Extending the Discrete-Time Hazard Model

James H. Steiger

Department of Psychology and Human Development
Vanderbilt University

GCM, 2010

Extending the Discrete-Time Hazard Model

Introduction

In this module, we explore ways of extending the basic discrete-time hazard model. In particular, we explore:

1. Modeling the shape of the baseline hazard function to be more constrained and parsimonious
2. Altering the link function from the more familiar logit function to a complementary log-log function
3. Using time-varying predictors, and dealing with the problems of interpretation that surround their use
4. Evaluating assumptions underlying the use of our models, the effect of violation of these assumptions, and ways of relaxing the assumptions

Alternative Specifications

The baseline model (i.e., the model with no covariates) discussed in the previous module assigned dummy variables to each of the discrete time periods. This offers some advantages:

1. The baseline function is guaranteed to essentially mimic the shape observed in the life table
2. The model coefficients in $\boldsymbol{\alpha}$ are easy to interpret

Alternative Specifications

However, nothing in the basic model specification requires such a completely general and non-restrictive model. We might choose any functional shape to relate the hazard function with time, and numerous simple alternatives come to mind:

1. Linear
2. Quadratic
3. Higher order polynomials
4. Etc.

Moreover, the completely general specification, besides being rather unparsimonious, might capitalize excessively on chance variations around a parametric functional form, especially when sample size is small.

Alternative Specifications

Ideally, of course, one should begin with a specific theoretical orientation that yields a strong prediction about the shape of the hazard function. In this case, the discrete-time hazard modeling process becomes essentially *confirmatory* in nature, and our task is simplified.

Often, of course, we are operating in an exploratory mode, and all the standard caveats and trade-offs that apply more generally in regression analysis remain in force. More complex models fit better, unless we compensate our "goodness of fit" evaluation for model complexity. Data-snooping and *post hoc* model modification without careful statistical control and/or cross-validation can lead to serious errors.

Alternative Specifications

A functional specification for our base model becomes a more urgent necessity under the following conditions:

1. When the study involves many time periods, in which case the number of dummy predictors will be excessively large, thereby reducing statistical power
2. When hazard probability is expected to be near zero in some time periods, or when some time periods have very small risk sets, in which case ML estimation may become unstable

Basic Polynomial Models for Time

We can gain substantial simplicity by replacing the dummy variables (one for each period) with a single `PERIOD` variable, and then modeling logit $h(t_j)$ as a polynomial function of `PERIOD`.

Table 12.1 of Singer and Willett (p. 411) summarizes the polynomials, starting from a constant, and working through linear and quadratic up to a 5th order polynomial. Table 12.2 shows the application of these models to data from Gamse and Conger (1997) on achievement of academic tenure.

Basic Polynomial Models for Time

Here is the code to generate the tabled values:

```
> tenure<-read.table("tenure_pp.csv", sep=",", header=T)
> attach(tenure)
> PERIOD2 <- PERIOD^2
> PERIOD3 <- PERIOD^3
> PERIOD4 <- PERIOD^4
> PERIOD5 <- PERIOD^5
> PERIOD6 <- PERIOD^6
> general <- glm(EVENT ~ D1 + D2 + D3 + D4 + D5 + D6 +
+   D7 + D8 + D9, family = "binomial")
> order0<-glm(EVENT~1, family="binomial")
> order1<-glm(EVENT~PERIOD, family="binomial")
> order2<-glm(EVENT~PERIOD + PERIOD2, family="binomial")
> order3<-glm(EVENT~PERIOD + PERIOD2 + PERIOD3,
+   family="binomial")
> order4<-glm(EVENT~PERIOD + PERIOD2 + PERIOD3 +
+   PERIOD4, family="binomial")
> order5<-glm(EVENT~PERIOD + PERIOD2 + PERIOD3 +
+   PERIOD4 + PERIOD5, family="binomial")
> dev <- c(order0$deviance, order1$deviance, order2$deviance,
+ order3$deviance, order4$deviance, order5$deviance, general$deviance)
> dev.diff.p <- c(0, dev[1:5] - dev[2:6],0)
> dev.diff.gen <- c(dev - dev[7])
> aic <- c(order0$aic, order1$aic, order2$aic, order3$aic,
+  order4$aic, order5$aic, general$aic)
> n.parameters <- c(1:6,9)
> bic <- dev + n.parameters * log(166)
> table12.2 <- cbind(n.parameters,dev, dev.diff.p,
+  dev.diff.gen, aic,bic)
```

Basic Polynomial Models for Time

Here is the table. Note that the AIC value for the quadratic model has a typographical error in the Singer-Willett text.

```
> table12.2
```

	n.parameters	dev	dev.diff.p	dev.diff.gen	aic	bic
[1,]	1	1037.6	0.00000	206.361	1039.6	1042.7
[2,]	2	867.5	170.10332	36.258	871.5	877.7
[3,]	3	836.3	31.15780	5.100	842.3	851.6
[4,]	4	833.2	3.13159	1.969	841.2	853.6
[5,]	5	832.7	0.42981	1.539	842.7	858.3
[6,]	6	832.7	0.01052	1.528	844.7	863.4
[7,]	9	831.2	0.00000	0.000	849.2	877.2

Evaluating the Models' Fit

There is an extensive discussion of how to evaluate the fit of these models on pages 415–417 of Singer and Willett. The quadratic model seems to be the winner here.

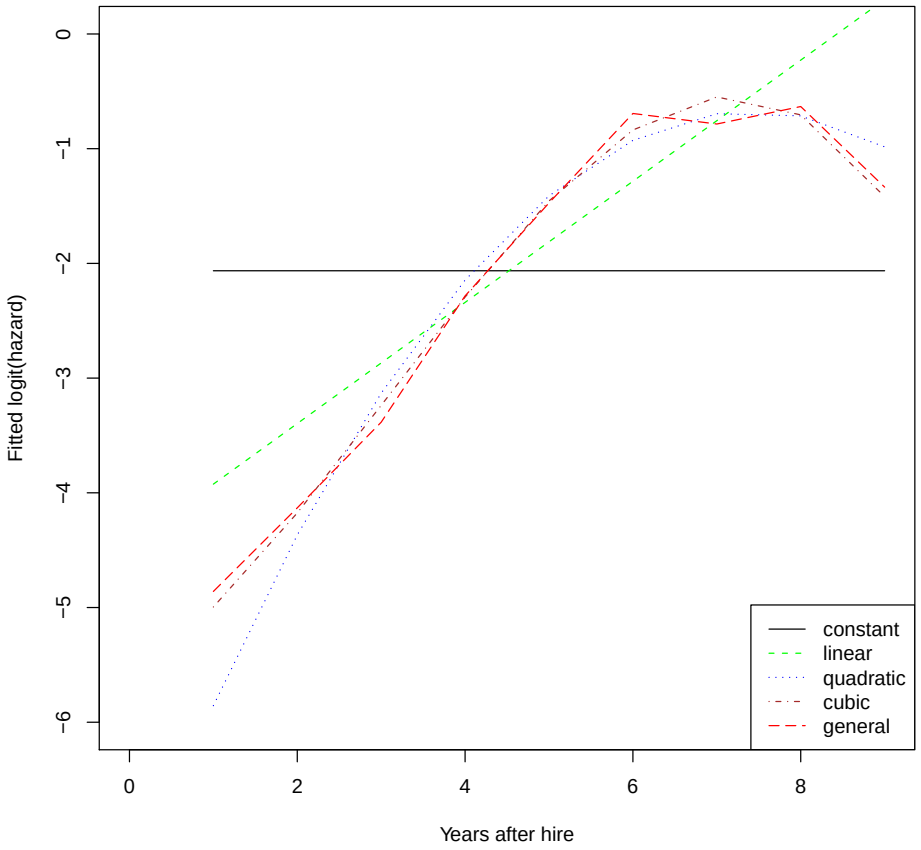
1. The chi-square difference test moving from quadratic to cubic is not significant. ($\chi^2 = 3.13, p = .077$)
2. The AIC is just a smidgen smaller for the cubic model than for the quadratic, while the BIC is smallest for the quadratic
3. There is not much difference in fit between the quadratic and the completely general model

Plotting the Fitted Logit Hazard Functions

Here is code to plot the fitted logit hazard functions for all the models.

```
> general <- glm(EVENT ~ D1 + D2 + D3 + D4 + D5 + D6 + D7 + D8 + D9 - 1, family = "binomial")
> fits <- c()
> survivor.quad = 1
> survivor.gen = 1
> for (i in 1:9){
+   constant = order0$coef[1]
+   linear = order1$coef[1] + order1$coef[2]*i
+   quadratic = order2$coef[1] + order2$coef[2]*i + order2$coef[3]*i**2
+   cubic = order3$coef[1] + order3$coef[2]*i + order3$coef[3]*i**2 + order3$coef[4]*i**3
+   hazard.quad = 1/(1 + exp(-quadratic));
+   survivor.quad = (1 - hazard.quad)*survivor.quad;
+   generalval = general$coef[i]
+   hazard.gen = 1/(1 + exp(-generalval));
+   survivor.gen = (1 - hazard.gen)*survivor.gen;
+   z <- c(i, constant, linear, quadratic, cubic, generalval, hazard.quad, survivor.quad, hazard.gen, survivor.gen)
+   fits <- rbind(fits, z)}
> par(mfrow=c(1,1))
> plot(fits[,1], fits[,2], type = "l", lty = 1, col="black",
+   xlim = c(0,9), ylim = c(-6,0), xlab = "Years after hire", ylab = "Fitted logit(hazard)")
> points(fits[,1], fits[,3], type = "l", lty = 2,col="green")
> points(fits[,1], fits[,4], type = "l", lty = 3,col="blue")
> points(fits[,1], fits[,5], type = "l", lty = 4,col="brown")
> points(fits[,1], fits[,6], type = "l", lty = 5,col="red")
> legend("bottomright", c("constant", "linear", "quadratic", "cubic",
+   "general"), lty = c(1, 2, 3, 4, 5),col=c("black","green","blue","brown","red"))
```

Plotting the Fitted Logit Hazard Functions

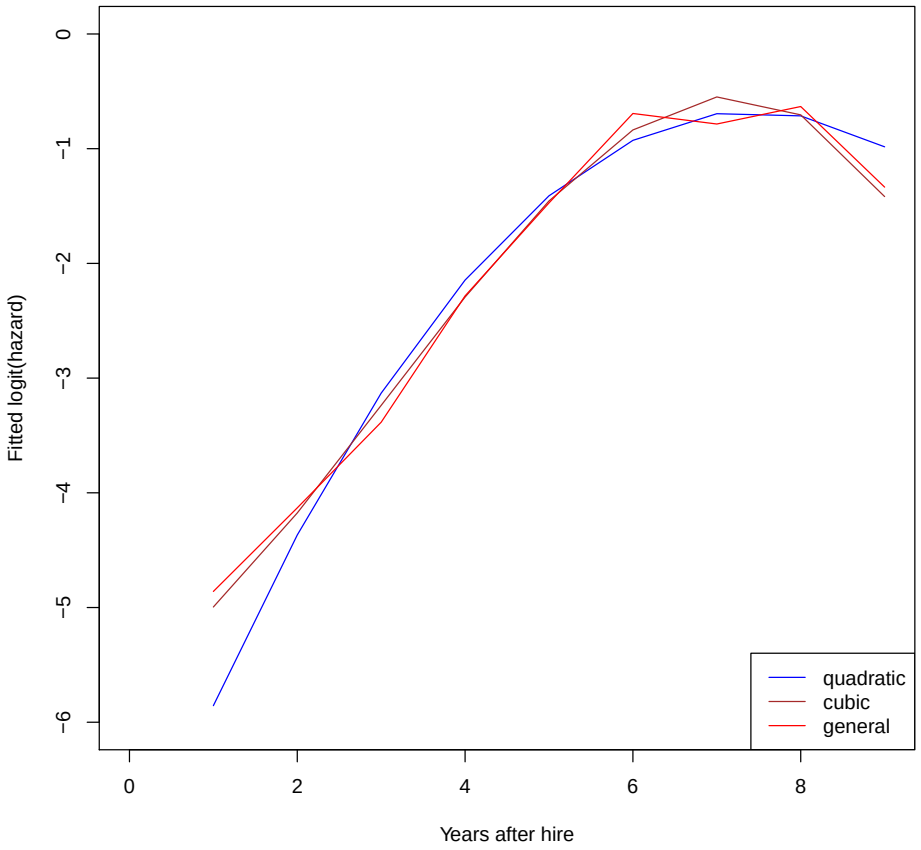


Plotting the Fitted Logit Hazard Functions

We can clean up the plot by eliminating the non-contenders, and it seems that, in the peripheral areas of the plot range, the cubic does a better job than the quadratic. The final choice between cubic and quadratic remains somewhat ambiguous.

```
> par(mfrow=c(1,1))
> plot(fits[,1], fits[,4], type = "l", lty = 1, col="blue",
+   xlim = c(0,9), ylim = c(-6,0), xlab = "Years after hire",
+   ylab = "Fitted logit(hazard)")
> points(fits[,1], fits[,5], type = "l", lty = 1,col="brown")
> points(fits[,1], fits[,6], type = "l", lty = 1,col="red")
> legend("bottomright", c("quadratic", "cubic",
+   "general"), lty = c(1, 1, 1),col=c("blue","brown","red"))
```

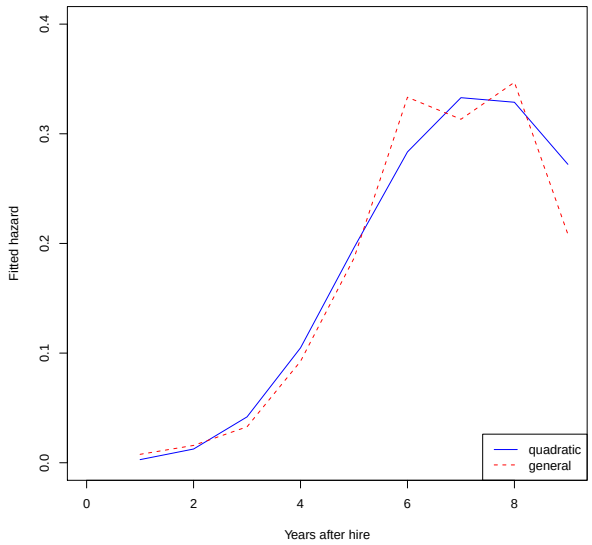
Plotting the Fitted Logit Hazard Functions



Plotting the Fitted Hazard Functions

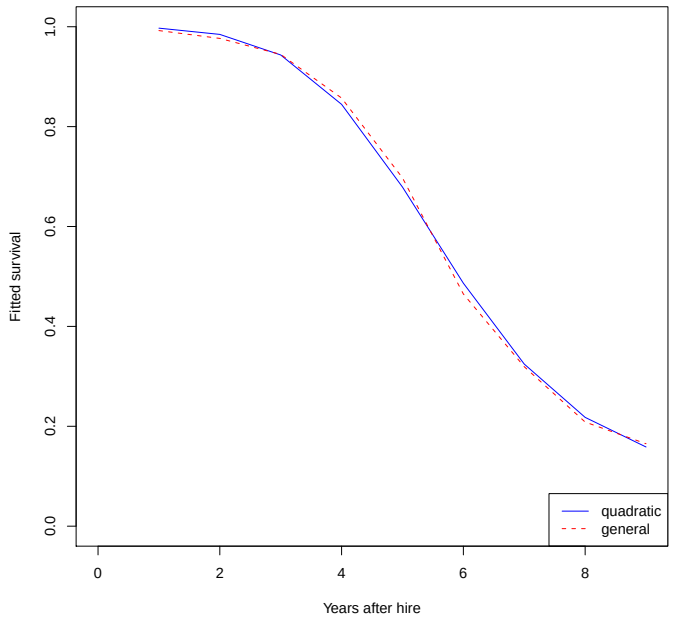
The quadratic plot looks much better when you plot the fitted hazard and survival function, as shown on this slide and the following one.

```
> plot(fits[,1], fits[,7], type = "l", lty = 1, col="blue",xlim = c(0,9),
+ ylim = c(0,.4), xlab = "Years after hire", ylab = "Fitted hazard")
> points(fits[,1], fits[,9], type = "l", lty = 2,col="red")
> legend("bottomright", c("quadratic","general"), lty = c(1, 2),col=c("blue","red"))
```



Plotting the Fitted Survival Functions

```
> plot(fits[,1], fits[,8], type = "l", lty = 1, col="blue",
+ xlim = c(0,9), ylim = c(0,1), xlab = "Years after hire", ylab = "Fitted survival")
> points(fits[,1], fits[,10], type = "l", lty = 2,col="red")
> legend("bottomright", c("quadratic","general"), lty = c(1, 2),col=c("blue","red"))
```



The Complementary Log-Log Link Function

So far, we have discussed the discrete-time survival analysis model as a special case of logistic regression. However, other link functions may be employed, and one particularly interesting example is the `cloglog` family. This function is defined as

$$\text{cloglog}(p) = \log(-\log(1-p)) \tag{1}$$

The function is invertible, since

$$p = 1 - \exp(-\exp(\text{cloglog}(p))) \tag{2}$$

and so

$$\text{cloglog}^{-1}(x) = 1 - \exp(-\exp(x)) \tag{3}$$

Benefits and Drawbacks of the cloglog Function

A plot of the cloglog and logit functions reveals some key facts:

1. At low hazard values, the functions are virtually identical
2. The logit function is symmetric around 0.5, while the cloglog function is not symmetric.
3. While odds of 1 (probability of 0.5) correspond to a convenient and easily remembered value of 0 on the logit scale, the corresponding value on the cloglog scale is a not-so-memorable -0.3665.
4. While the logit link builds in a *proportional odds* assumption in the discrete-time model, the cloglog function builds in a *proportional hazards* assumption. Later, we will discover that one of the most popular continuous survival analysis models, the Cox regression model, also builds in a proportional hazard assumption. Consequently, some analysts might prefer the cloglog link, as it provides analytic continuity not present with the logit link.

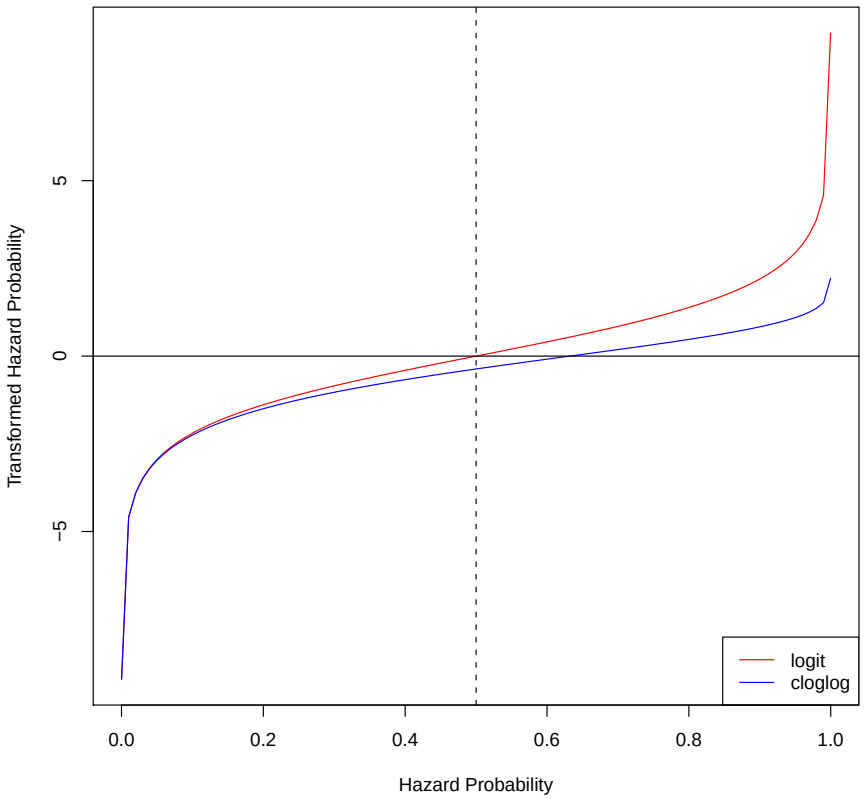
Graphical Comparison of the cloglog and logit Functions

Here is some R code for generating a comparison plot.

```
> curve(logit(x),.0001,.9999,col="red",
+ xlab="Hazard Probability",
+   ylab="Transformed Hazard Probability")
> curve(cloglog(x),.0001,.9999,col="blue",add=TRUE)
> abline(h=0)
> abline(v=0.5,lty=2)
> legend("bottomright",c("logit","cloglog"),
+ lty=c(1,1),col=c("red","blue"))
```

Graphical Comparison of the cloglog and logit Functions

The plot illustrates many of our points of discussion.



A cloglog-based Discrete-Time Model

Modifying the discrete-time model to employ the `cloglog` function is straightforward. We have, for person i,

$$\text{cloglog}\, \boldsymbol{h}_i = \boldsymbol{D}_i\boldsymbol{\alpha} + \boldsymbol{X}_i\boldsymbol{\beta} \tag{4}$$

where, in the fully general model, the matrix $\boldsymbol{D}$ contains the dummy time variables coded 0-1.

Incorporating Covariates

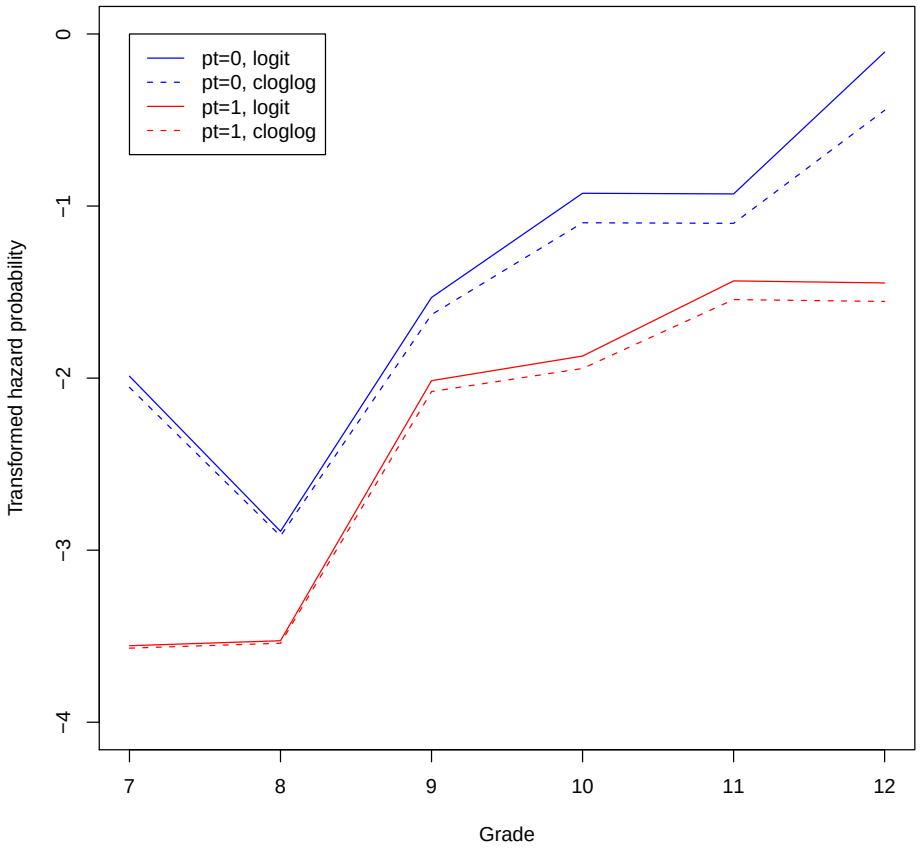
In the next plot, we compare the logit and cloglog hazard functions for $PT = 1$ and $PT = 0$ for our data from the study on age of first heterosexual intercourse in at-risk boys.

Incorporating Covariates

Here is the code for producing the plots:

```
> firstsex<-read.table("firstsex_pp.csv", sep=",", header=T)
> firstsex0  <- subset(firstsex, pt==0)
> firstsex1  <- subset(firstsex, pt==1)
> fs0.logit<-glm(event~d7+d8+d9+d10+d11+d12 - 1,
+ family=binomial(link = "logit"), data = firstsex0)
> fs1.logit<-glm(event~d7+d8+d9+d10+d11+d12 - 1,
+ family=binomial(link = "logit"), data = firstsex1)
> fs0.loglog<-glm(event~d7+d8+d9+d10+d11+d12 - 1,
+ family=binomial(link = "cloglog"), data = firstsex0)
> fs1.loglog<-glm(event~d7+d8+d9+d10+d11+d12 - 1,
+ family=binomial(link = "cloglog"), data = firstsex1)
> fig12.3 <- cbind(time = c(7, 8, 9 , 10, 11, 12),
+ fs0.logit = fs0.logit$coef,
+ fs0.loglog = fs0.loglog$coef, fs1.logit = fs1.logit$coef,
+ fs1.loglog = fs1.loglog$coef)
> par(mfrow=c(1,1))
> plot(fig12.3[,1], fig12.3[,2], type = "l",
+ ylab = "Transformed hazard probability",
+ xlab = "Grade", ylim = c(-4,0),lty=1,col="red")
> points(fig12.3[,1], fig12.3[,3], type = "l", lty=2,col="red")
> points(fig12.3[,1], fig12.3[,4], type = "l", lty=1,col="blue")
> points(fig12.3[,1], fig12.3[,5], type = "l", lty=2,col="blue")
> legend(7, 0, c("pt=0, logit", "pt=0, cloglog",
+ "pt=1, logit", "pt=1, cloglog"), lty = c(1, 2, 1, 2),
+ col=c("blue","blue","red","red"))
```

Graphical Comparison of the cloglog and logit Functions



Time-Varying Predictors

Discrete-time survival analysis is easily adaptable to inclusion of time-varying predictors, which are simply added to the person-period data set. Recall that the model is of the form

$$\text{logit}\, \boldsymbol{h}_i = \boldsymbol{D}_i\boldsymbol{\alpha} + \boldsymbol{X}_i\boldsymbol{\beta} \tag{5}$$

Our statement of the model includes covariates in the matrix $\boldsymbol{X}$ for each individual. These covariates can be either time-varying or time-invariant. If a covariate in column k of matrix $\boldsymbol{X}_i$ for individual i is time-invariant, then values in each row of $\boldsymbol{X}_i$ will be the same.

Modeling Depression Onset

Wheaton, Rozell, and Hall (1997) examined the relationship between life stresses and the onset of depression symptoms in a random sample of adults aged 17–57. The stresses included major hospitalization, physical abuse, and parental divorce. As an example analysis, we assess the influence of parental divorce (PD).

PD is coded 0 for all time periods before the period in which divorce occurred, and is coded 1 in the period in which the divorce occurred and all subsequent periods. Hence, PD is a *time-varying* predictor.

Modeling Depression Onset

We begin by ignoring the PD predictor and assessing the general shape of the relationship between onset of depression and age.

Given that there were 36 time periods and only 387 events, the data are sparse, and a fully general time specification is not practical. Singer and Willett claimed they examined several polynomial models, and found that a cubic model worked best. (You will reproduce this analysis as a homework assignment.)

For convenience, they center the onset age around a time of 18.

Modeling Depression Onset

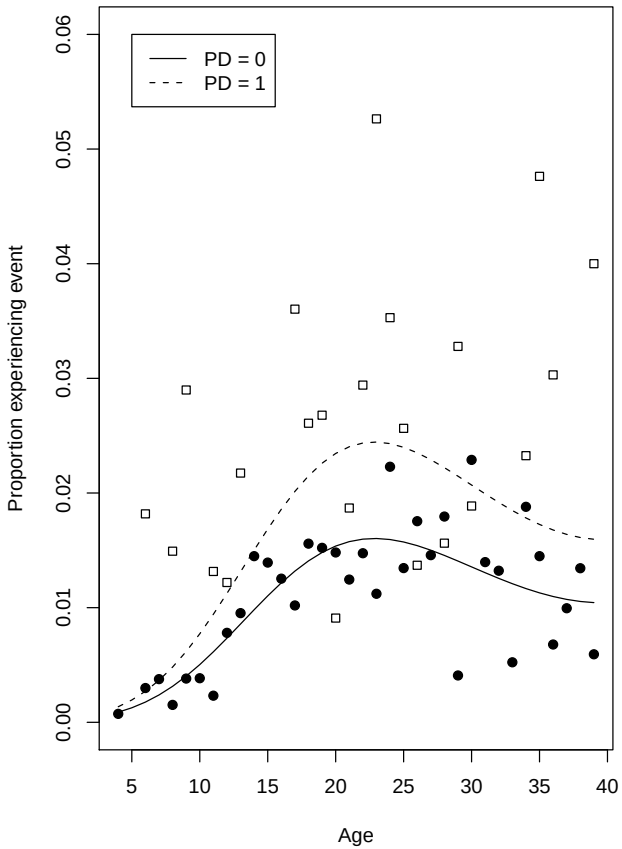
Singer and Willett spend a substantial amount of time discussing the implications of the model, a key one of which is that, in the logit metric, the effect of either a time-invariant or time-varying predictor does not change over time. This allows us to construct functions that represent and "envelope" of the possible individual functions.

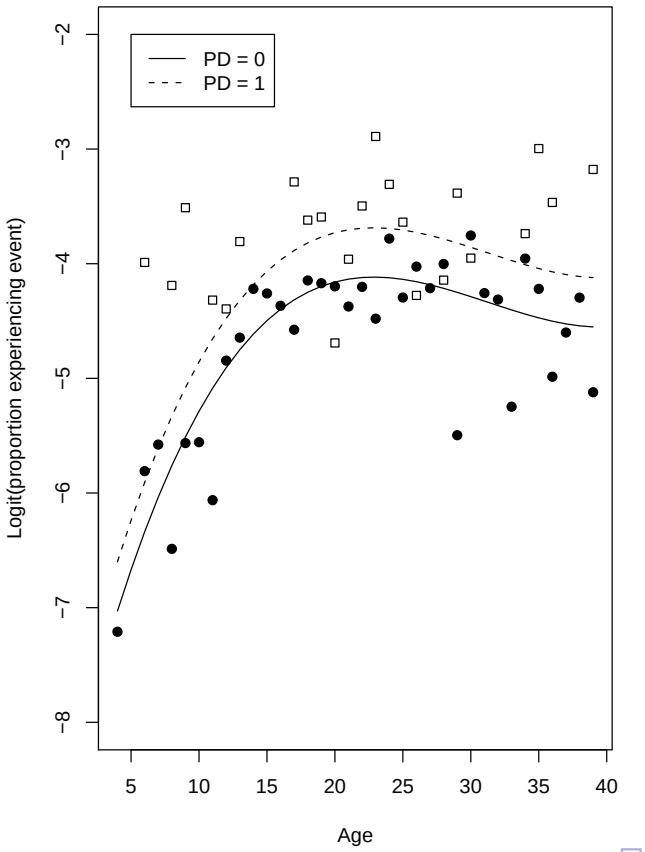
Modeling Depression Onset

```
> depression<-read.table("depression_pp.csv", sep=",", header=T)
> percents <- c()
> for (i in 4:39){
+   for (j in 0:1){
+     x <- subset(depression, PERIOD==i & PD==j)
+     y <- table(x$EVENT)
+     if (dim(y)==2){
+       z <- c(i, j, y[[2]], y[[1]], (y[[2]]/(y[[1]]+y[[2]])))}
+     if (dim(y)==1){
+       z <- c(i, j, 0, y[[1]], NA)}
+     percents <- rbind(percents, z)
+   }
+ }
> percents <- cbind(percents, log(percents[,5]/(1-percents[,5])))
> colnames(percents) <- c("age", "parent", "event", "nonevent", "percent", "logpercent")
> percents.nm <- as.data.frame(na.omit(percents))
> percent.pd0 <- subset(percents.nm, parent == 0)
> percent.pd1 <- subset(percents.nm, parent == 1)
> dmodel<-glm(EVENT ~ ONE + age_18 + age_18sq + age_18cub + PD - 1,
+   family=binomial(link = "logit"), data = depression)
> modelfit<- c()
> for (i in 0:1){
+   for (j in 4:39){
+     fitx = dmodel$coef[1] + dmodel$coef[2]*(j-18) + dmodel$coef[3]*(j-18)^2 +
+        dmodel$coef[4]*(j-18)^3 + dmodel$coef[5]*i
+     fithazard = 1/(1 + exp(-fitx))
+     modelfit <- rbind(modelfit, c(i, j, fitx, fithazard))
+   }
+ }
> modelfit.0 <- subset(data.frame(modelfit), modelfit[,1]==0)
> modelfit.1 <- subset(data.frame(modelfit), modelfit[,1]==1)
```

```
> par(mfrow=c(1,2))
> plot(percent.pd0$age, percent.pd0$percent, pch = 19,
+   ylim = c(0, .06), xlab = "Age",
+   ylab = "Proportion experiencing event")
> points(percent.pd1$age, percent.pd1$percent, pch = 22)
> points(modelfit.0[,2], modelfit.0[,4], type = 'l', lty = 1)
> points(modelfit.1[,2], modelfit.1[,4], type = 'l', lty = 2)
> legend(5, 0.06, c("PD = 0", "PD = 1"), lty = c(1, 2))
> plot(percent.pd0$age, percent.pd0$logpercent, pch = 19,
+   ylim = c(-8, -2), xlab = "Age",
+   ylab = "Logit(proportion experiencing event)")
> points(percent.pd1$age, percent.pd1$logpercent, pch = 22)
> points(modelfit.0[,2], modelfit.0[,3], type = 'l', lty = 1)
> points(modelfit.1[,2], modelfit.1[,3], type = 'l', lty = 2)
> legend(5, -2, c("PD = 0", "PD = 1"), lty = c(1, 2))
```

Hazard Function Envelope





Interpreting Model Coefficients

Here is the model fit with both PD and FEMALE as predictors.

```
> quadratic.pd.gender <- glm(EVENT~ 1 + age_18 +
+ I(age_18^2) +  I(age_18^3) + PD + FEMALE,
+  family="binomial"(link="logit"),data=depression)
```

Interpreting Model Coefficients

```
> summary(quadratic.pd.gender)

Call:
glm(formula = EVENT ~ 1 + age_18 + I(age_18^2) + I(age_18^3) +
    PD + FEMALE, family = binomial(link = "logit"), data = depression)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.2452  -0.1759  -0.1431  -0.0994   3.6949

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.586640   0.107024  -42.86  < 2e-16 ***
age_18       0.059599   0.011653    5.11  3.1e-07 ***
I(age_18^2) -0.007360   0.001224   -6.01  1.8e-09 ***
I(age_18^3)  0.000185   0.000079    2.34    0.019 *
PD           0.415055   0.162021    2.56    0.010 *
FEMALE       0.545451   0.109409    4.99  6.2e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 4299.5  on 36996  degrees of freedom
Residual deviance: 4139.2  on 36991  degrees of freedom
AIC: 4151

Number of Fisher Scoring iterations: 8
```

Interpreting Model Coefficients

```
> exp(coefficients(quadratic.pd.gender)["PD"])

   PD
1.514

> exp(coefficients(quadratic.pd.gender)["FEMALE"])

FEMALE
 1.725
```

We see from the standard logistic regression analytic approach that, in this model, controlling for gender and time, parental divorce in the life history increases the likelihood of depression onset by 51%.

Correspondingly, controlling for parental divorce and time, females are 72.5% more likely than males to report depression onset.

Plotting Prototypical Functions

In attempting to portray the effects of various predictors on hazard and survival functions, we often rely on the device of plotting these functions for hypothetical individuals who are, in some sense, prototypical.

Singer and Willett present a set of such plots in their Figure 12.5.

They suggest approaches for selecting "prototypical" time-variant values.

Plotting Prototypical Time-Varying Values

In the code below, we compute the model fit, then use the model coefficients to generate predicted values for all of the time/PD/sex combinations we are interested in. This is done in a loop. Within the loop, we use the product-limit formula to compute survival rate at each time point.

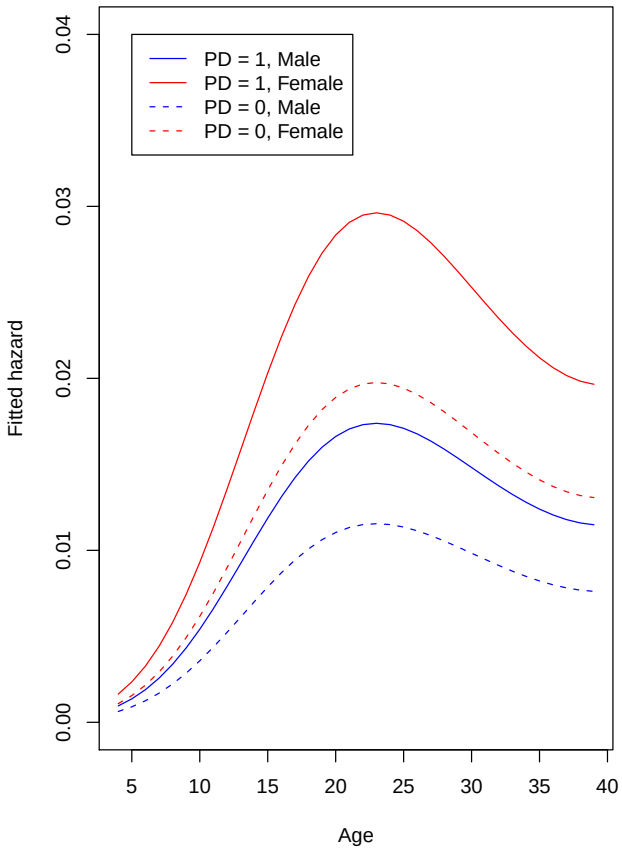
```
> depression<-read.table("depression_pp.csv", sep=",", header=T)
> dmodel<-glm(EVENT ~ 1 + age_18 + I(age_18^2) + I(age_18^3) + PD
+  + FEMALE, family=binomial(link = "logit"), data = depression)
> modelfit<- c()
> for (i in 0:1){
+   for (k in 0:1){
+   survivor <- 1
+     for (j in 4:39){
+     fitx = dmodel$coef[1] + dmodel$coef[2]*(j-18) + dmodel$coef[3]*(j-18)^2 +
+       dmodel$coef[4]*(j-18)^3 + dmodel$coef[5]*i + dmodel$coef[6]*k
+     hazard = 1/(1 + exp(-fitx))
+     survivor = (1-hazard)*survivor
+     modelfit <- rbind(modelfit, c(i, j, k, fitx, hazard, survivor))}}}
> colnames(modelfit) <- c("pd", "age", "female", "fit", "hazard", "survival")
> modelfit.0male <- subset(data.frame(modelfit), modelfit[,1]==0 & modelfit[,3]==0)
> modelfit.0female <- subset(data.frame(modelfit), modelfit[,1]==0 & modelfit[,3]==1)
> modelfit.1male <- subset(data.frame(modelfit), modelfit[,1]==1 & modelfit[,3]==0)
> modelfit.1female <- subset(data.frame(modelfit), modelfit[,1]==1 & modelfit[,3]==1)
```

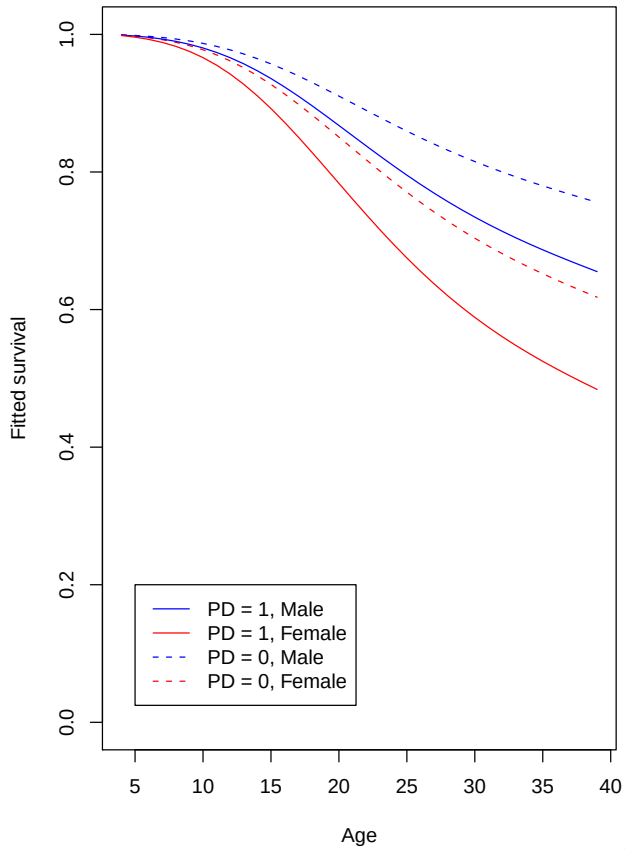
Plotting Prototypical Time-Varying Values

Here is the code to generate the plot. Note, to enhance readability, I am coding males in blue, females in red, and PD=1 with a solid line, PD=0 with dashed line.

```
> par(mfrow=c(1,2))
> plot(modelfit.0male$age, modelfit.0male$hazard,
+ type = 'l', lty = 2, ylim = c(0, .04), xlab = "Age",
+ ylab = "Fitted hazard",col="blue")
> points(modelfit.0female$age, modelfit.0female$hazard,
+ type = 'l', lty = 2,col="red")
> points(modelfit.1male$age, modelfit.1male$hazard,
+ type = 'l', lty = 1,col="blue")
> points(modelfit.1female$age, modelfit.1female$hazard,
+ type = 'l', lty = 1,col="red")
> legend(5, 0.04, c("PD = 1, Male", "PD = 1, Female", "PD = 0, Male",
+ "PD = 0, Female"), lty = c(1, 1, 2, 2),col = c("blue","red","blue","red"))
> plot(modelfit.0male$age, modelfit.0male$survival, type = 'l', lty = 2,
+ ylim = c(0, 1), xlab = "Age", ylab = "Fitted survival",col="blue")
> points(modelfit.0female$age, modelfit.0female$survival, type = 'l',
+  lty = 2,col="red")
> points(modelfit.1male$age, modelfit.1male$survival, type = 'l',
+ lty = 1,col="blue")
> points(modelfit.1female$age, modelfit.1female$survival, type = 'l',
+ lty = 1,col="red")
> legend(5, 0.2, c("PD = 1, Male", "PD = 1, Female", "PD = 0, Male",
+ "PD = 0, Female"), lty = c(1, 1, 2, 2),col = c("blue","red","blue","red"))
```

Plotting Prototypical Time-Varying Values





State Dependence

A time-varying predictor is *state-dependent* if its values at time t_j are affected by an individual's state (i.e., event-occurrence status) at time t_j.

Rate Dependence

A time-varying predictor is *rate-dependent* if its values at time t_j are affected by an individual's hazard rate at time t_j.

An Example

Example (State and Rate Dependence)

- In a study assessing marriage breakups, time-varying predictors are marital satisfaction and employment status
- Either of these covariates could affect the likelihood of a marriage breakup
- However, the occurrence of a marriage breakup or an increased likelihood of a marriage breakup could also cause a change in marital satisfaction or employment status
- In other words, there is a problem in assessing causal direction.

Use of Lagged Predictors

In an attempt to eliminate competing causal attributions, some researchers use time-varying predictors that have been lagged for one or more time periods.

However, lagging is no panacea, because:

1. The first value in the sequence may need to be imputed.
2. *Anticipatory effects* can also occur.

Introduction

All the discrete-time models we've examined so far are linear, with all the restrictions that linearity entails. Linearity can be violated in numerous ways. For example:

1. Predictors can interact
2. The basic functional form can be nonlinear

Interactions between Substantive Predictors

When there are several predictors, there are many potential interactions to investigate, and one must be careful to avoid capitalizing on chance when declaring interactions "significant."

Of course, when substantive theory strongly suggests an interaction, you can propose to test it *a priori* and escape this problem to some extent.

Searching for interactions often can begin by plotting hazard or survival functions within groups that are defined by values on one of the predictors

Uncovering Statistical Interactions

Here, we load the data from Keiley and Martin (2002). These authors examined the effect of child abuse on the risk of first juvenile arrest. The variable *ABUSED* was the focal predictor, and a key question was whether or not its effect varied across race. The variable *BLACK* was therefore examined for an interaction with *ABUSED*. In the next slide, we present the code (again adapted from the UCLA website) for reading in the data and computing separate *sample* logit-hazard curves for the 4 combinations of the variables *BLACK* and *ABUSED*.

Uncovering Statistical Interactions

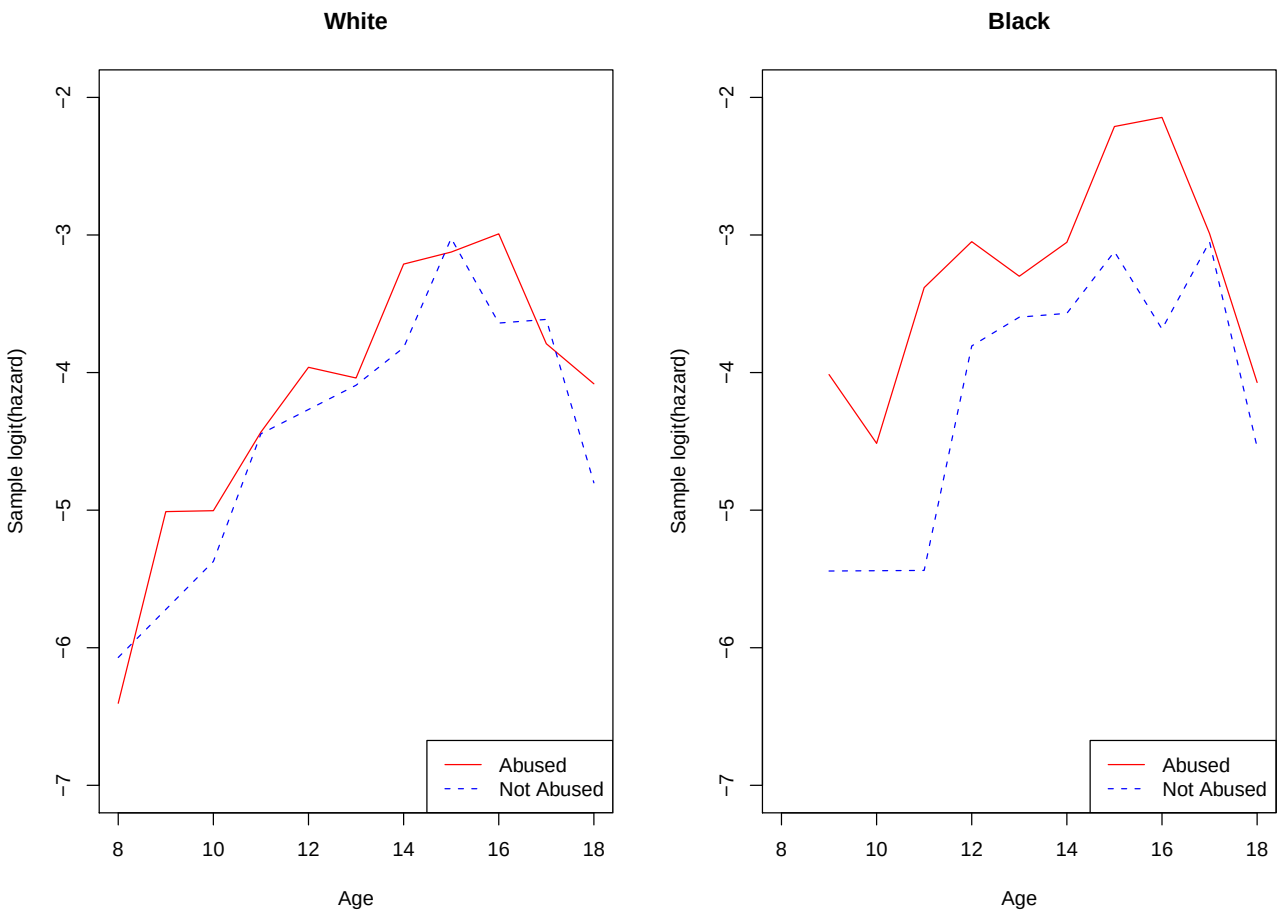
```
> fa<-read.table("firstarrest_pp.csv", sep=",", header=T)
> percents <- c()
> for (i in 8:18){
+   for (j in 0:1){
+     for (k in 0:1){
+       x <- subset(fa, PERIOD==i & BLACK==j & ABUSED==k)
+       if (sum(x$EVENT) > 0){
+         y <- mean(x$EVENT)}
+       if (sum(x$EVENT)==0){
+         y <- NA}
+       logity <- log(y/(1-y))
+       z <- c(i, j, k, y, logity)
+       percents <- rbind(percents, z)}}}
> colnames(percents) <- c("age", "black", "abused", "pct", "hazard")
> percents.nm <- as.data.frame(na.omit(percents))
> percents.w.a <- subset(percents.nm, black==0 & abused==1)
> percents.b.a <- subset(percents.nm, black==1 & abused==1)
> percents.w.na <- subset(percents.nm, black==0 & abused==0)
> percents.b.na <- subset(percents.nm, black==1 & abused==0)
```

Uncovering Statistical Interactions

Here is code to construct plots, which are shown on the next slide.

```
> par(mfrow=c(1,2))
> plot(percents.w.a$age, percents.w.a$hazard, type = "l",
+ lty = 1, col="red",ylim = c(-7, -2), xlim = c(8,18),
+ main = "White", xlab = "Age", ylab = "Sample logit(hazard)")
> points(percents.w.na$age, percents.w.na$hazard,
+  col="blue",type = "l", lty = 2)
> legend("bottomright", c("Abused", "Not Abused"),
+ col=c("red","blue"),lty = c(1, 2))
> plot(percents.b.a$age, percents.b.a$hazard, type = "l",
+ lty = 1, ylim = c(-7, -2), xlim = c(8,18),
+ main = "Black", xlab = "Age", col="red",ylab = "Sample logit(hazard)")
> points(percents.b.na$age, percents.b.na$hazard,
+ type = "l", lty = 2,col="blue")
> legend("bottomright", c("Abused", "Not Abused"),
+ col=c("red","blue"),lty = c(1, 2))
```

Uncovering Statistical Interactions



Uncovering Statistical Interactions

Next, we construct similar plots based on the *fitted models*.
Here is the code (compare my succinct expression of the model with the UCLA version):

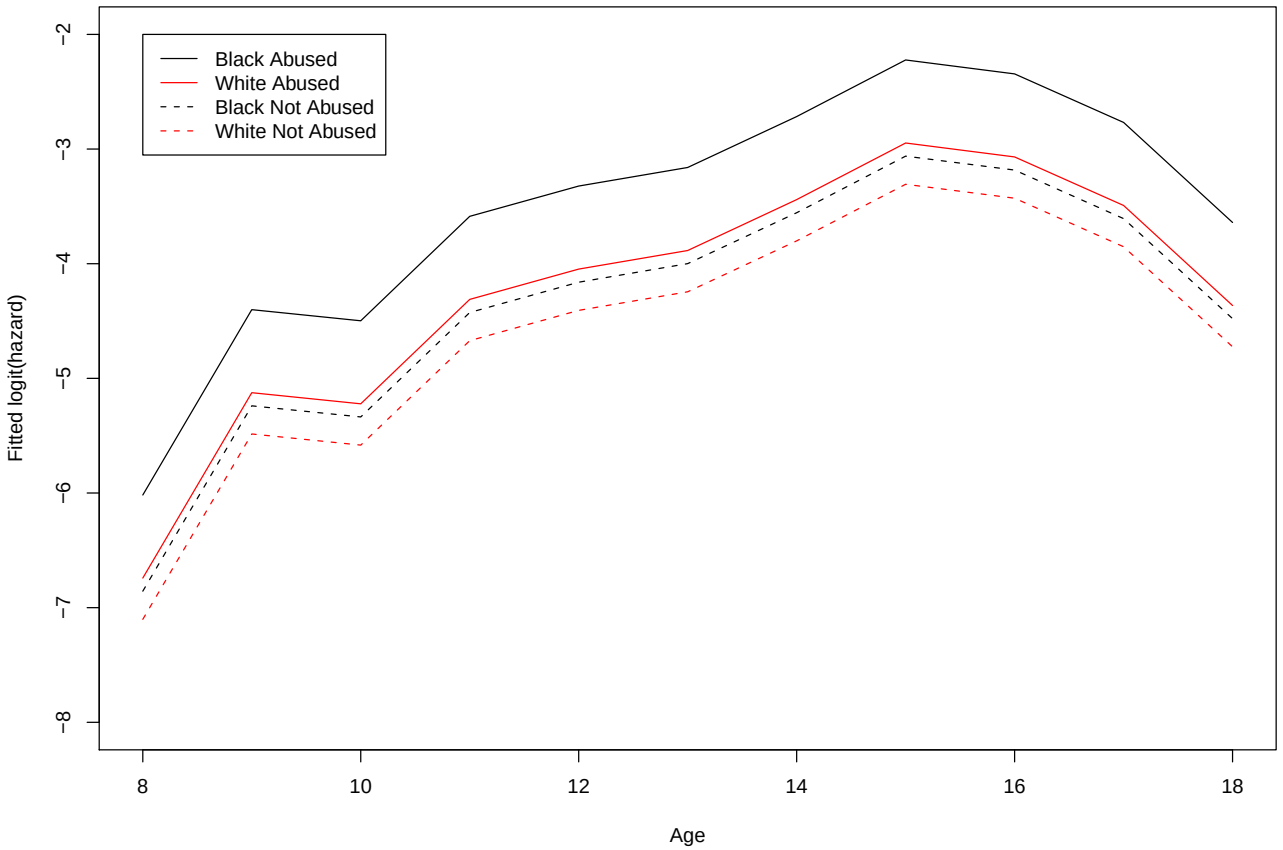
```
> famodel <- glm(formula = EVENT ~ factor(PERIOD) + ABUSED * BLACK - 1,
+ family = "binomial"(link="logit"),data = fa)
> modelfit <- c()
> for (j in 0:1){
+   for (k in 0:1){
+   survivor <- 1
+     for (i in 8:18){
+       logitfit <- famodel$coef[i-7] +
+        famodel$coef[12]*j + famodel$coef[13]*k + famodel$coef[14]*j*k
+       hazard = 1/(1 + exp(-logitfit))
+       survivor = (1-hazard)*survivor
+       z <- c(i, j, k, j*k, logitfit, hazard, survivor)
+       modelfit <- rbind(modelfit, z)}}}
> colnames(modelfit) <- c("age", "abused", "black",
+   "ablack", "logitfit", "hazard", "survival")
> modelfit <- as.data.frame(modelfit)
> modelfit.w.a <- subset(modelfit, black==0 & abused==1)
> modelfit.b.a <- subset(modelfit, black==1 & abused==1)
> modelfit.w.na <- subset(modelfit, black==0 & abused==0)
> modelfit.b.na <- subset(modelfit, black==1 & abused==0)
```

Uncovering Statistical Interactions

Here is the code to draw the plots:

```
> par(mfrow=c(1,1))
> plot(modelfit.w.a$age, modelfit.w.a$logitfit, type = "l",
+ lty = 1, col="red",ylim = c(-8, -2), xlim = c(8,18),
+   xlab = "Age", ylab = "Fitted logit(hazard)")
> points(modelfit.w.na$age, modelfit.w.na$logitfit,
+   type = "l", lty = 2,col="red")
> points(modelfit.b.a$age, modelfit.b.a$logitfit,
+   type = "l", lty = 1,col="black")
> points(modelfit.b.na$age, modelfit.b.na$logitfit,
+   type = "l", lty = 2,col="black")
> legend(8, -2, c("Black Abused", "White Abused",
+   "Black Not Abused","White Not Abused"),
+   lty = c(1, 1,2, 2),col=c("black","red","black","red"))
```

Uncovering Statistical Interactions



Nonlinear Effects

On pages 447–451, Singer and Willett illustrate an exploratory approach to evaluating nonlinearity in the discrete-time hazard model, using the depression-onset data analyzed previously. In this example, besides *PD* and *FEMALE*, and additional predictor, the number of siblings (*NSIBS*) is added.

The first model simply adds *NSIBS* as a linear predictor.

```
> model.a <- glm(EVENT ~ 1 + age_18 + I(age_18^2) +
+ I(age_18^3) + PD + FEMALE + NSIBS,
+ family="binomial"(link="logit"),data=depression)
```

Nonlinear Effects

```
> summary(model.a)

Call:
glm(formula = EVENT ~ 1 + age_18 + I(age_18^2) + I(age_18^3) +
    PD + FEMALE + NSIBS, family = binomial(link = "logit"), data = depression)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-0.272  -0.173  -0.140  -0.095   3.847

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.358696   0.121599  -35.84  < 2e-16 ***
age_18       0.061061   0.011661    5.24  1.6e-07 ***
I(age_18^2) -0.007308   0.001224   -5.97  2.3e-09 ***
I(age_18^3)  0.000182   0.000079    2.30  0.02145 *
PD           0.372601   0.162379    2.29  0.02175 *
FEMALE       0.558686   0.109472    5.10  3.3e-07 ***
NSIBS       -0.081411   0.022272   -3.66  0.00026 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 4299.5  on 36996  degrees of freedom
Residual deviance: 4124.3  on 36990  degrees of freedom
AIC: 4138

Number of Fisher Scoring iterations: 8
```

Nonlinear Effects

The second model looks for linearity, by breaking down the *NSIBS* variable into a set of category variables, and fitting them, while looking for a divergent pattern of coefficients. In line with their own suggestion, Singer and Willett employ a small number of dummy variables, corresponding to roughly equal-spaced groups of observations.

```
> model.b<-glm(EVENT ~ 1 + age_18 + I(age_18^2) +
+  I(age_18^3) + PD + FEMALE + SIBS12 + SIBS34 +
+  SIBS56 + SIBS78 + SIBS9PLUS,
+  family=binomial(link = "logit"), data = depression)
```

Nonlinear Effects

```
> summary(model.b)

Call:
glm(formula = EVENT ~ 1 + age_18 + I(age_18^2) + I(age_18^3) +
    PD + FEMALE + SIBS12 + SIBS34 + SIBS56 + SIBS78 + SIBS9PLUS,
    family = binomial(link = "logit"), data = depression)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.2563  -0.1738  -0.1401  -0.0929   3.6864

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.500069   0.206684  -21.77  < 2e-16 ***
age_18       0.061452   0.011663    5.27  1.4e-07 ***
I(age_18^2) -0.007289   0.001223   -5.96  2.6e-09 ***
I(age_18^3)  0.000181   0.000079    2.30    0.022 *
PD           0.372713   0.162483    2.29    0.022 *
FEMALE       0.559590   0.109528    5.11  3.2e-07 ***
SIBS12       0.020851   0.197602    0.11    0.916
SIBS34       0.010761   0.210029    0.05    0.959
SIBS56      -0.494220   0.254537   -1.94    0.052 .
SIBS78      -0.775399   0.343704   -2.26    0.024 *
SIBS9PLUS   -0.658483   0.344042   -1.91    0.056 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 4299.5  on 36996  degrees of freedom
Residual deviance: 4118.0  on 36986  degrees of freedom
AIC: 4140

Number of Fisher Scoring iterations: 8
```

Nonlinear Effects

A quick examination of the coefficients from the output of model B suggests that, in fact, there is a significant breakpoint between large and small families, starting at families with 5 or 6 siblings. To investigate this with a more parsimonious model, Singer and Willett created a binary 0-1 dummy variable *BIGFAMILY*, and fit a model including it, along with *PD* and *FEMALE*

```
> model.c <- glm(EVENT ~ 1 + age_18 + I(age_18^2) +
+  I(age_18^3) + PD + FEMALE + BIGFAMILY,
+  family=binomial(link = "logit"), data = depression)
```

Nonlinear Effects

```
> summary(model.c)

Call:
glm(formula = EVENT ~ 1 + age_18 + I(age_18^2) + I(age_18^3) +
    PD + FEMALE + BIGFAMILY, family = binomial(link = "logit"),
    data = depression)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.2554  -0.1743  -0.1411  -0.0957   3.7132

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.482812   0.108709  -41.24  < 2e-16 ***
age_18       0.061410   0.011663    5.27  1.4e-07 ***
I(age_18^2) -0.007291   0.001224   -5.96  2.5e-09 ***
I(age_18^3)  0.000181   0.000079    2.30    0.022 *
PD           0.371032   0.162293    2.29    0.022 *
FEMALE       0.558050   0.109471    5.10  3.4e-07 ***
BIGFAMILY   -0.610782   0.144574   -4.22  2.4e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 4299.5  on 36996  degrees of freedom
Residual deviance: 4118.8  on 36990  degrees of freedom
AIC: 4133

Number of Fisher Scoring iterations: 8
```

Nonlinear Effects

Model C has a somewhat lower AIC than model A, and seems to capture the essence of the situation better and just as compactly as model A.

The Proportionality Assumptions

A built-in assumption in the models we have examined so far is that a predictor's effect does not depend on the respondent's duration in the initial state.

This would imply, for example, that in the data set we just examined, the effect of *BIGFAMILY* would be the same in childhood as in adulthood.

One can easily imagine that this might not be true.

Types of Violations

There are several types of potential violations. For example:

1. The predictor's effect may increase with time, and as a result the hazard plots will diverge.
2. The predictor's effect may decrease with time, and consequently the hazard plots will converge.
3. Effect of the predictor will vary from period to period in a more complex fashion.

Types of Violations

Models that include an interaction between time and the predictor can handle the above kinds of violations. For example, if the effect increases or decreases linearly with time, a model that adds a single interactio term can work well.

On the other hand, if the effect varies from period to period, a more general interaction model that includes a separate intercation term (each with its own regression coefficient) for each time period may be called for.

Strategies for Investigating Violations

Strategies for studying interactions between a predictor and *TIME* are illustrated with data from a study by Graham (1997), who tracked the mathematics course-taking history of 3790 high school students.

The key questions were:

1. When would students "leave mathematics," i.e., take their last course?
2. To what extent would the patterns differ by gender?

Strategies for Investigating Violations

These questions were investigated by means of a sequence of 3 models:

1. Model A was a standard model using gender ($FEMALE$) as a predictor, thus incorporating the standard proportionality assumption
2. Model B was a completely general interaction model, incorporating a separate interaction term at each of the 5 time points
3. Model C was the standard model, augmented by a single linear interaction term and its coefficient

The code on the following slides fits the 3 models, but also computes and displays the fitted logit hazard functions for the sample, as well as for all 3 fitted models.

Strategies for Investigating Violations

```
> math<-read.table("mathdropout_pp.csv", sep=",", header=T)
> ##########################
> #  First, for each combination of time and sex, we use a loop
> #  to generate the proportion of subjects experiencing the
> #  event and calculate the logit of the proportion.
> #  Then we subset this proportion and logit data by sex.
> ##########################
> percents <- c()
> for (i in 1:5){
+   for (j in 0:1){
+     x <- subset(math, PERIOD==i & FEMALE==j)
+     if (sum(x$EVENT) > 0){
+       y <- mean(x$EVENT)}
+     if (sum(x$EVENT)==0){
+       y <- NA}
+     logity <- log(y/(1-y))
+     z <- c(i, j, y, logity)
+     percents <- rbind(percents, z)}}
> colnames(percents) <- c("term", "female", "pct", "logit")
> percents <- as.data.frame(na.omit(percents))
> percents.m <- subset(percents, female==0)
> percents.f <- subset(percents, female==1)
```

Strategies for Investigating Violations

```
> ####################
> ## Next, we run the first of three models, Model A.
> ## Then, using a loop, we generate predicted proportion
> ## values for each combination of time and sex with the
> ## coefficients from Model A and calculate the logit of
> ## the fitted proportion.  Then we subset this proportion
> ## and logit data by sex.
> ####################
> modelA<-glm(EVENT ~ HS11 + HS12 + COLL1 + COLL2 +
+  COLL3 + FEMALE - 1, family=binomial(link = "logit"),
+  data = math)
> modelfitA <- c()
> for (i in 1:5){
+   for (j in 0:1){
+     logitfit <- modelA$coef[i] + modelA$coef[6]*j
+     hazard = 1/(1 + exp(-logitfit))
+     z <- c(i, j, logitfit, hazard)
+     modelfitA <- rbind(modelfitA, z)}}
> colnames(modelfitA) <- c("term", "female", "logitfit", "hazard")
> modelfitA <- as.data.frame(modelfitA)
> modelfitA.m <- subset(modelfitA, female==0)
> modelfitA.f <- subset(modelfitA, female==1)
```

Strategies for Investigating Violations

```
> ############################
> ## Next, we run the second of three models, Model B,
> ## and go through the same steps as for Model A.
> ############################
> modelB <- glm(EVENT ~ HS11 + HS12 + COLL1 + COLL2 +
+   COLL3 + FHS11 + FHS12 + FCOLL1 + FCOLL2 + FCOLL3 - 1,
+   family=binomial(link = "logit"), data = math)
> modelfitB <- c()
> for (i in 1:5){
+   for (j in 0:1){
+     logitfit <- modelB$coef[i] + modelB$coef[i+5]*j
+     hazard = 1/(1 + exp(-logitfit))
+     z <- c(i, j, logitfit, hazard)
+     modelfitB <- rbind(modelfitB, z)}}
> colnames(modelfitB) <- c("term", "female", "logitfit", "hazard")
> modelfitB <- as.data.frame(modelfitB)
> modelfitB.m <- subset(modelfitB, female==0)
> modelfitB.f <- subset(modelfitB, female==1)
```

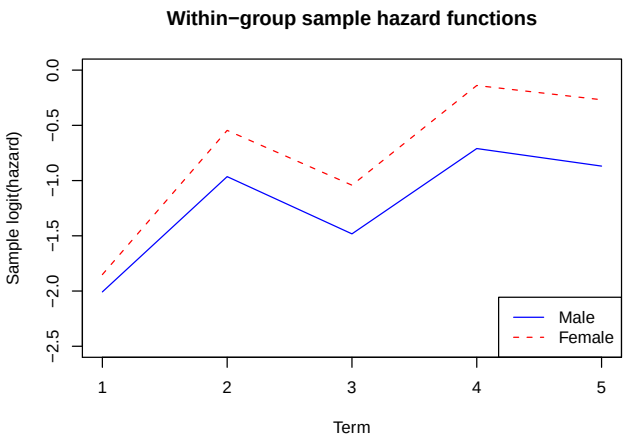
Strategies for Investigating Violations

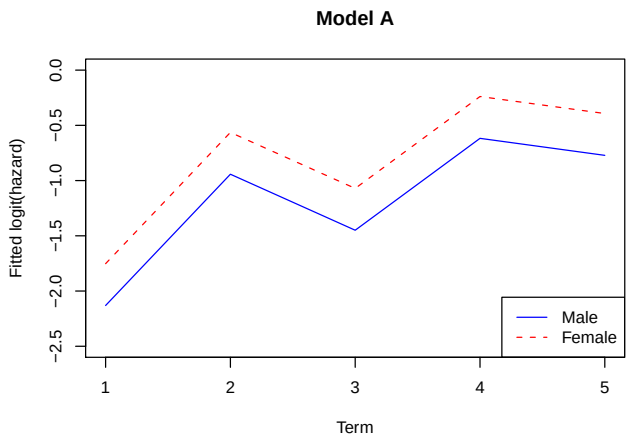
```
> #############
> ## Next, we run the last of three models, Model C,
> ## and go through the same steps as for Model A and Model B.
> #############
> modelC <- glm(EVENT ~ HS11 + HS12 + COLL1 + COLL2 +
+   COLL3 + FEMALE + FLTIME - 1,
+   family=binomial(link = "logit"), data = math)
> modelfitC <- c()
> for (i in 1:5){
+   for (j in 0:1){
+     logitfit <- modelC$coef[i] + modelC$coef[6]*j + modelC$coef[7]*i*j
+     hazard = 1/(1 + exp(-logitfit))
+     z <- c(i, j, logitfit, hazard)
+     modelfitC <- rbind(modelfitC, z)}}
> colnames(modelfitC) <- c("term", "female", "logitfit", "hazard")
> modelfitC <- as.data.frame(modelfitC)
> modelfitC.m <- subset(modelfitC, female==0)
> modelfitC.f <- subset(modelfitC, female==1)
```

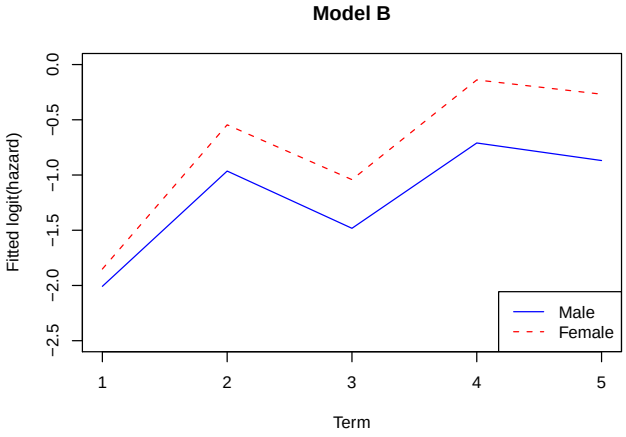
Strategies for Investigating Violations

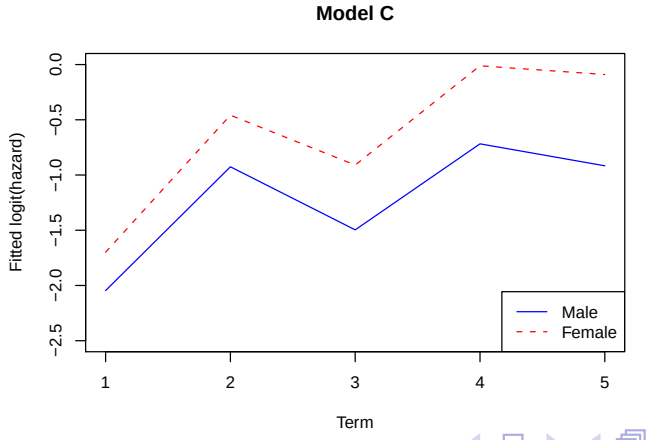
```
> ##########################
> ## Lastly, we create four plots: one for the unfitted hazard
> ## and one for each of the three fitted hazards.
> ## For each graph, we start by plotting one line
> ## for males and then overlay a line for females.
> ##########################
> par(mfrow=c(2,2))
> plot(percents.m$term, percents.m$logit, type = "l",
+  col="blue",lty = 1, ylim = c(-2.5, 0),
+ main = "Within-group sample hazard functions", xlab = "Term", ylab = "Sample logit(hazard)")
> points(percents.f$term, percents.f$logit, type = "l", lty = 2,col="red")
> legend("bottomright", c("Male", "Female"), lty = c(1, 2),col=c("blue","red"))
> plot(modelfitA.m$term, modelfitA.m$logitfit, type = "l", lty = 1,
+ col="blue",ylim = c(-2.5, 0),
+ main = "Model A", xlab = "Term", ylab = "Fitted logit(hazard)")
> points(modelfitA.f$term, modelfitA.f$logitfit, type = "l", lty = 2,col="red")
> legend("bottomright", c("Male", "Female"), lty = c(1, 2),col=c("blue","red"))
> plot(modelfitB.m$term, modelfitB.m$logitfit, type = "l", lty = 1,
+ col="blue",ylim = c(-2.5, 0),
+ main = "Model B", xlab = "Term", ylab = "Fitted logit(hazard)")
> points(modelfitB.f$term, modelfitB.f$logitfit, type = "l", lty = 2,col="red")
> legend("bottomright", c("Male", "Female"), lty = c(1, 2),col=c("blue","red"))
> plot(modelfitC.m$term, modelfitC.m$logitfit, type = "l", lty = 1,
+ col="blue",ylim = c(-2.5, 0),
+ main = "Model C", xlab = "Term", ylab = "Fitted logit(hazard)")
> points(modelfitC.f$term, modelfitC.f$logitfit, type = "l", lty = 2,col="red")
> legend("bottomright", c("Male", "Female"), lty = c(1, 2),col=c("blue","red"))
```

Strategies for Investigating Violations









Strategies for Investigating Violations

```
> summary(modelA)

Call:
glm(formula = EVENT ~ HS11 + HS12 + COLL1 + COLL2 + COLL3 + FEMALE -
    1, family = binomial(link = "logit"), data = math)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.078  -0.811  -0.566  -0.474   2.118

Coefficients:
       Estimate Std. Error z value Pr(>|z|)
HS11    -2.1308     0.0567  -37.56  < 2e-16 ***
HS12    -0.9425     0.0479  -19.68  < 2e-16 ***
COLL1   -1.4495     0.0634  -22.84  < 2e-16 ***
COLL2   -0.6176     0.0757   -8.16  3.4e-16 ***
COLL3   -0.7716     0.1428   -5.40  6.5e-08 ***
FEMALE   0.3786     0.0501    7.55  4.3e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 13250.2  on 9558  degrees of freedom
Residual deviance:  9804.3  on 9552  degrees of freedom
AIC: 9816

Number of Fisher Scoring iterations: 4
```

Strategies for Investigating Violations

```
> summary(modelB)

Call:
glm(formula = EVENT ~ HS11 + HS12 + COLL1 + COLL2 + COLL3 + FHS11 +
    FHS12 + FCOLL1 + FCOLL2 + FCOLL3 - 1, family = binomial(link = "logit"),
    data = math)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.119  -0.804  -0.540  -0.502   2.066

Coefficients:
       Estimate Std. Error z value Pr(>|z|)
HS11    -2.0077     0.0715  -28.09  < 2e-16 ***
HS12    -0.9643     0.0585  -16.47  < 2e-16 ***
COLL1   -1.4824     0.0847  -17.50  < 2e-16 ***
COLL2   -0.7100     0.1007   -7.05  1.8e-12 ***
COLL3   -0.8690     0.1908   -4.56  5.2e-06 ***
FHS11    0.1568     0.0978    1.60  0.10879
FHS12    0.4187     0.0792    5.28  1.3e-07 ***
FCOLL1   0.4407     0.1158    3.81  0.00014 ***
FCOLL2   0.5707     0.1445    3.95  7.9e-05 ***
FCOLL3   0.6008     0.2857    2.10  0.03550 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 13250.2  on 9558  degrees of freedom
Residual deviance:  9796.3  on 9548  degrees of freedom
AIC: 9816

Number of Fisher Scoring iterations: 4
```

Strategies for Investigating Violations

```
> summary(model1C)

Call:
glm(formula = EVENT ~ HS11 + HS12 + COLL1 + COLL2 + COLL3 + FEMALE +
    FLTIME - 1, family = binomial(link = "logit"), data = math)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.122  -0.817  -0.548  -0.493   2.082

Coefficients:
       Estimate Std. Error z value Pr(>|z|)
HS11    -2.0459     0.0646  -31.65  < 2e-16 ***
HS12    -0.9255     0.0482  -19.20  < 2e-16 ***
COLL1   -1.4966     0.0665  -22.51  < 2e-16 ***
COLL2   -0.7178     0.0861   -8.34  < 2e-16 ***
COLL3   -0.9166     0.1557   -5.89  3.9e-09 ***
FEMALE   0.2275     0.0774    2.94   0.0033 **
FLTIME   0.1198     0.0470    2.55   0.0108 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 13250.2  on 9558  degrees of freedom
Residual deviance:  9797.8  on 9551  degrees of freedom
AIC: 9812

Number of Fisher Scoring iterations: 4
```

Strategies for Investigating Violations

```
> anova(modelA,modelB)

Analysis of Deviance Table

Model 1: EVENT ~ HS11 + HS12 + COLL1 + COLL2 + COLL3 + FEMALE - 1
Model 2: EVENT ~ HS11 + HS12 + COLL1 + COLL2 + COLL3 + FHS11 + FHS12 +
    FCOLL1 + FCOLL2 + FCOLL3 - 1
  Resid. Df Resid. Dev Df Deviance
1      9552       9804
2      9548       9796  4     8.04

> anova(modelA,modelC)

Analysis of Deviance Table

Model 1: EVENT ~ HS11 + HS12 + COLL1 + COLL2 + COLL3 + FEMALE - 1
Model 2: EVENT ~ HS11 + HS12 + COLL1 + COLL2 + COLL3 + FEMALE + FLTIME -
    1
  Resid. Df Resid. Dev Df Deviance
1      9552       9804
2      9551       9798  1      6.5

> anova(modelC,modelB)

Analysis of Deviance Table

Model 1: EVENT ~ HS11 + HS12 + COLL1 + COLL2 + COLL3 + FEMALE + FLTIME -
    1
Model 2: EVENT ~ HS11 + HS12 + COLL1 + COLL2 + COLL3 + FHS11 + FHS12 +
    FCOLL1 + FCOLL2 + FCOLL3 - 1
  Resid. Df Resid. Dev Df Deviance
1      9551       9798
2      9548       9796  3     1.54
```

The No Unobserved Heterogeneity Assumption

Just as in other forms of regression modeling, we ultimately must assume that our covariates account for all substantive sources of variation. If they don't, the trends we observe may, for example, be a blended average of the trends in several disparate populations, none of which have trends matching this blended average.

Residual Analysis

Residual analysis in survival analysis is complicated by the fact that the observed value in any time period is either 0 or 1, while the expected value generally lies between 0 and 1.

So if the residual were defined simply as the difference between observed and expected values, then the model would "underpredict" in any period in which the event occurs, and "overpredict" in any period in which it did not occur.

Deviance Residuals

To overcome this problem, Singer and Willett recommend use of *deviance residuals* defined as

$$DEV_{ij} = sign(EVENT_{ij} - \hat{h}_{ij}) \times \sqrt{-2\left[EVENT_{ij}\log(\hat{h}_{ij}) + (1 - EVENT_{ij})\log(1 - \hat{h}_{ij})\right]}$$

Deviance Residuals

In the textbook, S&W examine the residuals for 8 boys in "grade of first intercourse" study. The code to obtain and collect these residuals in order to reproduce Table 12.6 is rather convoluted.

```
> firstsex<-read.table("firstsex_pp.csv", sep=",", header=T)
> model12.6<-glm(event ~ d7 + d8 + d9 + d10 + d11 +
+  d12 + pt+pas- 1, family=binomial(link = "logit"),
+  data = firstsex)
> firstsex.r <- cbind(firstsex, dev.res = residuals(model12.6,
+  type="deviance"))
> get12.6 <- function(id.num){
+   x <- subset(firstsex.r, id==id.num)
+   pt <- max(x$pt)
+   pas <- max(x$pas)
+   grade <- max(x$period)
+   censor <- abs(max(x$event)-1)
+   gr <- x$dev.res
+   gr.l <- length(gr)
+   if (gr.l < 6){
+     gr <- c(gr, c(rep(NA, (6-gr.l))))}
+   ss.dev <- sum(na.omit(gr*gr))
+   z <- c(id.num, pt, pas, grade, censor, gr, ss.dev)
+   return(z)}
> tab12.6 <- rbind(get12.6(22), get12.6(112), get12.6(166),
+ get12.6(89), get12.6(102), get12.6(87), get12.6(67), get12.6(212))
> colnames(tab12.6) <- c("id", "pt", "pas", "grade", "censor",
+ "gr7", "gr8", "gr9", "gr10", "gr11", "gr12", "ss.dev")
```

Deviance Residuals

```
> tab12.6
```

	id	pt	pas	grade	censor	gr7	gr8	gr9	gr10	gr11	gr12	ss.dev
[1,]	22	1	-0.64965	12	0	-0.4117	-0.2944	-0.5840	-0.7176	-0.7748	1.4145	3.713
[2,]	112	1	-0.66093	12	1	-0.4111	-0.2940	-0.5831	-0.7166	-0.7737	-0.9563	2.622
[3,]	166	1	2.78141	11	0	-0.6615	-0.4807	-0.9108	-1.0903	1.1914	NA	4.106
[4,]	89	0	-0.07516	11	0	-0.3248	-0.2314	-0.4645	-0.5752	1.8624	NA	4.174
[5,]	102	1	0.60493	8	0	-0.4913	2.3695	NA	NA	NA	NA	5.856
[6,]	87	1	2.67790	7	0	1.8176	NA	NA	NA	NA	NA	3.304
[7,]	67	1	2.27465	12	0	-0.6180	-0.4477	-0.8559	-1.0294	-1.1007	1.0430	4.674
[8,]	212	0	-0.96179	12	1	-0.2857	-0.2032	-0.4098	-0.5090	-0.5524	-0.6958	1.339

Deviance Residuals

Here is the code for generating plots of the residuals. One plot presents individual residuals, the other aggregates them at the person level.

```
> firstsex<-read.table("firstsex_pp.csv", sep=",", header=T)
> model12.6<-glm(event ~ d7 + d8 + d9 + d10 + d11 +
+  d12 + pt+pas- 1, family=binomial(link = "logit"),
+  data = firstsex)
> firstsex.r <- cbind(firstsex,
+  dev.res = residuals(model12.6, type="deviance"))
> ## Utility function to compute SSresiduals for a given ID.
> get12.9 <- function(id.num){
+   x <- subset(firstsex.r, id==id.num)
+   gr <- x$dev.res
+   ss.dev <- sum((gr*gr))
+   return(ss.dev)}
> ## Use the function to loop through IDs
> ## and collect information
> ss.devs <- c()
>   for (i in 1:216){
+   z <- c(i, get12.9(i))
+   if (z[2]!=0){
+     ss.devs <- rbind(ss.devs, z)}}
> ## Create the two plots
> par(mfrow=c(2,1))
> plot(firstsex.r$id, firstsex.r$dev.res, ylab = "Deviance residual", xlab = "ID")
> plot(ss.devs[,1], ss.devs[,2], ylab = "SS Deviance residual", xlab = "ID")
```

Deviance Residuals