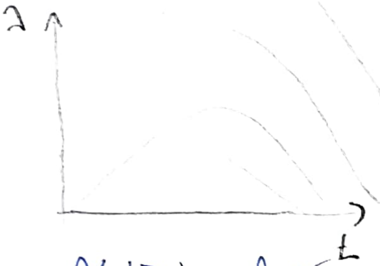
EXERCISE 7 COXMODEL WITH TIMEINVARIANT REGRESSORS

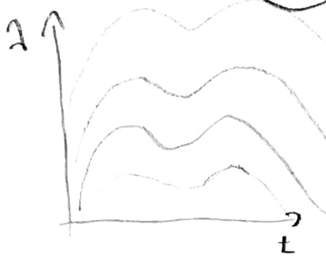
n independent triplets $\begin{pmatrix} X_i = T_i \wedge C_i \\ \delta_i = \mathbb{1}(T_i \le C_i) \\ Z_i \end{pmatrix}_{i=1,\dots,n} \rightsquigarrow \begin{pmatrix} N_i(t)\,, t>0 \\ Y_i(t)\,, t>0 \\ Z_i \end{pmatrix}$ $i=1,\dots,n$

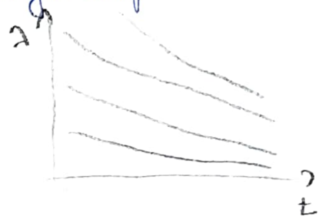
(Ex 1) Exponential regression for censored data $\lambda(\cancel{t}|Z) = \boxed{\lambda_0}\, \exp\{\beta^T Z\}$

→ constants in time

(Ex 7) Cox proportional hazards model $\lambda(t|Z) = \boxed{\lambda_0(t)} \cdot \exp\{\beta^T Z\}$ ← an arbitrary hazard function







$$\frac{\lambda(t|Z_1)}{\lambda(t|Z_2)} = \frac{\cancel{\lambda_0(t)}\cdot \exp\{\beta^T Z_1\}}{\cancel{\lambda_0(t)}\cdot \exp\{\beta^T Z_1\}} = \exp\{\beta^T (Z_1 - Z_2)\}$$ ← constant in time t

baseline hazard $\lambda_0(t) = \lambda(t|Z=0)$

If you shift covariates by some a and use $\lambda(t|Z) = \lambda_0(t)\cdot\exp\{\beta^T(Z-a)\}$

then $\lambda_0(t) = \lambda(t|Z=a)$

(R) predict.coxph uses as default $a = \overline{Z}_n$ - mean of each of the covariate

→ it uses it even for categorical covariates and dummy variables ⇝ baseline may correspond to non-existing subject

example: Z is education ⇝ baseline corresponds to someone who is educated from 30% on basic level, 50% on high sch. level and 20% on unive

But usually we don't care about the meaning of baseline hazard, we simply want to compute S or Λ for specific Z.

Under $\lambda(t|Z) = \lambda_0(t)\cdot\exp\{\beta^T Z\}$ we obtain estimates for individual with $Z = c$ easily as $\lambda_0(t)\cdot e^{\beta^T c}$

Under $\lambda(t|Z) = \lambda_0(t)\cdot \exp\{\beta^T(Z-\overline{Z})\} = \underbrace{\lambda_0(t)\cdot\exp\{-\beta^T \overline{Z}\}}_{\text{new baseline}} \cdot \exp\{\beta^T Z\}$ we need to use $Z = \overline{Z} + c$

to obtain $\lambda_0(t)\, e^{\beta^T c}$

That is why using predict.coxph is problematic - it subtracts $\overline{Z}$ without you actively trying to

⇝ so to obtain prediction for someone of age=50, chol=300, hepato=1

⇝ you should plug (50+mean(age), 300+mean(chol), ~~hepato~~ 1+mean(hepato)) to predict.coxph

fit ← coxph(....)

predict.coxph(fit, newdata= c(50,300,1) + fit$means)

or predict(fit, type="lp", newdata) + sum(coef(fit) * fit$means) to get $\beta^T c$