A joint newsletter of the Statistical Computing & Statistical Graphics Sections of the American Statistical Association.

Summer 2004
Vol.15 No.1

Statistical COMPUTING & GRAPHICS

A WORD FROM OUR CHAIRS

Statistical Computing

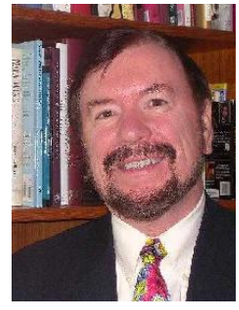
Carey Priebe is the 2004 Chair of the Statistical Computing Section.

Statistical computing, as we all know, is becoming ever-more fundamental to applications AND research in statistics. The days when the preferred methodologies were those for which closed-form expressions can be derived — and these methodologies were preferred for precisely that reason — are long gone. Applications for which data are high-dimensional, sample sizes are massive, or both are driving the development of imaginative, computation-based solution methodologies. The future of statistics is exciting indeed; especially for those of us involved in statistical computing ... and Statistical Computing!

I hope to see you all at The 2004 Joint Statistical Meetings, August 8-12 in Toronto, where we will once again enjoy a stellar scientific program (thanks to our Program Chair for 2004, David Madigan, among many others) and an always-fun mixer with our friends in the Section on Statistical Graphics.

Two other meetings of interest — there are, of course, many others — are Interface 2004, occurred May 26-29 in Baltimore, with a "Computational Biology and Bioinformatics" theme, and the International Statistical Institute (ISI) International Association of Statistical Computing (IASC) COMPSTAT 2004, is coming up August 23-27 in Prague. To get an indication of IASC's priorities, notice that the keynote lectures for the Prague meeting include:

Statistical Graphics



Jim Gentle is the 2004 Chair of the Statistical Graphics Section.

It's that time of year again. Time to make final preparations for JSM (or time to start thinking about it if you haven't yet.) On the ASA website giving the program, there are 23 sessions with sponsorship from our Section, plus posters on two days, roundtable luncheons on three days, and a continuing ed course on Tuesday (by Rich Heiberger and Burt Holland). Also, don't forget the Section business meeting, joint with the Section on Statistical Computing, at the usual time on Monday evening.

The ASA's website makes it easy to view the JSM program — but it still doesn't solve my annual problem — how to choose between conflicting sessions. There are 8 time slots with two or more Section-sponsored sessions. However, there is no time slot without at least one Section-sponsored session; there was no room to maneuver. Maybe the meetings just need to be longer — no, no; that is not a good idea. We just have to live with our embarassment of riches.

While as I write this, JSM 04 is on our minds, when you read this, you should be thinking ahead to JSM 05. A major reason we have so many good sessions at the meetings is because people — you — suggest topics for sessions. Our Program Chair for JSM 05 is Dan Rope; deluge him with suggestions!

This is also the time of year to congratulate various

CONTINUED ON PAGE III

Prosper: A LaTeX documentclass for presentations

Rouben Rostamian
Department of Mathematics and Statistics
University of Maryland, Baltimore County
Baltimore, MD 21250, USA.
rouben@math.umbc.edu

Introduction

Microsoft's PowerPoint is a ubiquitous tool for screen-projected presentations in these days, but it is not the only such tool. In this article I will describe Prosper, a LaTeX-based alternative. The output of Prosper is a PDF file which is suitable for both viewing on a computer screen and projecting on a wall screen for presentation.

I am writing with the assumption that those interested in Prosper will be using it on a UNIX or UNIX-like platform, such as Linux. Chances are that the guidelines provided here may be applicable to other platforms as well, however since I have no experience with anything other than UNIX, I will comment no more on such possibilities.

I also assume that you, the reader, have the LaTeX technical typesetting system installed on your computer and know how to use it. This is not a LaTeX tutorial. Prosper is a set of macros layered on top of LaTeX. Actually, Prosper is layered on top of the `seminar`, and `pstricks` and `hyperref` packages, which in turn are layered on LaTeX.

Prosper was created by FRÉDÉRIC GOUALARD and is in active development. More information about Prosper is available online at its development site at `http://sourceforge.net/projects/prosper/`.

This article is a brief tutorial on Prosper. I will describe its syntax, give some examples, and provide hints for its use. My intention is to get you started with using Prosper quickly. For that reason, I will skip over a few secondary details. If you find the instructions in this tutorial inadequate for your needs, then you should consult Prosper's manual to see whether Prosper provides what you need.

Installation

To use Prosper, you need a fully functional LaTeX installation. A good deal of information about LaTeX is available on the WWW at `http://www.latex-project.org`. (Or `teTeX` at `http://www.tug.org/teTeX/`, or `vtex` at `http://www.micropress-inc.com/linux/`.) If you are a Linux user, you may install LaTeX, as well as the other utilities mentioned below, in a trivial way using your package manager tool. On other platforms the installation may be just as easy or only a bit more cumbersome. Ask for your sysadmin's help if necessary. All the required software is widely available and can be obtained at no cost.

Chances are that your LaTeX installation comes with the `dvips` utility which is used to convert LaTeX's output to the PostScript format. In the unlikely event that it doesn't, you will have to install it as well. Information about `dvips` can be found at `http://www.radicaleye.com/dvips.html`.

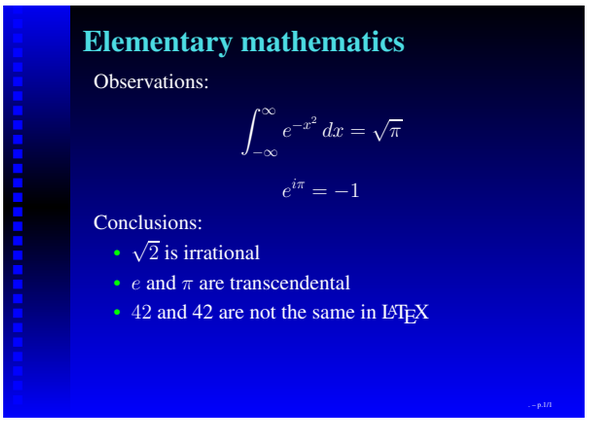
You will also need the `ghostscript` utility to convert the output of `dvips` to the PDF format. Various flavors of ghostscript are available. See `http://www.ghostscript.com` for details.

To view your slides, you will need a PDF viewer such as the Adobe Reader which you may download from `http://www.adobe.com/products/acrobat/readstep2.html`.

Finally, you will need to install Prosper itself. Refer to `http://prosper.sourceforge.net` for information about Prosper, and follow the links there to download the complete Prosper package, `prosper-1.00.4.tar.gz`. Read the INSTALL file in that package for the pretty straightforward installation instructions.

Getting started

The figure below shows a representative Prosper slide. Following it is the contents of the corresponding source file `sample.tex`. I have reduced things to a bare minimum to bring out the simple structure of a Prosper input document. A typical Prosper input file will contain several slide definitions and possibly a titlepage slide.



Every Prosper slide has a mandatory slide title. The

line `\begin{slide}{Elementary mathematics}` in `sample.tex` specifies "Elementary mathematics" as the slide title. It is an error to omit the slide title.

The `\begin{slide}{...} ... \end{slide}` block may be repeated any number of times to produce a sequence of slides.

file: sample.tex

```
\documentclass[pdf,azure,slideColor,
  colorBG]{prosper}
\begin{document}

\begin{slide}{Elementary mathematics}

Observations:
\[ \int_{-\infty}^\infty e^{-x^2} \,
  dx = \sqrt{\pi} \]
\[ e^{i\pi} = -1 \]

Conclusions:
\begin{itemize}
    \item $\sqrt{2}$ is irrational
    \item $e$ and $\pi$ are
                transcendental
    \item $42$ and 42 are not the
                same in \LaTeX
\end{itemize}
\end{slide}

% insert more \begin{slide}{...} ...
%             \end{slide} blocks here
% to make a sequence of slides
\end{document}
```

Compiling the input into PDF

To compile the `sample.tex` into the PDF file, do:

```
# create sample.dvi
 latex sample.tex
# create sample.ps
 dvips -t a4 sample.dvi
# create sample.pdf
 ps2pdf -dPDFSETTINGS=/prepress \
        sample.ps
```

The `latex` command runs LaTeX over the input file and produces a so-called device-independent `sample.dvi`. The `dvips` converts `sample.dvi` to the PostScript file `sample.ps`. The resulting PostScript file may be viewed on screen with any PostScript viewer, such as `ghostview` (`http://www.cs.wisc.edu/~ghost`) or sent to the printer for paper output. But for screen presentation almost surely you will want a PDF file. The `ps2pdf` utility, which comes with `ghostscript`, converts `sample.ps` to the PDF file `sample.pdf`.

Remark 1: The "`-t a4`" flag tells `dvips` to treat its input as having the dimensions of the international A4 paper size rather than the US $8.5'' \times 11''$ paper size. This is because Prosper's design is based on the A4 geometry. Without that flag, parts of the slide may be cut off!

Remark 2: The `dvips` utility reads its configuration information from several places, including a file named `.dvipsrc` in your home directory. (Note that the first character of the file name is a period.) You should create a `.dvipsrc` file, if it doesn't exist, or examine its contents if it does, and make sure that it has the following two entries in it:

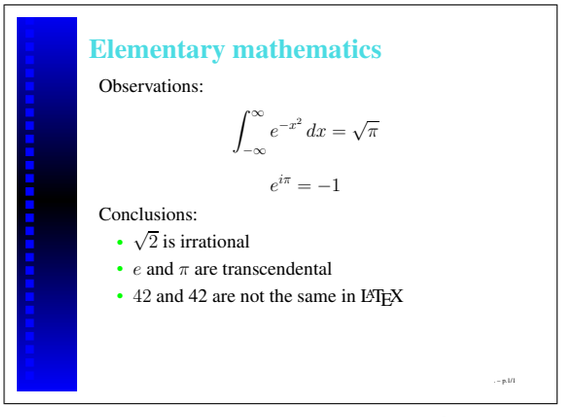
```
p +psfonts.cmz
p +psfonts.amz
```

Without this, characters and symbols used in mathematical formulae will have coarse outlines and will look ugly, especially when magnified and projected on a screen. Some characters may be badly mangled too.

Remark 3: The "`-dPDFSETTINGS=/prepress`" flag to `ps2pdf` optimizes the output resolution for producing sharper graphics. Without it, included graphics and images may look fuzzy. You may omit this flag if you don't have graphics in your slides.

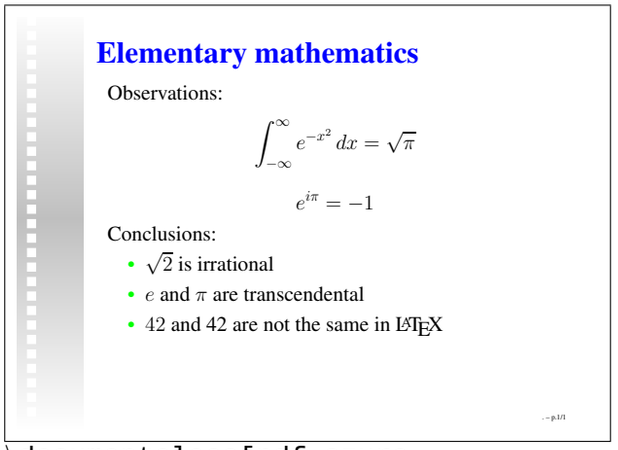
Slide styles

Prosper comes with several pre-defined slide styles. The `azure` option on the `sample.tex`'s `\documentclass` line selects Prosper's `azure` style and results in the slide shown in in the first sample slide. The `colorBG` option on that same line may be negated by replacing it with `nocolorBG`, in which case coloring of the slide's text area is disabled and we obtain the slide shown below.



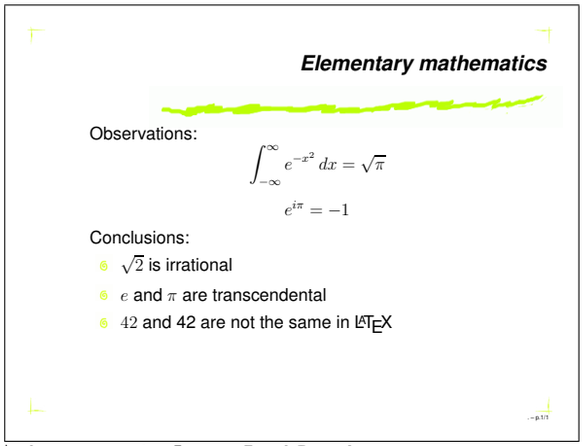
We may disable almost all coloring of the slide by replacing the `slideColor` option with `slideBW`. The result is shown below. This option is intended for printing

slides on black-and-white printers, but I know people who actually like this minimal color style for on-screen presentations.

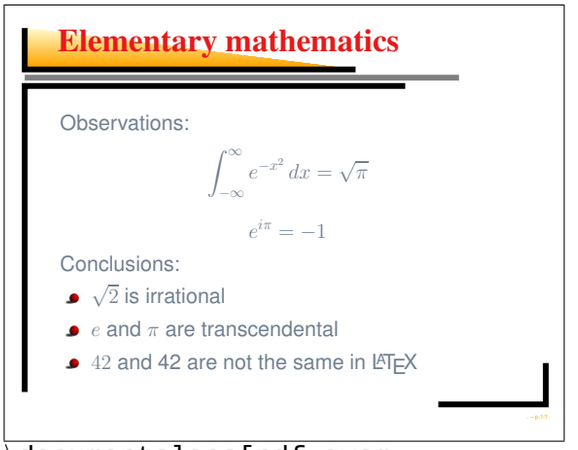


`\documentclass[pdf,azure,`
`slideBW,nocolorBG]{prosper}`

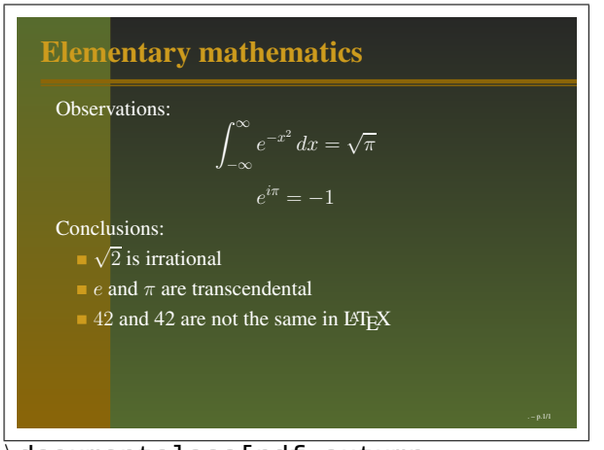
Prosper comes with a large number styles in addition to `azure`. The style definition for `azure` is in the file `PPRazure.sty`. Examine your Prosper installation directory for its list of `PPR*.sty` files. There are a selection of representative samples below. The caption of each figure shows the `\documentclass` line used to produce that slide. Otherwise the input files are identical to that in `sample.tex`. Observe the drastic effect of the change of single keyword on the page layout, fonts, and the "bullet" markers.



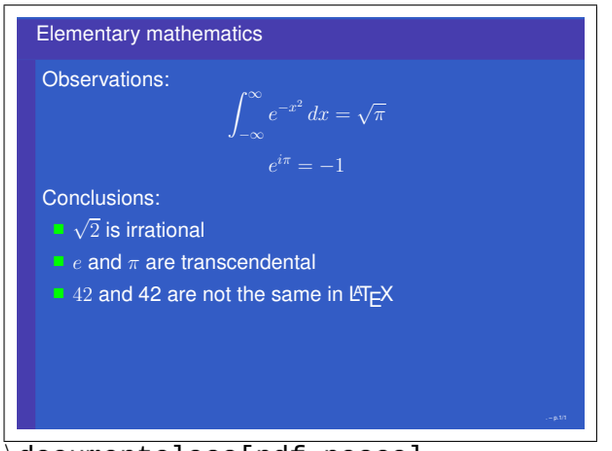
`\documentclass[pdf,rico,`
`slideColor,colorBG]{prosper}`



`\documentclass[pdf,gyom,`
`slideColor,colorBG]{prosper}`



`\documentclass[pdf,autumn,`
`slideColor,colorBG]{prosper}`

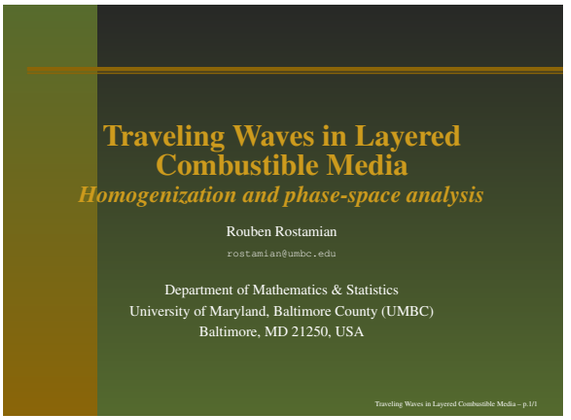


`\documentclass[pdf,pascal,`
`slideColor,colorBG]{prosper}`

For additional samples see the *examples* link in `http://prosper.sourceforge.net/` and the *gallery* link in `http://www.math.umbc.edu/~rouben/prosper/`.

The title page

The layout of the first page, the title page, of a presentation generally is different from the rest of the slides. The figure below illustrates how a title page is created. A typical slideshow sequence will consist of a title page followed by several slides; the latter are not shown in this sample for brevity.




```
\documentclass[pdf,autumn,slideColor,
                    colorBG]{prosper}
\title{Traveling Waves in Layered
               Combustible Media}
\subtitle{Homogenization and
               phase-space analysis}
\author{Rouben Rostamian}
\email{rostamian@umbc.edu}
\institution{
        Department of Mathematics
           \& Statistics \\
        University of Maryland,
           Baltimore County (UMBC) \\
        Baltimore, MD~21250, USA
}
\begin{document}
\maketitle   % generate the title slide
% enter the rest of the slides here
\end{document}
```

This figure above shows a sample "title slide" and the LaTeX input that produced it. Generally the `\maketitle` command will be followed by a sequence of `\begin{slide}{...} ... \end{slide}` blocks as described before.

Overlays

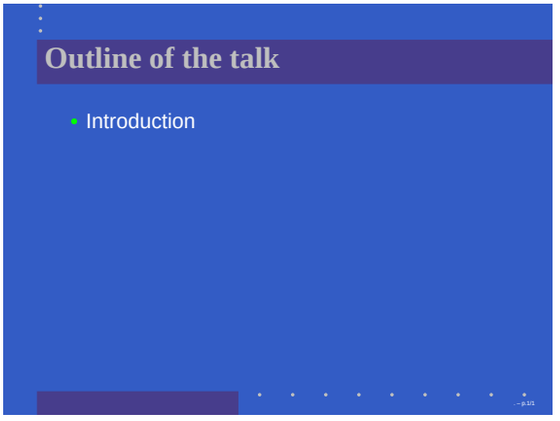
Probably the most interesting effect achieved by Prosper is *overlays*, which gives the appearance of incremental exposure of a given slide. Prosper provides two different mechanisms for achieving this. The first, and the easier one to describe, allows incremental exposure of an itemized list. The second, which is only slightly more complicated, allows incremental exposure of arbitrary parts of a slide, not necessarily entries of an itemized list. I will describe these in the next two subsections.

Overlays for itemized lists

Prosper defines a new LaTeX environment, named `itemstep`, which is closely related to LaTeX's `itemize` environment. `Itemstep`'s purpose is to expose an itemized list one item at a time in a slideshow.

The slide's initial exposure displays only the slide title and the first bulleted item. The action corresponding to moving to the next slide, e.g., pressing of the right-arrow key in the Adobe Reader, has the effect of exposing the second bulleted item. Repeated right-arrow presses expose the remaining bullets, one at a time, until the last bullet is reached. Only then the right-arrow key resumes its normal function of taking us to the next slide.

The sample below illustrates the concept.



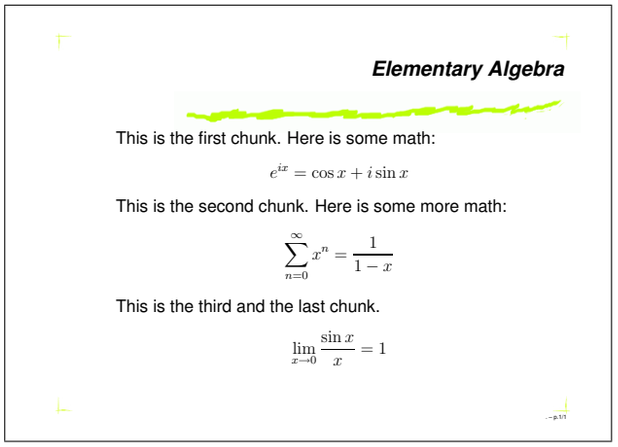

```
\documentclass[pdf,contemporain,
     slideColor,colorBG]{prosper}
\begin{document}
\overlays{5}{
\begin{slide}{Outline of the talk}

\begin{itemstep}
        \item Introduction
        \item Statement of the main
                theorem
        \item Technical lemmata
        \item Proof of the main
                theorem
        \item Conclusions
\end{itemstep}
\end{slide}
```

```
} % closing brace of \overlays
\end{document}
```

The displayed figure is the final state of a slide which will be exposed in five steps. The definition of entire slide is enclosed within the opening and closing braces of the `\overlays{5}{...}` command. The first argument of `overlays` must give the number of incremental steps leading to the fully exposed slide.

Overlays in general

Prosper provides a `fromSlide` command which may be used to expose incrementally a slide containing material of more general nature than just an itemized list. The figure below shows an example. The resulting effect is just like that achieved with `itemstep`, however the gradually exposed chunks are not necessarily items in an itemized list.




```
\documentclass[pdf,rico,slideColor,
                  colorBG]{prosper}
\begin{document}

\overlays{3}{
\begin{slide}{Elementary Algebra}

This is the first chunk.  Here is
  some math:
\[  e^{ix} = \cos x + i \sin x  \]

\fromSlide{2}{
This is the second chunk.  Here is
  some more math:
\[  \sum_{n=0}^\infty x^n =
     \frac{1}{1-x}  \]
} % closing brace of \fromSlide

\fromSlide{3}{
This is the third and the last chunk.
\[  \lim_{x\to0} \frac{\sin x}{x} = 1
  \]
} % closing brace of \fromSlide

\end{slide}
} % closing brace of \overlays
\end{document}
```

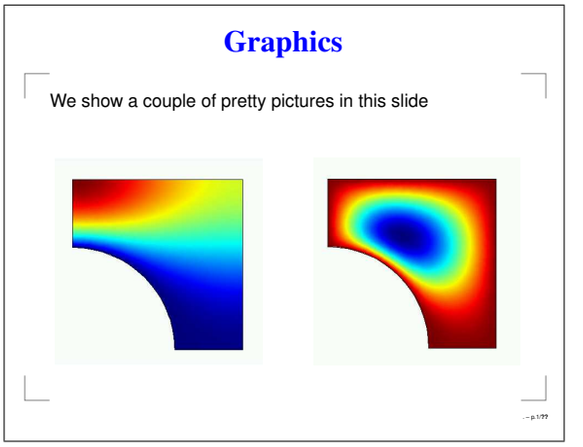
The shown slide will be exposed in three steps. The initial exposure will display the slide title and the paragraph containing the “first chunk”. The next increment will will add the paragraph spanned by the `\fromSlide{2}{...}` command. The definition of entire slide is enclosed within the opening and closing braces of the `\overlays{3}{...}` command.

Including graphics

In this section I will describe how to include graphics in your slides. Actually this topic is not specific to Prosper at all; the same instructions apply to including graphics in any LaTeX document. Although this is not supposed to be a LaTeX tutorial, I have included this section because almost any slideshow presentation involves some graphics.

Encapsulated Postscript is the most common image format for inclusion in LaTeX documents. I assume that you know how to create and save graphics as encapsulated PostScript files.

The figure below shows how to include two side-by-side pictures in a slide.




```
\documentclass[pdf,default,slideColor,
                    colorBG]{prosper}
\begin{document}
\begin{slide}{Graphics}

We show a couple of pretty pictures
     in this slide \vspace{10mm}
```

```
\begin{center}
 \includegraphics[height=50mm]
        {ns-2d-u.eps}\hspace{10mm}
 \includegraphics[height=50mm]
        {ns-2d-v.eps}
\end{center}

\end{slide}
\end{document}
```

Navigating with hyperlinks

During a live slide presentation it is often helpful being able to jump several slides back to remind the audience of a formula or a picture. You don't want to page through 17 slides to go to that particular slide and back through 17 slides to return to where you were.

Hyperlinks in a PDF file enable you to jump from one slide to another with one mouse click.

The sample input file below defines two slides. You should imagine that these are two (not necessarily consecutive) slides out of a large number of slides in a slideshow. The rest of the slides are not shown for brevity.

file: hyperlinks.tex

```
\documentclass[pdf,pascal,slideColor,
                    colorBG]{prosper}
\hypersetup{colorlinks=true,
            linkcolor=red}
\begin{document}
\begin{slide}{Starting slide}

\hyperlink{mytarget}{Click here} to
       go to the target slide.
\end{slide}

\begin{slide}{Target slide}
\hypertarget{mytarget}{}

Yup, we are there!

\end{slide}
\end{document}
```

One slide contains a `\hyperlink` command and the other slide contains a `\hypertarget` command. If you click on the `\hyperlink` with the mouse, the Adobe Reader will jump to displaying the slide containing the corresponding `\hypertarget` command.

A `\hyperlink{...}{...}` command takes two arguments. The first argument (the word "mytarget" in our example,) is an arbitrary but unique label which associates a hyperlink/hypertarget pair. The second argument (the words "Click here" in our example,) is the text which will receive the mouse click. Normally that text will be shown in red to give a visual clue that it is a hyperlink. (The color of the hyperlink may by customized through the `\hypersetup` command.)

A `\hypertarget{...}{...}` command also takes two arguments. The first argument is identical to that of the corresponding hyperlink. The second argument can contain an arbitrary text. In a slide presentation, the second argument has no useful role, therefore it is left empty in the following sample.

It is difficult to illustrate the dynamic action of hyperlinks with words. You should experiment to see it for yourself.

Returning to the calling slide

Now you know how to jump from slide n_1 to slide n_2. But how to you return to slide n_1? Well, you can page through all the intervening slides to get there, but there ought to be a better way.

If you are using the Adobe Reader for your presentation, which very likely you are, then press control-left-arrow (that is, hold down the control key and press the left-arrow key.) That will take you back to the slide that you came from in one single step!

Yet another way to return to the calling slide would be by adding a second hyperlink/hypertarget pair that works in reverse. Thus you put a hyperlink in slide n_1 that takes you to a hypertarget in slide n_2, and you put a hyperlink in slide n_2 that takes you to a hypertarget in slide n_1.

This could be cumbersome in a complex document. Suppose that you have a frequently referred formula on slide n_1. You want to put links in slides n_2 and n_3 and n_4, each of which takes you to slide n_1. But once in slide n_1, which of the slides n_2 or n_3 or n_4 are you supposed to return to? Well, you can have three hyperlinks on slide n_1 that take you to each of the slides n_2 and n_3 and n_4. You just have to remember which one is which. The control-left-arrow method is easier to use, in my opinion.

Odds and ends

In this section I have gathered a miscellany of topics which may be of occasional use but which are not absolutely essential for running Prosper. You may wish to skip this on the first reading.

Controlling type size

To change the default font size within the body of a specific slide, put the command `\ptsize{n}` after

`\begin{slide}{...}`. The effect of the `\ptsize` command will be local to that slide. Legal values for n are 8, 9, 10, 11, 12, 14, 17. The default is 14.

The `\ptsize` command does not affect the font size of the title of the slide. That may be a good thing or a bad thing depending on your perspective.

Setting the slide caption

Prosper puts a caption at the foot of each slide. The default caption looks like this:

title of this slideshow - p.7/12

where "title of this slide show" is the text specified in the `\title` command. The "p.7/12" indicates that this is the seventh of a set of 12 slides. If you don't specify a title, only the "- p.7/12" will be shown.

You may override the text that appears in the caption using the `\slideCaption` command. For instance, if you put:

```
\slideCaption{Joe Smith, Maryland
Institute of Technology}
```

somewhere before `\begin{document}`, then the caption will change to:

Joe Smith, Maryland Institute of Technology - p.7/12

Putting a slide number and total slide count at the foot of each slide is a courtesy to your audience. Unfortunately not everyone is aware of this. All too often I have sat through long presentations where the speaker's slides have given no clue whether the current slide is near the beginning or near the end of the talk. I think that's rude and inconsiderate.

Prosper does not permit you to be rude and inconsiderate.; it will put down that p.7/12 in the slide's caption whether you like it or not (but I hope that you do).

Having said how strongly I feel in favor of numbering slides, I am going to show you how to disable the numbering.

A colleague of mine has a strong *dislike* of numbered slides. He pled for my help to achieve his sinister goal of removing the numbers. So I gave in and showed him how. I am including the solution here in case you will have a use for it in some rare situations.

To disable page numbers on slide captions, put the following immediately after the `\documentclass` line and before the `\begin{document}` line:

```
\makeatletter
\newpagestyle{MyPageStyle}{\hfill}
    {\hfill\@colorFoot\tiny\@cartouche}
\makeatother
\slidepagestyle{MyPageStyle}
```

Putting a logo on your slides

The command `\Logo{someobject}` (spelled with capital L) puts the TEX object `someobject`, which is typically an image, such as a company logo, on each slide other than the titlepage slide. Put the command somewhere before the `\begin{document}`. Typically you will do:

```
\Logo{\includegraphics{logo.eps}}
```

where `logo.eps` is the name of an Encapsulated PostScript file that contains the logo image.

The logo's default position depends on Prosper's style option. It's possible to override the default by specifying the logo's x and y coordinates, as in `\Logo(x,y){someobject}`. The precise definition of the origin of coordinates and the units of measurement are arcane; it is easier to experiment with small values of x and y and gradually change them until you achieve the desired location.

A Makefile for Prosper

The repeated typing of the compilation commands during a write/test/debug cycle can be tiresome and error-prone. Those who have programmed under the UNIX environment may be familiar with the `make` utility and `Makefile`s. These are intended to organize and automate repetitive tasks when compiling programs. A `Makefile` is a text file that contains a sequence of instructions to be performed. UNIX's `make` utility reads and executes the instructions specified in a Makefile.

The description of the `make` utility is outside of the scope of this tutorial so I will not get into that. However, you don't need to be a `make` expert in order to use it. I suggest that you download the Makefile pointed at by the URL:

```
http://www.math.umbc.edu/~rouben/prosper/
    large_sample/Makefile
```

and modify it for your own use. Beware that with some web browsers you will have to right-click on a link to download it. Also beware that the layout of a Makefile follows certain very precise rules. Cutting and pasting by the mouse almost certainly will destroy that precise layout.

The provided Makefile assumes that the name of the LATEX file that contains your slides is `slides.tex`. To change that, just edit the makefile and replace the single occurrence of `slides.tex` by the name of your slides file.

To compile your slides file, just type `make` on the command line. The `make` utility will read both the Makefile and your LaTeX file, will apply the compiling commands to the LaTeX file and will produce the corresponding PDF file. That's all there is to it!

As a bonus for using the Makefile, you may type `make clean` on the command line to remove all the files generated by running `make` thus restoring your directory to the state it was in before the `make` command was run.

Miscellaneous Editorial Notes

More recent alternatives to latex are available, including `pdflatex`, and `vlatex`. These packages are useful for inserting non-ps figures, such as jpg, pdf and they create good quality `pdf` documents. We've tested `pdflatex` with prosper and it produces attractive slides, although the overlay features of prosper don't work. We have not managed to produce slides with `vlatex`.

FROM OUR CHAIRS (Cont.). . .

Statistical Computing

CONTINUED FROM PAGE 1

Sabine Van Huffel, Katholieke Universiteit Leuven, "Bridging the Gap between Statistics, Computational Mathematics and Engineering"; Andrew Barron, Yale University, "Function Fitting with Many Variables; Neural Networks and Beyond"; and Chun-houh Chen, Academia Sinica Taipei, "Dimension Free Data Visualization and Information Mining".

I would also ask that you keep an eye on the journals in our field. In addition to the general statistics journals, I recommend perusal of two journals, in particular, which focus on statistical computing: our own *Journal of Computational and Graphical Statistics* and IASC's *Computational Statistics and Data Analysis*.

Finally, we plan to reinvigorate our Statistical Computing webpage, `http://www.statcomputing.org/`, to make this site a central resource for our Section. We will do our best, but we need your help. Mark Hansen has graciously agreed to remain as our Webmaster, but this initiative cannot be a success without the active participation of YOU, our members. This is just our first shot across the bow on this issue; please contact Mark `cocteau@stat.ucla.edu` or me `cep@jhu.edu` with your ideas for the web site, or to volunteer as a "local editor" in an area such as software releases, conference announcements, member news, section news, paper announcements, etc.

For any success I have in my tenure as Chair I thank all the Chairs that have come before me (including, notably, my immediate predecessor Lee Wilkinson) and those who shall come after me (including, notably, my immediate successor, our current Chair-Elect, Tim Hesterberg). (Any failures are, of course, mine and mine alone!)

I leave you with a quote from Leopold Kronecker to Hermann von Helmholtz which, I think, succinctly captures the happy juxtaposition of application and theory that characterizes our ASA Section on Statistical Computing: "The wealth of your practical experience with sane and interesting problems will give to mathematics a new direction and a new impetus."

Statistical Graphics

CONTINUED FROM PAGE 1

members of our Section. The newly-elected officers, whose terms will begin January 1, 2005, are Chair-Elect, Paul Murrell; Program Chair-Elect, Jürgen Symanzik; Publications Officer, Linda W. Pickle; Secretary/Treasurer, John Castelloe; and the Council of Sections Representatives, Dan Carr and Ed Wegman. Also, congratulations are in order for the members of our Section who were chosen this year as Fellows of ASA: I. Elaine Allen, Michael P. Cohen, William T. Dunsmuir, Fred L. Hulting, Xiao-Li Meng, R. Daniel Meyer, Mario Peruggia (our current Chair-Elect), Robert N. Rodriguez, Dongchu Sun, and Terry M. Therneau.

A perennial concern for our Section, and indeed for any learned or professional association is membership numbers. Membership in our Section has not grown proportionate to the growth of the importance of graphical methods in understanding increasingly complex sets of data. Benefits from membership in an organization rarely derive from the relatively passive step of paying dues; benefits flow in proportion to the level of particiption. That early step of "enrolling", however, is an important one. Those of us in education – or who know students, or who are students – can help by bringing new members into our association. Full-time students can join ASA for $10 for the first year! (The dues then go to only $25/year.) We should encourage all of our students to join ASA — for full-time students, it's too good a bargin to pass up. Part-time students should con-

sider it a very worthwhile investment (and an even a better investment if they can get their employers to pay!).

Coincident with joining ASA should be joining the Section. Incidentally, I will propose to the Exceutive Commitee that the Section institute a lower rate for student dues, maybe $1/year. Our Section is one of the few ASA sections that does not have special rates for students.

Our student paper competition (joint with the Stat Comp Section) are great, but we need to get more students and students from different institutions involved in these competitions.

Speaking of Section awards, this is also a topic for consideration: we should have more awards from the Section. (Writing this newletter article has prompted me to surf through a slew of ASA web pages. It's been interesting. In looking at the ASA website, it appears that the "Student Paper Competition" is sponsored only by our sister section. That may be true. If so, it's OK; let's just proceed to develop some more awards in Statistical Graphics.) Some possible awards for our Section are: best invited paper, best contributed paper, best poster, best newsletter article, best use of graphical displays in a refereed statistics journal, best use of graphical displays in a refereed non-statistics journal, best use of graphical displays in the news media (separate categories for print and video), worst graphical display in the news media, best graphics software, student travel award, distinguished achievement.

Maybe we could get contributions from our commercial software friends to put some tangible rewards (money, software, or books) behind the awards.

Looking forward to seeing you all at the JSM mixer!

Editorial

Time has flown by again! Now that its time to eat watermelon, smack mosquitoes, and recover from the year's teaching we're finally pulling together the pieces for the next newsletter. We have two very substantial contributed articles in this issue.

The first article describes a utility package for typesetting with latex called Prosper. Prosper helps latex users to create attractive presentations with special effects, similar to those created using Microsoft's Powerpoint. Prosper gives latex users more options to create visually rich slides. The typeset presentation can be given using Adobe Reader on any operating system, linux, Mac, Windows. Ranjan and Di have both been using Prosper for several years now. This article has been written by a former colleague of Ranjan's at the University of Maryland.

The second article describes a new plotting package for R, called iPlots, that has direct manipulation capabilities. This arises from the creative energy of Martin Theus' and Simon Urbanek's graphics research at the University of Augsburg. For ease of use the command line arguments are carefully produced to match the regular R plotting functions. Yet the scatterplots and histograms are interactive and linked, so that a user can highlight a point in one plot which will highlight the relevant parts in other open plots. Its a very exciting development!

Now about the newsletter operation. We have had some trouble announcing recent newsletters. Since shifting to a purely electronic format its been harder to get announcements about a new issue out to you all. We first experimented with emailing announcements, using the mailing list provided by the American Statistical Association. These mailing lists are supposed to be kept up to date. What we found is that they were not up to date and we received numerous email responses requesting that we correct the addresses. We can't do that. We also received numerous bounced emails, and automated vacation notices. And the first time we sent emails we inadvertently forgot to use blind carbon copy, which resulted in an enormous number of flaming responses yelling at us for exposing their email address. This was a tad disturbing for our volunteer spirits. So for the most recent newsletter, November 2003, we investigated having the ASA staff send the announcement. It seems the announcement didn't get sent out to most of us! A very few members did receive the news. This didn't include either Ranjan or I! We will work on this approach for this issue. But in the future issues we are contemplating the possibility of mailing a postcard to each member. There'll be some expense to produce and mail a postcard, but not as much as printing and mailing a full newsletter. And the postcards could be designed to have some attractive graphic on the main side, perhaps becoming collector's items! Let us know what you think. We do apologize for the slow publication of our newsletter.

Finally, we are very thankful to our contributors for the articles in this issue. This is a great outlet for new experimental research, short descriptions of useful software, and general information about computing. We are open to new ideas and writings for the next issue, due December, 2004.

Dianne Cook and Ranjan Maitra

iPlots: Interactive Graphics for R

Martin Theus, Simon Urbanek
University of Augsburg,
Department of Computeroriented Statistics and Data Analysis
Universitätsstr. 14, 86135 Augsburg, Germany
{martin.theus, simon.urbanek}@math.uni-augsburg.de

Introduction

After more than 10 years of ongoing software development in interactive statistical graphics software at the Department of Computeroriented Statistics and Data Analysis at Augsburg University, a whole suite of software packages emerged. Some of the most recent and still supported packages include KLIMT [4] by Simon Urbanek, MANET [1] by Heike Hofmann and Mondrian [3] by Martin Theus. What all these packages have in common are the interactive features, which are the building blocks for a graphical data analysis. Although interactive statistical graphics plays a major role in a data analysis process, it usually needs to be complemented with classical statistical methods or other data mining tools. None of the interactive tools mentioned above ever intended to incorporate these methods, although it would be desirable. There are some exceptions to this rule, e.g. MANET offers Biplots and Mondrian can setup loglinear models. Nonetheless, the big drawback when implementing these algorithms is of course the amount of testing needed in order to "get them right". Looking at KLIMT, we see a different option. KLIMT does not implement algorithms to build CARTs, but relies on R. It either imports the output of R via a ASCII file, or directly talks to R. Since R's communication skills for various systems are still pretty limited, this option is not very satisfactory and it is platform dependent.

When bringing R's method richness to interactive software is too cumbersome, why not bring interactive graphics to R? The basic graphics system in R dates back to times when many of the readers were still in kindergarten, and implementation was putting ink on paper. This was perfectly justified at the time, but it does not allow for interactive features, which have developed in recent times. Thus we were forced to develop the interactive features outside of R. For a maximum platform independence JAVA was the choice to work with. The iPlots package is designed as a library for R which adds interactive graphics to R. iPlots uses the R/SJava Interface to communicate with R.

The first versions of iPlots implements interactive barcharts, histograms and scatterplots. All iPlots can be manipulated from within R and can be enhanced with static elements like lines and points.

Design Choices

When integrating interactive graphics into R it is obviously desirable not to introduce a new way of handling objects and devices, but to stick as close to R syntax and semantics as possible. iPlots come as an R library and it is invoked simply by the command `library(iplots)`. Plots in iPlots use the same names as their static counterparts, except for the leading '`i`'. E.g. to bring up an interactive histogram one simply types `ihist(data)`, where `data` is any arbitrary R vector. The most important plot options which specify colors, labels and scale information are supported transparently by iPlots.

The concept of linked highlighting, i.e. cases in a plot can be selected and light up in all other plots, as well as color brushing, i.e. all points belonging to a specific group are painted in the same color, is an attribute of the rows of a data set, linking cases across variables. The corresponding R construct of a dataframe would be the solution of choice. Unfortunately it is not possible to associate a variable from a dataframe with the dataframe after it is passed to a function. This would force the user to explicitly add the dataframe name (as known from the `data=...` option in model functions) when calling an iplot. To be less restrictive and more compatible, we chose a different path. We introduced the iSets, which are data structures, which hold all data belonging to a relation. Within an iSet alls cases with corresponding row-ids are linked. The handling of iSets is adopted from the way devices are handled in R. When the first iPlot is generated, a corresponding iSet is created. As long as the user only uses data from the same relation – which is assumed when the data vectors are all of the same length – all data is collected in the initially created iSet and no new iSet is created. Whenever the user plots an iPlot with data from a different relation, i.e. the length of the vector is different from the number of rows in the current iSet, a new iSet is created. To avoid mistakes, a warning dialog prompts the user to confirm the creation of the new iSet. One can switch back and forth between iSets with next and previous functions as well as by directly specifying the iSet's id. The same scheme is used to manage iPlots within iSets.

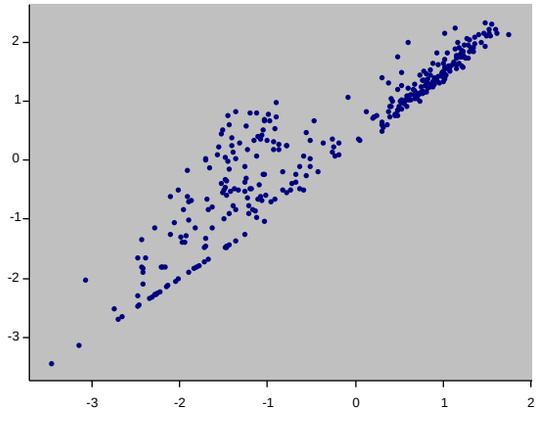
The figure below shows an example iPlot session. In this session data from three different relations is held in three iSets.iPlots are attached to each of the iSets.

Furthermore, iObj, like lines or points can be attached to an iPlot. The next section gives a brief example of how iPlots can be used in a data analysis session in R.

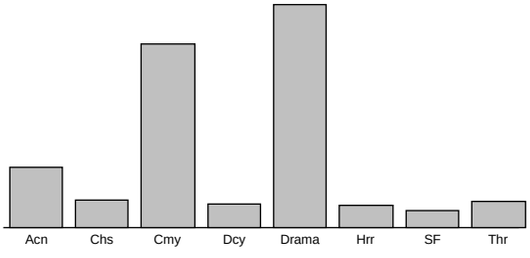
Sample Session: Movie Data

We want to investigate the movies-dataset from Simonoff's Summer 2000 Chance article [2]. The data is available at. `http://pages.stern.nyu.edu/~jsimonof/movies/`. This data contains both the 1998 and 1999 movies. The list of variables includes log(total US gross), log(first weekend gross), log(number of opening screens), Oscar Nominations, Genre (action, childrens, comedy, documentary, drama, horror, science fiction, thriller), and MPAA Rating. Simonoff's final result fits a different to movies with 10 or less opening screens and more than 10 opening screens. The most important variable is the log(revenue of the first weekend). He also includes the log(number of opening screens). For the part with more than 10 opening screens the genre of the movie and the number of oscar nominees are included. To investigate this model the following session could be used:

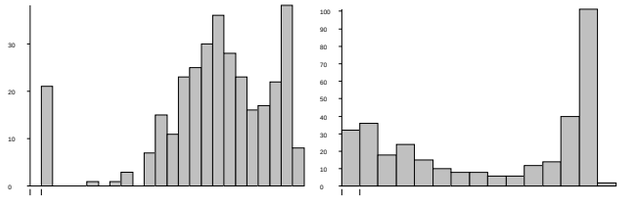
```
> movies <- read.table("F.txt", sep="\t",
+ header=T, quote='"')
> library(iplots)
> attach(movies)
> iplot(Log.1st.weekend, Log.domestic)
ID:1 Name: "Scatterplot
  (Log.1st.weekend : ...
```



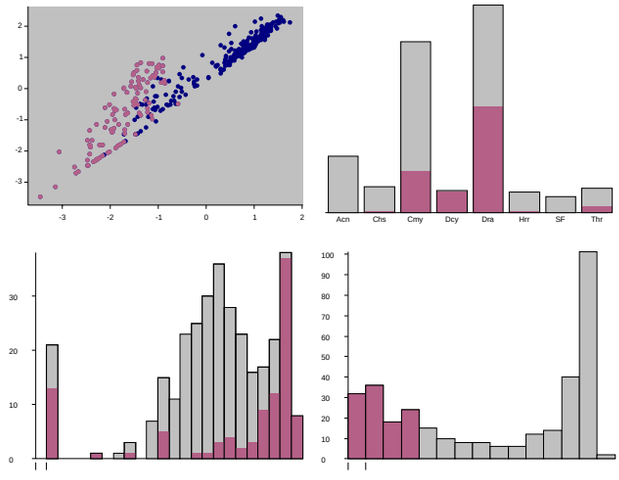

```
> ibar(Genre)
ID:2 Name: "Barchart (Genre)"
```




```
# How much of the total domestic gross was
# made after the 1st weekend?
> ihist((Total.domestic.gross -
+ First.weekend)
+ / Total.domestic.gross)
ID:3 Name: "Histogram ((
    Total.domestic.gross ...
> ihist(Log.opening.screens,
+ breaks=seq(0,4,.25))
ID:4 Name: "Histogram (
    Log.opening.screens.2)"
```



These four iPlots set up so far allow for a closer investigation of Simonoff's proposed model. The next Figure shows an example of the linked highlighting. All movies with less than 10 opening screens have been selected in the bottom right histogram.

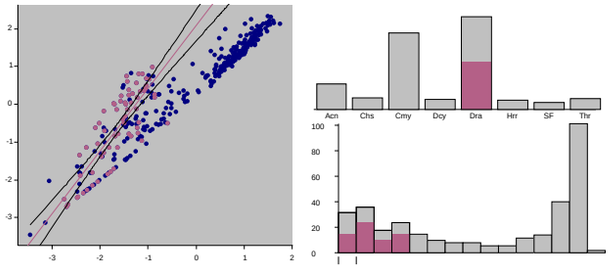


What do we see? The selected group, movies with less than 10 opening screens, made most of their total domestic gross after the first weekend (as seen by the large number of cases at higher values in the bottom left histogram). These movies also accounts for more than half of the movies which made their total gross only at the opening weekend. Furthermore these movies all have a rather small total domestic gross (top left). There is a strong interaction between the selected group and the genre of the movie, mostly drama, comedy and documentary (top right).

As with R graphics, its possible to overlay different statistical quantities on iP. A regression line is added to the plot along with 90% confidence bounds. The regression line is calculated for the currently selected points only, and changes whenever the selection changes, until the user exits the loop with the Break menu command in

iPlots. This example builds on the previous commands.

```
# Set the current plot to be
# the first histogram
iplot.set(1)r.line <- iabline(0,
  col="marked", visible=FALSE)
cb1<-ilines(0, col="black", visible=FALSE)
cb2<-ilines(0, col="black", visible=FALSE)
x.sorted<-sort(Log.1st.weekend)
while (!is.null(ievent.wait())) {
  if (iset.sel.changed()) {
    s<-iset.selected()
    if (length(s)>0) {
      y<-Log.domestic[s]
      x<-Log.1st.weekend[s]
      m<-lm(y ~ x)
      iobj.opt(r.line, reg=m,
        visible=TRUE)
      ci<-predict(m,
        data.frame(x=x.sorted),
        level=0.9, interval="confidence")
      iobj.opt(cb1, x.sorted, ci[,2],
        visible=TRUE)
      iobj.opt(cb2, x.sorted, ci[,3],
        visible=TRUE)
    }
    else {
      iobj.opt(list(r.line, cb1, cb2),
        visible=FALSE)
    }
  }
}
iobj.rm(list(r.line, cb1, cb2))
```



In this Figure only movies with less than 10 opening screens and genre “drama” are selected, showing a clear deviation from the rest of the points in the scatterplot.

Function Reference

Plots

This section gives a short listing of all functions related to iPlots implemented so far. For brevity reasons the parameter list may not be complete and doesn’t include default values.

- Histogram:
  `ihist(x, breaks, col, bwidth, anchor, right, main, xlab, ylab, xlim, ylim)`
- Scatter plot:
  `iplot(x, y, col, main, xlab, ylab, xlim, ylim, log)`
- Bar chart:
  `ibar(data, col)`
  Note: Unlike the `barplot` R function, this iPlot requires the entire factor as the `data` parameter, because the highlighting uses linking at the case level.

Manipulation of plots and data sets

iPlots support two methods for accessing individual plots. The first interface was defined to resemble the functions used in conjunction with graphics devices. The functions `iplot.set`, `iplot.cur`, `iplot.list`, `iplot.next`, `iplot.prev` and `iplot.off` provide means to select, retrieve or close the current iPlot. This corresponds to the `dev.`*xx* commands for R graphics devices. All plot-specific commands apply to the current iPlot, unless the `plot` parameter is set.

Some other iPlot and iSet manipulating functions include:

- Sets options or parameters of the plot. The possible options can include those specified in the initial plot invocation.
- `iset.new(name = NULL, data = NULL)`
  Creates a new iSet and makes it current. The iSet is assigned a name of the form “data.#”, where # is an unique ID.
- `iset.set(which = iset.next())`
  Change the current iSet.
- `iset.cur()`
  Returns the current iSet.
- `iset.list()`
  Lists all iSets.
- `iset.next(which = iset.cur())`
  Returns the next iSet in the list. The list is circular.
- `iset.prev(which = iset.cur())`
  Returns the previous iSet in the list.

The second means of accessing individual plots without changing the current plot is to use the plot object returned by plot creating commands. This object can be passed via `plot` parameter to most iPlot functions, for example:

```
p<-iplot(x,y); ibar(z);
ilines(lm(y~x), plot=p)
```

Selection and color brushing

- `iset.select(what, mode="replace")`
  Selects cases specified by `what`, which can be either a logical vector used as a mask or a numeric vector used as list of IDs. The option `mode` specifies which logical operation should be applied relative to the existing selection.
- `iset.selected()`
  Returns a vector of selected cases as IDs.
- `iset.col(color=NULL, what="all")`
  Brushes specified cases with `color`. `what` can be either a list of IDs, a logical vector or the string `"all"`.

iObjects *Toolkit*

Every iPlot can have an arbitrary number of additional graphical objects attached, such as lines, rectangles, polygons or labels. Those objects, called iObjects, appear in the plot graphics and are programmable as of shape, position and color.

- `ilines(x, y=NULL, col=NULL, fill=NULL, ...)`
  Takes given coordinates and joins the corresponding points with line segments. The resulting iObject is a polygon or a polyline.
- `iabline(a, b=0, ...)`
  Creates a line with the specified intercept and slope.
- `itext(x, y=NULL, labels = seq(along = x), ...)`
  Adds given text labels as an iObject to the plot at specified coordinates.
- `ipoints(x, y=NULL, col=NULL, ...)`
  Creates a new iObject which draws a sequence of points at the specified coordinates.
- `iobj.list(plot = iplot.cur())`
  Lists all iObjects of the plot.
- `iobj.cur(plot = iplot.cur())`
  Returns the ID of the current iObject of the plot.
- `iobj.rm(obj = iobj.cur())`
  Removes the iObject `obj` from the associated plot.
- `iobj.set(obj)`
  Make `obj` the current iObject.
- `iobj.opt(...,obj=iobj.cur())`
  Sets options and properties of the given iObject. The parameters are object-dependent.

Event handling

iPlots offer some basic functions to make interaction between iPlots and R more flexible. These functions can be used to build animations and small interaction loops in R.

- `ievent.wait(...)`
  This function waits until an iPlots event occurs. The result is `NULL` for the *break* event.
- `iset.sel.changed(iset=iset.cur(),...)`
  Returns `TRUE` if the selection of the specified iSet changed since the last call of this function.

Summary & Outlook

iPlots are a R library which brings interactive graphics into the R environment. iPlots offer a variety of customization options that can be modified and enhanced directly through the R interface. In the first release iPlots offers onlya few plots, histogram, barchart and scatterplot. The current development is focused on adding more plots like parallel coordinate plots, mosaic plots and maps.

Further effort is made to integrate iPlots and R in an unified R interface based on Java. This combination will enhance the user experience by providing flexibility and common tools, such as window management, menus and help for all four parts of the system: console, editor, R graphics and iPlots. Finally such a tight integration will allow for much improved event handling with callback support, without the need for `ievent.wait` loops.

More information on iPlots can be found at `http://stats.math.uni-augsburg.de/iPlots/`.

References

[1] Heike Hofmann. Simpson on Board the Titanic? Interactive methods for dealing with multivariate categorical data. *Statistical Computing & Statistical Graphics Newsletter*, 9(2):16–19, 1998.

[2] Jeff Simonoff and Ilana Sparrow. Predicting movie grosses: Winners and losers, blockbusters and sleepers. *CHANCE Magazine*, 13(3), Summer 2002.

[3] M. Theus. *Mondrian - Interactive Statistical Graphics in JAVA*. `http://www.theusRus.de/Mondrian`, 2001.

[4] Simon Urbanek and Antony R. Unwin. Making Trees Interactive - KLIMT. In *Proc. of the 33th Symposium of the Interface of Computing Science and Statistics*, 2001.

NEWS CLIPPINGS AND SECTION NOTICES

This years JSM program highlights for the Statistical Graphics section are:

- Mon 8:35am Evolutionary Graphics for Streaming Data, organized by Ed Wegman: Data Cleansing and Preparation at the Gates: A Data Streaming Perspective, Detecting Computer Masqueraders Using Online Monitoring.
- Mon 10:30am Contributed papers in graphics: Parallel Coordinate Plots of Gene Expression Data, Large Datasets in Drug Discovery: How to Collect, Summarize, and Visualize Data from Image-based Cellular Assays in Meaningful Way, Comparison of Data-driven vs. Theory-driven Method: Decision-tree Models as Data Discovery Tools in Social Science Research, Social Network Blockmodels and Allegiance, Exploratory Analysis of Graphical Summaries of Scenarios, Structured Multicategory Support Vector Machine with ANOVA Decomposition.
- Wed 10:30am History of Data Visualization, organized by Michael Friendly: The Milestones Project: A Case Study in Statistical Historiography, A Graphical Legacy of Charles Joseph Minard, AM Georg von Mayr and Emil Eugen Roesle–Two German Pioneers of Statistical Graphics.

Upcoming conferences: COMPSTAT '04, in Prague, August 23-27, 2004, has an outstanding program this year, put together by Jaromir Antoch and the program committee. There are speakers from the across the globe, USA, Canada, Australia, New Zealand, Germany, Austria, France, Romania, Finland, ... The topics of sessions include data mining, visualization, functional data analysis, bioinformatics, statistics teaching, and web mining. The details can be found at `http://compstat2004.cuni.cz/`.

SECTION OFFICERS

Statistical Graphics Section - 2004

Gentle, James E. , Chair
George Mason Univ
Phone: (703) 993-1994
jgentle@gmu.edu

James, David A. , Past-Chair
Bell Labs, Lucent Technologies
Phone: (908) 582-3082
dj@lucent.com

Peruggia, Mario , Chair-Elect
The Ohio State University
Phone: (614) 292-0963
peruggia@stat.ohio-state.edu

Symanzik, Juergen , COS Rep
Utah State University
Phone: (435) 797-0696
symanzik@math.usu.edu

Robbins, Naomi B. , COS Rep
Phone: (973) 694-2686
naomi@nbr-graphs.com

Solka, Jeffrey L. , Program Chair
NSWC
Phone: (540) 653-1982
solkajl@nswc.navy.mil

Rope, Daniel J. , Program Chair-Elect
SPSS
Phone: (703) 390-9346
drope@spss.com

Murrell, Paul, Publications Officer
and Electronic Communication Liaison
(649) 373-7599 x85392
Univ of Auckland
paul@stat.auckland.ac.nz

Smith, William B. ,
American Statistical Association
Phone: (703) 684-1221
williambsmith@amstat.org

Statistical Computing Section - 2004

Priebe, Carey, Chair
(410) 516-7200
The Johns Hopkins University
cep@jhu.edu

Hesterberg, Tim, Chair-Elect
(206) 802-2319
Insightful
timh@insightful.com

Wilkinson, Leland, Past-Chair
(312) 651-3270
SPSS
leland@spss.com

Madigan, David, Program Chair
(732) 445-3748
Rutgers University
dmadigan@rci.rutgers.edu

Holmes, Susan P., Program Chair–Elect
(650) 725-1925
Stanford University
susan@stat.stanford.edu

Ranjan Maitra, Newsletter Editor (02-04)
(515) 294-7757
Iowa State University
maitra@iastate.edu

Wills, Graham J. Secretary/Treasurer (04-04)
(312) 651-3671
SPSS
gwills@spss.com

Hesterberg, Tim, Publications Officer
(206) 283-8802
Insightful
timh@insightful.com

Pinheiro, Jose C., Awards Officer
(862) 778-8879
Novartis
jose.pinheiro@pharma.novartis.com

Koenker, Roger, Rep.(03-05) Council of Sections
(216) 333-4558
Univ of Illinois
rkoenker@uiuc.edu

Gentleman, Robert, Rep.(04-04) Council of Sections
(617) 632-5250
Harvard Biostatistics
rgentlem@hsph.harvard.edu

Hansen, Mark, Electronic Communications Officer
(310) 826-8375
UCLA
cocteau@stat.ucla.edu

Smith, William B. ,
American Statistical Association
Phone: (703) 684-1221
williambsmith@amstat.org

INSIDE

Statistical COMPUTING & GRAPHICS

The *Statistical Computing & Statistical Graphics Newsletter* is a publication of the Statistical Computing and Statistical Graphics Sections of the ASA. All communications regarding this publication should be addressed to:

Ranjan Maitra
Editor, Statistical Computing Section
Department of Statistics
Iowa State University
Ames, IA 50011-1210
(515) • FAX: (515) 294 4040
maitra@iastate.edu
http://www.stat.iastate.edu/∼maitra

Dianne Cook
Editor, Statistical Graphics Section
Department of Statistics
Iowa State University
Ames, IA 50011-1210
(515) 294 8865 • FAX: (515) 294 4040
dicook@iastate.edu
www.public.iastate.edu/∼dicook

All communications regarding ASA membership and the Statistical Computing or Statistical Graphics Sections, including change of address, should be sent to:

American Statistical Association
1429 Duke Street
Alexandria, VA 22314-3402 USA
(703) 684-1221 • FAX (703) 684-2036
asainfo@amstat.org