Running R from SAS

Dr. David Zeitler

Grand Valley State University

Code and slides available on our github site

http://github.com/WestMichiganRUserGroup/R-from-SAS

Setting up software

- I use a Mac with LaTeX and Office, and switch to Windows 8 for SAS 9.4 work.
    - You should have Windows with office installed.
- SAS (9.4 with SAS/IML preferred).
- R, either CRAN or MRAN (I have MRAN)
- RStudio
- Installed R packages: source('InstallPackages.R')

dplyr, ggplot2, car, mosaicData, lazyeval, MASS,

reshape2, readr, latticeExtra,

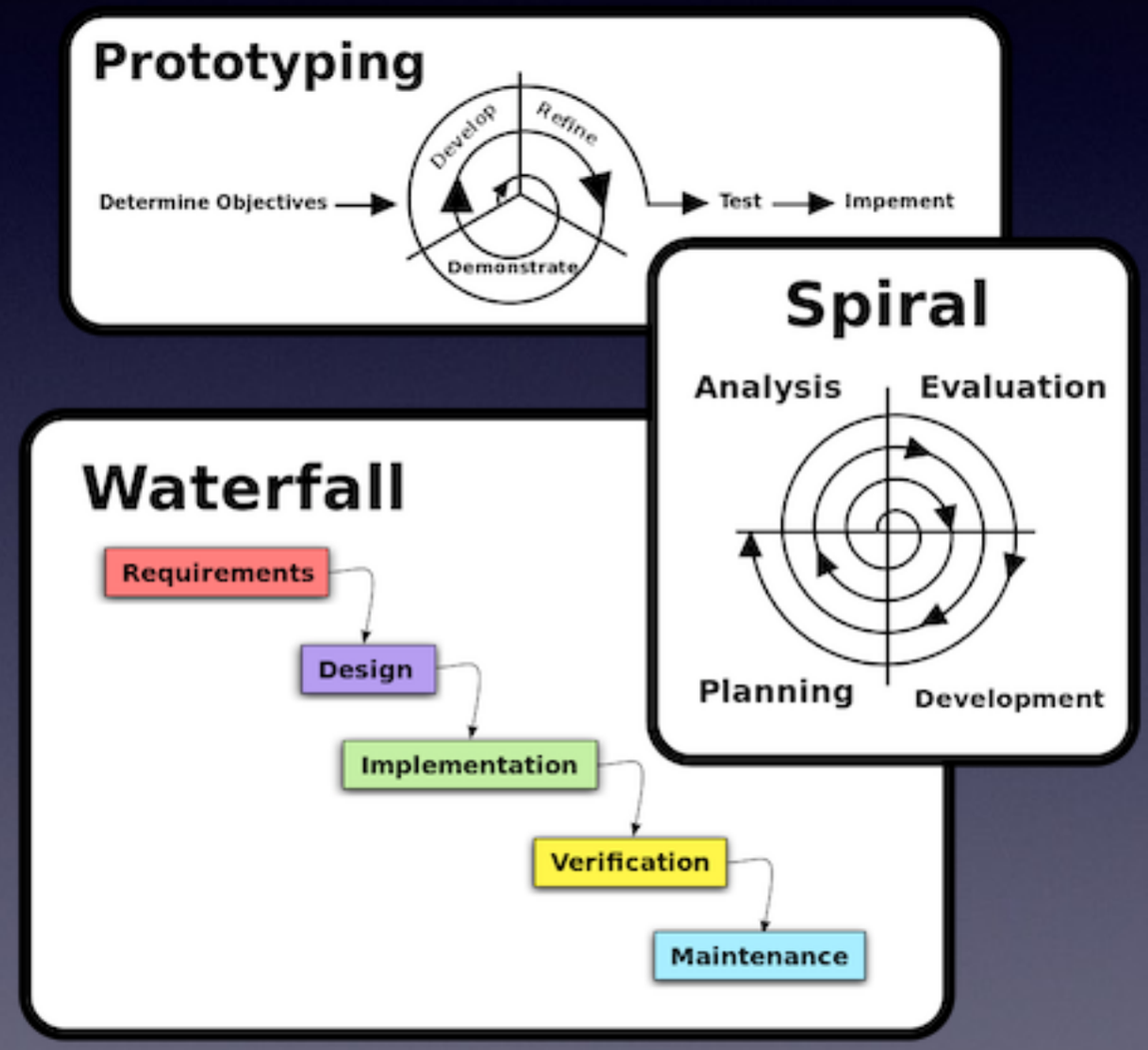
ggdendro, gridExtra, lubridate, fastR, magrittr, NHANES,

RCurl, sp, maptools, vcd, testthat, tidyr, knitr,

mapproj, rgl, manipulate, Rcmdr, RcmdrPlugin.EZR

Software Development

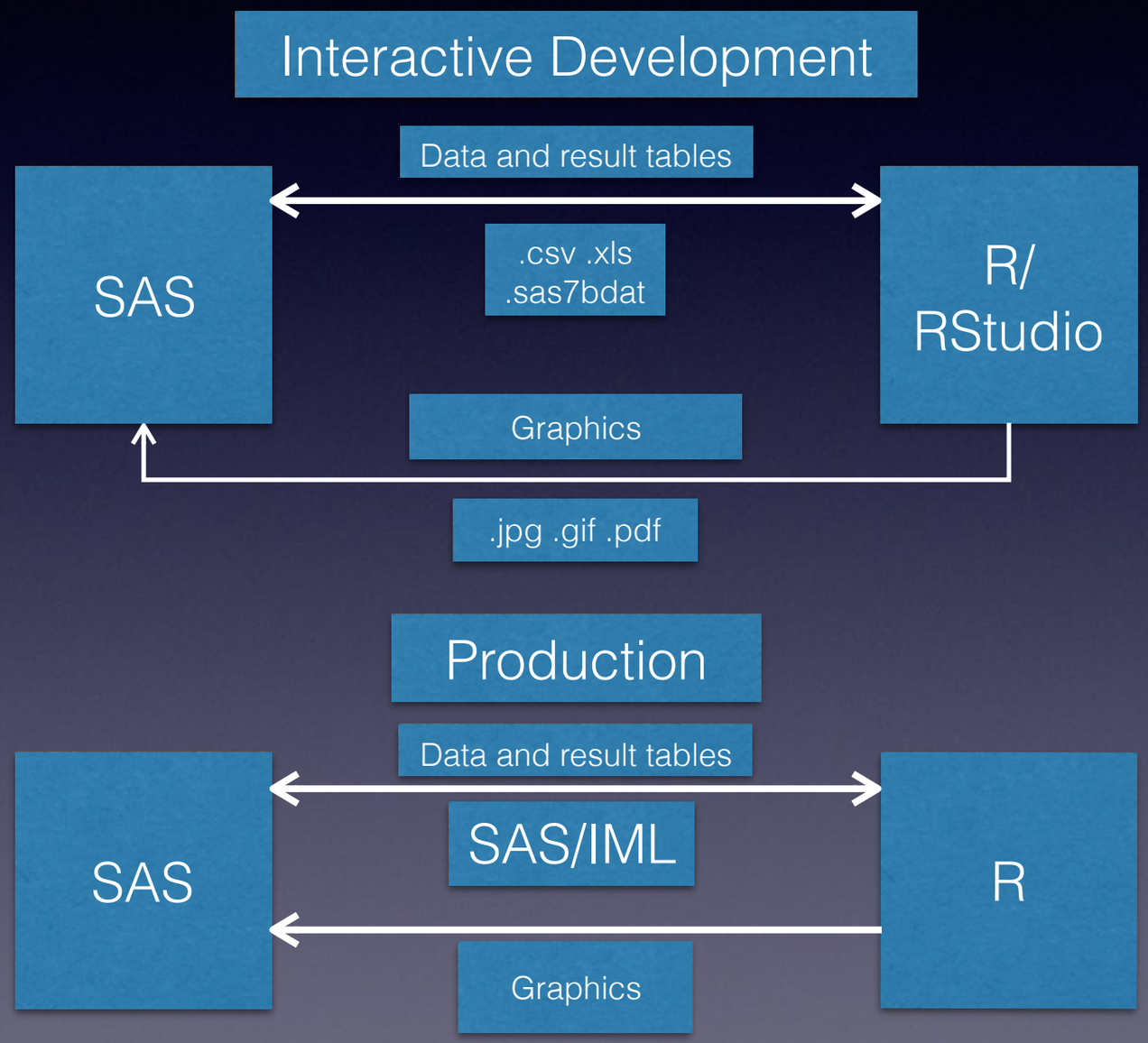
Developing code, whether SAS, R, Java, or any other is software development. The end result is code that produces something. For analysts, this is often a report. Regardless, software development is a process and can be viewed as sequential (waterfall) or iterative (spiral or prototyping).



Development vs Production

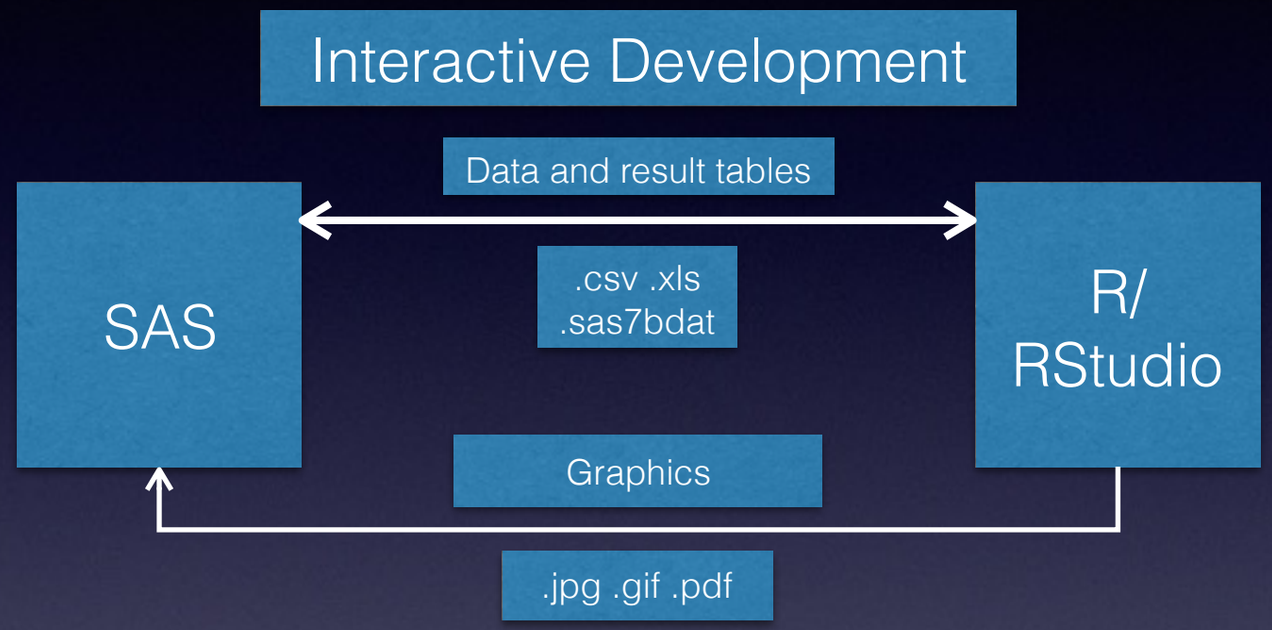
<u>Development</u> is done interactively, ending in a program for production.

<u>Production</u> runs without intervention.



Interactive development

- R work is done interactively
- Save SAS data to file and read into R
- Perform analysis in R writing result files for data tables and graphics images
- Read results back into SAS.



- Works with SAS 9.3
- Somewhat difficult to automate

Data import/export

- Write .csv file from SAS

```
ods csv body="c:\test.csv";
proc print data=sashelp.class;
run;
ods csv close;
```

- Read .csv file in SAS

```
proc import datafile="C:\temp\test.csv"
     out=shoes
     dbms=csv
     replace;
     getnames=no;
run;
```

- Read .sas7bdat into data frame in R

```
library(haven)
read_csv('c:\test.sas7bdat')
```

- Write dataframe to .csv in R

```
write.csv(test,file='c:\test.sas7bdat')
```

Using excel .xlsx files

- SAS read/write .xlsx files (http://www.ats.ucla.edu/stat/sas/faq/rwxls8.htm)

```
PROC IMPORT OUT= WORK.auto1 DATAFILE= "C:\auto.xlsx"
            DBMS=xlsx REPLACE;
     SHEET="auto";
     GETNAMES=YES;
RUN;
proc export data=mydata outfile='c:\dissertation\mydata.xlsx' dbms = xlsx
replace;
run;
```

- R read/write .xlsx files

```
library(xlsx)
# example of reading xlsx sheets
file <- system.file("tests", "test_import.xlsx", package = "xlsx")
res <- read.xlsx(file, 2) # read the second sheet
# example of writing xlsx sheets
file <- paste(tempfile(), "xlsx", sep=".")
write.xlsx(USArrests, file=file)
```

Graphics files

- Writing graphics from R.

```
# Using mosaic
png(‘mygraphic.png’)
boxplot(y~x,data=mydataframe)
device.off()
# With ggplot2
ggsave(‘mygraphic.png’)
```

- Including imported graphics in SAS ODS

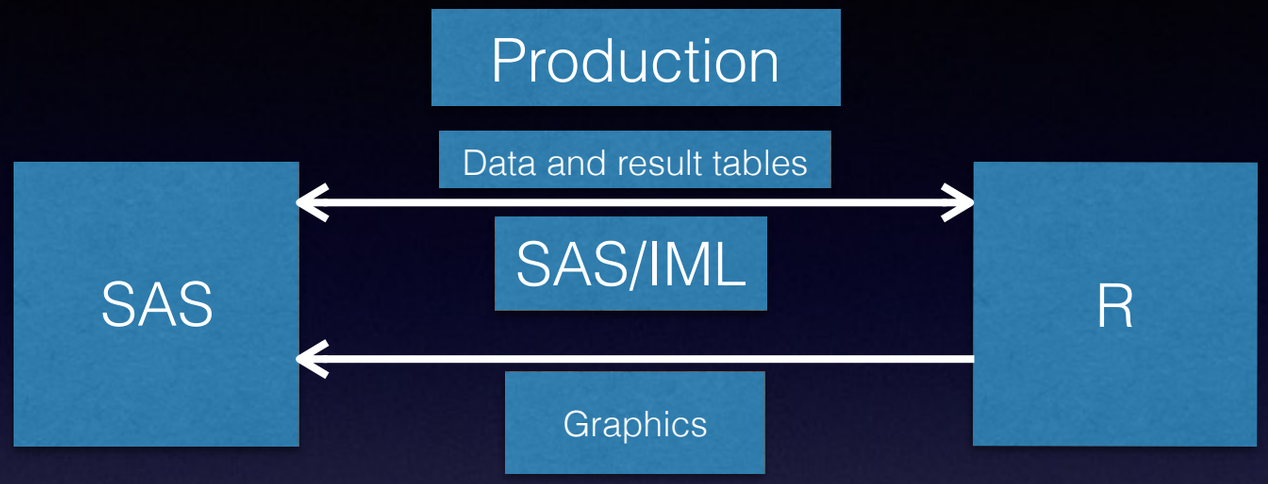
```
ods pdf file = "test.pdf" nogtitle nogfoot
 title = 'R graphic image';
ods escapechar='~';
ods text='~S={width=100% preimage="myplot.png"}';
ods pdf close;
```

- Note: SAS ODS output doesn’t appear to adhere well to standards.

Production - SAS 9.4 IML

Production runs without intervention, allowing automated report generation.

Make sure to add RLANG to your SASV9.CFG or -RLANG to your start icon properties. You'll need system privileges.



SASR.sas

```
proc iml; /* Comparison of matrix operations in IML and R */
print "----------  SAS/IML Results  ----------";
x = 1:3;                              /* vector of sequence 1,2,3 */
m = {1 2 3, 4 5 6, 7 8 9};            /* 3 x 3 matrix */
q = m * t(x);                         /* matrix multiplication */
print q;

print "-------------  R Results  ---------------------";
submit / R;
  rx <- matrix( 1:3, nrow=1)            # vector of sequence 1,2,3
  rm <- matrix( 1:9, nrow=3, byrow=TRUE) # 3 x 3 matrix
  rq <- rm %*% t(rx)                    # matrix multiplication
  print(rq)
endsubmit;
```

```
--------------------------------------------------------------------------------
The SAS System

---------- SAS/IML Results ----------


          q
         14
         32
         50


------------ R Results ----------


     [,1]
[1,]   14
[2,]   32
[3,]   50
```

SAS/IML R communication

Transferring from a SAS Source to an R Destination

Method or Module	SAS Source	R Destination
ExportDataSetToR	SAS data set	R data frame
ExportMatrixToR	SAS/IML matrix	R matrix
DataObject.ExportToR	DataObject	R data frame

Transferring from an R Source to a SAS Destination

Method or Module	R Source	SAS Destination
DataObject.AddVarFromR	R expression	DataObject variable
DataObject.CreateFromR	R expression	DataObject
ImportDataSetFromR	R expression	SAS data set
ImportMatrixFromR	R expression	SAS/IML matrix

- Export data from SAS to R (ExportDataSetToR)
- Submit R code to create data objects
- Export any desired R graphics to image files. Use full file paths or they may get buried in temp folders.
- Import data from R to SAS (ImportDataSetFromR)
- Complete any desired analysis in SAS
- Use ODS to format output for reporting in to incorporate R graphics image files.
- Examples in R_IML.sas

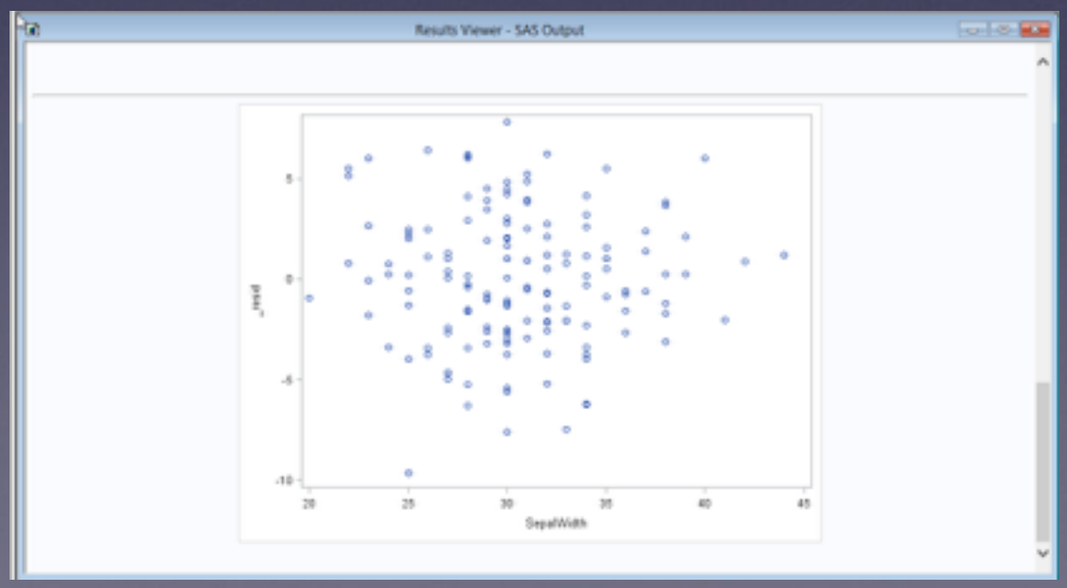
Put it all together

R_IML.sas

```
/* Full IML use of R from SAS */
proc iml;

/* Send the SAS help data set iris to R as SASIris */
run ExportDataSetToR("Sashelp.iris", "SASIris" );
submit / R;
        library(mosaic)
        str(SASIris);
        model <- lm(SepalLength ~ SepalWidth + PetalLength, data=SASIris)
        summary(model)
        anova(model)
        SASIris.Diag <- fortify(model)
        SASIris.Diag %>%
                ggplot(aes(x=SepalWidth,y=.resid)) +
                geom_point() +
                ggtitle("Residuals by Sepal Width")
        ggsave("w:\\IrisResid.png")
endsubmit;

/* Bring R results back to SAS */
run ImportDataSetFromR("Work.SASIrisDiag","SASIris.Diag");
use Work.SASIrisDiag;
show contents;
close Work.SASIrisDiag;
proc sgplot
   data=Work.SASIrisDiag
   description="Iris model from R - diagnostics";
   scatter
      x = SepalWidth
      y = _resid;
run;
```

```
'data.frame':   150 obs. of  5 variables:
 $ Species    : Factor w/ 3 levels 'Setosa','Versicolor',..: 1 1 1 1 1 1 1 1 1 1 ...
 $ SepalLength: num  50 46 46 51 55 48 52 49 44 50 ...
 $ SepalWidth : num  33 34 36 33 35 31 34 36 32 35 ...
 $ PetalLength: num  14 14 10 17 13 16 14 14 13 16 ...
 $ PetalWidth : num  2 3 2 5 2 2 2 1 2 6 ...

Call:
lm(formula = SepalLength ~ SepalWidth + PetalLength, data = SASIris)

Residuals:
    Min      1Q  Median      3Q     Max
-9.6159 -2.3489  0.0077  2.1453  7.8557

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 22.49140    2.47970    9.07 7.04e-16 ***
SepalWidth   0.59552    0.06933    8.59 1.16e-14 ***
PetalLength  0.47192    0.01712   27.57  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.333 on 147 degrees of freedom
Multiple R-squared:  0.8402,    Adjusted R-squared:  0.838
F-statistic: 386.4 on 2 and 147 DF,  p-value: < 2.2e-16

Analysis of Variance Table

Response: SepalLength
             Df Sum Sq Mean Sq F value    Pr(>F)
SepalWidth    1  141.2   141.2  12.714 0.0004902 ***
PetalLength   1 8442.7  8442.7 760.059 < 2.2e-16 ***
Residuals   147 1632.9    11.1
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
DATASET : WORK.SASIRISDIAG.DATA

 VARIABLE                        TYPE  SIZE
 ------------------------------- ----  ----
 SepalLength                     num      8
 SepalWidth                      num      8
 PetalLength                     num      8
 _hat                            num      8
 _sigma                          num      8
 _cooksd                         num      8
 _fitted                         num      8
 _resid                          num      8
 _stdresid                       num      8

Number of Variables   : 9
Number of Observations: 150
```



What if you don't have IML?

- Save SAS datasets to disk (either csv or sas7bdat)
- Put R code into a separate file, call it <myuRcode.R>.
- Use Powershell Integrated Scripting Environment (ISE).
- Create a batch R file <myRcode.R>.
    - Develop the script interactively until you can source it in a clean session.
    - Use sink('Routput.txt') to save R output to a text file.
    - In <myRcode.R> write data sets to csv file to send to SAS
    - Save graphics to image format files
- Run the batch file from SAS code using the x command.
    - http://www2.sas.com/proceedings/sugi31/036-31.pdf
    - x "c:\'Program Files'\Microsoft\MRO\R-3.2.3\bin\R.exe CMD BATCH myRcode.R";
- Incorporate R output and graphics into SAS ODS.

R batch example

odsimage.sas

```
/* Run the batch file to do work in R first */
options noxwait;
x "myR";

/* Set options for RTF output */
ods rtf file = "test.rtf" nogtitle nogfoot
        /* Titles and footnotes */
        title = 'R graphic image';
ods escapechar='~';
/* Import the image and output into the RTF */
ods text='~S={width=100% preimage="w:\\Iris.png"}';
ods rtf close;

/* ---------------------- */
/* Set options for PDF output */
ods pdf file = "test.pdf" nogtitle nogfoot
        /* Titles and footnotes */
        title = 'R graphic image';
ods escapechar='~';
/* Import the image and output into the RTF */
ods text='~S={width=100% preimage="w:\\iris.png"}';
ods pdf close;";
```

myR.bat

```
"C:\Program Files\R\R-3.2.5\bin\R.exe" CMD BATCH myRcode.R
```

myRcode.R

```
# Simple R script
library(mosaic)
library(ggplot2)
library(dplyr)
# sink not needed since we're just getting the
# graphic output this time
#   sink('myRoutput.txt')
iris %>%
  ggplot(aes(x=Sepal.Length,y=Sepal.Width)) +
  geom_point(aes(col=Petal.Length,
                 size=Petal.Width,
                 shape=Species),
             alpha=.5) +
  theme_light()
# Note the file path contains full path
# otherwise the file gets burried
ggsave('w:\\iris.png')
```