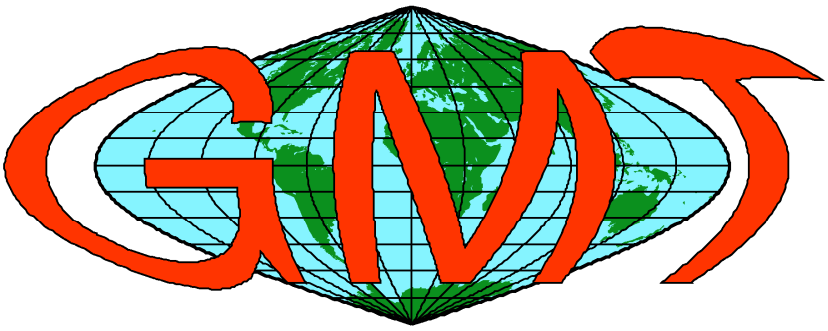
# GMT - GENERIC MAPPING TOOLS

*Data Processing and Plotting*




**Generic Mapping Tools Graphics**


4th Year Honours Course Unit (25 hours)
University of Sydney

Maria Seton and Jo Whittaker

# TABLE OF CONTENTS

# 1. Introduction

## The Course

This course is designed to introduce students to different types of spatial data, data processing, interpolation functions and data plotting using GMT (Generic Mapping Tools). GMT is a set of computer programs which will be used in conjunction with UNIX general processing tools (awk, grep) and basic shell programming. The examples used will be focussed on marine geophysical data, however many of the principles are applicable to any variety of scientific data. The learning outcomes include:

- ☑ an understanding of the differences between vector and raster data
- ☑ knowledge of different map projections and geodetic parameters
- ☑ understanding of different processing techniques used for vector and raster data, in particular an understanding of data interpolation techniques
- ☑ ability to use basic shell programming, UNIX commands and awk programming
- ☑ ability to use common GMT commands to plot, process and manipulate scientific data

The course is targeted towards honours (fourth year) students with a background in geosciences and an interest in developing skills in data processing and data visualisation. The course is designed for new users of GMT. A background in UNIX is useful but not essential.

It is recommended that you download the GMT Technical Reference and Cookbook from the Docs page on the GMT website:

http://gmt.soest.hawaii.edu/

You should also complete the online GMT Tutorial which you can access from the same website.

## What is GMT?

GMT stands for Generic Mapping Tools and is an open-source (i.e. free!) software package primarily used to process, manipulate and plot scientific data. It is released under the GNU General Public License.

GMT compiles under UNIX/Linux, Windows and Mac OSX operating systems (Figure 1.1). GMT was jointly developed by Professor Paul Wessel from the University of Hawaii and Professor Walter Smith from NOAA while both were postgraduate students at Lamont-Doherty Earth Observatory at Columbia University. GMT was born in 1987 and has been expanded and improved every year since and is currently used by over 15,000 people worldwide (this number includes registered users only). There is now a team of

dedicated GMT developers but is still ultimately maintained by Prof. Paul Wessel. For more information on the history of the development of GMT, read the Introduction section (Chapter 2, p32) of the GMT Technical Reference and Cookbook.

GMT is written in POSIX C making it very portable and easy to maintain. It consists of over 60 individual programs as well as many supplements which enable the processing, manipulation and visualisation of 2D time-series or x,y series or 3D data sets. The individual programs can be combined and integrated with UNIX commands in a variety of ways to achieve complex tasks. GMT can deal with vector (point, line, polygon) data and raster (or gridded) data.

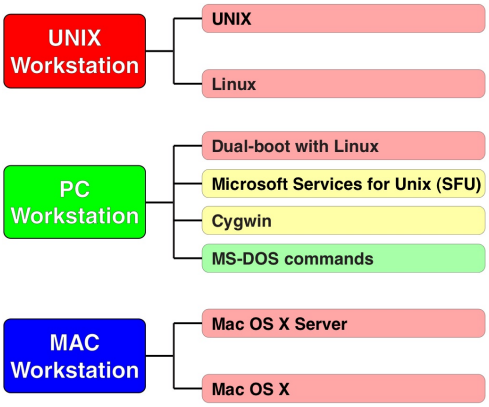



*Figure 1.1: GMT can be compiled on a UNIX, PC and Mac workstation.*

GMT is not a GIS (Geographical Information System). A GIS is made up of hardware, software, a network, a database, a management structure and procedures and people. The main point of difference between GMT and a GIS is the database. GMT does not have a back-end database for the storage and retrieval of data. Instead, users store their data on their computer or on a network as individual files that are not linked via a relational database.

The advantages of using GMT include:

- ☑ High quality ("camera-ready") PostScript output.
- ☑ Fit to process geophysical data.
- ☑ Support various geographical map projections.
- ☑ Support 3D plot with mesh, surface plot.
- ☑ Powerful gridding functions
- ☑ Grid, image data manipulation.
- ☑ Support 24 bit colour.
- ☑ Contains different resolution selectable coastline data.

- ☑ Easy to install on any platform (Windows, Mac OSX, Linux)
- ☑ NetCDF binary data support
- ☑ It's free (open source).
- ☑ The developers are scientists and users

For more detailed information about GMT, visit the GMT website http://gmt.soest.hawaii.edu/

An interactive mapping version of GMT with a Graphical User Interface (GUI) called iGMT has been developed. Due to its interactive mode, this software is more suitable for beginners or those without knowledge of GMT commands who want to plot simple maps.

## 2. Spatial Data

Spatial data includes any data/information that can be related to space or location. On a more technical note, spatial data is defined as "the conversion or abstraction of the earth and its properties to a database that defines location and properties of individual features of interest".

All measurable or describable properties of the world can be considered to fall into three components:

- ☑ Spatial - "where" - linking of data to a place/location
- ☑ Attribute or thematic - "what" - linking properties of the data
- ☑ Temporal - "when" - linking data to a time

For the purpose of this course, we will only look at the spatial component.

Spatial refers to any space. It is not only the space of the Earth's surface but can be applied to space on other planets, solar systems, galaxies, etc. It deals with the variation from place to place - the "where is" component. These locational data can have a precise georeference, often encoded as coordinates on a particular Cartesian coordinate system with a particular map projection (e.g. latitude and longitude coordinates, a national grid coordinate), or they can have an relative location (e.g. the name of a road, a postcode, a pixel in the middle of a satellite image, grid reference).

Spatial data can be translated into simple objects and come into four basic forms:

- ☑ *Points* are described as "zero-dimensional features" in the sense that they do not have a length or a width on a map. Examples include: a city on a global map, the location of a church or an an old mine on a local or regional map, a magnetic anomaly identification on the world's ocean floor.
- ☑ *Lines* are described as "one-dimensional features" as they have a length. Examples include: a terrain contour, a road or railway line, a seafloor spreading isochron.
- ☑ *Polygons* are described as "two-dimensional features" as they have a length and a width and are sometimes called areas. Examples include: land use type, a lake, country borders, an area of anomalous seafloor.
- ☑ *Surfaces* are described as "three-dimensional features" as they have length, width and a third dimension determined by the characteristics of that surface. Surfaces can sometimes be called volumes or grids. Examples include: topographic surfaces, the volume of a seamount, sea-surface temperature grid.

When we talk about dimensions, we must keep in mind that dimension is a function of scale. For example, a city can be represented as a point on a global map but we know

that in actuality a city has a length, a width and an area. Likewise, a river is abstracted to a one-dimensional feature but we know that it also has a width and area. We use the four forms of spatial data above to abstract features in a simple way. However, there are two fundamentally different ways of abstracting the four forms of spatial data - as *discrete objects* and *continuous fields*.

*Discrete objects* (Figure 2.1) represent the real world as entities or objects with well-defined boundaries in otherwise empty space. They are described by their attributes or properties and their position is mapped using a geometric coordinate system. Examples of discrete objects include a house, an animal, a fire hydrant, a road, a country, a rock outcrop, a fault line. Discrete objects are represented as a vector data in a vector data model.

*Continuous fields* (Figure 2.2) represent real world entities that vary continuously in space. Continuous fields can be represented as a continuous mathematical function or field but cannot be represented as a simple polynomial equation. Examples of continuous fields include: sea surface temperature, wind speed, elevation, vegetation type, electromagnetism, seismic velocities. Continuous fields are represented as raster data.

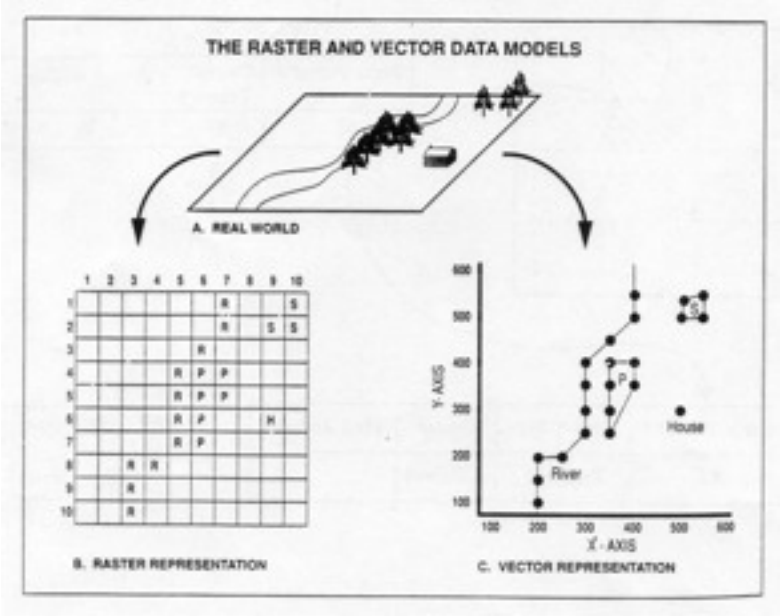



*Figure 2.1: Representation of the real world depicting a river, some trees and a cabin. The equivalent raster representation (left) codes the location of the river, trees and cabin as filled entries in grid cell. The equivalent vector representation (right) codes the river as a line, the trees as areas and the cabin as a point.*

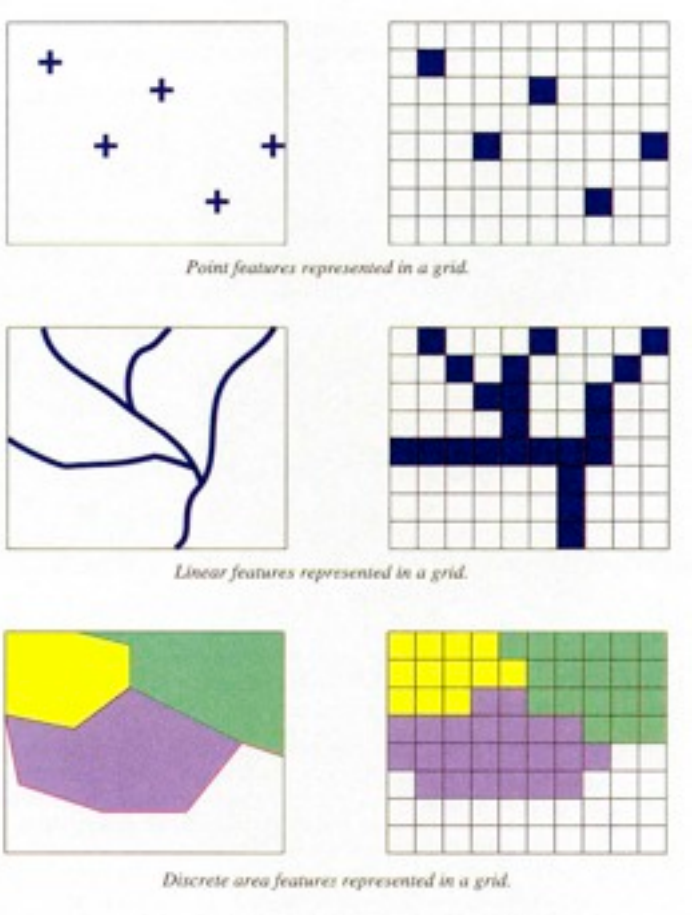



*Figure 2.2: Discrete objects on the left represented as points, line and areas (vector data). The equivalent data represented as a raster/gridded dataset (raster data).*

## Vector Data

Vector data has its origins in cartography where lines or vectors were always used to depict landscapes and features (think of vector data as line drawings). Surveying and map-making techniques were founded on the principles of geometry and trigonometry which employ vectors.

Vector data are a physical representation of points, lines and areas.

- Points with attributes are represented by a single coordinate and a vector of attributes. Point features are standalone nodes (Figure 2.3)
- Lines or linear features are an ordered set of point coordinates. At the beginning and end of every line feature is a node. At each bend (change of direction) is a vertex (plural: vertices). Nodes are end points and vertices are between, defining the shape (Figure 2.3). Chains connect the shape points to draw the feature's outline.
- Polygons or areas can be stored as a closed loop of coordinates. At the beginning and end of every polygon feature is a node (same coordinates). At each bend (change of direction) is a vertex (plural: vertices). Nodes are end points and vertices are between, de-

fining the shape (Figure 2.3). Chains connect the shape points to draw the feature's outline.

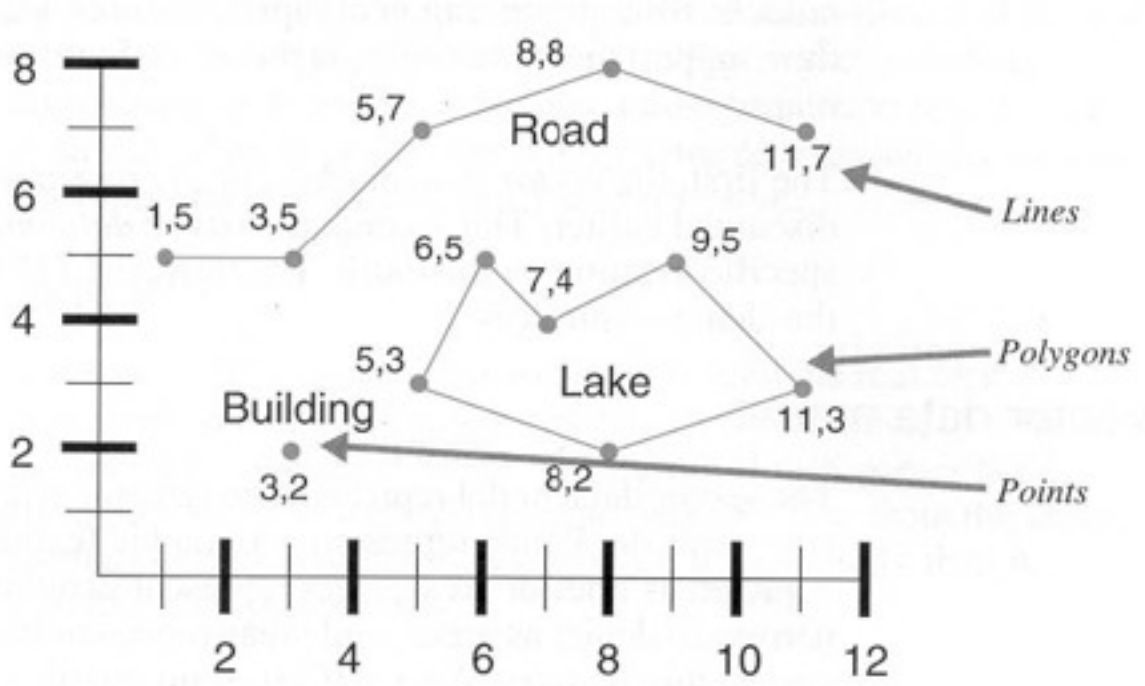



*Figure 2.3: An example of a vector data model on a simple Cartesian coordinate system. The building is represented as a point feature with x,y coordinate pair of (3,2) and is represented as a standalone node in the data model. The road is defined as a polyline with nodes on either end defining the end points of the line and with vertices in between. The only information that is stored in the data model are the 5 sets of coordinates. The lake is defined as a polygon or area, the node is the start point of the polygon. In this case (5,3) and it is repeated at the end of the polygon. A other points are vertices. The polygon is defined by 7 coordinate pairs.*

In the vector system, data files store only the coordinate of each node and vertex. Chains are vectors or data structure paths that are not part of the actual stored data elements; they are not real lines, but define and present the connection between shape points. The software draws the connecting chain segments. The vector data structure is also known as an arc-node model because it uses chains (arcs) and end points (nodes).

Unlike the raster model which only implicitly stores coordinate pairs, the vector model explicitly stores coordinate pairs. There is an x,y value for every point or node. The coordinate pair can be encoded with any conceivable degree of precision. However, this does not necessarily mean that the data is more accurate.

The vector data model does not have the generalising effect of a raster data model as it is the map-like drawing of features. Shape is better retained in the vector data model. Vector data is also much more spatially accurate than the raster format and allows for a precise representation of locations, lengths and areas.

The vector model is extremely useful for describing discrete features but less useful for describing continuously varying features.

The **advantages** of using a **vector data model** include:

- ☑ Very high resolution supports high spatial accuracy.
- ☑ The graphical output from vector data more closely resemble hand-drawn maps, making it easier for the general public to understand what is shown on a vector map and people are more familiar with it.
- ☑ Vector formats have storage advantages. You need only store nodes and vertices rather than a value for each grid cell that covers an area, requiring less disk space to store data.
- ☑ Vector data can also be topological which means that qualitative (nonmetric) properties of geographic objects can be assigned.

The **disadvantages** of using a **vector data model** include:

- ☑ Can be more difficult to manage and maintain than raster data because of a more complex data structure.
- ☑ Vector data require more powerful, high-tech machines meaning that hardware and software is more expensive.
- ☑ Often learning the technical aspects of vector systems is more difficult than understanding the simplicity of the raster format particularly when topology is introduced.

### *Vector Data Storage (File Formats)*

Vector data explicitly stores coordinate pairs and only stores the nodes and vertices of each each shape. There are several file formats that are commonly used to store vector data, including the following formats: ASCII, ESRI shapefile, postscript, XML, SVG.

ASCII (American Standard Code for Information Interchange) has become an international standard for the coding of alphanumeric and graphic characters used as computer input-output data. They are 1D tables that can be edited with a text editor (Figure 2.4). ASCII files can be read by GMT commands as long as the x and y coordinates are specified in the first 2 columns. Many computer programming languages such as awk, perl, python, can read and manipulate ASCII data.

```
>
-13.15900 149.85000 0.05
-13.36500 150.14800 0.45
-13.11000 150.28000 0.15
```

*Figure 2.4. Example of ASCII formatted data. The ">" sign denotes a heading, column 1 is the latitude, column 2 is the longitude and column 3 is a value corresponding to that coordinate point.*

## Raster Data

In a raster representation or model the landscape or space is divided into an array of rectangular cells or pixels. All geographical variation is then expressed by assigning properties or attributes to these cells (Figure 2.5). Raster data generalise reality (i.e. all of the features in the cell area are reduced to a single cell identity). Think of a raster data model as a chessboard.

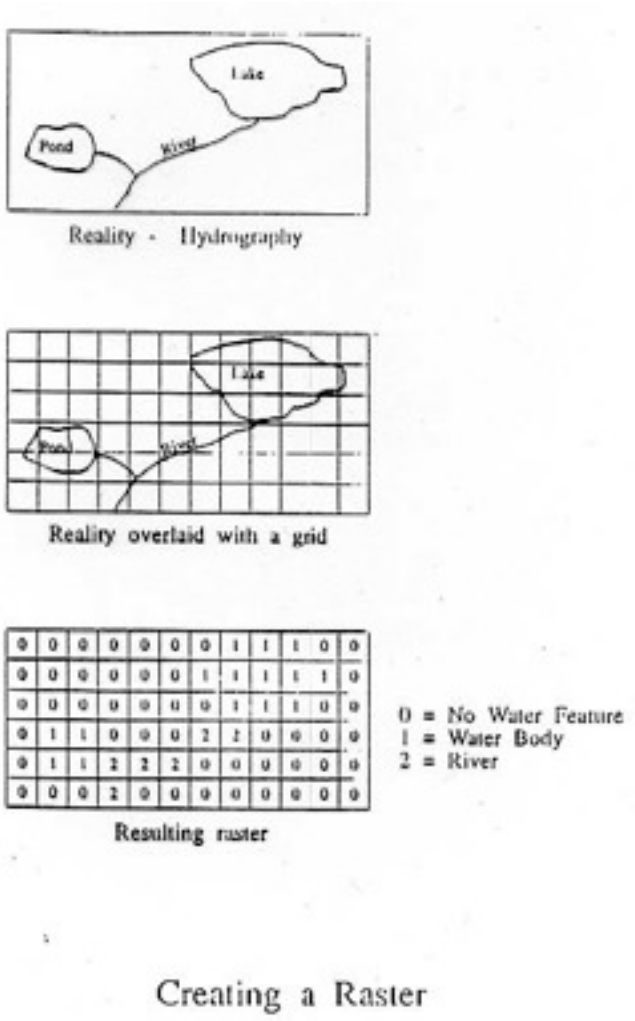



*Figure 2.5: In the real world, the landscape comprises a river with a large lake and small pond. To convert this into raster data, a rectangular grid is overlaid onto the landscape based on a chosen resolution. Each grid cell is then assigned a value based on the properties of that grid cells (no water, water, river). All the features are defined by their grid cells to build up an image of the landscape.*

With raster data, only data values are stored. The data are ordered either by column within row or row within column and is usually referred to as a lattice or grid. The size of the lattice or grid determines the resolution of the raster data. The cell is the minimum mapping unit. It is not necessary to record the locations of every cell. Instead, the metadata associated with the raster data contains all the required information (Figure 2.6), this includes:

- ☑ The geographic coordinate of the upper-left hand corner of the grid (the origin)
- ☑ The cell size or grid resolution
- ☑ The number of row and columns elements (number of x and y nodes) and
- ☑ The projection

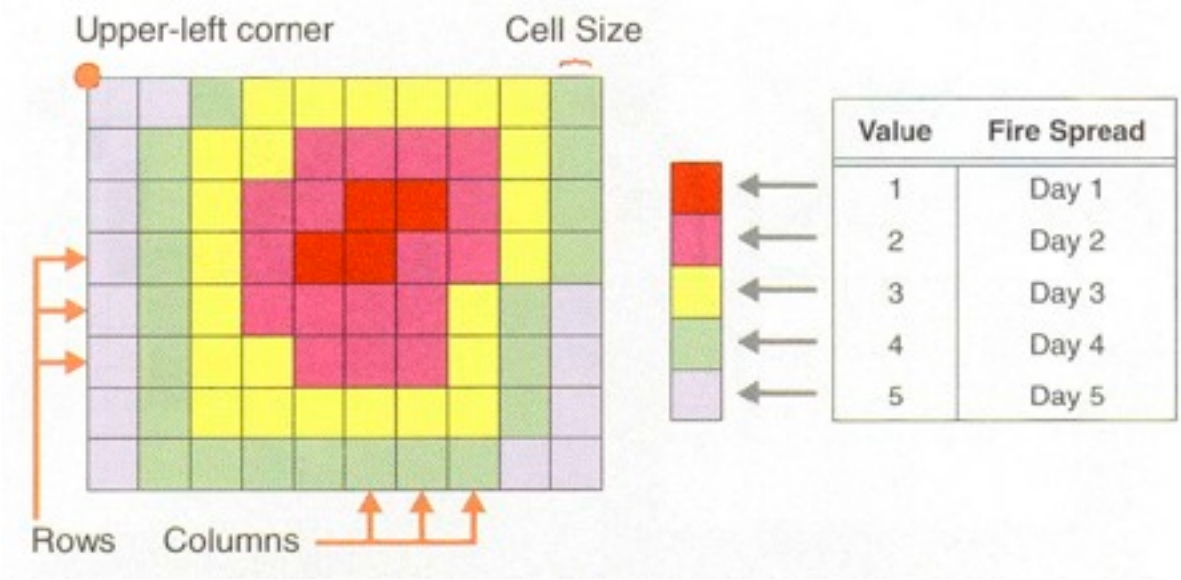



*Figure 2.6: This example is a fire spread model and includes both spatial and temporal data. The grid cell are assigned values based on days (temporal). The spatial data is implicit and defined by the origin in the upper-left corner, the cell size and the number of rows and columns.*

The attribute values for any grid cell can be based on several encoding schemes and represented as (Figure 2.7):

- ☑ Integer values which are just whole numbers. For example, 1, 2, 3, 100, 123. Examples may include a soil class
- ☑ Real decimal values which are just all numbers that can be expressed as decimals. For example, 2.1, 2.3, 3.0. 100.45. Examples may include elevation
- ☑ Non-numeric values or strings which comprise values that are not numbers. For example, a, b, c, norway spruce, radiata pine. Examples of non-numeric values might include a vegetation class.

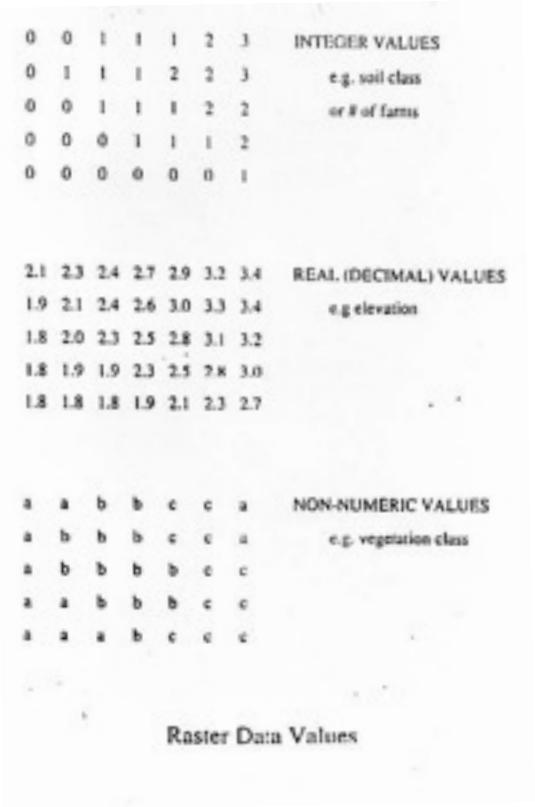



*Figure 2.7: Examples of raster data values. They can be numeric and non-numeric, integers and real values.*

The value assigned to a cell takes up the entire cell but the value may not be true for the entire cell (Figure 2.8). The values are assigned by taking:

- ☑ The average of all the values in the grid cell, can be called the presence/absence method
- ☑ A sample at the corners of the cells. This is often called gridline registration.
- ☑ A sample taken at the centre of the cell. This is often called pixel registration or the cell centre method.
- ☑ A value based on a dominant area principle whereby the cell code is assigned to the feature with the largest or dominant share of the cell.
- ☑ A value based on the percentage coverage method (more advanced method)

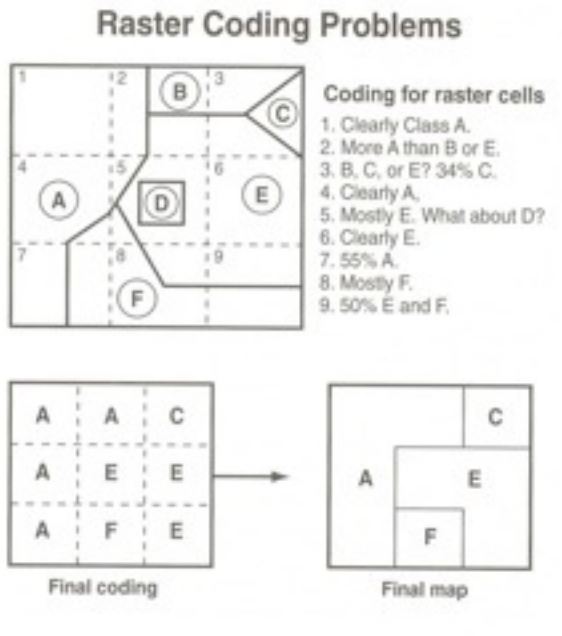



*Figure 2.8: This example clearly highlights the problems inherent in coding raster data. In our final coding we end up with one unique attribute value for each grid cell which is a representation of reality but does not match reality. For example, polygon D is completely lost in our final representation.*

A major problem with the raster structure is that the shape of features is forced into an artificial grid cell format which introduces spatial inaccuracies (Figure 2.9). For right-angled features, such as square agricultural fields or rectangular political districts, this may not present a major problem. However, for many features, size and shape can become undesirably distorted.

A solution to the artificial grid shaping of areas is to increase the grid resolution. Increasing the number of cells on a data set increases the spatial resolution, which helps to increase spatial accuracy (Figure 2.9). However, the one advantage to using relatively few cells is the short processing time and ease of analysis and in some cases this can outweigh any benefit that may be had from a higher resolution data model.

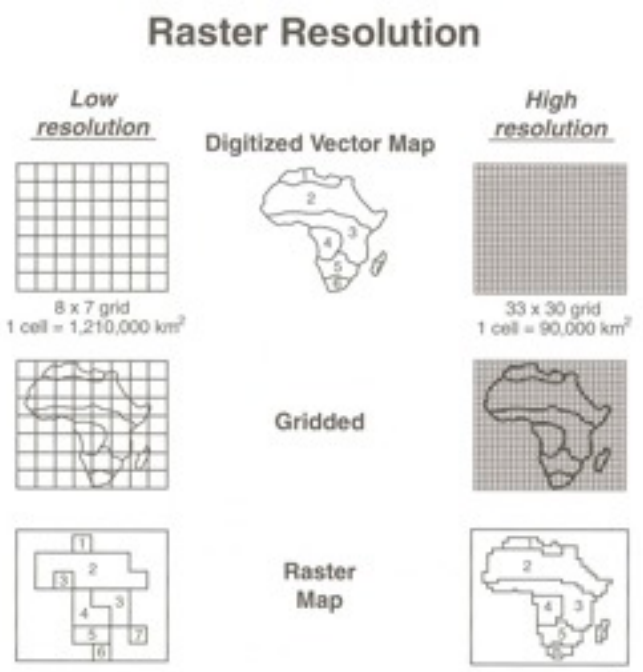



*Figure 2.9: Digitised vector map of the African continent with Africa divided into 6 different climatic zones. We want to represent the change in climate across the African continent as a raster data model. On the left we have a low resolution representation and on the right a high resolution. The initial grid in the low resolution case is 8 x 7 with each cell representing 1,210,000 square kilometers whereas in the high resolution case the initial grid is 33 x 30 with one grid cell equaling 90,000 square kilometers. When the vector model of Africa is superimposed onto the grid and assign attributes to each grid cell, we find that in the low resolution case, we have to use the proportion area technique on larger areas, forcing us to lose accuracy along the boundaries of the African continent. In the high resolution case, because the grid cells are so much smaller, less area has to be sacrificed. In the end the raster map of Africa's climate zones doe not resemble Africa at a in the low resolution case. A detail in lost to the point where the data is no longer in usable form. The low-resolution raster results in a rather generalized and crude shape. In the higher resolution case, we can still see that the boundaries of the African continent take the shape of the grid cells themselves but we at least can identify this as being the African continent. The high-resolution raster shape appears more realistic, though still a long way from the vector shape and spatial accuracy.*

Raster data represent continuous features, that is, things that continuously vary e.g. topography and represents one theme of geographic data.

The **advantages** of using a **raster data model** include:

- ☑ They have a relatively simple data structure - it has a grid defined by cells of square or rectangular shape. The simple grid structure makes analysis easier.
- ☑ The computer platform to run and store raster grids can be low tech and inexpensive.

- ☑ Remote sensing imagery like satellite data or aerial photographs are typically obtained in raster format.
- ☑ Spatial analysis procedures or modelling is typically quite simple.

The **disadvantages** of a **raster data model** include:

- ☑ Spatial inaccuracies, but the level of inaccuracy is based on the grid cell size
- ☑ Relatively low resolution compared to the vector format because each cell tends to generalise a landscape. As a result, a raster format cannot tell precisely what exists at a given location
- ☑ Each cell must have a code even where there is no data - where nothing exists. This is in contrast to the vector model where white space does not matter.

## Colour

Colour derives from the spectrum of light (distribution of light energy versus wavelength) interacting in the eye with the spectral sensitivities of the light receptors. To best understand colour, we must understand light and energy.

There are many sources of light (e.g. the sun, radar, camera flash, computer monitor or TV screen). The energy emitted from the Sun is known as electromagnetic radiation. All electromagnetic radiation has fundamental properties and behave according to the basics of wave theory. Two characteristics of electromagnetic radiation are particularly important for understanding colour: wavelength and frequency.

The wavelength is the length of one wave cycle, which can be measured as the distance between successive wave crests. Frequency refers to the number of cycles of a wave passing a fixed point per unit of time. Wavelength and frequency are related by a formula which shows that the two characteristics are inversely related to each other. The shorter the wavelength, the higher the frequency. The longer the wavelength, the lower the frequency. This is best represented in Figure 2.10. The long wavelength, low frequency signal also corresponds to low energy, while short wavelength and high frequency correspond to high energy. Understanding the characteristics of electromagnetic radiation in terms of their wavelength and frequency (and therefore energy) is crucial to understanding what colour actually is (i.e. a wave) and the information that can be extracted from remote sensing data.

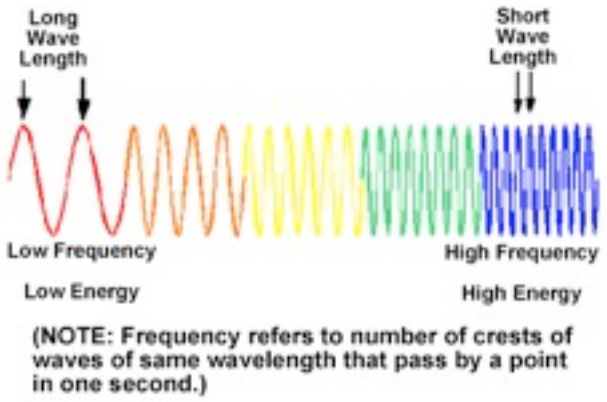



*Figure 2.10: Wavelength and frequency are inversely related to each other. The shorter the wavelength, the higher the frequency. The longer the wavelength, the lower the frequency. The receptors in our eyes view high frequency and short wavelengths as the colour blue and low frequency, long wavelength as the colour red.*

Electromagnetic radiation that varies from high to low energy levels comprises the ElectroMagnetic spectrum (EMS). The electromagnetic spectrum (Figure 2.11) ranges from the shorter wavelengths (gamma and x-rays) to visible light (mid-range) to the longer wavelengths (microwaves, radio waves). The light which our eyes can detect is part of the visible spectrum and only makes up a small portion of the EMS. There is a lot of radiation around us which is "invisible" to our eyes, but can be detected by other remote sensing instruments. The visible wavelengths (the wavelengths that our eyes can detect) cover a range from approximately 400 to 700 nanometers (or 0.4 - 0.7 micrometers). One cm equals 10,000,000 nanometers. The longest visible wavelength is red and the shortest is violet.

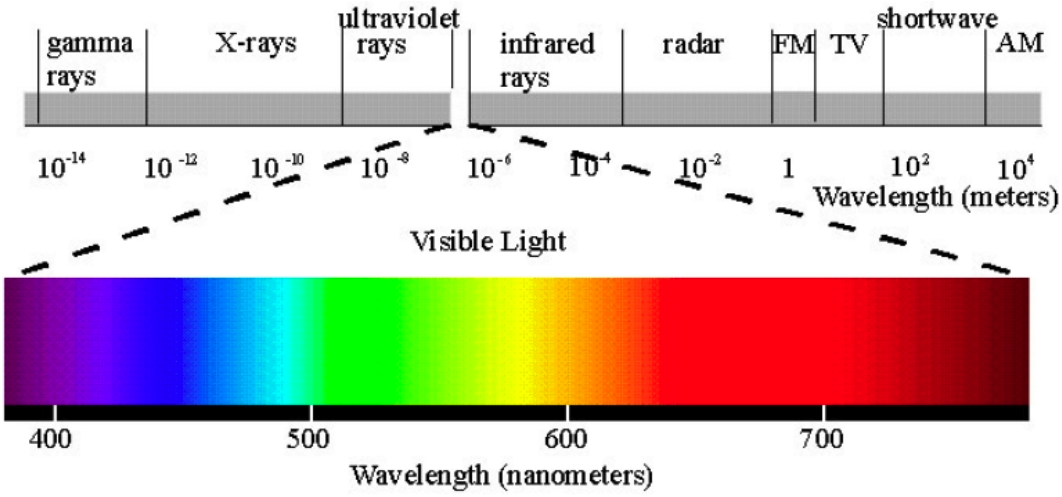



*Figure 2.10. The Electromagnetic Spectrum consists of many wavelengths of light that our eyes cannot detect. Instruments have been built to detect some of these higher and lower wavelengths and are important remote sensing tools. For our purpose, we are only interested in the colour that our eyes can detect. This is the visible light part of the spectrum.*

The colours in the electromagnetic spectrum are often referred to as ROYGBIV (red, orange, yellow, green, blue, indigo, violet) based on optics experiments conducted by Sir Issac Newton in 1666. Newton passed a beam of sunlight through a glass prism and found that the white light separated out into a spectrum of these colours. White light consists of all colours in the visible spectrum. Black is the absence of all these colours.

Blue, green and red are what we term the primary colours. Primary colours are not a fundamental property of light but are related to the physiological response of the eye to light. The reason that we only define three primary colours (as opposed to ROYGBIV, for example) is that the human eye contains only three types of colour receptors, each corresponding to different ranges of the colour spectrum, namely blue (400-500 nm), green (500-600 nm) and red (600-700 nm). These ranges vary slightly from person to person and vary greatly between species. Mixtures of these primary colours form other colours (see RGB Colour Model below). Species with a different number of colour receptors to our own will have a different number of primary colours. For example, some birds and marsupials have four colour receptors therefore, they would define four primary colours that could be mixed to create all the colours that they see. Most mammals (including dogs) have only two types of color receptors and so only need two primary colors to make the range of colours that they see.

### *Colour Models*

Colours can be defined numerically by defining a colour space or colour model. A colour model is defined as an abstract mathematical model describing the way colours can be represented as three or four values or colour components. Examples of colour models include CMYK (Cyan, Magneta, Yellow, Black), RGB (Red, Green, Blue) and HSV (Hue, Saturation, Value). There are fundamentally two different types of colour models (Figure 2.11):

- ☑ Additive colour models which "create" colour through the mixing of light
- ☑ Subtractive colour models which "create" colour through the mixing of paint or pigments

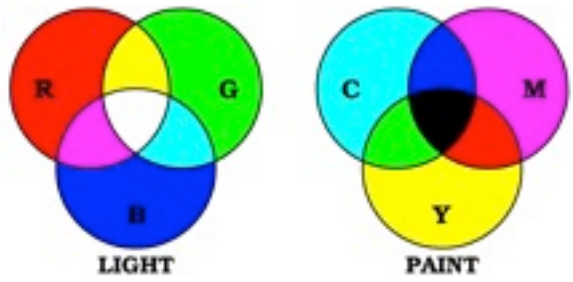



*Figure 2.11: Computer monitors use light to create colour (RGB system, left) whereas printers use the primary colours of pigment to create colour (CMYK system, right). This explains why you sometimes print out an image and wonder why the colours are different to what you observed on your computer monitor.*

***RGB Colour Model***

The RGB (red, green, blue) colour model is based on the three primary colours of light. Colours are formed through the mixing of light and consists of three components (red, green and blue). It is known as an additive colour model because red, green and blue light are added to produce colours.

Figure 2.12 displays the RGB colour model in 3 dimensional space within an orthogonal cartesian coordinate system. Here, the horizontal x-axis is red increasing to the right, the horizontal y axis is green increasing to the top and the vertical z axis increasing up is blue. The origin (xmin, ymin, zmin) is black and white is (xmax, ymax and zmax). The diagonal defines the location in space where there are equal values of red, green and blue and this is called the grey axis. The 8 cube corners define the colours of red, green, blue, white, cyan, magenta, yellow and black.

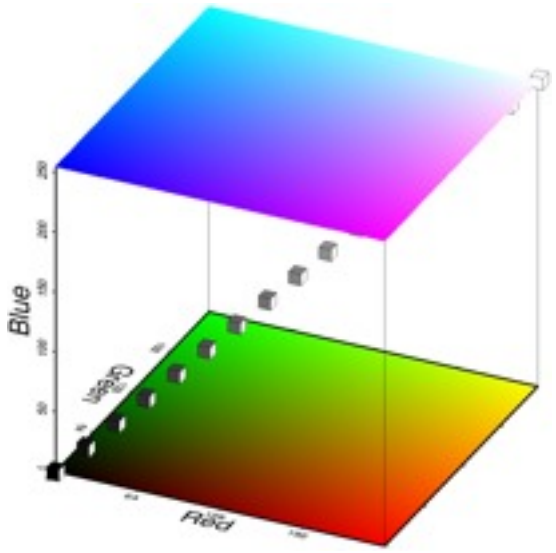



*Figure 2.12: The RGB colour model defined in three dimensional space. An orthogonal coordinate system on a basic level just means that the surfaces intersects the others at right angles. Only Cartesian coordinates have orthogonal coordinate systems.*

A color in the RGB color model is described by indicating how much of each of the red, green, and blue is included. The color is expressed as an RGB triplet (r,g,b), each component of which can vary from zero to a defined maximum value. If all the components are at zero the result is black (i.e. no light). If all are at maximum, the result is the brightest representable white (i.e. all light).

The values of light are measured as intensity. The ranges of these values can be defined in a variety of ways. For example from 0-1, as a percentage or as an integer number in the range of 0-255 or 0-65535 (depending on the computer you are using). In GMT, values from 0-255 are used to define colour in the RGB system.

☑ Red is defined as 255/0/0 corresponding to all (maximum) red, no green and no blue.

☑ Green is defined as 0/255/0 corresponding to no red, all (maximum) green and no blue.

☑ Blue is defined as 0/0/255 corresponding to no red, no green and all (maximum) blue).

Various shades of red, green and blue can be defined as decreasing amounts of red, green and blue, respectively (Figure 2.13). For example, 50/0/0 would correspond to a cherry red. The less colour (i.e. the lower the value), the darker the colour. The more colour (i.e. the higher the value), the lighter the colour. To create colours that are not a shade of the primary colours, you simply add colours together. For example, yellow is defined as equal quantities of red and green (Figure 2.13), magenta as equal quantities of red and blue (Figure 2.13).

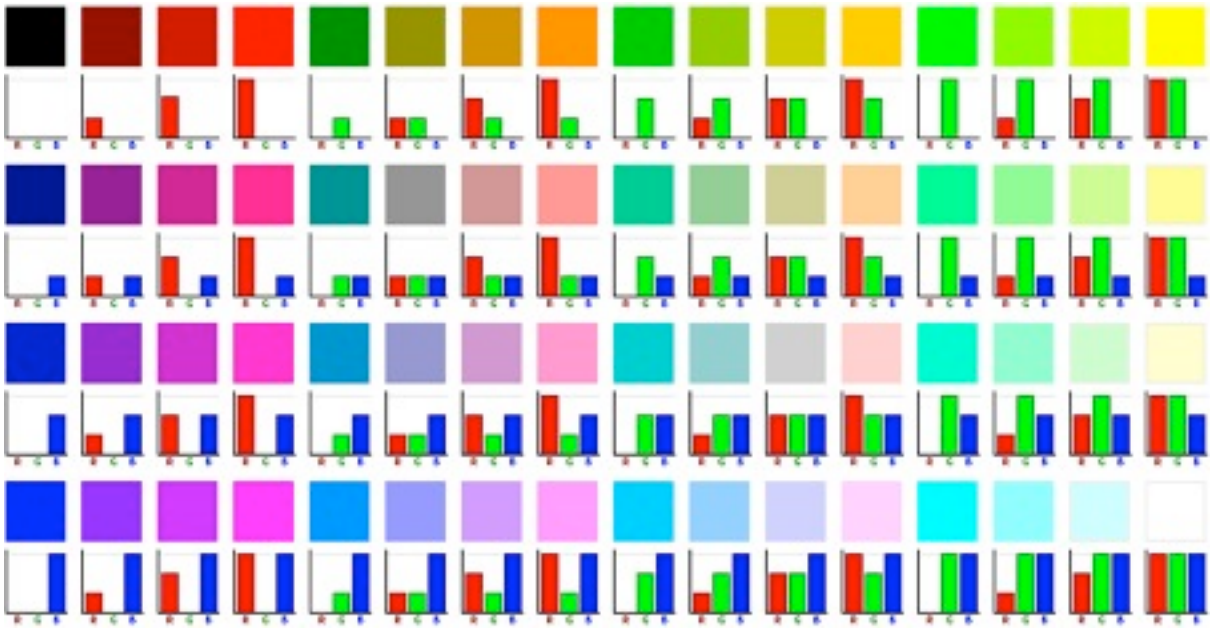


*Figure 2.13: Colours and their equivalent RGB proportions.*

***CMYK Colour Model***

In the previous sections, our definition of colour was in the context of light but colour can also be represented as paint, dyes or pigments. In the CMYK system, a wide range of colors can be created through mixing the primary colors of pigment (cyan (C), magenta (M), yellow (Y), and black (K)). Think of the CMYK system as what you used in art class.

Unlike the RGB system where colours are combined to make new colours (additive colour model), CMYK is a subtractive colour model where a colour is taken away (or masked) to create a new colour. For example, if we mix equal portions of blue, green and red paint, we get a really dark grey colour instead of white (Figure 2.14). If we projected white light onto a translucent filter made of yellow dye, the filter would subtract the blue light and allow the green and red light to pass through. If we used a magenta, cyan and yellow filter, they would filter out all the white light and we would be left with black. Think of the CYMK model as subtracting brightness from white (Figure 2.14).

*Figure 2.14: CMYK. All colours create black.*
*http://vladek.ehion.com/xchange/hsi/images/cmyk.jpg Accessed 30th March, 2010.*

In additive colour models such as RGB, white is the "additive" combination of all primary colored lights, while black is the absence of light. In the CMYK model, it is the opposite: white is the natural color of the paper or other background, while black results from a full combination of colored inks. Printers and painters use the CMYK colour model because there is no way of "printing" light.

To create a three-dimensional representation of a color space in the CMYK model, we can assign the amount of magenta color to the representation's X axis, the amount of cyan to its Y axis, and the amount of yellow to its Z axis. The resulting 3-D space provides a unique position for every possible color that can be created by combining those three pigments. However, when printing, the 'black' created by mixing Cyan, Magenta and Yellow is unsatisfactory, so black ink is used in addition.

***HSV Colour Model***

The HSV colour model stands for Hue, Saturation and Value (value being the brightness). It is a cylindrical-coordinate representations of points in an RGB color model (Figure 2.15) i.e. it rearranges the geometry of RGB from a 3D cube into a 3D cylinder. The HSV colour model is sometimes preferred to the RGB colour model because it does a better job of handling shading or illumination. Hue is represented on the x-axis, saturation on the y-axis and value on the z-axis.

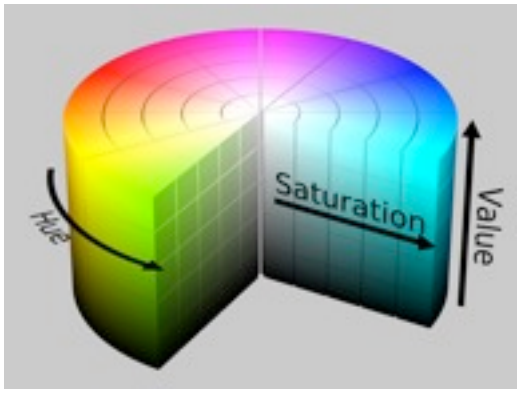


*Figure 2.15: The HSV colour model mapped to a cylinder. http://en.wikipedia.org/wiki/File:HSV_color_solid_cylinder_alpha_lowgamma.png Accessed 30th March, 2010*

Hue is the visual sensation according to which an area appears to be similar to one of the perceived colours and corresponds to the angular dimension, starting at the red primary at 0°, passing through the green primary at 120° and the blue primary at 240°, and then wrapping back to red at 360° (Figure 2.15, 2.16 and 2.17).

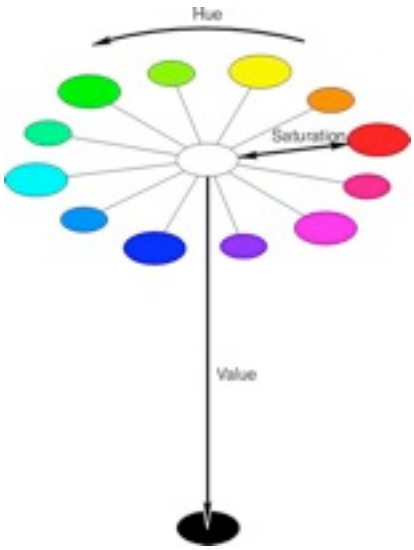



*Figure 2.16: Hue can be thought of as the colour on a colour wheel. In other colour models, hue is the only dimension of colour.*

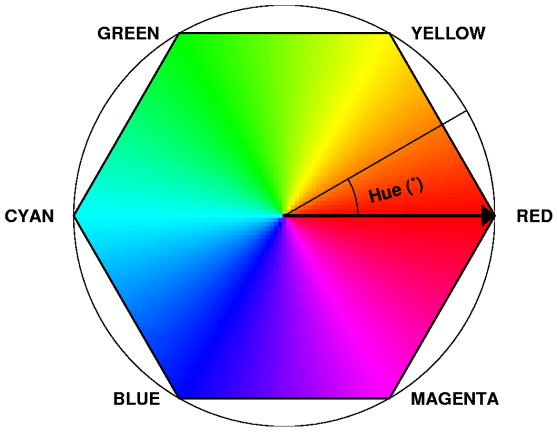



*Figure 2.17: Hue can be thought of as an angle from 0 to 360°. If we project the RGB colour cube onto the "chromaticity plane" perpendicular to the neutral axis, our projection takes the shape of a hexagon, with red, yellow, green, cyan, blue, and magenta at its corners. Hue is roughly the angle of the vector to a point in the projection, with red at 0°.*

Saturation is the “colorfulness of a stimulus relative to its own brightness” or can be thought of as a measure of the purity or vividness of colour. On the outer edge of the hue wheel are the 'pure' hues. As you move into the center of the wheel, the hue we are using to describe the color dominates less and less (Figure 2.15, 2.18 and 2.19). When you reach the center of the wheel, no hue dominates. The central vertical axis comprises the neutral, achromatic, or grey colors and range from black at 0 (the bottom) to white at 1 (the top).

In the HSV colour model, mixing pure colors with white (producing “tints”) reduces saturation, while mixing pure colours with black (producing shades) leaves saturation unchanged.

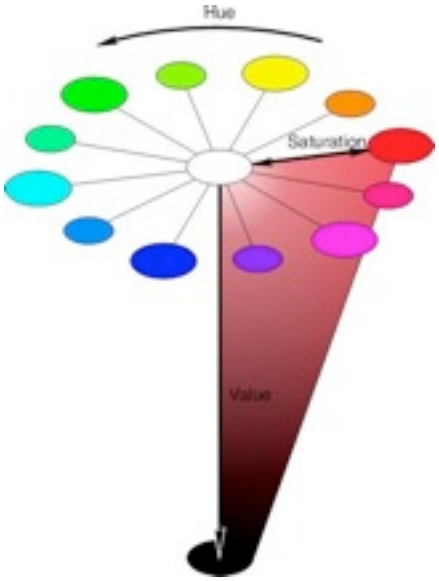



*Figure 2.18: Saturation refers to the dominance of hue in a colour. This example shows a slice along the saturation axis at a red hue. Notice that the vividness of red decreases as we move to the centre of the colour wheel.*

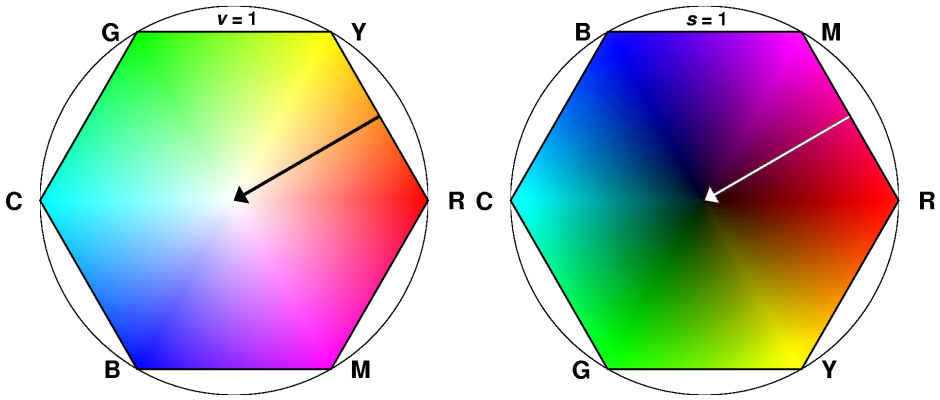



*Figure 2.19: Colour can be lightened by moving towards white or decreasing the saturation (left) and darkened by moving towards black or decreasing the value (right). Hue is constant.*

Value is a measure of the strength of the hue, whether an area appears to emit more or less light (i.e. its lightness-darkness). The value is the linear axis running through the middle of the wheel (Figure 2.15, 2.18 and 2.19).

In GMT, we must transform from using the RGB colour models to the HSV colour model when we are dealing with gridded data that have an illumination. The reason is that when we add illumination or intensity using an RGB colour model, the hue is changing as a result. The HSV system allows for illumination to be handled more naturally.

# 3. Historical View of Spatial Data

The representation of spatial data has its origins in pre-historic times when humans discovered a method of transposing the 3D images observed in the physical world (e.g. stars, animals, mountains, rivers) onto a 2D surface (e.g. cave walls, sand/gravel, tree trunk). Some of the earliest known spatial representations are over 35,000 years old preserved on cave walls across Europe, Africa and Australia. In Europe, these representations are of bulls, bison, stags, deer and horses in various states of motion some with associated track lines and tallies believed to depict the migration routes of these animals (Figure 3.1). The "mapping" of these migration routes forms one of the earliest examples of spatial data, well before the time of maps and coordinate systems.

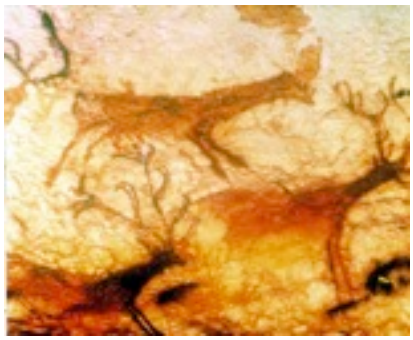


*Figure 3.1: Paintings of stags and their possible migration routes painted on the walls of a cave in Lascaux, France by Cro-Magnon hunters. It is believed that these paintings are one of the earliest known examples of spatial type data.*

Other examples of pre-historic spatial data are human hunting trails, signs and monuments such as stone circles (Figure 3.2), temples and burial mounds.

*Figure 3.2: Stonehenge in Southern England is a famous example of a neolithic stone circle. Some believe that it represents celestial configuration.*

With the birth of civilisation, spatial data were collected and became increasingly important to ancient navigators, astronomers, surveyors and geographers. The spatial data were then recorded by map-makers and cartographers in picture form. The first known

"map" of the world is preserved in an ancient Babylonian clay tablet from southern Iraq dating to around 700-500 BC. In addition to containing inscriptions and pictures of mythological beings, the tablet contains a unique map of the Mesopotamian world (Figure 3.3) with Babylon at its centre and its neighbours (Assyria, Armenia and other city states) surrounding it. This central region itself is surrounded by an ocean. Although the map correctly depicts the spatial locations and relationships between different locations, its purpose was as a map of the mythological world. It was not a map drawn to scale and there was no geo-referencing.

*Figure 3.3: The earliest known "world" map showing Babylon along the Euphrates River surrounded by its neighbours and an ocean. This clay tablet is currently housed in the British Museum.*

Another contender for the world's earliest map is a cartographic artifact found in 1963 in Çatalhöyük, Turkey dating to 6,200 years ago. The map is a wall painting believed to depict a town plan (Figure 3.4).

*Figure 3.4: An earlier contender for first map showing a town in modern day Turkey. The town plan and settlement is seen as black rectangles, a volcano which overlooks the town is seen in the background.*

Cartography/map-making as we know it was born with the ancient Greeks. The ancient Greek city of Miletus was a centre of Greek philosophy and scientific research starting around 600 BC. It was here that Thales, with his students Anaximander and Hecataeus, started to formulate ideas on the world outside of their borders, what the earth looked like and drew early maps of the known world. For more information on the earliest ideas on the shape of the earth, see Historical Note on the Shape of the Earth in the Geodetics section.

Anaximander was the first ancient Greek to draw a map of the known world. It is for this reason that he is considered by many to be the world's first mapmaker. Although the maps of Anaximander do not survive, Hecataeus of Miletus produced a map 50 years after Anaximander that he claimed was an improved version of the map of Hecataeus (Figure 3.5). Around -450 BC, Herodotus produced a map of the known world which was largely based on Anaximander's original map.


*Figure 3.5: A depiction of Anaximander's first world map based on a map by Hecataeus of Miletus. http://en.wikipedia.org/wiki/File:Anaximander_world_map-en.svg Accessed 21st February, 2010.*

Great advances in cartography occurred towards the end of the Greek Golden Age primarily in the city of Alexandria in present day Egypt by two main mathematicians: Eratosthenis of Cyrene and Ptolemy (Figure 3.6).

*Figure 3.6: The founder of geography, Eratosthenes (left) and the greatest of all geographers, Ptolemy (right)*

Eratosthenes was an ancient Greek mathematician, astronomer, geographer and head librarian at the Great Library of Alexandria around 250BC and is often credited with being the founder of geography. His main contributions to the field of geography was to devise the concept of a map grid (an early form of latitude and longitude), to incorporate meridians and parallels on maps and he was the first to measure the circumference of the Earth with great accuracy (see Historical Note on the Shape of the Earth in the Geodetics section to find out how he did this). A contemporary of Eratosthenes, Hipparchus of Nicaea, was critical of the map grid defined by Eratosthenes suggesting that his grid points were chosen arbitrarily. Instead Hipparchus designed a grid with astronomical significance and further developed the geographic coordinates (longitude/latitude) for describing geographic positions.

Claudius Ptolemaeus (more commonly known as Ptolemy) (Figure 3.6) was a Roman citizen of Greek ancestry who lived between 90-168 AD. He made the last major contribution by the Greeks to cartography. He is often termed the greatest ancient western scholar of geography and cartography and was a student at the Great Library of Alexandria well after the time of Eratosthenes. Ptolemy was most famous for his work on astronomy and his book the Almagest. However, he also produced an eight volume work called the Guide to Geography which included the theory and practical use of map projections, globe construction, 8000 place name with latitudes and longitudes, instructions for making maps and discussions on mathematical geography. Ptolemy's work also produced a map of known world from about 60°N to 30°S latitudes (Figure 3.7).

*Figure 3.7: A 15th century copy of the Ptolemy world map. Ptolemy's maps were distorted because he*

*incorrectly measured the circumference of the earth (assumed it was much smaller)*
*http://en.wikipedia.org/wiki/File:Claudius_Ptolemy-_The_World.jpg Accessed 25th February, 2010*

Ptolemy's works were so influential that any mistakes that Ptolemy made were accepted even though previous knowledge suggested otherwise. One such example is the circumference of the Earth. Ptolemy did not use the accurate measurement of Eratosthenes and instead underestimated the size of world based on later work. This had a lasting influence on global cartography (for example, it affected Christopher Columbus's world view). Ptolemy's Guide to Geography remained the authoritative reference on world geography until the Renaissance.

The Romans continued the Greek tradition of map-making but were less interested in mathematical and theoretical concepts. Instead the Romans focused on the military and administrative applications of map-making to manage their growing empire and new conquests. Land surveyors were an important part of the government and the results of their work can still be seen in the landscapes of Europe today. The Roman road network and series of road maps are also famous the world over (Figure 3.8). Many Roman roads still exist today throughout all of Europe. The fall of Rome curtailed geographic though and research.

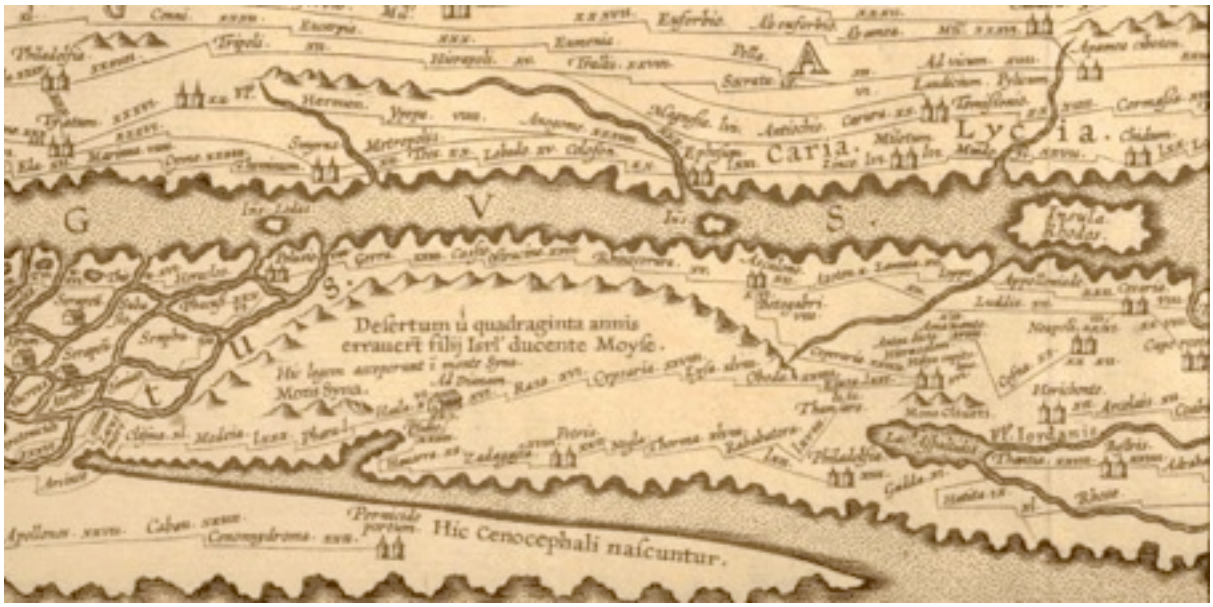



*Figure 3.8: A medieval copy of a 4th century revision of an original Roman map of the Roman road network around Jerusalem. North to the right. http://homepages.luc.edu/~avande1/jerusalem/~views/romanRoadMap.htm Accessed 14th March, 2010.*

There is evidence of Chinese surveying and cartography from the 4th century BC which developed independently from the ancient Greeks and Romans. The main driving force in China to survey and draw maps was often for military reasons but also to help solve problems such as water conservancy. Extensive maps of the river systems in China, population and military movements were common. Maps were often preserved on silk screen

or wood blocks providing a greater chance of survival than the ancient maps of the western world.

Ptolemy's Guide to Geography was translated into Arabic in the 9th century ensuring that a lot of the knowledge of the ancient Greeks and Romans was preserved. They made improvements to much of Ptolemy's work using information obtained from the extensive explorations of the Old World and developed many new map projections.

During the Medieval period, European maps were dominated by religious views and the maps were mainly symbolic rather than mathematical (Figure 3.9). Theologians argued that the truth about the world was contained in the Bible and not in the foundations of science. Good science was often dismissed when it appeared to contradict Bible quotations. The idea of a flat earth was advanced during this period but did not truly take hold.


*Figure 3.9: An example of a 16th century symbolic religious map, the Bünting clover leaf map. Jerusalem is in the center, surrounded by Europe, Asia and Africa and a vast ocean.*
*http://en.wikipedia.org/wiki/File:1581_Bunting_clover_leaf_map.jpg Accessed 15th March, 2010.*

The Renaissance was a cultural movement that started in present day Italy in the 14th century and spread throughout Europe. At its core was a rediscovery of classical philosophy and ideas including the knowledge of the ancient Greeks and Romans about geography, the shape of the Earth and map projections. This era also coincided with the age of discovery and exploration in Europe which began in the 15th century. The main motivation to improve cartography during this period came with the discoveries of new lands initially made by Portuguese explorers and later by Spanish, Dutch and English explorers. Some of the most famous of these explorers were Christopher Columbus because of his "discovery" of America and the "New World", Ferdinand Magellan and Sir Francis Drake

for their circumnavigation of the world and Vasco da Gama for his discovery of a route to the east from Europe.

*Figure 3.10: The Cantino planisphere (1502) is the earliest surviving map depicting Portuguese Discoveries in the east and west. http://en.wikipedia.org/wiki/File:CantinoPlanisphere.png Accessed 15th March, 2010*

As time progressed, better navigational knowledge, instruments and other advancements led to an increase in mapping accuracy and further exploration. One of the most important advances in the history of cartography was the invention of the printing press in 1440. This made maps much more widely available to the early explorers.

The first whole-world maps began to appear in the early 16th century, following the voyages of Columbus and others to the New World. The first true world map is generally credited to the German cartographer, Martin Waldseemüller in 1507 (Figure 3.11). Waldseemüller's map used an expanded Ptolemaic projection and was the first map to include the name America for the New World.

*Figure 3.11: . The Waldseemüller wall map printed in 1507*
*http://en.wikipedia.org/wiki/File:Waldseemuller_map_2.jpg Accessed 15th March, 2010*

During this period, skilled cartographers emerged leading to the development of better map projections. Gerardus Mercator (1512–1594) was a Flemish cartographer who developed new map projection (called Mercator projection) using mathematical formulas that was very popular amongst navigators because of the ease of measuring direction and distances. His Mercator projection, produced in 1569 has become a conventional view of the world that is still in use today.

The major advance in cartography to take place in the 1600s was to determine a better model for the shape of the world. There was a fierce debate between Sir Issac Newton who believed the earth was shaped like an oblate spheroid and the French Academy of Sciences who believed in a prolate spheroid (see Historical Note on the Shape of the Earth under Geodetics to find out who was right!).

The 17th and 18th centuries was the Golden Age of English navigation and exploration. This was further helped by the invention of the chronometer by John Harrison in the late 1700's for increased accuracy of longitude determination. The determination of longitude at sea had a great influence on mapping positions as well as aiding exploration and discovery. Many English explorers were active during this time. The most famous explorer in the context of Australia's history was Captain James Cook.

The 18th century saw the rise of scientific cartography based on more accurate observations and further map projection skills were developed. The need for agreed standards on spatial phenomena was recognised to provide a better model for land surveys, human settlement, navigation at sea, taxation, military strategy and an easier way to control vast empires. This led to many countries initiating national surveys but using different stan-

dards. For example, the coordinate system had different reference points for each country. In 1884, the International Meridian Conference was held in Washington D.C. (Figure 3.12) with the aim of standardising the origin of the coordinate system. It was only at this conference that the Greenwich Meridian was adopted as the zero for longitude by all countries.

*Figure 3.12: Participants at the International Meridian Conference, Washington DC, 1884. http://www.phys.uu.nl/~vgent/idl/idl_imc1884.htm Accessed 15th March, 2010.*

Significant developments in cartography were made during the late 19th and earliest 20th centuries The invention of photography in 1839 led to the development of aerial photography by the French photographer and balloonist Gaspard-Félix Tournachon in 1858. He took the world's first aerial photograph over Paris, France giving a bird's eye view of the Earth for the first time (Figure 3.13). The widespread use of aerial photography and photogrammetry particularly during World War I and II enabled a great picture of the world to emerge. Aerial photography is still used today and is an important mapping tool.

*Figure 3.13: The world's first aerial photograph, Paris, France. http://pablo.enlapc.com/wp-content/uploads/2008/07/primer-foto-aerea.jpg Accessed 15th March, 2010.*

The Cold War led to a new emphasis on military mapping and standardisation. Standard NATO symbols for military maps and charts, a global model of the shape of the Earth, a global datum and the Global Positioning System (GPS) were all developed as a response

to the military threat of the Cold War. The Cold War also led to the need to spy on enemies and rapid developments in satellite observation technologies were developed. Satellites were also crucial for taking geophysical measurements of the earth (Figure 3.14).

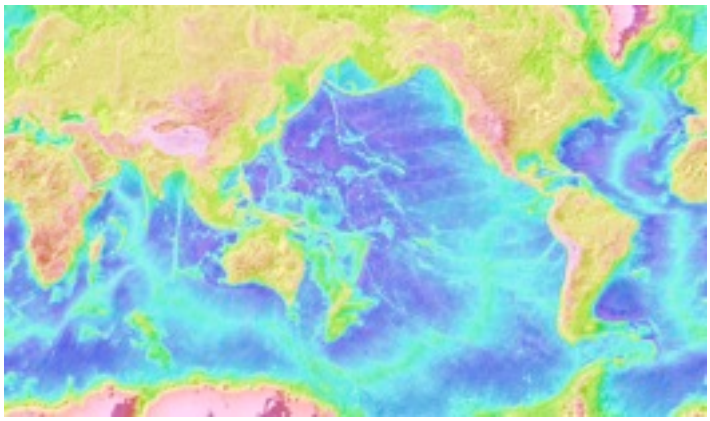


*Figure 3.14: Gravity anomalies from satellite altimetry. Two examples of the use of these data include: to better target missiles by assisting in determining a better model for the shape of the Earth and, to determine submarine "highways" and "parking bays" for submarines by producing a bathymetric map of the ocean floor. These data are now publicly available and use as a primary tool in plate tectonics. http://www.ngdc.noaa.gov/mgg/image/walter/World.jpg Accessed 15th March, 2010*

Geographic information systems (GIS) emerged in the 1970-80s as a significant new leap forward in cartography and the display and analysis of spatial data. Modern cartography involves these tools as well as combining ground observations and remote sensing techniques.

# 4. Coordinate Systems

## Cartesian Coordinate System

The Cartesian coordinate system is the x-y coordinate system and sometimes called the rectangular coordinate system. It is used to denote the unique location of a point along a plane through two numbers -the x value and y value. In the diagram below (Figure 4.1), the point (x,y) is defined by the distance relative from two perpendicular axes. The distance from the origin (0,0) to point (x,y) can be calculated using Pythagoras' Theorem.

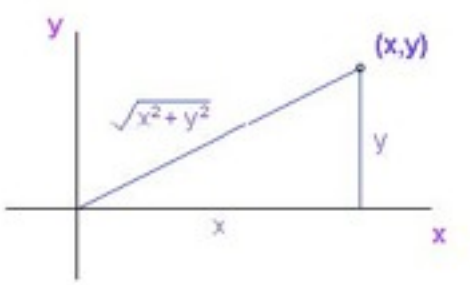



*Figure 4.1: 2D Cartesian coordinate system.*

The 3D Cartesian coordinate system provides three physical dimension of space, namely length, width and height (Figure 4.2). In the diagram below, the point (x,y,z) is defined by the distance relative from three perpendicular axes. Note: Look up your high school or 1st year textbooks for more information on 3D coordinate systems.

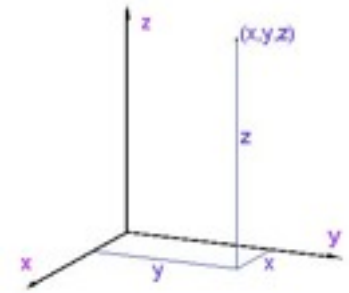



*Figure 4.2: 3D Cartesian coordinate system. The point (x,y,z) is defined by the distance of x, y and z from their respective axes.*

In cartography and surveying, the X axis coordinates are known as *Eastings* (aligned E-W), and the Y axis coordinates as *Northings* (aligned N-S). The Z axis coordinate is known as the height.

### *Geographic Coordinate System*

To understand the geographic coordinate system, we must first describe the Polar and Spherical coordinate systems.

The Polar coordinate system is a way of representing points in space. Instead of moving up or down and side to side from the origin along the axes as we do in the Cartesian system, we instead move at an angle from the pole (origin) and at a distance to a point (Figure 4.3). The angle from the pole (origin) is θ and the radial distance from the pole is r.

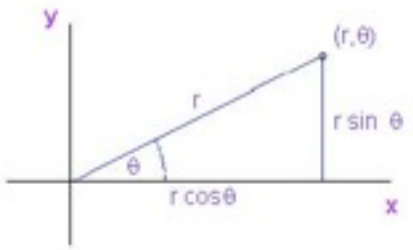



*Figure 4.3: 2D Polar coordinate system. The x axis is called the polar axis.*

When the x-axis of the Cartesian coordinates coincides with the polar axis, the relationship between the Cartesian coordinates (x, y) and polar coordinates (r, θ) is given by:

$x = r\cos\theta$

and

$y = r\sin\theta$

To convert the Cartesian coordinates to polar coordinates, the following relationships are useful:

$r_2 = x_2 + y_2$ and $\tan\theta = y/x$

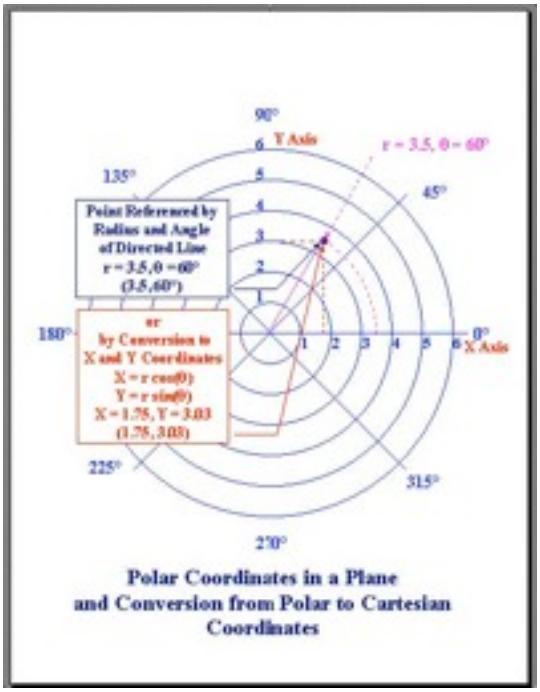



*Figure 4.4: Polar coordinates in a plane and conversion from Polar to Cartesian Coordinates.*

In the example shown in Figure 4.4, we have a point (3.5, 60) defined in the polar coordinate system. This means it has a radial distance of 3.5 and an angle of 60 degrees. To convert this polar coordinate into a Cartesian coordinate, we do the following:

$x = r\cos\theta$ $y = r\sin\theta$

$x = 3.5 \times \cos 60$ $y = 3.5 \times \sin 60$

$x = 1.75$ $y = 3.03$

So the x and y coordinates become (1.75,3.03)

To convert the Cartesian coordinate back into its polar position, we do the following:

$r_2 = x_2 + y_2$ $\tan\theta = y/x$

$r_2 = 1.75^2 + 3.03^2$ $\tan\theta = 3.03/1.75$

$r_2 = 12.24$ $\tan\theta = 1.73$

$r = 3.5$ $\theta = 60$

So the radial distance and angle become (3.5, 60)

Polar coordinates can be extended into three dimensions to locate positions on a sphere using the spherical coordinate system. The system defines 3 parameters ($\phi$, $\theta$, r):

where;

r = rho = the radial coordinate (the distance from the origin to the point in space)

$\theta$ = theta = the azimuth angle (the angle to the point from the positive x-axis)

$\phi$ = phi = the zenith angle (the angle to the point in space from the positive z-axis)

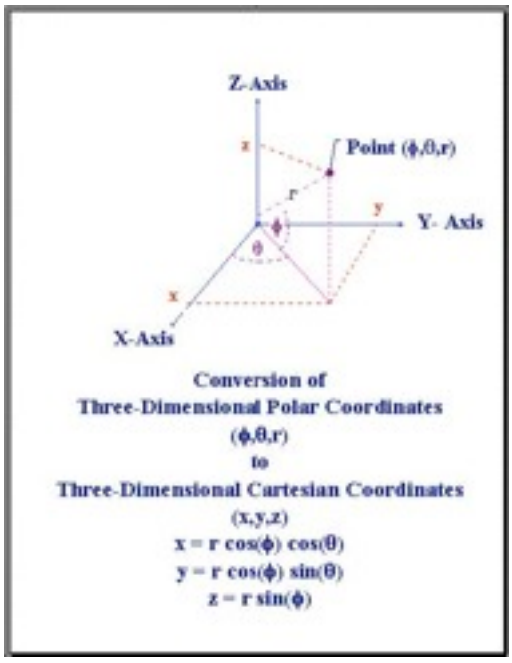



*Figure 4.5: Spherical coordinate system where the point P is defined by (ϕ, θ, r). The equivalent Cartesian coordinate system is also displayed (x,y,z).*

The three spherical coordinates are converted to Cartesian coordinates by:
x = r cos Φ cos θ (projection of r on the xz plane)
y = r cos Φ sin θ (projection of r on the yz plane)
z = r sin Φ (equivalent to the opposite side of the triangle with r as the hypotenuse and Φ as the angle)

The geographic coordinate system is a derivation of the spherical coordinate system, used mainly in geography, to express locations on Earth as three coordinates.

In the spherical coordinate system, a point in space is defined by (ϕ, θ, r). In the geographic coordinate system, this is replaced with (latitude, longitude, distance) (Figure 4.6). The origin of the geographic coordinate system is defined as being the centre of the Earth, assuming that the Earth is a perfect sphere. But as we know, the Earth is not a perfect sphere so complications arise.

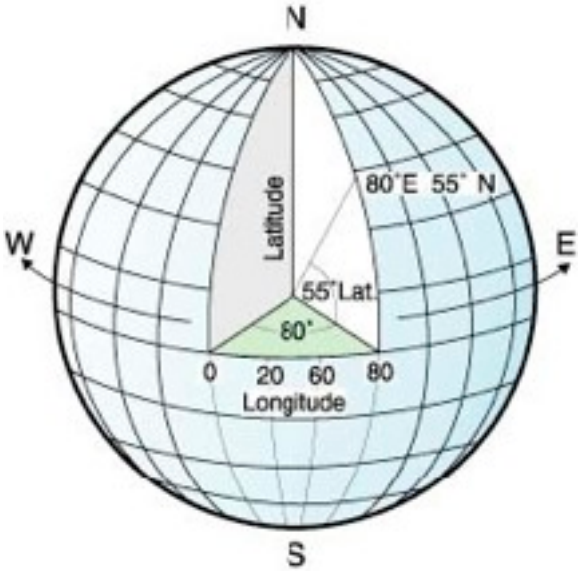



*Figure 4.6: Geographic coordinate system where the point P on the surface of the Earth is defined by a longitude and latitude - in this example 80°E and 55°N. The distance is just the radius of the Earth.*

Spherical coordinates can be converted to latitude and longitude (Figure 4.7). In spherical coordinates we measure the angle phi from the north pole. Thus, the north pole corresponds to phi = 0; the equator to phi = 90 degrees; and the south pole to phi = 180 degrees.

Thus, the following formula converts from latitude expressed in degrees to phi also expressed in degrees.

phi = 90 - latitude if latitude is in the Northern Hemisphere

phi = 90 + latitude if latitude is in the Southern Hemisphere

In spherical coordinates we measure the angle theta starting at the prime meridian (longitude 0) and moving east. Thus

theta = longitude if longitude is East

theta = -longitude if longitude is West

This is fine if both phi and theta as described above are measured in degrees. However, it is mathematically much better to measure angles in radians. The conversion formula is

angle in radians = ( angle in degrees * 2 * Pi ) / 360

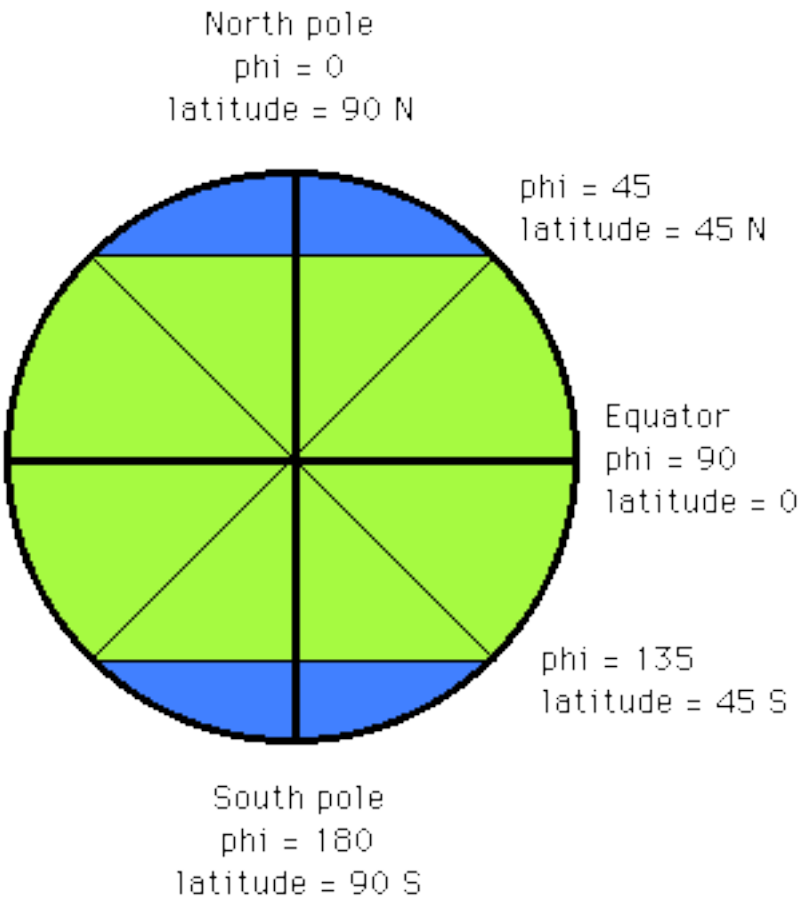



*Figure 4.7: Conversion from spherical coordinate phi to latitude.*

Latitude (ϕ) and longitude (θ) are coordinates that represent positions with angles instead of distances. The angles are measured in degrees (°) which are divided into 60 minutes of arc or minutes (') and 60 seconds of arc or seconds ("). There are 3600 seconds in a degree. Lines of latitude and longitude always cross each other at right angles. You may find that maps which depict a flat surface use angles as measurements instead of x,y coordinates. More on that in the Map Projections Section.

### *Latitude*

The latitude is the complement of ϕ (or the zenith angle or colatitude) in the spherical coordinate system. It represents the zenith angle originating from the xy plane with a domain -90° ≤ ϕ ≤ 90°. It is commonly denoted by the Greek letter ϕ.

The equator is the reference plane used to define latitude and is at 0°. The North Pole is at +90° or 90°N and the South Pole is at -90° or 90°S. Latitude is an angular measurement of the distance of a particular point north or south of the plane (equator) (Figure 4.6). When the latitude angle is measured from the centre of the Earth, this measurement is called the *geocentric latitude* and assumes a spherical Earth. However, as we will see in the next section, the Earth is not a perfect sphere and often the latitude measurements are not calculated from the centre of the Earth. Instead, latitude is often meas-

ured from the *geographic* or *geodetic latitude*. The geodetic latitude of a point is the angle from the equatorial plane to the vertical direction of a line normal to the reference ellipsoid (Figure 4.8). All this means is that the latitude is measured normal to the reference surface of the Earth. It is always based on a specific datum or ellipsoid. See Geodetics to understand the need for geodetic values.

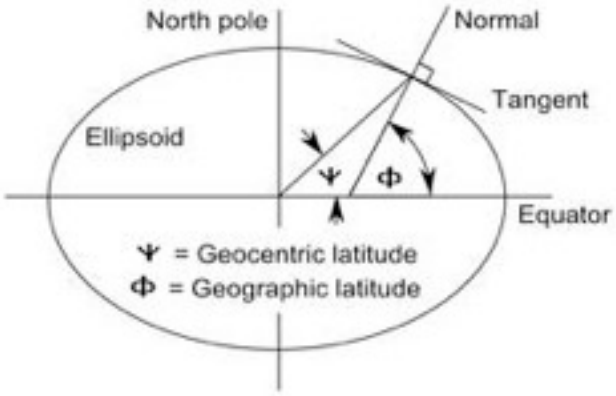



*Figure 4.8: The geographic or geodetic latitude (phi) differs from the geocentric latitude (psi). It corresponds to a point on the equatorial plane normal to the reference ellipsoid.*

The lines that run east and west each have a constant latitude value and are called *parallels* (Figure 4.9). They are equidistant and parallel to one another, and form concentric circles around the earth. The *equator* is the largest circle and divides the earth in half. It is equal in distance from each of the poles.

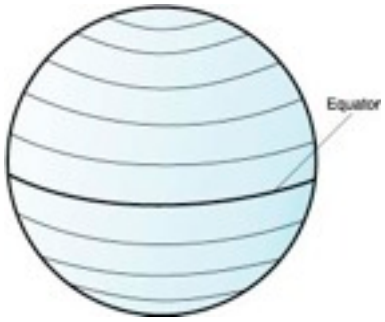



*Figure 4.9: Lines of latitude run East-West and are called parallels.*

### *Longitude*

The longitude is the complement of θ (or the azimuth angle) in the spherical coordinate system and is measured in degrees east and west from 0° within the domain -180° ≤ θ ≤ 180° or east from 0° within the domain 0° -360°. It is commonly denoted by the Greek letters θ or λ.

While latitude uses the midpoint between the two poles, the equator, as a base line or reference plane that separates the Earth into two equal parts, there is no naturally obvious reference plane for longitude. In the case of longitude, any longitude and its antip-

ode (the continuation of the prime meridian on the other side of the globe) can act as the reference plane. Longitude is a dihedral angle which means that it takes its measurement from two planes. The first plane passes through the point of interest and the second plane passes through a selected point denoting zero longitude (prime Meridian).

In 1884 at the International Meridian Conference in Washington D.C., the line of longitude that passes through the observatory in Greenwich, England was declared the Prime Meridian from which all longitude measurements would be taken (Figure 4.10). This was an essential agreement (but not without much bickering and debate) because up until then each country used its own prime meridian to make maps, calculate distances and navigate the world's oceans. Other longitude lines that were used previously included those that pass through Amsterdam, Bern, Bogota, Brussels, Copenhagen, Lisbon, Madrid, Munich, Rome, Paris, Stockholm, Warsaw and Washington D.C. The antipode of the Prime meridian roughly corresponds to the international dateline.

*Figure 4.10: Picture of the prime Meridian (National Maritime Museum).*

Locations east of the prime meridian up to its antipodal meridian have positive longitudes ranging from 0 to +180° or 0 to 180°E. Locations west of the prime meridian have negative longitudes ranging from 0 to -180° or positive values ranging from 0 to 180°W. Longitudes can also be measured as positive values from 0° to 360° but this is not common practice in map-making.

The *geodetic longitude* is the same as *geocentric longitude* because they share the same reference meridian and axis in contrast to latitude.

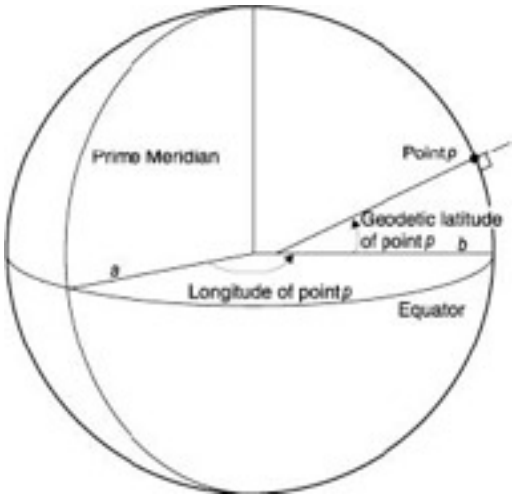



*Figure 4.11: The geodetic longitude is the angle in the equatorial plane between the line a that connects the Earth's center with the prime meridian and the line b that connects the center with the meridian on which the point lies.*

Lines of longitude run north and south and are called *meridians* (Figure 4.12). They form circles of the same size around the earth, and intersect at the poles. They are not equidistant and parallel to each other but converge at the poles. The *prime meridian* is the line of longitude that defines the origin (zero degrees) for longitude coordinates.

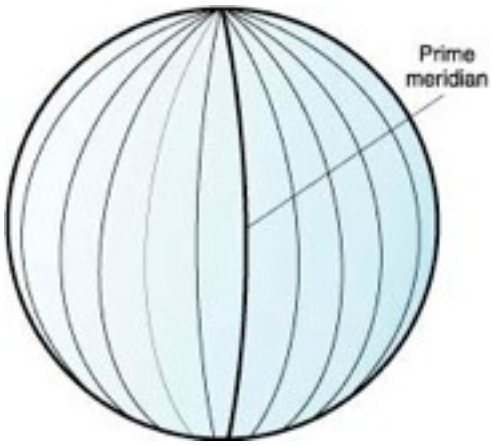



*Figure 4.12: Lines of longitude run North-South and are called meridians.*

### *Determining Longitude*

As the Earth turns on its rotational axis, it moves through 360° every 24 hours. The Greenwich meridian is used as a reference line from which to calculate any time difference between the time at Greenwich and the time at the location of interest. Since the earth rotates through 15° of angle or longitude per hour (360/24 hours = 15° per hour), longitude can be determined each day at noon.

For example, a ship sets sail west across the Atlantic Ocean, checking its longitude each day at noon (where the sun crosses the meridian of the observer directly overhead). One day when the sun is at the noon position, the captain checks the chronometer. It reads

16:18 hours. What is the ship's longitude?

16:18 hours = 4:18 pm

Earth rotates through a quarter of a degree (15 minutes) per minute of time

(one degree of arc is divided into 60 minutes)

4 hours x 15°/hr = 60° of longitude

18 minutes of time x 15 minutes of angle/minute of time = 270 minutes of arc

270 minutes / 60 minutes / degree = 4.5°of longitude

60°+ 4.5°= 64.5° W longitude

### *Measuring Distance*

Lines of latitude and longitude can cover the globe to form a grid, called a graticule (Figure 4.13). The point of origin of the graticule is (0,0), where the equator and the prime meridian intersect.

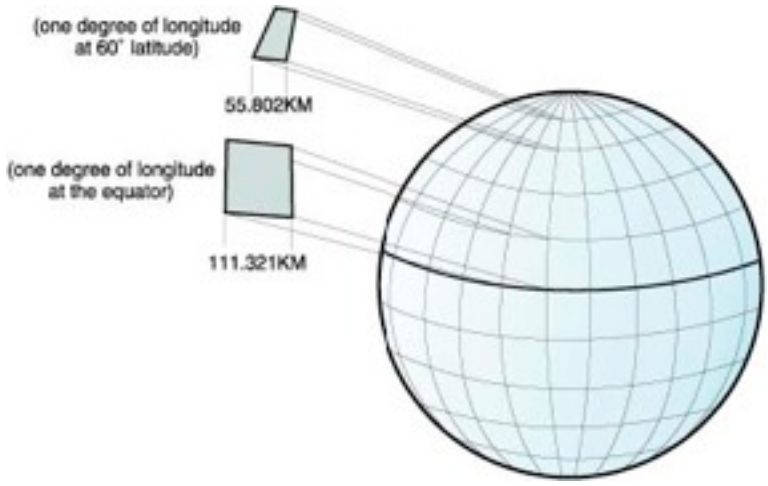



*Figure 4.13: Graticule showing the difference between one degree of longitude at 60 latitude and the equator. Lines of longitude converge at the poles.*

The equator is the only place on the graticule where the linear distance corresponding to one degree latitude is approximately equal the distance corresponding to one degree longitude. One degree of longitude at the equator is equivalent to 1/360 of the circumference of the Earth or a distance of 111.321 km (assuming a spherical Earth) (Figure 4.12).

As you move away from the equator, the distance between two meridians (or the length of a parallel) gets shorter and shorter until it is just a point at the poles. This is because lines of longitude converge at the poles. The degree of shortening is approximately equal to the cosine of latitude or cos ϕ. For example, at 60°N one degree of longitude is equivalent to a distance of 55.802 km (Figure 3.26) (i.e. Distance = 111.321 x cos60).

On a spherical Earth, the shortest path between two points is a great circle or the arc formed if the Earth is sliced through the two points and through its centre (Figure 4.14). It is the path with least curvature and is the largest possible surface that can be drawn from a sphere. Every meridian and the equator are great circles, There are an infinite number of great circles that can be drawn.

*Figure 4.14: A great circle divides the Earth into two equal parts and represents the line of shortest distance.*

Only one great circle will pass through any two specified points on the Earth's surface. The exception is at the antipodes. The practical uses of great circles are include finding the shortest route for ship, air and (unfortunately) missile travel.

The length of this arc or the *great-circle distance* on a spherical Earth of radius R is given by: distance = R arccos [sin$\phi_1$ sin$\phi_2$ + cos$\phi_1$ cos$\phi_2$ cos($\theta_1$ -$\theta_2$)] (See Figure 4.15)

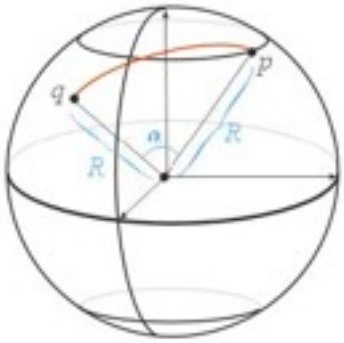



*Figure 4.15: The distance between point q and p (red line) follows a great circle. We must know the radius of the Earth (R), the latitude and longitude of the two points and we must recall the law of cosines*

A *small circle* is any circle produced by planes passing through a sphere anywhere except through its exact center.

# 5. Geodetics

According to Webster geodesy is defined as "that branch of applied mathematics which determines by observation and measurement the exact positions of points and the figures and areas of large portions of the earth's surface, the shape and size of the earth, and the variations of terrestrial gravity." If we want to have a mathematical representation of a point (coordinates of a point in space) we need to have a reference surface that we can refer to. If we have a good approximation for the surface of the Earth, then we can define this surface. Figure 5.1 shows the main components that need to be accounted for in determining the shape of the Earth: the ellipsoid, the geoid and the terrestrial surface.

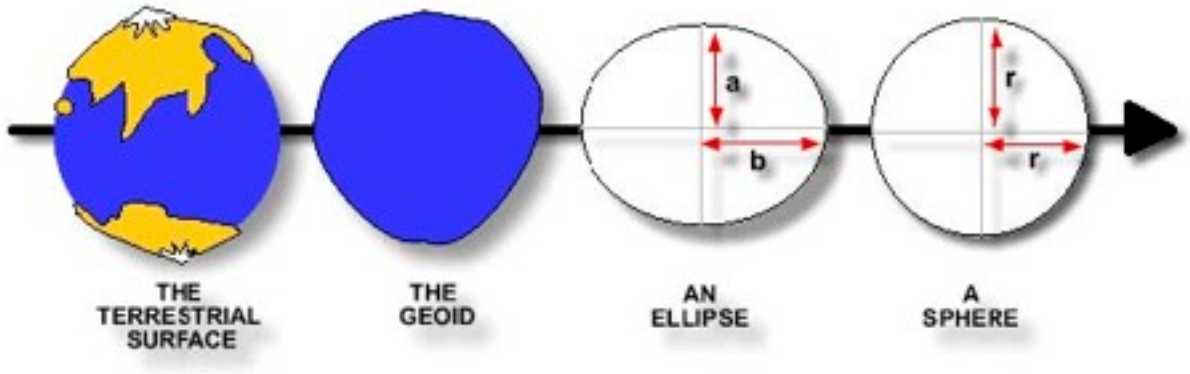



*Figure 5.1: Main parameters that affect the shape of the Earth. Taken from Geoscience Australia.*

## Geometric Earth Models

Ancient cultures had different ideas on the shape and size of the Earth. These were initially based on mythology: some of the more interesting ideas include the Earth as a turtle, an oyster (the Babylonians < 3000BC) and a giant igloo (the Inuits). Initially, the Ancient Greeks, through the mathematician and philosopher Thales of Mileteus, believed that the Earth was disk-shaped floating on water surrounded by a circular stream. The Greek philosopher Anaximander, a student of Thales of Mileteus, is credited with thinking that the Earth was both flat and cylindrical in shape.

The Pythagoreans (a group of philosophers, mathematicians headed by Pythagoras himself) around 500 B.C. were the first to argue that the Earth was a sphere. There is no record of their scientific argument but it may have been an aesthetic decision in part. because they believed that the sphere was the most beautiful shape and therefore, the Earth would have to have followed this form.

Many centuries later, Aristotle also proposed that the Earth was a sphere but unlike the people before him, he developed a proof by observation. He observed that during a lunar eclipse (and in every lunar eclipse recorded and from different locations) the Earth's shadow on the Moon always appears curved. If the Earth were flat or any shape other

than a globe, at least some of those eclipse shadows would be straight or angular. In addition, Aristotle believed that his physics of natural motion proved that the Earth must be round (i.e. all things strive toward their natural place; the natural place of earth is the center of the universe; earth falls toward the center from all sides, earth would naturally compact into a sphere). Other proofs included observations of ships on the horizon.

Once the shape of the Earth was firmly established, the Greeks attempted to calculate the size of the Earth.

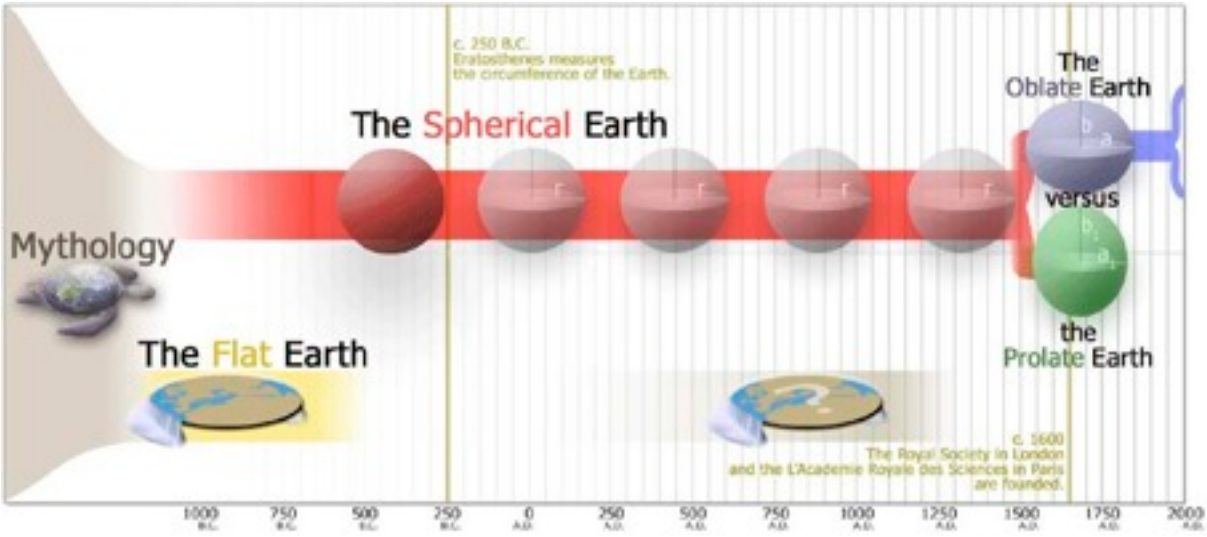



*Figure 5.2: Evolution of the ideas on the shape of the Earth (taken from NOAA).*

Eratosthenes (276 -196 BC) was a Greek mathematician, philosopher and chief librarian at the Great Library of Alexandria. Although he was not the first person to make attempts at calculating the circumference of the Earth, he was the mot accurate. He measured the circumference of the Earth using the following equation:

$$\text{circumference} = (360^\circ \div \theta) \times s$$

where

$s$ = distance between two points on the surface of the Earth at the same longitude

$\theta$ = the angle between these two points, if you were to draw a line from each point to the center of the Earth

Obviously, Eratosthenes could not go to the center of the Earth so he calculated the angle using the rays of the sun. At noon on the longest day of the year (the summer solstice) the sun shone directly into a deep well at Syene (present day Aswan, Egypt), casting no shadow (i.e. the rays were perpendicular to the surface). At the same time in Alexandria, Egypt, he found that the sun cast a shadow equivalent to about 1/50th of a circle or 7.12°. He figured that by measuring the shadow of a column (with known height) in Alexandria, he could figure out how much of the curve of the Earth separated Alexandria from Syene.

He combined this measurement with the distance between Syene and Alexandria, about 4,400 stades.

Using the above equation, we get: (360°÷ 7.12°) which equals 50; and 50 x 4,400 stades equals 220,000 stades, or about 39,376 kilometers (Figure 3.21). The accepted measurement of the Earth's circumference today is about 40,075 kilometers.

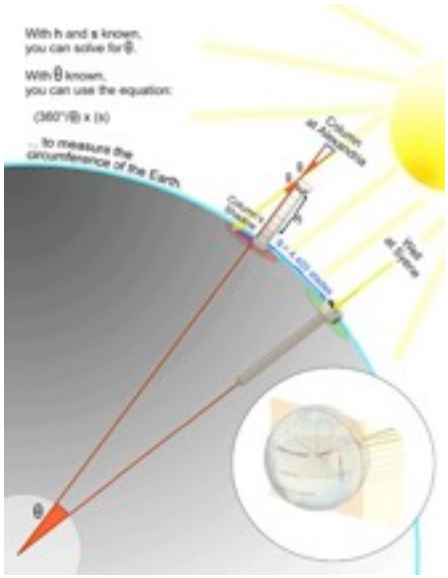



*Figure 5.3: This illustration shows how Eratosthenes calculated the circumference of the Earth. By calculating the angle of the between the column and its shadow and using the proof that alternate interior angles are equivalent to calculate, he calculated the angle to the centre of the Earth. Once he knew the that angle theta was equivalent to a distance, he could calculate the distance (circumference) for an angle of 360 degrees (whole Earth). Note that his calculations assume a spherical Earth and that the two points were at the same longitude. Taken from NOAA.*

In about 150 AD, Ptolemy wrote the greatest book of ancient astronomy, usually known from its Arabic title, Almagest, which means "The Greatest." In it Ptolemy listed some of the arguments for the sphericity of the Earth that persuaded earlier Greeks from Plato to Aristotle. He was the one who combined all the proofs into one book, mainly based on astronomical observations.

As Europe entered the Dark Ages, the teachings of the ancient Greeks were largely lost or forgotten but there is evidence that the idea of a spherical Earth was still accepted. Maps were dominated more and more by theological teachings and ideas of a flat Earth surfaced. However, the overall idea of a spherical earth was still common.

Fortunately, work by Ptolemy as well as other ancient Greek teachings were preserved in Arabic translation and these works eventually started to reenter Europe through trade and contacts.

The Book of Roger, commissioned by a Norman king in Sicily in the 12 century, had maps and a geography based on Ptolemy. Additional information, probably based on trade in

the East was used to update the Ptolemaic maps. Portuguese and Spanish navigators in the 15th and 16th centuries accepted the ideas of a spherical Earth, including Christopher Columbus However, there were different ideas on the size of the Earth. Some, including Columbus, questioned the calculations made by Eratosthenes over 1500 ago. See Historical Note on Maps in the next section for more information.

Around the end of the 16th century, the idea that the Earth was a perfect sphere evolved into a radical new idea: that the Earth was an imperfect sphere. This new way of thinking was initially divided into two major schools of thought. One believed the Earth was egg-shaped (prolate). The other believed the Earth was flattened at the poles (oblate). If the Earth were prolate, a degree would be longer at the equator than at the poles. If the Earth was oblate, the reverse would be true.

Isaac Newton developed the theory of gravity and proposed that the globe is actually an oblate spheroid due to angular momentum as the Earth spun around its axis. He argued that the reason why pendulum clocks beat more slowly at the equator than at higher latitudes was because there was reduced gravity at the equator.

In 1669, a French astronomer, Jean Picard developed a triangulation line near Paris (along the Paris Meridian) and measured a degree of latitude to be 56,996 toise (-111.09 km). In the 1690's Giovanni Cassini calculated that a degree of latitude was 57,097 toise (-111.28 km) in south of Picard's study area. His son, Jacques Cassini, calculates that a degree of latitude is longer north of Picard's study area. As a result, they proposed a prolate spheroid as the correct shape for the earth, in contrast to Newtown's reasoning that the Earth was oblate in shape.

To prove the idea of a prolate Earth, which was supported by the French, the French Academy of Sciences in Paris staged two expeditions, one to Peru (now Ecuador) at the equator, and the other to the border of Sweden and Finland in the northern hemisphere. Their objective was to measure the north-south curvature of the Earth at each location's latitude and determine whose concept of the Earth's shape was correct. The Academy's efforts proved that Newton was right. The Earth is flattened into the shape of an oblate sphere.

*"He confirmed with great effort at distance places*

*What Newton knew without ever leaving home" Voltaire*

In short, the shape of the Earth being a sphere was well accepted since about 500 BC. There was never really any widespread belief that the Earth was flat. Since the 19th century, the Earth as an oblate spheroid was accepted.

### *Historical Note on the Shape of the Earth*

School children are often wrongly taught that Columbus sailed Westward to China to prove that the Earth is round. Once launched on his journey Columbus is often portrayed

as heroically pressing on despite the opposition of his sailors, who feared their little fleet would fall off the edge of a flat Earth. That is almost the exact opposite of the truth.

Most educated people in Columbus's day knew the Earth was round. In fact, they not only knew the Earth was round they knew the size of the Earth as well. Almost everyone except Columbus accepted the estimate for the radius of the round Earth computed by Eratosthenes of Cyrene (276-195 B.C.). Eratosthenes figured the Earth's radius to be about 6267 kilometers, a figure remarkably close to the modern mean of about 6371 kilometers. In the 1490's educated people had known for over one thousand five hundred years the actual size of the round Earth. Since ancient days cartographers had even created projections to deal with the representation of a round earth on flat maps.

*Figure 5.4: Christopher Columbus.*

Even many uneducated people knew the Earth was round. Among uneducated people sailors especially believed the Earth to be round because of the frequent observation at sea that tall points such as mountains come into view above the horizon as the distance to an objective becomes closer. Many "round Earth" visual effects incompatible with a flat Earth are easily seen by the human eye at sea.

Columbus met much opposition at Court to his plan precisely because people knew the Earth was a very large sphere. The ships of Columbus's day were so slow that they could not be loaded with enough food and water to voyage directly to China westward from Europe. Without the then-unknown continents of North and South America to use as resupply points the direct voyage would be so long that the crew would die before making landfall.

Columbus based his plans for his voyage on the argument that the Earth is smaller than it truly is. Educated people were unimpressed with what they regarded as his chain of wishful thinking assumptions that "proved" Eratosthenes was wrong and that a Westward voyage was just barely feasible. When Columbus launched across the Atlantic his sailors were fearful that in the event his estimate of the Earth's size was wrong and everyone else was right they would expire of thirst and starvation.

As it turns out Eratosthenes was right and Columbus was wrong about the size of the Earth. Columbus simply had the good fortune of rediscovering a New World before he

and his crew died proving the true size of the round Earth. In all fairness it should be pointed out that despite his flawed belief in a small world Columbus was a master admiral of unparalleled skill, intelligence and personal courage. A failed and quarrelsome administrator on land, Columbus is indisputably one of the greatest leaders who ever took to sea. He is alone among the early voyagers in executing and surviving four successful voyages to the New World.

### *Figure of the Earth*

The figure of the earth has to do with the way the earth's size and shape is defined. When measurements are made on the surface of the Earth, they are taken on the apparent or topographic surface of the Earth. However, this is not a suitable surface for exact mathematical computations because of the complicated formulas which would be required to take the topographic irregularities into account.

The spherical Earth offers a simple surface which is mathematically easy to deal with. Many astronomical and navigational computations use it as a surface representing the Earth. While the sphere is a close approximation of the true figure of the earth and satisfactory for many purposes, a more exact figure is often necessary for detailed surveying and accurate distance calculations. Computations are therefore, often performed on an ellipsoid and geoid surface (see following sections).

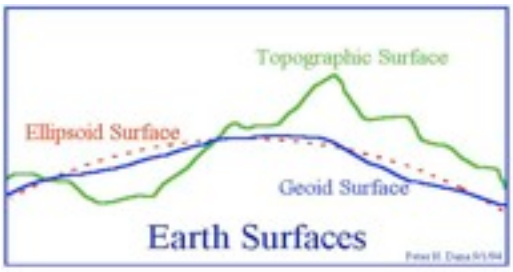



*Figure 5.5: The surfaces of the Earth: the ellipsoid, geoid and topographic surface.*

#### *Ellipsoid*

As stated above, a much better approximation for the shape of the Earth is an ellipsoid (or spheroid), a figure formed by taking a mathematical ellipse (Figure 5.6).

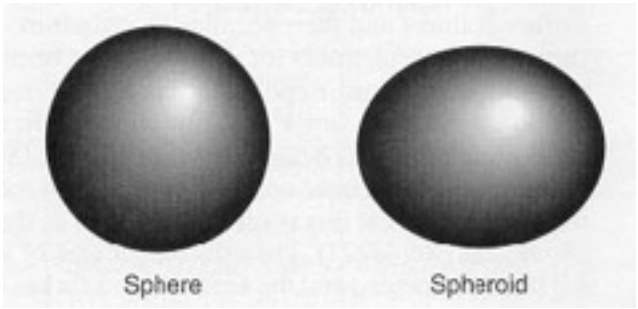



*Figure 5.6: The difference between a sphere and an oblate spheroid or ellipsoid.*

To measure the Earth and avoid problems associated with complex topography on the Earth's surface, geodesists use a theoretical mathematical surface called the ellipsoid. It is the reference surface used to approximate the shape of the earth in geodetic surveys. The ellipsoid can be completely smooth and does not take any irregularities - such as mountains or valleys -- into account because it exists only in theory and not in real life.

The ellipsoid is created when an ellipse is rotated around its minor (shorter) axis. The shape of an ellipsoid may be defined in a number of ways, but in geodetic practice the definition is usually by its semi-major axis and flattening. Flattening (f) is dependent on both the semi-major axis (a) and the semi-minor axis (b) (Figure 5.7).

f = (a -b) / a

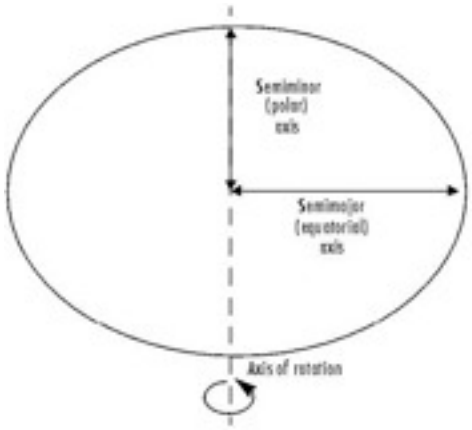



*Figure 5.7: The axes of an ellipsoid.*

Typical values of the parameters for an ellipsoid are:

**a** = 6378135.00m

**b** = 6356750.52m

**f** = 1/298.26

However, different calculations of a and b were made during different geodetic surveys and there was no real consensus on which estimates of a and b were correct (differences arose due to changes in the Earth's topography in different locations). As a result, different ellipsoid standards were defined for each country so that national mapping agencies could produce accurate maps and measure position (remember that the ellipsoid is just a mathematical model).

With the development of the ballistic missile program in the 1950s and the need to target them accurately, there was a need for an international ellipsoid standard. Without a single global standard, the maps produced by different countries using different ellipsoids could never be made to fit along their edges and offsets had to be calculated when moving from one country to another. This lead to the development of WGS84 which is now the

most commonly used ellipsoid today (reference ellipsoid used for GPS). The global ellipsoid have their origin at the centre of the Earth (geocentric systems) which differs from regional ellipsoids. Below are listed some of the more common ellipsoids in use today (Table 5.1).

| Name | *Equatorial/ Semi-major axis (m)* | *Polar/Semi-Minor axis (m)* |
|---|---|---|
| ***Clarke 1886*** | ***6 378 206*** | ***6 356 584*** |
| ***Bessel 1841*** | ***6 377 397*** | ***6 356 078*** |
| ***International 1924*** | ***6 378 388*** | ***6 356 912*** |
| ***Krasovsky 1940*** | ***6 378 245*** | ***6 356 863*** |
| ***GRS 1980*** | ***6 378 137*** | ***6 356 752*** |
| ***WGS 1984*** | ***6 378 137*** | ***6 356 752*** |
| ***Sphere (6371 km)*** | ***6 371 000*** | ***6 371 000*** |

*Table 5.1: Commonly used reference ellipsoids.*

For small scale maps the earth may be treated as a sphere. However, to maintain accuracy for larger scale maps the earth must be treated as a spheroid or ellipsoid.

***Geoid***

While the ellipsoid gives a common reference to geodesists, it is still only a mathematical concept and we still need a way of accounting for the reality of the Earth's surface. To meet this need, the geoid, a shape that refers to global mean sea level, was defined.

The definition of the geoid is:

**an equipotential surface which means that potential gravity is the same at every point on its surface (best fits the mean sea-level in the world).**

According to the laws of physics, the surface of the ocean is an "equipotential surface" of the earth's gravity field (ignoring waves, winds, tides and currents for now). This means that if one could place balls everywhere on the surface of the ocean, none of the balls would roll down hill because they are all on the same "level". The direction of gravity is always perpendicular to the geoid.

While the ellipsoid fits the earth quite well, the actual surface of the Earth deviates by up to 100 meters from this ideal ellipsoid and is due to variations in the earth's mass distribution. The resulting surface, the geoid, has an irregular shape with many bumps and dips. These bumps and dips are caused by minute variations in the earth's gravitational field, meaning that certain areas of the planet experience more or less gravitational "pull" than others. Take for example a massive mountain on the ocean floor, the extra gravitational attraction due to the extra mass of the mountain attracts water toward it causing a

local bump in the ocean surface. This bump cannot be seen with the naked eye because the slope of the ocean surface is very low. (Taken from Sandwell and Smith, NOAA).

Optical instruments containing leveling devices are commonly used to make geodetic measurements. When properly adjusted, the vertical axis of the instrument coincides with the direction of gravity and is, therefore, perpendicular to the geoid. The ellipsoid, which is a regular surface and the geoid surface, which is irregular do not coincide (Figure 5.8). The separation between the ellipsoid surface and the geoid surface are referred to as geoid undulations, geoid heights, or geoid separations.

The WGS84 (World Geodetic system 1984) defines geoid heights for the entire earth and it is the mathematical model to which GPS (Global Positioning System) are referenced.

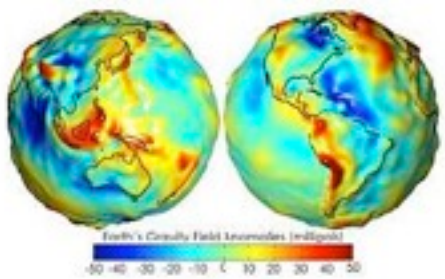


*Figure 5.8: The shape of the Earth's geoid. Red areas equal higher gravity (geoid) anomalies (the surface is being pushed up), blue areas denote negative gravity (geoid) anomalies (the surface is being pulled down).*

### *Projections*

Maps depict, most commonly on a flat surface, the spatial organisation of any part of the physical world at any scale and can be used to symbolise a wide variety of both static and dynamic information.

A globe is the only true representation or model of the earth's surface but its use to represent any area of the earth's surface has its limitations:

- ☑ On a globe you can only view one side of the earth at any one time - only have a hemispheric view
- ☑ You are confined by scale - usually the size or scale is too small to be useful
- ☑ Large globes are difficult to transport, store and reproduce.

Maps, on the other hand are much more versatile. They:

- ☑ Allow us to get a sense of relative location, of spatial relationships, on a scale that goes far beyond our own perspective
- ☑ Can convey all kinds of information beyond that of location, if the mapmaker chooses data and symbolism properly e.g. a topographic map

☑ Are portable - much more convenient than carrying around a globe!

The essential and biggest limitation of any map is the physical impossibility of transferring a curved surface, like the Earth's, onto a flat one without distortion and error. Try and peel an orange and flatten the peel without distorting it. Can it be done?

Map projections are the ways cartographers use to convert the Earth's 3D surface to a 2D representation and is a fundamental component of mapmaking. A map projection is:

**a mathematical means of transferring information from the Earth's three-dimensional curved surface (a global location defined by latitude and longitude [φ,λ]) to a two-dimensional medium (a planar position [x,y]).**

It is a way of moving from a spherical coordinate system (including geodetic parameters) to a Cartesian coordinate system. Every recognised map projection can be represented as a pair of mathematical functions:

$x = f(\varphi,\lambda)$

$y = g(\varphi,\lambda)$

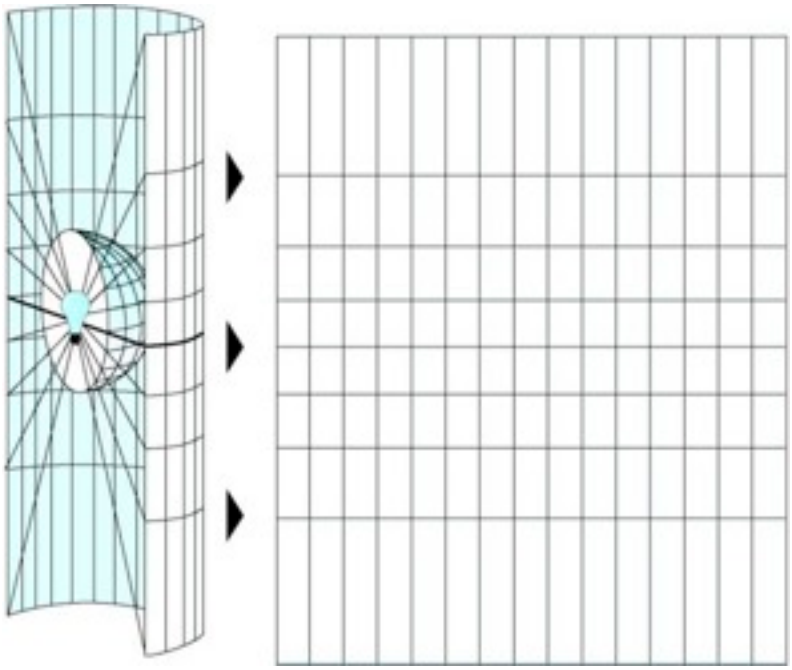


*Figure 5.9: The term 'projection' comes from the notion of placing a light source inside a transparent globe and projecting shadows of the meridians, parallels and other geographic features onto a sheet of paper placed tangent to the globe. Changing the position of the light source alters the pattern of parallels and meridians on the map, resulting in maps that have different geometric properties.*

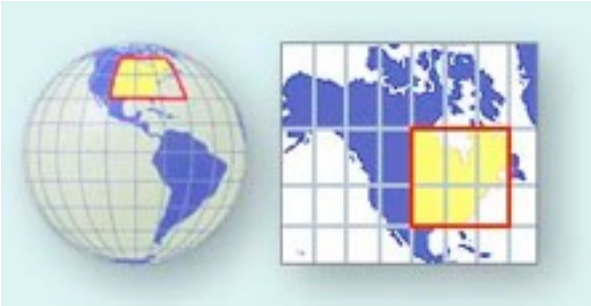


*Figure 5.10: The Eastern US represented on a globe (left) and its 2D representation (right). Source: USGS.*

### *Distortion Properties*

Different projections are used for different types of maps because each projection is particularly appropriate to certain uses. All map projections introduce distortions -they are inherent in flattening of the sphere. For example, a projection that accurately represents the shapes of the continents will distort their relative sizes. Some projections minimize distortions in some of these properties at the expense of maximizing errors in others. Some projections are attempts to only moderately distort all of these properties.

Although many different map projections exist, they all introduce distortion in one or more of the following measurement properties:

**Conformality** or **Shape** When the scale of a map at any point on the map is the same in any direction (i.e. the scale in the x and y direction are always equal), the projection is conformal. Meridians (lines of longitude) and parallels (lines of latitude) intersect at right angles. Shape is preserved locally on conformal maps but at the cost of accurate representation of area (Figure 5.11). Shapes are more or less distorted using equal-area projections.

**Area** When a map portrays areas over the entire map so that all mapped areas have the same proportional relationship to the areas on the Earth that they represent. If area is accurately shown, shape is thrown off. Conformality and true area are mutually exclusive virtues (Figure 5.11). If you have one, you sacrifice the other. Area is preserved by equal-area projections.

**Distance** A map is equidistant when it portrays distances from the center of the projection to any other place on the map. Equidistant projections preserve distance between points.

**Direction** A map preserves direction when azimuths (angles from a point on a line to another point) are portrayed correctly in all directions. On maps that are "true direction", a straight line is a great circle route, the direction of the shortest distance between two places. Azimuthal projections represent distortions correctly with respect to the centre.

**Scale** is the relationship between a distance portrayed on a map and the same distance on the Earth and often associated with true distance.

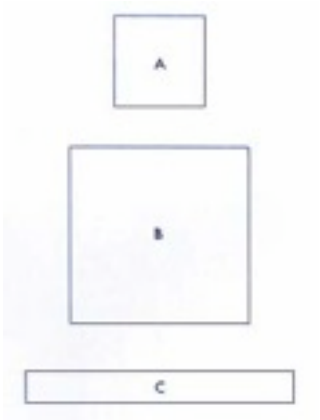



*Figure 5.11: A and B are conformal: They show the same shape. A and C are true area, but certainly NOT conformal in shape. Equivalence of area might not be important to a navigator, but it is very important for a lot of classroom and media uses to show various distributions.*

It is not possible for any one projection to retain more than one of these qualities over a large area of the Earth. Some projections minimise distortions in some of these properties at the expense of maximising errors in others. Some projections are attempts to only moderately distort all of these properties. It is impossible to have a perfect map.

It is important to note that every map is a purposeful selection from everything that is known, bent to the mapmaker's ends. The map-maker selects only the information that is essential to fulfill the purpose of the map and hence, every map advances an interest of some sort and has an inherent bias inbedded within it. Moreover, maps show only one perspective and are an abstraction of reality.

### *Projection Parameters*

Once the reference datum and type of projection have been chosen other parameters have to be defined in order to draw a map.

**Central meridian** is defined as a meridian of longitude, which is defined as the center of a particular projection zone. It is measured in degrees east or west of Greenwich.

**Scale factor** is defined as a value that describe the relationship of a distance measured between two points as measured on the ground and compared to the same two points as measured on the map projection. Mathematically it is defined:

SF=grid (map projection) distance/ ground (true) distance

e.g. 1/20,000 scale means that 1 cm on the map reprsents 20,000 cm (or 200 m) on the ground

**Standard parallels** are parallels of latitude, measured in degrees north or south of the equator, that are chosen to represent locations at which the scale factor will be unity (grid distance=true distance)

**Origin** is the point at which the projection originates

### *Projection Surfaces*

One method of classifying map projections is to group them by the type of surface onto which the graticule is theoretically being projected. There are three main projection surfaces (Figure 5.12):

- ☑ *Cylindrial* -projection surface forms a cylinder over the globe
- ☑ *Conical* - projection surface forms a cone over the globe
- ☑ *Azimuthal* or *Planar* - projection surface is a flat plane

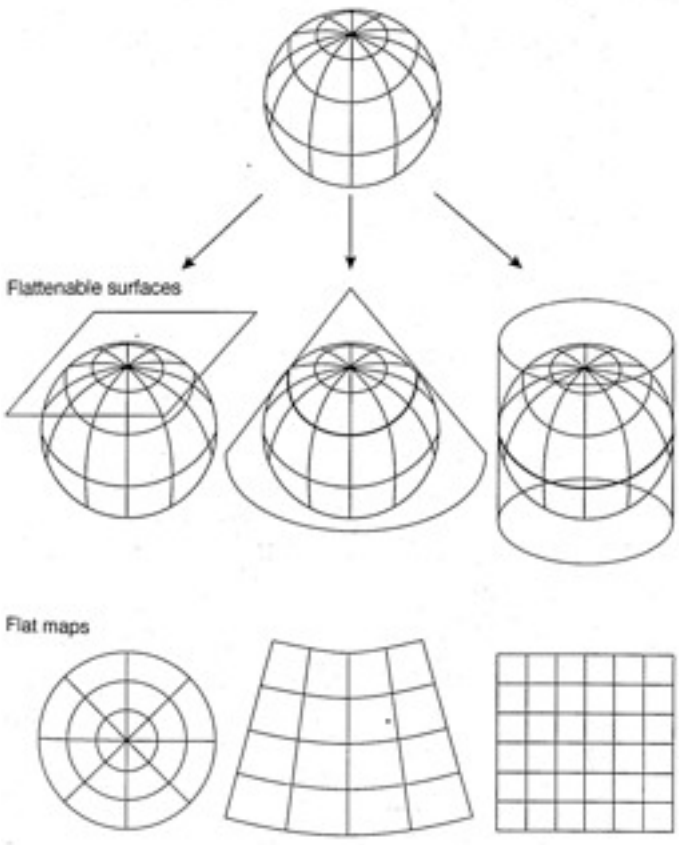



*Figure 5.12: Three main types of projection types and their flattenable surfaces: Azimuthal (left), Conic (middle) and Cylindrical (right).*

Map projections that do not fit within these three classes are described as *Pseudo* or *Misceaneous* projections.

The globe does not require a transformation to a map projection surface. On the globe, directions, distances, shape, scale and area are true.

***Cylindrical Projections***

A cylindrical projection is produced by wrapping a cylinder around a globe representing the Earth. The map projection is the image of the globe projected onto the cylindrical surface, which is then unwrapped onto a flat surface.

The meridians and parallels are "projected" onto the cylinder as straight, parallel lines crossing at right angles. The meridians in cylindrical projections do not converge at the poles, as they do on the globe, resulting in increased stretching and distortion toward the poles. The various types of cylindrical projections have different way of spacing the meridians and parallels to obtain certain desirable cartographic properties.

Cylindrical projections have three aspects:

- When the cylinder is parallel to the polar axis, the tangent to the Earth is aligned with latitude and has a normal (or regular) aspect (Figure 5.13)
- When the cylinder is perpendicular to the polar axis, the tangent to the Earth is aligned with the meridians and has a transverse aspect (Figure 5.13)
- When the cylinder is oblique to the polar axis, the tangent to the Earth is at an angle to both the parallels and meridians and has an oblique aspect (Figure 5.13)

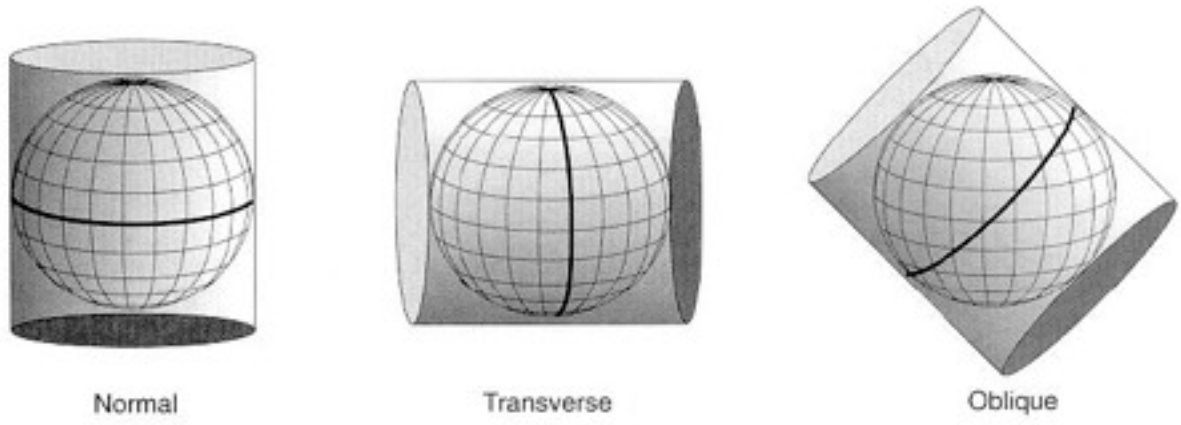



*Figure 5.13: Cylindrical projections have three aspects: Normal (left), Transverse (middle) and Oblique (right).*

Cylindrical projections can either be equal-area, conformal or equidistant. They are best used between the tropics and not at all adequate in representing higher latitudes.

***Mercator***

In 1569, the Flemish cartographer and mathematician Gerardus Mercator (Figure 5.14) published the first "Mercator" map. His projection revolutionised navigation by straightening out *rhumb lines* on a flat map and was the standard map for maritime mapping in the 17th and 18th centuries. The Mercator map has received much criticism recently be-

cause of the relative size of the European and colonial powers compared to the rest of the world. However, at the time Mercator clearly explained that he developed this projection for navigational purposes. He used equal-area maps to represent the whole world for his own map-making.

*Figure 5.14: The Flemish cartographer Gerardus Mercator*

The Mercator projection has straight meridians and parallels that intersect at right angles. The projection is often used for marine navigation because all straight lines on the map are lines of constant azimuth/direction (rhumb line). Directions along a rhumb line are true between any two points on a map, but a rhumb line is usually not the shortest distance between points (does not correspond to a great circle) (Figure 5.15).


*Figure 5.15: Mercator projection showing the difference between the rhumb line and the great circle between Washington D.C. and London. The great circle represents the shortest distance however, the rhumb line is more useful for navigation as it represents a line of constant bearing. A navigator wanting to travel from London to Washington D.C need only draw a straight line on a Mercator map, measure the angle and determine the bearing to take.*

On the Mercator projection, scale and distance are true at the equator but are reasonably correct within 15° of the equator. Two particular parallels can be made correct in scale instead of the equator. The Mercator is conformal but not equidistant (Figure 5.16). Notice how areas toward the poles seem stretched and artificially large. Countries toward the poles are significantly larger than they should be. A common example of stretching,

this occurs when areas are artificially enlarged in the projection process. In the case of Greenland, off the coast of North America, the country actually appears larger than South America.

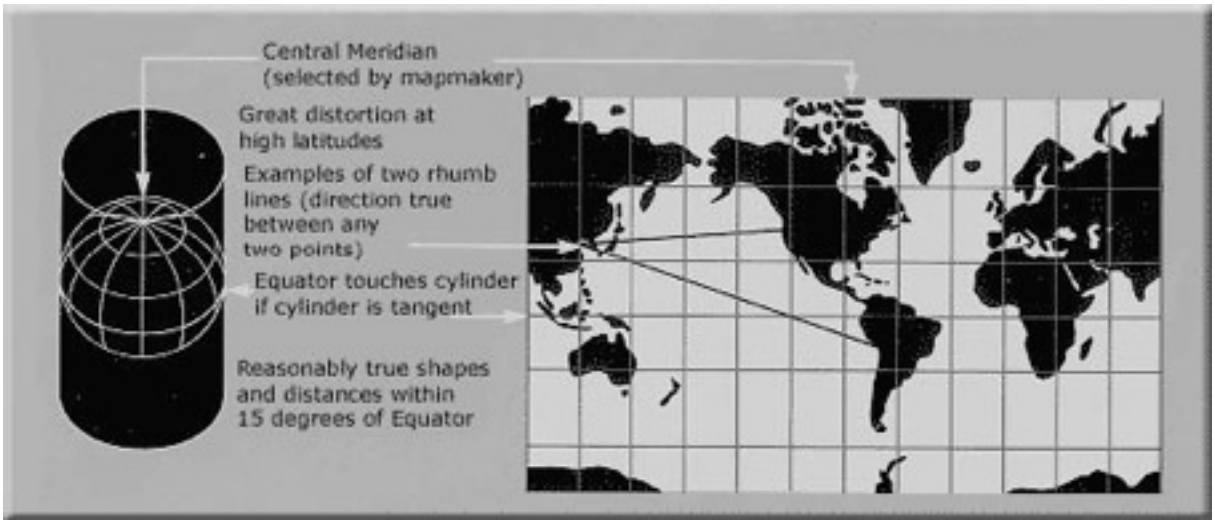



*Figure 5.16: Mercator projection. Source: USGS*

#### *Transverse Mercator*

Turning the cylinder so that it is tangent to the Earth along a meridian instead of tangent to the Equator results in what is called a transverse cylindrical projection. Scale and distance area true along the chosen meridian but are reasonably correct within 15° of the equator. Distortion increases rapidly outside the 15° band. If we pick a North-South line running through Athens we can make maps all the way from Scandinavia down the length of Africa, but any maps using this projection in North and South America would be hopelessly distorted (Figure 5.17).

Transverse Mercator maps conformal however, shapes and angles within any small area are essentially true. Graticule spacing increases way from the central meridian. The equator, central meridian and the meridians 90° from the central meridian are straight but other parallels are complex curves concave toward the nearest pole. This projection is often used to portray areas with larger north-south than east-west extent. Distortion of scale, distance, direction and area increase away from the central meridian.

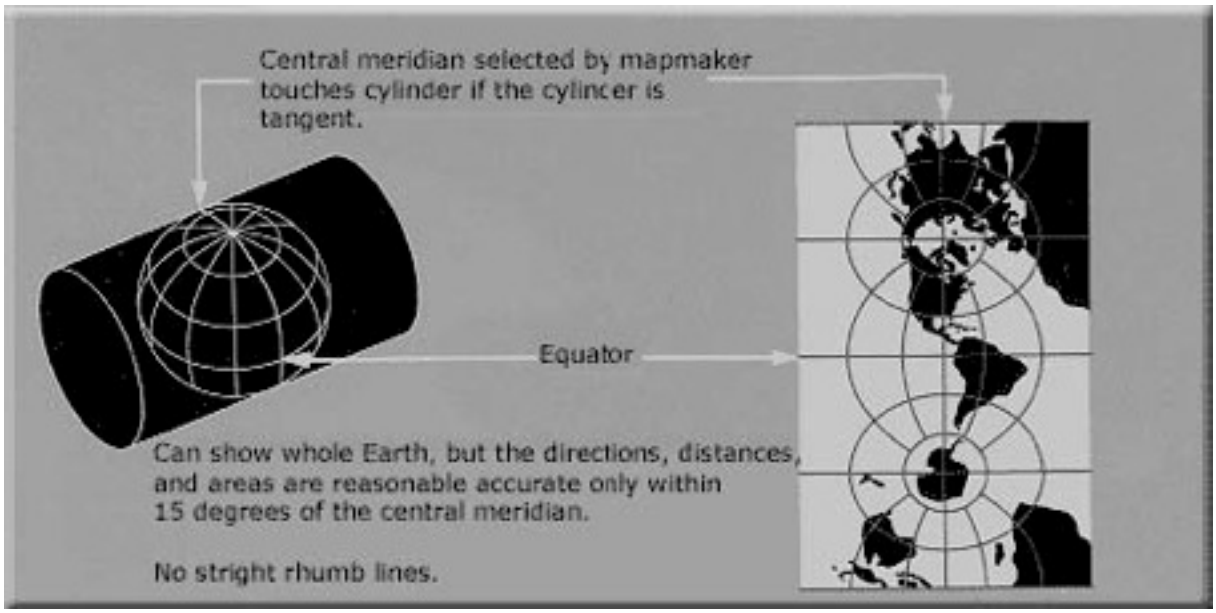



*Figure 5.17: Transverse Mercator projection. Source: USGS*

***Universal Transverse Mercator***

The Universal Transverse Mercator (UTM) has become a popular projection for use with GIS because of its level of accuracy, near global coverage and ease of use. The UTM projection was developed for military purposes. After World War II, all of the nations in NATO (the North Atlantic Treaty Organization) agreed that a standard spatial coordinate system was needed so that they can more precisely coordinate military movements between nations. It is now the most commonly used map projection.

The Universal Transverse Mercator (UTM) projection is used to define horizontal positions worldwide by dividing the surface of the Earth into 60 zones, each 6 degrees of longitude wide (Figure 5.18). Each zone extends from 80°S to 84°N; the reason for the asymmetry is that 80°S falls in the southern ocean, south of any southern continental landmass and hence accuracy is not needed beyond this point. In contrast, you have to be at 84°N to reach a point north of Greenland. The zones are numbered, starting with 1 which runs from 180° to the 174°W, and progressing eastward t zone 60. These zones cover almost the entire planet, omitting only the Arctic Ocean in the north and central Antarctica in the South. At the poles, a Universal Polar Stereographic projection (UPS) is used.

As its name implies, the UTM system is based on the Transverse Mercator projection. Each UTM zone is mapped using the Transverse Mercator projection with a central meridian in the centre of the zone (Figure 5.19). Thus, UTM zone 1, which extends from 180° to 174°W, has a central meridian running down 177°W. Therefore, the UTM projection assures that all spots on Earth are within 3 degrees of the central meridian.

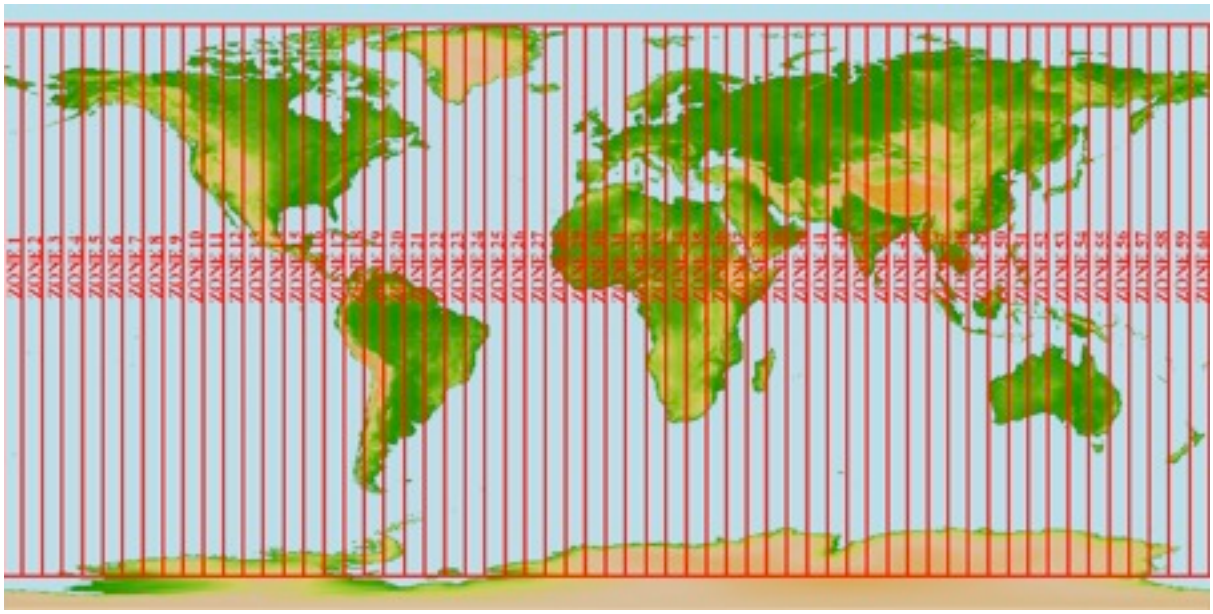


*Figure 5.18: Zones defined in the UTM.*


*Figure 5.19: A UTM zone.*

Each UTM zone is divided further into strips of 8 degrees latitude beginning at 80 degrees S. They are assigned letters C through X with O and I omitted (Figure 5.20). This division is mainly used in the military.

The UTM system does not use the "standard" Transverse Mercator projection, which is tangent. Instead, it uses a secant variation that has two lines of tangency located approximately 180 kilometers on either side of the central meridian. Since the ratio of actual map scale to nominal scale is 1.0 only along the map's line of tangency, this ratio must be different from 1.0 along the central meridian. The formal definition of the UTM sys-

tem states that along the central meridian, the ratio of actual map scale to nominal scale must be 0.9996.

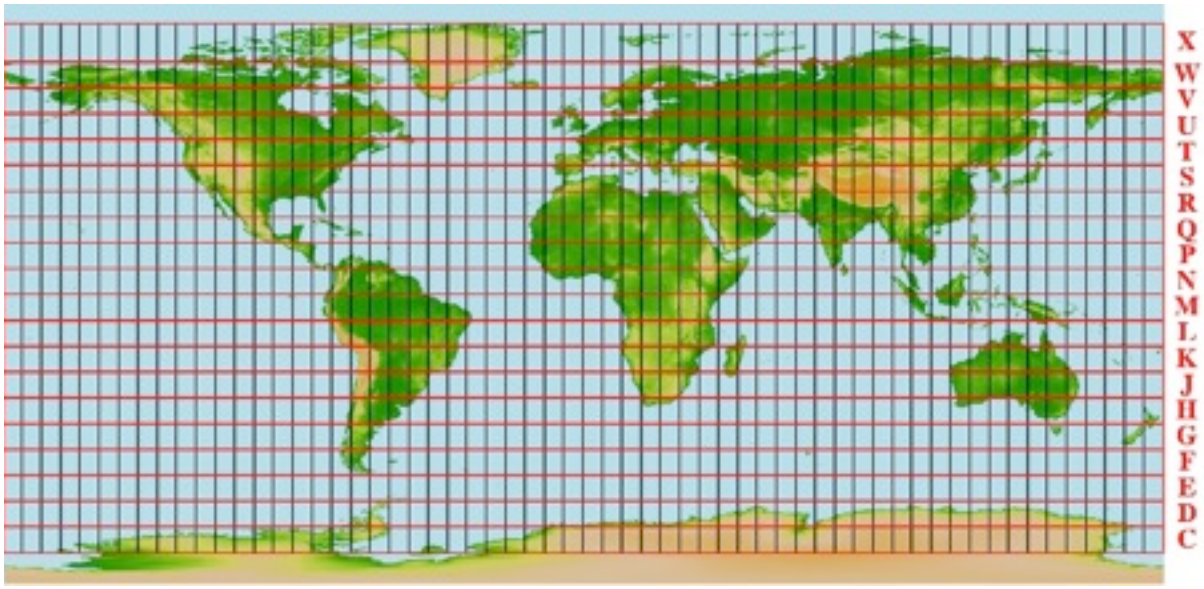



*Figure 5.20: UTM subzones.*

The projection is conformal, so small features appear with the correct shape and scale is the same in all directions.

The coordinates of a UTM zone are defined in meters east or west of the zone's origin. The distances east of the origin (x coordinates) are called eastings. Distances north of the origin (y coordinates) are called northings. One of the big benefits of using UTM coordinates is that they are always positive numbers.

An example of UTM coordinates are:

44030976, 38403088

44030984, 38403080

44030900, 38403077

and not longitude, latitude coordinates, such as:

-110.3484, 44.2856

-110.3463, 44.2889

-110.3511, 44.2902

Many countries define their own UTM coordinate system (see Map Grid Australia section for Australia's standard reference system).

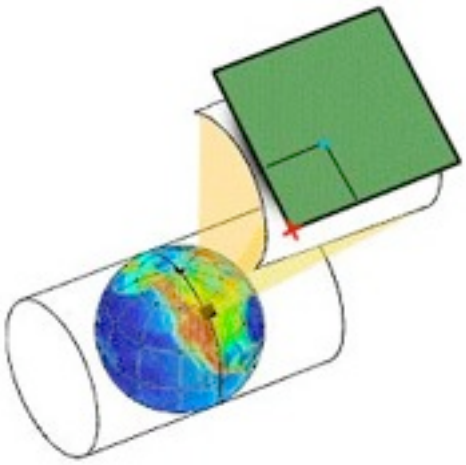


*Figure 5.21: Projected UTM coordinates on a flat piece of paper. A small section of the earth near the tangent line projected onto the cylinder, and then the cylinder being unrolled into a flat sheet. If we want to save the X,Y locations of points on our flat sheet we can now measure them as though the flat sheet were graph paper and use the resulting coordinates in a digital, flat map. This illustrates a key concept that often proves confusing to GIS newcomers: although "unprojected' data about locations on the Earth are specified in degrees, a projected maps specify the coordinates of the objects on them using X,Y coordinates using meters, feet or other linear measures. These coordinates are computed relative to some origin on the flat sheet established by the projection in use.*

#### *Oblique Mercator*

Oblique Mercator projections are used to portray regions along great circles. Distances are true along a great circle defined by the tangent line formed by the sphere and the oblique cylinder. Hence, it is used to used to show the distance between any two points as a straight line. Distances, directions, areas and shapes are reasonable accurate within 15° of the great circle but are greatly distorted as you move away from the great circle (Figure 5.22).

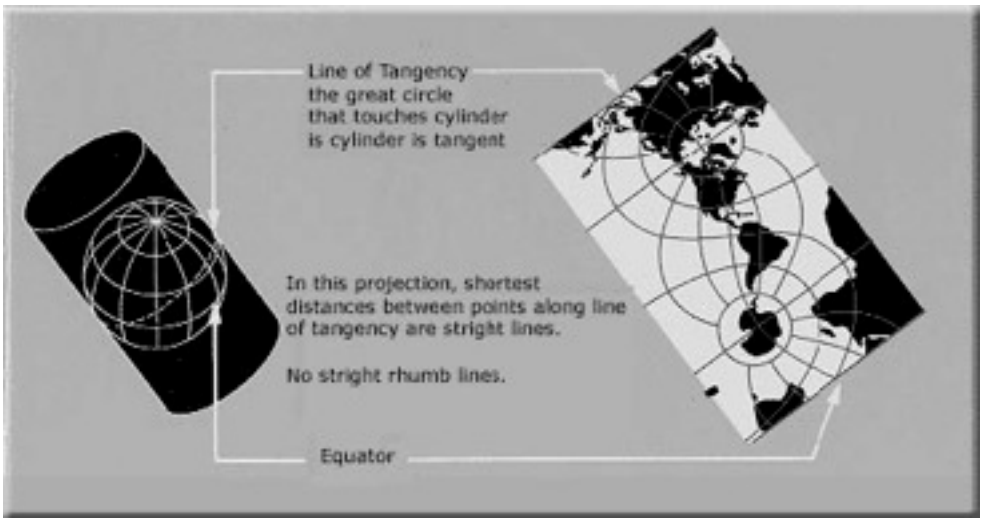



*Figure 5.22: Oblique Mercator. Source: USGS*

Oblique mercator maps are conformal but not perspective, equal area or equidistant. Graticule spacing increases away from the great circle. The equator and other parallels are complex curves concave toward the nearest pole. Two meridians (180° apart) are straight lines - all others are complex curves concave toward the great circle.

#### *Cylindrical Equal Area*

Cylindrical Equal-Area projections (e.g. Lambert Equal Area) have straight meridians and parallels, the meridians are equally spaced, the parallels unequally spaced. Scale is true along the central line and along two lines equidistant from the central line. Shape and scale distortions increase near points 90 degrees from the central line.

Cylindrical Equal Area projections (Figure 5.23) are most frequently used as textbook examples of the most easily constructed equal-area projection. However, distortions are so great using this projection that there has been little use of any of the forms for world maps by professional cartographers. The Peters Projection is an example of a Cylindrical Equal Area projection (Figure 5.24).

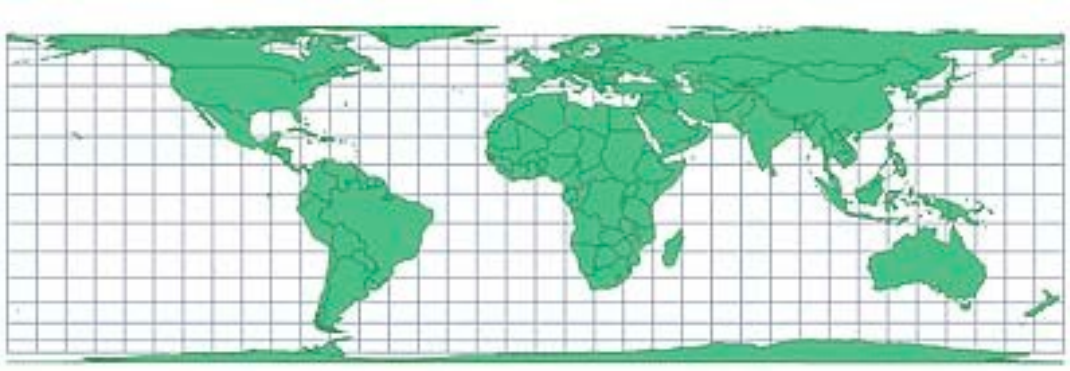


*Figure 5.23: Cylindrical Equal Area.*

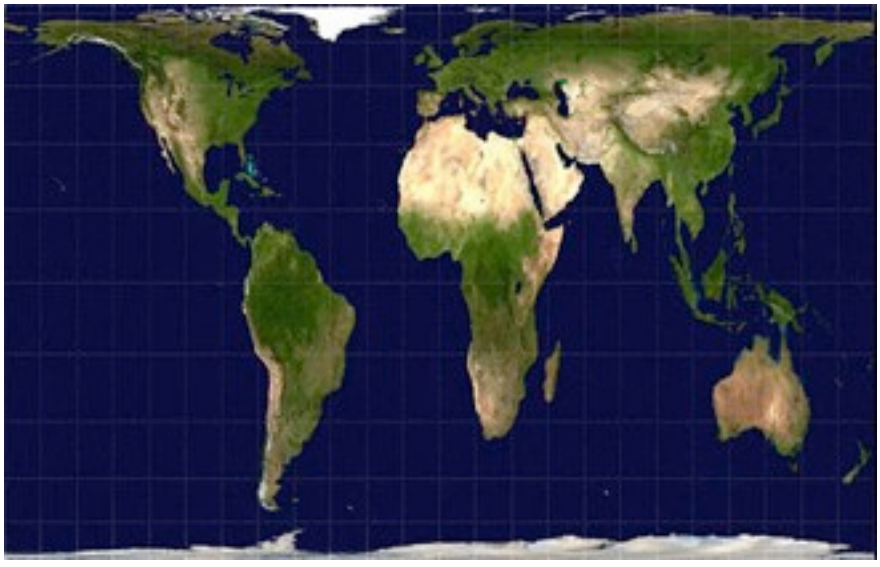


*Figure 5.24: Peters Projection.*

***Miller***

The Miller cylindrical projection (Figure 5.25) is often used to represent the whole world and looks like a Mercator projection but the latitudes are scaled. It avoids some of the scale exaggerations of the Mercator map but is not useful for navigation and shows areas and shapes with distortion.

Directions and distances are true only along the equator. Distortion is extreme at higher latitudes. It is not equal area, equidistant, conformal or perspective.

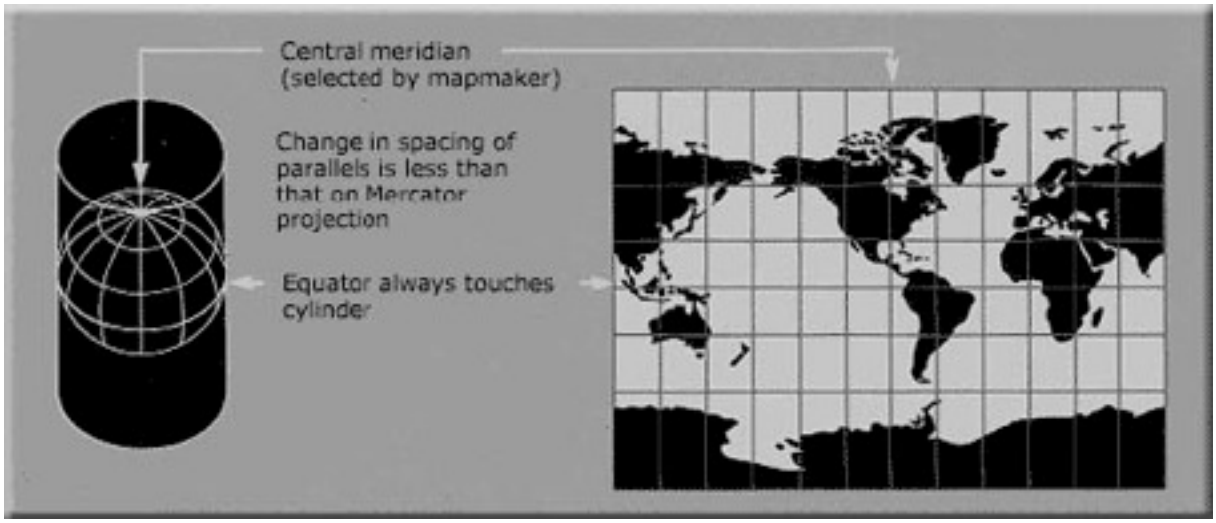



*Figure 5.25: Miller Cylindrical Source: USGS*

## *Conical Projections*

A conical projection is produced by wrapping a cone around a globe representing the Earth. The map projection is the image of the globe projected onto the conical surface, which is then unwrapped onto a flat surface (Figure 5.26).

There are 2 different types of conical projections (Figure 5.26):

- *Conic tangent* where the cone contacts the surface of the globe along one standard parallel
- *Conic secant* where the cone is sliced through the globe and intersects it twice along two standard parallels.

The standard parallels are where the cone touches or slices through the globe. There is no distortion along the standard parallels - true scale, shape and area are preserved. The map is equidistant, conformal, and equivalent along the standard parallel/s. The parallel is determined by the angle of the cone. A low angle cone (fatter cone) produces a tangent at a high latitude. A high angle cone (like a witch's hat), produces a tangent at a lower latitude. A secant is produced by reducing the size of the cone. The central meridian is opposite the edge where the cone is sliced open.

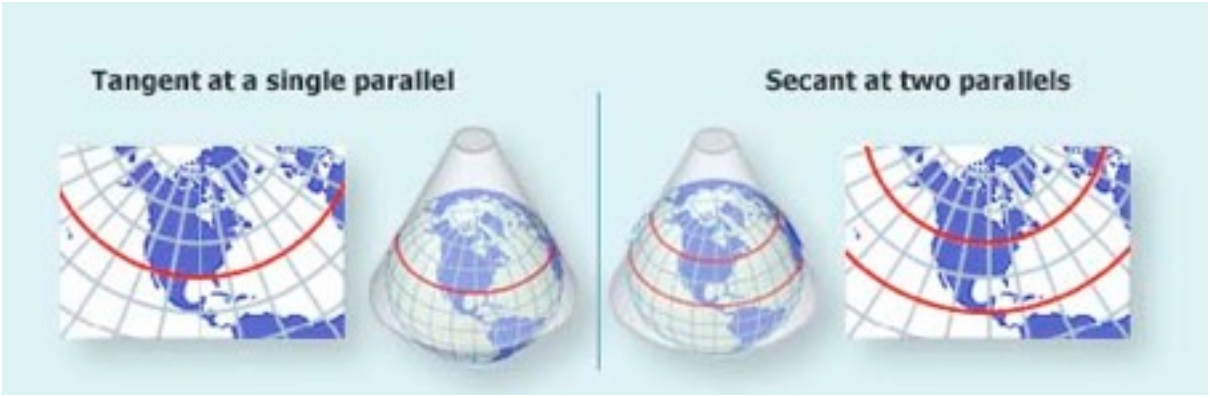



*Figure 5.26: Two types of conical projects: conic tangent (left) with one standard parallel and the conic secant (right) with two standard parallels. Standard parallels marked in red. Source: USGS.*

Meridians show up as straight lines radiating from the middle of the top edge, while parallels show up as concentric semi-circles spaced farther apart as you move away from the standard parallel/s (Figure 5.26). This means scale, shape, and area become more and more distorted away from the standard parallel(s). In the case of secant conic projects, there is minimal distortion between the two parallels. Overall, these maps are neither conformal nor equal area, but are a compromise between the two.

Conical projections are well-suited for showing areas in the middle-latitudes with a mostly east-west extent (like the United States) because they achieve less distortion at these latitudes than cylindrical projections. They are also frequently used to map large areas (e.g., states, large countries, or continents).

#### *Albers Equal Area Conic*

The Albers Equal Area Conic projection is a conic secant (two standard parallels) (Figure 5.27) that distorts scale and distance except along the standard parallels. The parallels are brought closer to one another to compensate for the increased area created by those stretched-out meridians.

All areas on the map are proportional to the same areas on the Earth - it preserves area. Deformation of shapes increases away from the two standard parallels. Directions are reasonably accurate in limited regions. Distances are true on both standard parallels. Scale true only along standard parallels. Albers Equal Area Conic maps are not conformal, perspective, or equidistant.

This projection is used in the United States and other large countries with a larger east-west than north-south extent and ones that require equal-area representation. It is also used for many thematic maps. Maps showing adjacent areas can be joined at their edges only if they have the same standard parallels (parallels of no distortion) and the same scale.

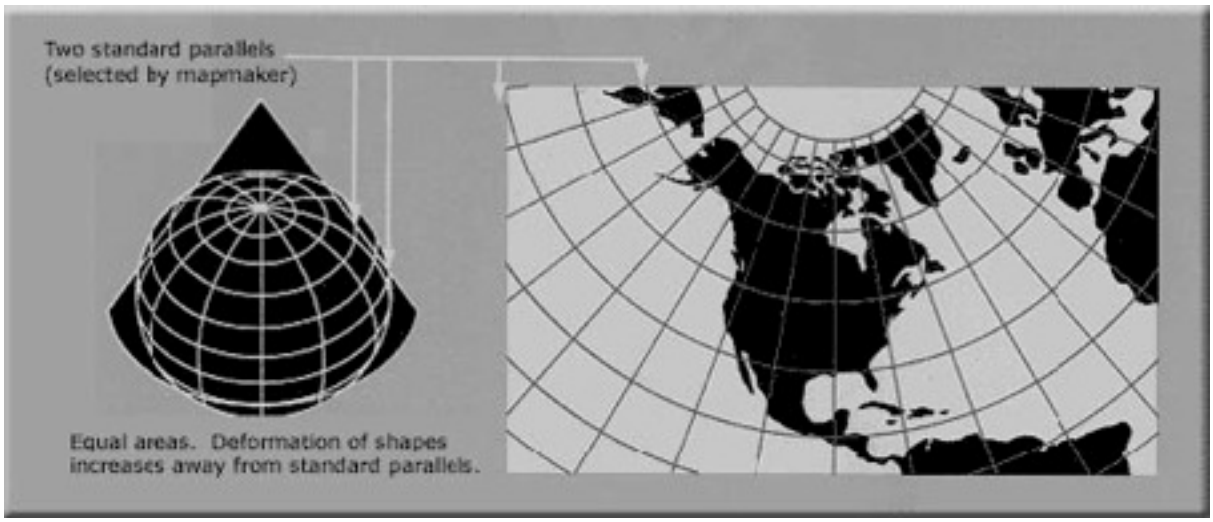



*Figure 5.27: Albers Equal Area Conic. Source: USGS.*

#### *Equidistant Conic*

The Equidistant Conic projection can be a conic secant (two standard parallels) or conic tangent (one standard parallel) (Figure 5.28). The parallels are equally spaced. This projection was first described by the Greek philosopher Claudius Ptolemy about A.D. 150.

Distances and scale are true only along all meridians and along one or two standard parallels. Directions, shapes and areas are reasonably accurate, but distortion increases away from standard parallels. Scale is constant along any given parallel. The resultant map is not conformal, perspective, or equal area, but a compromise between Lambert Conformal Conic and Albers Equal Area Conic.

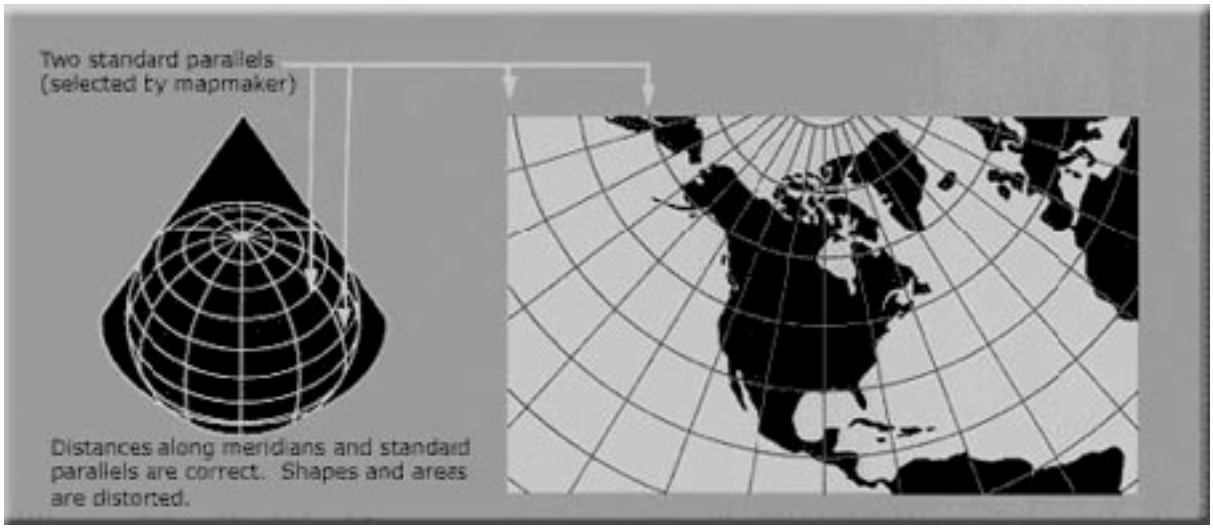



*Figure 5.28: Equidistant Conic. Source: USGS.*

The Equidistant Conic is used to show areas in the middle latitudes and is good for showing regions within a few degrees of latitude and lying on one side of the Equator. It is the most common projection in atlases for small countries.

#### *Lambert Conformal Conic*

The Lambert Conformal Conic projection is a conic secant (two standard parallels) (Figure 5.29) that distorts area and shape away from the standard parallels. The parallels are pulled apart a bit to compensate for the exaggerated separation of the meridians at higher latitudes as they approach the tip of the cone (rather than the pole of the globe).

This projection retains conformality. Distances are true only along standard parallels but are reasonably accurate elsewhere in limited regions. Directions are reasonably accurate. The distortion of shapes and areas is minimal at, but increases away from standard parallels. Shapes on large-scale maps of small areas are essentially true. The resultant map is conformal but not perspective, equal area, or equidistant.

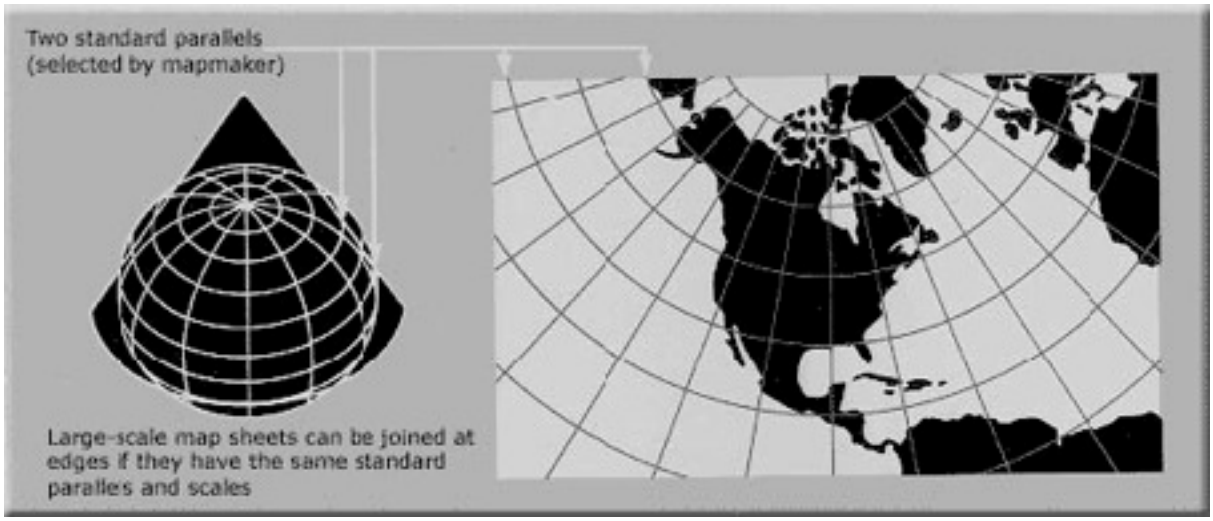



*Figure 5.29: Lambert Conformal Conic. Source: USGS.*

The Lambert Conformal Conic projection is useful for countries or regions that are mainly east-west in extent. It is one of the most widely used projections in the USA. It looks like the Albers Equal Area Conic, but the graticule spacings differ.

### *Azimuthal or Planar Projections*

An azimuthal, planar or, less commonly, zenithal projection is a projection of the globe onto a plane. Azimuthal projections usually have a point of tangency where the plane comes into contact with the generating sphere along a specific point. Any straight line drawn from the point of tangency is a true great circle. Distortion values increase from the point of tangency to the outer edges of the map.

Azimuthal projections have three aspects:

- When the plane is parallel to the polar axis, the tangent to the Earth is aligned with latitude and has a polar (or normal) aspect (Figure 5.30). The meridians are projected as straight lines radiating from the pole, and parallels are shown as complete circles centered at the pole.

☑ When the plane is perpendicular to the polar axis, the tangent to the Earth is aligned with the meridians and has an equatorial (or transverse) aspect (Figure 5.30)

☑ When the plane is oblique to the polar axis, the tangent to the Earth is at an angle to both the parallels and meridians and has an oblique aspect (Figure 5.30)

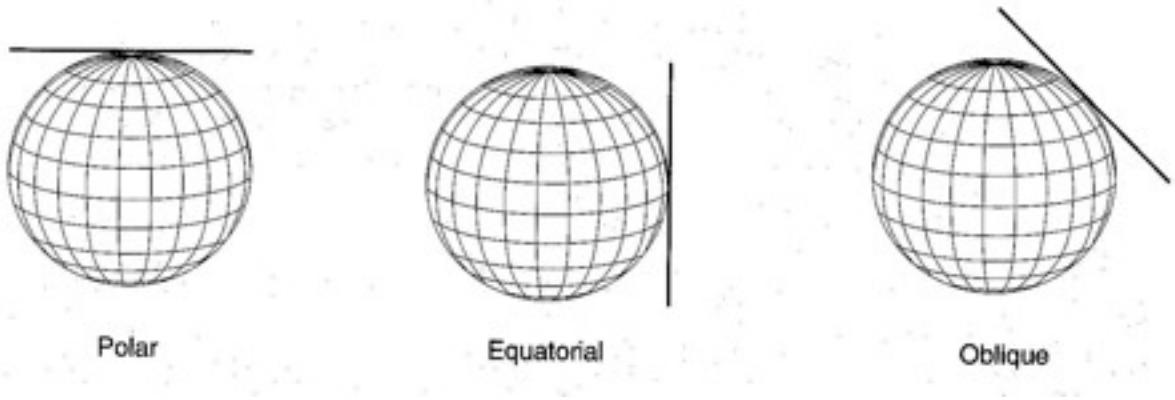



*Figure 5.30: Azimuthal projections have three aspects: polar or normal (le'), equatorial or transverse (middle) and oblique (right).*

In addition to the three aspects, you can have a tangent and secant azimuthal projection (Figure 5.31).

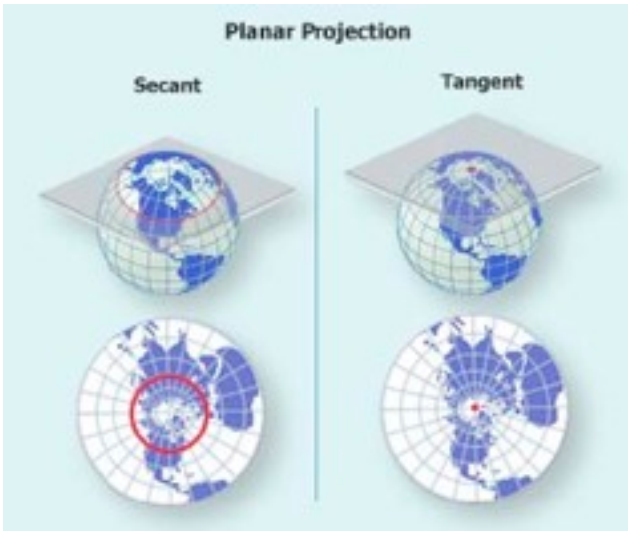



*Figure 5.31: The secant (left) and tangent (right) azimuthal projections. The tangent has a standard point, a point of tangency at the pole (red point). The secant slices the globe and has a standard parallel (red line).*

Most planar projections are not suitable for displaying the entire Earth in one view, but give a sense of the globe. They are best for displaying the polar regions and also for perspective views of the earth.

#### *Azimuthal Equidistant*

Azimuthal Equidistant (Figure 5.32) is a projections whereby all distances measured from the center of the map are true. Distortion of other properties (e.g. areas and shapes) increases dramatically away from the center point. Any straight line drawn through center point is on a great circle.

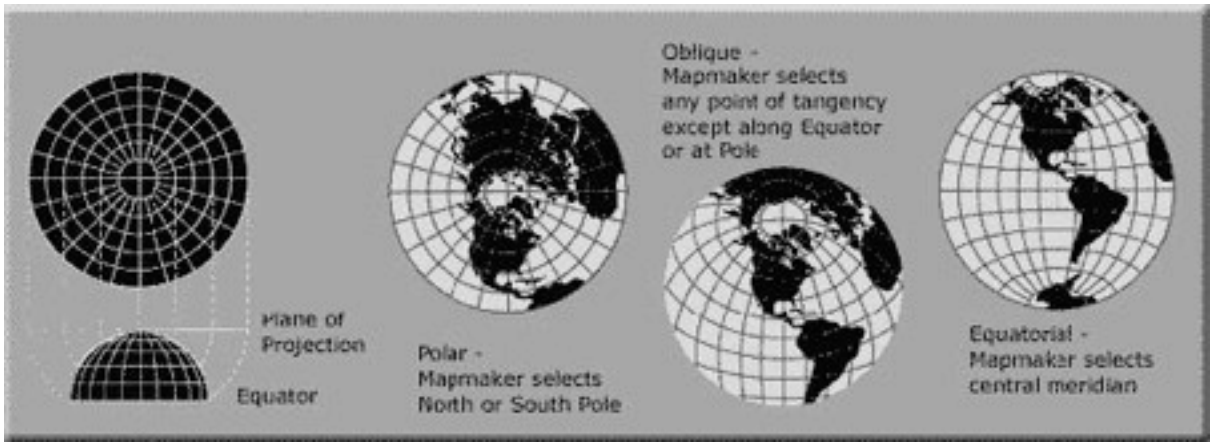



*Figure 5.32: Azimuthal Equidistant. Source: USGS.*

This map projection is useful for showing air-route distances from center point of projection. The oblique aspect is used for atlas maps of continents and world maps for radio, seismic and aviation use. The polar aspect is used for world maps, maps of polar hemispheres, and United Nations emblem.

#### *Lambert Azimuthal Equal Area*

The Lambert azimuthal equal-area projection (Figure 5.33) accurately represents area in all regions of the sphere, but does not accurately represent angles. The central meridian is a straight line, others are curved. A straight line drawn through the center point is on a great circle.

Areas on the map are shown in true proportion to the same areas on the Earth. Quadrangles (bounded by two meridians and two parallels) at the same latitude are uniform in area. Directions are true only from center point. Scale decreases gradually away from center point. Distortion of shapes increases away from center point. Any straight line drawn through the center point is on a great circle. The resultant map is equal area but not conformal, perspective, or equidistant.

The Lambert azimuthal equal-area projection is sometimes used to map large ocean areas, for example the circum-Pacific. It is suited for regions extending equally in all directions from center points.

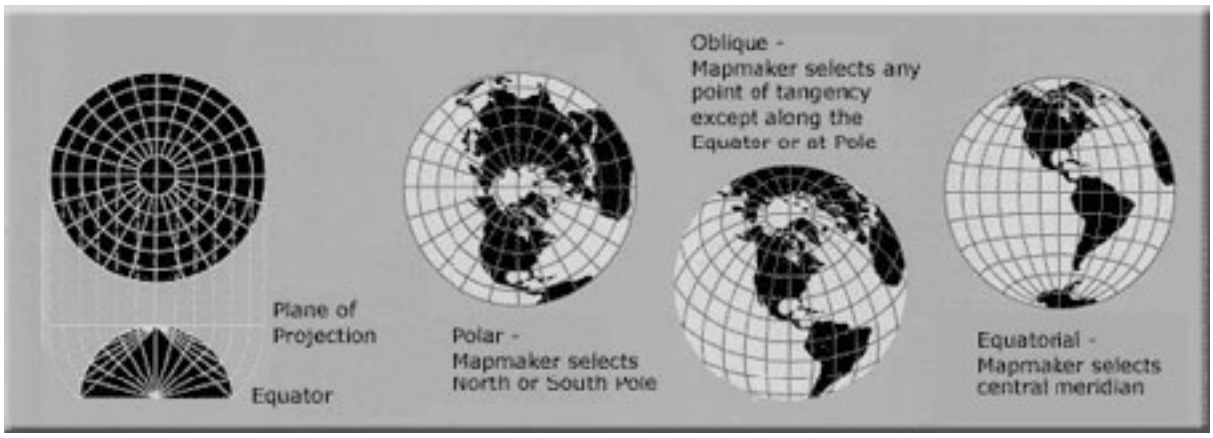



*Figure 5.33: Lambert Azimuthal Equal Area. Source: USGS.*

***Orthographic***

Orthographic projections (Figure 5.34) are used for perspective views of hemispheres. This projection was mentioned by the Greek philosopher Hipparchus in the 2nd century B.C., but it was probably known earlier.

Distances are true along the equator and other parallels. Sscale is exactly correct along any circle whose center coincides with the projection center, like parallels in the polar aspect but decreases along all lines radiating out from the centre. Any straight line through the centre point is a great circle. Area and shape are distorted. In particular, there is severe shape and area distortion near the map borders. The resultant map is perspective but not conformal or equal area.

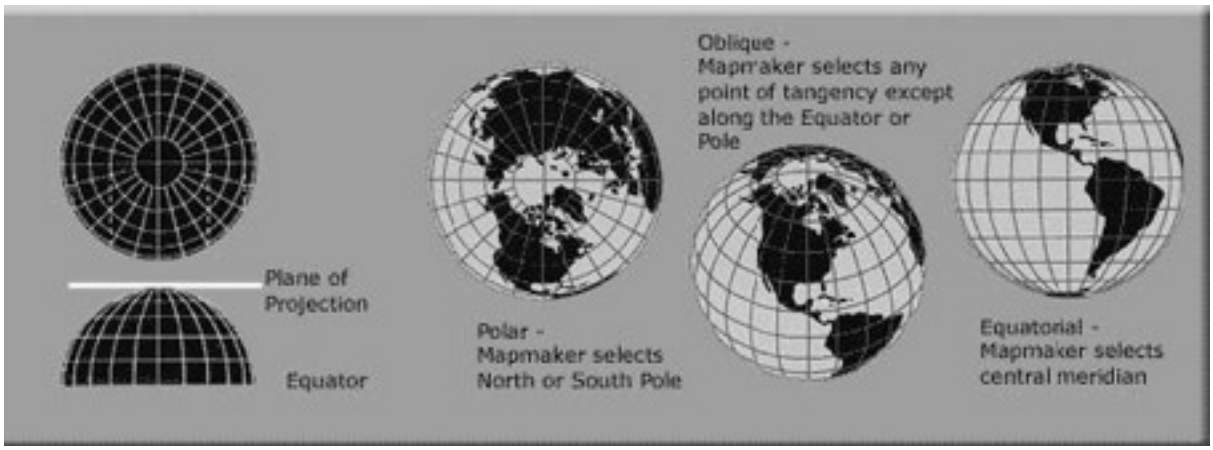



*Figure 5.34: Azimuthal Orthographic. Source: USGS.*

The orthographic projection is mainly used for perspective views of the Earth and other planets, It is used for illustration purposes, since it clearly shows the Earth as seen from space infinitely far away.

***Stereographic***

Stereographic projections (Figure 5.35) are probably the most widely used azimuthal projection and is usually attributed to Greek philosopher Hipparchus in the 2nd century B.C for use as star maps. Only in modern times has this projection been used for mapping the Earth surface and is commonly used for navigation in polar regions.

The Stereographic projection is a conformal projection - over a small area, angles are the same as on the Earth's surface. It also preserves circles, no matter how large (great circles passing on the central point are mapped into straight lines), although concentric circles on the sphere will not generally remain concentric on the map. Directions are true from the center point and scale increases away from the center point as does distortion in area and shape.

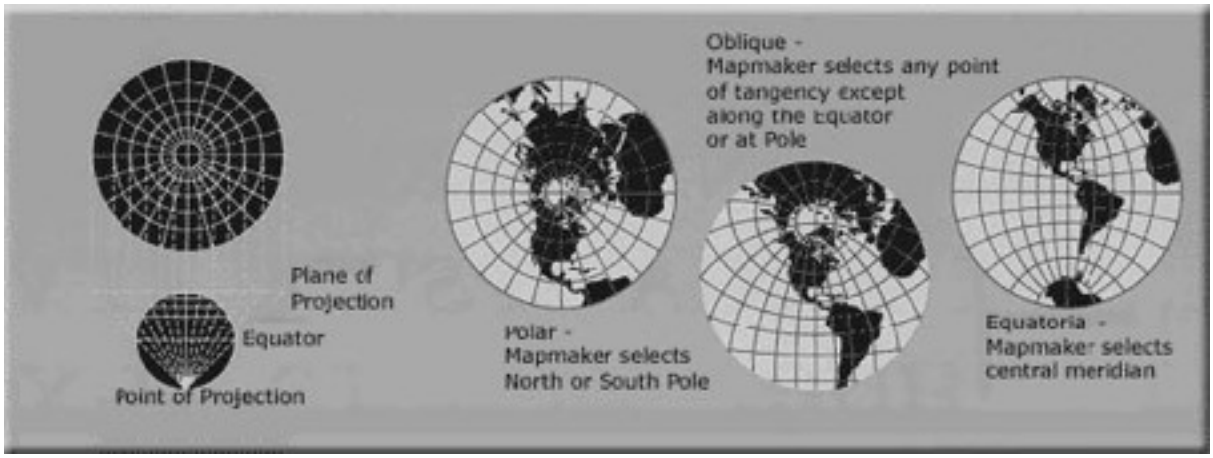



*Figure 5.35: Stereographic. Source: USGS.*

Stereographic projections are used for maps of Antarctica and the Arctic and may be used to map large continent-sized areas of similar extent in all directions. Generally used to map higher latitudes.

***Gnonomic***

The gnomonic (also called central, azimuthal centrographic or rarely gnomic) projection (Figure 5.36) is thought to be the oldest projection, ascribed to the Greek philosopher, Thales of Miletus in the 6th century B.C.

It is constructed much like the azimuthal stereographic, but the ray source is located exactly on the sphere's center; therefore it can present even less than one hemisphere at a time. Distance and shape distortions are pronounced except very near the tangent point.

This unique property of this projection is that every geodesic, including the Equator and all meridians, is mapped to a straight line, making it easy to find the shortest route between any two points. Great circle routes (the shortest distance between two points on the globe) appear as straight lines. Directions are true only from the center point of projection. The scale increases very rapidly away from center point. Distortion of shapes

and areas increases away from the center point. The resultant map is perspective (from the center of the Earth onto a tangent plane) but not conformal, equal area, or equidistant.

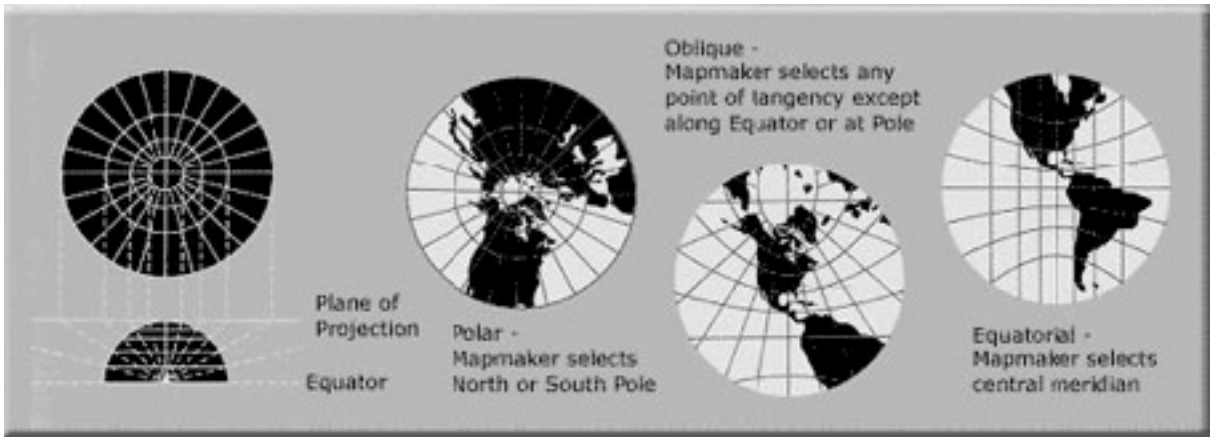



*Figure 5.36: Gnonomic. Source: USGS.*

The Gnonomic projection is used along with the Mercator by some navigators to find the shortest path between two points and in seismic work because seismic waves tend to travel along great circles.

### *Pseudo or Miscellaneous Projections*

Map projections that do not fit within these three classes of cylindrical. conical and azimuthal projections are described as Pseudo or Miscellaneous projections.

Pseudocylindrical projections are the most common sub-types of projections and resemble cylindrical projections, with straight and parallel latitude lines and equally spaced meridians, but the other meridians are curves. We will describe some pseudocylindrical projections below.

#### *Mollweide*

The Mollweide projection (Figure 5.37) is pseudocylindrical and equal-area and was developed by German Karl Mollweide in 1805.

This classic equal-area projection was designed to inscribe the world into a 2:1 ellipse, keeping parallels as straight lines while still preserving areas. All meridians but the central one map to elliptical arcs. The only standard parallels are 40°44'12"N and S. The vertical scale is compressed beyond them and stretched between them. The only two points with no distortion are the intersection of the central meridian and standard parallels. It use is primarily for world maps.

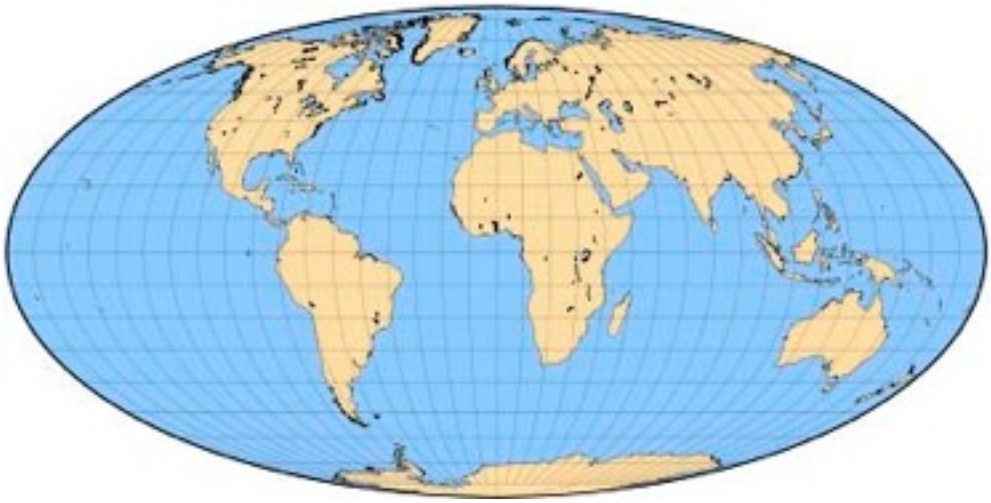


*Figure 5.37: Mollweide.*

***Robinson***

The Robinson projection (Figure 5.38) was designed by Arthur Robinson in 1963 and is a good compromise projection.

It distorts all areas and shapes a little bit, but overall the distortion is minimal (except around the edges). It is also a very attractive map. It shows parallels and the central meridian as straight lines and then uses a table of longitude coordinates for every 5° of latitude instead of a mathematical formula.

The projection distorts shape, area, scale, and distance in an attempt to balance the errors of projection properties. Directions are true along all parallels and along central meridian. Distances are constant along Equator and other parallels, but scales vary. Scale is true along 38° N & S. This projection is not conformal, equal area, equidistant, or perspective.

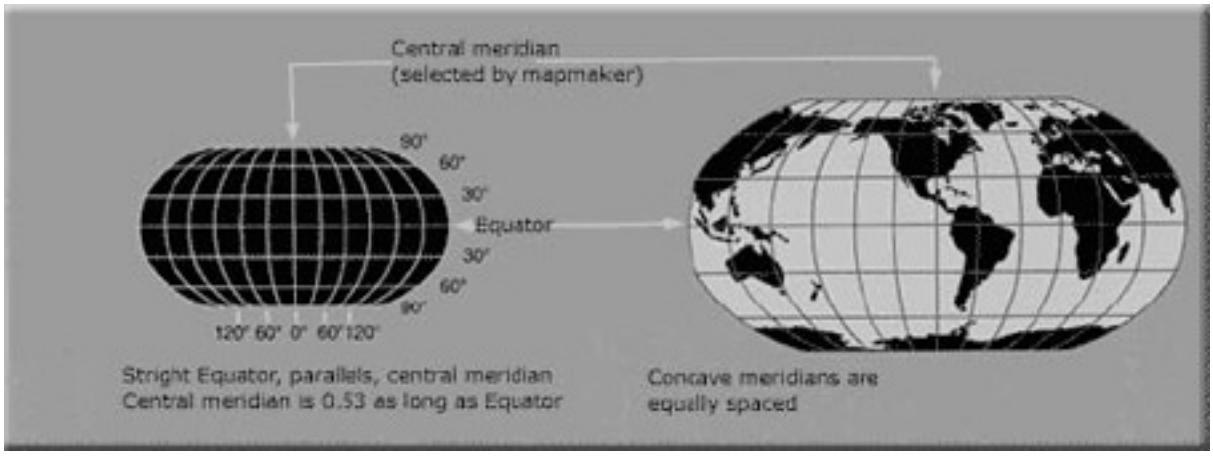



*Figure 5.38: Robinson. Source: USGS.*

The Robinson projection has become very popular after it was chosen as the reference projection for the world map of the National Geographic Society. It is used as a global mapping projection.

#### *Sinusoidal Equal Area*

The sinusoidal equal-area projection (Figure 5.39) is also commonly known as the Sanson-Flamsteed projection. It is an easily plotted equal-area projection for world maps.

It has straight parallels at right angles to a central meridian. Other meridians are sinusoidal curves. It may have a single central meridian or, in interrupted form, several central meridians. Graticule spacing retains property of equivalence of area. The areas on the map are proportional to same areas on the Earth. Distances are correct along all parallels and the central meridian/s. Shapes are increasingly distorted away from the central meridian(s) and near the poles. Scale is true only on the central meridian and the parallels. The resultant map is not conformal, perspective, or equidistant.

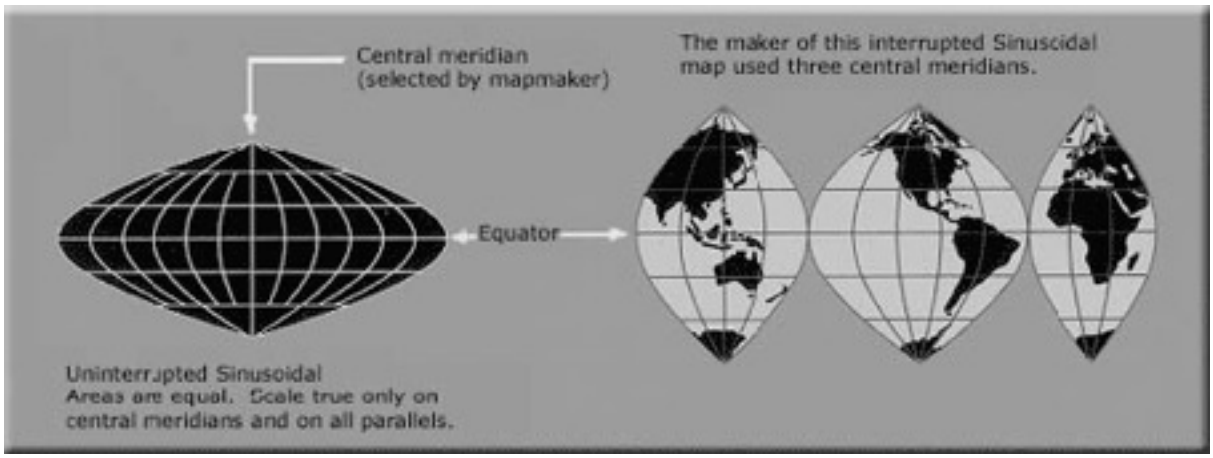



*Figure 5.39: Sinusoidal Equal Area. Source: USGS.*

This projection is frequently used in atlases to show distribution patterns and is often used in countries with a larger north-south than east-west extent such as Africa and South America.

## SUMMARY

**Cylindrical projections** are true at the equator and distortion increases toward the poles.

**Conic projections** are true along some parallel somewhere between the equator and a pole and distortion increases away from this standard.

**Azimuthal projections** are true only at their center point, but generally distortion is worst at the edge of the map.

***Summary of Projection Properties***

| Key:* = Yes x= Partly | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Projection** | **Type** | ***Conformal*** | ***Equal area*** | ***Equidistant*** | ***True direction*** | ***Perspective*** | ***Compromise*** | ***Straight rhumbs*** |
| ***Globe*** | ***Sphere*** | * | * | * | * | | | |
| ***Mercator*** | ***Cylindrical*** | * | | | x | | | * |
| ***Transverse Mercator*** | ***Cylindrical*** | * | | | | | | |
| ***Oblique Mercator*** | ***Cylindrical*** | * | | | | | | |
| ***Space Oblique Mercator*** | ***Cylindrical*** | * | | | | | | |
| ***Miller Cylindrical*** | ***Cylindrical*** | | | | | | * | |
| ***Robinson*** | ***Pseudo-cylindrical*** | | | | | | * | |
| ***Sinusoidal Equal Area*** | ***Pseudo-cylindrical*** | | * | x | | | | |
| ***Orthographic*** | ***Azimuthal*** | | | | x | * | | |
| ***Stereographic*** | ***Azimuthal*** | * | | | x | * | | |
| ***Gnomonic*** | ***Azimuthal*** | | | | x | * | | |
| ***Azimuthal Equalidistant*** | ***Azimuthal*** | | | x | x | | | |
| ***Lambert Azimuthal Equal Area*** | ***Azimuthal*** | | * | | x | | | |
| ***Albers Equal Area Conic*** | ***Conic*** | | * | | | | | |
| ***Lambert Conformal Conic*** | ***Conic*** | * | | | x | | | |
| ***Equidistant Conic*** | ***Conic*** | | | x | | | | |
| ***Polyonic*** | ***Conic*** | | | x | | | * | |
| ***Biplolar Oblique Conic Conformal*** | ***Conic*** | * | | | | | | |

*Table 5.1: Summary of Projection Properties. Source: USGS.*

A country in the tropics asks for a cylindrical projection.
A country in the temperate zone asks for a conical projection.
A polar area asks for an azimuthal projection.

| ***Summary of Areas Suitable of Mapping with Projections*** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Key:* = Yes x = Partly** | | | | | | | |
| **Projection** | **Type** | **World** | ***Hemisphere*** | ***Continent/ Ocean*** | ***Region/ sea*** | ***Medium scale*** | **Large scale** |
| ***Globe*** | ***Sphere*** | * | | | | | |
| ***Mercator*** | ***Cylindrical*** | x | | | * | | |
| ***Transverse Mercator*** | ***Cylindrical*** | | | * | * | * | * |
| ***Oblique Mercator*** | ***Cylindrical*** | | | * | * | * | * |
| ***Space Oblique Mercator*** | ***Cylindrical*** | | | | | | * |
| ***Miller Cylindrical*** | ***Cylindrical*** | * | | | | | |
| ***Robinson*** | ***Pseudo-cylindrical*** | * | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ***Sinusoidal Equal Area*** | ***Pseudo-cylindrical*** | * | | * | | | |
| ***Orthographic*** | ***Azimuthal*** | | x | | | | |
| ***Stereographic*** | ***Azimuthal*** | | * | * | * | * | * |
| ***Gnomonic*** | ***Azimuthal*** | | | | x | | |
| ***Azimuthal Equalidistant*** | ***Azimuthal*** | x | * | * | * | | x |
| ***Lambert Azimuthal Equal Area*** | ***Azimuthal*** | | * | * | * | | |
| ***Albers Equal Area Conic*** | ***Conic*** | | | * | * | * | |
| ***Lambert Conformal Conic*** | ***Conic*** | | | * | * | * | * |
| ***Equidistant Conic*** | ***Conic*** | | | * | * | | |
| ***Polyonic*** | ***Conic*** | | | | | x | x |
| ***Biplolar Oblique Conic-Conformal*** | ***Conic*** | | | * | | | |

*Table 5.2: Summary of areas suitable of mapping with projections. Source: USGS.*

# 6. Map Making

GMT is a command-line oriented package. This was a deliberated decision: a graphical user interface (GUI) would place limits on what can be done; in return it would make the system more user-friendly. The premium was placed on flexibility and performance. In order to best use GMT some knowledge of the UNIX system is required.

## Basic UNIX

If you are not familiar with Unix commands, an online manual is available by typing man command name. To exit the man page, type q

A useful web page is http://www.mines.utah.edu/gg_computer_seminar/unix

This section will introduce some helpful UNIX commands and concepts. Throughout this section the commands to be typed will, by convention be contained with []. The brackets must not be typed in. Note: UNIX is case sensitive, all commands are written in lower case.

### *Command Line Navigation*

Like in Windows, the files on a Linux system are arranged in a hierarchical directory structure. This means that the files are organized the same way as in Windows: in a tree-like pattern of directories (or folders in window speak), and those directories may contain files or other directories, which in turn may contain more files or directories, and so on...

This section of the manual will teach you how to navigate your way through the file system using the CLI.

***Find out where you are [pwd]***

The first command you should try is the **pwd** command. Type **pwd** at the command line in your Terminal window. This command shows you the present working directory and is where you are currently located in the file system e.g. /Users/studentXXX/GMTcourse/Exercise5

***List the contents of the current directory [ls]***

list all the files and folders in the current directory [**ls**]

list all files and folders, in long format [**ls -l**]

list all files and folders including ones beginning with . and .. [**ls -a**]

list all files and folders sorted by time, oldest first [**ls -r**]

list all files and folders, sorted by timestamp [**ls -t**]

list all files and folders, long format, sorted by timestamp, with most recently modified last [**ls -rtl**]

To see all the options of the **ls** command, type **man ls**. This will display information about the command **ls** such as how to use it and also what all it's options are.

***Change Directory [cd]***

So now you know how to display the contents of a directory. How do you move through the directories? You use the **cd** command (change directory).

move up to the directory that is one level higher [**cd ..**]

move to a folder within the folder you are already in [**cd foldername**] For example, say you were in your home directory /home/user/ and you wanted to change to the /mnt/cdrom/ directory, you would type:

cd /mnt/cdrom

move to the users home directory [**cd**]

HINT: A major time saver in Linux is the auto-complete function. If you press the <TAB> key, Linux will attempt to complete the command or argument you are typing. For example, type:

cd /mnt/cd  then press the <TAB> key. Linux will complete the argument for you.

***Making Directories***

Obviously you will want to create directories in your home directory to organise and store all your work.

Make a new folder/directory [**mkdir**]

make a new folder in your current location [**mkdir foldername**] For example, to create a directory called prac_1, type [**mkdir prac_1**]

make a new folder two levels higher [**mkdir ../../foldername**]

NOTE: Never put a space in a directory or file name. If you want to separate words, use the underscore symbol. There is no restriction to the length of directory and file names under Linux.

***Deleting Files and Directories***

To delete files and directories the command line, use the rm command. The rm command can be VERY, VERY dangerous, so always be careful when you use it. Unlike in Windows or the Linux GUI, using rm deletes a file or directory for good, it's gone and it ain't never coming back!

Delete a file [**rm file_name**]

Delete a directory and all it's contents [**rm -rf directory_name**] Here the –r option stands for recursive.

Delete all the files with the same file extension [**rm *.ext**], where * is a wildcard

BE CAREFUL If you were to type [**rm ***] you would delete EVERY file in the current directory. If you don't want to accidentally delete everything when using the * wildcard, I would suggest to always use the –i option with the rm command, this will ask you before deleting a file e.g. [**rm -i *.ext**]

***Copying and Moving***

To copy a file under Linux, use the **cp** command.

Copy a file and rename [**cp file1.ext file2.ext**] Now there will be two identical files name file1.ext and file2.ext.

To move a file under Linux, use the mv command.

Move the contents of the file file1.ext [**mv file1.ext file2.ext**]. Now there will only be one file with the name file2.ext.

Move a file to another directory [**mv file1.ext /foldername/foldername/**]

## *Other useful UNIX commands and processes*

***Shell redirection: [<] [>] [>>]***

Most UNIX processes (commands and programs) have input and output. If no information about input and output is passed to UNIX, it will use the standard input or output if necessary. The redirection operators redirect the input and output of processes:

process1 < file1 file1 is input for process1.

process2 > file2 process2 outputs to file2.

process3 >> file3 process3 appends its output to file3.

Examples:

cat <[file1] file1 is input for cat and will be displayed on screen.

cat <[file1] >[file2] reads file1 and writes it to disk as file2.

***Pipes: [ | ]***

Pipes allow data to flow from one process to another:

process4 | process5 The output from process4 is the input for process5.

Example:

cat [file1] | grep [pattern] will pass line after line from file1 to grep.

***Permissions:***

All files on a UNIX system have permission rights associated with them. These permissions can be changed with the command chmod. All script files must be executable. This can be achieved by typing:

Make a file executable [**chmod +x filename**]

Make a file readable, writable and executable for user, group and everyone [**chmod ugo+rwx filename**]

Many different combinations are possible, the options are summarised in the table below.

| Letter | Meaning |
|---|---|
| u | Owner of file |
| g | Group to which the owner belongs |
| o | Everyone |
| a | Means u+g+o |
| **Operator** | **Meaning** |
| + | Add permission |
| - | Delete Permission |
| = | Set permission for specific user |
| **Letter** | **Meaning** |
| r | Read permission |
| w | Write permission |
| x | Execute permission |

*Table 6.1: Summary of permission changing options*

***UNIX Summary***

The following list of UNIX commands is not complete; only a selection of commands, which the author considers to be helpful for the understanding and reproduction of the processing done in this report, is presented.

| LINUX Command | What does it do? | Type in Terminal | Comments |
|---|---|---|---|
| | | | |
| **pwd** | Tells you what directory you are in, prints the working directory | pwd | */home/user/* |

| | | | |
|---|---|---|---|
| **whoami** | Find out who you are | whoami | *marias* or *belinda* or *craig* or *christian* |
| **cd** | Changes the working directory | cd *sapphire1* | If you just type cd, it will take you to your home directory |
| **cd ..** | Moves up a directory | cd .. | Move up 1 directory |
| **ls** | Lists all the files in the directory you are in | ls | If you place *ps for example, only the .ps files will be listed |
| **mk** | Makes/creates | mk *file.ext* | A new file called *file.ext* |
| **mkdir** | Makes/creates a directory | mkdir *sapphire1* | A new directory called *sapphire1* |
| **rm** | Removes/deletes | rm *file.ext* | *file.ext* is removed |
| **rmdir** | Removes/deletes an empty directory | rmdir *sapphire1* or rm –rf *sapphire1* | Directory *sapphire1* is removed (only when it's empty) |
| **mv** | Moves/cuts and pastes to as another name or to a different directory | mv *file1.ext file2.ext* | *file1.ext* is renamed to *file2.ext* |
| **cp** | Copy as another name or to a different directory | cp *file1.ext file2.ext* | Using cp –a ( or –R) copies whole directories and a good way of backing up |
| **lpr** | Prints a file | lpr *file.ext* | |
| **lpq** | Lists the queue for the printer | lpq | Append a –P*printername* for the queue of that particular printer |
| **lsd** | Lists the directories in that directory | lsd | |
| **ls *** | Lists all files (excluding hidden ones) including the files in the directories | ls * | |
| **ls -a** | Lists all files including hidden files (start with .) | ls –a | |
| **ls -rtl** | The last thing that you created will be listed in long format (l), in order of creation (t), then in reverse order (r) | ls –rtl | Can type ls -rtl *ps and this will list the last postscript file you created last |
| **history** | Displays the history of what you have typed | history | h –10 gives the last 10 things |
| **man** | The online manual, gives info about a certain command | man *command* | Detailed description of the command is given e.g. man lpr |
| **more** | Displays the file in the terminal | more *file.ext* | Useful for a quick look at the file to see if it's the right one |
| **head** | Displays the first 10 lines (default) of the file, in the terminal | head *file.ext* | Useful for a quick look at the beginning of a file |
| **tail** | Displays the last 10 lines (default) of the file, in the terminal | tail *file.ext* | Useful for a quick look at the end of a file |
| **talk** | To chat to someone logged onto the network | talk *belinda* | The other person has to be logged on |
| **finger** | Lists data about the other users that are logged in | finger | finger *craig* gives more info about that particular user |
| **passwd** | Allows the user to change their password | passwd | Must be 6 characters long and at least 2 letters and one number and cannot be a derivative of your login name |
| **vim** | Allows you to create and edit a file in the terminal | vim *file.ext* | |
| **Ctrl C** | Goes to new terminal line – cancels what you are doing | Ctrl C | |
| **Ctrl K** | Kill to end up line forwards | Ctrl K | |
| **Ctrl A** | Goes to the beginning of the line | Ctrl A | |
| **Ctrl E** | Goes to the end of the line | Ctrl E | |
| **Tab key** | Attemps a filename completion of the current word | Tab key | |

| | | | |
|---|---|---|---|
| **grep** | Allows you to search for certain strings of characters | grep '*coffee*' | Displays the line that has the specified word or phrase in the terminal |
| **sort** | Sorts in alphabetical or numerical order | sort *file.ext* | |
| **whereis** | Locates a utility | whereis *craig* | |
| **&** | Means that the command will run in the background | gedit *file.ext* & | |
| **gunzip** | Unzips a .gz file to make them readable | gunzip *file.ext.gz* | A file followed by a .gz means that it requires this command to make it readable again. Its basically a compressed file |
| **tar** | Decompresses .tar files to make them readable | tar –xvf *file.ext.tar* | Another compressor |
| **ssh** | To remotely log into another computer using a secure shell | ssh *craig@amethyst.es.usyd.edu.au* | A secure shell |
| **mount** | Mounts a device | mount /mnt/zip | This mounts the zip drive |
| **umount** | Unmounts a device | umount /mnt/zip | This unmounts the zip drive but only if all other processes involving the zip drive are not running. –f forces it to mount (only as root) |
| **slogin** | To remotely log into another computer using a secure login | slogin *sapphire* | |
| **cp** | Copy files from one place to another | cp *file.ext marias/dir/* | Append –r to copy several files |
| **scp** | Means secure copy so you can copy files from one place to another securely | scp *file.ext marias@sapphire.es.usyd.edu.au:~marias/dir/* | Append –r to copy several files |

*Table 6.2: Summary of basic UNIX commands*

## GMT FUNDAMENTALS

### *GMT Default Parameters*

The GMT defaults contain pre-set parameters for the GMT system such as annotation and layout parameters for plotting, colour system parameters, postscript parameters, parameters for user input and output and projection parameters. In total there are more than 100 of these default parameters. These reside in a file called .gmtdefaults4 as a simple ASCII file. The . in front of the filename is deliberate - it denotes that the GMT defaults file is a system file and hides it from general view.

A user will typically have a "master" .gmtdefaults4 file in their home directory and possibly more specialised files in sub-directories. If no .gmtdefaults4 file exists in the current working directory then GMT will search the home directory. If a file does not exist there, it will use the system defaults.

To produce a .gmtdefaults4 file, type the following command in the terminal:

```
gmtdefaults -D > .gmtdefaults4
```

Here is an example of a .gmtdefaults4 file:

```
#	GMT-SYSTEM 4.4.0 Defaults file
#-------- Plot Media Parameters ------------
PAGE_COLOR		= 255/255/255
PAGE_ORIENTATION	= portrait
PAPER_MEDIA		= a4
#-------- Basemap Annotation Parameters ------
ANNOT_MIN_ANGLE		= 20
ANNOT_MIN_SPACING	= 0
ANNOT_FONT_PRIMARY	= Helvetica
ANNOT_FONT_SIZE_PRIMARY	= 10p
ANNOT_OFFSET_PRIMARY	= 0.2c
ANNOT_FONT_SECONDARY	= Helvetica
ANNOT_FONT_SIZE_SECONDARY	= 16p
ANNOT_OFFSET_SECONDARY	= 0.2c
DEGREE_SYMBOL		= ring
HEADER_FONT		= Helvetica
HEADER_FONT_SIZE	= 36p
HEADER_OFFSET		= 0.5c
LABEL_FONT		= Helvetica
LABEL_FONT_SIZE		= 24p
LABEL_OFFSET		= 0.3c
OBLIQUE_ANNOTATION	= 1
PLOT_CLOCK_FORMAT	= hh:mm:ss
PLOT_DATE_FORMAT	= yyyy-mm-dd
PLOT_DEGREE_FORMAT	= +ddd:mm:ss
Y_AXIS_TYPE		= hor_text
#-------- Basemap Layout Parameters ---------
BASEMAP_AXES		= WESN
BASEMAP_FRAME_RGB	= 0/0/0
BASEMAP_TYPE		= fancy
FRAME_PEN		= 1.25p
FRAME_WIDTH		= 0.2c
GRID_CROSS_SIZE_PRIMARY	= 0c
GRID_PEN_PRIMARY	= 0.25p
GRID_CROSS_SIZE_SECONDARY	= 0c
```

```
GRID_PEN_SECONDARY = 0.5p
MAP_SCALE_HEIGHT = 0.2c
POLAR_CAP = 85/90
TICK_LENGTH = 0.2c
TICK_PEN = 0.5p
X_AXIS_LENGTH = 25c
Y_AXIS_LENGTH = 15c
X_ORIGIN = 2.5c
Y_ORIGIN = 2.5c
UNIX_TIME = FALSE
UNIX_TIME_POS = BL/-2c/-2c
UNIX_TIME_FORMAT = %Y %b %d %H:%M:%S
#-------- Color System Parameters -----------
COLOR_BACKGROUND = 0/0/0
COLOR_FOREGROUND = 255/255/255
COLOR_NAN = 128/128/128
COLOR_IMAGE = adobe
COLOR_MODEL = rgb
HSV_MIN_SATURATION = 1
HSV_MAX_SATURATION = 0.1
HSV_MIN_VALUE = 0.3
HSV_MAX_VALUE = 1
#-------- PostScript Parameters -------------
CHAR_ENCODING = ISOLatin1+
DOTS_PR_INCH = 300
N_COPIES = 1
PS_COLOR = rgb
PS_IMAGE_COMPRESS = lzw
PS_IMAGE_FORMAT = ascii
PS_LINE_CAP = butt
PS_LINE_JOIN = miter
PS_MITER_LIMIT = 0
PS_VERBOSE = FALSE
GLOBAL_X_SCALE = 1
GLOBAL_Y_SCALE = 1
```

```
#-------- I/O Format Parameters -------------
D_FORMAT            = %lg
FIELD_DELIMITER                 = tab
GRIDFILE_SHORTHAND = FALSE
GRID_FORMAT             = nf
INPUT_CLOCK_FORMAT= hh:mm:ss
INPUT_DATE_FORMAT   = yyyy-mm-dd
IO_HEADER           = FALSE
N_HEADER_RECS           = 1
OUTPUT_CLOCK_FORMAT     = hh:mm:ss
OUTPUT_DATE_FORMAT          = yyyy-mm-dd
OUTPUT_DEGREE_FORMAT    = +D
XY_TOGGLE               = FALSE
#-------- Projection Parameters -------------
ELLIPSOID           = WGS-84
MAP_SCALE_FACTOR    = default
MEASURE_UNIT            = cm
#-------- Calendar/Time Parameters ----------
TIME_FORMAT_PRIMARY         = full
TIME_FORMAT_SECONDARY   = full
TIME_EPOCH              = 2000-01-01T12:00:00
TIME_IS_INTERVAL        = OFF
TIME_INTERVAL_FRACTION  = 0.5
TIME_LANGUAGE           = us
TIME_UNIT           = d
TIME_WEEK_START             = Sunday
Y2K_OFFSET_YEAR             = 1950
#-------- Miscellaneous Parameters ----------
HISTORY                 = TRUE
INTERPOLANT             = akima
LINE_STEP           = 0.025c
VECTOR_SHAPE            = 0
VERBOSE                 = FALSE
```

The parameters are pretty self-explanatory but for a listing of a description of all of the parameters in the .gmtdefaults file, type the following in the terminal:

```
man gmtdefaults
```

The parameters in the GMT defaults file can be changed by the user. This can be done in numerous ways:

1. Open the .gmtdefaults4 file and manually change an entry
2. Use the **gmtset** command to set a value for a new parameter on the fly. This can either be done on the command line or in a script:

```
gmtset ANOT_FONT Times-Bold ANOT_FONT_SIZE 12
```

3. For some parameters, values can be changed within various GMT commands (more advanced step - no part of this course)

***Map boundary parameters in GMT: GMT defaults***

GMT is a map-making software package and as such provides great flexibility in the way maps are created and the way they look. An important aspect of map creation is the boundary of the map, i.e. what the border around the map looks like, how big or small the annotations and labels are, how thick or thin and long the tick marks are, etc.

Figures 6.1, 6.2 and 6.3 illustrate what each GMT plot related default parameter controls. For a detailed listing of what each parameter does and how values are defined, see the man page. Here are a few examples:

HEADER_FONT defines the font type that will be used for the title of your plot. GMT supports 35 fonts and they can be defined either by their name (e.g. Helvetica, Times-Roman) or by a number corresponding to font name (e.g. 1 = Helvetica, 4 = Times New Roman). The same applies for ANOT_FONT and LABEL_FONT.

HEADER_FONT_SIZE defines the font size in pts for the title of a plot. e.g. 12, 36, 72. The same applies for ANOT_FONT_SIZE and LABEL_FONT_SIZE.

BASEMAP_TYPE defines what the basemap frame looks like. Options include fancy (as seen in Figure 6.1), plain (as seen in Figure 6.2), inside (annotations and tick marks are located within the map) and graph (as seen in Figure 6.3).

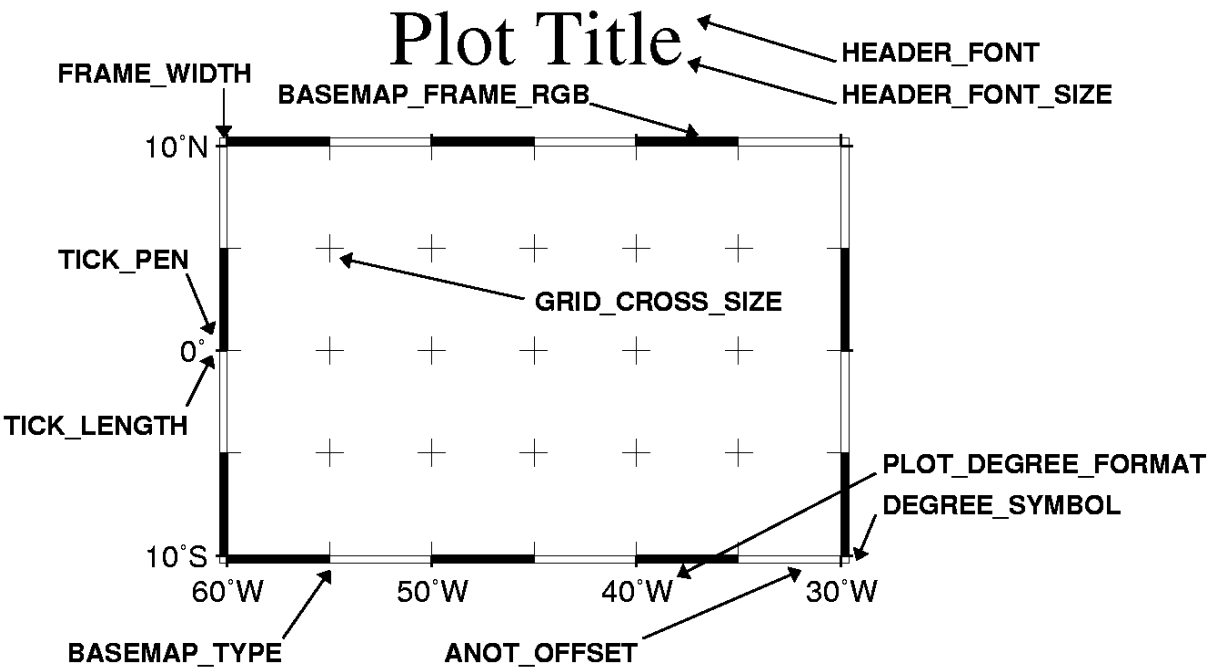



*Figure 6.1: Plot related GMT default parameters.*

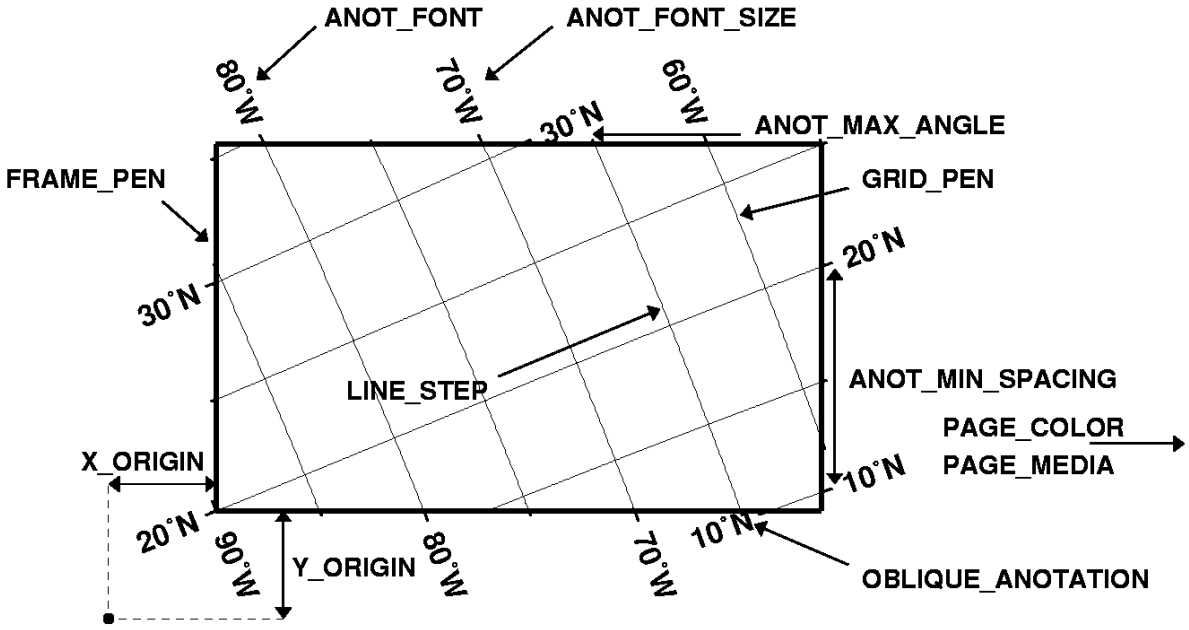



*Figure 6.2: Plot related GMT default parameters.*

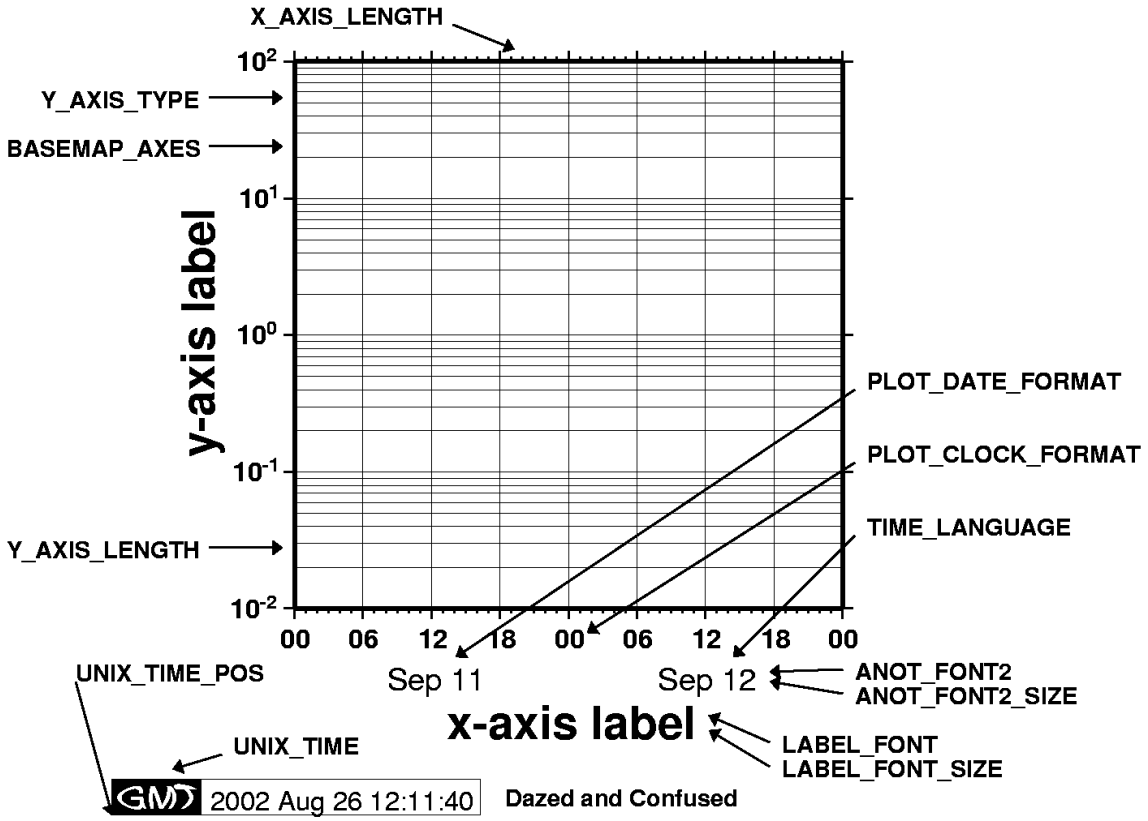



*Figure 6.3: Plot related GMT default parameters.*

***Defining geodetic parameters in GMT: GMT Defaults***

GMT can handle different ellipsoid models of the Earth. The model you would like to use is defined in the GMT defaults file. There are close to 100 different ellipsoids available in GMT. The default is WGS-84. The ellipsoid is defined using the ELLIPSOID parameter followed by the name of the ellipsoid.

e.g. ELLIPSOID = WGS-84

***Measurement Units in GMT: GMT Defaults***

GMT can accept cm, inch, meter, or point as measurement units for maps and calculations. This can be set in the MEASURE_UNIT parameter. The default is cm.

## *Creating a map boundary in GMT: -B option*

The -B option is the most complicated option in GMT but most examples of its usage are quite simple. It controls the map boundary annotation, tick mark and gridline intervals as well as the labelling of axes and setting the title of the plot.

The general syntax for -B is as follows:

[p|s]xinfo[/yinfo[/zinfo]][:."Title":][W|w][E|e][S|s][N|n][Z|z[+]]

It looks very complicated but when you break it down into its components, it is not that difficult.

p and s just specifies whether you are using the primary or secondary annotation information. In the vast majority of cases (and all the ones presented in this course) we will only be using primary annotation information (i.e. only have one set of annotations and labels per axis). The default is just primary so we can ignore this step. See Figure 4.3 for an example of a situation where you may require secondary annotation information.

xinfo, yinfo and zinfo are the annotation, tick mark and gridline interval parameters and the label for each axis. There is a lot going on in this component of -B so let's expand it out step-by-step.

xinfo (and corresponding yinfo or xinfo for y and z axes , respectively) can be expanded out to:

info[:"axis label":][:,"unit label"]

where

info is defined as one or more of [t]stride[u]

axis label is the label you want to give your axis

unit label is the unit that you wanted to append to your axis

[t]stride[u] is where all the intervals are set for annotations, tick marks and gridlines:

where

t = type such as a for annotation, f for tick mark and g for gridline

stride = interval or spacing

u = units (if you wanted to change the units from the default unit)

An example of the [t]stride[u] would be a10f5g10 which stands for annotate every 10 degrees, place a tick mark every 5 degrees and create gridlines every 10 degrees. In creating maps, usually only the [t]stride[u] parameters will be used instead of axis labels and units (you don't really label the latitude and longitude axes!).

However, if you wanted to plot a graph, you will likely need to complete more of the xinfo string. For example: a10f5g10:"Sea level"::,m: would create annotations every 10 m, tick marks every 5 m and gridline every 10 m and then label the axis "Sea level".

If we wanted to have different parameters for the x-axis and y-axis, we do this by putting a / after the info section. For example: a30f10g30/a10f5g10 will give the a30f10g30 parameters to the x-axis and a10f5g10 to the y-axis. We add another / if we wanted to define parameters for the z-axis.

You can also append a title to your plot using [:."Title":]. An example would be :."Long-term Sea level for the last 200 Million Years":

By default, annotation and label information for all axes are plotted. This is denoted by [W|w][E|e][S|s][N|n][Z|z[+]] in the -B option syntax. WESN which stands for West, East, South, North. The Z is if you have a 3D plot. To change the axes that you would like to annotate and label, change from uppercase to lowercase. For example, if you wanted annotations and labels only along the western and southern boundary but you still wanted a boundary outline draw on the eastern and northern boundary, you would type WeSn. If you wanted no boundary, annotations or labels on the eastern and northern boundaries, you would omit e and n altogether e.g. WS.

***Exercise 6.1:*** Create a -B option which will define annotations every 10 degrees for latitude and longitude, tick marks every 5 degrees and gridlines every 10 degrees only along the western and southern boundaries

-Ba10f5g10WeSn

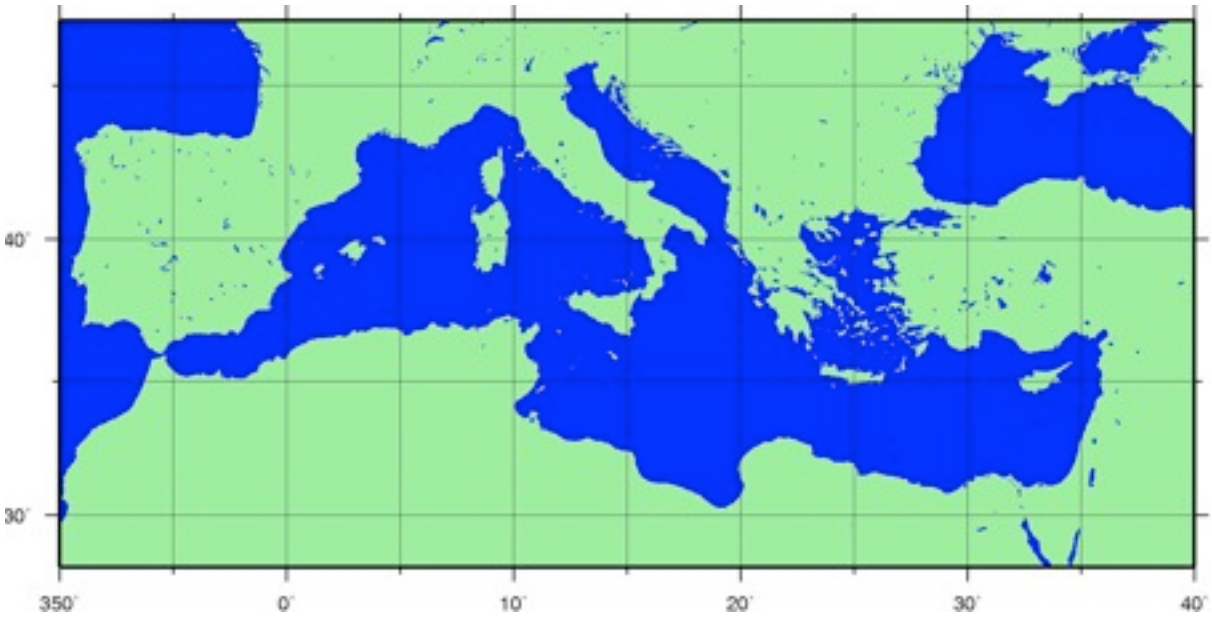



*Figure 6.4: Map of the Mediterranean with the boundary annotations listed in Exercise 4.1.*

***Exercise 6.2:*** Create a -B option which will define annotations every 10 degrees, tick marks every 5 degrees and gridlines every 10 degrees for latitude only. For longitude, annotations every 30 degrees, tick marks every 10 degrees and gridlines every 30 degrees. The annotations and tick marks should only be along the western and southern boundaries and there should be a title of "Present Day Global Distribution of Bearded Geologists".

-Ba30f10g30/a10f5g10:."Present Day Global Distribution of Bearded Geologists":WeSn

***Exercise 6.3:*** Create a -B option which will define annotate every 10 million years and place tick marks every 1 million year on the x-axis and annotate every 100 m and place tick marks every 10 m on the y-axis. The x-axis should be labelled as "Time (millions of years)" and the y-axis should be labelled as "Sea level (m)". The title of the plot should be "Long-term Sea level for the last 200 Million Years". Annotations, tickmarks and labels should only be along the western and southern boundary. There should be no boundary outline in the north and east.

-Ba10f1:"Time (millions of years)":/a100f10:"Sea level (m)"::."Long-term Sea level for the last 200 Million Years":WS

The -B option is used in all GMT mapping commands (e.g. psbasemap, pscoast, psxy, grdimage). It is only a mandatory option for psbasemap. It is optional for all other commands. The way that the title, labels, annotations, tick marks and gridlines look like are determined in the GMT defaults file.

### *Defining the map frame in GMT: -R option*

The -R option specifies the region of interest. The region can be specified in 2 ways (Figure 6.5):

- ☑ As min and max coordinates (most common method), or
- ☑ As coordinates of lower left and upper right corners (append r to signal intent)

To specify min and max coordinates, you do the following: -Rxmin/xmax/ymin/ymax (in geographical coordinates, it corresponds to -Rwest/east/south/north). For example, the image to the left in Figure 6.5 would have a region of interest of -R-90/-70/18/36

To specify lower left and upper right corners, you do the following: -Rxlleft/ylleft/xuright/yuright and then add a lower case r at the end to signal your intent. For example, the image to the right in Figure 6.5 would have a region of interest of -R-90/20/-60/39r

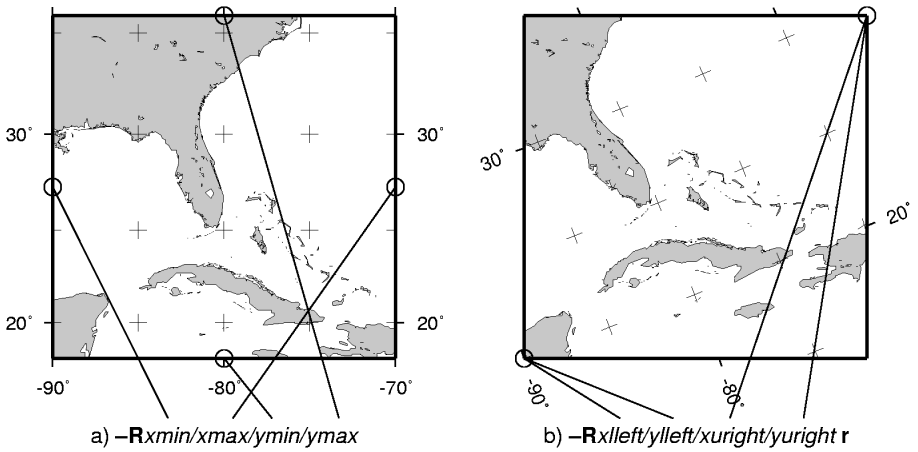



*Figure 6.5: Two maps of Florida and Cuba showing the two ways of specifying the region of interest. The methods you use is dependent on the map projection you use.*

***Exercise 6.4:*** What is the region of interest in Figure 6.1?

-R-60/-30/-10/10

***Exercise 6.5:*** What is the region of interest in Figure 6.2?

-R-90/20/-55/25r

There is a minor complication when defining the region of interest and that has to do with the way that longitude is defined. In cartography, longitude are usually represented as values from -180/+180 (-180/+180 corresponding to the International Dateline). However, mathematically the globe is thought of as a sphere and so longitudes are represented as values from 0/360 (0/360 corresponding to the Prime Meridian). By default, GMT uses the mathematical 0/360 representation of longitude. This can be changed to the -180/180 format in the GMT defaults file under OUTPUT_DEGREE_FORMAT.

***Exercise 6.4:*** What is the region of interest in Figure 6.4?

Reading off the map, the western boundary is 350 degrees, the eastern boundary is 40 degrees, the southern boundary is 28 degrees and the northern boundary is 47 degrees. However, we cannot assign the -R parameters to be -R350/40/28/47 because 350 is greater than 40 (remember, it corresponds to xmin/xmax/ymin/ymax). Instead we have to define the western boundary as -10 instead. So the region of interest is:

-R-10/40/28/47

When dealing with global maps, there are two useful shorthands: -Rg defines -R0/360/-90/90 and -Rd defines -R-180/180/-90/90

### *Defining map projections in GMT: -J option*

GMT can handle 28 map projections (Figure 6.6). Map projection is specified using the -J option. Each projection has a letter associated with it. For example, Mercator is Mm, Orthographic is Gg, Mollweide is Ww and linear is Xx.

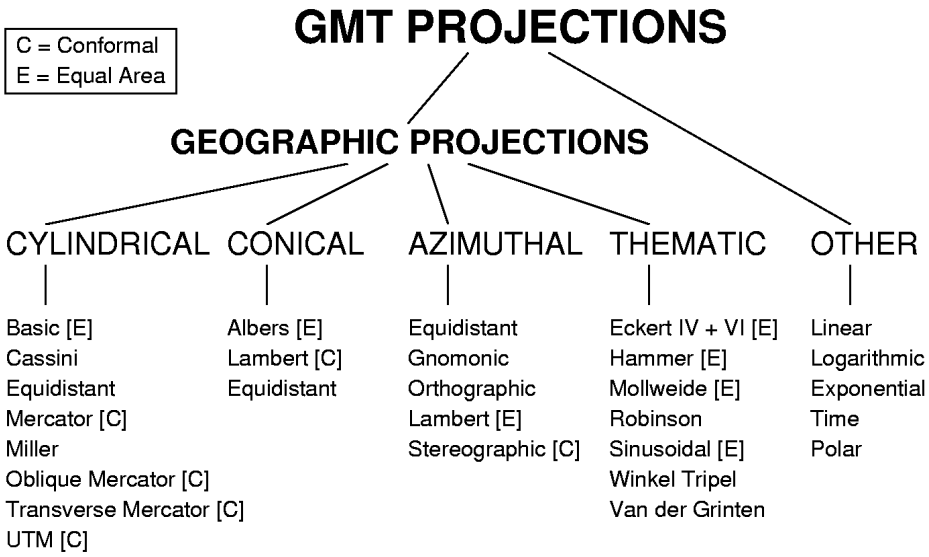



*Figure 6.6: The 28 map projections that GMT can handle.*

There are 2 general approaches:

☑ Specify the map scale **–Jδ[parameters/]scale**

☑ Specify the map width **–JΔ[parameters/]width**

where: **δ** or **Δ** specifies which projection you are using (denoted by unique letter for each projection)

**parameters** (0 or more) depends on δ

**scale** is the map scale and depends on the region of interest

**width** is the map width e.g. 12 corresponds to 12 cm (if cm if your default measurement unit)

In both cases, the map height is automatically calculated from the scale or width of the map

#### *Linear projection -Jx or -JX*

The linear projection is used when you want to plot graphs of any sort. To create a linear projection where the width of the map is defined use:

–JXwidth[/height]

For example, -JX12 creates a linear projection with a map width of 12 cm (i.e. when I print out the map and I measure the x-axis, it will be 12 cm). If you wanted to define a different width for the height, just add /height. For example, -JX12/6 creates a linear projection with a map width (x-axis) of 12 cm and a map height (y-axis) of 6 cm.

To create a linear projection where the scale of the map is defined use:

–Jxxscl[/yscl]

The scale in plot units per user unit. The user unit is given in -R (e.g. 0.1). Scale may also be 1:xxxx (e.g. 1:10000). For example, -Jx0.1 means a linear projection with a map scale of 0.1. Again, you can define a different scale for the height (y-axis).

***Mercator projection -JM or -Jm***

The Mercator projection is a conformal (preserves shape) and uses a cylindrical projection surface (see Map Projections section of notes for more details). It is still one of the most commonly used projections.

To create a Mercator projection where the width of the map is defined use:

–JMwidth

The height of the map is calculated automatically based on the -R values. For example, -JM6.5 will create a Mercator projected map with a width of 6.5 cm.

To create a Mercator projection where the scale of the map is defined use:

–Jmscale

The height of the map is calculated automatically based on the -R values. Scale can be

defined as plot units per degree or as 1:xxxxx. For example, -Jm0.2 will create a Mercator projected map with a scale of 0.2.

***UTM projection -JU or -Ju***

The UTM or Universal Transverse Mercator projection is a conformal (preserves shape) and uses a cylindrical projection surface (see Map Projections section of notes for more details). It has become the most widely used projection in the world.

To create a UTM projection use either:

-JUzone/width or -Juzone/scale

To determine the zone and how the zone is determined, you need to look up the UTM zone map (see Map Projections section of notes).

An example of a UTM projection for Australia with a width of 12 cm would be -JU53/12. When using the -R option with the UTM projection, it is best to define -Rxlleft/ylleft/xuright/yuright

***Conical projections***

GMT handles several conical projections. The syntax for these projections is as follows:

–JΔlon0/lat0/slat1/slat2/width

or

–Jδlon0/lat0/slat1/slat2/scale

Depending on Δ (or δ) we get

B (or b): Albers Equal-Area

D (or d): Equidistant

L (or a): Lambert Conformal

As described earlier, conical projections are defined by either one or two standard parallels (where the cone intersects the surface) (Figure 6.7).

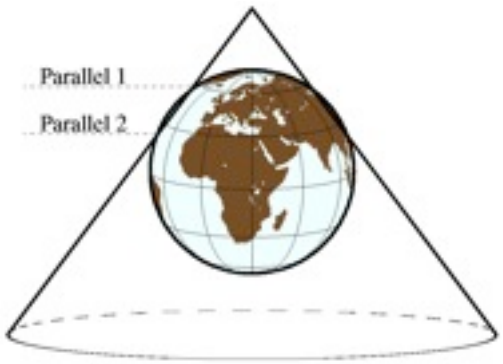



*Figure 6.7: Conical projection*

This is why we have several parameters to define for conical projections:

$long_0$ and $lat_0$ define the projection centre (usually the centre of the map)

$slat_1$ is the first standard parallel

$slat_2$ is the second standard parallel

An example of an Albers Equal-Area projection for the continental US is -JB265/35/33/45/12 with a corresponding region of interest being -R230/300/25/50. The projection centre is 265/35 which is roughly in the middle of the area of interest. The two standard parallels (i.e. the places with no distortion) are along 25 degrees and 50 degrees latitude. The other conical projections follow the same format.

***Azimuthal projections***

GMT handles several azimuthal projections. The syntax for these projections is as follows:

–JΔlon$_o$/lat$_o$/width

or

–Jδlon$_o$/lat$_o$/scale

Depending on Δ (or δ) we get

A (or a): Lambert Equal-Area

E (or e): Equidistant

G (or g): Orthographic

S (or s): Stereographic Conformal

F (or f): Gnomonic (takes lath/scale)

As described earlier, azimuthal projections define one plane (can have more) tangent to the surface (Figure 6.8). lon$_o$/lat$_o$ corresponds to this point.

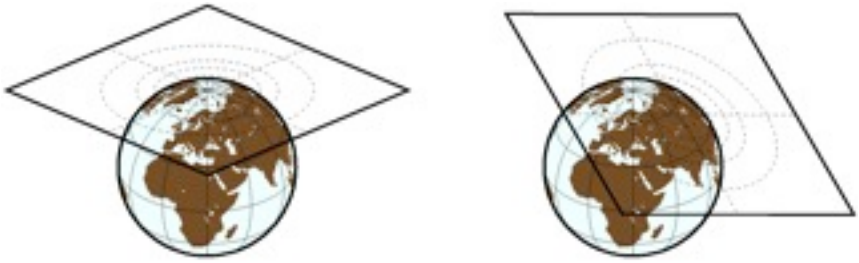


*Figure 6.8: Azimuthal projection*

An example of an orthographic projection is a globe centered on the central Pacific with a width of 10 cm. You would use a global -Rg and -JG220/-10/10.

***Thematic (global) projections***

GMT handles several thematic/global projections. Most of them have the following syntax:

–JΔlon$_o$/width

or

–Jδlon$_o$/scale

Depending on Δ (or δ) we get

H (or h): Hammer

R (or r): Robinson (National Geographic Society)

I (or i): Sinusoidal

W (or w): Mollweide

$lon_0$ corresponds to the central meridian.

### *Creating a basemap using psbasemap*

psbasemap is the simplest GMT command. It plots a basemap given a projection (-J), a region of interest (-R) and boundary parameters (-B).

***Exercise 6.5:*** Create a basemap that corresponds to a linear projection with annotations, tickmarks and gridlines every 10 units on the x-axis and every 20 units on the y-axis. Only the western and southern boundaries should be plotted. You can chose whichever map width or scale you want but the x-axis width should be different to the y-axis height. The x-label is time (Ma) and y-axis is convergence rate (mm/yr). The data that needs to be plotted has minimum value of 0/0 and a maximum value of 200/367.

psbasemap -R0/200/0/367 -JX16/8 -Ba10f10g10:"Time (Ma)":/a20f20g20:"Convergence Rate (mm/yr)":WS > psfile.ps

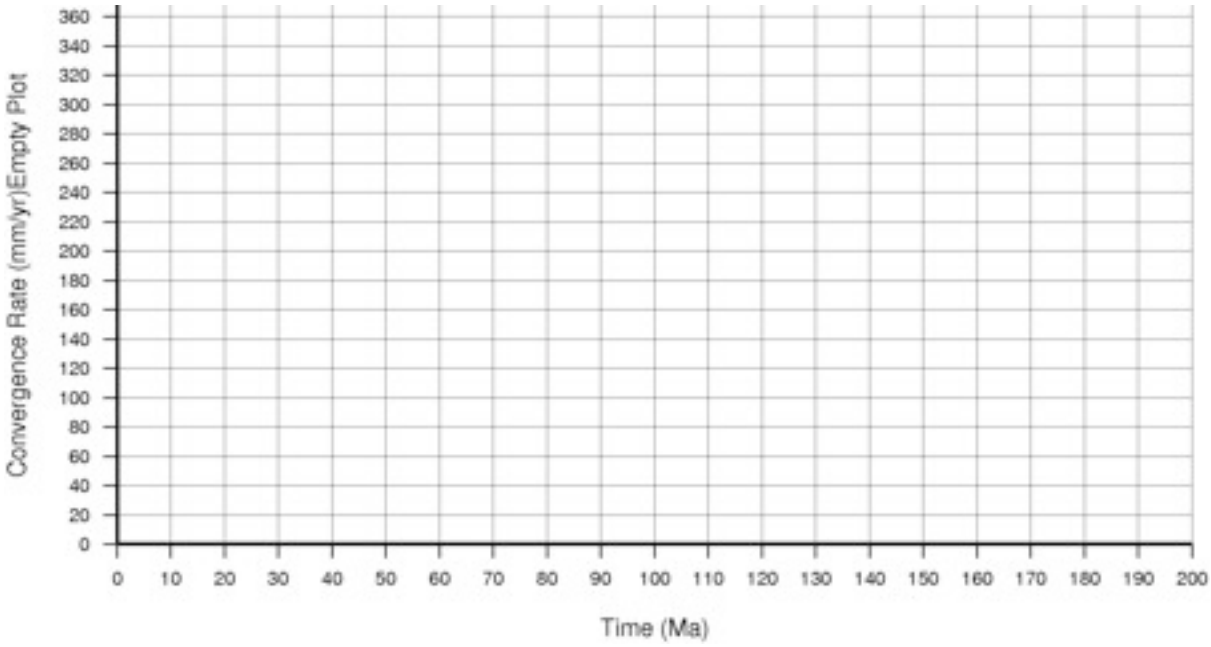



*Figure 6.9: Output of Exercise 6.5*

***Exercise 6.6:*** Create a basemap that corresponds to a Mercator projection, region of interest being the Australian continent, map width of 12 cm with annotations, tickmarks and gridlines every 10 degrees latitude and every 20 degrees longitude. The title of the basemap should be "Australia should be here".

psbasemap -R110/155/-45/-10 -JM12 -Ba20f20g20/a10g10f10:."Australia should be here": > basemap_australia.ps

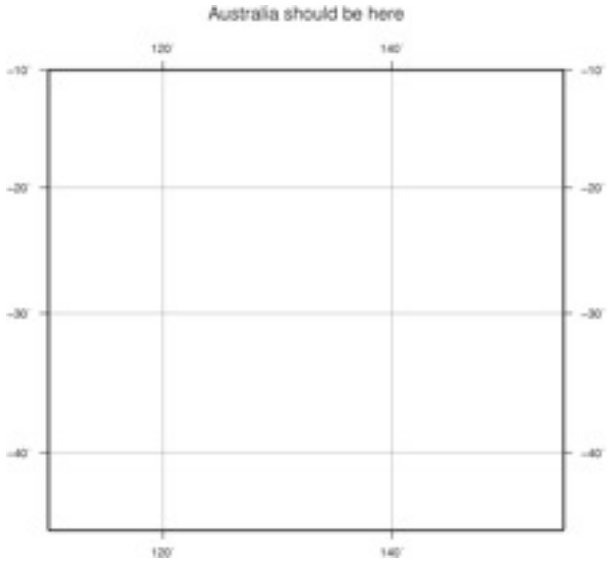



*Figure 6.10: Output of Exercise 6.6*

As you can see, there is no Australia! We will plot a map of Australia in the next section.

## Plotting Vector Data

Vector data can be plotted with GMT using the command psxy. Lines, closed polygons, standard geometric shapes (e.g. circle, square, triangle, star, etc), and custom geometric shapes can all be plotted.

As with psbasemap, the GMT commands psxy and pscoast need to know the projection (-J), area (-R) and map settings (-B). In addition, psxy requires either -W or -S.

### *Plotting Symbols: -S[symbol][size] option*

Point data is plotted using the -S option, where the type of symbol to be used and the size can be specified. Standard geometric shapes are built into GMT, e.g. circle (-Sc), star (-Sa), diamond (-Sd), for full list see table below. It is also possible to create and then plot your own custom symbols using the -Sk option.

| Code | Symbol | Code | Symbol | Code | Symbol |
|---|---|---|---|---|---|
| **–** | x-dash (–) | **g** | octagon | **r** | rectangle |
| **a** | star | **h** | hexagon | **s** | square |
| **b** | bar | **i** | invtriangle | **t** | triangle |
| **c** | circle | **k** | kustom | **v** | vector |
| **d** | diamond | **l** | letter | **w** | wedge |
| **e** | ellipse | **n** | pentagon | **x** | cross (x) |

| | | | | | |
|---|---|---|---|---|---|
| **f** | front | **p** | point | **y** | y-dash (\|) |

*Table 6.3: Summary of symbol codes for use with the -S option.*

The size of the symbol is specified second. By default, the size will be in the units specified in .gmtdefaults4, unless centimeters (c), inches (i), meters (m), or points (p) are specified.

Lower case letters (a, c, d, g, h, i, n, s, t, x) fit inside a circle of the given diameter

Upper case letters (A, C, D, G, H, I, N, S, T, X) have an area equal to a circle of the given diameter

-Sc0.2c - A circle 0.2 centimeters in diameter

-Sa0.5i - A star where the circumscribing circle is 0.5 inches in diameter

-Ss10p - A square with the same area as a circle 10 points in diameter

### *Plotting Vector Data: psxy command*

psxy reads in data points supplied by the user in a text file. The most basic format is:

x y

x - x-value of data point (e.g. longitude)

y - y-value of data point (e.g. latitude)

To create an input file that we can plot with psxy copy or type the following x, and y values into a text file called testpoints.txt. In this case the x-values are seafloor age and the y-values are seafloor roughness.

```
0     9.5857
5     9.48536
10    9.27587
15    8.81607
20    8.63073
25    8.41677
30    8.18178
35    9.35036
40    9.81617
45    9.97597
50    9.69928
55    9.2697
60    8.27442
65    7.96095
70    8.1061
```

75 7.82319
80 7.49408
85 7.1468
90 7.17136
95 6.86081
100 6.64573

***Exercise 6.7:*** Plot point data using psxy -S . Use the following GMT command and options to plot this data.

psxy testpoints.txt -JX12/6 -R0/100/0/12 -Ba10g5:"Seafloor Age (Ma)":/a2g2:"Roughness (mGal)":SW -Sc0.2 > testpoints_1.ps

-JX12/6 - We are plotting non-geographic data (i.e. these are not latitudes and longitudes) so we need to use -JX. 12/6 sets the width=12cm, and height=6cm.

-R0/100/0/12 - Sets the region of the plot from 0 to 100 for the x-axis and 0 to 12 for the y-axis

-Ba10g5/a2g2::WS - For the x-axis, sets the annotation interval to 10 and grid interval to 5. For the y-axis, sets the annotation and grid interval to 2. WS specifies that only the west and south axes of the plot will be plotted and labelled.

-Sc0.2c - This option tells GMT how to treat the data points that are in the file testpoints.txt. 'c' specifies a circle, and 0.2c specifies the size of the circles.

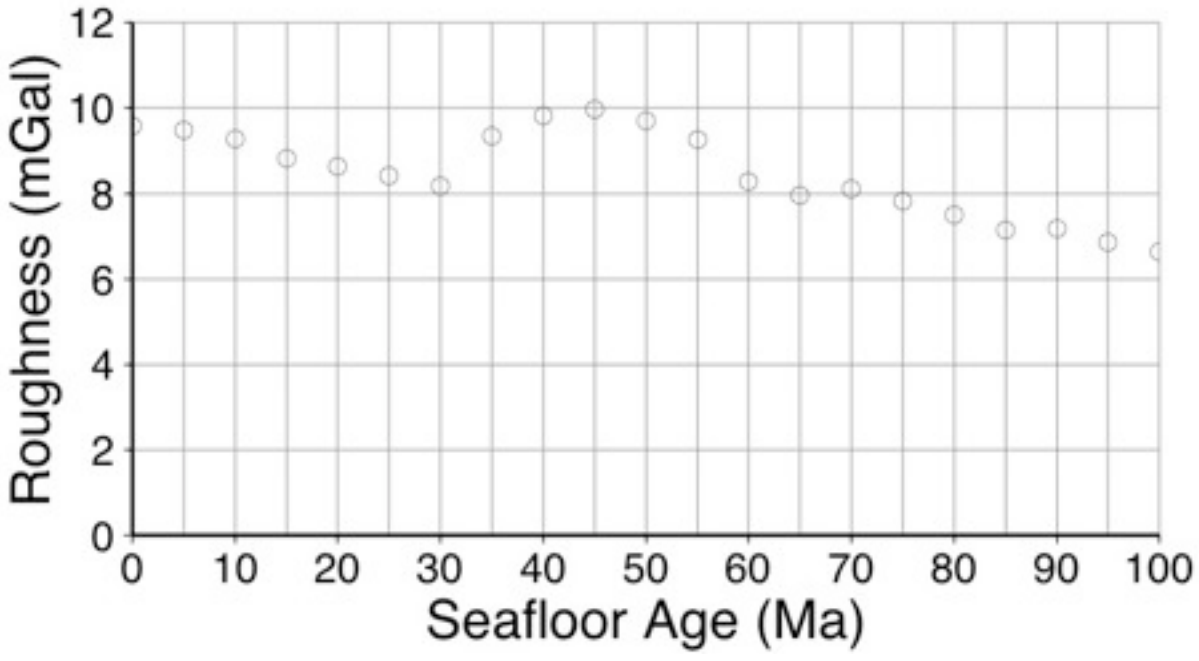



*Figure 6.11: Plot that is the product of Exercise 6.7*

### *Colouring Interiors : -G[fill] option*

The fill colour and texture of symbols and polygons is controlled by the -G option. Colour can be mixed either by light or by paint. Computer monitors mix light in order to make colours, which is the RGB colour scheme (Figure 6.12). Printers on the other hand mix paint to make colours, which is the CMYK colour scheme. This is the reason why sometimes colours are different on the screen to what gets printed.

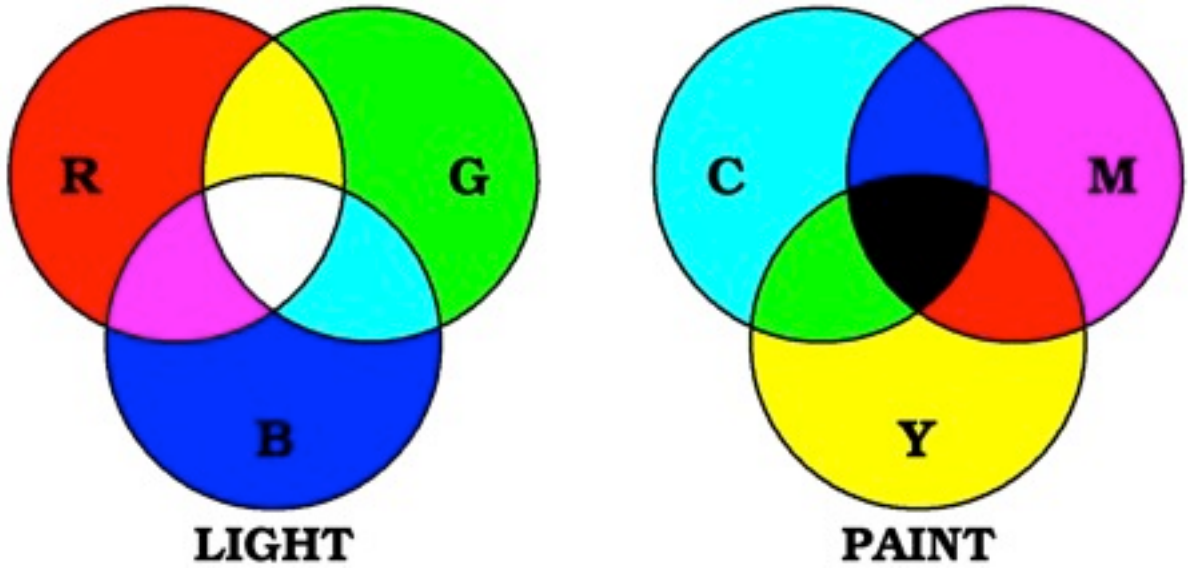



*Figure 6.12: Comparison of RGB and CMYK colour models*

Colour is specified in one of four ways:

- Colour names: Give standard names such as red (-Gred), green (-Ggreen), violet (-Gviolet), etc. (For full list of valid colour names type man gmtcolours).
- RGB (red green blue) system: Give r/g/b where each integer indicates intensity of light from 0 to 255. For example -G255/0/0 specifies red; -G0/255/0 specifies green, -G0/0/255 specifies blue If r = g = b we have gray and only r needs to be specified. For example 125/125/125 could be specified as -G125/125/125 or -G125.
- HSV system: Give h-s-v for hue, saturation, and value.
- CMYK system: Give c/m/y/k values, each in the 0-100% range.

***Exercise 6.8:*** Plotting point data with -S and -G. Plot green circles -Sc0.2c -Ggreen

psxy testpoints.txt -JX12/6 -R0/100/0/12 -Ba10g5:"Seafloor Age (Ma)":/a2g2:"Roughness (mGal)":SW -Sc0.2c -Ggreen > testpoints_green.ps

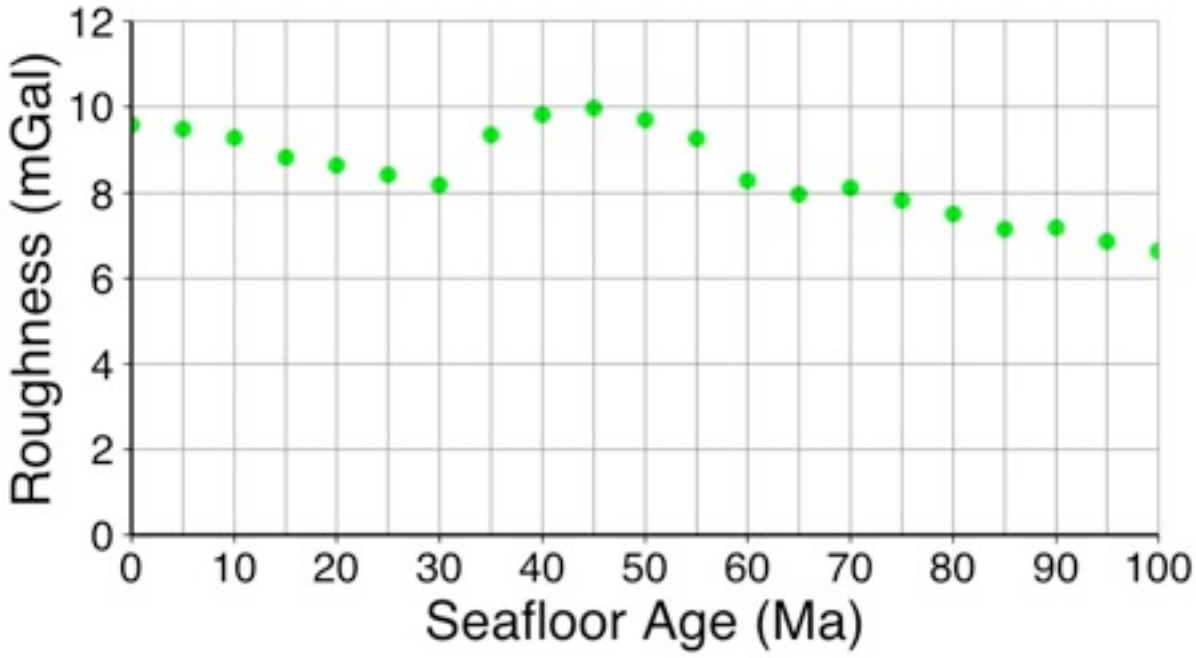



*Figure 6.13: Plot that is the product of Exercise 6.8*

***Exercise 6.9:*** Plot violet stars (-SA0.2c -G255/0/255)

psxy testpoints.txt -JX12/6 -R0/100/0/12 -Ba10g5:"Seafloor Age (Ma)":/a2g2:"Roughness (mGal)":SW -SA0.2c -G255/0/255 > testpoints_violet.ps

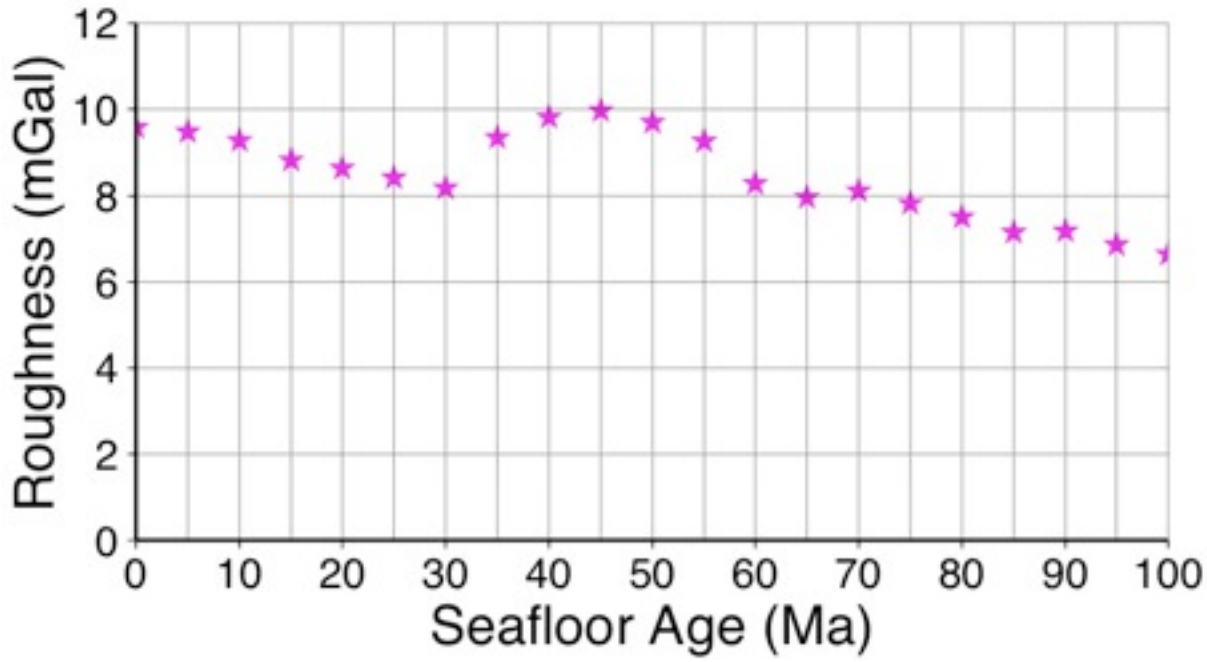



*Figure 6.14: Plot that is the product of Exercise 6.9*

### *Colouring Outlines and Lines: -W[pen] option*

Lines and symbol outlines are controlled with the -Wpen option. The pen is defined by a comma separated list of width, colour and texture, each of which is optional.

-Wwidth,colour,texture

width - is specified as

(1) a measure e.g. points, cm, inches (e.g. -W6, -W6c, -W6i for a pen that is 6 points, centimeters and inches wide, respectively)

(2) faint, thin[ner|nest], thick[er|est], fat[ter|test], or obese (e.g. -Wthickest, -Wfat)

colour - see previous section -Gfill that describes specifying colour

texture - is specified by a combination of dashes '-' and dots '.'

***Exercise 6.10:*** Plot lines using -W. Plot the data points in testpoints.txt as a fat, black line.

psxy testpoints.txt -JX12/6 -R0/100/0/12 -Ba10g5:"Seafloor Age (Ma)":/a2g2:"Roughness (mGal)":SW -Wfat > testpoints_line.ps

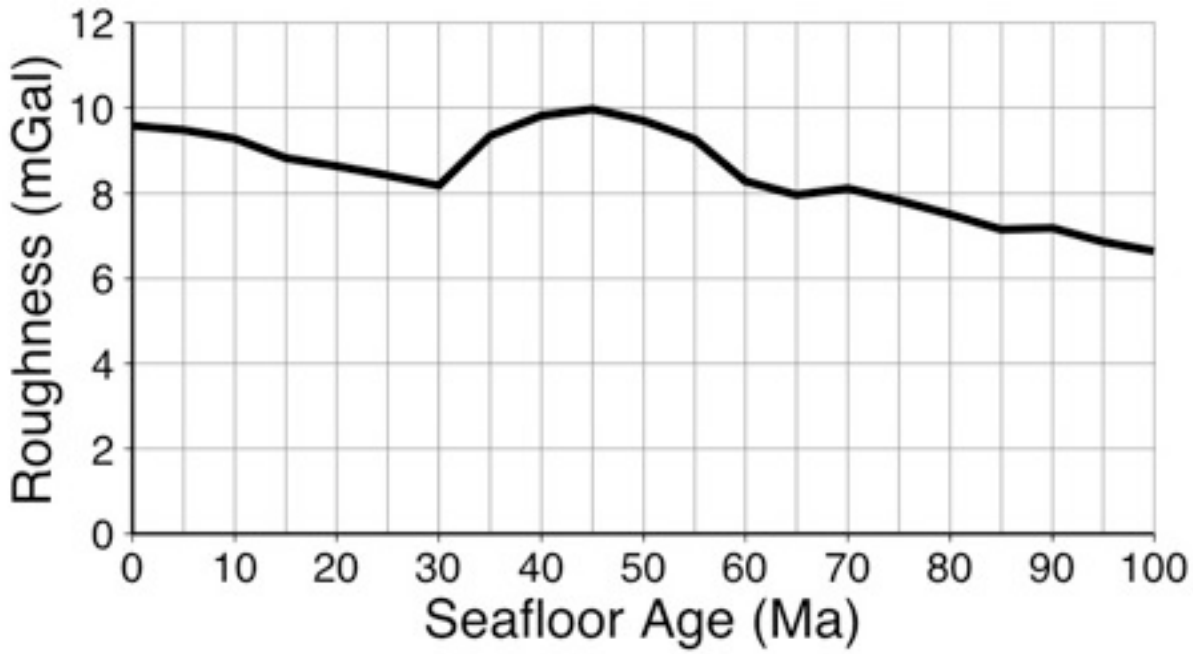



*Figure 6.15: Plot that is the product of Exercise 6.10*

***Exercise 6.11:*** Plot the data points as a line but thinner and red.

psxy testpoints.txt -JX12/6 -R0/100/0/12 -Ba10g5:"Seafloor Age (Ma)":/a2g2:"Roughness (mGal)":SW -Wthinner,red > testpoints_redline.ps

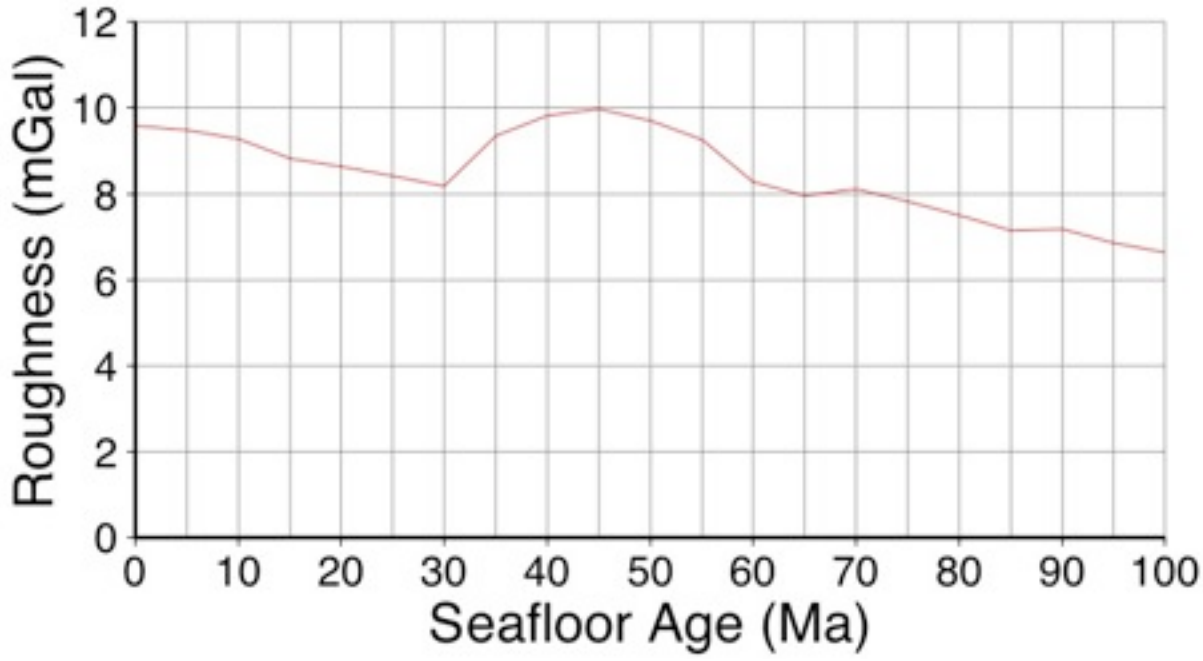



*Figure 6.16: Plot that is the product of Exercise 6.11*

***Esercise 6.12:*** Plot the data points as a thick, orange, dashed line.

psxy testpoints.txt -JX12/6 -R0/100/0/12 -Ba10g5:"Seafloor Age (Ma)":/a2g2:"Roughness (mGal)":SW -Wthickest,255/127/0,-.. > testpoints_orangeline.ps

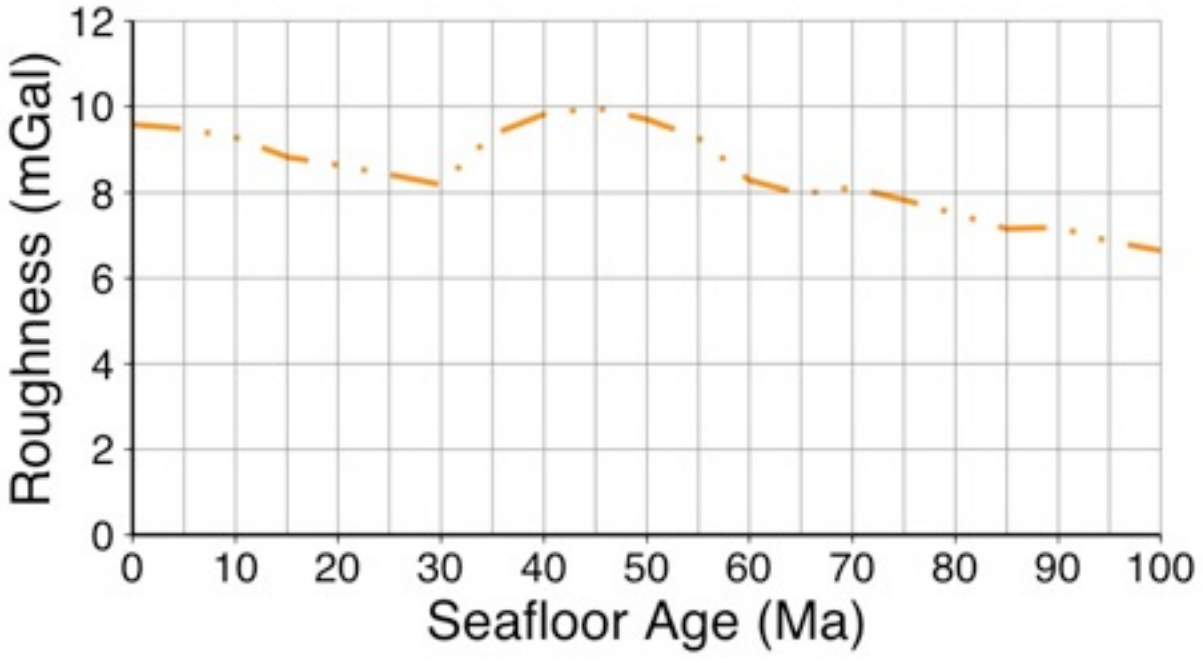



*Figure 6.17: Plot that is the product of Exercise 6.12*

***Exercise 6.13:*** Plot the data as blue squares with a red outline

psxy testpoints.txt -JX12/6 -R0/100/0/12 -Ba10g5:"Seafloor Age (Ma)":/a2g2:"Roughness (mGal)":SW -SS0.5c -Wthicker,255/0/0 -Gblue > testpoints_bluesquares.ps

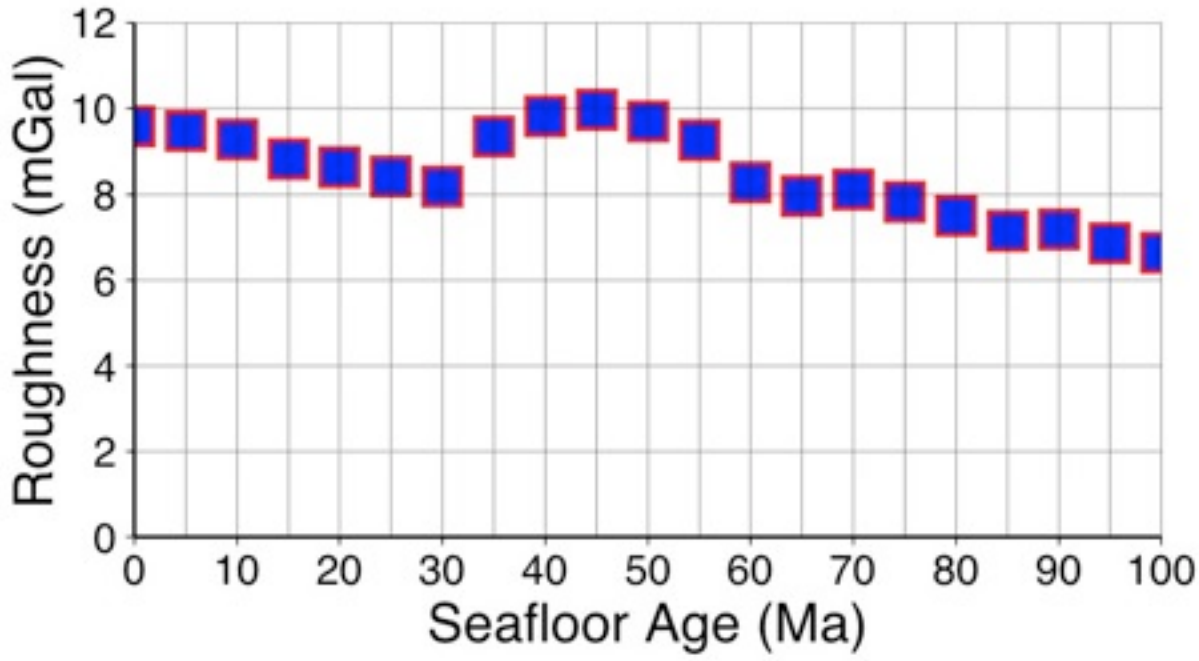



*Figure 6.18: Plot that is the product of Exercise 6.13*

***Plotting Polygons: -L option***

The -L option forces GMT to draw a closed polygon

***Exercise 6.14:*** Plot a closed polygon using -L

psxy testpoints.txt -JX12/6 -R0/100/0/12 -Ba10g5:"Seafloor Age (Ma)":/a2g2:"Roughness (mGal)":SW -L -Wthicker,255/0/0 -Gblue > testpoints_bluepolygon.ps

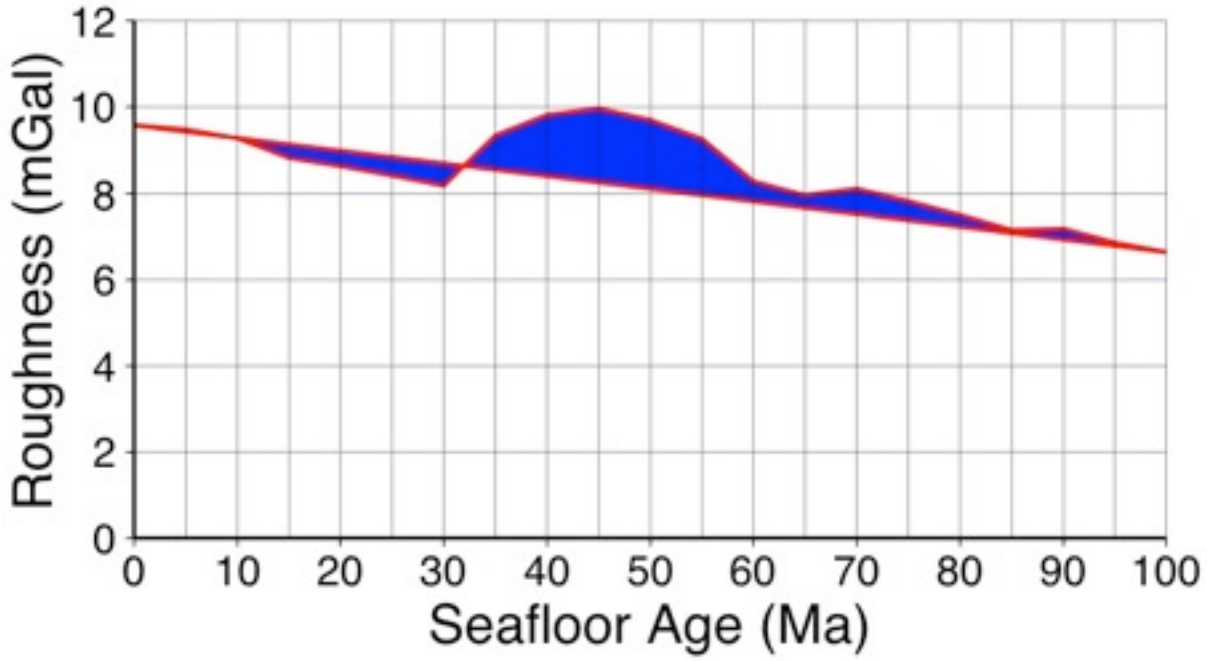



*Figure 6.19: Plot that is the product of Exercise 6.14*

### *Plotting Coastlines: pscoast command*

GMT is installed with pre-defined coastlines files which are stored as vector data. The coastlines come in several resolutions: full(f), high (h), intermediate (i), low (l) and crude (c) (Figure 6.20). The type of resolution you want can be defined as the -D flag followed by the letter corresponding to the resolution. For example, -Df means use full resolution. The type of resolution you use is highly dependent on the scale of your map and the speed with which you want your process to run. You would not use a full resolution coastline file (55MB file) to plot a global map, for example. The default is low resolution.

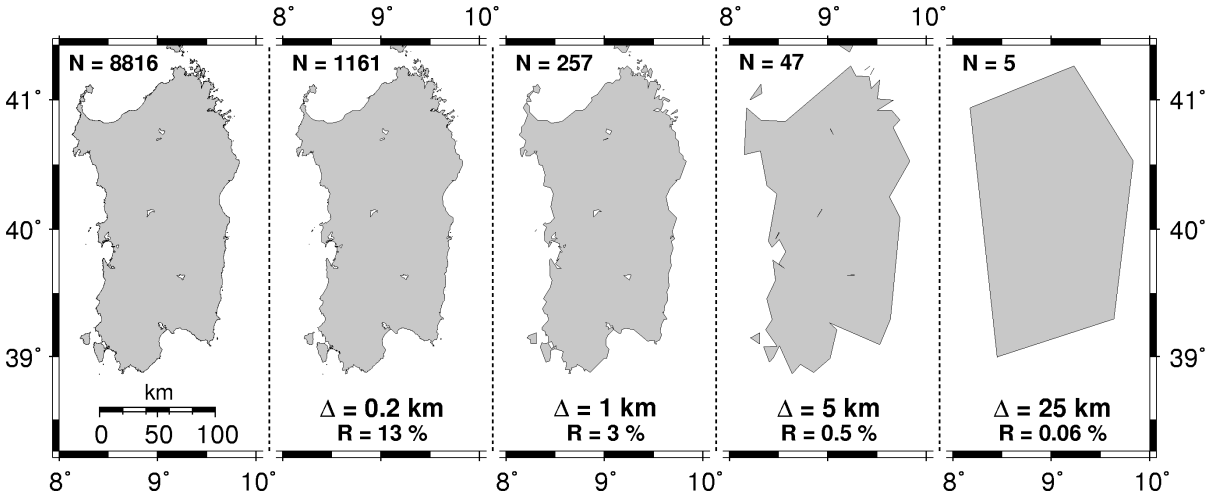



*Figure 6.20: The five coastline resolutions supplied with GMT.*

To control the outline of the continents, you use -W, to control the fill of the continents you use -G. In addition, you have the option to fill "wet" areas or the oceans and lakes by

specifying the -S option (Note: -S means something different here than in psxy) followed by the colour.

***Exercise 6.15:*** Create a map of Australia based on the parameters defined in exercise 6.6 but change the title to be "My First GMT Map of Australia". Use a low resolution coastline. Colour the continents in brown and the oceans in light blue.

pscoast -R110/155/-45/-10 -JM12 -Dl -Ba20f20g20/a10g10f10:."My First GMT Map of Australia": -Gsienna -Slightblue > australia.ps


*Figure 6.21: Output of Exercise 4.15*

The coastline file also gives you the option of plotting rivers and political boundaries and of excluding data from the coastline file based on area (for example to exclude lakes). See the man page for pscoast to get further details about these options.

***Exercise 6.16:*** Create a map of the continental US based on the following parameters: Albers Equal-Area projection with a projection centre of 265/35 and two standard paraellels at 33 and 45 degrees. The region of interest is -R230/300/25/50. Use an intermediate resolution coastline. Colour the continents and oceans in some weird colours. Plot also the national political boundaries and the state boundaries in the US.

pscoast -R230/300/25/50 -JB265/35/33/45/12 -Ba20f5g20/a10f2g10 -Di -N1 -N2 -Ghotpink -Sblack > us_funky_map.ps

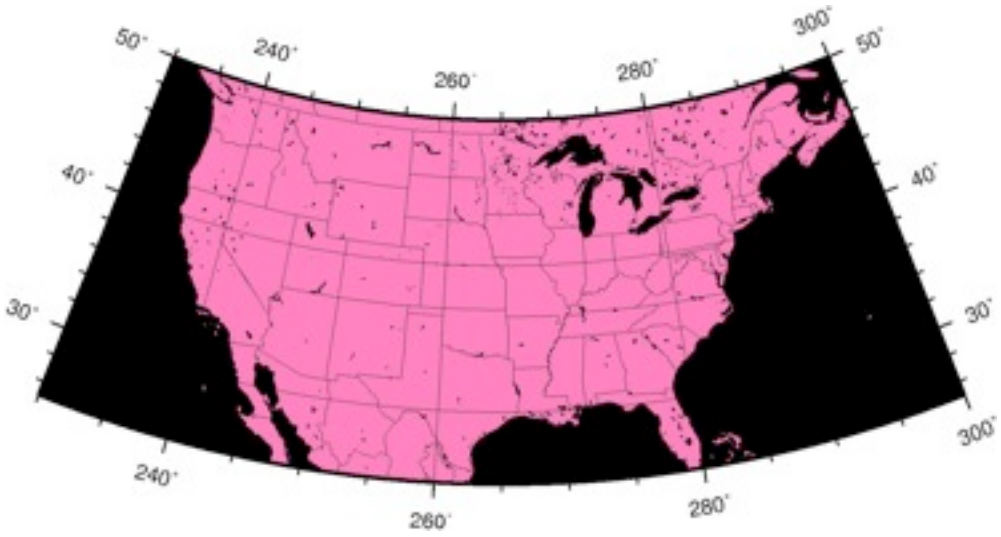



*Figure 6.22: Funky map of the continental US - output of Exercise 6.16*

## Creating Colour Tables

### *Defining colours for gridded data: -C option*

Before we can plot raster data as a gridded data set, we need to define a colour palette or colour table in order to assign colours to each value when plotting. As we mentioned in an earlier section on colour, there are several colour models that one can use to define colour. GMT can handle the following three colour models:

- RGB (r,g,b in 0-255 range)
- HSV (h in 0-360, s,v in 0-1 range)
- CMYK (c,m,y,k in 0-1 range)

The default colour model can be defined in the GMT defaults file under COLOR_MODEL. The system default is rgb. Colour palette files usually have the extension .cpt.

Colour palettes are created in GMT in four ways:

- Using the GMT command **makecpt**
- Using awk
- By manually typing values into a txteditor into the format required (not recommended!)
- Using the GMT command **grd2cpt**

The general format of a colour palette (cpt) file is one or more records of the following:

z0 colour0 z1 colour1 [U|L|B]

where

z0–z1 is the "z" range of this particular slice

colour0 and colour1 specify colour for this range (for constant colour, color0 = color1, otherwise colour will vary linearly from z0 to z1 )

U, L, B indicates we want to annotate the Upper, Lower, or Both ends of the slice (optional)

Colors can be given in greyscale, RGB, HSV, CMYK or name components but they must be separated by space or tab.

***Creating a colour table using makecpt***

To make a simple, linear colour table given a master colour table (several are in-built in GMT) and the desired z-values at colour boundaries, we can use **makecpt**. There are a few key options that are used with makecpt:

-C sets the name of the master cpt file to use. Type **makecpt** in the terminal to get a list of the in-built colour tables).

-I reverses the sense of the colour progression

-Z makes a continuous rather than discrete (default) colour table

-T defines the zmin, zmax and the interval as zmin/zmax/interval

To make a discrete colour palette file for data that ranges from 0 to 1 with colour changes at every 0.1 units, do the following:

```
makecpt -Crainbow -T0/1/0.1 > discrete.cpt
```

The file discrete.cpt contains the following (using the RGB colour model):

```
    cpt file created by: makecpt -Crainbow -T0/1/0.1
#COLOR_MODEL = RGB
0     255   0     242   0.1   255   0     242
0.1   255   0     217   0.2   255   0     217
0.2   255   0     191   0.3   255   0     191
0.3   255   0     166   0.4   255   0     166
0.4   255   0     140   0.5   255   0     140
0.5   255   0     115   0.6   255   0     115
0.6   255   0     89    0.7   255   0     89
0.7   255   0     64    0.8   255   0     64
0.8   255   0     38    0.9   255   0     38
```

```
0.9    255    0      13     1      255    0      13
B      0      0      0
F      255    255    255
N      128    128    128
```

or using HSV colour model:

```
#      cpt file created by: makecpt -Crainbow -T0/1/0.1
#COLOR_MODEL = +HSV
0      285    1      1      0.1    285    1      1
0.1    255    1      1      0.2    255    1      1
0.2    225    1      1      0.3    225    1      1
0.3    195    1      1      0.4    195    1      1
0.4    165    1      1      0.5    165    1      1
0.5    135    1      1      0.6    135    1      1
0.6    105    1      1      0.7    105    1      1
0.7    75     1      1      0.8    75     1      1
0.8    45     1      1      0.9    45     1      1
0.9    15     1      1      1      15     1      1
B      0      0      0
F      0      0      1
N      0      0      0.50196078
```

Notice that the colour values for the z-slices are the same.

To make a continuous colour palette file for data that ranges from -20 to 60 with colour changes at every 10 units, do the following:

```
makecpt -Crainbow -T0/1/0.1 -Z > continuous.cpt
```

The file continuous.cpt contains the following (using the RGB colour model):

```
#      cpt file created by: makecpt -Crainbow -T0/1/0.1 -Z
#COLOR_MODEL = RGB
0      255    0      255    0.1    255    0      229
0.1    255    0      229    0.2    255    0      204
0.2    255    0      204    0.3    255    0      178
0.3    255    0      178    0.4    255    0      153
```

```
0.4   255   0    153   0.5   255   0    127
0.5   255   0    127   0.6   255   0    102
0.6   255   0    102   0.7   255   0    76
0.7   255   0    76    0.8   255   0    51
0.8   255   0    51    0.9   255   0    25
0.9   255   0    25    1     255   0    0
B     0     0    0
F     255   255  255
N     128   128  128
```

Notice that the colour values for the z-slices are different meaning that there is a gradation of colour within z-slices.

***Creating a colour table using grd2cpt***

To create a colour table based on a master cpt file and the histogram-equalised distribution of z-values in a gridded data file, we can use the **grd2cpt** command. This is advantageous if you want to create the best colour distribution for your gridded data set.

There are a few key options that are used with **grd2cpt**:

-C sets the name of the master cpt file to use.

-I reverses the sense of the colour progression

-Z makes a continuous rather than discrete (default) colour table

-D defines the upper and lower limit if you wanted it different to the actual min/max values of the gridfile

To create the simplest cpt file based on your gridded data using grd2cpt, do the following:

```
grd2cpt your_grid_file -Crainbow > new_cpt.cpt
```

## PLOTTING RASTER DATA

Firstly a recap on some points about raster data:

- ☑ They are stored as binary files
- ☑ They are netCDF files, commonly with the extension .grd or .nc
- ☑ The grids are equidistant and the grid spacing is fixed ($\Delta x$, $\Delta y$ are constants)
- ☑ Header section of the grid file contains all information such as the w/e/s/n region, the grid spacing and various text strings describing the data
- ☑ There are two types of grid file registrations (Figure 6.23):

- pixel registration - grid points are located in the centre of grid cells
- gridline registration - grid points are located in the corners of grid cells

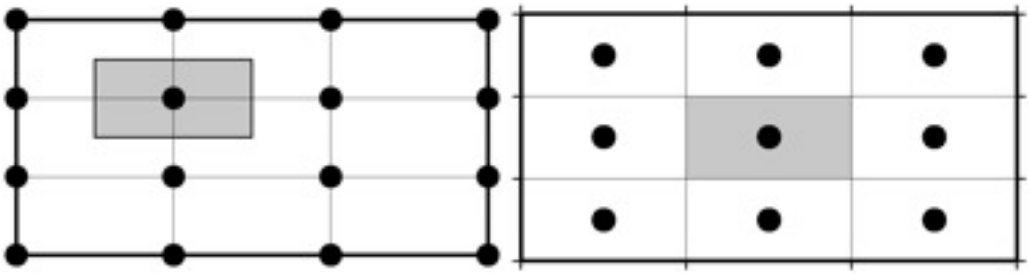


*Figure 6.23: The two different grid file registrations: gridline (left) and pixel (right). The difference between the two is that there is one more grid point (one extra row/column) in gridline registered grids than pixel registered grids.*

Raster data can be plotted in GMT using the commands **grdcontour** and **grdimage**.

### *Querying raster data using grdinfo*

Raster data cannot be opened in a texteditor as you can with vector data stored in ASCII format, for example. Instead, we need to use a program to extract relevant information from the raster data (e.g. the min and max coordinates of the grid, the min, max, mean and other statistical parameters from the grid, the grid spacing, the number of nodes in the x and y direction and the grid registration.

The GMT program that enables the querying of raster data is **grdinfo**. It is really easy to use. Just type:

```
grdinfo your_grid_file
```

where your_grid_file corresponds to the filename of your gridded data set

An example of grdinfo output is shown below:

WDMAM_NGDC_V1.1.grd: Title: z

WDMAM_NGDC_V1.1.grd: Command: grdmath WDMAM.grd mask.grd OR = WDMAM_NGDC_V1.1.grd

WDMAM_NGDC_V1.1.grd: Remark:

WDMAM_NGDC_V1.1.grd: Gridline node registration used

WDMAM_NGDC_V1.1.grd: Grid file format: nf (# 18) GMT netCDF format (float) (COARDS-compliant) [DEFAULT]

WDMAM_NGDC_V1.1.grd: x_min: -180 x_max: 180 x_inc: 0.05 name: Longitude [degrees_east] nx: 7201

WDMAM_NGDC_V1.1.grd: y_min: -90 y_max: 90 y_inc: 0.05 name: Latitude [degrees_north] ny: 3601

WDMAM_NGDC_V1.1.grd: z_min: -3691.49 z_max: 3373.3899 name: z

WDMAM_NGDC_V1.1.grd: scale_factor: 1 add_offset: 0

The output of this is pretty self-explanatory. The important ones are:

☑ x_min, x_max, y_min and y_max define the boundaries of the grid

☑ x_inc and y_inc define the x and y increments or grid spacing

☑ nx and ny define the number of nodes (or number of points) in the x and y directory

☑ z_min and z_max are the min and max values of the data

To extract statistical information from the grid, use the -L1 and -L2 options.

### *Plotting contours using grdcontour*

**grdcontour** is a GMT command that traces each contour through a grid and plots the data. It can also dump the resultant vector data (of contours which are line features) into a separate file.

**grdcontour** requires –J and optionally –R (default region is assumed to be the grid region) as well as an input grid file. Several options determine how the contouring will take place:

☑ contour interval (-C option)

☑ annotation interval (-A option)

☑ contour limits (-L option)

☑ pen for the contours (-W option)

There are several other options but you can look them up in the man page.

The -C option sets the contour interval or the colour palette file. We will not be describing how to use the colour palette file for contouring in these course notes. To work out an appropriate contour interval, you would firstly want to find out what the min and max z values of your grid is. In the example above, the min/max values are -3691.49/3373.3899 so as a first pass, we can select a contour interval of 250 nT.

If we wanted to annotate those contours, we will likely want not to be at every contour interval. To change this, use the -A option. We might choose to annotate every 500 nT.

***Exercise 6.17:*** Create a contour map of the World Digital Magnetic Anomaly Map (WDMAM) for a small region in the Pacific with contours every 50 nT, annotations every 100 nT and the contours drawn in blue.

grdcontour -A100 -C50 WDMAM_NGDC_V1.1_mask.grd -W1/blue -R180/200/40/50 -JM14 -B10 > WDMAM_contour_map.ps

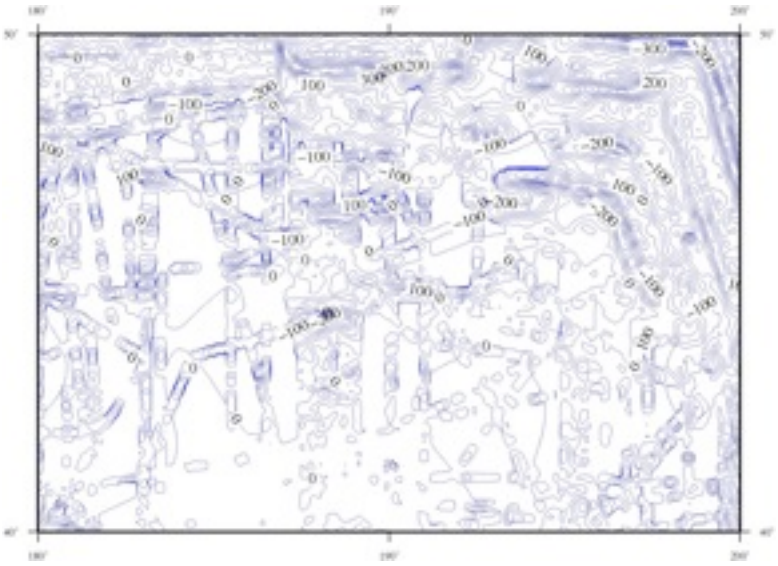


*Figure 6.24: A magnetic grid contour map of the North Pacific. The output from Exercise 6.17.*

### *Plotting gridded data using grdimage*

Once a cpt file has been made it is relatively straight-forward to generate a colour image of gridded data. Colour images are made with **grdimage** which takes the usual common command options (such as -J, -R and -B) and a cpt file (using -C with your colour palette).

**grdimage** works by reading in a 2-D grid file and producing a grey-shaded or coloured map by plotting rectangles centered on each grid node and assigning them a grey-shade or color based on the z-value. Inherently, interpolation occurs between each grid cell. By default this is done by bi-cubic interpolation which produces a smoooth image with few artifacts.

In addition to -J, -R, -B and -C, the following options can be used with **grdimage**:

-S controls the interpolation and aliasing by allowing you to change the interpolation mode. Options include B-spline smoothing (b), bicubic (c), bilinear (l) and near neighbour (n). Bi cubic is the default.

-E sets the resolution of the image in dpi

-I gives the name of a grid file with intensities in the (-1,+1) range

To create an image of the present day agegrid of the world, do the following (see Figure 6.25):

```
grdimage agegrid.grd -Cage.cpt -Rg -JW200/20 -Ba30f10g30 > agegrid.ps
```

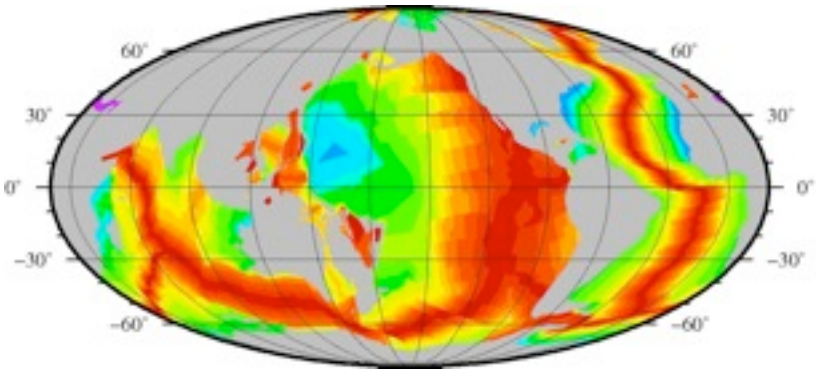



*Figure 6.25: Present day grid of the age of the oceanic lithosphere. Red denotes young crust, blue is old crust. Grey areas mask out continental crust.*

The plotting of gridded data often involves using an intensity or shading gradient (-I option). This is because flat images usually fail to show any details about the fabric. This gradient grid is created using **grdgradient**

grdgradient is based on the concept of artificial illumination. Artificial illumination simulates light from a source placed at infinity at a given azimuth (degrees from north) and elevation (in degrees). It is often used to shade maps in different ways so that you can highlight the trends and textures of features.

When slopes are facing the light source, these slopes should lighten while slopes that face away from the light source should darken (Figure 6.26). In Figure 6.26, we have a light source placed to the upper right of the image. The azimuth of the illumination is s which is parallel to the rays of the light source. The n1 values represent the normal to the topographic surface. It is the angle between s and n1 that will determine the shading (small angle, less shading - larger angle, more shading). In this case on the right, the slope will be exposed to much of the light and will lighten whereas the example of the left is facing away from the light and so will darken.

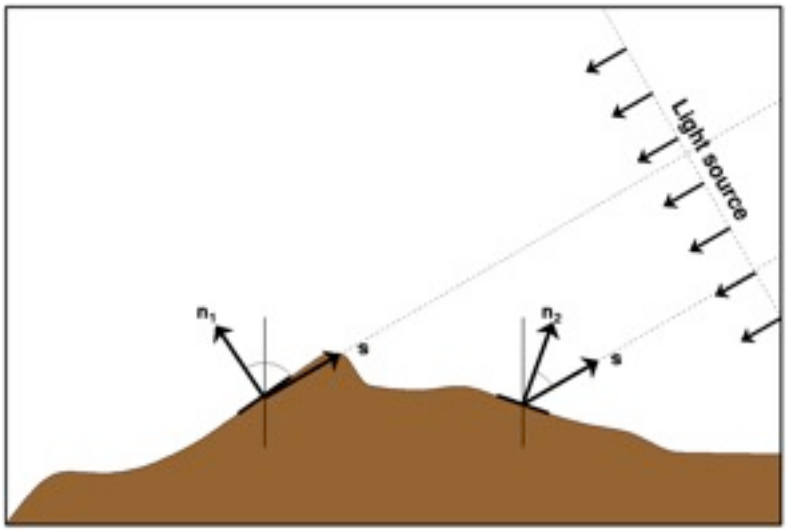



*Figure 6.26: An example of artificial illumination*

In the case of GMT, we use artificial illumination on gridded data to highlight features. In GMT, shadows that are cast by the topography are not used. For example, we do not account for the shadow that the Mount Everest itself casts on the surrounding area. We only use the shadows generated by the light source. Since angles of normal vectors are only meaningful for topography, we generalise by using the data gradient dz/dn (slope) instead, where n is the direction to the light source. The resulting gradients are normalised to the -1 to +1 range and then transformed to give smoothly varying intensities.

The common options used with grdgradient include:

-A defines the azimuth (the azimuth to the light source)

-N defines the normalisation settings. The options are no normalisation (default), cumulative Laplace distribution (e) or cumulative Caucy distribution (t).

-G defines the output gradient file

To create a gradient grid for the agegrid plotted above, we do the following:

```
grdgradient agegrid.grd -Ggrad.grd -A120 -Ne0.8
```

To now plot the agegrid as in Figure 6.25 but with the gradient grid, do the following:

```
grdimage agegrid.grd -Cage.cpt -Rg -JW200/20 -Ba30f10g30 -Igrad.grd >
agegrid_grad.ps
```

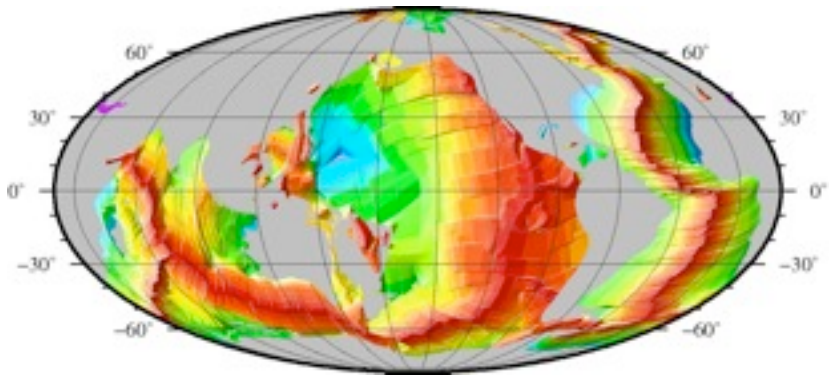



*Figure 6.27: Present day grid of the age of the oceanic lithosphere with a gradient/shading grid. Red denotes young crust, blue is old crust. Grey areas mask out continental crust.*

## Building a GMT Script

The combination of GMT and shell-scripting allows for very powerful programming and automation of repetitive tasks.

A shell script is a text file containing commands that can be executed by a UNIX shell. The text file contains one or more commands, information about their input and output, and the order in which they should be processed. Commands can be from a mixture of

sources e.g. UNIX (ls, cd, mkdir etc), GMT (pscoast, psxy, etc) and many others (e.g. awk, sed etc).

### *Starting a Shell Script*

You can start writing a script using any sort of text editor that does not embed control characters. e.g. Notepad++ (Windows), Textedit (Mac OSX).

The first line of a script always calls the executing shell. The Syntax is:

#! /bin/bash

This would call the Bash shell to execute the commands in the script. Other possibilities are Korn (#!/bin/ksh), C-Shell (#!/bin/ csh). Note, any other line that starts with the # character is a comment and will not be recognized by the shell.

Save the script to a filename, e.g. myfirstscript.sh. It is convention to give a shell script the file extension .sh so it is easily recognised as a shell script.

### *Building a Shell Script*

Once your shell script exists it is then possible to add any number of commands to it.

If you wanted to add comments to your script, you would start lines with the # symbol. This tells the program to skip these lines as they are not to be processed. An example of a comment line would look like

# This line is a comment and is ignored by the program

### *Make a Shell Script Executable*

In order to run the script, we need to make it executable by typing
chmod +x myfirstscript.sh

### *Run a Shell Script*

To run the shell script you have created, simply type the name of the script preceded by ./

./myfirstscript.sh

***Exercise 6.18:*** Make a basic shell script. Open a blank script with your favourite text editor. Copy/type the following commands and comments.

```
#!/bin/bash
# This line is a comment
# Make a map of the world between the latitudes of 70°N and 70°S in Mercator projection, with the coastlines shown in blue.
pscoast -JM14 -R0/360/-70/70 -Wblue -B60/30 -Dc > BlueMercatorWorld.ps
```

Save the file as myfirstscript.sh

Make the file executable, by typing in the terminal window [**chmod +x myfirstscript.sh**]

Run the script, by typing in the terminal window [**./myfirstscript.sh**]

### *Assigning Variables in a Shell Script*

Variables can be defined with a = sign and referenced with the $ character. Some examples

width=14  defines the variable *width* to 14

region=0/360/-70/70  defines the variable *region* to 0/360/-90/90

Note, spaces are not permitted on either side of the = sign.

Once assigned a variable can be used later on in the script by typing the variable name with a $ sign in front, e.g.

$width

$region

***Exercise 6.19:*** Use Variables in a Basic Shell Script. Make a new shell script with the text below. Save it, make it executable and run it

```
#!/bin/bash

# Make a map of the world between the latitudes of 70°N and 70°S in Mercator projection, with the coastlines shown in blue.

#Assign Variable
width=14
region=0/360/-70/70

#Make GMT map
pscoast -JM$width -R$region -Wblue -B60/30 -Dc > BlueMercatorWorld.ps
```

Now change the region, and the width of the map.

Assign another variable and use it in the script. For example you could assign colour, or the psfilename (e.g. colour=blue or psfile=BlueMercatorWorld.ps)

## POSTSCRIPT

All the images you have created so far have been Postscript files, denoted with the extension ".ps". PostScript is a programming language for describing how a page is to be

printed or displayed. PostScript files are plain text files that contain PostScript code. All plotting programs use the PostScript page description language to define plots. Usually they have a ".ps" or a ".eps" termination. They are device-independent and can be viewed using different UNIX tools such as imagetool, ghostscript (gs) or ghostview (gv).

PostScript files can be opened with any text editor and can be edited, assuming you have some knowledge of the syntax. An example of the top (header) and bottom (trailer) of a PostScript file is shown below.

```
%!PS-Adobe-3.0 EPSF-3.0
%%BoundingBox: 0 0 340 340
%%Title: GMT v3.0 Document from pscoast
%%Creator: Username Gaina
%%DocumentNeededResources: font Times-Roman
%%CreationDate: Mon Feb 15 12:30:19 1999
%%Orientation: Portrait
%%EndComments

%%BeginProlog
……
% End of basemap
S 0 A
%%Trailer
% Reset translations and scale and call showpage
S -353 -353 T 4.97417 4.97417 scale showpage

end
```

### *Building a PostScript file in GMT: -K and -O options*

In GMT, plotting is done through the PostScript programming language. The huge advantage of using PostScript is that multiple plot files can be layered to create one image. For example, you can use grdimage to plot a raster, and then use pscoast to plot coastlines over the top. To do this PostScript needs to know something about the order of the layers. In GMT, specifying the -K and -O options tells PostScript how to treat each plot layer.

If your script only has one GMT command in it (as all our examples have so far) then PostScript does not need to know anything about layering, so no -K or -O options are required.

When you want to combine more than one GMT plot layer, -K and -O must be used.

The -K option tells PostScript that another layer will be appended later. Using the -K option without the -O option MUST be done for the first line of your because it will create the correct header record of your PostScript file (see example of PostScript code above). The last command in your script should not have the -K option.

The -O option tells PostScript to overlay the current plot onto the preceding layers. This must be specified for all layers except the first one.

The -K and -O options used in combination means that no header or trailer will be created but rather it is telling the program that more code will be appended and overlayed.

In summary, your commands should have the following pattern of -K and -O.

```
First_command          -K        > $psfile
Inbetween_commands     -K  -O    >> $psfile
Final_command          -O        >> $psfile
```

Note that the first line outputs using a >, which means overwrite any existing file.

Subsequent lines use a >>, which means append this information to the existing file.

NOTE: Some of the most common beginner errors involve incorrect use of the -K and -O options and > and >>. If your script isn't working, check these things are all correct FIRST.

### *Using Multiple GMT Commands*

***Exercise 6.20:*** Use a shell script to plot multiple GMT commands, using the -K and -O options. Plot topography/bathymetry, with coastlines and hotspot locations overlaid.

```
#!/bin/bash

# Bathymetric map of the Indian Ocean with coastlines and hotspot locations

#Assign Variable
width=14
region=60/110/-60/10
colour=blue
psfile=IndianOceanHotspots.ps

#Make GMT map
grdimage etopo2.grd -JM$width -R$region -B60/30  -Ctopo.cpt      -K       > $psfile
pscoast             -JM$width -R$region -Wblue  -Dc              -K  -O >> $psfile
psxy hotspots.gmt   -JM$width -R$region -Sa0.4c -Gred -Wthin,black -O >> $psfile
```

***Exercise 6.21:*** Multi-Line Shell Script. Plot topography/bathymetry, with coastlines and the locations of the Emperor-Hawaii Seamounts colour coded by age overlaid, with a scalebar.

```
#!/bin/ksh
# Project: Global Volcanoes Exercise
# Date:
# Author:   Jo Whittaker
```

```
psfile=globalvolcanoes.ps
cpt=volcanoes.cpt
region=140/300/-60/60
width=15
makecpt -Crainbow -T0/70/10 > $cpt

grdimage etopo2.grd -JM$width -R$region -B60/30  -Ctopo.cpt       -K         > $psfile

pscoast -R$region -JM$width -Ba60g30/a30g30:."Emperor-Hawaii Seamounts":neSW -K -O
-V -Dc -Glightbrown -Slightblue -P -Wwhite >> $psfile

psxy seamount_pac.d -R -J -O -K -V -Sa0.4 -Wthin  -C$cpt >> $psfile

psscale -C$cpt -O -K -V -D7/-1/12/0.5h >> $psfile
```

# 7. Data Processing and Interpolation

## Modifying Vector Data

***Using common UNIX commands***

***Using basic awk programming***

## How to use AWK

Awk is an powerful command language that allows the user to manipulate files containing columns of data and strings. Awk is extremely useful, both for general operation of Unix commands, and for data processing and map-making with GMT. There are two ways to run awk. A simple awk command can be run from a single command line. More complex awk scripts should be written to a command file. Awk takes each line of input and tries to match the 'pattern' (see below), and if it succeeds it will do whatever you tell it to do within the {} (called the action). Awk works best on files that have columns of numbers or strings that are separated by whitespace (tabs or spaces).

Awk refers to the first column as $1, the second column as $2, etc., and the whole line as $0. If you have a file (such as a catalog) that always has numbers in specific columns, you may also want to run the command 'colrm' and combine it with awk. There is a manual page on colrm.

First, suppose you have a file called 'file1' that has 2 columns of numbers, and you want to make a new file called 'file2' that has columns 1 and 2 as before, but also adds a third column which is the ratio of the numbers in columns 1 and 2. Suppose you want the new 3-column file (file2) to contain only those lines with column 1 smaller than column 2. Either of the following two commands does what you want:

awk '$1 < $2 {print $0, $1/$2}' file1 > file2

-- or --

cat file1 | awk '$1 < $2 {print $0, $1/$2}' > file2

Let's look at the second one. You all know that 'cat file1' prints the contents of file1 to your screen. The | (called a pipe) directs the output of 'cat file1', which normally goes to your screen, to the command awk. Awk considers the input from 'cat file1' one line at a time, and tries to match the 'pattern'. The pattern is whatever is between the first ' and the {, in this case the pattern is $1 < $2. If the pattern is false, awk goes on to the next line. If the pattern is true, awk does whatever is in the {}. In this case we have asked awk to check if the first column is less than the second. If there is no pattern, awk assumes the pattern is true, and goes onto the action contained in the {}.

What is the action? Almost always it is a print statement of some sort. In this case we want awk to print the entire line, i.e. $0, and then print the ratio of columns 1 and 2, i.e. $1/$2. We close the action with a }, and close the awk command with a '. Finally, to store the final 3- column output into file2 (otherwise it prints to the screen), we add a '> file2'.

As a second example, suppose you have several thousand files you want to move into a new directory and rename by appending a .dat to the filenames. You could do this one by one (several hours), or use vi to make a decent command file to do it (several minutes), or use awk (several seconds). Suppose the files are named junk* (* is wildcard for any sequence of characters), and need to be moved to ../gmt and have a '.dat' appended to the name. To do this

type

ls junk* | awk '{print "mv "$0" ../gmt/"$0".dat"}' | csh

ls junk* lists the filenames, and this output is piped into awk instead of going to your screen.

There is no pattern (nothing between the ' and the {), so awk proceeds to print something for each line. For example, if the first two lines from 'ls junk*' produced junk1 and junk2,

respectively, then awk would print:

mv junk1 ../gmt/junk1.dat mv junk2 ../gmt/junk2.dat

More complex awk scripts need to be run from a file. The syntax for such cases is:

cat file1 | awk -f a.awk > file2

where file1 is the input file, file2 is the output file, and a.awk is a file containing awk commands. Examples below that contain more than one line of awk need to be run from files. Some useful awk variables defined for you are NF (number of columns), NR (the current line that awk is working on), END (true if awk reaches the EOF), BEGIN (true before awk reads anything), and length (number of characters in a line or a string). There is also looping capability, a search (/) command, a substring command (extremely useful), and formatted printing available. There are logical variables || (or) and && (and) that can be used in 'pattern'.

You can define and manipulate your own user defined variables. Examples are outlined below. The only bug I know of is that Sun's version of awk won't do trig functions, though it does do logs. There is something called gawk, which does a few more things, than awk, but they are basically the same. Note the use of the 'yes' command below. Coupled with 'head' and 'awk' you save an hour of typing if you have a lot of GMT files to process or rename.

EXAMPLES # is the comment character for awk. 'field' means 'column'

```
# Print first two fields in opposite order:
awk '{ print $2, $1 }' file
# Print lines longer than 72 characters:
awk 'length > 72' file
# This program prints every line that has at least one field. This is an easy way to delete
# blank lines from a file (or rather, to create a new file similar to the old file but from
# which the blank lines have been deleted).
awk 'NF > 0' file
# Print length of string in 2nd column
awk '{print length($2)}' file
# This programs counts lines in a file.
awk 'END { print NR }'
# This program prints 7 random numbers from 0 to 100, inclusive.
awk 'BEGIN { for (i = 1; i <= 7; i++)
print int(101 * rand()) }' file
# Add up first column, print sum and average:
{ s += $1 }
END { print "sum is", s, " average is", s/NR }
# Print fields in reverse order:
awk '{ for (i = NF; i > 0; --i) print $i }' file
# Print the last line
{line = $0}
END {print line}
# Print the total number of lines that contain the word Pat
/Pat/ {nlines = nlines + 1}
END {print nlines}
# Print all lines between start/stop pairs:
```

```
awk '/start/, /stop/' file
# Print all lines whose first field is different from previous one:
awk '$1 != prev { print; prev = $1 }' file
# Print column 3 if column 1 > column 2:
awk '$1 > $2 {print $3}' file
# Print line if column 3 > column 2:
awk '$3 > $2' file
# Count number of lines where col 3 > col 1
awk '$3 > $1 {print i + "1"; i++}' file
# Print sequence number and then column 1 of file:
awk '{print NR, $1}' file
# Print every line after erasing the 2nd field
awk '{$2 = ""; print}' file
# Print hi 28 times
yes | head -28 | awk '{ print "hi" }'
# Print hi.0010 to hi.0099 (NOTE GMT USERS!)
yes | head -90 | awk '{printf("hi00%2.0f \n", NR+9)}'
# Replace every field by its absolute value
{ for (i = 1; i <= NF; i=i+1) if ($i < 0) $i = -$i print}
# Some looping for printouts
BEGIN{
for (i=875;i>833;i--){
printf "lprm -Plw %d\n", i
} exit
}
```

Formatted printouts are of the form printf( "format\n", value1, value2, ... valueN)

e.g. printf("howdy %-8s What it is bro. %.2f\n", $1, $2*$3)

%s = string

%-8s = 8 character string left justified

%.2f = number with 2 places after .

%6.2f = field 6 chars with 2 chars after .

\n is newline

\t is a tab

# Print frequency histogram of column of numbers

$2 <= 0.1 {n=n+1}

($2 > 0.1) && ($2 <= 0.2) {nb = nb+1}

($2 > 0.2) && ($2 <= 0.3) {nc = nc+1}

($2 > 0.3) && ($2 <= 0.4) {nd = nd+1}

($2 > 0.4) && ($2 <= 0.5) {ne = ne+1}

($2 > 0.5) && ($2 <= 0.6) {nf = nf+1}

($2 > 0.6) && ($2 <= 0.7) {ng = ng+1}

($2 > 0.7) && ($2 <= 0.8) {nh = nh+1}

($2 > 0.8) && ($2 <= 0.9) {ni = ni+1}

($2 > 0.9) {nj = nj+1}

END {print na, nb, nc, nd, ne, nf, ng, nh, ni, nj, NR}

# Find maximum and minimum values present in column 1

NR == 1 {m=$1 ; p=$1}

$1 >= m {m = $1}

$1 <= p {p = $1}

END { print "Max = " m, " Min = " p }

# Example of defining variables, multiple commands on one line

NR == 1 {prev=$4; preva = $1; prevb = $2; n=0; sum=0}

$4 != prev {print preva, prevb, prev, sum/n; n=0; sum=0; prev = $4; preva = $1; prevb = $2}

$4 == prev {n++; sum=sum+$5/$6}

END {print preva, prevb, prev, sum/n}

#Example of using substrings

{print "imarith", substr($2,1,7) " - " $3, "out."substr($2,5,3)}

{print "imarith", substr($2,9,7) " - " $3, "out."substr($2,13,3)}

{print "imarith", substr($2,17,7) " - " $3, "out."substr($2,21,3)}

{print "imarith", substr($2,25,7) " - " $3, "out."substr($2,29,3)}

## Processing Vector Data

***filter1d***

***filter2d***

***trend1d***

## Modifying Raster Data

In GMT grids can be manipulated using different commands:

**grdcut**: for extracting a subregion from a larger grid

**grdpaste**: paste together 2 *.grd files on a common edge

**grdmath:** perform mathemetical operations (add, substract, multiply, divide) with *grd files

**grdsample**: resample a *.grd file

**grdproject**: forward and inverse map transformation.

To obtain information about a particular *.grd file, use **grdinfo** input *.grd file.

In order to perform more complex operations with grdfiles, see **grdfft**, **grdfilter**, **grdmask**.

A *.grd file can also be transformed into a *.xyz ASCII file (**grd2xyz**) or you can obtain/ plot

contours of equal value from your *.grdfiles by using **grdcontour**.

***Querying raster data: grdinfo***

***Resizing or merging gridding data: grdcut, grdpaste and grdblend***

***Resampling gridded data: grdsample***

***Performing mathematical operations on gridded data: grdmath***

## Processing Raster Data

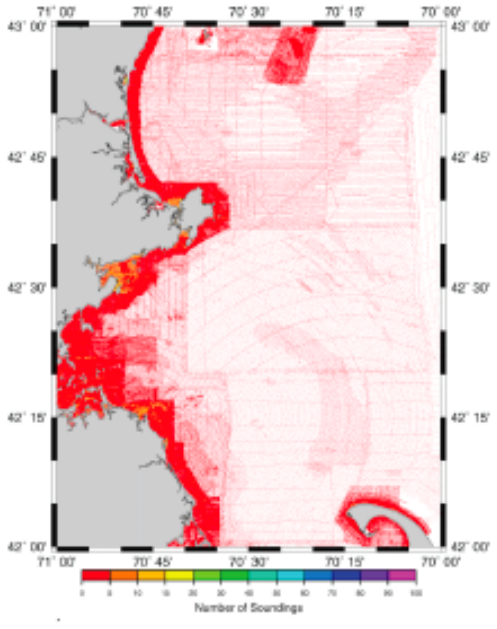
***grdfilter***

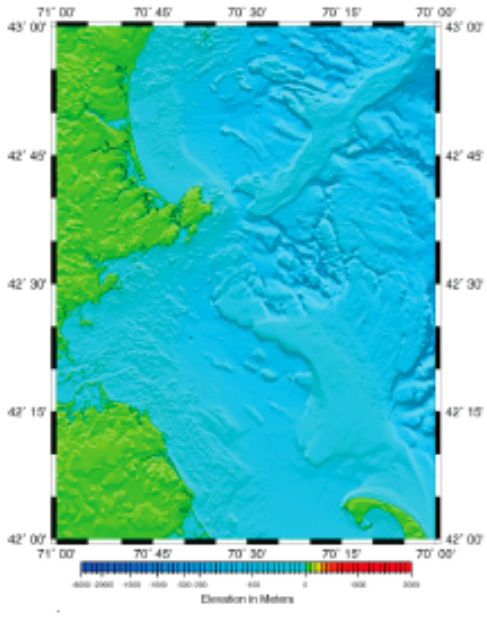
***grdfft***

## Data Interpolation

### *Data Gridding*

The 2 figures below show the eastern coast of the USA where bathymetry has been mapped using echosounders. The resulting data was subsequently transformed into a dense grid. We will apply a similar technique (gridding=data interpolation) to the offshore Antarctica magnetic previously plotted. Below you will find a script that extracts the magnetic data from the GMT database for the same area offshore Antarctica, but now the magnetic data are interpolated to construct a 2D grid.




After setting the parameters, selecting the data in a given area (with gmtlegs), processing the magnetic data (filtering and detrending) the script uses **blockmedian** and **surface** to grid the magnetic data. **Blockmean** or **blockmedian** should always be run prior to running surface (they preprocess data to avoid aliasing). **Surface** is used to generate a high-order

interpolation of z(x.y) onto a grid. A simpler method for gridding is **triangulate** (Delaunay triangulation) which forms z(x,y) as a union of planar triangular facets (type man **blockmean** and man **surface** to find out the complete description of the 2 gmt commands). The resulting grid is then plotted with **grdimage. Grdgradient** is then used for creating a shading file. In this script we will create three different maps (magnetic grid image, magnetic tracks and coastline), and then we will superimpose them using the UNIX command

***xyz2grd***

***surface***

***nearneighbor***

# 8. Further Reading

It is recommended that you download the GMT Technical Reference and Cookbook from the GMT website:

http://gmt.soest.hawaii.edu/

You should also complete the online GMT Tutorial which you can access from the same GMT website: http://gmt.soest.hawaii.edu/

# 9. References

Wessel, P. and W.H.F.Smith, New version of the Generic Mapping Tools released, EOS Trans. Am. Geophys. Union, 76, 329, 1995.
Wessel, P. and W.H.F.Smith, Free software helps map and disply data, EOS Trans. Am. Geophys. Union, 72, 441, 445-446, 1991.
T. Becker, A. Braun, iGMT: Interactive Mapping of Geoscientific Datasets. User Manual for version0.5, 1998.
D. Wright, R. Wood, and B. Sylvander, ARCGMT: A suite of tools for conversion between Arc/Info and Generic Mapping Tools (GMT), Computers& Geosciences, 8, 737-744, 1998.
Aho, A., Kernighan, B., and Weinberger, P., The AWK Programming Language, Addison-Wesley Publishing Company, 1988.
Maling, D.H., Coordinate Systems and Map Projections, 2nd Ed. Pergamon Press. Oxford, 1992

Helpful Web sites:
Map projections: http://everest.hunter.cuny.edu/mp/index.html

ARCGMT: http://dusk.geo.orst.edu/arcgmt