

# Cox regression: Inference

Patrick Breheny

October 31

# Introduction

- Today we will discuss inference for the Cox model; this discussion will be brief, as all of our previous likelihood-based methods apply to the Cox partial likelihood
- We will also formally introduce `coxph`, the function in the `survival` package that fits Cox proportional hazards models
- Finally, we will take a look at how the results from the Cox model applied to several our example data sets compare to the results we obtained from parametric models

# Wald inference

- Just as in the case of parametric inference, Wald-based inference is based off of the asymptotic result

$$\hat{\beta} \sim N(\beta, (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1}),$$

where expressions for the elements of  $\mathbf{W}$  were derived in the previous lecture

- The composition of  $\mathbf{W}$  is of course very different for Cox regression than what we had in the exponential regression case, but all formulas and procedures remain the same
- In particular,  $100(1 - \alpha)\%$  confidence intervals are constructed via  $\hat{\beta}_j \pm z_{(\alpha/2)} \sqrt{(\mathbf{I}^{-1})_{jj}}$

# Likelihood ratio confidence intervals

- As we saw with parametric models, likelihood ratio methods are typically the most accurate of the asymptotic likelihood approaches
- However, they are somewhat cumbersome for the purposes of constructing confidence intervals, as they require profiling
- For this reason, likelihood ratio confidence intervals are rare in practice

# Likelihood ratio tests

- Likelihood ratio tests, however, are common and widely used, especially when comparing nested models that differ with respect to multiple parameters
- For example, in the pbc data, suppose we wished to compare the fit of a linear effect for stage versus the fit allowing separate parameters describing the relative risk of each stage
- Letting  $\hat{\beta}_0$  denote the fit of the first model and  $\hat{\beta}_1$  denote the fit of the second model, the likelihood ratio test (which only requires fitting two models) is based on

$$2\{\ell(\hat{\beta}_1) - \ell(\hat{\beta}_0)\} \sim \chi^2_2;$$

3 parameters for the four stages minus a single parameter assuming linearity = 2 df

# Score tests

- The score, like the likelihood ratio, requires profiling in order to construct confidence intervals and is thus rarely used for this purpose in practice
- Score tests for Cox regression are not particularly common either; however, they do have the advantage, as you hopefully saw in the current assignment, that the significance of adding new terms to a model can be tested without actually fitting any new models
- Nevertheless, there is an interesting connection between the score test in a Cox model and the log-rank test that is worth discussing

## Score and log-rank tests

- Consider the Cox regression score test in the special case with only one covariate, an indicator function
- In that case, the Cox score statistic for testing  $H_0 : \beta = 0$  is

$$\begin{aligned}u(0) &= \sum_j (x_j - \mathbb{E}_j x) \\ &= \sum_j \left( d_{1j} - d_j \frac{n_{1j}}{n_j} \right),\end{aligned}$$

or  $W$  from the log-rank test

- Thus, the Cox regression score test is in some sense equivalent to the log-rank test, although the variances are calculated differently and therefore don't produce the exact same  $p$ -value

## coxph

- The function for fitting Cox proportional hazards models in the `survival` package is called `coxph`
- Broadly speaking, the syntax is similar to other model-fitting functions in R:

```
fit <- coxph(S ~ trt + stage + hepato + bili, pbc)
```

where `S` is a `Surv` object

- Once we have fit the model, there are a number of functions that can be called on the fitted model object; we will go over most of them now, although some are more complex and we will save for a later time (e.g, `residuals(fit)`)



# coef, vcov, and model.matrix

Several functions should be familiar to you from your past experience with modeling functions in R:

- `coef(fit)`: Returns the MLE of the coefficient vector,  $\hat{\beta}$
- `vcov(fit)`: Returns the inverse of the information matrix,  $(\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1}$
- `model.matrix(fit)`: Returns the design matrix,  $\mathbf{X}$ ; this is particularly convenient when things like factors, interactions, and basis expansions are present in the model formula

## summary

```
> summary(fit)
n= 312, number of events= 144
```

	coef	exp(coef)	se(coef)	z	Pr(> z )	
trt	-0.15473	0.85664	0.16813	-0.920	0.357	
stage	0.62138	1.86149	0.12816	4.848	1.24e-06	***
hepato	0.34854	1.41700	0.21269	1.639	0.101	
bili	0.13353	1.14285	0.01392	9.591	< 2e-16	***

	exp(coef)	exp(-coef)	lower .95	upper .95
trt	0.8566	1.1673	0.6162	1.191
stage	1.8615	0.5372	1.4480	2.393
hepato	1.4170	0.7057	0.9340	2.150
bili	1.1429	0.8750	1.1121	1.174

...

## summary

```
> summary(fit)

...

Concordance= 0.797 (se = 0.026 )
Rsquare= 0.348 (max possible= 0.991 )
Likelihood ratio test= 133.3 on 4 df, p=0
Wald test = 154 on 4 df, p=0
Score (logrank) test = 212.1 on 4 df, p=0
```

- We will discuss concordance and  $R^2$  in a future lecture
- The Wald, Score, and LRT tests here are testing the global hypothesis  $H_0 : \beta_1 = \beta_2 = \beta_3 = \beta_4 = 0$

## anova

- Earlier, we proposed the idea of a likelihood ratio test for whether a linear effect for stage was adequate, or whether a three parameter representation would offer a better fit
- This can be carried out using the anova function:

```
> fit0 <- coxph(S ~ trt + hepato + bili + stage, pbc)
> fit1 <- coxph(S ~ trt + hepato + bili +
                factor(stage), pbc)
> anova(fit0, fit1)
Analysis of Deviance Table
Cox model: response is Surv(time, status != 0)
Model 1: ~ trt + stage + hepato + bili
Model 2: ~ trt + factor(stage) + hepato + bili
  loglik  Chisq Df P(>|Chi|)
1 -672.17
2 -671.34 1.6635 2    0.4353
```

# logLik and AIC

- Like many likelihood-based procedures, `coxph` allows you to extract the (partial) log-likelihood using `logLik`:

```
> logLik(fit0)
'log Lik.' -672.1719 (df=4)
> logLik(fit1)
'log Lik.' -671.3401 (df=6)
```

- This, in turn, means that other functions that depend on log-likelihoods, such as AIC, can be called:

```
> AIC(fit0)
[1] 1352.344
> AIC(fit1)
[1] 1354.68
```

## BIC

- Same with BIC:

```
> BIC(fit0)
[1] 1364.223
> BIC(fit1)
[1] 1372.499
```

- However, note that this BIC calculation uses the formula

$$\text{BIC} = -2\ell + \log(d)\text{df},$$

with  $d$ , the number of *events*, replacing  $n$

- This is supported by a paper from by Volinsky & Raftery (2000) showing that this yielded more accurate approximations to the true Bayes factors

# Caveats, Part 1

- Note that a likelihood ratio test only makes sense if the models are nested
- However, `anova()` does not check for this:

```
> anova(fit1, fit2)
Analysis of Deviance Table
Cox model: response is Surv(time, status != 0)
Model 1: ~ trt + stage
Model 2: ~ trt + hepato + bili
  loglik  Chisq Df P(>|Chi|)
1 -710.22
2 -685.25 49.941  1 1.584e-12 ***
```

- Use AIC or BIC instead in this scenario

## Caveats, Part 2

- On the other hand, `anova()` does check to see that the number of observations in the two models is the same (this is a common scenario if any data are missing):

```
fit1 <- coxph(Surv(time, status!=0) ~ trt, pbc)
fit2 <- coxph(Surv(time, status!=0) ~ trt + chol, pbc)
anova(fit1, fit2) # Error: models were not all fitted
                  # to the same size of dataset
```

- You can easily run into big problems with `AIC()`, however:

```
> AIC(fit1) # 1479.575
> AIC(fit2) # 1311.981 -- no errors or warnings!
```

- Better coding practice is `AIC(fit1, fit2)`, which will warn you if the number of observations is different between models



# Predictions

- Like many regression models, `coxph` also provides a `predict` method
- However, this is worth discussing carefully, as Cox regression does not provide true “predictions”
- In particular, the Cox model estimates only the relative risk for each subject compared to an unspecified baseline hazard
- As a consequence, the linear predictors  $\{\eta_i\}$  do not have any absolute meaning, in the sense that one could redefine them according to  $\{\tilde{\eta}_i = \eta_i + C\}$  for any constant  $C$  and the likelihood would remain the same

# Invariance

- This is unappealing because it means that if we code, say, treatment as 0/1, as opposed to -1/1 or 1/2, we will get different predicted values for  $\{\eta_i\}$
- To resolve this difficulty, standard practice is to center  $\mathbf{X}$  prior to fitting so that each column has mean zero
- This does not affect  $\hat{\beta}$  in any way, but it does mean that the linear predictors for Cox regression are invariant to changes of location and scale

# predict

- As a concrete example:

```
> new <- data.frame(trt=0, stage=2, hepato=1, bili=1)
> predict(fit, new)
-0.5416297
```

- This is different from

```
> XX <- as.matrix(new)
> XX %*% coef(fit)
1.724818
```

but the same as

```
> m <- apply(model.matrix(fit), 2, mean)
> (XX-m) %*% coef(fit)
-0.5416297
```

## PBC data

- We'll now carry out some comparisons between the Cox model and some parametric PH models of estimates and confidence intervals for various data sets
- First, the stage coefficient for PBC data:

	$\hat{\beta}$	Lower	Upper
Cox	0.621	0.370	0.873
Weibull	0.625	0.367	0.884
Exponential	0.564	0.317	0.811

- It is reassuring that Cox agrees with Weibull here, both in terms of estimates and width of confidence intervals, given that our diagnostic plots suggested that the Weibull is a reasonable parametric model for this data

## Pike data

- Now for the estimate of pretreatment regimen for the Pike data:

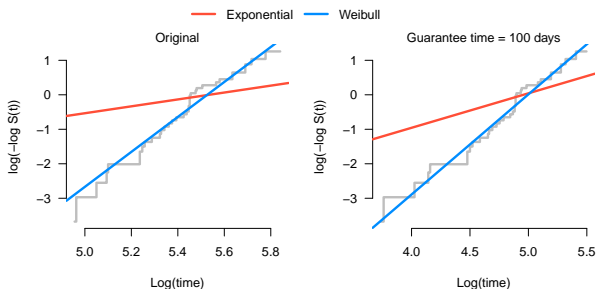
	$\hat{\beta}$	Lower	Upper
Cox	-0.569	-1.249	0.112
Weibull	-0.720	-1.375	-0.065
Exponential	-0.093	-0.747	0.561

## Guarantee time

- Our earlier diagnostic plots suggested that the Weibull was a reasonable model here (certainly much better than the exponential)
- The Weibull may provide a poor fit at earlier times, however, in that no failures occur before day 142
- To account for this, Pike's original analysis modeled time to failure *starting at day 100*
- These first 100 days are known as a *guarantee time*, in the sense that it assumes a guarantee that no rats will die during that time span

# Model comparison

- A comparison of the diagnostic plots:



- Log-likelihood provides an objective indication that the guarantee time model fits (slightly) better:  $\ell_0 = -193.4$ ;  $\ell_{100} = -191.9$

## Pike data; guarantee time = 100 days

Revisiting the Pike data with a guarantee time of 100 days, we find that the Weibull (and exponential) estimates and confidence intervals have moved closer to those of the Cox model

	$\hat{\beta}$	Lower	Upper
Cox	-0.569	-1.249	0.112
Weibull	-0.660	-1.314	-0.005
Exponential	-0.175	-0.830	0.479



## GVHD data

Our final data set for today is the GVHD data:

	$\hat{\beta}$	Lower	Upper
Cox	-1.152	-2.166	-0.138
Weibull	-1.317	-2.417	-0.216
Exponential	-1.529	-2.541	-0.517

Recall that there is no reason to think that either the Weibull or exponential estimates are particularly accurate here

## GVHD data

What about applying an artificial censoring time of 60 days to all subjects still at risk at that time, as we did on assignment 6?

	$\hat{\beta}$	Lower	Upper
Cox	-1.152	-2.166	-0.138
Weibull	-1.255	-2.361	-0.149
Exponential	-1.238	-2.250	-0.226

Again, the Weibull and exponential results move much closer to the Cox results