# Biostatistics for Pharmacy Research

Frank E Harrell Jr
James C Slaughter
Leena Choi
Department of Biostatistics
Vanderbilt University School of Medicine
f.harrell@vanderbilt.edu
james.c.slaughter@vanderbilt.edu
leena.choi@vanderbilt.edu



Updated October 19, 2017

# Contents

Blue symbols in the right margin starting with ABD designate section numbers (and occasionally page numbers preceeded by $p$) in *The Analysis of Biological Data, Second Edition* by MC Whitlock and D Schluter, Greenwood Village CO, Roberts and Company, 2015. Likewise, right blue symbols starting with RMS designate section numbers in *Regression Modeling Strategies, 2nd ed.* by FE Harrell, Springer, 2015.

 in the right margin indicates a hyperlink to a YouTube video related to the subject.

 in the right margin is a hyperlink to an audio file[a] elaborating on the notes. Red letters and numbers in the right margin are cues referred to within the audio recordings.

Here is a link to the playlist for all audio files in these notes.

Rotated boxed blue text in the right margin at the start of a section represents the mnemonic key for linking to discussions about that section in `vbiostatcourse.slack.com` channel `#bbr`. Anyone starting a new discussion about a topic related to the section should include the mnemonic somewhere in the posting, and the posting should be marked to `slack` as threaded. The mnemonic in the right margin is also a hyperlink to a search in the `bbr` channel for messages containing the mnemonic. When you click on it the relevant messages will appear in the search results on the right side of the `slack` browser window.

howto

Members of the `slack` group can also create submnemonics for subsections or other narrower-scope parts of the notes. When creating keys "on the fly," use names of the form `chapterkey-sectionkey-yourkey` where `sectionkey` is defined in the notes. That way a search on `chapterkey-sectionkey` will also bring up notes related to `yourkey`.

Several longer and more discussed subsections in the text have already been given short keys in these notes.

blog in the right margin is a link to a blog entry that further discusses the topic.

[a] The first time you click on one of these, some browsers download the audio file and give you the opportunity to right click to open the file on your local audio player, then the browser asks if you always want to open files of this type. Select "yes".

# Chapter 1

# R

## 1.1 Background

Computer code shown throughout these notes is R[7]. R is free and is the most widely used statistical software in the world. It has the best graphics, statistical modeling, nonparametric methods, survival analysis, clinical trials methods, and data manipulation capabilities. R has the most comprehensive genomics analysis packages and has advanced capabilities for reproducible analysis and reporting. R also has an excellent graphical front-end `RStudio` (`rstudio.org`) that has the identical look and feel on all operating systems and via a web browser. Part of R's appeal is the thousands of add-on packages available (at `http://cran.r-project.org/web/packages`), which exist because it is easy to add to R. Many of the add-on packages are specialty packages for biomedical research including packages for such widely diverse areas as

- interfacing R to `REDCap` (2 packages)
- interactive design of adaptive clinical trials
- analyzing accelerometer data
- flow cytometry
- genomics
- analyzing ICD9 codes and computing comorbidity indexes

- downloading all annotated NHANES datasets
- interfacing to clinicaltrials.gov
- detecting whether a dataset contains personal identifiers of human subjects
- analysis of early phase cardiovascular drug safety studies

The main R web site is www.r-project.org.

1.2

# Learning R

Start with *R Tutorials* at http://www.r-bloggers.com/how-to-learn-r-2 and *R Programming Tutorials* from Mike Marin at https://www.youtube.com/user/marinstatlectures. A good list of books and other material for learning R may be found at
http://www.revolutionanalytics.com/r-language-resources. Those who have used SPSS or SAS before will profit from R *for SAS and SPSS Users* by Robert Muenchen. A current list of R books on amazon.com may be found at http://amzn.to/15URiF6. http://www.ats.ucla.edu/stat/r and http://www.introductoryr.co.uk/R_Resources_for_Beginners.html are useful web sites. See also R *in Action, second ed.* by Robert Kavacoff. http://stackoverflow.com/tags/r is the best place for asking questions about the language and for learning from answers to past questions asked (see also the `R-help` email list).

Three of the best ways to learn how to analyze data in R quickly are

1. Avoid importing and manipulating data, instead using the R `load` function to load datasets that are already annotated and analysis-ready (see Section 1.6 for information about importing your own datasets)
2. Use example R scripts as analysis templates
3. Use `RStudio` (rstudio.org) to run R

On the first approach, the R `Hmisc` packages's `getHdata` function finds datasets on the Vanderbilt Biostatistics `DataSets` wiki, downloads them, and `load()`s them in your R session. These notes use only datasets available via this mechanism. These datasets are fully annotated with variable labels and units of measurements for many of the continuous variables. Concerning analysis scripts, Vanderbilt Biostatistics has collected template analysis scripts on https://github.com/harrelfe/rscripts[a] and the R `Hmisc` package has a function `getRs` to download these scripts and to automatically populate an `RStudio` script editor window with the script. Many of the scripts are in RMarkdown format for use with the R `knitr` package to allow mixing of text and R code to make reproducible reports. `knitr` is described in Section **??**.

[a] `github` has outstanding version control and issue reporting/tracking. It greatly facilitates the contribution of new scripts by users, which are most welcomed. Contact `f.harrell@vanderbilt` if you have scripts to contribute or suggestions for existing scripts.

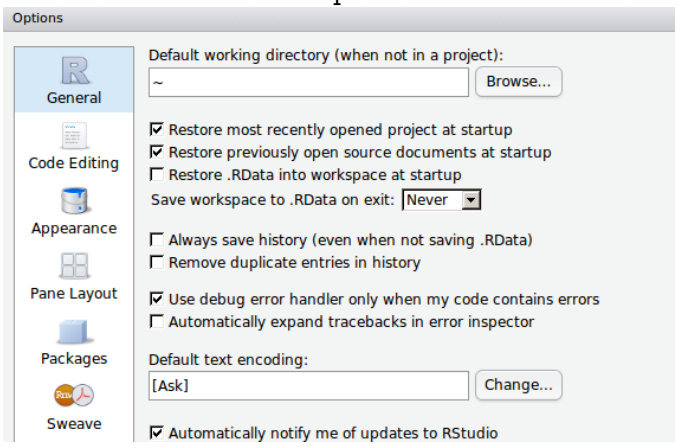
The RMarkdown scripts accessed through `getRs` use a template that makes the result part of a reproducible research process by documenting the versions of R and attached packages at the end of the report. Some of the scripts make use of the `knitrSet` function in the `Hmisc` package. When running Rmarkdown, call `knitrSet(lang='markdown')`. `knitrSet` gets rid of ## at the start of R output lines, and makes it easy to specify things like figure sizes in knitr chunk headers. It also causes annoying messages such as those generated from attaching R packages to be put in a separate file `messages.txt` rather than in the report.

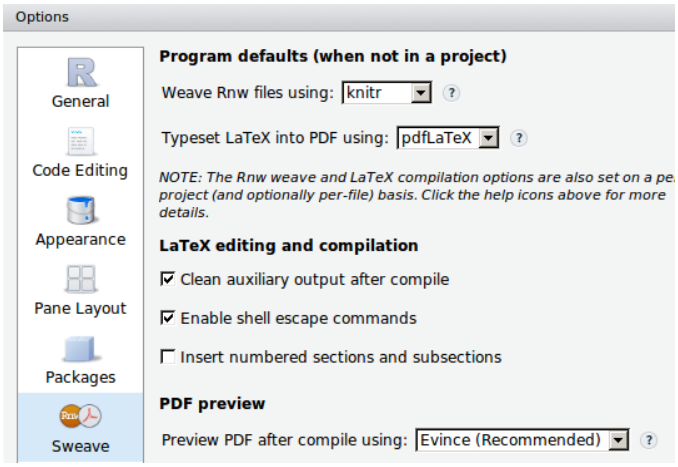
1.3

# Setting up R

Before running examples in these notes and R markdown example scripts, you need to do the following:

1. Make sure your operating system is up to date enough to run the most current version of R at www.r-project.org. For Mac you must have OS X Maverick or later.
2. Install R from www.r-project.org or upgrade your installation of R to the latest version.
3. Install `RStudio` from rstudio.org or update your `RStudio` to the latest version.
4. Run `RStudio` and get it to install the packages that allow `Rmarkdown` to run, by clicking on `File ... New File ... R Markdown`. Make sure that the `knitr` package is installed.
5. Have `RStudio` install the `Hmisc` and `rms` packages (which will make `RStudio` install several other packages). For packages you had installed previously, make sure you update them to have the latest versions.
6. Configure `RStudio Tools ... Global Options` to match the images below






Here are some examples of how `getRs` is used once you load the `Hmisc` package using a menu or by typing `require(Hmisc)` or `library(Hmisc)` in the console.

```
require(Hmisc)       # do this once per session (or library(Hmisc))
options(url.method='libcurl')     # sometimes needed if using Windows
getRs()              # list available scripts
getRs(browse='browser')  # open scripts contents in your web browser
scripts ← getRs()    # store directory of scripts in an object that can easily
                     # be viewed on demand in RStudio (right upper pane)
getRs('introda.r')   # download introda.r and open in script editor
getRs(cats=TRUE)     # list available major and minor categories
categories ← getRs(cats=TRUE)  # store results in a list for later viewing
getRs(cats='reg')    # list all scripts in a major category containing 'reg'
getRs('importREDCap.r', put='source')    # source() to define a function
```

You can also point your browser to https://github.com/harrelfe/rscripts/blob/master/contents.md to see the available scripts and categories, and to be able to click on links to see `html` report output.

To get started using R in `RStudio` to create reproducible annotated reports, finish the above configuration instructions and type the following in the `RStudio` console: `getRs('descriptives.Rmd')`. The above the script editor window click on `Knit HTML`.

## 1.4 Using R Markdown

See http://kbroman.org/knitr_knutshell/pages/Rmarkdown.html and print the R Markdown cheat sheet from http://www.rstudio.com/resources/cheatsheets.

To make the code listing pretty, put this chunk at the top of your report. echo=FALSE suppresses this setup chunk from printing in the report.

```
```{r setup,echo=FALSE}
require(Hmisc)
knitrSet('myreport', lang='markdown')
```
```

The argument 'myreport' is replaced with a string to use as a prefix to all the graphics file names, making each report in your working directory use graphics file names that do not collide with each other. For example if your report is called acidity_analysis.Rmd you might specify knitrSet('acidity_analysis.Rmd', lang='markdown'). There are many other options to knitrSet. A commonly used one is width=n to specify a line width for printing code of n letters. The default is 61. You can also specify echo,results, and other options. Type ?knitrSet for help.

The R knitr package is used to run the markdown report and insert graphics and text output into the report at appropriate slots. It is best to specify a name for each chunk, and you must use unique names. Each R code chunk must begin exactly with ```{r ...} and the chunk name is the first set of characters that appear after the space after r. Here are some example chunk headers. Chunk names must not contain a space.

```
```{r descriptives}
```{r anova}
```{r anova-y1}
```{r anova_y1}
```{r acidity_plot}
```{r plot_residuals,top=1}
```{r plot_residuals,mfrow=c(2,2),left=1,top=1,rt=1,bot=1}
```{r plot-residuals,w=5,h=4}
```

Chunk options that were used above are:

| Options | Description |
|---|---|
| `top=1` | Leave an extra line of space at top of graph for title |
| `mfrow=c(2,2)` | Use base graphics and put the next 4 plots into a single figure with 2 rows, 2 columns |
| `left=1,rt=1,bot=1` | Leave one extra line for margin for left, right, bottom of figure |
| `w=5,h=4` | Make the figure larger than the default that `knitrSet` uses (4 inch width by 3 inch height) |

Always having a chunk name also allows easy navigation of chunks by clicking to the right of the green `C` at the bottom of your script. This will show the names of all chunks and you can click on one to go there.

1.5

# Debugging R Code

When using `RStudio` and `knitr` as with `RMarkdown`, it is best to debug your commands a piece at a time. The fastest way to do this is to go to some line inside your first chunk and click the green `C` just above and to the right of your script. Click on `Run Current Chunk` then on `Run Next Chunk`. Shortcut keys for these are `Ctrl+Alt+C` and `Ctrl+Alt+N` (`Command+Option+C` and `Command+Option+N` for Mac). You can also click on a single line of code and run it by clicking on `Run`.

Whenever you get a strange execution error it is sometimes helpful to show the history of all the function calls leading to that error. This is done by typing `traceback()` at the command prompt.

## 1.6 Importing Other Datasets

Most of the work of getting some data sources ready for analysis involves reshaping datasets from wide to tall and thin, recoding variables, and merging multiple datasets. R has first-class capabilities for all of these tasks but this part of R is harder to learn, partly because there are so many ways to accomplish these tasks in R. Getting good variable names, variable labels, and value labels, can also be tedious but is highly worth the time investment.

### 1.6.1 Stata and SPSS

If you have Stata or SPSS files that are already shaped correctly and have variable labels and value labels, the R `Hmisc` package's `stata.get` and `spss.get` functions will produce fully annotated ready-to-analyze R data frames.

### 1.6.2 REDCap

`REDCap` exports data to R, and Biostatistics has an R function to make the import process much easier. Here is an example showing how to fetch and use the function. In this example, the user did not provide the name of the file to import but rather let the function find the last created REDCap export files in the current working directory.

```
require(Hmisc)
getRs('importREDCap.r', put='source')  # source() code to define function
mydata ← importREDCap()  # by default operates on last downloaded export
Save(mydata)   # Hmisc function to create mydata.rda in compressed format
```

Advanced users can hook into `REDCap` dynamically with R to avoid the need to export/import.

1.6.3

# Spreadsheets

If you have a properly formatted csv file (e.g., exported from a spreadsheet), the Hmisc csv.get function will read it, facilitate handling of date variables, convert column names to legal R names, and save the original column names as variable labels.

Here is an example of importing a csv file into R. First of all make sure your spreadsheet is a "spreadsheet from heaven" and not a "spreadsheet from hell" by reading http://biostat.mc.vanderbilt.edu/DataTransmissionProcedures. Then use your spreadsheet software to export a single worksheet to create a csv file. Small csv files may be pasted into your R script as is done in the following example, but in most cases you will call csv.get with an external file name as the first argument.

```
# What is between data ← .. and ') is exactly like an external .csv file
data ← textConnection('
Age in Years,sex,race,visit date,m/s
23,m,w,10/21/2014,1.1
14,f,b,10/22/2014,1.3
,f,w,10/15/2014,1.7
')
require(Hmisc)
d ← csv.get(data, lowernames=TRUE, datevars='visit.date',
            dateformat='%m/%d/%Y')
close(data)
# lowernames=TRUE: convert variable names to lower case
# Omit dateformat if dates are in YYYY-MM-DD format
contents(d)
```

```
Data frame:d    3 observations and 5 variables    Maximum # NAs:1


                  Labels Levels   Class Storage NAs
age.in.years Age in Years        integer integer   1
sex                   sex      2          integer   0
race                 race      2          integer   0
visit.date     visit date           Date  double   0
m.s                   m/s        numeric  double   0

+--------+------+
|Variable|Levels|
+--------+------+
|  sex   |  f,m |
+--------+------+
|  race  |  b,w |
+--------+------+
```

```
d
```

```
  age.in.years sex race visit.date m.s
1           23   m    w 2014-10-21 1.1
2           14   f    b 2014-10-22 1.3
3           NA   f    w 2014-10-15 1.7
```

In the `contents` output above you can see that the original column names have been placed in the variable labels, and the new names have periods in place of blanks or a slash, since these characters are illegal in R names.

You can have as the first argument to `csv.get` not only a file name but a URL to a file on the web. You can also specify delimiters other than commas.

Also see the excellent tutorial on importing from Excel found at http://www.r-bloggers.com/r-tutorial-on-reading-and-importing-excel-files-into-r.

The `Hmisc upData` function may be used to rename variables and provide variable and value labels and units of measurement. Here is another example where there is a `junk` variable to delete after importing, and a categorical variable is coded as integers and need to have value labels defined after importing. We show how `csv.get` automatically renamed one illegal (to R) variable name, how to redefine a variable label, and how to define the value labels. Suppose that file `test.csv` exists in our project directory and has the following contents.

```
age,sys bp,sex,junk,state
23,140,male,1,1
42,131,female,2,1
45,127,female,3,2
37,141,male,4,2
```

Now import and modify the file.

```
require(Hmisc)
d ← csv.get('test.csv')
names(d)    # show names after modification by csv.get
```

```
[1] "age"    "sys.bp" "sex"    "junk"   "state"
```

```
contents(d)   # show labels created by csv.get
```

```
Data frame:d    4 observations and 5 variables    Maximum # NAs:0


       Labels Levels   Class Storage
age       age         integer integer
sys.bp sys bp         integer integer
sex       sex      2          integer
junk     junk         integer integer
state   state         integer integer
```

```
+--------+-----------+
|Variable|Levels     |
+--------+-----------+
|   sex  |female,male|
+--------+-----------+
```

```
d ← upData(d,
           state=factor(state, 1:2, c('Alabama','Alaska')),
           rename=c(sys.bp='sbp'),
           labels=c(age = 'Age',
                    sbp = 'Systolic Blood Pressure'),
           drop='junk',    # for > 1: drop=c('junk1','junk2',...)
  units=c(sbp='mmHg'))
```

```
Input object size:       3848 bytes;     5 variables     4 observations
Renamed variable         sys.bp         to sbp
Modified variable       state
Dropped variable        junk
New object size:        3456 bytes;     4 variables     4 observations
```

```
contents(d)
```

```

Data frame:d    4 observations and 4 variables    Maximum # NAs:0


                     Labels Units Levels   Class Storage
age                     Age              integer integer
sbp   Systolic Blood Pressure  mmHg        integer integer
sex                     sex            2         integer
state                                  2         integer

+--------+--------------+
|Variable|Levels        |
+--------+--------------+
|  sex   |female,male   |
+--------+--------------+
|  state |Alabama,Alaska|
+--------+--------------+
```

```
describe(d)
```

```
d

 4  Variables      4  Observations
--------------------------------------------------------------------------------
age : Age
       n  missing distinct     Info     Mean      Gmd
       4        0        4        1    36.75    11.83

Value        23   37   42   45
Frequency     1    1    1    1
Proportion 0.25 0.25 0.25 0.25
--------------------------------------------------------------------------------
sbp : Systolic Blood Pressure [mmHg]
       n  missing distinct     Info     Mean      Gmd
```

```
       4         0         4         1     134.8       8.5

Value        127   131   140   141
Frequency      1     1     1     1
Proportion  0.25  0.25  0.25  0.25
-------------------------------------------------------------------------
sex
       n  missing distinct
       4         0         2

Value      female    male
Frequency       2       2
Proportion    0.5     0.5
-------------------------------------------------------------------------
state
       n  missing distinct
       4         0         2

Value      Alabama   Alaska
Frequency        2        2
Proportion     0.5      0.5
-------------------------------------------------------------------------
```

```
dim(d); nrow(d); ncol(d); length(d)   # length is no. of variables
```

```
[1] 4 4
```

```
[1] 4
```

```
[1] 4
```

```
[1] 4
```

### 1.6.4 Defining Small Datasets Inline

For tiny datasets it is easiest to define them as follows:

```
d ← data.frame(age=c(10,20,30), sex=c('male','female','male'),
               sbp=c(120,125,NA))
```

Large files may be stored in R binary format using `save(..., compress=TRUE)`, which creates an incredibly compact representation of the data in a file usually suffixed with `.rda`. This allows extremely fast loading of the data frame in your next R session using `load(...)`. The `Hmisc Save` and `Load` functions make this even easier.

## 1.7 Suggestions for Initial Data Look

The `datadensity` function in the `Hmisc` package gives an overall univariable graphical summary of all variables in the imported dataset. The `contents` and `describe` functions are handy for describing the variables, labels, number of `NA`s, extreme values, and other values.

## 1.8 Operating on Data Frames

One of the most common operations is subsetting. In the following example we subset on males older than 26.

```
young.males ← subset(d, sex == 'male' & age > 26)
# If you want to exclude rows that are missing on sex or age:
young.males ← subset(d, sex == 'male' & age > 26 & ! is.na(sex) &
                         ! is.na(age))
# f ← lrm(y ~ sex + age, data=subset(d, sex == 'male' & ...))
# f ← lrm(y ~ sex + age, data=d, subset=sex == 'male' & age > 26 ...)
```

# Chapter 2

# Algebra Review

## 2.1 Overview

Algebra and probability are underlying frameworks for basic statistics. The following elements of algebra are particularly important:

- Understanding symbols as variables, and what they can stand for
- Factoring out common terms: $axw + bx = x(aw + b)$
- Factoring out negation of a series of added terms: $-a - b = -(a + b)$
- Simplification of fractions
- Addition, subtraction, multiplication, and division of fractions
- Exponentiation with both fractional and whole number exponents
- Re-writing exponentials of sums: $b^{u+v} = b^u \times b^v$
- Logarithms
  - log to the base $b$ of $x = \log_b x$ is the number $y$ such that $b^y = x$
  - $\log_b b = 1$

  - $\log_b b^x = x \log_b b = x$
  - $\log_b a^x = x \log_b a$
  - $\log_b a^{-x} = -x \log_b a$
  - $\log_b(xy) = \log_b x + \log_b y$
  - $\log_b \frac{x}{y} = \log_b x - \log_b y$
  - When $b = e = 2.71828\ldots$, the base of the natural log, $\log_e(x)$ is often written as $\ln x$ or just $\log(x)$
  - $\log e = \ln e = 1$

- Anti-logarithms: anti-log to the base $b$ of $x$ is $b^x$
  - The natural anti-logarithm is $e^x$, often often written as $\exp(x)$
  - Anti-log is the inverse function of log; it "undoes" a log

- Understanding functions in general, including $\min(x, a)$ and $\max(x, a)$

- Understanding indicator variables such as $[x = 3]$ which can be thought of as true if $x = 3$, false otherwise, or 1 if $x = 3$, 0 otherwise
  - $[x = 3] \times y$ is $y$ if $x = 3$, 0 otherwise
  - $[x = 3] \times [y = 2] = [x = 3 \text{ and } y = 2]$
  - $[x = 3] + 3 \times [y = 2] = 4$ if $x = 3$ and $y = 2$, 3 if $y = 2$ and $x \neq 3$
  - $x \times \max(x, 0) = x^2[x > 0]$
  - $\max(x, 0)$ or $w \times [x > 0]$ are algebraic ways of saying to ignore something if a condition is not met

- Quadratic equations

- Graphing equations

Once you get to multiple regression, some elements of vectors/linear algebra are helpful, for example the vector or dot product, also called the inner product:

- Let $x$ stand for a vector of quantities $x_1, x_2, \ldots, x_p$ (e.g., the values of $p$ variables for an animal such as age, blood pressure, etc.)
- Let $\beta$ stand for another vector of quantities $\beta_1, \beta_2, \ldots, \beta_p$ (e.g., weights / regression coefficients / slopes)
- Then $x\beta$ is shorthand for $\beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p$
- $x\beta$ might represent a predicted value in multiple regression, and is known then as the *linear predictor*

## 2.2

# Some Resources

- http://tutorial.math.lamar.edu/pdf/Algebra_Cheat_Sheet.pdf
- https://www.khanacademy.org/math/algebra
- http://biostat.mc.vanderbilt.edu/PrereqAlgebra
- http://www.purplemath.com/modules/index.htm

# Chapter 3

# General Overview of Biostatistics

There are no routine statistical questions, only questionable statistical routines.

Sir David R. Cox

It's much easier to get a *result* than it is to get an *answer*.

Christie Aschwanden, FireThirtyEight

ABD1.1,p.23-4

## 3.1 What is Biostatistics?

- Statistics applied to biomedical problems
- Decision making in the face of uncertainty or variability
- Design and analysis of experiments; detective work in observational studies (in epidemiology, outcomes research, etc.)
- Attempt to remove bias or find alternative explanations to those posited by researchers with vested interests
- Experimental design, measurement, description, statistical graphics, data analysis, inference

To optimize its value, biostatistics needs to be fully integrated into biomedical research and we must recognize that experimental design and execution (e.g., randomization and masking) are all important.

### 3.1.1 Fundamental Principles of Statistics

- Use methods grounded in theory or extensive simulation
- Understand uncertainty
- Design experiments to maximize information and understand sources of variability
- Use all information in data during analysis
- Use discovery and estimation procedures not likely to claim that noise is signal
- Strive for optimal quantification of evidence about effects

- Give decision makers the inputs (*other* than the utility function[a]) that optimize decisions

- Present information in ways that are intuitive, maximize information content, and are correctly perceived

[a]The utility function is also called the loss or cost function. It specifies, for example, the damage done by making various decisions such as treating patients who don't have the disease or failing to treat those who do. The optimum Bayes decision is the one that minimizes expected loss. This decision conditions on full information and uses for example predicted risk rather than whether or not the predicted risk is high.

## 3.2

# Types of Data Analysis and Inference

- Description: what happened to *past* patients
- Inference from specific (a sample) to general (a population)
  - Hypothesis testing: test a hypothesis about population or long-run effects
  - Estimation: approximate a population or long term average quantity
  - Prediction: predict the responses of other patients *like yours* based on analysis of patterns of responses in your patients

## 3.3 Types of Measurements by Their Role in the Study

ABD1.3

- Response variable (clinical endpoint, final lab measurements, etc.)
- Independent variable (predictor or descriptor variable) — something measured when a patient begins to be studied, before the response; often not controllable by investigator, e.g. sex, weight, height, smoking history
- Adjustment variable (confounder) — a variable not of major interest but one needing accounting for because it explains an apparent effect of a variable of major interest or because it describes heterogeneity in severity of risk factors across patients
- Experimental variable, e.g. the treatment or dose to which a patient is randomized; this is an independent variable under the control of the researcher

Table 3.1: Common alternatives for describing independent and response variables

| Response variable | Independent variable |
|---|---|
| Outcome variable | Exposure variable |
| Dependent variable | Predictor variable |
| $y$-variables | $x$-variable |
| Case-control group | Risk factor |
| | Explanatory variable |

### 3.3.1 Proper Response Variables

It is too often the case that researchers concoct response variables $Y$ in such a way that makes the variables *seem* to be easy to interpret, but which contain several hidden problems:

- $Y$ may be a categorization/dichotomization of an underlying continuous response variable. The cutpoint used for the dichtomization is never consistent with data

(see Figure **??**), is arbitrary (P. **??**), and causes a huge loss of statistical information and power (P. **??**).

- $Y$ may be based on a change in a subject's condition whereas what is truly important is the subject's most recent condition (P. **??**).

- $Y$ may be based on change when the underlying variable is not monotonically related to the ultimate outcome, indicating that positive change is good for some subjects and bad for others (Fig. **??**).

A proper response variable that optimizes power is one that

- Captures the underlying structure or process

- Has low measurement error

- Has the highest resolution available, e.g.
  - is continuous if the underlying measurement is continuous
  - is ordinal with several categories if the underlying measurement is ordinal
  - is binary only if the underlying process is truly all-or-nothing

- Has the same interpretation for every type of subject, and especially has a direction such that higher values are always good or always bad

## 3.4

# Types of Measurements According to Coding

ABD1.3

- Binary: yes/no, present/absent
- Categorical (nominal, polytomous, discrete): more than 2 values that are not necessarily in special order
- Ordinal: a categorical variable whose possible values are in a special order, e.g., by severity of symptom or disease; spacing between categories is not assumed to be useful
  - Ordinal variables that are not continuous often have heavy ties at one or more values requiring the use of statistical methods that allow for strange distributions and handle ties well
  - Continuous are also ordinal but ordinal variables may or may not be continuous
- Count: a discrete variable that (in theory) has no upper limit, e.g. the number of ER visits in a day, the number of traffic accidents in a month
- Continuous: a numeric variable having many possible values representing an underlying spectrum
- Continuous variables have the most statistical information (assuming the raw values are used in the data analysis) and are usually the easiest to standardize across hospitals
- Turning continuous variables into categories by using intervals of values is arbitrary and requires more patients to yield the same statistical information (precision or power)
- Errors are not reduced by categorization unless that's the only way to get a subject to answer the question (e.g., income[b])

[b] But note how the Census Bureau tries to maximize the information collected. They first ask for income in dollars. Subjects refusing to answer are asked to choose from among 10 or 20 categories. Those not checking a category are asked to choose from fewer categories.

## 3.5 Preprocessing

- In vast majority of situations it is best to analyze the rawest form of the data
- Pre-processing of data (e.g., normalization) is sometimes necessary when the data are high-dimensional
- Otherwise normalizing factors should be part of the final analysis
- A particularly bad practice in animal studies is to subtract or divide by measurements in a control group (or the experimental group at baseline), then to analyze the experimental group as if it is the only group. Many things go wrong:
  - The normalization assumes that there is no biologic variability or measurement error in the control animals' measurements
  - The data may have the property that it is inappropriate to either subtract or divide by other groups' measurements. Division, subtraction, and percent change are highly parametric assumption-laden bases for analysis.
  - A correlation between animals is induced by dividing by a random variable
- A symptom of the problem is a graph in which the experimental group starts off with values 0.0 or 1.0
- The only situation in which pre-analysis normalization is OK in small datasets is in pre-post design or certain crossover studies for which it is appropriate to subject baseline values from follow-up values

See also Section 4.3.1.

## 3.6 Random Variables

- A potential measurement $X$
- $X$ might mean a blood pressure that will be measured on a randomly chosen US resident
- Once the subject is chosen and the measurement is made, we have a sample value of this variable
- Statistics often uses $X$ to denote a potentially observed value from some population and $x$ for an already-observed value (i.e., a constant)

# Chapter 4

# Descriptive Statistics, Distributions, and Graphics

## 4.1

## Distributions

ABD1.4

The *distribution* of a random variable $X$ is a profile of its variability and other tendencies. Depending on the type of $X$, a distribution is characterized by the following.

- Binary variable: the probability of "yes" or "present" (for a population) or the proportion of same (for a sample).

- $k$-Category categorical (polytomous, multinomial) variable: the probability that a randomly chosen person in the population will be from category $i, i = 1, \ldots, k$. For a sample, use $k$ proportions or percents.

- Continuous variable: any of the following 4 sets of statistics

  - probability density: value of $x$ is on the $x$-axis, and the relative likelihood of observing a value "close" to $x$ is on the $y$-axis. For a sample this yields a histogram.

  - cumulative probability distribution: the $y$-axis contains the probability of observing $X \leq x$. This is a function that is always rising or staying flat, never

decreasing. For a sample it corresponds to a cumulative histogram[a]

  - all of the *quantiles* or *percentiles* of $X$
  - all of the *moments* of $X$ (mean, variance, skewness, kurtosis, . . . )
  - If the distribution is characterized by one of the above four sets of numbers, the other three sets can be derived from this set

- Ordinal Random Variables
  - Because there many be heavy ties, quantiles may not be good summary statistics
  - The mean may be useful if the spacings have reasonable quantitative meaning
  - The mean is especially useful for summarizing ordinal variables that are counts
  - When the number of categories is small, simple proportions may be helpful
  - With a higher number of categories, exceedance probabilities or the empirical cumulative distribution function are very useful

- Knowing the distribution we can make intelligent guesses about future observations from the same series, although unless the distribution really consists of a single point there is a lot of uncertainty in predicting an individual new patient's response. It is less difficult to predict the average response of a group of patients once the distribution is known.

- At the least, a distribution tells you what proportion of patients you would expect to see whose measurement falls in a given interval.

### 4.1.1 Continuous Distributions

```
x ← seq(-3, 35, length=150)
par(mfrow=c(1,2)); xl ← expression(x)    # Fig. 4.1:
plot(x, dt(x, 4, 6), type='l', xlab=xl, ylab='Probability Density Function')
plot(x, pt(x, 4, 6), type='l', xlab=xl, ylab='Cumulative Distribution Function')
```

[a] But this *empirical cumulative distribution function* can be drawn with no grouping of the data, unlike an ordinary histogram.

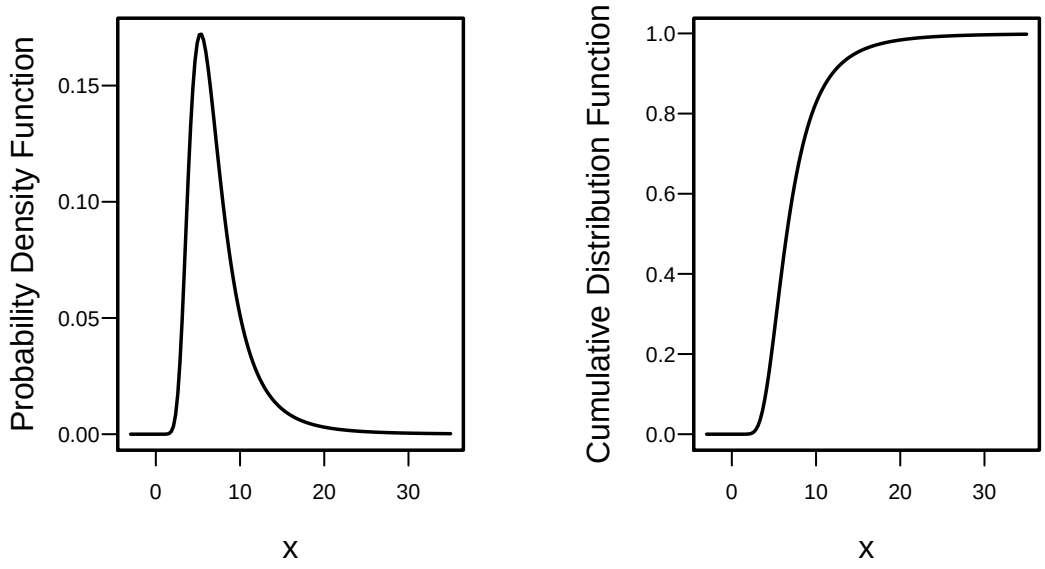



Figure 4.1: Example probability density (a) and cumulative probability distribution (b) for a positively skewed random variable (skewed to the right)

```
set.seed(1); x ← rnorm(1000)    # Fig. 4.2:
hist(x, nclass=40, prob=TRUE, col=gray(.9), xlab=xl, ylab='')
x ← seq(-4, 4, length=150)
lines(x, dnorm(x, 0, 1), col='blue', lwd=2)
```

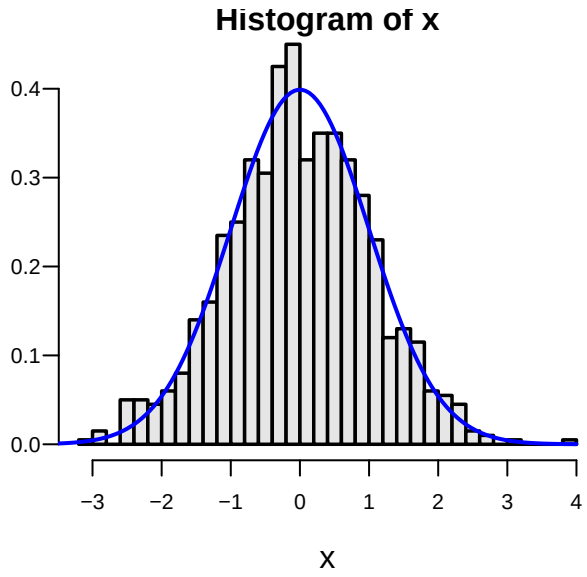



Figure 4.2: Example of a continuous distribution that is symmetric: the Gaussian (normal) distribution with mean 0 and variance 1, along with a histogram from a sample of size 1000 from this distribution

```
set.seed(2)
x ← c(rnorm(500, mean=0, sd=1), rnorm(500, mean=6, sd=3))
hist(x, nclass=40, prob=TRUE, col=gray(.9), xlab=xl, ylab='')
lines(density(x), col='blue', lwd=2)
abline(v=c(0, 6), col='red')    # Fig. 4.3
```

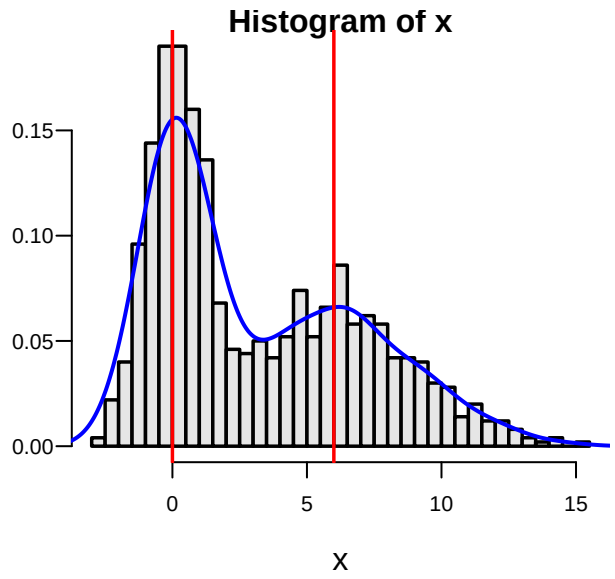



Figure 4.3: Example of a bimodal distribution from sampling from a mixture of normal distributions with different means and variances and estimating the underlying density function. Vertical red lines indicate true population means of the two component populations. Such a distribution can occur naturally or by failing to condition on a binary characteristic such as sex.

4.1.2

## Ordinal Variables

- Continuous ratio-scaled variables are ordinal
- Not all ordinal variables are continuous or ratio-scaled
- Best to analyze ordinal response variables using nonparametric tests or ordinal regression
- Heavy ties may be present
- Often better to treat count data as ordinal rather than to assume a distribution such as Poisson or negative binomial that is designed for counts
  - Poisson or negative binomial do not handle extreme clumping at zero
- Example ordinal variables are below

```
x ← 0:14
y ← c(.8, .04, .03, .02, rep(.01, 11))
plot(x, y, xlab=xl, ylab='', type='n')    # Fig. 4.4
segments(x, 0, x, y)
```

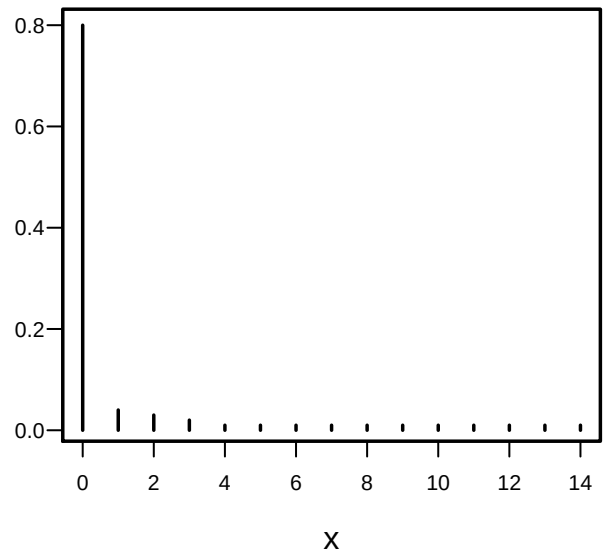



Figure 4.4: Distribution of number of days in the hospital in the year following diagnosis

```
x ← 1:10
y ← c(.1, .13, .18, .19, 0, 0, .14, .12, .08, .06)
plot(x, y, xlab=xl, ylab='', type='n')    # Fig. 4.5
segments(x, 0, x, y)
```

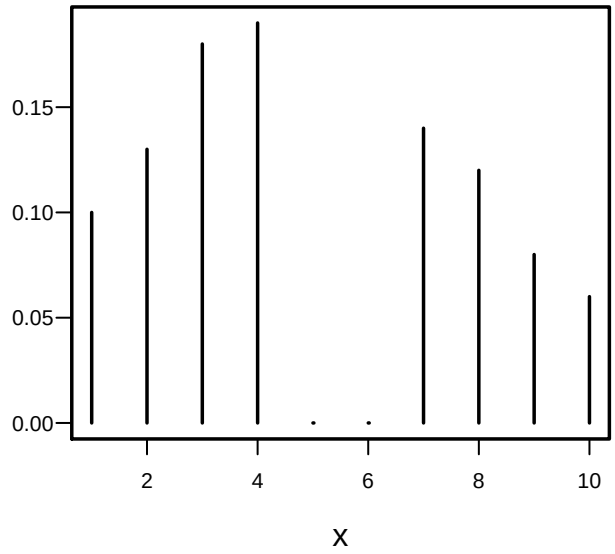



Figure 4.5: Distribution of a functional status score that does not have points in the middle

The `getHdata` function in the `Hmisc` package[4] finds, downloads, and `load()`s datasets from biostat.mc.vanderbilt.edu/DataSets.

```
require(Hmisc)
getHdata(nhgh)    # NHANES dataset     Fig. 4.6:
scr ← pmin(nhgh$SCr, 5)    # truncate at 5 for illustration
scr[scr == 5 | runif(nrow(nhgh)) < .05] ← 5   # pretend 1/20 dialyzed
hist(scr, nclass=50, xlab='Serum Creatinine', ylab='Density', prob=TRUE)
```

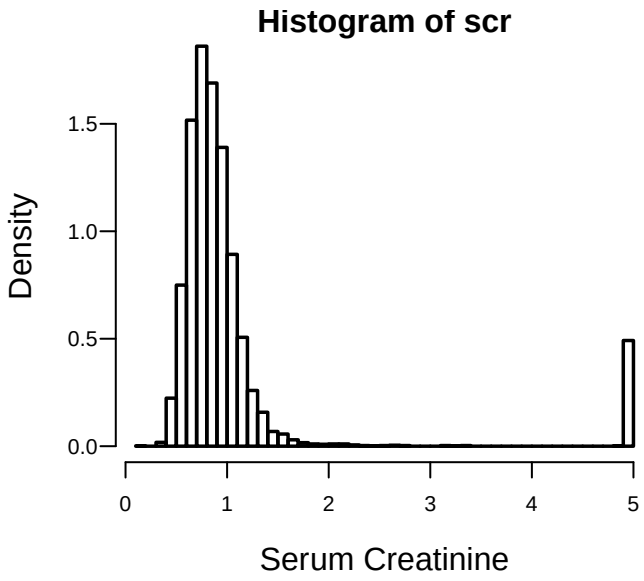



Figure 4.6: Distribution of serum creatinine where the patient requiring dialysis is taken to have the worst renal function. The variable is mostly continuous but is best analyzed as ordinal so that no assumption is made about how to score dialysis except for being worse than all non-dialysis patients. Data taken from NHANES where no patients were actually dialyzed.

# 4.2 Descriptive Statistics



## 4.2.1 Categorical Variables

- Proportions of observations in each category
  Note: The mean of a binary variable coded 1/0 is the proportion of ones.
- For variables representing counts (e.g., number of comorbidities), the mean is a good summary measure (but not the median)
- Modal (most frequent) category

## 4.2.2 Continuous Variables

Denote the sample values as $x_1, x_2, \ldots, x_n$

### Measures of Location

"Center" of a sample

- Mean: arithmetic average

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

  Population mean $\mu$ is the long-run average (let $n \to \infty$ in computing $\bar{x}$)

  – center of mass of the data (balancing point)

  – highly influenced by extreme values even if they are highly atypical

- Median: middle sorted value, i.e., value such that $\frac{1}{2}$ of the values are below it and above it
  - always descriptive
  - unaffected by extreme values
  - not a good measure of central tendency when there are heavy ties in the data
  - if there are heavy ties and the distribution is limited or well-behaved, the mean often performs better than the median (e.g., mean number of diseased fingers)
- Geometric mean: hard to interpret and effected by low outliers; better to use median

## Quantiles

Quantiles are general statistics that can be used to describe central tendency, spread, symmetry, heavy tailedness, and other quantities.

- Sample median: the 0.5 quantile or $50^{th}$ percentile
- Quartiles $Q_1, Q_2, Q_3$: 0.25 0.5 0.75 quantiles or $25^{th}, 50^{th}, 75^{th}$ percentiles
- Quintiles: by 0.2
- In general the $p$th sample quantile $x_p$ is the value such that a fraction $p$ of the observations fall below that value
- $p^{th}$ population quantile: value $x$ such that the probability that $X \leq x$ is $p$

## Spread or Variability

- Interquartile range: $Q_1$ to $Q_3$
  Interval containing $\frac{1}{2}$ of the subjects

Meaningful for any continuous distribution

- Other quantile intervals

- Variance (for symmetric distributions): averaged squared difference between a randomly chosen observation and the mean of all observations

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

The $-1$ is there to increase our estimate to compensate for our estimating the center of mass from the data instead of knowing the population mean.[b]

- Standard deviation: $s$ — $\sqrt{}$ of variance
  - $\sqrt{}$ of average squared difference of an observation from the mean
  - can be defined in terms of proportion of sample population within $\pm$ 1 SD of the mean **if the population is normal**

- SD and variance are not useful for very asymmetric data, e.g. "the mean hospital cost was \$10000 $\pm$ \$15000"

- range: not recommended because range $\uparrow$ as $n \uparrow$ and is dominated by a single outlier

- coefficient of variation: not recommended (depends too much on how close the mean is to zero)

[b] $\bar{x}$ is the value of $\mu$ such that the sum of squared values about $\mu$ is a minimum.

## 4.3 Graphics

ABD2.3.4

Cleveland[2,1] is the best source of how-to information on making scientific graphs. Much information may be found at http://biostat.mc.vanderbilt.edu/StatGraphCourse, especially these notes: http://goo.gl/DHE0a2. Murrell[6] has an excellent summary of recommendations:

- Display data values using position or length.
- Use horizontal lengths in preference to vertical lengths.
- Watch your data–ink ratio.
- Think very carefully before using color to represent data values.
- Do not use areas to represent data values.
- *Please* do not use angles or slopes to represent data values.
- *Please, please* do not use volumes to represent data values.

R has superior graphics implemented in multiple models, including

- Base graphics such as `plot()`, `hist()`, `lines()`, `points()` which give the user maximum control and are best used when not stratifying by additional variables other than the ones being summarized
- The `lattice` package which is fast but not quite as good as `ggplot2` when one needs to vary more than one of color, symbol, size, or line type due to having more than one categorizing variable
- The `ggplot2` package which is very flexible and has the nicest defaults especially for constructing keys (legends/guides)

For `ggplot2`, http://www.cookbook-r.com/Graphs contains a nice cookbook. See also http://learnr.wordpress.com. To get excellent documentation with examples for any `ggplot2` function, google `ggplot2` *`functionname`*.

### 4.3.1

## Graphing Change vs. Raw Data

A common mistake in scientific graphics is to cover up subject variability by normalizing repeated measures for baseline (see Section 3.5). Among other problems, this prevents the reader from seeing regression to the mean for subjects starting out at very low or very high values, and from seeing variation in intervention effect as a function of baseline values. It is highly recommended that all the raw data be shown, including those from time zero. When the sample size is not huge, spaghetti plots are most effective for graphing longitudinal data because all points from the same subject over time are connected. An example [3, pp. 161-163] is below.

```
require(Hmisc)    # also loads ggplot2
getHdata(cdystonia)
ggplot(cdystonia, aes(x=week, y=twstrs, color=factor(id))) +
       geom_line() + xlab('Week') + ylab('TWSTRS-total score') +
       facet_grid(treat ~ site) +
       guides(color=FALSE) # Fig. 4.7
```

Graphing the raw data is usually essential.

### 4.3.2

## Categorical Variables

- pie chart
  - high ink:information ratio
  - optical illusions (perceived area or angle depends on orientation vs. horizon)
  - hard to label categories when many in number
- bar chart
  - high ink:information ratio
  - hard to depict confidence intervals (one sided error bars?)
  - hard to interpret if use subcategories
  - labels hard to read if bars are vertical

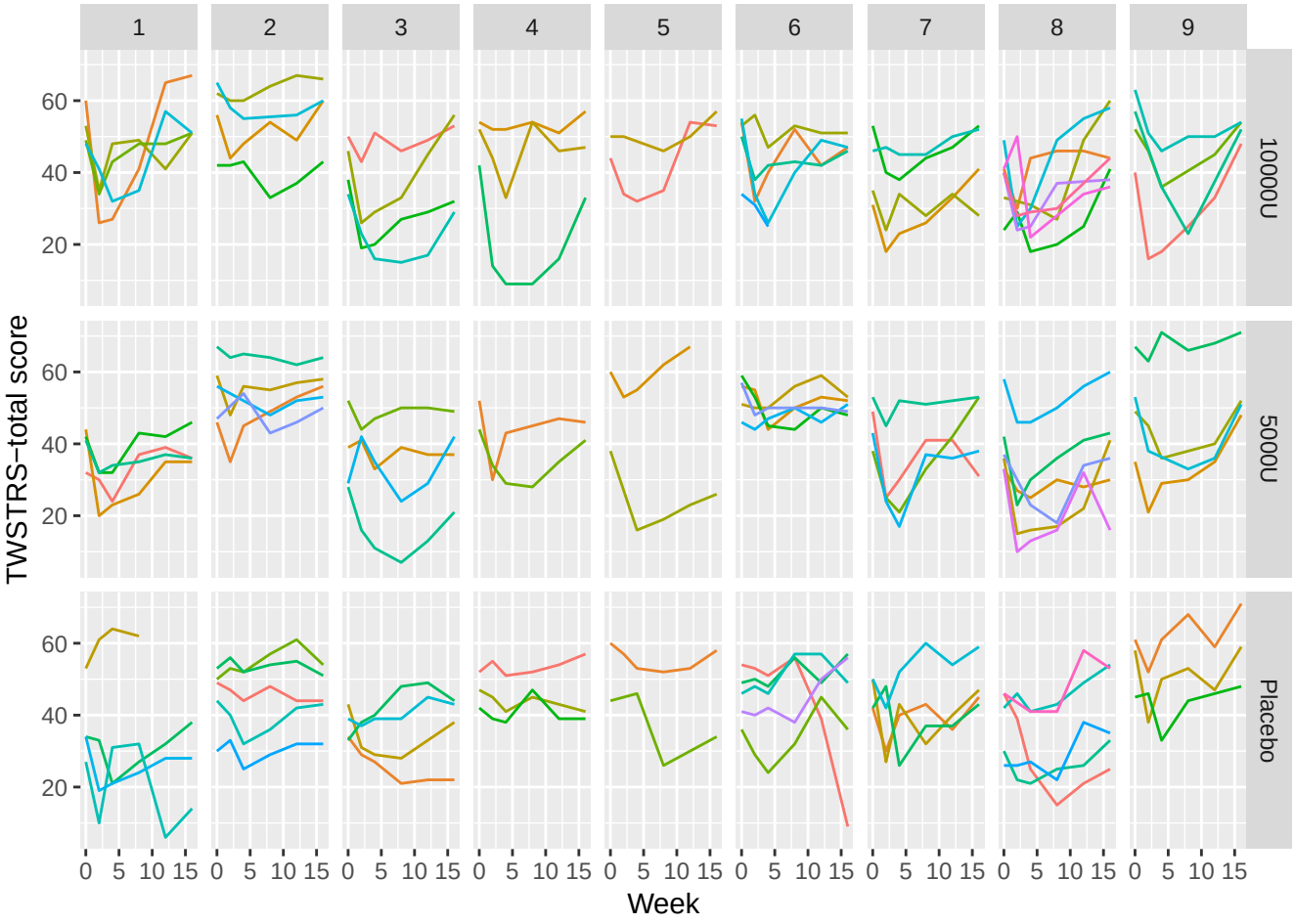



Figure 4.7: Spaghetti plot showing all the raw data on the response variable for each subject, stratified by dose and study site (1–9). Importantly, week 0 (baseline) measurements are included.

- dot chart
  - leads to accurate perception
  - easy to show all labels; no caption needed
  - allows for multiple levels of categorization (see Figures 4.8 and 4.9)

```
getHdata(titanic3)
d ← upData(titanic3,
           agec     = cut2(age, c(10, 15, 20, 30)), print=FALSE)
d ← with(d, as.data.frame(table(sex, pclass, agec)))
d ← subset(d, Freq > 0)
ggplot(d, aes(x=Freq, y=agec)) + geom_point() +
  facet_grid(sex ~ pclass) +
  xlab('Frequency') + ylab('Age')
```

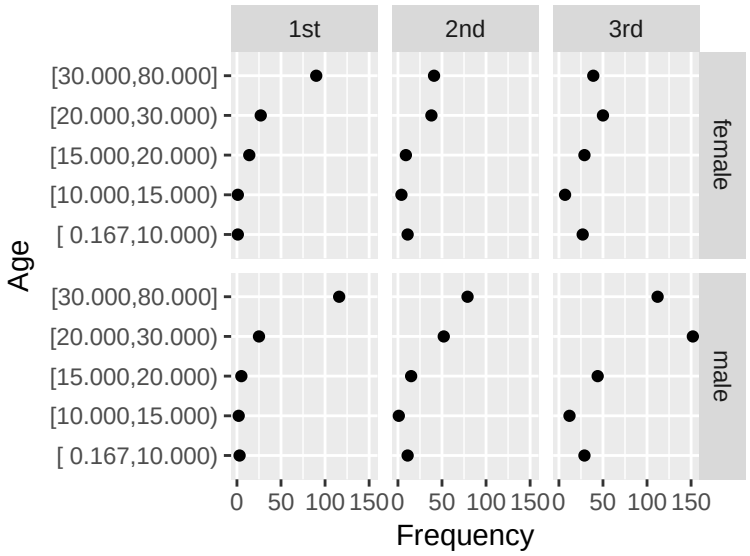



Figure 4.8: Dot chart showing frequencies from cross-classifications of discrete variables for Titanic passengers

    - multi-panel display for multiple major categorizations
    - lines of dots arranged vertically within panel
    - categories within a single line of dots
  - easy to show 2-sided error bars

- Avoid chartjunk such as dummy dimensions in bar charts, rotated pie charts, use of solid areas when a line suffices

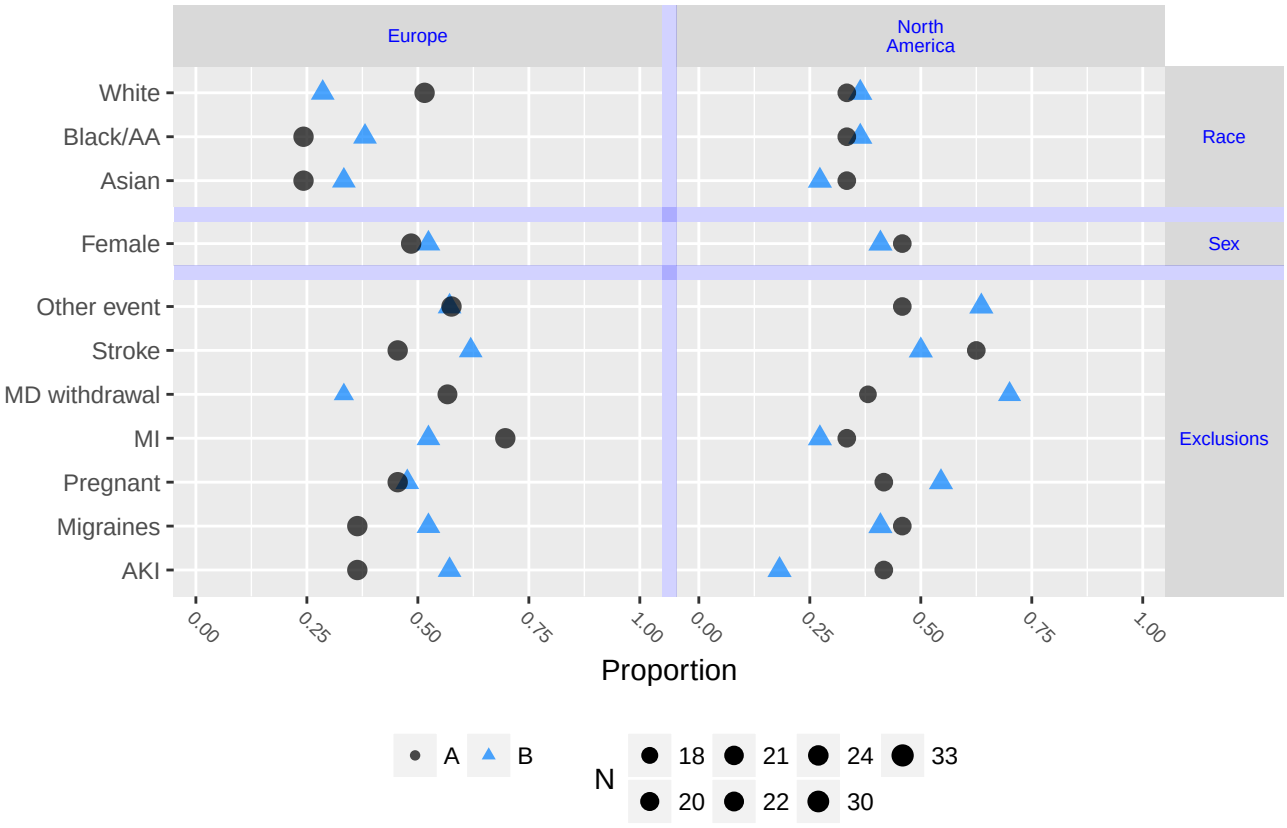



Figure 4.9: Dot chart for categorical demographic variables, stratified by treatment and region

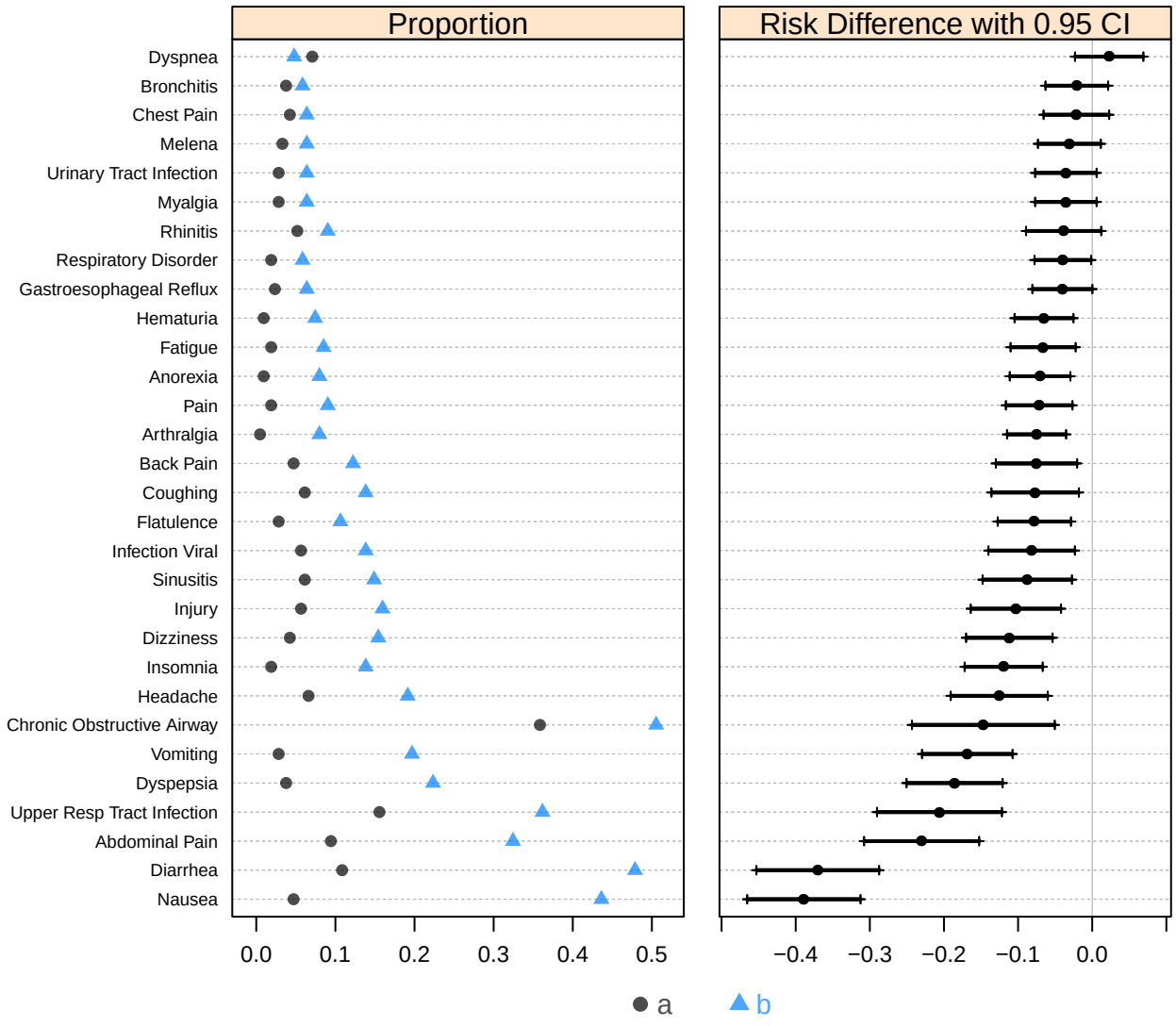



Figure 4.10: Dot chart showing proportion of subjects having adverse events by treatment, sorted by risk difference, produced by the R `greport` package. See `test.Rnw` at biostat.mc.vanderbilt.edu/Greport

4.3.3

# Continuous Variables

## Raw Data

For graphing two continuous variable, scatterplots are often essential. The following example draws a scatterplot on a very large number of observations in a measurement comparison study where the goal is to measure esophageal pH longitudinally and across subjects.

```
getHdata(esopH)
contents(esopH)
```

```
Data frame:esopH        136127 observations and 2 variables    Maximum # NAs:0


                                       Labels   Class Storage
orophar Esophageal pH by Oropharyngeal Device numeric  double
conv     Esophageal pH by Conventional Device numeric  double
```

```
xl ← label(esopH$conv)
yl ← label(esopH$orophar)
ggplot(esopH, aes(x=conv, y=orophar)) + geom_point(pch='.') +
  xlab(xl) + ylab(yl) +    # Fig. 4.11
  geom_abline(intercept = 0, slope = 1)
```

With large sample sizes there are many collisions of data points and hexagonal binning is an effective substitute for the raw data scatterplot. The number of points represented by one hexagonal symbol is stratified into 20 groups of approximately equal numbers of points. The code below is not currently working for the `ggplot2` package version 2.1.0.

```
ggplot(esopH, aes(x=conv, y=orophar)) +
  stat_binhex(aes(alpha=..count.., color=Hmisc::cut2(..count.., g=20)),
              bins=80) +
  xlab(xl) + ylab(yl) +
  guides(alpha=FALSE, fill=FALSE, color=guide_legend(title='Frequency'))
```

Instead we use the `Hmisc ggfreqScatter` function to bin the points and represent frequencies of overlapping points with color and transparency levels.

```
with(esopH, ggfreqScatter(conv, orophar, bins=50, g=20) +
              geom_abline(intercept=0, slope=1))    # Fig. 4.12
```

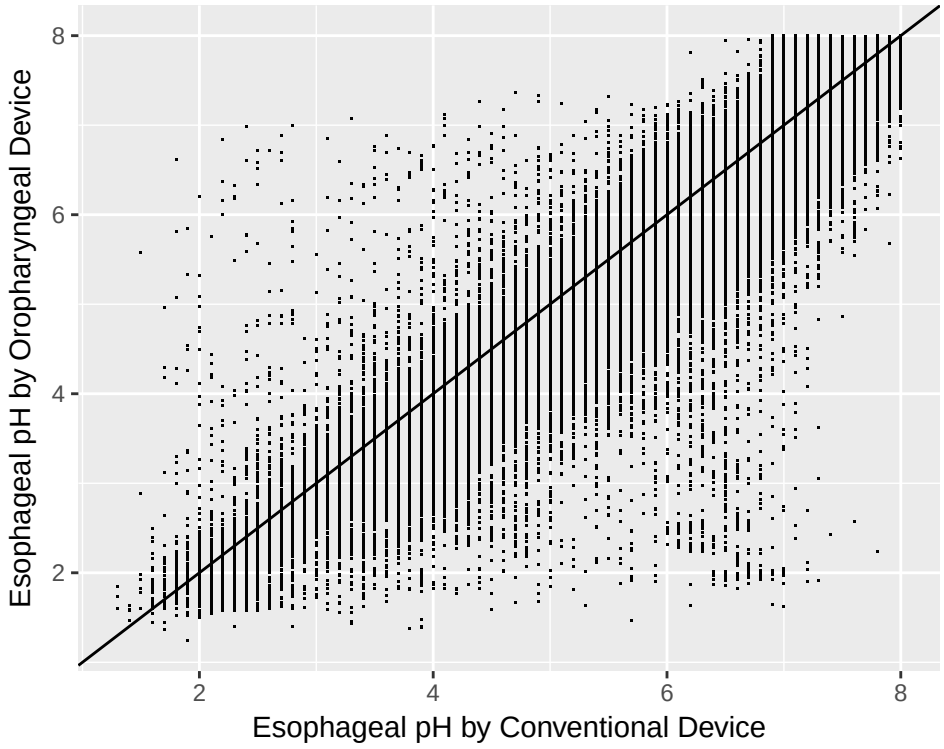



Figure 4.11: Scatterplot of one measurement mode against another

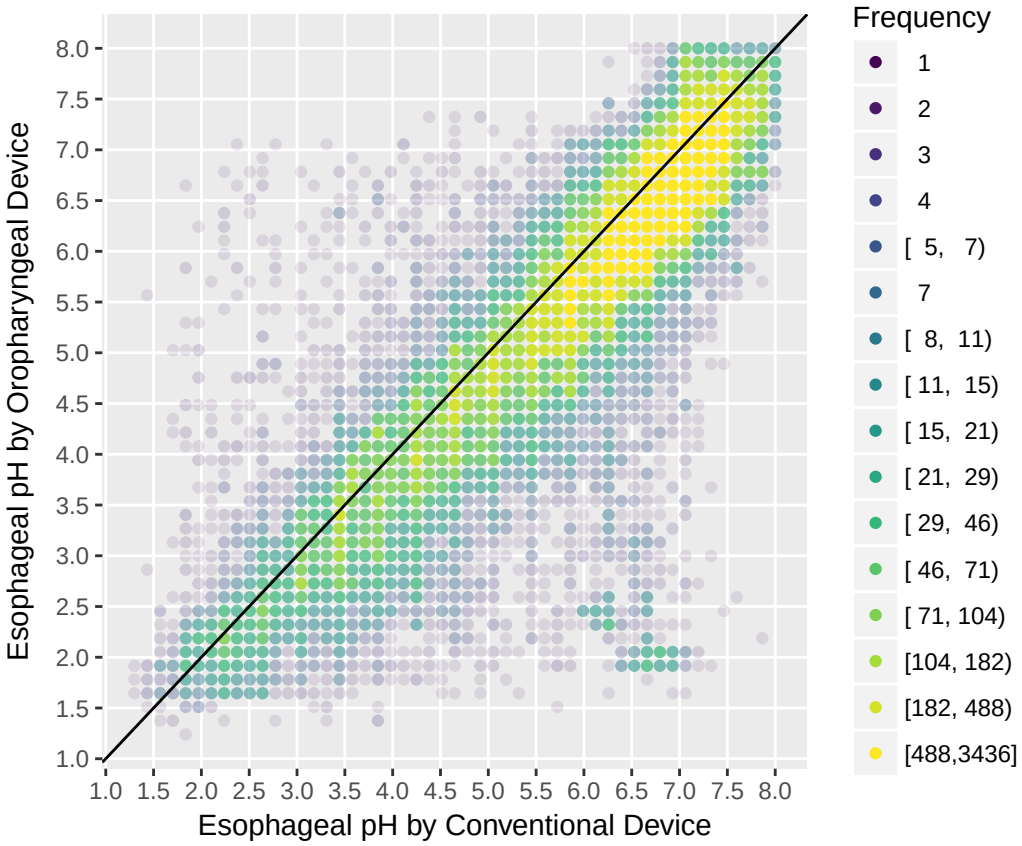



Figure 4.12: Binned points (2500 total bins) with frequency counts shown as color and transparency level

# Distributions



- histogram showing relative frequencies
  - requires arbitrary binning of data
  - not optimal for comparing multiple distributions
- cumulative distribution function: proportion of values $\leq x$ vs. $x$ (Figure 4.13)
  Can read all quantiles directly off graph.

```
getHdata(pbc)
pbcr ← subset(pbc, drug != 'not randomized')
Ecdf(pbcr[,c('bili','albumin','protime','sgot')],   # Fig. 4.13
     group=pbcr$drug, col=1:2,
     label.curves=list(keys='lines'))
```

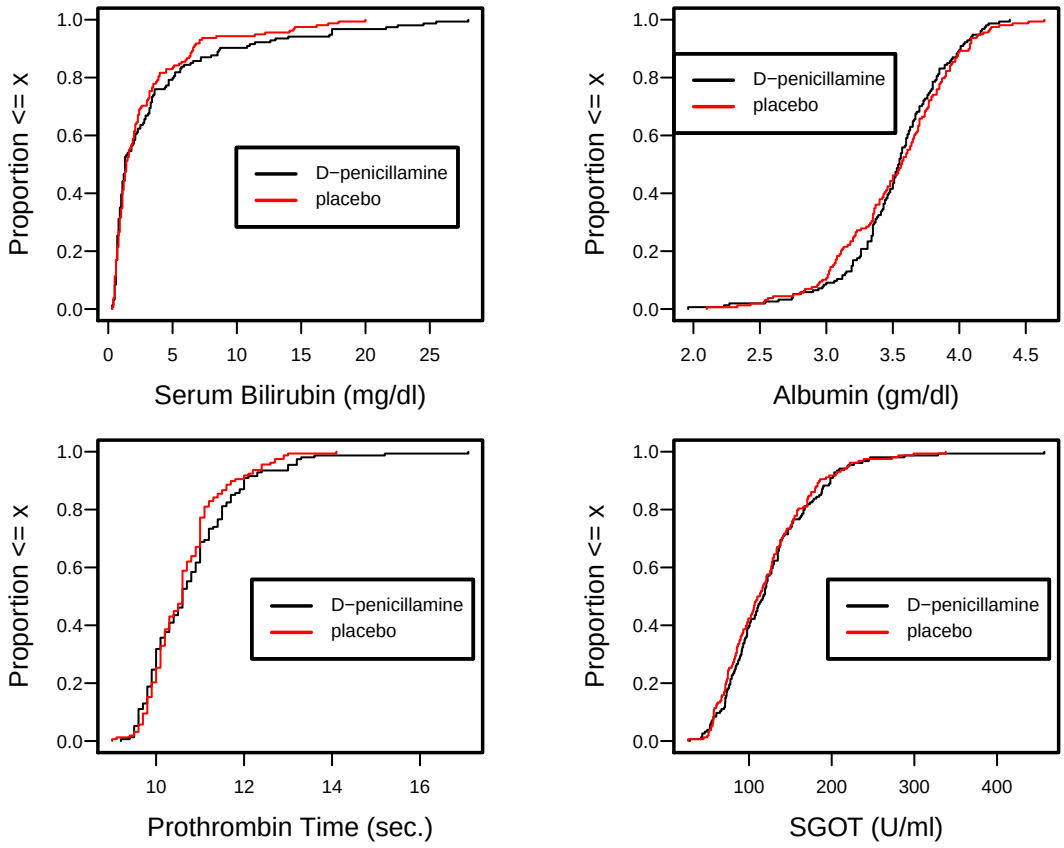



Figure 4.13: Empirical cumulative distributions of baseline variables stratified by treatment in a randomized controlled trial. $m$ is the number of missing values.

- Box plots shows quartiles plus the mean. They are a good way to compare many groups as seen in Figures 4.14 and 4.16.

```
getHdata(support)    # Fig. 4.14
bwplot(dzgroup ~ crea, data=support, panel=panel.bpplot,
       probs=c(.05,.25), xlim=c(0,8), xlab='Serum Creatinine')
```

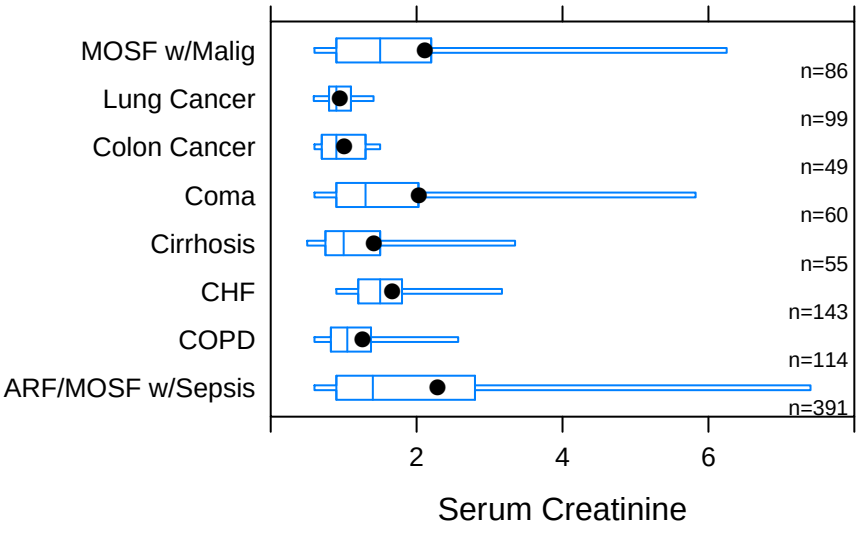



Figure 4.14: Box plots showing the distribution of serum creatinine stratified by major diagnosis. Dot: mean; vertical line: median; large box:interquartile range. The 0.05 and 0.95 quantiles are also shown, which is not the way typical box plots are drawn but is perhaps more useful. Asymmetry of distributions can be seen by both disagreement between $Q_3 - Q_2$ and $Q_2 - Q_1$ and by disagreement between $Q_2$ and $\bar{x}$.

Figure 4.16 uses extended box plots. The following schematic shows how to interpret them.

```
bpplt()    # Fig. 4.15
```

```
require(lattice)    # Fig. 4.16:
getHdata(diabetes)
wst ← cut2(diabetes$waist, g=2)
levels(wst) ← paste('Waist', levels(wst))
bwplot(cut2(age,g=4) ~ glyhb | wst*gender, data=diabetes,
       panel=panel.bpplot, xlab='Glycosylated Hemoglobin', ylab='Age
            Quartile')
```

Box plots are inadequate for displaying bimodality. Violin plots show the entire distribution well if the variable being summarized is fairly continuous.

## Relationships

- When response variable is continuous and descriptor (stratification) variables are categorical, multi-panel dot charts, box plots, multiple cumulative distributions,

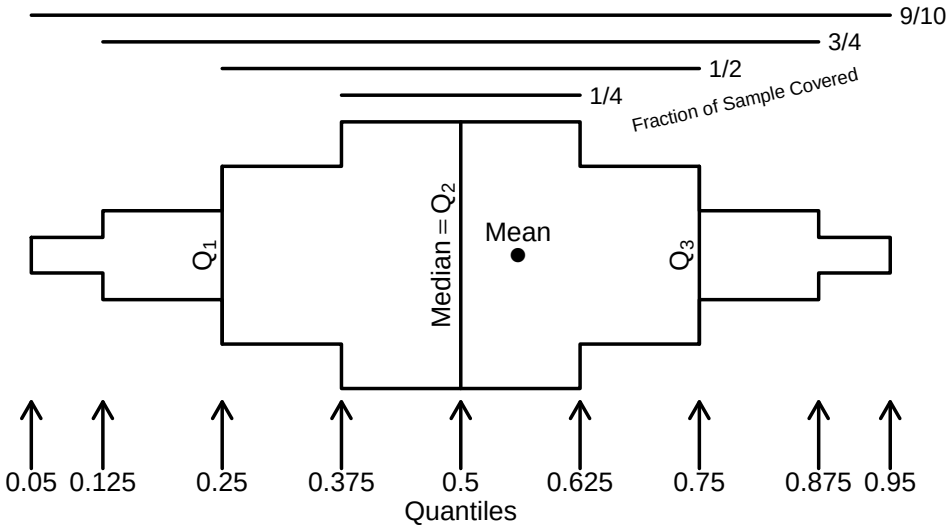



Figure 4.15: Schematic for extended box plot

etc., are useful.

- Two continuous variables: scatterplot

### 4.3.4 Graphs for Summarizing Results of Studies

- Dot charts with optional error bars (for confidence limits) can display any summary statistic (proportion, mean, median, mean difference, etc.)

- It is not well known that the confidence interval for a difference in two means cannot be derived from individual confidence limits.[c]
  Show individual confidence limits as well as actual CLs for the difference.

```
attach(diabetes)
set.seed(1)
male   ← smean.cl.boot(glyhb[gender=='male'],   reps=TRUE)
female ← smean.cl.boot(glyhb[gender=='female'], reps=TRUE)
dif ← c(mean=male['Mean']-female['Mean'],
        quantile(attr(male, 'reps')-attr(female,'reps'), c(.025,.975)))
plot(0,0,xlab='Glycated Hemoglobin',ylab='',   # Fig. 4.18
     xlim=c(5,6.5),ylim=c(0,4), axes=F)
```

[c] In addition, it is not necessary for two confidence intervals to be separated for the difference in means to be significantly different from zero.

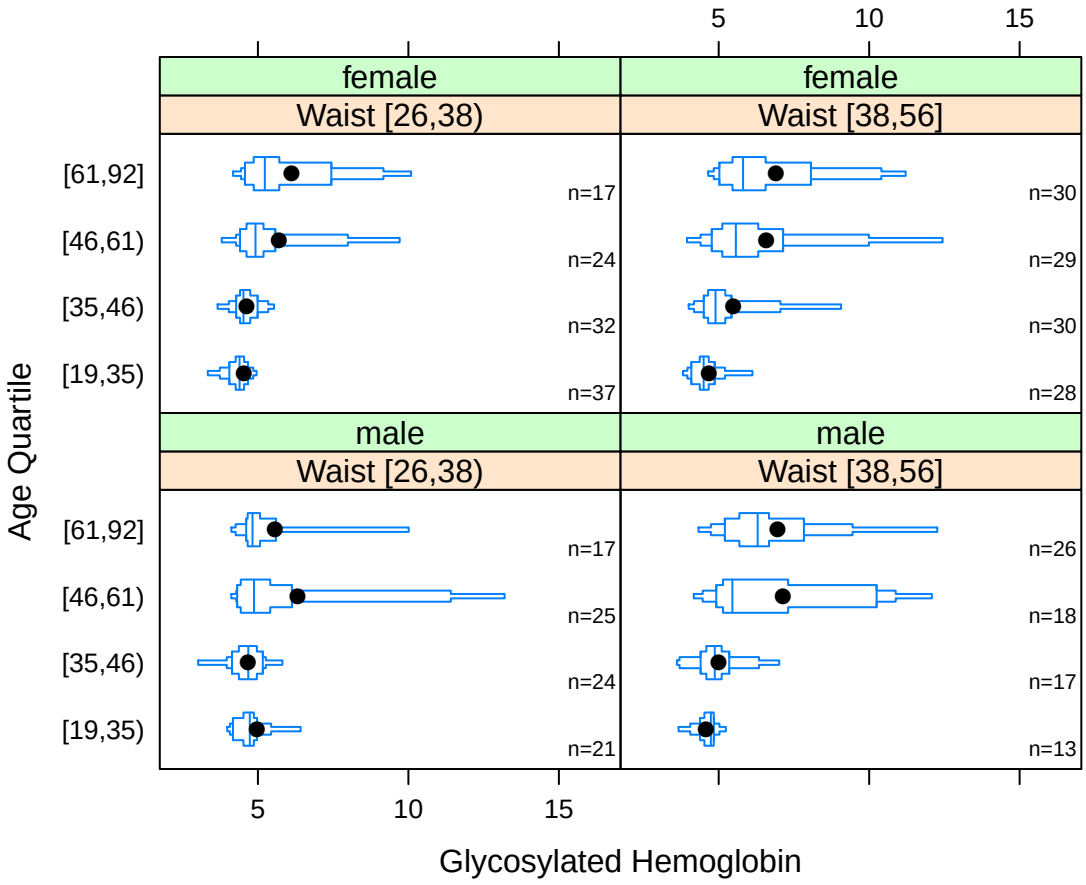



Figure 4.16: Extended box plots for glycohemoglobin stratified by quartiles of age (vertical), two-tiles of waist circumference (horizontal), and sex (vertical)

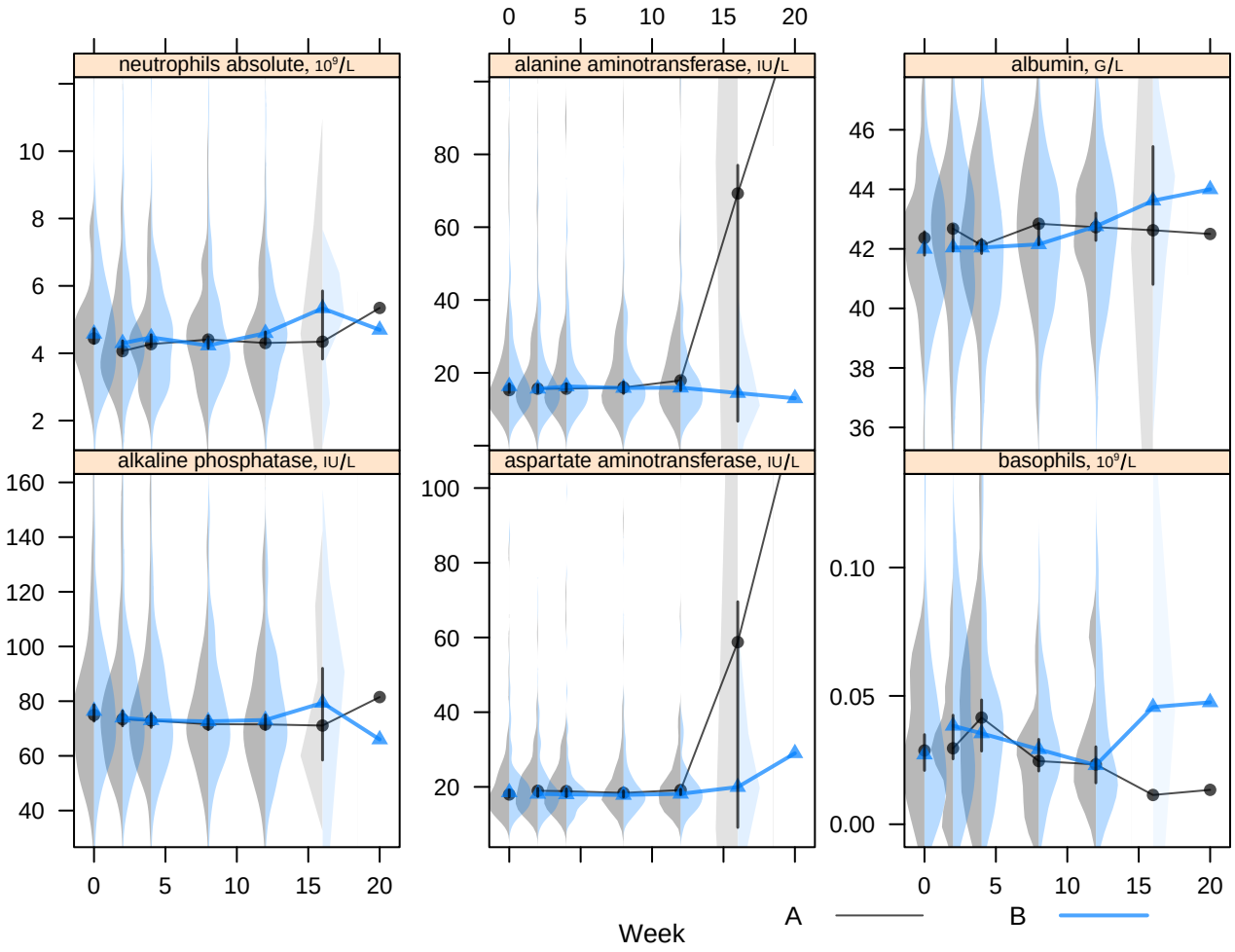



Figure 4.17: One-half violin plots for longitudinal data, stratified by treatment. Density estimates for groups with insufficient sample sizes are faded. Density plots are back–to–back for treatment A and B. Points are treatment medians. When the black vertical line does not touch the two medians, the medians are significantly different at the $\alpha = 0.05$ level. Graphic was produced by the R `grepport` package.

```
axis(1, at=seq(5, 6.5, by=0.25))
axis(2, at=c(1,2,3.5), labels=c('Female','Male','Difference'),
     las=1, adj=1, lwd=0)
points(c(male[1],female[1]), 2:1)
segments(female[2], 1, female[3], 1)
segments(male[2], 2,   male[3], 2)
offset ← mean(c(male[1],female[1])) - dif[1]
points(dif[1] + offset, 3.5)
segments(dif[2]+offset, 3.5, dif[3]+offset, 3.5)
at ← c(-.5,-.25,0,.25,.5,.75,1)
axis(3, at=at+offset, label=format(at))
segments(offset, 3, offset, 4.25, col=gray(.85))
abline(h=c(2 + 3.5)/2, col=gray(.85))
```

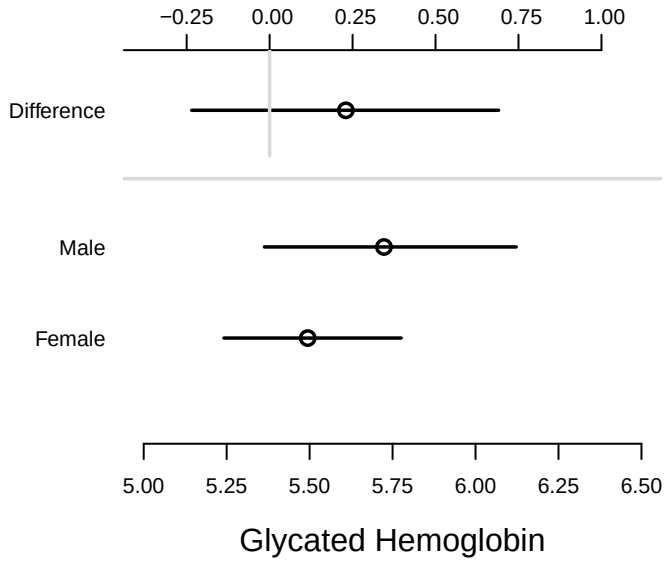



Figure 4.18: Means and nonparametric bootstrap 0.95 confidence limits for glycated hemoglobin for males and females, and confidence limits for males - females. Lower and upper $x$-axis scales have same spacings but different centers. Confidence intervals for differences are generally wider than those for the individual constituent variables.

- For showing relationship between two continuous variables, a trend line or regression model fit, with confidence bands

### 4.3.5 Graphs for Describing Statistical Model Fits

Several types of graphics are useful. These are all implemented in the R `rms` package[5].

**Partial effect plots** : Show the effect on $Y$ of varying one predictor at a time, holding the other predictors to medians or modes, and include confidence bands. This is the best approach for showing shapes of effects of continuous predictors.

**Effect charts** : Show the difference in means, odds ratio, hazard ratio, fold change, etc., varying each predictor and holding others to medians or modes[d]. For continuous variables that do not operate linearly, this kind of display is not very satisfactory because of its strong dependence on the settings over which the predictor is set. By default inter-quartile-range effects are used.

**Nomograms** : Shows the entire model if the number of interactions is limited. Nomograms show strengths and shapes of relationships, are very effective for continuous predictors, and allow computation of predicted values (although without confidence limits).

Here are examples using NHANES data to predict glycohemoglobin from age, sex, race/ethnicity, and BMI.

**Note**: ordinary regression is not an adequate fit for glycohemoglobin; an excellent fit comes from ordinal regression. BMI is not an adequate summary of body size. The following ordinary regression model in the $-1.75$ power of glycohemoglobin resulted in approximately normal residuals and is used for illustration. The transformation is subtracted from a constant just so that positive regression coefficients indicate that increasing a predictor increases glycohemoglobin. The inverse transformation raises predicted values to the $-\frac{1}{1.75}$ power after accounting for the subtraction, and is used to estimate the median glycohemoglobin on the original scale[e]. Restricted cubic spline functions with 4 default knots are used to allow age and BMI to act smoothly but nonlinearly. Partial effects plots are in Fig. 4.19.

```
require(rms)
```

```
getHdata(nhgh)    # NHANES data
dd ← datadist(nhgh); options(datadist='dd')
g         ← function(x) 0.09 - x ^ - (1 / 1.75)
ginverse ← function(y) (0.09 - y) ^ -1.75
f ← ols(g(gh) ~ rcs(age, 4) + re + sex + rcs(bmi, 4), data=nhgh)
cat('{\\small\n')
```

```
f
```

**Linear Regression Model**

```
ols(formula = g(gh) ~ rcs(age, 4) + re + sex + rcs(bmi, 4), data = nhgh)
```

| | Model Likelihood Ratio Test | | Discrimination Indexes | |
|---|---|---|---|---|
| Obs 6795 | LR $\chi^2$ | 1861.16 | $R^2$ | 0.240 |
| $\sigma$ 0.0235 | d.f. | 11 | $R^2_{\text{adj}}$ | 0.238 |
| d.f. 6783 | Pr($>\chi^2$) | 0.0000 | $g$ | 0.015 |

[d] It does not matter what the other variables are set to if they do not interact with the variable being varied.

[e] If residuals have a normal distribution after transforming the dependent variable, the estimated mean and median transformed values are the same. Inverse transforming the estimates provides an estimate of the median on the original scale (but not the mean).

Residuals

| Min | 1Q | Median | 3Q | Max |
|---|---|---|---|---|
| −0.09736 | −0.01208 | −0.002201 | 0.008237 | 0.1689 |

| | $\hat{\beta}$ | S.E. | $t$ | Pr($>\|t\|$) |
|---|---|---|---|---|
| Intercept | -0.2884 | 0.0048 | -60.45 | <0.0001 |
| age | 0.0002 | 0.0001 | 3.34 | 0.0008 |
| age' | 0.0010 | 0.0001 | 7.63 | <0.0001 |
| age'' | -0.0040 | 0.0005 | -8.33 | <0.0001 |
| re=Other Hispanic | -0.0013 | 0.0011 | -1.20 | 0.2318 |
| re=Non-Hispanic White | -0.0082 | 0.0008 | -10.55 | <0.0001 |
| re=Non-Hispanic Black | -0.0013 | 0.0009 | -1.34 | 0.1797 |
| re=Other Race Including Multi-Racial | 0.0006 | 0.0014 | 0.47 | 0.6411 |
| sex=female | -0.0022 | 0.0006 | -3.90 | <0.0001 |
| bmi | -0.0006 | 0.0002 | -2.54 | 0.0111 |
| bmi' | 0.0059 | 0.0009 | 6.44 | <0.0001 |
| bmi'' | -0.0161 | 0.0025 | -6.40 | <0.0001 |

```
print(anova(f), dec.ss=3, dec.ms=3)
```

**Analysis of Variance for** `g(gh)`

| | d.f. | Partial SS | MS | $F$ | $P$ |
|---|---|---|---|---|---|
| age | 3 | 0.732 | 0.244 | 441.36 | <0.0001 |
| *Nonlinear* | 2 | 0.040 | 0.020 | 35.83 | <0.0001 |
| re | 4 | 0.096 | 0.024 | 43.22 | <0.0001 |
| sex | 1 | 0.008 | 0.008 | 15.17 | <0.0001 |
| bmi | 3 | 0.184 | 0.061 | 110.79 | <0.0001 |
| *Nonlinear* | 2 | 0.023 | 0.011 | 20.75 | <0.0001 |
| TOTAL NONLINEAR | 4 | 0.068 | 0.017 | 30.94 | <0.0001 |
| **REGRESSION** | 11 | 1.181 | 0.107 | 194.29 | <0.0001 |
| **ERROR** | 6783 | 3.749 | 0.001 | | |

```
cat('}\n')
```

```
# Show partial effects of all variables in the model, on the original scale
ggplot(Predict(f, fun=ginverse),    # Fig. 4.19
       ylab=expression(paste('Predicted Median ', HbA['1c'])))
```

An effect chart is in Fig. 4.20 and a nomogram is in Fig. 4.21. See http://stats.stackexchange.com/questions/155430/clarifications-regarding-reading-a-nomogr for excellent examples showing how to read such nomograms.

```
plot(summary(f))    # Fig. 4.20
```

```
plot(nomogram(f, fun=ginverse, funlabel='Median HbA1c'))  # Fig. 4.21
```

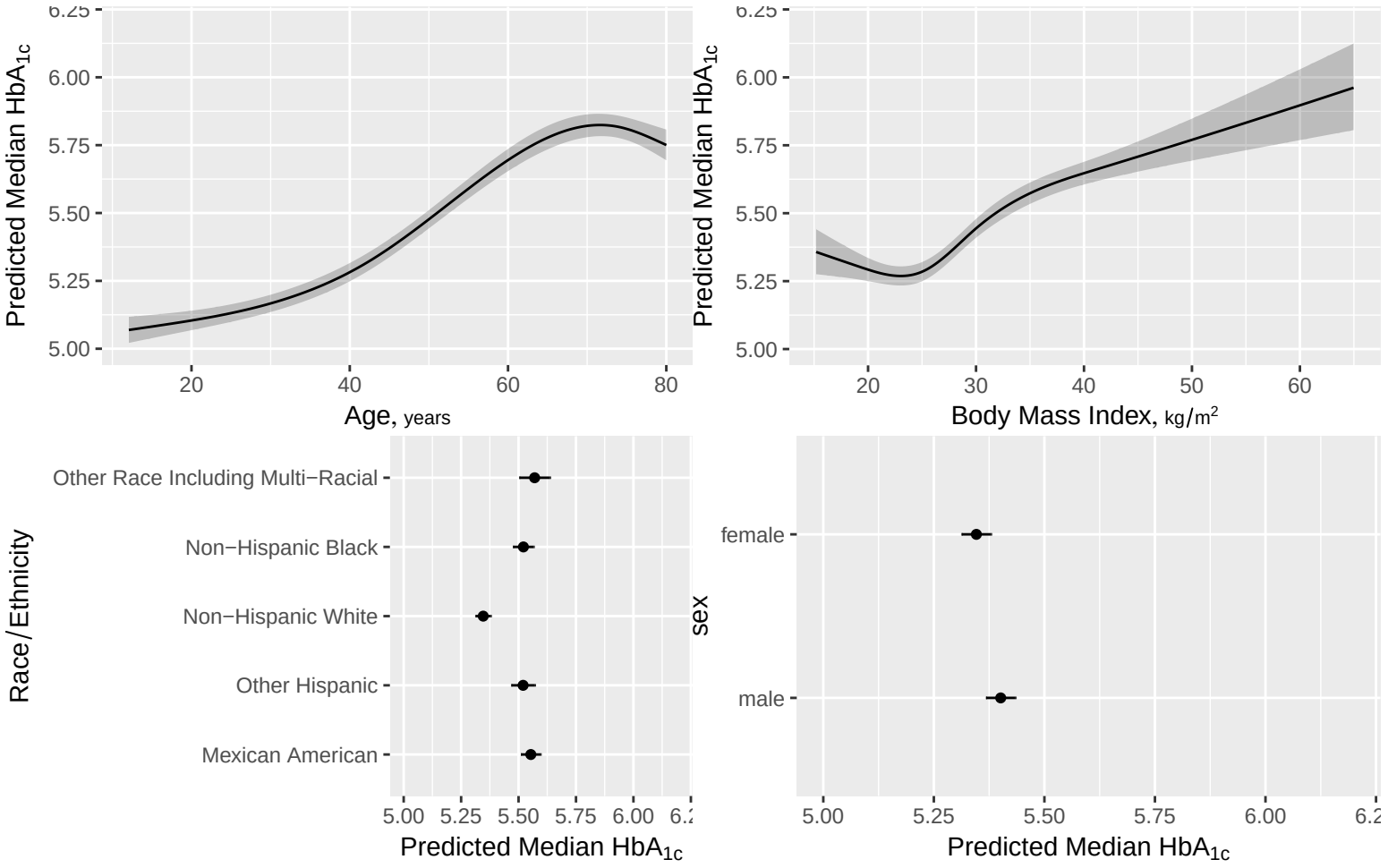



Figure 4.19: Partial effects in NHANES HbA1c model

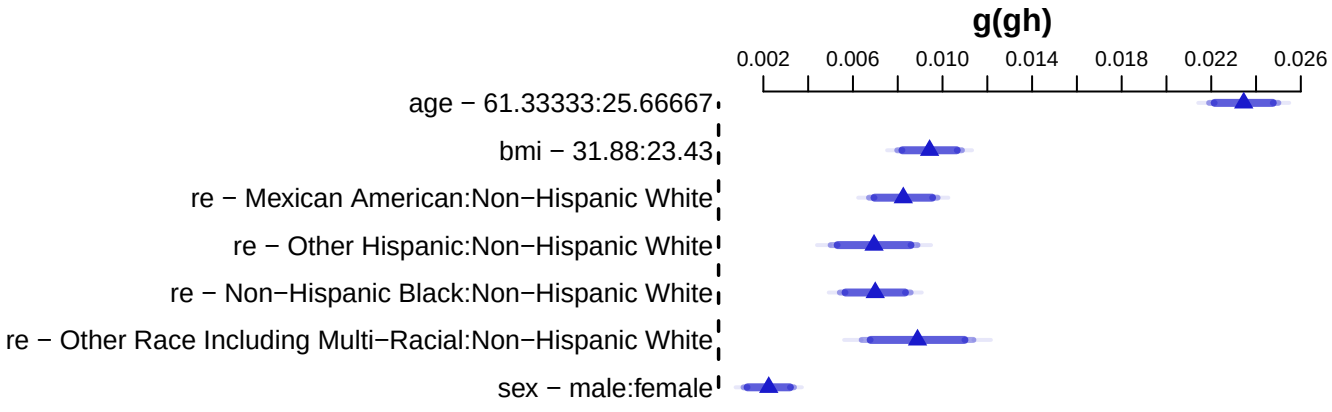



Figure 4.20: Partial effects chart on the transformed scale. For age and BMI, effects are inter-quartile-range effects. 0.9, 0.95, and 0.99 confidence limits are shown.

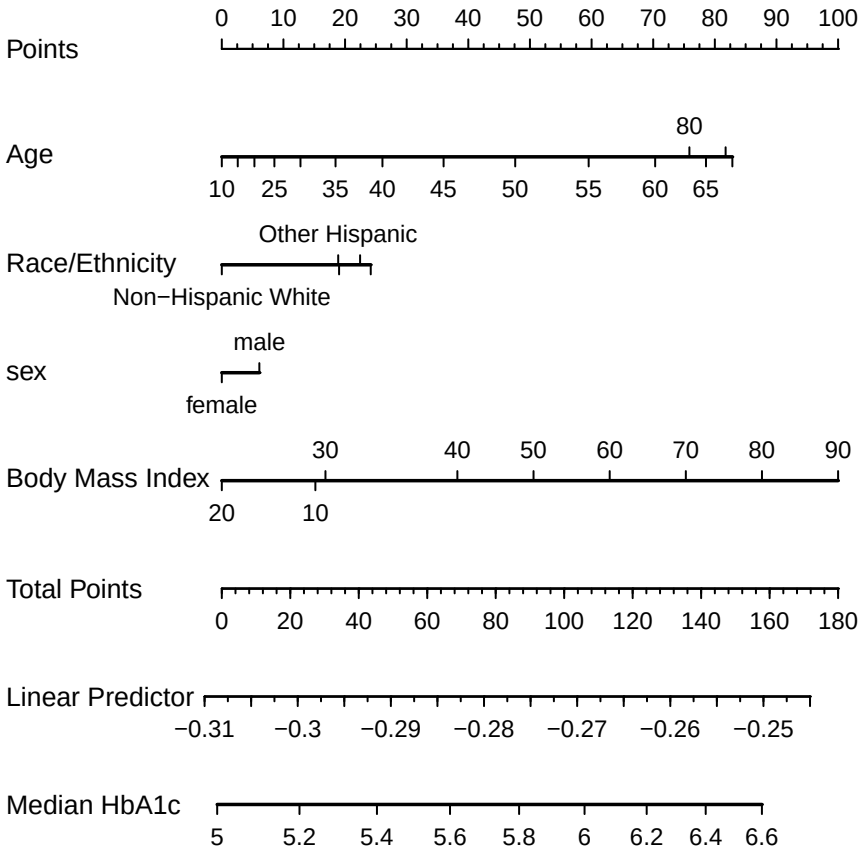



Figure 4.21: Nomogram for predicting median $HbA_{1c}$. To use the nomogram, use the top `Points` scale to convert each predictor value to a common scale. Add the points and read this number on the `Total Points` scale, then read down to the median.

## Graphing Effect of Two Continuous Variables on $Y$

The following examples show the estimated combined effects of two continuous predictors on outcome. The two models included interaction terms, the second example using penalized maximum likelihood estimation with a tensor spline in diastolic $\times$ systolic blood pressure.

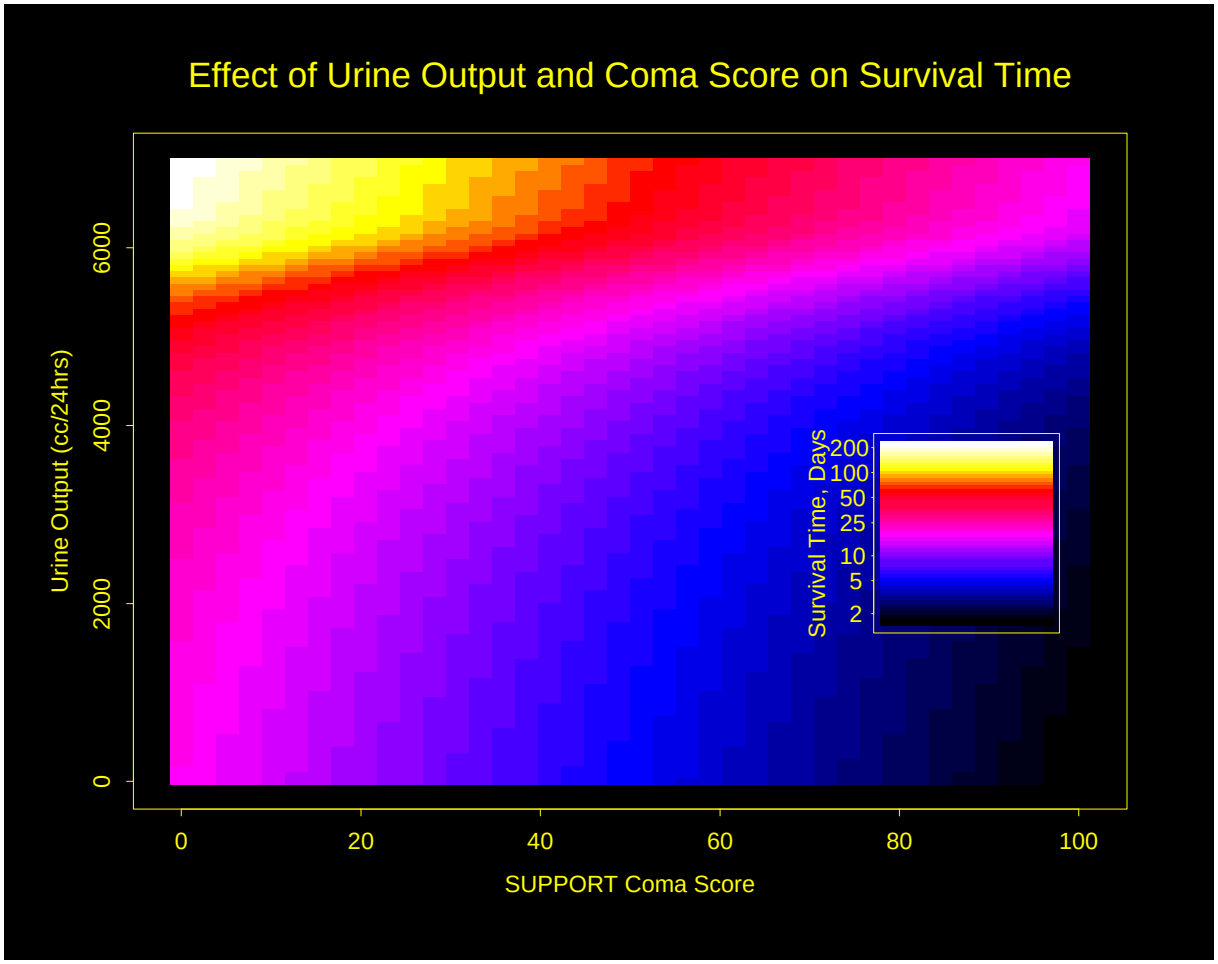



Figure 4.22: Estimated median survival time for critically ill adults

Figure 4.23 is particularly interesting because the literature had suggested (based on approximately 24 strokes) that pulse pressure was the main cause of hemorrhagic stroke whereas this flexible modeling approach (based on approximately 230 strokes) suggests that mean arterial blood pressure (roughly a $45^{\circ}$ line) is what is most important over a broad range of blood pressures. At the far right one can see that pulse pressure (axis perpendicular to $45^{\circ}$ line) may have an impact although a non-monotonic one.

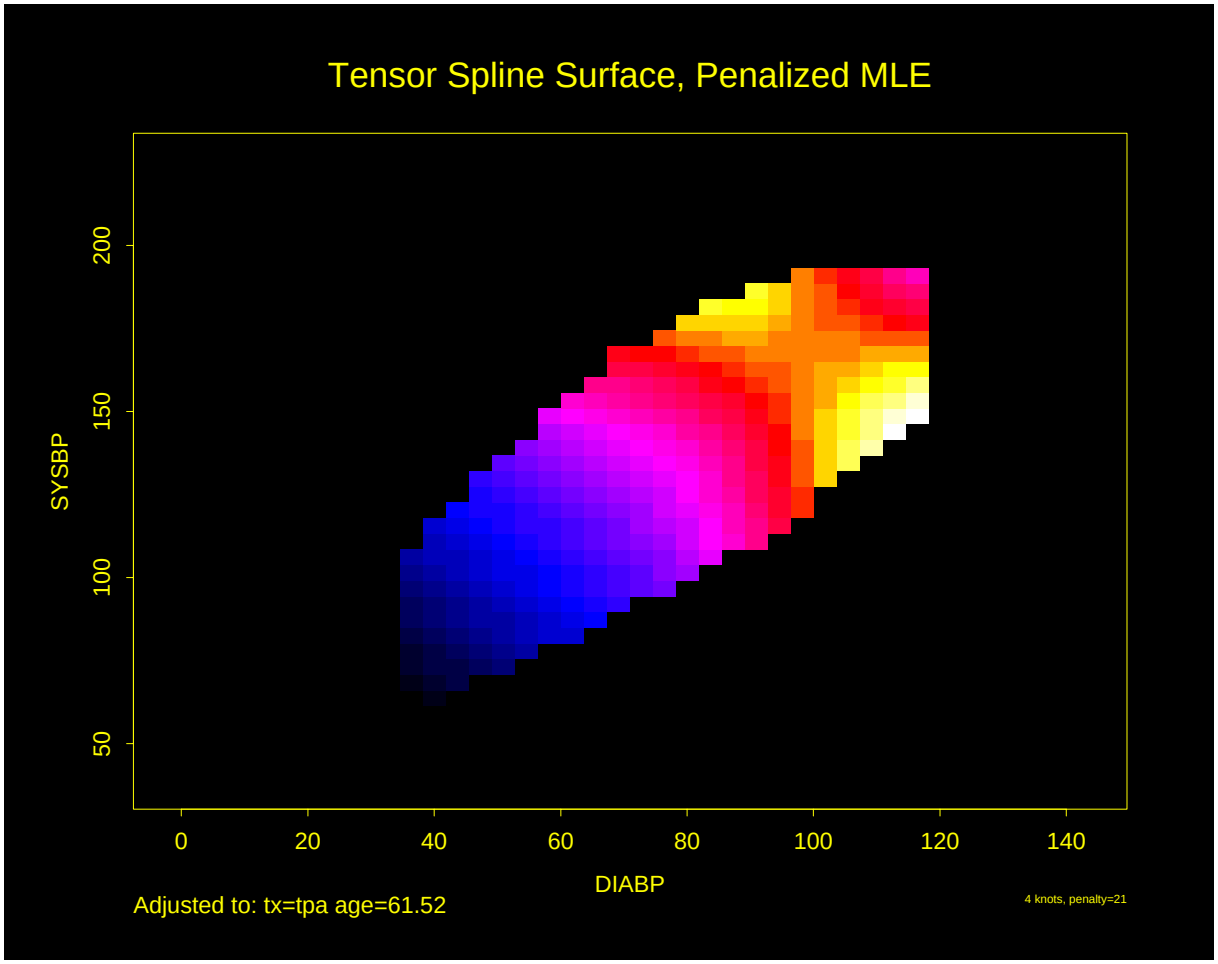



Figure 4.23: Logistic regression estimate of probability of a hemorrhagic stroke for patients in the GUSTO-I trial given $t$-PA, using a tensor spline of two restricted cubic splines and penalization (shrinkage). Dark (cold color) regions are low risk, and bright (hot) regions are higher risk.

4.4

# Tables

- Binary variables: Show proportions first; they should be featured because they are normalized for sample size
  - Don't need to show both proportions (e.g., only show proportion of females)
  - Proportions are better than percents because of reduced confusion when speaking of percent difference (is it relative or absolute?) and because percentages such as 0.3% are often mistaken for 30% or 3%.
- Make logical choices for independent and dependent variables.
  E.g., less useful to show proportion of males for patients who lived vs. those who died than to show proportion of deaths stratified by sex.
- Continuous variables
  - to summarize distributions of raw data: 3 quartiles
    recommended format: $_{35}$ **50** $_{67}$ or 35/50/67
  - summary statistics: mean or median and confidence limits (without assuming normality of data if possible)
- Show number of missing values
- Add denominators when feasible

Table 4.1: Descriptive Statistics: Demographic and Clinical variables

| | N | |
|---|---|---|
| Age | 27 | $_{28}$ 32 $_{52}$ |
| C reactive protein | 27 | $_{1.0}$ 1.8 $_{10.1}$ |
| Fecal Calprotectin | 26 | $_{128}$ 754 $_{2500}$ |
| Gender | 27 | |
| Female | | 0.52 $\frac{14}{27}$ |
| Location of colitis | 27 | |
| Left side | | 0.41 $\frac{11}{27}$ |
| Middle | | 0.52 $\frac{14}{27}$ |
| Right side | | 0.07 $\frac{2}{27}$ |

$_a\, b\, _c$ represent the lower quartile $a$, the median $b$, and the upper quartile $c$ for continuous variables.
$N$ is the number of non–missing values.

## 4.5

# Bar Plots with Error Bars

- "Dynamite" Plots
- Height of bar indicates mean, lines represent standard error
- High ink:information ratio
- Hide the raw data, assume symmetric confidence intervals
- Replace with
  - Dot plot (smaller sample sizes)
  - Box plot (larger sample size)

```
getHdata(FEV); set.seed(13)
FEV ← subset(FEV, runif(nrow(FEV)) < 1/8)    # 1/8 sample
require(ggplot2)
s ← with(FEV, summarize(fev, llist(sex, smoke), smean.cl.normal))
ggplot(s, aes(x=smoke, y=fev, fill=sex)) +     # Fig. 4.24
    geom_bar(position=position_dodge(), stat="identity") +
    geom_errorbar(aes(ymin=Lower, ymax=Upper),
                  width=.1,
                  position=position_dodge(.9))
```

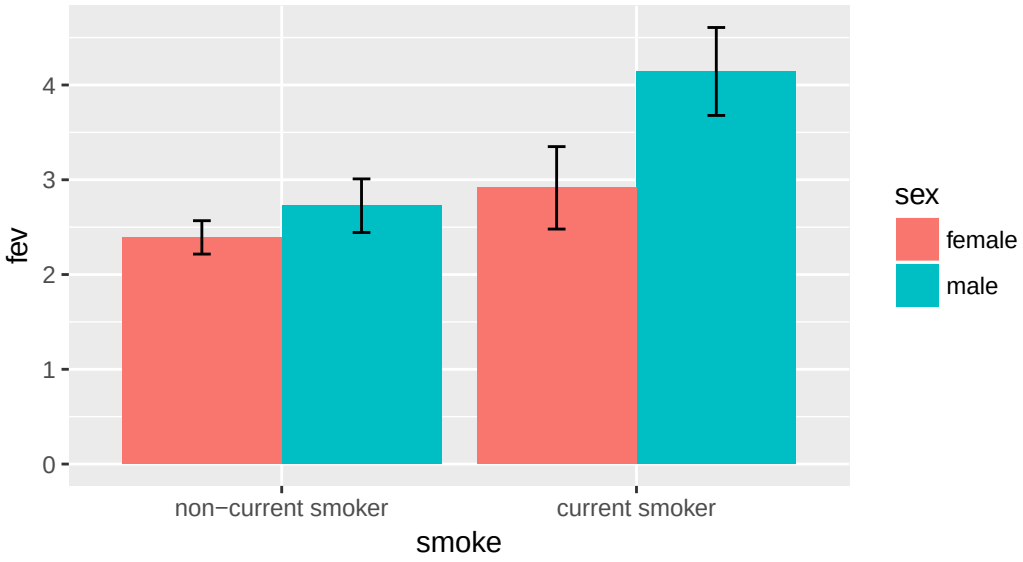



Figure 4.24: Bar plot with error bars—"dynamite plot"

See http://biostat.mc.vanderbilt.edu/DynamitePlots for a list of the many problems caused by dynamite plots, plus some solutions.

Instead of the limited information shown in the bar chart, show the raw data along with box plots. Modify default box plots to replace whiskers with the interval between 0.1 and 0.9 quantiles.

```
require(ggplot2)    # Fig. 4.25
stats ← function(x) {
  z ← quantile(x, probs=c(.1, .25, .5, .75, .9))
  names(z) ← c('ymin', 'lower', 'middle', 'upper', 'ymax')
  if(length(x) < 10) z[c(1,5)] ← NA
  z
}
ggplot(FEV, aes(x=sex, y=fev)) +
  stat_summary(fun.data=stats, geom='boxplot', aes(width=.75), shape=5,
               position='dodge', col='lightblue') +
  geom_dotplot(binaxis='y', stackdir='center', position='dodge', alpha=.4) +
  stat_summary(fun.y=mean, geom='point', shape=5, size=4, color='blue') +
  facet_grid(~ smoke) +
  xlab('') + ylab(expression(FEV[1])) + coord_flip()
```

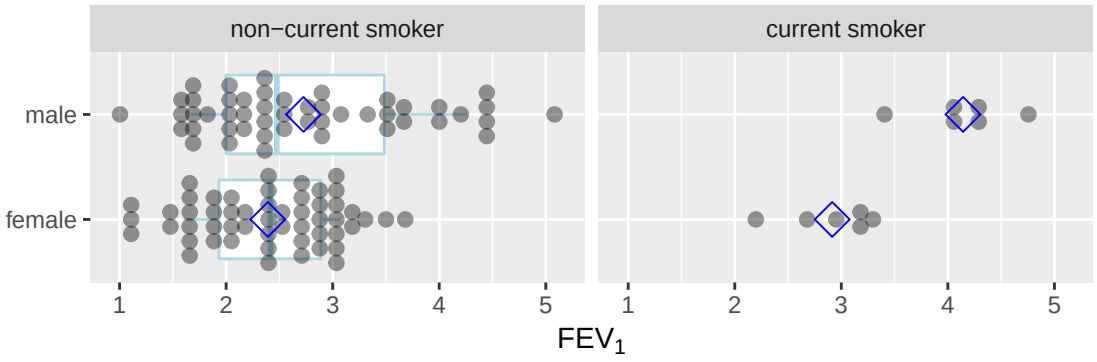



Figure 4.25: Jittered raw data and box plots. Middle vertical lines indicate medians and diamonds indicate means. Horizontal lines indicate 0.1 to 0.9 quantiles when $n \geq 10$. The ink:information ratio for this plot is far better than a dynamite plot.

Use a violin plot to show the distribution density estimate (and its mirror image) instead of a box plot.

```
ggplot(FEV, aes(x=sex, y=fev)) +
  geom_violin(width=.6, col='lightblue') +
  geom_dotplot(binaxis='y', stackdir='center', position='dodge', alpha=.4) +
  stat_summary(fun.y=median, geom='point', color='blue', shape='+', size=12) +
  facet_grid(~ smoke) +
  xlab('') + ylab(expression(FEV[1])) + coord_flip()
```

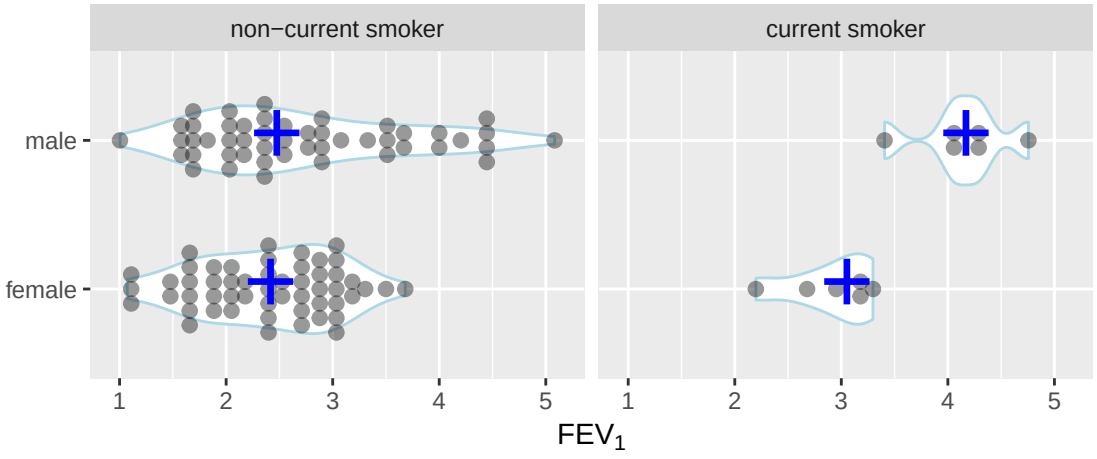



Figure 4.26: Jittered raw data and violin plots with median indicated by +

# Annotated Bibliography


[1] William S. Cleveland. "Graphs in scientific publications". In: *Am Statistician* 38 (1984). C/R 85v39 p238-9, pp. 261–269 (cit. on p. 4-10).

[2] William S. Cleveland. *The Elements of Graphing Data*. Summit, NJ: Hobart Press, 1994 (cit. on p. 4-10).

[3] Charles S. Davis. *Statistical Methods for the Analysis of Repeated Measurements*. New York: Springer, 2002 (cit. on p. 4-11).

[4] Frank E. Harrell. *Hmisc: A package of miscellaneous* R *functions*. 2015. url: http://biostat.mc.vanderbilt.edu/Hmisc (cit. on p. 4-5).

[5] Frank E. Harrell. *rms: R functions for biostatistical/epidemiologic modeling, testing, estimation, validation, graphics, prediction, and typesetting by storing enhanced model design attributes in the fit*. biostat.mc.vanderbilt.edu/Rrms. Implements methods in Regression Modeling Strategies, 2nd edition. 2016. url: http://biostat.mc.vanderbilt.edu/rms (cit. on p. 4-23).

[6] Paul Murrell. "InfoVis and statistical graphics: Comment". In: *J Comp Graph Stat* 22.1 (2013), pp. 33–37. eprint: http://www.tandfonline.com/doi/pdf/10.1080/10618600.2012.751875. url: http://www.tandfonline.com/doi/abs/10.1080/10618600.2012.751875 (cit. on p. 4-10).

Excellent brief how-to list; incorporated into graphscourse
.

[7] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. Vienna, Austria, 2016. isbn: 3-900051-07-0. url: http://www.R-project.org (cit. on p. 1-1).