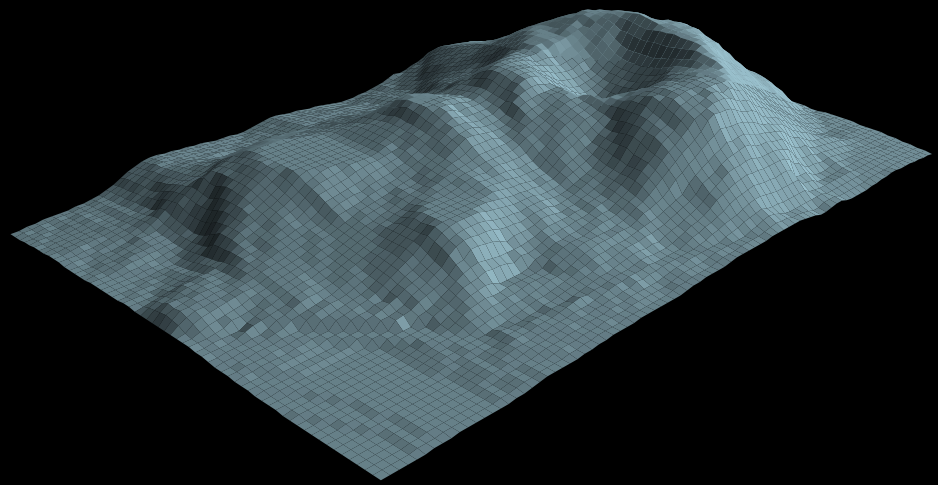
Introducing 

JEFF GILL
Department of Political Science
Division of Biostatistics
Department of Surgery (Public Health Sciences)
Washington University, St. Louis



Volcano Code

```
dim(volcano)
[1] 87 61

z <- 2 * volcano        # Exaggerate the relief
x <- 10 * (1:nrow(z))   # 10 meter spacing (S to N)
y <- 10 * (1:ncol(z))   # 10 meter spacing (E to W)
par(bg = "black",mar=c(0,0,0,0))
persp(x, y, z, theta = 135, phi = 30, col = "lightblue", scale = FALSE,
      ltheta = -120, shade = 0.75, border = NA, box = FALSE)
```

Reminders

- Thank R Its Friday (TRIF) is supported by the Washington University TREC Center (`www.obesity-cancer.wustl.edu`), School of Medicine.
- Meeting times and topics (3-4PM):
  - October 25: Basic data analysis and introduction to graphics
  - November 22: Running regression models in R
  - January 31, 2014: Analysis of Variance
  - February 28, 2014: Logistic Regression
  - March 28, 2014: Principle Components Analysis
  - April 25, 2014: Survival Analysis
  - May 30, 2014: Nonparametric Data Analysis
- The slides for today are available at `http://jgill.wustl.edu/slides/trif2.pdf`.

Loading Data

- There are several different ways of loading data, depending on what format it comes in.
- For manually inputting (small) datasets, create the first line of the data with an R assignment command:

  ```
  x <- c(1,2,3)
  ```

  then you can enter more with:

  ```
  edit(x)
  ```

  which gives a pop-up window with: `c(1,2,3)` and room to edit/extend, or

  ```
  data.entry(x)
  ```

  which gives a different pop-up window with the data down a column and room to edit/extend.
- If you want bring code into R as if were typed in the environment window use `source("filename")`, which is the opposite of `sink("filename")`, `sink()`.

Loading Data

- The most common is `read.table`, which has many options, the most common are:

```
read.table(file, header = FALSE, sep = "", quote = "\"'", dec = ".", row.names,
           col.names, na.strings = "NA", strip.white = FALSE,
           blank.lines.skip = TRUE, comment.char = "#")
```

  Also, `file` here can be a file on your hard-drive or a file from a URL.

- Here is an example of `read.table`:

```
mouse <- read.table("/Users/jgill/Grant.TREC/CompiledMouseData.dat",header=TRUE)
mouse[1,]
Mouse_Number Age_when_used Body_weight UGS_weight Prostate_Weight
          56           119          NA         NA              NA
Number_male_pups_in_cage Cage_Number Diet_Treatment Age_parents_at_birth
                       4          31              0                  97
Time_Parents_on_diet_before_birth Total_Acini Normal
                               69         170    165
Number_Hyperproliferative
                        5
```

Loading Data

- `read.table` is fussy about having the exact same number of items on each line, so to check this in advance use:

```
count.fields("/Users/jgill/Grant.TREC/CompiledMouseData.dat")
[1] 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
```

- `read.table` actually uses another function called `scan` and formats it according to the `read.table` parameters specified.

- `scan` is used directly to bring in data that is not structured since it treats the data file as one long vector:

```
scan("Class.Multilevel/CODAchain1.txt")[1:10]
Read 400000 items
 [1] 1.0000 1.3832 2.0000 1.6580 3.0000 1.8051 4.0000 1.8193 5.0000 1.9530
```

- The command `read.csv()` is really `read.table` with the option `sep=‘‘,’’`.

Loading Data

- The command `load` brings in an R formatted file that has been saved with the command `save`.
- An example:

```
load("Class.Stat.Comp/prostate.sav")
names(prostate)
 [1] "patno"  "stage"  "rx"     "dtime"  "status" "age"    "wt"     "pf"
 [9] "hx"     "sbp"    "dbp"    "ekg"    "hg"     "sz"     "sg"     "ap"
[17] "bm"     "sdate"
```

- `load` does not have many options:

```
load(file, envir = parent.frame(), verbose = FALSE)
```

where `envir` is used when you have multiple R environments simultaneously (most people do not do this), and `verbose` set equal to `TRUE` will print the variables names on loading.

Loading Data

- To use `load` from a website, do this:

```
connect1 <- url("http://jgill.wustl.edu/data/Pixel.rda")
load(connect1)
close(connect1)
Pixel[1:10,]
   Dog Side day  pixel
1    1    R   0 1045.8
2    1    R   1 1044.5
3    1    R   2 1042.9
4    1    R   4 1050.4
5    1    R   6 1045.2
6    1    R  10 1038.9
7    1    R  14 1039.8
8    2    R   0 1041.8
9    2    R   1 1045.6
10   2    R   2 1051.0
```

Loading Data

- Sometimes we get data files that have no field delimiters, and instead come with a recipe of column assignments.
- The function `read.fwf` performs this function (also with `scan`).
- For example:

```
pa.raw <- read.fwf("Article.P-Agent/mackenzie.fixed.dat", width=c(
    4,1,2,2,2,2,2,1,1,  #end of long XXX's
    1,2,1,2,1,8,1,2,1,  #column 37 finished
    1,1,2,8,1,1,1,1,1,  #colum 54 finished
    1,8,6,1,1,1,2,1,1,
    1,1,1,              #first deck finished
    5,1,6,1,1,1,7,1,    #colum 32, deck 2 starts
    1,1,1,1,1,1,1,1,1,  #var 59 done
    1,1,1,1,1,1,
    1,1,1,1,1,1,        #finished col42
    1,1,1,1,1,1,1,1,1,  #finished col53/var77
    1,1,1,1,1,1,1,      #finished col60/var84
    2,1,1,1,1,1,1,1,    #finished col69/var92
    1,1,2,1,1,1,1,1,1,1)    #finished col80/var102, deck2
```

Loading Data

- What about loading data from other statistical software?
- The R library for doing this `foreign` (we'll cover libraries shortly).
- So after typing `library(foreign)`, you have the following functions (and more):
  - `read.dbf`, reads a standard DBF database file into a data frame.
  - `read.dta`, reads a Stata binary data file into a data frame
  - `read.mtp`, reads a Minitab Portable Worksheet into a list
  - `read.octave`, reads Octave text data into a list
  - `read.spss`, reads SPSS data files into a data frame
  - `read.ssd`, uses SAS (you need it installed on the same machine), to convert a ssd file into a transport format, then reads it into a data frame
  - `read.systat`, reads a Systat file into a data frame
  - `read.xport`, reads a SAS XPORT format library and returns a list of data frames
  - `read.S`, reads an Splus version 3 data file.

What Are Packages?

- User contributed application packages for specific routines.
- Syntax for loading onto your system:

  ```
  install.packages("UsingR","my.library.directory")}.
  ```

- Syntax to start using the package once loaded:

  ```
  library(UsingR)
  ```

- To find your current library locations and packages that are available:

  ```
  library()
  ```

- To find your current *loaded* packages:

  ```
  search()
  ```

  some of which are "autoloaded" when you start R.

Accessing Packages and Their Data

- Start package: `library(glmdm)`
- Load data locally from that package: `data(asia)`
- Look at these data:

```
asia
    ATT DEM FED SYS AUT
1     0  -7   0   0   0
2     0  -8   0   0   0
3     0  -7   0   0   0
4     1   0   0   0   0
:
```

- Print the variable names:

```
names(asia)
[1] "ATT" "DEM" "FED" "SYS" "AUT"
```

Accessing Packages and Their Data

- Attaching related commands:

```
attach(asia)
mean(ATT)
[1] 0.4466667
detach(asia)
```

- This function is very general:

```
with(asia,FUNCTION)
```

which works like this:

```
with(asia,cor(ATT,DEM))
[1] 0.1526725
```

Lists

- Lists are simply collections of various objects of different types and sizes.
- Since regression procedures provide different types of information (coefficients, AIC, residuals, etc.), it make sense to deliver this in a list.
- For instance,

```
names(prostate.gam1)
 [1] "coefficients"      "residuals"         "fitted.values"     "family"
 [6] "deviance"          "null.deviance"     "iter"              "weights"
[11] "df.null"           "y"                 "converged"         "boundary"
[16] "reml.scale"        "aic"               "rank"              "sp"
[21] "outer.info"        "scale.estimated"   "scale"             "Vp"
[26] "Ve"                "edf"               "edf1"              "F"
[31] "nsdf"              "sig2"              "method"            "smooth"
[36] "var.summary"       "cmX"               "model"             "control"
[41] "pterms"            "assign"            "xlevels"           "offset"
[46] "min.edf"           "optimizer"         "call"
```

Manipulating Lists

```
temp.list <- list("X"=c(1,2,3),"chars"=c("name1","name2"),
                  "example.matrix"=matrix(c(1,2,3,4),ncol=2))
names(temp.list)
[1] "X"              "chars"          "example.matrix"
temp.list$example.matrix
     [,1] [,2]
[1,]    1    3
[2,]    2    4
temp.list[1]
$X
[1] 1 2 3
temp.list[[1]]
[1] 1 2 3
temp.list[[1]][1]
[1] 1
temp.list[[3]][1,1]
[1] 1
```

Manipulating Lists

```
temp.list$X
[1] 1 2 3
temp.list$chars
[1] "name1" "name2"
temp.list$example.matrix
     [,1] [,2]
[1,]    1    3
[2,]    2    4
```

Naming Matrices

- Matrices do not begin with names, unless inherited from other objects used to create them, but they can be assigned with the list format:

```
dimnames(ZZ)    NULL
dimnames(ZZ) <- list(c("r1","r2"),c("c1","c2"))
ZZ
         c1       c2
r1 5.000000 2.108736
r2 2.108736 3.145143
rownames(ZZ)
[1] "r1" "r2"
colnames(ZZ)
[1] "c1" "c2"
rownames(ZZ) <- c("R1","R2")
colnames(ZZ) <- c("C1","C2")
dimnames(ZZ)
[[1]]
[1] "R1" "R2"
[[2]]
[1] "C1" "C2"
```

Some Basic Plotting Commands

- `barplot(x, width, names, space=.2, beside=FALSE, horiz=FALSE, legend, angle, density, col)` (there are even more options)
- `hist(x, nclass, breaks, plot=TRUE, angle, density, col)`
- `boxplot(x,col="violet")`
- The most general: `plot(x,y)`

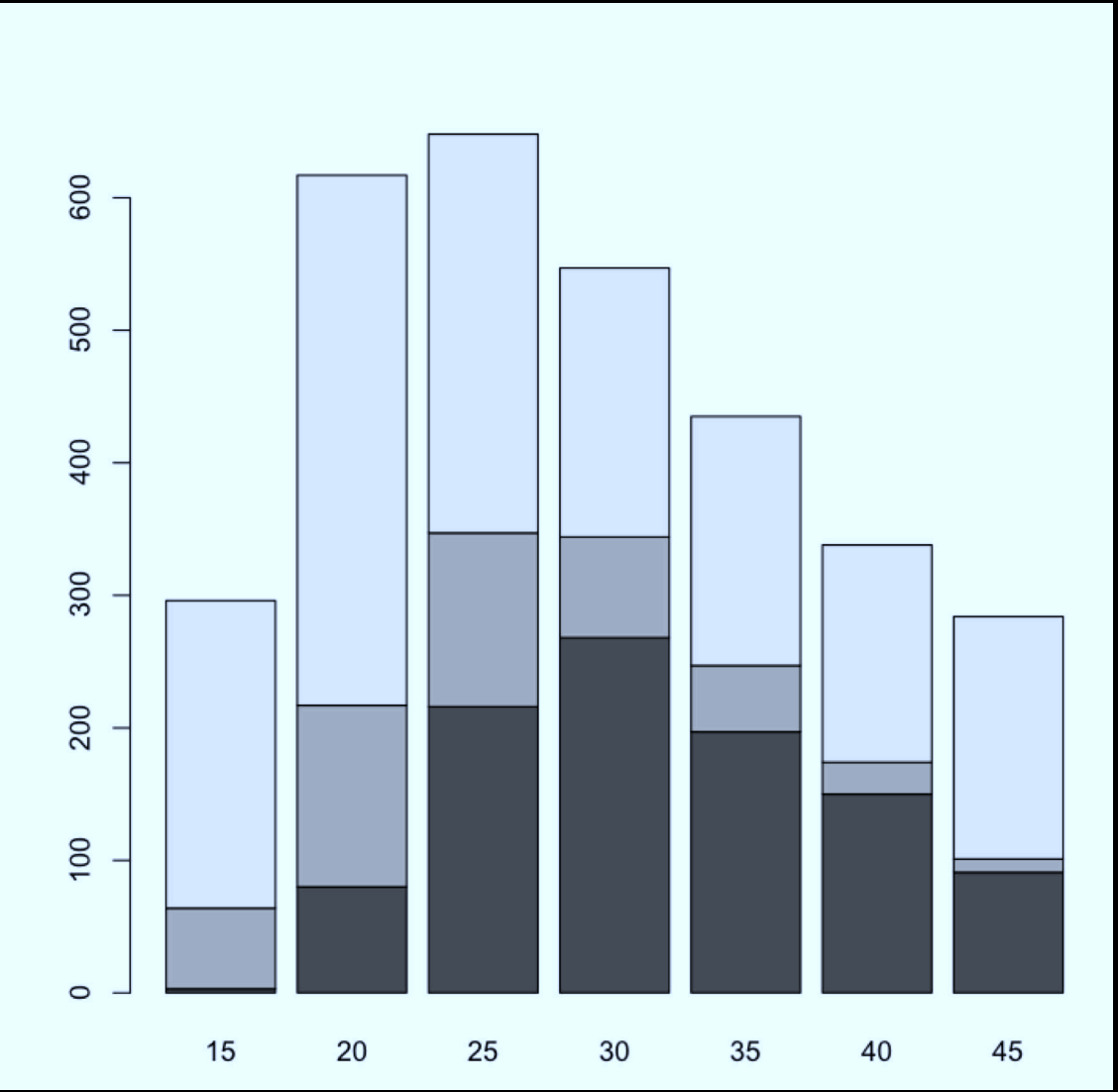
Some parameters for `plot`

- `main='title'`
- `new=<logical>`
- `pch='.'`, `pch=3`, `pch='+'`
- `xlab='x-axis label'`
- `ylab='y-axis label'`
- `xlim=c(xlo.value,xhi.value)`
- `ylim=c(ylo.value,yhi.value)`
- `type`, `"p"`, for points, `"l"` for lines, `"h"` for vertical lines.
- `bty`, shape of the outer box, o,l,c,etc.
- `cex`, magnification for characters, default is 1.
- `lty`, type of line, number from 1 to 10(ish).
- `lwd`, width of line, default is 1.
- `col`, color, use `color()` to see options.

Plotting Categorical Data

```
# Current Use of Contraception By Age, Currently Married Women. El Salvador, 1985.
# Table 6.1 was reconstructed from weighted percents found in Table 4.7 of the final
# report of the Demographic and Health Survey conducted in El Salvador in 1985
# (FESAL-1985). The table shows 3165 currently married women classified by age,
# grouped in five-year intervals, and current use of contraception, classified as
# sterilization, other methods, and no method.
contraception.df <- read.table(
                    "http://people.hmdc.harvard.edu/~jgill/contraception.dat",
                                        header=TRUE, row.names=1)
( contraception.df <- contraception.df[,-4] )
      Sterilization Other None
   15             3    61  232
   20            80   137  400
   25           216   131  301
   30           268    76  203
   35           197    50  188
   40           150    24  164
   45            91    10  183
barplot(t(as.matrix(contraception.df)))
```

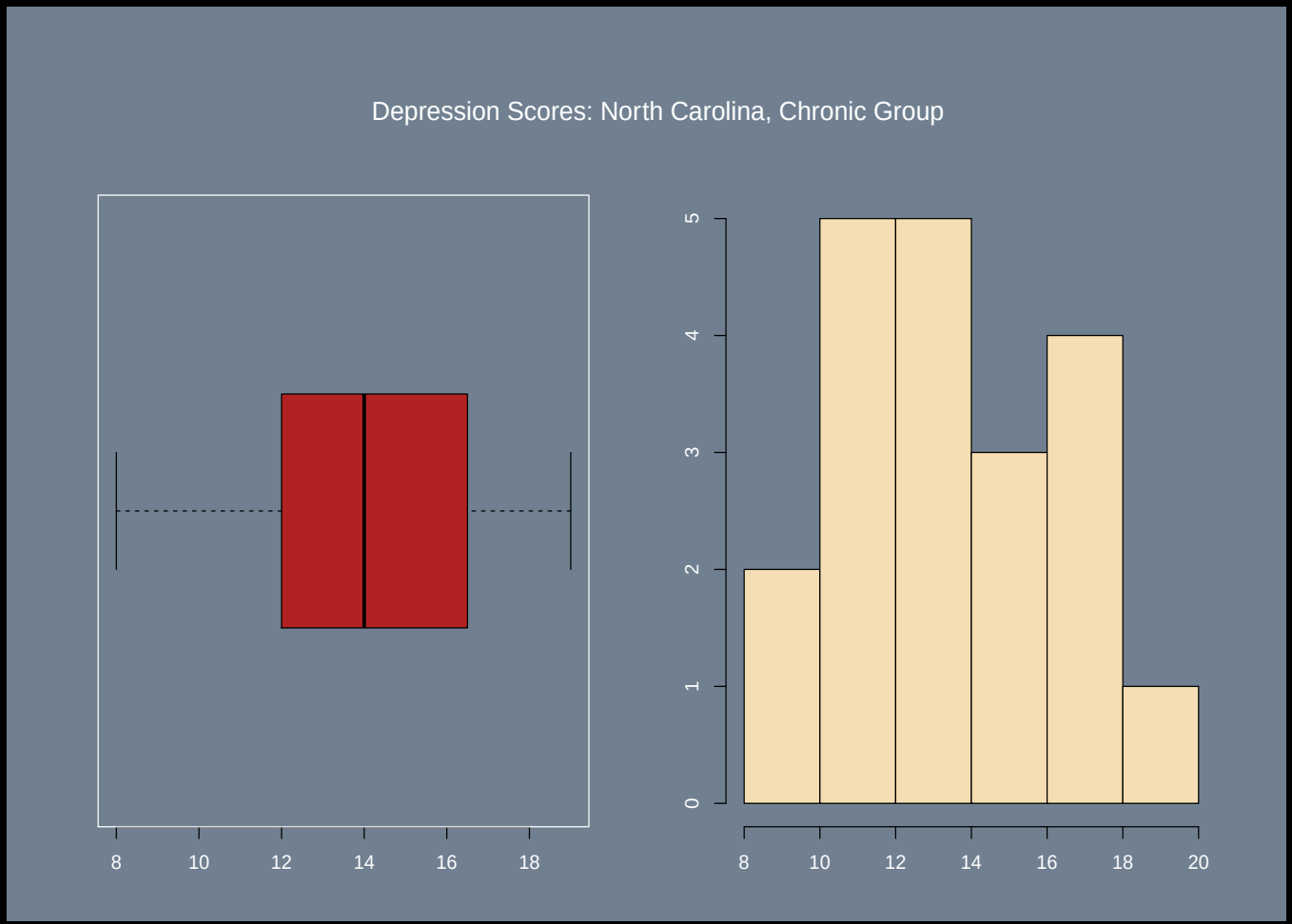
The Simplest Graph You'll See Today



Some Basic Two-Dimensional Plotting Commands

- `lines(x, y, type="l")` (overlays)
- `points(x, y, type="p")` (overlays)
- `abline(coef) abline(a, b) abline(reg) abline(h=) abline(v=)` (overlays)
- `qqplot(x, y, plot=TRUE)` (new plot)
- `qqnorm(x, datax=FALSE, plot=TRUE)` (new plot)

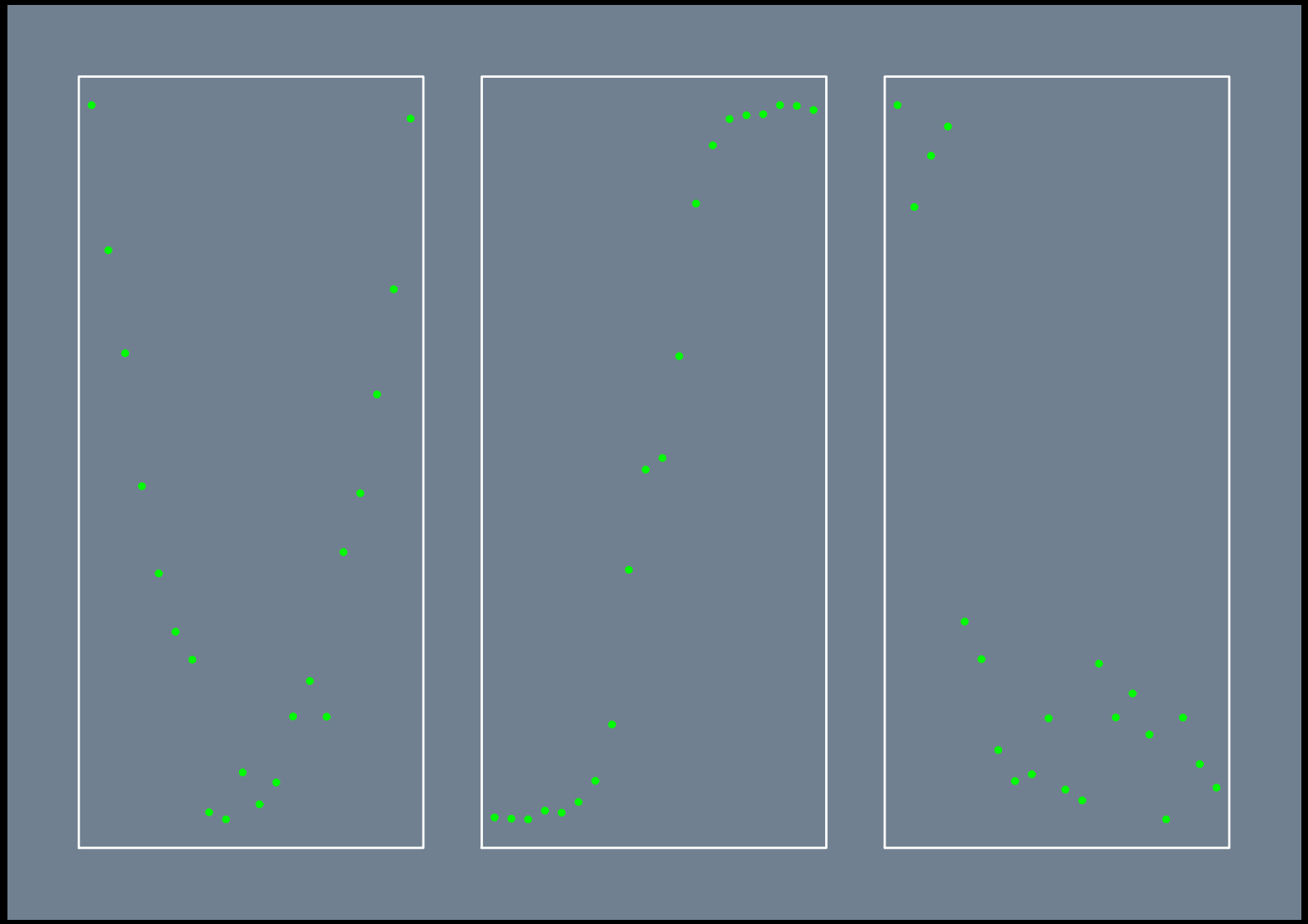
Combining Different Graphs



Combining Different Graphs

```
depression.vec <- scan("http://jgill.wustl.edu/data/depression.data")
depression.mat <- matrix(depression.vec,ncol=6,byrow=TRUE)
par(mfrow=c(1,2),mar=c(3,3,3,3),oma=c(1,1,5,0),col.axis="white",col.lab="white",
    col.sub="white",col="white",bg="slategray")
boxplot(depression.mat[,6],col="firebrick",horizontal=TRUE)
hist(depression.mat[,6],col="wheat",main="")
mtext(outer=TRUE,"Depression Scores: North Carolina, Chronic Group",cex=1.3)
```

When Not To Use Correlation



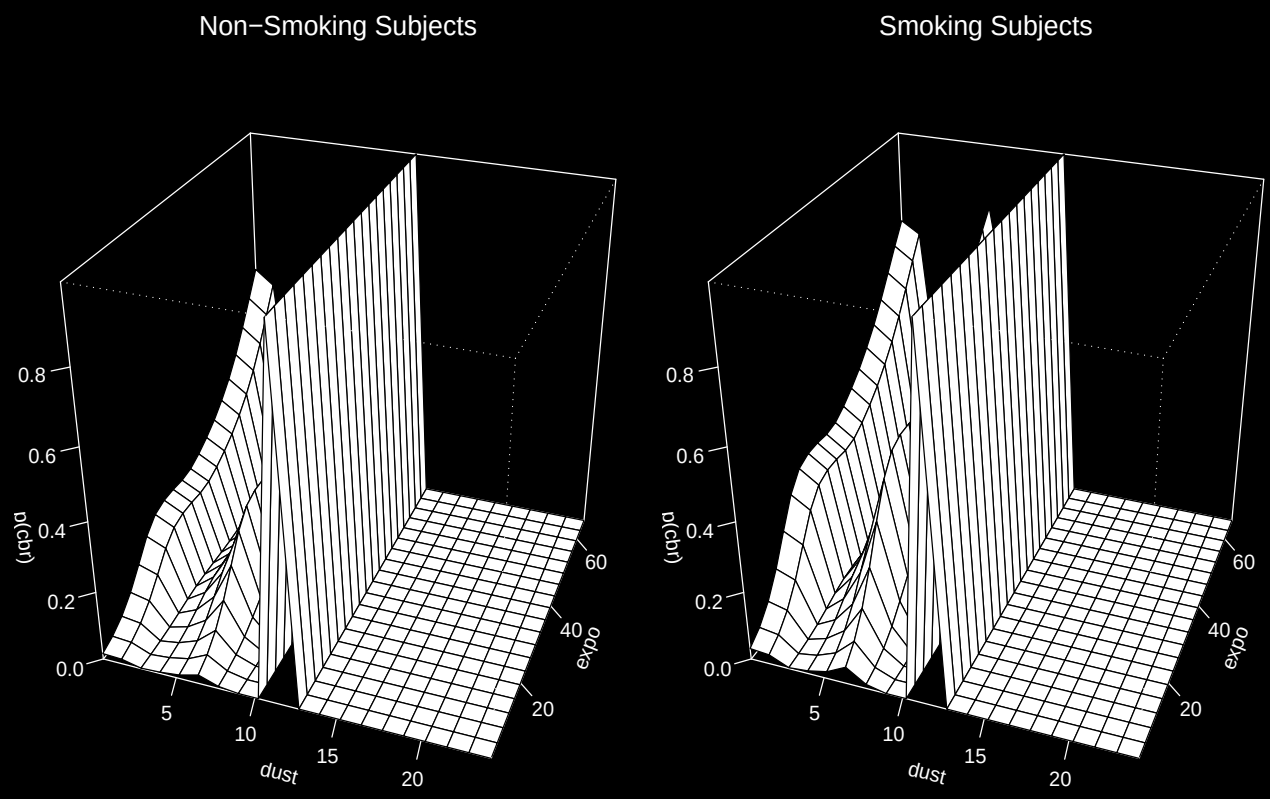
When Not To Use Correlation (or linear regression)

```
par(mfrow=c(1,3),mar=c(2,2,2,2),lwd=2,col.axis="white",col.lab="white",
    col.sub="white",col="white",bg="slategray",cex.lab=1.3,oma=c(1,1,1,1))
x <- seq(-5,5,length=20); y1 <- jitter(x^2,20);
y2 <- jitter(atan(x^3),30); y3 <- jitter(c(10,9,9.5,9.8,runif(16,3,5)),10)
plot(x,y1,pch=19,col="green",xaxt="n",yaxt="n",xlab="",ylab="")
plot(x,y2,pch=19,col="green",xaxt="n",yaxt="n",xlab="",ylab="")
plot(x,y3,pch=19,col="green",xaxt="n",yaxt="n",xlab="",ylab="")
```

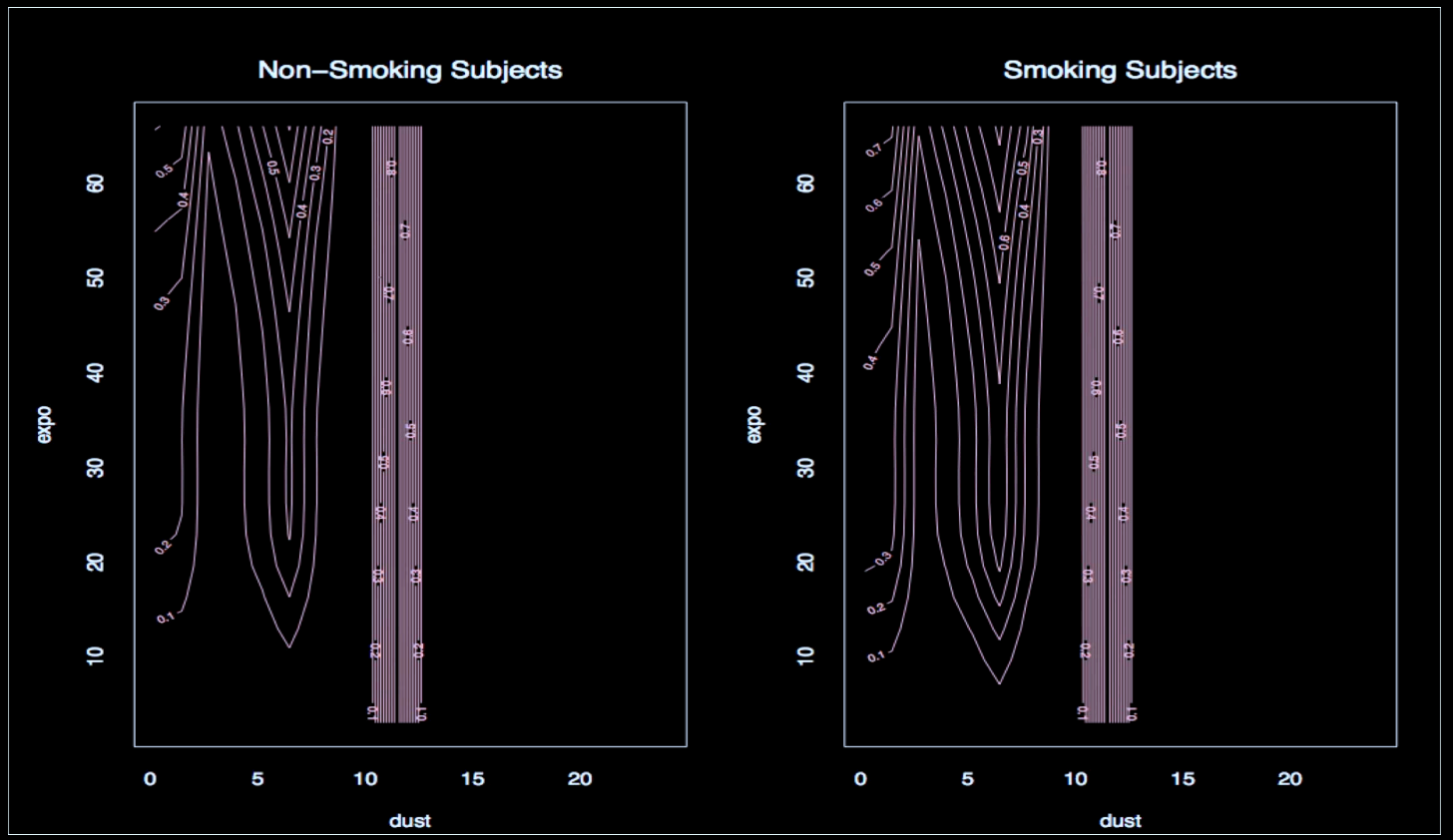
Some Basic Three-Dimensional Plotting Commands

- `contour(x, y, z, v, nint=5, add=FALSE, labex)`
- `persp(x, y, z, theta = 135, phi = 30, col = "lightblue", scale = FALSE, ltheta = -120, shade = 0.75, border = NA, box = FALSE)`

Predictions From GAM Output, Perspective Plot



Predictions From GAM Output, Contour Plot



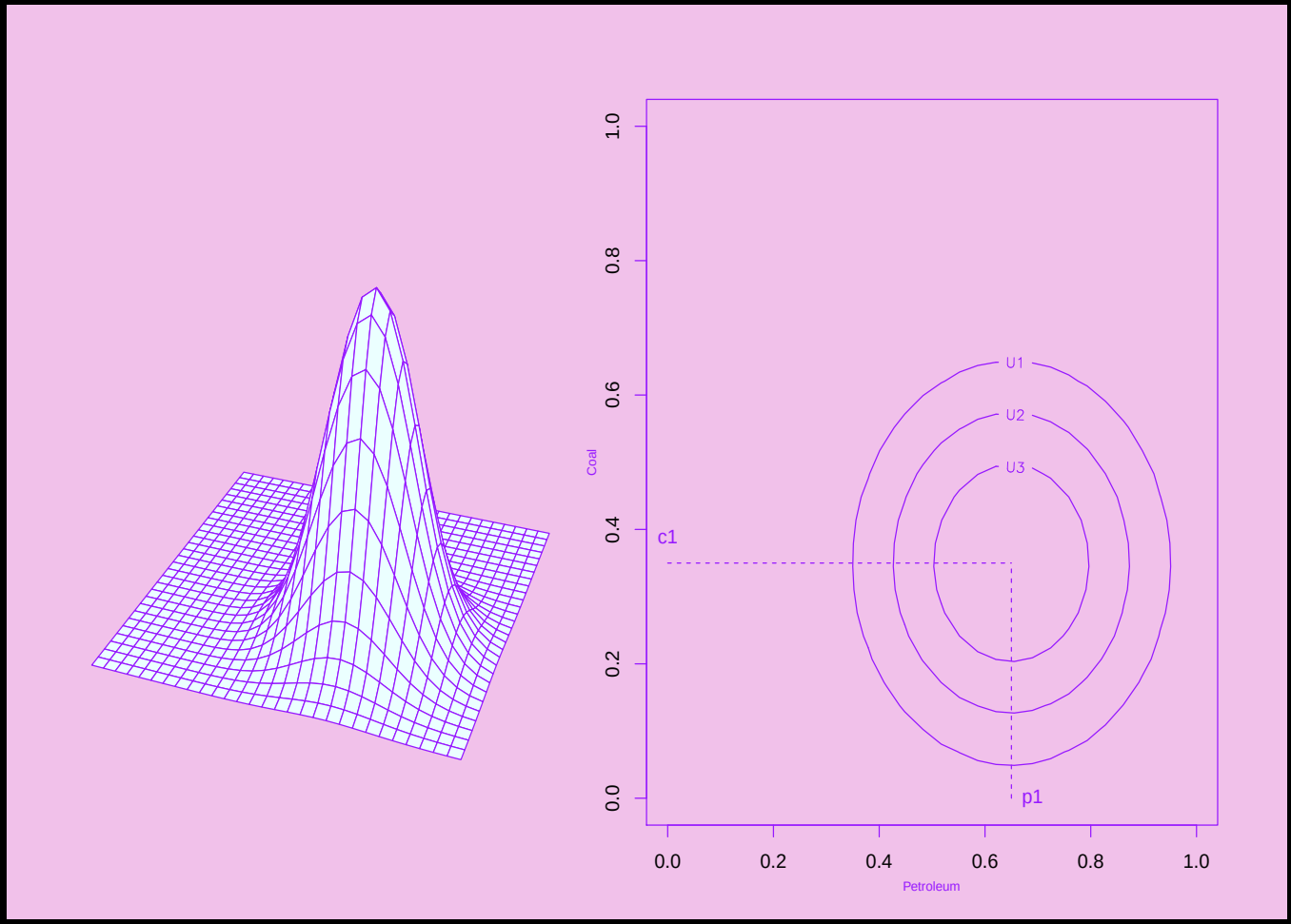
Detailed Example

```
library(mvtnorm)
ruler <- seq(0,1,length=30)
energy.dens <- energy.dens2 <- matrix(NA,length(ruler),length(ruler))
var.mat <- matrix(c(.02,0,0,.02),nrow=2)
mean.vec <- c(0.65,0.35)
for (i in 1:length(ruler))  {
    for (j in 1:length(ruler))
        energy.dens[i,j] <- dmvnorm(c(ruler[i],ruler[j]),mean.vec,var.mat)
}
for (i in 1:length(ruler))  {
    for (j in 1:length(ruler))
        energy.dens2[i,j] <- dmvnorm(c(ruler[i],ruler[j]),mean.vec,var.mat)
}
```

Detailed Example

```
par(mfrow=c(1,2),mar=c(3,0,3,2),oma=c(1,1,1,1),bg="pink",fg="purple")
persp(ruler,ruler,energy.dens, box=F,axes=TRUE,
        theta=30,phi=50,r=5,xlab="petroleum",ylab="coal",zlab="utility")
contour(ruler,ruler,energy.dens,levels=c(0.5,2,5.25),labels=c("U1","U2","U3"),
        vfont=c("sans serif","plain"),drawlabels=TRUE,labcex=0.8)
segments(0.65,0,0.65,0.35,lty=2)
segments(0,0.35,0.65,0.35,lty=2)
mtext(outer=F,side=1,line=2,"Petroleum",cex=0.7)
mtext(outer=F,side=2,line=2,"Coal",cex=0.7)
text(0.69,0,"p1")
text(0,0.39,"c1")
```

Detailed Example



Important Ways To Export Your Graphs

- `postscript(file, command, horizontal=F, width, height, rasters, pointsize=14, font=1, preamble=ps.preamble, fonts=ps.fonts)`
- `pdf((file = ifelse(onefile, "Rplots.pdf", "Rplot%03d.pdf"), width = 6, height = 6, onefile = TRUE, ...)`
- There are more options for both of these.
- You will not see your graph in the graphing window, all comannds direct to the file specified.
- End your code section with `dev.off()` to shut down the graphics device.
- Also, be sure to make use of:

  ```
  getwd()
  setwd(dir)
  ```

  or use full pathnames for `file`.

Setting Up The Graphics Window

- The command `par()` configures the general look of the graph with many options and only the main ones are described here. Order does not matter.
- `bg` sets the background color.
- `bty` determins the box type around the figure, one of "o" (the default), "l", "7", "c", "u", or "]" ("n" suppresses the box).
- `cex` a number that gives size adjustment for text and symbols, defaults to 1.
- `cex.axis` size adjustment for the axis values.
- `cex.lab`: size adjustment for the x and y labels.
- `cex.main` size adjustment for the main titles.
- `cex.sub` size adjustment for the sub-titles.
- `col` sets the default plotting color (defaults to black).
- `col.axis` sets the axis color (defaults to black).
- `col.lab` sets the x and y labels color (defaults to black).

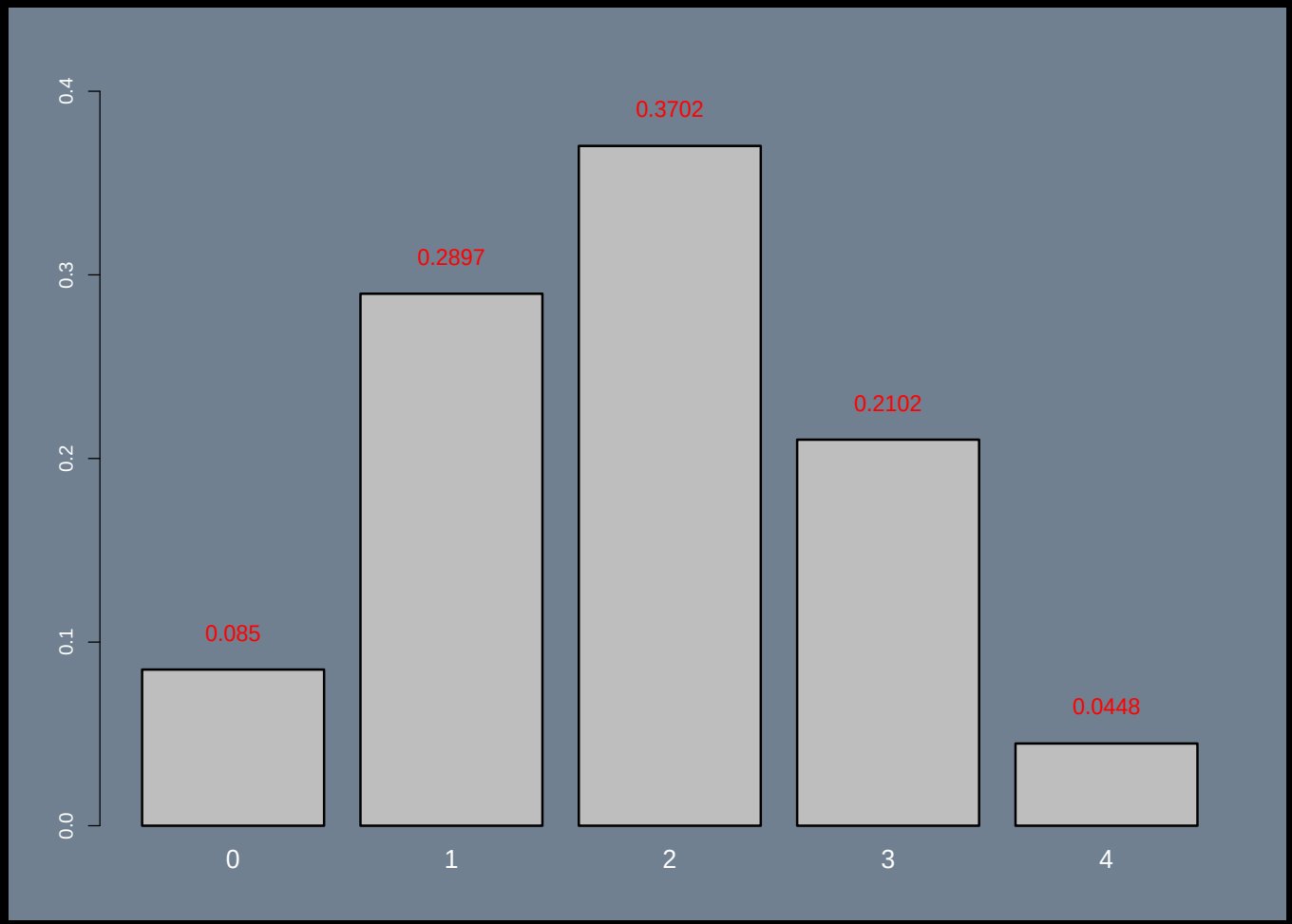
Setting Up The Graphics Window

- `col.main` sets the color for the main titles (defaults to black).
- `col.sub` sets the color for sub-titles (defaults to black).
- `fg` sets the color of all of the foreground objects (above) at once (defaults to black).
- `font` a number according to: 1 for plain text (the default), 2 to bold face, 3 to italic and 4 to bold italic.
- `font.axis` font number for the axes.
- `font.lab` font number for the x and y labels.
- `font.main` font number for the main titles.
- `font.sub` font number for sub-titles.
- `las` style of axis labels: 0 for always parallel to the axis (default), 1 for always horizontal, 2 for always perpendicular to the axis, 3 for always vertical.

Setting Up The Graphics Window

- `lty` line type: 0=blank, 1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash, or as one of the character strings "blank", "solid", "dashed", "dotted", "dotdash", "longdash", or "twodash".
- `lwd` line width, a positive number, defaulting to 1.
- `mar` numerical vector: c(bottom, left, top, right) giving of the margin for the four sides of the plot (default is c(5, 4, 4, 2) + 0.1).
- `mfcol`, `mfrow` vector of the form c(nr, nc) giving an nr-by-nc array on the device by columns (mfcol), or rows (mfrow), respectively. More advanced versions: `layout`, and `split.screen`.
- `oma`, vector c(bottom, left, top, right) giving the size of the outer margins in lines of text.
- `xaxt`, `yaxt` character giving x or y axis type "n" suppresses plotting of the axis, any other does not.

A Fancier Barplot From A Study of Acupuncture and Headaches Outcome Probabilities



A Fancier Barplot From A Study of Acupuncture and Headaches Outcome Probabilities

```
# WRITE TO A FILE
postscript("Class.Med.Stats/Images/binomial.plot.ps")
# GET CUMULATIVE BINOMIAL PROBABILITIES WHERE THE PROBABILITY OF A 1 IS 0.46
p.binom <- cum.binom <- pbinom(0:4,4,0.46)
# CALCUATE MARGINAL PROBABILITIES
p.binom[2:5] <- p.binom[2:5] - p.binom[1:4]
# SETUP GRAPHING WINDOW
par(mar=c(4,4,2,2),lwd=2,col.axis="white",col.lab="white",col.sub="white",
    col="white",bg="slategray")
# PLOT BARPLOT AND ALSO SAVE IT AS AN OBJECT (NOT REQUIRED FOR PLOTTING)
p.binom.bar <- barplot(p.binom,ylim=c(0,0.42),names.arg=0:4,cex.names=1.3,
    axisnames=TRUE)
# ADD NUMERICAL VALUES ON TOP OF BARS
text(round(p.binom,4),x=p.binom.bar,y=(p.binom+.02),col="red",cex=1.2)
dev.off()
```

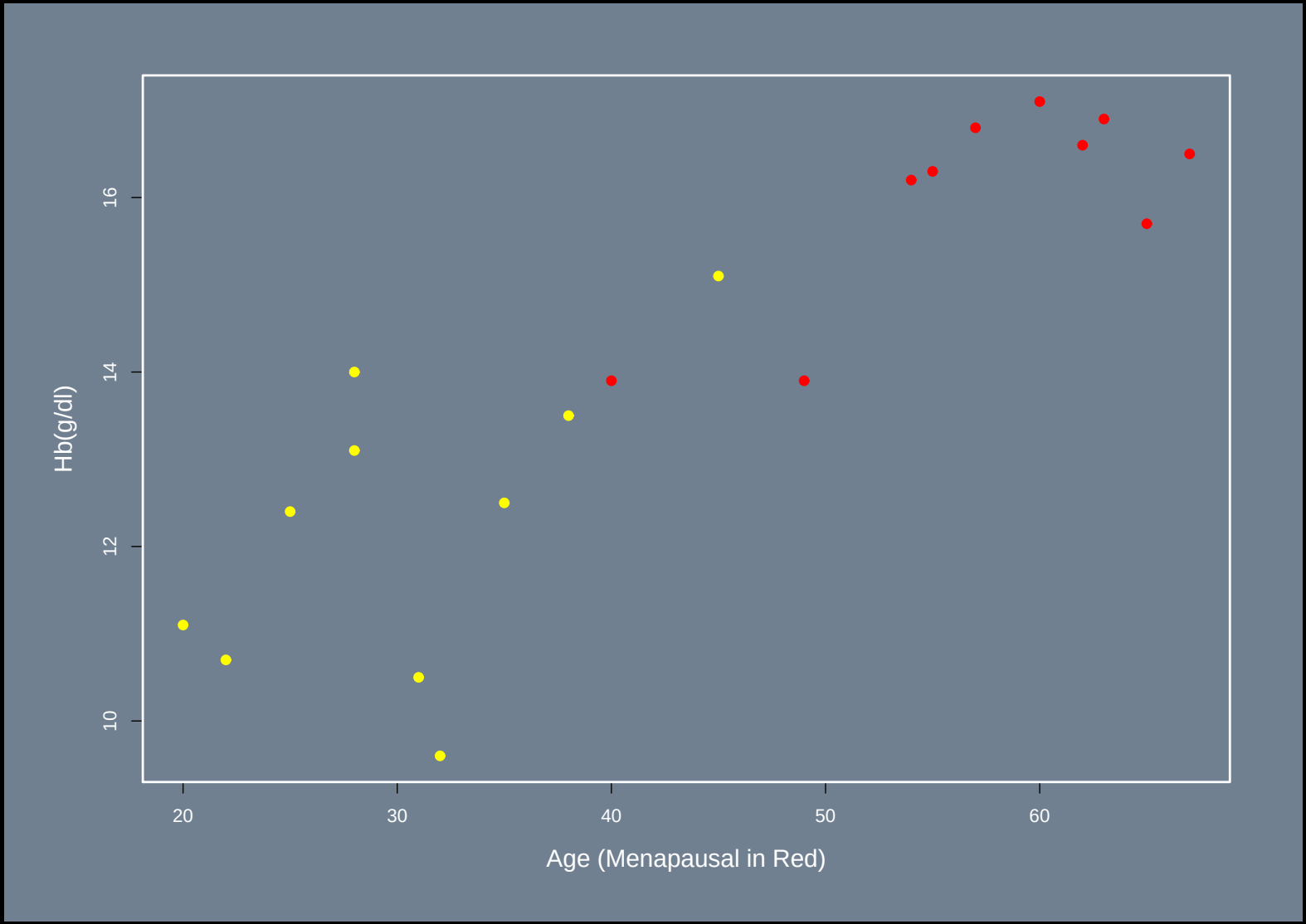
Anaemia Data

- Consider a study of anaemia in women in a given clinic where 20 cases are chosen at random from the full study to get the data here.
- From a blood sample we get:
  - haemoglobin level (Hb) in grams per deciliter (12–15 g/dl is normal in adult females)
  - packed cell volume (PCV) in percent of blood volume that is occupied by red blood cells (also called hematocrit, Ht or HCT, or erythrocyte volume fraction, EVF). 38% to 46% is normal in adult females.
- We also have:
  - age in years
  - menopausal (0=no, 1=yes)
- There is an obvious endogeneity problem in modeling Hb(g/dl) versus PCV(%).

Anaemia Data

Subject	Hb(g/dl)	PCV(%)	Age	Menopausal
1	11.1	35	20	0
2	10.7	45	22	0
3	12.4	47	25	0
4	14.0	50	28	0
5	13.1	31	28	0
6	10.5	30	31	0
7	9.6	25	32	0
8	12.5	33	35	0
9	13.5	35	38	0
10	13.9	40	40	1
11	15.1	45	45	0
12	13.9	47	49	1
13	16.2	49	54	1
14	16.3	42	55	1
15	16.8	40	57	1
16	17.1	50	60	1
17	16.6	46	62	1
18	16.9	55	63	1
19	15.7	42	65	1
20	16.5	46	67	1

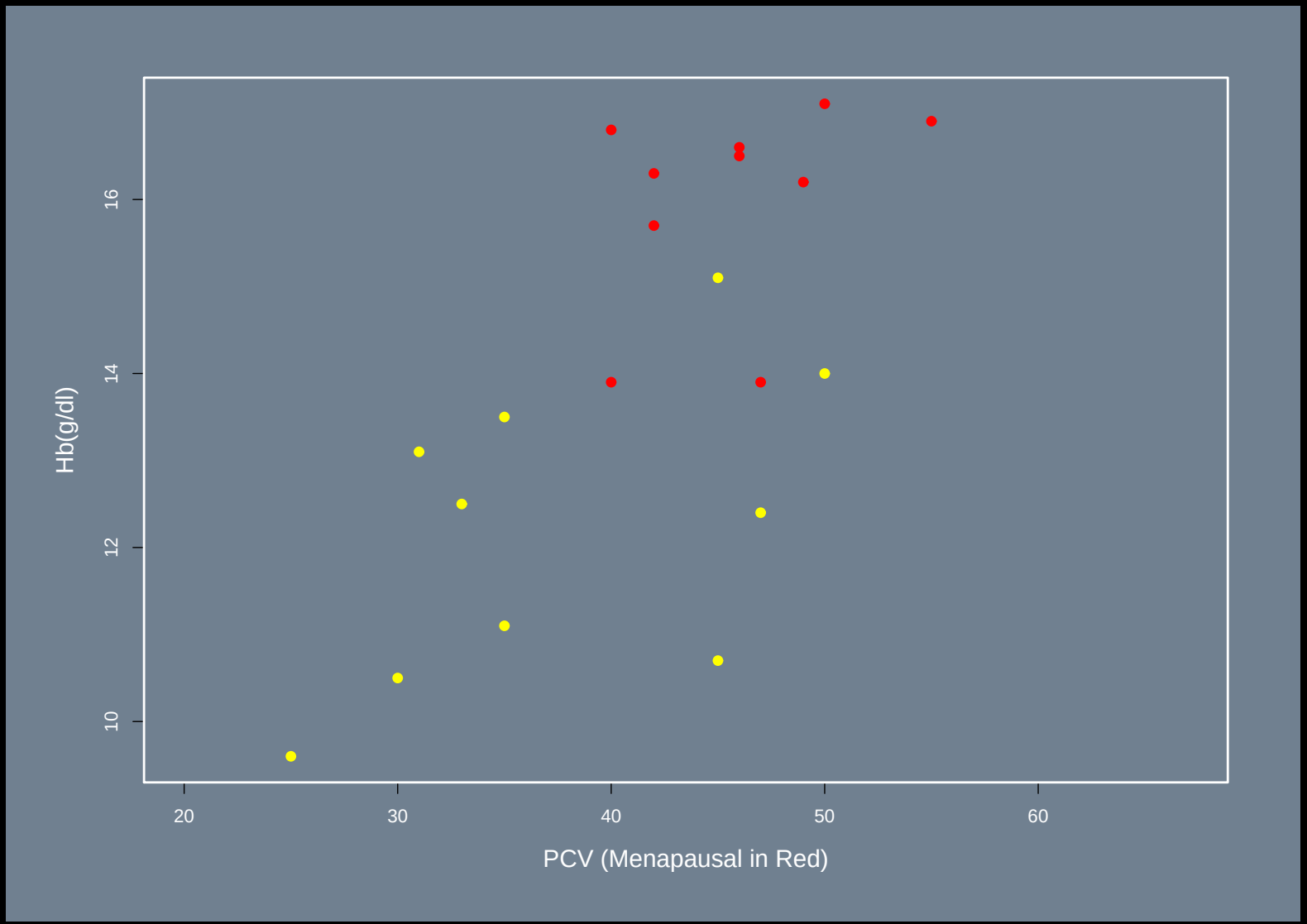
Scatterplot of the Anaemia Data



Scatterplot of the Anaemia Data

```
anaemia <- read.table("http://jgill.wustl.edu/data/anaemia.dat",
                      header=TRUE,row.names=1)

postscript("Class.Med.Stats/Images/anaemia1.fig.ps")
par(mfrow=c(1,1),mar=c(5,5,2,2),lwd=2,col.axis="white",col.lab="white",
    col.sub="white",col="white",bg="slategray", cex.lab=1.3)
plot(anaemia$Age[anaemia$Menapause==0],anaemia$Hb[anaemia$Menapause==0],pch=19,
     col="yellow", xlim=range(anaemia$Age),ylim=range(anaemia$Hb),
     xlab="Age (Menapausal in Red)",ylab="Hb(g/dl)")
points(anaemia$Age[anaemia$Menapause==1],anaemia$Hb[anaemia$Menapause==1],pch=19,
     col="red")
dev.off()
```

Scatterplot of the Anaemia Data



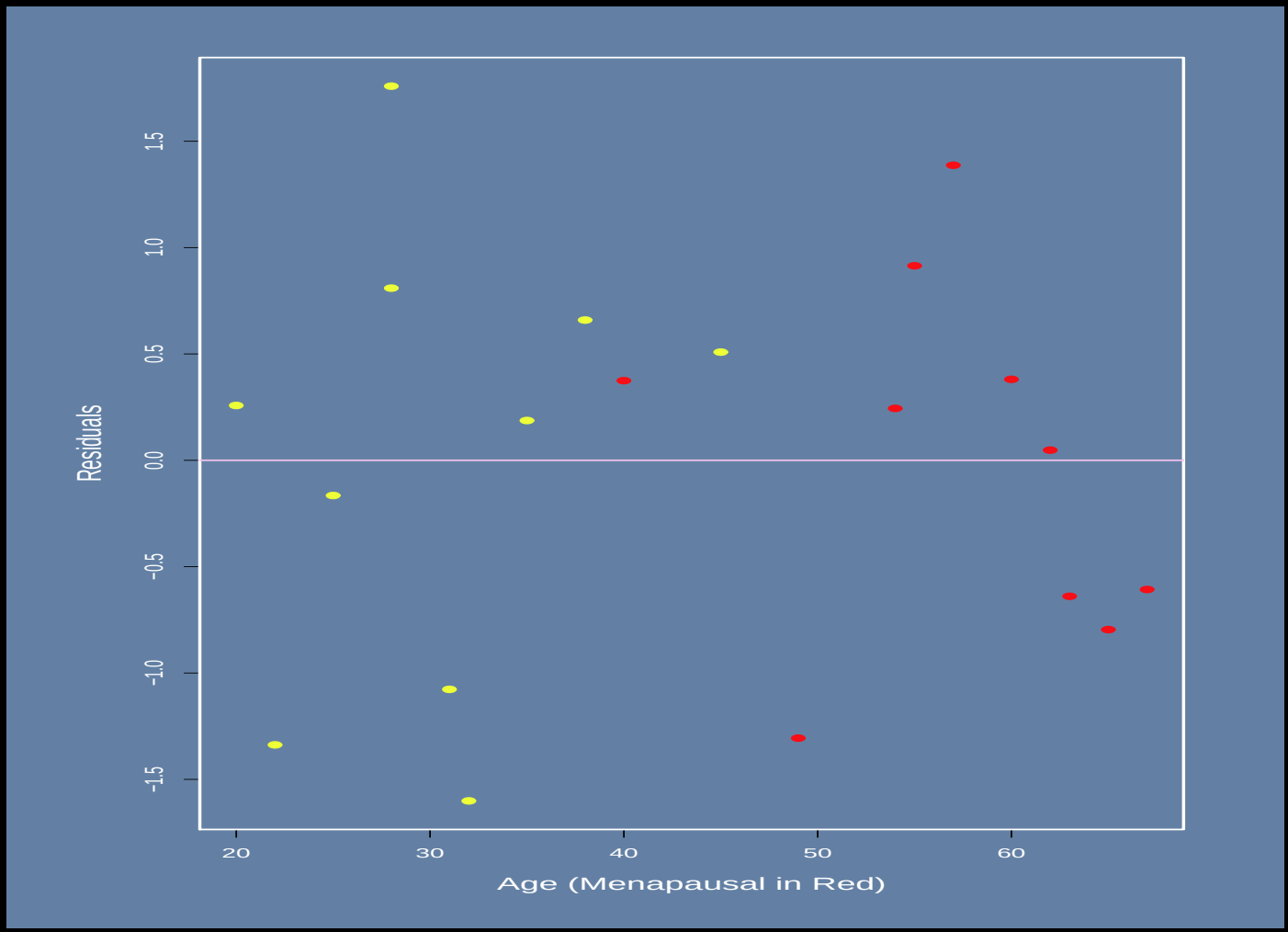
Scatterplot of the Anaemia Data

```
postscript("Class.Med.Stats/Images/anaemia2.fig.ps")
par(mfrow=c(1,1),mar=c(5,5,2,2),lwd=2,col.axis="white",col.lab="white",
    col.sub="white",col="white",bg="slategray", cex.lab=1.3)
plot(anaemia$PCV[anaemia$Menapause==0],anaemia$Hb[anaemia$Menapause==0],pch=19,
     col="yellow", xlim=range(anaemia$Age),ylim=range(anaemia$Hb),
     xlab="PCV (Menapausal in Red)",ylab="Hb(g/dl)")
points(anaemia$PCV[anaemia$Menapause==1],anaemia$Hb[anaemia$Menapause==1],pch=19,
       col="red")
dev.off()
```

Linear Model of Anaemia

```
anaemia.lm <- lm(Hb ~ PCV + Age + Menapause, data=anaemia)
summary(anaemia.lm)
Residuals:
   Min     1Q Median     3Q    Max
-1.601 -0.678  0.216  0.546  1.759

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5.2146     1.5718    3.32   0.0044
PCV           0.0973     0.0346    2.81   0.0125
Age           0.1110     0.0303    3.66   0.0021
Menapause    -0.0241     0.9540   -0.03   0.9802

Residual standard error: 1.01 on 16 degrees of freedom
Multiple R-squared: 0.851,   Adjusted R-squared: 0.823
F-statistic: 30.5 on 3 and 16 DF,  p-value: 7.46e-07
```

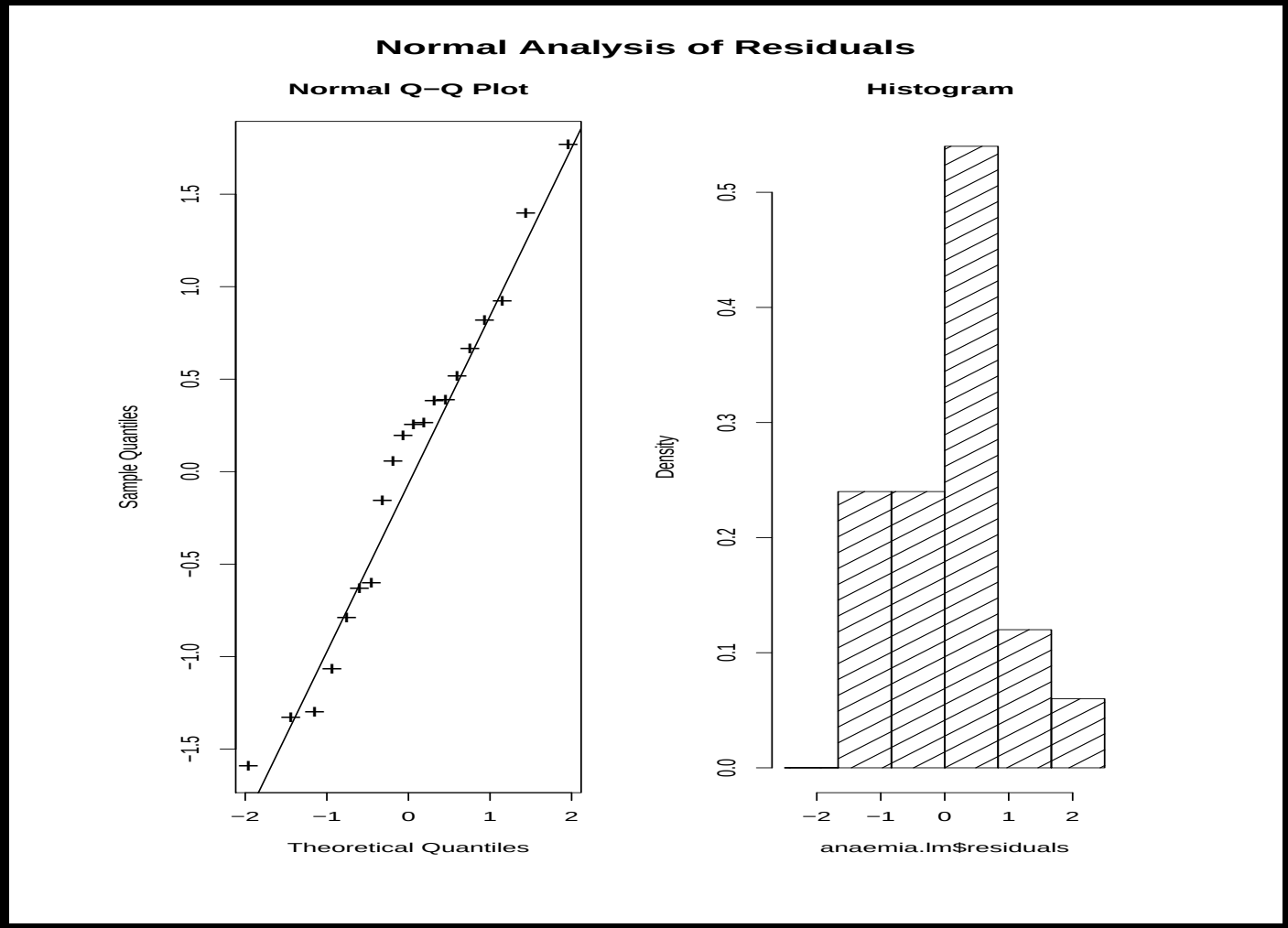
Trends In The Residuals



Trends In The Residuals

```
postscript("Class.PreMed.Stats/Images/anaemia.resids.fig.ps")
par(mfrow=c(1,1),mar=c(5,5,2,2),lwd=2,col.axis="white",col.lab="white",
    col.sub="white",col="white",bg="slategray", cex.lab=1.3)
plot(anaemia$Age[anaemia$Menapause==0],anaemia.lm$residuals[anaemia$Menapause==0],
     pch=19,col="yellow", xlim=range(anaemia$Age),ylim=range(anaemia.lm$residuals),
     xlab="Age (Menapausal in Red)",ylab="Residuals")
points(anaemia$Age[anaemia$Menapause==1],anaemia.lm$residuals[anaemia$Menapause==1],
       pch=19,col="red")
abline(h=0,lwd=2,col="pink")
dev.off()
```

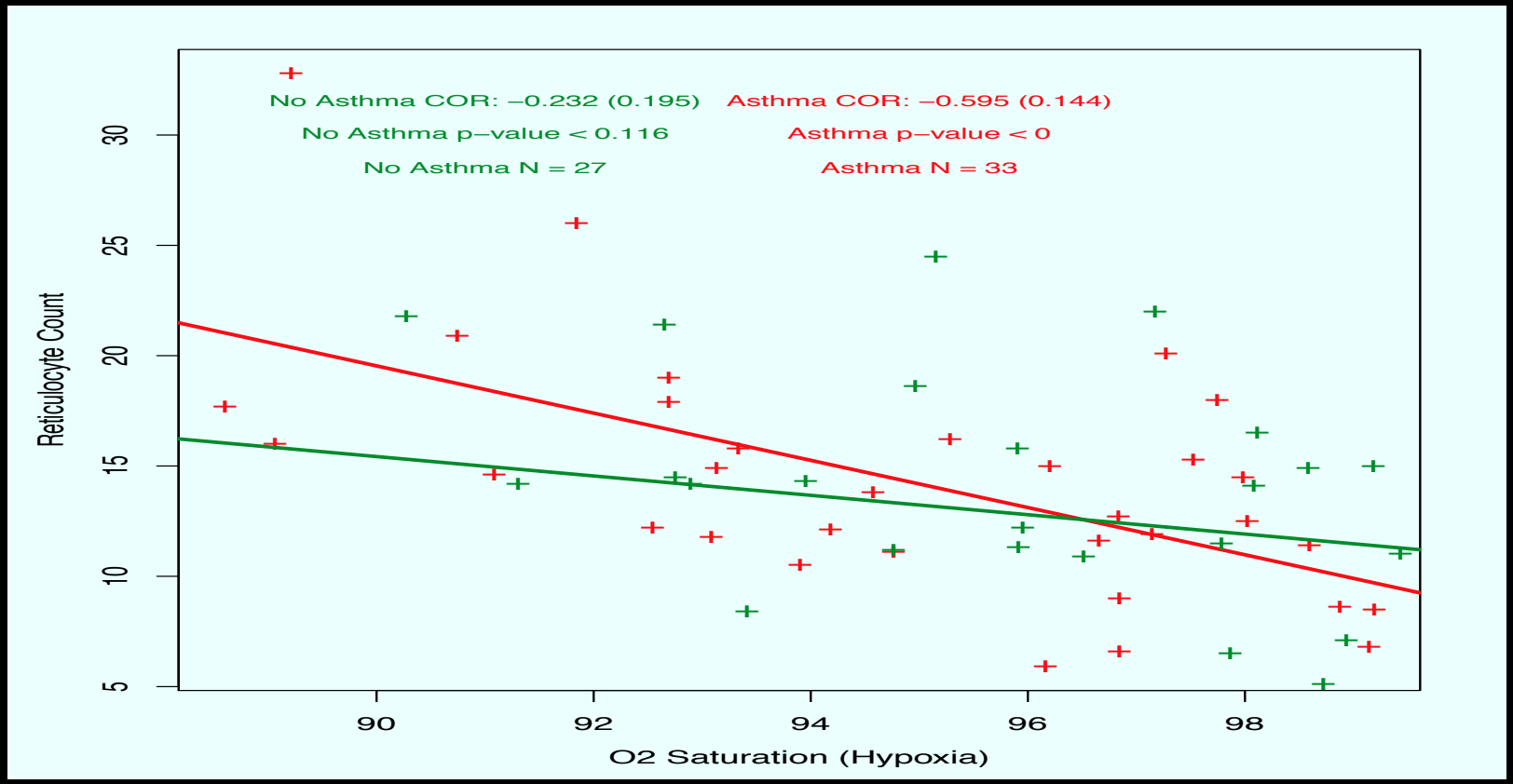
Normality Of The Residuals



Normality Of The Residuals

```
postscript("Class.Stat.Comp/Images/anaemia.qqnorm.ps")
par(mfrow=c(1,2),oma=c(2,2,2,2),bg="white")
qqnorm(anaemia.lm$residuals,pch="+",cex=1.5)
qqline(anaemia.lm$residuals)
hist(anaemia.lm$residuals,breaks=seq(-2.5,2.5,length=7),freq=FALSE,density=7,
     main="Histogram ")
mtext(side=3,font=2,cex=1.5,"Normal Analysis of Residuals",outer=TRUE)
dev.off()
```

Another Regression Plot



Plotting Two Regression Lines At Once

```
pdf("Article.Sickle.Cell/linear.plot.asthma.pdf")
par(mfrow=c(1,1),mar=c(4,4,2,2),col.axis="black",col.lab="black",col.sub="black",
    col="black",bg="white")
plot(sickle1$sac2.satmeantst,sickle1$reticulocyte,pch="+",xlab="",ylab="",
    cex=1.25,col="red")
mtext(side=1,line=2.5,"O2 Saturation (Hypoxia)"); mtext(side=2,line=2.5,
    "Reticulocyte Count")
abline(lm(sickle1$reticulocyte ~ sickle1$sac2.satmeantst),col="red",lwd=3)
sickle.cor <- cor(sickle1$reticulocyte, sickle1$sac2.satmeantst)
se.cor <- sqrt((1-sickle.cor^2)/(nrow(sickle1)-2))
p.cor <- pnorm(sickle.cor/se.cor)
text(95,31.5,paste("Asthma COR: ",round(sickle.cor,3),"
    (",round(se.cor,3),")",sep=""),cex=0.8,col="red")
text(95,30,paste("Asthma p-value < ",round(p.cor,3),sep=""),cex=0.8,col="red")
text(95,28.5,paste("Asthma N = ",nrow(sickle1),sep=""),cex=0.8,col="red")
```

Plotting Two Regression Lines At Once

```
points(sickle0$sac2.satmeantst,sickle0$reticulocyte,pch="+",xlab="",ylab="",
    cex=1.25,col="forestgreen")
abline(lm(sickle0$reticulocyte ~ sickle0$sac2.satmeantst),col="forestgreen",lwd=3)
sickle.cor <- cor(sickle0$reticulocyte, sickle0$sac2.satmeantst)
se.cor <- sqrt((1-sickle.cor^2)/(nrow(sickle0)-2))
p.cor <- pnorm(sickle.cor/se.cor)
text(91,31.5,paste("No Asthma COR: ",round(sickle.cor,3),"
    (",round(se.cor,3),")",sep=""),cex=0.8,col="forestgreen")
text(91,30,paste("No Asthma p-value < ",round(p.cor,3),sep=""),cex=0.8,
    col="forestgreen")
text(91,28.5,paste("No Asthma N = ",nrow(sickle0),sep=""),cex=0.8,col="forestgreen")
dev.off()
```

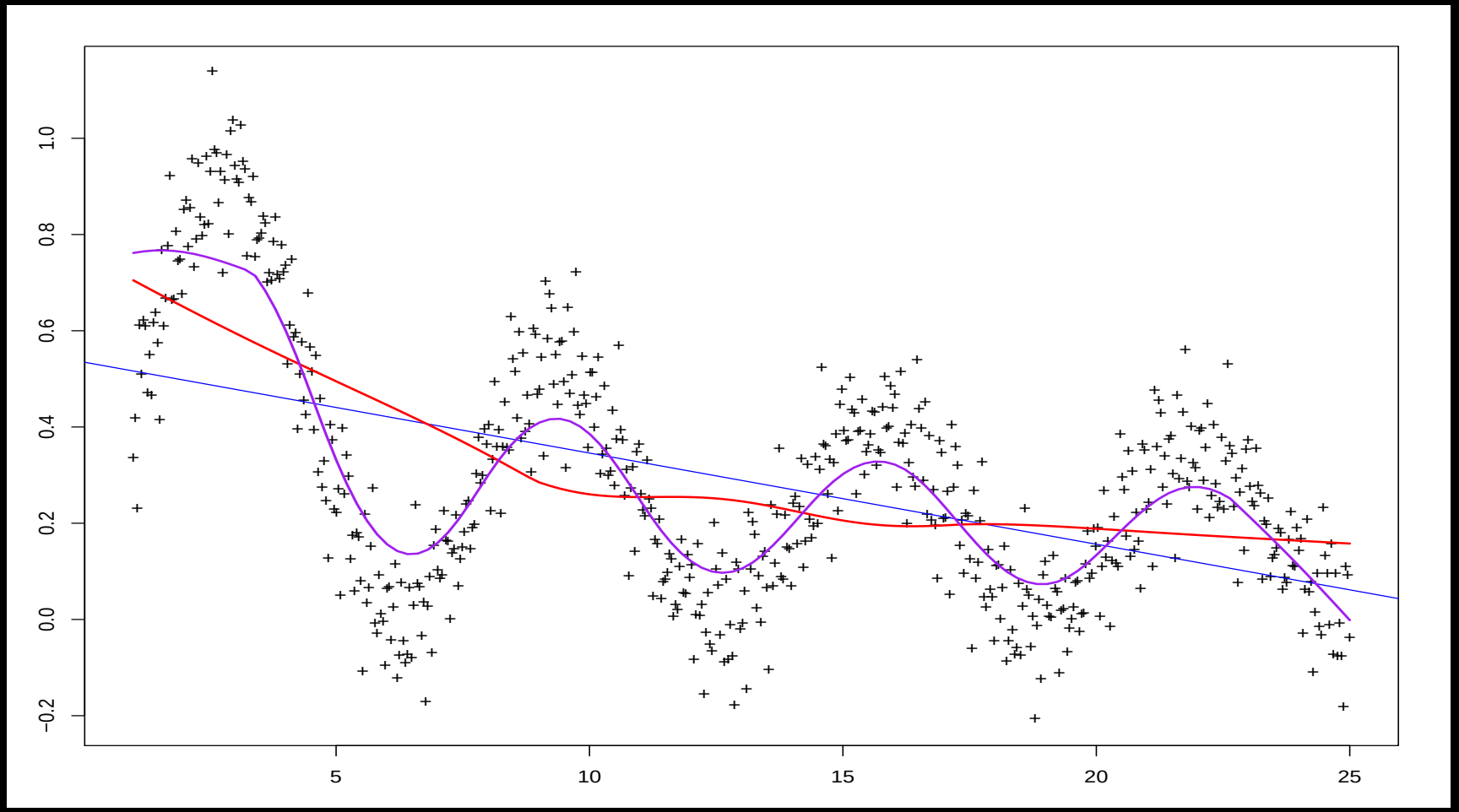
Following Trends

- Sometimes trends are obvious and easy to follow in data, but often they are not.
- Two standard tools: smoothing and linear regression.
- Usually one *or* the other is appropriate.
- Smoothers simply follow the trends in the data, with a given smoothing parameter.
- Main smoother: lowess, "locally weighted running line smoother."

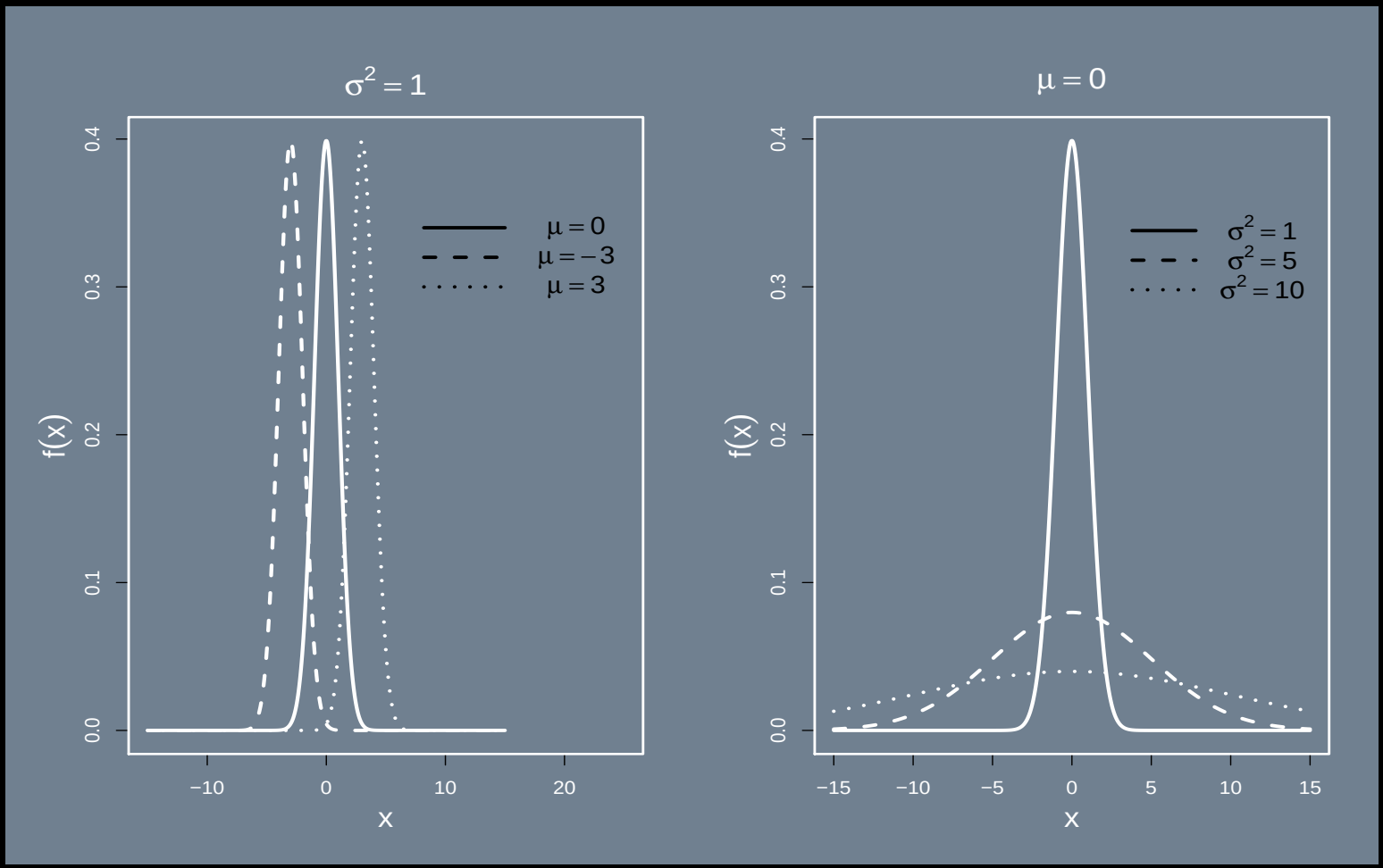
Running Lowess

```
x <- seq(1,25,length=600)
y <- (2/(pi*x))^(0.5)*(1-cos(x)) + rnorm(100,0,1/10)
plot(x,y,pch="+")
ols.object <- lm(y~x)
abline(ols.object,col="blue")
lo.object <- lowess(y~x,f=2/3)
lines(lo.object$x,lo.object$y,lwd=2,col="red")
lo.object <- lowess(y~x,f=1/5)
lines(lo.object$x,lo.object$y,lwd=2,col="purple")
```

Running Lowess



Normal Distribution Forms



Normal Distribution Graphing Different Forms

```
# SETUP DATA STRUCTURES
ruler <- seq(-15,15,length=600)
norm1 <- dnorm(ruler,-3,1)
norm2 <- dnorm(ruler,0,1)
norm3 <- dnorm(ruler,3,1)
norm4 <- dnorm(ruler,0,1)
norm5 <- dnorm(ruler,0,5)
norm6 <- dnorm(ruler,0,10)

# FIX THE GRAPHING PARAMETERS
postscript("Class.PreMed.Stats/Images/normal.plot.ps")
par(mfrow=c(1,2),mar=c(5,5,5,2),lwd=2,col.axis="white",col.lab="white",
    col.sub="white",col="white",bg="slategray")
```

Normal Distribution Graphing Different Forms

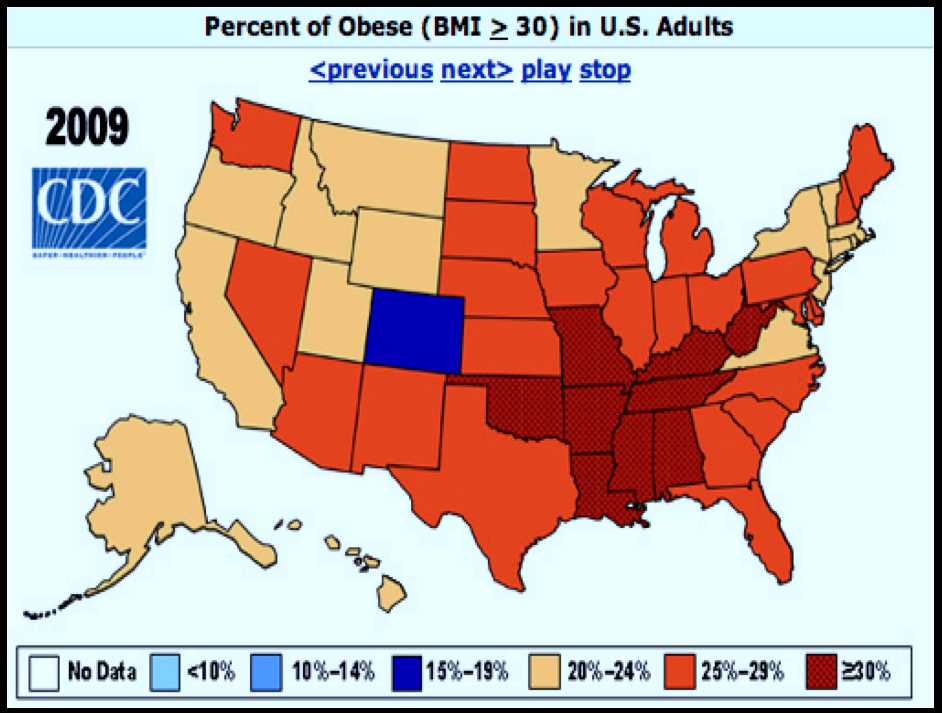
```
# PLOT FIRST DISTRIBUTION
plot(ruler,norm1,type="l",xlab="",ylab="",lty=2,xlim=c(-15,25),lwd=3)

# ADD X AND Y LABELS
mtext(outer=FALSE,side=1,cex=1.5,line=2.3,expression(x))
mtext(outer=FALSE,side=2,cex=1.5,line=2.3,expression(f(x)))
mtext(outer=FALSE,side=3,cex=1.6,line=1,expression(sigma^2==1))

# ADD SECOND AND THIRD DISTRIBUTIONS
lines(ruler,norm2,type="l",lty=1,lwd=3)
lines(ruler,norm3,type="l",lty=3,lwd=3)

# INCLUDE ILLUSTRATIVE SEGMENTS AND TEXT‘
segments(8.2,0.34,15,0.34,lty=1,lwd=3); text(21.0,0.34,expression(mu==0),cex=1.3,
    col="black")
segments(8.2,0.32,15,0.32,lty=2,lwd=3); text(21.0,0.32,expression(mu==-3),cex=1.3,
    col="black")
segments(8.2,0.30,15,0.30,lty=3,lwd=3); text(21.0,0.30,expression(mu==3),cex=1.3,
    col="black")
```

Normal Distribution Graphing Different Forms

```
# SAME STEPS FOR SECOND PANEL THAT VARIES SIGMA INSTEAD OF MU
plot(ruler,norm4,type="l",xlab="",ylab="",lty=1,lwd=3)
mtext(outer=FALSE,side=1,cex=1.5,line=2.3,expression(x))
mtext(outer=FALSE,side=2,cex=1.5,line=2.3,expression(f(x)))
mtext(outer=FALSE,side=3,cex=1.6,line=1,expression(mu==0))
lines(ruler,norm5,type="l",lty=2,lwd=3)
lines(ruler,norm6,type="l",lty=3,lwd=3)
segments(3.8,0.338,7.8,0.338,lty=1,lwd=3); text(12.0,0.34,expression(sigma^2==1),
         cex=1.3, col="black")
segments(3.8,0.318,7.8,0.318,lty=2,lwd=3); text(12.0,0.32,expression(sigma^2==5),
         cex=1.3, col="black")
segments(3.8,0.298,7.8,0.298,lty=3,lwd=3); text(12.0,0.30,expression(sigma^2==10),
         cex=1.3, col="black")
dev.off()
```

2009 BMI CDC Data by US State

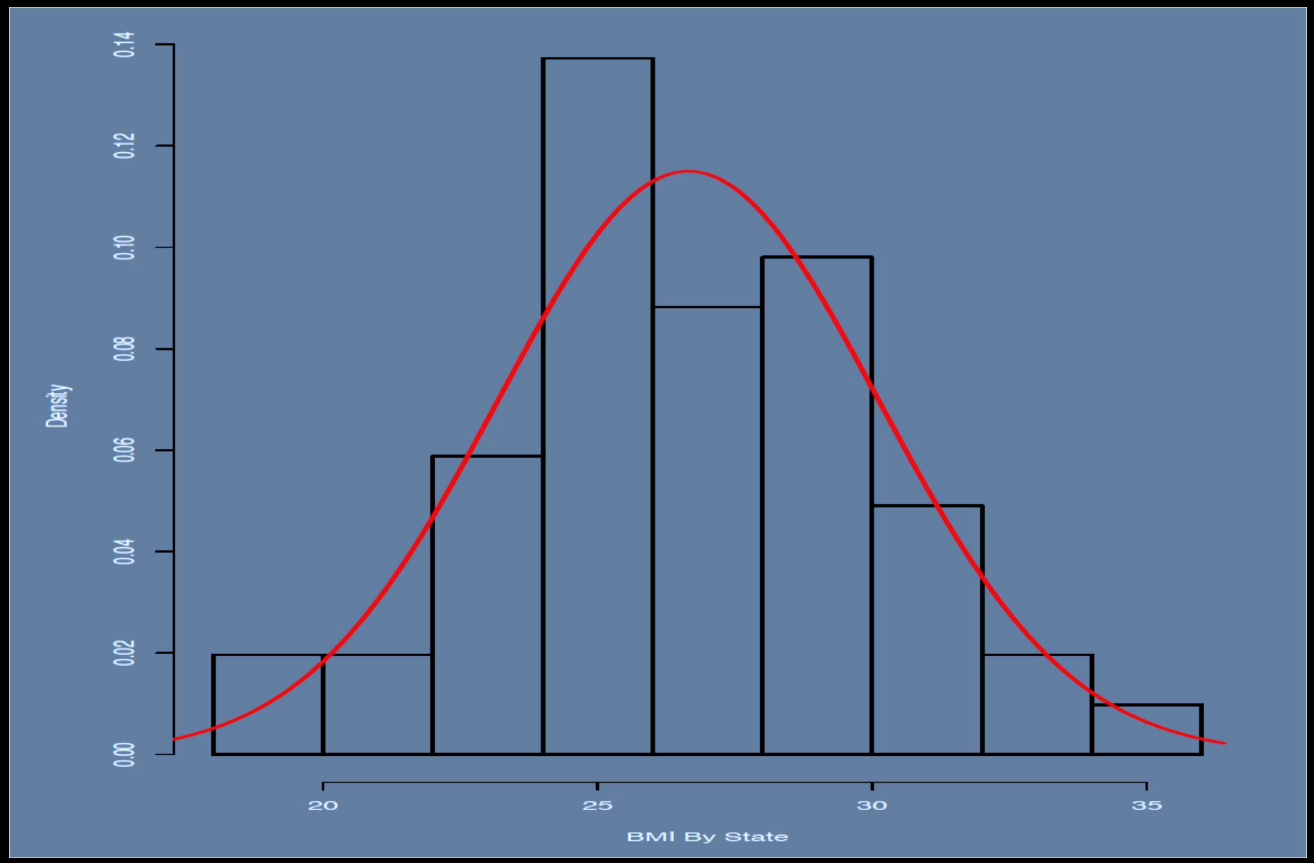
- $BMI = [kilograms]/[meters]^2 = [4.88 \times pounds]/[feet]^2$.
- Obesity is defined as a body mass index of 30 or greater.
- http://jgill.wustl.edu/data/2009.bmi.data



2009 BMI CDC Data by US State

Alabama	31.0	Alaska	24.8	Arizona	25.5	Arkansas	30.5
California	24.8	Colorado	18.6	Connecticut	20.6	Delaware	27.0
Florida	25.2	Georgia	27.2	Hawaii	22.3	Idaho	24.5
Illinois	26.5	Indiana	29.5	Iowa	27.9	Kansas	28.1
Kentucky	31.5	Louisiana	33.0	Maine	25.8	Maryland	26.2
Massachusetts	21.4	Michigan	29.6	Minnesota	24.6	Mississippi	34.4
Missouri	30.0	Montana	23.2	Nebraska	27.2	Nevada	25.8
New.Hampshire	25.7	New.Jersey	23.3	New.Mexico	25.1	New.York	24.2
North.Carolina	29.3	North.Dakota	27.9	Ohio	28.8	Oklahoma	31.4
Oregon	23.0	Pennsylvania	27.4	Rhode.Island	24.6	South.Carolina	29.4
South.Dakota	29.6	Tennessee	32.3	Texas	28.7	Utah	23.5
Vermont	22.8	Virginia	25.0	Washington	26.4	Washington.DC	19.7
West.Virginia	31.1	Wisconsin	28.7	Wyoming	24.6		

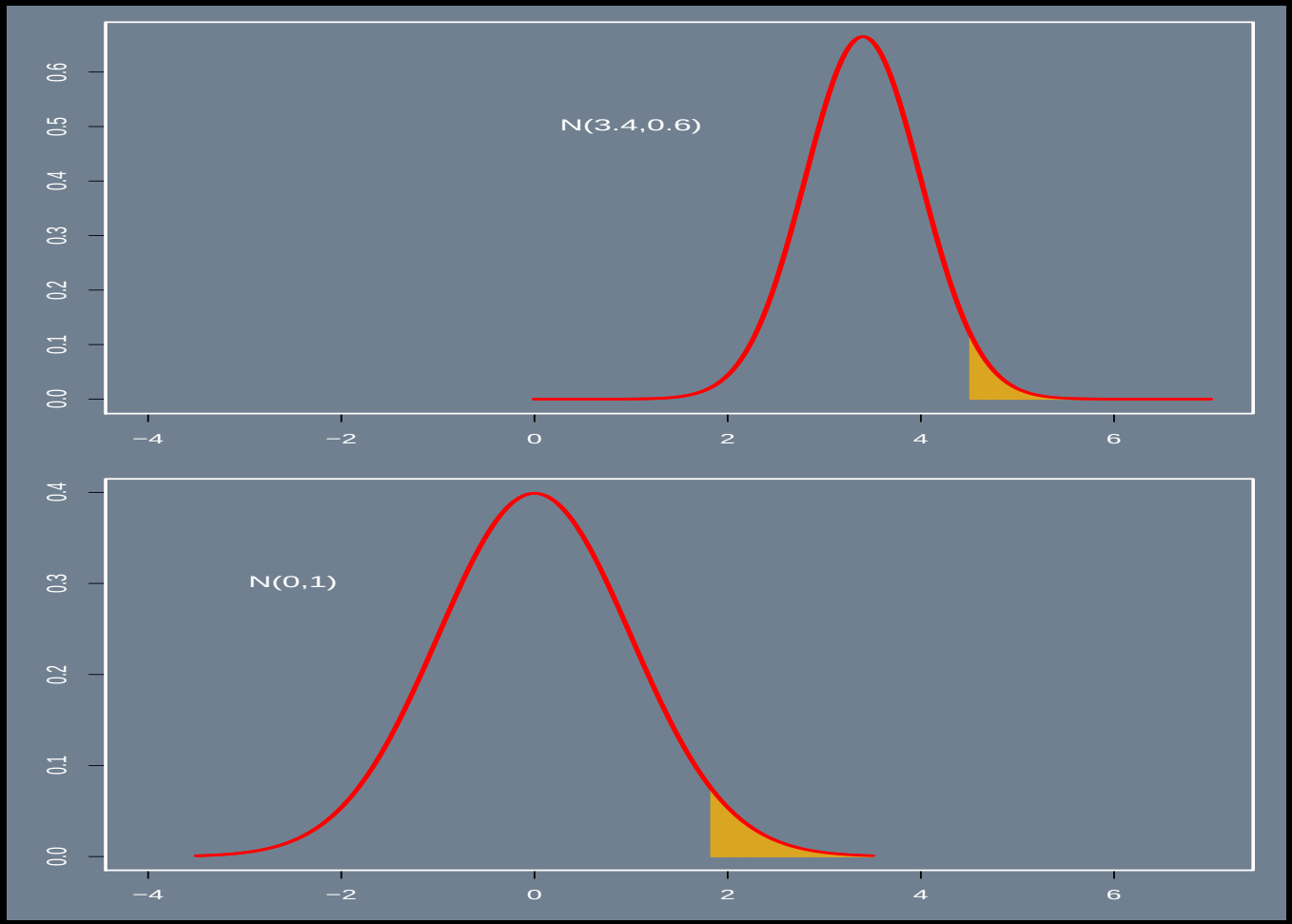
Normality of BMI Means



2009 BMI CDC Data by US State

```
bmi <- read.table("http://jgill.wustl.edu/data/2009.bmi.data", header=FALSE,
                  row.names=1)
postscript("Class.Med.Stats/Images/bmi.hist.ps")
par(mfrow=c(1,1),mar=c(5,5,2,2),lwd=2,col.axis="white",col.lab="white",
    col.sub="white",col="white",bg="slategray")
hist(bmi$V2,breaks=10,prob=TRUE,xlab="BMI By State", ylab="Density",
     main="")
lines(seq(min(bmi)-2,max(bmi)+2,length=100),
      dnorm(seq(min(bmi)-2,max(bmi)+2,length=100), mean(bmi$V2),sd(bmi$V2)),lwd=3,
      col="red")
dev.off()
```

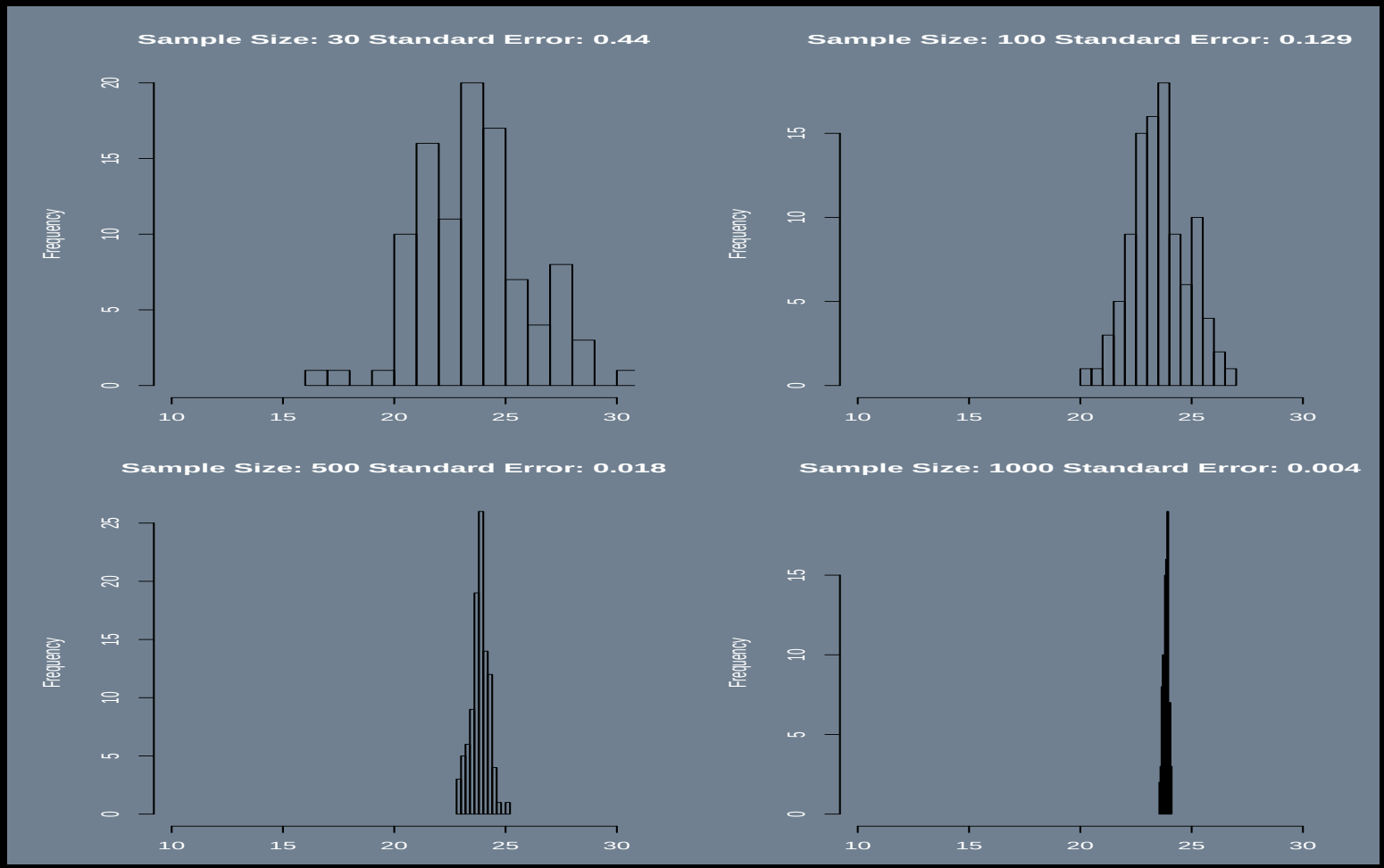
Standardized Normal Illustration



Standardized Normal Illustration

```
postscript("Class.Med.Stats/Images/standard.norm.ps")
par(mfrow=c(2,1),mar=c(3,3,1,1),lwd=2,col.axis="white",col.lab="white",
    col.sub="white",col="white",bg="slategray")
ruler <- seq(0,7,length=300)
plot(ruler, dnorm(ruler,3.4,0.6),lwd=3, col="red",type="l",xlab="",ylab="",
     xlim=c(-4,7))
lines(ruler[ ruler > 4.5 ],
      dnorm(ruler[ ruler > 4.5 ],3.4,0.6), col="goldenrod",type="h")
lines(ruler, dnorm(ruler,3.4,0.6),lwd=3, col="red",type="l",xlab="",ylab="Density")
text(1,0.5,"N(3.4,0.6)",col="white",cex=1.2)
ruler <- seq(-3.5,3.5,length=300)
plot(ruler, dnorm(ruler,0,1),lwd=3, col="red",type="l",xlab="",ylab="Density",
     xlim=c(-4,7))
lines(ruler[ ruler > 1.8333 ],
      dnorm(ruler[ ruler > 1.8333 ],0,1), col="goldenrod",type="h")
lines(ruler, dnorm(ruler,0,1),lwd=3, col="red",type="l",xlab="",ylab="Density")
text(-2.5,0.3,"N(0,1)",col="white",cex=1.2)
dev.off()
```

HIV Counts in South African Children, Histogram of Means from 100 Draws



HIV Counts in South African Children, Histogram of Means from 100 Draws

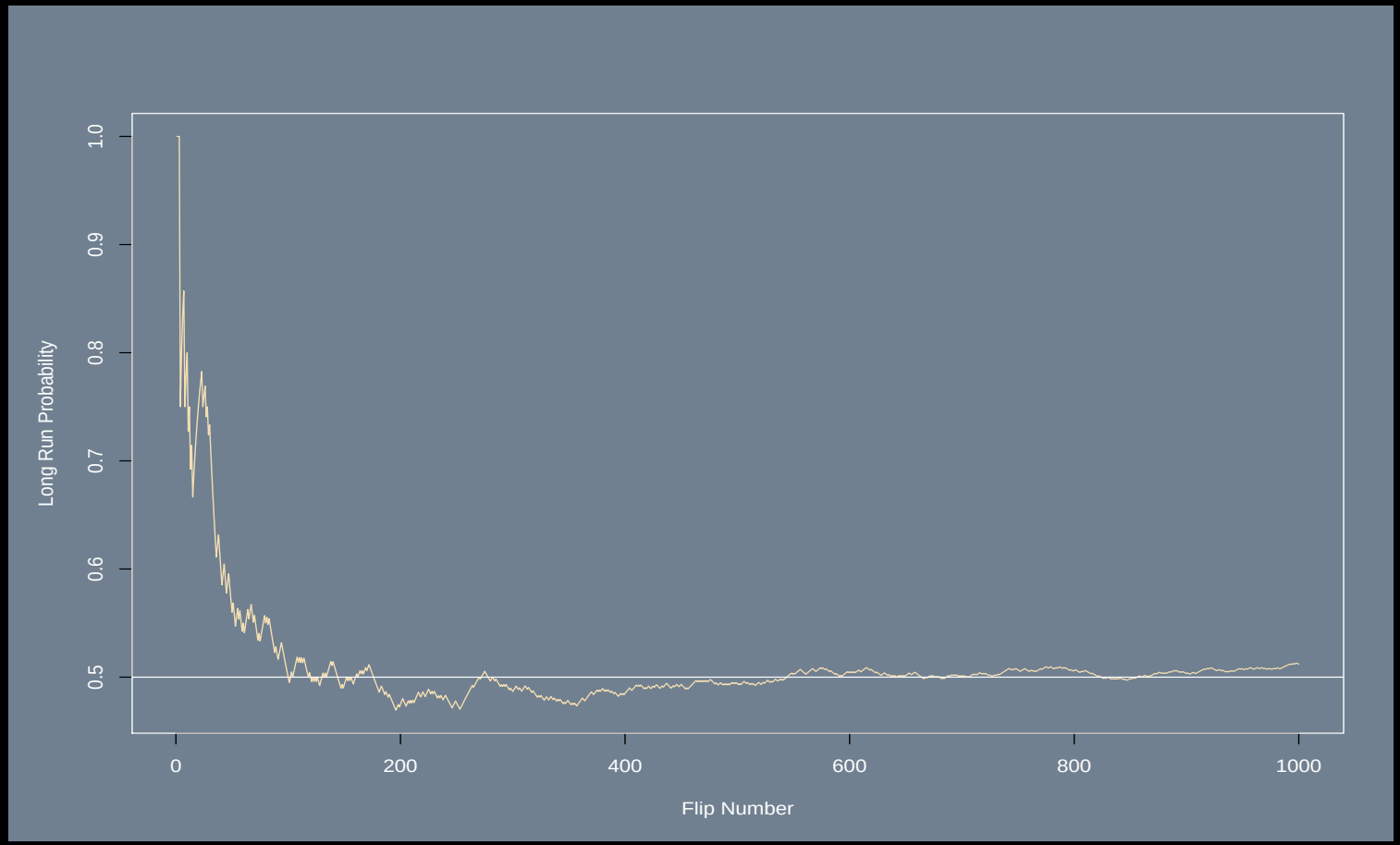
```
hiv.data <- read.table("Class.Multilevel/examples/cd4/allvar.csv",sep=",",
                       header=TRUE)
ok <- !is.na(hiv.data$CD4PCT); y <- hiv.data$CD4PCT[ok]
sample.size <- c(30,100,500,1000)
sample.100 <- rep(NA,length=100)
postscript("Class.Med.Stats/Images/hiv.means.ps")
par(bg="slategray",mfrow=c(2,2),mar=c(3,5,5,2),col.main="white",col.axis="white",
        col="white",col.lab="white")
for (i in 1:4)  {
    for (j in 1:100) sample.100[j] <- mean(sample(y,sample.size[i]))
    hist(sample.100,main=paste("Sample Size:",sample.size[i],"Standard Error:",
         round(sd(sample.100)/sqrt(sample.size[i]),3)),
               breaks=15,xlim=c(10,30), xlab="")
}
dev.off()
```

Illustrating The Law of Large Numbers

- Suppose we sample $X_1, X_2, \ldots, X_n$ such that they are iid from a certain population with finite mean and variance (does not have to be normal).
- Then as n gets big, $\bar{X}$ converges to μ, the true population mean.

```
x <- sample(0:1,1000,replace=TRUE,prob=c(0.5,0.5))
y <- cumsum(x)/seq(1:length(x))
postscript("Class.Med.Stats/Images/law.of.large.nums.ps")
par(bg="slategray",mfrow=c(1,1),mar=c(5,5,5,2),col.main="white",col.axis="white",
    col="white",col.lab="white")
x <- sample(0:1,1000,replace=TRUE,prob=c(0.5,0.5))
y <- cumsum(x)/seq(1:length(x))
ts.plot(y,col="wheat",xlab="Flip Number",ylab="Long Run Probability")
abline(h=0.5)
dev.off()
```

Illustrating The Law of Large Numbers

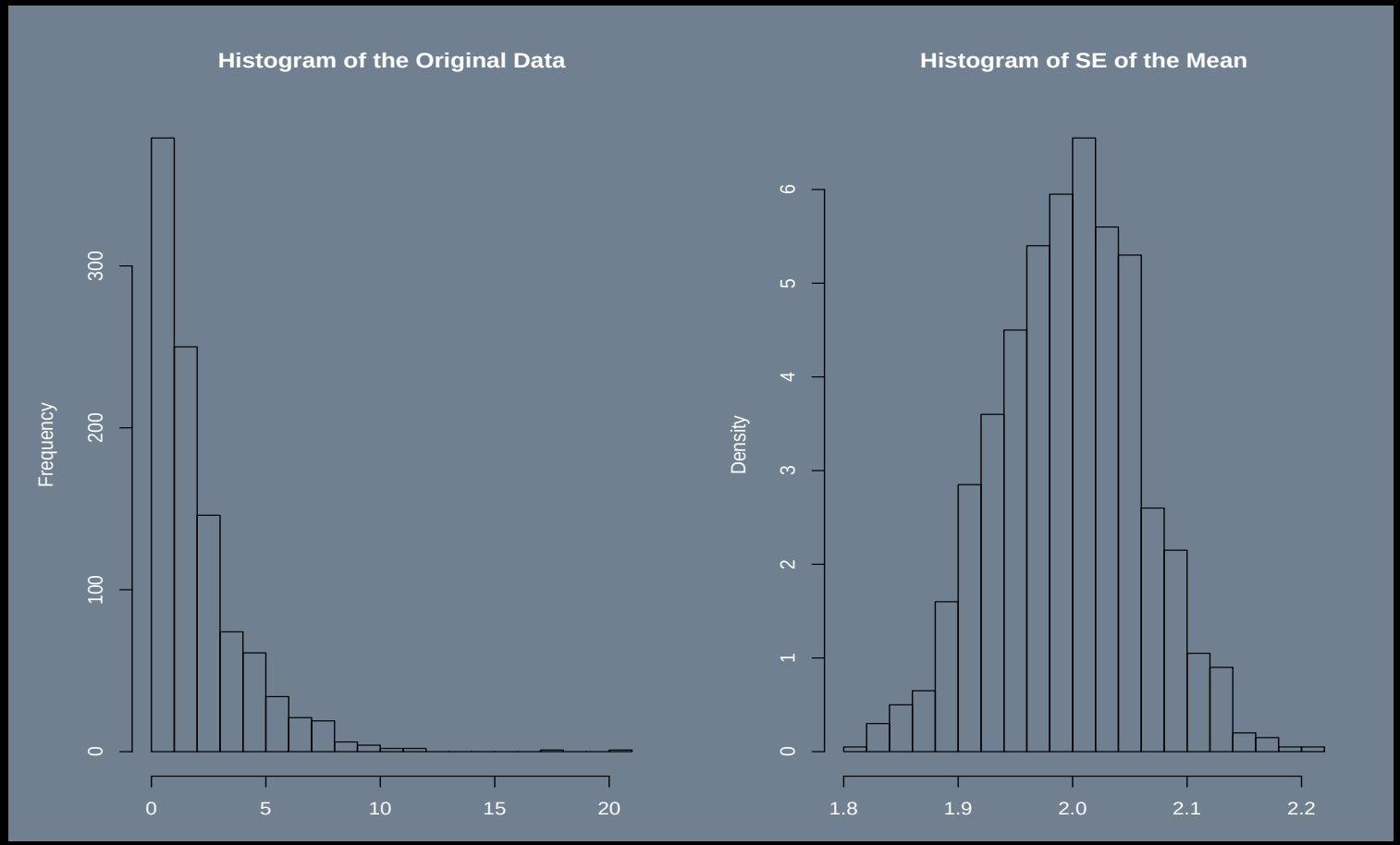


Illustrating The Central Limit Theorem

- Suppose $X_1, X_2, \ldots, X_n$ iid from a certain population with *finite mean and variance.*
- As n gets "sufficiently big" the sampling distribution of the mean, $\bar{X}$, converges to a normal form with mean μ and standard error $\sigma/\sqrt{n}$, regardless of the distribution of the data.

```
par(bg="slategray",mfrow=c(1,2),mar=c(3,5,5,2),col.main="white",col.axis="white",
        col="white",col.lab="white")
y <- rchisq(1000,2)
hist(y,breaks=15,main="Histogram of the Original Data")
x <- matrix(NA,nrow=1000,ncol=1000)
for (i in 1:nrow(x))  x[i,] <- rchisq(1000,2)
hist(apply(x,1,mean),breaks=15,probability=TRUE,main="Histogram of SE of the Mean")
```

Illustrating The Central Limit Theorem

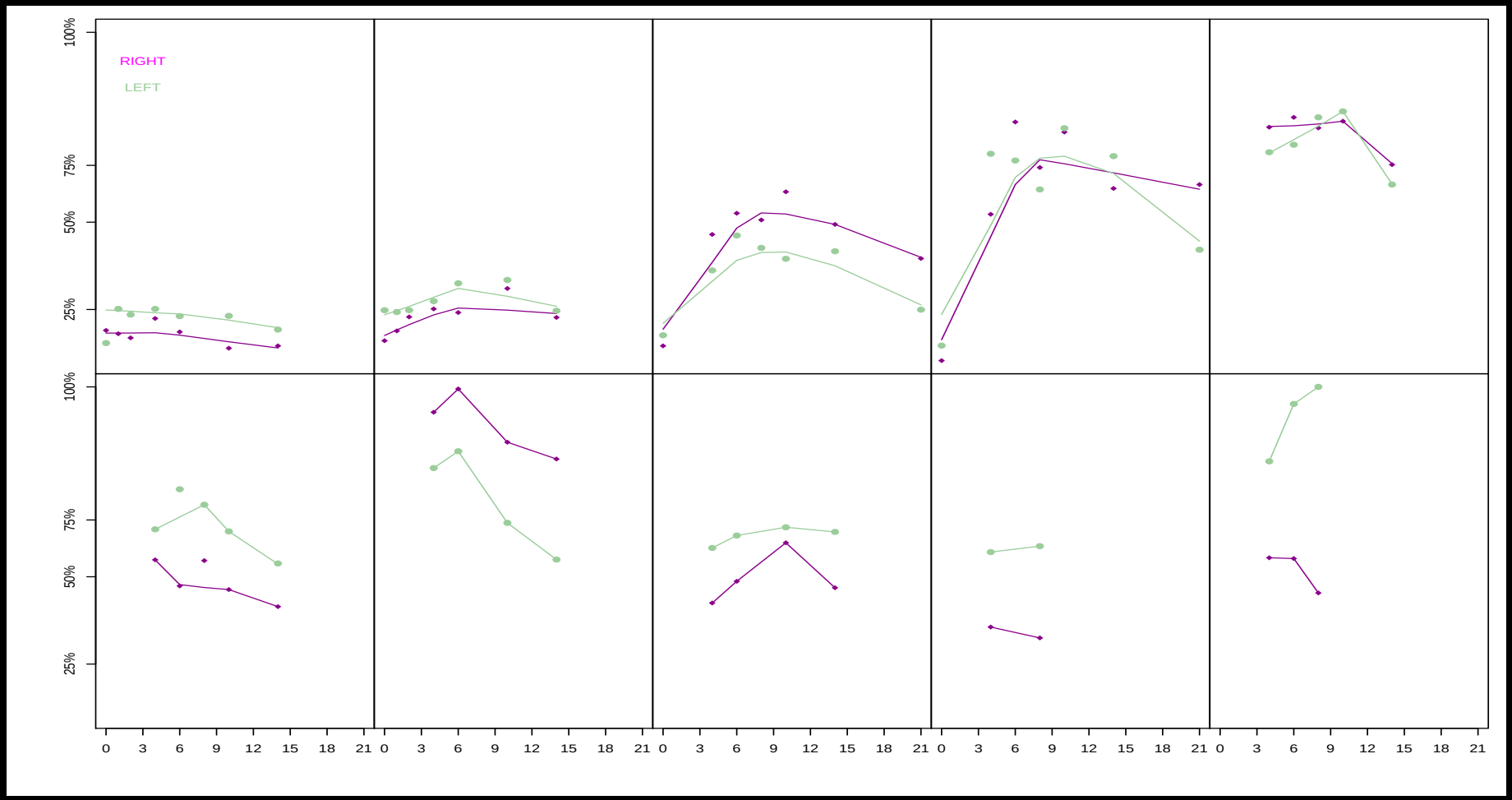


Using Lowess for Nested Classification Factors

- Experiment recording mean pixel intensity of the right and left lymphnodes in the axillary region from (tomography) CT scans of 10 dogs over 14 days after intravenous application of a dye contrast.
- Data:

```
connect1 <- url("http://jgill.wustl.edu/data/Pixel.rda")
load(connect1);   close(connect1)
Pixel[,"Side"] <- as.numeric( Pixel[,"Side"] )
Pixel[1:10,]
   Case Side day  pixel
1     1    2   0 1045.8
2     1    2   1 1044.5
3     1    2   2 1042.9
4     1    2   4 1050.4
5     1    2   6 1045.2
6     1    2  10 1038.9
7     1    2  14 1039.8
8     2    2   0 1041.8
9     2    2   1 1045.6
10    2    2   2 1051.0
```

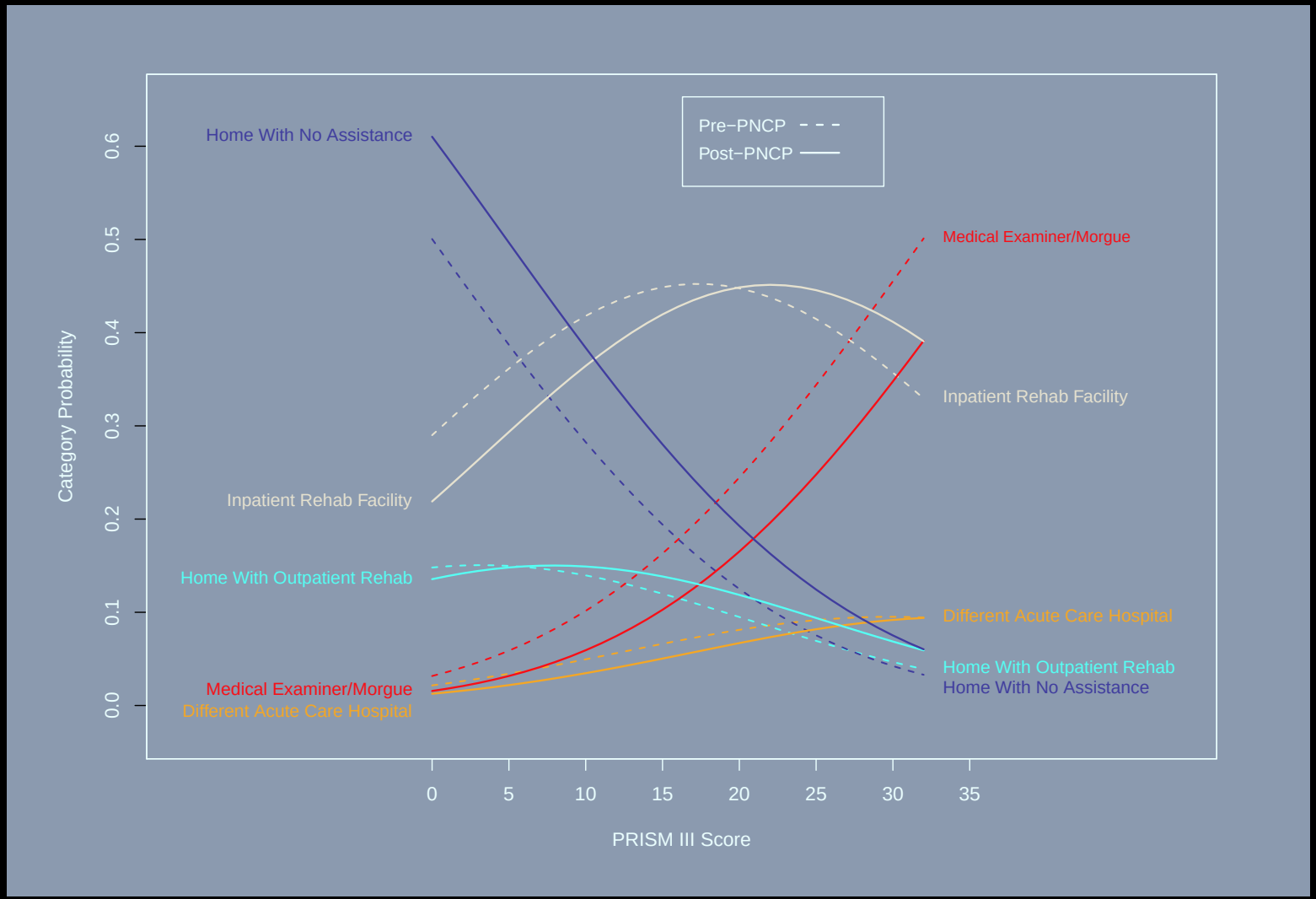
Using Lowess for Nested Classification Factors



Using Lowess for Nested Classification Factors

```
postscript("Class.Multilevel/patients.ps")
J = max(as.numeric(Pixel$Person))
day.range <- c(min(as.numeric(Pixel$day)), max(as.numeric(Pixel$day)))
pixel.range <- c(min(as.numeric(Pixel$pixel)), max(as.numeric(Pixel$pixel)))
par(oma=c(5,5,1,1),mar=c(0,0,0,0),mfrow=c(J/(J/2),J/2),bg="white")
for (i in 1:J)  {
    patient.i <- Pixel[Pixel["Dog"]==i,]
    plot(patient.i[patient.i["Side"]==2,][,3:4], xlim=day.range, ylim=pixel.range,
         pch=18,col="darkmagenta",yaxt="n",xaxt="n")
    lines(lowess(patient.i[patient.i["Side"]==2,][,3:4], f=0.9), col="darkmagenta")
    points(patient.i[patient.i["Side"]==1,][,3:4], pch=19,col="darkseagreen3")
    lines(lowess(patient.i[patient.i["Side"]==1,][,3:4], f=0.9), col="darkseagreen3")
    if (i > 5) axis(side=1,labels=seq(0,21,by=3),at=seq(0,21,by=3))
    if (i == 1 | i==6) axis(side=2,labels=names(quantile(Pixel$pixel)[-1]),
                            at=quantile(Pixel$pixel)[-1])
    if (i == 1)  { text(3,1150,"RIGHT",col="magenta")
                   text(3,1140,"LEFT",col="darkseagreen3") }
}
dev.off()
```

Pediatric Traumatic Brian Injury



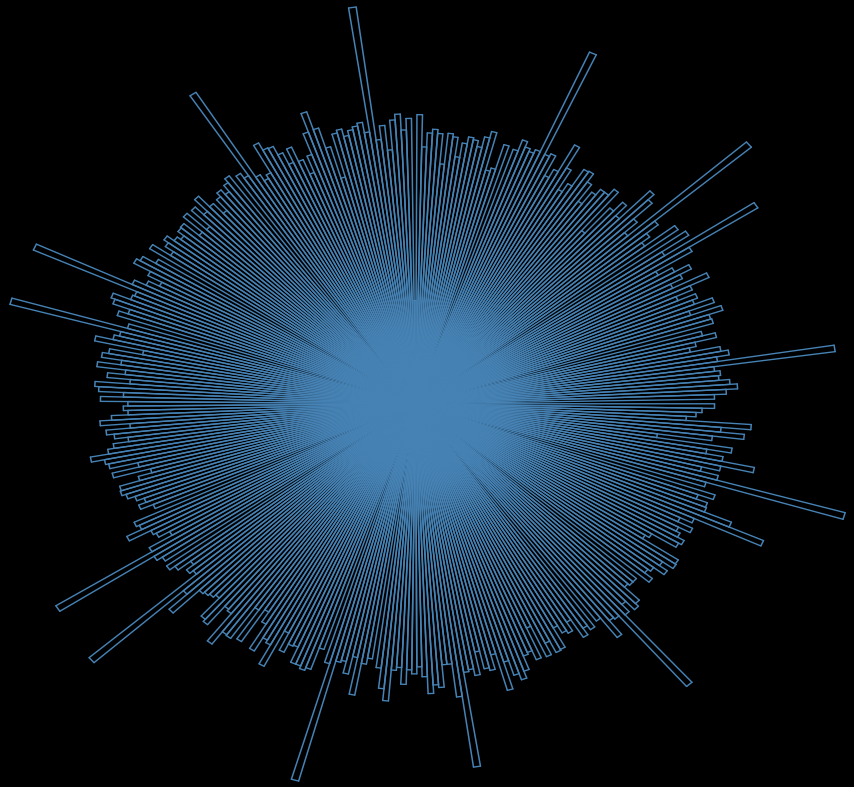
Pediatric Traumatic Brian Injury

```
postscript("Article.PTBI/outcomes.graph.col.ps",horizontal=TRUE)
head.col <- c(552,498,646,418,8,107)
par(mfrow=c(1,1),mar=c(6,6,3,4),cex.axis=1,cex.lab=1.0,
    col.axis="white",col.lab="white",col.sub="white",col="white",bg="grey60")
plot(prism.mat1[,1],type="l",lwd=1.45,col=colors()[head.col[1]],lty=2,
    xlab="PRISM III Score", ylab="Category Probability",xlim=c(-15,49.5),
    ylim=c(-0.03,0.65),xaxt="n")
box(lty = "solid", col = 'white')
axis(side=1,col="white",col.ticks="white",labels=c(0,5,10,15,20,25,30,35),
    at=c(1,6,11,16,21,26,31,36))
text(33.5,prism.mat1[nrow(prism.mat1),1],levels(head.inj.sub1$Destination)[1],
    cex=0.8, col=colors()[head.col[1]],pos=4)
for(i in c(2,3,5,6)) {
    lines(prism.mat1[,i],lwd=1.45,col=colors()[head.col[i]],lty=2)
        if (i != 6) text(33.5,prism.mat1[nrow(prism.mat1),i],
            levels(head.inj.sub1$Destination)[i], cex=0.92,
                col=colors()[head.col[i]],pos=4)
        else text(33.5,(prism.mat1[nrow(prism.mat1),i]-0.015),cex=0.92
            levels(head.inj.sub1$Destination)[i], col=colors()[head.col[i]],pos=4)
}
```

Pediatric Traumatic Brian Injury

```
for(i in c(1,2,3,5,6)) {
      lines(prism.mat2[,i],lwd=1.65,col=colors()[head.col[i]],lty=1)
          if (i != 2)
            text(0.5,prism.mat2[1,i],
                levels(head.inj.sub2$Destination)[i],cex=0.92,
                col=colors()[head.col[i]],pos=2)
          else
            text(0.5,(prism.mat2[1,i]-0.020),
                levels(head.inj.sub2$Destination)[i],cex=0.92,
                col=colors()[head.col[i]],pos=2)
}
legend(17.3,0.653,c("Pre-PNCP","Post -PNCP"),box.col="white",text.col="grey60",
    cex=1.3)
text(17.6,0.62,"Pre-PNCP",col="white",cex=0.9,pos=4)
segments(25.0,0.623,27.5,0.623,col="white",lty=2,lwd=1.65)
text(17.6,0.59,"Post-PNCP",col="white",cex=0.9,pos=4)
segments(25.0,0.593,27.5,0.593,col="white",lty=1,lwd=1.65)
dev.off()
```

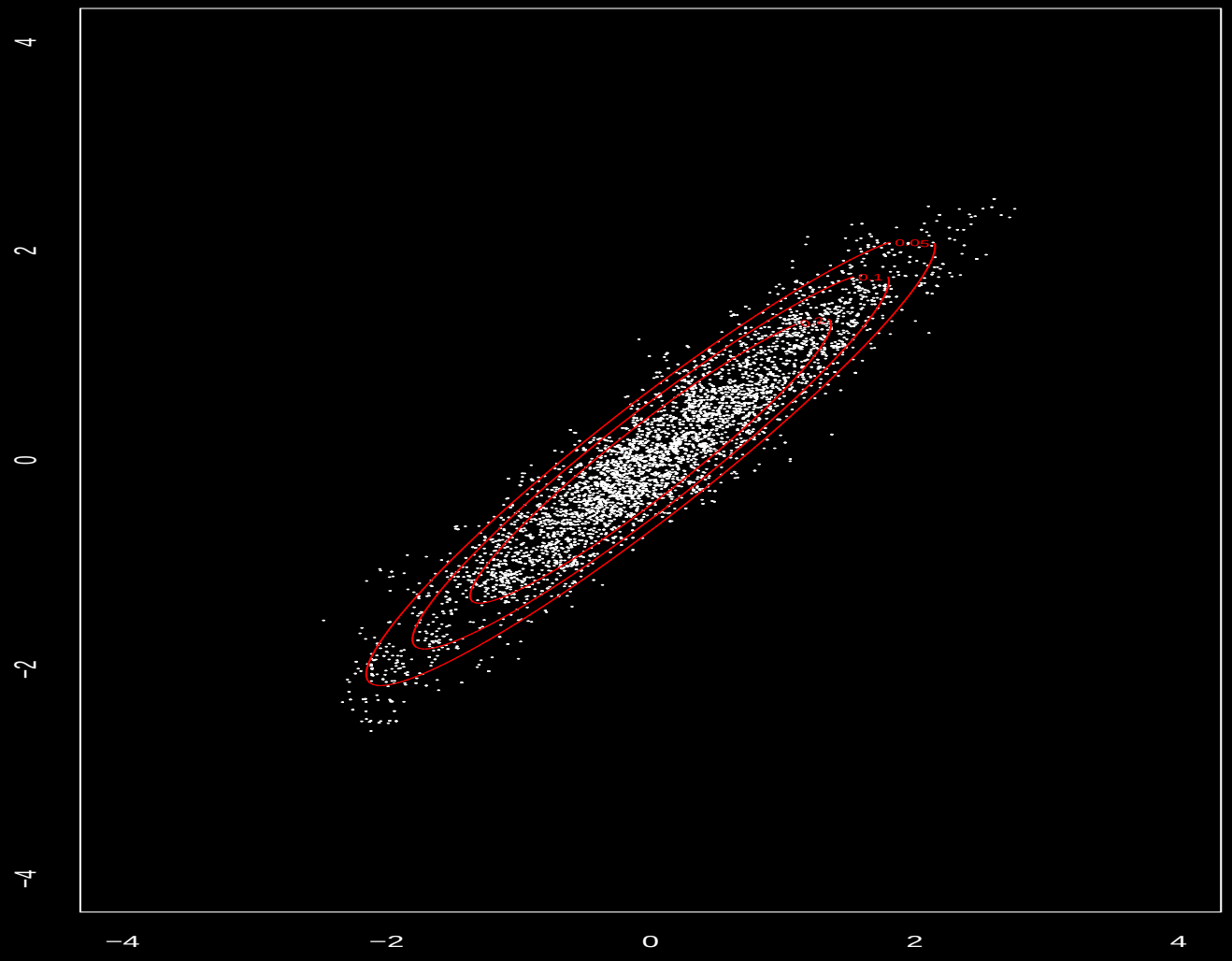
Rose Diagram: 1969-2004 Suicides in Switzerland by Numbered Date, $N = 51025$



Rose Diagram: 1969-2004 Suicides in Switzerland by Numbered Date, $N = 51025$

```
suicide <- read.dta("Article.Circular/suicide.dta")
par(col="steelblue")
rose.diag(suicide$edate, bins=365, prop=2,pts=TRUE,dotsep=20,shrink=0.15,
    col="steelblue")
dev.off()
```

Hit-and-Run Markov Chain Example

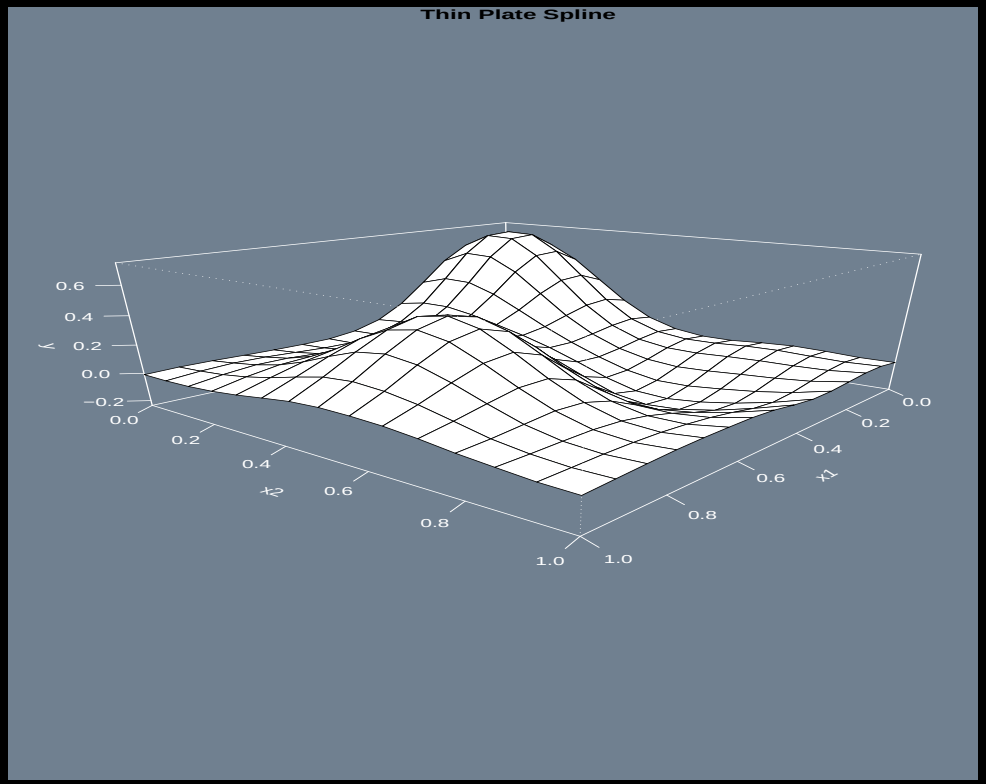


Hit-and-Run Markov Chain Example

```
postscript("Class.Multilevel/hit.run.contours.ps")
par(mfrow=c(1,1),mar=c(3,2,2,2),oma=c(3,3,1,1),col.axis="white",col.lab="white",
    col.sub="white",col="white",bg="black")
plot(mcmc.mat[(nrow(mcmc.mat)/2+1):nrow(mcmc.mat),],pch=1,cex=0.1,
    xlim=c(-4,4),ylim=c(-4,4))

alpha.grid <- seq(-4,4,length=300)
beta.grid <- seq(-4,4,length=300)
density <- outer(alpha.grid,beta.grid,dmultinorm,mu.vector=c(0,0),
    sigma.matrix=matrix(c(1,0.95,0.95,1),ncol=2))
contour(alpha.grid,beta.grid,density,levels=c(.2,.1,.05),col="red",add=TRUE)
dev.off()
```

Next Month

- Regression.



Tabular Analaysis

- Back to the psychology data:

```
   freq anxiety behavioral depression    sex
1     9     Low    Present     Absent   Male
2    32  Medium    Present     Absent   Male
3     4    High    Present     Absent   Male
4     1     Low     Absent     Absent   Male
:
```

- R uses its standard modeling language (discussed in detail in future sessions).
- Basic tables are relatively simple:

```
xtabs(freq ~ depression + sex, data=psych.df)
          sex
depression Male Female
    Absent   58     72
    Mild     52     46
    Severe   33    195
```

Tabular Analaysis

- Here's another, smaller, table:

```
xtabs(freq ~ behavioral + sex, data=psych.df)
          sex
behavioral Male Female
   Present   98    213
   Absent    45    100
```

- We can also add a χ^2 test by wrapping `summary()` around this statement:

```
summary(xtabs(freq ~ behavioral + sex, data=psych.df))
Call: xtabs(formula = freq ~ behavioral + sex, data = psych.df)
Number of cases in table: 456
Number of factors: 2
Test for independence of all factors:
    Chisq = 0.010443, df = 1, p-value = 0.9186
```

Tabular Analaysis

- More complex tables are created with `ftable`:

```
ftable(xtabs(freq ~ depression + behavioral + anxiety, data = psych.df))
                      anxiety Low Medium High
depression behavioral
Absent     Present             18     38   13
           Absent              42     10    9
Mild       Present              8      8   63
           Absent               9      3    7
Severe     Present             32     24  107
           Absent               3     15   47
```