GRASS GIS et R
main dans la main
dans un monde libre

Reinhard.Furrer@epfl.ch & Diego.Kuonen@epfl.ch, EPFL, Département de Mathématiques

Rappelons que **R**, un exemple parmi tant d'autres du succès incontestable des modèles libres, est un langage et un environnement pour les calculs statistiques et leurs représentations graphiques. Cet article constitue le troisième paru dans le Flash informatique au sujet de **R**, cette année. Il conclut la trilogie sur **R**. Le premier article paru dans le FI 2/01 [1] était un point de départ pour **R**, tandis que le but du deuxième paru dans ce Flash informatique spécial [2] est d'illustrer les interfaces entre **R** et les bases de données relationnelles. Le but de ce troisième document est de montrer l'utilisation de la géostatistique dans un monde libre en combinant **R** avec GRASS GIS.

Ce texte n'est pas un manuel pour les systèmes d'information géographique, et nous ne prétendons pas être des spécialistes de ce domaine. C'est plutôt un regard de deux statisticiens pour souligner l'importance des logiciels libres dans notre domaine de recherche et de travail.

Les systèmes d'information géographique

Les systèmes d'information géographique font partie de ces outils qui ont bénéficié des progrès réalisés dans le secteur de l'informatique. Le matériel et les logiciels sont de plus en plus performants et de plus en plus nombreux. Il est, aujourd'hui, difficile de concevoir que la réalisation d'un SIG (*Système d'Information Géographique*) se fasse sans les apports de l'informatique notamment pour la gestion, le traitement et l'analyse des données. Si les outils informatiques ont un rôle fondamental dans la conception des SIG, il faut toujours garder en mémoire qu'il ne s'agit que d'outils facilitant le travail du géographe. Bien connaître les logiciels de base de données géographiques ne suffit pas, il faut aussi et surtout connaître la démarche géographique et sa méthode.

Selon le *International GIS Dictionary* [3], un SIG ou un *GIS* (*Geographical Information System*) est un *computer system for capturing, managing, integrating, manipulating, analysing and displaying data which is spatially referenced to the Earth*.

C'est donc un ensemble organisé de matériel informatique, de logiciels, de données géographiques et de personnes capables de saisir, stocker, mettre à jour, manipuler, analyser et présenter toute forme d'information référencée géographiquement.

Ce qui distingue les SIG des autres formes de systèmes d'information, telles que les bases de données ou les tableurs, est qu'un SIG traite des informations spatiales. Un SIG possède la capacité d'associer plusieurs couches de données pour des points situés dans l'espace. L'information spatiale utilise une localisation avec un système de coordonnées ou système de référence (par exemple, un point est spécifié par la latitude et la longitude).

À l'origine, tout SIG repose sur une base de données géographiques. Un SIG n'est donc pas un simple système de cartographie assisté par ordinateur. S'il doit permettre la représentation des données qu'il gère et qu'il crée, celle-ci n'est pas impérativement cartographique. Les sorties peuvent être aussi alphanumériques et graphiques. Sans *Système de Gestion de Base de Données Localisées* (SBGDL), pas de base de données géographiques. C'est le SGBD qui permet à l'utilisateur d'agir en entrée du système et en sortie.

Quelques applications des SIG sont évidentes, par exemple les administrations peuvent utiliser des SIG pour contrôler les bornes de propriétés ou estimer les ressources environnementales. Des SIG peuvent également être employés pour planifier des services tels que la santé et l'éducation primaire, en tenant compte de la distribution de la population et de l'accès aux équipements. Des SIG sont de plus en plus employés pour aider des entreprises à identifier leurs marchés potentiels et en mettant à jour une base de données spatiales de leurs clients.

De manière globale, un SIG peut être défini comme un ensemble de principes et techniques utilisés pour réaliser des objectifs comme:

- trouver des localisations ayant des attributs spécifiques. Par exemple trouver l'endroit pour un nouveau aérodrome.
- examiner des attributs géographiques d'une localisation spécifique. Par exemple examiner la densité des routes d'une région particulière.

Les données des SIG sont souvent sauvées dans une ou plusieurs couches pour éviter des problèmes techniques causés par le traitement de grosses quantités de données. Il est plus facile de travailler avec des problèmes spatiaux complexes une couche à la fois pour permettre la révision des données sans devoir réviser le système d'information en entier.

Des données spatiales consistant en localisations et attributs sont représentées dans un SIG par les formats suivants:

- modèle vectoriel, sous forme d'objets géométriques: points, lignes, polygones qui représentent l'endroit et les bornes des entités géographiques;
- modèle *raster*, sous formes d'images composées de pixels.

Ces deux modèles ont des avantages et des désavantages. Le choix du modèle utilisé est souvent imposé par la nature du travail et les données disponibles.

Pour conclure cette section, voici quelques spécificités des données géographiques et spatiales:

- les observations ne sont pas indépendantes;
- les erreurs ont souvent une structure spatiale;
- les distributions sont non normales;
- la plupart des couches sont des données catégorielles ;
- les données sont rarement stationnaires;
- il y a souvent une interaction du temps avec l'espace, et
- il y a des données redondantes.

GRASS GIS

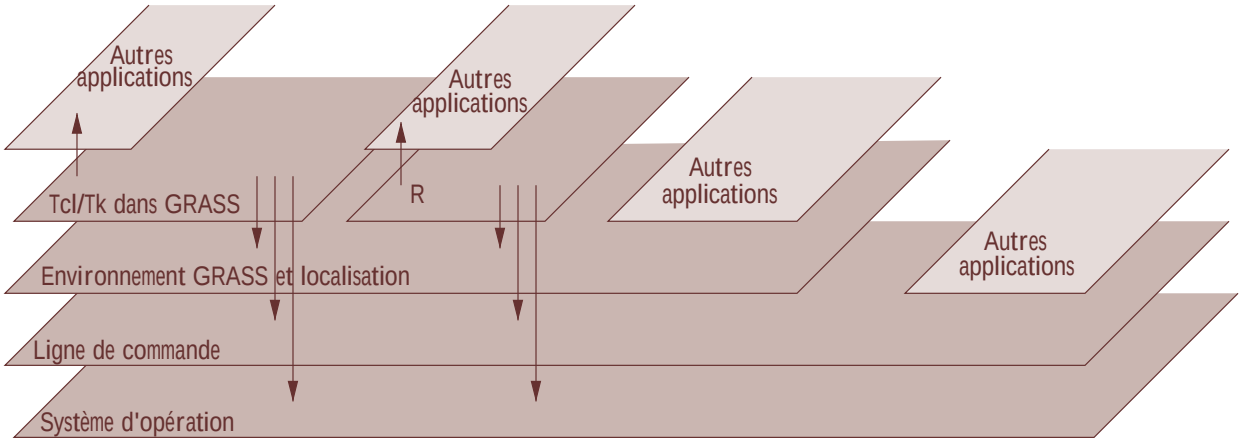

Figure 1 – Architecture de GRASS

GRASS *(Geographic Resources Analysis Support System)* est un système d'information géographique qui a été originellement programmé par des ingénieurs de l'armée américaine, plus précisément par l'*U.S. Army Construction Engineering Research Laboratories* (USA-CERL). Depuis, ce logiciel est maintenu et amélioré par une nouvelle armée de programmeurs, et maintenu officiellement par l'Université de Baylor aux États-Unis et l'Université d'Hanovre en Allemagne. Les utilisateurs de GRASS incluent la NASA, la NOAA, le USDA, le U.S. National Park Service, le U.S. Census Bureau, le USGS, et d'autres bureaux de *consulting environmental*; voir [4] pour plus de détails.

Les caractéristiques principales de GRASS sont:

- distribué librement sous licence GPL;
- disponible pour les principales architectures et systèmes d'exploitation (Unix/Linux, MacOS/X), avec une version préliminaire pour Windows NT/2000;
- lancé par une commande en ligne standardisée ou par une interface-utilisateurs graphique intuitive basée sur Tcl/Tk;
- accès à une DBMS (PostgreSQL, ou par ODBC, ...);
- 121 projections géographiques implantées; et support d'un vaste nombre de différents formats des données (*raster* et vecteur).

Comme GRASS est utilisé dans plusieurs universités, on a accès à plusieurs tutoriels et manuels [5, 6]. Comme mentionné, GRASS est lancé depuis une commande en ligne. Chaque commande GRASS n'est rien d'autre qu'un script *shell*. On peut donc lancer d'autres applications dans l'environnement GRASS, comme par exemple **R**. La figure 1 montre l'architecture de GRASS avec les différentes niveaux des applications.

A titre d'exemple, lançons GRASS avec les données Spearfish qui est un jeu de données libre sur lequel la plupart des manuels se basent [7]:

```
$FreeWorld$ grass5
```

Le programme demande de spécifier la *location* et le *mapset*. Une *location* définit les bornes et les projections d'un projet. A chaque projet (i.e. location) on peut associer des *mapsets* définissant des sous-régions.

Ouvrons une fenêtre graphique pour afficher l'image topologique de la région définie par Spearfish et superposons la région du Mount Rushmore (ci-après Mount Redmont):

```
> d.mon start=x0
> d.rast map=elevation.dem
> d.rast -o map=redmont
```

À noter que GRASS ne met pas les fenêtres automatiquement à jour. Les fenêtres graphiques doivent être fermées avec `d.mon stop`. De plus, il n'est pas conseillé de changer leur taille interactivement avec la souris.

L'aide pour la commande `ps.map` (génération des fichiers postscript) est obtenue par:

```
> g.manual entries=ps.map
```

Les modules et commandes de GRASS sont structurés grâce à un préfixe; voir le tableau ci-après.

Préfixe	Classe	Signification des commandes
`d.`	`display`	présentation graphique et requêtes visuelles
`s.`	`sites`	traitement des données semis de points
`r.`	`raster`	traitement des données en mode raster
`i.`	`imagery`	traitement des données en mode image
`v.`	`vector`	traitement des données vectorielles
`g.`	`general`	commandes générales des opérations des fichiers
`m.`	`misc`	commandes diverses
`p.`	`paint`	commandes d'établissement de cartes
`ps.`	`postscript`	commandes d'établissement de cartes postscript

Presque toutes les commandes de GRASS peuvent aussi être effectuées par l'interface Tcl/Tk (lancée depuis GRASS avec la commande `tcltkgrass`). Par exemple pour rajouter les sites archéologiques à l'image topologique on peut, au lieu de taper la commande:

```
>d.sites type=box sitefile=archsites color=red size=6
```

utiliser le menu `Site -> Display -> Display Site Markers` qui ouvre la fenêtre montrée dans la Figure 2.

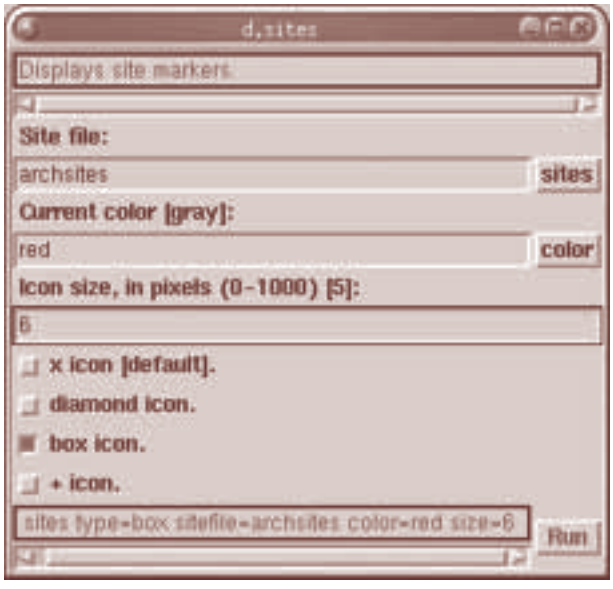

Figure 2 – Saisie de l'interface Tcl/Tk

Un petit exemple d'analyse géostatistique

Une grande partie des travaux faits avec GRASS consistent en l'analyse et la manipulation des données, qui peuvent être faites avec un logiciel statistique comme **R**, conçu pour ce type de travail. La Figure 3 montre le domaine d'application de **R** dans GRASS.

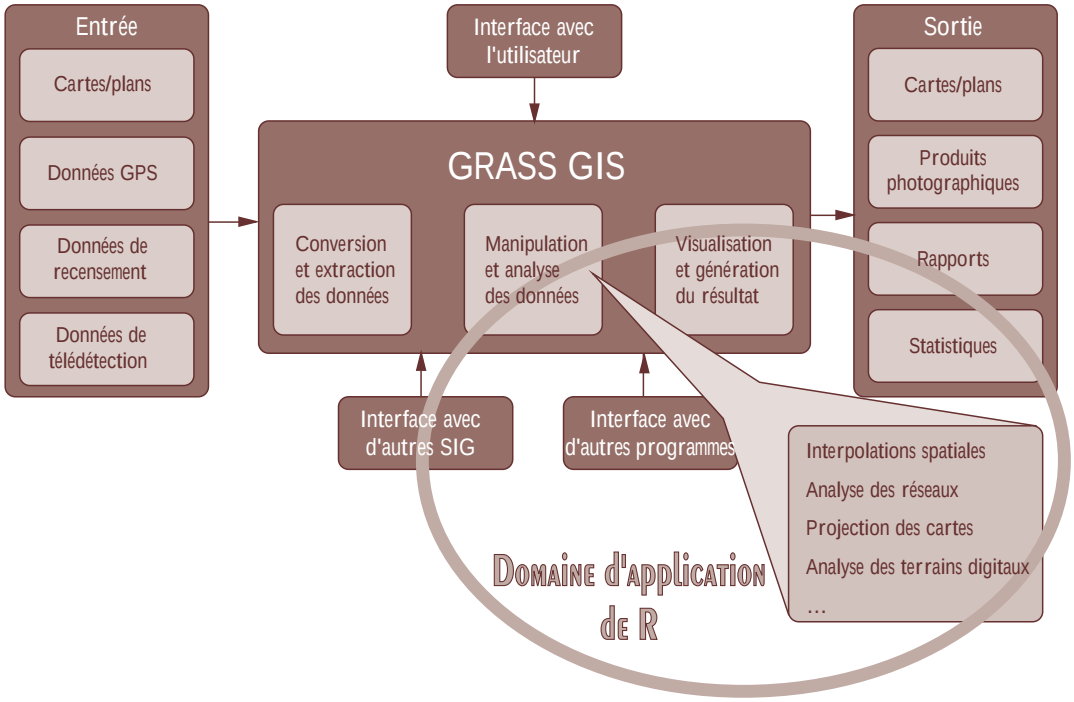

Figure 3 – Domaine d'application de **R** dans GRASS

Supposons qu'un travail consiste à comparer la densité des *bugs* (ci-après pingouins) avec une région comparable. La Figure 4 montre les pingouins dans une représentation tridimensionnelle (fait par la commande `nviz`). On peut voir les endroits comme des réalisations d'un processus ponctuel (inhomogène) - nous forçant de nous rappeler les statistiques spatiales. Pour des analyses géostatistiques comme l'interpolation spatiale, les corrélations spatiales, etc., il existe pour **R** plusieurs modules libres contenant ces fonctions [8].

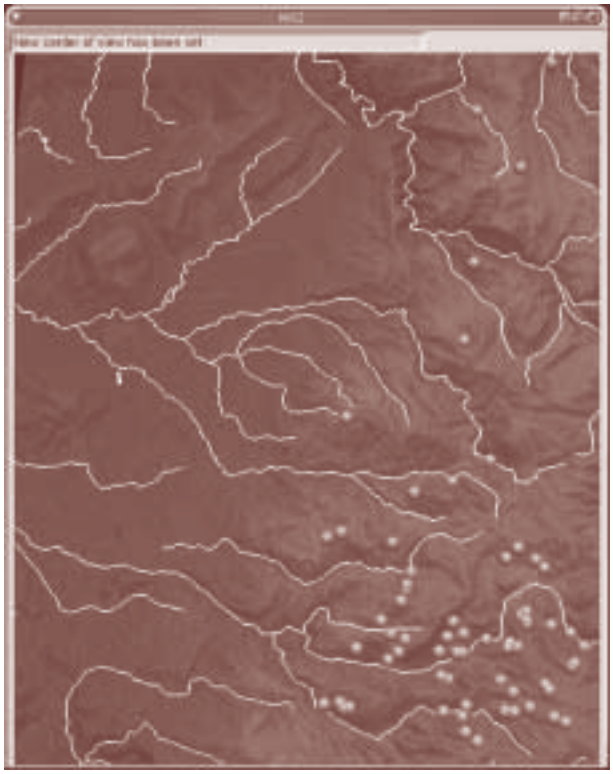

Figure 4 – Les endroits des pingouins à la conquête de Mount Redmont

Pour l'utilisation de l'interface **R**/GRASS il faut installer le module GRASS, disponible à [9]. Il est recommandé d'installer aussi les modules *`akima, VR, nlme, sgeostat, tripack`* de **R** [10].

Reprenons GRASS avec les données Spearfish, lançons **R** et chargeons le module **R**-GRASS dans **R** (ayant lancé `tcltkgrass` permet de travailler avec **R** et GRASS simultanément):

```
> R
> library(GRASS)
```

Dans la variable `G`, on affecte l'information géographique de notre jeu de données (pour les cracks, `G` est un objet de la classe `grassmeta`, ayant des méthodes `plot`, `summary`, etc.):

```
> G <- gmeta()
```

Nos données consistent en plusieurs différentes cartes *raster*, la commande suivante les affiches (pour ne pas alourdir l'article, on supprime la sortie de **R**):

```
> print(list.GRASS(type=»rast»))
> print(list.GRASS(type=»sites»))
```

Nous nous sommes intéressés aux pingouins et avons copié leurs positions dans la variable `world.domination` de **R**.

```
> world.domination <-sites.get(G,slist=c («bugsites»))[,1:2]
> names(world.domination) <- c('x','y')
```

Ripley [11] était le premier à réaliser l'importance de la fonction K(r) (i.e. espérance du nombre évènements dans une disque de rayon r) pour décrire la dépendance spatiale. La fonction `khat` du module `splancs` [8] estime K(r) dans un certain polygone.

```
> library(splancs)
> boundary <- cbind(x=c(590000, 609000, 609000, 590000),y=c(4914000,4914000,4928000,4928000))
> distance <- seq(0,to=10000,by=200)
> k.hat <- khat(world.domination, boundary, distance)
> plot(distance,k.hat,main='La fonction K')
> lines(distance, pi*distance^2)
```

La Figure 5 montre le résultat. Malheureusement il n'est pas toujours facile d'interpréter cette fonction. Quelquefois, il suffit de construire une simple interpolation de la densité (Figure 6).

```
> plot(G,kde2d.G(world.domination$x, world.domination$y, h=c(2500,2500), G=G), xlab='est',
                                                                                ylab='nord')
> title(«Densité des pingouins ('world.domination')»)
> points(world.domination)
> polygon(boundary)
```

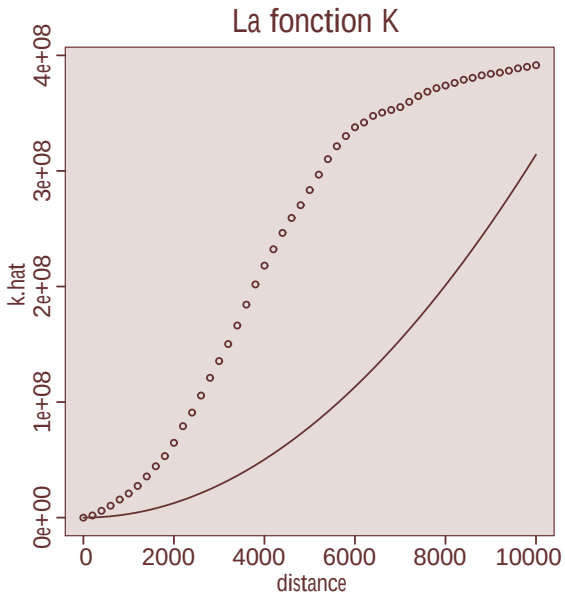

Figure 5 – La fonction K(r) estimée

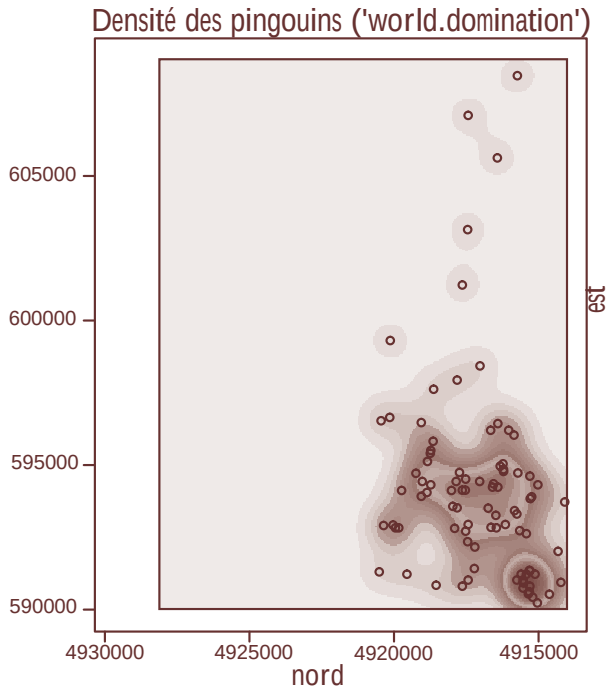

Figure 6 – La densité estimée des pingouins à la conquête du Mount Redmont

Pour aller plus loin

L'architecture de GRASS (et de **R**) quipermet d'utiliser plusieurs programmes en parallèle (voir Figure 1), est illustré ici par quelques exemples particuliers.

Le modèle de mémoire statique de **R** peut causer des problèmes dans le cas de bases de données immenses. Pour y remédier, on peut utiliser le module `RPgSQL` de **R** [2], qui permet de lire et écrire des tables dans et depuis une base de données. GRASS

lui-même met aussi à disposition des interfaces à des DBMS ou des *SGBD (Data Base Management Systems* ou *Système de Gestion de Base de Données*) et à PostgreSQL; voir [12] pour plus de détails.

Un deuxième exemple est le programme `xfig` qui permet de créer des figures postscript. Pour rajouter à une carte de GRASS des flèches, du texte ou d'autres éléments graphiques, on ouvre `xfig` avec l'image transformée en format tiff ou exportée par la commande `v.out.xfig`.

A première vue, on pourrait argumenter sur le fait que GRASS n'est pas un programme ayant tous les éléments nécessaires pour un bon SIG: du point de vue des capacités statistiques et graphiques, de l'accès intégré aux bases de données, etc. Est-ce que GRASS est *sous-développé* ou incomplet? Non, au contraire, dans le concept et l'idéologie d'un monde libre chaque programme s'occupe de son point fort et au lieu d'inventer la roue deux fois, on laisse - grâce à une architecture modulaire – les tâches particulières aux programmes conçus pour ces problèmes. Ceci selon le proverbe: *Schuster bleib bei deinen Leisten* que l'on pourrait traduire par *À chacun son métier*.

Conclusion

Dans cet article nous avons brièvement illustré l'utilisation de la géostatistique en combinant **R** avec GRASS. Résumons quelques points faibles et forts de notre liste subjective.

Points faibles de l'utilisation de **R** et de GRASS:

- L'utilisation de GRASS n'est pas évidente même avec Tcl/Tk. Il faut avoir une certaine expérience avec des SIG.
- Il est difficile de créer des images postscript de manière efficace. Il faut souvent un post processing (par exemple avec xfig).
- La composante temps n'est pas encore prise en compte.

Par contre, on peut profiter des points forts suivants (on remarque que la plupart de ces points restent valables pour tous les logiciels libres:

- La conception du programme permet d'utiliser plusieurs programmes en parallèle.
- Il y a une abondante documentation et de nombreux exemples traités online (des cours universitaires y compris).
- Pour un statisticien les environ 250 commandes et les modules supplémentaires suffisent largement pour le travail. Sinon la conception de GRASS permet facilement d'élargir ou de rajouter des fonctions personnalisées.
- And last but not least: it's free ☺.

Bibliographie

[1] **Diego Kuonen** & **Valerie Chavez-Demoulin** (2001), **R** - *un exemple du succès des modèles libres*, FI/02/01, *http://sic.epfl.ch/publications/FI01/fi-2-1/2-1-page3.html*

[2] **Diego Kuonen** & **Reinhard Furrer** (2001), *Data mining avec R dans un monde libre*, FI-sp-01, *http://sic.epfl.ch/publications/FI01/fi-sp-01/*

[3] **McDonnell Rachael** & **Kemp A. Karen** (1995). International GIS, Dictionary, Cambridge: GeoInformation International.

[4] GRASS Homepage: *http://www.geog.uni-hannover.de/grass/* & Information générale de GRASS: *http://www.geog.uni-hannover.de/grass/general.html*

[5] **Markus Neteler** (2001), *Short Introduction to Geostatistical and Spatial Data Analysis with GRASS and R statistical data language*, *http://www.geog.uni-hannover.de/grass/statsgrass/grass_geostats.html*

[6] **Markus Neteler** (2000), GRASS-Handbuch, Der praktische Leitfaden zum Geographischen Informationssystem GRASS. Geosynthesis 11,Geographisches Institut der Universität Hannover (ISBN 3-927053-30-9).

[7] Spearfish database description (postscript, 8 pages): *http://www.geog.uni-hannover.de/grass/gdp/tutorial/spearDB.ps.gz*

[8] Module geoS: *http://www.maths.lancs.ac.uk/~ribeiro/geoS.html*, module spatstat: *http://www.maths.uwa.edu.au/~adrian/spatstat.html*, module splancs: *http://www.maths.lancs.ac.uk/~diggle/*, module spatial: *http://www.stats.ox.ac.uk/pub/MASS3/R.shtml*, module spat01: *http://www.cnr.colostate.edu/~robin/*

[9] Module GRASS pour **R**: *ftp://reclus.nhh.no/pub/R/*

[10] CRAN Package Sources: *http://cran.r-project.org/src/contrib/PACKAGES.html*

[11] **Brian D. Ripley** (1976), *The second-order analysis of stationary point processes*, Journal of Applied Probability, 13, 255-266.

[12] **Roger Bivand** & **Markus Neteler** (2000), *Open Source geocomputation: using the R data analysis language integrated with GRASS GIS and PostgreSQL data base systems*, *http://reclus.nhh.no/gc00/gc009.htm*

[13] Free GIS Software and Free Geo-Data: *http://www.freegis.org*

[14] SAL (Scientific Applications on Linux) - Geographic Information Systems: *http://sal.kachinatech.com/E/6/index.shtml*

[15] Et si vous rêvez de télédétection plutôt que des SIG, alors voilà des logiciels libres pour vous: *http://www.remotesensing.org, http://www.ossim.org* ■