Functional Response Models in General

Consider functional-input functional-output regression

$$x(t) \rightarrow y(t)$$

So far we have considered the *concurrent linear model*

$$y(t) = \beta(t)x(t) + \epsilon(t)$$

but clearly this is unsatisfactory:

- $y(t)$ may depend on $x(t)$ at times other than the current
- $y(t)$ and $x(t)$ may be measured at different ranges

At Most General

Treat $y(t)$ as a scalar at each time t. The functional linear model is

$$y_t = \int \beta(s)x(s)ds + \epsilon$$

So that over all times t this becomes

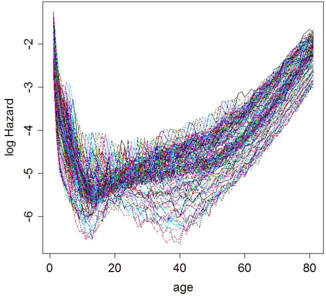
$$y(t) = \int \beta(s,t)x(s)ds + \epsilon(t)$$

As for the scalar response model, this is not identifiable without smoothing.

But we do know how to smooth bivariate functions!

Example: Swedish Lifetable Timeseries

Recall the Swedish mortality data: log hazard = instantaneous chance of death at each age.



Instead of treating time as a co-variate, we will consider a time-series model

$$y_i(t) = \int \beta(s,t) y_{i-1}(s) ds + \epsilon_i(t)$$

Estimating a Coefficient Function

We use an integrated squared error objective criterion:

$$\text{SISE} = \sum \left[\int \left(y_i(t) - \int \beta(s,t) x_i(s) ds \right)^2 dt \right]$$

with the usual bivariate roughness penalty.

Representing this by a bivariate basis expansion

$$\begin{aligned} \text{SISE} &= \sum \left[\int \left(y_i(t) - \psi(t) B \int \phi(s) x_i(s) ds \right)^2 dt \right] \\ &= \sum \left[\int \left(y_i(t) - \int \phi(s) x_i(s) ds \otimes \psi(t) \text{vec}(B) \right)^2 dt \right] \end{aligned}$$

Note $\text{vec}(B)$ vectorizes B *column-wise*.

Estimating B

The minimizer of SISE is given by

$$\left[\sum \left[\int \phi(s) x_i(s) ds \right] \left[\int \phi(s) x_i(s) ds \right]^T \otimes \int \Psi(t) \Psi(t)^T dt \right]^{-1} \left[\sum \int \phi(s) x_i(s) ds \otimes \int \Psi(t) y_i(t) dt \right]$$

Note separation into inner-products of basis defined w.r.t s and w.r.t. t.

Usual penalties result in additional penalty matrix inside the inverse.

Use of inprod to Define Functional Regression

```
intbasis = eval.penalty(tbasis,0)

xphi = inprod(sbasis,xfd)
sxphi = apply(xphi,1,sum)

ypsi = inprod(tbasis,yfd)
sypsi = apply(ypsi,1,sum)

yxphi = (xphi%x%matrix(1,23,1))*(matrix(1,23,1)%x%ypsi)
yxphi = apply(yxphi,1,sum)

X = xphi%*%t(xphi)
```

Penalties

As for bivariate smoothing, we have

$$P_{\lambda_s,\lambda_t}(x(s,t)) = \lambda_1 \int [L_s x(s,t)]^2 \, dsdt + \lambda_2 \int [L_t x(s,t)]^2 \, dsdt$$

```
insbasis = eval.penalty(sbasis,0)
inLsbasis = eval.penalty(sbasis,2)
inLtbasis = eval.penalty(tbasis,2)

Rs = intbasis%x%inLsbasis
Rt = inLtbasis%x%insbasis
```

Note that

```
eval.penalty(sbasis,0) = inprod(sbasis,sbasis)
```

With an Intercept

For simplicity, we have not considered an intercept.

In this context we have

$$y_i(t) = \beta_0(t) + \int \beta_1(s,t) x_i(s) ds + \epsilon_i(t)$$

and we can estimate co-efficients for all terms by $(X+R)^{-1}Y$ for

$$X = \begin{bmatrix} \int \Phi(t)\Phi(t)^T dt & \sum \left[\int \phi(s)x_i(s)ds\right]^T \otimes \int \Phi(t)\Phi(t)^T dt \\ \left[\int \phi(s)x_i(s)ds\right] \otimes \int \Phi(t)\Phi(t)^T dt & \left[\int \phi(s)x_i(s)ds\right]\left[\int \phi(s)x_i(s)ds\right]^T \otimes \int \Psi(t)\Psi(t)^T dt \end{bmatrix}$$

and

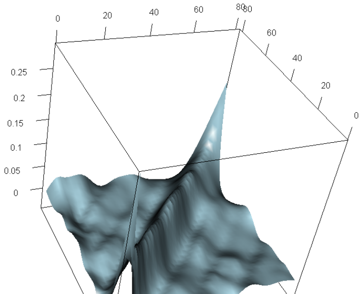
$$Y = \begin{bmatrix} \int \Phi(t) y_i(t) dt \\ \sum \int \phi(s)x_i(s)ds \otimes \int \Psi(t) y_i(t) dt \end{bmatrix}$$

Putting It All Together

```
lambda0 = 1e-5
lambdas = 1e3
lambdat = 1e3

R0 = inLtbasis

penmat = rbind( cbind(lambda0*R0, matrix(0,23,23^2)),
                cbind(matrix(0,23^2,23),lambdas*Rs+lambdat*Rt) )

Xmat = rbind( cbind( intbasis,          t(sxphi)%x%intbasis ),
              cbind( sxphi%x%intbasis, X%x%intbasis) )

ymat = c(sypsi,yxphi)
```

Obtaining an Estimate

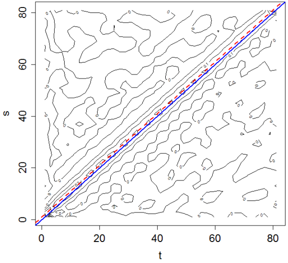
```
betacoefs = solve(Xmat+penmat,ymat)

beta0coefs = betacoefs[1:ntbasis]
beta0fd = fd(beta0coefs,tbasis)

beta1coefs = betacoefs[ntbasis+(1:(ntbasis*nsbasis)]
beta1Cmat = matrix(beta1coefs,ntbasis,nsbasis)
beta1fd = bifd(beta1Cmat,tbasis,sbasis)
```



Interpretation

Ridge in middle is not exactly diagonal (would imply *concurrent* model)

First off-diagonal $\Rightarrow$ events get passed one-year earlier.



Essentially, hazard-events happen to each cohort *at the same time*.

Confidence Intervals

As we had for the concurrent linear model, define B in terms of coefficients of $y(t)$.

$$\mathbf{y}(t) = \xi(t)^T C$$

This gives us

$$\hat{B} = X^{-1} \begin{bmatrix} \int \phi(t)\xi(t)^T dt & \cdots & \int \phi(t)\xi(t)^T dt \\ \int x_1(s)\Phi(s)ds \otimes \int \phi(t)\xi(t)^T dt & \cdots & \int x_n(s)\Phi(s)ds \otimes \int \phi(t)\xi(t)^T \end{bmatrix} \begin{bmatrix} \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_n \end{bmatrix}$$

or

$$\hat{B} = \text{c2bmap} \circ \text{vec}(C)$$

Confidence Intervals

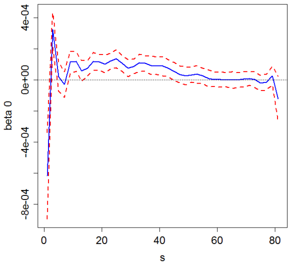
$$\mathrm{var}(\hat{B}) = \mathrm{c2bmap} \circ \begin{bmatrix} \Sigma & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \Sigma \end{bmatrix} \circ \mathrm{c2bmap}^T$$

```
beta1Cmat = matrix(betacoefs[ntbasis+(1:(tbasis*nbasis))],ntbasis,nsbasis)

yhatfd = fd(betacoefs[1:ntbasis]%x%matrix(1,1,143),tbasis) +
    fd( beta1Cmat%*%xphi,tbasis)

errvals = eval.fd(seq(1,81,by=0.5),yfd-yhatfd)
err = smooth.basis(seq(1,81,by=0.5),errvals,bbasis)$fd

Evar = var(t(err$coefs))

inbetay = inprod(betabasis,bbasis)
Ymat = rbind(matrix(1,1,143)%x%inbetay, xphi%x%inbetay)

c2bmap = solve(Xmat+penmat,Ymat)

Cvar = 0

for(i in 1:143){
  ind = bbasis$nbasis*(i-1) + 1:bbasis$nbasis
  Cvar = Cvar + c2bmap[,ind]%*%Evar%*%t(c2bmap[,ind])
}
```

Confidence Intervals $\beta_0(t)$

Based on the first entries in $\hat{B}$

```
tbvals = eval.basis(tfine,tbasis)
beta0bvals = cbind(tbvals,matrix(0,ntfine,nsbasis*ntbasis))
beta0std = sqrt(diag(beta0bvals%*%Cvar%*%t(beta0bvals)))
```



Confidence Intervals $\beta_1(t)$

We need to extract the latter entries of $\hat{B}$.

We also need to remember the order of the kroenecker product basis.

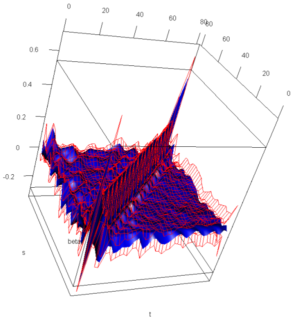
```
sbvals = eval.basis(sfine,sbasis)
beta1bvals = cbind(matrix(0,nsfine*ntfine,ntbasis),
                        sbvals%x%tbvals)
beta1var = diag(beta1bvals%*%Cvar%*%t(beta1bvals)
beta1std = matrix(sqrt(beta1var)),ntfine,nsfine,byrow=TRUE)
```

$$\Phi(s) \otimes \Psi(t) \to x(t,s)$$

remember that rows will be plotted *horizontally*

Some Plotting Tools

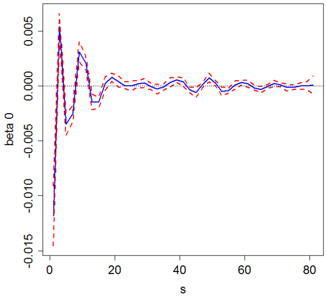
```
beta1vals = eval.bifd(tfine,tfine,beta1fd)

up = beta1vals + 2*beta1std
down = beta1vals - 2*beta1std

persp3d(tfine,tfine,beta1vals)
for(i in 1:nsfine){
 lines3d(tfine,rep(sfine[i],ntfine),up[,i])
 lines3d(tfine,rep(sfine[i],ntfine),down[,i])
}
for(i in 1:ntfine)
 lines3d(rep(tfine[i],nsfine),sfine,up[i,])
 lines3d(rep(tfine[i],nsfine),sfine,down[i,])
}
```

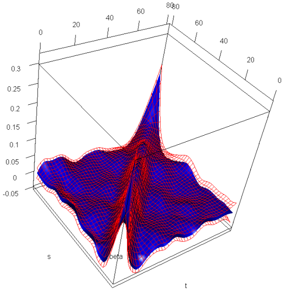


Effects of Smoothing Parameters

Making one λ larger can reduce confidence intervals for other components.

Set $\lambda_s = 10^3$, $\lambda_t = 10^3$:

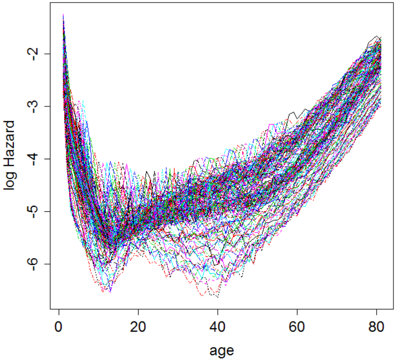


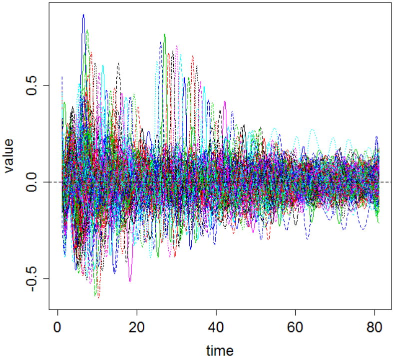


Cross-validation possible, but now three smoothing parameters.

Exploring Residuals

First, no clear trends along age





Yet More Models

fAR(p) Processes:

$$y_i(t) = \beta_0(t) + \sum_{j=1}^{k} \int \beta_j(s,t) y_{i-j}(t) dt + \epsilon_i(t)$$

fARMA(p,q) Processes:

$$y_i(t) = \beta_0(t) + \sum_{j=1}^{k} \int \beta_j(s,t) y_{i-j}(t) dt + \epsilon_i(t) + \sum_{k=1}^{q} \int \theta_k(s,t) \epsilon_{i-k}(t) dt$$

These are models on a *mixed continuous-discrete domain*.

Bivariate continuous analogue:

$$\frac{d}{dw} y(t,w) = \beta_0(t) + \int \beta(s,t) y(s,w) ds + \epsilon(s,w)$$

essentially a partial differential equation: dream up your own.

Some Useful Restrictions

Historical Linear Model: frequently $y_i(t)$ should only depend on $x_i(t)$ at times *before* t:

$$y_i(t) = \beta_0(t) + \int_0^t \beta_1(s,t)x_i(s)ds + \epsilon_i(t)$$

sets $\beta_1(s,t) = 0$ for $s > t$. Requires triangular bases.

Functional Convolution Model: Also restrict dependence to a short time window.

$$y_i(t) = \beta_0(t) + \int_{t-\delta}^t \beta_1(s,t)x_i(s)ds + \epsilon_i(t)$$

- Can be implemented with a kronecker product basis.
- Frequently, set $\beta_0(t) = 0$

Summary

- Most general functional response linear model $\Rightarrow$ bivariate coefficient function.
- Note some pathological cases: $y_i(t) = \beta(t)x_i(T) + \epsilon_i(t)$ for some fixed number T.
- Smooth cases efficiently computed with `inprod` and `eval.basis`
- Confidence intervals follow from concurrent linear model
- Direct extensions to time-series
- Restricted models can also be useful, but harder to code.