Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *http://www.jstatsoft.org/*

plotROC: An R Package to Improve the State of ROC Curve Plots in the Medical Literature

Michael C. Sachs
Biometric Research Branch, Division of Cancer Treatment and Diagnosis, National Cancer Institute

Abstract

Plots of the receiver operating characteristic (ROC) curve are ubiquitous in medical research. Designed to simultaneously display the operating characteristics at every possible value of a continuous diagnostic test, ROC curves are used in oncology to evaluate screening, diagnostic, prognostic and predictive biomarkers. We evaluate current trends in the design of ROC curve plots with a literature review. Our review suggests that ROC curve plots are often ineffective as statistical charts and that poor design obscures the relevant information the chart is intended to display. We describe our new R package that was created to address the shortcomings of existing tools. The package has functions to create informative ROC curve plots, with sensible defaults, for use in print or as an interactive web-based plot. A web application was developed to reach a broader audience of scientists who do not use R.

Keywords: ROC curves; graphics; interactive; plots.

1. Introduction

1.1. About ROC Curves

The Receiver Operating Characteristic (ROC) curve is used to assess the accuracy of a continuous measurement for predicting a binary outcome. In medicine, ROC curves have a long history of use for evaluating diagnostic tests in radiology and general diagnostics. ROC curves have also been used for a long time in signal detection theory.

The accuracy of a diagnostic test can be evaluated by considering the two possible types of errors: false positives, and false negatives. For a continuous measurement that we denote as M, convention dictates that a test positive is defined as M exceeding some fixed threshold c: $M > c$. In reference to the binary outcome that we denote as D, a good outcome of the

test is when the test is positive among an individual who truly has a disease: $D = 1$. A bad outcome is when the test is positive among an individual who does not have the disease $D = 0$.

Formally, for a fixed cutoff c, the true positive fractions is the probability of a test positive among the diseased population:

$$TPF(c) = P\{M > c|D = 1\}$$

and the false positive fraction is the probability of a test positive among the healthy population:

$$FPF(c) = P\{M > c|D = 0\}$$

Since the cutoff c is not usually fixed in advance, we can plot the TPF against the FPF for all possible values of c. This is exactly what the ROC curve is, $FPF(c)$ on the x axis and $TPF(c)$ along the y axis. A useless test that is not informative at all in regards to the disease status has $TPF(c) = FPF(c)$ for all c. The ROC plot of a useless test is thus the diagonal line. A perfect test that is completely informative about disease status has $TPF(c) = 1$ and $FPF(c) = 0$ for all c.

Given a sample of test and disease status pairs, $(M_1, D_1), \ldots, (M_n, D_n)$, we can estimate the ROC curve by computing proportions in the diseased and healthy subgroups separately. Specifically, given a fixed cutoff c, an estimate of the $TPF(c)$ is

$$\widehat{TPF}(c) = \frac{\sum_{i=1}^{n} 1\{M_i > c\} \cdot 1\{D_i = 1\}}{\sum_{i=1}^{n} 1\{D_i = 1\}},$$

where $1\{\cdot\}$ is the indicator function. An estimate for $FPF(c)$ is given by a similar expression with $D_i = 1$ replaced with $D_i = 0$. Calculating these proportions for c equal to each unique value of the observed M_i yields what is known as the empirical ROC curve estimate. The empirical estimate is a step function. Other methods exist to estimate the ROC curve, such as the binormal parametric estimate which can be used to get a smooth curve. There are also extensions that allow for estimation with time-to-event outcomes subject to censoring. For a more thorough reference on the methods and theory surrounding ROC curves, we refer interested readers to Pepe (2003).

A common way to summarize the value of a test for classifying disease status is to calculate the area under the ROC curve (AUC). The greater the AUC, the more informative the test. The AUC summarizes the complexities of the ROC curve into a single number and therefore is widely used to facilitate comparisons between tests and populations. It has been criticized for the same reason because it does not fully characterize the trade-offs between false- and true-positives.

1.2. Design Considerations

The main purpose of visually displaying the ROC curve is to show the trade-off between the FPF and TPF as the cutoff c varies. This can be useful for aiding viewers in choosing an optimal cutoff for decision making, for comparing a small number of candidate tests, and for

generally illustrating the performance of the test as a classifier. In practice, once the FPF and TPF are computed for each unique observed cutoff value, they can be plotted as a simple line chart or scatter plot using standard statistical plotting tools. This often leads to the unfortunate design choice of obscuring the critical and useful third dimension, the range of cutoff values c.

Another key design element is the use of a diagonal guideline for comparison. The diagonal guideline serves as a baseline for comparison, allowing observers to roughly estimate the area between the diagonal and the estimated ROC curve, which serves as a proxy for estimating the value of the test for classification above a coin-flip. Likewise, gridlines inside the plotting region and carefully selected axes allow for accurate assessment of the TPF and FPF at particular points along the ROC curve. Many medical studies use ROC curves to compare a multitude of candidate tests to each other and to the null diagonal. In those cases, curves need to be distinguished by using a legend combined with different colors or line types, or direct labels inside the plotting region.

In the medical literature, FPF and TPF are usually referred to in terms of the jargon terms Sensitivity and Specificity. Sensitivity is equivalent to the true positive fraction. Specificity is 1 - FPF, the true negative fraction. Sometimes, the FPF and TPF are incorrectly referred to as rates, using the abbreviations FPR and TPR. These are probabilities and their estimates are proportions, therefore we prefer the use of the term fraction as opposed to rate.

1.3. Existing Plotting Software

The ROC curve plot is, at the most basic level, a line graph. Therefore, once the appropriate statistics are estimated, existing plotting functions can be used to create an ROC curve plot. The addition of axis labels, titles, legends, and so on, indicate a chart as such. In our literature review, we observed plots with the distinctive characteristics of the base plotting functions from Microsoft Office, SAS, SPSS, and the base R plotting functions.

There are several R packages related to ROC curve estimation that contain dedicated plotting functions. The **ROCR** package (Sing, Sander, Beerenwinkel, and Lengauer 2005) plots the FPF versus TPF, as usual, and then takes the interesting approach of encoding the cutoff values as a separate color scale along the ROC curve itself. A legend for the color scale is place along the vertical axis on the right of the plotting region. The **pROC** package (Robin, Turck, Hainard, Tiberti, Lisacek, Sanchez, and Müller 2011) is mainly focused on estimating confidence intervals and regions for restricted ranges of the ROC curve. The plotting methods therein use the base R plotting functions to create nice displays of the curves along with shaded confidence regions.

1.4. Motivation

Anyone giving a cursory look at any of the major medical journals is likely to find at least one ROC curve plot. We sought assess the usage of ROC curve plots and to evaluate the design choices made in the current oncology literature by conducting a small literature review. We searched Pubmed for clinical trials or observational studies in humans reported in major oncology journals for the past 10 years for the terms "ROC Curve" OR "ROC Analysis" OR "Receiver operating characteristic curve". The search was conducted on October 8, 2014 and returned 54 papers. From those papers, 47 images were extracted and reviewed. The exact specifications for the Pubmed query are available in the supplementary materials.

Each image consisted of a single ROC curve plot or a panel of multiple plots. Each plot was inspected manually for the following design features: the number of curves displayed, the type of axis labels (sensitivity/ 1 - specificity or true/false positive fractions), presence or absence of grid lines, presence or absence of diagonal guide line, whether any cutpoint were indicated, the type of curve label (legend or direct label), and presence of other textual annotations such as the area under the curve. The numerical results of the survey are summarized in table 1.

	percent (count)
Number of curves	
1	19.6 (9)
2	43.5 (20)
3	10.9 (5)
4+	26.1 (12)
Average (SD)	2.6 (1.5)
Axis labels	
FPF/TPF	13.0 (6)
mixed	2.2 (1)
none	2.2 (1)
sens/spec	82.6 (38)
Diagonal Guide	43.5 (20)
Gridlines	17.4 (8)
Cutoffs indicated	15.2 (7)
AUC indicated	50.0 (23)
Curve Labels	
direct	10.9 (5)
legend	63.0 (29)
none	19.6 (9)
title	6.5 (3)

Table 1: Results of a literature review of major oncology journals for ROC curve plots. The rows indicate the frequency and count of key design elements of an ROC curve plot. FPR = False positive rate; TPR = True positive rate; sens = Sensitivity; spec = Specificity; AUC = Area under the Curve

The small minority of the figures make any attempt to indicate the values of the test cutoffs, which is an integral component of the ROC curve. We conjecture that this is mainly due to the use of default plotting procedures in statistical software. The software, by default, treats the ROC curve as a 2 dimensional object, obscuring the cutoff dimension. Gridlines and direct labels are also somewhat out of the ordinary. The absence of these features make accurate determination and comparison of the values more difficult. Many of the plots included large tables containing estimates and inference for AUCs, while the ROC curves themselves, numerous and without clear labels or reference lines, merely served as decoration. We aim to solve some of these problems by providing an easy-to-use plotting interface for the ROC curve that provides sensible defaults.

The panels of figure 1 illustrate the most common styles of ROC curve plots, and the associated design elements. We favor the use of gridline and reference lines to facilitate accurate readings off of the axes. Direct labels are preferred over legends because they omit the additional

cognitive step of matching line types to labels. Our **plotROC** package additionally provides plotting of cutoff values, using hover events in interactive use, and direct labels for print use. Exact confidence regions for points on the ROC curve are optionally calculated and displayed. Additionally, we use axis scales are adjusted to be denser near the margins 0 and 1. In medical applications, it is often necessary to have a very low FPR (less than 10%, for instance), therefore the smaller scales are useful for accurately determining values near the margins. The next section details the usage of the **plotROC** R package and these features.

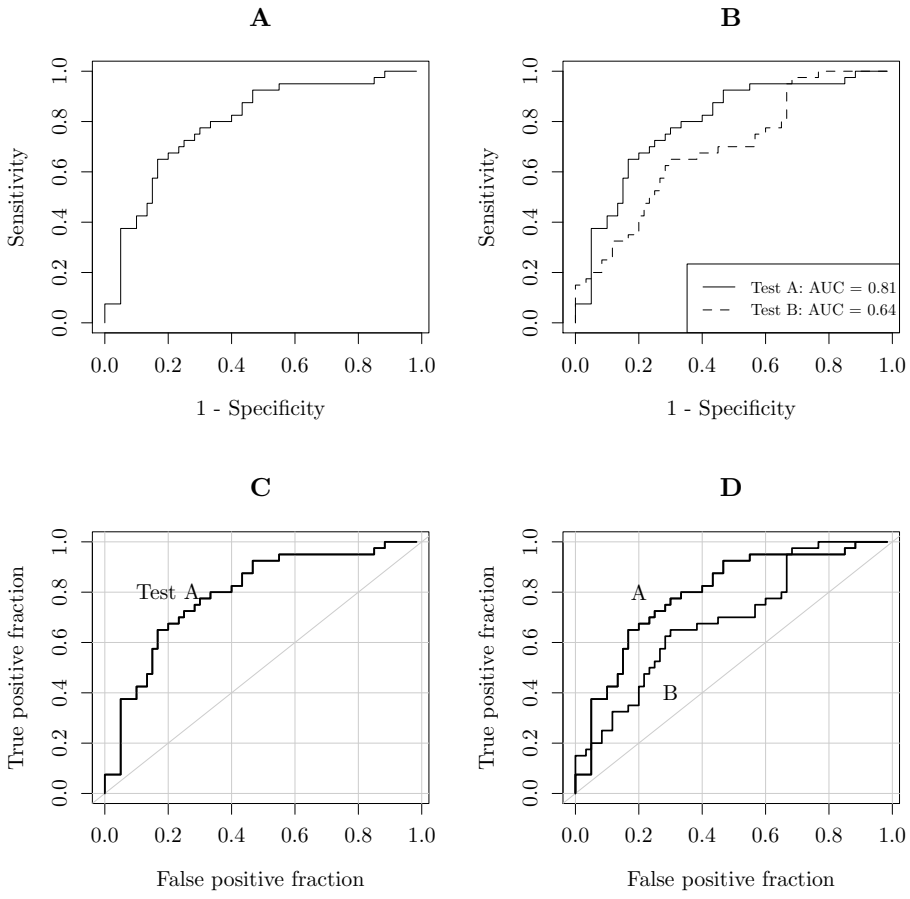

Figure 1: Illustration of design choices in plotting ROC curves. Panel A shows a sparse ROC curve, with no design additions inside the plotting region. The plot results in more white space than anything else. It is difficult to accurately determine values without reference lines. Panel B shows a plot comparing 2 curves, with different line types and a legend. AUCs are also given in the legend. Panels B and C add gridlines, diagonal reference lines, and direct labels.

2. Usage of the Package

2.1. Shiny application

We created a **shiny** application (RStudio and Inc. 2014b) in order to make the features more accessible to non-R users. A limited subset of the functions of the **plotROC** can be performed on an example dataset or on data that users upload to the website. Resulting plots can be saved to the users' machine as a pdf or as a stand-alone html file. It can be used in any modern web browser with no other dependencies at the website here: http://sachsmc.shinyapps.io/plotROC.

2.2. Quick start

After installing, the interactive Shiny application can be run locally.

```
shiny_plotROC()
```

2.3. Command line basic usage

We start by creating an example data set. The marker we generate is moderately accurate for predicting disease status.

```
library(plotROC)
D.ex <- rbinom(100, size = 1, prob = .5)
M.ex <- rnorm(100, mean = D.ex)
```

Next we use the `calculate_roc` function to compute the empirical ROC curve. The disease status need not be coded as 0/1, but if it is not, `plotROC` assumes (with a warning) that the lowest value in sort order signifies disease-free status. This returns a dataframe with three columns: the cutoff values, the TPF and the FPF.

```
rocdata <- calculate_roc(M.ex, D.ex)
str(rocdata)
```

```
'data.frame':   100 obs. of  3 variables:
 $ c  : num  -2.18 -1.67 -1.57 -1.55 -1.54 ...
 $ TPF: num  1 1 1 1 1 1 1 1 1 ...
 $ FPF: num  0.981 0.962 0.943 0.925 0.906 ...
```

The same data.frame `rocdata` can be used to generate an interactive plot to view in Rstudio viewer or web browser. A screen shot of an interactive plot is shown in figure 2. Hovering over the display shows the cutoff value at the point nearest to the cursor. Clicking makes the cutoff label stick until the next click, and if confidence regions are available, clicks will also display those as grey rectangles.

```
plot_interactive_roc(rocdata)
```

The `rocdata` is passed to the `ggroc` function with an optional label. This creates a ggplot object of the ROC curve using the **ggplot2** package (Wickham 2009).

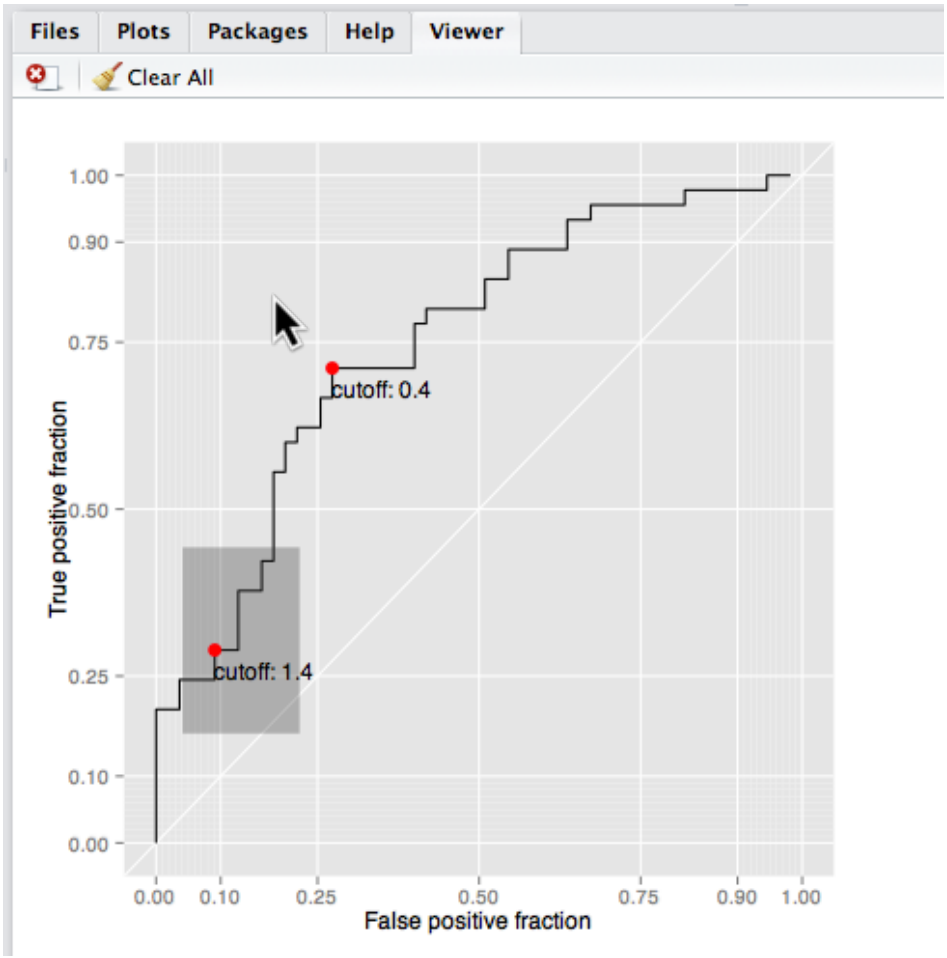

Figure 2: Screen shot of an interactive plot created with **plotROC** being displayed in the Rstudio viewer. Hovering the mouse cursor over the plot causes the cutoff label nearest to the cursor to be displayed. Clicking will display a confidence region, if available, and make the label stick until the next click. For live examples, see the package vignette, or go to http://sachsmc.github.io/plotROC.

```
myrocplot <- ggroc(rocdata, label = "Example")
```

We can create an interactive ROC plot using the `export_interactive_roc` function, which returns a character string containing the necessary HTML and JavaScript. The key interactive feature of these plots is that the cutoff values nearest to the mouse cursor are displayed when hovering over the figure. Clicking makes the cutoff label stick until the next click. The character string can be copy-pasted into an html document, or better yet, using **knitr** (Xie 2014), we can `cat` the results and use the option `results = 'asis'` so that the interactive plot is displayed correctly. For examples of interactive plots and how to incorporate them into `knitr` documents, see the package vignette (`vignette("examples", package = "plotROC")`) or the webpage https://sachsmc.github.io/plotROC/. Of note is the `prefix` option, which allows the user to assign each plot a unique identifier. This is necessary to prevent conflicts with multiple plots in a single webpage.

```
cat(
  export_interactive_roc(myrocplot, cutoffs = rocdata$c,
                         font.size = "12px", prefix = "a")
  )
```

The same `ggroc` object that we called `myrocplot` can be used to generate an ROC plot suitable for use in print. It annotates the cutoff values and is completely in black and white. A simple example with the default options is shown in figure 3.

```
plot_journal_roc(myrocplot, rocdata)
```

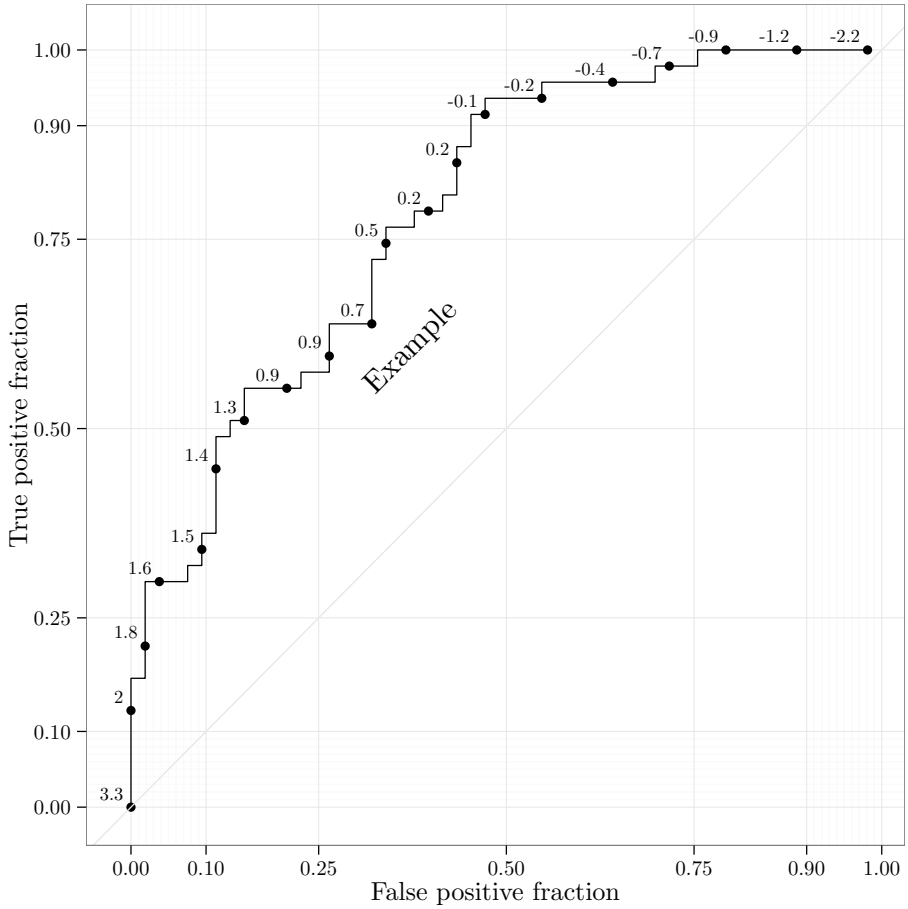

Figure 3: Illustration of ROC curve plot generated by **plotROC** for use in print.

2.4. Advanced options

Click to view confidence region

We use the `ci = TRUE` option in `calculcate_roc` and `ggroc` to compute confidence regions for points on the ROC curve using the Clopper and Pearson (1934) exact method. The significance level can be specified using the `alpha` option.

```
rocdata <- calculate_roc(M.ex, D.ex, ci = TRUE, alpha = 0.05)
myrocplot <- ggroc(rocdata, label = "Example", ci = TRUE)
```

For interactive plots, the confidence regions are automatically detected. When the user clicks on the ROC curve, the confidence region for the TPF and FPF is overlaid using a grey rectangle. The label and region stick until the next click.

```
cat(
  export_interactive_roc(myrocplot, cutoffs = rocdata$c,
                         font.size = "12px", prefix = "aci")
  )
```

For use in print, we pass a small vector of cutoff locations at which to display the confidence regions. This is shown in figure 4.

```
plot_journal_roc(myrocplot, rocdata, n.cuts = 10,
                 ci.at = c(-.5, .5, 2.1))
```

Multiple ROC curves

If you have multiple tests of different types on the same subjects, you can use the `calculate_multi_roc` function to compute the empirical ROC curve for each test. Then the `multi_ggroc` function creates the appropriate type of `ggplot` object. Confidence regions are not supported for multiple curves at the time of writing.

```
D.ex <- rbinom(100, 1, .5)

fakedata <- data.frame(M1 = rnorm(100, mean = D.ex),
                       M2 = rnorm(100, mean = D.ex, sd = .4),
                       M3 = runif(100), D = D.ex)

datalist <- calculate_multi_roc(fakedata, c("M1", "M2", "M3"), "D")
rocplot <- multi_ggroc(datalist)
```

This multi ggroc object can be passed to the `plot_journal_roc` and the `export_interactive_roc` functions, as desired.

Labels can be added easily with the `label` option. The length of the label element should match the number of plotted curves. The resulting plot is shown in figure 5.

```
rocplot <- multi_ggroc(datalist, label = c("M1", "M2", "M3"))
plot_journal_roc(rocplot, datalist)
```

Themes and annotations

`plotROC` uses the `ggplot2` package to create ggplot objects. Therefore, themes and annotations can be added to `ggroc` objects in the usual `ggplot` way. A `plotROC` figure with a new theme, title, axis label, and AUC annotation is shown in figure 6.

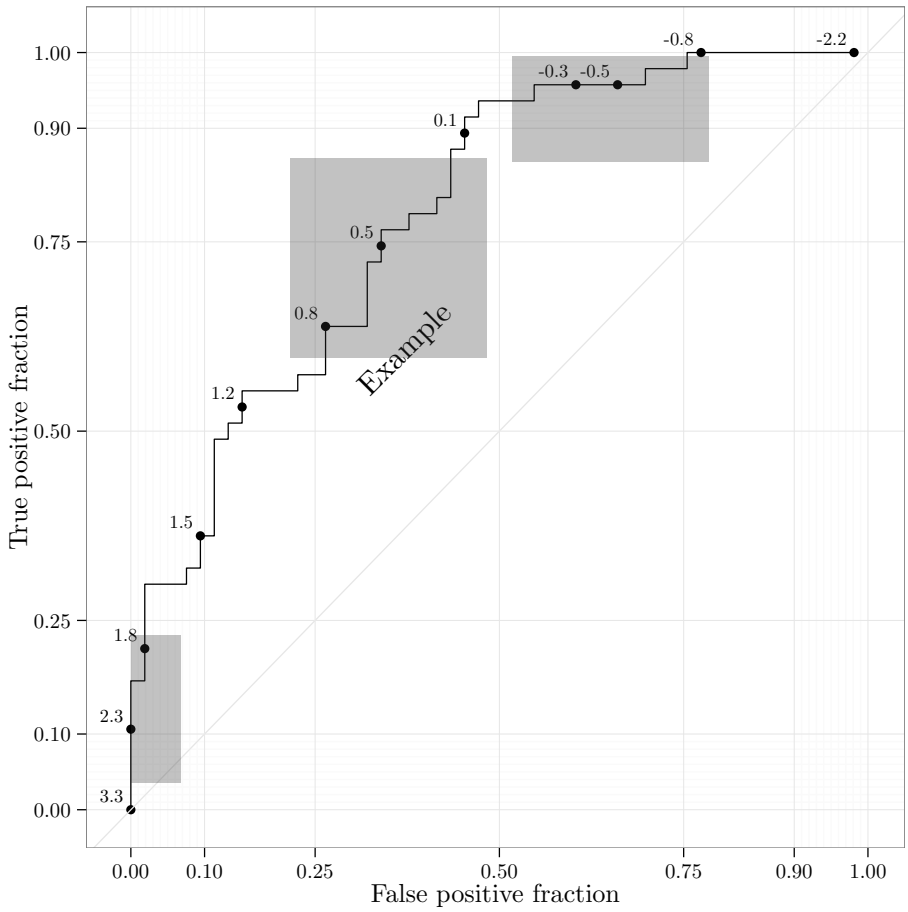

Figure 4: Illustration of **plotROC** plot with exact confidence regions.

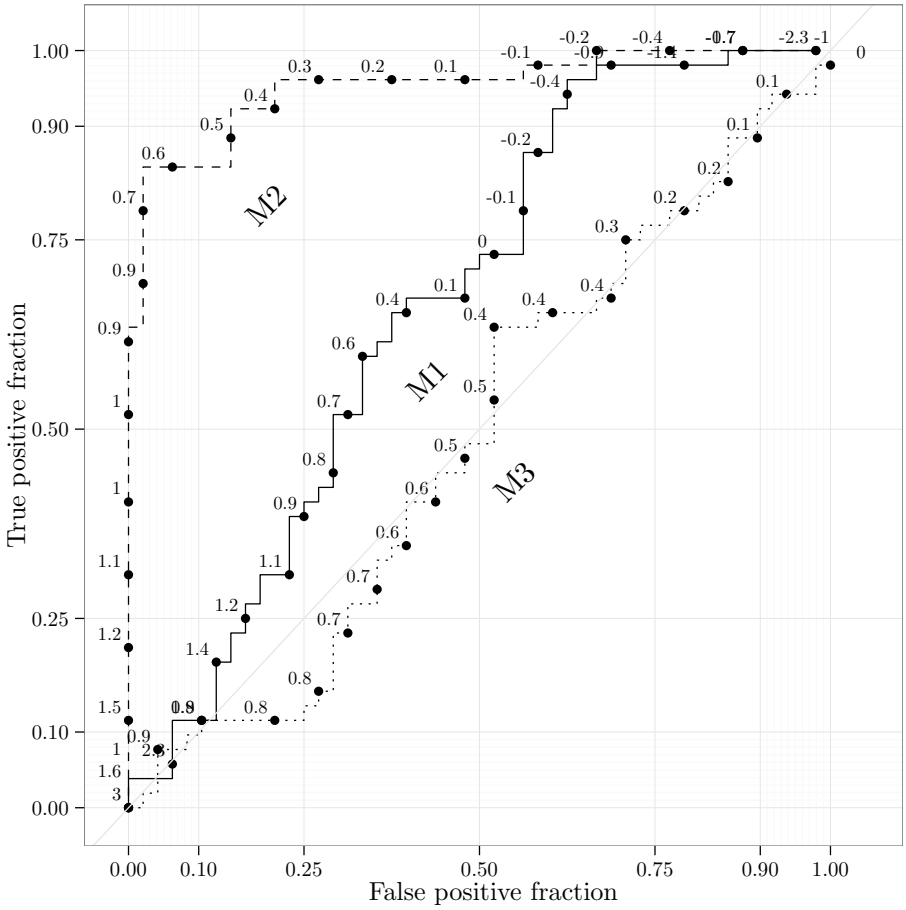

Figure 5: Illustration of **plotROC** plot with multiple curves.

```
library(ggplot2)
plot_journal_roc(myrocplot, rocdata) +
  theme_grey() +
  geom_abline(intercept = 0, slope = 1, color = "white") +
  ggtitle("Themes and annotations") +
  annotate("text", x = .75, y = .25,
           label = "AUC = 0.80") +
  ylab("Sensitivity")
```

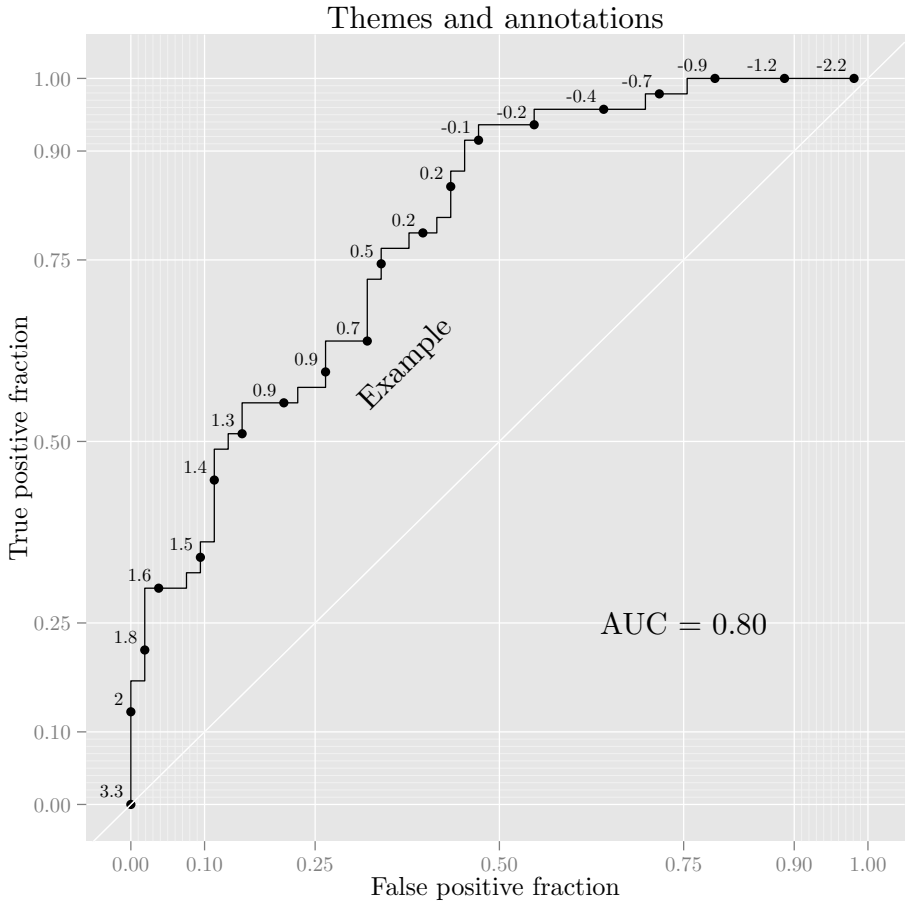

Figure 6: Using **ggplot2** themes and annotations with **plotROC** objects.

Other estimation methods

By default `calculate_roc` computes the empirical ROC curve. There are other estimation methods out there, as we have summarized in the introduction. Any estimation method can be used, as long as the cutoff, the TPF and the FPF are returned. Then you can simply pass those values in a data frame to the `ggroc` function. For example, let us use the binormal method to create a smooth curve. This approach assumes that the test distribution is normal conditional on disease status.

```
D.ex <- rbinom(100, 1, .5)
M.ex <- rnorm(100, mean = D.ex, sd = .5)

mu1 <- mean(M.ex[D.ex == 1])
mu0 <- mean(M.ex[D.ex == 0])
s1 <- sd(M.ex[D.ex == 1])
s0 <- sd(M.ex[D.ex == 0])
c.ex <- seq(min(M.ex), max(M.ex), length.out = 300)

binorm_rocdata <- data.frame(c = c.ex,
                             FPF = pnorm((mu0 - c.ex)/s0),
                             TPF = pnorm((mu1 - c.ex)/s1)
                             )
```

Then we can pass this data.frame to the `ggroc` function as before. The example is shown in figure 7.

```
binorm_plot <- ggroc(binorm_rocdata, label = "Binormal")
plot_journal_roc(binorm_plot, binorm_rocdata)
```

Another potential use of this approach is for plotting time-dependent ROC curves for time-to-event outcomes estimated as desribed in (Heagerty, Lumley, and Pepe 2000).

3. How it Works

plotROC makes use of **ggplot2** (Wickham 2009), **gridSVG** (Murrell and Potter 2014), and **d3.js** (Bostock, Ogievetsky, and Heer 2011) to create interactive plots. The first step in the process is to create `ggplot` objects using the `ggroc` or `multi_ggroc` functions. These functions return standard `ggplot` objects that include basic styling, hidden cutoff labels, and hidden confidence regions. They can be plotted and inspected in the R console. These form the basis for both the print versions and the interactive versions of the plots. Creating a print version by using the `plot_journal_roc` function simply makes visible a subset of the hidden cutoff labels and confidence regions, if available.

plotROC makes interactive plots by first converting the `ggplot` object into a scalable vector graphic (svg) object with the `grid.export` function. This function maps each element of the plot to a corresponding element of the svg markup language. We keep track of the names of the points and labels elements so that we can add interactivity using **d3.js** and JavaScript. The main interactive feature we wanted was to be able to display the cutoff labels at the points on the ROC curve closest to the mouse cursor.

There are many ways to solve this with **d3.js**, but we decided to use Voronoi polygons to map the cursor location to the nearest point on the ROC curve. The idea is that for the set of cutoff points along the ROC curve, the `d3.geom.voronoi` function chain computes a set of polygons overlaying the plotting region such that the area of each polygon contains the region of the plot closest to it's corresponding cutoff point. Hover events are bound to the polygons so that when the mouse cursor moves around the plotting region, the closest point on the ROC curve is made visible. Similarly, click events are bound to the polygons so that the

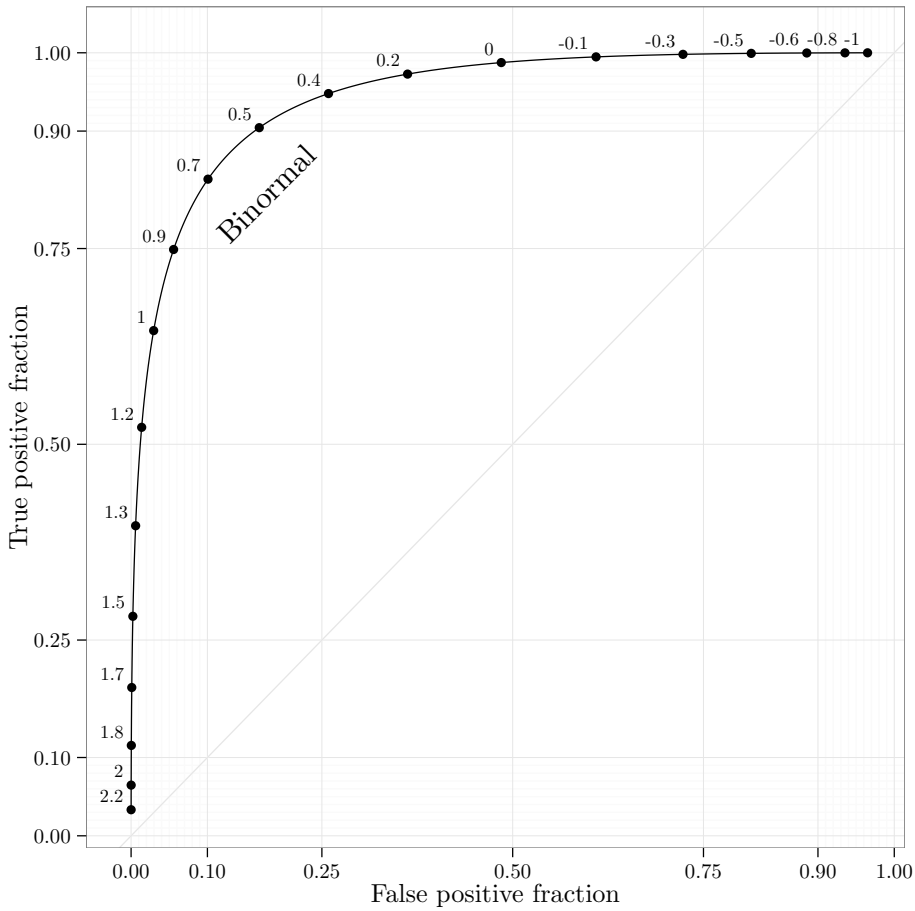

Figure 7: Illustration of smooth binormal ROC curve.

appropriate confidence region is made visible upon clicking. The svg code and all necessary JavaScript code is returned in the character string provided by `export_interactive_roc`. Figure 8 outlines the **plotROC** process.

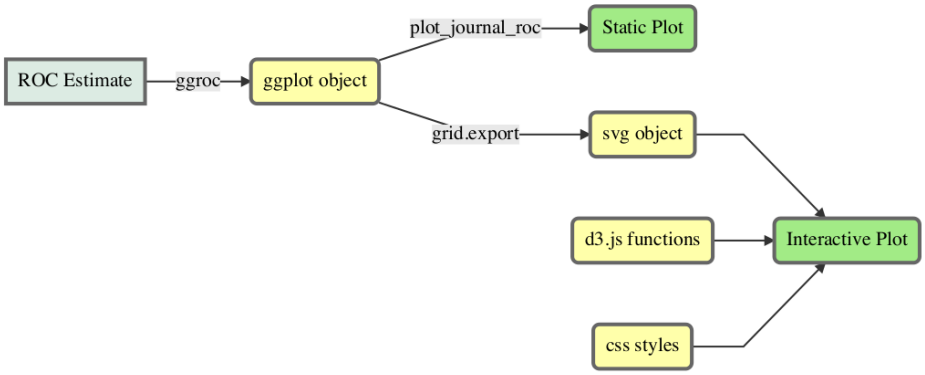

Figure 8: Flowchart illustrating the approach that **plotROC** takes to generate either static plots for print or interactive plots for web-use.

This approach is similar to what is done in the **gridSVG** `grid.animate` function, which uses the svg `<animate />` tags. However, the available features were not sufficient for our needs, which is why we used **d3.js**. There are several other R packages that aim to create interactive figures. The authors of **animint** (Hocking, Sievert, and VanderPlas 2014) created an extensive JavaScript library that creates plots in a similar way as **ggplot2**. A set of interactive features can be added to plots using **d3.js**. **ggvis** (RStudio and Inc. 2014a), **rCharts** (Reinholdsson, Russell, and Vaidyanathan 2014), and the more recently released **htmlwidgets** (Vaidyanathan, Cheng, Allaire, Xie, , and Russell 2014) all leverage existing charting libraries written in JavaScript. Their general approach is to manipulate the data and create options in R, and then let the charting libraries handle the rendering and interactivity. **plotROC** lets R do the rendering, allowing the figures to be consistent across print and web-based media, and retaining the distinctive R style.

4. Discussion

Here we have illustrated the usage and described the mechanics of a new R package for creating ROC curve plots. The functions are easy to use, even for non-R users *via* the web application, yet have sufficient flexibility to meet the needs of power users. Our approach to creating interactive plots differs from other interactive charting packages. We found that existing approaches did not meet the highly specialized needs of plotting ROC curves. While ROC curve plots can technically be created with even the most basic plotting tools, we find that specialized functions make the results clearer and more informative.

References

Bostock M, Ogievetsky V, Heer J (2011). "D^3 data-driven documents." *Visualization and Computer Graphics, IEEE Transactions on*, **17**(12), 2301–2309.

Clopper C, Pearson ES (1934). "The use of confidence or fiducial limits illustrated in the case of the binomial." *Biometrika*, **26**(4), 404–413.

Heagerty PJ, Lumley T, Pepe MS (2000). "Time-dependent ROC curves for censored survival data and a diagnostic marker." *Biometrics*, **56**(2), 337–344.

Hocking TD, Sievert C, VanderPlas S (2014). ***animint**: a Grammar for Interactive Animations*. R package version 2014.12.4, URL https://github.com/tdhock/animint.

Murrell P, Potter S (2014). *gridSVG: Export grid Graphics as SVG*. R package version 1.4-2, URL http://CRAN.R-project.org/package=gridSVG.

Pepe MS (2003). *The statistical evaluation of medical tests for classification and prediction*. Oxford University Press.

Reinholdsson T, Russell K, Vaidyanathan R (2014). *Interactive Charts using Javascript Visualization Libraries*. R package version 0.4.5, URL http://ramnathv.github.io/rCharts/.

Robin X, Turck N, Hainard A, Tiberti N, Lisacek F, Sanchez JC, Müller M (2011). "pROC: an open-source package for R and S+ to analyze and compare ROC curves." *BMC Bioinformatics*, **12**, 77.

RStudio, Inc (2014a). *ggvis: Interactive grammar of graphics*. R package version 0.4, URL http://CRAN.R-project.org/package=ggvis.

RStudio, Inc (2014b). *shiny: Web Application Framework for R*. R package version 0.10.2.2, URL http://CRAN.R-project.org/package=shiny.

Sing T, Sander O, Beerenwinkel N, Lengauer T (2005). "ROCR: visualizing classifier performance in R." *Bioinformatics*, **21**(20), 7881. URL http://rocr.bioinf.mpi-sb.mpg.de.

Vaidyanathan R, Cheng J, Allaire J, Xie Y, , Russell K (2014). *htmlwidgets: HTML Widgets for R*. R package version 0.3.2, URL http://CRAN.R-project.org/package=htmlwidgets.

Wickham H (2009). *ggplot2: elegant graphics for data analysis*. Springer New York. ISBN 978-0-387-98140-6. URL http://had.co.nz/ggplot2/book.

Xie Y (2014). *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.8.

Affiliation:

Michael C. Sachs
9609 Medical Center Drive, MSC 9735
Bethesda, MD 20892
telephone: 240-276-7888
michael.sachs@nih.gov

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume VV, Issue II *Submitted:* yyyy-mm-dd
MMMMMM YYYY *Accepted:* yyyy-mm-dd