Development of the GRASS/R interface - GIS and statistical data analysis*

Roger Bivand

September 2002

Outline

- Interfacing R and GRASS: data analysis, statistics, spatial statistics
- Interface status seen from GRASS
- Interface status seen from R
- Interface prospects: GDAL, connections, Postgresql/ODBC
Geographical Information System

Other geographical information systems

User interface

External statistical and modelling packages

Display and product generator

Data manipulation and analysis

Data storage retrieval and database management

Data input processing

Geographical Information System

Input

Data from maps

Census data

Field survey data, GPS

Remote sensing data

Data from other digital sources

Output

Photographic products

Maps

Statistics

Reports

Data inputs to models

INTERFACE BETWEEN GIS AND STATISTICS ENVIRONMENT

Input

Data from maps

Census data

Field survey data, GPS

Remote sensing data

Data from other digital sources

Output

Photographic products

Maps

Statistics

Reports

Data inputs to models
Interfacing R and GRASS: data analysis, statistics, spatial statistics

- Spatial data analysis
- GIS data models and their mapping into R objects
- Running R within GRASS: initial approach
- Building the raster interface on the GRASS libgis.a library
Spatial data analysis

• Both exploratory and confirmatory

• Fields/surfaces, point patterns, and lattice/area data

• Substantial literature, but widely spread and inconsistent implementations

• Some R packages cover fields/surfaces, point patterns, areal/lattice data (spatial, sgeostat, geoR, geoRglm, akima, fields, RandomFields, ash, tripack, splancs, spatstat, spdep)
GIS data models

- Spatial reference systems are used to register different data in the same framework

- Geographical metadata includes projection, ellipsoid, and datum, as temporal metadata includes series origin and time zone

- Spatial data regarding position are subject to measurement error and to numerical artifacts as a result of transformation

- The two major data models: raster and vector, handle resolution differently; neither handle “crispness” well
The R/GRASS interface

- Initial insight that R can be run from the GRASS prompt, which is a standard shell prompt including GRASS environment variables.

- GRASS is a GPL’ed development of a US Army raster GIS with vector capabilities; GRASS 5.0 beta supports NULL/NA and floating point raster values.

- First versions used intermediate text files, later versions added calls to the GRASS libgis.a to C functions called from R.

- At present, the interface supports raster and sites data, but not vector.
Underlying operating system
System shell and environment (csh, ksh, bash)
GRASS environment and location
R interactive session
`system("g.region −p")`
`system("psql midwest")`
`.Call("gmeta")`
dynamically loaded modules
Interface status seen from GRASS

- Project locations can contain raster data of varying extent and resolution, vector and site data, all in a single projection.

- The view of the location is chosen by the current window, which also sets the resolution of the raster data, resampling as required on raster data access.

- Most layers, including those with data values, are accompanied by attributes such as colour tables, category labels.
Data should look like GRASS data

- The current window should determine the way in which raster data are retrieved and transferred; this should also apply to vector/site data with regard to extent

- Data moved to GRASS over the interface should be furnished with attributes expected by GRASS, including category labels and colours

- These criteria are important when the interface is used in BATCH mode to supplement GRASS shell scripts
Interface status seen from R

- R offers access to a broad range of techniques for data analysis, visualization, modelling and simulation

- R is advancing in memory handling, but does require data being analysed to be held in memory unless special steps are taken to subset external data sources

- R graphics are vector-based, so that raster data are displayed using filled rectangles on screen and hard-copy — not ideal for large raster layers

- R needs a unifying package for handling vector data
Data objects should look like R objects

- R objects can have similar — but not the same — names as GRASS files (similar issue as in `r.mapcalc`)

- R objects typically have one or more class attributes, and in the future may be required to have one such class attribute (to enable data-driven analysis)

- Classes should be provided with implementations of generic functions for those classes

- Very often objects being analysed are expected to be data vectors, often assembled in data frames, rather than say 2D arrays with multiple bands
Alternative ways to handle metadata attributes

- At present, metadata are extracted once on the basis of the current window, and used in relation to all interface functions, including wrapper functions for interpolation.

- They could be attached to each object in a class-based capsule, which could then be copied to new objects created from them.

- The metadata content should probably be extended to cover both window, and projection/ellipsoid/datum, but it is not clear how much data integration should take place on which side of the interface.
Interface prospects: connections, GDAL, databases

- The `pixmap` package was written using connections mechanisms to permit portable anymaps (PNM) to be read without linking against an external library.

- Timothy Keitt has published an early version of wrappers for R to Frank Warnerdam’s GDAL library — if GDAL is compiled with GRASS support, this also lets us access GRASS raster data in R.

- Access to databases has been advanced in R, using generic and database-specific mechanisms, including `DBI`, a successor to Timothy Keitt’s `RPgSQL`.
Connections

- Connections provide functions that can be generalised to read and write files, sockets, pipes, and URLs, both for text and binary data.

- In a forthcoming version, the present interpreted version of the interface will be moved from temporary text files to temporary binary files.

- Metadata can also be transferred using connections.

- Connections could be run on a GRASS database without running GRASS itself — this may not be an advantage.
GDAL

- The R/GDAL interface is a promising way of accessing raster data of many kinds, not just GRASS — it is based on GDAL, which has to be installed first.

- The interface can be extended to handle OGR vector data.

- The R/GDAL interface can be run on a GRASS database without running GRASS itself — this may not be an advantage (see `deleteDataset()` which works, even for raster layers opened read-only!).

- `DBI` and `GDAL` use new-style classes in the `methods` package.
Where now?

- Interpreted mode interface to change to connections mechanisms

- Interface should migrate to new-style classes

- This raises the question of implementing stiffer object structures, probably multiple structures, and helper functions to move from one object class to another

- The interface needs to be extended to vector data (but should GRASS be needed to import polygons just for display? Does R need topology? Line thinning?)

- What about more demanding data structures (3D with time — R supports dates and times)?