

# MLE

Brad Jones<sup>1</sup>

<sup>1</sup>Department of Political Science  
University of California, Davis

April 7, 2009

Today: MLE

# Preliminaries

- ▶ The “concept” of maximum likelihood.
- ▶ The theory of MLE
- ▶ MLE in practice.

# Coin Flips and MLE

- ▶ Theory of ML has close connection to probability
- ▶ Suppose we want to know  $\theta$
- ▶ Let  $\theta = \Pr(H)$ .
- ▶ *a priori*, we know  $p = .5$  for fair coins.
- ▶ We could specify a “model”:  $\Pr(H | p = .5)$ .
- ▶ And formalize it in terms of the binomial distribution PDF:

$$\frac{n!}{y!(n-y)!} p^y (1-p)^{n-y} \quad (1)$$

# Binomial Distribution

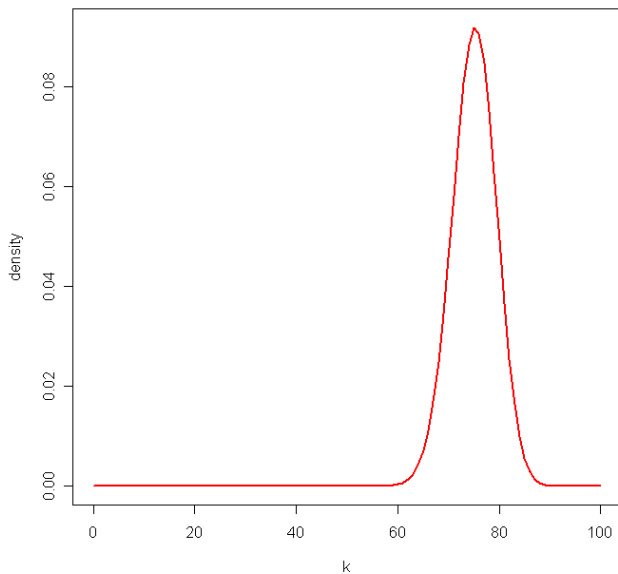
- ▶ The binomial PDF:

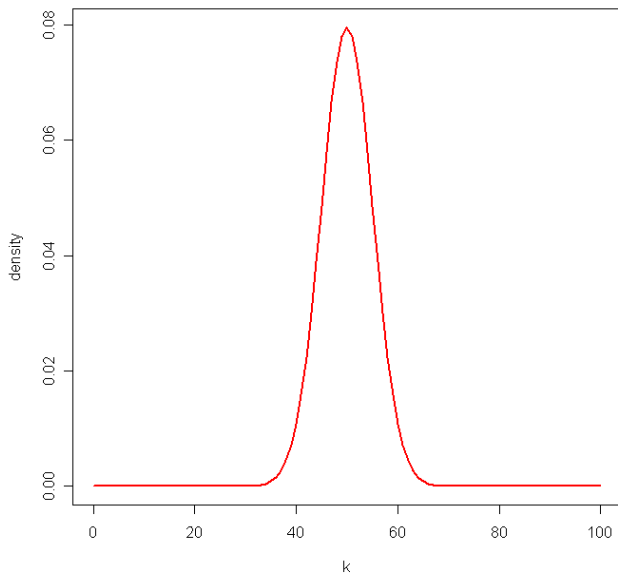
$$\frac{n!}{y!(n-y)!} p^y (1-p)^{n-y}$$

- ▶ The distribution has two parameters:  $n$  is the number of “Bernoulli trials”;  $p$  is the probability of a success.
- ▶ Suitable for Bernoulli processes
- ▶ ...or binary sequences (like a coin flip, for example)
- ▶ Let's take a look at the p.f.

```
n <- 100 k <- seq(0, n, by = 1) plot (k, dbinom(k, n, .75,
log=FALSE), type='l', ylab="density", main = "Binomial Density
(Prob. Scale)") lines(k, (dbinom(k, n, .75)), col='red', lwd=2)
```

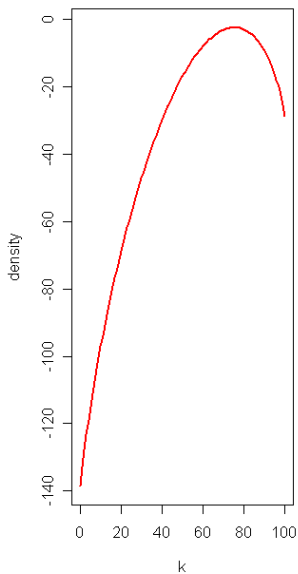
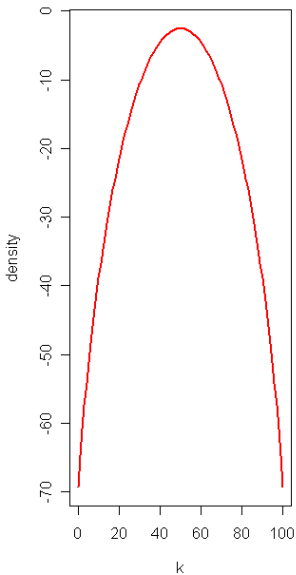
```
n <- 100 k <- seq(0, n, by = 1) plot (k, dbinom(k, n, .50,
log=FALSE), type='l', ylab="density", main = "Binomial Density
(Prob. Scale)") lines(k, (dbinom(k, n, .50)), col='red', lwd=2)
```

Binomial Density (Prob. Scale,  $p=.75$ )

Binomial Density (Prob. Scale,  $p=.50$ )

# Binomial Distribution

- ▶ The “peaks” represent maximum probability.
- ▶ The “location” of the peak depends on the parameter  $p$ .
- ▶ In passing (for now), note that the location is unaffected by rescaling the  $y$ -axis.
- ▶ Suppose we log it?

Binomial Density (Log Scale,  $p=.75$ )Binomial Density (Log Scale,  $p=.50$ )

# Binomial Distribution

- ▶ The “peaks” still lie above the same location on the  $x$ -axis.
- ▶ However,  $y$  is no longer a true probability.
- ▶ Coin flipping: suppose  $n$  flips=9 and  $n$  heads (call it  $y$ ) is 4?
- ▶ The joint probability is given by:

$$p \times p \times p \times p \times (1 - p) \times (1 - p) \times (1 - p) \times (1 - p) \times (1 - p)$$

- ▶ Or more succinctly:

$$p^4(1 - p)^5$$

- ▶ Note, this is a really small number (why?)

# Binomial Distribution

- ▶ To solve the probability, utilized the binomial distribution:

$$\frac{9!}{4!5!} .5^4 .5^5 = .246$$

- ▶ We've answered the following:

$$P(H | p)$$

- ▶ That is, the probability of 4 heads from 9 flips is .25, *conditional on the parameter  $p = .5$ .*
- ▶ This, of course is a bit trivial.
- ▶ But let's pretend we know nothing.

# Binomial Distribution and Likelihood

- ▶ Suppose we didn't know  $p$ ? Above, this is an assumption (and a pretty good one).
- ▶ Instead, we might want to posit:

$$\mathcal{L} = (p \mid H)$$

- ▶ Which in words asks, what is the value of  $p$  that *maximizes the likelihood* of a given sequence of coin flips?
- ▶ Here, we do not know the parameter but we know what the data look like.
- ▶ More generic:

$$\mathcal{L} = (\theta \mid \mathbf{X})$$

- ▶  $\theta$  is some parameter (could be a regression coefficient) and  $\mathbf{X}$  are data.
- ▶ What is the value of  $\theta$  that maximizes the likelihood of the observed data.

# Binomial Distribution and Likelihood

- ▶ What is the appeal of this statement?
- ▶ It explicitly acknowledges the fact that our data are a fixed sample.
- ▶ Given this, the parameters are not fixed (and are certainly not known in advance).
- ▶ The question is: what value of  $\theta$  is “most likely” given the observed data?
- ▶ The best that we can do is “maximize” the likelihood of  $\theta$  given the observed data.
- ▶ This is among the fundamental principles of the theory of maximum likelihood.
- ▶ The historically relevant figure here is R.A. Fisher.
- ▶ Among the most important statisticians in history.

# Binomial Distribution and Likelihood

- ▶ Sometimes, analytical solutions for  $\theta$  do not exist (or are very hard to solve).
- ▶  $\therefore$  we may need to “search” for the best answer.
- ▶ This entails *iterating over possible* values of  $\theta$ .
- ▶ Since some values of  $\theta$  may be more likely than other values, we apply some mathematical criteria to evaluate the possible solutions.
- ▶ We'll discuss this in a bit, but let's do a pedagogical “walk-thru.”

## Let's flip a coin

- ▶ `sequence<-rbinom(100, 1, .46);summary(sequence)`
- ▶ What did I just do?
- ▶ This returns a binary sequence of 100 trials with a mean of .42.
- ▶ The mean of a binary variable is equivalent to the proportion of “1s” (or “successes” [or heads]).
- ▶ What we want to do is “solve” this:  $\mathcal{L} = (p | H)$ .
- ▶ What is the value of  $p$  that maximizes the likelihood of this coin flip chain?
- ▶ Pretending we know nothing about binomials, we could just pick a value of  $p$  and evaluate.

# Wandering around.

- ▶ Let  $y$  be:  $y < -\text{mean}(\text{sequence}) * 100$
- ▶ We can derive the probability of  $n$  Bernoulli trials through the binomial distribution.
- ▶ i.e.:  $\frac{n!}{y!(n-y)!} p^y (1-p)^{n-y}$
- ▶ OK ... so where to start?
- ▶ Remember, any permissible value of  $p$  is possible (though some are not very likely).

Let's start with  $p=.10$ :

```
p1<-choose(100, 42)*(.10^42*(1-.10)^58)
```

This returns: 6.269305e-17 (basically 0).

-----  
Can we do better? Let's try  $p=.50$ :

```
p2<-choose(100, 42)*(.50^42*(1-.50)^58)
```

This returns: 0.02229227.

-----  
How about  $p=.40$ ?

```
p2<-choose(100, 42)*(.40^42*(1-.40)^58)
```

This returns: 0.0742072

-----

We could keep going: .41, .42, .43, ... .95, .96, .97

Or tell R to do it for us:

Probability sequence (coarse mesh)

```
p<-seq(0, 1, by=.1)
```

Grid Search, coarse mesh

```
s1<-choose(n, y)*(p^y*(1-p)^(n-y))
```

```
> cbind(p, s1)
```

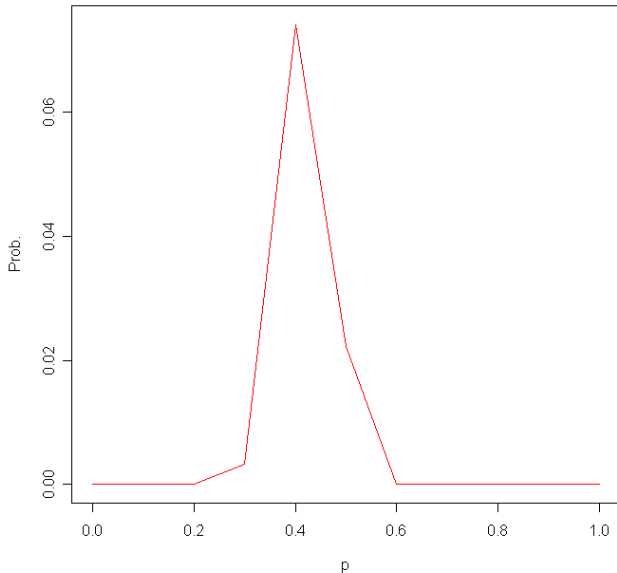
	p	s1
[1,]	0.0	0.000000e+00
[2,]	0.1	6.269305e-17
[3,]	0.2	2.976000e-07
[4,]	0.3	3.205774e-03
[5,]	0.4	7.420719e-02 <---"Biggest"
[6,]	0.5	2.229227e-02
[7,]	0.6	1.129759e-04
[8,]	0.7	4.152449e-09
[9,]	0.8	6.929040e-17
[10,]	0.9	3.383290e-32
[11,]	1.0	0.000000e+00

Plot this wrt p:

```
plot(p,s1, type="l", col='red', xlab="p", ylab="Prob.")  
title("Binomial Probabilities: 42 Heads, 58 Tails")
```

Returns:

## Binomial Probabilities: 42 Heads, 58 Tails



# Searches

- ▶ What do we see?
- ▶ When  $p = .40$ , the *likelihood* of the data is highest (about .074).
- ▶ We “iterated” to this solution by substituting in 11 possible values for  $p$ .
- ▶ Any issues? Points of concern?
- ▶ What we’ve just done is a very simple (and crude) “grid search.”
- ▶ The search was for the maximum.
- ▶ Given the “mesh” of the grid, we found the maximum at .40.
- ▶ Thus, our *MLE* of  $p$  is .40, *given this grid*.
- ▶ Can we do better?

```

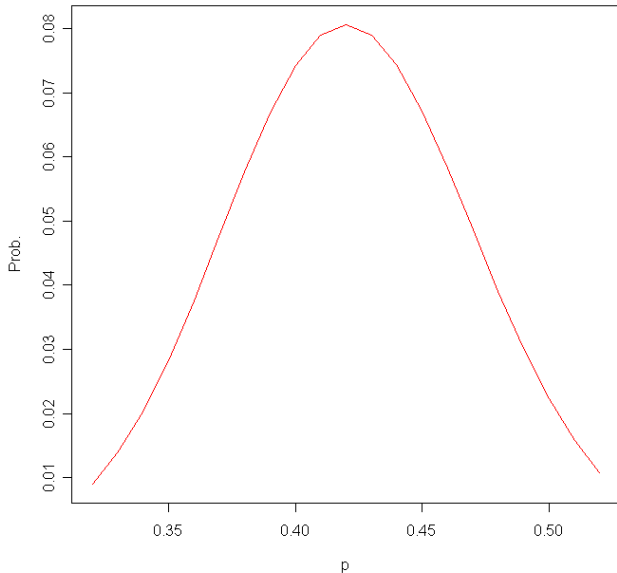
Probability sequence (fine mesh):
p<-seq(.32, .52, by=.01) <--Note beginning and ending points on grid? Why do this?
Grid Search, fine mesh
s2<-choose(n, y)*(p^y*(1-p)^(n-y))

> cbind(p, s2)
      p      s2
[1,] 0.35 0.028247915
[2,] 0.36 0.037522985
[3,] 0.37 0.047574927
[4,] 0.38 0.057647822
[5,] 0.39 0.066834050
[6,] 0.40 0.074207194
[7,] 0.41 0.078975523
[8,] 0.42 0.080620906 <---"Biggest"
[9,] 0.43 0.078989727
[10,] 0.44 0.074314204
[11,] 0.45 0.067160732
[12,] 0.46 0.058320448
[13,] 0.47 0.048670334
[14,] 0.48 0.039037298
[15,] 0.49 0.030092580
[16,] 0.50 0.022292270
[17,] 0.51 0.015866235
[18,] 0.52 0.010846411
[19,] 0.53 0.007118953
[20,] 0.54 0.004483798
[21,] 0.55 0.002708399

Plot it:

plot(p,s2, type="l", col='red', xlab="p", ylab="Prob.")
title("Binomial Probabilities: 42 Heads, 58 Tails")

```

**Binomial Probabilities: 42 Heads, 58 Tails**

# Searches

- ▶ What do we see?
- ▶ A finer grid mesh returns a different maximum.
- ▶ This one is clearly preferable to the course mesh search from before.
- ▶ In either case, however, we had to iterate (though in reality, no need to!)
- ▶ The probability of a binary sequence is maximal for  $p = \text{proportion of 1s}$
- ▶ But if we didn't know that, we could still get to the right answer.
- ▶ In fact, what we've just done is maximum likelihood estimation.
- ▶ The value of  $p$  that maximizes the likelihood of the observed data is .42.
- ▶ Its likelihood is .081.

## MLE

- ▶ Back to the theory.
- ▶ “Likelihood” is *not synonymous* with probability; it’s *proportional* to  $p$ :
- ▶ Proportionality:

$$\begin{aligned}\mathcal{L}(\theta | \mathbf{Y}) &= kp(\mathbf{Y} | \theta) \\ &\propto \prod_{i=1}^N p(\mathbf{Y} | \theta)\end{aligned}\quad (2)$$

- ▶ That is, the likelihood is proportional to the joint probability of the data.
- ▶ More generic:

$$\mathcal{L}(\theta | y) = \prod_{i=1}^N f(y_i | \theta)$$

- ▶ What is  $f(\cdot)$ ?

# MLE

- ▶ It's a density function!
- ▶ What about the product operator?
- ▶ Recall property of independence: if observations are independent, the joint probability is the product of the individual probabilities.
- ▶ iid: *independently and identically distributed*
- ▶ We just talked about independence (and we'll assume it).
- ▶ Identically distributed has something to do with the PDF.
- ▶ Coin flips: events are independent with a DGP governed by  
...
- ▶ The binomial distribution.

Lets generate a binary sequence using the binomial distribution:

```
lik<-function(p,y,n) choose(n,y)*(p^y)*(1-p)^(n-y)
```

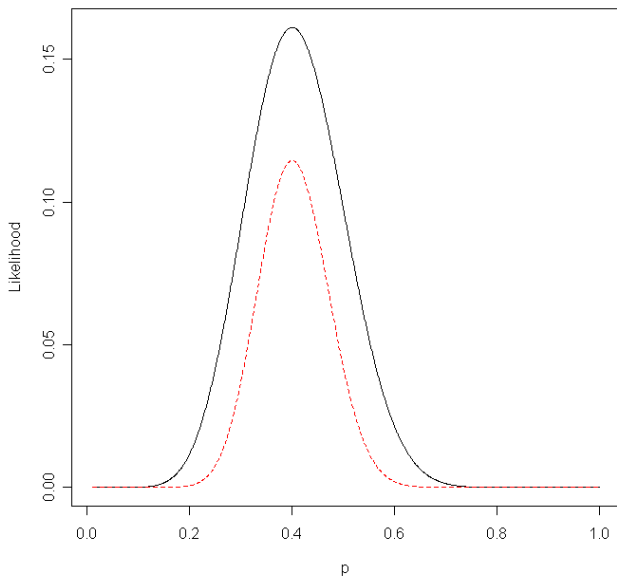
```
p<-seq(.01:.99, by=.001)
```

```
matplot(p, cbind(lik(p, 10, 25), lik(p,20,50)),
```

```
type="l", xlab="p", ylab="Likelihood")
```

```
title("Likelihood Functions for Two Binomial Sequences")
```

## Likelihood Functions for Two Binomial Sequences



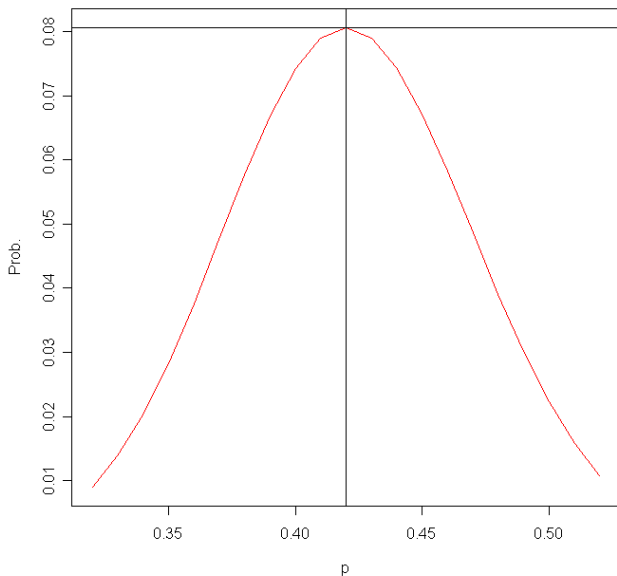
# MLE

- ▶ The MLE is found at the mode.
- ▶ Technically, it's the point at which the rate of change in the likelihood wrt a change in  $\theta$  is 0.
- ▶ That is:

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta} = 0 \quad (3)$$

- ▶ A tangent wrt the plot would have a slope of zero.
- ▶ Consider our coin flip example (in pictures)

## Likelihood: Coin Flips



# MLE

- ▶ In principle, the likelihood is obtained by specifying a probability function and deriving the joint likelihood:  
$$\prod_{i=1}^N \mathcal{L} = \ell_1 \times \ell_2 \times \dots \times \ell_N.$$
- ▶ Implicit here is some predefined  $f(\cdot)$  (like a binomial, for example).
- ▶ In reality, we always work with log of the likelihood functions.
- ▶ Why? The joint likelihood is way too small a number to handle!
- ▶  $.5^{1000} = 9.332636e - 302$
- ▶ On the other hand,  $\log(.5^{1000}) = -693.1472$ .
- ▶ Easy to work with and no trickery: the log transformation is a monotonic transformation of  $\mathcal{L}$ .
- ▶ The only effect it has is to change the scale.
- ▶ But since the scale has no intrinsic meaning (except in a relative sense), this is not a problem.

## MLE

- ▶ Note that since  $\mathcal{L}$  is a joint probability (and a very, very small number),  $\log \mathcal{L}$  must be negative.
- ▶ On your computer output, you will be given  $\log \mathcal{L}$  (and by itself, it's a meaningless number).
- ▶ MLE revealed.
- ▶ Step 1: find a suitable probability function.
- ▶ Step 2: Evaluate the *score function*:

$$\mathbf{U}(\theta) = \frac{\partial \log \mathcal{L}(\theta)}{\partial \theta} \quad (4)$$

- ▶ If multiple parameters are estimated, evaluate the elements of  $\mathbf{U}(\theta)$ :

$$u_k = \frac{\partial \log \mathcal{L}(\theta)}{\partial \theta_k} \quad (5)$$

- ▶ What is the score function?
- ▶ It is the derivative of the log-likelihood wrt the parameters.

## MLE

- ▶ “Evaluate” really means setting those derivatives to 0 and solving for  $\theta$ .
- ▶ Why 0? That is the point at which a tangent wrt log-likelihood function is flat.
- ▶ Flat is good. It means you hit the maximum (of course it may not be THE maximum).
- ▶ Coin flips redux.
- ▶ The first derivative for the binomial setting is:

$$\begin{aligned}\mathbf{U}(\theta) &= \frac{\partial \log \mathcal{L}(\theta)}{\partial \theta} \\ &= \frac{y}{p} - \frac{n-y}{1-p}\end{aligned}\tag{6}$$

- ▶ Find the value of  $p$  that gives a score of 0.
- ▶ This is your maximum.

# Coin flips MLE

- ▶ The log-likelihood function:

$$\log \mathcal{L} = \log \binom{n}{y} + y \log p + (n - y) \log(1 - p) \quad (7)$$

- ▶ The score function:

$$u(p) = \frac{\partial \log \mathcal{L}(p | y)}{\partial p} \quad (8)$$

- ▶ Easy to do in R

```
loglik<- lchoose(n,y) + y*log(p) + (n-y)*log(1-p)
```

```
score<-y/p - (n-y)/(1-p)
```

Let's compare:

```
cbind(p, score, s2)
```

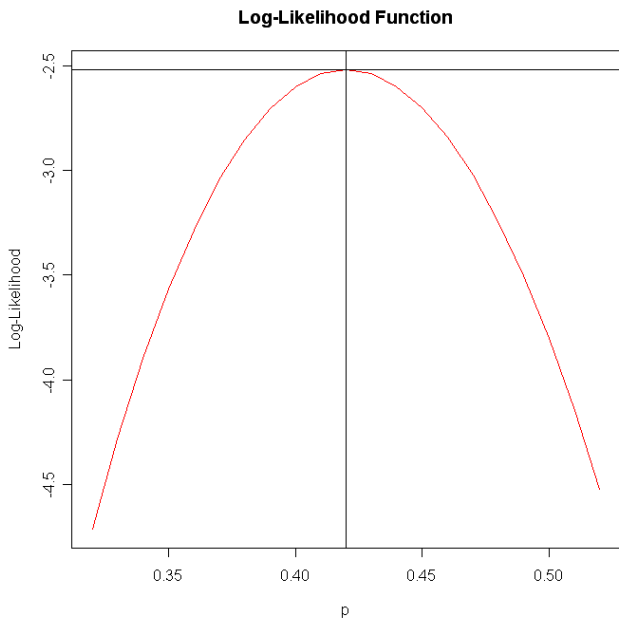
	p	score	s2
[1,]	0.32	4.595588e+01	0.008973666
[2,]	0.33	4.070556e+01	0.013838145
[3,]	0.34	3.565062e+01	0.020268268
[4,]	0.35	3.076923e+01	0.028247915
[5,]	0.36	2.604167e+01	0.037522985
[6,]	0.37	2.145002e+01	0.047574927
[7,]	0.38	1.697793e+01	0.057647822
[8,]	0.39	1.261034e+01	0.066834050
[9,]	0.40	8.333333e+00	0.074207194
[10,]	0.41	4.133940e+00	0.078975523
[11,]	0.42	-1.421085e-14	0.080620906<---"Maximum" (score is essentially 0!)
[12,]	0.43	-4.079967e+00	0.078989727
[13,]	0.44	-8.116883e+00	0.074314204
[14,]	0.45	-1.212121e+01	0.067160732
[15,]	0.46	-1.610306e+01	0.058320448
[16,]	0.47	-2.007226e+01	0.048670334
[17,]	0.48	-2.403846e+01	0.039037298
[18,]	0.49	-2.801120e+01	0.030092580
[19,]	0.50	-3.200000e+01	0.022292270
[20,]	0.51	-3.601441e+01	0.015866235
[21,]	0.52	-4.006410e+01	0.010846411

Plot it:

```
plot(p,loglik, type="l", col='red', xlab="p", ylab="Log-Likelihood")
```

```
abline(h=mlc1)
```

```
abline(v=.42)
```



# Are we done yet?

- ▶ Did the first-order condition really identify a maximum?
- ▶ Why might it be wrong? (coarse mesh in a grid search?)
- ▶ To establish if first-order condition truly yields a maximum, we need to compute the second derivatives.
- ▶ We'll call this the Hessian (named for Ludwig Otto Hesse).
- ▶ It looks like this:

$$\mathbf{H}\theta = \frac{\partial^2 \log \mathcal{L}(\theta)}{\partial \theta \partial \theta'} \quad (9)$$

- ▶ It's a bit ugly looking but it's really just a matrix of second derivatives.
- ▶ This thing needs to be **negative definite**. (Why?)

# Inverse of the Hessian

- ▶ The inverse of the expected value of the Hessian is called the *Fisher Information Matrix*.
- ▶ It looks like:

$$\mathbf{I}(\theta) = -E(\mathbf{H}(\theta)) \quad (10)$$

- ▶ The information in this matrix is really, really important!
- ▶ The inverse, if it exists, gives us the variance of  $\theta$ :

$$\begin{aligned} \mathbf{I}(\theta)^{-1} &= \text{var}(\theta) \\ &= (-E[\mathbf{H}(\theta)])^{-1} \\ &= \left( -E \left[ \frac{\partial^2 \log \mathcal{L}(\theta)}{\partial \theta \partial \theta'} \right] \right)^{-1} \end{aligned} \quad (11)$$

# Inverse of the Hessian

- ▶ If we can't invert the Hessian, we're stuck.
- ▶ The inverse of the information matrix gives us the variances down the main diagonal.
- ▶ Take the square root, you've got your standard errors.

```
llk.logit<-function(param,y,x) {
  cons <- rep(1, length(x[,1]))
  x<-cbind(cons, x)
  b<-param[1 : ncol(x) ]
  xb<-x%*%b
  sum( y*log(1+exp(-xb)) + (1-y)*log(1+exp(xb)))
}

ols.result <- lm(y~x); ols.result

stval <-ols.result$coeff

logit.result<-optim(stval, llk.logit, method="BFGS", hessian=TRUE, y=y, x=x)

parm_est<-logit.result$par; parm_est
var_cov<-solve(logit.result$hessian); var_cov
std_err<-sqrt(diag(var_cov)); std_err
log_like<- -logit.result$value; log_like
```

```
ols.result <- lm(y~x); ols.result

Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)          x
    0.5067         0.7750

>
> stval <-ols.result$coeff
>
> logit.result<-optim(stval, llk.logit, method="BFGS", hessian=TRUE, y=y, x=x)
>
> parm_est<-logit.result$par; parm_est
(Intercept)          x
 0.03303452  5.09369576
> var_cov<-solve(logit.result$hessian); var_cov
              (Intercept)          x
(Intercept)  0.043813933 -0.001781283
x            -0.001781283  0.569164199
> std_err<-sqrt(diag(var_cov)); std_err
(Intercept)          x
 0.2093178   0.7544297
> log_like<- -logit.result$value; log_like
[1] -71.13285
>
```

```
> logit1<-glm(d~x1, family=binomial(link="logit") ); summary(logit1)
```

Call:

```
glm(formula = d ~ x1, family = binomial(link = "logit"))
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-2.0330	-0.6081	0.1545	0.6256	2.0836

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	0.03303	0.20932	0.158	0.875
x1	5.09370	0.75442	6.752	1.46e-11 ***

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 234.14 on 168 degrees of freedom  
 Residual deviance: 142.27 on 167 degrees of freedom  
 AIC: 146.27

Number of Fisher Scoring iterations: 5

## MLE

- ▶ Last time, we covered basic
- ▶ MLE revealed.
- ▶ Step 1: find a suitable probability function.
- ▶ Step 2: Evaluate the *score function*:

$$\mathbf{U}(\theta) = \frac{\partial \log \mathcal{L}(\theta)}{\partial \theta} \quad (12)$$

- ▶ If multiple parameters are estimated, evaluate the elements of  $\mathbf{U}(\theta)$ :

$$u_k = \frac{\partial \log \mathcal{L}(\theta)}{\partial \theta_k} \quad (13)$$

- ▶ What is the score function?
- ▶ It is the derivative of the log-likelihood wrt the parameters.

# MLE

- ▶ “Evaluate” really means setting those derivatives to 0 and solving for  $\theta$ .
- ▶ Why 0? That is the point at which a tangent wrt log-likelihood function is flat.
- ▶ Often, however, an analytical solution may not be possible.
- ▶ This is particularly true for models with nonlinearities.
- ▶ As such, we need to iteratively find the maximum.
- ▶ “Iteratively” could mean a grid search . . .
- ▶ Or searches using different kinds of *optimizers*

# Optimizers

- ▶ Algorithms to find the maximum (or minimum, depending on how you parameterize these things).
- ▶ Technically, these are methods to solve unconstrained nonlinear optimization problems.
- ▶ aka “hill climbers.”
- ▶ There are a variety of them.
- ▶ They do not all behave the same way.
- ▶ The choice is usually not yours to make . . .
- ▶ although you can control the optimization choice.

# Optimizers

- ▶ The basic issue is: how do you know you've reached a maximum?
- ▶ Commonly used optimizers: Newton-Raphson or quasi Newton methods (BFGS=Broyden, Fletcher, Goldfarb, and Shanno).
- ▶ Both make use of Hessian (or approximate Hessian) and score vector to update estimates over  $\theta$ .
- ▶ The "hill climb" stops when "you reach the top."
- ▶ The top is usually determined to be reached when the change in parameter values from one iteration to the next is effectively nil.

## Hessian, importance of.

- ▶ The Hessian matrix:

$$\mathbf{H}(\boldsymbol{\theta}) = \frac{\partial^2 \log \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}'} \quad (14)$$

- ▶ Useful to us in order to compute estimates of uncertainty (i.e. variances and covariances).
- ▶ The negative of the expected value of the Hessian is called the *Fisher Information Matrix*.
- ▶ It looks like:

$$\mathbf{I}(\boldsymbol{\theta}) = -E(\mathbf{H}(\boldsymbol{\theta})) \quad (15)$$

- ▶ The inverse, if it exists, gives us the variance of  $\boldsymbol{\theta}$ :

$$\begin{aligned} \mathbf{I}(\boldsymbol{\theta})^{-1} &= \text{var}(\boldsymbol{\theta}) \\ &= (-E[\mathbf{H}(\boldsymbol{\theta})])^{-1} \\ &= \left( -E \left[ \frac{\partial^2 \log \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}'} \right] \right)^{-1} \end{aligned} \quad (16)$$

# Can we do this on our own?

- ▶ Let's play around with an optimizer.
- ▶ Consider the Newton-Raphson optimizer.
- ▶ Let's start with the Hessian (assume two parameters,  $\beta_1$  and  $\beta_2$ ).
- ▶ Hessian:

$$\mathbf{H}(\boldsymbol{\theta}) = \frac{\partial^2 \log \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}'} = \begin{pmatrix} \frac{\partial^2 \log \mathcal{L}(\boldsymbol{\theta})}{\partial \beta_1 \partial \beta_1} & \frac{\partial^2 \log \mathcal{L}(\boldsymbol{\theta})}{\partial \beta_1 \partial \beta_2} \\ \frac{\partial^2 \log \mathcal{L}(\boldsymbol{\theta})}{\partial \beta_2 \partial \beta_1} & \frac{\partial^2 \log \mathcal{L}(\boldsymbol{\theta})}{\partial \beta_2 \partial \beta_2} \end{pmatrix} \quad (17)$$

- ▶ Newton-Raphson algorithm:

$$\boldsymbol{\theta}_1 = \boldsymbol{\theta}_0 - \left( \frac{\partial^2 \log \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}'} \right)^{-1} \frac{\partial \log \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \quad (18)$$

# Newton-Raphson

- ▶ We've seen these parts before:

$$\theta_1 = \theta_0 - \mathbf{H}(\theta)^{-1}\mathbf{U}(\theta) \quad (19)$$

- ▶ The first part is the inverse of the Hessian (i.e. the var-cov matrix) and the second part is the *score vector*.
- ▶ Intuitively, it makes sense to consider both functions.
- ▶ Why? The score function gives you the gradient (or the direction of the change in parameter values) while the Hessian gives you the rate of change of the function (is it increasing “quickly” or “slowly”?)
- ▶ At a maximum, the first derivative should be 0. If the second derivative is 0, the function is “concave down.”
- ▶ This means, the tangent lines wrt the function are downwardly slope.

# Newton-Raphson

- ▶ You achieve “convergence” when the incremental change in  $\theta$  is effectively 0.
- ▶ Alternatives to Newton-Raphson?
- ▶ Method of Steepest ascent (only uses score function; often not a good choice)
- ▶ Scoring methods (replace inverted Hessian with inverse of information matrix).
- ▶ BHHH, BFGS, etc.
- ▶ They do not all behave exactly the same!
- ▶ However, with “well behaved” data and pdfs, they will all tend to the same answer.

# Logit and Newton-Raphson

- ▶ Let's work through an example to illustrate all the moving parts.
- ▶ We'll estimate a logit model.
- ▶ The distribution function for the logistic distribution is given by

$$\begin{aligned}\Pr(Y = 1) &= \frac{\exp(Z)}{1 + \exp(Z)} \\ &= \frac{1}{1 + \exp(-Z)} \\ &= \Lambda_i\end{aligned}\tag{20}$$

- ▶  $Z = \sum_{k=0}^K \beta_k x_{ik}$ .

# Logit

- ▶ The logit model is obtained by taking the logistic transformation, which yields:

$$\begin{aligned}\log\left(\frac{p_i}{1-p_i}\right) &= Z \\ &= \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_k x_{ik} \\ &= \sum_{k=0}^K \beta_k x_{ik}\end{aligned}\tag{21}$$

- ▶ In the language of GLMs,  $Z$  is a *link function*.
- ▶ The model is linear in the *log odds ratios*.
- ▶ The distribution is nice because:
  - a) It is unbounded in the log-odds.
  - b)  $\Pr(Y = 1)$  is bound in the interval  $(0,1)$ .
  - c) The probabilities are a nonlinear function of covariates.
- ▶ We have a distribution function. All we need are data.

# Logit and Newton-Raphson

- ▶ I use some simulated data and estimate a logit model with one covariate  $x_i$  and binary response variable  $y_i$ .
- ▶ R code: `logit1<-glm(d x1, family=binomial(link="logit") )`
- ▶ `glm` uses the Fisher scoring algorithm
- ▶ The logit function is globally concave and so Fisher method should be generally equivalent to other optimizers.
- ▶ “Known” results:  $\beta_0 = 0.033$  (0.209);  $\beta_1 = 5.094$  (0.754)
- ▶ log-likelihood:  $-71.13$

# Optimizing a Logit Model

- ▶ Let's start from scratch.
- ▶ We know the log-likelihood for a generic BRM is given by:

$$\log \mathcal{L}(\beta) = \sum_i \left\{ \log \binom{n_i}{y_i} + y_i \log F(\mathbf{x}_i' \beta) + (n_i - y_i) \log [1 - F(\mathbf{x}_i' \beta)] \right\} \quad (22)$$

- ▶ Look familiar? Think *binomial family*.
- ▶ Logit log-likelihood:

$$\log \mathcal{L}(\beta) = \sum_i \{y_i \log \Lambda_i + (1 - y_i) \log [1 - \Lambda_i]\} \quad (23)$$

- ▶ What did I just do?
- ▶ We've got a function, we've got data, but we don't have any estimates.
- ▶ Time to iterate using equation 8.

# Optimizing a Logit Model

- ▶ We've got to start somewhere.
- ▶ How about OLS? (i.e.  $y = \beta_0 + \beta_1 x_1$ ).
- ▶ This returns (.507, .775)'
- ▶ We'll start with these values.

The function:

```
llk.logit<-function(param,y,x) {  
  cons <- rep(1, length(x[,1]))  
  x<-cbind(cons, x)  
  b<-param[1 : ncol(x) ]  
  xb<-x%*%b  
  sum( y*log(1+exp(-xb)) + (1-y)*log(1+exp(xb)))  
}
```

# Iteration 1

Score Vector:

$$\mathbf{U}^1 = \begin{pmatrix} -18.263 \\ 18.095 \end{pmatrix} \quad (24)$$

Information Matrix

$$[\mathbf{I}^1]^{-1} = \begin{pmatrix} .026 & .003 \\ .003 & .146 \end{pmatrix} \quad (25)$$

Directional Vector ( $\Delta\hat{\beta}^1$ )

$$[\mathbf{I}^1]^{-1}\mathbf{U}^1 = \begin{pmatrix} -.413 \\ 2.589 \end{pmatrix} \quad (26)$$

“Updating”:

$$\begin{aligned} \hat{\beta}^1 + \Delta\hat{\beta}^1 &= \begin{pmatrix} .507 \\ .775 \end{pmatrix} + \begin{pmatrix} -.413 \\ 2.589 \end{pmatrix} \\ \hat{\beta}^2 &= \begin{pmatrix} .094 \\ 3.364 \end{pmatrix} \end{aligned} \quad (27)$$

Log-likelihood:  $-105.110$ ; Deviance:  $210.221$

## Iteration 2

Score Vector:

$$\mathbf{U}^2 = \begin{pmatrix} -1.603 \\ 4.262 \end{pmatrix} \quad (28)$$

Information Matrix

$$[\mathbf{I}^2]^{-1} = \begin{pmatrix} .034 & .001 \\ .001 & .299 \end{pmatrix} \quad (29)$$

Directional Vector ( $\Delta\hat{\beta}^0$ )

$$[\mathbf{I}^2]^{-1}\mathbf{U}^2 = \begin{pmatrix} -0.049 \\ 1.272 \end{pmatrix} \quad (30)$$

"Updating":

$$\begin{aligned} \hat{\beta}^2 + \Delta\hat{\beta}^2 &= \begin{pmatrix} -.413 \\ 2.589 \end{pmatrix} + \begin{pmatrix} -0.049 \\ 1.272 \end{pmatrix} \\ \hat{\beta}^3 &= \begin{pmatrix} .045 \\ 4.636 \end{pmatrix} \end{aligned} \quad (31)$$

Log-likelihood: -74.467; Deviance: 148.934

## Iteration 3

Score Vector:

$$\mathbf{U}^3 = \begin{pmatrix} -.249 \\ .875 \end{pmatrix} \quad (32)$$

Information Matrix

$$[\mathbf{I}^3]^{-1} = \begin{pmatrix} .041 & -.001 \\ -.001 & .481 \end{pmatrix} \quad (33)$$

Directional Vector ( $\Delta\hat{\beta}^0$ )

$$[\mathbf{I}^3]^{-1}\mathbf{U}^3 = \begin{pmatrix} -0.011 \\ .421 \end{pmatrix} \quad (34)$$

"Updating":

$$\begin{aligned} \hat{\beta}^3 + \Delta\hat{\beta}^3 &= \begin{pmatrix} .045 \\ 4.636 \end{pmatrix} + \begin{pmatrix} -0.011 \\ .421 \end{pmatrix} \\ \hat{\beta}^4 &= \begin{pmatrix} .034 \\ 5.057 \end{pmatrix} \end{aligned} \quad (35)$$

Log-likelihood: -71.329; Deviance: 142.658

## Iteration 4

Score Vector:

$$\mathbf{U}^4 = \begin{pmatrix} -.012 \\ .065 \end{pmatrix} \quad (36)$$

Information Matrix

$$[\mathbf{I}^4]^{-1} = \begin{pmatrix} .044 & -.002 \\ -.002 & .562 \end{pmatrix} \quad (37)$$

Directional Vector ( $\Delta\hat{\beta}^0$ )

$$[\mathbf{I}^4]^{-1}\mathbf{U}^4 = \begin{pmatrix} -0.001 \\ .036 \end{pmatrix} \quad (38)$$

"Updating":

$$\begin{aligned} \hat{\beta}^4 + \Delta\hat{\beta}^4 &= \begin{pmatrix} .034 \\ 5.057 \end{pmatrix} + \begin{pmatrix} -0.001 \\ .036 \end{pmatrix} \\ \hat{\beta}^5 &= \begin{pmatrix} .033 \\ 5.093 \end{pmatrix} \end{aligned} \quad (39)$$

Log-likelihood: -71.134; Deviance: 142.268

## Iteration 5

Score Vector:

$$\mathbf{U}^5 = \begin{pmatrix} -.000 \\ .000 \end{pmatrix} \quad (40)$$

Information Matrix

$$[\mathbf{I}^5]^{-1} = \begin{pmatrix} .044 & -.002 \\ -.002 & .569 \end{pmatrix} \quad (41)$$

Directional Vector ( $\Delta\hat{\beta}^5$ )

$$[\mathbf{I}^5]^{-1}\mathbf{U}^5 = \begin{pmatrix} -2.78e-06 \\ 2.44e-04 \end{pmatrix} \quad (42)$$

"Updating":

$$\begin{aligned} \hat{\beta}^5 + \Delta\hat{\beta}^5 &= \begin{pmatrix} .033 \\ 5.093 \end{pmatrix} + \begin{pmatrix} -0.000 \\ .000 \end{pmatrix} \\ \hat{\beta} &= \begin{pmatrix} .033 \\ 5.094 \end{pmatrix} \end{aligned} \quad (43)$$

Log-likelihood: -71.133; Deviance: 142.266

# Optimizing a Logit Model

- ▶ At iteration 5, we stop.
- ▶ Why? The incremental change in  $\beta$  is effectively 0.
- ▶ If we kept going, we'd get nowhere.
- ▶ Our MLE estimates are thus:  
 $\beta_0 = 0.033$  (0.209);  $\beta_1 = 5.094$  (0.754)  
log-likelihood:  $-71.133$ ; Deviance: 142.266
- ▶ Return to canned routine:

```
glm(formula = d ~ x1, family = binomial, trace = TRUE)
```

```
Deviance Residuals:
```

	Min	1Q	Median	3Q	Max
	-2.0330	-0.6081	0.1545	0.6256	2.0836

```
Coefficients:
```

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	0.03303	0.20932	0.158	0.875
x1	5.09370	0.75442	6.752	1.46e-11 ***

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)
```

```
Null deviance: 234.14 on 168 degrees of freedom
```

```
Residual deviance: 142.27 on 167 degrees of freedom
```

```
AIC: 146.27
```

```
Number of Fisher Scoring iterations: 5
```

# Optimizing a Logit Model

- ▶ Note again that GLM uses Fisher scoring.
- ▶ I used quasi-Newton methods.
- ▶ Both return same results.
- ▶ Again, this need not always be the case.
- ▶ Good data gone bad.

# Red Flags

- ▶ If a function is globally concave, the maximum is usually THE maximum.
- ▶ If not, local maxima may exist. What to do?
- ▶ Try different starting values; attempt a grid search over different ranges of  $\theta$ .
- ▶ Nonconcave functions.
- ▶ The Hessian isn't invertible.
- ▶ Can happen frequently.
- ▶ A BIG problem if your model stops iterating with this kind of error!
- ▶ Unproductive steps. (Not a big deal unless *all* the steps are unproductive.)
- ▶ Flat likelihoods, lots of iterations.
- ▶ Data problems: scaling, limited variance.

# MLE Estimators, Properties of

- ▶ MLE typically only has large-sample properties.
- ▶ That is, they hold asymptotically.
- ▶ There is a premium for large  $n$  . . .
- ▶ and a price to pay for small.
- ▶ Consider the optimizers: they are data intensive.
- ▶ Consistency is a probabilistic statement:

$$\lim_{n \rightarrow \infty} \Pr\{|\hat{\theta} - \theta| < \delta\} = 1 \quad \delta > 0. \quad (44)$$

- ▶ Or  $\text{plim}\hat{\theta} = \theta$ .
- ▶ Cramer-Rao Theorem:

$$\text{var}\boldsymbol{\theta} \geq \left( -E \left[ \frac{\partial^2 \log \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}'} \right] \right)^{-1} \quad (45)$$

- ▶ Under certain regularity conditions, the variance of the MLE will be lower bound. This is asymptotic *efficiency*.

# Binary Response Models: Logit and Probit

- ▶ We now have an idea how these estimators work.
- ▶ Finally (!), let's consider some regression models.
- ▶ Sometimes the OLS estimator simply will not work well for us.
- ▶ The assumption of normally distributed errors may not hold for some DGPs.
- ▶ Further, for some kinds of response variables, classic Gaussian assumptions will not hold.
- ▶ To begin moving toward ML methods, let's consider what happens to the OLS estimator in a very specific setting:
- ▶  $Y = 1, 0$
- ▶ In other words, let's consider a regression model again and hope that it gets us thinking about some other way of proceeding.

# When your response variable is binary: A regression sidetrip

- ▶ Typically assume  $y$  is unbounded.
- ▶ Since the distribution of  $y$  is a function of  $\epsilon$ , if we assume  $\epsilon \sim N$ , then  $\hat{y}$  must also be unbounded.
- ▶ Hence,  $y = \beta_0 + \beta_X + \epsilon$ .
- ▶  $\therefore \hat{y}$  must be unbounded.
- ▶ With a dichotomous response variable, this clearly isn't the case!
- ▶ Sometimes we think of dichotomous  $y$  as representing “latent utilities,” or “probabilities.”
- ▶ As such,  $y$  really measures  $y^*$ , which reflects underlying (and continuous) probability in the interval  $(0,1)$ .

## When your response variable is binary

- ▶ Problem: Linear change in  $E(y) \mid x$  not natural.
- ▶ Why?

$$P(y = 1) = P_i$$

$$P(y = 0) = (1 - P_i) = Q_i$$

$$E(y) = P_i(1) + Q_i(0)$$

$$E(y) = P_i(1)$$

- ▶ In OLS, the coefficients give us the expected change in  $y = 1$  for a unit increase in  $X$ .
- ▶ This won't always make sense.

## Illustration: Regression on dichotomous y:

```

lpdat<-read.dta('d:\\classes\\pol681\\logitprobit.dta')
summary(lpdat)
attach(lpdat)

p1<-plot(x1, d, title("Dichotomous y, Continuous x"))

reg<-lm(d~x1);summary(reg)

Call:
lm(formula = d ~ x1)

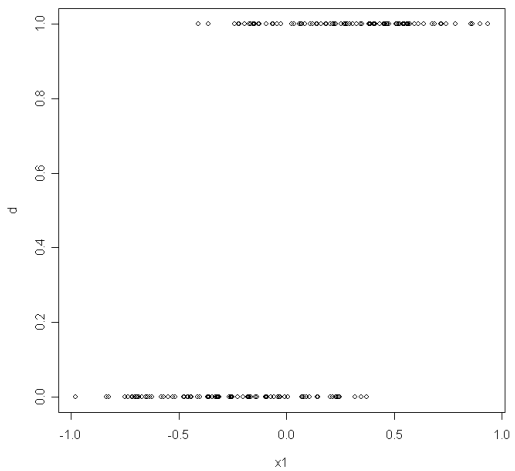
Residuals:
    Min       1Q   Median       3Q      Max
-0.79552 -0.25918  0.03246  0.26522  0.81012

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.50673    0.02904   17.45  <2e-16 ***
x1           0.77499    0.06814   11.37  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3774 on 167 degrees of freedom
Multiple R-Squared: 0.4365,    Adjusted R-squared: 0.4331
F-statistic: 129.3 on 1 and 167 DF,  p-value: < 2.2e-16

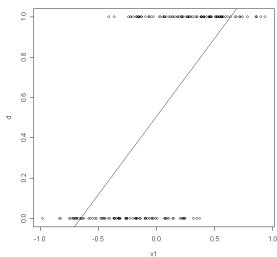
```

## Dichotomous y, Continuous x



# Interpretation

- ▶ For a unit change in  $x$ ,  $E(y)$  changes by .77 units.
- ▶ `yhat<-fitted.values(reg)`  
`plot(d x1) abline(reg)`



- ▶
- ▶ Oops.
- ▶  $\hat{y} = (-.25, 1.23)$ : Can't be probabilities!

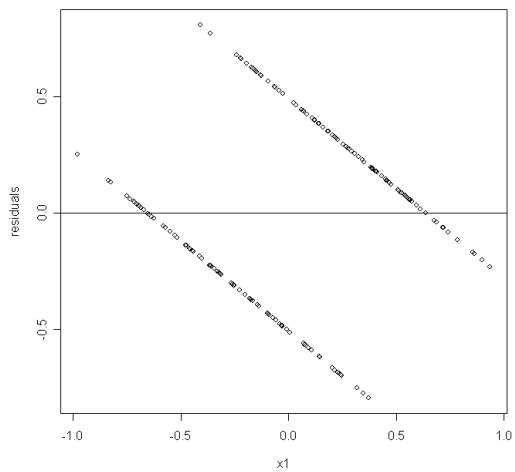
# Interpretation

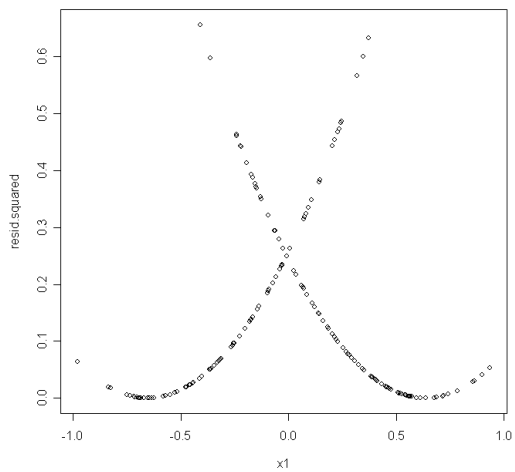
- ▶ We could impose constraints: fix  $\hat{y}$  to lie in unit interval (0,1).
- ▶ This is fairly *ad hoc*.
- ▶ But wait! There are other problems.
- ▶  $\hat{\epsilon} = y_i - \sum \hat{\beta}_k x_i$ .
- ▶

$$y = 1 : 1 - \sum \hat{\beta}_k x_i$$

$$y = 0 : 0 - \sum \hat{\beta}_k x_i$$

- ▶  $\forall x_i, \epsilon$  assumes two values.
- ▶ Plot them (and the squared residuals):





# Heteroskedasticity

- ▶ This is a problem:  $x$  is systematically linked to  $\hat{\epsilon}$ .
  - ▶ Since the variance is a function of the squared residuals and since the squared residuals are a function of  $x$ , the variance in the model is a function of  $x$ . This is heteroskedasticity.
  - ▶  $E(y) = \sum \hat{\beta}_k x_i = P_i$  and  $1 - \sum \hat{\beta}_k x_i = Q_i$
  - ▶ Noting (without proof) that
- $$\text{var}(\epsilon) = (1 - \sum \hat{\beta}_k x_i)^2 P_i + (-\sum \hat{\beta}_k x_i)^2 Q_i$$
- ▶  $\therefore$

$$\begin{aligned}
 \text{var}(\epsilon) &= (Q_i)^2 P_i + (-P_i)^2 Q_i \\
 &= Q_i P_i (Q_i P_i) \\
 &= Q_i P_i ([1 - P_i] + P_i) \\
 &= Q_i P_i \\
 &= (1 - \sum \hat{\beta}_k x_i) (\sum \hat{\beta}_k x_i)
 \end{aligned}$$

- ▶ The variance of  $\hat{\epsilon}$  is a direct function of  $x$ ; it's non-constant.

## OLS

- ▶ Heteroskedasticity means the estimator is no longer minimum variance.
- ▶ Thus, even though  $\hat{\beta}$  are unbiased, the estimator is no longer efficient.
- ▶ That is, it's not “best linear” anymore.
- ▶ There are “fix-ups”: weighted least squares.
- ▶ What about an alternative estimator?

## Initial consideration of a logit estimator:

```
> logit.mod <- glm(d~x1, binomial); summary(logit.dat)
```

Call:

```
glm(formula = d ~ x1, family = binomial)
```

Deviance Residuals:

	Min	1Q	Median	3Q	Max
	-2.0330	-0.6081	0.1545	0.6256	2.0836

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	0.03303	0.20932	0.158	0.875
x1	5.09370	0.75442	6.752	1.46e-11 ***

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

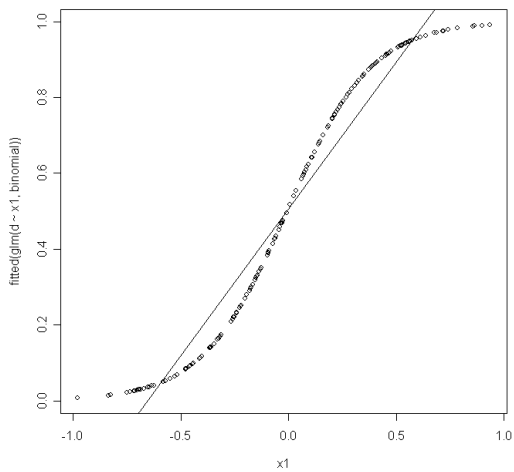
(Dispersion parameter for binomial family taken to be 1)

Null deviance: 234.14 on 168 degrees of freedom  
 Residual deviance: 142.27 on 167 degrees of freedom  
 AIC: 146.27

Number of Fisher Scoring iterations: 5

# Initial Logit

- ▶ Doesn't look like regression (of the OLS variety).
- ▶ It *is* a linear model, however.
- ▶ But only in one part of it.
- ▶ Suppose we like the probability interpretation?
- ▶ In regression, the response function for probabilities was awkward (i.e. a straight line).
- ▶ The  $\Pr(y = 1)$  or  $\Pr(y = 0)$  is ever increasing (decreasing) wrt  $\Delta(x)$ .
- ▶ Probabilities probably exhibit marginality.
- ▶ What do the logit probabilities look like?



# Logit

- ▶ Logit resolves the functional form problem (in terms of the response function in the probabilities.
- ▶ Note that in the probabilities, logit *is* a non-linear model.
- ▶ Suppose  $E(y) = P(y = 1 | x) = \beta_k x_{ik}$ , and  $y$  is binary.

$$\Pr(y = 1 | x) = \frac{1}{1 + \exp(-\sum \beta_k x_{ik})}$$

- ▶ Let  $Z = \sum \beta_k x_{ik}$ , then

$$\Pr(y = 1 | x) = \frac{1}{1 + \exp(-Z)} = \frac{\exp(Z)}{1 + \exp(Z)}$$

- ▶ This is the c.d.f. for the logistic distribution.
- ▶ Problems Solved:
  - $Z$  is unbounded;  $P_i$  must stay in unit interval.
  - $P_i$  is nonlinearly related to parameters (though logit is linear!)
  - Prediction of “1” or “0” impossible.

# Logit

- ▶ Why no perfect prediction?
- ▶ Consider log-odds:

$$\log\left(\frac{p_i}{1-p_i}\right) = Z$$

- ▶ If  $p = 1$ , division by 0.
- ▶ If  $p = 0$ ,  $\log(0)$  (undefined operation).
- ▶ So what?
- ▶ This also holds for probit.

# Interpreting Logit

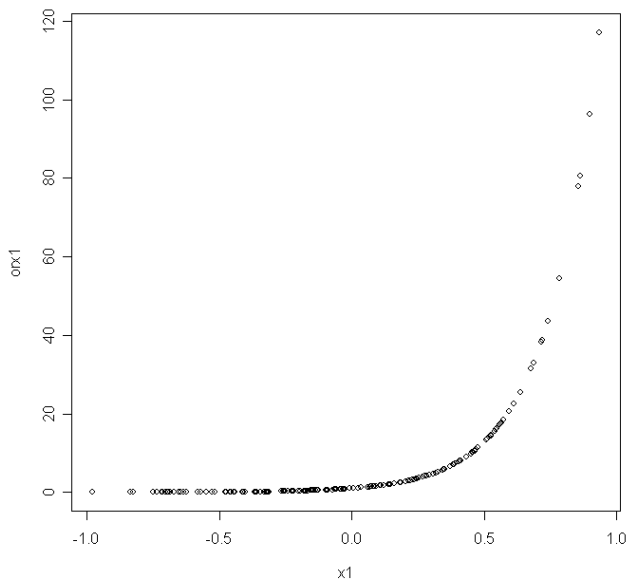
- ▶ Often stated: “logit coefficients are not naturally interpretable.”
- ▶ W R O N G
- ▶ They’re about as easily interpretable as any linear regression model.
- ▶ The metric of the logit coefficients are as log odds ratios.
- ▶ Thus, changes in  $x$  are associated with  $\beta$  change in the log-odds.
- ▶ Since the model is linear in the log-odds ratios, the printed logit coefficients are simply not “impossible” to interpret.
- ▶ Signs make sense.
- ▶ Hypothesis tests make sense.

# Interpreting Logit: Odds Ratios

- ▶ The big issue is the scale.
- ▶ Conversion from log-odds to odds is simple!
- ▶ Exponentiate  $\beta$  and you get an *odds ratio*.
- ▶ These are easy to interpret (though some argue against using them).
- ▶ Example: consider the model we estimated from before.
- ▶  $\log(p_i/1 - p_i) = .03 + 5.09x_1$
- ▶ Odds ratio for  $x_1$ :  $\exp(5.09) = 162.99$

# Interpreting Logit: Odds Ratios

- ▶ There are a couple of ways to interpret this.
- ▶  $\exp(x_1) \times 1 / \exp(x_1) \times 0 = 162.99$
- ▶ Note that the odds ratios will be proportional.
- ▶ Take the ratio of odds for adjacent scores and they will all be the same.
- ▶  $\exp(\text{coef}[2] * 1) / \exp(\text{coef}[2] * .99)$  gives: 1.052257
- ▶  $\exp(\text{coef}[2] * .99) / \exp(\text{coef}[2] * .98)$  gives: 1.052257
- ▶ Odds ratios are really useful (perhaps most useful) for dummy variables.



# Interpreting Logit: Probabilities

- ▶ Probabilities are easy:

$$\Pr(y = 1 | x) = \frac{\exp(Z)}{1 + \exp(Z)}$$

- ▶ Compute the linear prediction; substitute it into the above; out pops a probability value.
- ▶ Logit model from before:  $\Pr(Y = 1 | x_1 = \bar{x})$ .
- ▶  $\exp(\text{coef}[1] + \text{coef}[2] * \text{mean}(x_1)) / (1 + \exp(\text{coef}[1] + \text{coef}[2] * \text{mean}(x_1)))$
- ▶ Yields  $\Pr(Y = 1 | x_1 = \bar{x}) = .521$ .
- ▶ Probabilities for certain covariate profiles may be interesting.
- ▶ But be careful.
- ▶ More on this next week.