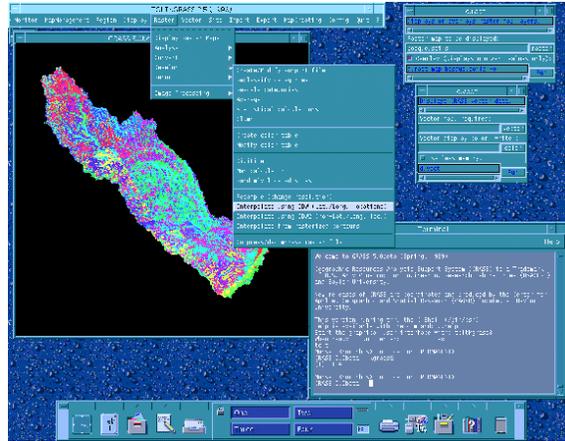
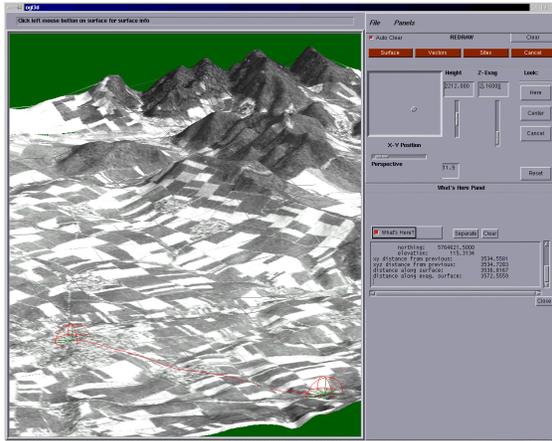


GRASS Reference Manual

Sites 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/>

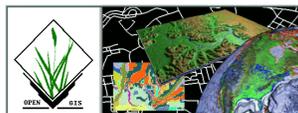


Table of Contents

Topic	Page
GRASS Introduction	2
s.buffer	4
s.db.rim	5
s.geom.....	19
s.in.ascii.....	21
s.in.grid	23
s.info	24
s.kcv.....	27
s.label	28
s.medp.....	30
s.menu.....	32
s.normal	36
s.out.ascii	38
s.perturb.....	40
s.probplt	41
s.qcount	43
s.rand	45
s.sample	46
s.surf.idw.....	48
s.surf.krig	50
s.surf.rst	51
s.surf.tps.....	56
s.sv.....	61
s.to.rast	63
s.to.vect	65
s.univar	66

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 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

Please allow us to extend our most cordial thanks to all of you. If you contributed to GRASS at nce, let us know your name and e-comprehensive on-

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

P.O. Box 97351
Waco, Texas, U.S.A. 76798-

s.buffer

NAME

s.buffer generates vector buffer around a site point. The area is labeled with site category.
(S Sites Program)

GRASS VERSION

s.buffer

s.buffer help

DESCRIPTION

generates vector buffer around a site point. The area is labeled with site category. If the value is not given or its 0, then the radius is taken from the attribute of the site point.

OPTIONS

Parameters:

site_list file to read

output vector (digit) file to create

radius

Values for radius:

0=take category for radius,

1=the 4th value of site list (category label),

positive value= use given value for radius. Then the buffer is the same with all points.

SEE ALSO

Janne Soimasuo, Finland

s.db.rim

NAME

s.db.rim - RIM data base management/query interface for GRASS sites data.
(GRASS Sites Management Program)

GRASS VERSION

4.x

SYNOPSIS

s.db.rim

s.db.rim data_base

DESCRIPTION

For some information on LINUX and RIM look at the BUGS section

s.db.rim allows users to create, manage and query information about site locations (sites) across the landscape. Required inputs can be entered interactively, or from the command line. Command line input may be entered through a prepared text file or from the keyboard (standard input). The *s.db.rim* command language is defined in SECTION ONE below. The menu-driven interactive version is described in SECTION TWO of these manual pages.

These programs are actually a marriage of the GRASS environment and the programmer's interface library of the relational data base management program RIM, distributed publicly by the University of Washington Academic Computing Services as FORTRAN 77 code. Your system must have a FORTRAN 77 compiler to use *s.db.rim*.

SECTION ONE -- THE COMMAND VERSION

The command-line driven version of *s.db.rim* is run by typing the below command, where *data_base* is the name of an existing RIM sites data base:

```
s.db.rim data_base
```

The sites data bases are stored in a subdirectory named 'rim/sites' in the user's current mapset. Data bases in other mapsets, selectable through the GRASS *g.mapsets* command, can be accessed for 'read-only' retrieval of records. Each mapset may have many data bases. Each data base within a mapset must have a different name; user-supplied names for data bases are limited to seven (7) characters in order to maintain compatibility with the standard version of RIM. As with other GRASS commands, mapsets are searched in the mapset SEARCH_PATH order when a data base needs to be opened.

Each site data base is composed of multi-field records (rows or tuples, in DBMS jargon). Each field and its position in the site form is defined via input to the *.make* command when a data base is originally defined. It is possible to add new fields or change the length of existing fields after data has been loaded, however this is not straightforward and is not described here; deleting of fields is also possible, but requires even more experience and knowledge. The user needs to carefully design the data base fields and form (layout) and check the operation with a few pieces of test data before loading data for a large number of sites.

COMMANDS

(Note: For each of the "dot commands", i.e., *.make*, described below there is a menu choice to select when running the interactive version. The interactive menus are described in the SECTION TWO of this document. Some display capabilities exist in the interactive version, which are not directly implemented in the command version.)

The commands are given alphabetically here for easy reference. The `.make` command is required to create a data base shown in () are accepted; this is primarily for those giving *s.db.rim* abbreviations may also be used in batch files.

Each command is introduced with an input record (line), which starts with a period and is followed by one of the words shown below; for some commands the command line also contains one or more required or options/data for a command are supplied on successive lines; a `.end` line is needed by some commands to signify the end of these input lines.

Alphabetical Command Summary

"!command"

This is the only *s.db.rim* which is executed by a "G_system()" call (see GRASS gis library). Many UNIX utilities (e.g., vi, ls, print) and most GRASS commands (e.g., *d.points g.list*, *d.zoom r.mask*) may be executed. It is often useful, to change "region" and "MASK" within *s.db.rim* be separated by ";" in the standard UNIX way. Note that a "!cd directory; ls" will change to the specified directory and list files, but the effective working directory for `!cd` will not be changed when the command terminates.

".add (.a)"

Add a new site record (row) to the open data base. Each line following contains a field name followed by spaces and/or tabs then the value or character string to store for that field with `.end`. Some fields may be absent and fields may appear in any order. Checks are made for the input of data for the one required field (site number), for string length for string type fields, and for duplicate fields to be added as a separate line. If there are any problems, the record will not be stored and a message will be output. This format makes it relative to the current mapset. -a"

```
.add
north      4690673.30
east       601410.00
.end
```

".backup (.b) file_name"

The `.backup` command is used to dump the entire data base from the RIM binary files to a text file format (see UNLOAD in the RIM User's Manual). The `file_name` can be a relative path name or full path name. This text file is portable to RIM or running on any other computer; it may also be reloaded to recreate the *s.db.rim* only be done on data bases in the user's current mapset.

To reload your data base from the backup file (normally not necessary):

```
GRASS 4.1> cd $LOCATION/rim/sites #right directory
GRASS 4.1> rm db_name.rimdb2 #remove data base (or mv to somewhere)
GRASS 4.1> rim #run RIM manually
RIM> input "path/file" #RIM rebuilds data base from data
```

RIM> exit

".change (.c) [-l]"

Without the "-l" flag, each line following .change is in the same format as for the .add command. The site number field is required and the site number must match an existing site in the data base. Only those fields for which lines are provided are changed in the record. After the .end the changed record is stored, if all is ok, otherwise a message is output.

If the "-l" flag (for "list") is given, the site number field is omitted and the specified field values are changed for all sites currently selected by .find and/or .query.

".delete (.d)"

This command is used to delete data records for sites. Deletion of sites is permanent. A backup of the data base, or copies of the data base files, are the ways to protect your valuable data.

The lines following the .delete command should contain only the site numbers, with a .end line being last.

The following command sequence will delete all the sites currently on the internal site list (the result of the last .query or .find command) after asking for approval.

```
.delete
.end
```

".end (.e)"

Ends multi-line input for several other commands.

".exit (.ex)"

Use .exit to end operation of *s.db.rim* cleanly. In general, do not use CTRL-C to exit unless absolutely necessary. When .exit is encountered in a batch file, input will revert back to the previous file, or the terminal, if any, which called the batch file.

".find (.f) [-a | -d] [-m] [-r]"

The .find command is used to find the site(s) closest to a given point (the target). The target can be defined in one of several ways. The found sites are stored on an internal sites list for output by other commands; however, see note 2, below. The found sites are stored on the internal sites list in order of proximity to the target location.

The optional .find command line parameter specifies the current MASK (-m), if any, or the current region (-r), as a filter on the retrieved sites. -m automatically implies -r, as the MASK is not defined outside the current region. If the -a flag is given, the retrieved sites will be appended to those previously retrieved with a .query or .find; duplicates will be automatically discarded. The -d flag causes the retrieved sites to be deleted from the internal site list, if present there. Very complex selections can be done by interspersing appends and deletes to arrive at a final list of sites. For instance, selecting those sites within 2000 meters of a target and then deleting those within 1500 meters of the target will give a final list of those from 1500 to 2000 meters.

The single required line following the .find line gives the program the necessary target information. The following examples show the possibilities.

```
find> 602793.90 4379010.00
```

will find the one site nearest these coordinates and store it on the internal site list.

```
find> 619840 4599000 10
```

will find the 10 sites (or fewer, if there are not that many) closest to the given location.

```
find> site 132 10
```

will find the 10 sites closest to the location of site 132 in the data base (including site 132). If site 132

-Long coordinates) of the target location.

```
find> distance from site 16 2500
```

will find all sites greater than 2500 (meters) from the location of site 16.

from target).

2. The number of sites found is automatically printed to the active output device/file.
3. If mask is specified, the effective region is automatically set to the current region (because the GRASS mask is only defined for the current region).
4. Region and mask filtering uses the current region and mask in the masking map.
5. In the last two examples the string "distance from" must be exactly matched. Also, the word "site" must be exactly matched.
6. If the "distance from" radius is given as a negative value, points outside the target circle are selected;

y

be set or removed with a variety of GRASS commands.

8. The "find>" prompt is given only when input is from a terminal.

".help (.h)"

Prints a help screen to the output device or file. Useful to have when using *s.db.rim* when writing a script file of commands.

".input (.i) [file]"

until a .exit command is found. Input files may call other input files to a nesting depth of eight. Without

Lists the available data bases in the current mapset search path.

".make"

Using the .make command you create a new data base in the current mapset by specifying the following items, which define the screen (page) layout for displaying and printing the site records, as well as the information fields:

- 1) The fixed text part of the screen layout.
- 2) The positions, types, names and lengths of data fields.

Three fields must always exist in a data base; each of these field types may only occur once in a data base layout:

- 1) Type 's' Site identification number field (an integer).
- 2) Type 'x' Easting coordinate of the site (a double float).
- 3) Type 'y' Northing coordinate of the site (a double float).

The other field types, which may occur in any combination and order, are:

- 4) type 'i' An integer field.
- 5) type 'f' A double precision float field.
- 6) type 't' A text field.

Each of the fields can be positioned anywhere within the screen layout, which has a limit of 19 lines by 80 columns. A maximum of 70 fields may be defined within this space. A field is specified in the screen layout by a tilde (~), a field type character, a field name and enough trailing tildes to fill out the desired field length.

Each line following the .make command is taken to define a line of the screen layout until a .end is reached. If a mistake is made on any of the input lines, the .make will fail. The .make information may be prepared in advance as a text file (this facilitates fixing mistakes) and the .input command can be used to read in this file. An example text file for a data base screen layout follows, with some important explanatory notes.

```
.make
      Archaeological Sites Database
      =====

Site #: ~sSite~~~      Entered By: ~tEnter_by~~~~~
Description:          C-14 Date: ~iAge~~~
  ~tDescript.1~~~~~
  ~tDescript.2~~~~~
  ~tDescript.3~~~~~
Type: ~tType~~~~~ (Should be Arch. or Hist.)
Date: ~tEnter_Date~~~~~      Square Miles: ~fArea.4~~~
North: ~yNorth~~~~~      East: ~xEast.1~~~~~
.end
```

Notes:

- 1) Any text not preceded by a tilde (~) character is taken to be part of the constant or fixed text portion of the form.
- 2) A field definition begins with a tilde (~) character immediately followed by a single character which indicates the data type of the field (s, x, y, i, f, or t). Immediately following the data type character is the

field name of 1 to 16 characters. Field names can be composed of any characters from the following set: [A-Z,a-z,_,0-9]; the RIM program and library do not distinguish upper and lower case in field names, so you should avoid making names which differ only in case. Field names may not begin with a numeral [0-9]. The rest of the field length is padded with tilde (~) characters to the required maximum length.

3) The minimum field width is three characters; e.g., "~tA". Be sure field widths for all fields are wide enough for the values and strings you expect to store there; e.g., UTM northings require at least 11 spaces.

4) For text fields it is possible to continue a field across more than one line. This is done by appending a .1 to the field name forming first portion of this "split field", a .2 for the second portion, etc. This text field splitting affects how information is organized for input and output; the composite text string is concatenated (unused portions of fields are retained as spaces) and treated as a unit for storage and queries to the data base.

5) For the double precision floating point fields (types x, y and f), the number of decimal places to print may be specified by appending ".n" to the field name, where n is the number of decimals places required. Values of n from 1 to 12 may be used. Two decimal places is the default if ".n" is not specified. Be certain that you make the field wide enough to print the integer and decimal portions of the values that will be stored in the field; include space for a sign and the decimal point. (If it is desired to print zero decimals, whole numbers, use an "i" type field.) In the example above, the northing coordinate (y) would be output with two decimals, the easting (x) would have only one decimal place, and the "Area" would be printed with four decimal places.

".output (.o) [file or | process]"

Causes all output (except some error messages) from *s.db.rim*, including that from the *.print* command, to go to the named path/file (may be a full or relative path name), or to be used as standard input by the process (a pipe). If no parameter is given, output returns to stdout, usually the user's terminal. An example of the pipe usage would be

```
.output | grep "easting" | wc -l > /tmp/my_count
```

A pipe is closed whenever the *.output* command is given again, or on a *.exit* command.

".pack (.pa)"

This should be used when numerous data records have been deleted to recover disk space in the RIM binary data base files. It works by doing a *.backup* to a temporary file; moving the data base files to new names (*.bakdb*); running RIM to rebuild the data base; and, if the rebuilt data base can be opened and read, the temporary files are deleted. The user is informed if this process fails. Packing can only be done on an open data base located in the user's current mapset.

".print (.p) [-a | -l] "

This command outputs the full site records for the sites currently stored on the internal sites list (result of last *.query* or *.find*). Without the flag, the screen layout format is used. With the *-l* flag, for list format, the field name followed by the contents are output one field per line. The *-a* flag also outputs in the list format but with a *.add* line and a *.end* line surrounding each site record printed; data files in this form can be read with *.input*, thus they form one kind of backup mechanism and can be used to transfer data (not the data base layout) from one GRASS system to another. The destination for the output is set by a previous *.output* command (default is stdout).

".query (.q) [-a | -d] [-m] [-r]"

The *.query* command is used to retrieve sites via an SQL-like request to RIM, including a user specified "where clause." All fields for each site meeting the selection criteria are retrieved.

The optional .query command line parameters cause points not in the region (-r) and/or mask (-m) to be rejected, so these conditions need not be tested in the "where clause." The -a flag causes the retrieved sites to be appended to those previously retrieved by .query or .find; duplicate entries are automatically discarded. The -d flag causes selected sites to be deleted from the current list, if present.

After the query command line, any number of lines may be entered to define the SQL "where" clause. A .end line is required to finish the request and begin data retrieval. See examples below.

The "distance from" clause may also be used as additional selection criteria exactly as described in the examples and notes for .find. It must be entered as a separate line to the query prompt.

The retrieved records may be printed at time of retrieval, rather than after the completion of the query command by including a .print (.p) line with the same options for print format as in the .print command (see above); e.g. .p -a to output in the "list add" format. The .print clause must be entered as a separate line to the query prompt. This feature is most useful when working with very large data bases where retrieval time is significant. See example 2 below.

Example 1

```
query> where density < 20 and (date = "10/14/89"  
query> or county eq "San Marcos")  
query> .end
```

Example 2

```
query> where east <600000 and name like "**Jones*"  
query> distance from site 12 3000  
query> .print -a  
query> .end
```

Example 3

```
query>.end
```

The where and distance from clauses are each optional. If both are omitted, only the mask and region on the .query command line restrict the search; if mask and region are also omitted, all sites will be retrieved (Example 3). When querying for sites the where clause is processed first, the current region and mask tested next (if requested), then the distance from clause is applied; a site must pass all tests to be put on the internal site list for output by other commands.

Notes: (Also see Notes for .find)

1. The retrieved sites are stored on the internal site list in the order returned from the data base by RIM, not necessarily in site number order or the order the data was loaded. A "distance from" clause results in a final sorting by proximity to target.
2. See the RIM User's Manual for additional information on the "where" clause in the "select" command, especially the quotes required for matching character string fields, and the allowed comparison operators.
3. In the where clauses of the examples, "density", "date", "county", "east", and "name" are field names (column names in RIM) defined when the user initially makes the data base.
4. Each .query or .find resets the internal site list (even unsuccessful ones), unless the -a or -d flags are used.

".read_site (.re) site_list [comment_field]"

This command reads an existing GRASS site list and creates a data base record for each site. If the comment or description field of all entries in the site list begin with # and a number, the number becomes the site number in the data base. If some of the sites in the GRASS site list do not have a # at the beginning of the comment field, the sites are numbered sequentially starting with 1. (These options are similar to the way the GRASS sites-to-raster [in *s.menu*] works.)

if a data base field name "comment_field" is entered on the command line, the comment will be stored in that field for each site. If an integer or float field is specified, and attempt is made to interpret the comment as that type of number; if this interpretation fails, 0 or 0.0 is stored.

If the site number duplicates one already in the data base or found earlier in the site list, it is not added.

Once the sites have been loaded by *.read_site*, use *.change* (or the interactive version) to add data to other fields for those sites.

".remove"

This command, which requires a "y" as confirmation on the next line, entirely removes the three binary files, which constitute your RIM data base. Use with care. Backup files must be removed individually by the user, if desired, from the \$LOCATION/rim/sites directory.

".show (.sh)"

This command is used to output the screen or page layout as defined for the current data base. It serves as documentation of the data base definition and as a reminder for field names, types and lengths. By using an editor to surround the output of *.show* with *.make* and *.end* lines, it can be used to reload the data base definition with *.input*.

".site_list (.si) file_name [field_name]"

This command writes the site locations and the site numbers to the specified file in the *site_list* directory in the current mapset. If the file exists, the sites are appended to the current list, otherwise, a new site list file is created. A "field_name" may be optionally specified; if so, the contents of that field (retrieved from the appropriate site record) are inserted as the comment (following a '#') in the site list. The site number is used if no field name is supplied.

A comment line is inserted in the *site_list* file with the current date and time and the name of the data base producing the site locations. The format used for each site is:

```
eastings|northings|#comment
```

".tables (.t)"

Prints the table structure of the currently opened RIM data base. This is the same output generated by a "list *" command when running RIM manually. The information for the table named "data" is useful for review of the user's field definitions. The information for the two other tables is for internal use by *s.db.rim*.

SECTION TWO -- THE INTERACTIVE VERSION MENUS AND COMMENTS

SYNOPSIS

s.db.rim

DESCRIPTION

The interactive version of *s.db.rim* allows you to create, manage and query information about site locations (sites) across the landscape. Operations are done on a data base through a series of menus explained below. Most of the menus use VASK screens; the user should become familiar with keys that move the cursor among the fields to be entered (RETURN, ENTER, CTRL-L, CTRL-K, etc.).

THE MAIN MENU

Below is the main menu.

Option 1 is the default. Note the status line at the top of the menu, and the fact that 8 records have been selected by the latest find or query operation (between items 2 and 3). Note, also, that CTRL-C can be used to exit from this menu (and most other menus in the program) back to the GRASS prompt. The specifics of each menu choice are described below.

```
s.db.rim                MAIN MENU                Version 1.4
  Data base <water> in mapset <rono> open.  25 records.

  1  Open a data base
  2  List available data bases
----- Retrieve/Output Site Records (8 currently) --
  3  Find sites in proximity to a Target point
  4  Query to select site records (SQL)
  5  Show selected site records on Terminal
  6  Display maps/selected sites on graphics terminal
  7  Output selected site records to Printer or File
  8  Create a site_list from selected records
----- Add/Edit Site Records -----
  9  View a single site record
 10  Add a site record
 11  Change a site record
 12  Delete a single record or all selected records
----- Other functions -- Shell Command -- Exit -----
 13  Make a new data base & Management Functions
 14  Execute a shell command

  0  Done -- Exit from s.db.rim

AFTER COMPLETING ALL ANSWERS, HIT <ESC> TO CONTINUE
      (OR <Ctrl-C> TO EXIT THIS PROGRAM)
```

1. Open a data base. If a data base is already open, it is closed before the requested one is opened. Only data bases in the user's current mapset may be modified; others are opened in read-only mode; this will be indicated on line 2.

2. List available data bases. For each mapset in the current GRASS mapset search path, the names of the existing data bases are listed.

3. "Find" sites in the data base relative to a specified target location. This is used to select sites based on proximity to the target and, optionally, sites within the current region and, optionally, sites falling in active cells within the current GRASS mask. Two modes of targeting are provided: the N sites closest to the target, and all sites within (or outside) a circle of specified radius from the target. The FIND/QUERY TARGET MENU discussed below accepts region/mask/target specifications from the user. The selected sites are then displayed one at a time until CTRL-C is entered; then other operations, choices 5-8, can be done with these sites. The line on the menu between 2 and 3 shows the number of sites currently selected by choices 3 or 4.

4. "Query" sites in the data base using an SQL-like "where clause," including specifications for region/mask/target (circle only) as in 3, above; see FIND/QUERY TARGET MENU section below. The where clause can test for ranges or matches for numeric data base fields, or matches on full strings or

substrings for text fields. The selected sites are then displayed one at a time until CTRL-C is entered; then other operations, choices 5-8, can be done with these sites. This clause is entered on a menu described below; see QUERY COMMAND MENU section, below.

The where clause may use parentheses () to control the order of comparisons. Field names are not case sensitive within where clauses. The following comparison operators are valid for all types of fields:

```
eq  or  =          ne  or  <>
ge  or  >=        le  or  <=
gt  or  >         lt  or  <
```

String comparisons are case sensitive and are done character by character. Substrings comparisons may be done with the "like" operator as in:

```
where name like "*Jones*"
```

Note that the string being tested against the name field for each record is in quotes (single or double) and that wild card comparisons can be done in the standard way with '*' and '?' characters.

Logical comparisons may also be combined with those operators above. The permitted logical operators are:

```
and  or  not
```

The following complex example should be examined. The line breaks can occur between any tokens (words, values, operators), except within quoted strings.

```
where (name like "*Jones*" or name = "Smith")
and ( ( site < 300 and not (site = 251 or site eq 15) )
or east < 601000 )
```

5. This choice will display the site records resulting from the last find/query one at a time on the terminal. Use ESC or enter a number to display another record and CTRL-C to end the display.

6. If a graphics monitor is active, the locations of the selected sites will be displayed. The user may choose to erase the screen; display raster, vector, and/or site maps; or display the selected sites from the data base. These maps are requested through the following interactive screen. Just enter ESC to skip this step. If no data base sites are currently selected, that section of the menu will not appear; but the menu can still be used to display the other types of maps. This display function is a major added function of the interactive version of the program; display is not so easy in the command version.

SELECTION MENU FOR ITEMS TO DISPLAY

Enter raster and/or vector map names, if desired

```
_____ Raster map to display
_____ Vector map to display in color: _____
_____ Site list to display
_____ Dpoints with: size=3_ type=box_____ color=white_____
_ Display currently selected sites (enter x)
_ d.sites with: size=6_ type=x_____ color=red_____
_ Erase graphics screen (enter x)
_ d.erase black_____
```

AFTER COMPLETING ALL ANSWERS, HIT <ESC> TO CONTINUE
(OR <Ctrl-C> TO CANCEL)

7. This selection results in a screen prompting for the name of the file to output the selected site records to, and for optional formatting selection. If the file name is lp, the site records are sent to the printer. The optional formatting choices are for export of data in list format (see .print in the first part of this manual page for *s.db.rim* for information and examples).

8. Using this choice you can write (or append) the currently selected sites to a GRASS site_list file in your current mapset. A short menu prompts for the name of the site_list file, and also for the name of a field to be used for the "comment" in the site_list (the site number is the default field). The current date and time, and the names of the mapset and data base in use are entered as an information line in the site_list file. Note that various kinds of raster map layers can be produced from a *s.db.rim* data base by writing site_lists with different fields as "comments" then converting the site_lists to raster files with *s.menu*.

9. Choices 9-12 operate on only a single site and do not use or modify the internal list of sites selected by find/query (choices 3 or 4). Choice 9 is the way to view a single site record, selected by site number. After viewing, ESC will allow entry of another site number and CTRL-C will exit to the main menu.

10. Use this selection to add a new site record to the data base. (A new site is one whose site number does not currently exist in the open data base.) After making this selection, the data base layout will be displayed and you should enter the available information appropriate to each field; the only required entry is the site number field. If values for numeric fields are not entered, zero values will be stored. Unused portions of text fields are stored as strings of spaces.

11. After making this selection and specifying the site number to change field information for, the data is entered as for choice 10, except that the site number cannot be changed. (The command version of the program has provision for making bulk changes after a find or query; see .change.)

12. To delete a single record, enter its site number when requested. All site records chosen by the last find/query operation may be deleted by entering "list" in place of the site number. BE CAREFUL with this, deleted records are really gone.

13. This choice starts a new menu with less commonly used functions. See MANAGEMENT MENU section below.

14. The program will prompt you for one-line Shell Commands until you enter just a <RETURN> to return to the main menu. Often useful for changing the GRASS region, setting a MASK, etc.

FIND/QUERY TARGET MENU

This is the screen to set up the region/mask/target information for the find choice (3) and the query choice (4), except that item B is omitted for choice (4). The choice to append or delete selected records will only be given after a successful find or query has stored some records on the internal record list. See .find and .query for more information.

If a graphics monitor is not active, the "mouse" item is omitted from the screen; and, if a mask is not set, that choice is omitted. The choices entered on this example screen will result in all the sites within a 1500 (meters) radius of the target point (to be chosen with the mouse) being selected and stored on the internal site list by find or query. They are stored in order of proximity to the target. If a site is used as the target, it is always the first in the retrieved list (useful for just selecting one site by number). If a mouse is

chosen to select the target point, a menu to display reference maps is presented, exactly as in choice (6), prior to actually activating the mouse.

```
QUERY/FIND: REGION/MASK/TARGET SELECTION MENU
Data base <arch> (READONLY) in mapset <PERMANENT> open. 113 records.
Mark requests with 'x' and enter required values.
```

```
    Respect current region  _
    Respect current MASK   x
    (forces current region)
```

- A. Find all sites within (or outside) a circular target
 and give the radius (negative for outside) 1500.00_____
 OR
- B. Find a number of sites nearest a point
 and the number of sites requested _____

After selecting A or B, complete one(!) of these:

1. x to select target point with mouse
2. Enter site number for target point _____
3. Target coordinates east 0.00_____
 north 0.00_____

```
Append/Delete to current FIND/QUERY site list (a | d) _
Reset to default choices for this menu _
```

```
AFTER COMPLETING ALL ANSWERS, HIT <ESC> TO CONTINUE
(OR <Ctrl-C> TO CANCEL)
```

QUERY COMMAND MENU

The following screen completes the information for a query (choice 4). It may be left blank if no "where clause" is required. After a successful query, the selected records are displayed one at a time by hitting escape; CTRL-C will quit the display and return to the main menu where several choices of operation on the retrieved sites are offered.

```
QUERY COMMAND CONSTRUCTION SCREEN
Data base <A> in mapset <rim_test> open. 25 records.
The SQL select query will use the current region
and a target clause of 'distance from 596463.15 4919041.88'
```

```
where date = 10/16/89_____
_____
_____
_____
_____
_____
_____
```

(Enter .show on a line to review screen layout and field names.)

```
AFTER COMPLETING ALL ANSWERS, HIT <ESC> TO CONTINUE
(OR <Ctrl-C> TO CANCEL)
```

MANAGEMENT MENU

Choice 13 from the main menu presents this menu. Each item is discussed below.

```
s.db.rim DATA BASE MANAGEMENT MENU
Data base <A> in mapset <rim_test> open. 15025 records.
```

- 1 Make a New Data Base in Current Mapset
- 2 List Available Data Bases
- 3 Remove (PERMANENTLY) Data Base from Current Mapset

```

4 Recover a Data Base from a RIM ASCII File
5 Show Screen Layout of Current Data Base
6 Backup (UNLOAD) Data Base to RIM ASCII Format File
7 Pack the Current Data Base
8 Read a Site list into the Current Data Base

```

```
0 Return to Main Menu
```

```
0_ Your selection
```

```
AFTER COMPLETING ALL ANSWERS, HIT <ESC> TO CONTINUE
      (OR <Ctrl-C> TO CANCEL)
```

1. Use this choice to create a new *s.db.rim* data base in the current GRASS mapset. See section below on MAKE A NEW DATA BASE.

2. List available data bases. Like 2 on MAIN MENU.

3. Delete an entire data base from the current mapset. The name of the data base and additional confirmation of the action are prompted for.

4. Choice 6 allows backup of the definition and data parts of a data base to a transportable text file. To rebuild (or build for the first time) a *s.db.rim* data base from one of these text files do the following steps:

```

# see if the rim directory exists.
ls $LOCATION/rim/sites
# if the directory was not found, make it.
mkdir $LOCATION/rim/sites
# change directory to it.
cd $LOCATION/rim/sites
# have rim build the binary data base files.
rim
RIM> input '/path/to/your/textfile'
RIM> exit

```

The data base is thus created in the current mapset. Several *s.db.rim* commands should be run to verify the integrity of the newly created data base.

5. This merely shows the screen layout of the currently open data base. It is a useful way to quickly see the layout and review the field names and types.

6. When backing up to a text file, the RIM UNLOAD command is run with the output directed to a file of the user's choice. See 4 above. It is wise to do this operation after extensive changes or additions of data records. The resulting text file can be written to tape for preservation, or shared with other GRASS systems, if desired.

7. After deleting a large number of site records, some "wasted" disk space will be present in the binary data base files. This procedure will perform an unload and a reload automatically to recover this unusable disk space. If there is any problem reopening the data base after packing, the user is notified and can recover in various ways depending on the backups which have been done.

8. Data may be loaded into a data base from an existing GRASS *site_list*. This procedure will prompt for the *site_list* name and then add the sites to the currently open data base. If all sites in the list have a comment field of the form "#value ...", the value is used as the data base site number, otherwise the sites are numbered sequentially beginning with 1. Only the site number and location coordinates are loaded for each site record by this procedure; other fields may be later added with the "change" function. See *.read_sites*.

MAKE A NEW DATA BASE

After entering the name of the new data base you wish to create (7 characters maximum), you then decide how to input the information required. This input may be from a text file, or may be entered directly using the editor of your choice; the former is recommended.

See `.make` for the way to define a data base and record (form) layout.

NOTES

This program is included in the GRASS 4.0 release, but is not automatically compiled with other GRASS commands. The user must compile this program separately.

`s.db.rim` interfaces to the RIM program. Both `rim` and `s.db.rim` contain FORTRAN code. The user must have access to a FORTRAN compiler in order to compile and use `s.db.rim`. See the FILES section, below, for the location of source code.

A "date" type field should be added to future versions. This version only allows storing of dates as strings (unless the user codes them to integers), and thus only string type searches can be made for dates.

BUGS

On LINUX I have some problems:

- If you select menu 13/8 and hit ESC, there is no output to the screen. `s.db.rim` expects here the name of the site file or the keyword list.
- If you select menu 8, you should be asked for the `site_list` filename, but the screen remains empty. Simply enter the name.

FILES

The source code for RIM is located under `$GISBASE/./src.related/rim`

The source code for `s.db.rim` is located under `$GISBASE/./src.garden/grass.rim/s.db.rim`

SEE ALSO

The RIM User's Manual by Jim Fox, Academic Computing Services, Univ. of Washington. See especially Appendix B on redistribution of RIM. The RIM Installers manual.

GRASS 4.1 Installation Guide, by Jim Westervelt and Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

d.icons, d.points, g.mapsets, g.region, r.mask, s.in.ascii, s.menu, s.out.ascii, v.db.rim

AUTHORS

James Hinthorne and David Satnik, GIS Laboratory, Central Washington University, Ellensburg, WA.

s.geom

NAME

s.geom - Computes Delaunay triangulation, MinMax-Angle triangulation, MinMax-Slope triangulation, MaxMin-Height triangulation, Regular triangulation, planesweep triangulation, Voronoi diagram, and convex hull of sites in 2 and 2 1/2 dimensions.

GRASS VERSION

4.x

SYNOPSIS

s.geom

s.geom help

s.geom input=name output=name [precision=value] [operation=name]

DESCRIPTION

s.geom takes a sites file as input and computes various triangulations, the Voronoi diagram, or the convex hull of the sites. The z-coordinate is read from the description field if it is specified, otherwise 0 is assumed. The z-coordinate is used for the MinMax-slope triangulation and for the regular triangulation (where it is interpreted as the weight of the site). For all other computations the z-coordinate is ignored.

The MinMax-angle triangulation is the triangulation for the sites, which minimizes (lexicographically) the sorted vector of all the angles of triangles in the triangulation. The MaxMin-height and MinMax-slope triangulations are similar. The algorithms used for the computations are not heuristics, they actually achieve the optimum.

The regular triangulation is the weighted version of the Delaunay triangulation (weights are assigned to the sites, the delaunay triangulation corresponds to the regular triangulation where all the sites have identical weights).

The output is saved in vector file format.

OPTIONS

Parameters:

input=name Input vector (level 2) file.

output=name Output vector file.

precision=value Number of significant positions after the decimal point. (default is 0).

operation=name One of the following: sweep, Delaunay, angle, height, slope, hull. These correspond to the planesweep triangulation, Delaunay triangulation, MinMax-angle triangulation, MaxMin-height triangulation, MinMax-slope triangulation, regular triangulation, Voronoi diagram, and convex hull, respectively. (default is Delaunay triangulation).

NOTE

Only the sites which fall into the current region are used for the computations.

The computation times for the various operations depends strongly on the algorithm used.

The plansweep triangulation and convex hull computation require $O(n \log n)$ operations in the worst case [Ed]. The Delaunay heuristic needs $O(n^2)$ time in the worst case, however it performs much faster in

practice. The MinMax-angle and MaxMin-height triangulations need $O(n^2 \log n)$ operations [BeEd, EdTa], and the MinMax-slope triangulation needs $O(n^3)$ operations [BeEd].

Internally, the coordinates of the sites are stored in fix-point format. Therefore, the number of decimal digits cannot exceed 64 bit (or approx. 16 decimal digits).

BUGS

Some fields of the header in the output file are not properly set.

REFERENCES

M. Bern, H. Edelsbrunner, D. Eppstein, S. Mitchel, T. S. Tan. Edge Insertion for Optimal Triangulations.

In Proc. 1st Latin American Sympos. Theoret. Informatics 1992, 46--60.

H. Edelsbrunner. Algorithms in Combinatorial Geometry. Springer-Verlag, Heidelberg, Germany, 1987.

H. Edelsbrunner, N. R. Shah. Incremental Flipping Works for Regular Triangulations. In Proc. 8th Ann. Sympos. Comput. Geom. 1992, 43-52.

H. Edelsbrunner, T.S. Tan and R. Waupotitsch. An $O(n^2 \log n)$ Time Algorithm for the MinMax Angle Triangulation. SIAM J. Sci. Statist. Comput. 13 1992, 994-1008.

SEE ALSO

v.geom

AUTHOR

Roman Waupotitsch

s.in.ascii

NAME

s.in.ascii - Converts an ASCII listing of site locations and their descriptions into a GRASS site list file. (GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.in.ascii

s.in.ascii help

s.in.ascii [-e]sites=name [input=name] [fs=character|space|tab]

DESCRIPTION

s.in.ascii converts an ASCII listing of site locations and category labels into a file in GRASS site list file format.

Input can be entered via standard input or from the file *input=name*. Each line of input should contain the easting, northing, and (optionally) the category label associated with a site. The *fs=name* option (where name is either a character, a space, or a tab) can be used to specify the use of a particular field separator between these three input fields. This is useful when input is obtained from other programs (see NOTES, below). Output is stored in the file *sites=name* and placed in the *site_lists* directory under the user's current mapset.

The GRASS program *s.out.ascii* can be used to perform the reverse function, converting a file in GRASS site list format into an ASCII listing of eastings, northings, and category labels associated with site locations.

OPTIONS

Flag:

-e Attribute field to import represents elevation heights.

Parameters:

sites=name Name of the new GRASS site list file to be output.

input=name Name of an existing ASCII file containing site locations and labels.

"fs=character|space|tab" The field separator separating the easting, northing, and category label in each line of the input file. The field separator can be a character, a space, or a tab.

Default: space

s.in.ascii can be run either non-interactively or interactively. The program will be run non-interactively if the user specifies a name to be assigned to the sites file output, the name of an existing ASCII file containing input, and (optionally) a field separator *fs* appearing in the input file, using the form:

```
s.in.ascii sites=name [input=name] [fs=character|space|tab]
```

Alternately, the user can simply type *s.in.ascii* on the command line, without program arguments. In this case, the user will be prompted for parameter values using the standard GRASS *parser* interface described in the manual entry for *parser*. If the user does not specify the name of an input file containing site locations and (optionally) category labels, these should be entered to the program via standard input. If

importing elevation heights, the *-e* flag must be used. Otherwise the GRASS interpolation and conversion modules will not detect this points data attribute properly.

NOTES

Other GRASS programs can be used to produce output in a format suitable for input to *s.in.ascii*. For example, the user might pipe output produced by *d.where* into *s.in.ascii* to create a site list file called *my.sites* containing site locations pointed to with the mouse, as illustrated below. In this example it was unnecessary to specify the field separator used in the input, since *d.where* output separates the easting and northing values with spaces, and spaces are the default field separator assumed by *s.in.ascii*.

d.where | s.in.ascii sites=my.sites

SEE ALSO

d.points, d.sites, d.what.rast, d.where, s.out.ascii

AUTHOR

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

s.in.grid

NAME

s.in.grid - convert Arc GRIDASCII output to site list.
(GRASS Shell Script)

GRASS VERSION

4.x

SYNOPSIS

s.in.grid file

DESCRIPTION

s.in.grid is a awk script that reads the output of Arc/INFO's GRIDASCII command and converts it to a GRASS site list.

Parameters:

file Name of the output of GRIDASCII.

EXAMPLE

Typing the following:

```
Arc: GRIDASCII nitrates tmp
```

```
Arc: quit
```

```
GRASS GRID> s.in.grid tmp | s.in.ascii nitrates fs=''
```

```
GRASS GRID> rm tmp
```

This procedure exports the GRID file nitrates from Arc/INFO and imports it as the GRASS site list nitrates.

NOTES

awk may not be able to handle large ASCII grids. In these instances, try using nawk, or better yet, gawk.

FILES

\$GISBASE/scripts/s.in.grid

SEE ALSO

s.in.ascii

AUTHOR

James Darrell McCauley, Agricultural Engineering, Purdue University

s.info

NAME

s.info - reports attribute, label, and other information about a sites file.
(GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.info

s.info help

s.info [-qwQ] sites=name

DESCRIPTION

s.info reads the entire sites file and reports header & label information, type of category, number of attributes & dimensions, and min and max values for each dimension, attribute, or category. For best results sites should conform to the multiple attribute format as specified in GRASS5.0.

OPTIONS

This program can be run non-interactively or interactively. The program will be run non-interactively; if the user specifies program arguments and flag settings on the command line using the form

s.info [-qwQ] sites=name

Alternately, the user can type simply *s.info* 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*.

Flags:

-q Load quietly

-w Only count sites in current window for min/max.

-Q Show quartiles for decimal attributes

Parameter:

sites = name Name of sites file.

Example output:

```
SITES FILENAME: r_analys
```

```
-----
```

```
Header Information:
```

```
-----
```

```
name          r_analys
description    soil data from Germany
format         |||%%%%%%%%
labels         X|Y|Horizon| %pH%Kcal%Pcal%Corg%Ccarb%Nt%Trd
```

```
Number of DIMENSIONS:          3
```

```
-----
```

```
      - - MIN - -      - - MAX - -
dim 1  4457986.000000  4459536.000000  X
dim 2  5372514.000000  5374314.000000  Y
dim 3      0.000000      120.000000  Horizon
```

Type of CATEGORY information: NO CATEGORY

Number of DOUBLE attributes: 7

	MIN	Q1	MED	Q3	MAX	
dbl 1	0	5.4	6	6.2	7.7	pH
dbl 2	0	0.058	0.1	0.174	1.212	Kcal
dbl 3	0	0.009	0.017	0.044	0.415	Pcal
dbl 4	0	0.18	0.34	1.09	13.68	Corg
dbl 5	0	0	0	0	3.82	Ccarb
dbl 6	0	0.031	0.05	0.117	0.744	Nt
dbl 7	0.31	1.45	1.55	1.6	1.89	Trd

Number of STRING attributes: 0

TOTAL SITES COUNTED: 2343

SITES FILENAME: US.places

Header Information:

```

name          US Places
description    names of populated places in the United States
format        | |#@@%%%
labels
easting|northing|#FIPS@place@state@place_type%population%households%land_area%water_area%
opulation_density
time          21 Jul 1995

```

Number of DIMENSIONS: 2

	MIN	MAX	
dim 1	-124.610970	-67.012039	easting
dim 2	24.562840	48.996136	northing

Type of CATEGORY information: CELL_TYPE

	MIN	MAX	
	100124	5686665	FIPS

Number of DOUBLE attributes: 5

	MIN	Q1	MED	Q3	MAX	
dbl 1	0	437	1371	4605	7.32256e+06	population
dbl 2	0	190	583	1905	2.99217e+06	households
dbl 3	5	1659	3973	10396	1.96495e+06	land_area
dbl 4	0	0	0	76	781914	water_area
dbl 5	0	0.191793	0.393334	0.712237	17.692	population_density

Number of STRING attributes: 3

	MIN_LEN	MAX_LEN	
str 1	3	35	place
str 2	2	2	state
str 3	3	7	place_type

TOTAL SITES COUNTED: 22993

BUGS

If the *-Q* flag is given, the sites are cached in order to determine quartile values; otherwise the sites are read sequentially without caching. Therefore, use of the *-Q* option may result in an out-of-memory error for very large sites files. The algorithm used to find quartile values matches that used by *s.univar*, so if the number of sites is not evenly divisible by 4, the resulting Q1 and Q3 values may not exist in the data, but is an average of the two neighboring values. Likewise, if the number of sites is not evenly divisible by 2, the resulting median will be an average of its neighbors.

SEE ALSO

d.sites, d.siter, d.sites.qual, s.univar

AUTHOR

Bill Brown, UI GMS Laboratory

s.kcv

NAME

s.kcv - Randomly partition sites into test/train sets. (GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.kcv

s.kcv help

s.kcv [-dq] k=value sites=name

DESCRIPTION

s.kcv randomly divides a sites lists into k sets of test/train data (for k-fold cross validation). Test partitions are mutually exclusive. That is, a site will appear in only one test partition and k-1 training partitions. The program generates a random point using the selected random number generator and then finds the closest site to it. This site is removed from the candidate list (meaning that it will not be selected for any other test set) and saved in the first test partition file. This is repeated until enough points have been selected for the test partition. The number of sites chosen for test partitions depends upon the number of sites available and the number of partitions chosen (this number is made as consistent as possible while ensuring that all sites will be chosen for testing). This process of filling up a test partition is done k times.

OPTIONS

Flags:

-d Use drand48() (default is rand()).

-q Run quietly. Don't report progress.

Parameters:

k=value Positive integer value indicating the number of partitions.

sites=name Name of a sites file to store random points in.

Test/train pairs are saved as sites list using name as a basename. Test sites are saved in name-test.i while training sites are saved in name-train.i, where i ranges from zero to k.

NOTES

Existing files are silently overwritten.

An ideal random sites generator will follow a Poisson distribution and will only be as random as the original sites. This program simply divides sites up in a random manner.

Be warned that random number generation occurs over the intervals defined by the current region.

This program may not work properly with Lat-long data.

SEE ALSO

s.rand, *g.region*

AUTHOR

James Darrell McCauley, Purdue University

s.label

NAME

s.label - makes a label-file from a GRASS sites list.
(GRASS Sites Program)

GRASS VERSION

4.x

SYNOPSIS

s.label

s.label help

s.label [-r] sites=name label=name [xoffset=value] [yoffset=value] [reference=name] [font=name] [ps_font=name] [size=value] [color=name] [width=value][hcolor=name] [hwidth=value] [background=name] [border=name] [opaque=name]

DESCRIPTION

s.label makes a label-file from a GRASS sites list.

OPTIONS

Flag:

-r make sites from label file

Parameters:

sites Name of a site list

label Name of a paint-label file

xoffset Offset label in x-direction
Default: 0

yoffset Offset label in y-direction
Default: 0

reference Reference position
Options: center, left, right, upper, lower
Default: center

font Font type.
Options: standard, cyrilc, gothgbt, gothgrt, gothitt, greekc, greekcs, grrekp, greeks, italicc, italiccs, italict, romanc, romancs, romand, romanp, romans, scriptc, scripts
Default: standard

ps_font PostScript font (for use with ps.map); N.B. screen display will be in standard font!

Options: Helvetica, Helvetica-Bold, Helvetica-Oblique, Helvetica-BoldOblique, Times-Roman, Times-Bold, Times-Italic, Times-BoldItalic, Courier, Courier-Bold, Courier-BoldOblique, Symbol, AvantGarde-Book, AvantGarde-BookOblique, AvantGarde-Demi, AvantGarde-DemiOblique, Bookman-Demi, Bookman-DemiItalic, Bookman-Light, Bookman-LightItalic, Helvetica-Narrow, Helvetica-Narrow-Bold, Helvetica-Narrow-Oblique, Helvetica-Narrow-BoldOblique, NewCenturySchlbk-Roman, NewCenturySchlbk-Italic, NewCenturySchlbk-Bold, NewCenturySchlbk-BoldItalic, Palatino-Roman, Palatino-Italic, Palatino-BoldItalic, ZapfChancery-MediumItalic

size Label size (in map-units)

Options: 1-1000

Default: 100

color Text color.

Options: aqua, black, blue, brown, cyan, gray, green, grey, indigo, magenta, orange, purple, red, violet, white, yellow

Default: black

width Line width of text (only for *p.map* output).

Options: 1-100

Default: 1

hcolor Highlight color for text (only for *p.map* output).

Options: none, aqua, black, blue, brown, cyan, gray, green, grey, indigo, magenta, orange, purple, red, violet, white, yellow

Default: none

hwidth Line width of highlight color (only for *p.map* output).

Options: 0-100

Default: 0

background Background color.

Options: none, aqua, black, blue, brown, cyan, gray, green, grey, indigo, magenta, orange, purple, red, violet, white, yellow

Default: none

border Border color.

Options: none, aqua, black, blue, brown, cyan, gray, green, grey, indigo, magenta, orange, purple, red, violet, white, yellow

Default: none

opaque Opaque to vector (only relevant if background color is selected).

Options: yes, no

Default: yes

AUTHOR

Philip Verhagen, S t i c h t i n g R A A P, Regionaal Archeologisch Archiverings Projekt

address: Plantage Muidergracht 14

1018 TV Amsterdam

THE NETHERLANDS

e-mail: motte@xs4all.nl

s.medp

NAME

s.medp - Median polish for a GRASS sites list.
(GRASS Sites Program)

GRASS VERSION

4.x,5.x

SYNOPSIS

s.medp

s.medp help

s.medp [-aeqs] sites=name vect=name output=name [report=name] [thresh=n]

DESCRIPTION

s.medp performs median polish on an existing sites list. The sites are overlaid onto a grid (vect=name) and each site is associated with the closest node. Therefore, sites do not necessarily have to be oriented on a grid. Each node may be associated with zero, one, or more sites and the grid may be rotated. Also, the length of each box does not have to be equal to width of each box defining the grid.

Once sites are associated with nodes on a p by q grid, p+q+1 extra storage locations are created (initialized to zero) to store all, row, and column effects. The median of each row is removed from the data and added to the extra p cells. Then, medians are removed from the data as well as from the p cells containing row effects. The medians of data in this pass are stored in the extra q cells (column effects) and the median of the row effects is stored in the extra (p+1, q+1) cell (all effect). This is repeated until each successive iteration leaves each site unchanged (within a tolerance, described by [thresh=n]).

OPTIONS

Flags:

-a Use all sites found in the named sites file, rather than limiting output to sites falling within the current geographic region.

-e Store row, column, and all effects in output file.

-s Write results to a sites list file (default is to write points in a binary vector file).

-q Quiet. Cut out the chatter.

Parameters:

sites=name Name of an existing sites file.

vect=name Name of the grid file (binary vector file).

output=name Name of the output file (binary vector file or sites list).

report=name Name of an ASCII file which shows original and residual data in tabular form.

thresh=n Threshold to determine when convergence of median polish is obtained.
(default = 1).

s.medp can be run either non interactively or interactively. The program will be run non-
the user specifies the name of an existing sites list file, name for a vect file, and name of an output file
using the form

```
s.medp [ aeqs] sites=name vect=name output=name [report=name] [thresh=n]
```

Alternately, the user can si *s.medp* on the command line, without program arguments. In this
parser interface.

NOTES

When using the report option, the tabular report may be more than 128 columns wide. Therefore, it may
be useful to add a TeX or PostScript wrapper before printing (e.g., `enscript h - -fCourier Bold5 -
report.ps report`) so that the page is in landscape orientation and a very small font is used.

The input vector file defining the grid should be large enough so that the extreme east west and north-
-rotated grids) can be used to store row and column effects. Otherwise, when

when written to a vector output file.

Currently, nothing is done with the residuals, but I would like implement some sort of analysis of these in
the future (suggestions?). Kriging using these residuals as new data set is being planned

The *s* flag is yet to be implemented.

This should probably be re written to use points in a vector file as input (thus creating *v.medp*
g.gnuplot), but

trend can be obtained by extrapolation and/or interpolation.

Extrapolating and interpolating using the row, column, and all effects to create a raster file would be a
and creating the surface with separate program.

Please send all bug fixes and comments to the author.

SEE ALSO

187.

AUTHOR

James Darrell McCauley, Agricultural Engineering Purdue University

s.menu

NAME

s.menu - Accesses and manipulates GRASS site location data.
(GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.menu

DESCRIPTION

The *s.menu* program provides the user with the capability of interfacing site location data with the geographic data in raster map layers. Two types of spatial analysis reports on sites can be generated, and an interface to the "S" statistical package is provided.

The *s.menu* program is an interface to functions that allow the user to manipulate GRASS site "lists". A site list is a list of eastings and northings describing the location of some point feature. It can also contain a category value and category label for each site location. The program is interactive. After typing *s.menu* on the command line, the user selects site functions from a menu.

The main menu is shown below:

```
SITES MAIN MENU      (current list: no sites)

LOCATION: spearfish    REGION 4928000.00(N) 4914000.00(S) 100.00 (RES)
MAPSET:  PERMANENT   609000.00(E)  590000.00(W) 100.00 (RES)
MASK:  none

Please select one of the following

 1 Read an existing site list
 2 Mask current site list
 3 Save the current site list in your mapset

 4 Check site list for duplicate sites
 5 Edit site list using a UNIX editor

 6 Convert site list to raster file (0/1)
 7 Convert site list to raster file (frequency of occurrence)

 8 Run reports on the current site list

stop Leave the s.menu program
```

At the top of the menu is general information about the user's current MAPSET, LOCATION, etc. Note the above message in parentheses "current list: no sites." This message will vary depending on the status of the list. For example, after the user reads the existing site list file *arch_sites*, the message would read (given the geographic region indicated): "current list: 25 sites, 24 in current region."

Read an existing site list

This option will copy an existing site list file into the current site list within the sites server. Existing site lists are stored under a GRASS data base and are pulled into the *s.menu* server via this option. Other sites menu functions operate only on the current site list file in the server -- you must therefore "read an existing site list" file BEFORE performing any of the other sites menu functions. Note: Site lists can be

s.menu option 5 (edit) followed by option 3 (save).

to create a site list file in the appropriate format using any text editor (e.g., "vi"), and to put this site list

\$LOCATION/site_lists). The user can do this either inside of GRASS or outside of GRASS. Alternately, the user can run other GRASS programs which format their output as a GRASS site list file (`r.site` , `s.db.rim`, `v.mkquads`, `v.to.sites`

GRASS programs' output in the form of a site list file (`d.what.rast` `d.what.vect`

Mask current site list

The site list can be reduced to a subset that includes only sites which fall in specific categories within a specified raster map layer. The user will be asked to specify the name of a raster map layer to form the

site list. As with *r.mask*,

will remain. Areas assigned category values not selected will be re assigned to category value "0" ("no data"). Note: This masking operation is performed only against the site list itself and not against other

the *r.mask*

Save the current site list in your mapset

The current site list can be stored permanently in your current mapset with this option. You will be asked to name the saved site list and to provide a short description of it. Sa

1) during later runs of *s.menu*

like *d.sites*, *d.points*, *d.icons*, *p.icons*, *s.surf.idw* *s.db.rim*.

Note: Saved lists will be removed if the GRASS mapset under which they are stored is removed.

Check site list for duplicate sites

It is not desirable that a site list contain multiple references to the same site. This option attempts to recognize duplicate sites. Duplicates are displayed to th

user desires. Duplicates can also be removed by hand using option 5(edit).

Edit site list using a UNIX editor

The user can modify the current site list or create a new site list by hand using a UNIX editor. You will be asked to specify the text editor you prefer to use. You should exercise some care if you select this

s.menu. See the

r a description of the site list format.

list stored under a GRASS mapset. If you wish to modify a stored site list file, you will have to combine options 1 (read), 5 (edit) and 3 (save).

Convert site list file to raster file (0/1)

You can create a raster data representation of the site list in your GRASS mapset. Once created, this

urther analyses. Allowing the user to create

a raster map layer of sites opens up the full analysis capabilities for site data that are available for raster

You have the option of specifying the number of cells to represent a site. The minimum is one cell per site. The alternatives are squares around the site: 3x3, 5x5, 7x7, etc.

The number of categories present in the new raster map layers will depend on the format of your site list file (see sites.format). You can create a non-binary raster map layer representation of your site list by creating the site list in the format:

"E|N|" "#n label" where E is the Easting, N is the Northing, and #n label is the description field. The description field consists of a pound sign # followed by the category value n to be associated with the site's cell location, and the category label for n. If the user does not include a description field starting with #n beside the Easting and Northing on every line in the sites list file, a binary raster map layer will be created instead. In the binary raster file, each site will be represented as the category value 1. Non-site cells will be coded as category value 0.

Note that only sites within the current geographic region will be considered. However, if the size of the sites is more than one cell (3x3, 5x5, etc.) and a site lies near an edge of the geographic region, some of the cells for the site may fall outside the geographic region. These cells will not appear in the raster map layer, and the site will no longer be 3x3 or 5x5 but will be clipped to fit the geographic region.

Convert site list file to raster file (frequency of occurrence)

You can also create a frequency of occurrence raster map layer representation of the site list file.

The raster category values will be coded as the number of sites that fall within the cell.

In this function, you do NOT have the option of specifying the number of cells to represent a site.

Run reports on the current site list

The current list of sites is passed to the report server after removing sites, which do not fall within the geographic region of the user's current GRASS mapset (*g.region*). The user then selects the names of one or more raster map layers for analysis. Data at (or near) the sites extracted from these raster map layers form the basis for all reports.

The user specifies the 'size' of a site in cells. The 'size' may be specified as a single cell, or as a 3x3 square around the site, or 5x5, or 7x7, etc (where the size is an odd integer).

The following menu of reports is then presented:

```
SITES REPORT MENU

Please select one of the following

  1 Site characteristics report
  2 Site occurrence report

  3 Convert data to S input format
  4 Produce machine-readable data file

stop return to SITES MAIN MENU
```

Site characteristics report

This report provides geographic information about each site.

Each site is identified by description and locational information. The 'description' is an identification of location is an easting and a northing. (The location does not denote the extent of the site, since, for example, an archeological site, which takes up two hectares, would be represented as a single

the categories (i.e., characteristics) that occurred at the site (as well as a count of the number of cells in each category).

This can easily generate a massive amount of information, which is difficult. Therefore, option 2 produces a synopsis of the information.

Site occurrence report

This report provides aggregate site characteristic information organized by raster map layer. The report provides square statistics for each raster map layer, measuring number of expected sites (assuming a occurring cell category in the site (i.e., the statistical mode). See the GRASS manual entry for `sites.occure`

Convert data to S input format

This function converts the GRASS data extracted for the sites into a form usable by the S statistical package. The user provides a file to contain the information. Once the file is written, the user must exit GRASS, run S on an S data base, and issue the S command source file to bring the data into the S data base. (Of course, file would be the name of the actual file supplied by the user.) See manual entry `sites.S`

Produce machine-

user-

-written

`$LOCATION/site_list/<file >`

SEE ALSO

d.icons, d.graph, d.points, d.sites, p.icons, r.random, d.what.rast, d.what.vect, r.what, s.in.ascii, sites.occure, sites.S

AUTHORS

occurrence sites to cell function

s.normal

NAME

s.normal - tests for normality for sites.
(GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.normal

s.normal help

s.normal [-q] input=name [tests=range[,range,...]]

DESCRIPTION

s.normal computes tests of normality on a site list.

OPTIONS

Flag:

-q Quiet. Cut out the chatter when reading site list.

Parameters:

input=name Name of sites list.

tests=range[,range,...]] Tests of normality requested (see below).

NOTES

The tests that *s.normal* performs are indexed below. The tests that are performed are specified by giving an index, ranges of indices, or multiple thereof.

Sample skewness and kurtosis

Geary's a-statistic and an approximate normal transformation

Extreme normal deviates

D'Agostino's D-statistic

Modified Kuiper V-statistic

Modified Watson U²-statistic

Durbin's Exact Test (modified Kolmogorov)

Modified Anderson-Darling statistic

Modified Cramer-Von Mises W²-statistic

Kolmogorov-Smirnov D-statistic (modified for normality testing)

Chi-Square test statistic (equal probability classes) and the number of degrees of freedom

Shapiro-Wilk W Test

Weisberg-Binghams W" (similar to Shapiro-Francia's W')

Royston's extension of W for large samples

Kotz Separate-Families Test for Lognormality vs. Normality

EXAMPLE

s.normal input=soils tests=1-3,14

-statistic and an approximate normal

s.univar

AUTHOR

James Darrell McCauley, Agricultural Engineering, Purdue University

s.out.ascii

NAME

s.out.ascii - Converts a GRASS site list file into an ASCII listing of site locations and their descriptions. (GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.out.ascii

s.out.ascii help

s.out.ascii [-ad] sites=name [fs=character/space/tab]

DESCRIPTION

s.out.ascii converts an existing site list file (*sites=name*) into an ASCII listing of site locations and (optionally) their category labels, in a format suitable for input to other programs (e.g., *d.points*, *m.u2ll*, etc.).

Each line of output consists of the easting, northing, and category label for a site listed in the named sites file. The *fs=name* option (where *name* is either a character, a space, or a tab) can be used to place a particular field separator between these three output fields. This is useful when output is to be manipulated by other programs, like *awk* or *sed*.

The GRASS program *s.in.ascii* can be used to perform the reverse function, converting a UNIX file containing eastings, northings, and category labels associated with site locations into GRASS site list file format.

OPTIONS

Flags:

-a Output all sites found in the named sites file, rather than limiting output to sites falling within the current geographic region.

-d Include site descriptions (category labels) in the output.

Parameters:

sites=name Name of an existing site list file.

fs=character/space/tab The field separator to be placed between the easting, northing, and (optionally) category label on each line of output. The field separator can be a character, a space, or a tab.

Default: space

s.out.ascii can be run either non-interactively or interactively. The program will be run non-interactively if the user specifies the name of an existing site list file and (optionally) a value for *fs*, using the form

```
s.out.ascii [-ad] sites=name [fs=character/space/tab]
```

where *name* is the name of an existing site list file to be converted to a brief ASCII listing, and *fs* is the field separator to be placed between output fields. The user can also the *-a* and *-d* options to use all sites in the named sites file and to include site descriptions in the output.

Alternately, the user can simply type *s.out.ascii* on the command line, without program arguments. In this case, the user will be prompted for parameter values using the standard GRASS *parser* interface.

NOTES

The output from *s.out.ascii* may be placed into a file by using the UNIX redirection mechanism; e.g.:

```
s.out.ascii sites=archsites > out.file
```

s.out.ascii output may also be redirected into other programs; e.g.:

```
s.out.ascii sites=archsites | d.points color=red size=10 type=diamond
```

SEE ALSO

d.points, d.sites, m.ll2u, m.u2ll, s.in.ascii

AUTHOR

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

s.perturb

NAME

s.perturb - Random location perturbations of GRASS sites.
(GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.perturb

s.perturb help

s.perturb [-q] input=name output=name distribution=[uniform/normal] parameters=value,[value]

DESCRIPTION

s.perturb reads a site list and writes the same list but perturbs the eastings and northings by adding either a uniform or normal delta value.

OPTIONS

Flags:

-q Quiet. Cut out the chatter.

Parameters:

input=name Name of an existing sites file.

output=name Name of output sites file.

distribution=[uniform/normal] Distribution of perturbation.

parameters=value,[value] Parameter(s) of distribution. If the distribution is uniform, only one parameter, the maximum, is needed. For a normal distribution, two parameters, the mean and standard deviation, are required.

NOTES

The uniform distribution is always centered about zero. The associated parameter is constrained to be positive and specifies the maximum of the distribution; the minimum is the negation of that parameter.

Usually, the mean (first parameter) of the normal distribution is zero (i.e., the distribution is centered at zero). The standard deviation (second parameter) is naturally constrained to be positive.

Output sites are not guaranteed to be contained within the current geographic region.

SEE ALSO

g.region, s.rand, s.univar, s.kcv

AUTHOR

James Darrell McCauley, Agricultural Engineering, Purdue University

Random number generators originally written in FORTRAN by Wes Peterson and translated to C using f2c

s.probplt

NAME

s.probplt - Normal probability plot of a GRASS site list.
(GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.probplt

s.probplt help

s.probplt [-alq] sites=name width=value [graph=name]

DESCRIPTION

s.probplt does normal or lognormal probability plots of site values.

OPTIONS

Flags:

-a Use all sites found in the named sites file, rather than limiting output to sites falling within the current geographic region.

-l Lognormal probability plot instead of normal.

-q Quiet. Cut out the chatter.

Parameters:

sites=name Name of an existing sites file.

width=value Width of bins.

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

EXAMPLE

Given a sites file named example in the following format:

```
83.8 | 92.2 | 3.5689
83.8 | 82.2 | 3.9269
83.8 | 80.2 | 3.5389
83.8 | 69.2 | 3.7452
```

Suppose that we wish to examine normality of the site values (third column). The first step is to examine minimum and maximum and determine with the histogram bin width using *s.univar*:

```
s.univar -gq sites=example
```

This command outputs:

```
n=216
min=1.489
max=3.9419
```

along with other useful statistics. For this range and number of observations, we randomly select 0.1 as the histogram bin width. Then, the following command graphs a probability plot in the GRASS graphics window and saves it in the current working directory with a basename of myplot:

```
s.problt -q sites=example width=0.1 graph=myplot
```

To view the graph again, try

```
g.gnuplot myplot.gp
```

Using *g.gnuplot*, the graphs may be output as PostScript, LaTeX, FrameMaker MIF, or many other formats.

SEE ALSO

s.univar, s.normal

AUTHOR

James Darrell McCauley, Agricultural Engineering, Purdue University

s.qcount

NAME

s.qcount - indices for quadrat counts of sites lists
(GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.qcount

s.qcount help

s.qcount [-ciq] input=name n=value r=value

DESCRIPTION

s.qcount chooses *n* circular quadrats of radius *r* such that they are completely within the bounds of the current region and no two quadrats overlap. The number of sites falling within each quadrat is counted and indices are calculated to estimate the departure of site locations from complete spatial randomness.

OPTIONS

Flags:

-c Print count data for each quadrat in the form of a sites list (E|N|#n desc, where E is the easting and N is the northing of the quadrat center, n is the quadrat number, and desc is an integer indicating the number of sites contained within the quadrat).

-i Suppress printing table of indices (default is to print indices and not the count data).

-q Quiet. Cut out the chatter.

Parameters:

input=name Name of sites list defining qcount points.

n=value Number of quadrats (integer).

r=value Radius of quadrats (floating point).

NOTES

This program may not work properly with lat-long data. It uses `hypot()` in two files: `count.c` and `findquads.c`.

SEE ALSO

s.rand

REFERENCES

Noel A. C. Cressie. *Statistics for Spatial Data*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York, NY, 1st edition, 1991.

Brian D. Ripley. *Spatial Statistics*. John Wiley & Sons, New York, NY, 1981.

References to the indices include:

F. N. David and P. G. Moore. Notes on contagious distributions in plant populations. *Annals of Botany*, 18:47-53, 1954.

J. B. Douglas. Clustering and aggregation. *Sankhya B*, 37:398-417, 1975.

R. A. Fisher, H. G. Thornton, and W. A. Mackenzie. The accuracy of the plating method of estimating the density of bacterial populations. *Annals of Applied Biology*, 9:325-359, 1922.

M. Lloyd. Mean crowding. *Journal of Animal Ecology*, 36:1-30, 1967.

M. Morista. Measuring the dispersion and analysis of distribution patterns. *Memoires of the Faculty of Science, Kyushu University, Series E. Biology*, 2:215-235, 1959.

AUTHOR

James Darrell McCauley, Agricultural Engineering, Purdue University

s.rand

NAME

s.rand - Randomly generate a GRASS sites list.
(GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.rand

s.rand help

s.rand [-r/d] n=value sites=name

DESCRIPTION

s.rand randomly generates sites using the selected random number generator.

OPTIONS

Flags:

-r Use rand() (the default).

-l Use drand48().

Parameters:

n=value Positive integer value indicating the number of sites to be created.

sites=name Name of a sites file to store random points in.

SEE ALSO

UNIX man pages for rand(3) and drand48(3).

AUTHOR

James Darrell McCauley, Agricultural Engineering, Purdue University

s.sample

NAME

s.sample - Sample a raster file at site locations.
(GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.sample

s.sample help

s.sample [-BCcdlq] input=name [output=name] rast=name [z=name]

DESCRIPTION

s.sample samples a GRASS raster map at the site locations in the input file by either cubic convolution interpolation, bilinear interpolation, or nearest neighbor sampling (default). Categories values are sampled. This program may be especially useful when sampling for cross validation of interpolations whose output is a raster map.

OPTIONS

Flags:

-B Use bilinear interpolation.

-C Use cubic convolution interpolation.

-c Use numeric category labels instead of category values.

-d Calculate difference between raster value and site value (raster minus site).

-l If site is in the E|N|#n label format (instead of the E|N|label format), use the numeric label as the site value. Implies the *-d* flag.

-q Quiet. Cut out the chatter.

Parameters:

input=name Name of sites list defining sample points.

output=name Optional name of sites list in which output will be stored. Standard output is used if this is missing.

rast=name Name of raster map to be sampled.

z=value Option scaling factor for values read from raster map. Sampled values will be multiplied by this factor. If omitted, this is set to 1.0.

NOTES

If any of *-cdl* are specified, it is important that the raster category label and/or the site description are numeric. No error checking is done except for "no data" values as raster category labels. In this instance, a warning is issued and a zero value is assumed.

This program may not work properly with lat-long data when the *-BC* flags are used.

When interpolation is done (i.e., the -BC flags are used), values are assumed to be located at the centroid of grid cells. Therefore, current resolution settings are important.

SEE ALSO

s.rand, s.kcv, g.region

Image Sampling Methods - GRASS Tutorial on *s.sample* (available as *s.sample-tutorial.ps.gz*)

AUTHOR

James Darrell McCauley, Agricultural Engineering, Purdue University

s.surf.idw

NAME

s.surf.idw - Surface generation from sites data program.
(GRASS Raster Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.surf.idw *input=name*
output=name
[*npoints=count*]

DESCRIPTION

s.surf.idw fills a raster matrix with interpolated values generated from a set of irregularly spaced data points using numerical approximation (weighted averaging) techniques. The interpolated value of a cell is determined by values of nearby data points and the distance of the cell from those input points. In comparison with other methods, numerical approximation allows representation of more complex surfaces (particularly those with anomalous features), restricts the spatial influence of any errors, and generates the interpolated surface from the data points. It is the most appropriate method to apply to most spatial data.

This program allows the user to use a GRASS site list file, rather than a raster map layer, as input.

The program will be run non-interactively if the user specifies the values of needed program parameters and any desired optional parameter values on the command line, using the form:

s.surf.idw *input=name* *output=name* [*npoints=count*]

Alternately, the user can simply type *s.surf.idw* on the command line, without program arguments. In this case, the user will be prompted for needed inputs and option choices using the standard GRASS *parser* interface.

OPTIONS

Parameters:

input=name Name of an input site list file that contains a set of irregularly spaced data values; i.e., some cells contain known data values while the rest contain zero (0).

output=name Name to be assigned to the new output raster map layer containing a smooth surface generated from the known data values in the input site list file.

npoints=count The number of points to use for interpolation. By default, the 12 nearest points are used for interpolation.

Default: 12

NOTES

The amount of memory used by this program is related to the number of non-zero data values in the input sites list file. If the input site list is very dense (i.e., contains many non-zero data points), the program may not be able to get all the memory it needs from the system. The time required to execute increases with the number of input data points.

If the user has a mask set, then interpolation is only done for those cells that fall within the mask. However, all non-zero data points in the input map are used even if they fall outside the mask.

SEE ALSO

d.sites, g.region, r.mask, r.surf.contour, r.surf.idw, r.surf.idw2, s.db.rim, s.menu

AUTHOR

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

s.surf.krig

NAME

s.surf.krig Surface interpolation from site data via kriging?

GRASS VERSION

4.x,5.x

SYNOPSIS

s.surf.krig

s.surf.krig -help

s.surf.krig input=name outz=name outvarz=name model=name [npoints=count] [range=semivariogram range] [power=exponential power] [nugget=semivariogram nugget] [sill=semivariogram sill] [max_lag=max_lag]

Parameters:

input=name Name of input site map

outz=name Name of output z value raster map

outvarz=name Name of output z variance raster map

model=type Type of semivariogram model to be used options: power, spherical, exp, log, gaussian

npoints=value Number of interpolation points
Default: 12

range=value Not used with log or power models
Default: 1.0

power=value Only for use with the power model
Default: 1.0

nugget=value The variogram nugget variance
Default: 0.0

sill=value The variogram sill scaled to the sample variance
Default: 1.0

max_lag=value The max_lag is only used with the power model
Default: 1000.0

NOTE

You have to play with the parameters to get a reasonable result and not a segmentation violation

AUTHORS

Chris Skelly, School of Earth Sciences, Macquarie University, North Ryde 2109 NSW Australia

Darrel McCauley

s.surf.rst

NAME

s.surf.rst - interpolation and topographic analysis from given site data to GRASS floating point raster format using regularized spline with tension (this program replaces *s.surf.tps*)
(GRASS Site Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.surf.rst

s.surf.rst help

s.surf.rst [-dt] input = name [elev = name] [elatt=val] [slope = name] [aspect = name] [pcurv = name] [tcurv = name] [mcurv = name] [maskmap = name] [dmin = val] [zmult = val] [tension = val] [smooth = val] [smatt=val] [segmax = val] [npmin = val] [devi = name] [treefile = name] [overfile = name]

DESCRIPTION

This program interpolates the z_i -values from point data (e.g., elevations, climatic stations, drill holes, etc.) given in a sites file named input as $x|y|z_1 z_2 \dots z_i \dots$ to grid cells in the output raster file elev representing surface. The user can select which floating point attribute will be interpolated by setting the parameter elatt to value i for the i-th floating point attribute.

As an option, simultaneously with interpolation, topographic parameters slope, aspect, profile curvature (measured in the direction of steepest slope), tangential curvature (measured in the direction of a tangent to contour line) or mean curvature are computed and saved as raster files specified by the options slope, aspect, pcurv, tcurv, mcurv respectively. If *-d* flag is set the program outputs partial derivatives $f_x, f_y, f_{xx}, f_{yy}, f_{xy}$ instead of slope, aspect, profile, tangential, and mean curvatures respectively.

User can define a raster file named maskmap, which will be used as a mask. The interpolation is skipped for cells that have zero or NULL value in mask. Data points are checked for identical points and the points that are closer to each other than the given dmin are removed. Parameter zmult allows user to rescale the z-values for sites (useful e.g. for transformation of elevations given in feet to meters, so that the proper values of slopes and curvatures can be computed).

Regularized spline with tension and smoothing is used for interpolation and approximation. The tension parameter tunes the character of the resulting surface from thin plate to membrane. For noisy data, it is possible to define either a constant smoothing parameter smooth which must be a positive number or a variable smoothing parameter by setting the parameter smatt to value j for the j-th floating point attribute representing the smoothing parameter for each point. The resulting surface passes exactly through the data points that have smoothing parameter zero. When smoothing is used, it is possible to output site file devi containing deviations of the resulting surface from the given data.

If the number of given points is greater than segmax, segmented processing is used. The region is split into rectangular segments, each having less than segmax points and interpolation is performed on each segment of the region. To ensure smooth connection of segments the interpolation function for each segment is computed using the points in given segment and the points in its neighborhood that are in the rectangular window surrounding the given segment. The number of points taken for interpolation is controlled by npmin, the value of which must be larger than segmax. User can choose to output vector files treefile and overfile which represent quad tree used for segmentation and overlapping neighborhoods from which additional points for interpolation on each segment were taken. The program writes several

important values to history file of raster map elev. If the input data have time stamp, the program creates time stamp for all output files.

The user must run *g.region* before the program to set the region and resolution for interpolation.

OPTIONS

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

```
s.surf.rst [-dt] input = name elev = name [elatt=val] [slope = name] [aspect = name] [pcurv = name] [tcurv = name] [mcurv = name] [maskmap = name] [dmin = val] [zmult = val] [tension = val] [smooth = val] [smatt=val] [segmax = val] [npmin = val] [devi = name] [treefile = name] [overfile = name]
```

Alternately, the user can simply type *s.surf.rst* on the command line without program arguments. In this case, the user will be prompted for parameter values and flag settings using the standard GRASS *parser* interface described in the manual entry for *parser*.

Flags:

-d Output partial derivatives instead of aspect, slope and curvatures.

-t Use dnorm independent tension.

Parameters:

input=name Use the existing site file name as input.

elev=name Output elevation values to raster file name.

elatt=val Order of floating point site attribute to use for interpolation (1=first).
Options (1-100)
Default is 1

slope=name Output slope or dx values to raster file name.

aspect=name Output aspect or dy values to raster file name.

pcurv=name Output profile curvature or dxx values to raster file name.

tcurv=name Output tangential curvature or dyy values to raster file name.

mcurv=name Output mean curvature or dxy values to raster file name.

maskmap=name Use the existing raster file name as a mask.

dmin=val Set min distance between points to val.
Default value is set to 0.5 grid cell size.

zmult=val Convert z-values using conversion factor val.
Default value is 1.

tension=val Set tension to val.
Default value is 40, appropriate for smooth surfaces at scales 1-100m distances/cell size.

smooth=val Set smoothing parameter to val. Default value is 0.0.

smatt=val Order of floating point attribute to use for variable smoothing parameter

segmax=val Set max number of points per segment to val. Default value is 40.

npmin=val Set min number of points for interpolation to val. Default value is 200, for data with heterogeneous spatial distribution higher value is suggested (see notes).

devi=name Output deviations to a site file name. (Used for smoothed surfaces)

treefile=name Output quad tree used for segmentation to vector file name

overfile=name Output overlapping neighborhoods used for segmentation to vector file name.

NOTES

s.surf.rst uses regularized spline with tension for interpolation from point data. Point data should be in a new site format, that means format *x|y|%z₁ %z₂ ...%z_i ...*, instead of the old format which confused categories with values (*x|y|#z*). If program detects the old format it will allow users to have the site file rewritten to a new format automatically.

The implementation has a segmentation procedure based on quadtrees, which enhances the efficiency for large data sets. The GRASS5.0 version has enhanced segmentation, which takes more points for the large segments, to reduce the potential for visibility of segments in areas with sparse data.

Special color tables are created by the program for output raster files.

Topographic parameters are computed directly from the interpolation function so that the important relationships between these parameters are preserved. The equations for computation of these parameters and their interpretation is described in (Mitasova and Hofierka 1993). Slopes and aspect are computed in degrees (0-90 and 1-360 respectively). The aspect raster file has value 0 assigned to flat areas (with slope less than 0.1%) and to singular points with undefined aspect. Aspect points downslope and is 90 to the North, 180 to the West, 270 to the South and 360 to the East, the values increase counterclockwise. Curvatures are positive for convex and negative for concave areas. Singular points with undefined curvatures have assigned zero values.

Tension and smoothing allow user to tune the surface character. For most landscape scale applications the default should work fine. The program gives warning when significant overshoots appear in the resulting surface and higher tension or smoothing should be used. While it is possible to automatize the selection of suitable tension and smoothing, it has not been done yet, so here are some hints which may help to choose the proper parameters if the results look "weird". It is useful to know that the method is scale dependent and the tension works as a rescaling parameter (high tension "increases the distances between the points" and reduces the range of impact of each point, low tension "decreases the distance" and the points influence each other over longer range). Surface with tension set too high behaves like a membrane (rubber sheet stretched over the data points) with peak or pit ("crater") in each given point and everywhere else the surface goes rapidly to trend. If digitized contours are used as input data, high tension can cause artificial waves along contours. Lower tension and higher smoothing is suggested for such a case. Surface with tension set too low behaves like a stiff steel plate and overshoots can appear in areas with rapid change of gradient and segmentation can be visible. Increase tension should solve the problems.

There are two options how tension can be applied in relation to *dnorm* (*dnorm* rescales the coordinates depending on the average data density so that the size of segments with *segmax=40* points is around 1 - this ensures the numerical stability of the computation):

1. Default (used also in *s.surf.tps*): the given tension is applied to normalized data (x/dnorm.), that means that the distances are multiplied (rescaled) by tension/dnorm. If density of points is changed, e.g., by using higher dmin, the dnorm changes and tension needs to be changed too to get the same result. Because the tension is applied to normalized data its suitable value is usually within the 10-100 range and does not depend on the actual scale (distances) of the original data (which can be km for regional applications or cm for field experiments).
2. Flag *-t* (experimental for *s.surf.rst*): The given tension is applied to un-normalized data (rescaled tension = tension*dnorm/1000 is applied to normalized data (x/dnorm) and therefore dnorm cancels out) so here tension truly works as a rescaling parameter. For regional applications with distances between points in km the suitable tension can be 0.1 or smaller, for detailed field scale analysis with distances in cm it can be 500 or more. To help select how much the data need to be rescaled the program writes dnorm and rescaled tension=tension*dnorm/1000 at the beginning of the program run. This rescaled tension should be around 20-30. If it is lower or higher, the given tension parameter should be changed accordingly.

The default is a recommended choice, however for the applications where the user needs to change density of data and preserve the interpolation character the *-t* flag can be helpful.

For data with values changing over several magnitudes (sometimes the concentration or density data) it is suggested to interpolate the log of the values rather than the original ones.

The program checks the numerical stability of the algorithm by computing the values in given points. The root mean square deviation (rms) between interpolated and given values is written into the history file of raster map elev. For computation with smoothing set to 0, the rms should be 0. Significant increase in tension is suggested if the rms is unexpectedly high for this case. With smoothing parameter greater than zero the surface will not pass exactly through the data points and the higher the parameter the closer the surface will be to the trend. The rms then represents a measure of smoothing effect on data. More detailed analysis of smoothing effects can be performed using the output deviations option and running *s.univar* on the site file with deviations.

The program writes the values of parameters used in computation into the comment part of history file elev as well as the following values which help to evaluate the results and choose the suitable parameters: minimum and maximum z values in the data file (zmin_data, zmax_data) and in the interpolated raster map (zmin_int, zmax_int), rescaling parameter used for normalization (dnorm), which influences the tension.

When the number of points in a site file is not too large (less than 800), the user can skip segmentation by setting segmax to the number of data points or segmax=700.

The program gives warning when user wants to interpolate outside the rectangle given by minimum and maximum coordinates in site file, zoom into the area where the points are is suggested in this case.

When a mask is used, the program takes all points in the given region for interpolation, including those in the area that is masked out, to ensure proper interpolation along the border of the mask. It therefore does not mask out the data points, if this is desirable, it must be done outside *s.surf.rst* (e.g. using *r.mask.points*).

For examples of applications see <http://www.cecer.army.mil/grass/viz/VIZ.html> and <http://www2.gis.uiuc.edu:2280/modviz/>

SEE ALSO

r.slope.aspect, *r.surf.idw*, *r.surf.idw2*, *r.surf.contour*, *s.surf.idw*, *v.to.sites*, *g.region*, *r.mask*, *s.surf.tps*, *v.surf.rst*, *r.resamp.rst*

AUTHORS

Original version of program (in FORTRAN) and GRASS enhancements:

Lubos Mitas, NCSA, University of Illinois at Urbana-Champaign, Illinois, USA

Helena Mitasova, Department of Geography, University of Illinois at Urbana-Champaign, Champaign, Illinois, USA

Modified program (translated to C, adapted for GRASS, new segmentation procedure):

Irina Kosinovsky, US Army CERL, Champaign, Illinois, USA

Dave Gerdes, US Army CERL, Champaign, Illinois, USA

Modifications for new sites format and timestamping:

Darrel McCauley, Purdue University, West Laffayette, Indiana, USA

REFERENCES

Mitas, L., Mitasova, H., 1999, Spatial Interpolation. In: P. Longley, M.F. Goodchild, D.J. Maguire, D.W. Rhind (Eds.), *Geographical Information Systems: Principles, Techniques, Management and Applications*, Wiley, pp.481-492

Mitasova H., Mitas L., Brown W.M., D.P. Gerdes, I. Kosinovsky, Baker, T.1995, Modeling spatially and temporally distributed phenomena: New methods and tools for GRASS GIS. *International Journal of GIS*, 9 (4), special issue on Integrating GIS and Environmental modeling, 433-446.

Mitasova H. and Mitas L. 1993: Interpolation by Regularized Spline with Tension: I. Theory and Implementation, *Mathematical Geology* 25, 641-655.

Mitasova H. and Hofierka L. 1993: Interpolation by Regularized Spline with Tension: II. Application to Terrain Modeling and Surface Geometry Analysis, *Mathematical Geology* 25, 657-667.

Mitasova, H., 1992 : New capabilities for interpolation and topographic analysis in GRASS, *GRASSclippings* 6, No.2 (summer), p. 13.

Mitas, L., Mitasova H., 1988 : General variational approach to the interpolation problem, *Computers and Mathematics with Applications* 16, p. 983

Talmi, A. and Gilat, G., 1977 : Method for Smooth Approximation of Data, *Journal of Computational Physics*, 23, p.93-123.

Wahba, G., 1990, : *Spline Models for Observational Data*, CNMS-NSF Regional Conference series in applied mathematics, 59, SIAM, Philadelphia, Pennsylvania.

s.surf.tps

NAME

s.surf.tps - Interpolates and computes topographic analysis from given site data to GRASS raster format using spline with tension (integer version from GRASS4.1, use *s.surf.rst* for floating point in GRASS5.0). (GRASS Raster Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.surf.tps

s.surf.tps help

s.surf.tps input = name elev = name [slope = name] [aspect = name] [pcurv = name] [tcurv = name] [mcurv = name] [maskmap= name] [dmin1 = val] [zmult = val] [tension = val] [smooth = val] [segmax = val] [npmin = val]

DESCRIPTION

s.surf.tps program interpolates the values to grid cells from point data (digitized contours, climatic stations, drill holes, etc.) given in a sites file named input. The output raster file is elev. As an option, simultaneously with interpolation, topographic parameters slope, aspect, profile curvature (measured in the direction of steepest slope), tangential curvature (measured in the direction of a tangent to contour line) or mean curvature are computed and saved as raster files as specified by the options slope, aspect, pcurv, tcurv, mcurv respectively.

User can define a raster file named maskmap, which will be used as a mask. The interpolation is skipped for cells which have zero value in mask. Zero values will be assigned to these cells in all output raster files.

Data points are checked for identical points and points that are closer to each other, then the given dmin1 are removed (this is necessary especially for digitized contours). Parameter zmult allows the user to rescale the z-values for sites (useful, e.g., for transformation of elevations given in feet to meters, so that the proper values of slopes and curvatures can be computed).

Regularized spline with tension is used for the interpolation. The tension parameter tunes the character of the resulting surface from thin plate to membrane. Higher values of tension parameter reduce the overshoots that can appear in surfaces with rapid change of gradient (see suggested values for different types of surfaces given in notes). For noisy data, it is possible to define a smoothing parameter, smooth. With the smoothing parameter set to zero (smooth=0), the resulting surface passes exactly through the data points.

If the number of given points is greater than 400, segmented processing is used. The region is split into rectangular segments, each having less than segmax points and interpolation is performed on each segment of the region. To ensure the smooth connection of segments the interpolation function for each segment is computed using the points in a given segment and the points in its neighborhood. The minimum number of points taken for interpolation is controlled by npmin, the value of which must be larger than segmax and less than 400. This limit of 400 was selected to ensure the numerical stability and efficiency of the algorithm. The program writes important values related to the computation to the history file of raster map elev.

OPTIONS

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

```
s.surf.tps input=name elev=name [slope=name] [aspect=name] [pcurv=name] [tcurv=name]  
[mcurv=name] [maskmap=name] [dminl=val] [zmult=val] [tension=val] [smooth=val]  
[segmax=val] [npmin=val]
```

Alternatively, the user can simply type *s.surf.tps* on the command line without program arguments. In this case, the user will be prompted for parameter values and flag settings using the standard GRASS *parser* interface.

Parameters:

- input=name* Use the existing site file name as input.
- elev=name* Output elevation values to raster file named *name*.
- slope=name* Output slope values to raster file named *name*.
- aspect=name* Output aspect values to raster file named *name*.
- pcurv=name* Output profile curvature values to raster file named *name*.
- tcurv=name* Output tangential curvature values to raster file named *name*.
- mcurv=name* Output mean curvature values to raster file named *name*.
- maskmap=name* Use the existing raster file name as a mask.
- dminl=val* Set min distance between points to *val*.
Default value is set to 0.5 grid cell size
- zmult=val* Convert z-values using conversion factor *val*.
Default value is 1
- tension=val* Set tension to *val* .
Default value is 40, appropriate for smooth surfaces
- smooth=val* Set smoothing parameter to *val* .
Default value is 0, no smoothing is performed
- segmax=val* Set max number of points per segment to *val*.
Default value is 40
- npmin=val* Set min number of points for interpolation to *val*.
Default value is 150, for data with heterogeneous spatial distribution higher value is suggested

NOTES

LINUX

There are some peculiar differences between *s.surf.tps* for LINUX and for SUN.

- The usage of the same parameters is causing a segmentation violation on LINUX, on SUN *s.surf.tps* is running well - e.g. the interpolation of the heroldsberg elevation maps was using on SUN a temporary space of 170Mb, on LINUX approx. 80Mb.

Please play with the parameters for a reasonable result.

s.surf.tps uses regularized spline with tension for interpolation from point data. The implementation has an improved segmentation procedure based on quadtrees, which enhances the efficiency for large data sets. Special color tables are created by the program for output raster files.

Topographic parameters are computed directly from the interpolation function so that the important relationships between these parameters are preserved. The equations for computation of these parameters and their interpretation is described in Mitasova and Hofierka, 1993. Slopes and aspect are computed in degrees (0-90 and 1-360 respectively). The aspect raster file has value 0 assigned to flat areas (with slope less than 0.1%) and to singular points with undefined aspect. Aspect points downslope and is 90 to the North, 180 to the West, 270 to the South and 360 to the East, the values increase counterclockwise. Curvatures are positive for convex and negative for concave areas. Original values of curvatures are multiplied by 100000, to conform with GRASS integer raster files. Therefore any curvature lower than 0.00001 will be zero. Flat areas have zero curvatures and singular points have codes 1000000, 2000000, 3000000 for peak, pit and saddle respectively. We suggest to use these codes only to distinguish areas (grid cells) with undefined curvature because the codes are assigned using the theorems from differential geometry but have never been tested.

The program gives warning when significant overshoots appear and higher tension should be used. However, with tension too high the resulting surface changes its behavior to membrane (rubber sheet stretched over the data points resulting in a peak or pit in each given point and everywhere else the surface goes rapidly to trend). Smoothing can also be used to reduce the overshoots if the resulting surface should be smooth.

For data with values changing over several magnitudes (sometimes the concentration or density data) it is suggested to interpolate the log of the values rather than the original ones.

The program checks the numerical stability of the algorithm by computation of values in given points, and prints the maximum difference found into the history file of raster map elev. Significant increase in tension is suggested if the difference is unacceptable. For computation with smoothing set to 0 this difference should be 0. With smoothing parameter greater than zero the surface will not pass through the data points and the higher the parameter the closer the surface will be to the trend. The maximum difference between the given and approximated value in this case reflects the smoothing effect on data. For theory on smoothing with splines and their statistical interpretation see Talmi and Gilat 1977, Wahba 1990, and Hutchinson 1992, where you can find also a comparison of smoothing splines with kriging.

The program writes the values of parameters used in computation into the comment part of the history file elev as well as the following values which help to evaluate the results and choose the suitable parameters: minimum and maximum z values in the data file (*zmin_data*, *zmax_data*) and in the interpolated raster map (*zmin_int*, *zmax_int*), maximum difference between the given and interpolated z value in a given point (*errtotal*), rescaling parameter used for normalization (*dnorm*), which influences the tension (see Mitasova, 1992; Mitasova and Mitas, 1993).

If a visible connection of segments appears, the program should be rerun with higher *npmin* to get more points from the neighborhood of the given segment.

If the number of points in a site file is less then 400, *segmax* should be set to 400 so that segmentation is not performed when it is not necessary.

The program gives a warning when the user wants to interpolate outside the rectangle given by the minimum and maximum coordinates in the site file, zooming into the area where the points are is suggested in this case.

When a mask is used, the program takes all points in the given region for interpolation, including those in the area which is masked out, to ensure proper interpolation along the border of the mask. It therefore does not mask out the data points; if this is desirable, it must be done outside *s.surf.tps*.

The program was used for various applications with the following parameters :

"interpolation of DEM from digitized contours"

```
tension      20. - 80.  
smoothing    0.01 - 1.0  
segmax       40  
npmin 200 - 300
```

(low tension was used for relatively flat terrain, high tension was necessary for terrain with sharp changes in slope, low value of smoothing is usually sufficient for dense and accurately digitized contours, for less dense and not very carefully digitized contours, higher smoothing is suggested)

"interpolation of precipitation from climatic stations"

```
tension      40. - 150.  
smoothing    0. - 2.  
segmax       40  
npmin 200
```

"interpolation of concentration of chemicals"

```
tension      20. - 60.  
smoothing    0.5 - 5.0
```

The user must run *g.region* before the program to set the region for interpolation.

SEE ALSO

v.to.sites, g.region, r.surf.contour, r.surf.idw, r.surf.idw2, s.surf.idw, s.surf.rst

AUTHOR

Original version of program (in FORTRAN):

Helena Mitasova, Illinois Natural History Survey and US Army CERL, Champaign, Illinois Comenius University, Bratislava, Czechoslovakia

Lubos Mitas, Department of Physics, University of Illinois at Urbana Champaign, Illinois Institute of Physics, Bratislava, Czechoslovakia

Modified program (translated to C, adapted for GRASS , segmentation procedure):

Irina Kosinovsky, U.S. Army Construction Engineering Research Laboratory

Dave Gerdes, U.S. Army Construction Engineering Research Laboratory

REFERENCES

Hutchinson, M. K. and Gessler, P. E., 1992. Splines: More than just a smooth interpolator, Geoderma.

- Mitasova, H. and Mitas, L., 1993. Interpolation by regularized spline with tension: I. Theory and implementation, *Mathematical Geology*.
- Mitasova, H. and Hofierka, L., 1993. Interpolation by regularized spline with tension: II. Application to terrain modeling and surface geometry analysis, *Mathematical Geology*.
- Mitasova, H., 1992. New capabilities for interpolation and topographic analysis in GRASS, *GRASS Clippings*, v. 6, No.2 (summer), p 13.
- Mitasova, H., 1992. Surfaces and modeling, *GRASS Clippings*, v. 6, No.3 (winter), pp 16-18.
- Talmi, A. and Gilat, G., 1977. Method for smooth approximation of data, *Journal of Computational Physics*, 23, pp 93-123.
- Wahba, G., 1990. Spline models for observational data, *CNMS-NSF Regional Conference series in applied mathematics*, 59, SIAM, Philadelphia, Pennsylvania.

s.sv

NAME

s.sv - Sample semivariogram of a GRASS sites list.
(GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.sv

s.sv help

s.sv [-alq] [sites=name] lag=value [lagtol=value] [direction=value] [angtol=value] [graph=name]

DESCRIPTION

s.sv calculates a sample semivariogram and either plots it or writes it to standard output.

OPTIONS

Flags:

-q Quiet. Cut out the chatter.

-p Plot the sample semivariogram in the GRASS graphics window (requires *g.gnuplot*).

Parameters:

sites=name Name of an existing sites file. Default is standard input with no field separators.

lag=value Nominal lag distance.

lagtol=value Tolerance on lag distance. Default is half of nominal distance.

direction=value Direction of semivariogram. Default is omnidirectional semivariogram.

angtol=value Angular tolerance on direction.

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. If unspecified, semivariogram is written to standard output.

NOTES

Without the *-p* flag, three columns of data are written to standard output: lag distance (h), semivariogram value (gamma), and the number of data pairs used to compute it (N(h)). When the graph parameter is set, these same three columns of data are written to name.dat. Therefore, to replot a sample semivariogram, use:

```
g.gnuplot name.gp
```

To plot a histogram of N(h), simply edit name.gp and redo the previously given command.

SEE ALSO

s.univar, *s.normal*, *g.gnuplot*, *m.svfit*

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

BUGS

Will not work correctly with lat-long data. Should `G_azimuth()` be used to calculate the angle between points?

Only Matheron's classical estimator is available with *s.sv*. Others may be added in the future.

AUTHOR

James Darrell McCauley, Agricultural Engineering, Purdue University

s.to.rast

NAME

s.to.rast - Create a raster map from site list
(GRASS Sites Program)

SYNOPSIS

s.to.rast

s.to.rast help

s.to.rast [*-qs*] *input=name output=name [size=value] [title=name]*

GRASS VERSION

4.x, 5.x

DESCRIPTION

s.to.rast creates a raster map representation of a site list in your GRASS mapset. (The program actually is a non-interactive subset of the functionality of the *s.menu* program.) A site list is a list of eastings and northings describing the location of some point feature. It can also contain a category value and category label for each site location. Once created, the raster map created from the site list can be used with other raster map layers in further analyses. Allowing the user to create a raster map layer of sites opens up the full analysis capabilities for site data that are available for raster map layers within GRASS. The number of categories present in the new raster map layers will depend on the format of your site list file (see *sites.format*). A multi-valued (more than one category) raster map layer representation of a site list can be created with a site list in the format: *.in +2 "E|N|" "#n label" .in -2* where E is the Easting, N is the Northing, and #n label is the description field. The description field consists of a pound sign # followed by the category value n to be associated with the site's cell location, and the category label for n. If the user does not include a description field starting with *.in +2 #n .in -2* beside the Easting and Northing on every line in the sites list file, a single-valued raster map layer will be created instead. In the single-valued raster file, each site will be represented as the category value 1. Non-site cells will be coded as category value 0. The creation of a single-valued map can be forced - see the *-s* flag below. *.sp .25* Note that only sites within the current geographic region will be considered. However, if the size of the sites is more than one cell (3x3, 5x5, etc.) and a site lies near an edge of the geographic region, some of the cells for the site may fall outside the geographic region. These cells will not appear in the raster map layer, and the site will no longer be 3x3 or 5x5 but will be clipped to fit the geographic region.

Flags:

-q Run quietly. (The default is verbose mode).

-s Force the creation of a (0/1) raster map. Thus, even if the site file contains properly formatted description fields denoting category values, each site will be represented in the resulting raster map as the category 1; non-sites will be 0 (no data).

Parameters:

input=name Name of site list in the mapset to use as input.

output=name Name of raster map to be created in user's mapset.

size=value This option can be used to set the number of cells to represent a site. The minimum is one cell per site (e.g. the default value of 0). The alternatives are squares (which can be visualized as "layers" of cells) around the cell at the site, such as 3x3 (*size=1*), 5x5 (*size=2*), 7x7 (*size=3*), etc.

title=name Title to be used when creating the new raster map. If not specified, the program will come up with a default title.

NOTE

Be aware that duplicate sites or sites that exist close to each other may not appear as separate values in the raster map that is created. In other words, if more than one site is within the confines of one raster grid cell, only the last site read from the list in the given cell will be represented. If this is a potential problem, use the *s.menu* programs to check for multiple occurrences. Also note that if a size parameter of 1 or greater is used, the site areas created may have an overlapping effect, with sites found toward the end of the site file ending up "on top" of those they overlap.

SEE ALSO

d.sites, s.menu, s.in.ascii, s.out.ascii, s.db.rim, s.surf.idw, s.surf.tps, v.to.sites, r.support sites.format, sites.report.

AUTHORS

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

s.to.vect

NAME

s.to.vect - Converts a GRASS site_lists file into a vector file.
(GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.to.vect

s.to.vect help

s.to.vect input=name output=name

DESCRIPTION

s.to.vect converts GRASS site_lists file into vector files. The resulting vector file can be treated as any other vector file. The requirements of the site_lists file are standard (i.e., a regular site_lists file format is required). site_lists file values are used as dig_cats category values.

OPTIONS

Parameters:

input=name Name of input GRASS <KBD>site_lists</KBD> file to be converted.

output=name Name to be assigned to the vector output file.

AUTHOR

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

s.univar

NAME

s.univar - Univariate statistics for a GRASS sites list.
(GRASS Sites Program)

GRASS VERSION

4.x, 5.x

SYNOPSIS

s.univar

s.univar help

s.univar [-aglq] sites=name

DESCRIPTION

s.univar calculates univariate statistics of sites lists. This includes the number of sites, mean, standard deviation, coefficient of variation, minimum, first quartile, median, third quartile, and maximum.

OPTIONS

Flags:

-a Use all sites found in the named sites file, rather than limiting output to sites falling within the current geographic region.

-g Print the statistics in shell script style (similar to *g.region*).

-l Use site descriptions (category labels) for z values. Sites must be in the (E|N|#n desc) format for this to work. Default format is just three pipe-separated floating point numbers (E|N|desc).

-q Quiet. Cut out the chatter.

Parameters:

sites=name Name of an existing sites file.

SEE ALSO

s.normal

BUGS

Needs to do as many calculations as possible without storing individual site records (requires a lot of memory).

AUTHOR

James Darrell McCauley, Agricultural Engineering, Purdue University