Streamlining Data Analysis, Reporting and Website Publishing Using R

Mike Wallinga
Director of Institutional Research
Northwestern College
mwalling@nwciowa.edu

NORTHWESTERN
COLLEGE

My Previous Fact Book Process

- Cumbersome, with lots of different tools in the chain
  - mostly Excel, some Word, Access, Cognos Impromptu, FrontPage, Adobe Acrobat, etc.
- Updating pages was a highly manual process
- Lots of “round-tripping” required to make changes
- Safety and stability of the source data wasn’t assured

NORTHWESTERN COLLEGE

Goals

- Reduce the number of tools and steps needed to build the Fact Book
- Keep the data safe!
- Lose as little functionality and richness as possible
- Automate page generation and “flipping” over to the next year, freeing up time for other tasks

NORTHWESTERN COLLEGE

My Solution: Use R to Do Everything

- "R is a free software environment for statistical computing and graphics." (http://www.r-project.org/)
- A recent eAIR Tech Tip on Using R as a Calculator
- In addition to the base software, there are over 4000 add-on packages at the Comprehensive R Archive Network (CRAN)
- With that many packages available, surely I could find the functionality I needed, right?

NORTHWESTERN COLLEGE

Step 1: Extract the data

- Determine the year, use the RODBC package to connect to the database, use SQL to retrieve data:

```
1  suppressPackageStartupMessages(library(RODBC))
2
3  curY = 1900+as.POSIXlt(Sys.Date())$year
4
5  server <- odbcDriverConnect('driver={SQL Server};
6                               server=[redacted];
7                               database=[redacted];
8                               trusted_connection=true')
9
10 day5 <- sqlQuery(server, paste("select * from DAY5.Enrollment
11                                 where year >= ", curY, "-4"))
12
```

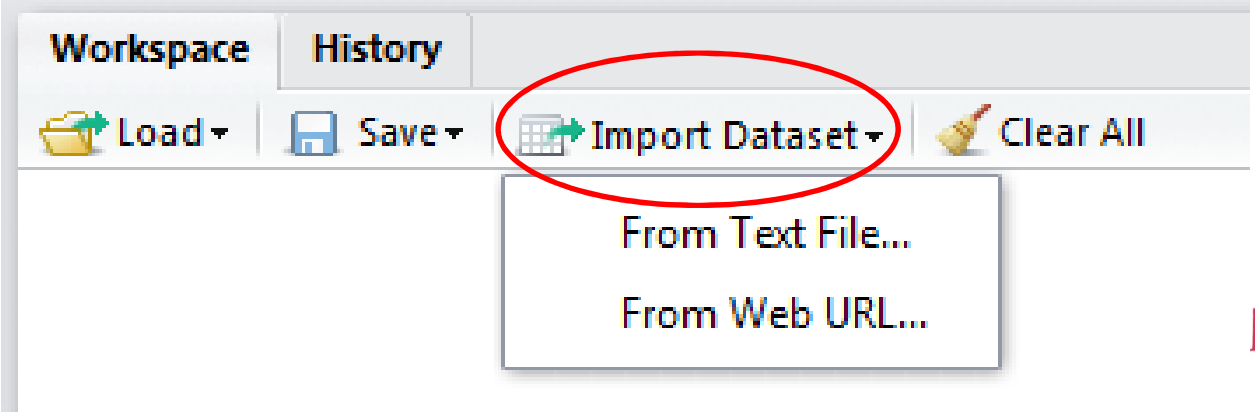
NORTHWESTERN COLLEGE

No SQL? No Problem!

- If the data is in a flat file, such as a CSV, the previous step can be accomplished in one line of code:

```
1  day5 <- read.csv("C:/Users/mwalling/Desktop/website/data/day5.csv")
```

- Can also be done using the RStudio interface



NORTHWESTERN COLLEGE

The Data Set in R

- The data from the database is now stored in R's memory, represented in a grid like so:

	SESS	YEAR	ID	Name	State	Major1	Major2
1	FA	2012	[redacted]	[redacted]	IA	ENG	NA
2	FA	2011	[redacted]	[redacted]	IA	ACC	BUS

- We can now use R to manipulate the data and calculate the necessary statistics for our report

NORTHWESTERN COLLEGE

Step 2: Analyze the Data

- This excerpt counts the total students enrolled in Fall 2012, and calculates the FTE and %FTE of Total

```
13  enrollment <- nrow(subset(day5, ID > 0 & YEAR == 2012 & SESS == 'FA'))
14
15  full_time <- nrow(subset(day5, ID > 0 & YEAR == 2012 & SESS == 'FA'
16                           & RegHrs >= 12))
17
18  part_time_equiv <- sum(subset(day5, ID > 0 & YEAR == 2012 & SESS == 'FA'
19                                & RegHrs > 0 & RegHrs < 12)
20                         $RegHrs)/12
21
22  fte = round(full_time + part_time_equiv)
23
24  percent_fte_of_total = sprintf("%.1f", fte/enrollment*100)
```

NORTHWESTERN COLLEGE

What Did We Calculate?

- We can print these variables at the R console to see their values:

```
> print(enrollment)
[1] 1241
> print(fte)
[1] 1198
> print(percent_fte_of_total)
[1] "96.5"
```

NORTHWESTERN COLLEGE

Step 3: Design Table and Page Layouts

- The xtable library allows us to print data from R in tabular form into multiple formats, such as HTML, LaTeX (for PDFs), etc.
  - But first, we need to collect our multiple variables into a single structure, called a data frame

```
26  library(xtable)
27  enrolled.df <- data.frame(enrollment, fte, percent_fte_of_total,
28                            row.names=c("FA 2012"))
29  colnames(enrolled.df) <- c("Total Enrolled", "Total FTE",
30                             "%FTE of Total")
31  print(xtable(enrolled.df), type = "html", include.rownames = T)
```

NORTHWESTERN COLLEGE

What Did We Just Print? And Why?

- The xtable function created our table in HTML format.
- Putting that command inside the print command printed the resulting HTML at the console

```
> print(xtable(enrolled.df), type = "html", include.rownames = T)
<!-- html table generated in R 2.15.1 by xtable 1.7-0 package -->
<!-- Fri Nov 02 10:08:39 2012 -->
<TABLE border=1>
<TR> <TH>  </TH> <TH> Total Enrolled </TH> <TH> Total FTE </TH> <TH> %FTE of Total </TH>  </TR>
  <TR> <TD align="right"> FA 2012 </TD> <TD align="right"> 1241 </TD> <TD align="right"> 1198.00
</TD> <TD> 96.5 </TD> </TR>
   </TABLE>
>
```

NORTHWESTERN COLLEGE

Step 4: Publish to the Web

- There are multiple packages in R that take R code/results, and convert it to a publishable format (postscript, PDF, HTML, etc.)
- One such package is Knitr. It intertwines R code/results with Markdown-formatted text and produces HTML output
- Even better, support for it is built into RStudio

NORTHWESTERN COLLEGE

What a minute… what's Markdown?

- Markdown is a plain-text formatting syntax that can be interpreted into HTML
- Goal: easier to read and write than actual HTML, but still produce validly coded web pages
- Key idea: You can turn a Markdown file into HTML, but you can also share it as-is and it will be easily readable by the average person
- More: Markdown Project Home Page

NORTHWESTERN COLLEGE

A Very Simple Markdown Example

THIS:

```
Snack Foods
-------------------
* Candy.
* Gum.
* Soda.
```

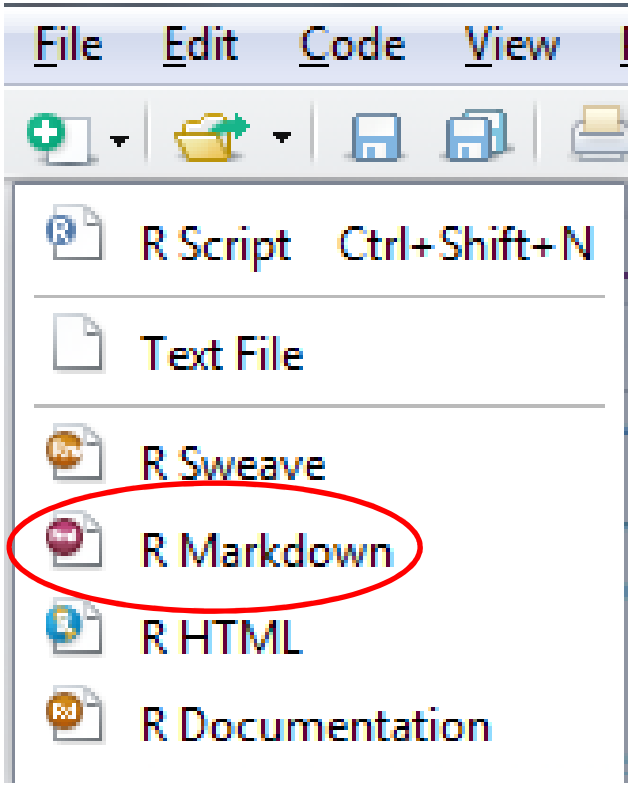
BECOMES THIS:

```
<h2>Snack Foods</h2>
<ul>
<li>Candy.</li>
<li>Gum.</li>
<li>Soda.</li>
</ul>
```

NORTHWESTERN COLLEGE

Back to Using Knitr

- Create a new R Markdown file in RStudio



NORTHWESTERN COLLEGE

Placing R Code in the R Markdown File

- Copy our code into the new file, splitting it into two sections, as described here:

````
```{r echo=FALSE, results='hide'}

# Code that you want to execute, but you do not
# want the code or its results to appear in the
# HTML output

# In our case, everything up to the xtable command

```
````

````
```{r echo=FALSE, results='asis'}

# Code that you want to execute, and have the
# code's results show up in the web page as-is

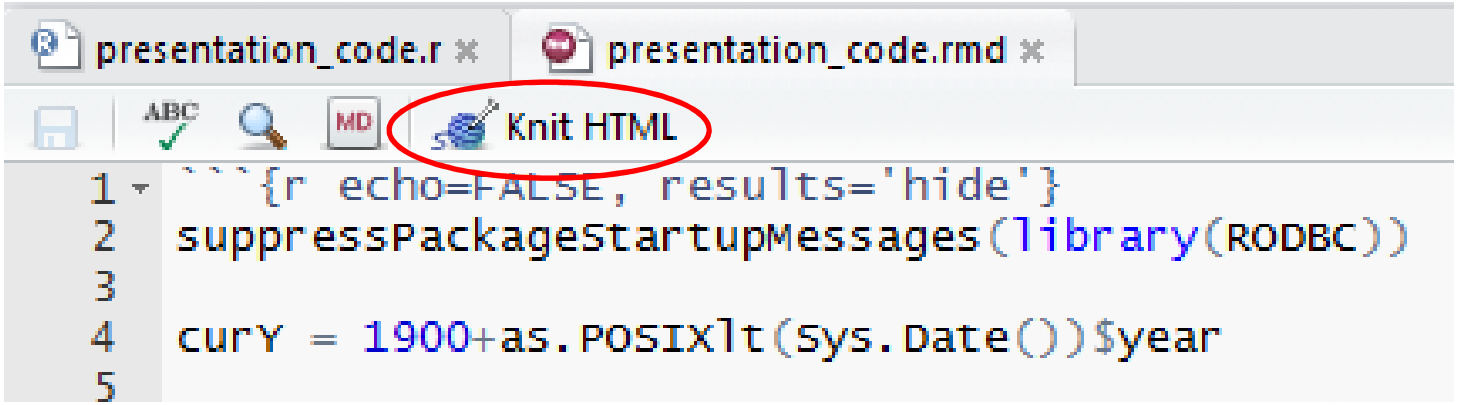
# In our case, the print(xtable()) line of code
# would go here, and the resulting table will
# get placed into the HTML file

```
````

NORTHWESTERN COLLEGE

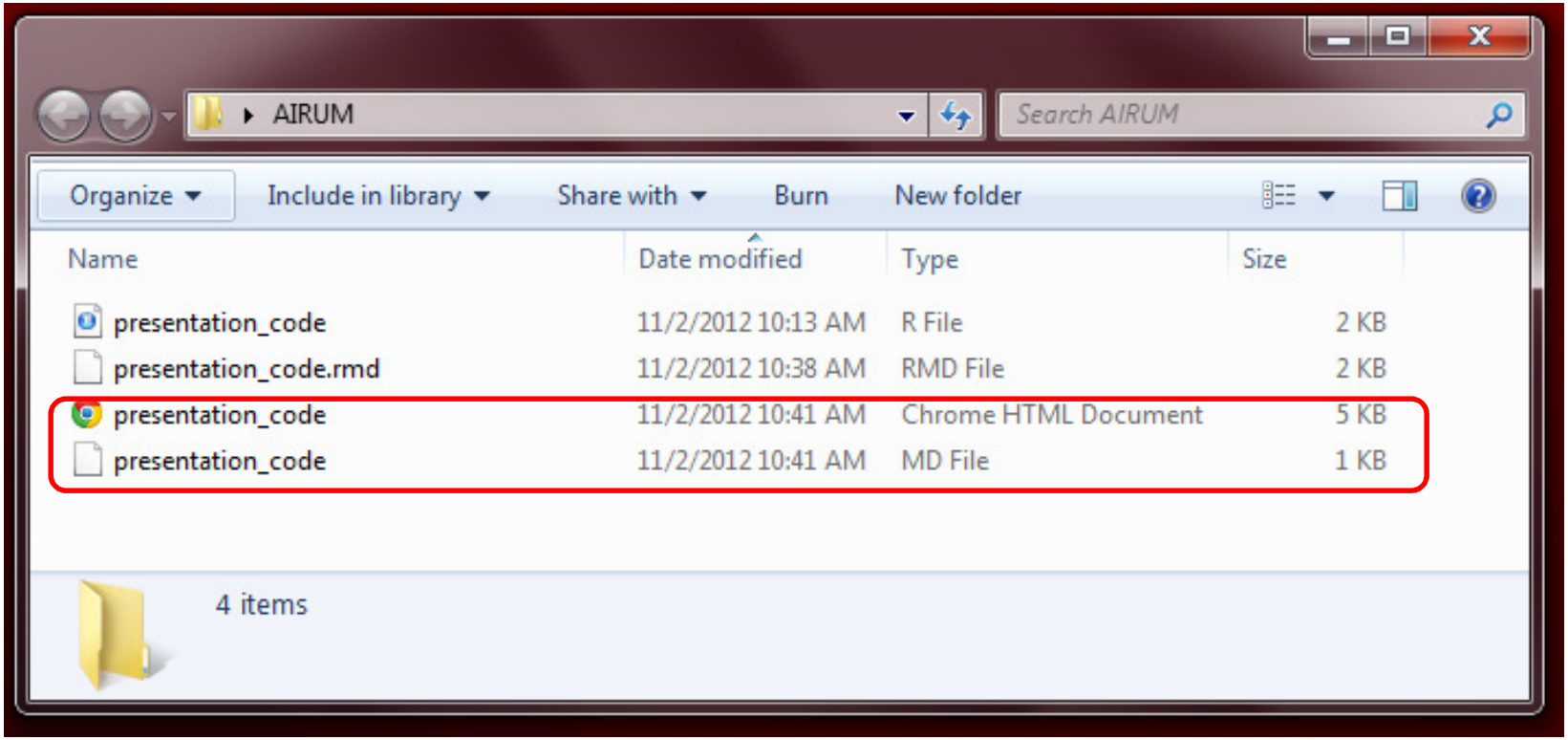
Next… Time to Knit!

- RStudio has a convenient “Knit HTML” button that automatically executes the commands to convert our RMarkdown file into a plain Markdown file, and from that, into a pure HTML file:



NORTHWESTERN COLLEGE

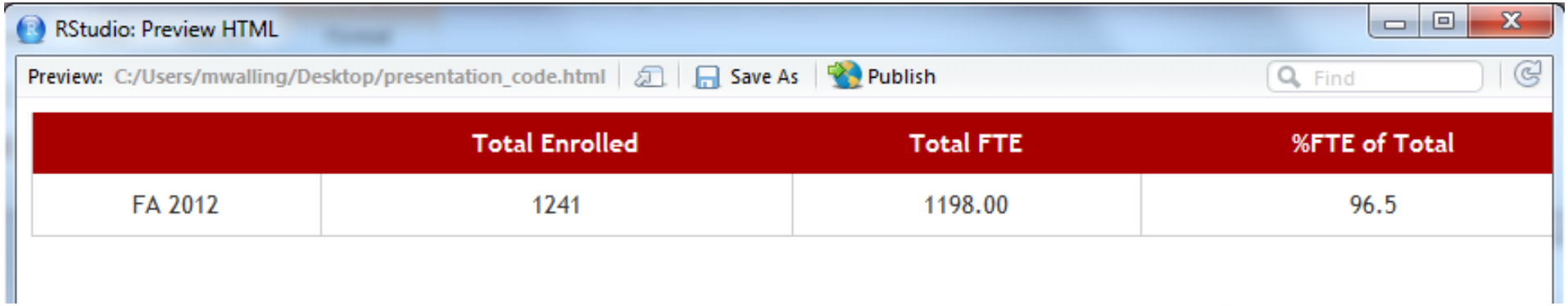
The Resulting Files



NORTHWESTERN COLLEGE

The Resulting Web Page

- These results are fairly plain, but provides an example of what can be done.
  - There is some custom CSS applied to the HTML file



	Total Enrolled	Total FTE	%FTE of Total
FA 2012	1241	1198.00	96.5

NORTHWESTERN COLLEGE

More Pages: Adding Visualizations

- One of my goals was to not lose any functionality/features of the existing web site
- That includes charts and graphs
- Luckily, R excels at producing visualizations (no pun intended!)

NORTHWESTERN COLLEGE

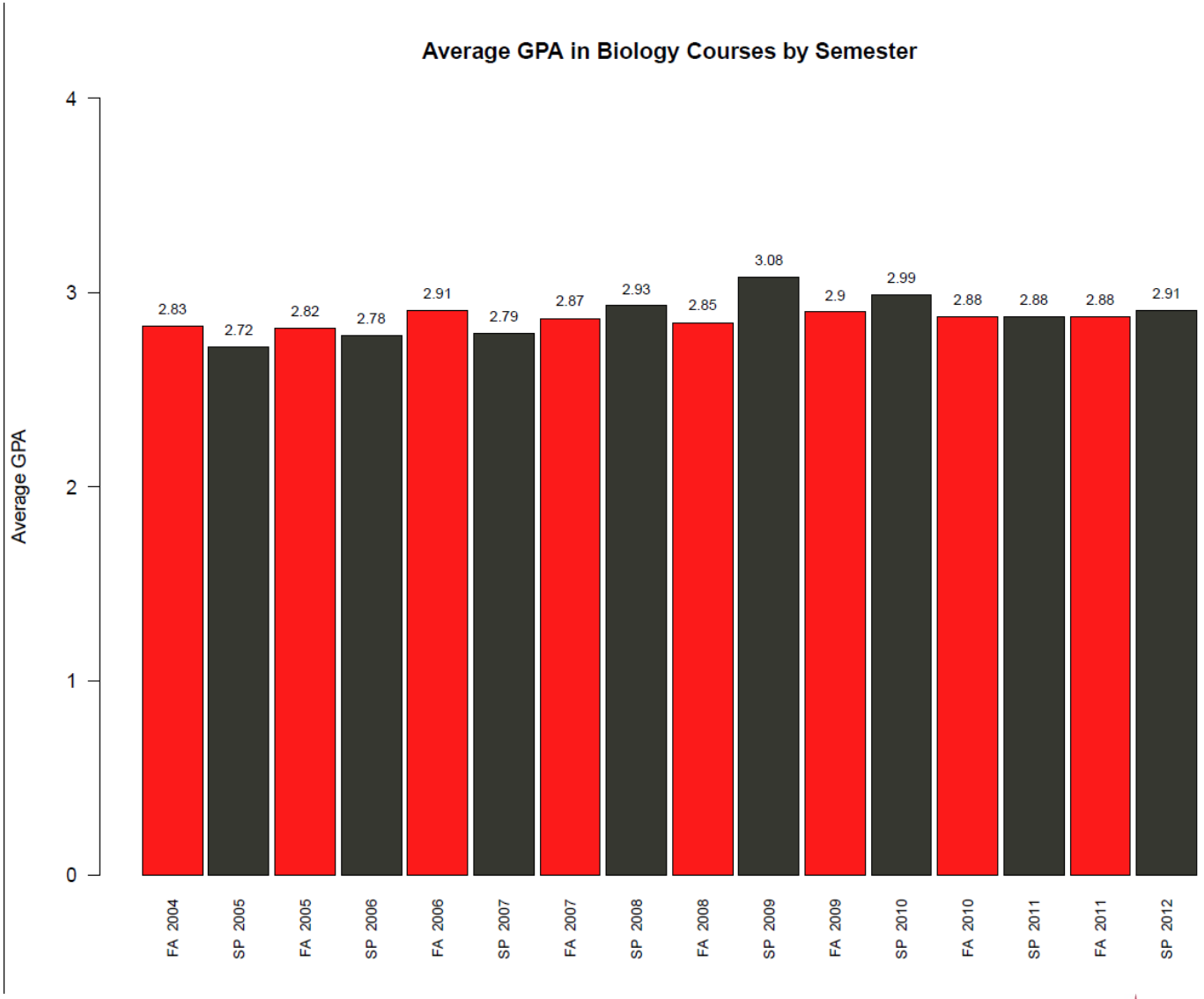
Tracking Average Student GPA in Courses within Each Department

- Read the data (either via SQL or from a CSV)
- Split the data according to departments
- For each department, graph the results and print the graph to a PDF file
- The code to accomplish this is put into an RMarkdown file that also describes the web page linking to these reports

NORTHWESTERN COLLEGE

```
dept_gpa <- read.csv("C:/Users/mwalling/Desktop/AIRUM/website/data/gpa.csv")
depts <- isplit(dept_gpa, dept_gpa$department)
foreach(dept = depts) %do% {
  pdf(paste(gsub(" ","", strsplit(dept$key[[1]],"/")[[1]][1] , fixed=TRUE),
            ".pdf",sep=""), height=8.5, width=11)
  curr.dept = dept
  plot.title = curr.dept$value$department
  bar.labels <- round(dept$value$average_gpa, 2)
  bar.plot <- barplot(height=dept$value$average_gpa,
          names.arg=dept$value$semester,
          space = 0.1,
          beside=TRUE,
          col=c("red","#33342C"),
          border="black",
          main= paste("Average GPA in", plot.title[1], "Courses by Semester"),
          ylab="Average GPA",
          ylim=c(0, 4.0),
          cex.axis=par("cex.axis"),
          cex.names=0.75,
          las=2
  )
  text(bar.plot, bar.labels, labels=bar.labels, cex=0.75, pos=3)
  dev.off()
}
```

NORTHWESTERN COLLEGE



NORTHWESTERN COLLEGE

Many Options for Graphics

- The previous bar graph was created using the built-in “base” graphics features
- There are other popular, powerful graphics packages you could use instead:
  - Lattice
  - ggplot2
- Google also provides a version of its googleVis JavaScript library

NORTHWESTERN COLLEGE

Student Enrollment by State

- One of the Fact Book reports is a listing of how many students in the student body come from each state
- This can be easily done by creating a data frame and publishing it using xtable, as before
- But, this also begs for a neat, interactive map using googleVis!

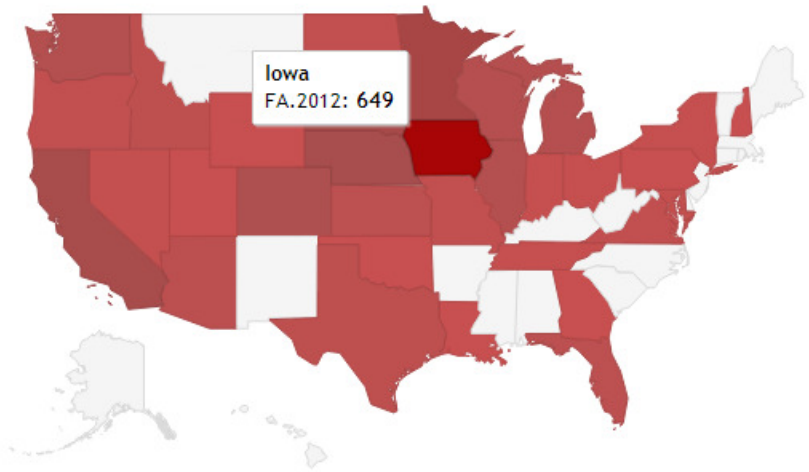
NORTHWESTERN COLLEGE

The Code

```
```{r echo=FALSE, results='asis'}
suppressPackageStartupMessages(library(googleVis))
state.chart <- gvisGeoChart(by.state, colnames(by.state)[1], colnames(by.state)[6],
                            options=list(region="US", resolution="provinces",
                                         width="900", keepAspectRation="true",
                                         legend="none",
                                         colorAxis="{colors:[\'#F5F5F5\',
                                         \'#C85050\', '#B85050\', '#A85050\',
                                         \'#A80000\'], values:[0,1,10,50,700]}"))
print(state.chart, "chart")
library(xtable)
print(xtable(by.state), type = "html", include.rownames = F)
```
```

NORTHWESTERN COLLEGE

The Results



State	FA 2008	FA 2009	FA 2010	FA 2011	FA 2012
Alaska	1	1	1	1	0
Arizona	11	8	7	6	8
Arkansas	1	2	3	0	0
California	39	42	59	66	85
Colorado	30	25	29	24	32

NORTHWESTERN COLLEGE

Automating the Process

- OK, we've created a few web pages… but can we automate the creation of the entire web site?
- Sure we can!
- To do so, we need to write some code to mimic what the "Knit HTML" button does

NORTHWESTERN COLLEGE

Scripting a Single Web Page

- This is more or less what the “Knit HTML” button is doing:

```
require('knitr')
require('markdown')
setwd('C:/Users/mwalling/Desktop/AIRUM/website')

knit(input='./rmd/enrollment_departments.rmd',
     output='./md/enrollment_departments.md',
     envir=new.env())

markdownToHTML(file='./md/enrollment_departments.md',
               output='../html/enrollment_departments.html',
               stylesheet=getOption('bad_file_name'),
               title='Student Enrollment by Academic Departments')

```

NORTHWESTERN COLLEGE

Create a Function to Handle an Arbitrary Web Page

```
generate_webpage <- function(page_name) {

  require('knitr')
  require('markdown')
  setwd('C:/Users/mwalling/Desktop/AIRUM/website')

  knit(input=paste('./rmd/', page_name, '.rmd', sep=''),
       output=paste('./md/', page_name, '.md', sep=''),
       envir=new.env())

   markdownToHTML(file=paste('./md/', page_name, '.md', sep=''),
                  output=paste('../html/', page_name, '.html', sep=''),
                  stylesheet=getOption('bad_file_name')
                  title=pagename)
```

NORTHWESTERN COLLEGE

Finally, Automate the Process

- Write a script to call that function repeatedly…

```
generate_website <- function() {
  source('C:/Users/mwalling/Desktop/AIRUM/website/generate_webpage.r')
  generate_webpage('index')
  generate_webpage('enrollment_summary')
  generate_webpage('enrollment_departments')
  generate_webpage('enrollment_states')
  generate_webpage('dept_gpa')
}
```

- … and run the script at the R console

```
> source('C:/Users/mwalling/Desktop/AIRUM/website/generate_website.r')
> generate_website()
Loading required package: knitr
Loading required package: markdown


processing file: ./rmd/index.rmd
```

Conclusions

- Did I accomplish my goals? Yes!
  - The data is safe in a database
  - The reports are written once, and will automatically update for the next year (or semester)
  - The same data is being presented, and in same cases extra functionality was added (interactive maps)
  - Steps involved: load new data into database, start the R script, and let it go to work!
- Can this be done in other tools? Probably!
  - Avoid the “all you have is a hammer” syndrome

NORTHWESTERN COLLEGE

A Caveat

- Resulting web pages are NOT active content
  - They are static .html files with snapshot data
  - This is fine for once-a-semester updates
  - But not appropriate for “live” dashboards

NORTHWESTERN COLLEGE

Obstacles

- Fairly difficult learning curve
  - Even if you're a programmer, R can be a little… unique
- All of the functionality isn't always apparent
  - So many features are "hidden" in add-on packages that you don't know what's possible!
- Multiple packages to accomplish the same thing
  - Base graphics? Lattice? ggplot2? GoogleVis?
  - Knitr? Sweave?

NORTHWESTERN COLLEGE

Future Projects (for version 2.0!)

- Fact Book conversion to PDF is still a work in progress
  - Convert the HTML pages to *styled* PDF pages
  - Combine multiple PDF pages into a single file
- Add error handling to the scripts
  - What to do if a page causes an error message?
  - R supports try/catch blocks like many other programming languages, so I'm hopeful

NORTHWESTERN COLLEGE

Inspiration and Further Reading

- R Studio documentation on using Markdown
- Reproducible Research with Markdown, Knitr, and Pandoc
- Getting Started with R Markdown and Knitr

NORTHWESTERN COLLEGE