 - determine latitude angle phi-2
   PREC  function phi20(eccent, ts, err)
   PREC  eccent, ts, eccnth, phi, sinpi, SIN, con,
          ABS, dphi, ATAN
   integer i, err

      eccnth = C(.5) * eccent
      phi = HALFPI - C(2) * ATAN(ts)
      for (i = 1; i <= MAXITER ; i = i + 1) {
         sinpi = SIN(phi)
         con = eccent * sinpi
         dphi = HALFPI - C(2) * ATAN(ts * ((C(1) - con) / %
                (C(1) + con)) ** eccnth) - phi
         phi = phi + dphi
         if (ABS(dphi)  <= TOL)
            break
      }
      phi20 = phi
      if (i > MAXITER) {
         err = ERR
         call remark("phi20 - convergence failure.")
         }
      else
         err = OK

   return
   end


#-h- PHI30 562 asc TUE.,  1  FEB., 1983 10:52:16.12
# phi30 - determine latitude angle phi-3
   PREC  function phi30(ml, e0, e1, e2, err)
   PREC  ml, e0, e1, e2, phi, SIN, ABS, dphi
   integer i, err

      phi = ml
      for (i = 1; i <= MAXITER; i = i + 1) {
         dphi = (ml + e1 * SIN(C(2) * phi) - e2 * SIN(C(4) %
             * phi)) / e0 - phi
         phi = phi + dphi
         if (ABS(dphi) <= TOL)
            break
         }
      phi30 = phi
      if (i > MAXITER) {
         err = ERR
         call remark("phi30 - convergence failure.")
         }
      else
         err = OK

   return
   end
```

```
#-h- PHI4O 1191 asc TUE.,  1  FEB., 1983 10:49:45.63
# phi4O - determine latitude angle phi-4
   PREC  function phi4O(eccnts, eO, e1, e2, p, b, c, err)
   PREC  eccnts, eO, e1, e2, p, b, c, phi, sinphi, tanphi,
          SIN, TAN,    SQRT, sin2ph, ml, mlp, con1, con2, con3,
          dphi, ABS, COS
   integer i, err

      phi = p
      for (i = 1; i <= MAXITER; i = i + 1) {
         sinphi = SIN(phi)
         tanphi = TAN(phi)
         c = tanphi * SQRT(C(1) - eccnts * sinphi * sinphi)
         sin2ph = SIN(C(2) * phi)
         ml = eO * phi - e1 * sin2ph + e2 * SIN(C(4) * phi)
         mlp = eO - C(2) * e1 * COS(C(2) * phi) + C(4) * e2 * %
               COS(C(4) * phi)
         con1 = C(2) * ml + c * (ml * ml + b) - C(2) * p *  %
                (c * ml + C(1))
         con2 = eccnts * sin2ph * (ml * ml + b - C(2) * p * %
            ml) /(C(2) * c)
         con3 = C(2) * (p - ml) * (c * mlp - C(2) / sin2ph) %
             - C(2) * mlp
         dphi = con1 /(con2 + con3)
         phi = phi + dphi
         if (ABS(dphi) <= TOL)
            break
      }
      phi4O = phi
      if (i > MAXITER) {
         err = ERR
         call remark("phi4O - convergence failure.")
         }
      else
         err = OK

   return
   end


#-h- QSFNO 405 asc TUE.,  1  FEB., 1983 10:49:47.88
# qsfnO - determine small q
   PREC  function qsfnO(eccent, sinphi, cosphi)
   PREC  eccent, sinphi, cosphi, con, LOG

      if (eccent >= EPSLN) {
         con = eccent * sinphi
         qsfnO = (C(1) - eccent * eccent) * (sinphi /(C(1) - %
            con * con) - (C(.5) / eccent) * LOG((C(1) - con) %
            / (C(1) + con)))
         }
      else
         qsfnO = C(2) * sinphi
   return
   end
```

```
#-h- TSFN0 287 asc TUE.,  1  FEB., 1983 10:49:49.03
# tsfn0 - determine small t
   PREC  function tsfn0(eccent, phi, sinphi)
   PREC  eccent, phi, sinphi, con, com, TAN

      con = eccent * sinphi
      com = C(.5) * eccent
      con = ((C(1) - con) /(C(1) + con)) ** com
      tsfn0 = TAN(C(.5) * (HALFPI - phi)) / con

   return
   end
```

Appendix E

This appendix provides listings of test values for each of the projections in this report. In each case, a set of cartographic attributes for the test are displayed in the 'consts' list (note that although radian arguments are required by the procedures, they are displayed in degrees for clarity). For a range of input geographic coodinates ('lamin-phiin'), the forward cartesian values are determined and listed ('xout'-'yout') and then passed as input to the inverse projection procedure. The absolute difference of the inverse geographic coordinates and original input coordinates are then tabulated in the last two columns ('del lam'-'del phi'). If all processes are functioning properly, the variation in the differences should only reflect the mantissa significance capability of the host computer.

Test of projection 1 Universal Transverse Mercator (UTM)

Attributes

consts(1): 6378206.4000
consts(2): .67686580000E-02
consts(5): -75.000000000
consts(6): 45.500000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	!del lam!	!del phi!
-76.00	25.00	399085.20	2765162.13	0.0E+00	5.8E-11
-76.00	30.00	403547.90	3319026.20	0.0E+00	1.2E-10
-76.00	35.00	408744.64	3873302.29	0.0E+00	0.0E+00
-76.00	40.00	414637.30	4428026.02	0.0E+00	0.0E+00
-76.00	45.00	421182.37	4983219.57	0.0E+00	1.2E-10
-76.00	50.00	428331.20	5538890.89	0.0E+00	1.2E-10
-75.00	25.00	500000.00	2764789.92	0.0E+00	0.0E+00
-75.00	30.00	500000.00	3318605.33	0.0E+00	5.8E-11
-75.00	35.00	500000.00	3872845.49	0.0E+00	0.0E+00
-75.00	40.00	500000.00	4427547.16	0.0E+00	1.2E-10
-75.00	45.00	500000.00	4982733.19	0.0E+00	1.2E-10
-75.00	50.00	500000.00	5538411.76	0.0E+00	0.0E+00
-74.00	25.00	600914.80	2765162.13	0.0E+00	5.8E-11
-74.00	30.00	596452.10	3319026.20	0.0E+00	1.2E-10
-74.00	35.00	591255.36	3873302.29	0.0E+00	0.0E+00
-74.00	40.00	585362.70	4428026.02	0.0E+00	0.0E+00
-74.00	45.00	578817.63	4983219.57	0.0E+00	1.2E-10
-74.00	50.00	571668.80	5538890.89	0.0E+00	1.2E-10
-73.00	25.00	701849.49	2766279.19	4.7E-10	2.3E-10
-73.00	30.00	692919.00	3320289.29	4.7E-10	1.2E-10
-73.00	35.00	682520.32	3874673.10	4.7E-10	2.3E-10
-73.00	40.00	670729.98	4429462.97	4.7E-10	2.3E-10
-73.00	45.00	657635.29	4984679.02	4.7E-10	1.2E-10
-73.00	50.00	643333.82	5540328.49	4.7E-10	2.3E-10
-72.00	25.00	802824.00	2768142.45	2.3E-09	4.7E-10
-72.00	30.00	789415.50	3322395.95	3.0E-09	5.2E-10
-72.00	35.00	773804.43	3876959.21	3.7E-09	7.0E-10
-72.00	40.00	756106.38	4431859.11	4.4E-09	3.5E-10
-72.00	45.00	736453.01	4987112.42	4.7E-09	2.3E-10
-72.00	50.00	714991.28	5542725.22	5.4E-09	5.8E-10
-71.00	25.00	903858.24	2770754.19	1.6E-08	2.9E-10
-71.00	30.00	885956.39	3325348.46	2.1E-08	4.1E-10
-71.00	35.00	865117.26	3880162.74	2.6E-08	2.3E-10
-71.00	40.00	841496.41	4435216.32	3.1E-08	2.3E-10
-71.00	45.00	815270.74	4990521.28	3.5E-08	1.0E-09
-71.00	50.00	786637.34	5546082.16	3.9E-08	1.9E-09

Test of projection 3 Alber's equal area

Attributes

consts(1): 6378206.4000
consts(2): .67686580000E-02
consts(3): 29.500000000
consts(4): 45.500000000
consts(5): 0.0000000000
consts(6): 40.000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	!del lam!	!del phi!
-15.00	25.00	-1526420.62	-1548089.38	1.2E-10	2.9E-10
-15.00	30.00	-1439774.98	-1003708.85	0.0E+00	7.6E-10
-15.00	35.00	-1352158.65	-453229.62	0.0E+00	5.8E-10
-15.00	40.00	-1264109.17	99970.99	5.8E-11	4.7E-10
-15.00	45.00	-1176284.18	651761.20	5.8E-11	2.3E-10
-15.00	50.00	-1089503.29	1196991.48	0.0E+00	3.5E-10
0.00	25.00	0.00	-1668805.04	0.0E+00	8.1E-10
0.00	30.00	0.00	-1117572.21	0.0E+00	1.4E-09
0.00	35.00	0.00	-560163.92	0.0E+00	1.0E-09
0.00	40.00	0.00	.00	0.0E+00	9.3E-10
0.00	45.00	0.00	558735.78	0.0E+00	1.2E-10
0.00	50.00	0.00	1110829.05	0.0E+00	3.5E-10
15.00	25.00	1526420.62	-1548089.38	1.2E-10	2.9E-10
15.00	30.00	1439774.98	-1003708.85	0.0E+00	7.6E-10
15.00	35.00	1352158.65	-453229.62	0.0E+00	5.8E-10
15.00	40.00	1264109.17	99970.99	5.8E-11	4.7E-10
15.00	45.00	1176284.18	651761.20	5.8E-11	2.3E-10
15.00	50.00	1089503.29	1196991.48	0.0E+00	3.5E-10
30.00	25.00	3014891.80	-1188943.59	1.2E-10	5.8E-11
30.00	30.00	2843754.68	-664949.59	0.0E+00	7.6E-10
30.00	35.00	2670700.31	-135085.28	0.0E+00	5.8E-10
30.00	40.00	2496790.42	397398.50	0.0E+00	4.7E-10
30.00	45.00	2323323.91	928524.70	1.2E-10	2.3E-10
30.00	50.00	2151919.66	1453336.62	0.0E+00	3.5E-10
45.00	25.00	4428407.62	-600296.66	1.2E-10	2.3E-10
45.00	30.00	4177033.78	-109716.57	1.2E-10	0.0E+00
45.00	35.00	3922843.80	386359.48	0.0E+00	2.3E-10
45.00	40.00	3667397.19	884887.97	1.2E-10	2.3E-10
45.00	45.00	3412601.84	1382145.45	1.2E-10	4.7E-10
45.00	50.00	3160835.63	1873491.30	0.0E+00	3.5E-10

Test of projection 4 Lambert conformal conic

Attributes

```
consts( 1):     6378206.4000
consts( 2):     .67686580000E-02
consts( 3):     33.000000000
consts( 4):     45.000000000
consts( 5):     0.0000000000
consts( 6):     40.000000000
consts( 7):     0.0000000000
consts( 8):     0.0000000000
```

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	!del lam!	!del phi!
-15.00	25.00	-1542643.24	-1541818.71	0.0E+00	0.0E+00
-15.00	30.00	-1450296.14	-987445.20	5.8E-11	5.8E-10
-15.00	35.00	-1359078.38	-439851.31	5.8E-11	1.2E-10
-15.00	40.00	-1268331.47	104915.99	0.0E+00	5.8E-10
-15.00	45.00	-1177373.94	650947.70	5.8E-11	3.5E-10
-15.00	50.00	-1085466.32	1202682.88	5.8E-11	1.2E-10
0.00	25.00	0.00	-1669425.69	0.0E+00	1.7E-10
0.00	30.00	0.00	-1107413.26	0.0E+00	0.0E+00
0.00	35.00	0.00	-552273.85	0.0E+00	3.5E-10
0.00	40.00	0.00	.00	0.0E+00	7.0E-10
0.00	45.00	0.00	553555.69	0.0E+00	5.8E-10
0.00	50.00	0.00	1112893.44	0.0E+00	5.8E-10
15.00	25.00	1542643.24	-1541818.71	0.0E+00	0.0E+00
15.00	30.00	1450296.14	-987445.20	5.8E-11	5.8E-10
15.00	35.00	1359078.38	-439851.31	5.8E-11	1.2E-10
15.00	40.00	1268331.47	104915.99	0.0E+00	5.8E-10
15.00	45.00	1177373.94	650947.70	5.8E-11	3.5E-10
15.00	50.00	1085466.32	1202682.88	5.8E-11	1.2E-10
30.00	25.00	3043350.99	-1162466.68	0.0E+00	0.0E+00
30.00	30.00	2861167.16	-630802.28	1.2E-10	5.8E-10
30.00	35.00	2681211.32	-105639.78	0.0E+00	0.0E+00
30.00	40.00	2502184.38	416811.91	0.0E+00	2.3E-10
30.00	45.00	2322741.92	940476.21	0.0E+00	1.2E-10
30.00	50.00	2141425.13	1469610.36	0.0E+00	1.2E-10
45.00	25.00	4461327.73	-541681.95	0.0E+00	4.1E-10
45.00	30.00	4194259.70	-47179.53	0.0E+00	8.1E-10
45.00	35.00	3930457.73	441275.46	0.0E+00	1.2E-10
45.00	40.00	3668017.46	927209.11	0.0E+00	2.3E-10
45.00	45.00	3404968.07	1414270.63	1.2E-10	1.2E-10
45.00	50.00	3139171.05	1906419.66	0.0E+00	1.2E-10

Test of projection 5 Mercator

Attributes

consts(1): 6378206.4000
consts(2): .67686580000E-02
consts(5): 0.0000000000
consts(6): 0.0000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

forward			inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
0.00	0.00	0.00	0.00	0.0E+00	0.0E+00
0.00	5.00	0.00	553548.58	0.0E+00	4.4E-11
0.00	10.00	0.00	1111404.92	0.0E+00	8.7E-11
0.00	15.00	0.00	1678043.12	0.0E+00	8.7E-11
0.00	20.00	0.00	2258286.10	0.0E+00	1.7E-10
0.00	25.00	0.00	2857523.32	0.0E+00	2.3E-10
0.00	30.00	0.00	3481989.83	0.0E+00	5.8E-10
0.00	35.00	0.00	4139145.66	0.0E+00	1.2E-10
0.00	40.00	0.00	4838218.95	0.0E+00	2.3E-10
0.00	45.00	0.00	5591021.00	0.0E+00	1.2E-10
0.00	50.00	0.00	6413230.50	0.0E+00	1.2E-10
0.00	55.00	0.00	7326528.18	0.0E+00	1.2E-10
0.00	60.00	0.00	8362377.87	0.0E+00	1.2E-10
0.00	65.00	0.00	9569276.28	0.0E+00	2.3E-10
0.00	70.00	0.00	11028186.67	0.0E+00	2.3E-10
0.00	75.00	0.00	12890594.86	0.0E+00	0.0E+00
0.00	80.00	0.00	15496270.75	0.0E+00	0.0E+00
0.00	85.00	0.00	19928981.89	0.0E+00	2.3E-10
10.00	0.00	1113207.02	0.00	2.9E-11	0.0E+00
10.00	5.00	1113207.02	553548.58	2.9E-11	4.4E-11
10.00	10.00	1113207.02	1111404.92	2.9E-11	8.7E-11
10.00	15.00	1113207.02	1678043.12	2.9E-11	8.7E-11
10.00	20.00	1113207.02	2258286.10	2.9E-11	1.7E-10
10.00	25.00	1113207.02	2857523.32	2.9E-11	2.3E-10
10.00	30.00	1113207.02	3481989.83	2.9E-11	5.8E-10
10.00	35.00	1113207.02	4139145.66	2.9E-11	1.2E-10
10.00	40.00	1113207.02	4838218.95	2.9E-11	2.3E-10
10.00	45.00	1113207.02	5591021.00	2.9E-11	1.2E-10
10.00	50.00	1113207.02	6413230.50	2.9E-11	1.2E-10
10.00	55.00	1113207.02	7326528.18	2.9E-11	1.2E-10
10.00	60.00	1113207.02	8362377.87	2.9E-11	1.2E-10
10.00	65.00	1113207.02	9569276.28	2.9E-11	2.3E-10
10.00	70.00	1113207.02	11028186.67	2.9E-11	2.3E-10
10.00	75.00	1113207.02	12890594.86	2.9E-11	0.0E+00
10.00	80.00	1113207.02	15496270.75	2.9E-11	0.0E+00
10.00	85.00	1113207.02	19928981.89	2.9E-11	2.3E-10

Test of projection 6 Polar stereographic - Opt. A

Attributes

consts(1): 6378388.0000
consts(2): .67226700000E-02
consts(5): 0.0000000000
consts(6): 81.083000000
consts(7): 0.0000000000
consts(8): 0.0000000000
consts(9): 1.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	!del lam!	!del phi!
-10.00	76.00	-272746.14	-1546820.22	2.9E-11	0.0E+00
-10.00	78.00	-233484.55	-1324156.66	2.9E-11	2.3E-10
-10.00	80.00	-194361.07	-1102276.39	2.9E-11	0.0E+00
-10.00	82.00	-155352.15	-881045.83	2.9E-11	0.0E+00
-10.00	84.00	-116434.53	-660333.04	2.9E-11	2.3E-10
-10.00	86.00	-77585.18	-440007.44	2.9E-11	2.3E-10
-10.00	88.00	-38781.25	-219939.41	2.9E-11	0.0E+00
-10.00	90.00	-.00	-.00	2.9E-11	2.3E-10
0.00	76.00	0.00	-1570682.42	0.0E+00	2.3E-10
0.00	78.00	0.00	-1344583.91	0.0E+00	0.0E+00
0.00	80.00	0.00	-1119280.78	0.0E+00	0.0E+00
0.00	82.00	0.00	-894637.38	0.0E+00	0.0E+00
0.00	84.00	0.00	-670519.74	0.0E+00	2.3E-10
0.00	86.00	0.00	-446795.26	0.0E+00	2.3E-10
0.00	88.00	0.00	-223332.33	0.0E+00	0.0E+00
0.00	90.00	0.00	-.00	0.0E+00	2.3E-10
10.00	76.00	272746.14	-1546820.22	2.9E-11	0.0E+00
10.00	78.00	233484.55	-1324156.66	2.9E-11	2.3E-10
10.00	80.00	194361.07	-1102276.39	2.9E-11	0.0E+00
10.00	82.00	155352.15	-881045.83	2.9E-11	0.0E+00
10.00	84.00	116434.53	-660333.04	2.9E-11	2.3E-10
10.00	86.00	77585.18	-440007.44	2.9E-11	2.3E-10
10.00	88.00	38781.25	-219939.41	2.9E-11	0.0E+00
10.00	90.00	.00	-.00	2.9E-11	2.3E-10
20.00	76.00	537205.03	-1475958.68	1.2E-10	0.0E+00
20.00	78.00	459874.78	-1263495.58	1.2E-10	2.3E-10
20.00	80.00	382816.57	-1051779.89	5.8E-11	0.0E+00
20.00	82.00	305984.00	-840684.14	5.8E-11	0.0E+00
20.00	84.00	229331.26	-630082.46	5.8E-11	2.3E-10
20.00	86.00	152812.98	-419850.21	5.8E-11	2.3E-10
20.00	88.00	76384.15	-209863.74	5.8E-11	0.0E+00
20.00	90.00	.00	-.00	1.2E-10	2.3E-10

Test of projection 6 Polar stereographic - Opt. B

Attributes

consts(1): 6378206.4000
consts(2): .67686580000E-02
consts(3): .99400000000
consts(5): 0.0000000000
consts(6): 90.000000000
consts(7): 0.0000000000
consts(8): 0.0000000000
consts(9): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	!del lam!	!del phi!
-10.00	76.00	-272853.45	-1547428.82	2.9E-11	0.0E+00
-10.00	78.00	-233576.50	-1324678.13	2.9E-11	2.3E-10
-10.00	80.00	-194437.67	-1102710.83	2.9E-11	0.0E+00
-10.00	82.00	-155413.42	-881393.30	2.9E-11	0.0E+00
-10.00	84.00	-116480.47	-660593.60	5.8E-11	2.3E-10
-10.00	86.00	-77615.81	-440181.12	0.0E+00	2.3E-10
-10.00	88.00	-38796.56	-220026.24	2.9E-11	0.0E+00
-10.00	90.00	-.00	-.00	2.9E-11	2.3E-10
0.00	76.00	0.00	-1571300.41	0.0E+00	0.0E+00
0.00	78.00	0.00	-1345113.43	0.0E+00	0.0E+00
0.00	80.00	0.00	-1119721.92	0.0E+00	0.0E+00
0.00	82.00	0.00	-894990.21	0.0E+00	0.0E+00
0.00	84.00	0.00	-670784.32	0.0E+00	2.3E-10
0.00	86.00	0.00	-446971.62	0.0E+00	2.3E-10
0.00	88.00	0.00	-223420.50	0.0E+00	0.0E+00
0.00	90.00	0.00	-.00	0.0E+00	2.3E-10
10.00	76.00	272853.45	-1547428.82	2.9E-11	0.0E+00
10.00	78.00	233576.50	-1324678.13	2.9E-11	2.3E-10
10.00	80.00	194437.67	-1102710.83	2.9E-11	0.0E+00
10.00	82.00	155413.42	-881393.30	2.9E-11	0.0E+00
10.00	84.00	116480.47	-660593.60	5.8E-11	2.3E-10
10.00	86.00	77615.81	-440181.12	0.0E+00	2.3E-10
10.00	88.00	38796.56	-220026.24	2.9E-11	0.0E+00
10.00	90.00	.00	-.00	2.9E-11	2.3E-10
20.00	76.00	537416.39	-1476539.40	1.2E-10	0.0E+00
20.00	78.00	460055.89	-1263993.16	5.8E-11	2.3E-10
20.00	80.00	382967.45	-1052194.43	5.8E-11	0.0E+00
20.00	82.00	306104.68	-841015.70	5.8E-11	0.0E+00
20.00	84.00	229421.75	-630331.08	5.8E-11	2.3E-10
20.00	86.00	152873.30	-420015.94	5.8E-11	2.3E-10
20.00	88.00	76414.31	-209946.60	0.0E+00	0.0E+00
20.00	90.00	.00	-.00	5.8E-11	2.3E-10

Test of projection 7 Polyconic

Attributes

consts(1): 6378206.4000
consts(2): .6768658000E-02
consts(5): 0.0000000000
consts(6): 40.000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-15.00	25.00	-1511190.28	-1579737.27	5.2E-10	7.0E-10
-15.00	30.00	-1443193.70	-1014793.68	3.8E-10	4.1E-10
-15.00	35.00	-1364213.84	-452304.23	2.0E-10	3.5E-10
-15.00	40.00	-1274904.31	107525.21	0.0E+00	3.5E-10
-15.00	45.00	-1175993.70	664570.16	2.9E-11	1.2E-10
-15.00	50.00	-1068276.73	1218791.04	1.5E-10	0.0E+00
0.00	25.00	0.00	-1663422.61	0.0E+00	1.1E-09
0.00	30.00	0.00	-1109385.59	0.0E+00	8.7E-10
0.00	35.00	0.00	-554923.64	0.0E+00	7.0E-10
0.00	40.00	0.00	.00	0.0E+00	7.0E-10
0.00	45.00	0.00	555408.19	0.0E+00	4.7E-10
0.00	50.00	0.00	1111309.12	0.0E+00	4.7E-10
15.00	25.00	1511190.28	-1579737.27	5.2E-10	7.0E-10
15.00	30.00	1443193.70	-1014793.68	3.8E-10	4.1E-10
15.00	35.00	1364213.84	-452304.23	2.0E-10	3.5E-10
15.00	40.00	1274904.31	107525.21	0.0E+00	3.5E-10
15.00	45.00	1175993.70	664570.16	2.9E-11	1.2E-10
15.00	50.00	1068276.73	1218791.04	1.5E-10	0.0E+00
30.00	25.00	3003900.22	-1329704.63	8.1E-10	5.8E-10
30.00	30.00	2861693.96	-732636.43	2.9E-10	3.5E-10
30.00	35.00	2697724.30	-146755.58	0.0E+00	2.3E-10
30.00	40.00	2513790.19	427063.05	3.5E-10	2.3E-10
30.00	45.00	2311801.69	988325.82	5.2E-10	1.2E-10
30.00	50.00	2093730.87	1536928.31	0.0E+00	1.2E-10
45.00	25.00	4459875.49	-916382.34	1.6E-09	1.7E-10
45.00	30.00	4231229.84	-267741.64	4.7E-10	1.2E-10
45.00	35.00	3970519.04	354845.56	3.5E-10	1.2E-10
45.00	40.00	3681656.81	949585.98	0.0E+00	1.2E-10
45.00	45.00	3368611.48	1515611.89	7.1E-09	3.5E-10
45.00	50.00	3035256.42	2052968.19	4.7E-10	2.3E-10

Test of projection 8 Equidistant conic - Opt. A

Attributes

consts(1): 6378206.4000
consts(2): .67686580000E-02
consts(3): 40.000000000
consts(5): 0.0000000000
consts(6): 40.000000000
consts(7): 0.0000000000
consts(8): 0.0000000000
consts(9): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-15.00	25.00	-1553508.09	-1532400.00	2.9E-11	2.9E-10
-15.00	30.00	-1460713.39	-986189.27	0.0E+00	1.2E-10
-15.00	35.00	-1367847.51	-439559.62	0.0E+00	1.2E-10
-15.00	40.00	-1274904.31	107525.21	0.0E+00	2.3E-10
-15.00	45.00	-1181879.95	655087.74	0.0E+00	0.0E+00
-15.00	50.00	-1088773.06	1203136.05	2.9E-11	1.2E-10
0.00	25.00	0.00	-1663422.61	0.0E+00	7.0E-10
0.00	30.00	0.00	-1109385.59	0.0E+00	5.2E-10
0.00	35.00	0.00	-554923.64	0.0E+00	5.8E-10
0.00	40.00	0.00	.00	0.0E+00	8.1E-10
0.00	45.00	0.00	555408.19	0.0E+00	4.7E-10
0.00	50.00	0.00	1111309.12	0.0E+00	5.8E-10
15.00	25.00	1553508.09	-1532400.00	2.9E-11	2.9E-10
15.00	30.00	1460713.39	-986189.27	0.0E+00	1.2E-10
15.00	35.00	1367847.51	-439559.62	0.0E+00	1.2E-10
15.00	40.00	1274904.31	107525.21	0.0E+00	2.3E-10
15.00	45.00	1181879.95	655087.74	0.0E+00	0.0E+00
15.00	50.00	1088773.06	1203136.05	2.9E-11	1.2E-10
30.00	25.00	3063126.68	-1143033.82	1.2E-10	2.9E-10
30.00	30.00	2880158.90	-620080.85	2.3E-10	1.2E-10
30.00	35.00	2697050.78	-96726.80	1.2E-10	1.2E-10
30.00	40.00	2513790.19	427063.05	1.2E-10	2.3E-10
30.00	45.00	2330369.58	951310.26	1.2E-10	0.0E+00
30.00	50.00	2146786.24	1476022.57	2.3E-10	1.2E-10
45.00	25.00	4486206.23	-506324.38	0.0E+00	1.2E-10
45.00	30.00	4218234.54	-21403.57	0.0E+00	2.9E-10
45.00	35.00	3950057.33	463889.16	0.0E+00	1.2E-10
45.00	40.00	3681656.81	949585.98	0.0E+00	0.0E+00
45.00	45.00	3413021.92	1435706.91	0.0E+00	2.3E-10
45.00	50.00	3144148.71	1922259.11	2.3E-10	1.2E-10

Test of projection 8 Equidistant conic - Opt. B

Attributes

consts(1): 6378206.4000
consts(2): .67686580000E-02
consts(3): 33.000000000
consts(4): 45.000000000
consts(5): 0.0000000000
consts(6): 40.000000000
consts(7): 0.0000000000
consts(8): 0.0000000000
consts(9): 1.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-15.00	25.00	-1540683.68	-1536444.23	5.8E-11	5.8E-11
-15.00	30.00	-1449975.79	-989883.08	5.8E-11	1.2E-10
-15.00	35.00	-1359198.33	-442902.73	5.8E-11	3.5E-10
-15.00	40.00	-1268345.28	104533.09	5.8E-11	3.5E-10
-15.00	45.00	-1177412.90	652446.91	5.8E-11	2.3E-10
-15.00	50.00	-1086399.85	1200846.83	5.8E-11	1.2E-10
0.00	25.00	0.00	-1663422.61	0.0E+00	7.0E-10
0.00	30.00	0.00	-1109385.59	0.0E+00	7.6E-10
0.00	35.00	0.00	-554923.64	0.0E+00	5.8E-10
0.00	40.00	0.00	.00	0.0E+00	5.8E-10
0.00	45.00	0.00	555408.19	0.0E+00	4.7E-10
0.00	50.00	0.00	1111309.12	0.0E+00	5.8E-10
15.00	25.00	1540683.68	-1536444.23	5.8E-11	5.8E-11
15.00	30.00	1449975.79	-989883.08	5.8E-11	1.2E-10
15.00	35.00	1359198.33	-442902.73	5.8E-11	3.5E-10
15.00	40.00	1268345.28	104533.09	5.8E-11	3.5E-10
15.00	45.00	1177412.90	652446.91	5.8E-11	2.3E-10
15.00	50.00	1086399.85	1200846.83	5.8E-11	1.2E-10
30.00	25.00	3039789.12	-1158935.87	1.2E-10	5.8E-11
30.00	30.00	2860821.26	-634600.55	0.0E+00	1.2E-10
30.00	35.00	2681716.15	-109863.09	1.2E-10	1.2E-10
30.00	40.00	2502461.89	415311.32	1.2E-10	0.0E+00
30.00	45.00	2323051.11	940944.30	1.2E-10	0.0E+00
30.00	50.00	2143481.16	1467043.61	1.2E-10	1.2E-10
45.00	25.00	4456860.17	-541085.28	1.2E-10	1.2E-10
45.00	30.00	4194462.13	-53125.97	1.2E-10	5.8E-11
45.00	35.00	3931862.83	435207.59	1.2E-10	1.2E-10
45.00	40.00	3669044.88	923947.77	1.2E-10	0.0E+00
45.00	45.00	3405997.44	1413114.72	1.2E-10	0.0E+00
45.00	50.00	3142716.63	1902715.64	1.2E-10	1.2E-10

Test of projection 9 Transverse Mercator

Attributes

consts(1): 6378206.4000
consts(2): .67686580000E-02
consts(3): .99960000000
consts(5): 0.0000000000
consts(6): 0.0000000000
consts(7): 500000.00000
consts(8): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/vin	|del lam|	|del phi|
-1.00	25.00	399085.20	2765162.13	1.8E-12	5.8E-11
-1.00	30.00	403547.90	3319026.20	1.8E-12	1.2E-10
-1.00	35.00	408744.64	3873302.29	5.5E-12	0.0E+00
-1.00	40.00	414637.30	4428026.02	1.8E-12	0.0E+00
-1.00	45.00	421182.37	4983219.57	1.1E-11	1.2E-10
-1.00	50.00	428331.20	5538890.89	1.1E-11	1.2E-10
0.00	25.00	500000.00	2764789.92	0.0E+00	0.0E+00
0.00	30.00	500000.00	3318605.33	0.0E+00	5.8E-11
0.00	35.00	500000.00	3872845.49	0.0E+00	0.0E+00
0.00	40.00	500000.00	4427547.16	0.0E+00	1.2E-10
0.00	45.00	500000.00	4982733.19	0.0E+00	1.2E-10
0.00	50.00	500000.00	5538411.76	0.0E+00	0.0E+00
1.00	25.00	600914.80	2765162.13	3.6E-12	5.8E-11
1.00	30.00	596452.10	3319026.20	1.8E-12	1.2E-10
1.00	35.00	591255.36	3873302.29	3.6E-12	0.0E+00
1.00	40.00	585362.70	4428026.02	1.1E-11	0.0E+00
1.00	45.00	578817.63	4983219.57	0.0E+00	1.2E-10
1.00	50.00	571668.80	5538890.89	0.0E+00	1.2E-10
2.00	25.00	701849.49	2766279.19	1.2E-10	2.3E-10
2.00	30.00	692919.00	3320289.29	1.7E-10	1.2E-10
2.00	35.00	682520.32	3874673.10	2.1E-10	3.5E-10
2.00	40.00	670729.98	4429462.97	2.5E-10	2.3E-10
2.00	45.00	657635.29	4984679.02	2.5E-10	1.2E-10
2.00	50.00	643333.82	5540328.49	2.9E-10	2.3E-10
3.00	25.00	802824.00	2768142.45	2.1E-09	4.7E-10
3.00	30.00	789415.50	3322395.95	2.8E-09	5.2E-10
3.00	35.00	773804.43	3876959.21	3.5E-09	7.0E-10
3.00	40.00	756106.38	4431859.11	4.1E-09	3.5E-10
3.00	45.00	736453.01	4987112.42	4.6E-09	2.3E-10
3.00	50.00	714991.28	5542725.22	5.2E-09	5.8E-10
4.00	25.00	903858.24	2770754.19	1.6E-08	2.9E-10
4.00	30.00	885956.39	3325348.46	2.1E-08	4.1E-10
4.00	35.00	865117.26	3880162.74	2.6E-08	2.3E-10
4.00	40.00	841496.41	4435216.32	3.1E-08	2.3E-10
4.00	45.00	815270.74	4990521.28	3.7E-08	1.2E-09
4.00	50.00	786637.34	5546082.16	3.9E-08	1.7E-09

Test of projection 10 Stereographic

Attributes

consts(1): 6370977.0000
consts(5): 0.0000000000
consts(6): 45.000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-10.00	35.00	-917238.11	-1062997.11	5.8E-11	0.0E+00
-10.00	45.00	-785260.74	48579.24	0.0E+00	3.5E-10
-10.00	55.00	-641400.72	1157927.54	2.9E-11	4.7E-10
-10.00	65.00	-483213.77	2281914.75	8.7E-11	7.0E-10
-10.00	75.00	-307349.51	3438304.84	4.4E-10	1.2E-09
-10.00	85.00	-109252.39	4646910.74	1.7E-09	2.6E-09
0.00	35.00	0.00	-1114776.53	0.0E+00	0.0E+00
0.00	45.00	0.00	.00	0.0E+00	2.3E-10
0.00	55.00	0.00	1114776.53	0.0E+00	4.7E-10
0.00	65.00	0.00	2246750.28	0.0E+00	7.0E-10
0.00	75.00	0.00	3414196.28	0.0E+00	2.3E-10
0.00	85.00	0.00	4637691.98	0.0E+00	3.7E-09
10.00	35.00	917238.11	-1062997.11	5.8E-11	0.0E+00
10.00	45.00	785260.74	48579.24	0.0E+00	3.5E-10
10.00	55.00	641400.72	1157927.54	2.9E-11	4.7E-10
10.00	65.00	483213.77	2281914.75	8.7E-11	7.0E-10
10.00	75.00	307349.51	3438304.84	4.4E-10	1.2E-09
10.00	85.00	109252.39	4646910.74	1.7E-09	2.6E-09
20.00	35.00	1830818.26	-906477.48	5.8E-11	1.2E-10
20.00	45.00	1564373.23	195049.19	5.8E-11	2.3E-10
20.00	55.00	1275104.50	1287668.28	5.8E-11	4.7E-10
20.00	65.00	958422.57	2387319.05	3.5E-10	7.0E-10
20.00	75.00	608054.84	3510323.95	4.1E-10	1.2E-09
20.00	85.00	215524.69	4674344.66	2.8E-09	2.6E-09
30.00	35.00	2736358.34	-641681.16	0.0E+00	0.0E+00
30.00	45.00	2330538.77	441564.13	5.8E-11	2.3E-10
30.00	55.00	1892928.45	1504805.07	2.9E-10	8.1E-10
30.00	65.00	1417357.14	2562646.77	4.1E-10	1.2E-09
30.00	75.00	895424.65	3629310.12	4.1E-10	7.0E-10
30.00	85.00	315889.80	4719327.27	4.1E-09	2.6E-09
40.00	35.00	3627956.35	-262753.77	1.2E-10	0.0E+00
40.00	45.00	3075622.91	791560.21	4.7E-10	2.3E-10
40.00	55.00	2485715.30	1810484.52	1.2E-10	8.1E-10
40.00	65.00	1851235.59	2807205.11	9.3E-10	7.0E-10
40.00	75.00	1162688.58	3793612.71	1.3E-09	1.2E-09
40.00	85.00	407528.63	4780750.55	4.3E-09	2.6E-09

Test of projection 11 Lambert azimuthal equal-area

Attributes

consts(1): 6370977.0000
consts(5): 0.0000000000
consts(6): 45.000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-10.00	35.00	-911719.91	-1056602.01	0.0E+00	1.2E-10
-10.00	45.00	-783768.09	48486.89	2.9E-11	3.5E-10
-10.00	55.00	-637967.43	1151729.38	1.2E-10	8.1E-10
-10.00	65.00	-475315.43	2244615.87	8.7E-11	1.2E-09
-10.00	75.00	-296655.57	3318672.27	3.5E-10	7.0E-10
-10.00	85.00	-102636.45	4365510.08	1.1E-09	3.7E-09
0.00	35.00	0.00	-1110534.46	0.0E+00	0.0E+00
0.00	45.00	0.00	.00	0.0E+00	2.3E-10
0.00	55.00	0.00	1110534.46	0.0E+00	4.7E-10
0.00	65.00	0.00	2212617.09	0.0E+00	7.0E-10
0.00	75.00	0.00	3297860.37	0.0E+00	2.3E-10
0.00	85.00	0.00	4358004.93	0.0E+00	2.6E-09
10.00	35.00	911719.91	-1056602.01	0.0E+00	1.2E-10
10.00	45.00	783768.09	48486.89	2.9E-11	3.5E-10
10.00	55.00	637967.43	1151729.38	1.2E-10	8.1E-10
10.00	65.00	475315.43	2244615.87	8.7E-11	1.2E-09
10.00	75.00	296655.57	3318672.27	3.5E-10	7.0E-10
10.00	85.00	102636.45	4365510.08	1.1E-09	3.7E-09
20.00	35.00	1807730.72	-895046.34	0.0E+00	1.2E-10
20.00	45.00	1552535.54	193573.24	5.8E-11	2.3E-10
20.00	55.00	1262401.21	1274839.82	5.8E-11	4.7E-10
20.00	65.00	939466.95	2340102.81	2.3E-10	4.7E-10
20.00	75.00	585596.34	3380670.13	4.1E-10	1.2E-09
20.00	85.00	202313.74	4387822.79	2.8E-09	2.6E-09
30.00	35.00	2672124.97	-626618.31	5.8E-11	0.0E+00
30.00	45.00	2291177.25	434106.35	5.8E-11	2.3E-10
30.00	55.00	1859733.74	1478416.56	5.8E-11	8.1E-10
30.00	65.00	1381343.85	2497533.09	4.1E-10	1.2E-09
30.00	75.00	859212.64	3482536.63	1.0E-09	2.3E-10
30.00	85.00	296144.60	4424338.24	2.4E-09	1.2E-09
40.00	35.00	3488590.98	-252660.27	0.0E+00	0.0E+00
40.00	45.00	2984322.90	768062.71	0.0E+00	3.5E-10
40.00	55.00	2416340.37	1759954.91	1.2E-10	8.1E-10
40.00	65.00	1789953.47	2714277.19	9.3E-10	7.0E-10
40.00	75.00	1110111.46	3622064.42	1.9E-09	1.6E-09
40.00	85.00	381385.20	4474059.90	6.6E-09	2.6E-09

Test of projection 12 Azimuthal equidistant

Attributes

consts(1): 6370977.0000
consts(5): 0.0000000000
consts(6): 45.000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-10.00	35.00	-913552.65	-1058726.00	2.9E-11	0.0E+00
-10.00	45.00	-784265.07	48517.64	2.9E-11	3.5E-10
-10.00	55.00	-639108.18	1153788.78	1.2E-10	4.7E-10
-10.00	65.00	-477922.23	2256926.14	8.7E-11	1.2E-09
-10.00	75.00	-300144.80	3357706.09	4.1E-10	1.2E-09
-10.00	85.00	-104759.74	4455821.51	1.7E-09	2.6E-09
0.00	35.00	0.00	-1111945.25	0.0E+00	1.2E-10
0.00	45.00	0.00	.00	0.0E+00	2.3E-10
0.00	55.00	0.00	1111945.25	0.0E+00	4.7E-10
0.00	65.00	0.00	2223890.50	0.0E+00	7.0E-10
0.00	75.00	0.00	3335835.76	0.0E+00	7.0E-10
0.00	85.00	0.00	4447781.01	0.0E+00	2.6E-09
10.00	35.00	913552.65	-1058726.00	2.9E-11	0.0E+00
10.00	45.00	784265.07	48517.64	2.9E-11	3.5E-10
10.00	55.00	639108.18	1153788.78	1.2E-10	4.7E-10
10.00	65.00	477922.23	2256926.14	8.7E-11	1.2E-09
10.00	75.00	300144.80	3357706.09	4.1E-10	1.2E-09
10.00	85.00	104759.74	4455821.51	1.7E-09	2.6E-09
20.00	35.00	1815368.06	-898827.75	1.2E-10	1.2E-10
20.00	45.00	1556463.47	194062.98	5.8E-11	2.3E-10
20.00	55.00	1266610.23	1279090.32	1.2E-10	8.1E-10
20.00	65.00	945709.94	2355653.37	1.7E-10	4.7E-10
20.00	75.00	592914.24	3422916.63	4.1E-10	1.2E-09
20.00	85.00	206551.55	4479733.33	1.7E-09	1.2E-09
30.00	35.00	2693231.81	-631567.91	0.0E+00	0.0E+00
30.00	45.00	2304163.93	436566.92	1.7E-10	2.3E-10
30.00	55.00	1870681.44	1487119.55	2.9E-10	8.1E-10
30.00	65.00	1393163.48	2518903.52	4.1E-10	1.2E-09
30.00	75.00	870985.83	3530255.39	5.2E-10	7.0E-10
30.00	85.00	302473.46	4518890.00	4.1E-09	2.6E-09
40.00	35.00	3533959.45	-255946.07	0.0E+00	0.0E+00
40.00	45.00	3014207.98	775754.11	0.0E+00	3.5E-10
40.00	55.00	2439073.94	1776513.03	2.3E-10	8.1E-10
40.00	65.00	1809969.93	2744630.05	7.0E-10	7.0E-10
40.00	75.00	1127153.31	3677668.45	9.3E-10	7.0E-10
40.00	85.00	389755.82	4572256.40	5.5E-09	3.7E-09

Test of projection 13 Gnomonic

Attributes

consts(1): 6370977.0000
consts(5): 0.0000000000
consts(6): 45.000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-10.00	35.00	-928511.78	-1076062.30	8.7E-11	0.0E+00
-10.00	45.00	-788266.04	48765.15	5.8E-11	3.5E-10
-10.00	55.00	-648398.36	1170560.44	2.9E-11	4.7E-10
-10.00	65.00	-499967.79	2361033.48	1.2E-10	7.0E-10
-10.00	75.00	-331694.61	3710652.35	1.5E-10	1.2E-09
-10.00	85.00	-126022.89	5360222.55	1.1E-09	3.7E-09
0.00	35.00	0.00	-1123375.14	0.0E+00	0.0E+00
0.00	45.00	0.00	.00	0.0E+00	2.3E-10
0.00	55.00	0.00	1123375.14	0.0E+00	4.7E-10
0.00	65.00	0.00	2318845.99	0.0E+00	7.0E-10
0.00	75.00	0.00	3678285.29	0.0E+00	2.3E-10
0.00	85.00	0.00	5345884.45	0.0E+00	2.6E-09
10.00	35.00	928511.78	-1076062.30	8.7E-11	0.0E+00
10.00	45.00	788266.04	48765.15	5.8E-11	3.5E-10
10.00	55.00	648398.36	1170560.44	2.9E-11	4.7E-10
10.00	65.00	499967.79	2361033.48	1.2E-10	7.0E-10
10.00	75.00	331694.61	3710652.35	1.5E-10	1.2E-09
10.00	85.00	126022.89	5360222.55	1.1E-09	3.7E-09
20.00	35.00	1879123.45	-930394.42	5.8E-11	1.2E-10
20.00	45.00	1588692.36	198081.35	1.2E-10	2.3E-10
20.00	55.00	1301428.32	1314251.47	5.8E-11	4.7E-10
20.00	65.00	999148.98	2488763.81	5.8E-11	7.0E-10
20.00	75.00	659619.88	3808010.88	3.5E-10	1.2E-09
20.00	85.00	249121.94	5403008.77	1.6E-09	1.2E-09
30.00	35.00	2876303.34	-674498.52	0.0E+00	1.2E-10
30.00	45.00	2414201.34	457415.57	0.0E+00	2.3E-10
30.00	55.00	1963653.24	1561028.54	1.2E-10	4.7E-10
30.00	65.00	1496400.16	2705560.17	2.9E-10	7.0E-10
30.00	75.00	979749.10	3971091.62	4.1E-10	7.0E-10
30.00	85.00	366373.83	5473548.07	2.2E-09	1.2E-09
40.00	35.00	3949843.79	-286066.39	1.2E-10	0.0E+00
40.00	45.00	3279343.45	843990.92	0.0E+00	3.5E-10
40.00	55.00	2639452.29	1922459.71	0.0E+00	4.7E-10
40.00	65.00	1989817.58	3017350.20	1.2E-10	4.7E-10
40.00	75.00	1287537.49	4200968.92	9.3E-10	7.0E-10
40.00	85.00	474862.28	5570646.92	4.4E-09	2.6E-09

Test of projection 14 Orthographic

Attributes

consts(1): 6370977.0000
consts(5): 0.0000000000
consts(6): 45.000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-10.00	35.00	-906234.91	-1050245.39	2.9E-11	1.2E-10
-10.00	45.00	-782278.27	48394.73	5.8E-11	4.7E-10
-10.00	55.00	-634552.51	1145564.39	2.9E-11	4.7E-10
-10.00	65.00	-467546.19	2207926.66	2.3E-10	7.0E-10
-10.00	75.00	-286333.72	3203202.20	4.1E-10	1.2E-09
-10.00	85.00	-96421.14	4101150.06	1.7E-09	2.6E-09
0.00	35.00	0.00	-1106308.55	0.0E+00	1.2E-10
0.00	45.00	0.00	.00	0.0E+00	2.3E-10
0.00	55.00	0.00	1106308.55	0.0E+00	4.7E-10
0.00	65.00	0.00	2179002.47	0.0E+00	7.0E-10
0.00	75.00	0.00	3185488.50	0.0E+00	1.2E-09
0.00	85.00	0.00	4095185.08	0.0E+00	2.6E-09
10.00	35.00	906234.91	-1050245.39	2.9E-11	1.2E-10
10.00	45.00	782278.27	48394.73	5.8E-11	4.7E-10
10.00	55.00	634552.51	1145564.39	2.9E-11	4.7E-10
10.00	65.00	467546.19	2207926.66	2.3E-10	7.0E-10
10.00	75.00	286333.72	3203202.20	4.1E-10	1.2E-09
10.00	85.00	96421.14	4101150.06	1.7E-09	2.6E-09
20.00	35.00	1784934.33	-883759.36	1.2E-10	1.2E-10
20.00	45.00	1540787.42	192108.46	5.8E-11	2.3E-10
20.00	55.00	1249824.47	1262139.17	1.7E-10	8.1E-10
20.00	65.00	920886.23	2293820.41	3.5E-10	7.0E-10
20.00	75.00	563967.34	3255805.08	4.7E-10	1.2E-09
20.00	85.00	189912.58	4118863.76	3.5E-09	1.2E-09
30.00	35.00	2609399.42	-611909.05	0.0E+00	0.0E+00
30.00	45.00	2252480.52	426774.54	1.7E-10	2.3E-10
30.00	55.00	1827121.14	1452490.80	2.9E-10	8.1E-10
30.00	65.00	1346245.61	2434073.86	4.1E-10	1.2E-09
30.00	75.00	824465.09	3341698.82	1.5E-09	7.0E-10
30.00	85.00	277633.62	4147787.96	4.9E-09	3.7E-09
40.00	35.00	3354579.23	-242954.51	0.0E+00	0.0E+00
40.00	45.00	2895733.14	745262.74	4.7E-10	2.3E-10
40.00	55.00	2348901.66	1710835.55	2.3E-10	8.1E-10
40.00	65.00	1730700.00	2624425.49	7.0E-10	7.0E-10
40.00	75.00	1059911.89	3458273.59	1.7E-09	1.6E-09
40.00	85.00	356918.90	4187043.80	6.5E-09	2.6E-09

Test of projection 15 General vertical near-side persp.

Attributes

consts(1): 6370977.0000
consts(3): 1000000000.0
consts(5): 0.0000000000
consts(6): 45.000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-10.00	35.00	-906096.41	-1050084.88	2.9E-11	0.0E+00
-10.00	45.00	-782240.42	48392.39	0.0E+00	3.5E-10
-10.00	55.00	-634466.20	1145408.57	0.0E+00	4.7E-10
-10.00	65.00	-467353.11	2207014.86	8.7E-11	7.0E-10
-10.00	75.00	-286084.47	3200413.80	4.1E-10	1.2E-09
-10.00	85.00	-96277.07	4095021.90	1.7E-09	2.6E-09
0.00	35.00	0.00	-1106201.48	0.0E+00	1.2E-10
0.00	45.00	0.00	.00	0.0E+00	2.3E-10
0.00	55.00	0.00	1106201.48	0.0E+00	4.7E-10
0.00	65.00	0.00	2178165.58	0.0E+00	7.0E-10
0.00	75.00	0.00	3182771.85	0.0E+00	7.0E-10
0.00	85.00	0.00	4089090.18	0.0E+00	2.6E-09
10.00	35.00	906096.41	-1050084.88	2.9E-11	0.0E+00
10.00	45.00	782240.42	48392.39	0.0E+00	3.5E-10
10.00	55.00	634466.20	1145408.57	0.0E+00	4.7E-10
10.00	65.00	467353.11	2207014.86	8.7E-11	7.0E-10
10.00	75.00	286084.47	3200413.80	4.1E-10	1.2E-09
10.00	85.00	96277.07	4095021.90	1.7E-09	2.6E-09
20.00	35.00	1784364.51	-883477.23	5.8E-11	1.2E-10
20.00	45.00	1540491.48	192071.56	1.2E-10	2.3E-10
20.00	55.00	1249508.82	1261820.40	1.2E-10	8.1E-10
20.00	65.00	920426.91	2292676.28	1.7E-10	4.7E-10
20.00	75.00	563446.79	3252799.93	4.1E-10	1.2E-09
20.00	85.00	189625.45	4112636.39	2.8E-09	2.6E-09
30.00	35.00	2607857.68	-611547.51	0.0E+00	0.0E+00
30.00	45.00	2251519.63	426592.48	5.8E-11	2.3E-10
30.00	55.00	1826312.14	1451847.67	2.9E-10	8.1E-10
30.00	65.00	1345385.52	2432518.78	2.9E-10	7.0E-10
30.00	75.00	823633.42	3338327.91	8.1E-10	1.2E-09
30.00	85.00	277205.85	4141397.25	4.1E-09	2.6E-09
40.00	35.00	3351361.44	-242721.46	0.0E+00	0.0E+00
40.00	45.00	2893576.66	744707.73	0.0E+00	3.5E-10
40.00	55.00	2347255.49	1709636.55	3.5E-10	8.1E-10
40.00	65.00	1729265.33	2622249.97	8.1E-10	7.0E-10
40.00	75.00	1058719.42	3454382.80	1.3E-09	1.2E-09
40.00	85.00	356355.01	4180428.76	6.5E-09	2.6E-09

Test of projection 16 Sinusoidal

Attributes

consts(1): 6370977.0000
consts(5): 0.0000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

forward			inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-10.00	0.00	-1111945.25	0.00	0.0E+00	0.0E+00
-10.00	20.00	-1044886.75	2223890.50	2.9E-11	5.8E-11
-10.00	40.00	-851799.48	4447781.01	0.0E+00	1.2E-10
-10.00	60.00	-555972.63	6671671.51	0.0E+00	1.2E-10
-10.00	80.00	-193087.27	8895562.02	4.1E-10	2.3E-10
0.00	0.00	0.00	0.00	0.0E+00	0.0E+00
0.00	20.00	0.00	2223890.50	0.0E+00	5.8E-11
0.00	40.00	0.00	4447781.01	0.0E+00	1.2E-10
0.00	60.00	0.00	6671671.51	0.0E+00	1.2E-10
0.00	80.00	0.00	8895562.02	0.0E+00	2.3E-10
10.00	0.00	1111945.25	0.00	0.0E+00	0.0E+00
10.00	20.00	1044886.75	2223890.50	2.9E-11	5.8E-11
10.00	40.00	851799.48	4447781.01	0.0E+00	1.2E-10
10.00	60.00	555972.63	6671671.51	0.0E+00	1.2E-10
10.00	80.00	193087.27	8895562.02	4.1E-10	2.3E-10
20.00	0.00	2223890.50	0.00	0.0E+00	0.0E+00
20.00	20.00	2089773.50	2223890.50	5.8E-11	5.8E-11
20.00	40.00	1703598.96	4447781.01	0.0E+00	1.2E-10
20.00	60.00	1111945.25	6671671.51	0.0E+00	1.2E-10
20.00	80.00	386174.53	8895562.02	8.1E-10	2.3E-10
30.00	0.00	3335835.76	0.00	5.8E-11	0.0E+00
30.00	20.00	3134660.24	2223890.50	5.8E-11	5.8E-11
30.00	40.00	2555398.44	4447781.01	5.8E-11	1.2E-10
30.00	60.00	1667917.88	6671671.51	5.8E-11	1.2E-10
30.00	80.00	579261.80	8895562.02	1.0E-09	2.3E-10
40.00	0.00	4447781.01	0.00	0.0E+00	0.0E+00
40.00	20.00	4179546.99	2223890.50	1.2E-10	5.8E-11
40.00	40.00	3407197.93	4447781.01	0.0E+00	1.2E-10
40.00	60.00	2223890.50	6671671.51	0.0E+00	1.2E-10
40.00	80.00	772349.07	8895562.02	1.6E-09	2.3E-10

Test of projection 17 Equirectagular (plate caree)

Attributes

consts(1): 6370977.0000
consts(5): 0.0000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

forward			inverse		
lamin	phiin	xout/xin	yout/yin	!del lam!	!del phi!
-10.00	-10.00	-1111945.25	-1111945.25	2.9E-11	2.9E-11
-10.00	0.00	-1111945.25	0.00	2.9E-11	0.0E+00
-10.00	10.00	-1111945.25	1111945.25	2.9E-11	2.9E-11
-10.00	20.00	-1111945.25	2223890.50	2.9E-11	5.8E-11
-10.00	30.00	-1111945.25	3335835.76	2.9E-11	5.8E-11
0.00	-10.00	0.00	-1111945.25	0.0E+00	2.9E-11
0.00	0.00	0.00	0.00	0.0E+00	0.0E+00
0.00	10.00	0.00	1111945.25	0.0E+00	2.9E-11
0.00	20.00	0.00	2223890.50	0.0E+00	5.8E-11
0.00	30.00	0.00	3335835.76	0.0E+00	5.8E-11
10.00	-10.00	1111945.25	-1111945.25	2.9E-11	2.9E-11
10.00	0.00	1111945.25	0.00	2.9E-11	0.0E+00
10.00	10.00	1111945.25	1111945.25	2.9E-11	2.9E-11
10.00	20.00	1111945.25	2223890.50	2.9E-11	5.8E-11
10.00	30.00	1111945.25	3335835.76	2.9E-11	5.8E-11
20.00	-10.00	2223890.50	-1111945.25	5.8E-11	2.9E-11
20.00	0.00	2223890.50	0.00	5.8E-11	0.0E+00
20.00	10.00	2223890.50	1111945.25	5.8E-11	2.9E-11
20.00	20.00	2223890.50	2223890.50	5.8E-11	5.8E-11
20.00	30.00	2223890.50	3335835.76	5.8E-11	5.8E-11
30.00	-10.00	3335835.76	-1111945.25	5.8E-11	2.9E-11
30.00	0.00	3335835.76	0.00	5.8E-11	0.0E+00
30.00	10.00	3335835.76	1111945.25	5.8E-11	2.9E-11
30.00	20.00	3335835.76	2223890.50	5.8E-11	5.8E-11
30.00	30.00	3335835.76	3335835.76	5.8E-11	5.8E-11
40.00	-10.00	4447781.01	-1111945.25	1.2E-10	2.9E-11
40.00	0.00	4447781.01	0.00	1.2E-10	0.0E+00
40.00	10.00	4447781.01	1111945.25	1.2E-10	2.9E-11
40.00	20.00	4447781.01	2223890.50	1.2E-10	5.8E-11
40.00	30.00	4447781.01	3335835.76	1.2E-10	5.8E-11

Test of projection 18 Miller cylindical

Attributes

consts(1): 6370977.0000
consts(5): 0.0000000000
consts(7): 0.0000000000
consts(8): 0.0000000000

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	!del lam!	!del phi!
0.00	0.00	0.00	0.00	0.0E+00	0.0E+00
0.00	5.00	0.00	556424.80	0.0E+00	8.7E-11
0.00	10.00	0.00	1115575.95	0.0E+00	3.5E-10
0.00	15.00	0.00	1680247.17	0.0E+00	0.0E+00
0.00	20.00	0.00	2253371.01	0.0E+00	1.7E-10
0.00	25.00	0.00	2838099.07	0.0E+00	3.5E-10
0.00	30.00	0.00	3437896.47	0.0E+00	0.0E+00
0.00	35.00	0.00	4056658.18	0.0E+00	3.5E-10
0.00	40.00	0.00	4698857.49	0.0E+00	0.0E+00
0.00	45.00	0.00	5369741.95	0.0E+00	2.3E-10
0.00	50.00	0.00	6075599.81	0.0E+00	1.2E-10
0.00	55.00	0.00	6824133.45	0.0E+00	0.0E+00
0.00	60.00	0.00	7624999.22	0.0E+00	2.3E-10
0.00	65.00	0.00	8490614.67	0.0E+00	7.0E-10
0.00	70.00	0.00	9437413.37	0.0E+00	4.7E-10
0.00	75.00	0.00	10487885.59	0.0E+00	4.7E-10
0.00	80.00	0.00	11674085.34	0.0E+00	2.3E-10
0.00	85.00	0.00	13044087.09	0.0E+00	4.7E-10
10.00	0.00	1111945.25	0.00	2.9E-11	0.0E+00
10.00	5.00	1111945.25	556424.80	2.9E-11	8.7E-11
10.00	10.00	1111945.25	1115575.95	2.9E-11	3.5E-10
10.00	15.00	1111945.25	1680247.17	2.9E-11	0.0E+00
10.00	20.00	1111945.25	2253371.01	2.9E-11	1.7E-10
10.00	25.00	1111945.25	2838099.07	2.9E-11	3.5E-10
10.00	30.00	1111945.25	3437896.47	2.9E-11	0.0E+00
10.00	35.00	1111945.25	4056658.18	2.9E-11	3.5E-10
10.00	40.00	1111945.25	4698857.49	2.9E-11	0.0E+00
10.00	45.00	1111945.25	5369741.95	2.9E-11	2.3E-10
10.00	50.00	1111945.25	6075599.81	2.9E-11	1.2E-10
10.00	55.00	1111945.25	6824133.45	2.9E-11	0.0E+00
10.00	60.00	1111945.25	7624999.22	2.9E-11	2.3E-10
10.00	65.00	1111945.25	8490614.67	2.9E-11	7.0E-10
10.00	70.00	1111945.25	9437413.37	2.9E-11	4.7E-10
10.00	75.00	1111945.25	10487885.59	2.9E-11	4.7E-10
10.00	80.00	1111945.25	11674085.34	2.9E-11	2.3E-10
10.00	85.00	1111945.25	13044087.09	2.9E-11	4.7E-10

Test of projection 20 Oblique Mercator - Opt. A

Attributes

```
consts( 1):     6378206.4000
consts( 2):     .67686580000E-02
consts( 3):     .99960000000
consts( 4):     0.0000000000
consts( 7):     0.0000000000
consts( 8):     0.0000000000
consts( 9):     0.0000000000
consts(10):     45.000000000
consts(11):     5.0000000000
consts(12):     50.000000000
consts(13):     0.0000000000
```

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-2.00	44.00	1947421.21	4968755.86	2.5E-10	5.8E-10
-2.00	46.00	1913976.63	5188356.57	4.5E-10	5.8E-10
-2.00	48.00	1880703.29	5408166.77	1.4E-10	8.1E-10
-2.00	50.00	1847710.16	5628272.07	3.0E-10	7.0E-10
0.00	44.00	2105574.01	4994850.71	1.0E-10	1.2E-10
0.00	46.00	2066746.87	5213513.85	1.0E-10	2.3E-10
0.00	48.00	2027962.61	5432286.33	3.1E-10	5.8E-10
0.00	50.00	1989329.24	5651263.15	3.6E-10	8.1E-10
2.00	44.00	2263073.28	5024806.34	2.8E-10	3.5E-10
2.00	46.00	2218809.61	5242483.36	1.2E-10	3.5E-10
2.00	48.00	2174473.06	5460159.46	1.2E-10	5.8E-10
2.00	50.00	2130170.38	5677939.03	2.3E-10	7.0E-10
4.00	44.00	2419883.74	5058668.52	5.5E-10	3.5E-10
4.00	46.00	2370123.93	5275300.22	1.4E-10	3.5E-10
4.00	48.00	2320188.70	5491811.37	1.4E-10	5.8E-10
4.00	50.00	2270183.28	5708315.91	5.5E-10	8.1E-10
6.00	44.00	2575961.88	5096487.01	5.2E-10	3.5E-10
6.00	46.00	2520641.53	5312002.91	5.2E-10	3.5E-10
6.00	48.00	2465057.12	5527270.15	3.1E-10	5.8E-10
6.00	50.00	2409311.94	5742412.37	2.0E-10	8.1E-10
8.00	44.00	2731255.56	5138314.98	3.8E-10	3.5E-10
8.00	46.00	2670306.37	5352632.76	4.8E-10	2.3E-10
8.00	48.00	2609019.00	5566566.17	3.8E-10	8.1E-10
8.00	50.00	2547494.41	5780248.80	2.8E-10	7.0E-10

Test of projection 20 Oblique Mercator - Opt. B

Attributes

```
consts( 1):     6378206.4000
consts( 2):     .67686580000E-02
consts( 3):     .99960000000
consts( 4):     45.000000000
consts( 5):     0.0000000000
consts( 6):     45.000000000
consts( 7):     0.0000000000
consts( 8):     0.0000000000
consts(13):     1.0000000000
```

	forward		inverse		
lamin	phiin	xout/xin	yout/yin	|del lam|	|del phi|
-2.00	44.00	4142841.12	4194069.73	2.5E-10	1.2E-10
-2.00	46.00	4148297.34	4416215.75	4.5E-10	2.3E-10
-2.00	48.00	4153926.33	4638595.24	3.5E-10	7.0E-10
-2.00	50.00	4159867.55	4861332.74	3.5E-10	7.0E-10
0.00	44.00	4303181.05	4192104.49	4.2E-10	5.8E-10
0.00	46.00	4303186.67	4414282.98	7.3E-10	5.8E-10
0.00	48.00	4303259.85	4636607.89	4.2E-10	8.1E-10
0.00	50.00	4303536.28	4859213.84	1.0E-10	4.7E-10
2.00	44.00	4463555.78	4194063.15	6.9E-11	1.2E-10
2.00	46.00	4458042.95	4416206.64	6.9E-11	2.3E-10
2.00	48.00	4452504.04	4638402.41	4.8E-10	7.0E-10
2.00	50.00	4447070.56	4860795.13	1.7E-10	5.8E-10
4.00	44.00	4623915.35	4200000.19	3.4E-10	3.5E-10
4.00	46.00	4612813.27	4422030.94	3.4E-10	1.2E-10
4.00	48.00	4601603.50	4644013.60	2.4E-10	5.8E-10
4.00	50.00	4590412.84	4866102.88	5.5E-10	8.1E-10
6.00	44.00	4784201.94	4209972.02	6.3E-10	1.2E-10
6.00	46.00	4767437.76	4431801.67	2.0E-10	2.3E-10
6.00	48.00	4750496.67	4653477.52	3.1E-10	5.8E-10
6.00	50.00	4733500.27	4875164.35	4.1E-10	5.8E-10
8.00	44.00	4944349.58	4224036.38	1.7E-10	3.5E-10
8.00	46.00	4921849.32	4445565.60	3.8E-10	1.2E-10
8.00	48.00	4899115.71	4666830.96	2.8E-10	7.0E-10
8.00	50.00	4876264.55	4888007.31	5.8E-11	7.0E-10

:

Appendix F

The following is a listing of the program for testing the projection procedures. It is presented as an example of usage of the projection procedures and contains nonstandard RATFOR code.

```
# main program for testing projections
   program ptest

         call initr4
         call bench
         call endr4

      end


# block data section
   block data chpblk
   include chpdcb
   data maxsys / 3 /
   end


# bench - benchmark for projections
   subroutine bench
include prjsym
define(PRTLU, 6)
define(DATAIN, 5)
   PREC lat, lon, lati, loni, x, y, consts(MAXCONSTS)
   PREC minlat, maxlat, inclat, minlon, maxlon, inclon, ABS
   integer setxx, forxx, invxx, i, prjno, title(30), c

   repeat {
      read(DATAIN, 100) prjno, title
      if (prjno == 99)
         break
      100 format(i2,30a2)
      write(PRTLU, 1) prjno, title
      1 format("1 Test of projection", i3,2x,30a2,
         /"   "/" Attributes"/"   ")
      read(DATAIN, *) minlon, maxlon, inclon
      read(DATAIN, *) minlat, maxlat, inclat
      repeat {
         read(DATAIN, *) i, consts(i), c
         if (i ^= 30)
            write(PRTLU,3) i,consts(i)
            3 format(" consts(",i2,"):",g20.11)
         if (c ^= 0)
            consts(i) = consts(i) * DEGTORAD
         }
      until (i == 30)
```

```
      write(PRTLU, 10)
      10 format("    "/t15,"forward",t42,"inverse"/,
         t4,"lamin",t12,"phiin",t23,"xout/xin",t36,"yout/yin",
         t47,"!del lam!",t58,"!del phi!"/"    ")

      if (setxx(prjno, consts) == ERR)
         call error("initialization failure.")

      for (lon = minlon; lon <= maxlon; lon = lon + inclon)
         for (lat = minlat; lat <= maxlat; lat = lat + inclat) {
            if (forxx(prjno, lon * DEGTORAD, lat * DEGTORAD, x,
                y, consts) == ERR)
               call error("forward error.")
            if (invxx(prjno, x, y, loni, lati, consts) == ERR)
               call error("inversion error.")
            lati = ABS(lati * RADTODEG - lat)
            loni = ABS(loni * RADTODEG - lon)
            write(PRTLU, 2) lon, lat, x, y, loni, lati
            2 format(2f8.2,f14.2,f13.2,1p,t49,e7.1,t60,e7.1)
            }
      }

   return
   end


# - setxx - switch routine to initialization
   integer function setxx(prjno, consts)
   PREC consts(ARB)
   integer prjno, set01, set03, set04, set05, set06, set07,
       set08, set09, set10, set11, set12, set13, set14, set15,
       set16, set17, set18, set19, set20

      switchto prjno {
         setxx = set01(consts)
         setxx = ERR
         setxx = set03(consts)
         setxx = set04(consts)
         setxx = set05(consts)
         setxx = set06(consts)
         setxx = set07(consts)
         setxx = set08(consts)
         setxx = set09(consts)
         setxx = set10(consts)
         setxx = set11(consts)
         setxx = set12(consts)
         setxx = set13(consts)
         setxx = set14(consts)
         setxx = set15(consts)
         setxx = set16(consts)
         setxx = set17(consts)
         setxx = set18(consts)
         setxx = set19(consts)
         setxx = set20(consts)
         }
```

```
      else
         setxx = ERR

   return
   end


# - forxx - switch routine to forward
   integer function forxx(prjno, lamin, phiin,
                  xout, yout, consts)
   PREC consts(ARB), lamin, phiin, xout, yout
   integer prjno, for01, for03, for04, for05, for06, for07,
       for08, for09, for10, for11, for12, for13, for14, for15,
       for16, for17, for18, for19, for20

      switchto prjno {
         forxx = for01(lamin, phiin, xout, yout, consts)
         forxx = ERR
         forxx = for03(lamin, phiin, xout, yout, consts)
         forxx = for04(lamin, phiin, xout, yout, consts)
         forxx = for05(lamin, phiin, xout, yout, consts)
         forxx = for06(lamin, phiin, xout, yout, consts)
         forxx = for07(lamin, phiin, xout, yout, consts)
         forxx = for08(lamin, phiin, xout, yout, consts)
         forxx = for09(lamin, phiin, xout, yout, consts)
         forxx = for10(lamin, phiin, xout, yout, consts)
         forxx = for11(lamin, phiin, xout, yout, consts)
         forxx = for12(lamin, phiin, xout, yout, consts)
         forxx = for13(lamin, phiin, xout, yout, consts)
         forxx = for14(lamin, phiin, xout, yout, consts)
         forxx = for15(lamin, phiin, xout, yout, consts)
         forxx = for16(lamin, phiin, xout, yout, consts)
         forxx = for17(lamin, phiin, xout, yout, consts)
         forxx = for18(lamin, phiin, xout, yout, consts)
         forxx = for19(lamin, phiin, xout, yout, consts)
         forxx = for20(lamin, phiin, xout, yout, consts)
         }
      else
         forxx = ERR

   return
   end


# - invxx - switch routine to inversion
   integer function invxx(prjno, xin, yin,
                    lamout, phiout, consts)
   PREC xin, yin, lamout, phiout, consts(ARB)
   integer prjno, inv01, inv03, inv04, inv05, inv06, inv07,
       inv08, inv09, inv10, inv11, inv12, inv13, inv14, inv15,
       inv16, inv17, inv18, inv19, inv20

      switchto prjno {
         invxx = inv01(xin, yin, lamout, phiout, consts)
         invxx = ERR
```

```
        invxx = inv03(xin, yin, lamout, phiout, consts)
        invxx = inv04(xin, yin, lamout, phiout, consts)
        invxx = inv05(xin, yin, lamout, phiout, consts)
        invxx = inv06(xin, yin, lamout, phiout, consts)
        invxx = inv07(xin, yin, lamout, phiout, consts)
        invxx = inv08(xin, yin, lamout, phiout, consts)
        invxx = inv09(xin, yin, lamout, phiout, consts)
        invxx = inv10(xin, yin, lamout, phiout, consts)
        invxx = inv11(xin, yin, lamout, phiout, consts)
        invxx = inv12(xin, yin, lamout, phiout, consts)
        invxx = inv13(xin, yin, lamout, phiout, consts)
        invxx = inv14(xin, yin, lamout, phiout, consts)
        invxx = inv15(xin, yin, lamout, phiout, consts)
        invxx = inv16(xin, yin, lamout, phiout, consts)
        invxx = inv17(xin, yin, lamout, phiout, consts)
        invxx = inv18(xin, yin, lamout, phiout, consts)
        invxx = inv19(xin, yin, lamout, phiout, consts)
        invxx = inv20(xin, yin, lamout, phiout, consts)
        }
    else
        invxx = ERR

return
end
```

Appendix G

A distribution tape (9-track, ASCII) of the projection software discussed in this report is available from the author in either 800- or 1600-bpi recording density. The data are recorded as 80-character logical records in 3600-character blocks and are divided into five files:

file 1 - macro definitions (Appendix B),

file 2 - projections procedures (Appendix C),

file 3 - support procedure library (Appendix D),

file 4 - FORTRAN version of projections, and

file 5 - FORTRAN version of support library.

Files 4 and 5 are the output of the RATFOR preprocessor with the 'double precision' option and are included for those without access to a RATFOR system.