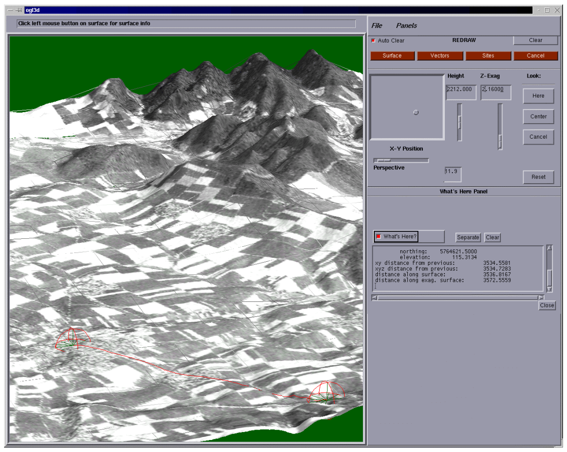
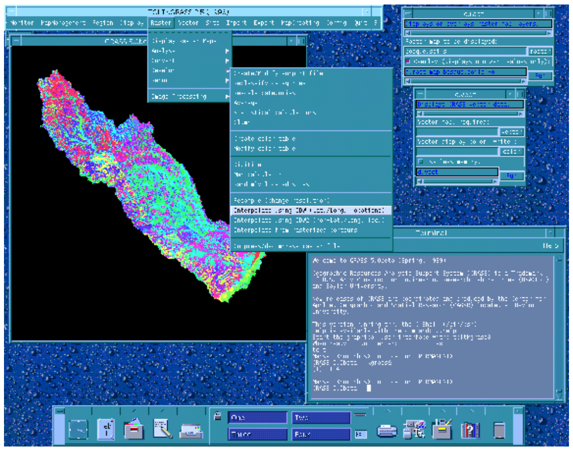
GRASS Reference Manual

Miscellaneous Commands





GRASS Development Team

USA Headquarters
Center for Applied Geographic & Spatial Research
Baylor University
P.O. Box 97351
Waco, Texas 76798-7351
USA

European Headquarters
Institute of Physical Geography-Landscape Ecology
University of Hannover
Schneiderberg 50
30167 Hannover
Germany

grass@baylor.edu

http://www.baylor.edu/~grass
http://www.geog.uni-hannover.de/grass/

Table of Contents

GRASS Introduction

GRASS (Geographic Resources Analysis Support System) is a raster based GIS, vector GIS, image processing system, and graphics production system. GRASS contains over 200 programs and tools to render maps and images on monitor and paper; manipulate raster, vector, and sites data; process multi-spectral image data; and create, manage, and store spatial data. GRASS uses both an intuitive windows interface as well as command line syntax for ease of operations. GRASS can interface with commercial printers, plotters, digitizers, and databases to develop new data as well as manage existing data.

GRASS is ideal for use in engineering and land planning applications. Like other GIS packages, GRASS can display and manipulate vector data for roads, streams, boundaries, and other features. GRASS can also be used to keep maps updated with its integral digitizing functions. Another feature of GRASS is its ability to use raster, or cell, data. This is particularly important in spatial analysis and design. GRASS functions can convert between vector data to raster data for seamless integration.

GRASS' strengths lie in several fields. The simple user interface makes it an ideal platform for those learning about GIS for the first time. GRASS is capable of reading and writing maps and data to many popular commercial GIS packages including ARC/Info and Idrisi. Users wishing to write their own code can do so by examining existing source code, interfacing with the documented GIS libraries, and using the GRASS Programmers Manual. This allows more sophisticated functionality to be integrated in GRASS.

The ability to work with raster data gives GRASS the unique ability to function as a surface modeling system. GRASS contains more than 100 multi-function raster analysis and manipulation commands. Surface processes such as rainfall-runoff modeling, flowline construction (as shown), slope stability analysis, and spatial data analysis are just a few of the many applications of GRASS to engineering and land planning. Since many of the raster tools are multi-functional, users can create their own maps from GRASS data analysis.

In addition to standard two-dimensional analysis, GRASS allows users to view data in three-dimensions. Raster maps, vector maps, and sites data can be used for visualization. Example applications of such capabilities include airspace analysis for airport planning (as shown), terrain analysis and “flybys”, and spatial trends. Tools in GRASS allow the user to animate any spatial data available with options to switch between data layers “on-the-fly”. Data used in 3-D visualization may also be saved as still pictures, or as mpeg movie files for later replay and analysis.

Accompanying its land planning and engineering applications, GRASS contains a suite of tools to aid in hydrologic modeling and analysis. Currently, tools are also available for performing such functions as watershed analysis, curve number generation, flood analysis, and stream channel characteristics for comprehensive watershed modeling. Other GRASS programs can generate graphs, statistics, and charts of modeled and calibrated data. Additionally, GRASS can use field data for model input or simulate parameters based on numerical data.

In addition to the traditional command line version of GRASS, a new user interface, based on Tcl/Tk has been written. This puts the power of spatial analysis and modeling into an easy to use Graphical User Interface that is platform-independent. This intuitive user interface lets users quickly and easily view, manipulate, and use data. Nearly all of the programs available in GRASS are available in the new GUI, with the standard command-line still available, giving users all of the functionality of GRASS.

This manual is part of a comprehensive set of documentation written to support GRASS. This Users Guide consists of a complete set of command references for all current GRASS functions and tools, including examples. An installation guide and fact sheet guides users through the installation process. For those wishing to write their own spatial analysis and modeling applications for GRASS, a Programmers Guide is also available. GRASS runs on a variety of UNIX and Linux platforms including SUN SPARCstations and Ultras, HP, Silicon Graphics, and PC’s running Windows 95 and Windows NT.

The GRASS Development Team is currently working to further upgrade and enhance the capabilities of GRASS. Future developments include tools that give the user the ability to work completely in 3-D, a capability that does not exist in any other GIS package. Users will be able to work with raster elevation data as well as vector and sites data in the 3-D environment, adding to the visualization capabilities of GRASS. Enhancements in the numerical processing functions of GRASS also now allow for floating-point operations to be performed on data.

For the latest information on GRASS contact the GRASS Development Team at grass@baylor.edu or visit our web sites at:

http://www.baylor.edu/~grass if you're in the U.S.

http://www.geog.uni-hannover.de/grass if you're in Europe

Look for our worldwide mirrors!

The GRASS Development Team is:

Bruce Byars and Markus Neteler are the development team leaders and coordinators.

Helena Mitasova and Bill Brown of the GMS Lab at UIUC have made significant contributions with the development of GRASS 5.

Additional authors include:
Lisa Zygo, Edward Zarecky, Jacques Bouchard, Steve Clamons, Brent Duncan, Jason Cipriano, Jim Westervelt, Michael Shapiro, Darrell McCauley, Dave Gerdes, Bill Hughes, Bernhard Reiter, Brook Milligan, Eliot Cline, Jaro Hofierka, Clay Cockrell, and Bob Lozar. See the web pages for author affiliations.

Note:

Many other people have contributed to the GRASS GIS. Without any one of them, GRASS would not exist in its current form. The authors of the individual programs are listed at the end of their manual page in the GRASS users manual, however, numerous authors of bug fixes and enhancements as well as people who have been working on coordination, integration, documentation and testing are not mentioned.

Please allow us to extend our most cordial thanks to all of you. If you contributed to GRASS at any point during its existence, let us know your name and e-mail address so we can add your name to the comprehensive on-line list.

To reference GRASS:

GRASS Development Team, 1999, Geographic Resources Analysis and Support System - GRASS: Baylor University, Waco, Texas.

GRASS Development Team
Center for Applied Geographic and Spatial Research
Baylor University
P.O. Box 97351
Waco, Texas, U.S.A. 76798-7351

m.bsplit

NAME
m.bsplit **-** Splits a large (greater than 1,048,000 megabyte) file into smaller portions.
(SCS GRASS File Management Program)

GRASS VERSION
4.x,5.x

SYNOPSIS
m.bsplit
m.bsplit help
m.bsplit input=name [size=value] [prefix=name]

DESCRIPTION
m.bsplit works much like the UNIX command csplit; however, it was developed to be used on binary, not ASCII, files.

The first parameter, size, has a default value of 500000. The user may assign larger or smaller values, as long as 1000000 is not exceeded.

The second parameter, prefix, is comparable to the UNIX csplit command use of "x" as a prefix to output files. *m.bsplit* uses the default of "bx" as a prefix on output.

OPTIONS
Parameters:

input=name	The name of the binary input file to be split.
size=value	The value, in megabytes, of one of the new, split-up files created from the input file.
prefix=name	File name prefix assigned to the new (split) files.

SEE ALSO
The UNIX csplit command.

AUTHOR
R.L. Glenn, USDA, SCS, NHQ-CGIS

m.datum.shift

NAME
m.datum.shift **-** Datum shift program.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x

SYNOPSIS
m.datum.shift lat=dd.mm.ss{n|s} lon=dd.mm.ss{e|w} is=input_spheroid os=output_spheroid dx=xshift dy=yshift dz=zshift

DESCRIPTION
m.datum.shift returns geographic coordinates based on a different spheroid (and datum) than the one used to obtain the original coordinates.

The input and output spheroids, is and os, are the spheroids for two different datums. The input spheroid is the one on which the original coordinates are based. The output spheroid is that on which the resultant coordinates will be based. The "shifting" occurs between the two datums. The shift values, dx, dy, and dz, are constants. They indicate the mean differences between points in the second datum versus the first as measured in meters.

The list of spheroids available is somewhat dynamic. It may not contain exactly the ones listed below. To determine the current list of possible spheroids, type in the command:

m.datum.shift lat=0n lon=0w dx=0 dy=0 dz=0 is=help os=help

A list of available spheroids will be printed on the screen. If the spheroid desired is not on the list, the values for the semi-major axis and the eccentricity squared for the spheroid may be entered in place of a spheroid name in the following format:

s=a=semi-major_axis,e=eccentricity_squared

SOME POSSIBLE SPHEROIDS
(The on-line listing includes only the spheroid names)

Spheroid	Commonly used for:	Semi-major axis	Eccentricity sqrd
australian	Australia	6378160.0	0.0066945419
bessel	Japan	6377739.155	0.0066743722
clark66	N. America	6378206.4	0.006768658
clark80	France, Africa	6378249.145	0.0068035113
everest	India, Burma	6377276.345	0.0066378466
international	Europe	6378388.0	0.00672267
wgs72	worldwide coverage	6378135.0	0.006694317778

EXAMPLE

m.datum.shift lat=0n lon=175w is=clark66 os=wgs72 dx=-22 dy=157 dz=176

Results:
```
    lat=0.00.05.72999N
    lon=174.59.55.004133W
```

NOTES
Essentially, the program follows these steps. The original point, as defined by a latitude and a longitude, is converted to geocentric coordinates. The shift values are added to the geocentric coordinates. The summed values are then converted to latitude and longitude based on the output spheroid.

For a brief discussion of spheroids and datums see *m.ll2u*. For a brief discussion of geocentric coordinates see *m.ll2gc*.

This remains under testing is still an experimental program. It is part of an initial effort to incorporate geographic coordinates into GRASS.

SEE ALSO
m.gc2ll, m.ll2gc, m.ll2u, m.u2ll

AUTHOR
Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

m.dem.examine

NAME
m.dem.examine **-** Provides a terse description of USGS Digital Elevation Model (DEM) data files stored on 1/2-inch magnetic tape.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.dem.examine input=name blocksize=value

DESCRIPTION
m.dem.examine reads Digital Elevation Model (DEM) terrain elevation data produced and supplied by the USGS from the input file specified by name, in the block size specified by value. If no input file is named on the command line, the program assumes that /dev/rmt0 is the input file. The information provided to the user comes from each file included on tape. Included are the maximum and minimum elevation values and the approximate UTM boundaries of each file. The program assumes unlabeled tapes in ASCII format with no header or trailer files.

SEE ALSO
g.region, m.dem.extract, m.examine.tape, r.rescale, r.support

AUTHOR
Andrew Heekin, U.S. Army Construction Engineering Research Laboratory

m.dem.extract

NAME
m.dem.extract **-** Extracts USGS Digital Elevation Model (DEM)
data from 1/2-inch magnetic tape.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.dem.extract
m.dem.extract help
m.dem.extract input=name output=name blocksize=value start=value end=value

DESCRIPTION
m.dem.extract extracts USGS Digital Elevation Model (DEM) elevation data that fits into the user's current geographic region from the input file input, in blocks of blocksize bytes. If no input file is specified by the user, input is taken from /dev/rmt0, by default. Results are placed in the named output file, and stored beneath the cell directory of the user's current mapset. *m.dem.extract* will only extract data that fall within the boundaries of the user's current geographic region. Data falling outside this region will be ignored. *m.dem.extract* will not complain if the input file does not cover the entire geographic region. The user should run *m.dem.extract* without specifying output file (which only examines the input file) prior to extracting the data to determine the size of the geographic region needed. If the block size is unknown run the command *m.examine.tape*.

The user can run this program either non-interactively or interactively. The program will be run non-interactively if the user specifies program arguments on the command line, using the form:

m.dem.extract input=name [output=name] [blocksize=value] [start=value] [end=value]

Alternately, the user can simply type *m.dem.extract* on the command line, without program arguments. In this case, the user will be prompted for needed parameter values using the standard GRASS interface described in the manual entry for *parser*.

Parameters:
input=name The full path name of tape device or file from which DEM data are to be extracted.
Default: /dev/rmt0

output=name The name to be assigned to the output file containing raster DEM data extracted from 1/2-inch tape.

blocksize=value The physical block size (record length) of each record, in bytes. *m.examine.tape* can be used to determine block size.

start=value First DEM quad to look at (default 1).

end=value Last DEM quad to look at (default 9999999).

NOTES
The user should check the boundaries and resolution of the current region setting (see *g.region*) BEFORE extracting data, since *m.dem.extract* will only extract the data that falls within these boundaries and only use the set resolution during extraction.

Warning: This program assumes that the tape has ONLY DEM data. If the tape contains other files (DLG, for example) the program will not skip over them. In this case, forward the tape to the first DEM file. Use the appropriate UNIX commands (mt to forward the tapes past the non-DEM files, and dd to extract and copy the correct files) on the tape on another machine first to extract the non-DEM data from the tape.

USGS Tapes: This version of *m.dem.extract* is sensitive to formatting errors. If the format on a tape is wrong, the program exits with the error message: "The file with incorrect data format encountered ..." Nevertheless, the program continues. In this case, run the program without specifying input files. If the program exits again with the same error message, there is an inconsistency in the data format.

The format error: Each USGS Dem file consists of A and B, and sometimes C, records--

type A record	header
type B record	collection of profiles with various numbers of elevations
type C record	accuracy

This new version of *m.dem.extract* cannot skip over any records to the end of a DEM file. It first determines how many records there are in a file, reads those records, and then proceeds to the header (record A) of the next file. Thus it is very important for the program to know whether or not there is a C record to be read at the end of each file. This information is given in the file's header record (type A) which has a field set to 1 when a C-record exits, and to 0 when a C-record does not exist.

In some DEM files, the accuracy field is set to 0, even though a C-record does exist. Because of this error, when the program tries to read the data, it will not try to read the C-record, but instead attempts to go on to the next file and read its header--a type A record. This results in the format error because type A records and type C records have different formats, and the program is trying to read a type A record when the tape is presenting a type C record.

When this format error in a file is encountered, forward the tape to the next file.

SEE ALSO
g.region, m.examine.tape, r.rescale, r.support, parser

AUTHOR
Andrew Heekin, U.S. Army Construction Engineering Research Laboratory

Improvements to program code were made for GRASS 4.0 by David Satnik, Central Washington University

Major enhancements for GRASS 4.1 to allow the input to be a file or tape were made by Olga Waupotitsch, U.S. Army Construction Engineering Research Laboratory

m.dmaUSGSread

NAME
m.dmaUSGSread **-** Extracts digital terrain elevation data (DTED) produced by the Defense Mapping Agency (DMA) but supplied by the USGS (in a different tape format) on 1/2- inch magnetic tape.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
dd [if=tapedev] ibs=input-block-size | m.dmaUSGSread top=value bottom=value left=value right=value output=name logfile=name

DESCRIPTION
m.dmaUSGSread extracts digital terrain elevation data (DTED) that was produced by the Defense Mapping Agency (DMA) but bought from the USGS; these two agencies distribute the same data in a slightly different tape formats. This program requires the use of the UNIX command dd to read the 1/2-inch magnetic tape. *m.dmaUSGSread* should be used prior to using the GRASS programs *m.rot90* and *r.in.ll*.

Parameters for the UNIX dd command are listed below:
if=tapedev The path name of the tape drive from which input will be taken (usually /dev/rmt0).

ibs=input-block-size The physical blocking value of the data on tape, usually written on the tape spool. If this value is unknown, run *m.examine.tape* prior to running *m.dmaUSGSread*.

Parameters:
top=value Beginning row number of data.
Options: 1-1201

bottom=value Ending row number of data.
Options: 1-1201

left=value Beginning column number of data.
Options: 1-1201

right=value Ending column number of data.
Options: 1-1201

output=name Name of the output file to hold the extracted DEM data.

logfile=name Name of a file to hold related information about the extracted data.

EXAMPLE
The command:

```
dd if=/dev/rmt0 ibs=10240 | \ m.dmaUSGSread top=1 bottom=400 left=1 right=500 \ output=dem logfile=log
```

will extract data from /dev/rmt0 and put the first 400 rows and first 500 columns into the file dem located in the user's current directory.

SEE ALSO
Options for Acquiring Elevation Data by Stuart Bradshaw (ADP Report N-87/ USACERL1988).
DTED and DEM Elevation Data Extraction, by Stuart Bradshaw, Mary Martin, and Chester Kos (ADP Report N-87/22, USACERL, 1988).

UNIX manual entry for dd

g.region, *m.dted.examine*, *m.dted.extract*, *m.examine.tape*, *m.region.ll*, *m.rot90*, *r.describe*, *r.in.ll*, *r.rescale*, *r.slope.aspect*

AUTHORS
James Farley, Arkansas Archeological Survey, University of Arkansas
Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

m.dted.examine

NAME
m.dted.examine **-** Provides a terse description of level 1 and 2 digital terrain elevation data (DTED) files produced and distributed by the Defense Mapping Agency (DMA) on 1/2-inch magnetic tapes.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.dted.examine [tapedev]

DESCRIPTION
m.dted.examine scans digital terrain elevation data (DTED - Levels 1 and 2) that was produced and supplied by the Defense Mapping Agency (DMA) on 1/2-inch magnetic tape from the tape device specified by tapedev, and prints to the screen a terse description of each file.

If the user specifies no tapedev on the command line the program assumes that input is coming from /dev/rmt0. The information provided by *m.dted.examine* includes the UTM borders, cell resolution, and number of elevation profiles in each file.

BUGS
The header file for DTED Level 1 and 2 data was changed in 1987. *m.dted.examine* and *m.dted.extract* operate only on DTED data containing pre-1987 headers. DTED data containing the pre-1987 headers may be purchased from the Defense Mapping Agency (DMA) upon request.

SEE ALSO
g.region, m.dmaUSGSread, m.dted.extract, m.examine.tape, m.region.ll, m.rot90, r.describe, r.in.ll, r.rescale, r.slope.aspect

Pursuance of Elevation Data by Stuart Bradshaw
USACERL DEM and DTED Elevation Extraction by Stuart Bradshaw, Mary Martin, and Chester Kos, USACERL

AUTHORS
Michael Shapiro, U.S. Army Construction Engineering Research Laboratory
Andrew Heekin, U.S. Army Construction Engineering Research Laboratory

m.dted.extract

NAME
m.dted.extract **-** Extracts digital terrain elevation data (DTED **-** levels 1 and 2) produced and supplied by the Defense Mapping Agency (DMA) on 1/2-inch magnetic tapes.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.dted.extract if=tapedev of=outfile hf=headfile n=lat s=lat e=lon w=lon

DESCRIPTION
m.dted.extract extracts DTED (Levels 1 and 2) digital terrain elevation data produced by the Defense Mapping Agency (DMA) from 1/2-inch magnetic tapes obtained from DMA. Data is read from the input file (if) specified by tapedev. If the user does not specify any input file name on the command line, the program assumes that input is coming from /dev/rmt0. The extracted data is placed in the output file (of) specified by outfile. This program should be used in conjunction with the programs *m.rot90* and *r.in.ll* to convert DTED data to GRASS raster format.

Parameters:
if The pathname of the tape device where the raw DTED data exists (default is /dev/rmt0)

of The full pathname of the output file into which the extracted tape data is to be copied

hf The full pathname of a file to contain descriptive information about the extracted data; should be placed in the same directory as the output file

n North latitude value defining the boundaries of the extraction geographic region (format: dd.mm.ss[n|s])

s South latitude value defining the extraction geographic region (format: dd.mm.ss[n|s])

ea East longitude value defining the extraction geographic region (format: dd.mm.ss[e|w])

w West longitude value defining the extraction geographic region (format: dd.mm.ss[e|w])

The *n, s, e,* and *w* parameters define a geographic region that should completely encompass the data set. dd.mm.ss are degree, minute, and second values. Only data that falls within this defined geographic region will be extracted from the tape.

EXAMPLE
The command

```
m.dted.extract if=/dev/rmt0 of=dted.out hf=dted.head n=37.30.00n s=37.15.00n e=103.30.00w w=103.45.00w
```

will extract DTED data from /dev/rmt0, store it in a file named dted.out, and store some supporting information in the file dted.head. Only data that falls within the geographic region defined by the coordinates n,s,e and w will be extracted.

NOTE

The user should examine the contents of the header file produced by this program; it contains information needed as inputs to the data rotation and raster file import programs *m.rot90* and *r.in.ll*, respectfully.

BUGS
The format of the header file for DTED Level 1 and 2 data was changed in 1987. *m.dted.extract* and *m.dted.examine* only operate on DTED data containing pre-1987 headers. DTED data containing the pre-1987 headers may be purchased from the DMA upon request.

SEE ALSO
g.region, m.dmaUSGSread, m.dted.examine, m.examine.tape, m.region.ll, m.rot90, r.describe, r.in.ll, r.rescale, r.slope.aspect

Pursuance of Elevation Data by Stuart Bradshaw
USACERL DEM and DTED Elevation Extraction by Stuart Bradshaw, Mary Martin, and Chester Kos, USACERL

AUTHOR
Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

m.eigensystem

NAME
m.eigensystem **-** Computes eigen values and eigen vectors for square matrices.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x

SYNOPSIS
m.eigensystem < *inputfile*

DESCRIPTION
m.eigensystem determines the eigen values and eigen vectors for square matrices. The inputfile must have the following format: the first line contains an integer K which is the number of rows and columns in the matrix; the remainder of the file is the matrix, i.e., K lines, each containing K real numbers. For example:

```
3
462.876649   480.411218   281.758307
480.411218   513.015646   278.914813
281.758307   278.914813   336.326645
```

The output will be K groups of lines; each group will have the format:

```
E   real part   imaginary part   relative importance
V   real part   imaginary part
    ... K lines ...
N   real part   imaginary part
    ... K lines ...
W   real part   imaginary part
    ... K lines ...
```

The E line is the eigen value. The V lines are the eigen vector associated with E. The N lines are the V vector normalized to have a magnitude of 1. The W lines are the N vector multiplied by the square root of the magnitude of the eigen value (E).

For the example input matrix, the output would be:

```
E       1159.7452017844               0.0000000000   88.38
V       0.6910021591          0.0000000000
V       0.7205280412          0.0000000000
V       0.4805108400          0.0000000000
N       0.6236808478          0.0000000000
N       0.6503301526          0.0000000000
N       0.4336967751          0.0000000000
W       21.2394712045         0.0000000000
W       22.1470141296         0.0000000000
W       14.7695575384         0.0000000000

E       5.9705414972          0.0000000000   0.45
V       0.7119385973          0.0000000000
V       -0.6358200627         0.0000000000
V       -0.0703936743         0.0000000000
N       0.7438340890          0.0000000000
N       -0.6643053754         0.0000000000
N       -0.0735473745         0.0000000000
W       1.8175356507          0.0000000000
W       -1.6232096923         0.0000000000
W       -0.1797107407         0.0000000000

E       146.5031967184        0.0000000000   11.16
V       0.2265837636          0.0000000000
```

```
V       0.3474697082           0.0000000000
V       -0.8468727535          0.0000000000
N       0.2402770238           0.0000000000
N       0.3684685345           0.0000000000
N       -0.8980522763          0.0000000000
W       2.9082771721           0.0000000000
W       4.4598880523           0.0000000000
W       -10.8698904856         0.0000000000
```

NOTES
The relative importance of the eigen value (E) is the ratio (percentage) of the eigen value to the sum of the eigen values. Note that the output is not sorted by relative importance.

In general, the solution to the eigen system results in complex numbers (with both real and imaginary parts). However, in the example above, since the input matrix is symmetric (i.e., inverting the rows and columns gives the same matrix) the eigen system has only real values (i.e., the imaginary part is zero). This fact makes it possible to use eigen vectors to perform principle component transformation of data sets. The covariance or correlation matrix of any data set is symmetric and thus has only real eigen values and vectors.

PRINCIPLE COMPONENTS
To perform principle component transformation on GRASS data layers, one would use *r.covar* to get the covariance (or correlation) matrix for a set of data layers, *m.eigensystem* to extract the related eigen vectors, and *r.mapcalc* to form the desired components. For example, to get the eigen vectors for 3 layers:

(echo 3; r.covar map.1,map.2,map.3) | m.eigensystem

Note that since *m.covar* only outputs the matrix, we must manually prepend the matrix size (3) using the echo command.

Then, using the W vector, new maps are created:

*r.mapcalc 'pc.1 = 21.2395*map.1 + 22.1470*map.2 + 14.7696*map.3'*
*r.mapcalc 'pc.2 = 2.9083*map.1 + 4.4599*map.2 - 10.8699*map.3'*
*r.mapcalc 'pc.3 = 1.8175*map.1 - 1.6232*map.2 - 0.1797*map.3'*

NOTES
The source code for this program lives under /src.contrib/CERL/misc/m.eigensystem and requires a Fortran compiler.

SEE ALSO
r.covar, r.mapcalc, r.pca, r.rescale

AUTHOR
This code uses routines from the EISPACK system. The interface was coded by Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

m.examine.tape

NAME
m.examine.tape **-** Provides a description of the files on a 1/2-inch magnetic tape.
(GRASS File Management Tool)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.examine.tape

DESCRIPTION
m.examine.tape is a convenient tool for previewing a tape and obtaining block size values. After the user enters the command

m.examine.tape

the program will prompt the user for the full path name of the tape device on which the tape is mounted. In most cases, this will be /dev/rmt0. However, the user must enter this, as there is no default value for the tape device name. The user is then prompted to enter the buffer size to be used when reading the tape. The buffer size is the amount of memory allocated to read one physical record of the tape. If the user hits RETURN, the program will assume a default buffer size of 32767 bytes.

m.examine.tape reads the tape specified by the user and reports back the block size (record length) and the number of blocks for each file contained on the tape. *m.examine.tape* can be used on any 1/2-inch magnetic tape. The user has the option of sending the output into a file or viewing the output on the terminal screen.

NOTES
The buffer size is the amount of memory allocated to read one physical record on the tape. Magnetic tape devices will accept a request to read more bytes than actually exist in any given record on a tape. However, *m.examine.tape* reads only as many bytes as physically exist on the tape and returns the number of bytes actually read. The user is allowed to alter the buffer size in order to request smaller reads for tape devices unable to handle requests this large (32767 bytes), or to request larger reads for tapes with larger record sizes read on drives able to handle larger record sizes.

Note that *m.examine.tape* cannot be used to examine the content of either ordinary files or 1/4-inch tape cartridges.

SEE ALSO
m.dem.examine, m.dem.extract, m.dmaUSGSread, m.dted.examine, m.dted.extract, m.lulc.USGS, m.lulc.read

AUTHOR
Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

m.flip

NAME
m.flip **-** Flips elevation data extracted from systems that retrieve data by rows from south to north.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.flip
m.flip help
m.flip [-q] input=name output=name rows=value cols=value bpc=value

DESCRIPTION
m.flip is similar to *m.rot90*. However, rather than rotating a raster file, *m.flip* flips the contents of the raster generated by tape extraction programs about the east-west axis. Flipping may be necessary to compensate for the orientation of the data read from tape. This program can be used in conjunction with the program *r.in.ll* to convert this rotated data into GRASS raster format.

m.flip requires five inputs to be entered by the user. These parameters and the optional flag setting are described below.

Flag:
-*q* Run quietly, suppressing output of messages on program progress to standard output.

Parameters:
input=name The full pathname of an existing file containing the data to be flipped.

output=name The full pathname of the output file in which the rotated data are to be stored.

rows=value The number of rows of data in the input file. Values must be positive integers.

cols=value The number of columns of data in the input file. Values must be positive integers.

bpc=value The number of bytes per cell (i.e., per data value) in the input file. Values must be positive integers.

EXAMPLE
The following command:

```
m.flip  input=/tmp/foo.out  output=/tmp/flip.out rows=301  cols=358  bpc=2
```

will rotate the file /tmp/foo.out, and place the rotated file in /tmp/flip.out. Here, the input file is 301 rows by 358 columns, at 2 bytes per data value.

SEE ALSO
g.region, m.dmaUSGSread, m.dted.examine, m.dted.extract, m.examine.tape, m.region.ll, m.rot90, r.describe, r.in.ll, r.rescale, r.slope.aspect

Pursuance of Elevation Data by Stuart Bradshaw
USACERL DEM and DTED Elevation Extraction by Stuart Bradshaw, Mary Martin, and Chester Kos, USACERL

AUTHORS

Donald Newcomb, U.S. Naval Oceanographic Office, borrowing heavily from the *m.rot90* program written by Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

m.gc2ll

NAME
m.gc2ll **-** Converts geocentric to geographic coordinates.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.gc2ll x=# y=# z=# s=spheroid

DESCRIPTION
m.gc2ll returns geographic coordinates for geocentric ones supplied by the user. It performs the reverse operation of the GRASS program *m.ll2gc*. The x, y and z values are the three dimensions needed to locate a point in three- dimensional space. The values that are printed include the latitude, the longitude and the height above (or distance below) the spheroid.

The list of spheroids available is somewhat dynamic. It may not contain exactly the ones listed below. To determine the current list of possible spheroids, simply type in:

m.gc2ll x=0 y=0 z=0 s=help

A list of available spheroids will be printed on the screen. If the spheroid desired is not on the list, the values for the semi-major axis and the eccentricity squared for the spheroid may be entered in place of a spheroid name in the following format:

s=a=semi-major_axis,e=eccentricity_squared

SOME POSSIBLE SPHEROIDS
(The on-line listing includes only the spheroid names)

Spheroid	Commonly used for:	Semi-major axis	Eccentricity sqrd
australian	Australia	6378160.0	0.0066945419
bessel	Japan	6377739.155	0.0066743722
clark66	N. America	6378206.4	0.006768658
clark80	France, Africa	6378249.145	0.0068035113
everest	India, Burma	6377276.345	0.0066378466
international	Europe	6378388.0	0.00672267
wgs72	worldwide coverage	6378135.0	0.006694317778

EXAMPLE

m.gc2ll x=0.0 y=0.0 z=6356750.520017 s=wgs72

```
Results:
   lat=90N
   lon=90W
   h=0.0000
```

NOTES

This is an experimental program. It is part of an initial effort to incorporate geographic coordinates into GRASS.

SEE ALSO
m.datum.shift, m.ll2gc, m.ll2u, m.u2ll

AUTHOR
Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

m.geo

NAME
m.geo **-** Calculates conversion coordinates for geographic positions.
(SCS GRASS Map Development Program)

GRASS VERSION
4.x

SYNOPSIS
m.geo
m.geo help

DESCRIPTION
Note: The use of this module has been replaced by *m.proj*.

This program allows a user to interactively: convert projection coordinates Northings and Eastings to Latitude and Longitude values, or allows a user to interactively: convert Latitude and Longitude values to projection coordinate Northings and Eastings.

It allows a user to do all of the above: reading from a file, writing to the screen, or reading from the keyboard, writing to a file, or reading from a file, writing to a file.

Note: The program does not transform GRASS files, it is designed to determine coordinate values on an individual position.

Several map projections are currently supported:

stp - State Plane, for all CONUS, Alaska, Hawaii, Puerto Rico, Guam, Virgin Islands, American Samoa, Northern Mariana Islands, Palau, and U.S. Minor Outlying Islands

airy - Airy-F
aea - Albers Equal Area-FI
apian - Apian Globular I-F
aeqd - Azimuthal equidistant-FI
aitoff - Aitoff-F
august - August Epicycloidal-F

bacon - Bacon Globular-F
bipc - Bipolar Conic-FI
boggs - Boggs Eumorphic-F
bonne - Bonne-FI

cass - Cassini-FI
cc - Central Cylindrical-FI
cea - Cylindrical Equal Area-FI
collg - Collignon-FI

dense - Denoyer Semi-Elliptical-F

eck1 - Eckert I-FI eck2 - Eckert II-FI
eck3 - Eckert III-FIeck4 - Eckert IV-FI
eck5 - Eckert V-FI eck6 - Eckert VI-FI

eisen - Eisenlohr-F
eqc - Equidistant Cylindrical-FI
eqdc - Equidistant Conic-FI

fourn - Fournier Globular-F

gall - Gall (Stereographic)-FI
goode - Goode Homolosine-F
gnom - Gnomonic-FI

hammer - Hammer (Elliptical)-F
hataea - Hatano Asymmetrical Equal Area-FI

lagrng - Lagrange-F
laea - Lambert Azimuthal Equal Area-FI
leac - Lambert Equal Area Conic-FI
lcc - Lambert Conformal Conic-FI
loxim - Loximuthal-FI

mbtfpp - McBryde-Thomas Flat-Polar Parabolic-FI
mbtfps - McBryde-Thomas Flat-Polar Sinusoidal-FI
mbtfpq - McBryde-Thomas Flat-Polar Quartic-FI
merc - Mercator-FI
mill - Miller-FI
moll - Mollweides-FI

nicol - Nicolosi Globular-F
nsper - General Vertical Persepective-FI

ocea - Oblique Cylindrical Equal Area-FI
omerc - Oblique Mercator-FI
ortel - Ortelius-F
ortho - Orthographic-FI

parab - Caster Parabolic-FI
pconic - Perspective Conic-F
poly - Polyconic (American)-FI
putp2 - Putnins P2'-FI
putp5 - Putnins P5-FI

quau - Quartic Authalic-FI

rpoly - Rectangular Polyconic-F
robin - Robinson-FI

sinu - Sinusoidal-FI
stere - Stereographic-FI

tcc - Transverse Central Cylindrical-FI
tcea - Transverse Cylindrical Equal Area-FI
tmerc - Transverse Mercator-FI
tpers - Tilted perspective-FI

ups - Universal Polar Stereographic-FI

utm - Universal Transverse Mercator-FI

vandg - Van der Grinten-FI
vandg2 - Van der Grinten II-F
vandg3 - Van der Grinten III-F
vandg4 - Van der Grinten IV-F

wag7 - Wagner VII-F
wink1 - Winkel I-FI
wintri - Winkel Tripel-F

Each of the above projections (with the exception of State Plane) can be computed with the following spheroids:

MERIT - MERIT 1983
GRS80 - GRS 1980(IUGG, 1980)
IAU76 - IAU 1976
airy - Airy 1830
aust_ntl - Australian Natl, S. Amer., IAU 64
GRS67 - GRS 67(IUGG 1967)
bessel - Bessel 1841
clrk66 - Clarke 1866
clrk80 - Clarke 1880 mod.
everest - Everest 1830
hough - Hough
intl - International 1909 (Hayford)
krass - Krassovsky, 1942
mercury - Mercury 1960
mod_airy - Modified Airy
mod_ever - Modified Everest
mod_merc - Modified Merc 1968
new_intl - New International 1967
SEasia - Southeast Asia
walbeck - Walbeck
WGS66 - WGS 66
WGS72 - WGS 72
sphere - Sphere of 6370997 m

INPUT FILE FORMAT

When reading from a file of LATITUDE/LONGITUDE data the file will contain three (3) columns of information: the first column - latitude   - in degrees minutes seconds, the second column - longitude - in degrees minutes seconds, the third column - zone - zero(0) if not required.

For example:

```
        +40 36 31.4563   -87 2 7.8193   16
        40n 36 31.4563   87w 2 7.8193   16
```

When reading from a file of PROJECTION COORDINATES data the file will contain three (3) columns of information:
first column - easting - ground coordinates
second column - northing - ground coordinates
third column - zone - zero(0) if not required.

For example:

```
        500000.00    4496918.64   16   <- utm
        -424489.11   1908736.13   0    <- lambert
```

Note: NO column headings are required, just the numbers.

SEE ALSO
Mapgen proj, *m.proj*

AUTHOR
R.L. Glenn, USDA, SCS, NHQ-CGIS

m.get.fips

NAME
m.get.fips **-** Finds a state and county FIPS code for the user.
(SCS GRASS Data Import/Processing)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.get.fips
m.get.fips help

DESCRIPTION
m.get.fips prompts the user for a State FIPS or two-letter state abbreviation code, and then for the name of a county located in that state. The program then prints the State and County FIPS code numbers to the user's terminal screen (standard out).

EXAMPLE

m.get.fips

```
        Enter State FIPS code or press RETURN if unknown:
        Enter Two-Letter State or Territory Abbreviation:
        pa
        Enter County Name:  cambria
        42  21
```

AUTHOR
M.L.Holko, USDA, SCS, NHQ-CGIS

m.get.stp

NAME
m.get.stp **-** Finds a State Plane projection zone code for the user.
(SCS GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.get.stp
m.get.stp help

DESCRIPTION
m.get.stp prompts the user for a State FIPS code or two-letter state abbreviation, and then for the name of a county located within this state. The program then prints to standard output the USGS State Plane zone code number associated with this county.

EXAMPLE

m.get.stp

```
Enter State FIPS code or RETURN if unknown:
Enter Two-Letter State or Territory Abbreviation:  pa
Enter County Name:  cambria
3702
```

AUTHOR
M.L. Holko, USDA, SCS, NHQ-CGIS

m.in.e00

NAME
m.in.e00 **-** Read an ESRI e00 file
(GRASS Raster Data Import Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.in.e00
m.in.e00 help
m.in.e00 [-s] input=name [mapset=name] [action=what to do] [verbose=debug level] [logfile=name]

DESCRIPTION
The *m.in.e00* program is designed to import ESRI Arc/Info e00 ASCII archives. The input file must be in ASCII (no binary compressed) e00 file.

The program not only can analyze the content of an Arc/Info file, but try to create the objects described (geometry and attributes).

m.in.e00 will be run non-interactively if the user specifies program arguments on the command line, using the form:

m.in.e00 [-s] input=name [mapset=name] [action=what to do] [verbose=debug level] [logfile=name]

Alternately, the user can simply type:

m.in.e00

on the command line without program arguments. In this case, the user will be prompted for parameter values using the standard GRASS user interface described in the manual entry for *parser*.

FEATURES
m.in.e00 attempts to retrieve all information in an Arc/Info export file (.e00) : points, line, polygon, and grid coverage. The attribute of each "vector" object (points, line, and polygon) is the Arc/Info coverage-ID (and not the coverage-#).

Grass files created have the name extracted from the first line of the e00 file, i.e. the name of the coverage.

The following rules are used : A line vector file is created when there is a non-empty ARC section, and neither PAL (Polygon Attribute Label) section, neither PAT (Point/Polygon Attribute Table) table in IFO section. Otherwise a polygon vector file is created, excepted when the ARC section is empty or doesn't exist, in which case a site file is created.

Dig_cats files are created if the PAT or AAT (Arc Attribute Table) tables have more attributes than the standard one. If there is one extra attribute, the dig_cats file has the name of the vector (dig) file created. If there is more than one extra attribute, the dig_cats files have names of the form cover_name.attribute_name. If you want to use one of them, you must rename it so it matches the name of the vector file. Doing this, *v.to.rast* will use it for the raster file created.

OPTIONS
m.in.e00 requires the user to enter the following information:

Parameters:
mapset=name For creating a new mapset for the data imported. This may be useful, since we cannot create a projection info file and a default window in an existing mapset. When a file is imported in the current mapset, you should take care that they are in the same projection. Unfortunately, you may run in trouble after that when using *g.region* (Cf BUGS).

action=what to do Five options: analyze, raster, lines, vector and all. By default, all is used and everything (grid, points, lines, polygons) is imported. Analyze produce no files, but only a log output (level 5) on stderr, and is useful to see whether the e00 file is clean or not. Raster imports only grid section. Lines imports only the geometry (no label are attached to the lines or polygons) of a point/line/polygon coverage. Vector imports a point/line/polygon coverage with their attributes.

verbose=debug level Number between 0 (no trace of what's happening) and 9 (very verbose log).

logfile=name Name of file where log info will be written. By default log info are directed to stderr.

BUGS AND CAVEAT
Binary files are rejected. You must use e00b2a before to convert binary e00 files to ASCII.

Example: `e00b2a < binary.e00 > ascii.e00`

You must know what to do with all the dig_cats file generated when importing a vector coverage.

No attribute (cats) file is produced when importing a grid.

The program may crash when unexpected data (mainly in IFO section) are found, or when the input file is corrupted.

There is no support for projection. using verbose=1 let you see the parameters of the e00 file, but you must create the DEFAULT_WIND and the PROJ file yourself.

New mapset are always created with proj=0 zone=99. If the default proj and zone are not the same, *g.region* complaints and *d.vect* or *d.rast* refuse to display your data. The only thing to do is to import in the current mapset.

SEE ALSO
e00b2a, g.mapsets, g.region, g.setproj, v.support, v.to.rast, v.in.shape

AUTHOR
Michel J. Wurtz, Laboratoire Territoires & Environnement, Ecole Nationale du Genie de l'Eau et de l'Environnement.
mw@engees.u-strasbg.fr

m.in.pl94.db3

NAME
m.in.pl94.db3 **-** Import Demographic records from Census PL94-171 dBase3 (CDROM) Files (GRASS Sites Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.in.pl94.db3 help
m.in.pl94.db3 [-f] [-p] [-n] sc=string [sc=str . .] in=/cdrom/pl9417ss.dbf > result_file

DESCRIPTION
This command may be run in command line mode only. This program is provided to read the Census PL94 data records from a file for one state. The file is usually on a CD-ROM, but this is not necessary providing the proper path is given for the *in*= parameter and the contents of the file duplicate that on CD-ROM.

The output consists of records of 517 characters extracted from the input file with a suitable line termination character added (0 to create a normal text file.

Users of GRASS 4.1 and the Census PL94 files on CD-ROM will generally use this program to select appropriate subsets of PL94 records; write them to one or more files; and then either process these files directly with *v.apply.census* to print demographics, create site maps or label vector maps, or use *m.in.stf1.tpe* to extract further subsets. NOTE: Unix utility programs such as grep or awk can also be used for processing the extracted PL94 files.

The PL94 CD-ROM data file for most states is very large, 100+ MB. This program may take 20 to 60 minutes for a single extraction, so plan carefully and don't discard useful subsets that you create on disk.

Parameters:
Parameters used must be in the order specified.

-*f* Output to stdout the list of all field name and starting positions for the Identification Section of the PL94 file. Other parameters are not necessary and will be ignored.

-*p* Print several information items to stderr input file is opened.

-*n* Output only the sequential number and PL94 logical record number (LOGRECNU) of matching records to stdout.

sc=string sc is starting column or a capitalized reference name from the Identification Section of the PL94 Data Dictionary; string is characters to match there. A negative test may also be made with: *sc!=string*. If more than one test is specified the results are "anded" resulting in conjunction of the results of the expressions. (See *m.in.stf1.tpe* for complete details.)

in Path/file for dBase files for state ss.

EXAMPLES
Insert the appropriate CD-ROM and mount it (see your system administrator). (We assume here that it is mounted as the file system /cdrom.)

To extract all Census tract summary records for the state of South Dakota:

m.in.pl94.db3 -p 11=050 in=/cdrom/pl9417sd.dbf > counties.sd

The 11=050 parameter selects those records which have the character string Summary Level (SUMLEV) field which is used in almost every selection from CD-ROM.

To extract all block records for Lawrence County, South Dakota:

m.in.pl94.db3 -p SUMLEV=100 CNTY=081 in=/cdrom/pl9417sd.dbf >

100 is the SUMLEV for blocks and 081 is the county code for Lawrence Co.

AUTHOR
Dr. James Hinthorne, GIS Laboratory, Central Washington University, August 1992.

m.in.stf1.db3

NAME
m.in.stf1.db3 **-** Import Demographic records from Census STF1A dBase3 (CDROM) Files
(GRASS Sites Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.in.stf1.db3 help
m.in.stf1.db3 [-f] [-p] [-n] sc=string [sc=str . .] in=/cdrom/stf1a0ss.dbf > *result_file*

DESCRIPTION
This command may be run in command line mode only. This program is provided to read and assemble the Census STF1A data records from a set of 10 files for one state. The files are usually on a CD-ROM, but this is not necessary providing the proper path is given for the in= parameter and all 10 files are in the same directory with their normal names.

The output consists of data from the 10 input files assembled into records identical to the STF1A tape distribution format. The output records are each about 4800 characters and a pair constitute a logical STF1A record.

Users of GRASS 4.1 and the Census STF1A files on CD-ROM will generally use this program to select appropriate subsets of STF1A records; write them to one or more files (in the STF1A tape format); and then either process these files directly with *v.apply.census* to print demographics, create site maps or label vector maps, or use *m.in.stf1.tpe* to extract further subsets.

The STF1A CD-ROM data sets for most states are very large, 100+ MB. This program may take 20 to 60 minutes for a single extraction, so plan carefully and don't discard useful subsets that you create on disk.

m.in.stf1.tpe is used as a post-processor to select further subsets of STF1A records.
NOTE: Unix utility programs such as awk and grep cannot be used for processing the STF1 files due to their very long record lengths.

Flags:
-f Output to stdout the list of all field name and starting positions for the Identification Section of the STF1A file. Other parameters are not necessary and will be ignored.

-*p* Print several information items to stderr as the 10 input files are opened.

-*n* Output only the sequential number and STF1A logical record number (LOGRECNU) of matching records to stdout.

Parameters:
Parameters used must be in the order specified.

sc=string sc is starting column or a capitalized reference name from the Identification Section of the STF1A Data Dictionary; string is characters to match there. A negative test may also be made with: sc!=string. If more than one test is specified the results are "anded" resulting in a conjunction of the results of the expressions. (See *m.in.stf1.tpe* for complete details.)

in= Path/file for dBase files for state ss.

EXAMPLES
Insert the appropriate CD-ROM and mount it (see your system administrator). (We assume here that it is mounted as the file system /cdrom.)

To extract all Census tract summary records for the state of South Dakota:

m.in.stf1.db3 -p 11=140 in=/cdrom/stf1a0sd.dbf > tracts.SD

The 11=140 parameter selects those records which have the character string Summary Level (SUMLEV) field which is used in almost every selection from CD-ROM.

To extract all block group records for Lawrence County, South Dakota:

m.in.stf1.db3 -p SUMLEV=150 CNTY=081 in=/cdrom/stf1a0sd.dbf

150 is the SUMLEV for block groups and 081 is the county code for Lawrence Co.

AUTHOR
Dr. James Hinthorne, GIS Laboratory, Central Washington University, May 1992.

m.in.stf1.tape

NAME
m.in.stf1.tape **-** Filter to extract lines from a text file based on column contents, especially for Bureau of the Census STF1 files.
(GRASS Support Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.in.stf1.tape
m.in.stf1.tape help
m.in.stf1.tape -f
m.in.stf1.tape [-n] sc=str [sc=str . . .] < infile > outfile

DESCRIPTION
This program must be run in command mode only.

This is a text filter program written in C especially to work with the Census STF1 files, but useful for selecting subsets of lines from any text file. It will work with arbitrarily long input lines, up to 10,000 characters. Input lines may be of variable length.

A sc=str condition which refers to columns beyond the end of the input line is assumed to be true.

Multiple tests are 'anded' into a single test; that is, lines which pass all tests are written to the output file.

Null characters are always filtered out; thus, this program can be used as a null filter by specifying a sc=str condition which will always be true (e.g., '1=?'). In the same way, files with lines terminated with <LF><CR> or just <CR> can be "fixed" to have the standard Unix <LF> terminator.

NOTE: One special property of *m.in.stf1.tape* is that it will pad (with 0) its output lines to 4806 characters if the line begins with the characters "STF1" and is greater than 4000 characters long. Experience has shown that some STF1 records are a few characters short (but no data has been omitted), and this corrects them so that other programs will be able to read full lines.

Input lines must be terminated with <LF>, <CR> or <LF><CR>. Output lines will be terminated with <LF> only.

For extracting lines from files, UNIX programs such as grep or awk are somewhat more flexible than *m.in.stf1.tape.tmp*, bur have line length limitations, and do not adapt to lines terminated with <LF> and <CR>.

This program was created as a preprocessor for Census STF1 files, to produce input files for the program *v.apply.census*. The program *m.in.stf1.db3* performs the same preprocessing function for the Census STF1 files distributed in "dBase 3" format on CD-ROM.

Parameters:
sc starting column number of a desired field in the input file, or is the name of one of the Identification Section field names for the STF1 records (all upper case letters). str is a string to match against input lines starting at column sc. sc=str may be repeated resulting in a conjunction (anding) of the results of each sc=str expression. A '?' may be used as a single character wild card in str; if sc=str

contains '?' or other shell interpreted characters, it should be protected in quotes. Preceding sc by 'N', or preceding the = by '!' reverses the sense of the test ("not equals").

EXAMPLES
m.in.stf1.tape 11=050 < infile > outfile
m.in.stf1.tape SUMLEV=050 < infile > outfile
m.in.stf1.tape 1=T450 '7=Bu??s' < infile > outfile
m.in.stf1.tape 51=tract N37=9753 < infile > outfile
m.in.stf1.tape 51=tract 37!=9753 < infile > outfile

Running the program with the ***-f*** flag generates the list of STF1 Identification Section field names to stdout.

BUGS
Input characters lexically less than 'space' (32 decimal; the "control" characters) which are not line terminators will be perceived as line terminators and thus cause improper functioning.

AUTHOR
Dr. James Hinthorne, GIS Laboratory, Central Washington University.

m.ipf

NAME
m.ipf **-** Iterative proportional fitting for error matrices.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x

SYNOPSIS
m.ipf
m.ipf help
m.ipf [-emz] [input=name] [format=string] [stop=value]

DESCRIPTION
m.ipf uses an error or confusion matrix produced by *r.coin* or *r.kappa*, smoothes zero counts, and does iterative proportional fitting to normalize the matrix.

OPTIONS
Flags:

-*e*	Indicate when the iterative algorithm finished.
-*m*	Print the marginals (row and column totals) with each matrix.
-*z*	Print the intermediate (smoothed) matrix.

Parameters:
input=name The input file must have the following format: the first line contains an integer K that is the number of rows and columns in the matrix; the remainder of the file is the matrix, i.e., K lines, each containing K integers. If the input is not specified on the command line, it may come from standard input.

format=string Specifies the format conversion string used to print the results. Default is %7.3f. For details, see the UNIX man page for printf.

stop=value The stopping criteria is a floating point number which actually specifies an integer maximum number of iterations and a fractional change in marginal. The default, 100.01, specifies that the iterative proportional fitting will stop at 100 iterations or when marginals do not change by 0.01, whichever comes first.

EXAMPLE
For the following input,

```
3
712    0   12
  0  584    2
 18    0  434
```

zero counts in the matrix will be smoothed:

```
711.249   0.438  12.314
  0.443 583.289   2.268
 18.309   0.273 433.418
```

and the matrix will be normalized to yield:

```
0.969 0.001 0.022
0.001 0.999 0.004
0.031 0.001 0.973
```

NOTES
Iterative proportional curve fitting is useful when comparing the output of image classification algorithms (for example, *i.maxlik* and *i.smap*), especially when training fields (signatures) and/or test fields are different. The diagonals of the normalized matrix can be used in a Tukey multiple comparison test.

SEE ALSO
Assessing Multiple Classifications - GRASS Tutorial on *m.ipf*

r.coin, *r.kappa*

Zhuang, X., B.A. Engel, X. Xiong, and C. Johanssen. 1994. Analysis of Classification Results of Remotely Sensed Data and Evaluation of Classification Algorithms, Photogrammetric Engineering and Remote Sensing (in press)

AUTHOR
James Darrell McCauley, Agricultural Engineering, Purdue University

m.ll2gc

NAME
m.ll2gc **-** Converts geographic coordinates to geocentric coordinates.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.ll2gc lat=dd.mm.ss{n|s} lon=dd.mm.ss{e|w} [h=height] s=spheroid

DESCRIPTION
m.ll2gc returns geocentric coordinates for geographic coordinates (latitude and longitude). Geographic coordinates are in degrees, minutes, and seconds and must include designation of north or south {n|s} and east or west {e|w}. The spheroid on which they are based must also be entered. The height (in meters) above the spheroid is optional.

The list of spheroids available is somewhat dynamic. It may not contain exactly the ones listed below. To determine the current list of possible spheroids, type in:

m.ll2gc lat=0n lon=0w s=help

A list of available spheroids will be printed on the screen. If the spheroid desired is not on the list, the values for the semi-major axis and the eccentricity squared for the spheroid may be entered in place of a spheroid name in the following format:

s=a=semi-major_axis,e=eccentricity_squared

SOME POSSIBLE SPHEROIDS
(The on-line listing includes only the spheroid names)

Spheroid	Commonly used for:	Semi-major axis	Eccentricity sqrd
australian	Australia	6378160.0	0.0066945419
bessel	Japan	6377739.155	0.0066743722
clark66	N. America	6378206.4	0.006768658
clark80	France, Africa	6378249.145	0.0068035113
everest	India, Burma	6377276.345	0.0066378466
international	Europe	6378388.0	0.00672267
wgs72	worldwide coverage	6378135.0	0.006694317778

EXAMPLE
m.ll2gc lat=0n lon=90w s=wgs72

```
Results:
   x=0.0
   y=6378135.0
   z=0.0
```

The distances designated by "x," "y" and "z" are in meters from the center of the earth. Because the sample point is on the equator and is at 90 degrees west, two of the three parameters have a value of zero (i.e., the point is at the origin of those two dimensions).

NOTES
For a brief discussion of spheroids see *m.ll2u*.

Geographic coordinates (latitude/longitude) describe the location of a point on the earth relative to the equator in the north/south directions and relative to Greenwich in the east/west directions. The same point will have different coordinates depending on the spheroid used as the approximation of the shape of the earth. The geocentric coordinates given here describe the point on the basis of the same spheroid, but they use the center of the spheroid as their origin. This can be thought of as being the center of the earth, which is why they are called "geocentric." Describing the location of a point in a spheroid requires three points: x, y, and z; these are measured out from the center along three different axes. The distances given are in meters.

This is an experimental program. It is part of initial efforts to incorporate geographic coordinates into GRASS.

SEE ALSO
m.datum.shift, m.gc2ll, m.ll2u, m.u2ll

AUTHOR
Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

m.ll2u

NAME
m.ll2u **-** Converts geographic coordinates to Universal Transverse Mercator (UTM) coordinates.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.ll2u
m.ll2u help
m.ll2u [-rwoz] spheroid=name [zone=value] [input=name] [output=name]

DESCRIPTION
m.ll2u converts geographic coordinates (i.e., latitudes and longitudes) to Universal Transverse Mercator (UTM) eastings and northings. The user must specify the spheroid on which to base the UTM conversion. The user may optionally specify the UTM zone; however, the program can determine this from the geographic coordinates submitted.

The list of spheroids available is somewhat dynamic. At the time of this release, available spheroids included: airy, australian, bessel, clark66, everest, grs80, hayford, international, krasovsky, wgs66, wgs72, and wgs84 (see table below).

This command can be run either non-interactively or interactively. The user can run the program non-interactively by entering desired flag settings and parameter values on the command line using the following format:

m.ll2u [-rwoz] spheroid=name [zone=value] [input=name] [output=name]

Alternately, the user can simply type:

m.ll2u

on the command line. In this case, the user will be prompted for parameter values and flag settings through the standard interface described in the manual entry for *parser*.

Input can be entered from the keyboard or from an input file. In either case, input should be entered with one longitude and latitude pair per line, in either of the below forms:

```
degrees:minutes:seconds{E|W} degrees:minutes:seconds{N|S}
degrees:minutes:seconds{E|W} degrees:minutes:seconds{N|S}
degrees:minutes:seconds{E|W} degrees:minutes:seconds{N|S}
degrees:minutes:seconds{E|W} degrees:minutes:seconds{N|S}
  .

end

degrees.decimal{E|W} degrees.decimal{N|S}
degrees.decimal{E|W} degrees.decimal{N|S}
degrees.decimal{E|W} degrees.decimal{N|S}
degrees.decimal{E|W} degrees.decimal{N|S}
  .
```

```
.
end
```

If the user sets the *-r* flag, *m.ll2u* will expect the order of the coordinates to be reversed, and stated as latitude, longitude pairs, rather than as longitude, latitude pairs.

Similarly, the user can elect to send output to an output file or (by default) to standard output (the user's terminal screen). If the user sets the *-w* flag, output will be printed in a format suitable for input to programs like *d.points*. Example input and output are shown below (see EXAMPLE).

Program flag settings and parameters have the following meanings.

Flags:

-r The order of coordinates is reversed in the input, and entered as: lat lon

-w Do not flag invalid lon,lat input lines as errors.

-o Flag other invalid input lines as errors.

-z Suppress printing of the UTM zone in the output. (Note. This will produce output in a format suitable for direct input to programs like *d.points*.)

Parameters:

spheroid=name Reference spheroid (ellipsoid).
Options: airy, australian, bessel, clark66, everest, grs80, hayford, international, krasovsky, wgs66, wgs72, wgs84

zone=value UTM zone number (results will be forced into this UTM zone).
Options: 1-60

input=name Name of an existing input file containing longitude, latitude coordinates to be converted. Input lines may either be input in the form of degrees:minutes:seconds, or as decimal degrees. If input as decimal degrees, *m.ll2u* recognizes negative numbers.

output=name Name to be assigned to the output file containing UTM coordinates and zone designations.

AVAILABLE SPHEROIDS
(The on-line listing includes only the spheroid names)

Spheroid:	Semi-major axis (Equatorial Radius) (a):	Eccentricity sqrd (e), Flattening (f), or Polar Radius (b):	Commonly used for:
airy	a=6377563.396	e=.006670540	
australian	a=6378160	f=1/298.25	Australia
bessel	a=6377397.155	e=.006674372	Japan
clark66	a=6378206.4	b=6356583.8	N. America
everest	a=6377276.345	e=.0066378466	India, Burma
grs80	a=6378137	f=1/298.257	
hayford	a=6378388	f=1/297	
international	a=6378388	f=1/297	Europe
krasovsky	a=6378245	f=1/298.3	
wgs66	a=6378145	f=1/298.25	worldwide coverage
wgs72	a=6378135	f=1/298.26	worldwide coverage
wgs84	a=6378137	f=1/298.257223563	worldwide coverage

EXAMPLE
Assume the user has input the command:

m.ll2u spheroid=wgs72 input=ll.infile output=utm.outfile

where the input file ll.infile contains the following longitude, latitude values:

```
        155:30:00W 19:35:00N
        165:30:00W 19:35:00N
        145:30:00W 20:00:00N
        135:30:00W 21:00:00N
        end
```

Output would then be sent to the output file utm.outfile, containing the below Easting and Northing coordinate values and UTM zone designations:

```
        237740.85270818 2167292.1076231 5
        447560.64349407 2165450.19058336 3
        656921.61734802 2212183.40032627 6
        448035.11644906 2322228.3038167 8
        end
```

NOTES
Spheroids, the solids associated with an ellipse, are also known as ellipsoids. They are used as the best possible model to simulate the shape of the earth with its flattened poles and bulging equator. They are the mathematical means for establishing control points to use as a reference when determining exact locations on the earth. A cohesive set of these control points is called a datum.

The spheroids listed above have been used as the basis for a number of different datums. The North American Datum of 1927 (NAD 27) was based on the Clark 1866 ("clark66") spheroid. This was a recent current standard datum for North America. Be aware, however, that a new datum, NAD 83, has been developed using the Geodetic Reference System 1980 spheroid; this is now available in *m.ll2u* as the "grs80" spheroid. The "wgs66", "wgs72", and "wgs84" spheroids are for worldwide use. The "wgs72" spheroid has been used fairly widely by the Department of Defense (DOD) and is the basis for the World Geodetic System 1972 datum. This datum has also been recently replaced; the new DOD datum is WGS 84 (wgs84). Both datums are available within *m.ll2u*.

To use spheroids other than those listed here, the user can add lines to the ellipsoid parameter definition file in $GISBASE/etc/ellipse.table.

Read the marginalia of your source map to determine which spheroid was used to produce the map on which you are working.

This program recognizes negative numbers if coordinates are input in decimal degrees rather than in the form of degrees:minutes:seconds.

This program has received only limited testing. It should be used with some caution.

FILES
See ellipsoid parameter definition file in $GISBASE/etc/ellipse.table.

SEE ALSO
d.labels, d.points, d.sites, d.where, m.datum.shift, m.gc2ll, m.ll2gc, m.u2ll, parser

For australian, clark66, grs80, hayford, international, krasovsky, and wgs72 ellipsoid parameters, see:
John P. Snyder, Map Projections - A Working Manual, U.S. Government Printing Office, Washington DC, 1989. U.S. Geological Survey Professional Paper 1395; from Table 1, p.12.

For bessel, airy, everest, and wgs66 ellipsoid parameter values, see:
Thomas O. Seppelin, The Department of Defense World Geodetic System 1972, presented at the International Symposium on Problems Related to the Redefinition of North American Geodetic Networks, Fredericton, New Brunswick, Canada in May, 1974; see Table 9, p.35.

For wgs84 parameter values, see:
U.S. Naval Oceanographic Labs.

AUTHOR
Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

m.lulc.read

NAME
m.lulc.read **-** Extracts Landuse/Landcover data in the ASCII Composite Theme Grid (CTG) data format distributed by the USGS in to a working file for *m.lulc.USGS*.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
dd [if=tapedev] ibs=input block_size cbs=80 conv=unblock | m.lulc.read arg1

DESCRIPTION
m.lulc.read extracts USGS Land Use/Land Cover data distributed in the ASCII CTG format. Extracted data is placed into a file specified by arg1. *m.lulc.read* should be used prior to using the GRASS program *m.lulc.USGS*.

m.lulc.read reads the data from standard input, allowing the user to pipe in the data from a file.

Data can also be read directly from the 1/2-inch magnetic tape distributed by USGS, by using the UNIX command dd.

Parameters:

if=tapedev	The pathname of the input tape drive (usually /dev/rmt0).
ibs=input	The physical blocking value of the data on tape, usually written on the tape spool. If this value is unknown, run *m.examine.tape* prior to *m.lulc.read*.
cbs	This is the conversion blocking factor for ASCII CTG file and is by default set to 80.
conv	Set conv to unblock when extracting ASCII CTG data.
argl	The name of the output file where the data will be stored in binary format.

EXAMPLES
READING FROM A FILE:
```
        m.lulc.read outfile < infile
```

The above command will read the data from infile and place the results into outfile. Note that infile must be extracted from the USGS CTG 1/2 magnetic tape using the UNIX dd command:

```
        dd if=/dev/rmt0 of=infile ibs=32000 cbs=80 conv=unblock
```

READING DIRECTLY FROM 1/2 INCH MAGNETIC TAPE:

```
        dd if=/dev/rmt0 ibs=32000 cbs=80 conv=unblock | m.lulc.read outfile
```

The above command will extract data with a blocksize of 32000 from /dev/rmt0, and put the results in the file outfile.

SEE ALSO
m.examine.tape, m.lulc.USGS, UNIX manual entry for dd

AUTHOR

Kenneth Shepardson, Spectrum Sciences & Software, Inc

m.lulc.USGS

NAME
m.lulc.USGS **-** Creates raster map layers from a Composite Theme Grid (CTG) file created by *m.lulc.read*. *m.lulc.read* extracts the CTG data from an ASCII landuse/landcover (lulc) CTG format file supplied by the USGS.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.lulc.USGS input_file

DESCRIPTION
m.lulc.USGS creates the following raster map layers from the CTG file created by *m.lulc.read*, and places them into the user's current mapset:

1. Land Use & Land Cover
2. Political Units
3. Census Units
4. Hydrologic Units
5. Federal Land Ownership
6. State Land Ownership

Since the CTG file may not contain all of layers listed above, *m.lulc.USGS* will extract only the raster map layers, which exist in the CTG file.

The CTG file contains all of the geographic region definition information necessary for creating the raster map layers, including:

1. Grid Zone
2. Projection Type (UTM)
3. Cell Resolution (ew_res = nw_res)
4. Multi-byte data formation
5. Geographic region coordinates

m.lulc.USGS will use the geographic region information definition supplied with the CTG file and NOT the geographic region definition currently set for the user's mapset; (note that this is different than is the case with many of the other GRASS map development commands).

When *m.lulc.USGS* is executed, it will check the file for a specified layer and then ask the user if he wishes to extract the associated raster map layer. The user is then prompted for the name of a new raster file in which output is to be placed. *m.lulc.USGS* will then create this raster file and all supporting (e.g., category, cell header, color table, etc.) files.

EXAMPLE
In the example below, raster data is extracted from a CTG file (named lulc) that contains only LAND USE/LAND COVER data:

m.lulc.USGS lulc

```
MAP TITLE: DODGE CITY, KS 1:250,000 LU, PB, CN, HU, FO
The Composite Theme Grid file contains <1> overlays
```

```
        and has a map code type of <01>

        Do you Wish to Create <LAND USE/LAND COVER> Raster File (y/n) [y] y

        Enter File Name for LAND USE/LAND COVER Overlay
        Enter 'list' for a list of existing raster files
        Hit RETURN to cancel request
        > landuse

        Creating <landuse> from <LAND USE/LAND COVER> Overlay: 95%
        Number of Categories: 76  (UNLABELED)
        Writing Cell Header Information
        Writing Color Table Information

        Conversion is Complete
```

BUGS
m.lulc.USGS does not currently extract Census unit data. Also, only the cataloging unit is extracted from the CTG file (see the Land Use and Land Cover Data User's Guide supplied by the USGS).

SEE ALSO
g.region, m.lulc.read
USGS Land Use and Land Cover Data User's Guide

AUTHOR
Kenneth Shepardson, Spectrum Sciences & Software, Inc.

m.proj

NAME
m.proj **-** Calculates conversion coordinates for geographic positions.
(GRASS Map Development Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.proj
m.proj help

DESCRIPTION
This program allows a user to interactively convert coordinates from one projection to another. It allows a user to do all of the following: reading from a file, writing to the screen, or reading from the keyboard, writing to a file, or reading from a file, writing to a file.

Note: The program does not transform GRASS files, it is designed to determine coordinate values on an individual position.

The map projections currently supported are listed in etc/projections file

Each of the above projections (with the exception of State Plane) can be computed with the spheroids listed in etc/ellipse.table file

INPUT FILE FORMAT
When reading from a file of LATITUDE/LONGITUDE data the file will contain two (2) columns of information: the first column **-** latitude **-** in degrees minutes seconds, the second column **-** longitude **-** in degrees minutes seconds,

For example:

```
+40 36 31.4563   -87 2 7.8193
40n 36 31.4563   87w 2 7.8193
```

When reading from a file of PROJECTION COORDINATES data the file will contain two (2) columns of information: the first column **-** easting **-** ground coordinates the second column **-** northing **-** ground coordinates

For example:

```
500000.00    4496918.64
-424489.11   1908736.13
```

Note: NO column headings are required, just the numbers.

AUTHORS
Irina Kosinovsky, U.S. Army CERL
R.L. Glenn, USDA, SCS, NHQ-CGIS
Morten Hulden, morten@tor.ngb. se **-** rewrote and added 121 projections

m.qcalc

NAME
m.qcalc **-** Creates tables, performs conversions, and performs simple math calculations.
(SCS GRASS Data Import/Processing Program)

GRASS VERSION
4.x

SYNOPSIS
m.qcalc
m.qcalc help
m.qcalc [-sahcm] init=value [end=value] [incr=value] [unit=name]

DESCRIPTION
This program

- allows a user to create a table of cell sizes, showing how many square feet, acres, and hectares each cell would represent.

- allows a user to create a table of acreage sizes, showing how many square feet, hectares, and what cell size would be represented.

- allows a user to create a table of hectare sizes, showing how many square feet, acres, and what cell size would represented.

- allows a user to convert values among feet, meters, miles, and kilometers.

- allows a user to use the UNIX bc simple math calculator.

OPTIONS
Flags:

-*s*	Create a cell size table.
-*a*	Create an acres size table.
-*h*	Create a hectares size table.
-*c*	Convert value to user-stated units.
-*m*	Perform a math calculation.

Parameters:

init=value	Initial value.
end=value	Ending value.
incr=value	Increment value.
unit=name	Units of measure.

Options: ft, mt, mi, km

EXAMPLE

The command:

g.qcalc **-***s init=100 end=400 incr=100*

produces the following table:

```
       Cell Size      Sq.Ft.         Acres          Hectares
       100 x 100      107639.31      2.47           1.00
       200 x 200      430557.23      9.88           4.00
       300 x 300      968753.77      22.24          9.00
       400 x 400      1722228.93     39.54          16.00
```

The command:

g.qcalc **-***c init=100 unit=ft*

produces the following:

```
       Feet   Meters  Miles   Kilometers
       100.0  30.4    0.02    0.03
```

AUTHOR
R.L. Glenn, USDA, SCS, NHQ-CGIS

m.region.ll

NAME
m.region.ll **-** Converts Universal Transverse Mercator (UTM) coordinates falling within the current geographic region from UTM coordinates to geographic (latitude/longitude) coordinates.
(GRASS Data Import/Processing)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.region.ll
m.region.ll help
m.region.ll spheroid=name

DESCRIPTION
m.region.ll takes current geographic region settings in UTM coordinates, and converts them to geographic coordinates (i.e., latitudes and longitudes). It also prints the length (in meters) of one arc-second at each of the four edges of the geographic region. The user must enter the spheroid upon which to base the geographic coordinates. The list of spheroids available is somewhat dynamic. It may not contain exactly the ones listed below. To determine the current list of possible spheroids, simply type in:

m.region.ll help

A list of available spheroids will be printed on the screen.

AVAILABLE SPHEROIDS
(The on-line listing includes only the spheroid names.)

Spheroid:	Semi-major axis (Equatorial Radius) (a):	Eccentricity sqrd (e), Flattening (f), or Polar Radius (b):	Commonly used for:
airy	a=6377563.396	e=.006670540	
australian	a=6378160	f=1/298.25	Australia
bessel	a=6377397.155	e=.006674372	Japan
clark66	a=6378206.4	b=6356583.8	N. America
everest	a=6377276.345	e=.0066378466	India, Burma
grs80	a=6378137	f=1/298.257	
hayford	a=6378388	f=1/297	
international	a=6378388	f=1/297	Europe
krasovsky	a=6378245	f=1/298.3	
wgs66	a=6378145	f=1/298.25	worldwide coverage
wgs72	a=6378135	f=1/298.26	worldwide coverage
wgs84	a=6378137	f=1/298.257223563	worldwide coverage

EXAMPLE
m.region.ll spheroid=clark66

Results:
```
       WINDOW  4928000.00N   609000.00E   ZONE 13
               4914000.00S   590000.00W

        44.30.06N    44.29.57N
       103.52.04W   103.37.44W
```

```
 44.22.32N    44.22.23N
103.52.13W   103.37.55W

At northern edge 1 arc-second longitude=22.088500m
At southern edge 1 arc-second longitude=22.135998m
At western edge 1 arc-second latitude=30.860285m
At eastern edge 1 arc-second latitude=30.863082m
```

The values for the geographic coordinates are rounded to the nearest second in this example. They would be more precise in the actual output that is printed on the screen.

SEE ALSO
m.datum.shift, m.ll2u, m.u2ll, r.in.ll

AUTHOR
Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

m.rot90

NAME
m.rot90 **-** Rotates elevation data extracted by either *m.dted.extract* or *m.dmaUSGSread*
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.rot90
m.rot90 help
m.rot90 [-q] input=name output=name rows=value cols=value bpc=value

DESCRIPTION
m.rot90 is used as a companion program to the DTED and DEM digital elevation data tape extraction programs *m.dted.extract* and *m.dmaUSGSread.* *m.rot90* rotates the contents of the output files generated by these tape extraction programs 90 degrees. Rotation is necessary to compensate for the orientation of the data read from tape. This program can be used in conjunction with the program *r.in.ll* to convert this rotated data into GRASS raster map layer form.

m.rot90 requires five inputs to be entered by the user. These parameters and the optional flag setting are described below.

Flags:
-*q* Run quietly, suppressing output of messages on program progress to standard output.

Parameters:
input=name The full pathname of an already-existing file containing the data to be rotated. This input file is usually the output file created by *m.dted.extract* or *m.dmaUSGSread.*

output=name Name to be assigned to the resultant, rotated output file. The full pathname of the output file in which the rotated data are to be stored.

rows=value The number of rows of data in the input file.

cols=value The number of columns of data in the input file.

bpc=value The number of bytes per cell in the input file.

EXAMPLE
The following command:

```
m.rot90  input=/tmp/dma.out  output=/tmp/rot.out rows=301  cols=358  bpc=2
```

will rotate the file /tmp/dma.out, and place the rotated file in /tmp/rot.out. Here, the input file is 301 rows by 358 columns, at 2 bytes per data value.

NOTES
The user should note that since the output file is rotated 90 degrees from the original input file, the rows and columns have been interchanged. Hence, in the above example, the number of rows (301) and columns (358) stated by the user on the command line were those present in the input file. The output file, however, will have 358 rows and 301 columns.

SEE ALSO

g.region, m.dmaUSGSread, m.dted.examine, m.dted.extract, m.examine.tape, m.flip, m.region.ll, r.describe, r.in.ll, r.rescale, r.slope.aspect

Pursuance of Elevation Data by Stuart Bradshaw USACERL DEM and DTED Elevation Extraction, by Stuart Bradshaw, Mary Martin, and Chester Kos, USACERL

AUTHOR

Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

m.sdts.read

NAME
m.sdts.read **-** Program for reading SDTS or other data from files in ISO 8211 (FIPS 123) format.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x

SYNOPSIS
m.sdts.read
m.sdts.read help m.sdts.read [-s] input=name [output=name]

DESCRIPTION
The program enables the user to read SDTS and other data files in ISO 8211 (FIPS 123) format. Output read from the file can be displayed to the screen and/or dumped to a user-specified file.

Data read from the ISO 8211 file is displayed or dumped record by record. The contents of each field of each record are displayed along with field name, subfield name, format, and length. Binary field data are displayed in hex format.

COMMAND LINE OPTIONS
Flags:

-*s*	suppress screen display of data; dump to specified file only.

Parameters:

input=name	name of SDTS ISO 8211 file to be read.
output=name	name of file to store output read from the ISO 8211 file.

NOTES
The original version of this program, *sdtsdump*, is included along with other utility program accompanying the U.S. Geological Survey's distribution of their public domain FIPS 123 Function Library. Both *sdtsdump* and *m.sdts.read* make use of this library. To create *m.sdts.read, sdtsdump* was only modified enough to make it conform to GRASS program conventions.

SEE ALSO
m.sdts.read, v.in.sdts, v.out.sdts, v.sdts.dp.cp, v.sdts.meta.cp, v.sdts.meta

AUTHORS
Robert Lazar, U.S. Geological Survey, National Mapping Division
David Stigberg, U.S. Army Construction Engineering Research Laboratory

m.setproj

NAME
m.setproj **-** Allows the user to create the PROJ_INFO and the PROJ_UNITS files to record the projection information associated with a specified mapset.
(SCS GRASS Data Import/Processing Program)

GRASS VERSION
4.x

SYNOPSIS
m.setproj
m.setproj help
m.setproj [set=name] proj=name

DESCRIPTION
Note: This module has been replaced by *g.setproj*.

Allows a user to create a new PROJ_INFO file in the specified mapset. The file is used to record the projection information associated with the specified mapset.

Note:

The "set" mapset must exist and must not contain a PROJ_INFO or PROJ_UNITS file. The specification of any projection other than ll and xxx will generate a request to the user for a name of a standard ellipse. The projections of aea, lcc, merc, and tmerc will generate a request to the user for the prime meridian and standard parallel for the output map. The projection of stp will generate a request to the user for the state abbreviation and choice of zone for the output map. The projection of xxx will request mapgen proj parameters. See the mapgen proj documentation for the uses of this software.

COMMAND LINE OPTIONS
Parameters:

set=name	Mapset in which the projection information file is to be stored.
proj=name	Map projection name.

Options: utm, aea, stp, ll, lcc, merc, tmerc, xxx

EXAMPLE
To create a PROJ_INFO file recording mapset SAMPLE as being a UTM projection in zone13:

m.setproj set=SAMPLE proj=utm

The user will be prompted for the spheroid and zone of the UTM projection.

SEE ALSO
Mapgen proj

AUTHOR
M.L. Holko, USDA, SCS, NHQ-CGIS

m.stp.proj

NAME
m.stp.proj **-** Finds a State Plane projection for the user.
(SCS GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.stp.proj
m.stp.proj help

DESCRIPTION
m.stp.proj prompts the user for a state FIPS code or its two-letter abbreviation, and then for the name of a county located within this state. The program then prints the MAPGEN style parameters for the State Plane Projection.

EXAMPLE
m.stp.proj

```
Enter State FIPS code or press RETURN if unknown:
Enter Two-Letter State or Territory Abbreviation:  pa
Enter County Name:  cambria
+proj=lcc +a=0.63782064e+07 +es=0.6768657997291094e-02
+x_0=0.6096012192024384e+06 +y_0=0 +lon_0=77d45'w
+lat_0=39d20'n +lat_1=40d58'n +lat_2=39d56'n
```

AUTHOR
M.L. Holko, USDA, SCS, NHQ-CGIS

m.svfit

NAME
m.svfit **-** Semivariogram model fitting.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x

SYNOPSIS
m.svfit
m.svfit help
m.svfit [-pqw] [input=name] model=value range=value [graph=name]

DESCRIPTION
m.svfit calculates a sample semivariogram and either plots it in the GRASS graphics window or writes the estimated parameters to standard output, or both.

For more information, refer to the tutorial or see the example below.

OPTIONS
Flags:

-*q*	Quiet. Cut out the chatter.
-*p*	Plot model and sample semivariogram (requires *g.gnuplot*).
-*w*	Use weighted least squares (default is general least squares)

Parameters:

sites=name UNIX file containing sample semivariogram (see NOTES). Default is standard input.

model=value Integer Model Index, one of
1. Linear,
2. Spherical,
3. Exponential,
4. Gaussian,
5. Quadratic, or
6. Wave or Hole Effect.

range=value Range of semivariogram.

graph=name Basename to save graphing data/commands files. Graphs are saved in the current working directory with the extensions .gp and .dat. Implies the -*p* flag.

NOTES
Three columns of data are expected as input: lag distance (h), semivariogram value (gamma), and the number of data pairs used to compute it (N(h)). This may either be from a UNIX file, entered from the command line (terminated by control-d), or via a pipe or redirection.

EXAMPLE
m.svfit was designed to be used in conjunction with *s.sv*, a GRASS sites program for calculating sample semivariograms. The following example calculates a sample semivariogram of the sites list wells with a nominal lag distance of 5 and then fits a linear model with a range of 100. The sample semivariogram and

model are plotted in the GRASS graphics monitor and the graphing instructions and data are saved to files with the basename svwells in the current working directory:

```
s.sv -q wells lag=5 | m.svfit -p m=1 r=100 g=svwells
```

By saving the graphing instructions and data, the semivariogram may be plotted again later by the following command:

```
g.gnuplot svwells.gp
```

SEE ALSO
s.univar, s.normal, g.gnuplot, s.sv

Semivariogram Modeling **-** A GRASS Tutorial on Exploratory Data Analysis and Semivariogram Modeling.

BUGS
Please send all bug fixes and comments to the author.

AUTHOR
James Darrell McCauley, Agricultural Engineering, Purdue University

m.tiger.region

NAME
m.tiger.region **-** Finds geographic region information for U.S. Census Bureau TIGER input data.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.tiger.region
m.tiger.region help
m.tiger.region infile=name [zone=value] [spheroid=name]

DESCRIPTION
m.tiger.region is a program designed to evaluate a file of raw type 1 Census (TIGER) data and determine the geographic region covered by that input file. Output is sent to standard out, and gives the east, west, north, and south boundaries for the given input data file.

If the user specifies the input file name and (optionally) the zone number or spheroid to be used on the command line, the program will run non-interactively; if no zone number or spheroid name is given, the default(s) will be used (see below). Alternately, the user may simply type *m.tiger.region* on the command line; in this case, the program will prompt the user for parameter values using the standard GRASS *parser* interface described in the manual entry for *parser*.

OPTIONS
Parameters:
infile=name Input file name in which raw TIGER data (type 1) are stored.

zone=value Universal Transverse Mercator (UTM) zone for this county.
  Options: **-**60 **-** 60
  Default: 0

spheroid=name Name of spheroid to be used.
  Default: clark66

Available spheroids are:

australian
bessel
clark66
clark80
everest
international
wgs72
wgs84

It is recommended that the user choose the clark66 (default) spheroid when dealing with TIGER data as it is the most consistent with the original data.

EXAMPLES
If the user typed simply:

m.tiger.region infile=inputfilename

program output would look similar to this:

```
Number of calculated zones is: 2

INFO FOR ZONE 1:
zone number: 13
percentage of data points
in this zone: 0.799489
regional spread of points
within this zone:
north:  5092049.155918
south:  5049238.983803
east:    734139.517650
west:    732514.747908

INFO FOR ZONE 2:
zone number: 14
percentage of data points
in this zone:99.200508
regional spread of points
within this zone:
north:  5092041.463966
south:  5036134.342322
east:    398030.217441
west:    265527.656108
```

If the user does not input the UTM zone number, it is calculated for them. Then the zone number and region information are output, and if the program finds that the input data contains information in more than one UTM zone, then the output is given for all applicable zones.

If instead the user supplies the UTM zone number, the output would look like that shown below:

```
REGION FOR THIS DATA FILE:

north border: 5092049.155918
south border: 5036134.342322
east border:   398030.217441
west border:   265527.656108
(zone number: 14)
```

NOTES
This command must be compiled separately. It will not automatically be included in the compile of the main GRASS code. Although *m.tiger.region* does not need a FORTRAN compiler, it is used to support other TIGER data functions (like *v.db.rim*, *v.in.tiger*, and *rim*) which do require access to a FORTRAN compiler.

TIGER data are presented in latitude/longitude format, and are converted to UTM coordinates using coordinate conversion routines contained in the GRASS library. If no UTM zone number is supplied by the user, the program calculates the appropriate zone(s) based on the input data provided. The output then provides the UTM zone numbers found (if more than one), the geographic region covered within each zone, and the percentage of data points found in each zone. The user must then decide which of these UTM zones contains the major or most important portion of data values, so that the zone number can be supplied in creating the GRASS location to hold the imported data and can be provided to the importing program (*v.in.tiger*). Zone edges will be extended (reasonably) to include data values lying outside the chosen zone. If desired, *m.tiger.region* can be re-run, supplying the chosen zone number, in order to evaluate the region edges of the input data set (with the extended zone).

FILES
Source code for RIM is located under $GISBASE/../src.related/rim

The RIM Users manual.
The RIM reference manual.

Source code for *v.db.rim* is located under $GISBASE/../src.garden/grass.rim/v.db.rim

Source code for *v.in.tiger* is located under $GISBASE/../src.garden/grass.tiger/v.in.tiger

Source code for *m.tiger.region* is located under $GISBASE/../src.garden/grass.tiger/m.tiger.region

SEE ALSO
Gen.Maps, Gen.tractmap, g.region, v.db.rim, v.in.tiger, tiger.info.sh

AUTHOR
Marjorie Larson, U.S. Army Construction Engineering Research Laboratory

m.u2ll

NAME
m.u2ll **-** Converts Universal Transverse Mercator (UTM) coordinates to geographic (latitude/longitude) coordinates.
(GRASS Data Import/Processing Program)

GRASS VERSION
4.x, 5.x

SYNOPSIS
m.u2ll
m.u2ll help
m.u2ll [-srwod] spheroid=name [zone=value] [input=name] [output=name]

DESCRIPTION
m.u2ll converts Universal Transverse Mercator (UTM) northings and eastings to geographic coordinates (i.e., latitudes and longitudes). The user must specify the UTM coordinates to be converted and the spheroid on which the geographic coordinates will be based. The program also needs to know the UTM zone in which the input coordinates are located. However, if the user is running GRASS from a UTM database LOCATION, *m.u2ll* will use this database's UTM zone designation, if no zone is specified by the user.

The GRASS program *m.ll2u* performs the reverse operation, converting geographic coordinates to UTM coordinates.

The list of spheroids available is somewhat dynamic. At the time of this release, available spheroids included: airy, australian, bessel, clark66, everest, grs80, hayford, international, krasovsky, wgs66, wgs72, and wgs84 (see table below).

This command can be run either non-interactively or interactively. The user can run the program non-interactively by entering desired flag settings and parameter values on the command line using the following format:

m.u2ll [-srwod] spheroid=name [zone=value] [input=name] [output=name]

Alternately, the user can simply type:

m.u2ll

on the command line. In this case, the user will be prompted for parameter values and flag settings through the standard interface described in the manual entry for *parser*.

Input can be entered from the keyboard or from an input file. In either case, input should be entered with one UTM easting and northing pair per line, in the format shown below:

```
easting northing
easting northing
easting northing
easting northing
  .
  .
end
```

If the user sets the -*r* flag, *m.u2ll* will expect the order of the coordinates to be reversed, and stated as northing, easting pairs, rather than as easting, northing pairs. This is useful for passing ASCII GRASS vector (/dig) files, whose coordinates are stated as northing, easting pairs, directly through *m.u2ll*.

Similarly, the user can elect to send output to an output file or (by default) to standard output (the user's terminal screen). Example input and output are shown below (see EXAMPLE).

Flags:
-*s* Specified UTM zone is in the southern hemisphere.

-*r* The order of coordinates is reversed in the input, and entered as: north east. This option allows the user to pass an ascii vector file through *m.u2ll*.

-*w* Do not flag invalid east, north input lines as errors.

-*o* Flag other invalid input lines as errors.

-*d* Output latitude/longitude values in decimal degrees, rather than in the form dd:mm:ss.

Parameters:
spheroid=name Reference spheroid (ellipsoid).
Options: airy, australian, bessel, clark66, everest, grs80, hayford, international, krasovsky, wgs66, wgs72, wgs84

zone=value UTM zone in which UTM coordinates are located.
Options: 1-60

input=name Name of input file containing UTM values to be converted.

output=name Name to be assigned to output file containing longitude and latitude values.

AVAILABLE SPHEROIDS
(The on-line listing includes only the spheroid names.)

Spheroid:	Semi-major axis (Equatorial Radius) (a):	Eccentricity sqrd (e), Flattening (f), or Polar Radius (b):	Commonly used for:
airy	a=6377563.396	e=.006670540	
australian	a=6378160	f=1/298.25	Australia
bessel	a=6377397.155	e=.006674372	Japan
clark66	a=6378206.4	b=6356583.8	N. America
everest	a=6377276.345	e=.0066378466	India, Burma
grs80	a=6378137	f=1/298.257	
hayford	a=6378388	f=1/297	
international	a=6378388	f=1/297	Europe
krasovsky	a=6378245	f=1/298.3	
wgs66	a=6378145	f=1/298.25	worldwide coverage
wgs72	a=6378135	f=1/298.26	worldwide coverage
wgs84	a=6378137	f=1/298.257223563	worldwide coverage

EXAMPLE
Assume the user has input the command:

m.u2ll -s spheroid=wgs72 zone=4 input=utm.infile output=ll.outfile

where the input file utm.infile contains the following easting and northing UTM coordinate values and zone designations:

```
        237740.85 2167292.10
        238740.00 2167000.00
        239000.00 2167100.00
        237100.00 2166000.00
        end
```

Output would then be sent to the output file ll.outfile, containing the below longitude and latitude coordinate values:

```
        166:02:25.645137W 70:27:46.615528S
        166:00:53.237056W 70:27:59.692673S
        166:00:27.23258W 70:27:57.454281S
        166:03:41.428895W 70:28:25.61617S
        end
```

NOTES

Users can add information to the ellipsoid parameter definition file on their systems (located in $GISBASE/etc/ellipse.table) to add spheroids not now among those supported by GRASS.

See *m.ll2u* for a brief discussion of spheroids.

The UTM zone designation determines on what area of the earth a point is found. The same UTM coordinates will be found in each different UTM zone. Look at the marginalia of your source map to determine into which UTM zone your UTM coordinates fall. Although the user can permissibly omit specification of a UTM zone when running this program under a UTM data base LOCATION, it is safer to specify it (see **DESCRIPTION**, above).

m.u2ll converts the first pair of coordinates on each line of input and leaves anything else on the line alone. If a line begins:

xxxxxx.xx xxxxxxx.xx

then the xxxxxx.xx xxxxxxx.xx UTM coordinate pair is converted to a longitude, latitude pair. Any other information appearing on the line is left alone. If the line doesn't begin with a pair of coordinates in the above format, then the line is left as it is.

See ellipsoid parameter definition file in $GISBASE/etc/ellipse.table.

SEE ALSO

d.labels, d.points, d.sites, d.where, m.datum.shift, m.gc2ll, m.ll2gc, m.ll2u, parser

For australian, clark66, grs80, hayford, international, krasovsky, and wgs72 ellipsoid parameters see:
P. Snyder, Map Projections - A Working Manual U.S. Government Printing Office, Washington DC, 1989. U.S. Geological Survey Professional Paper 1395, from table 1, p.12.

For bessel, airy, everest, and wgs66 ellipsoid parameter values, see:
Thomas O. Seppelin, The Department of Defense World Geodetic System 1972, presented at the International Symposium on Problems Related to the Redefinition of North American Geodetic Networks, Fredericton, New Brunswick, Canada in May, 1974; see Table 9, p.35.

For wgs84 parameter values, see: U.S. Naval Oceanographic Labs.

AUTHOR
Michael Shapiro, U.S. Army Construction Engineering Research Laboratory