Zelig: Everyone’s Statistical Software
Toward A Common Framework for Statistical Analysis & Development

Kosuke Imai¹  Gary King²  Olivia Lau³

¹Department of Politics
Princeton University

²Department of Government
Harvard University

³Center for Drug Evaluation and Research
Food and Drug Administration

February 26, 2010
Motivation for the Zelig Project

The Problem:
- Quantitative methodology is thriving like never before
- It is wonderful, but results in different jargon, notation, syntaxes, etc.

The Consequence:
- Hard to learn useful methods developed in various disciplines
- Despite their common underlying statistical foundation

Possible Solutions:
- Top-down approaches: possible (though inefficient) in commercial packages but contradict with the nature of scientific inquiry
- Open source approaches such as R (and Zelig)
What’s R? Why Should I Use R and Zelig?

What’s R?
- Canned statistical packages
- An open-source project (free and reliable)
- An object-oriented programming language

Why should I use R?
- Most methodologists and statisticians use R
- More statistical procedures than other software

Why should I use Zelig?
- With Zelig, R is easy to learn and use
- No need to wait until a commercial statistical package decides to include a procedure
- R and Zelig create a community of users and developers of statistics with a common language
What Does Zelig Do?: A Unified User Interface

- Interpreting and presenting statistical results:
  1. Focus on the scientific quantities of interest
  2. Point and uncertainty estimates

- Providing additional infrastructure:
  1. multiple imputation
  2. matching methods
  3. counterfactual evaluations
  4. replication, etc.

- Encompassing a large fraction of statistical models:
  1. Bayesian and frequentist models
  2. single and multiple equations models
  3. cross-section and time-series models
  4. time-series-cross-section models
  5. single and multi-level models, etc.
What Does Zelig Do?: A Developer’s Interface

1. **Tools for writing new models** so that developers can easily transform user inputs into mathematically convenient forms.

2. **Methods to wrap existing packages** so that developers do not have to modify their packages in order to include them into Zelig.

3. **A dynamically-generated GUI** so that those who do not know R can easily use developer’s packages.
Primary Zelig Commands: An Example

```r
z.out <- zelig(vote ~ race + educate,
               data = turnout,
               model = "probit")

x.out <- setx(z.out, educate = 12)

s.out <- sim(z.out, x = x.out)
```

Select vars
Select data set
Select model
Select QIs
Calculate QIs

Imai, King, & Lau (Princeton, Harvard, FDA)  Zelig: Everyone’s Statistical Software  February 26, 2010
Statistical Ontology: R Formula and Its Extension

- **R formula is simple yet comprehensive:**

  \[ f \leftarrow y \sim x_1 + x_2:x_3 + I\left(x_4^2/\sqrt{x_3}\right) + \log(x_5) \]

- **Zelig’s extension of R formula to multiple equations:**

  \[ f \leftarrow \text{list(} \mu_1 = y_1 \sim x_1 + x_2 + x_3, \mu_2 = y_2 \sim x_1 + x_4 + x_5 \) \]

  \[ f \leftarrow \text{list(} \mu_1 = y_1 \sim x_1 + \text{tag}(x_2, \beta_2), \mu_2 = y_2 \sim x_3 + \text{tag}(x_4, \beta_2), \rho = \sim z_1 - 1 \) \]

  \[ f \leftarrow \text{list(} \text{cbind}(y_1, y_2) \sim x_1 + x_2 \) \]
Built-in Zelig functionality

- Handle multiply-imputed data frames for missing data problems
- Stratifies data and fits a statistical model within each strata
- Works with MatchIt which implements a variety of matching methods to reduce model dependence for causal inference
- Works with WhatIf which evaluates the validity of counterfactual questions
- Computes various quantities of interest and uncertainties via simulation (bootstrap or Bayesian posterior simulation)
- Numerically and graphically summarizes the results
A Big Picture

Preprocessing (Matching, Imputation, Outlier Removal, etc.)

Estimate Statistical Method: \texttt{zelig()}

Choose quantity of interest: \texttt{setx()}

Simulate quantity of interest: \texttt{sim()}

\texttt{summary()}

\texttt{whatif()}

\texttt{plot()}
Getting Started

- **Working directory**: `setwd("/Users/kimai/research")`
- **Workspace (or global environment)**
- **Store objects in the workspace**: `a <- 5`
- **Choose intuitive names for your objects**
- **R is case sensitive!** `(Hello ≠ hello ≠ HELLO)`
- **Get help using** `help.zelig()`
Different Types of R Objects

- **Scalar:** numbers, character strings, logical values
  
a <- 5; b <- "hi"; c <- TRUE;

- **Vector:** sets of one type of scalar value
  
a <- c(1,2,3); b <- rep(1, 5);

- **Matrix:** 2-D sets of one type of scalar value
  
a <- matrix(c(1,2,3,4), ncol = 2, nrow = 2)

- **Array:** K-D sets of one type of scalar value
  
a <- array(1:30, dim=c(2,3,5))

- **List:** Any combination of the above!
  
obj <- list("first" = a, "second" = b)

- **Data frame:** A special list containing variables of different types
Helpful Functions

- Display objects in the workspace:
  ```r
  > ls()
  [1] "z.out"  "turnout"
  ```

- Display elements in an list:
  ```r
  > names(turnout)
  [1] "race"  "age"  "educate"  "income"  "vote"
  ```

- Display the dimensions of a data structure:
  ```r
  > dim(turnout)
  ```
Helpful Operators

- Extract one element of a vector, array, or matrix
  
  ```r
  > turnout[25,]
  race age educate income vote
  25 white 47 16 5.233 1
  ```

- Extract an element from a list
  
  ```r
  > turnout[[4]]
  ```

- Extract a named element from a list
  
  ```r
  > turnout$race <- as.integer(turnout$race)
  ```
Loading Data

- **Tab- or space- delimited .txt file:**
  ```r
  white 60 14 3.346 1
  ...
  > mydata <- read.table("data.txt")
  ```

- **Comma-separated value .csv file**
  ```r
  white, 60, 14, 3.346, 1
  ...
  > mydata <- read.csv("data.csv")
  ```

- **Stata .dta file**
  ```r
  > library(foreign)
  > mydata <- read.dta("data.dta")
  ```

- **SPSS .sav files**
  ```r
  > library(foreign)
  > mydata <- read.spss("data.sav",
                        to.data.frame = TRUE)
  ```
Data Verification

*Check* to see if the data loaded correctly

- **Basic commands:**
  ```
  dim(mydata)
  summary(mydata)
  ```

- **Check variable names:**
  ```
  names(data)
  names(data) <- c("income", "educate", "year")
  ```

- **Display specified observations:**
  ```
  mydata[2:8, ]
  ```
Creating New Variables

1. Insert a new variable
   mydata$new <- new.var

2. Merge two data frames
   new <- merge(x, y)
   new <- merge(x, y, by.x = "x1", by.y = "y2")
   new <- merge(x, y, all = TRUE)

3. Edit your data frame like a spreadsheet
   turnout <- edit(turnout)

   (Not recommended, but may be useful for some)
Recoding Variables

1. Extract the variable you would like to recode
   \[\text{var} \leftarrow \text{mydata}\$\text{var1}\]

2. Recode the variable
   \[\text{var}[\text{var} < 0] \leftarrow 0\]

3. Return the variable to your data frame
   \[\text{mydata}\$\text{var1} \leftarrow \text{var}\]

Keep the rows in the same order!
Saving R Objects to Disk

- After cleaning your data, you should save it:
  - As an R data file:
    ```
    save(mydata, file = "mydata.RData")
    ```
  - As a tab-delimited file:
    ```
    write.table(mydata, file = "mydata.tab")
    ```
  - As a stata file:
    ```
    library(foreign)
    write.dta(mydata, file = "mydata.dta", version = 10)
    ```
- Alternatively, save your entire R workspace:
  ```
  save.image(file = "Sept1.RData")
  save(mydata, my.function, file = "mydata.RData")
  ```
- To load your `.RData` files back into R:
  ```
  load("mydata.RData")
  ```
Example 1: Logistic Regression

- **Question:** *Ceteris paribus*, how does age affect voting behavior among
  - High school graduates (12 years of education)?
  - College graduates (16 years of education)?

- **The model:**

  \[
  Y_i \sim \text{Bernoulli}(y_i \mid \pi_i), \\
  \pi_i \equiv \Pr(y_i = 1 \mid x_i) = \frac{1}{1 + \exp(-x_i\beta)}
  \]

- **Estimate the model via** `zelig()`:

  \[
  x_i\beta = \beta_0 + \beta_1\text{Race} + \beta_2\text{Educate} + \beta_3\text{Age} + \beta_4\text{Age}^2 + \beta_5\text{Income}
  \]

  \[
  z\text{.out} \leftarrow \text{zelig(vote } \sim \text{ race + educate + age + } \\
  \text{I(age}^\text{2}) + \text{income, model = "logit", data = turnout)}
  \]}
Set explanatory variables via `setx()`:

<table>
<thead>
<tr>
<th></th>
<th>Intercept</th>
<th>Race</th>
<th>Educate</th>
<th>Age</th>
<th>Age$^2$</th>
<th>Income</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{12}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>12</td>
<td>18</td>
<td>324</td>
<td>3.9</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>12</td>
<td></td>
<td></td>
<td>3.9</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>12</td>
<td>95</td>
<td>9,025</td>
<td>3.9</td>
</tr>
<tr>
<td>$x_{16}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>16</td>
<td>18</td>
<td>324</td>
<td>3.9</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>16</td>
<td></td>
<td></td>
<td>3.9</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>16</td>
<td>95</td>
<td>9,025</td>
<td>3.9</td>
</tr>
</tbody>
</table>

`x.lo <- setx(z.out, educate = 12, age = 18:95)`

`x.hi <- setx(z.out, educate = 16, age = 18:95)`
Simulate quantities of interest:

- Simulate $\hat{\beta}$ from
  1. asymptotic distribution
  2. Bayesian posterior distribution
  3. sampling distribution using bootstrap

- Calculate quantities of interest
  1. predicted probabilities: $\hat{\pi}_i = \frac{1}{1 + \exp(-x_i\hat{\beta})}$ for $i = (12, 16)$
  2. first differences: $\hat{\pi}_{16} - \hat{\pi}_{12}$
  3. predictive draws: $Y_i \sim \text{Binomial}(\hat{\pi}_i)$

```
s.out <- sim(z.out, x = x.lo, x1 = x.hi)
```

Summarize the results:

```
summary(s.out)
plot(s.out)
plot.ci(s.out, xlab = "Age in Years",
        ylab = "Predicted Probability of Voting",
        main = "Effect of Education and Age")
```
Effect of Education and Age on Voting Behavior

Age in Years

Predicted Probability of Voting

- College Education (16 years)
- High School Education (12 years)
Example 2: Fixed Effects Models

- **Question:** Does volume of trade affect unemployment?
  - controlling for overall health of the economy (GDP)
  - controlling for degree of exposure to trade shocks (CapMob)

- **Estimate the model:**
  ```r
  z.out <- zelig(unem ~ gdp + trade + capmob + as.factor(country),
                 model = "ls", data = macro)
  ```

- **Set explanatory variables:**
<table>
<thead>
<tr>
<th></th>
<th>Intercept</th>
<th>GDP</th>
<th>Trade</th>
<th>CapMob</th>
<th>US</th>
<th>Japan</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_{US}$</td>
<td>1</td>
<td>3.25</td>
<td>57.08</td>
<td>-0.89</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$X_{Japan}$</td>
<td>1</td>
<td>3.25</td>
<td>57.08</td>
<td>-0.89</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

  ```r
  x.US <- setx(z.out, country = "United States")
  x.Japan <- setx(z.out, country = "Japan")
  ```
Simulate quantities of interest:

```r
s.out <- sim(z.out, x = x.US, x1 = x.Japan)
summary(s.out)
```

Ceteris paribus, unemployment is lower in Japan than in the United States:

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>2.5%</th>
<th>97.5%</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E(Y \mid X_{US})$</td>
<td>11.37</td>
<td>0.65</td>
<td>10.14</td>
<td>12.67</td>
</tr>
<tr>
<td>$E(Y \mid X_{Japan}) - E(Y \mid X_{US})$</td>
<td>-4.63</td>
<td>0.55</td>
<td>-5.67</td>
<td>-3.54</td>
</tr>
</tbody>
</table>
Example 3: Model Fitting in Strata

Let \texttt{data} be a data set with variables \texttt{vote}, \texttt{age}, \texttt{race}, and \texttt{state}

To run a model on each state:

1. By hand: a batch file (containing all 50 commands):
   
   \begin{verbatim}
   AL.data <- subset(data, state == "Alabama")
   AL <- zelig(vote ~ age + race, data = AL.data,
               model = "logit")
   AZ.data <- subset(data, state == "Arizona")
   ...
   \end{verbatim}

2. A loop:
   
   \begin{verbatim}
   state.name <- order(unique(data$state))
   results <- list()
   for (i in 1:length(state.name)) {
     tmp <- subset(data, state = state.name[i])
     results[[i]] <- zelig(vote ~ age + race,
                          data = tmp, model = "ls")
   }
   \end{verbatim}
With Zelig:

```r
z.out <- zelig(vote ~ age + race, data = data, by = "state")
```
Example 4: Multiply Imputed Data Sets

- Many data sets come with missing values
- Listwise deletion assumes “missing completely at random”
- Multiple imputation: multiply impute missing values based on the prediction of a statistical model while accounting for the uncertainty about the imputation

Zelig syntax for the ordinal logit model:

```r
z.out <- zelig(as.factor(ipip) ~ wage1992 + prtyid + ideol, model = "ologit", data = mi(immi1, immi2, immi3, immi4, immi5))
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

- `setx()`, `sim()`, `summary()` do their jobs using all multiply imputed data sets., i.e., no syntax change
Example 5: Matching for Causal Inference

Matching as nonparametric preprocessing for reducing model dependence in causal inference (Ho, Imai, King, & Stuart, 2007)

The basic idea: making the treatment and control groups similar to each other in terms of pre-treatment covariates

Question: Do job training programs affect an individual’s real earnings?

**MatchIt** implements a variety of matching techniques:

```r
m.out <- matchit(treat ~ age + educ + black + hispan + nodegree + married + re74 + re75, method = "nearest", data = lalonde)
```
After matching, fit the model you would have fitted without matching anyway:

```r
z.out <- zelig(re78 ~ treat + age + educ + black + hispan + n degree + married + re74 + re75 + distance, 
data = match.data(m.out1), model = "ls")
```

where `distance` is the estimated propensity score

Computation of the average treatment effect for the treated:

```r
x.out0 <- setx(z.out, fn = NULL, treat = 0, 
data = match.data(m.out, "treat"))
x.out1 <- setx(z.out, fn = NULL, 
data = match.data(m.out, "treat"))
s.out <- sim(z.out, x = x.out0, x1 = x.out1)
summary(s.out)
```
Example 3: Multiple Equations Models

Question: Are import sanctions and export sanctions likely to occur in the same state?

Bivariate probit model:

1. Observation mechanism:

   \[ Y_j = \begin{cases} 
   1 & \text{if } Y_j^* \geq 0, \\
   0 & \text{otherwise.} 
   \end{cases} \]

2. Latent (unobserved) variable:

   \[
   \begin{pmatrix} Y_1^* \\ Y_2^* \end{pmatrix} \sim N_2 \left\{ \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \right\},
   \]

   where \( \mu_j \) is a mean for \( Y_j^* \) and \( \rho \) is a scalar correlation parameter given by,

   \[
   \mu_j = x_j \beta_j \quad \text{for } j = 1, 2, \\
   \rho = \frac{\exp(x_3 \beta_3) - 1}{\exp(x_3 \beta_3) + 1}.
   \]

Imai, King, & Lau (Princeton, Harvard, FDA)
Default: estimate only the two conditional mean equations with the same set of $X$ and have no $X$ for the correlation parameter

```r
z.out <- zelig(cbind(import, export) ~ coop +
               cost + target, model = "bprobit",
               data = sanction)
```

```r
x.lo <- setx(z.out, cost = 1)
x.hi <- setx(z.out, cost = 4)
s.out <- sim(z.out, x = x.lo, x1 = x.hi)
```

It’s possible to specify different variables in each equation:

```r
z.out <- zelig(list(mu1 = import ~ coop,
                    mu2 = export ~ cost + target),
               model = "bprobit", data = sanction)
```
With Zelig, it is even easy to constrain the parameters across different equations:

```r
z.out <-
zelig(list(mu1 = import ~ tag(coop,"coop") +
tag(cost, "cost") + tag(target, "target"),
        mu2 = export ~ tag(coop,"coop") +
tag(cost, "cost") + tag (target, "target")),
model = "bprobit", data = sanction)
```

- `setx()` and `sim()` steps are identical
Concluding Remarks

- Zelig provides a unified interface for both users and developers
- Makes R and its numerous functionalities accessible to applied researchers
- Many more improvements planned for Zelig
  1. Adding more models
  2. Collaboration with the Dataverse Network
  3. API to encourage more contributions
- Visit Zelig on the web at
  
  http://gking.harvard.edu/zelig/