IN-61
145557
P.39

NASA Technical Memorandum 4445

MAPPER: A Personal Computer Map Projection Tool

Steven A. Bailey

FEBRUARY 1993

NASA

NASA Technical Memorandum 4445

MAPPER: A Personal Computer Map Projection Tool

Steven A. Bailey
Wallops Flight Facility
Wallops Island, Virginia

NASA
National Aeronautics and
Space Administration
Office of Management
Scientific and Technical
Information Program
1993

Contents

Summary

MAPPER is a set of software tools designed to let users create and manipulate map projections on a personal computer (PC). The capability exists to generate five popular map projections. These include azimuthal, cylindrical, mercator, lambert, and sinusoidal projections. Data for projections are contained in five coordinate databases at various resolutions. **MAPPER** is managed by a system of pull-down windows. This interface allows the user to intuitively create, view, and export maps to other platforms.

Introduction

Mapping programs and coordinate databases were developed by the Observational Science Branch for use in scientific instrument development flights on aircraft. A subset of these programs was initially written for an aircraft UNIX microcomputer. Maps were displayed along with real-time position data. Interest in maps then carried over to pre and post flight data analysis. **MAPPER** was then created for the PC. The programs contained herein make up a general purpose package for map generation. Maps can be displayed directly on a PC monitor as well as exported to a low cost CAD program for further manipulation. At the present time, translation is only supplied for DesignCAD, which is produced by American Small Business Computers of Pryor, Oklahoma. Once in DesignCAD, further translations can be made along with output to a multitude of printers and plotters. The map coordinate databases supplied are of various resolutions; catering to the needs of the Wallops Flight Facility. Other databases of higher resolution and larger area are available and will be incorporated when needed.

Installation

These programs **must** run from a hard disk with at least **4 megabytes** free. This is due to the immense size of two of the map databases.

To install **MAPPER**, place floppy diskette number **1** in any one of your floppy drives. While logged onto your hard disk, type:

A:INSTALL A: C:

This assumes your floppy drive is **A** and your hard disk is **C**. If this is not the case, simply change the command line with the appropriate drive information.

The installation procedure requires two floppy diskettes. All programs and databases have been packed, so unpacking will take a few minutes...be patient!

It is suggested that your **AUTOEXEC.BAT** file contain the statement:

GRAPHICS

GRAPHICS.COM should also exist...check your **DOS** directory. This will allow you to make a crude copy on your printer from the screen plot by typing **Ctrl PrtSc**.

Startup

Projec is a user interface program used to run **MAPPER** and all of its utilities. **Projec** is designed with pull-down windows to ease user interaction with **MAPPER**.

At the **DOS** prompt, type **projec** <ENTER> (see Figure 1).

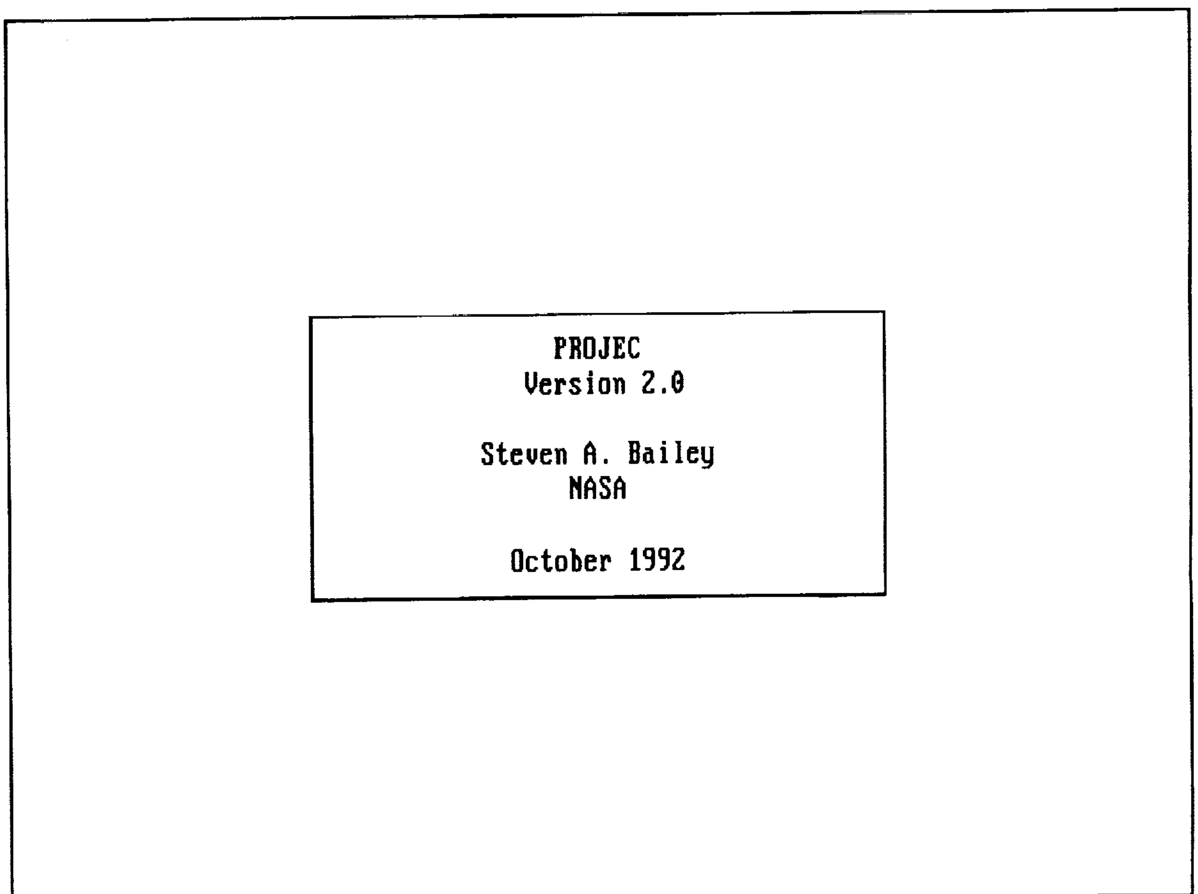


Figure 1. Author title window.

Strike any key to enter the startup window.

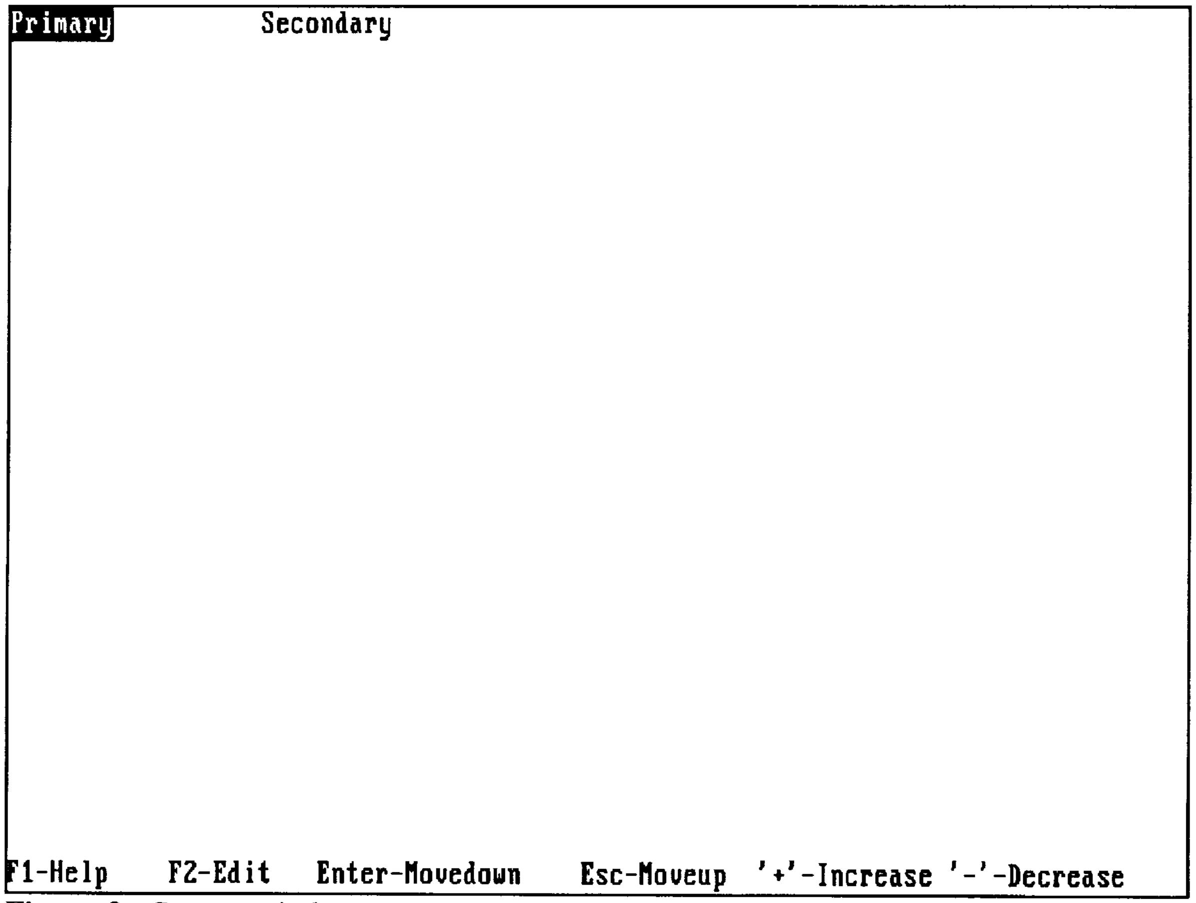


Figure 2. Startup window.

As evident, the main window is quite unassuming (see Figure 2). At the top of the screen are **2** categories. By using the left- and right-arrow keys, the highlighted bar moves from category to category. Also, by striking the letter key for the first letter of each category word, the highlighted bar moves directly to that category.

At the bottom of the screen are **5** function key descriptors. The highlight bar does not move here. These function keys are used throughout all windows for various purposes. Since we are now in the startup window, only the **F1** key applies. Move to either of the **2** categories at the top of the screen and strike the **F1** key. You should see a help window appear, explaining the particular category you have chosen.

Primary

The left-most section of the startup window is called **Primary**. This is where the most used parameters dealing with **MAPPER** are found. To activate the **Primary** window, simply strike the **ENTER** key (see Figure 3).

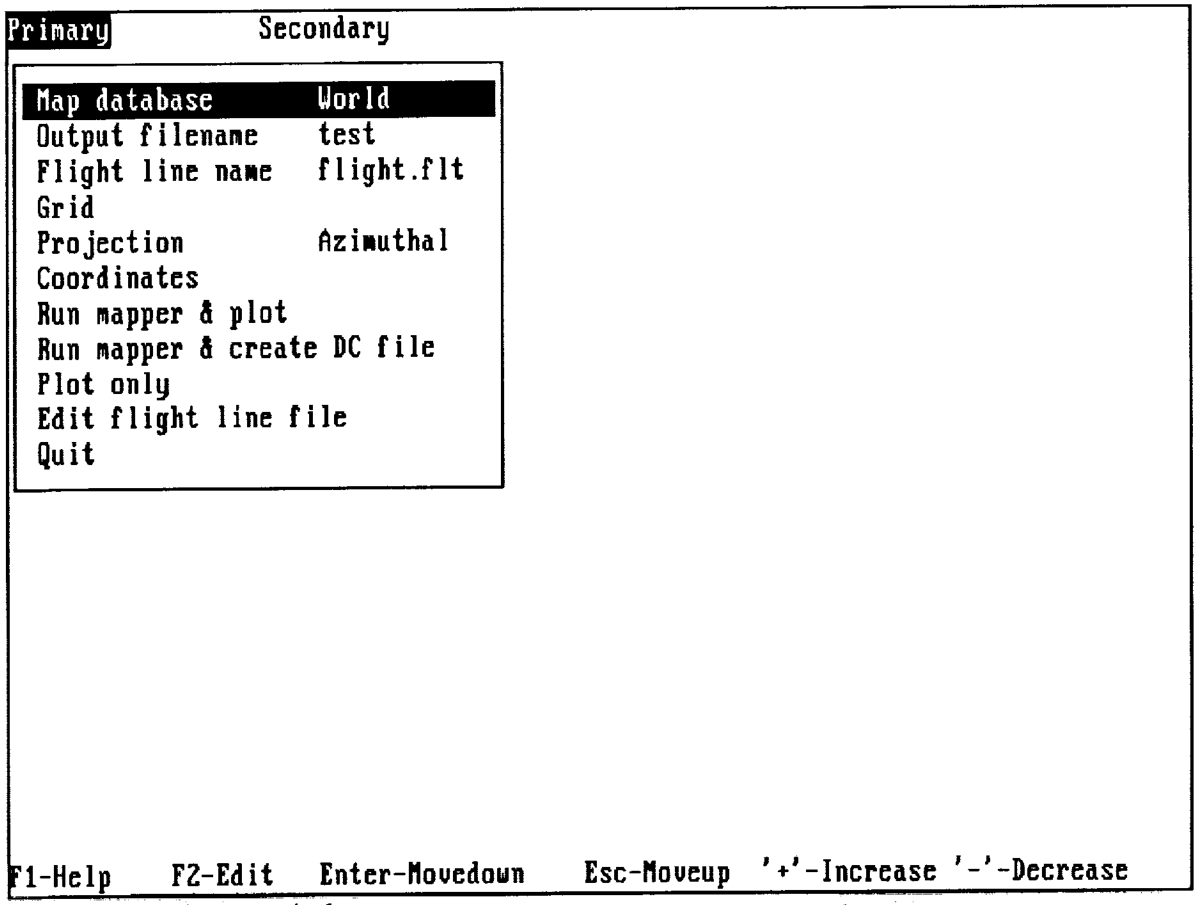


Figure 3. Primary window.

As evident, there are **11** parameters in the **Primary** window. Parameters shown in **blue** text indicate that you must move farther down to a subsequent window using the **ENTER** key. Parameters shown in **yellow** text mean you can edit using the **F2** key. Finally, parameters shown in **black** text mean you can change these variables using the **+** and **-** keys. For **black** parameters, the **PgUp**, **PgDn**, and **Home** keys can also be used. **PgUp** and **PgDn** increase and decrease, respectively, by a factor of 10. The **Home** key sets **black** parameters to their initial or **Home** state. To remember how parameters are color-coded, look at the **5** function key descriptors at the bottom of the screen. The **F1** key displays help (see Figure 9 for an example).

Map database - this defines the database file you use to create a map projection (see Figure 4). At present, there are **5** static databases and **1** user-defined database to choose from:

World - a general-purpose database that contains low-resolution geographic data of the entire world.

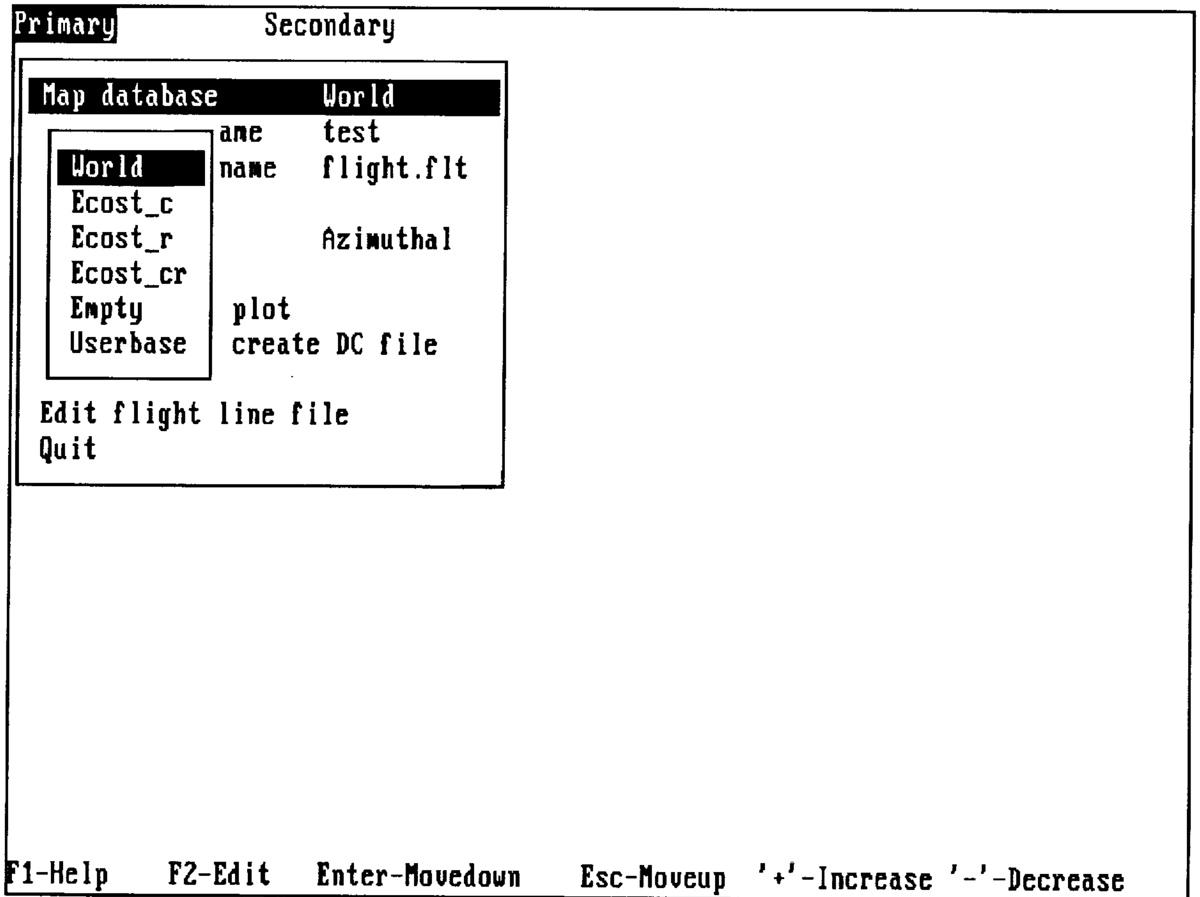


Figure 4. Database window.

Ecost_c - a high-resolution database containing Eastern U.S. coastal bays and shoreline.

Ecost_r - a high-resolution database containing Eastern U.S. rivers.

Ecost_cr - a high-resolution database containing Eastern U.S. coastal bays, shoreline, and rivers.

Empty - an empty database used when only a **lat., lon.** grid of an area is needed.

Userbase - an editable parameter for entering a user-defined database created when using the **Create database from flight line** found under the **Secondary** category.

Output filename - this defines the name of your output file (see Figure 3).

Flight filename - this defines the name of the datafile you use to plot a flight line with your map. This name **must** be different from your **output filename** (see Figure 3).

Grid - this defines your map grid in **degrees** of **latitude** and **longitude**. A grid entry of **0** for **latitude** and **0** for **longitude** produces a map with **NO** grid lines (see Figure 5).

Projection - this defines **1** of **5** types of map projections available (see Figure 6):

Azimuthal - used when true direction and distance are needed from a center point. It is commonly used for areas up to and including **1** hemisphere. Any center point on the globe is valid, making this projection the most useful. **DO NOT USE THIS PROJECTION UNLESS A MATH COPROCESSOR IS INSTALLED IN YOUR COMPUTER.**

Mercator - used for navigation purposes, because a course of constant bearing can be drawn between **2** points that maintains directional accuracy. This projection becomes **very** distorted above **85** degrees **North** or below **85** degrees **South latitude**.

Lambert - used when the "**right shape**" is needed. It is only good for areas with an **East-West** extent of temperate latitudes. The U.S. is a good example. It is valid between **10** and **80** degrees **North** and **South latitudes**.

Sinusoidal - used for small areas, or those of a **North-South** extent. Can also be used to show the entire Earth or its quadrants.

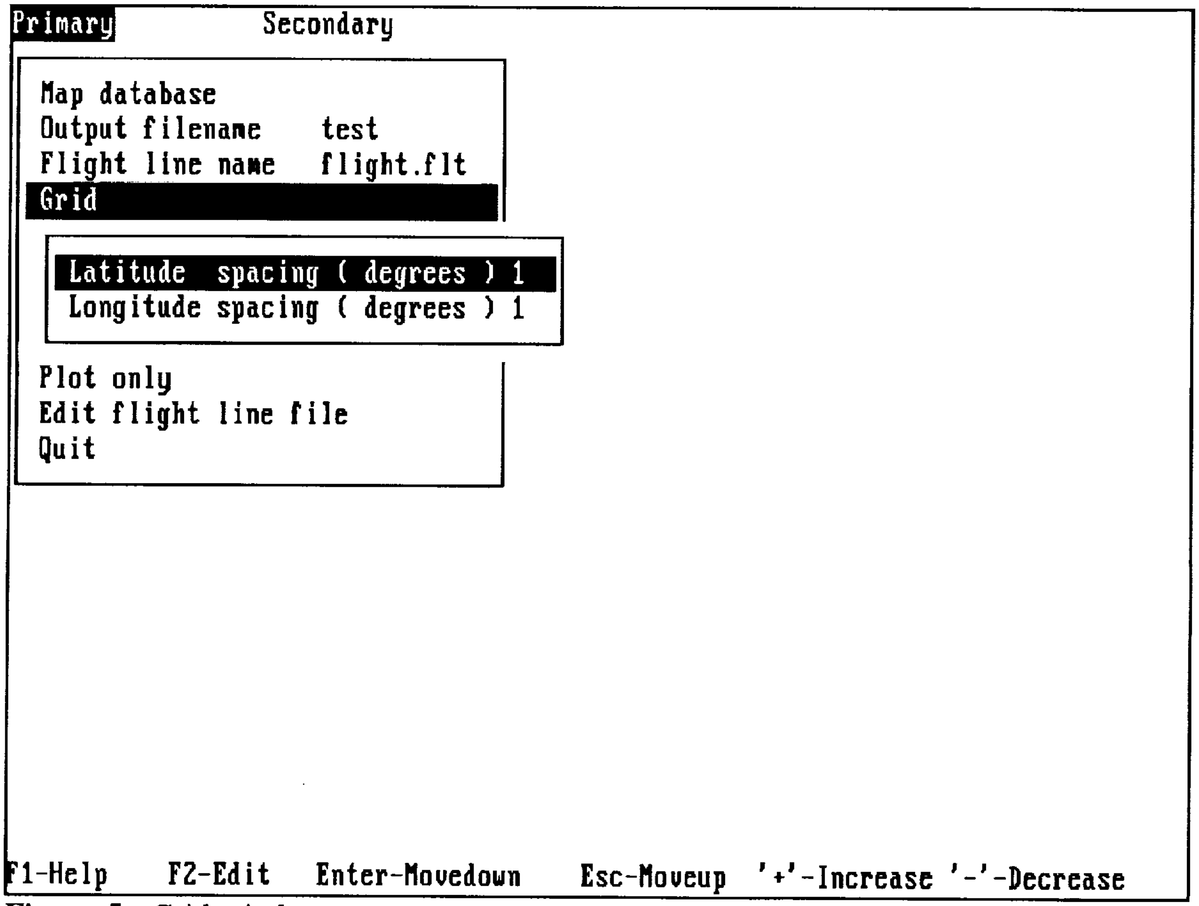


Figure 5. Grid window.

Cylindrical - map area corresponds with area found on projection. It is most commonly used in areas near the Equator. It is valid for any part of the globe, but the poles do become distorted.

Coordinates - this defines the map scale the user creates for a given projection type. When using the **Azimuthal** projection, choose the **Azimuthal coordinates** sub-window. When using a projection other than azimuthal, choose the **All other coordinates** sub-window. **Remember** to follow the convention of the **cartesian coordinate system** (see Figures 7 and 8).

Center latitude - this is the center point in latitude needed when using the **Azimuthal** projection.

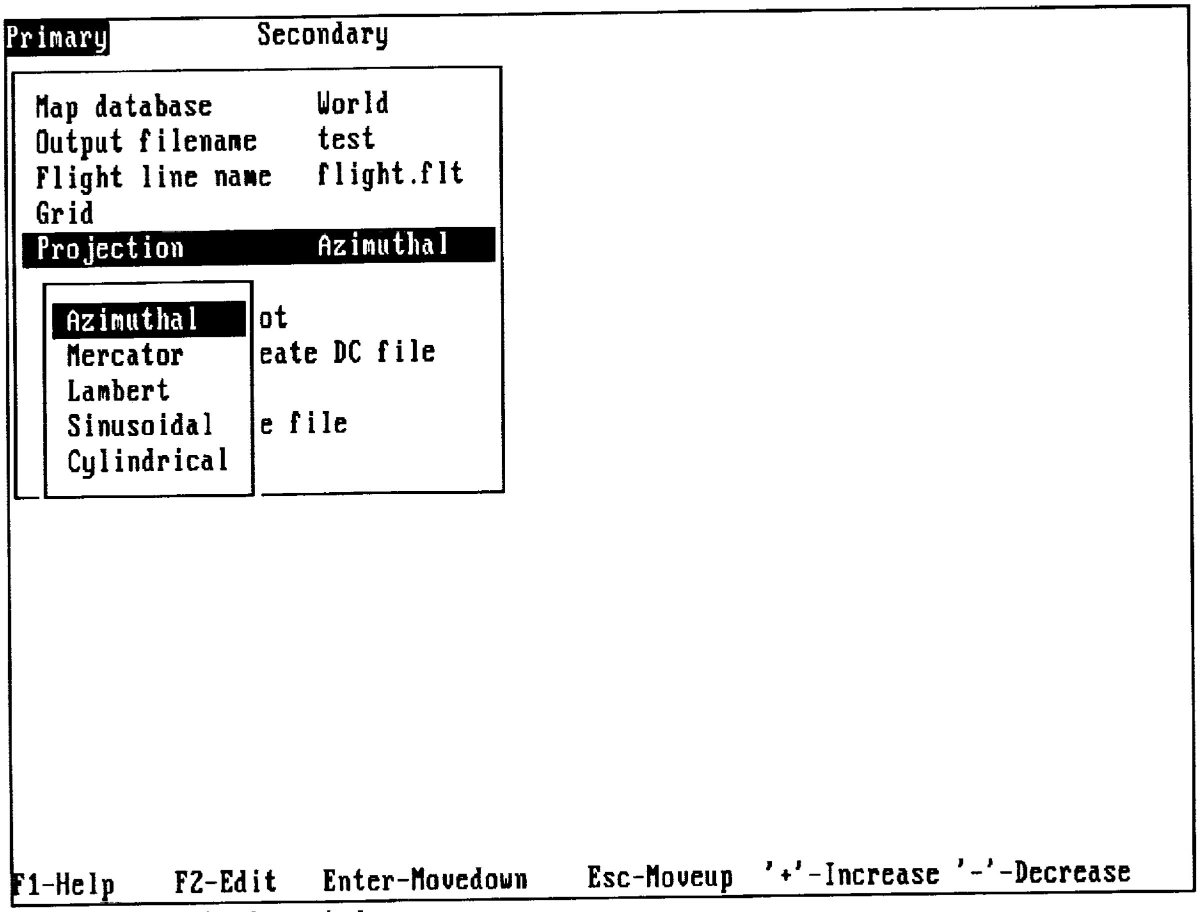


Figure 6. Projection window.

Center longitude - this is the center point in longitude needed when using the **Azimuthal** projection.

Range from center - this is the range in nautical miles from the center point that the **Azimuthal** map projection will span.

Min. latitude - this is the minimum latitude needed when using all projections except **Azimuthal**.

Max. latitude - this is the maximum latitude needed when using all projections except **Azimuthal**.

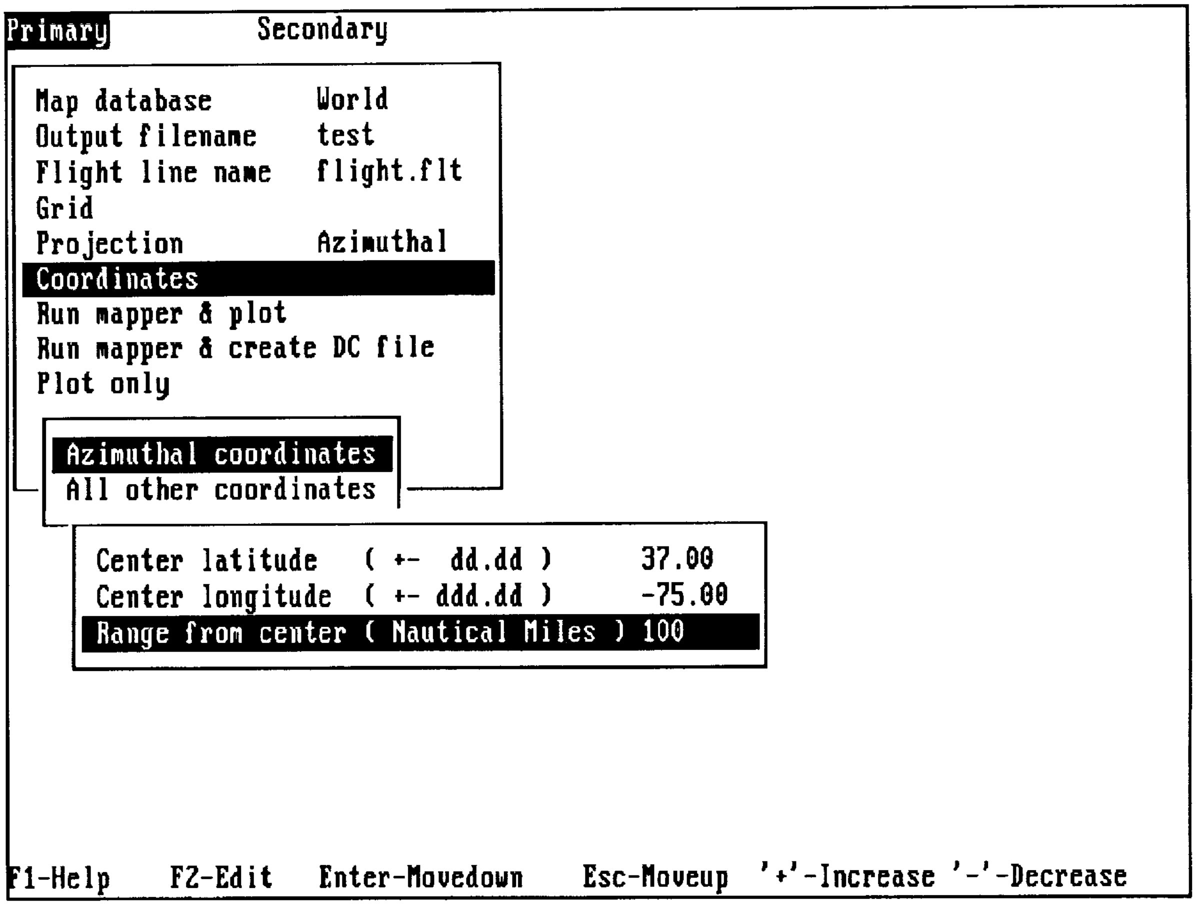


Figure 7. Azimuthal window.

Min. longitude - this is the minimum longitude needed when using all projections except **Azimuthal.**

Max. longitude - this is the maximum longitude needed when using all projections except **Azimuthal**.

Run mapper & plot - this runs **MAPPER**, creates a plot file, and then plots that file to the screen (see Figure 3).

Run mapper & create DC file - this runs **MAPPER** and creates a **DesignCAD** file from the **MAPPER** output (see Figure 3).

Plot only - this only plots a screen plot file which was previously created during a call to **Run mapper & plot** (see Figure 10).

```
Primary            Secondary

 Map database       World
 Output filename    test
 Flight line name   flight.flt
 Grid
 Projection         Azimuthal
 Coordinates
 Run mapper & plot
 Run mapper & create DC file
 Plot only

   Azimuthal coordinates
   All other coordinates

     Min. latitude   ( +-  dd.dd )  36.25
     Max. latitude   ( +-  dd.dd )  38.50
     Min. longitude  ( +- ddd.dd )  -77.00
     Max. longitude  ( +- ddd.dd )  -74.50

F1-Help   F2-Edit   Enter-Movedown   Esc-Moveup  '+'-Increase '-'-Decrease
```

Figure 8. Others window.

Edit flight line file - this is used to edit the ASCII flight-line file. The editor is a public-domain **Wordstar** clone. You must have a filename entered in the **Flight filename** parameter before calling this editor (see Figure 3).

Quit - this is used to quit or exit from **PROJEC**. You are then returned to the point at which you called **PROJEC** (see Figure 3).

Secondary

The right-most section of the startup window is called **Secondary**. This is where the least used parameters dealing with **MAPPER** are found. Strike the **ENTER** key to find which window follows **Secondary** (see Figure 10).

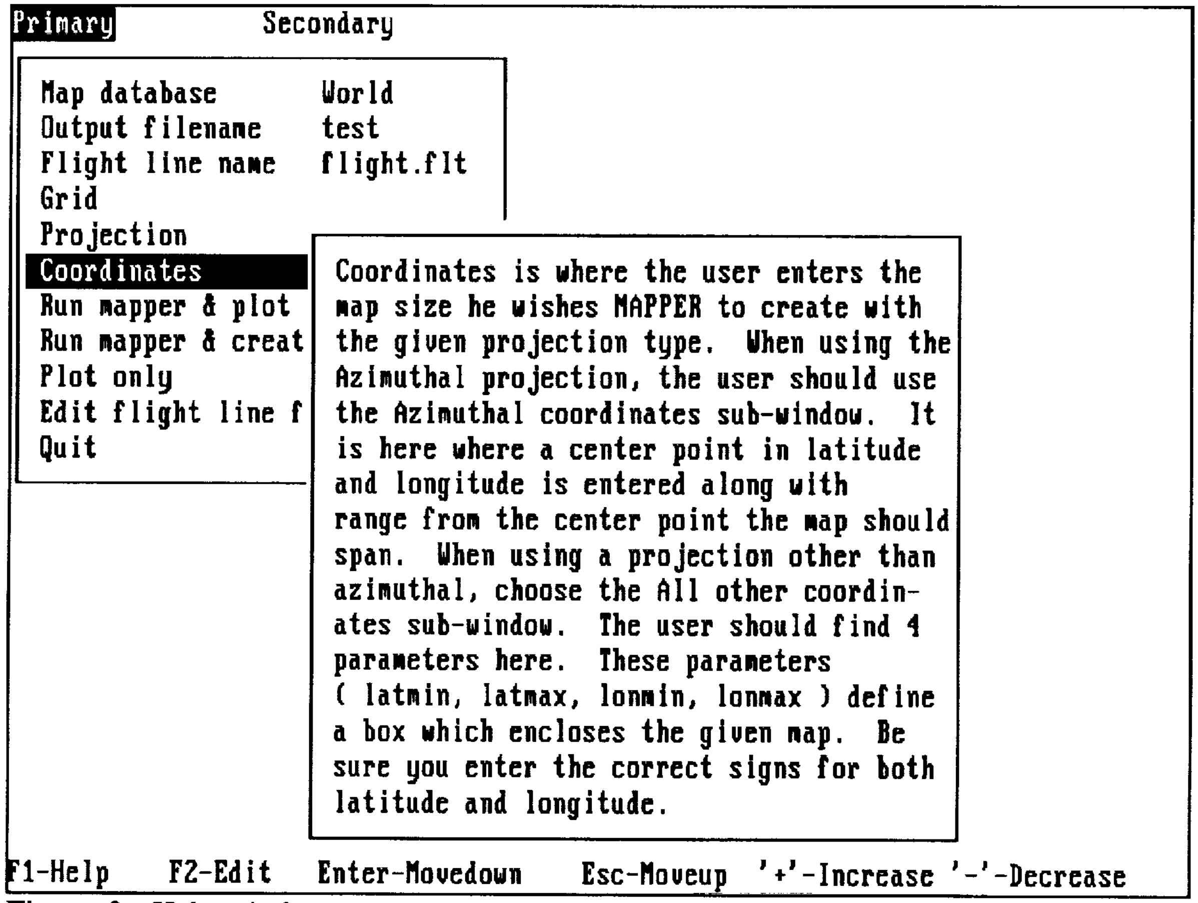


Figure 9. Help window.

Run mapper & plot - this runs **MAPPER**, creates a plot file, and then plots that file to the screen (see Figure 10).

Run mapper & create DC file - this runs **MAPPER** and creates a **DesignCAD** file from the **MAPPER** output (see Figure 10).

Plot only - this only plots a screen plot file which was previously created during a call to **Run mapper & plot** (see Figure 10).

Create DC file only - this only creates a DesignCAD file from an already created map projection file. A map projection file occurs if you ran **Run mapper & plot** prior to this command (see Figure 10).

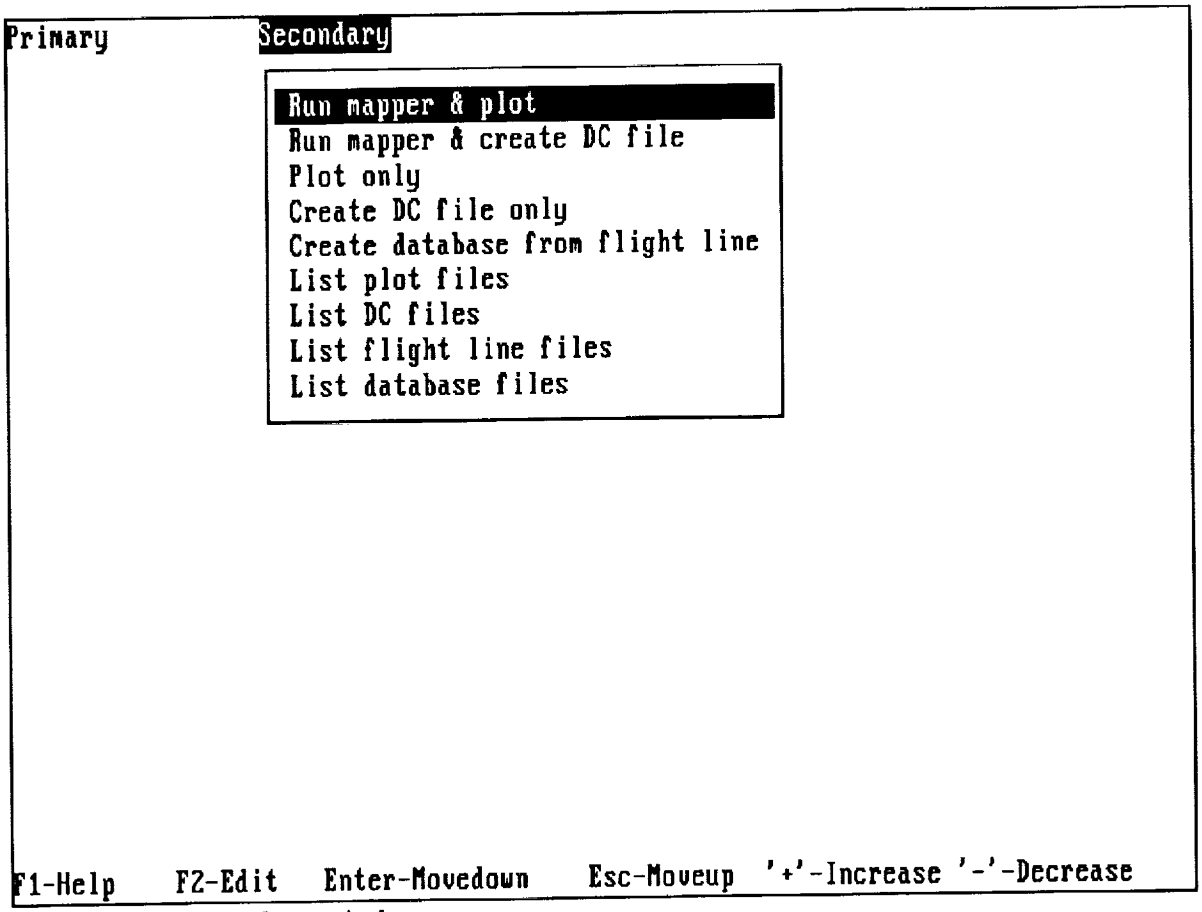


Figure 10. Secondary window.

Create database from flight line - this creates a binary map database from a user-defined ASCII file. This ASCII file is in the **flight.flt** format. To create this binary map database file, the appropriate filename is the **Flight line name** parameter found under the **Primary** category (see Figure 10).

List plot files - this executes a **DOS dir *.tp** command to display all screen plot files in the current directory (see Figure 10).

List DC files - this executes a **DOS dir *.dc2** command to display all DesignCAD files in the current directory (see Figure 10).

List flight line files - this executes a **DOS dir *.flt** command to display all flight-line files in the current directory (see Figure 10).

List database files - this executes a **DOS dir *.bin** command to display all database files in the current directory (see Figure 10).

Acknowledgements

I gratefully acknowledge the contributions of the following people:

Carl Ulbrich of Clemson University, who developed the original BASIC program and database for the Wallops area;

John Cavanaugh of Goddard Space Flight Center, who incorporated the original world database;

Dave Clem of Wallops Flight Facility, whose inspiration, editing, and initial debugging made this set of programs possible;

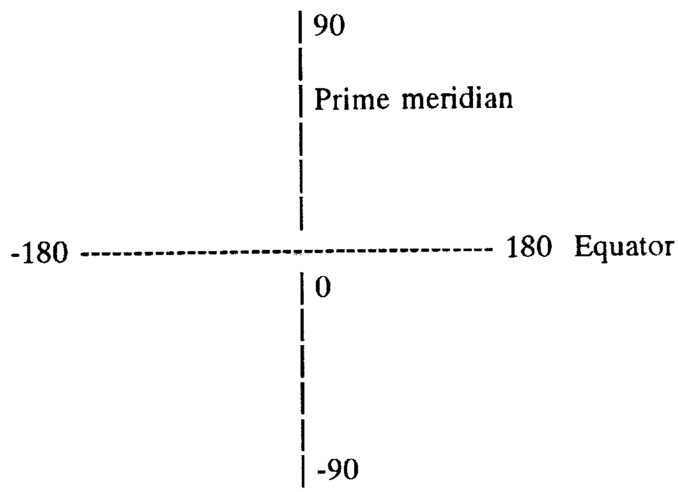
Wayne Wright of Wallops Flight Facility, who sparked my interest in map projections; and

Charles Vaughn of Wallops Flight Facility, whose persistence in finding software bugs inspired me to update **MAPPER**.

Appendix A

Coordinate System

MAPPER uses a cartesian coordinate system. All parallels north of the Equator are positive and those south, are negative. All meridians west of the prime meridian are negative and those east, are positive.



Appendix B

Explanation of Mapper

MAPPER.EXE is the program that creates map projections. It can be run without the aid of **PROJEC** and is set up like a UNIX system program. All interaction with the program is made on the command line. Here is an example:

mapper -mf world.bin -df wff -g 1 1 -a 37 -75 100

Now...let's dissect this command line.

1.	**mapper**	This is the name of the mapping program.
2.	**-mf**	This is a flag meaning **map file**. The next parameter is the name of a map database file.
3.	**world.bin**	This is the name of the map database you will read in.
4.	**-df**	This is a flag meaning **data file**. The next parameter is the name of the output file.
5.	**wff**	This is the name of the data file where your map projection will go.
6.	**-g**	This a flag meaning **grid**. The next two parameters refer to spacing in degrees of the produced map grid.
7.	**1 1**	The first **1** refers to latitude spacing, while the second **1** refers to longitude spacing. Zeroes here produce no grid.
8.	**-a**	This is the flag defining projection type. In this instance, **a** means azimuthal. The next three parameters follow this flag.
9.	**37 -75 100**	The first number is the center point in latitude. The second number is the center point in longitude. The third number is the range from the center point in nautical miles.

To see the **MAPPER** help file, type: **mapper** <ENTER>

```
Enter:  mapper [options]

        -mf                   = map database filename.
        -df                   = output filename.
        -ff                   = flight line or new database.
        -g    [ degrees  ]    = turn grid on with desired accuracy.
        -m    [ window   ]    = mercator projection.
        -l    [ window   ]    = lambert projection.
        -s    [ window   ]    = sinusoidal projection.
        -c    [ window   ]    = cylindrical projection.
        -a    [ azimuth  ]    = azimuthal projection.

              window          = latmin latmax lonmin lonmax.
              azimuth         = lat lon range.
              degrees         = lat lon.

Examples:

        mapper -mf world.bin -df wallops -g 1 1 -a 37 -75 100
        mapper -mf world.bin -df whole -g 30 30 -s -90 90 -180 180
        mapper -mf empty.bin -df grid -g 30 30 -a 0 0 5400
        mapper -mf ecost_cr.bin -df wff -ff flight.flt -g 1 1 -a 37 -75 100
```

After producing a map projection, you should view it. I have provided two methods of viewing. The first produces a map on your PC, provided you have CGA, EGA, or VGA capabilities. This utility is helpful when a quick view is needed.

The second method involves exporting your map projection to DesignCAD.

To create a screen plot file and plot, type: **rplot filename** <ENTER>

To view a plot file already created, type: **plot filename** <ENTER>

To create a DesignCAD file, type: **cplot filename** <ENTER>

Now, when you run DesignCAD, retrieve: **filename.dc2**

Appendix C

General Information

Several steps are involved when producing a map. All steps listed are handled by the **PROJEC** shell. You should be aware what steps are used in case you want to bypass the **PROJEC** shell. When making a map, the desired database is searched using the appropriate projection and coordinate information. Running **MAPPER.EXE** directly will do this. The file created as the **Output filename** is in a **type-length** binary format containing various datatypes. When doing screen plots, the program **READTYPE.EXE** reads this binary file of **type-length** format and creates an ASCII file (with the **.tp** extension) which is used by **NEWPLOT.EXE**. **NEWPLOT.EXE** can be called directly, but it is easier to use one of the two batch files supplied. **RPLOT.BAT** is used when binary to ASCII translation is needed. **PLOT.BAT** is used when an ASCII translated plot file (with a **.tp** extension) exists and only a screen plot is desired.

When a DesignCAD file is needed, a program called **RTYPECAD.EXE** is used to convert the binary file of **type-length** format to an ASCII file in the DesignCAD or **.dc2** format. For convenience, use the supplied batch file called **CPLOT.BAT**. A **.dc2** file can be read directly by DesignCAD for further manipulation. Using the **DCEXPORT.EXE** utility of DesignCAD, other file formats can be created from your **.dc2** format. They include IGES, VENTURA (GEM), and POSTSCRIPT. Also, the utility **DCPRINT.EXE** allows you to output to many printers now on the market. Run the DesignCAD utility **DCSETUP.EXE** to choose your output devices, including plotters. To plot to a device, you must be in DesignCAD, where you load your **.dc2** drawing, and plot.

The directory on which **MAPPER** resides can become congested after creating just a few maps, screen plots, and DesignCAD files. Erase all files with the **.tp** and **.dc2** extensions if you need more space. As long as you keep the binary projection files (the files with no extensions), these two file types can be regenerated.

Appendix D

Flight-Line Files

Files with the **.flt** extension are created by you, the user. These files are in ASCII format and represent data you want plotted on your map projection. The following example represents two flight lines which have a break between them. This example can be found in the file **FLIGHT.FLT**:

```
37 -75
37 -76
37 -77
# #
34 -75
34.5 -75
35 -75.3
35.5 -75.5
36 -76
# #
```

Each line represents a point made up of latitude, a space, longitude, and then a hard return. The **# #** symbols mark the end of a flight line and are absolutely necessary. If your data is one continuous flight line, you only need **# #** symbols at the end. To represent single data points, repeat coordinates twice followed by the **# #** symbols. A flight-line file must be created by a text editor or word processor, which **does not** insert control characters in your text. **WORDSTAR** (in the non-document mode) and the DOS utility **EDLIN** will suffice. **WORDPERFECT** will work **only** if your file is exported as a **DOS text** file or a **Generic** file. For convenience, use the supplied editor found under the **Edit flight line file** parameter.

When using projections, such as the azimuthal or lambert, lines of latitude and longitude can become curved. This is evident by the grid system you use. Sometimes, geographical boundaries on the map you produce do not properly follow the grid lines as they should. This is no fault of the program. The fault lies in the map database. When using a low-resolution database like **World**, it is not uncommon for a line (which represents hundreds of miles) to be made up of only two points. This can be found in many of the lines which mark state boundaries. Projections (like the azimuthal and lambert) which make curves cannot make a curve from two points; hence a straight line is drawn. Likewise, when you create flight lines, make them as high a resolution as you can. In other words, make the distance from one point to the next as close as possible. Experiment with this until you have a suitable resolution.

Appendix E

Program Files

CBAY	.	Map projection of Chesapeake Bay
SC	.	" " " South Carolina.
US_A	.	" " " U.S. using azimuthal projection.
US_C	.	" " " U.S. using cylindrical projection.
US_L	.	" " " U.S. using lambert projection.
US_M	.	" " " U.S. using mercator projection.
US_S	.	" " " U.S. using sinusoidal projection.
INSTALL	.BAT	Batch file used to install mapper on hard disk.
CPLOT	.BAT	Batch file that generates cad file.
PLOT	.BAT	Batch file that plots translated data to screen.
RPLOT	.BAT	Batch file that translates data, then plots to screen.
EMPTY	.BIN	Database used to produce grid only.
WORLD	.BIN	Database of world coordinates.
ECOST_C	.BIN	Database of Eastern U.S. coastal shore lines and bays.
ECOST_R	.BIN	Database of Eastern U.S. rivers.
ECOST_CR	.BIN	Database combining ECOST_C.BIN and ECOST_R.BIN.
*	.TP	All filenames with this extension are translated projection files, used for screen plots.
*	.DC2	All filenames with this extension are translated, ASCII maps in the DesignCAD format.
*	.FLT	All filenames with this extension are user-generated ASCII files in the lat, lon format.
*	.BGI	All filenames with this extension are graphic display setup files needed for screen plots.
MAPPER	.EXE	The main program.
PROJEC	.EXE	The user-friendly, shell program.
RTYPECAD	.EXE	Program converts binary map file to DesignCAD file with **dc2** extension.
READTYPE	.EXE	Program converts binary map file to ASCII plot file with **tp** extension.
NEWPLOT	.EXE	Program reads in plot file with **tp** extension and plots it to the screen.
FLTOWMAP	.EXE	Program used to convert files with **flt** extension to map database format of **bin** extension.
MAPVAR	.EXE	Program that strips out variables from binary map projection files.

EDIT	.EXE	Public-domain WordStar-like editor.
MERGEPRO	.EXE	Program that merges two binary map projection files.
JECTIONS	.C	This is a **C** source file which contains functions for all five projections along with preparatory information.
MAPPER	.HLP	Mapper help file.

Appendix F

Projection Source File

```
/*--------------------------------------------------------------------------------------------
 * jections.c
 *--------------------------------------------------------------------------------------------
 * The following functions each generate a map projection when called with
 * the variables lat and lon. These variables correspond to the latitude
 * and longitude in question. Prior to these function calls, a few variables
 * need to be defined. These variables are generated when a map projection is
 * created. A program called mapvar.exe is executed afterwards to extract these
 * variables. The projections azimuthal, mercator, cylin, and sinu only
 * require 3 variables to operate. These variables are latmid, lonmid, and
 * fact. Again, they are printed out when the program mapvar.exe is run. The
 * lambert projection requires 7 variables. They are latmid, lonmid,
 * fact, latmdis, base, bangle, and tot. They too are printed when
 * mapvar.exe is run.
 *
 * The following functions where written in the C programming language and
 * have been slightly altered for readability. The intention of this printout
 * is to show the user the structure of each projection. It is his/her job to
 * rewrite each projection in the appropriate language. Follow each algorithm
 * exactly! They have all been tested and they do work. They make up the heart of
 * MAPPER. Good luck!
 *--------------------------------------------------------------------------------------------
 */

/*--------------------------------------------------------------------------------------------
 * The following are static global variables needed for the general operation
 * of all projections.
 *--------------------------------------------------------------------------------------------
 */

double rad    = 3.14159265 / 180.0;    /* Converts degrees to radians */
double radius = 3437.746;              /* Radius of Earth at equator */
double add    = 250.0;                 /* Scaling factor needed to convert */
                                       /* screen coordinates from -250 through 250 */
                                       /* to 0 through 500 */
```

```
/*------------------------------------------------------------------------------------------
 * The following are global variables needed for the general operation of all
 * projections.
 *------------------------------------------------------------------------------------------
 */

double  x;          /* The x component of the final screen coordinate */
double  y;          /* The y component of the final screen coordinate */
double  crange;     /* Range in nautical miles from initial reference point */
double  lonrad;     /* Radius of sphere in nautical miles from given latitude */
double  lattemp;    /* Temporary storage */
double  lontemp;    /* Temporary storage */

/*------------------------------------------------------------------------------------------
 * The following are the variables needed to drive all projections.  They
 * must be determined using mapvar.exe before any projection function can
 * be called.
 *------------------------------------------------------------------------------------------
 */

double  latmid;     /* Center reference point in latitude for each projection */
double  lonmid;     /* Center reference point in longitude for each projection */
double  fact;       /* Factor which converts nautical miles to screen coordinates */
double  latmdis;    /* Distance in nautical miles from point above pole to lat. mid. point */
double  base;       /* Distance in nautical miles from Earth center to base triangle point */
double  bangle;     /* Angle bet. base and hypotenuse of triangle */
double  tot;        /* Distance in nautical miles of hypotenuse */

/*------------------------------------------------------------------------------------------
 * The following are macros which need to be defined.  They are presented in
 * C fashion.
 *------------------------------------------------------------------------------------------
 */

#define  SIGN( x )   (( x < 0.0 ) ? -1.0 : 1.0 )           /* Return sign of given 'x' variable */
#define  ABS( x )    (( x < 0.0 ) ? ( x * -1.0 ) : x )     /* Return abs. value of 'x' variable */
```

```
azimuthal( lat, lon )
double lat, lon;
{
        static double a,b,c,B,C,tC, cosa, cosb, sina;

        /* Return for polar projection */

        if ( ( lat == latmid ) && ( lon == lonmid ) ) {
          lattemp = 90.0;
          lontemp = 0.0;
          crange  = 0.0;
        }
        else {
          if ( ABS( latmid ) != 90.0 )
            a = 90.0 - latmid;          /* Arc-length of given point */
          else
            a = 90.0 - 89.9999 * SIGN( latmid );

          b = 90.0 - lat;               /* Arc-length of center point */
          C = lon - lonmid;

          if ( ABS(C) > 180.0 ) {
           if ( C < 0.0 )
              C += 360.0;
            else
              C -= 360.0;
         }

          tC = C;

          a *= rad;
          b *= rad;
          C *= rad;

          cosa = cos( a );
          cosb = cos( b );
          sina = sin( a );

          c = cosa * cosb + sina * sin( b ) * cos( C );
          c = acos( c );

          B = ( cosb - cosa * cos( c ) ) / ( sina * sin( c ) );
```

```
            if ( B <= -1.0 )
              B =  180.0;
            else if ( B >= 1.0 )
              B =  0.0;
            else {
              B = acos( B );
              B /= rad;
            }

            if ( tC < 0.0 )
              B *= -1.0;

            crange  = c * radius;
            lontemp = B;
          }

        x = crange * sin( rad * lontemp ) * fact + add;
        y = crange * cos( rad * lontemp ) * fact + add;
}

/*------------------------------------------------------------------------------------------*/

mercator( lat, lon )
double  lat, lon;
{
        lonrad = radius * cos( rad * lat );
        x = radius * ( rad * ( lon - lonmid ) ) * fact + add;
        y = (lonrad * rad) / 60.0;
        y = radius * ( rad * ( lat   - latmid ) ) * fact / y + add;
}
```

```
lambert( lat, lon )
double  lat, lon;
{
        static double temp;

        temp = base * sin( rad * ABS( lat ) ) / sin( rad * ( 180.0 - bangle - ABS( lat ) ) );
        crange = tot - temp;

       temp = lon - lonmid;

        x = crange * sin( rad * temp ) * fact + add;
        y = crange * cos( rad * temp );
        y = ( latmdis - y ) * fact;

        if ( latmid < 0.0 )
          y = ( y * -1.0 ) + add;              /* Must flip for S. hemp. */
        else
          y += add;
}

/*-------------------------------------------------------------------------------------------*/

cylin( lat, lon )
double  lat, lon;
{
           x   =   radius * ( rad * ( lon - lonmid ) ) * fact + add;
            y   = ( radius * sin( rad * lat ) - radius * sin( rad * latmid ) ) * fact + add;
}

/*-------------------------------------------------------------------------------------------*/

sinu( lat, lon )
double  lat, lon;
{
        lonrad = radius * cos( rad * lat );
        x      = lonrad * ( rad * ( lon - lonmid ) ) * fact + add;
        y      = radius * ( rad * ( lat - latmid ) ) * fact + add;
}
```

Appendix G

Sample Maps

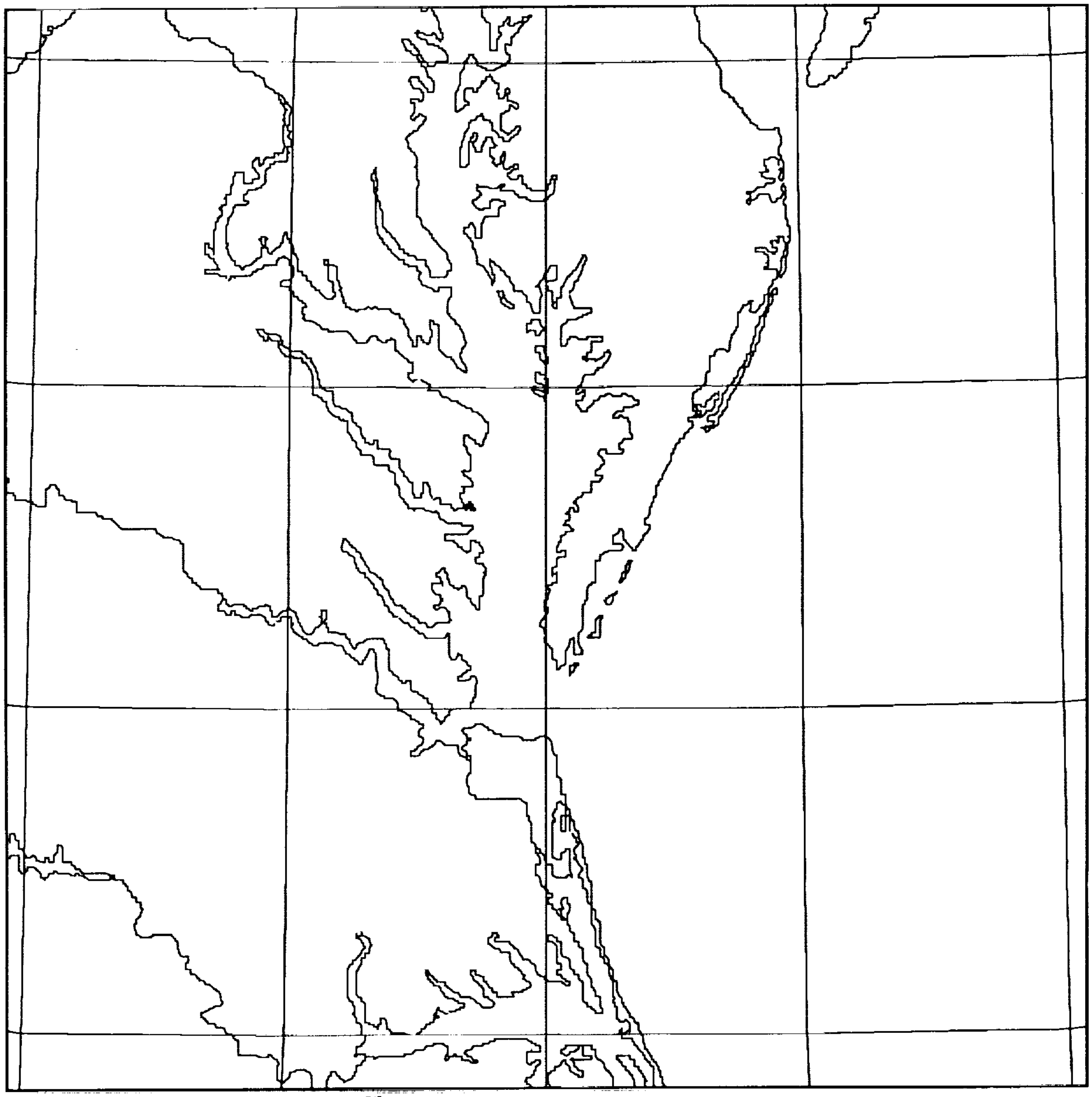


Figure 11. Map: Cbay
Database: Ecost_cr.bin
Projection: Azimuthal
Coordinates: 37.5 -76 100
Grid: 1 1

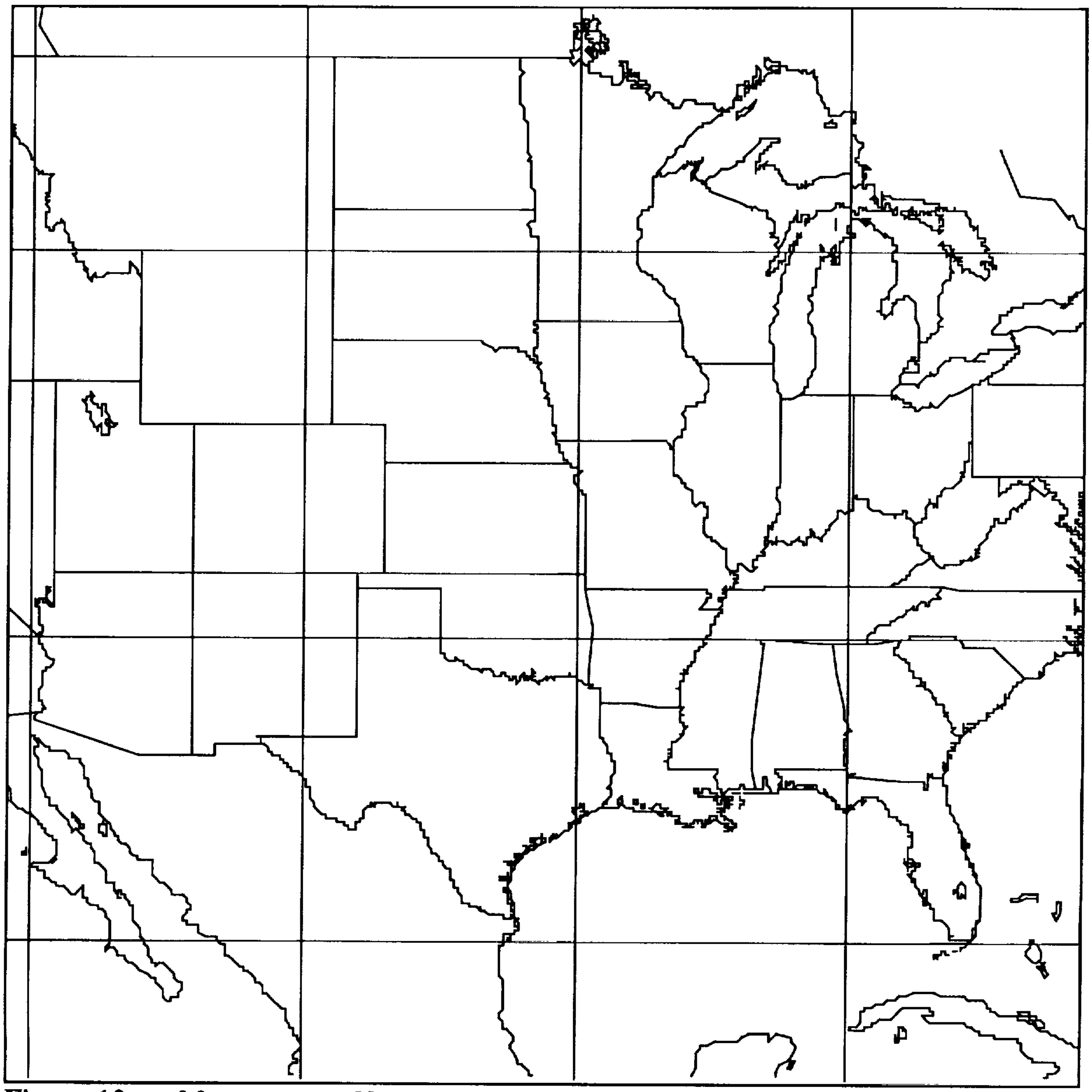


Figure 12. Map: Usa_m
Database: World.bin
Projection: Mercator
Coordinates: 20 50 -130 -60
Grid: 10 10

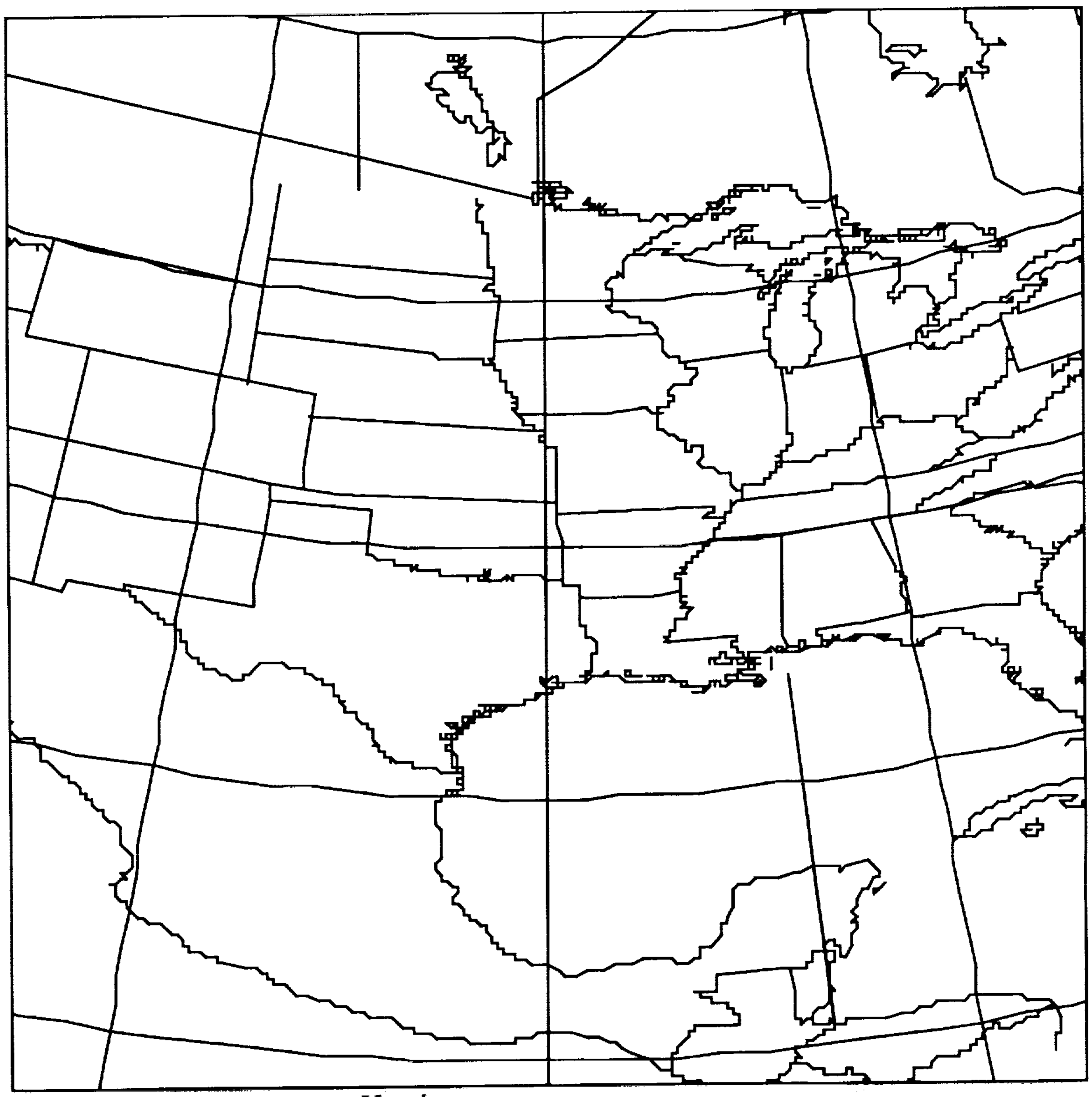


Figure 13.	Map:	Usa_1
	Database:	World.bin
	Projection:	Lambert
	Coordinates:	20 50 -130 -60
	Grid:	10 10

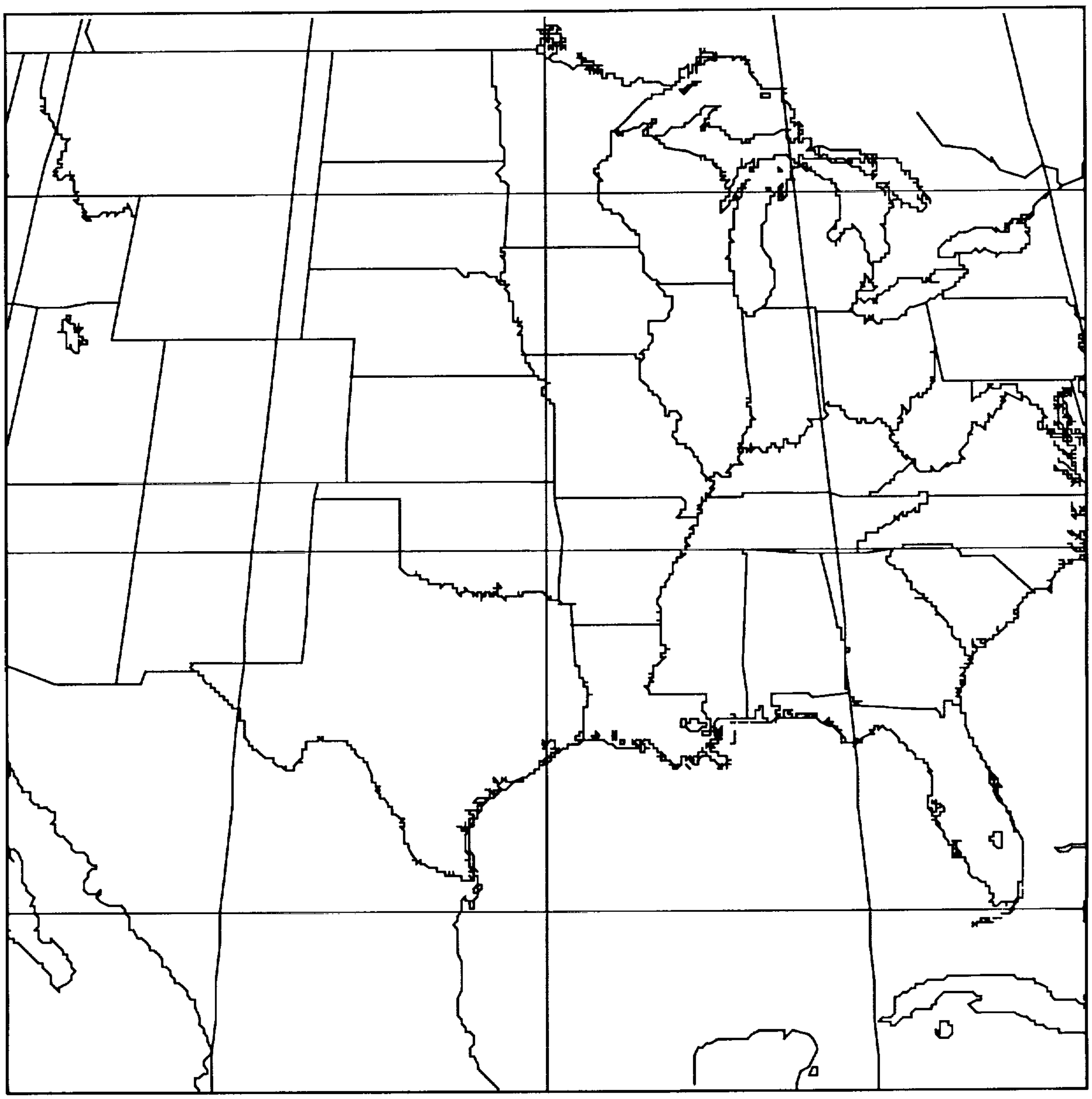


Figure 14. Map: Usa_s
Database: World.bin
Projection: Sinusoidal
Coordinates: 20 50 -130 -60
Grid: 10 10

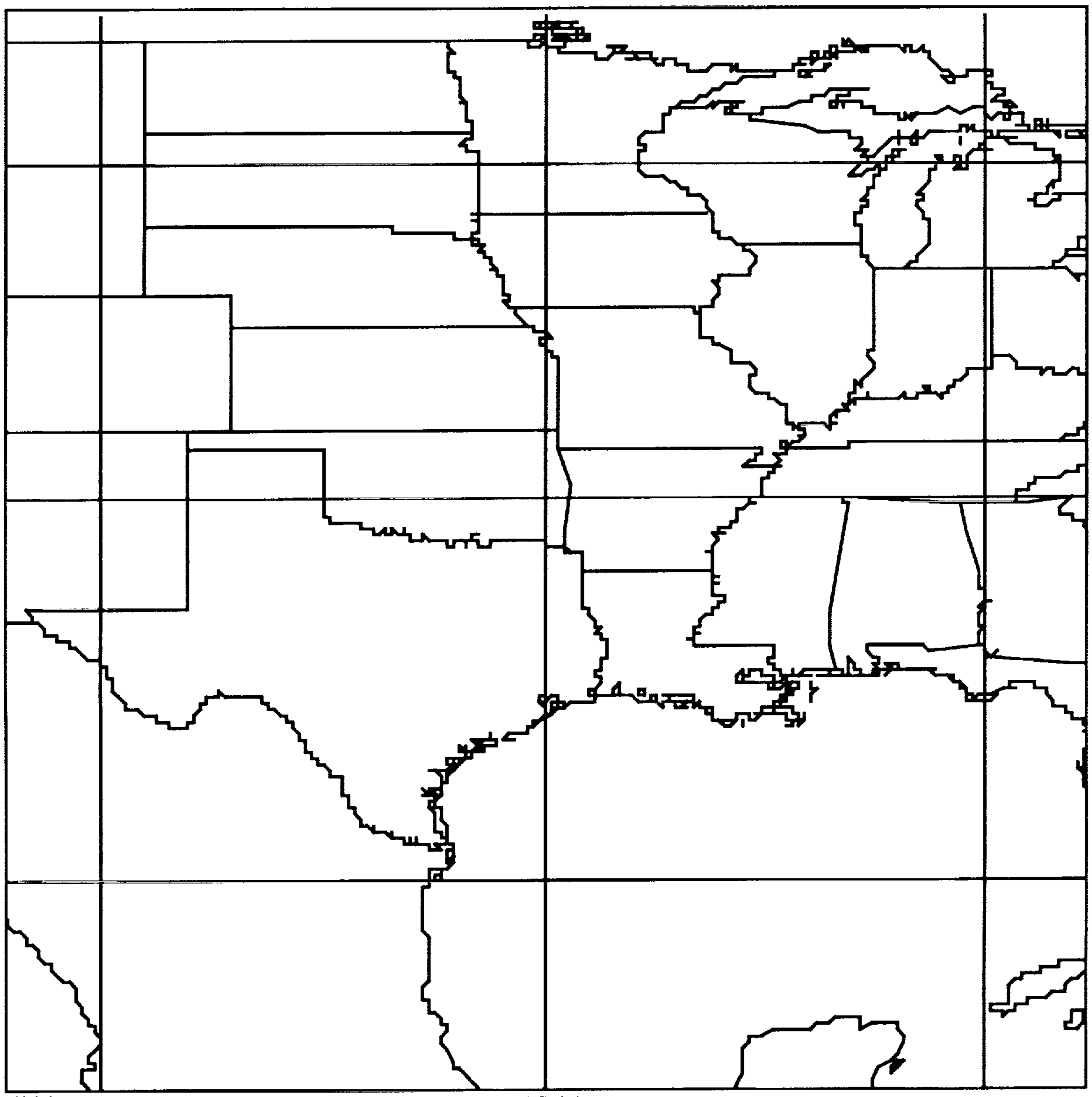


Figure 15. Map: Usa_c
Database: World.bin
Projection: Cylindrical
Coordinates: 20 50 -130 -60
Grid: 10 10

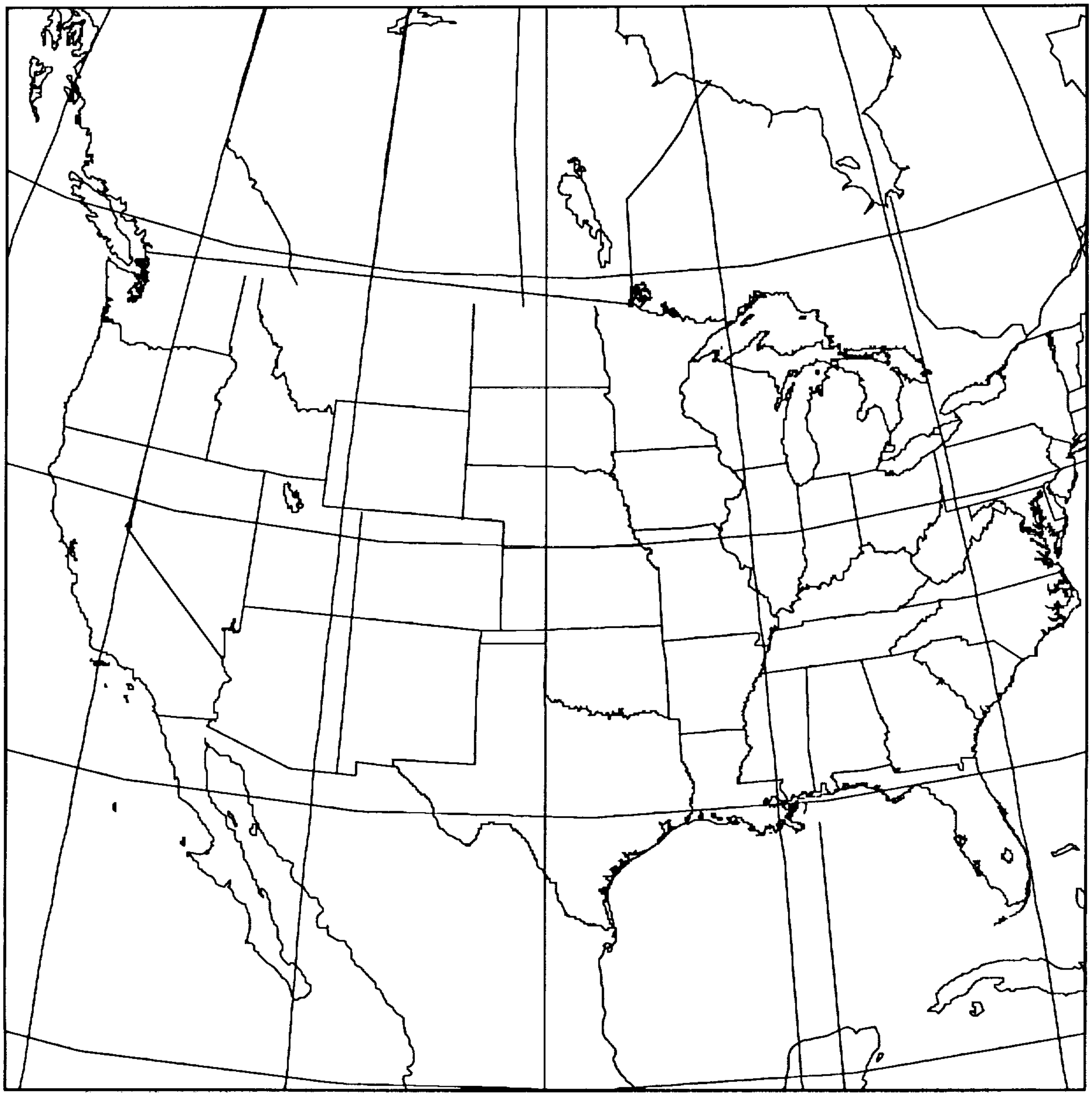


Figure 16.

Map:	Usa_a
Database:	World.bin
Projection:	Azimuthal
Coordinates:	40 -100 1200
Grid:	10 10

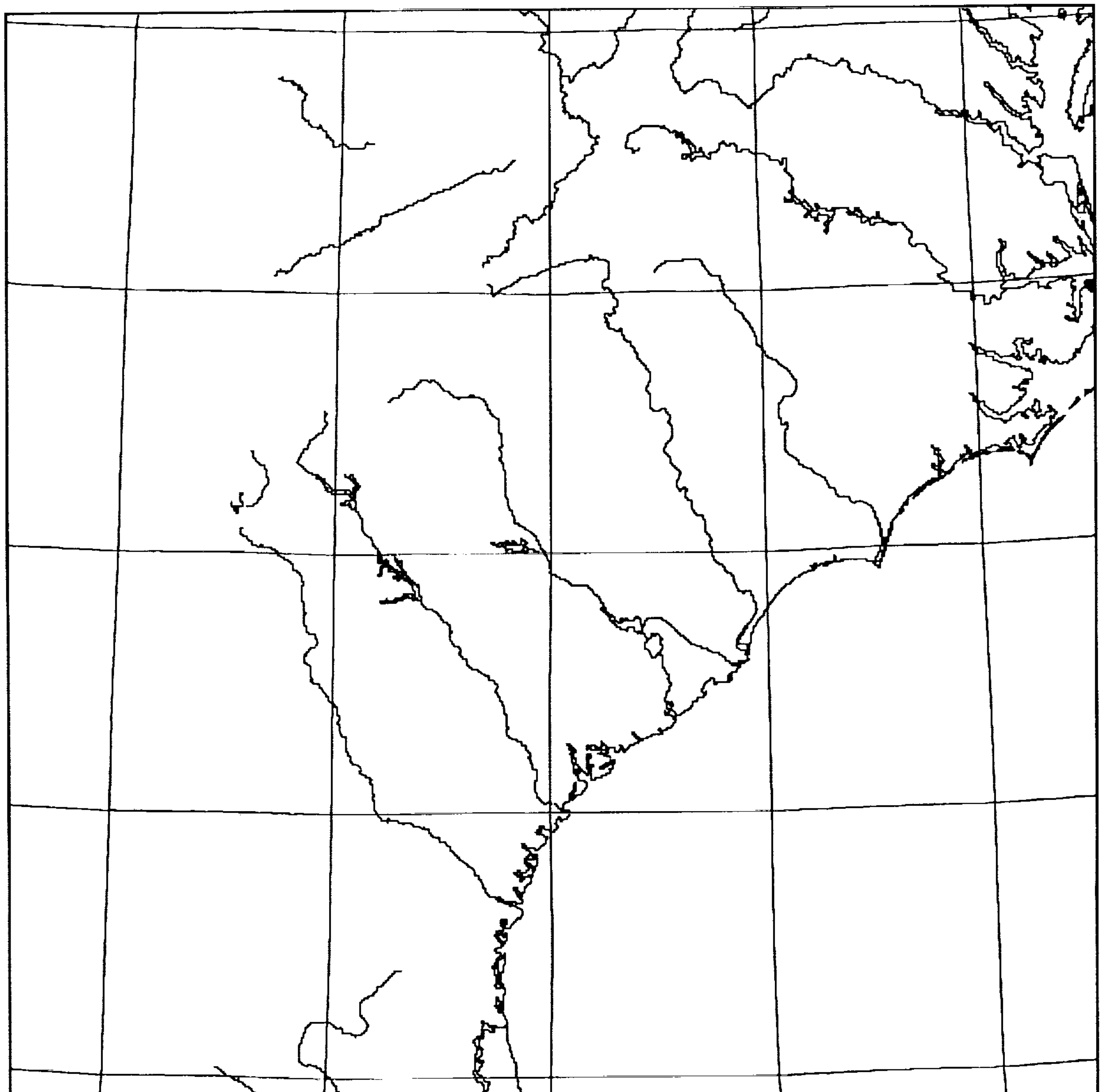


Figure 17. Map: Sc
Database: Ecost_cr.bin
Projection: Azimuthal
Coordinates: 34 -81 250
Grid 2 2

Index

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
February 1993

3. REPORT TYPE AND DATES COVERED
Technical Memorandum

4. TITLE AND SUBTITLE
MAPPER: A Personal Computer Map Projection Tool

5. FUNDING NUMBERS
972

6. AUTHOR(S)
Steven A. Bailey

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Wallops Flight Facility
Wallops Island, Virginia 23337

8. PERFORMING ORGANIZATION REPORT NUMBER
93B00028

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, D.C. 20546–0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA TM-4445

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified–Unlimited
Subject Category 61

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

MAPPER is a set of software tools designed to let users create and manipulate map projections on a personal computer (PC). The capability exists to generate five popular map projections. These include azimuthal, cylindrical, mercator, lambert, and sinusoidal projections. Data for projections are contained in five coordinate databases at various resolutions. MAPPER is managed by a system of pull–down windows. This interface allows the user to intuitively create, view and export maps to other platforms.

14. SUBJECT TERMS
Maps, Personal Computer (PC), Software Documentation

15. NUMBER OF PAGES
40

16. PRICE CODE
A03

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18, 298-102

NASA-Langley, 1993