# CSSS 510: Lab 3

Logistic Regression

*2017-10-13*

## 0. Agenda

1. Deriving a likelihood function for the logistic regression model

2. Fitting a logit model using `optim()` and `glm()`

3. Simulating predicted values and confidence intervals

4. Simulating first differences

## 1. Deriving a likelihood function for the logistic regression model
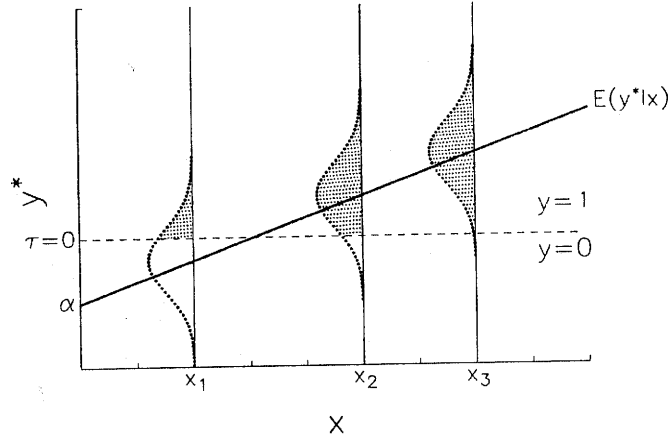
Recall from lecture the logit model:

$$y_i \sim \text{Bern}(y_i|\pi_i)$$
$$\pi_i = \text{logit}^{-1}(\boldsymbol{x}_i\boldsymbol{\beta})$$
$$\pi_i = \frac{\exp(\boldsymbol{x}_i\boldsymbol{\beta})}{1 + \exp(\boldsymbol{x}_i\boldsymbol{\beta})} = \frac{1}{1 + \exp(-\boldsymbol{x}_i\boldsymbol{\beta})}$$

In the simple case, this stems from the latent variable model:

$$y^* = \beta_0 + \beta_1 x + \epsilon$$

where the relationship between latent variable $y^*$ and the explanatory variable $x$ is modeled using simple linear regression, and the binary outcome $y$ is a function of the sign of $y^*$:

$$y = \begin{cases} 1, & \text{if } y^* > 0 \\ 0, & \text{if } y^* \leq 0 \end{cases} \tag{1}$$

**Figure 3.2.** The Distribution of $y^*$ Given $x$ in the Binary Response Model
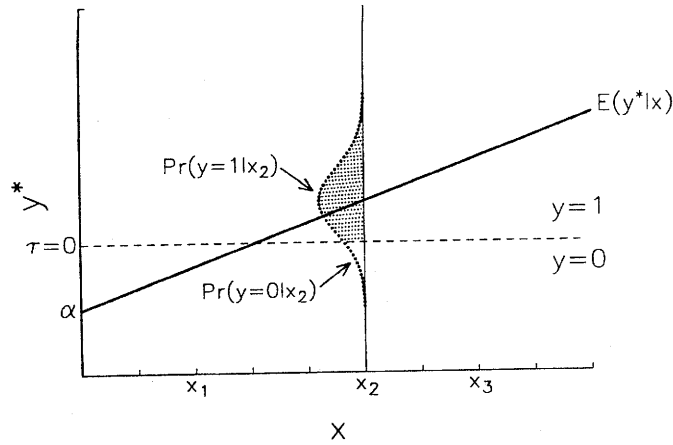
The logistic regression model is obtained if we assume the errors of this latent variable model follow a standard logistic distribution. Recall that the pdf and cdf of the standard logisitic distribution are as follows:
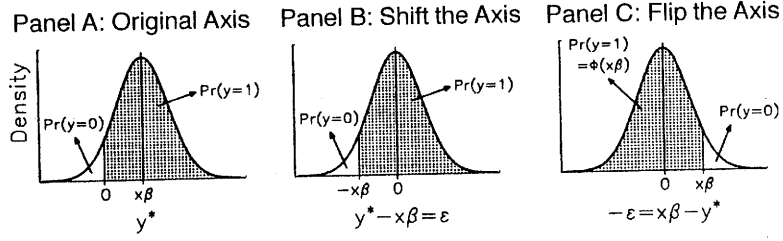
$$f(t) = \frac{\exp(t)}{(1 + \exp(t))^2}$$

$$F(t) = \frac{\exp(t)}{1 + \exp(t)}$$

We therefore have the following:

$$\begin{aligned}
\Pr(y = 1|x) &= \Pr(y^* > 0|x) \\
&= \Pr(\beta_0 + \beta_1 x + \epsilon > 0|x) \\
&= \Pr(\epsilon > -(\beta_0 + \beta_1 x)) \\
&= \Pr(\epsilon < \beta_0 + \beta_1 x) \\
&= F(\beta_0^L + \beta_1^L x)
\end{aligned}$$



**Figure 3.4.** Probability of Observed Values in the Binary Response Model
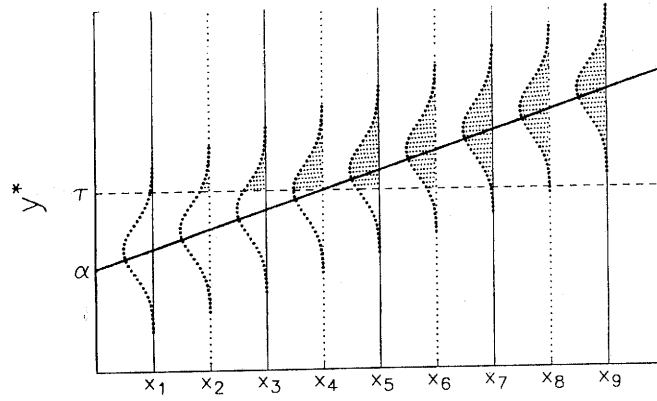
2

**Figure 3.5.** Computing $\Pr(y = 1 \mid x)$ in the Binary Response Model

Since we assume the errors follow a standard logistic distribution, we have
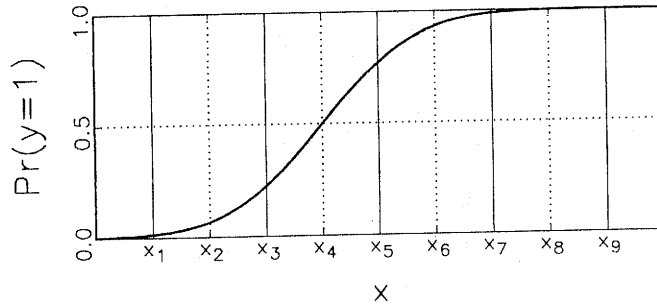
$$\Pr(y = 1 | x) = F(\beta_0^L + \beta_1^L x) = \frac{\exp(\beta_0^L + \beta_1^L x)}{1 + \exp(\beta_0^L + \beta_1^L x)}$$

and $E(\epsilon) = 0$ and $\text{Var}(\epsilon) = \frac{\pi^2}{3}$.



**Figure 3.6.** Plot of $y^*$ and $\Pr(y = 1 | \mathbf{x})$ in the Binary Response Model

The logit function is the inverse of the logistic function:

$$\text{logit}(p) = \log \frac{p}{1 - p}$$

3

or

$$\text{logit}^{-1}(p) = \frac{\exp(x)}{1 + \exp(x)}$$

We therefore have the following

$$\Pr(y = 1|x) = \text{logit}^{-1}(\beta_1^L + \beta_1^L x)$$

or

$$\text{logit}(\Pr(y = 1|x)) = \beta_1^L + \beta_1^L x$$

or

$$\log\frac{\Pr(y = 1|x)}{\Pr(y = 0|x)} = \beta_0^L + \beta_1^L x.$$

Recall from lecture that a Bernoulli distribution has the following pdf:

$$\Pr(y_i = 1|\pi_i) = \pi_i^{y_i}(1 - \pi_i)^{1-y_i}$$

And the likelihood function can be derived from the joint probability:

$$\mathcal{L}(\boldsymbol{\pi}|\boldsymbol{y}) \propto \prod_{i=1}^{n} \pi_i^{y_i}(1 - \pi_i)^{1-y_i}$$

$$\mathcal{L}(\boldsymbol{\beta}|\boldsymbol{y}) \propto \prod_{i=1}^{n} \left(\frac{1}{1 + \exp(-\boldsymbol{x}_i\boldsymbol{\beta})}\right)^{y_i} \left(1 - \frac{1}{1 + \exp(-\boldsymbol{x}_i\boldsymbol{\beta})}\right)^{1-y_i}$$

$$\mathcal{L}(\boldsymbol{\beta}|\boldsymbol{y}) \propto \prod_{i=1}^{n} (1 + \exp(-\boldsymbol{x}_i\boldsymbol{\beta}))^{-y_i}(1 + \exp(-\boldsymbol{x}_i\boldsymbol{\beta}))^{-(1-y_i)}$$

$$\log\mathcal{L}(\boldsymbol{\beta}|\boldsymbol{y}) \propto \sum_{i=1}^{n} -y_i\log(1 + \exp(-\boldsymbol{x}_i\boldsymbol{\beta})) - (1 - y_i)\log(1 + \exp(\boldsymbol{x}_i\boldsymbol{\beta}))$$

## 2. Fitting a logit model using **optim**() and **glm**()

```
rm(list = ls()) # clear up the memory

#install and load the packages needed
#from CRAN: install.packages("MASS", dependencies = TRUE)
library(MASS)
library(RColorBrewer)

#download simcf and tile packages from- http://faculty.washington.edu/cadolph/software
# don't unzip the archive (tar) file
library(simcf)
library(tile)
```

```
## Loading required package: grid
```

4

```r
# Load data
file <- "nes00a.csv"
data <- read.csv(file, header=TRUE)

# attach(data)

# Estimate logit model using optim()
# Construct variables and model objects
y <- data$vote00
x <- cbind(data$age,data$hsdeg,data$coldeg)

# Likelihood function for logit
llk.logit <- function(param,y,x) {
  os <- rep(1,length(x[,1])) # constant
  x <- cbind(os,x) # constant+covariates
  b <- param[ 1 : ncol(x) ]
  # number of parameters to be estimated equals number of columns in x
  # (i.e, one for constant and one for each covariates : total 4)
  xb <- x%*%b
  sum( y*log(1+exp(-xb)) + (1-y)*log(1+exp(xb))) # log-likelihood function for logit model
  # (based on our choice of standard logistic cdf as the systematic component)
              # optim is a minimizer, so use -lnL here
}

# Fit logit model using optim
ls.result <- lm(y~x)  # use ls estimates as starting values (for convenience)
stval <- ls.result$coefficients  # initial guesses
logit.result.opt <- optim(stval,llk.logit,method="BFGS",hessian=T,y=y,x=x)
                # call minimizer procedure or max by adding control=list(fnscale=-1)
pe.opt <- logit.result.opt$par   # point estimates
vc.opt <- solve(logit.result.opt$hessian)  # var-cov matrix
se.opt <- sqrt(diag(vc.opt))     # standard errors
ll.opt <- -logit.result.opt$value  # likelihood at maximum

logit.optim<-data.frame(cbind(round(pe.opt,3), round(se.opt,3)))
rownames(logit.optim)<-c("intercept", "age", "highschool" , "college")
colnames(logit.optim)<-c("pe", "std.err")
logit.optim
```

```
##                pe std.err
## intercept  -2.149   0.257
## age          0.031   0.003
## highschool   1.213   0.179
## college      1.102   0.130
```

```r
#p-value based on t-statistics
2*pt(abs(logit.optim$pe/logit.optim$std.err), df=length(y)-length(pe.opt) , lower.tail = FALSE)
```

```
## [1] 1.229175e-16 2.393414e-24 1.668155e-11 4.780627e-17
```

```r
# Estimate logit model using glm()

# Run logit & extract results using glm.
# GLM solves the likelihood equations with a common numeric algorithm
# called iteratively re-weighted least squares (IRWLS).
```

```r
logit.result <- glm(vote00~age +hsdeg+coldeg, family=binomial, data=data)
# family "binomial" calls logit transformation (log(pi/1-pi)) as a "link function"
# corresponding to logistic distribution.
# Link function transforms pi so that it follows a linear model.
# (So although pi itself is dependent on covariates in a non-linear way,
# logit transformed pi is dependent on covariates in a linear way.)

summary(logit.result)
```

```
##
## Call:
## glm(formula = vote00 ~ age + hsdeg + coldeg, family = binomial,
##     data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4487  -1.1347   0.6360   0.8973   1.8854
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.147997   0.256608  -8.371  < 2e-16 ***
## age          0.030885   0.003386   9.121  < 2e-16 ***
## hsdeg        1.212882   0.179447   6.759 1.39e-11 ***
## coldeg       1.102465   0.130426   8.453  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2293.5  on 1782  degrees of freedom
## Residual deviance: 2076.0  on 1779  degrees of freedom
## AIC: 2084
##
## Number of Fisher Scoring iterations: 4
```

```r
# now a new model adding age^2
model <- vote00 ~ age + I(age^2) + hsdeg + coldeg
mdata <- extractdata(model, data, na.rm=TRUE) # needs library(simcf)

logit.result <- glm(model, family=binomial, data=mdata)
summary(logit.result)
```

```
##
## Call:
## glm(formula = model, family = binomial, data = mdata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2045  -1.1145   0.6335   0.8743   1.9841
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.0193891  0.4181899  -7.220 5.19e-13 ***
## age          0.0747252  0.0168440   4.436 9.15e-06 ***
```

```
## I(age^2)    -0.0004427  0.0001655  -2.674  0.00749 **
## hsdeg        1.1243908  0.1800069   6.246 4.20e-10 ***
## coldeg       1.0795702  0.1312113   8.228  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2293.5  on 1782  degrees of freedom
## Residual deviance: 2069.0  on 1778  degrees of freedom
## AIC: 2079
##
## Number of Fisher Scoring iterations: 4
```

```
pe <- logit.result$coefficients  # point estimates
vc <- vcov(logit.result)         # var-cov matrix


pe
```

```
##   (Intercept)          age      I(age^2)         hsdeg        coldeg
## -3.0193890720  0.0747251922 -0.0004427014  1.1243907749  1.0795702357
```

```
sqrt(diag(vc))
```

```
##   (Intercept)          age      I(age^2)         hsdeg        coldeg
## 0.4181899367 0.0168440337 0.0001655466 0.1800068893 0.1312113265
```

## 3. Simulating predicted values and confidence intervals

```
# Simulate parameter distributions
sims <- 10000
simbetas <- mvrnorm(sims, pe, vc) #needs library(MASS) # draw 10000 sets of simulated
# parameter (beta) estimates from a multivariate normal distribution with
# mean pe and variance-covariance vc

# Now let's plan counterfactuals: We will have three sets of counterfactuals based
# on education level (less than hs edu, hs edu, college or higher edu), and for each set
# we will make age varies between 18 years old and 97 years old.

# Set up counterfactuals:  all ages, each of three educations
xhyp <- seq(18,97,1)  # create age vector
nscen <- length(xhyp)  # we will have total 80 different age scenarios for each education level

nohsScen <- hsScen <- collScen <- cfMake(model, mdata, nscen)  #this is just to initialize
# 80 scenarios for each education level. As default, all covariate values are set at the mean.

# Create three sets of education counterfactuals


for (i in 1:nscen) {
  # No High school scenarios (loop over each age, total 80 scenarios)
  nohsScen <- cfChange(nohsScen, "age", x = xhyp[i], scen = i)
  nohsScen <- cfChange(nohsScen, "hsdeg", x = 0, scen = i) # no hs degree
```

```r
  nohsScen <- cfChange(nohsScen, "coldeg", x = 0, scen = i) # no college degree

  # HS grad scenarios (loop over each age, total 80 scenarios)
  hsScen <- cfChange(hsScen, "age", x = xhyp[i], scen = i)
  hsScen <- cfChange(hsScen, "hsdeg", x = 1, scen = i) # has hs degree
  hsScen <- cfChange(hsScen, "coldeg", x = 0, scen = i)  # no college degree

  # College grad scenarios (loop over each age, total 80 scenarios)
  collScen <- cfChange(collScen, "age", x = xhyp[i], scen = i)
  collScen <- cfChange(collScen, "hsdeg", x = 1, scen = i) # has hs degree
  collScen <- cfChange(collScen, "coldeg", x = 1, scen = i) # has college degree
}

# # Now given the counterfactual covariates (nohsScen/hsScen/collScen) and simulated
# parameters (simbetas), we can calculate expected value of the response.
# In this case, expected probability of voting!

head(nohsScen$x) #we will fit the counterfactual data
```

```
##      vote00 age hsdeg coldeg
## 1 0.6567583  18     0      0
## 2 0.6567583  19     0      0
## 3 0.6567583  20     0      0
## 4 0.6567583  21     0      0
## 5 0.6567583  22     0      0
## 6 0.6567583  23     0      0
```

```r
nohsScen$model # in the model specification
```

```
## vote00 ~ age + I(age^2) + hsdeg + coldeg
```

```r
nohsSims <- logitsimev(nohsScen, simbetas, ci=0.95) # using simulated betas to get expected values.
# Built-in function "logitsimev" calculates the expected value
# for every individual scenario you created.

nohsSims #reports lower and upper confidence intervals as well as expected probabilities
```

```
## $pe
##  [1] 0.1422056 0.1492928 0.1565685 0.1640250 0.1716533 0.1794440 0.1873866
##  [8] 0.1954699 0.2036821 0.2120108 0.2204428 0.2289649 0.2375632 0.2462237
## [15] 0.2549320 0.2636738 0.2724348 0.2812007 0.2899573 0.2986908 0.3073875
## [22] 0.3160341 0.3246179 0.3331263 0.3415476 0.3498703 0.3580837 0.3661775
## [29] 0.3741420 0.3819682 0.3896476 0.3971725 0.4045354 0.4117296 0.4187490
## [36] 0.4255880 0.4322414 0.4387046 0.4449733 0.4510440 0.4569131 0.4625778
## [43] 0.4680354 0.4732837 0.4783206 0.4831445 0.4877538 0.4921473 0.4963239
## [50] 0.5002827 0.5040231 0.5075443 0.5108460 0.5139276 0.5167890 0.5194299
## [57] 0.5218500 0.5240494 0.5260279 0.5277855 0.5293222 0.5306381 0.5317332
## [64] 0.5326076 0.5332614 0.5336949 0.5339083 0.5339018 0.5336758 0.5332307
## [71] 0.5325670 0.5316854 0.5305865 0.5292712 0.5277406 0.5259957 0.5240379
## [78] 0.5218688 0.5194902 0.5169041
##
## $lower
##  [1] 0.09462001 0.10076701 0.10711318 0.11338659 0.11979642 0.12636419
##  [7] 0.13324556 0.13995945 0.14664476 0.15381488 0.16084469 0.16798761
## [13] 0.17535866 0.18278946 0.18991940 0.19764948 0.20513516 0.21270613
```

```
## [19]  0.22012991 0.22709536 0.23448093 0.24188292 0.24973687 0.25713582
## [25]  0.26412924 0.27165695 0.27907124 0.28631854 0.29342732 0.30073112
## [31]  0.30785122 0.31518519 0.32214795 0.32914661 0.33586961 0.34289391
## [37]  0.34947846 0.35580593 0.36229721 0.36875320 0.37475110 0.38081153
## [43]  0.38632098 0.39195289 0.39706840 0.40201306 0.40719532 0.41216726
## [49]  0.41661905 0.42096286 0.42472045 0.42799629 0.43091905 0.43398261
## [55]  0.43611760 0.43897488 0.44031202 0.44167518 0.44325491 0.44358500
## [61]  0.44372495 0.44272892 0.44153218 0.43978132 0.43775872 0.43549057
## [67]  0.43230510 0.42806518 0.42391334 0.42024731 0.41456811 0.40877094
## [73]  0.40271664 0.39675468 0.39058225 0.38363108 0.37534414 0.36709831
## [79]  0.35923807 0.35126345
##
## $upper
##  [1]  0.2038870 0.2113949 0.2197686 0.2277887 0.2363825 0.2443352 0.2531059
##  [8]  0.2618440 0.2711937 0.2803819 0.2897733 0.2993146 0.3089029 0.3187653
## [15]  0.3284401 0.3380427 0.3480391 0.3577787 0.3678968 0.3780474 0.3874300
## [22]  0.3968302 0.4063089 0.4156068 0.4248966 0.4334294 0.4420197 0.4507426
## [29]  0.4588226 0.4672926 0.4751844 0.4831463 0.4904675 0.4978749 0.5046963
## [36]  0.5111686 0.5179373 0.5243776 0.5302215 0.5357234 0.5408494 0.5458061
## [43]  0.5508134 0.5551815 0.5601691 0.5645725 0.5682190 0.5716638 0.5752057
## [50]  0.5786059 0.5824781 0.5858825 0.5890275 0.5917610 0.5949825 0.5986554
## [57]  0.6021808 0.6058911 0.6090835 0.6118537 0.6148080 0.6177133 0.6210798
## [64]  0.6243420 0.6279782 0.6310718 0.6353195 0.6388926 0.6420775 0.6452119
## [71]  0.6484632 0.6519164 0.6556097 0.6593564 0.6628853 0.6661776 0.6694192
## [78]  0.6737447 0.6773373 0.6808137
```

```r
# same thing for two other sets of sceneraios
hsSims <- logitsimev(hsScen, simbetas, ci=0.95)
collSims <- logitsimev(collScen, simbetas, ci=0.95)

# Get 3 nice colors for traces
col <- brewer.pal(3,"Dark2")

# Set up lineplot traces of expected probabilities

#Traces are elements of tile : lines, labels, legends...

# no hs
nohsTrace <- lineplot(x=xhyp, # age on x-axis
                      y=nohsSims$pe, #expected probability on y-axis
                      lower=nohsSims$lower, # lower confidence interval
                      upper=nohsSims$upper, #upper confidence interval
                      col=col[1], #color choice
                      extrapolate=list(data=mdata[,2:ncol(mdata)],
                                       #actual covariates (i.e., values in your data)
                                  cfact=nohsScen$x[,2:ncol(hsScen$x)],#counterfactual covariates
                        omit.extrapolated=FALSE), #don't show extrapolated values
                      plot=1)

# hs but no college
hsTrace <- lineplot(x=xhyp,
                    y=hsSims$pe,
                    lower=hsSims$lower,
                    upper=hsSims$upper,
```

```
                    col=col[2],
                    extrapolate=list(data=mdata[,2:ncol(mdata)],
                                     cfact=hsScen$x[,2:ncol(hsScen$x)],
                                     omit.extrapolated=FALSE),
                    plot=1)

#college
collTrace <- lineplot(x=xhyp,
                      y=collSims$pe,
                      lower=collSims$lower,
                      upper=collSims$upper,
                      col=col[3],
                      extrapolate=list(data=mdata[,2:ncol(mdata)],
                                       cfact=collScen$x[,2:ncol(hsScen$x)],
                                       omit.extrapolated=FALSE),
                      plot=1)

# Set up traces with labels

labelTrace <- textTile(labels=c("Less than HS", "High School", "College"),
                       x=c( 55,     49,      30),
                       y=c( 0.26,  0.56,    0.87),
                       col=col,
                       plot=1)

# For legend

legendTrace <- textTile(labels=c("Logit estimates:", "95% confidence", "interval is shaded"),
                        x=c(82, 82, 82),
                        y=c(0.2, 0.16, 0.12),
                        plot=1)


#options(device="quartz")

# Plot traces using tile
voting<-tile(nohsTrace,
     hsTrace,
     collTrace,
     labelTrace,
     legendTrace,
     limits=c(18,94,0,1),
     xaxis=list(at=c(20,30,40,50,60,70,80,90)),
     yaxis=list(label.loc=-0.5, major=FALSE),
     xaxistitle=list(labels="Age of Respondent"),
     yaxistitle=list(labels="Probability of Voting"),
     width=list(null=5,yaxistitle=4,yaxis.labelspace=-0.5)
     #,output=list(file="educationEV",width=5.5)
)

## Loading required package: WhatIf

## Loading required package: lpSolve

## ######################################################
```
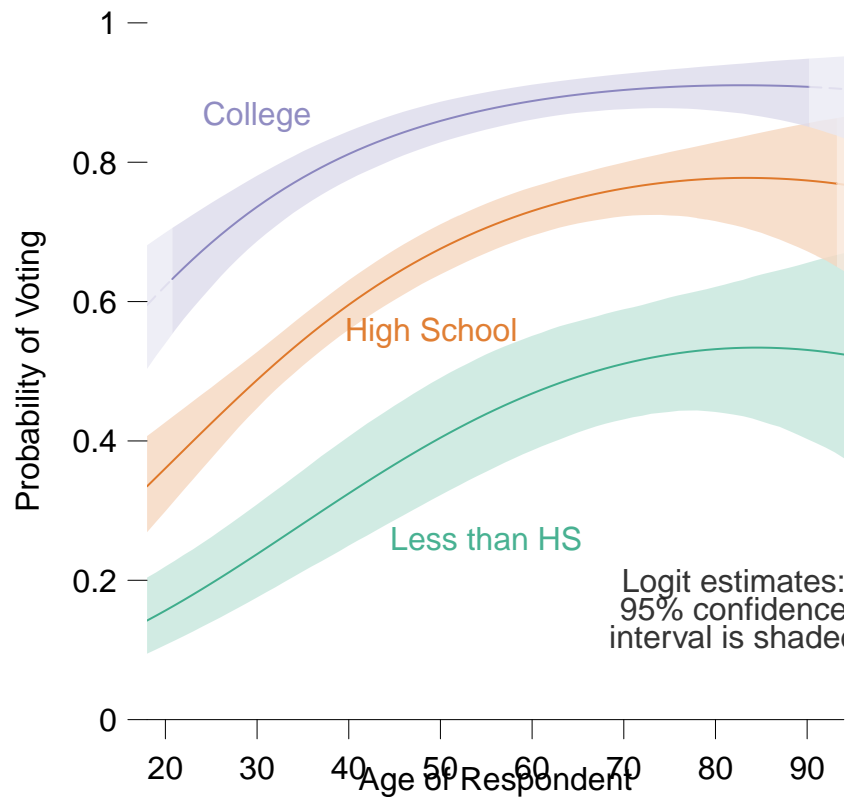
```
## ##
## ##  WhatIf (Version 1.5-6, built 2014-01-06)
## ##  Complete documentation available from http://gking.harvard.edu/whatif
## ##
## ###########################################################

## [1] "Running whatif"
## [1] "Preprocessing data ..."
## [1] "Performing convex hull test ..."
## [1] "Calculating distances ...."
## [1] "Calculating the geometric variance..."
## [1] "Calculating cumulative frequencies ..."
## [1] "Finishing up ..."
## [1] "Whatif finished; returning to tile"
## [1] "Running whatif"
## [1] "Preprocessing data ..."
## [1] "Performing convex hull test ..."
## [1] "Calculating distances ...."
## [1] "Calculating the geometric variance..."
## [1] "Calculating cumulative frequencies ..."
## [1] "Finishing up ..."
## [1] "Whatif finished; returning to tile"
## [1] "Running whatif"
## [1] "Preprocessing data ..."
## [1] "Performing convex hull test ..."
## [1] "Calculating distances ...."
## [1] "Calculating the geometric variance..."
## [1] "Calculating cumulative frequencies ..."
## [1] "Finishing up ..."
## [1] "Whatif finished; returning to tile"
```

# 4. Simulating first differences

```
################################################################
#
# Now consider a new specification adding the variable
# "ever married", or marriedo
#
# We will estimate this new model with glm(), then
# simulate new scenarios for marrieds and non-marrieds


# Estimate logit model using glm()


# Set up a new model formula and model specific data frame

model2 <- vote00 ~ age + I(age^2) + hsdeg + coldeg + marriedo
mdata2 <- extractdata(model2, data, na.rm=TRUE)

# Run logit & extract results
logit.m2 <- glm(model2, family=binomial, data=mdata2)
pe.m2 <- logit.m2$coefficients  # point estimates
vc.m2 <- vcov(logit.m2)          # var-cov matrix


# Simulate parameter distributions
```

```r
sims <- 10000
simbetas.m2 <- mvrnorm(sims, pe.m2, vc.m2)


# Set up counterfactuals:  all ages

xhyp <- seq(18,97,1)
nscen <- length(xhyp)
marriedScen <- notmarrScen <- cfMake(model2, mdata2, nscen)
for (i in 1:nscen) {


  #  - we will use the marriedScen counterfactuals in FDs and RRs as well as EVs
  # Note below the careful use of before scenarios (xpre) and after scenarios (x)
  # :i.e., use of the same age range (18-97) for both x and xpre, only marriedo values differ.

  # Married (loop over each age)
  marriedScen <- cfChange(marriedScen, "age", x = xhyp[i], xpre= xhyp[i], scen = i)
  marriedScen <- cfChange(marriedScen, "marriedo", x = 1, xpre= 0, scen = i)

  # Not Married (loop over each age)
  notmarrScen <- cfChange(notmarrScen, "age", x = xhyp[i], scen = i)
  notmarrScen <- cfChange(notmarrScen, "marriedo", x = 0, scen = i)
}

# Simulate expected probabilities for all age scenarios for married and not married respectively
marriedSims <- logitsimev(marriedScen, simbetas.m2, ci=0.95)
notmarrSims <- logitsimev(notmarrScen, simbetas.m2, ci=0.95)

# Simulate first difference of voting wrt marriage: E(y|married)-E(y|notmarried)
marriedFD <- logitsimfd(marriedScen, simbetas.m2, ci=0.95)

# Simulate relative risk of voting wrt marriage: E(y|married)/E(y|notmarried)
marriedRR <- logitsimrr(marriedScen, simbetas.m2, ci=0.95)


## Make plots using tile

# Get 3 nice colors for traces
col <- brewer.pal(3,"Dark2")

# Set up lineplot traces of expected probabilities
marriedTrace <- lineplot(x=xhyp,
                         y=marriedSims$pe,
                         lower=marriedSims$lower,
                         upper=marriedSims$upper,
                         col=col[1],
                         extrapolate=list(data=mdata2[,2:ncol(mdata2)],
                                          cfact=marriedScen$x[,2:ncol(marriedScen$x)],
                                          omit.extrapolated=TRUE),
                         plot=1)

notmarrTrace <- lineplot(x=xhyp,
```

```r
                          y=notmarrSims$pe,
                          lower=notmarrSims$lower,
                          upper=notmarrSims$upper,
                          col=col[2],
                          ci = list(mark="dashed"),
                          extrapolate=list(data=mdata2[,2:ncol(mdata2)],
                                          cfact=notmarrScen$x[,2:ncol(notmarrScen$x)],
                                          omit.extrapolated=TRUE),
                          plot=1)


# Set up traces with labels and legend
labelTrace <- textTile(labels=c("Currently Married", "Not Married"),
                       x=c( 35,    53),
                       y=c( 0.8,  0.56),
                       col=col,
                       plot=1)

legendTrace <- textTile(labels=c("Logit estimates:", "95% confidence", "interval is shaded"),
                        x=c(80, 80, 80),
                        y=c(0.2, 0.15, 0.10),
                        cex=0.9,
                        plot=1)

# Plot traces using tile
tile(marriedTrace,
     notmarrTrace,
     labelTrace,
     legendTrace,
     limits=c(18,94,0,1),
     xaxis=list(at=c(20,30,40,50,60,70,80,90)),
     yaxis=list(label.loc=-0.5, major=FALSE),
     xaxistitle=list(labels="Age of Respondent"),
     yaxistitle=list(labels="Probability of Voting"),
     width=list(null=5,yaxistitle=4,yaxis.labelspace=-0.5)
     #,output=list(file="marriedEV",width=5.5)
)
```
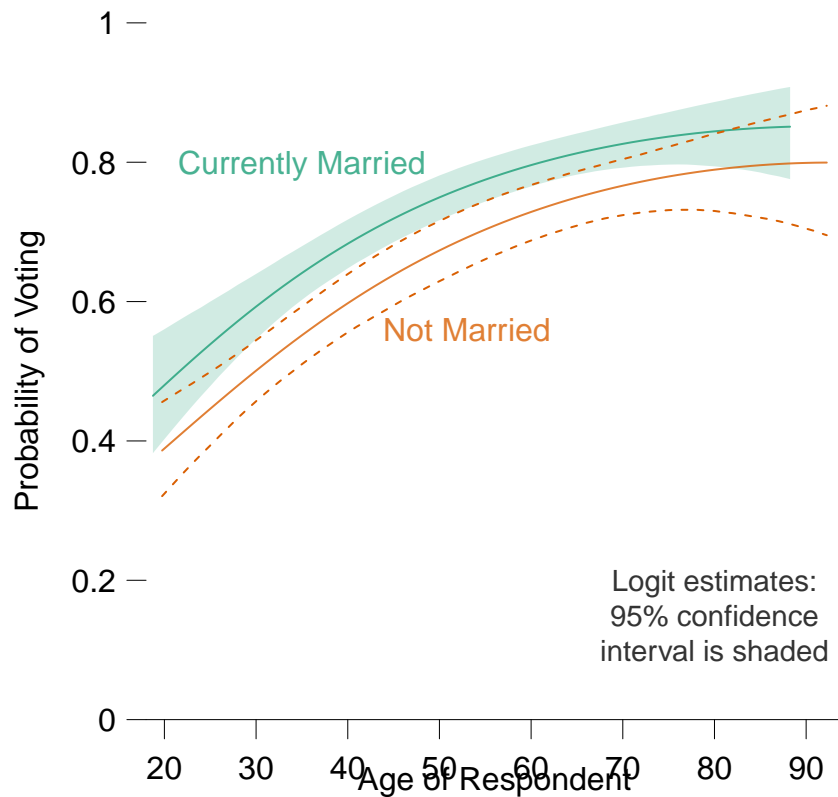
```
## [1] "Running whatif"
## [1] "Preprocessing data ..."
## [1] "Performing convex hull test ..."
## [1] "Calculating distances ...."
## [1] "Calculating the geometric variance..."
## [1] "Calculating cumulative frequencies ..."
## [1] "Finishing up ..."
## [1] "Whatif finished; returning to tile"
## [1] "Running whatif"
## [1] "Preprocessing data ..."
## [1] "Performing convex hull test ..."
## [1] "Calculating distances ...."
## [1] "Calculating the geometric variance..."
## [1] "Calculating cumulative frequencies ..."
## [1] "Finishing up ..."
## [1] "Whatif finished; returning to tile"
```

The chart shows "Probability of Voting" (y-axis) versus "Age of Respondent" (x-axis), with curves labeled "Currently Married" and "Not Married", and a note "Logit estimates: 95% confidence interval is shaded".

```
# Plot First Difference

# Set up lineplot trace of first difference

marriedFDTrace <- lineplot(x=xhyp,
                           y=marriedFD$pe,
                           lower=marriedFD$lower,
                           upper=marriedFD$upper,
                           col=col[1],
                           extrapolate=list(data=mdata2[,2:ncol(mdata2)],
                                            cfact=marriedScen$x[,2:ncol(marriedScen$x)],
                                            omit.extrapolated=TRUE),
                           plot=1)


# Set up baseline: for first difference, this is 0
baseline <- linesTile(x=c(18,94),
                      y=c(0,0),
                      plot=1)

# Set up traces with labels and legend
labelFDTrace <- textTile(labels=c("Married compared \n to Not Married"),
                         x=c( 40),
                         y=c( 0.20),
                         col=col[1],
                         plot=1)

legendFDTrace <- textTile(labels=c("Logit estimates:", "95% confidence", "interval is shaded"),
```

```
                        x=c(80, 80, 80),
                        y=c(-0.02, -0.05, -0.08),
                        cex=0.9,
                        plot=1)

# Plot traces using tile
tile(marriedFDTrace,
     labelFDTrace,
     legendFDTrace,
     baseline,
     limits=c(18,94,-0.1,0.5),
     xaxis=list(at=c(20,30,40,50,60,70,80,90)),
     yaxis=list(label.loc=-0.5, major=FALSE),
     xaxistitle=list(labels="Age of Respondent"),
     yaxistitle=list(labels="Difference in Probability of Voting"),
     width=list(null=5,yaxistitle=4,yaxis.labelspace=-0.5)
     #,output=list(file="marriedFD",width=5.5)
)
```
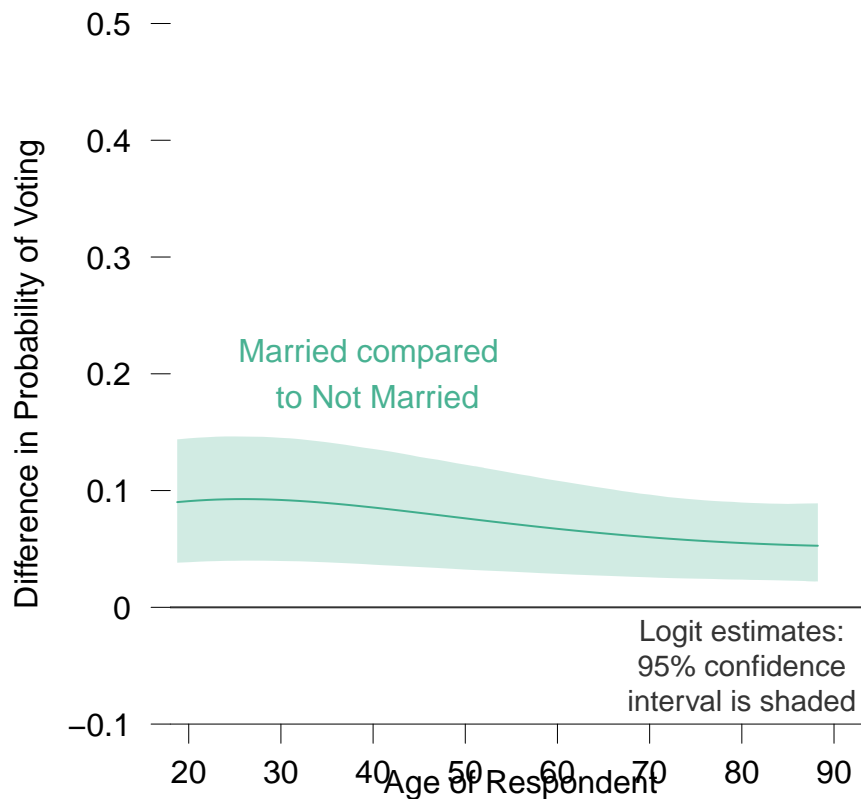
```
## [1] "Running whatif"
## [1] "Preprocessing data ..."
## [1] "Performing convex hull test ..."
## [1] "Calculating distances ...."
## [1] "Calculating the geometric variance..."
## [1] "Calculating cumulative frequencies ..."
## [1] "Finishing up ..."
## [1] "Whatif finished; returning to tile"
```

```r
# Plot Relative Risk

# Set up lineplot trace of relative risk
marriedRRTrace <- lineplot(x=xhyp,
                           y=marriedRR$pe,
                           lower=marriedRR$lower,
                           upper=marriedRR$upper,
                           col=col[1],
                           extrapolate=list(data=mdata2[,2:ncol(mdata2)],
                                            cfact=marriedScen$x[,2:ncol(marriedScen$x)],
                                            omit.extrapolated=TRUE),
                           plot=1)



# Set up baseline: for relative risk, this is 1

baseline <- linesTile(x=c(18,94),
                      y=c(1,1),
                      plot=1)

# Set up traces with labels and legend
labelRRTrace <- textTile(labels=c("Married compared \n to Not Married"),
                         x=c( 55),
                         y=c( 1.25),
                         col=col[1],
                         plot=1)

legendRRTrace <- textTile(labels=c("Logit estimates:", "95% confidence", "interval is shaded"),
                          x=c(80, 80, 80),
                          y=c(0.98, 0.95, 0.92),
                          cex=0.9,
                          plot=1)

# Plot traces using tile
tile(marriedRRTrace,
     labelRRTrace,
     legendRRTrace,
     baseline,
     limits=c(18,94,0.9,1.5),
     xaxis=list(at=c(20,30,40,50,60,70,80,90)),
     yaxis=list(label.loc=-0.5, major=FALSE),
     xaxistitle=list(labels="Age of Respondent"),
     yaxistitle=list(labels="Relative Risk of Voting"),
     width=list(null=5,yaxistitle=4,yaxis.labelspace=-0.5)
     #,output=list(file="marriedRR",width=5.5)
)

## [1] "Running whatif"
## [1] "Preprocessing data ..."
## [1] "Performing convex hull test ..."
## [1] "Calculating distances ...."
## [1] "Calculating the geometric variance..."
## [1] "Calculating cumulative frequencies ..."
## [1] "Finishing up ..."
```