Integrating GRASS 5.0 and R: GIS and modern statistics for data analysis

Roger S. Bivand

Department of Geography, Norwegian School of Economics and Business Administration, Breiviksveien 40, N-5045 Bergen, Norway, Roger.Bivand@nhh.no.

Abstract. With the release of the open-source GIS GRASS 5.0 in early 1999, opportunities are presented for integration with the open-source R statistical data analysis programming environment. After reviewing these two software systems, an example is given of the advantages yielded by the complementing of GIS techniques with modern statistical analysis. The example shows how GTOPO30 digital elevation models, with a resolution of 30 seconds, may be subjected to geomorphometric analysis; the data are taken from the Kosovo region. In these examples, R is run interactively within the GRASS 5.0 environment, transfering data by writing and reading text files; the operating system is Linux.

1 Introduction

Development of the leading Open Source GIS — GRASS — has been moved to Baylor University in Texas, where work on a new release incorporating floating-point raster cell values and NULL values different from zero is now in beta testing. In parallel with this, the R statistical and data analysis language, also Open Source, is maturing very rapidly, and can now execute most S and S-PLUS code in an unmodified form. In the past, when S was available on academic license, integration between GRASS and S existed in a loose-coupled form for integer raster cell values sampled at points given in a site layer.

The issues involved in linking two complex and fast-changing program environments are presented in a comprehensive way, with particular reference to the spatial analysis of data stored in the chosen GIS. While the progress reported in this paper is based on Open Source Unix-like operating systems, begun under NetBSD 1.3, and concluded under Linux 2.0.36 (RedHat 5.2), it is worth noting that both GRASS and R have been compiled for MS Windows systems. A third software package used for data integration here is Generic Mapping Tools (GMT).

In work to date, the interface used is that of the statistical analysis system, run from within the GIS environment. Given major design differences in memory management — GRASS uses the underlying file system, while R maps all active objects into memory managed by a garbage collector — and other problems, it has been necessary to decide on a representation suiting the data analysis and visualization tasks being performed. This means here that the statistical programming environment is run from within GRASS, permitting GRASS command line instructions, including those requiring interaction, to be issued from within R using the `system()` function (in the code

examples, $ is the operating system shell prompt, GRASS> is the GIS prompt, and > is the R prompt; the + sign is used for line continuation permitted in R , while * is used where R does not permit command lines to be broken within text strings):

```
$ grass5.0
Welcome to GRASS 5.0beta (Feb 1999)

Geographic Resources Analysis Support System (GRASS) is a Trademark
of U.S. Army Construction Engineering Research Laboratories (USACERL)
and Baylor University.
New releases of GRASS are coordinated and produced by the Center for
Applied Geographic and Spatial Research (CAGSR) located at Baylor
University.
...
When ready to quit enter:           exit
GRASS> g.gisenv LOCATION_NAME
kosovoutm34
GRASS> g.list type=rast mapset=rsb
----------------------------------------------
raster files available in mapset rsb:
D            PLAN         bordermask1  kPLAN100     topo.z3      z11
E            PLAN100      hbar         kPROF100     topo.z4      z12
...
----------------------------------------------
GRASS> R
R : Copyright 1999, The R Development Core Team
Version 0.64.0  (April 8, 1999)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
...
Type    "q()" to quit R.

> system("g.gisenv LOCATION_NAME")
kosovoutm34
> lname <- system("g.gisenv LOCATION_NAME", intern=T)
> lname
[1] "kosovoutm34"
> system("g.list type=rast mapset=rsb")
----------------------------------------------
raster files available in mapset rsb:
D            PLAN         bordermask1  kPLAN100     topo.z3      z11
E            PLAN100      hbar         kPROF100     topo.z4      z12
...
----------------------------------------------
> q("no")
GRASS> exit

GRASS SESSION WRAPUP

......

                  Goodbye from GRASS
$
```

Running under Unix-family operating systems, GRASS only customizes the user's program execution environment, adding specific definitions needed for GRASS programs to be able to find the files and metadata required for their work. GRASS does not then represent a major memory overhead, and R can be launched with plenty of space for its computations. The examples reported below did not need more than 16Mb heap memory for analysis of a data set with 26732 raster cells and eight initial attributes, and with the judicious deletion of data objects from the heap, much less would have surficed.

Following a review of GRASS and R in the context of open-source software, an example will be presented. It shows how a combination of GRASS and R can be employed to conduct a rapid geomorphometric analysis of the terrain in Kosovo. Without NULL and floating point raster cell values in GRASS, this would be more complicated, but now seems to function well. GRASS is used for the filtering operations used to construct the terrain indices to be used, while statistical tools in R are deployed to squeeze information out of the data. In particular, modern statistics stress the importance of exploratory and graphical data analysis, functions which GIS are not designed to support. Prior to data import into GRASS, Generic Mapping Tools (GMT) were used for accessing GTOPO30 digital elevation models from two tiles, and for projecting from geographical coordinates to UTM zone 34, to convert position to metre units.

2 Open-source software, GRASS and R

Several years have passed since the advantages of integrating GIS and spatial analysis were described by Bailey (1994), Haining (1994), and Anselin, Dodson and Hudak (1993), among others. It has taken time to define practical solutions, and even then, problems have arisen with changes in underlying operating systems, and in the interfaces used by the software systems to be integrated. Further, it has not always been the case that the interfaces, whether through file transfer, remote procedure calls (RPC), or application programming interfaces (API) have been sufficiently well documented for clean design. These considerations, while not preventing progress — witnessed by work reported by Haining (1996), Anselin and Bao (1996), Bao and Anselin (1997), and Can (1996) and others, do raise the question of access to source code for the software systems being integrated.

One argument for using open-source software is that it is cheaper than commercial alternatives for obvious reasons, but comes with no guarantees, and requires a willingness on the part of the user to commit time to its configuration, possibly compilation, and installation. This is perhaps not the key reason for seeing open-source software as bringing signal advantages to work in analysis and prototyping when routine tasks are seldom encountered (Ousterhout, 1997). The two that are stressed in current discussions are, related to the skills of the user, that there will exist a community (”bazaar”) of other users who most likely will already have met and solved the user’s problem, and that with a large enough bazaar, no problem that needs to be solved is unsolvable. This is termed the parallelizability of debugging, and requires all interested in advancing a given software system to have unrestricted access to the source code. When debugging is spread across many different users and programmers, almost certainly at least one of the participants will have encountered a similar problem before, and be able to point to a diagnosis (Raymond, 1997). Access to source code in the present example made it possible to find out how the revised GRASS `r.in.ascii` command reads NA values, although the manual page does not document this.

While there are several statistical analysis systems, most prominently R and LispStat, with open-source status, the only major geographical information system is GRASS (Geographic Resources Analysis Support System), now based at the Center for Applied Geographic and Spatial Research of Baylor University, Texas (Byars and Clam-

ons, 1998). Attempts to implement selected spatial statistics techniques within GRASS by J. Darell McCauley, reviewed in Bivand (1996), are still extant in the code base, but are now unsupported. Following uncertainty about the future of GRASS after the cessation in 1996 of support from its originating institution, the U.S. Army Construction Engineering Research Laboratory, it seems that the value of an open source GIS, albeit with much better support for raster than vector representations, has been recognized, and that effort is being put into development. GRASS has moved from version 4.1.5, the last CERL release, to 4.2.1, while version 5.0beta from Baylor University was released on 5 February 1999. Until now, GRASS has stored raster cell values as integers only, using the zero value both as numeric zero and as NULL (VOID, not available, NA). Version 5.0 introduces both a separate NULL value, and floating-point raster cell values, both of which are necessary for a viable interface to statistics software.

Turning to `R` , it is possible to see a clearer bazaar-type process than in the case of GRASS (Ihaka, 1998). `R` was envisioned as a programming environment for data analysis and graphics not dissimilar to `S` (Itaka and Gentleman, 1996, see also Becker et al., 1988, Chambers and Hastie, 1992, Becker, 1994, Venables and Ripley, 1997). `R` differs from `S` and its derivative `S-PLUS` by placing its objects in a workspace in memory rather than in separate data files on disk; in this way `S` is more like GRASS. `R` supports functions written by the user — indeed, it is a sophisticated programming language permitting both the user and the wider bazaar community to develop and exchange ideas.

The following example shows how the strengths of the `R` interpreted language can ease the housekeeping of generating the intermediate layers required for quantitative analysis of land surface topography in the fashion of Zevenbergen and Thorne (1987). To generate the coefficients A–I (see also Burrough and McDonnell, 1998, p. 191), single cell shifts are required from each cell to each of its eight neighbours. In the syntax of the GRASS `r.mapcalc` command, this involves appending in square brackets the desired shift (negative for leftward and upward, positive for rightward and downward) to the name of the raster cell layer involved. Running through `R` , we can execute the command with or without interaction:

```
> system("r.mapcalc")

mapcalc> z11 = topo[-1,-1]

EXECUTING z11 = ...  100%
CREATING SUPPORT FILES FOR z11
range: 4 2565.3898925781

mapcalc>
> system("r.mapcalc z11 = topo[-1,-1]")

EXECUTING z11 = ...  100%
CREATING SUPPORT FILES FOR z1
range: 4 2565.3898925781
>
```

We have however eight raster cell layers to create, and can automate the process using `R` . Use is made of the `:` operator, creating a sequence from its first argument to its second with an increment of unity, and then of `for()` loops to change both the name of the calculated raster cell layers, and their shift:

```
> system("g.list rast")
----------------------------------------------
raster files available in mapset rsb:
topo
----------------------------------------------
> x <- -1:1
> x
[1] -1  0  1
> y <- -1:1
> for (i in 1:length(x)) {
+    for (j in 1:length(y)) {
+       string <- paste("r.mapcalc z", i, j, " = topo[",
+          x[i], ",", y[j], "]", sep="")
+       cat(string, "\n")
+       system(string)
+    }
+ }
r.mapcalc z11 = topo[-1,-1]

EXECUTING z11 = ...  100%
CREATING SUPPORT FILES FOR z11
range: 4 2565.3898925781
r.mapcalc z12 = topo[-1,0]

EXECUTING z12 = ...  100%

...

r.mapcalc z33 = topo[1,1]

EXECUTING z33 = ...  100%
CREATING SUPPORT FILES FOR z33
range: 171.4170074463 2565.3898925781
> system("g.list rast")
----------------------------------------------
raster files available in mapset rsb:
topo        z11         z12         z13         z21
z22         z23         z31         z32         z33
----------------------------------------------
```

If this function prototype was appropriately packaged and supplemented by the necessary calls to `r.mapcalc` to compute the coefficients needed for analysis, the whole procedure could be automated. By checking values returned by GRASS programs used, it is possible to ensure that the procedure runs correctly.

In addition to GRASS and `R` , use has been made in the second example of Generic Mapping Tools (GMT, see Wessel and Smith, 1998). In particular, the GTOPO30 tiles for the Kosovo area, crossing the 20°E tile border, were imported using `gmtraster` and `grdpaste`, and converted to the UTM zone 34 projection using `grd2xyz` to write a text file of elevation values for the selected region by geographical coordinates, `mapproject` to convert to UTM zone 34, `blockmedian` and `surface` to interpolate to a 1000m grid spacing, roughly equivalent in W-E resolution to the 30 second input data, and finally `grd2xyz` again to output the data for reading into GRASS. The GMT tool `grdmask` was also used to create the mask for Kosovo, digitized from a 1:1 million thematic map (on which the line thickness varied with type, leading to potential errors of roughly $\pm$ 2000m. The data files output by `grd2xyz` were massaged using `awk` to convert them to a suitable format for GRASS `r.in.ascii`.

The versions of the software used are: Linux kernel 2.0.36 (RedHat 5.2); GRASS 5.0beta, released 5 February 1999, installed from the Linux binary from `http://www.baylor.edu/~grass` — including the installation of a nonstandard library as

mentioned in the installation guide; the R 0.64.0 source distribution of 8 April 1999, downloaded from http://www.ci.tuwien.ac.at/R/, configured, compiled using standard compilers and libraries located by the automatic configuration procedure, installed, and supplemented by a number of contributed packages, in particular MASS and cluster, compiled and installed from source distributions using the R INSTALL library command; and the GMT 3.2 source distribution of 19 March 1999 from http://www.soest.hawaii.edu/gmt, compiled and installed, and supplemented by the compiling and installation of the optional grdraster command. The GTOPO30 data tiles W020N90 and E020N90 were acquired from http://edcwww.cr.usgs.gov/landdaac/gtopo30/gtopo30.html, and installed following instructions given in the archives of the GMT discussion list.

3 Raster data integration

The research problem considered here has two major facets: firstly, to test the integration of GRASS and R with respect to floating point raster cell values and NAs, but not least importantly to use an example of a realistic size and format. In the context of the 1999 Kosovo crisis, the land surface topography of the region came into sharp focus, both as regards the plight of refugees and the conduct of land-based peace enforcement measures. Kosovo is known to pose many problems in this respect, being made up of a number of upland basins largely surrounded by mountain ranges, and draining in three directions: to the Adriatic in the west, to the Aegean to the south-east, and northward towards the Danube. After consulting the literature on quantitative analysis of land surface topography, is was decided to create a selection of indices, to explore their values, and to make a classification (Sulebak, 1997, Guzzetti and Reichenbach, 1994, Brown, Lusch and Duda, 1998, Jones, 1998, Zevenbergen and Thorne, 1987). Since the purpose of this paper is not geomorphometry *sensu stricto*, suffice it to note that Jones (1998) finds that the slope algorithm implicit in the Zevenbergen/Thorne approach outperforms all alternatives on his test data.

Elevation data for a 163 × 164 km region including Kosovo was extracted from GTOPO30 and converted to UTM zone 34 using GMT, and imported into GRASS. Shift layers for the computation of Zevenbergen/Thorne indices, and the indices themselves, were calculated using r.mapcalc in GRASS. The elevation data imported into GRASS were already in floating point format, as were all of the computed indices. The profile and plan curvature indices were scaled to represent curvature per 100m (negative values are concave and positive convex), slope gradient is scaled in dimensionless m/m units (Zevenbergen and Thornes, 1987, p. 50), elevation and local relief (maximum - minimum elevation in a moving 3×3 window) are scaled in metres, and the local elevation-relief ratio (or hypsometric integral — Pike and Wilson, 1971; here taken within a moving 3×3 window) is scaled between zero and unity.

A mask raster cell layer was constructed using the digitized borders of Kosovo, and imported into GRASS from GMT with zero coding cells outside Kosovo, and one coding those within and on the border. The r.reclass command was used to replace the zeros with NULL values, and r.mapcalc to multiply the resulting NULL/1 layer with the indices to be analysed in R . The following display shows the report window

returned by GRASS for the NULL/1 mask layer, indicating that Kosovo makes up about 40% of the selected region. We note that the NULL value is represented by an asterisk.

```
> system("r.report map=border1 units=c,h,k")
r.stats:  100%
+----------------------------------------------------------------------------+
|                         RASTER MAP CATEGORY REPORT                         |
|LOCATION: kosovoutm34                              Thu Apr 29 19:25:49 1999|
|----------------------------------------------------------------------------|
|          north: 253500    east: 283500                                     |
|REGION    south:  90500    west: 119500                                     |
|          res:     1000    res:    1000                                     |
|----------------------------------------------------------------------------|
|MASK:none                                                                   |
|----------------------------------------------------------------------------|
|MAP: (untitled) (border1 in rsb)                                            |
|----------------------------------------------------------------------------|
|             Category Information               | cell|          |  square  |
|#|description                                    |count|  hectares|kilometers|
|----------------------------------------------------------------------------|
|1| . . . . . . . . . . . . . . . . . . . . . . . |10948| 1,094,800|10,948.000|
|*|no data. . . . . . . . . . . . . . . . . . . . |15784| 1,578,400|15,784.000|
|----------------------------------------------------------------------------|
|TOTAL                                          |26732| 2,673,200|26,732.000|
+----------------------------------------------------------------------------+
```

Having masked the elevation layer and five index layers, they were moved to `R` for further analysis. The GRASS `r.stats` command is used with arguments for the additional output of the grid locations for each raster cell, saving a text file with eight blank separated columns. As can be seen from the output of the `head` command, displaying the first few lines of the file, the NULL values are shown again as asterisks. Furthermore, the first cell written to file is the top left cell, running rightwards along the top raster row. The data file is then read in to `R` using the `read.table()` command, specifying the string used to represent NA values. Finally, names are given to the columns of the data table. As can be seen from the output of the `dim()` command, returning the size of the data table, it has 26732 rows and eight columns, as expected:

```
> system("r.stats -1g input=kER,kLR,kPLAN100,kPROF100,kSLOPE,kTOPO
* output=morph.dat")
r.stats:  100%
> system("head morph.dat")
120000 253000 * * * * * *
121000 253000 * * * * * *
122000 253000 * * * * * *
123000 253000 * * * * * *
...
> morph <- read.table("morph.dat", na.strings="*")
> names(morph)
[1] "V1" "V2" "V3" "V4" "V5" "V6" "V7" "V8"
> names(morph) <-c("x","y","ER","LR","plan","prof","slope","elev")
> dim(morph)
[1] 26732     8
```

At this point, graphical and statistical analysis can begin, once a new data frame has been created excluding the NA cells beyond Kosovo's borders. In order to keep a record of the original ordering of the cells, their numbering is first prepended to the data frame, and next rows for which elevation is NA are dropped are copied to data frame `morph1`. Finally, the cell number and grid coordinates of the NA cells are stored in a separate data frame, so that the classification result can be merged back into the original grid.

```
> obsno <- 1:dim(morph)[1]
> morphno <- data.frame(obsno=obsno, morph)
> morph1 <- morphno[!is.na(morphno$elev),]
> NAs <- morphno[is.na(morphno$ER),1:3]
```

Figure 1 shows the results of four graphical analyses of the elevation variable for the remaining cells. A number of auxiliary values were also used in plotting lines on the figures; these were calculated first. The computation of the mean and median elevation is obvious; less so is the finding of the proportion of Kosovo over mean elevation, first creating a new data frame with elevation values and their ranks, and next displaying those around the mean. Since R is also a calculator, the proportion could be obtained by dividing the closest rank to the mean by the total number of cells. The hypsometric integral is also computed directly. The four diagrams shown were prepared using the `truehist()` function from the `MASS` package for the histogram, the `density()` function to calculate a Gaussian kernel density estimate with default bandwidth, a standard geomorphometric hypsometric integral diagram using built-in R functions, and finally an empirical cumulative distribution function using the `ecdf()` function from the `stepfun` package to show how modern applied statistics approaches the same task. Apart from the hypsometric integral diagram, all these methods are described in detail by Jacoby (1997). In addition to these functions, graphical "icing" was added using built-in functions, which are not reproduced here. All of the figures in this paper have been prepared in R without subsequent editing.

```
> mean(morph1$elev)
[1] 816.4051
> median(morph1$elev)
[1] 689.452
> rmorph1 <- cbind(morph1$elev, rank(morph1$elev))
> rmorph1[rmorph1[,1] > 816 & rmorph1[,1] < 816.5,]
        [,1] [,2]
[1,] 816.215 7095
[2,] 816.474 7096
[3,] 816.036 7094
> 1-7096/10948
[1] 0.3518451
> (mean(morph1$elev) - min(morph1$elev))/(max(morph1$elev) - min(morph1$elev))
[1] 0.2409858
> truehist(morph1$elev, col="gray", xlab="elevation", xlim=c(0,2500))
> plot(density(morph1$elev), xlim=c(0,2500))
> rel.elev <- (morph1$elev - min(morph1$elev))/(max(morph1$elev) - min(morph1$elev))
> plot((seq(1:length(rel.elev)))/length(rel.elev), rev(sort(rel.elev)), type="l")
> plot(ecdf(morph1$elev), verticals=F, do.p=F)
```

Initial attempts to use the `image()` function to display the geomorphometric indices — four of which are shown in Figure 2, and that of elevation in Figure 3 — were frustrated by inversion; GRASS assumes that all grids begin from top left, while R assumes that they begin from bottom left. The `order()` function was used to generate an ordering vector, which reversed the rows of the grid matrix to be displayed. Finally, the display needed to be made square to retain dimensional symmetry by setting a graphics parameter in `par()`, and reducing the number of columns by one to 163, removing the last column on the eastern side. Once again, graphic "icing" was added, but the functions used are not recorded here.

```
> reverse <- order(morph$y, morph$x)
> ER.im <- t(matrix(morph$ER[reverse], nrow=163, ncol=164, byrow=T))
```

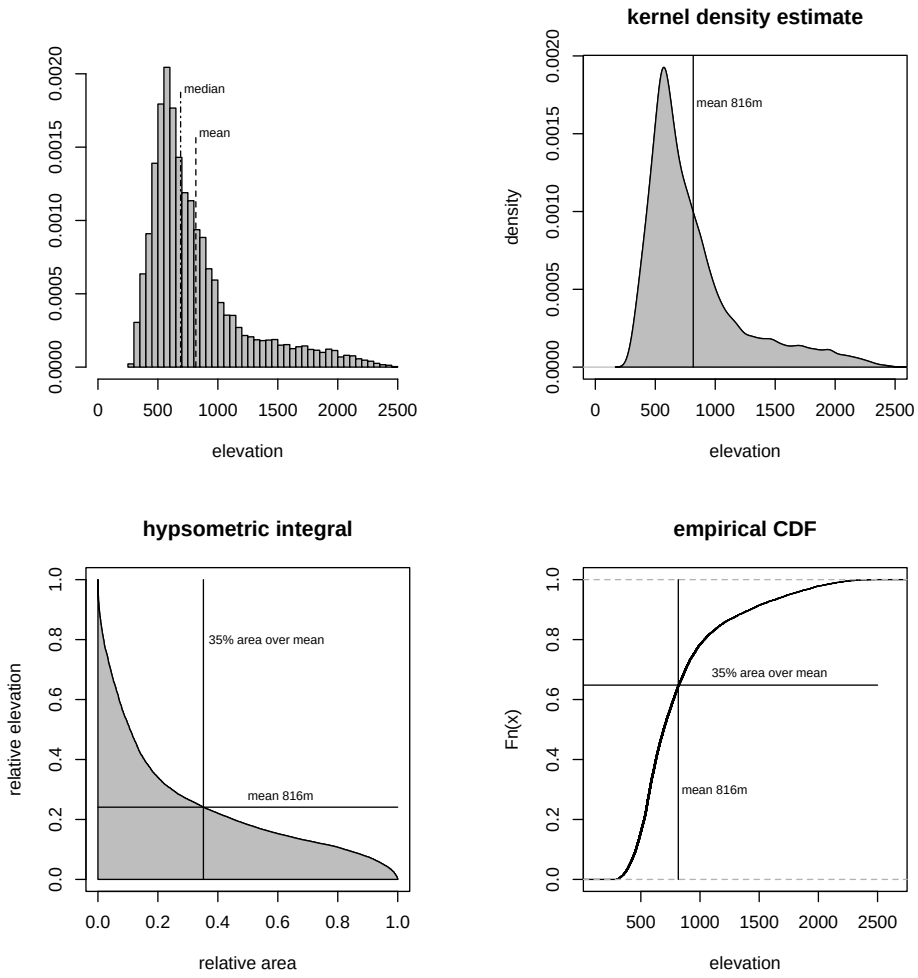

Fig. 1. Graphical data analysis of elevation in Kosovo, using a variety of `R` functions.

```
> xseq <- seq(120000, 283000, 1000)
> yseq <- seq(91000, 253000, 1000)
> image(xseq[1:163], yseq, ER.im[1:163,], col=gray(15:36/36),
+ xlab="", ylab="", main="elevation-relief ratio")
```

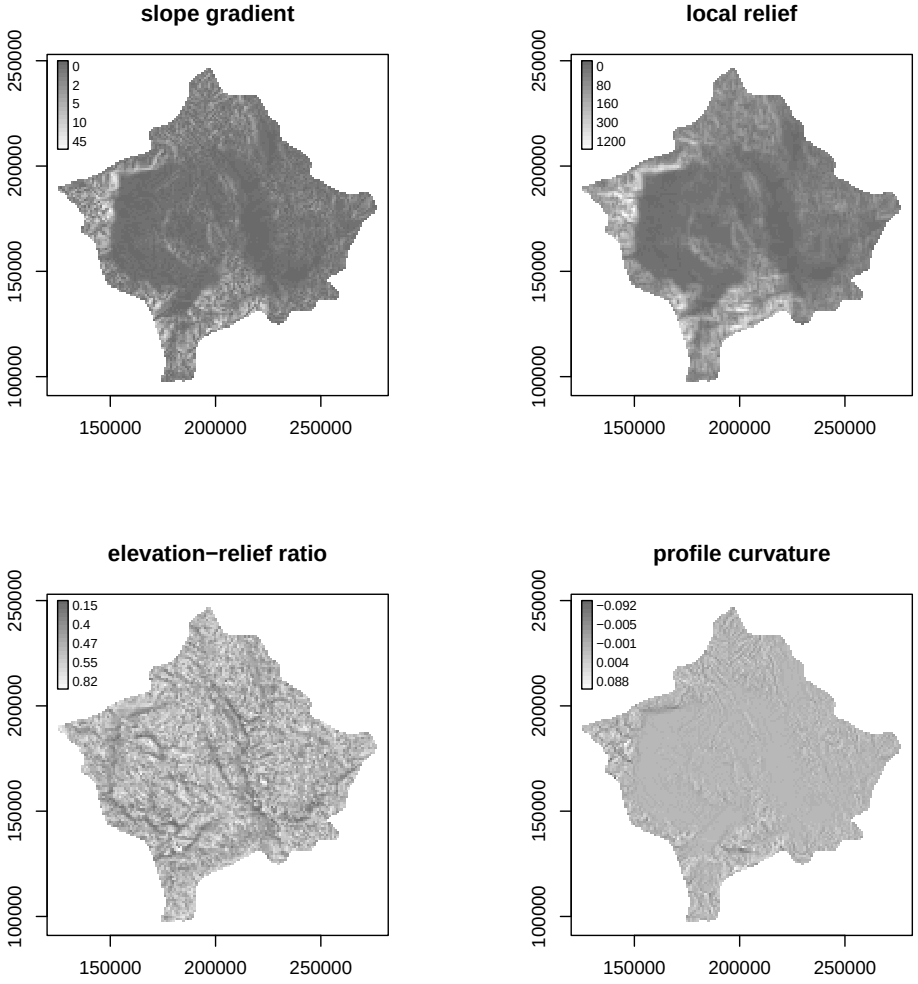

Fig. 2. Dimensionally correct maps of the values of four geomorphological indices for Kosovo; grid north is given by the vertical axes, and the axes are scaled in metres.

Inspection of the distributions of the indices suggested that logarithms should be used for classification in respect of elevation, slope gradient, and local relief. A further data frame was constructed for analysis, and its product-moment correlation matrix computed:

```
> cmorph <- data.frame(cbind(log(morph1$elev), log(morph1$slope),
+ morph1$plan, morph1$prof, log(morph1$LR), morph1$ER))
> names(cmorph) <- c("log.elev", "log.slope", "plan", "prof", "log.LR", "ER")
> cor(cmorph)
           log.elev  log.slope        plan       prof     log.LR          ER
log.elev  1.0000000 0.61874888  0.16443093 0.22153405 0.69853665  0.23094710
log.slope 0.6187489 1.00000000  0.08993179 0.01846930 0.86781107  0.06796537
plan      0.1644309 0.08993179  1.00000000 0.58341693 0.09662602 -0.16233969
prof      0.2215341 0.01846930  0.58341693 1.00000000 0.03158738  0.11551428
log.LR    0.6985366 0.86781107  0.09662602 0.03158738 1.00000000  0.03871261
ER        0.2309471 0.06796537 -0.16233969 0.11551428 0.03871261  1.00000000
```

All the indices were retained in this case, although it might have been appropriate to drop some of the highly correlated ones. The clustering procedure used was the `clara()` function from the `cluster` package; the indices were standardised before classification. Of course, classifying 10948 objects is a demanding task, but this function is designed to classify arbitrarily large data sets (Kaufman and Rousseeuw, 1990, Struyf, Hubert and Rousseeuw, 1997), using sampling to establish cluster mediods, iterating until they cease moving, for the chosen number of classes, in this case five. Accessing the outcome of the clustering procedure, we can see the number of observations, and values reporting residual dissimilarity within the clusters. In our case, however, most interest is connected to mapping where the five classes are located. To do this, use is made of the data frame recording the observation numbers and grid coordinates of the NA grid cells, which are merged back into the classification results, after the addition of NA classification values for cells outside Kosovo. The image matrix file is constructed as before, reversing the row order to accord with `R` 's assumptions about grid row ordering; the output is shown in Figure 3.

```
> cmorph.clara <- clara(cmorph, k=5, stand=T)
> print(data.frame(cmorph.clara$clusinfo))
  size  max.diss  av.diss isolation
1 2649  5.813854 1.911473  3.218603
2 1717 14.698209 2.642273  4.505136
3 1762 11.696241 2.866027  3.229143
4 2396  6.671776 1.862958  3.693557
5 2424  5.068890 1.712805  1.982502
> morph2 <- data.frame(morph1, Class=cmorph.clara$clustering)
> Class.0 <- rbind(cbind(morph2[, c(1:3, 10)]),
+ cbind(NAs, Class=matrix(nrow=dim(NAs)[1], ncol=1)))
> Class.order <- order(Class.0$obs)
> Class.1 <- Class.0[Class.order,]
> Class.image <- t(matrix(Class.1$Class[reverse], nrow=163, ncol=164, byrow=T))
```

It is not our present purpose to interpret the results in any substantive way, but without doubt perusal of such diagrams does yield quite adequate bases for such analyses. In addition, we can tabulate a series of values for the distributions of elevation values by classification classes (other indices are not shown here):

```
> res <- tapply(morph2$elev, morph2$Class, summary)
> dfres <- data.frame(matrix(unlist(res), nrow=5, ncol=6, byrow=T))
> names(dfres) <- names(res$"1")
> print(dfres)
   Min. 1st Qu. Median   Mean 3rd Qu. Max.
1 290.3   572.1  655.6  690.9   771.5 1811
2 318.1   717.0  908.7 1036.0  1298.0 2244
3 513.1   939.2 1223.0 1327.0  1674.0 2474
4 332.3   589.0  699.6  747.0   856.3 2405
5 293.8   428.9  505.5  495.8   558.1 1023
```

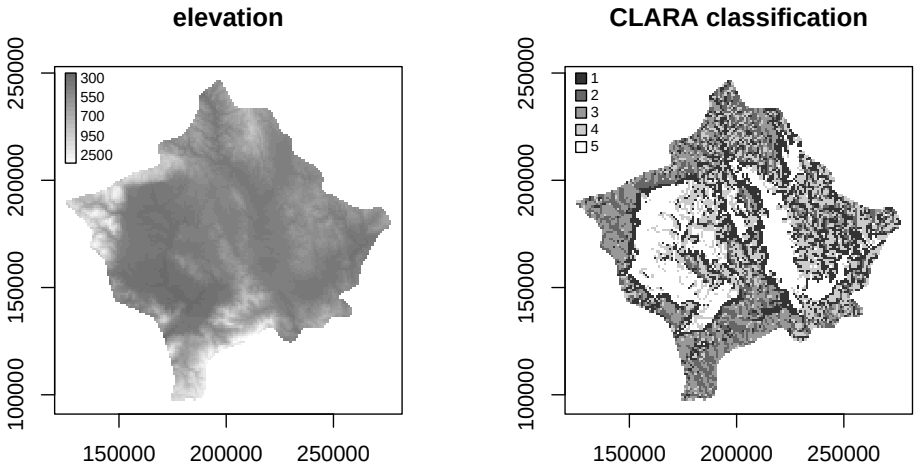

Fig. 3. Elevation in Kosovo viewed together with a geomorphometric classification into five surface topography types.

Thse analyses can be deepened using further graphical data analysis tools, such as the `boxplot` function, in which the distributions of each of the indices is conditioned by the classes resulting from this classification. As can be seen from Figure 4, qualitative interpretation of the results is aided by such graphical methods. In addition, it may be mentioned that, since `R` runs fast, it is possible not only to try out many views of the data, but also alternative classifications, perhaps without standardisation, using a different distance metric, or without logarithms, in the course of a session of interactive analysis.

In conclusion, it remains to show how the classification result may be moved back to GRASS. Since our data frame is still in the same order as was read in from GRASS, we can generate the output matrix directly. Next, we are required by GRASS to enter six header lines into the file to be used; the values given have been pasted into the concatenation function, rather than typed in, since they are all present in the data in one form or other. Since the grid coordinates are taken by GRASS as grid centres, half a spacing unit has to be accommodated on each edge. The `wc` command shows that the output file has 169 lines and 26744 separate data entities after the output matrix has been written, 6 header lines and 163 data lines, and 12 header entities and 26732 data entities. The GRASS `r.in.ascii` command is employed to move the classification back into the GRASS database; note that NULL values output by the `R write()` function are coded NA, and that `r.in.ascii` is passed this value as an argument to the (undocumented) `nv` option. The existence of the option was established by examining the source code. Following import, the new raster cell layer was displayed using `d.rast` in the normal fashion, and geographical analysis, using a location pointer to query chosen raster cell layers at a given location, could proceed without further delay.

```
> names(Class.1)
[1] "obsno" "x"     "y"     "Class"
```

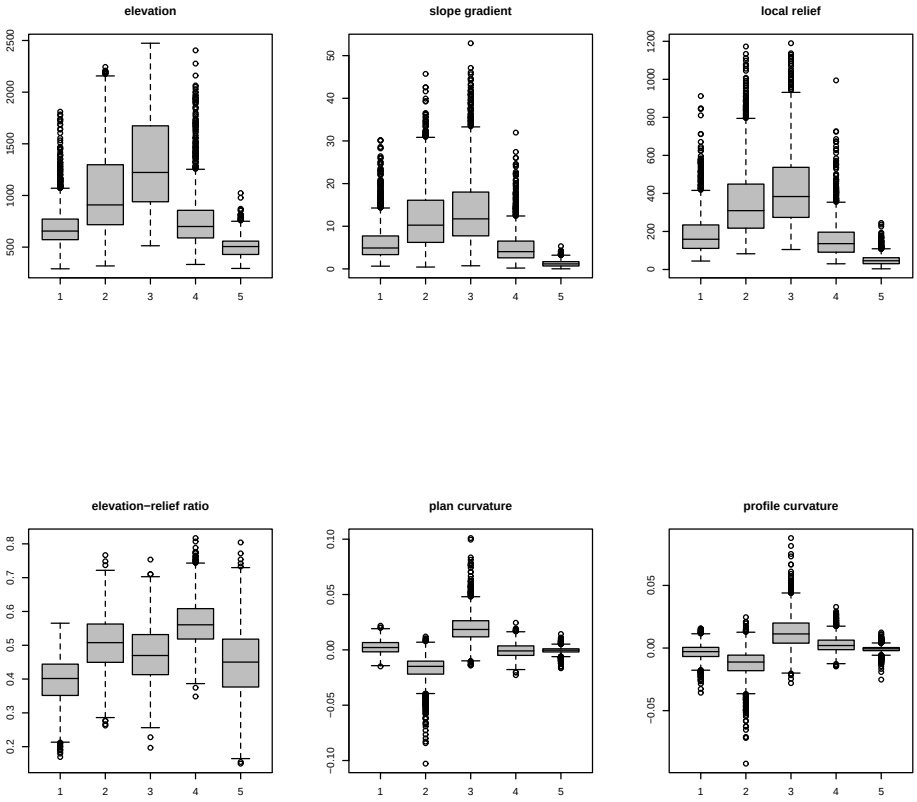

Fig. 4. Boxplots of the six geomorphometric indices by five classification classes.

```
> Ncol <- table(Class.1$y)[1]
> Nrow <- table(Class.1$x)[1]
> we.range <- range(Class.1$x)
> sn.range <- range(Class.1$y)
> xstep <- (we.range[2]-we.range[1])/(Ncol-1)
> ystep <- (sn.range[2]-sn.range[1])/(Nrow-1)
> cat("north:   ", sn.range[2]+(ystep/2), "\nsouth:   ", sn.range[1]-(ystep/2),
+ "\neast:    ", we.range[2]+(xstep/2), "\nwest:    ", we.range[1]-(xstep/2),
+ "\nrows:    ", Nrow, "\ncols:    ", Ncol, "\n",
+ file="Class-to-grass.ascii", sep="")
> Class.GRASS <- t(matrix(Class.1$Class, nrow=Nrow, ncol=Ncol, byrow=T))
> write(Class.GRASS, file="Class-to-grass.ascii", append=T, ncolumns=Ncol)
> system("wc Class-to-grass.ascii")
    169   26744   69337 Class-to-grass.ascii
> system("r.in.ascii input=Class-to-grass.ascii nv=NA output=Class
* title=classification")
CREATING SUPPORT FILES FOR Class
> system("d.rast Class")
> system("d.what.rast Class,topo,SLOPE,ER,LR,PLAN100,PROF100")

Buttons
 Left:  what's here
 Right: quit

155000(E) 191000(N)
  Class in rsb  (1)
   topo in rsb, quant   (542)
   topo in rsb, actual  (542.004000)
  SLOPE in rsb, quant   (9)
  SLOPE in rsb, actual  (9.449297)
     ER in rsb, quant   (0)
     ER in rsb, actual  (0.353796)
     LR in rsb, quant   (211)
     LR in rsb, actual  (210.835022)
...
```

With regard to the Kosovo example, it may already be seen how important the existence of major watersheds appears to be, with the western zone in class 5 showing the upland plains in the basin of the Beli Drin river, draining out through Kukës, and divided from eastern Kosovo by a north-south belt of more broken terrain. The eastern basin mostly drained northward by the river Sitnica towards the river Ibar, while south of Uroševac, it drains south-eastward via the river Lepanac, joining the Vardar just west of Skopje. These three valleys are also the main lines of communication linking Kosovo with its neighbours. This fifth class only makes up 22% of the area of the region, however, while almost 32% is in the mountainous second and third classes.

```
> system("r.report -N map=Class units=c,k,p")
r.stats:  100%
...
|-----------------------------------------------------------------------------|
|MAP: classification (Class in rsb)                                           |
|-----------------------------------------------------------------------------|
|              Category Information               | cell|  square  |   %  |
|#|description                                     |count|kilometers| cover|
|-----------------------------------------------------------------------------|
|1| . . . . . . . . . . . . . . . . . . . . . . . | 2649|  2649.000| 24.20|
|2| . . . . . . . . . . . . . . . . . . . . . . . | 1717|  1717.000| 15.68|
|3| . . . . . . . . . . . . . . . . . . . . . . . | 1762|  1762.000| 16.09|
|4| . . . . . . . . . . . . . . . . . . . . . . . | 2396|  2396.000| 21.89|
|5| . . . . . . . . . . . . . . . . . . . . . . . | 2424|  2424.000| 22.14|
|-----------------------------------------------------------------------------|
|TOTAL                                           |10948|10,948.000|100.00|
+-----------------------------------------------------------------------------+
```

Finally, GRASS is used to check the visual impression that class 5 cells tend to neighbour classes 1 and 4 rather than 2 or 3 — it turns out that less than 3% of neighbouring cells belonged to the mountainous classes:

```
> system("r.reclass input=Class output=Class5")
...
> system("r.buffer input=Class5 output=Class5b distances=1000")
> system("r.cross -z input=Class,Class5b output=C5Neigh")
> system("r.reclass input=C5Neigh output=C5Neigh1")
...
> system("r.report -N C5Neigh1 units=c,p")
r.stats:  100%
...
|-----------------------------------------------------------------------------|
|MAP: Neighbours of Class 5 (C5Neigh1 in rsb)                                 |
|-----------------------------------------------------------------------------|
|                    Category Information                      | cell|   %   |
|#|description                                                  |count| cover|
|-----------------------------------------------------------------------------|
|1|Class 1. . . . . . . . . . . . . . . . . . . . . . . . . . .|  743| 51.92|
|2|Class 2. . . . . . . . . . . . . . . . . . . . . . . . . . .|   32|  2.24|
|3|Class 3. . . . . . . . . . . . . . . . . . . . . . . . . . .|    4|  0.28|
|4|Class 4. . . . . . . . . . . . . . . . . . . . . . . . . . .|  652| 45.56|
|-----------------------------------------------------------------------------|
|TOTAL                                                          | 1431|100.00|
+-----------------------------------------------------------------------------+
```

Further analysis could be conducted with this data set, but it would be of advantage to be able to add land cover, drainage, and other data layers. In addition, use of higher resolution data would permit more of the topographic detail to be picked out.

4 Concluding remarks

The example presented above has exercised the new features of GRASS 5.0 needed for statistical analysis: NULL values for data not available, and the floating point representation of cell values. It has been shown how raster cell data in these formats may be moved from GRASS to R and back again, permitting both graphical and statistical data analysis to be conducted beyond the standard functions available in geographical information systems. It has further been demonstrated that GRASS commands can be run from within `R` using the `system()` function, and that because `R` is a programming language of substantially greater power than the shell usually employed in GRASS, the commands to be executed can be programmed where this is convenient or appropriate, and saved as user-defined functions for future use. It should also be noted that `R` has a comprehensive journalling facility, both logging all commands executed, and permitting their interactive recall and editing.

Since the `read.table()`, `write()`, and `system()` functions are common to `R` and `S` (though `system()` is known as `unix()` in `S`), the latter software system could also have been used for the same analysis under Unix-like operating systems. Major GIS systems run from a Unix shell, and supporting basic raster functionality, could also have been used in place of GRASS. In the case of GIS using graphical user interfaces, it would have been necessary to run the GIS and statistics software in parallel in the same data directory, rather than nesting the statistics program within the GIS shell, as was done here.

The arguments for using open-source software as described here are threefold: firstly cost, ease of acquisition, and the broad community support available through discussion lists; secondly, in the absence of fuller documentation — something which can affect commercial software too, source code can be read to determine how exactly the software is intended to function; and thirdly that the user can both modify the distributed source code, adding extra functionality. This has been done in GRASS in the past by creating new compiled commands, but in `R` is facilitated by the fact that it is in itself an interpreted programming language of substantial power. In the case of this example, the first two arguments apply, while the third remains for development. Such plans would include an interface such as that described here, but where data transfer would be provided in the form of `R` functions, and would use class mechanisms to provide for the reliable treatment of metadata and other issues. It also remains to provide adequate mechanisms for site and vector data layers, and for integrating GIS spatial data query mechanisms with `R` table lookup functionality. A first step will be to secure the same level of data exchangeability between GRASS and `R` as already exists between `R` and `xgobi`, a program used for dynamic data visualization.

References

1. Anselin, L. and Bao S. 1996 Exploratory spatial data analysis linking SpaceStat and ArcView; `http://www.rri.wvu.edu/papers/arcspace.pdf`.
2. Anselin, L., Dodson, R. and Hudak, S. 1993 Linking GIS and spatial data analysis in practice. *Geographical Systems*, 1, 3-23.
3. Bailey, T. 1994 A review of spatial statistical analysis for geographical information systems. In *Spatial analysis and GIS* (ed. A. S. Fotheringham A and P. Rogerson), pp. 13–44. London: Taylor & Francis.
4. Bao, S. and Anselin, L. 1997 Linking spatial statistics and GIS: operational issues in the SpaceStat–ArcView link and the S+Grassland link; `http://www.wvu.edu/~regional/wpapers/pdffiles/asawp.pdf`.
5. Becker, R. 1994 A brief history of S. In P. Dirschedl and R. Osterman (eds) *Computational statistics*, (Heidelberg: Physica Verlag), 81–110.
6. Becker, R., Chambers, J. and Wilks, A. 1988 *The new S language.* (New York: Chapman and Hall).
7. Bivand, R. 1996 Scripting and toolbox approaches to spatial analysis in a GIS context. In *Spatial analytical perspectives on GIS in the environmental and socio-economic sciences.* (ed. M. Fischer, H. Scholten, and D. Unwin), pp. 39–52. London: Taylor & Francis.
8. Brown, D. G., Lusch, D. P. and Duda, K. A. 1998 Supervised classification of types of glaciated landscapes using digital elevation data. *Geomorphology*, 21, pp. 233–250.
9. Burrough, P. A. and McDonnell, R. A. 1998 *Principles of geographical information systems.* (Oxford: Oxford University Press).
10. Byars, B. and Clamons, S. 1998 GRASS is back! *GIS World*, 10(2), 60–63.
11. Can, A. 1996 Weight matrices and spatial autocorrelation using a topological vector data model. *International Journal of Geographical Information Systems*, 10, 1009–1017.
12. Chambers, J. and Hastie, T. (eds) 1992 *Statistical models in S.* (New York: Chapman and Hall).
13. Guzzetti, F. and Reichenbach, P. 1994 Toward the definition of topographic divisions for Italy. *Geomorphology*, 11, pp. 57–75.

14. Haining, R. 1994 Designing spatial data analysis modules for geographical information systems. In *Spatial analysis and GIS* (ed. A. S. Fotheringham A and P. Rogerson), pp. 45–64. London: Taylor & Francis.
15. Haining, R. 1996 Designing a health needs GIS with spatial analysis capability. In M. M. Fischer, H. J. Scholten and D. Unwin (eds) *Spatial analytical perspectives on GIS*, (London: Taylor & Francis), 53–65.
16. Ihaka R, 1998 `R` : past and future history. Unpublished paper, `http://www.ci.tuwien.ac.at/R/doc/interface98-paper/paper.html`.
17. Ihaka, R. and Gentleman, R. 1996 `R` : a language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5, 299-314.
18. Jacoby, W. G. 1997 *Statistical graphics for univariate and bivariate data*. (Thousand Oaks: Sage Publications).
19. Jones, K. H. 1998 A comparison of algorithms used to comute hill slope as a property of the DEM. *Computers and Geosciences*, 24, pp. 315–323.
20. Kaufman, L. and Rousseeuw, P.J. 1990. *Finding Groups in Data: An Introduction to Cluster Analysis.* (New York: Wiley).
21. Ousterhout, J. K. 1997 Scripting: higher level programming for the 21st century. `http://??`.
22. Pike, R. J. and Wilson, S. E. 1971 Elevation-relief ratio, hypsometric integral, and geomorphic area-altitude analysis. *Geological Society of America Bulletin*, 82, pp. 1079–1084.
23. Raymond, E. 1997 The cathedral and the bazaar. `http://www.ccil.org/~esr/writings/cathedral-paper.html`.
24. Struyf, A., Hubert, M. and Rousseeuw, P.J. 1997 Integrating robust clustering techniques in S-PLUS. *Computational Statistics and Data Analysis*, 26, pp. 17–37.
25. Sulebak, J. R. 1997 Geomorphometric studies of different topographic regions; analyses and applications from Norway and Sweden. Rapportserie i naturgeografi, rapport nr. 8, University of Oslo.
26. Venables, W. N. and Ripley, B. D. 1997 *Modern applied statistics with S-PLUS.* (New York: Springer); `http://www.stats.ox.ac.uk/pub/MASS2`.
27. Wessel, P., and Smith, W. H. F. 1998, New, Improved Version of Generic Mapping Tools Released, *EOS Trans., AGU*, 79 (47), p. 579.
28. Zevenbergen, L. W. and Thorne, C. R. 1987 Quantitative analysis of land surface topography. *Earth Surface Processes and Landforms*, 12, pp. 47–56.