Subroutine Library PMPLOT

A Geographical Extension to PGPLOT

User's Guide and Reference

Remko Scharroo

Version 9607 – March 5, 2007

Delft Institute for Earth-Oriented Space Research
Delft University of Technology
Kluyverweg 1, 2629 HS Delft
The Netherlands

Contents

1 Introduction **2**
- 1.1 History . . . 2
- 1.2 Projections . . . 3
- 1.3 World Data Bank . . . 5

2 How to use PMPLOT **9**
- 2.1 Example program . . . 9
- 2.2 Program structure . . . 11

3 Subroutine description **11**
- 3.1 PMPLOT routines . . . 11
- 3.2 Stand-alone routines . . . 12
- 3.3 Additional or redefined PGPLOT routines . . . 12
- 3.4 Additional markers . . . 13

4 Subroutine synopsis **15**
- 4.1 GRTCIR – compute coordinates of points on a great circle . . . 15
- 4.2 PGCLIP – Select clipping mode . . . 15
- 4.3 PGNORM - normalize X and Y coordinates . . . 15
- 4.4 PGNUMB – convert a number into a plottable character string . . . 16
- 4.5 PGPT – draw several graph markers . . . 17
- 4.6 PGPTX – draw one or more graph markers at given size and angle . . . 18
- 4.7 PGSVPX – set viewport (any unit) . . . 19
- 4.8 PMBAR – draw scale-bar . . . 20
- 4.9 PMBOX – draw labeled frame around viewport . . . 21
- 4.10 PMCINV – convert X and Y coordinates of several points (inverse). . . . 22
- 4.11 PMCONV – convert X and Y coordinates of several points. . . . 22
- 4.12 PMCVEC – convert coordinates of several vectors. . . . 23
- 4.13 PMCPOL – convert X and Y coordinates of a polygon. . . . 23
- 4.14 PMDEF – set geographical projection type and scale . . . 23
- 4.15 PMQDEF – inquire projection type and scale . . . 25
- 4.16 PMQINF – return general PMPLOT information . . . 25
- 4.17 PMRND – find the smallest "round" number greater than X (in degrees) . . . 25
- 4.18 PMWDB – draw coastlines, rivers, boundaries or land masses. . . . 26
- 4.19 PMSWIN – set geographical projection window and adjust viewport . . . 27
- 4.20 PMX – determine map X coordinate . . . 27
- 4.21 PMY – determine map Y coordinate . . . 27
- 4.22 RNGCIR – compute coordinates of points at equal distance to location . . . 28
- 4.23 SATOBS – compute sub-satellite point from range, elevation and azimuth. . . . 28

1 Introduction

PMPLOT is the Geographical Extension to the PGPLOT subroutine library. It is intended to be used by those that want to include maps in their plots. However, many extra more-or-less plain PGPLOT subroutines have been added or changed. Also additional Hershy fonts are added to the font file that are only excessable through PMPLOT.

The PMPLOT library has been ported to various systems. Presently it is only available on the IBM Risk 6000 workstations of the Section Space Research and Technology. The library can be linked to any of your FORTRAN programs by specifying a linkage `-lpmplot`, *e.g.*

```
xlf example.f -o example -lpmplot
```

The PMPLOT library is courtesy of Remko Scharroo, Delft University of Technology, Section Space Research & Technology (E-mail: `Remko.Scharroo@lr.tudelft.nl`)

1.1 History

Since creation the following changes have been made to the software:

- Version 9101
  - Another 7 projections were added to PGGPLOT;
  - PGGENV, PGGXINV, and PGGYINV were removed from the library;
  - PGGX and PGGY are devised to also support non-linear projections;
  - PGGCINV was added as a new routine taking care of the inverse coordinate conversion;
  - RNGCIR was created to compute range circles; GRTCIR computes great circles;
  - PGGWDB has been changed (also the input) to improve flexibility.
  - The syntax of PGGBOX and PGGDEF have been changed slightly in order to support azimuthal and conical projections.
- Version 9102
  - PGGCVEC is a new routine that converts the position coordinates and azimuth of a vector to map coordinates and angle.
  - PGMARK is created to facilitate the drawing of several markers in different sizes and at different tilts in one call.
  - Markers 0858 and 0859 are added to the font library.
- Version 9103
  - Markers 0791 through 0795 are added to the font library.
  - Mountains and bathymetry has been added to the World Data Bank.
  - The definition of PGGWDB has been changed slightly in order to support the mountain and bathymetry files. This change is important for those who want to plot all ranks.
- Version 9104
  - PGGPLOT has been removed from dutlru4. However it works properly on both dutlru2 and dutlru4.
  - NEW: PGGPLOT is now also available on the Convex.
  - All WDB files have been compressed. Again the PGGWDB call has been altered. Read the manual carefully.
- Version 9105
  - Again 3 projection have been added to the PGGPLOT library: Mollweide projection, Gall's stereographic projection, and Bartholomew's 'The Times' projection.
- Version 9200
  - PGGPLOT is renamed to PMPLOT. Thus all routines starting with PGG, start now with PM.
- Version 9201
  - Land fill in PMWDB included. See description.
  - Extra PG-routine PGVP.
  - Extra fonts added to the font library to plot oceanic platform and SLR systems. This meant extending the size of the library.
- Version 9300
  - Extra projection: tilted rectangular (type 34).

  - Plot in Color PostScript mode included. Devices /CPS and /CVPS. This has effect for all plotting routines, like PGSCI and PGPIXL.
- Version 9304.x
  - PostScript device standardised to one source. `BoundingBox` is determined and rewritten at the top of the PostScript file. Also generating command is included in the file.
  - PMQINF routine is introduced.
  - First manual typeset in LaTeX.
  - (9304.1) Allow 255 grey shades and setting of grey shade with PGSCR.
  - (9304.2) Include new options in PMBOX to give nice degree-minutes notation. PGNUMB has changed as well.
- Version 9306.x
  - Small unnoticeable changes.
  - FASTIO incorporated in PMWDB. Much faster.
- Version 9308
  - Fill area and pixel dump routines improved.
  - PPM and VPPM device types introduced.
- Version 9310.x
  - Plate boundary data set and supporting change to PMWDB implemented.
  - Clipping of symbols or lines switchable.
- Version 9404
  - Small changes to PSDRIV to improve transparent use of PGSCR
- Version 9601
  - Bug removed from PMBOX (wrong major ticks in projections not equal to 1)
  - Alignment with PGPLOT 5.0. PGMARK renamed to PGPTX, PGVP to PGSVPX.
- Version 9607
  - Aligned with PGPLOT 5.1.
  - PMWINDOW renamed to PMSWIN. PMPOLY renamed to PMCPOL, and most of its functionality transferred to PMCONV.

1.2 Projections

PMPLOT routines take care of scaling and transformation of geographical data under certain projections. The present version (9300) supports fifteen types of projections.

- Cylindrical projections:
  1. Equi-Rectangular;
  2. Peters;
  3. Mercator;
  4. Miller;
  5. Gall's stereographic projection.
- Azimuthal projections:
  11. Orthographic;
  12. Perspective;
  13. Azimuthal Equal-Area;
  14. Azimuthal Equi-Distant.
- Conic projections:
  21. Ptolemy Conic Equal-Interval;
  22. Kavraiskiy IV Conic Equal-Interval.
- Miscellaneous projections:
  31. Sinusoidal or Mercator Equal-Area;
  32. Mollweide projection;
  33. Bartholomew's 'The Times' projection;

34. Tilted rectangular projection.

- Polar projections:

  41. Polar projection centred on North Pole;

  42. Polar projection centred on South Pole;

Equi-rectangular projection (1)

The Equi-Rectangular Projection, often referred to as the plane chart, is the only projection that is linear in both directions, i.e. tick marks along the x-axis are spaced equally, as well as along the y-axis. The ratio of the spacing between meridians on the one hand and parallels on the other depends on the latitude of the true-scale parallel. If the scale must be 'true' at the equator, the ratio is 1 to 1 (even if the equator is not in the map). This projection is most used for large-scale maps such as city plans. Hand plotting over this type of map is very easy.

Peters projection (2)

This non-linear projection does not preserve distances, nor shapes, but preserves areas. Thus areas in the map are at scale; one square millimeter in the map is the same number of square kilometers in reality anywhere in the map. Areas further form the true-scale latitude become strongly distorted.

Mercator projection (3)

The spacing of parallels on the Mercator Projection increases progressively poleward from the equator in a way that makes the projection conformal, i.e. by increasing the North-South scale to exactly the East-West scale at every latitude. However, the scale is not the same at each latitude. The conformality means that any straight line on the Mercator Projection describes a constant compass course, which makes the projection very useful for navigation purposes. The user should be aware that distances and areas are seriously exaggerated at high latitudes. In fact, the parallels are mapped so far apart at the extreme latitudes that the map should then be infinitely large. We speak of a singularity at the poles. Therefore, maps of this type are restricted to 89 degrees North and 89 degrees South.

Miller projection (4)

This is in fact an imperfect Mercator projection. It is neither conformal nor equal-area and is limited to small-scale maps. Fortunately the Miller projection is not singular at the poles and looks very much like the Mercator projection at lower latitudes.

Gall's stereographic projection (5)

This projection is a stereographic projection from an antipodal point on the equator, on to a cylinder which cuts the Earth at 45 degrees North and 45 degrees South. It's easy to construct and has been widely used for world maps including those showing distribution data. The projection is neither conformal nor equal-area. Its principal merit is that it reduces greatly the distortion at higher latitudes as in the Mercator projection.

Orthographic projection (11)

The Orthographic Projection presents the globe as viewed from infinity, centered at any point on the globe. This is in fact a special case of the Perspective Projection, where the observer is infinitely far. The scale of the Orthographic Projection (as all other azimuthal projections) is true at the center of the 'map'. This projection is very useful for presentations of entire hemispheres.

Perspective projection (12)

Unlike the Orthographic with the observer at 'infinity', the Perspective focus is from a point in near rather than deep space. One gets the spherical illusion of viewing from a high satellite. Its primary use is for orientation with the map centered at on the area of interest. In addition this gives the opportunity to plot areas less than a hemisphere in Perspective.

Azimuthal equal-area projection (13)

This projection, invented by Lambert, is best suited to small-scale maps of continental areas or hemispheres. The equivalent property is especially valuable for depicting distributions or measuring defined areas. The azimuths from the center point are correct and all straight lines from the center are great circles.

Azimuthal equi-distant projection (14)

The Azimuthal Equal-Distant projection is employed when the distance and azimuth are required from a central point to any other point. A popular application is plotting a hemisphere or even the entire earth around one radio station.

Ptolemy projection (21)

This conic equal-interval projection was invented by Claudius Ptolemy in the 2nd century B.C.. It has a single standard parallel and equally spaced parallels and possesses the virtues of being easy to manually construct or plot on.

Kavraiskiy IV projection (22)

The Kavraiskiy IV Projection with two standard parallels is the result of Russian efforts to develop a projection "with the least mean square linear distortion" for the area of the Soviet Union. Because it displays areas of large longitudinal extent so well, it is the choice for most maps of the Soviet Union. The scale of the map is true at both standard parallels. Between the standard parallels distances are slightly underestimated. Outside the standard parallels distances are slightly exaggerated.

Sinusoidal projection (31)

The Sinusoidal or Mercator Equal-Area Projection was created to reduce the distortions of shape present in the cylindrical Mercator Projection. It is an excellent choice for maps with significant North-South dimensions, such as hemispheres and continents. The equally spaced parallels and true meridional divisions are also convenient for manual plotting of data. This projection has not one standard parallels. Any parallel has a true scale.

Mollweide projection projection (32)

In this equal-area projection the central meridian is a straight line at right angles to the equator and all other parallels, all of which are straight lines subdivided equally. The spacing of the parallels are derived mathematically from the fact that the meridians 90 East and West of the central meridian form a circle equal in area to a hemisphere.

Bartholomew's 'The Times' projection (33)

This projection was designed to reduce the distortions in area and shape which are inherent in cylindrical projections, whilst, at the same time, achieving an approximately rectangular shape overall. It falls in the category of pseudo-cylindrical. Parallels are projected stereographically as in Gall's projection. The meridians are less curved than the sine curves of the Sinusiodal projection. Scale in preserved at 45 degrees North and South (always).

Tilted rectangular projection (34)

This is a rather odd projection that allows you to plot a map in 3D, like it is viewed from the southern boundary of the map. As parameters one can define the vertical scaling and a horizontal tilt of the map. It can be used to show vertical displacements.

1.3 World Data Bank

The World Data Bank is a digital representation of the World's coastlines, islands, lakes, boundaries, and rivers. To plot these items the subroutine PMWDB was designed. The World Data Bank (WDB) comes in three levels of detail:

- WDB 0 represents the coastlines and major islands and lakes in 446 lines of a total of 8512 points. Because of the lack of detail WDB 0 is only useful for small scale maps (of hemispheres or the entire earth). The resolution is 1 arc-minute (about 2 km). In addition to this a data set `0.plt` containing plate boundaries is introduced.
- WDB 1 is divided into of two data sets. One consists of 87457 points along 974 line segments representing the coastlines, islands and lakes of the world in much more detail than WDB 0. It is thus especially appropriate for larger scale maps of continents. The second data set incorporates the international boundaries of the world (300 line segments of 32258 points in total). The resolution of both data sets is 6 arc-seconds (about 200 m).
- Since Version 9200 the WDB 1 data set also has a version to plot landmasses to be plotted as filled polygons. The data set is called `1.lnd`. Like any other WDB data set it can be plotted with the PMWDB subroutine.
- WDB 2 was digitised at scales of approximately 1:3000000, and should therefore preferably be used at these large scales. The WDB 2 is divided into 5 groups of 3 or 4 data sets. Each group is associated with one geographic area: 1) AFR: Africa and the Middle East; 2) ASI: Asia incl. Russia, Australia, New Zealand and Oceania; 3) EUR: Europe; 4) NAM: North America and Greenland; 5) SAM: South America, Middle America and Antarctica. The 3 or 4 files in each group refer to each a different type of data: a) BDY: international boundaries; b) CIL: coastlines, islands and lakes; c) RIV: rivers; d) PBY: US state borders and Canadian province boundaries (only for NAM). All these standard WDB data sets have a resolution of 15 m.
- An extension to WDB contains altitude contours for both land and ocean bottom and was converted from the ETOPO5 topography data set. The altitude contours on land (MNT) are separated by 500 meter, starting at 500 meter altitude. This part of the WDB is also separated into 5 data sets, one for each geographical region as specified above.

  The depth contours for the sea bottom (or bathymetry, BTH) start at sea level (0 meter) and run up to -10000 meter, each contour separated by 500 meter. Again there are 5 data sets: 1) IND: Indian Ocean; 2) NAT: North Atlantic; 3) NPA: North Pacific; 4) SAT: South Atlantic; 5) SPA: South Pacific. The resolution of the contours is 500 meters in altitude (as said before) and 10 arc seconds (about 300 m) in horizontal direction, based on a grid with a 10-arc-minute resolution.

Because of their superiority in detail, the WDB 2 data sets are considerably larger than each individual WDB 0 or 1 file.

All lines in each file are attached a rank number. These ranks are hierarchically structured, and are useful for output plotting symbol or line style definition. Depending on the type of data the rank numbers have a different meaning, as described in Table 1.

	BDY		*PBY*
1.	Demarcated or delimited.	1.	First order admin.
2.	Indefinite or in dispute.		
3.	Other lines of separation.		
	CIL		*RIV*
1.	Coasts, major islands and lakes.	1.	Permanent major rivers.
2.	Additional major isles and lakes.	2.	Additional major rivers.
3.	Intermediated islands and lakes.	3.	Additional rivers.
4.	Minor islands and lakes.	4.	Minor rivers.
		5.	Double-lined rivers.
6.	Intermittent major lakes.	6.	Major intermittent rivers.
7.	Intermittent minor lakes.	7.	Additional intermittent rivers.
8.	Reefs.	8.	Minor intermittent rivers.
9.	Major salt pans.		
10.	Minor salt pans.	10.	Major canal.
		11.	Canals of lesser importance.
		12.	Irrigation type canals.
13.	Major ice shelves.		
14.	Minor ice shelves.		
15.	Glaciers.		
	MNT		*BTH*
$\geq$0.	Altitude in meters.	$\leq$0.	Depth in meters (negative).
	LND		
1.	Land.		
2.	Lakes or inner seas.		

Table 1. Meaning of the rank numbers in the WDB files.

To plot lines use PMWDB. The range of ranks can be specified in your call to PMWDB along with the name of the data set. It is not required to give the directory name of the WDB data sets, provided that they are stored in the directory specified by the `WDB_DIR` environment variable. The names of these files are:

```
0               0.plt           0.lnd

1.bdy           1.cil           1.lnd

2.afr.bdy       2.asi.bdy       2.eur.bdy       2.nam.bdy       2.sam.bdy
2.afr.cil       2.asi.cil       2.eur.cil       2.nam.cil       2.sam.cil
2.afr.riv       2.asi.riv       2.eur.riv       2.nam.riv       2.sam.riv
2.afr.mnt       2.asi.mnt       2.eur.mnt       2.nam.mnt       2.sam.mnt
                                                2.nam.pby

2.ind.bth       2.nat.bth       2.npa.bth       2.sat.bth       2.spa.bth
```

Although it would be of no interest to most users, we give here a short summary of the WDB format. Each data set mentioned above consists actually of two files: one table (with extension `.TAB`) and one data file (with extension `.DAT`). The table (a direct access binary file) has one entry for each individual line segment and contains information about the rank of the segment and the minimum and maximum longitude and latitude of the points in that segment. Also one field points to the record of the first of a number of points in that segment, points that are stored in the data file (a direct access binary file). The formats are given in Tables 2 through 4.

The longitudes as stored in the WDB are confined to the range –180 (180W) to +180 (180E), whereas the latitudes do not exceed, of cause, –90 (90S) or +90 (90N). Nevertheless the PMWDB routine can manage longitudes in any other range (even spanning the world more than once).

If you use the land-fill data sets (`0.lnd` or `1.lnd`) the land masses and lakes will be filled in two different colors (or shades). In stead of rank numbers you will have to give the color indices for plotting the land and lakes.

TABLE FILE HEADER		
Item	*Type*	*Explanation*
1	A4	File specifier (`@WDB` or `@LND`).
2	I4	Resolution (units per degree).
TABLE RECORDS		
Item	*Type*	*Explanation*
1	I4	Rank.
2	I4	Number of points in the segment.
3	I4	Record number of the first point.
4	I4	Minimum longitude (in units of resolution).
5	I4	Maximum longitude (in units of resolution).
6	I4	Minimum latitude (in units of resolution).
7	I4	Maximum latitude (in units of resolution).
8	I4	Start longitude (in units of resolution).
9	I4	Start latitude (in units of resolution).
The last record in the file contains all zeros.		

Table 2. WDB table file format (all types)

DATA FILE HEADER		
Item	*Type*	*Explanation*
1	A2	File specifier (`@@`).
DATA RECORDS		
Item	*Type*	*Explanation*
1	A1	Longitude increment (in units of resolution) + 127, stored as a character.
2	A1	Latitude increment (in units of resolution) +127, stored as a character.

Table 3. WDB data file format (.cil .bdy .pby .riv .mnt .bth)

DATA FILE HEADER		
Item	*Type*	*Explanation*
1	A4	File specifier (`@@LN`).
DATA RECORDS		
Item	*Type*	*Explanation*
1	I2	Longitude increment (in units of resolution).
2	I2	Latitude increment (in units of resolution).

Table 4. WDB data file format (.lnd only)

2 How to use PMPLOT

PMPLOT is an extension to the PGPLOT subroutine library. Like PGPLOT, PMPLOT consists mainly of FORTRAN-coded subroutines. Originally, PMPLOT was designed primarily to include maps in plots generated with the PGPLOT library. Presently, PMPLOT serves more purposes: it introduced Color PostScript to the plotting devices, extended the font set, and included new general-purpose routines or modified some of the standard PGPLOT routines.

The following Sections describe the usage of PMPLOT routines in conjunction with the standard PGPLOT routines.

2.1 Example program

The following program is an example FORTRAN program that uses some of the PMPLOT routines. We will discuss some of the calls made in this program. To emphasise the PMPLOT routines, they are capitalised in the source code.

```
* file: example.f
*
      program example
      real x(2),y(2),a(100),b(100),scale
      character*80 text

      call getarg(1,text)
      if (text.eq.' ') text='?'
      call pgopen(text)
      call pgvstd
*
* Select projection and window (conic, Europe).
*
      call PMDEF(22,0.,100.,100.)
      call PMSWIN(-10.,40.,30.,75.)
*
* Fill land.
* Draw coastlines (thick) and boundaries (thin, dashed) (WDB 1; all ranks)
*
      call pgscr(2,0.75,0.75,0.75)
      call PMWDB('1.lnd',2,0)
      call pgslw(2)
      call PMWDB('1.cil',1,0)
      call pgslw(1)
      call pgsls(2)
      call PMWDB('1.bdy',1,0)
      call pgsls(1)
*
* Draw two markers (Madrid and Moscow) and connect them by a great circle
*
      x(1)=-3.74
      y(1)=40.48
      x(2)=37.4
      y(2)=56.0
      call grtcir(x(1),y(1),x(2),y(2),100,a,b)
      call PMCONV(100,a,b)
      call pgline(100,a,b)
      a(1)=90.0
      a(2)=90.0
      call PMCVEC(2,x,y,a)
      call pgpt(2,x,y,850)
      call pgptxt(x(1),y(1),a(1),1.2,'Madrid')
      call pgptxt(x(2),y(2),a(2),-.2,'Moscow')
*
```

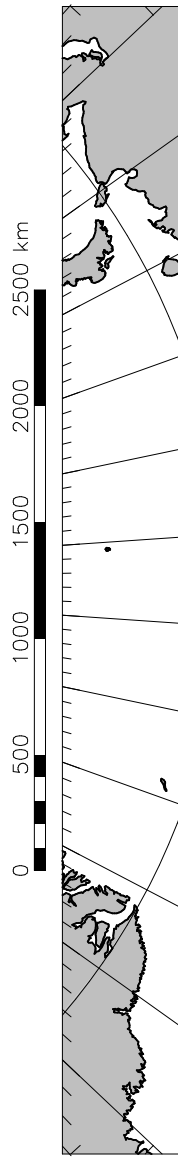

Figure 1. Output of `example.f`.

```
* Draw meridians and parallels.
* Draw scale bar, etc.
*
      call PMBOX('BCNGST',10.,5,'BCNGST',10.,5)
      call pgsch(.6)
      call PMBAR('CT',1.,0.,5,4)
      call PMQDEF(text,scale)
      call pgmtxt('R',1.,0.,0.,text)
      call pgend
      end
```

This program can be compiled by executing the command

```
xlf example.f -o example -lpmplot
```

And the output should look like displayed in Figure 1.

2.2 Program structure

Before using any of the other PMPLOT routine, PMDEF must be called to define the projection type and scale. This call can be followed by calls to other PMPLOT procedures to define the size and shape of viewport and the map boundaries. Furthermore all usual PGPLOT routines can be used as defined in the PGPLOT manual. One has to keep in mind, however, that the map (x,y) coordinates do not have to be the same as the true (longitude,latitude) coordinates. To plot lines and points with, for instance, PGLINE or PGPOINT one has to convert the true world coordinates (longitude and latitude in degrees) to map coordinates that are linearly mapped into viewport. One is adviced to use PMCONV for this purpose. Only in case of the Equi-Rectangular Projection, which is linear in both direction, the use of PMCONV is optional since then true world and map coordinates are identical. Referring to the Subroutine Synopsis below for more detailed description of the routine inputs and outputs, the setup of a map could be lined out as follows:

- Open device with PGBEG or PGOPEN, as usual.
- Open a viewport with PGVSTD, PGVSIZ or PGSVP (if necessary, preceded by PGPAGE to advance the page).
- Define the projection type and scale with PMDEF. Additionally, include the projection parameters.
- Define the map boundaries with PMSWIN. The viewport size and shape will automatically be adjusted conform the scale and projection type define in PMDEF. If no scale is defined, the map will be the largest that can fit within the original viewport. For some projections the mapped area will be larger than the one defined, because of the fact that the area will not be mapped rectangularly. However, the map is the smallest that can fit around the defined area.
- Call PMWDB to draw coastlines, lakes, rivers, and/or borders.
- PGCONV converts the latitude and longitude of two points to map coordinates. Afterwards the standard PGPLOT routines PGPOINT and PGPTXT can be used to plot markers and text.
- RNGCIR and GRTCIR can be used to compute range and great circles. A sequence of PMCONV and PGLINE can be called afterwards to draw the circles.
- Use PMBOX to draw meridians, parallels, ticks, annotations, etc.
- Finally, PMBAR plots a scale bar above the map.

As long as no other call to PMDEF is made, the projection type and scale remain unchanged. If the scale was not set in PMDEF (i.e. the parameter `SCALE` was set to `0.0`) the actual scale of your map will be computed by PMSWIN, along with the reshaping of the viewport, such that it exactly encompasses the area to be mapped. Once set or computed, the scale will not be altered, even if you call PMSWIN again.

As far as the map boundaries are concerned, they must be, as said before, defined in a call to PMSWIN. If the projection type is Cylindrical, the boundaries entered by the user directly define the edges of the window. On the other hand, if a projection is chosen that is not Cylindrical, such as the Conic ones where the meridians are tilted and the parallels are curved, the area boundaries as mapped in your chart do not coincide with the edges of the window. Therefore PMSWIN, by itself, slightly adjusts the area boundaries, such that (1) the window edges touch all corners of the mapped original area and is thus the smallest that can fit around the area, and (2) the entire window is covered by the new area. In other words, the area defined by the new boundaries is the smallest that can fit around the rectangular window.

In case of Azimuthal projections, which are essentially meant to be used to portrait the whole world, or at least a hemisphere, the boundaries are automatically set to –180 through +180 degrees longitude and –90 through +90 degrees latitude.

3 Subroutine description

3.1 PMPLOT routines

Apart from the abovementioned PMPLOT routines PMDEF, PMSWIN, and PMWDB, there are a couple of routines in conjunction with them to make maps. These routines can only be used *after* calling PMDEF.

PMDEF This routine has to be called before any other PMPLOT routine. The routine has four parameters: the projection type, as described in Section 1.2; scale; and two projection parameters. By giving a scale of '0.0', the plot will be scaled automatically. The projection parameters usually refer to the 'true scale parallels' (latitudes at which the plot is at true scale both in longitudinal and latitudinal direction).

PMQDEF This routine queries the scale and projection type.

PMQINF Queries more information on the PMPLOT version and projection parameters.

PMSWIN Defines the map boundaries. The viewport size and shape will automatically be adjusted conform the scale and projection type define in PMDEF. If no scale is defined, the map will be the largest that can fit within the original viewport. For some projections the mapped area will be larger than the one defined, because of the fact that the area will not be mapped rectangularly. However, the map is the smallest that can fit around the defined area.

PMWDB is a routine that is especially designed to plot maps of coastlines, islands, lakes, rivers, state and national boundaries, bathymetry, and mountains as lines, or filled-in land masses. The usage is extensively described in Section 1.3.

PMCONV and PMCINV These routines convert true world (longitude,latitude) coordinates to map (x,y) coordinates (PMCONV) or vice versa (PMCINV). These routines convert the coordinates of an indefinite number of points in one call.

PMCVEC and PGPTX In order to be able to draw vectors of different size and azimuth in a geographical projection PMCVEC and PGPTX can be used. First PMCVEC converts the real-world coordinates (longitude,latitude) and the given azimuth of the direction of the vector to map coordinates (x,y) and a tilt. This tilt is measured from the +x axis. The second step is to call PGPTX (which is and is in fact NOT a PMPLOT routine, but an extension to the regular PGPLOT routines), with a proper symbol. Symbol numbers 0858 (an arrow with a small head pointing to the right) and 0859 (a wind vane) are especially useful for this purpose. If you want to draw a legend, use e.g. PGMTXT to plot the string `'\(0858) = 30 m/s'`.

PMX is a function that returns the plot coordinate X for given longitude and plot coordinate Y.

PMY is a function that returns the plot coordinate Y for given latitude and plot coordinate X.

PMBOX This routine is very similar to PGBOX, however, PMBOX is also able to draw tilted or curved meridians or parallels and not-equally spaced ticks and annotations. This routine, rather than PGBOX, should be used, after PMSWIN, to draw these items. PMBOX has all kinds of special options to give a nice annotation and plot curved grid lines.

PMBAR With this routine a scale bar can be drawn at the top or the bottom of the window, either aligned with the left or the right of the viewport or centered.

PMRND Finds a suitable 'round' number just above a value given by the user.

3.2 Stand-alone routines

To facilitate plotting of some generally used items as range circles and great circles, two routines have been generated to compute the coordinates of points along these lines. However, neither of the routines produce graphics. For instance, one can combine calls to RNGCIR, PMCONV and PGLINE to draw a range circle.

RNGCIR Computes the coordinates of points at a specified range and azimuth range from a point. This routine does not draw anything, nor does it call PGPLOT routines or is it used explicitly with PMPLOT; it simply computes, nothing else.

GRTCIR Computes the points along a great circle between to locations on the earth. Especially useful to draw the shortest route between these points. As RNGCIR, this is a stand-alone subroutine.

SATOBS Computes the coordinates of the sub-satellite point from range, elevation, and azimuth from an earthly observer.

LENCHR Function that returns the number of characters in a string, excluding trailing spaces.

3.3 Additional or redefined PGPLOT routines

Some routines have been incorporate in the PMPLOT library that are actually an extension to PGPLOT. Consequently, they can be used without any call to PMDEF, or any other PMPLOT routine. Most of them are made to improve the PGPLOT plotting facilities.

PGSVPX can be used to set the viewport in any preferred units (absolute device coordinates, normalised device coordinates, inches, or centimeters), unlike PGSVP, that only allows normalised device coordinates.

PGCLIP allows you to regulate the clipping of lines or symbols. Usually, lines are always clipped at the area boundaries and symbols are only drawn when the center is within the viewport, and are not clipped at the boundaries. With PGCLIP you can specify several stages of clipping of lines and symbols.

PGPIXL This routine was created by Ge van Geldorp to make pixel dumps of arrays. In the specially adjusted version, PGPIXL is able to make pixel dumps to any device, including PostScript. For screen devices, colors are generated according to the color map that is set with PGSCR. The same works now also for a color PostScript device /CPS and /CVPS. A grey map is generated on regular PostScript devices.

PGPTX As described above, PGPTX draws marks with various magnifications and at various tilts.

PGNORM Normalises a vector.

3.4 Additional markers

In order to facilitate vector plotting two markers were added to the PGPLOT font library (See Figure 2):

0858 An arrow with a small head pointing to the right, and centered in the middle of the horizontal line.

0859 A wind vane pointing to the right and centered at left edge of the horizontal line.

Another five additional fonts were generated with the intention to be used for notations. The five markers (0791 through 0795) represent line styles 1 through 5 as defined in PGSLS. Use for instance `'\(0793) velocity'` as text in your call to PGTEXT, PGPTXT or PGMTXT to indicate that the dash-dot-dash-dot line represents velocity.

0791-0795 Line-styles 1 (full line), 2 (dashed line), 3 (dash-dot line), 4 (dotted line), and 5 (dash-dot-dot-dot-dash line).

Satellite Laser Ranging systems can be plotted with the markers 0879 through 0891. Since markers 0881–0890 have no width, you can also combine an SLR system as a string `'\bs(0879)\bs(0881)\bs(0886)\bs(0891)'`.

0879 Fixed SLR bed.

0880 Mobile SLR system.

0881–0885 SLR objective in five angles.

0886–0890 Camera in five angles, to be 'mounted' on characters 0881–0885.

0891 Reference point or center-of-mass. It completes a sequence of characters 0879–0880, 0881–0890, and 0891, when used as a string.

Also two markers are designed to draw an ocean platform. When fitted on top of eachother, characters 0892 and 0893 form a tower at sea.

0892 Ocean platform, including submarine structure.

0893 Water level.

Figure 2. Some Hershy fonts defined in PMPLOT.

4 Subroutine synopsis

4.1 GRTCIR – compute coordinates of points on a great circle

```
    SUBROUTINE GRTCIR (LON1, LAT1, LON2, LAT2, N, X, Y)
    INTEGER N
    REAL LON1, LAT1, LON2, LAT2, X(N), Y(N)

Compute the coordinates of a number of points on a great circle through
two points.

Arguments:
 LON1   (input) : Longitude (degrees) of the first point.
 LAT1   (input) : Latitude (degrees) of the first point.
 LON2   (input) : Longitude (degrees) of the second point.
 LAT2   (input) : Latitude (degrees) of the second point.
 N      (input) : Number of points on the great circle. (N should be greater
                  than 1)
 X     (output) : Array of longitude coordinates (degrees) of points
                  1 through N.
 Y     (output) : Array of latitude coordinates (degrees) of points
                  1 through N.
```

4.2 PGCLIP – Select clipping mode

```
    SUBROUTINE PGCLIP (MODE)
    INTEGER MODE

Select whether symbols or line should be clipped. Clipping is switched
on or off with the MODE parameter (see below).

Argument:
MODE (input) : = 0 : Clip all graphics.
                     Draw symbols completely, but only if the center is
                     inside the viewport. (DEFAULT)
               = 1 : Clip all graphics and all symbols.
               = 2 : Clip all graphics.
                     Let symbols be drawn completely, also when they are
                     outside the viewport.
               = 3 : Do not clip any graphics nor symbols.
```

4.3 PGNORM - normalize X and Y coordinates

```
    SUBROUTINE PGNORM (R, X, Y)
    REAL R, X, Y

This routine normalizes the X and Y coordinates of a vector (X,Y),
such that it obtains a length R.

Arguments:
 R           (input): New length of the vector.
 X, Y (input/output): Coordinates of the vector.
```

4.4 PGNUMB – convert a number into a plottable character string

```
    SUBROUTINE PGNUMB (MM, PP, FORM, STRING, NC)
    INTEGER MM, PP, FORM
    CHARACTER*(*) STRING
    INTEGER NC
```

```
This routine converts a number into a decimal character
representation. To avoid problems of floating-point roundoff, the
number must be provided as an integer (MM) multiplied by a power of 10
(10**PP).  The output string retains only significant digits of MM,
and will be in either integer format (123), decimal format (0.0123),
or exponential format (1.23x10**5). Standard escape sequences \u, \d
raise the exponent and \x is used for the multiplication sign.
This routine is used by PGBOX to create numeric labels for a plot.

Specially for degree-minute notation options FORM=3 and FORM=4 are added.
These will return a string in the format <degrees>\u<minutes>\d.
Option FORM=5 and FORM=6 allow <hour>:<min> and <hour>:<min>:<sec>

Formatting rules:
  (a) Decimal notation (FORM=1):
      - Trailing zeros to the right of the decimal sign are
        omitted
      - The decimal sign is omitted if there are no digits
        to the right of it
      - When the decimal sign is placed before the first digit
        of the number, a zero is placed before the decimal sign
      - The decimal sign is a period (.)
      - No spaces are placed between digits (ie digits are not
        grouped in threes as they should be)
      - A leading minus (-) is added if the number is negative
  (b) Exponential notation (FORM=2):
      - The exponent is adjusted to put just one (non-zero)
        digit before the decimal sign
      - The mantissa is formatted as in (a), unless its value is
        1 in which case it and the multiplication sign are omitted
      - If the power of 10 is not zero and the mantissa is not
        zero, an exponent of the form \x10\u[-]nnn is appended,
        where \x is a multiplication sign (cross), \u is an escape
        sequence to raise the exponent, and as many digits nnn
        are used as needed
  (c) Automatic choice (FORM=0):
        Decimal notation is used if the absolute value of the
        number is less than 10000 or greater than or equal to
        0.01. Otherwise exponential notation is used.

  (d) Write as <degrees>\u<minutes>\d (FORM=3)
      - The input is considered to be an angle in degrees. It is
        converted to degrees and minutes and plotted as an integer
        number of degrees and integer number of minutes, written as
        superscript.

  (e) Write as <degrees>\u<minutes>\d, only if min<>0 (FORM=4)
      - As FORM=4, only <minutes> is not written if it is zero.

  (f) Write as <hour>:<min> (FORM=5)
      - The input is considered to be time in hours. It is converted
        to hours and minutes and returned as a string <hours>:<min>

  (g) Write as <hour>:<min>:<sec> (FORM=6)
      - Similar to FORM=5, but including seconds as well.

Arguments:
```

4.5 PGPT – draw several graph markers

```
    SUBROUTINE PGPT (N, XPTS, YPTS, SYMBOL)
    INTEGER N
    REAL XPTS(*), YPTS(*)
    INTEGER SYMBOL
```

```
Primitive routine to draw Graph Markers (polymarker). The markers
are drawn using the current values of attributes color-index,
line-width, and character-height (character-font applies if the symbol
number is >31).  If the point to be marked lies outside the window,
no marker is drawn.  The "pen position" is changed to
(XPTS(N),YPTS(N)) in world coordinates (if N > 0).

Arguments:
 N      (input)  : number of points to mark.
 XPTS   (input)  : world x-coordinates of the points.
 YPTS   (input)  : world y-coordinates of the points.
 SYMBOL (input)  : code number of the symbol to be drawn at each
                   point:
                   -1, -2  : a single dot (diameter = current
                             line width).
                   -3..-31 : a regular polygon with ABS(SYMBOL)
                             edges (style set by current fill style).
                   0..31   : standard marker symbols.
                   32..127 : ASCII characters (in current font).
                             e.g. to use letter F as a marker, let
                             SYMBOL = ICHAR('F').
                   > 127   :  a Hershey symbol number.

Note: the dimension of arrays X and Y must be greater than or equal
to N. If N is 1, X and Y may be scalars (constants or variables). If
N is less than 1, nothing is drawn.
```

4.6 PGPTX – draw one or more graph markers at given size and angle

```
    SUBROUTINE PGPTX (N, XPTS, YPTS, SIZE, ANGLE, SYMBOL)
    INTEGER N
    REAL XPTS(*), YPTS(*), SIZE(*), ANGLE(*)
    INTEGER SYMBOL

Primitive routine to draw Graph Markers (polymarker). The markers
are drawn using the current values of attributes color-index and
line-width (character-font applies if the symbol number is >31).
The Markers are drawn SIZE times greater than the current character size.
The angle at which the markers are drawn is given by ANGLE.
If the point to be marked lies outside the window, no marker is drawn.
The "pen position" is changed to (XPTS(N),YPTS(N)) in world coordinates
(if N > 0).

Arguments:
 N      (input)  : number of points to mark.
 XPTS   (input)  : world x-coordinates of the points.
 YPTS   (input)  : world y-coordinates of the points.
 SIZE   (input)  : size of the points measured in units of current character
                   size.
 ANGLE  (input)  : tilt of the character, measured anti-clockwise (degrees).
 SYMBOL (input)  : code number of the symbol to be drawn at each
                   point:
                   -1, -2  : a single dot (diameter = current
                             line width).
                   -3..-31 : a regular polygon with ABS(SYMBOL)
                             edges (style set by current fill style).
                   0..31   : standard marker symbols.
                   32..127 : ASCII characters (in current font).
                             e.g. to use letter F as a marker, let
                             SYMBOL = ICHAR('F').
                   > 127   :  a Hershey symbol number.
```

4.7 PGSVPX – set viewport (any unit)

```
    SUBROUTINE PGSVPX (UNITS, XLEFT, XRIGHT, YBOT, YTOP)
    INTEGER UNITS
    REAL    XLEFT, XRIGHT, YBOT, YTOP
```

Change the size and position of the viewport, specifying
the viewport in device coordinates (several possible units specified by
UNITS). The viewport is the rectangle on the view surface "through"
which one views the graph. All the PG routines which plot lines
etc. plot them within the viewport, and lines are truncated at
the edge of the viewport (except for axes, labels etc drawn with
PGBOX or PGLAB). The region of world space (the coordinate
space of the graph) which is visible through the viewport is
specified by a call to PGSWIN. It is legal to request a
viewport larger than the view surface; only the part which
appears on the view surface will be plotted.

Arguments:

UNITS	(input)	: used to specify the units of the output parameters: UNITS = 0 : normalised device coordinates (as PGSVP). UNITS = 1 : inches UNITS = 2 : millimeters UNITS = 3 : pixels Other values give an error message, and are treated as 0.
XLEFT	(input)	: x-coordinate of left hand edge of viewport
XRIGHT	(input)	: x-coordinate of right hand edge of viewport
YBOT	(input)	: y-coordinate of bottom edge of viewport
YTOP	(input)	: y-coordinate of top edge of viewport

4.8 PMBAR – draw scale-bar

```
    SUBROUTINE PMBAR (OPTIONS, DISP, LENGTH, NMAIN, NSUB)
    CHARACTER*(*) OPTIONS
    REAL DISP, LENGTH
    INTEGER NMAIN, NSUB
```

Draw a horizontal scale-bar at the top or bottom of the viewport. This routine is only valid when the geographical projections are used. The scale can be defined with PMDEF; the window must have been specified by PMWINDOW.

```
The bar will look like this   0        10       20        30 km
                              II  II  II    *    IIIIIIIIII
(* is the center of the bar)          Scale 1:40000
```

The dimension of the various items in the bar (in units of character size):

Kilometer scale	100%
Height of the scale-bar	50%
Scale	120%
Total height (excl. scale)	180%
Total height (incl. scale)	310%

Arguments:

OPTIONS	(input)	String of plot options as defined below.
DISP	(input)	Displacement of the center of the scale bar from the top or the bottom of the viewport, measured outwards in units of character height. Use a negative value to put the bar inside the viewport, a positive to put it outside.
LENGTH	(input)	Length of the scale-bar in kilometers (Example: 30.). If LENGTH=0.0, PMBAR will choose the length such that it is at least 20 character units long, but no more than 60. NMAIN and NSUB will then also be set automatically.
	(output)	Length chosen by PMBAR.
NMAIN	(input)	Number of main intervals (Example: 3).
NSUB	(input)	Number of sub-intervals (Example: 5).

Options (for parameter OPTIONS):

- T : draw scale-bar at the top of viewport (Default).
- B : draw scale-bar at the bottom of viewport.
- L : align the left of the scale-bar with the left of the viewport.
- C : align the center of the scale-bar with the center of the viewport (Default).
- R : align the right of the scale-bar with the right of the viewport.
- S : put the scale at the bottom of the scale-bar.

4.9 PMBOX – draw labeled frame around viewport

```
    SUBROUTINE PMBOX (XOPT, XTICK, NXSUB, YOPT, YTICK, NYSUB)
    CHARACTER*(*) XOPT, YOPT
    REAL XTICK, YTICK
    INTEGER NXSUB, NYSUB
```

Annotate the viewport with frame, grid, ticks, numeric labels, etc.
PMBOX is rather similar to PGBOX, but is used especially for non-linear geographical projections.
PMDEF and PMWINDOW must have been called before PMBOX.

Arguments:

XOPT	(input)	: String of options for X (horizontal, longitude) axis of plot. Options are single letters, and may be in any order (see below).
XTICK	(input)	: Longitude interval (degrees) between major tick marks on X axis. If XTICK=0.0, the interval is chosen by PMBOX.
NXSUB	(input)	: Number of subintervals to divide the major coordinate interval into. If XTICK=0.0 or NXSUB=0, the number is chosen by PMBOX.
YOPT	(input)	: String of options for Y (vertical, latitude) axis of plot. Coding is the same as for XOPT.
YTICK	(input)	: Latitude interval (degrees) between major tick marks on Y axis. If YTICK=0.0, the interval will be the same as XTICK.
NYSUB	(input)	: Like NXSUB for the Y axis.

Options (for parameters XOPT and YOPT) common to all projections:

- B : Draw bottom (X) or left (Y) edge of frame.
- C : Draw top (X) or right (Y) edge of frame.
- G : Draw Grid of meridians (X) or parallels (Y).
- H : (projections 32 & 34 only) Draw frame around area and remove anything outside.
- I : Invert the tick marks (plot them outside the viewport).
- P : extend ("Project") major tick marks outside the box (if option I is not set) or inside the box (if option I is set).
- 2 : Increase the number of steps used to draw curved meridians (X) or parallels (Y) by a factor 2. The default number is 50 or half the length of the meridians or parallels measured in degrees (whichever is greater).
- 5 : Increase the number of steps used to draw curved meridians (X) or parallels (Y) by a factor 5. If both options 2 and 5 are used the number of steps is increased by a factor 10.

To get a complete frame, specify BC in both XOPT and YOPT.

Options used with projections other than azimuthal:

- N : Write Numeric labels in the conventional location below the viewport (X) or to the left of the viewport (Y).
- M : Write numeric labels in the unconventional location above the viewport (X) or to the right of the viewport (Y).
- . : Write numeric labels in a decimal notation. The default notation is to have arcminutes as a superscript (unless they are zero).
- : : Write numeric labels in a notation degrees:minutes. The default notation is to have arcminutes as a superscript (unless they are zero).
- – : Write – for West or South. Default is no sign.
- T : Draw major Tick marks at the major coordinate interval. (Ignored if option G is specified. Tick marks are only drawn when B or C are used.)
- S : Draw minor tick marks (Subticks).

```
 V : Orient numeric labels Vertically. This is only applicable to Y.
     The default is to write Y-labels parallel to the axis

Options used for azimuthal and polar projections only:
 H : Draw observer's Horizon and remove all graphics outside the horizon.
     Options B and C will have the same effect.
 T : Omit the meridians in the vicinity of the poles. Meridians will extend
     up to one major Y-interval from the poles. (Effective for XOPT only.)
 S : Meridians will not be draw within one sub-interval from the poles.
     (Used only for XOPT).
 (Ticks or numerals can not be drawn in this mode. N, M, and V are ignored.)
```

4.10 PMCINV – convert X and Y coordinates of several points (inverse).

```
    SUBROUTINE PMCINV (N, X, Y)
    INTEGER N
    REAL X(N), Y(N)

Convert map coordinates (x,y) to real-world coordinates (longitude,latitude).
The real-world coordinates are the position coordinates in degrees,
the map coordinates are mapped linearly in the viewport and
depend on the projection type used. For equi-rectangular projection
real-world and map coordinates are identical, and makes this conversion
optional.

Arguments:
 N    (input) : Number of points of which the coordinates must be converted.
 X,Y  (input) : Map coordinates.
     (output) : Longitude and latitude of the points (degrees).
```

4.11 PMCONV – convert X and Y coordinates of several points.

```
    FUNCTION PMCONV (N, X, Y)
    INTEGER PMCONV, N
    REAL    X(N),Y(N)

Convert real-world coordinates (longitude,latitude) to map coordinates (x,y).
The real-world coordinates are the position coordinates in degrees,
the map coordinates are mapped linearly in the viewport and
depend on the projection type used. For equi-rectangular projection
real-world and map coordinates are identical, and makes this conversion
optional. For non-linear projections real-world coordinates must be
changed into map coordinates by means of PMCONV before using PGLINE, PGPTXT,
etc.
Upon return PMCONV contains the number of points that do not have
invalid values, occurring when a point is not convertable.

Arguments:
 N    (input) : Number of points of which the coordinates must be converted.
 X,Y  (input) : Longitude and latitude of the points (degrees).
     (output) : Map coordinates.
 PMCONV (out) : Number of valid points after conversion.
```

4.12 PMCVEC – convert coordinates of several vectors.

```
    SUBROUTINE PMCVEC (N, X, Y, A)
    INTEGER N
    REAL X(N), Y(N), A(N)
```

```
Convert real-world coordinates (longitude,latitude) and the azimuth of a
vector to map coordinates (x,y) and a tilt.
The real-world coordinates are the position coordinates in degrees, and the
azimuth is measured eastward from the north in degrees. The map coordinates
are mapped linearly in the viewport and depend on the projection type used;
the tilt is measured anti-clockwise (in degrees) starting horizontal pointing
to the right. Even for linear projections (where the real-world coordinates
are the same as the map coordinates) the azimuth is different from the tilt,
not only because they are measured in different ways, but also simply because
the linear rectangular projections do not preserve angles.
This routine is very useful as a preparatory step to PGMARK.

Arguments:
 N    (input) : Number of points of which the coordinates must be converted.
 X,Y  (input) : Longitude and latitude of the points (degrees).
     (output) : Map coordinates.
 A    (input) : Direction of the vector, measured from the north eastward
                (in degrees) = Azimuth.
     (output) : Tilt (degrees) corresponding to the azimuth, measured
                anti-clockwise starting horizontal, pointing to the right.
```

4.13 PMCPOL – convert X and Y coordinates of a polygon.

```
    FUNCTION PMCPOL (N, X, Y)
    INTEGER PMCPOL, N
    REAL X(N),Y(N)
```

```
Convert real-world coordinates (longitude,latitude) to map coordinates (x,y).
The real-world coordinates are the position coordinates in degrees,
the map coordinates are mapped linearly in the viewport and
depend on the projection type used.
For map projections other than azimuthal, PMCPOL is equivalent to
PMCONV. When an azimuthal projection is chosen, PMCPOL maps
out-of-area coordinates to the circular edge.

Arguments:
 N       (input) : Number of points in the polygon.
 X,Y     (input) : Longitude and latitude of the points (degrees).
        (output) : Map coordinates.
 PMCPOL (output) : Number of points inside window.
```

4.14 PMDEF – set geographical projection type and scale

```
    SUBROUTINE PMDEF (PROJECT, SCALE, PARA1, PARA2)
    INTEGER PROJECT
    REAL SCALE, PARA1, PARA2
```

```
Define projection type and open possibility to use the PMPLOT routines.
Also choose the map scale to be used and the standard or true-scale
parallels.
If you select any of the non-linear projection types, conversion
```

```
between real-world coordinates (longitude,latitude) and map coordinates
(x,y) is mandatory, and PMCONV must be used. For equi-rectangular projection
it is optional.
If you want to use any of the other PMPLOT routines, call PMDEF first.

If you select a projection type less or equal to zero, standard PGPLOT
routines will be called by PMWINDOW and PMBOX.

Arguments:
 PROJECT  (input) : Projection type:

 [ <=0: Non-geographical projections:]
  =  0, Adjusted window (PGWNAD) (same scale both direction)
  = -1, Maximum window (PGSWIN) (unequal scales)

 [1-10: Cylindrical projections:]
  =  1, Equi-rectangular (linear).
  =  2, Peters (non-linear).
  =  3, Mercator (non-linear).
  =  4, Miller (non-linear).
  =  5, Gall's stereographic (non-linear).
  =  6, ERS projection (non-linear).

[11-20: Azimuthal projections (non-linear):]
  = 11, Orthographic.
  = 12, Perspective.
  = 13, Azimuthal equal-area.
  = 14, Azimuthal equi-distant.

[21-30: Conic projections (non-linear):]
  = 21, Ptolemy or conic equal-interval.
  = 22, Kavraiskiy IV or conic equal-interval.

[31-40: Miscellaneous projections (non-linear):]
  = 31, Sinusoidal or Mercator equal-area.
  = 32, Mollweide projection.
  = 33, Bartholomew's 'The Times' projection.
  = 34, Tilted rectangular.

[41-50: Polar projections (non-linear):]
  = 41: Polar projection centred on the North Pole
  = 42: Polar projection centred on the South Pole

For all projections other than Azimuthal:
 SCALE (input) : Scale of the map (1:SCALE) along the true-scale or standard
                 parallel(s). If the plot does not have to be at any
                 particular scale, specify SCALE=0.0.
 PARA1 (input) : Latitude of the standard (or true-scale) parallel.
 PARA2 (input) : (only for projections having two standard parallels; e.g.
                 Kavraiskiy IV) Latitude of the second standard parallel.

For Azimuthal projections:
 SCALE (input) : Scale of the map (1:SCALE) at the center of the plot.
                 If the plot does not have to be at any particular scale,
                 specify SCALE=0.0.
 PARA1, PARA2  : (not used).

For tilted rectangular projection (34):
 SCALE (input) : Scale of the map (1:SCALE) at the center of the plot.
```

```
                If the plot does not have to be at any particular scale,
                specify SCALE=0.0.
 PARA1 (input) : Vertical scaling (unity).
 PARA2 (input) : Tilt (degrees).
```

4.15 PMQDEF – inquire projection type and scale

```
    SUBROUTINE PMQDEF (TYPE, SCALE)
    CHARACTER*(*) TYPE
    REAL SCALE
```

```
Query the type of geographical projection defined in PMDEF and the
scale of the map defined in PMDEF or computed by PMWINDOW.

Arguments:
 TYPE   (output) : Character string quoting the projection type used.
 SCALE  (output) : Scale (1:SCALE) of the map
```

4.16 PMQINF – return general PMPLOT information

```
    SUBROUTINE PMQINF (NAME, VALUE)
    CHARACTER*(*) NAME
    REAL VALUE(*)
```

```
Query some of the PMPLOT settings such as version-number, window settings,
etc. NAME is a string identifying which information should be returned.
This character value can be given both in lower and upper case.
VALUE is a buffer of returns values.

Arguments:
 NAME   (input) : Type of information to query (case is irrelevant):
      'VERSION' : Version number will be returned.
         'AREA' : Min/Max longitude and Min/Max latitude in plot.
     'USERAREA' : Area set by the user in PMDEF.
        'SCALE' : Scale (1:SCALE) of the plot.
 VALUE (output) : Value(s) returned.
```

4.17 PMRND – find the smallest ”round” number greater than X (in degrees)

```
    REAL FUNCTION PMRND (X, NSUB)
    REAL X
    INTEGER NSUB
```

```
Routine to find the smallest "round" number larger than X to be used in
maps and the appropriate sub intervals. Round numbers are:
180 (4), 90 (3), 60 (4), 30 (3), 10 (5), 5 (5), 2 (4), 1 (4),
30' (3), 10' (5), 5' (5), 1' (4).
This routine is used by PMBOX for choosing  tick intervals.

Arguments:

 PMRND (output): the "round" number.
 X    (input)  : the number to be rounded.
 NSUB (output) : a suitable number of subdivisions
```

4.18 PMWDB – draw coastlines, rivers, boundaries or land masses.

```
    SUBROUTINE PMWDB (NAME, I1, I2)
    CHARACTER*(*) NAME
    INTEGER I1, I2

Draw coastlines/rivers/boundaries/landmasses/lakes from World Data
Bank (WDB) in the present PGPLOT window. The map boundaries and projection
type must have been specified using the proper PMPLOT routines (PMDEF and
PMWINDOW). The map drawn by this routine may extend to any longitude
(x-coordinate) and in latitude (y-coordinate) from -90 to +90 degrees.

** In case a line-segment data set (.bth, .cil, .bdy, .pby, .mnt, .bth, .plt)
   is used:

The line segments stored in the WDB all have rank-numbers. The
most important line-segments have rank 1, then comes rank 2, etc.
You may specify the ranks to be drawn by selecting proper values for
I1 (minimum rnak number) and I2 (maximum rank number). Use I2<I1 if all
ranks must be plotted.
All relevant line segments are drawn with line properties set by
PGSCI, PGSCR, PGSLS, PGSLW.

Arguments:
 NAME (input) : Name of the WDB dataset to be drawn, without the
                extensions '.DAT' or '.TAB'. If the dataset cannot
                be found in the current directory, the default directory
                specified by the environment variable WDB_DIR is searched.
 I1   (input) : Minimum rank.
 I2   (input) : Maximum rank.

** In case a landmasses data set (.lnd) is used:

Landmasses and lakes are drawn as polygons with color indices I1 and I2,
respectively. The color index selection before the call to PMWDB is
restored after execution.

Arguments:
 NAME (input) : Name of the WDB dataset to be drawn, without the
                extensions '.DAT' or '.TAB'. If the dataset cannot
                be found in the current directory, the default directory
                specified by the environment variable WDB_DIR is searched.
 I1   (input) : Color index for landmasses.
 I2   (input) : Color index for lakes.
```

4.19 PMSWIN – set geographical projection window and adjust viewport

```
    SUBROUTINE PMSWIN (A, B, C, D)
    REAL A, B, C, D

Change the window in real-world coordinate space that is to be mapped on
to the viewport, and simultaneously adjust the viewport so that the
projection type and scale selected in PMDEF are satisfied. If the
scale was set to 0.0 in PMDEF, the new viewport is the largest one
that can fit within the previously set viewport while retaining the
required aspect ratio, and the scale will be computed.
The window's coordinates will be map coordinates (x,y). However,
the input is in real-world (longitude,latitude) coordinates (degrees).

Arguments:
For all projections others than Azimuthal:
 A (input) : The most western longitude of the area to be mapped (degrees).
 B (input) : The most eastern longitude of the area to be mapped (degrees).
 C (input) : The most southern latitude of the area to be mapped (degrees).
 D (input) : The most northern latitude of the area to be mapped (degrees).

For Azimuthal projections:
 A (input) : The longitude of the central point in the map (degrees).
 B (input) : The latitude of the central point in the map (degrees).
 C (input) : (for Perspective projection) The height of the observer
             above the earth's surface (kilometers).
             (for Azimuthal Equi-distant projection) The maximum range
             to be plotted (kilometers). If C=0.0 the whole world will be
             plotted.
 D         : (not used).
```

4.20 PMX – determine map X coordinate

```
    REAL FUNCTION PMX (Y, LON)
    REAL Y, LON

Compute map x-coordinate of a point of given map y-coordinate and longitude.

Arguments:
 Y    (input) : Map y-coordinate.
 LON  (input) : Real-world longitude (degrees).
 PMX (output) : Map x-coordinate.
```

4.21 PMY – determine map Y coordinate

```
    REAL FUNCTION PMY (X, LAT)
    REAL X, LAT

Compute map y-coordinate of a point of given map x-coordinate and latitude.

Arguments:
 X    (input) : Map x-coordinate.
 LAT  (input) : Real-world latitude (degrees).
 PMY (output) : Map y-coordinate.
```

4.22 RNGCIR – compute coordinates of points at equal distance to location

```
    SUBROUTINE RNGCIR (LON, LAT, RADIUS, N, AZI1, AZIN, X, Y)
    INTEGER N
    REAL LON, LAT, RADIUS, AZI1, AZIN, X(N), Y(N)
```

Compute the coordinates of a number of points with a specified distance to a certain location. In other words, compute the coordinates of a range circle.

```
Arguments:
 LON    (input) : Longitude (degrees) of the center of the range circle.
 LAT    (input) : Latitude (degrees) of that point.
 RADIUS (input) : Radius of the first range circle (kilometers).
 N      (input) : Number of points on the range circle. (N should be greater
                  than 1)
 AZI1   (input) : Lower azimuth boundary (degrees) measured from the north
                  eastward.
 AZIN   (input) : Upper azimuth boundary (degrees).
 X     (output) : Array of longitude coordinates (degrees) of points
                  1 through N.
 Y     (output) : Array of latitude coordinates (degrees) of points
                  1 through N.
```

4.23 SATOBS – compute sub-satellite point from range, elevation and azimuth.

```
    SUBROUTINE SATOBS (OBSLON, OBSLAT, RANGE, ELEV, AZIM, LON, LAT)
    REAL OBSLON, OBSLAT, RANGE, ELEV, AZIM, LON, LAT
```

Compute the coordinates of a the sub-satellite point from range, elevation and azimuth observation of a satellite from a given station.

```
Arguments:
 OBSLON (input) : Longitude of the observer (degrees).
 OBSLAT (input) : Latitude of the observer (degrees).
 RANGE  (input) : Distance between observer and satellite (kilometers).
 ELEV   (input) : Elevation of the range observation (degrees).
 AZIM   (input) : Azimuth of the range observation (degrees).
 LON   (output) : Longitude of the sub-satellite point (degrees).
 LAT   (output) : Latitude of the sub-satellite point (degrees).
```