Should wildlife biologists use free software?

Authors: Tufto, Jarle, and Cavallini, Paolo

Source: Wildlife Biology, 11(1) : 67-76

Published By: Nordic Board for Wildlife Research

URL: https://doi.org/10.2981/0909-6396(2005)11[67:SWBUFS]2.0.CO;2

RENDEZ-VOUS

Rendez-vous is a forum for promoting discussions among and between scientists and other professionals. New ideas and questions raised may be merely scientific presented in a scientific way, or they may be literary and political contributions to the environmental discussion. Rendez-vous articles typically deal with enthusiastic ideas and expressions of opinion which may lack firm data basis.

Should wildlife biologists use free software?

Jarle Tufto & Paolo Cavallini

Tufto, J. & Cavallini, P. 2005: Should wildlife biologists use free software? - Wildlife Biology 11: 67-76.

Jarle Tufto, Department of Mathematics Sciences, Norwegian University of Science and Technology, NO-7491 Trondheim, Norway - e-mail: jarlet@math.ntnu.no
Paolo Cavallini, Faunalia, Piazza Garibaldi 5, I-56025 Pontedera, Italy - e-mail: cavallini@faunalia.it

Corresponding author: Jarle Tufto

Received 13 August 2004, accepted 14 October 2004

Associate Editor: Nigel G. Yoccoz

As wildlife scientists and professionals, most of us rely on computer software in our daily work. We invest a large proportion of our time in learning how to use different kinds of software effectively to do our jobs. Choosing which software to learn in the first place is therefore an important long-term decision. Many of us rely on commercial software such as Word, Excel, Outlook and SPSS or SAS when we analyse data, correspond with our colleagues by e-mail and write papers.

In this paper, we will briefly discuss some important aspects of commercial software in general. More importantly, we will introduce the reader to a specific alternative to commercial software known as free software. Our impression is that the awareness of this phenomenon is much smaller among biologists than in other fields of science such as mathematics, statistics, physics and engineering. Free software is software which is distributed freely, typically over the Internet, under a licence known as the GNU General Public Licence (GPL; Free Software Foundation, Inc. 1991). Like commercial software licences, this licence relies on standard copyright law. The important difference is that the recipient of the software is granted a number of additional rights by the copyright holder which permit the recipient of the software to make changes to the source code, provided that any modified version of the source code is also made freely available under the same licence. This contract therefore guarantees that any improvement made to the software becomes freely available to the software's initial author. This potential reward becomes one of the main motivations for releasing the software as free software in the first place.

This licencing regime has dramatic consequences on the effectiveness of the software development process. To understand why, we need to understand approximately how the process of complex software development typically works in general. When an architect designs a new building, almost all the details of the design are typically completed before any building work is initiated. A common misconception about software development is that software is designed in a way similar to this. In reality, however, most software, both com-

mercial and free, changes through gradual small incremental modifications of the source code after reaching a certain level of complexity. Complex software is therefore better described as the result of an evolutionary process similar to evolution in biology (Torvalds 2001). In this sense software is similar to living organisms; no grand design plan or vision necessarily exists behind it (there is no need for neither a God nor Bill Gates).

Under the free software development model, anyone having an interest in the software can contribute changes ('mutations') to the source code. Except for very specialised software, the number of competent people interested in making such contributions is typically moderately or very large. Their motivation for doing so typically comes from a need to solve some real world problem as part of their ordinary job for which they are paid. The other motivation lies in their interest in seeing their own modification (and perhaps further improvements on it) becoming part of software they use on a daily basis. Depending on how the 'mutation' is judged by the community of other developers and users of the software ('natural selection'), the modification may be incorporated into the common version of the software.

In contrast, for commercial software, only a number of people employed inside the company which owns the software have access to and can change ('mutate') the source code, thus making the pool of 'genetic variation' on which selection can act much smaller. Also, the kind of 'natural selection' provided through traditional economic market mechanisms is likely to be less effective because selection acts not on single 'mutations' but on the software package as a whole. However, this is no more than some theories that have been put forward to explain free software as a phenomenon to disbelievers. The best evidence that free software is a very viable alternative to commercial software is provided through some examples.

Examples of free software

R

R is a software environment for statistical computing and graphics. The main component of the system is being developed by a core group of 16 members and is released under the GPL. It runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), different versions of Microsoft Windows and Mac OS X.

One of its main advantages is that the result of statistical procedures (e.g. estimated models) can be stored as objects and used as input in further computations. This is often very useful, for example when an assumption underlying some statistical model is violated, say the assumption normality of a standard regression model. In such cases, it is straightforward to fit a regression model to simulated data and use the output from each fit of the model (say an estimate or a test statistic) in further analyses to investigate the impact of different forms of violations of the model assumptions. An object can also be used as input to graphical procedures, for example applying the plot function to an estimated model object produces a number of relevant diagnostic plots (e.g. residuals, Q-Q plots, Cook's distances) depending on the class of the estimated model object. A large number of high and low-level functions are available for producing publication quality advanced graphics.

R has a simple menu-based graphical user interface from which different statistical methods can be selected (through the add-on package Rcmdr). Its functions can also be called directly from other programs, including the user-friendly spreadsheet Gnumeric. However, most interaction with R goes through a command-line based shell. A typical user works directly in this shell or writes a short script which is then pasted into the shell. Compared to menu-based systems which may appear more user-friendly at first sight, this may seem like a shortcoming. However, in the long run, the use of command-based scripts is typically more efficient in that the whole analysis, including generation of plots of the data, can be quickly rerun if some mistake is discovered or some assumptions change. Also, through the use of scripts the user has full knowledge over the exact details of what has been done if the analysis has to be revised or checked several months or years later.

R is a simple and effective formal complete programming language, and the R environment is therefore highly extensible. Being an interpreted language, R is not as fast as compiled languages such as C, but computationally intensive tasks can be written in C and then called from within R.

Statistical methods which are available through R include general linear models (regression and anova), generalised linear models, linear and non-linear mixed models, generalised linear mixed models, kernel density estimation and regression, survival analysis, analysis of contingency tables, time-series analysis, and essentially all basic statistical methods which one expects to find in an advanced statistical software package. In addition to a large number of statistical methods that come with the standard distribution of R (mostly written in the R language itself), a large number of add-on packages (461 at the time of writing) are freely available from CRAN, a network of Internet archives carrying R

software. One example is given in the section Geographic Information System on page 73-74.

Functionality for working with standard statistical distributions (random number generation, density functions, cumulative density functions and quantiles) is built into R. While being mostly oriented towards statistical inference, R also has much of the functionality found in software packages such as Matlab and Scilab (http://scilabsoft.inria.fr/e) such as functions for numerical solution of ordinary differential equation, numerical integration, optimisation and linear algebra. R is therefore a highly suitable platform for carrying out simulation for both deterministic and stochastic models and for doing numerical analyses of ecological models (Petzoldt 2003).

R is very similar to the commercial software S-plus, and most code written for S-plus can run unaltered in R. Because R is widely used and preferred among leading professional statisticians, a lot of new methodology is often implemented first as add-on packages in R before appearing in commercial statistical software.

To summarise, R can be recommended to both beginners and advanced users of statistics in various fields including wildlife biology. Of course, R will not teach you basic statistics. A typical user in wildlife biology will probably benefit from reading Crawley (2005), Dalgaard (2002) or Venables & Ripley (2002) as well as any introductory statistics textbook. The software can be downloaded from www.r-project.org.

BUGS

R has been mostly oriented towards the classical (or frequentist) school of thought within statistics. According to the frequentist view, one is not allowed to assign probabilities to different hypotheses or values of unknown parameters of a statistical model, because the parameters of a model are fixed quantities and not random variables. Instead, all statistical inference is based on the frequency distribution of various functions of the data such as estimators or test statistics.

Bayesian statistics offers an alternative to the frequentist tradition and has seen a tremendous revival during the last 20 years, mainly as a result of advances in algorithms for carrying out Bayesian inference in practice. According to the Bayesian point of view, probabilities represent our often limited state of knowledge about nature. In this sense, all probabilities are subjective rather than objective. Thus, there is no fundamental difference between assigning probabilities to competing scientific theories and to the result of a coin-flipping experiment.

Bayesian statistical inference is based on Bayes theorem which states that the posterior, the probability of different parameter values given the data, $P(\theta|X) \propto P(X|\theta)P(\theta)$, where $P(X|\theta)$ represents the probability of the data given different parameter values and $P(\theta)$ represents prior subjective beliefs of the experimenter. Within this framework so-called graphical models can be formulated where data and parameters are represented as nodes in a graph. Parameter nodes have unknown values and these are updated repeatedly according to their conditional distribution given neighbouring nodes. Keeping data nodes fixed, it then turns out that the joint distribution of all parameters then converges towards the desired posterior distribution (Gilks et al. 1996).

This framework offers enormous flexibility in terms of the large variety of realistic models that can be analysed (Best et al. 1996). Programming the above simulation algorithm (known as the Gibbs sampler) is in many cases a somewhat tedious task requiring a fair amount of knowledge in probability theory. The software BUGS (Bayesian inference Using Gibbs Sampling) creates and runs this simulation algorithm given a specification of the model provided by the user in a specialised language.

BUGS is freely available on the Internet and runs on a number of platforms (Spiegelhalter et al. 1996). A version with a graphical user interface (WinBUGS) which includes some additional sampling algorithms is available on the Windows platform. BUGS (including WinBUGS) integrates well with R through a number of R packages. In September 2004, the source code of BUGS became open and a beta version of OpenBUGS is now available under the GPL (see http://mathstat. helsinki.fi/openbugs/).

Examples of its use in the ecological literature include Meyer & Millar (1999), Bjørnstad et al. (1999) and Tufto et al. (2000).

OpenOffice

OpenOffice.org (OOo) is a free software office suite compatible with the file formats of the Microsoft Office package. It runs on Microsoft Windows, Linux, Sun Solaris, Mac OS X and FreeBSD. It has essentially the same functionality as the more famous (and expensive) package (including a word processor, a spreadsheet and a slide presentation program), with minor differences, both positive and negative. It has also a program for drawing. Support for non-English languages (from Arabic to Vietnamese, including bidirectional and vertical writing) is very extensive in OpenOffice, whereas the grammar checking function is missing (the spell-checker, however, is equivalent). Writing math for-

mulas is more flexible and reliable than in commercial word processors. One very convenient feature is the ability to export and send via e-mail Acrobat Reader (pdf) files natively, without the need for additional software. For scientific use, the main problem in using it is the lack of a fully fledged bibliographic software (like EndNote or similar products) integrated with it; it comes with a very simple one, unlikely to be useful for professional use, whereas more robust programs are well integrated with LaTeX rather than with OOo. A solution is under study, and will probably be ready in the next major version of the program. It does not have an integrated database, but acts as a front end to a variety of database engines, allowing the use of different sources of data, local as well as remote, as elements for articles, graphs and spreadsheets (Fig. 1). The software can be downloaded from www.openoffice.org.

LaTeX

LaTeX is a document preparation system quite different from normal word-processing programs, excellent for preparing scientific and technical documents. It was created in 1984 by Leslie Lamport (Lamport 1986). It uses TeX, a typesetting system created by Donald Knuth and released in 1982, as its formatting engine.

In most commercial word-processing software, the author sees a close approximation, on the fly, of what the document being edited will look like when printed. This principle, however, known as WYSIWYG (what you see is what you get), discourage most authors from using so-called logical markup of different components of the documents such as section headings, figures and equations, in part, because such logical markup according to the WYSIWYG principle should be hidden from the authors view of the document. It is well known that instead most authors tend to format their section headings physically, and to number equations, figures and tables as well as references to these manually. Users who attempt to use 'advanced' features for logical markup in software such as Microsoft Word typi-

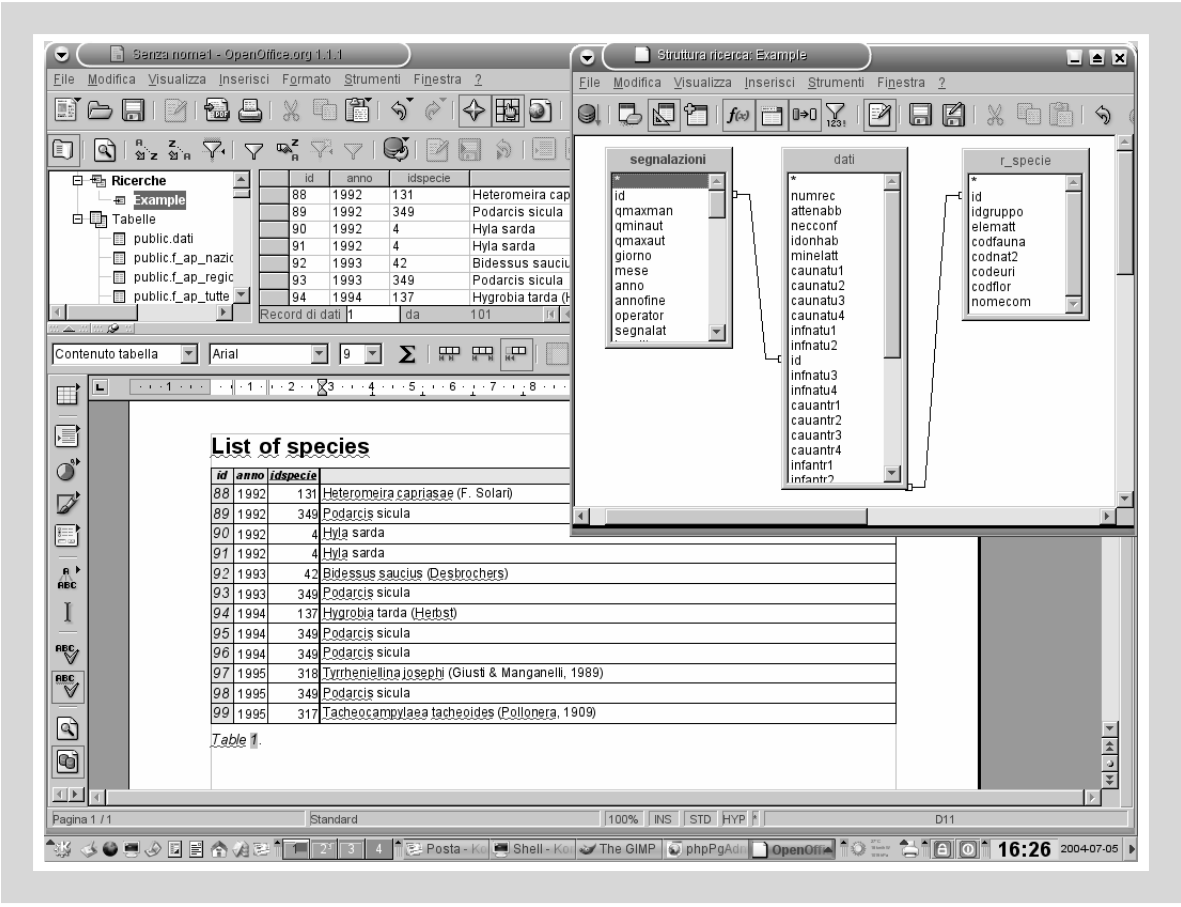

Figure 1. Screenshot of the Italian version of OpenOffice showing some word-processing and database front end functionality. Versions for an increasing number of other languages are available.

cally give up because it is viewed as too unreliable and confusing.

LaTeX is based on a different philosophy in that the author's view of the document being edited differs from that of the final document as seen by the reader. The main source file for the document is a simple plain text file which is then compiled, typeset and translated into a number of formats including postscript, pdf and html. Logical markup of different parts of the document is done by entering simple codes in the source file.

LaTeX has simple, straightforward mechanisms for automatic numbering of equations, figures, tables and sections, for making references to these, and for generating a list of cited literature (from a BibTeX bibliographic database file) in a citation style chosen by the author. It is therefore highly suitable for creating and maintaining large complex documents with many cross references. A number of predefined layout styles can be chosen for creating e.g. articles, letters and books. Many other layout styles and add-on packages are available from CTAN, an Internet network carrying TeX-related software. Among these are packages such as Prosper and Beamer for creating PowerPoint-like hypertext presentations in the pdf file format.

LaTeX is highly suitable for writing documents containing mathematics, in fact, so much so that it has become a standard document preparation system within mathematics, statistics and physics. Almost all major journals and many books within these research fields are typeset using systems based on LaTeX. Its use is spreading to more of the theoretically oriented biological journals. Again, mathematical equations are entered through the use of codes in the main plain-text source file. These codes are then translated into nicely typeset equations in the final document. Although the code needed to write a complex equation in LaTeX may look archaic to beginners, experienced users find LaTeX to be a far more efficient and reliable solution in the long run than the WYSIWYG menu-driven approach found in competing software. This is not to say that LaTeX is difficult to learn; Oetiker et al. (2003) is a good introduction for interested readers.

The fact that the source file for a document is in an open plain-text format means that other software can easily be written to generate and process text formatted as LaTeX. The R packages Hmisc, Design and xtable, for example, use this mechanism to export different types of objects (analysis of variance tables, other types of data of matrix form, various fitted models) to LaTeX. Also see the next subsection on concurrent version control systems.

The most widely used software implementations are TeTeX on UNIX systems and MikTeX on the Windows platform; both are freely available on the Internet. Recommended software for editing LaTeX source files include Winedt (a commercial Windows-only editor) and Emacs, both of which have special editing modes for LaTeX.

CVS and Subversion

If you work in a team on some project writing an article or a book, analysing a set of data or writing software, then CVS (Concurrent Version Control) or its replacement, a system known as Subversion, is for you. Consider the process of writing a manuscript in a team. Typically, the present revision of the manuscript is sent by e-mail to each coauthor, each coauthor makes a number of additions and corrections which are then sent back to the first author who then edits these changes into the main version of the manuscripts. This working mode is quite inefficient in that different coauthors will be unaware of each other's changes, and in that changes must be merged manually. Alternatively, the manuscript is passed around by e-mail, each coauthor sits on the manuscript for some time while changes are edited in, until the manuscript is passed on to the next coauthor. This working mode is also inefficient in that only one author works on the manuscript at the time, keeping the other authors waiting.

Both CVS and Subversion (which will be discussed very briefly below) aim at providing a solution to these inefficiencies. A repository of all files shared among project participants is first set up on a server. Each participant can then check out a working copy of the project files from the repository (using appropriate client software) and then work concurrently on their file copies. Changed files are then committed (using the client software) back into the repository by different coauthors. When someone, say Harry, attempts to commit his changes to the repository he may be notified by Subversion that his working copy is out of date because Sally committed her changes to the repository first. Sally's changes are therefore downloaded into Harry's working directory. In most cases Sally's changes will then be merged with Harry's changes automatically by the Subversion system. However, if the changes made to a file overlap (this is known as a conflict), both Harry's and Sally's changes appear as a marked region in the new version of Harry's working file copy, and Harry must resolve the conflict by manually editing the conflicted region of the file (possibly after first making a phone call to discuss things with Sally). After this the resolved working copy can be committed as usual.

Both binary files and text files can be controlled by

Subversion. However, automatic merges only work on plain text files, for example, R-scripts and program source files, LaTeX source files, BibTeX bibliographic database files, or files of field data if these are stored as plain text files. Conversely, Subversion can not merge binary files such as Word documents or Excel spreadsheets. Subversion is of course completely unaware of the semantic meaning of different files. Thus, the state of a merged file may not make sense and it is the responsibility of the participant carrying out the merge (Harry in the above example) to check that the merged file makes sense before committing his resolved merged file.

We believe that concurrent version control systems like CVS and Subversion provide an ideal environment for scientific collaborative work. Unlike lead teams where the team collectively or a team leader decides what each participant should do, concurrent version control systems create a loosely controlled environment where each participant can test out new ideas and contribute changes to any part of the project at any time. It facilitates closer peer review within the team. CVS and Subversion are used extensively in free software development and are probably one of the keys behind the success of this development model. Client and server software for various operating systems and extensive documentation is available at subversion.tigris.org.

GNU/Linux

GNU/Linux is a UNIX-like operating system which runs on a large number of hardware platforms including PCs and Macs. Its development began as the GNU project launched in 1984 with the aim of developing a complete free software-based UNIX-style operating system. At the time, UNIX was already a well-designed multi-user, multi-tasking, networking operating system unlike MS-DOS and many later versions of Microsoft Windows. Many basic command-line based tools and utilities have therefore remained relatively unchanged since then.

Linux is really only the central component, the operating system kernel, of GNU/Linux. The kernel is an essential software component of any operating system acting as a layer between the hardware of the computer and software applications. Development of the Linux kernel was started by Linus Torvalds in 1991.

Various distributions of GNU/Linux exist such as Fedora (fedora.redhat.com; a widely used distribution sponsored by Redhat), Debian, Mandrake and Suse. Linux typically comes bundled with a large variety of free software including compilers and software development tools for a large number of programming languages, command shells and scripting languages, different editors, spreadsheets, word-processors, LaTeX, e-mail clients, web-browsers, the Apache web-server, database servers, image processing software, multimedia tools, octave and hundreds of other utilities. Today, most distributions come with graphical desktop user interfaces such as KDE or Gnome. These are quite similar to the user interfaces found in different versions of Microsoft Windows. Thus, to an inexperienced user, although the user interfaces differ somewhat, learning basic skills like file management and simple tasks like reading e-mail and word processing in Linux is probably not any more or less difficult than learning the same skills in Microsoft Windows.

While ever more polished graphical desktop user interfaces certainly have been good selling points for new versions of Microsoft Windows, recent usability studies suggest that the command-line based interface preferred by more experienced users of UNIX-like operating systems is seen as simpler and less confusing also to beginners (Wareham 2004). In essence, a command-line interface, it is argued, puts the user in control of the computer rather than the other way around.

From a pragmatic point of view, one reason for choosing Linux over Windows is the much greater availability of high-quality free software on the Linux platform. Installing modern distribution of GNU/Linux is in most cases very straightforward except perhaps on brand new laptop models with non-standard hardware. These kind of hardware incompatibilities are typically resolved within a period of about six months (see www.linux-laptop.net for details). People interested in seeing Linux in action on their own PC without permanently installing the OS may download the Knoppix CD (from www.knoppix.org) which is a complete Linux distribution which can be run off a bootable CD without actually installing any of the software on the hard drive.

For users who depend on both Linux and Windows software a number of solutions exist. First, it is possible to create a dual-boot system during installation with Linux coexisting with Windows on a separate partition of the hard drive. The Windows OS and Linux can also be made to run concurrently on the same PC through commercial products such as Vmware and Win4lin. Windows applications can also be run on top of Wine, a free software 'compatibility layer' allowing windows applications to be run under Linux without installing the actual Windows OS. Linux applications can also be run on a remote server and displayed on Windows based (and of course other Linux based) PCs through the networking functionality of X Window

System. Finally, file sharing in a mixed environment of Windows and Linux machines is straightforward through the Samba file-server software.

Regardless of whether you are running Linux or Windows, if your computer is permanently connected to the Internet, it is highly recommended to install security fixes once they appear. On Linux distributions such as Fedora and Debian, this, and installation of new software in general, is very easy because almost all software is distributed in a special standardised package format through a network of Internet software repositories. With tools such as apt-get or yum, downloading and installing the newest versions of all packages currently installed on a system is therefore a simple matter of entering single commands in the shell, or the process can be automated completely and run in the background, say once a day, through the use of so-called cron-jobs.

Geographic Information System

The Geographic Information System (GIS) market is dominated by ESRI, with the two products ArcView and ArcInfo. While the latter is a large and powerful (but very expensive) package, the former is essentially a viewer of geographic information. Its limited capabilities have been variously enhanced by programming specific extensions.

Many free software packages are available for the same analyses. Of these, the Geographic Resources Analysis Support System (GRASS) is the most comprehensive and powerful. Historically born as a raster program, it now has good vector (topological) support. As with many free packages, it has a modular structure into which it is very easy to plug new routines, programmed in a variety of languages (from C to shell scripts). It has a complete set of commands for the creation, manipulation and visualisation of raster and vector data, e.g.:

- Raster analysis:
  - Automatic rasterline and area to vector conversion
  - Buffering of line structures
  - Cell and profile dataquery
  - Colourtable modifications
  - Conversion to vector and point data format
  - Correlation / covariance analysis
  - Expert system analysis
  - Map algebra (map calculator)
  - Interpolation for missing values
  - Neighbourhood matrix analysis
  - Raster overlay with or without weight
  - Reclassification of cell labels
  - Resampling (resolution)
  - Rescaling of cell values
  - Statistical cell analysis
  - Surface generation from vector lines
- 3D-Raster (voxel) analysis:
  - 3D data import (intuitive ASCII: x y z Format)
  - 3D masks
  - 3D map algebra (r3.mapcalc)
  - 3D interpolation (IDW, Regularised Splines with Tension)
  - 3D Visualisation (isosurfaces)
  - Interface to Vis5D visualisation tool
- Vector analysis:
  - Contour generation from raster surfaces
  - Conversion to raster and point data format
  - Digitising with board or on screen (scanned raster image) with mouse
  - Reclassification of vector labels
  - Superpositioning of vector layers
- Point data analysis:
  - Delaunay triangulation
  - Surface interpolation from spot heights
  - Thiessen polygons
  - Topographic analysis (curvature, slope, aspect)
- Image processing:
  - Canonical component analysis (CCA)
  - Colour composite generation
  - Edge detection
  - Frequency filtering (Fourier, convolution matrices)
  - Fourier and inverse fourier transformation
  - Histogram stretching
  - IHS transformation to RGB
  - Image rectification (affine and polynomial transformations on raster and vector targets)
  - Ortho photo rectification
  - Principal component analysis (PCA)
  - Radiometric corrections (Fourier)
  - Resampling
  - Resolution enhancement (with RGB/IHS)
  - RGB to IHS transformation
  - Texture oriented classification (sequential maximum *a posteriori* classification)
  - Shape detection
  - Supervised classification (training areas, maximum likelihood classification)
  - Unsupervised classification (minimum distance clustering, maximum likelihood classification)
- DTM-Analysis:
  - Contour generation
  - Cost / path analysis
  - Slope / aspect analysis
  - Surface generation from spot heights or contours
- Visualisation:
  - 3D surfaces with 3D query

Figure 2. Given the Digital Terrain Model, GRASS can identify automatically river basins with a single command (i.e. r.watershed).

- ◦ Colour assignments
- ◦ Histogram presentation
- ◦ Map overlay
- ◦ Point, raster and vector maps
- ◦ Zoom / unzoom -function

• Map creation:

- ◦ PNG maps
- ◦ PPM-image maps
- ◦ Postscript maps
- ◦ HTML maps

• More specialised programs currently included in the distribution include:

- ◦ Erosion modelling
- ◦ Rainfall-runoff and hydrological modelling
- ◦ Landscape structure analysis
- ◦ Solution transport
- ◦ Watershed analysis
- ◦ Wildfire spread simulation
- ◦ Evaluation of solar radiation.

For a complete list see grass.itc.it. At the time of writing, a new version (6.0), with much improved user interface and vector capabilities, is about to be released.

One of the strongest points in using this software for scientific purposes is its deep integration with the statistical package R (see above) and relational databases (especially PostgreSQL; see below). From inside GRASS it is possible to read data from a relational database (e.g. locations with associated data), call R functions, run statistical analyses (e.g. logistic multivariate regression and kriging), display the results, and store them back in the database. Many operations can be done with different methods (e.g. random points can be generated either by GRASS, by R or by the database, and then displayed in GRASS), thus overcoming limitations of single applications. This flexibility gives the researcher a power that is difficult (and expensive) to replicate in commercial systems, by necessity less inclined to interact with each other.

On the other hand, there are still rough edges in using the program; generally speaking, some complex tasks are surprisingly easy. The calculation of watersheds in a complex environment is, for example, done by just a single command (r.watershed; Fig. 2), whereas some of the apparently easy operations are rather convoluted (e.g. printing some combination of raster and vector maps with legends). For some of these tasks, simple (yet powerful) programs for desktop mapping (e.g. QuantumGIS and Thuban) are available. In the near future, Quantum GIS will integrate with GRASS, acting as its graphical front end.

During the last few years, the availability of mapping tools through the Internet has increased greatly, allowing the dissemination of results of research and management towards the public at large. Several commercial products are available, but high licencing cost prevents most research groups from their widespread use. Among free software, the most powerful tool is University of Minnesota's Mapserver (available at: mapserver.gis.umn.edu). It is a development environment, where one can set up from simple to very complicated applications (many fully functional examples can be browsed at mapserver.gis.umn.edu/gallery.html). Thanks to the GDAL/OGR libraries, it can use geographic data in a variety of formats and its performances are excellent as it can serve thousands of simultaneous accesses.

With the growth in quality and quantity of geographical information, it is becoming increasingly important to manage the data with appropriate tools, and databases are particularly suited for these tasks; until recently, however, databases could not store geographic information as such (including e.g. projection systems and datums), but tools are now available for this (see the next section).

The database issue

The growing power and friendliness of spreadsheets make the complexity of relational database systems (RDBMS) rather unattractive. However, it is clear that handling more than a few thousand records is unfeasible by spreadsheets. Besides performance and stability issues (treating millions of records with many concurrent accesses is quite feasible with RDBMS), the main advantage of using an RDBMS resides probably in the possibility of linking tables of data, thus enhancing internal consistency and reducing redundancy. Structured Query Language (SQL) is an industry standard language used to interact with relational databases.

The whole structure is composed of three layers: the data (stored as tables), the management system (the DB engine), and the front end (either graphical or not). In professional DBMS, the three layers are completely distinct, and often several front ends are available. By contrast, the most popular commercial RDBMS for small applications is Microsoft Access, where the three layers are completely integrated. This approach has the advantage of hiding part of the underlying complexity to final users, at the expense of reliability and performance. The Access 'file' is in fact a runtime DB engine, stored together with data; the resulting file is therefore much larger (a 10x factor is common) and more prone to internal errors.

Among free software, the choice of RDBMS is wide; especially popular are MySQL (www.mysql.com) and PostgreSQL (www.postgresql.org). From a user's point of view, differences among the different systems are not great; the main reason for a choice is the interaction capabilities with other programs. In this respect, PostgreSQL is generally superior (as it is the case for GRASS; see above). Administration of an RDBMS is a rather complicated and tedious task, because it requires the manager of the DB to grant privileges to users for specific tasks. While undoubtedly useful when many users have access to a single database, and for security across the web, this is overkill for single users. In such cases, simpler RDBMS are available (e.g. SQLite); these programs have no administration overhead, but are not suited for web applications.

One great advantage of RDBMS consists in the fact that data remain available for a variety of other applications, from GIS to spreadsheets and word processors, without the need for conversion and the associated risk of data duplication. In synthesis, the data are managed by a single system, and many programs act as front ends to these. Specialised data management can also be done with simple programming languages (among which php and perl are probably the most popular), designing e.g. ad hoc forms, reports and web interfaces.

Databases can now understand and manage geographic information; an extension for PostgrSQL (Post GIS with GEOS; see postgis.refractions.net) allows not only the management of data, but also their display and interrogation (questions like: "find all the points within distance x to rivers" can be answered directly from the database). This is a very powerful tool which can solve many of the problems that normally require a specialised GIS.

Recommendations

In conclusion, should wildlife biologists use free software? Our answer is generally positive, both for scientists and for professionals. The main advantages are the availability of powerful tools, the low cost of acquisition (especially important for underfunded universities and young professionals; students may profit from having the programs used at the university and in the profession legally installed on their private computers), and the ample scope for customisation.

Free software could not just substitute commercial software, but also offers a chance for cultural growth (and scientific/professional empowerment) among wildlife biologists, both because we can go deeper in understanding the information technology (and we know that many of us are lagging behind in this sense), and because we do not have 'entrance barriers' (i.e. costs for acquisition), so we can try and test all sorts of different techniques.

So what are the problems? Of course switching has a cost: some training and assistance may be necessary, and learning new programs takes some time. In addition, programs for some very specific application might not be available.

We therefore advise our colleagues to seriously evaluate this option. Several strategies are available for switching, e.g.:

- start moving to free software piece by piece; good starting points could be OpenOffice and R, as they are available also for Windows platforms;

- install a Linux distribution (on a test machine or on a dual-boot machine), and learn and test all the necessary programs before switching.

In any case, the help from Linux user groups and, for problems relative to specific applications, from their communities of users will make the transition much easier.

Acknowledgements - without the help of the community of GNU/Linux users all this would not have been possible. Giovanni Bacci and Cesare Furlanello provided useful comments on earlier drafts of this paper, and we thank Leonardo Lami for preparing the GRASS figures and Emilia Venturato for OpenOffice figures and comments.

References

Best, N.G., Spiegelhalter, D.J., Thomas, A. & Brayne, C.E.G. 1996: Bayesian analysis of realistically complex models. - Journal of the Royal Statistical Association 159: 323-342.

Bjørnstad, O.N., Fromentin, J.M., Stenseth, N.C. & Gjosaeter, J. 1999: Cycles and trends in cod populations. - Proceedings of the National Academy of Sciences 96: 5066-5071.

Crawley, M.J. 2005: Statistics: an introduction using R. - John Wiley & Sons, 288 pp.

Dalgaard, P. 2002: Introductory Statistics with R. - New York, Springer Verlag, 267 pp.

Free Software Foundation, Inc. 1991: GNU General Public License, version 2. - Available at: www.gnu.org/copy left/gpl.html.

Gilks, W., Richardson, S. & Spiegelhalter, D. (Eds.) 1996: Markov Chain Monte Carlo in Practice. - Chapman & Hall, London, 486 pp.

Lamport, L. 1986: LaTeX - A Document Preparation System. - Addison-Wesley, 242 pp.

Meyer, R. & Millar, R.B. 1999: Bugs in bayesian stock assessments. - Canadian Journal of Fisheries and Aquatic Sciences 56: 1078-1087.

Oetiker, T., Partl, H., Hyna, I. & Schlegl, E. 2003: The not so short introduction to LaTeX 2e. - Available at: www.ctan.org/tex-archive/info/lshort/english/lshort.pdf.

Petzoldt, T. 2003: R as a simulation platform in ecological modelling. - R News 3: 8-16.

Spiegelhalter, D., Thomas, A., Best, N. & Gilks, W. 1996: BUGS 0.5 Bayesian inference Using Gibbs Sampling. Version 0.5, (version ii). - Cambridge: MRC Biostatistics Unit. Available at: www.mrc-bsu.cam.ac.uk/bugs/

Torvalds, L. 2001: Re: Coding style - a non-issue. - Available at: lwn.net/2001/1206/a/mutation.php3.

Tufto, J., Sæther, B-E., Engen, S., Arcese, P., Jerstad, K., Røstad, O.W. & Smith, J. 2000: Bayesian meta-analysis of demographic parameters in three small temperate passerines. - Oikos 88: 273-281.

Venables, W.N. & Ripley, B.D. 2002: Modern Applied Statistic with S. - Springer, New York, 495 pp.

Wareham, R. 2004: The command line - the best newbie interface? - OS News. Available at: www.osnews.com/story.php? news_id=6282.