# Estimating Uncertainty in Inferential Models

ME104: Linear Regression Analysis
Kenneth Benoit

August 24, 2012

# Simulation and bootstrapping

Used for:

- Gaining intuition about distributions and sampling
- Providing distributional information not distributions are not directly known, or cannot be assumed
- Acquiring uncertainty estimates

Both simulation and bootstrapping are numerical approximations of the quantities we are interested in. (Run the same code twice, and you get different answers)

We have already seen simulation in the illustrations of the Central Limit Theorem, in applications to estimating the mean of spending from sample means.

# Bootstrapping

- *Bootstrapping* refers to repeated resampling of data points with replacement
- Used to estimate the error variance (i.e. the standard error) of an estimate when the sampling distribution is unknown (or cannot be safely assumed)
- Robust in the absence of parametric assumptions
- Useful for some quantities for which there is no known sampling distribution, such as computing the standard error of a median

# Bootstrapping illustrated

```
. /*** illustrate bootstrap sampling ***/
. /* using sample to generate permutations of the sequence 1:10 */
. clear

. set obs 10
obs was 0, now 10

. gen x = _n

. list, clean

        x
  1.    1
  2.    2
  3.    3
  4.    4
  5.    5
  6.    6
  7.    7
  8.    8
  9.    9
 10.   10
```

# Bootstrapping illustrated

```
. bsample

. list, clean

        x
  1.    1
  2.    5
  3.    8
  4.    3
  5.    9
  6.    6
  7.    2
  8.    5
  9.    5
 10.    9
```

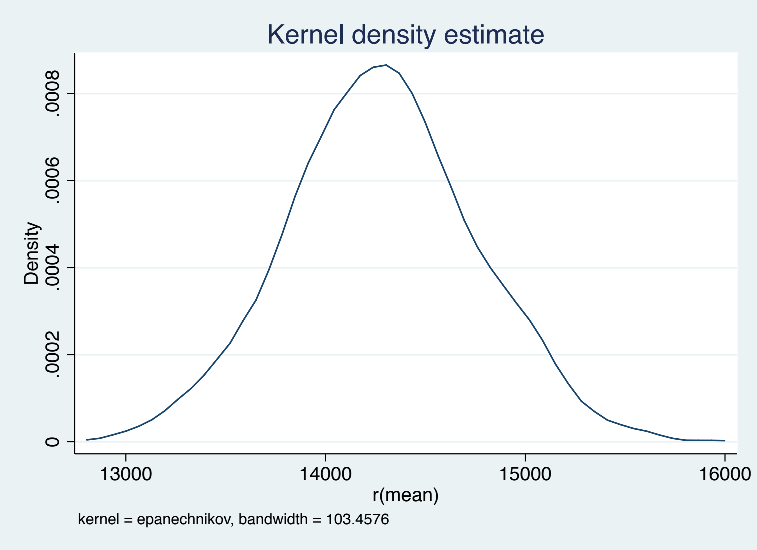
# Bootstrapping illustrated

```
. bsample

. list, clean

        x
  1.    5
  2.    1
  3.    8
  4.    5
  5.    6
  6.    3
  7.    9
  8.    2
  9.    8
 10.    3
```

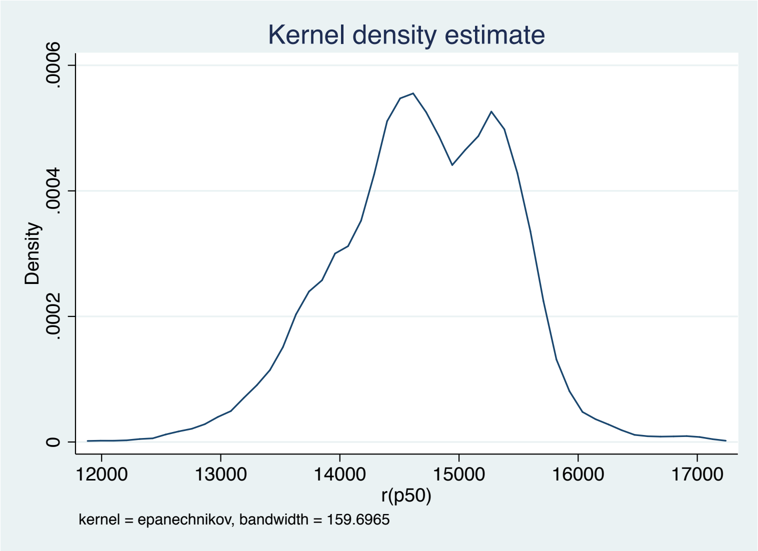
# Bootstrapping the standard error of the median

```
/* boostrap SE of median */
use dail2002.dta, clear
/* analytic std error of mean */
quietly summ spend_total, detail
di "mean = " r(mean) "    median = " r(p50)
di "analytic SE of mean = " r(mean) / sqrt(r(N))
bootstrap r(mean) r(p50), reps(1000) saving(day10bs1.dta, replace): ///
  summ spend_total, detail
use day10bs1, clear
list in 1/10, clean
rename _bs_1 BSmean
rename _bs_2 BSmedian
kdensity BSmean, name(meandens)
kdensity BSmedian, name(meddens)
```

# Bootstrapping the standard error of the mean



# Bootstrapping the standard error of the median



# Bootstrapping the standard errors of regression coefficients

```
/* boostrap SE of median */
use dail2002.dta, clear
/* analytic std error of mean */
quietly summ spend_total, detail
di "mean = " r(mean) "    median = " r(p50)
di "analytic SE of mean = " r(mean) / sqrt(r(N))
bootstrap r(mean) r(p50), reps(1000) saving(day10bs1.dta, replace): ///
   summ spend_total, detail
use day10bs1, clear
list in 1/10, clean
rename _bs_1 BSmean
rename _bs_2 BSmedian
kdensity BSmean, name(meandens)
kdensity BSmedian, name(meddens)
```

# Uncertainty in regression models: the linear case revisited

- Suppose we regress $y$ on $X$ to produce $b = (X'X)^{-1}X'y$
- Then we set explanatory variables to new values $X^p$ to predict $Y^p$
- The prediction $Y^p$ will have two forms of uncertainty:
  1. estimation uncertainty that can be reduced by increasing the sample size. Estimated a $\hat{y}^p = X^p b$ and depends on sample size through $b$
  2. fundamental variability comes from variability in the dependent variable around the expected value $\mathrm{E}(Y^p) = \mu = X^p\beta$ – even if we knew the true $\beta$

# Estimation uncertainty and fundamental variability

- We can decompose this as follows:

$$
\begin{aligned}
Y^p &= X^p b + \epsilon^p \\
\mathrm{Var}(Y^p) &= \mathrm{Var}(X^p b) + \mathrm{Var}(\epsilon^p) \\
&= X^p \mathrm{Var}(b)(X^p)' + \sigma^2 I \\
&= \sigma^2 X^p((X^p)'X^p)^{-1} + \sigma^2 I \\
&= \text{estimation uncertainty} + \text{fundamental variability}
\end{aligned}
$$

- It can be shown that the distribution of $\hat{Y}^p$ is:

$$\hat{Y}^p \sim N(X^p\beta, X^p \mathrm{Var}(b)(X^p)')$$

- and that the unconditional distribution of $Y^p$ is:

$$Y^p \sim N(X^p\beta,\ X^p\,\mathrm{Var}(b)(X^p)' + \sigma^2 I)$$

# Confidence intervals for predictions

- For any set of explanatory variables $x_0$, the predicted response is $\hat{y}_0 = x_0' \hat{\beta}$
- But this prediction also comes with uncertainty, and by extension, with a confidence interval
- Two types:
  - *predictions of future observations*: based on the prediction plus the variance of $\epsilon$ (Note: this is what we usually want)

$$\hat{y}_0 \pm t_{n-k-1}^{\alpha/2} \hat{\sigma} \sqrt{1 + x_o'(X'X)^{-1} x_0}$$

  - *prediction of mean response*: the average value of a $y_0$ with the characteristics $x_0$ – only takes into account the variance of $\hat{\beta}$

$$\hat{y}_0 \pm t_{n-k-1}^{\alpha/2} \hat{\sigma} \sqrt{x_0'(X'X)^{-1} x_0}$$

# Confidence intervals for predictions in R

```
> summary(m1)$coeff
                       Estimate   Std. Error   t value     Pr(>|t|)
(Intercept)         464.5955332 162.59752848  2.857335 4.466694e-03
spend_total           0.2041449   0.01155236 17.671273 1.154515e-53
incumb             4493.3251289 478.80828470  9.384393 2.962201e-19
spend_total:incumb   -0.1068943   0.02254283 -4.741832 2.832798e-06
> fivenum(dail$spend_total)    # what is typical spending profile
[1]     0.00  5927.32 14699.12 20812.66 51971.28
> x0 <- c(1, 75000, 1, 75000)  # set some predictor values
> (y0 <- sum(x0*coef(m1)))     # compute predicted response
[1] 12251.71
> fivenum(dail$votes1st)       # how typical is this response?
[1]    19.0  1151.5  3732.0  6432.0 14742.0
> quantile(dail$votes1st, .99, na.rm=T)  # versus 99th percentile
     99%
11138.44
> x0.df <- data.frame(incumb=1, spend_total=75000)
> predict(m1, x0.df)
       1
12251.71
> predict(m1, x0.df, interval="confidence")
       fit      lwr      upr
1 12251.71 10207.33 14296.09
> predict(m1, x0.df, interval="prediction")
       fit      lwr      upr
1 12251.71 8153.068 16350.36
```

# Fundamental and estimation variability for non-linear forms

- For well-known cases, we known both the expectation and the fundamental variability, e.g.
  - *Poisson* $E(Y) = e^{X\beta}$, $\text{Var}(Y) = \lambda$
  - *logistic* $E(Y) = \frac{1}{1+e^{-X\beta}}$, $\text{Var}(Y) = \pi(1-\pi)$
- Calculating the estimation variability is harder, but can be done using a linear approximation from the Taylor series. The Taylor series approximation of $\hat{y}^p = g(b)$ is:

  $$\hat{y}^p = g(b) = g(\beta) + g'(\beta)(b - \beta) + \cdots$$

  where $g'(\beta)$ is the first derivative of the functional form $g(\beta)$ with respect to $\beta$
- If we drop all but the first two terms, then

  $$\begin{aligned} \text{Var}(\hat{Y}^p) &\approx \text{Var}[g(\beta)] + \text{Var}[g'(\beta)(b - \beta)] \\ &= g'(\beta)\text{Var}(b)g'(\beta)' \end{aligned}$$

- This is known as the Delta method for calculating standard errors of predictions

# Example: Delta method for Poisson

- Consider the Poisson model, where the stochastic component is $Y \sim \frac{e^{-\lambda}\lambda^y}{y!}$ and the systematic component is $\lambda = e^{X\beta}$
- The fundamental variability is $\mathrm{Var}(Y|\lambda) = \lambda$
- To calculate the estimation variability:
  - calculate the first derivative matrix:

$$\begin{aligned} g'(\beta) &= \frac{\delta e^{X\beta}}{\delta\beta} \\ &= X \cdot e^{X\beta} \end{aligned}$$

    where the $\cdot$ operator is element-by-element multiplication
  - Then the estimated variance matrix of $\hat{Y}^p$ is:

$$\mathrm{Var}(\hat{Y}^p = (X^p \cdot e^{X^p b})\hat{\mathrm{Var}}(b)(X^p \cdot e^{X^p b})'$$

# Alternative: Estimating uncertainty through simulation

- King, Timz, and Wittenberg (2000) propose using statistical simulation to estimate uncertainty
- Notation:

  stochastic component $Y_i \sim f(\theta_i, \alpha)$
  systmatic component $\theta_i = g(X_i, \beta)$

  For example in a linear-normal model,
  $Y_i = N(\mu_i, \sigma^2)$ and $\mu_i = X_i\beta$

  simulated parameter vector $\hat{\gamma} = \text{vec}(\hat{\beta}, \hat{\alpha})$

  The central limit theorem tells us we can simulate $\gamma$ as

$$\tilde{\gamma} \sim \mathrm{N}(\hat{\gamma}, \hat{V}(\hat{\gamma}))$$

# Simulating predicted values

1. Using the algorithm in the previous subsection, draw one value of the vector $\tilde{\gamma} = \text{vec}(\tilde{\beta}, \tilde{\alpha})$.
2. Decide which kind of predicted value you wish to compute, and on that basis choose one value for each explanatory variable. Denote the vector of such values $X_c$.
3. Taking the simulated effect coefficients from the top portion of $\tilde{\gamma}$, compute $\tilde{\theta}_c = g(X_c, \tilde{\beta})$, where $g(\cdot,\cdot)$ is the systematic component of the statistical model.
4. Simulate the outcome variable $\tilde{Y}_c$ by taking a random draw from $f(\tilde{\theta}_c, \tilde{\alpha})$, the stochastic component of the statistical model.

Repeat this $M = 1000$ times to approximate the entire probability distribution of $Y_c$. Using this estimated distribution we can compute mean and SDs which will approximate the predicted values and their error.

# Simulating expected values

1. Following the procedure for simulating the parameters, draw one value of the vector $\tilde{\gamma} = \text{vec}(\tilde{\beta}, \tilde{\alpha})$.
2. Choose one value for each explanatory variable and denote the vector of values as $X_c$.
3. Taking the simulated effect coefficients from the top portion of $\tilde{\gamma}$, compute $\tilde{\theta}_c = g(X_c, \tilde{\beta})$, where $g(\cdot,\cdot)$ is the systematic component of the statistical model.
4. Draw $m$ values of the outcome variable $\tilde{Y}_c^{(k)}$ ($k = 1, \ldots, m$) from the stochastic component $f(\tilde{\theta}_c, \tilde{\alpha})$. This step simulates fundamental uncertainty.
5. Average over the fundamental uncertainty by calculating the the mean of the $m$ simulations to yield one simulated expected value $\tilde{E}(Y_c) = \sum_{k=1}^{m} \tilde{Y}_c^{(k)} \Big/ m$.

Note: It is $m$ that approximates the fundamental variability but Step 5 averages it away. A large enough $m$ will purge the simulated result of any fundamental uncertainty.

Repeat the entire process $M = 1000$ times to estimate the full probability distribution of $E(Y_c)$.

# Calculating standard errors in Zelig

```
## Examples from Homework 6
## titanic data qn3
titanic <- read.dta("titanic.dta")
levels(titanic$class) <- c("first","second","third","crew")
z.out <- zelig(survived ~ age+sex+class, model="logit", data=titanic)
summary(z.out)
x.kate <- setx(z.out, ageadults=1, sexman=1,
               classsecond=0, classthird=0, classcrew=0)
x.kate[1,] <- c(1,1,0,0,0,0)
x.leo <- setx(z.out, ageadults=1, sexman=1,
              classsecond=0, classthird=1, classcrew=0)
x.leo[1,] <- c(1,1,1,0,1,0)
summary(s.out <- sim(z.out, x=x.leo, x1=x.kate))
```

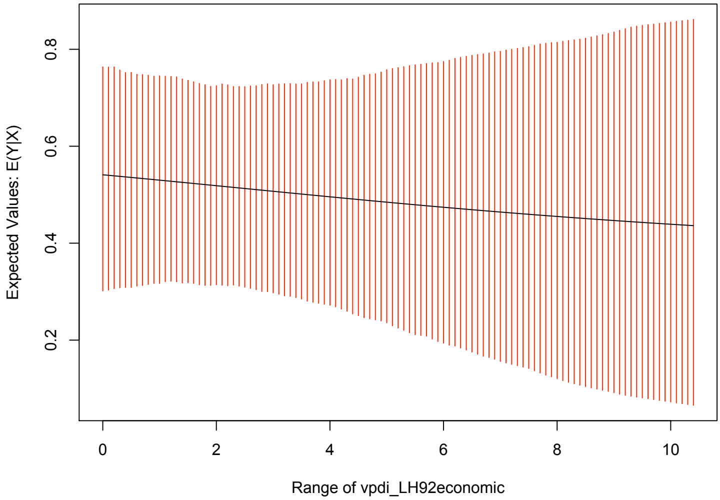
# Calculating standard errors in Zelig

```
> summary(s.out <- sim(z.out, x=x.leo, x1=x.kate))

Values of X
  (Intercept) ageadults sexman classsecond classthird classcrew
1           1         1      1           0          1         0

Values of X1
  (Intercept) ageadults sexman classsecond classthird classcrew
1           1         1      0           0          0         0

Expected Values: E(Y|X)
   mean      sd    2.5%  97.5%
1 0.105 0.01205 0.08251 0.1290

Predicted Values: Y|X
      0     1
1 0.888 0.112

First Differences in Expected Values: E(Y|X1)-E(Y|X)
    mean      sd   2.5%  97.5%
1 0.7791 0.02423 0.7291 0.8227

Risk Ratios: P(Y=1|X1)/P(Y=1|X)
   mean    sd  2.5% 97.5%
1 8.538 1.062 6.723 10.89
```

# More standard errors in Zelig

```
## economic_bills data qn4b
ecbills <- read.dta("economic_bills.dta")
z.out <- zelig(status ~ cabinet + vpdi_LH92economic + xland,
               model="logit", data=ecbills)
x.out <- setx(z.out)
x.out[1,] <- c(1,0,0,0,0,1)
summary(sim(z.out, x.out))
# for comparison:
predict(log2,new=data.frame(cabinet=0,vpdi_LH92economic=0,xland="UK"),
        type="response", se=T)

## economic_bills data qn4c
x.out[1,] <- c(1,1,5,1,0,0)
summary(sim(z.out, x.out))
# for comparison:
predict(log2,new=data.frame(cabinet=1,vpdi_LH92economic=5,xland="FRA"),
        type="response",se=T)

## economic_bills data qn4d
(x.out <- setx(z.out, vpdi_LH92economic=seq(0,10.4,.1)))
x.out[,2] <- 0
x.out[,5] <- 1
s.out <- sim(z.out, x.out)
plot.ci(s.out)
lines(seq(0,10.4,.1), apply(s.out$qi$ev,2,mean))
```

# Plot from Homework 6 Question 4d



# Predicted values from Benoit (1996)

```
> weede <- read.dta("weede.dta")
> z.out <- zelig(ssal6080 ~
+                 fh73+lpopln70+lmilwp70, model="poisson", data=weede)
> (x.out <- setx(z.out, fh73=2:14))
   (Intercept) fh73 lpopln70 lmilwp70
1            1    2    4.036    0.954
2            1    3    4.036    0.954
3            1    4    4.036    0.954
4            1    5    4.036    0.954
5            1    6    4.036    0.954
6            1    7    4.036    0.954
7            1    8    4.036    0.954
8            1    9    4.036    0.954
9            1   10    4.036    0.954
10           1   11    4.036    0.954
11           1   12    4.036    0.954
12           1   13    4.036    0.954
13           1   14    4.036    0.954
> s.out <- sim(z.out, x=x.out)
> summary(s.out)

  Model: poisson
  Number of simulations: 1000

Mean Values of X (n = 13)
(Intercept)        fh73     lpopln70     lmilwp70
      1.000       8.000        4.036        0.954

Pooled Expected Values: E(Y|X)
  mean     sd   2.5%  97.5%
0.3221 0.1085 0.1449 0.5697

Pooled Predicted Values: Y|X
  mean     sd   2.5%  97.5%
0.3259 0.5840 0.0000 2.0000
```

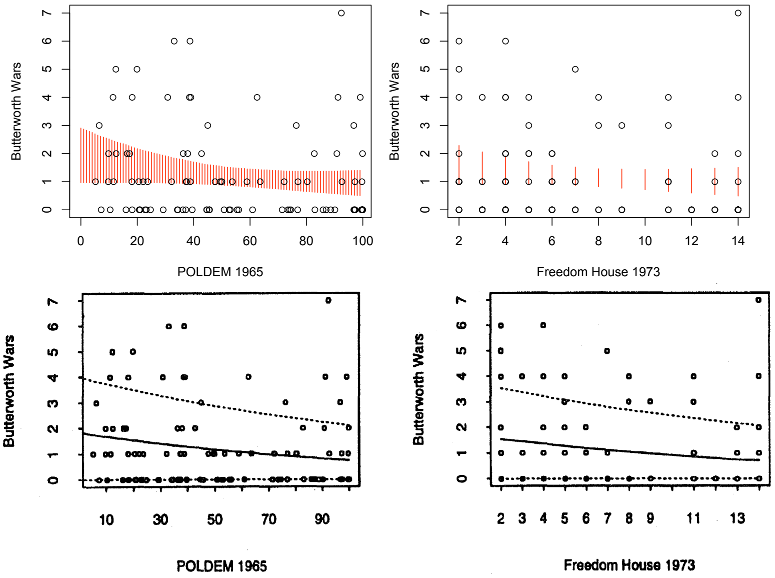
# Replicate part of Table 3 from Benoit (1996)

```
> ## replicate part of Table 3 from Benoit (1996)
> z.tab2NBpoldem <- zelig(butterw ~ poldem65, model="negbin", data=weede)
> x.tab2NBpoldem <- setx(z.tab2NBpoldem, poldem65=c(0,20,55,85,100))
> s.tab2NBpoldem <- sim(z.tab2NBpoldem, x=x.tab2NBpoldem)
> cbind(apply(s.tab2NBpoldem$qi$ev, 2, mean),
+       apply(s.tab2NBpoldem$qi$ev, 2, sd))
       [,1]   [,2]
[1,] 1.7378 0.4969
[2,] 1.4819 0.3092
[3,] 1.1445 0.1644
[4,] 0.9364 0.1971
[5,] 0.8532 0.2290
> x.tab2NBfh73 <- setx(z.tab2NBfh73, fh73=c(2,4,7,12,14))
> s.tab2NBfh73 <- sim(z.tab2NBfh73, x=x.tab2NBfh73)
> cbind(apply(s.tab2NBfh73$qi$ev, 2, mean),
+       apply(s.tab2NBfh73$qi$ev, 2, sd))
       [,1]   [,2]
[1,] 1.4611 0.3421
[2,] 1.3210 0.2414
[3,] 1.1470 0.1709
[4,] 0.9308 0.2273
[5,] 0.8642 0.2674
```

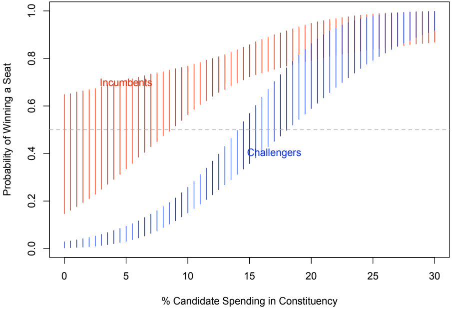
TABLE 3
Fitted Values: Bivariate Negative Binomial Model

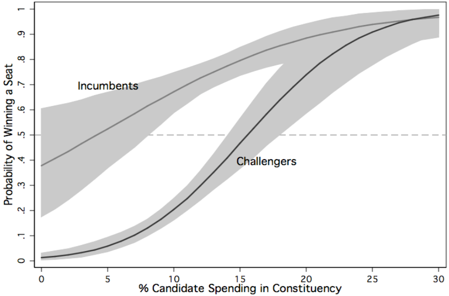
| | *Expected War Count* | | | *Expected War Count* | |
|---|---|---|---|---|---|
| *POLDEM 1965* | *Butterworth* | *Small-Singer* | *Freedom House 1973* | *Butterworth* | *Small-Singer* |
| 0 | 1.84 | 0.79 | 2 | 1.55 | 0.66 |
| 20 | 1.53 | 0.62 | 4 | 1.36 | 0.55 |
| 55 | 1.10 | 0.42 | 7 | 1.12 | 0.42 |
| 85 | 0.84 | 0.30 | 12 | 0.81 | 0.27 |
| 100 | 0.73 | 0.25 | 14 | 0.71 | 0.23 |
| Mean *SE* | (0.27) | (0.14) | | (0.23) | (0.11) |

# Replicate top part of Figure 1 from Benoit (1996)



# Replicate Benoit and Marsh (PRQ, 2009) Figure 2



# Compare models fits using a Receiver Operating Characteristic (ROC) plot

```
## plot an ROC plot comparing challengers v. incumbent predictions
dail.incumb <- subset(dail, incumb==1, select=c(wonseat,pspend_total,incumb,m))
dail.chall <- subset(dail, incumb==0, select=c(wonseat,pspend_total,incumb,m))
z.out.i <- zelig(wonseat ~ pspend_total+m, model="probit", data=dail.incumb)
z.out.c <- zelig(wonseat ~ pspend_total+m, model="probit", data=dail.chall)
rocplot(z.out.i$y, z.out.c$y, fitted(z.out.i), fitted(z.out.c),
        lty1="solid", lty2="solid", col2="blue", col1="red")
text(.6,.55,"Incumbents",col="red")
text(.8,.85,"Challengers",col="blue")
```