wxGRASS: Una interfaz Gráfica de Usuario para la integración de diversos programas libres para SIG

F. Alonso Sarría[(1)] *y J.A. Palazón Ferrando* [(2)]

[(1)] Departamento de Geografía, Facultad de Letras, Universidad de Murcia. Santo Cristo 1 30001 Murcia, alonsarp@um.es.
[(2)] Departamento de Ecología e Hidrología, Facultad de Biología, Universidad de Murcia. Santo Cristo 1 30001 Murcia, palazon@um.es.

RESUMEN

Se presenta una Interfaz Gráfica de Usuario, desarrollada con python y las librería gráfica wxPython, para el programa GRASS. Esta IGU, permite acceder a algunos de los módulos del programa de manera gráfica y en castellano. Su propósito es, en un principio, básicamente docente, por lo que sólo permite acceder a los módulos básicos para el trabajo con SIG. Permite además organizar los diversos elementos de la Interfaz en función de los aspectos que se quieran resaltar en una determinada sesión de trabajo con el programa.

El primer objetivo fue suavizar la curva de aprendizaje de GRASS, y de los SIG en general, a alumnos de licenciatura. Sin embargo presenta otras posibilidades como la integración, bajo una misma IGU de diversos programas (R, postgresql, GMT) que complementan las funcionalidades de GRASS haciéndolo un SIG aún más potente.

La principal diferencia entre wxGRASS y otros desarrollos de IGU para GRASS es, además del idioma castellano y de estar escrito en python, la posibilidad de definir distintos niveles de usuario que tengan acceso a diferentes aspectos del programa. Como ya se ha dicho el objetivo es docente, pero puede permitir a diversos miembros de una misma organización acceder a la información espacial con interfaces de complejidad variada.

Palabras clave: *IGU, GRASS, python, wxpython.*

ABSTRACT

This paper presents a Graphical User Interfaca, developed with python and the graphic library wxpython, to GRASS GIS. This GUI allows to access several modules with a graphic interface written in Spanish. Its main purpouse is to be a teaching tool, that is the reason way it only allows to access several basic but crucial modules. It also allows the user to organize the elements presented to stress the aspects to be resalted in a particular working sesion with the program.

The first goal was to smooth the GIS, an particulary GRASS, learning curve to graduate students. However it presents other possibilities as the integration within the same GUI of several programs (R, postgresql, GMT) that complements GRASS creating with it an even more powerful GIS.

The main difference between wxGRASS and other GRASS GUI projects is, not only the languages (Spanish and python), but also that allows the user to define several use levels with access to different components of the program. It has already been said that the main objective is to be used for teaching, but it allows several members from the same organization to access spatial information with several interfaces of different complexity

Key words: *GUI, GRASS, python, wxpython.*

INTRODUCCIÓN

wxGRASS es una Interfaz Gráfica de Usuario (IGU) concebida tanto para acceder a algunos de los módulos de GRASS como para facilitar el uso conjunto de GRASS con otros programas, creando una interfaz consistente para el conjunto de herramientas que acaba siendo utilizado cuando se trabaja en un entorno SIG. Está programada con python, un lenguaje para scripts de gran sencillez, utilizando wxPython para desarrollar la parte gráfica.

La motivación inicial fue fundamentalmente didáctica, siendo el primer objetivo del programa suavizar la curva de aprendizaje, no siempre sencilla, que deben atravesar los alumnos de asignaturas de SIG, especialmente cuando se trabaja con un programa potente, y por tanto complejo, como GRASS. Un desarrollo más profundo acerca de los problemas de la docencia de SIG que llevaron a iniciar este proyecto puede encontrarse en Alonso Sarría (2006).

El objetivo docente se abordó teniendo en cuenta el problema, más general, de las IGU en los SIG. En este sentido resultó particularmente interesante el informe final del proyecto BESTSIG (GISIG).

OBJETIVOS DEL PROYECTO

El propósito de wxGRASS no es convertirse en una IGU exhaustiva que permita el acceso a todos los módulos y opciones del programa, sino más bien ser una interfaz configurable para que aquellas personas con pocos conocimientos acerca de los SIG o sobre GRASS puedan iniciarse en el mismo con mayor facilidad. Como es lógico, se han incorporado preferentemente aquellas herramientas más básicas que suelen abordarse en cursos de SIG de tipo introductorio o medio.

Más adelante, el desarrollo del proyecto podría ir hacia la creación de IGU para un SIG corporativo en un entorno cliente-servidor. Esta IGU tendría un carácter modular y configurable de manera que existan una serie de perfiles de usuario a los que

corresponderán diferentes apariencias y potencialidades. De este modo, la apariencia del sistema sería diferente en función del perfil y necesidades del usuario.

La interfaz no pretende ocultar la complejidad de GRASS al usuario sino mostrársela de manera sencilla. Así cada vez que este pulsa el botón “Ejecución” de un módulo, wxGRASS genera una orden para GRASS a partir de la información suministrada por el usuario y, además de ejecutarla, la muestra al usuario, resaltada en rojo, en un control de texto (figura 1) en el que van quedando registradas todas las órdenes junto con las salidas de los módulos de GRASS utilizados. El contenido de este control puede copiarse y pegarse en un editor de texto para genera un script.

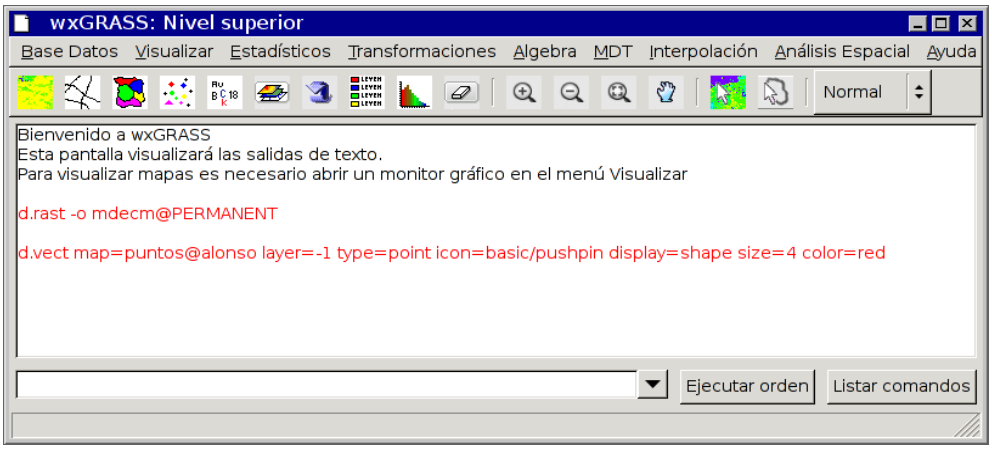

Figura 1. Pantalla principal de wxGRASS. El control de texto muestra las órdenes ejecutadas

De este modo se permite a los usuarios ejecutar módulos de GRASS de un modo interactivo pero viendo al mismo tiempo cuales son las lineas de comando. Con ello se consigue además que el usuario no olvide el carácter computacional del trabajo con los SIG en general y con GRASS en particular. No hay que olvidar que para los objetivos iniciales de este proyecto *usuario* equivale a *alumno*.

Al contrario de lo que ocurre con las IGUs que vienen con GRASS, se pretende que esta sea todo lo sencilla y clara que el usuario requiere en cada momento. Así se han definido diferentes niveles de usuario a partir de las prácticas de alumno de la asignatura de SIG.

La primera versión de la wxGRASS se desarrolló para GRASS 5.4, la segunda se ha modificado para adaptarla a GRASS 6.

Otro de los objetivos de este proyecto es tratar de presentar una IGU unificada para diversos programas que se utilizan de manera conjunta en el trabajo con datos geoespaciales. A nadie le sorprende ya que se trabaje de forma más o menos integrada con un SIG, un gestor de bases de datos y un programa de análisis de datos. Mientras que al principio los desarrolladores trataron de integrar todo en una misma herramienta, la tendencia es cada vez más el desarrollo de interfaces y librerías que permitan el trabajo de forma coordinada con diversos programas, cada uno especializado en una acción

Desde el punto de vista del usuario novato, esto supone el problema de tener que trabajar con diversos entornos, problema que se vuelve dramático en entornos docentes.

PYTHON Y WXPYTHON

Python es un lenguaje de scripting que permite dividir el programa en módulos reutilizables desde otros programas Python (van Rossum & Drake., 2000). Viene con una gran colección de módulos estándar que se pueden utilizar como base de los programas. Estos módulos incluyen funciones para la E/S de ficheros, llamadas al sistema, *sockets,* etc. Admite diversos tipos compuestos de datos cómo listas, tuplas, diccionarios o conjuntos, además de clases, lo que facilita el proceso de información estructurada de diversos modos.

Python es un lenguaje interpretado, lo que ahorra un tiempo considerable en el desarrollo del programa. El intérprete se puede utilizar de modo interactivo, lo que facilita experimentar con características del lenguaje, escribir programas desechables o probar funciones durante el desarrollo del programa.

Entre las ventajas que se suelen ver a este tipo de lenguajes de scripts:

1. Desarrollo más rápido, no es necesario compilar y enlazar
2. Multiplataforma, funciona en cualquier arquitectura siempre y cuando disponga del intérprete del lenguaje.
3. Facilidad y rapidez de aprendizaje del lenguaje, la estructura básica de wxGRASS estaba creada tras unas pocas horas de lectura sobre el lenguaje.

Como principal inconvenientes, se suele aducir su lentitud ya que los programas interpretados se ejecutan más lentamente que los compilados.

Una buena manera de aprovechar las ventajas que ofrece python y minimizar su inconveniente es el desarrollo de interfaces para funciones o módulos desarrollados con lenguajes de programación de más bajo nivel, como C, tal es el caso del proyecto que se presenta en el que se plantea una interfaz para GRASS, R PostgreSQL y GMT.

Una de las características de python es disponer de diversas posibilidades para la creación de entornos gráficos. En concreto wxPython es un módulo de python que permite acceder desde este lenguaje a wxWidgets. Se trata de una librería multiplataforma para el desarrollo de IGUs escrita con C++.

Al igual que python, wxPython ofrece una curva de aprendizaje suave que permite crear aplicaciones funcionales con gran rapidez (Rappin & Dunn, 2006).

¿COMO FUNCIONA WXGRASS?

La filosofía básica de wxGRASS se basa en dejar hacer al máximo a GRASS. Los módulos de GRASS hacen su trabajo de manera óptima y el papel de la IGU es actuar de mero interlocutor. De este modo se minimiza una de las desventajas de los lenguajes de scripting que es la lentitud en la ejecución.

La orden para activar GRASS es **`wxGRASS.py`**, a la que se puede añadir un número que indica el nivel de usuario, por el momento los números equivalen a la práctica que debe hacer el alumno de la asignatura de SIG. Si se introduce un número mayor que 200 se introducen todas las opciones más las opciones en desarrollo (en este momento la integración de R y GMT), si no se introduce ningún número se presentan todas las opciones excepto las que están en desarrollo.

La IGU se basa en una pantalla principal (figura 1) con un un sistema de menús y una barra de herramientas. Cada una de las opciones de menú y los botones de la barra constituyen la interfaz a un módulo de GRASS o a combinaciones de módulos que llevan a cabo una determinada acción.

Los botones de la barra de herramientas controlan la visualización de capas (**`d.rast`**, **`d.vect`**), zoom y consultas interactivas (**`d.what.*`**). Las opciones de menú controlan operaciones de gestión de archivos, álgebra de mapas, interpolación, etc. Se ha evitado la habitual redundancia que presentan las IGU en las que los botones de la barra de herramientas se conciben como simples atajos a las opciones de menú.

La figura 2 muestra como ejemplo la ventana que aparece al usuario al seleccionar la opción de menú para generar isolineas. Tal cómo aparece en el título de la ventana, es una interfaz para el módulo de GRASS **`r.contour`** que se encarga de recopilar del usuario el conjunto de parámetros necesarios para formar la orden que, tras pulsar el usuario el botón “Ejecutar”, es enviada tanto al sistema para su ejecución como al control de texto principal de wxGRASS para su visualización por el usuario.

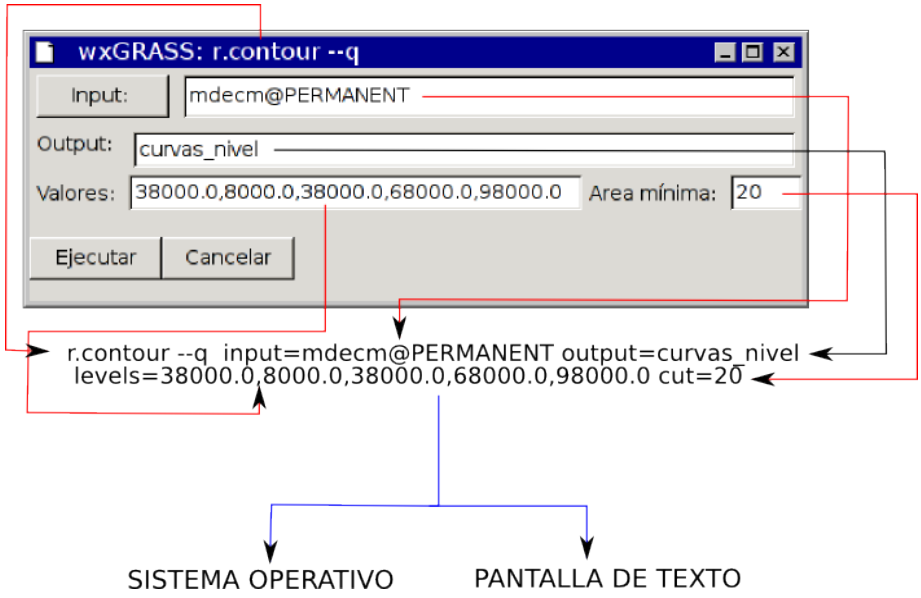

Figura 2: Ventanas de usuario y construcción de una instrucción.

Cada una de estas ventanas se definen en el programa como una clase en un módulo propio. Alguno de los módulos de wxGRASS realizan también un preprocesado de las opciones de usuario para facilitarle el trabajo. En el caso de la figura 2, cuando el usuario ha seleccionado una capa raster, el módulo calcula unos valores provisionales para las isolineas, que muestra al usuario, calculados a partir de los valores mínimo y máximo de la capa.

Las órdenes de visualización generadas por los correspondientes módulos de wxGRASS para obtener la visualización en el monitor gráfico activo, pueden visualizarse y editarse pulsando el botón “Control de la visualización” (sexto botón de

la barra de herramientas en la figura 1). Una vez editada, esta lista puede reejecutarse con el séptimo botón que refresca el contenido del monitor gráfico. De este modo se obtiene la misma funcionalidad que con las listas de capas cargadas y/o visualizadas de los entornos tipo Arc/view.

Tal como se ha dicho ya, wxGRASS no permite acceder a toda la potencialidad de GRASS; en aquellos casos en que se necesite utilizar un módulo no disponible mediante la interfaz, deberá utilizarse la linea de comandos o escribir la orden en la caja de texto situada en la parte inferior de la ventana principal (figura 1). En todo caso, es preferible que wxGRASS se ejecute siempre en segundo plano (**`wxGRASS.py &`**), dentro de una sesión de GRASS de manera que el usuario tenga en todo momento acceso a la linea de comandos de GRASS.

En la página http://www.um.es/geograf/sigmur/wxgrass/wxGRASS.html, puede obtenerse un manual más completo de wxGRASS.

INTEGRACIÓN DE OTROS PROGRAMAS

Tal cómo se ha mencionado, con wxGRASS se explora la posibilidad de trabajar de forma integrada con 5 programas de características muy diferentes:

- Un SIG, GRASS
- Un sistema de gestión de bases de datos, PostgreSQL
- Un programa de análisis y representación gráfica de datos, R
- Una librería de R para análisis geoestadístico, gstat
- Un programa para la maquetación de mapas, GMT

Estos programas, salvo quizás GMT, tienen una gran tradición en cuanto a su utilización de forma integrada con GRASS (Bivand y Neteler, 2000; Neteler y Mitasova, 2002; Alonso Sarría y Palazón Ferrando, 2004)

Integrando PostgreSQL

Una de las tendencias más notables en los últimos años respecto al trabajo con SIG es el cada vez mayor protagonismo de los Sistemas de Gestión de Bases de Datos externos al propio programa de SIG. En el caso de GRASS, es PostgreSQL el más utilizado.

Hasta su versión 5.4, GRASS disponía sólo de módulos que actuaban como programas clientes de base de datos (utilizando PostgreSQL como servidor. La versión de wxGRASS para GRASS 5.4 permitía hacer consultas a bases de datos y enlazar tablas a mapas vectoriales aprovechando los módulos de conexión a PostgreSQL de python.

En la nueva versión, GRASS se encarga de realizar estos enlaces por lo que se han sustituido las operaciones en python por interfaces simples a los módulos de GRASS correspondientes.

Sin embargo se ha optado por mantener una opción de consultas SQL a bases de datos PostgreSQL que, además, permite crear un fichero vectorial de puntos a partir de una consulta a una tabla.

En la figura 3 aparece la ventana de este módulo. El botón “Base de datos” permite seleccionar la base de datos con que queremos trabajar. Los botones “Tabla:” y “Variables” permiten determinar que variables se quieren consultar y de que tabla, de

manera que al seleccionar aquellas se esboza una consulta SQL en el control de texto. Esta se puede editar, guardar (también puede cargarse un fichero preexistente de consultas SQL) o ejecutar y los resultados, que se muestran en una tabla aparte, pueden almacenarse como un fichero de texto o como una capa vectorial de puntos.

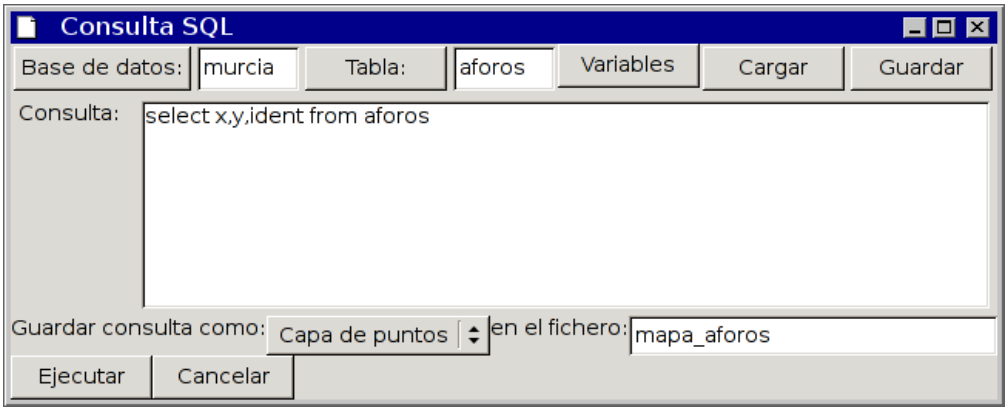

Figura 3. Ventana para conexión a PostgreSQL

Integrando R

R es uno de los programas de análisis de datos más utilizados en entornos de software libre, constituye un completo lenguaje de programación de muy alto nivel para aplicaciones de análisis de datos, una referencia completa puede conseguirse en Venables et al. (1999).

Uno de los puntos más potentes de R es el amplio conjunto de librerías desarrollado por diversos usuarios. Algunos de estos paquetes permiten integrar R con otros programas. La integración con la versión 5.4 de GRASS se hacía mediante la librería **GRASS** y la integración con la versión 6 se hace mediante la librería **spgrass6**. Esta librería se apoya en otra, **sp,** que aspira a convertirse en los formatos de referencia de R para datos espaciales.

La librería spgrass6 dispone de funciones para leer y escribir capas de información raster y vectoriales así como crear objetos que contienen la región de trabajo. Los objetos que crea son compatibles con los definidos en la librería sp.

Puesto que R es básicamente un lenguaje de programación, una interfaz gráfica para R deberá generar estas lineas de código. wxGRASS incorpora un botón de R que activa una ventana (figura 4) en la que escribir código de R que puede almacenarse, recuperarse o ejecutarse utilizando los botones que aparecen en la ventana.

A esta ventana se ha añadido una lista de selección que permite introducir en el programa código genérico para la realización de determinadas operaciones:

- Lectura de datos de GRASS (raster y vectorial)
- Escritura de datos de GRASS (raster y vectorial)
- Representación de gráficos (boxplot e histogramas)
- Modelos de regresión lineal

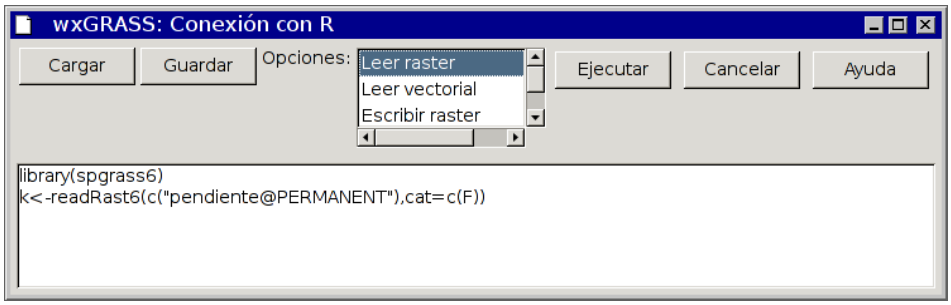

Figura 4. Ventana del módulo de acceso a R

El código del programa se almacena en un fichero temporal. A continuación el módulo abre R pasándole este fichero para su ejecución, tras la que se cierra sin intervención del usuario. Por tanto para poder visualizar los gráficos generados en el proceso es necesario almacenarlos en un fichero temporal que, tras cerrarse R se abre con el programa **display** del paquete **Imagemagick**, por lo que es necesario disponer de este para poder visualizar las salidas gráficas de R. Este paquete puede descargarse de http://www.imagemagick.org/script/index.php

Antes de la ejecución del código puede modificarse en este el nombre del archivo gráfico que almacenará los gráficos producidos. De este modo el usuario puede guardarlos como desee.

Integrando gstat

El programa Gstat (Pebesma y Wesseling, 1998) permite ejecutar análisis geoestadístico de un nivel de sofisticación (cokrigging, kriggeado en una vecindad, kriggeado por bloques, etc.) superior a la mayoría de los paquetes disponibles. Sin embargo carece de la capacidad de representar de forma gráfica los resultados. Recientemente se ha incorporado a la creciente lista de paquetes de GRASS uno que introduce la mayor parte de las capacidad de Gstat en R incluyendo la posibilidad de visualizar la información de forma gráfica (Pebesma, 2003).

El menú principal de wxGRASS incluye un menú de operaciones de interpolación. El kriggeado y la interpolación IDW (media ponderada por inverso de la distancia) se llevan a cabo mediante las funciones de la librería gstat de R.

Estas acciones se llevan a cabo de forma similar a como se opera en el módulo de conexión con R pero esta vez de forma transparente para el usuario. Este debe simplemente introducir en la ventana que aparece en la figura 5 los parámetros que se requieren e ir obteniendo los resultados,

En todo caso el funcionamiento de esta ventana es algo más complejo que el de las demás, el usuario de en primer lugar elegir el mapa de puntos con el que va a trabajar y la columna de la base de datos asociada, si en ese momento se pulsa el botón “Ejecutar” se obtiene un gráfico con el semivariograma muestral.

A continuación debe ajustarse este a un modelo, para ello basta con seleccionar el tipo de semivariograma, unos valores iniciales de sus parámetros (meseta, alcance, pepita) y el valor de distancia máximo a partir del cual no se tendrán en cuenta los

pares de valores para realizar la interpolación. El resultado de pulsar el botón "Ejecutar será entonces un semivariograma ajustado (figura 6).

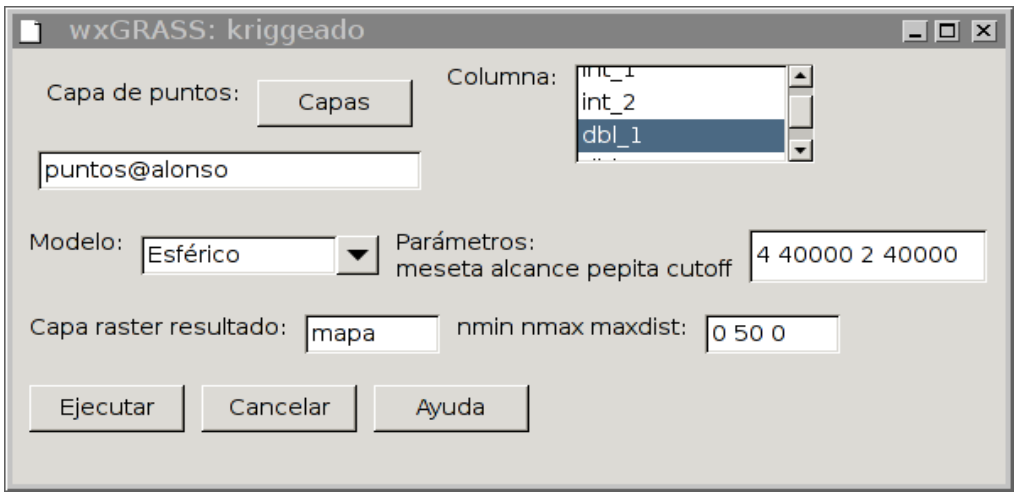

Figura 5. Ventana del módulo de interpolación por kriggeado

Finalmente, si el usuario indica un mapa raster de salida y los parámetros para determinar los puntos de interpolación local (mínimo número de puntos, máximo número de puntos y máxima distancia) obtendrá la interpolación por kriggeado ajustada a los parámetros de la región actual.

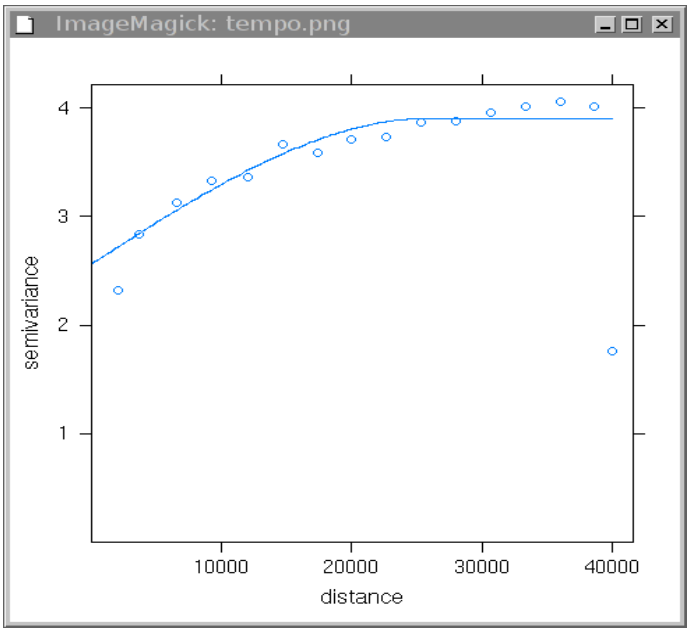

Figura 6. Semivariograma con modelo ajustado generado por la librería gstat de R a petición de wxGRASS.

Integrando GMT

GMT (Generic Mapping Tools) es un entorno de trabajo destinado a la maquetación de mapas para su impresión en papel. Pone a disposición del usuario un

conjunto de módulos orientados a la producción de cartografía a partir de datos codificados según los diferentes modelos lógicos (raster, vectorial, lista de puntos) habituales en los SIG. El resultado es un fichero postscript que contiene el mapa (Wessel y Smith, 2006).

El manejo de estos módulos en linea de comandos y la posibilidad de combinarlos (entre si y con otras herramientas UNIX) así como el elevado número de opciones de cada uno de ellos, convierte GMT en un entorno de maquetación de mapas extremadamente flexible. La contrapartida de esta flexibilidad es un lenguaje que puede llegar a ser bastante complejo y hermético. Existen proyectos para la creación de interfaces gráficas para GMT (Becker y Braun, 2001)

De este modo la integración de GRASS con GMT parece muy conveniente, especialmente si se hace en un entorno gráfico que facilite el acceso a las funciones de GMT. Para la integración de GMT con GRASS existen dos vías:

- Importación de la información raster o vectorial de GRASS a formatos compatibles con GMT (**r.out.bin**) y programas *ad hoc* con la información vectorial.
- Utilización del *driver* PNG de GRASS para generar la base del mapa, a esta base se pueden añadir, posteriormente, con GMT diversos elementos.

La segunda opción resulta preferible ya que el formato de dato vectorial de GMT resulta algo pobre y, por otra parte, en las últimas versiones de GRASS se han mejorado mucho las posibilidades de visualización de datos vectoriales.

La ventana de wxGRASS que actúa como interfaz para GMT (figura 7) permite al usuario introducir diversos parámetros como la resolución; el tamaño del mapa; la escala numérica, permite introducir la anchura del mapa o su escala numérica obteniendo una al modificar la otra; el tamaño de papel, la orientación, y los parámetros para confeccionar una escala gráfica (posición, anchura, longitud y número de intervalos.

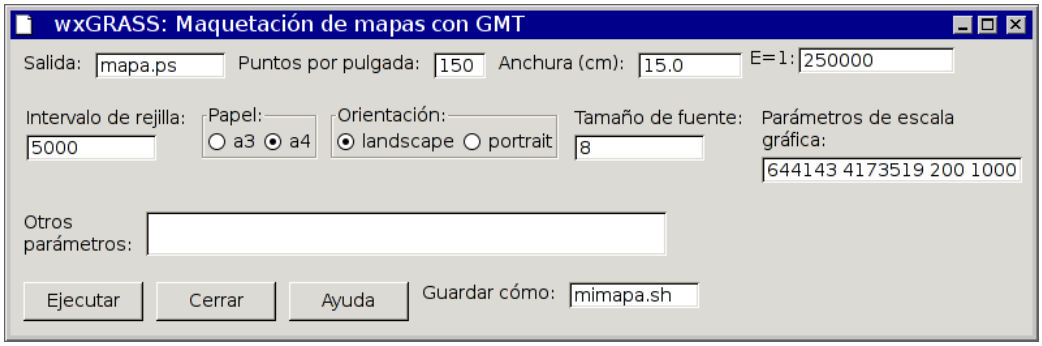

Figura 7. ventana de acceso a GMT.

El módulo recoge las órdenes gráficas utilizadas para componer el mapa situado en el monitor gráfico activo de GRASS y las envía al *driver* PNG. Posteriormente genera la base del mapa y le superpone el fichero de imagen creado. Finalmente añade otros elementos como la malla o la escala gráfica (figura 8).

En realidad este módulo es una interfaz para dos módulos desarrollados previamente por los autores para integrar GRASS y GMT (**d.out.GMT** y **escala.sh**). El primero calcula los parámetros básicos de la región y los que pasa el usuario para determinar los parámetros que se pasan al módulo psbasemap de GMT (el que se encarga de crear la base para el mapa). El segundo toma los parámetros que pasa el usuario y generan las instrucciones de GMT para superponer la escala gráfica.

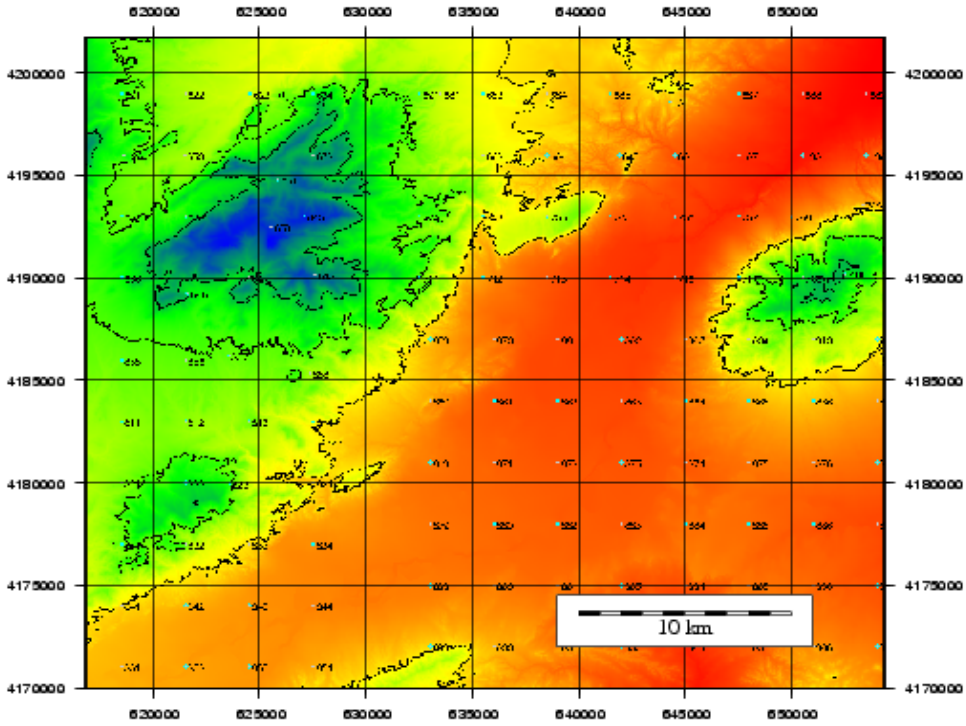

Figura 8. Mapa generado por GMT a petición de wxGRASS

Evidentemente, GMT dispone de muchas más posibilidades, por lo que el módulo permite también guardar en un fichero de texto (figura 9) el código utilizado para generar el mapa que el usuario puede posteriormente refinar. En dicha figura se aprecia como se inserta un fichero de imágen (**mapa.ras)** que ha sido previamente generado a partir de las órdenes gráficas utilizadas para crear la visualización presente en el monitor gráfico activo.

```
gmtset D_FORMAT %7.0f PAPER_MEDIA a4  ANNOT_FONT_SIZE_PRIMARY 8
psbasemap -R616850/654350/4169980/4201740 -JX15.0c/12.704c -B0 \
 -P -K -X5c >mapa.ps
psimage mapa.ras -W15.0c/12.704c -C0c/0c -O -K>>mapa.ps
psbasemap -R -J  -B5000g5000  -O -K>>mapa.ps
sh escala_km.sh 640000 4173519 200 1000 10 mapa.ps
psbasemap -R -J -B5000  -O>>mapa.ps
```

Figura 9. Código de GMT wwpara generar el mapa de la figura 8

RESULTADO E IDEAS PARA SEGUIR DESARROLLANDO

El uso de wxGRASS para la enseñanza de asignaturas de SIG ha tenido una respuesta muy positiva por parte del alumnado de la licenciatura de Ciencias Ambientales.

Tal como esta concebido hasta ahora, wxGRASS presenta unos módulos u otros en función del nivel del usuario. En algunos módulos, como puede ser el de kriggeado, sería preferible ir introduciendo al usuario los parámetros en función de su nivel de conocimiento. Se evitaría así el típico "y aquí que se pone",tener que explicar que esos valores se dejan por defecto y explicar a medias y sobre la marcha conceptos más avanzados.

Existe la posibilidad de utilizar la librería **tcltk** de R para que los programa de R que se ejecutan desde wxGRASS interactúen con el usuario, de este modo se podrían incorporar más opciones para análisis de datos complejos sin recargar excesivamente las ventanas principales de los módulos y se podría además mostrar gráficos sin necesidad de almacenarlos en fichero temporales.

Finalmente, la incorporación de pintar polígonos mediante tramas con el módulo **ps.map** de GRASS debería incorporarse al módulo de maquetación de mapas.

REFERENCIAS

- ALONSO SARRIA, F. (2006) *wxGRASS. Una interfaz gráfica de usuario para GRASS desarrollada con python y wxPython* XII Congreso Nacional de Tecnologías de la Información Geográfica
- ALONSO SARRÍA,F. y PALAZÓN FERRANDO,J.A. (2004) *Software libre para SIG* Curso impartido en el XI Congreso de Métodos Cuantitativos, SIG y Teledetección (http://www.um.es/geograf/sigmur/cursos/cursoGRASS.pdf)
- BECKER,T.W. Y BRAUN,A. (2001), *iGMT: Interactive Mapping of Geoscientific Datasets, User manual for version 1.2* 28 pp. (www-bprc.mps.ohio-state.edu/tools/igmt/manual_v1.2.pdf)
- BIVAND,R. y NETELER,M. (2000) *Open Source geocomputation: using the R data analysis language integrated with GRASS GIS and PostgreSQL data base systems* in Geocomputation 2000 (http://reclus.nhh.no/gc00/gc009.htm)
- GISIG *Guidelines for Best Practice in User Interface for GIS* (http://www.gisig.it/best-gis/)
- NETELER,M. y MITASOVA,H. (2002) *Open Source GIS. A GRASS GIS Approach* Kluwer, Boston/Dodrecht/London, 434 pags.
- PEBESMA, E.J. (2003) *Gstat: Multivariable Geostatistics for R* 3rd International Workshop on Distributed Statistical Computing Vienna, Austria. (http://www.ci.tuwien.ac.at/Conferences/DSC-2003/Proceedings/Pebesma.pdf)
- PEBESMA,E.J y WESSELING,C.G. (1998) *Gstat, a program for geostatistical modelling, prediction and simulation* en Computers & Geosciences, 24 (1), pp. 17-31 (http://www.gstat.org)
- RAPPIN,N. y DUNN,R. (2006) *wxPython in action* Manning 551 pp.
- VAN ROSSUM,G. & DRAKE, F.L. (2000): *Guía de aprendizaje de Python* (es.tldp.org/Tutoriales/Python/Tutorial-Python/)
- VENABLES, B, SMITH,D., GENTLEMAN,R. e IHAKA,R., 1999: *Notes on R: A Programming Environment for Data Analysis and Graphics (http://www.math.montana.edu/Rweb/Rnotes/R.html)*

♦ WESSEL,P. y SMITH, W.H.F., 2006: *The Generic Mapping Tools GMT.* http://www.soest.hawaii.edu/gmt/ddoc/pdf/GMT_Docs.pdf