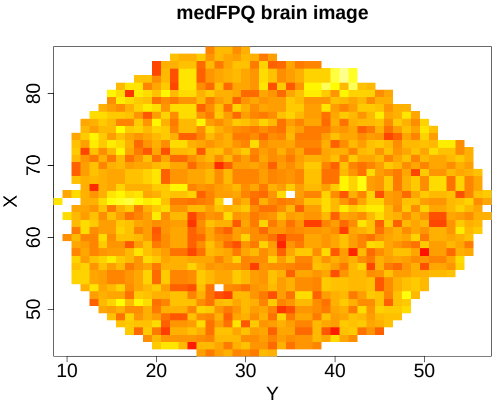
Generalized Additive Smooth Modelling

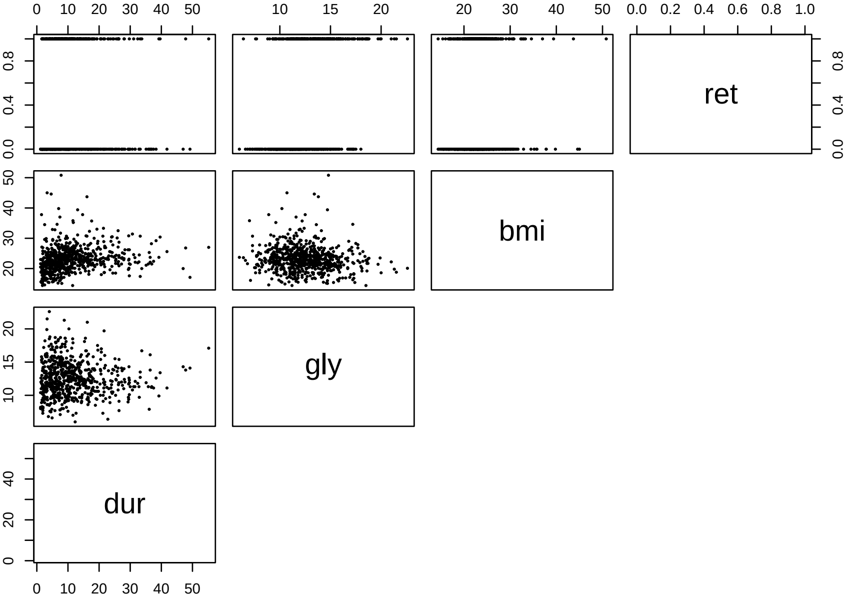
Simon Wood
Mathematical Sciences, University of Bath, U.K.

Example: brain scan data



- Objective: just de-noise the image.
- Model: $\mathbb{E}(\texttt{FPQ}) = f(X, Y)$, f smooth, $\texttt{FPQ} \sim$ Gamma.

Example: retinopathy data



Retinopathy models?

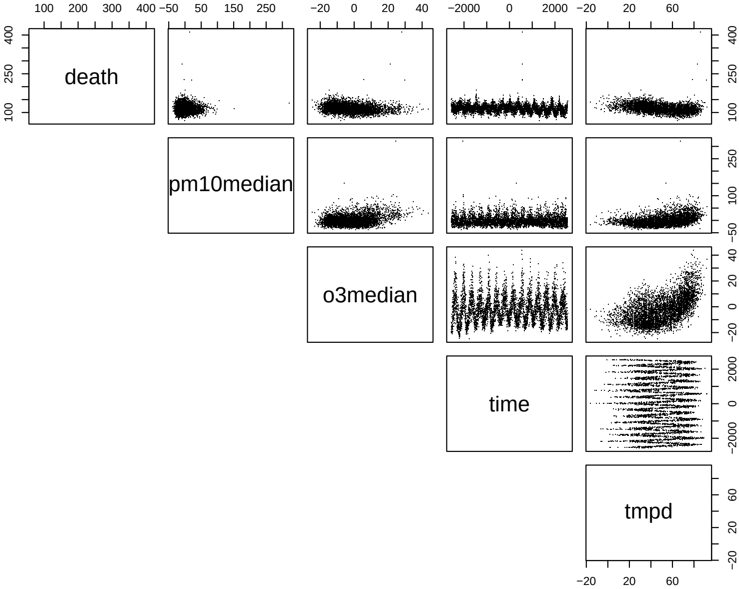
- Question: How is development of *ret*inopathy in diabetics related to *dur*ation of disease at baseline, body mass index (*bmi*) and percentage glycosylated haemoglobin (*gly*)?
- Possible model:

$$\text{logit}\{\mathbb{E}(\texttt{ret})\} = f_1(\texttt{dur}) + f_2(\texttt{bmi}) + f_3(\texttt{gly})$$

  where $\texttt{ret} \sim$ Bernoulli.
- Or would this be better?...

$$\text{logit}\{\mathbb{E}(\texttt{ret})\} = f_1(\texttt{dur}, \texttt{bmi}) + f_2(\texttt{dur}, \texttt{gly}) + f_3(\texttt{gly}, \texttt{bmi})$$

Example: Death & air quality in Chicago



Death & air quality models

- Question: how is air pollution related to daily death rate?
- Model? `death` $\sim$ Poisson

$$\log\{\mathbb{E}(\texttt{death})\} = f_t(\texttt{time}) + f_h(\texttt{tmpd}) + f_z(\texttt{o3}) + f_p(\texttt{pm10})$$

- Or similar with pollutants and temperature smoothly interacting?

Generalized Additive Models

- Examples are all generalized linear models (GLM) in which linear predictor (η, say) is specified (partly) in terms of a sum of smooth functions of predictors.
- Such models are *generalized additive models*.
- Form is something like:

$$g\{\mathbb{E}(y_i)\} = \eta_i = \mathbf{X}_i^* \boldsymbol{\beta}^* + f_1(x_{1i}) + f_2(x_{2i}, x_{3i}) + f_3(x_{4i}) + \cdots$$

  - g is a known *link function*.
  - y_i independent with some exponential family distribution. Crucially this $\Rightarrow \text{var}(y_i) = V(\mathbb{E}(y_i))\phi$, where V is a distribution dependent known function.
  - f_j are smooth unknown functions (subject to centering conditions).
  - $\mathbf{X}^* \boldsymbol{\beta}^*$ is parametric bit.

Making GAM practical

- How should the smooth functions f_j be represented and estimated?
  - Spline type basis expansions (but keep rank low to avoid computational cost).
- How should smoothness of the functions be controlled?
  - By penalizing wiggly functions in the GAM fitting.
- How should smoothness be selected?
  - Prediction error criteria like GCV, AIC etc.
- How should inference be done now?
  - Use wiggliness penalties to define priors in a Bayesian way.

Basis - penalty approach

- Represent the smooths with *basis expansions*; measure function wiggliness using quadratic penalties.
- If the $b_{jk}(\cdot)$ are known basis functions then

$$
\begin{aligned}
\eta_i &= \mathbf{X}_i^* \boldsymbol{\beta}^* + f_1(x_{1i}) + f_2(x_{2i}, x_{3i}) + f_3(x_{4i}) + \cdots \\
&= \mathbf{X}_i^* \boldsymbol{\beta}^* + \sum_k \beta_{1k} b_{1k}(x_{1i}) + \sum_k \beta_{2k} b_{2k}(x_{2i}, x_{3i}) + \cdots \\
&= \mathbf{X}_i \boldsymbol{\beta}
\end{aligned}
$$

- Function wiggliness is measured by quadratic penalties, $\boldsymbol{\beta}_j^{\mathrm{T}} \mathbf{S}_j \boldsymbol{\beta}_j$, typically evaluating things like $\int f_1''(x)^2 dx$. [$\mathbf{S}_j$ is a matrix of coefficients.]

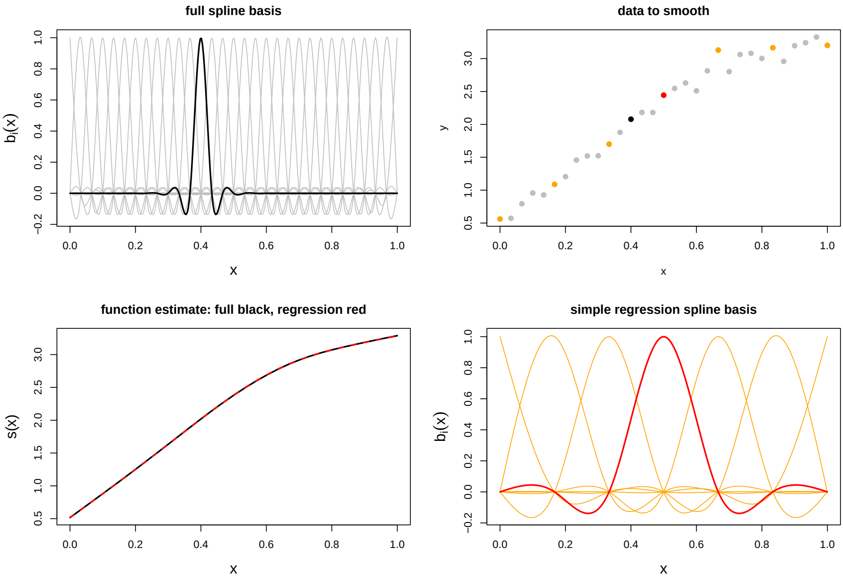
Simple 1D basis

- Usually the bases and penalties for the smooths are based on spline smoothing. (Good approximation properties.)
- i.e. we use bases that arise naturally from solving smoothing problems like: *find the f minimizing*

$$\sum_{i=1}^{n} \{y_i - f(x_i)\}^2 + \lambda \int f''(x)^2 dx$$

- However, such bases are of rank n. To avoid excessive computational cost, basis and penalty are produced for a representative subset of data (but used for whole data set).

Knot selection regression splines



GAM estimation

- Basis dimensions are chosen to be large enough to be sure not to over smooth.
- Consequently if estimation is by MLE then we will under smooth.
- So estimation is by *penalized* MLE. i.e. seek $\boldsymbol{\beta}$ to minimise

  $$D(\boldsymbol{\beta}) + \sum \lambda_j \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S}_j \boldsymbol{\beta}$$

  Where $D = 2[(\text{Max possible log.lik}) - (\text{log.lik for GAM})]$, the *deviance*.
- The *smoothing parameters*, $\boldsymbol{\lambda}$ control the fit vs smoothness tradeoff.

Computing the fit

- Given $\boldsymbol{\lambda}$, $\hat{\boldsymbol{\beta}}$ computed by Penalized Iteratively Re-weighted Least Squares (P-IRLS).
- From guess at $\mu_i = \mathbb{E}(y_i)$, iterate following to convergence
  1. Evaluate pseudodata $z_i = g'(\mu_i)(y_i - \mu_i) + \eta_i$ and iterative weights $W_{ii} = g'(\mu_i)^{-1} V(\mu_i)^{-0.5}$
  2. Minimize
     $$\|\mathbf{W}(\mathbf{z} - \mathbf{X}\boldsymbol{\beta})\|^2 + \sum \lambda_j \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S}_j \boldsymbol{\beta}$$
     w.r.t. $\boldsymbol{\beta}$ to get next $\hat{\boldsymbol{\beta}}$ and hence $\boldsymbol{\eta}$ and $\boldsymbol{\mu}$ estimates.
- Effective degrees of freedom of fit $=$ tr($\mathbf{F}$) where $\mathbf{F} = (\mathbf{X}^{\mathrm{T}}\mathbf{W}^2\mathbf{X} + \sum \lambda_j \mathbf{S}_j)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{W}^2\mathbf{X}$ (at convergence).

Inference

- Fact that $\mathbb{E}(\hat{\boldsymbol{\beta}}) \neq \boldsymbol{\beta}$ makes frequentist approaches poor, except for some H_0 tests.
- Instead put prior on function wiggliness

$$\propto \exp\left(-\frac{1}{2}\sum \lambda_j \boldsymbol{\beta}^{\mathrm{T}}\mathbf{S}_j\boldsymbol{\beta}\right)$$

  — an improper Gaussian on $\boldsymbol{\beta}$.

- Bayes' rule and some asymptotics then $\Rightarrow$

$$\boldsymbol{\beta}|\mathbf{y} \sim N(\hat{\boldsymbol{\beta}}, (\mathbf{X}^{\mathrm{T}}\mathbf{W}^2\mathbf{X} + \sum \lambda_j \mathbf{S}_j)^{-1}\phi)$$

- Posterior $\Rightarrow$ e.g. CIs for f_j, but can also simulate from posterior very cheaply, to make inferences about anything the GAM predicts.

Smoothness selection

- Choose $\boldsymbol{\lambda}$ to try and minimize prediction error (e.g. predictive deviance).
- In known ϕ case use scaled generalized AIC

$$D + 2\mathrm{tr}(\mathbf{F})$$

- Otherwise a generalized GCV

$$\frac{nD}{[n - \mathrm{tr}(\mathbf{F})]^2}$$

- GCV or AIC optimization by Newton method — challenging to get required derivatives in a quick and stable manner.

GAM issues

- Preceding framework quite general, but presents 2 problems.
- How should smooth functions of several variables best be represented while keeping the computations feasible?
- How can we compute in a stable and efficient manner with integrated smoothness selection?

Thin plate splines

- Thin plate splines are functions minimizing (isotropic) objectives like

$$\sum\{y_i - f(x_i)\}^2 \quad + \quad \lambda \int f_{xx}^2 dx$$

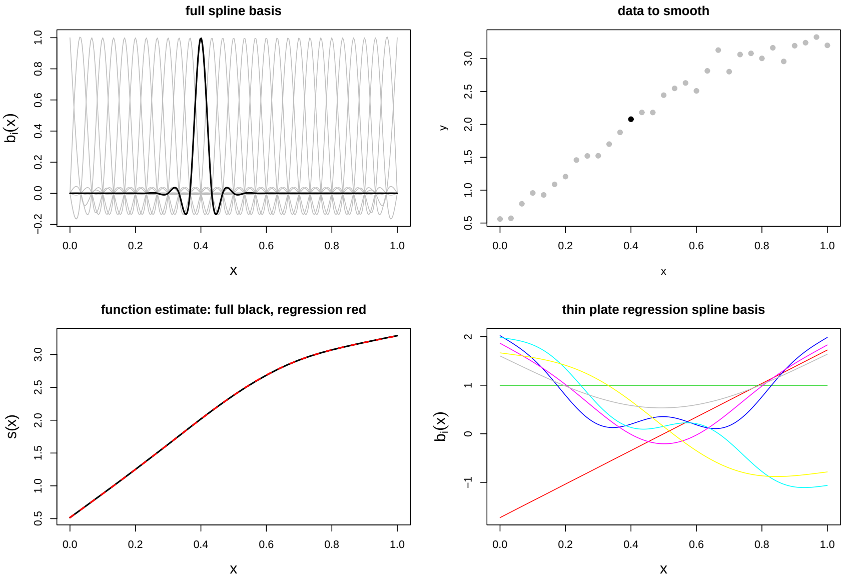
$$\sum\{y_i - f(x_i, z_i)\}^2 \quad + \quad \lambda \int f_{xx}^2 + 2f_{xz}^2 + f_{zz}^2 dxdz$$

- Solution is of form $\hat{f}(\cdot) = \sum_{i=1}^{n} \delta_i \eta_i(\cdot) + \sum_{j=1}^{M} \alpha_j \phi_j(\cdot)$ where
  - $\boldsymbol{\alpha}$ and $\boldsymbol{\delta}$ minimize $\|\mathbf{y} - \mathbf{E}\boldsymbol{\delta} - \mathbf{T}\boldsymbol{\alpha}\|^2 + \lambda \boldsymbol{\delta}^{\mathrm{T}}\mathbf{E}\boldsymbol{\delta}$ subject to $\mathbf{T}^{\mathrm{T}}\boldsymbol{\delta} = \mathbf{0}$.
  - η_i and ϕ_j are known and give **E** and **T**.

Thin plate regression splines

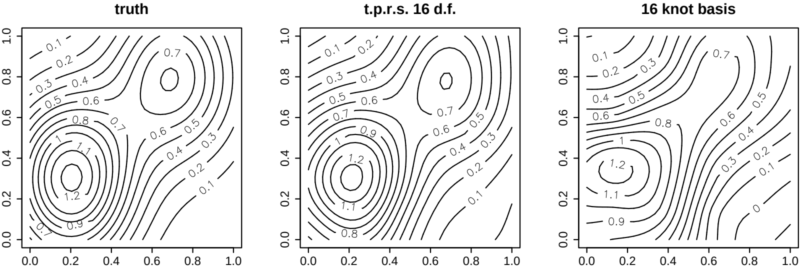
- Thin plate splines are computationally expensive [$O(n^3)$].
- By replacing **E** by its rank k truncated eigen-decomposition we can get an ‘optimal’ (Wood, 2003, JRSSB) rank k approximation to a thin plate spline that is much cheaper to work with.
- Lanczos iteration gives this truncated decomposition relatively cheaply.
- General method for efficient isotropic smoothing. No need to choose ‘knots’!

TPRS 1D example



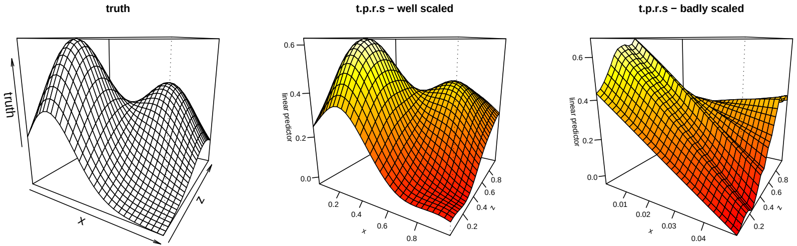
Multi-D smooths: isotropic

- TPRS also provide the optimal low rank basis for multi-dimensional isotropic smoothing. Here is a comparison with a knot based alternative basis.



TPRS limitations

- TPRS work well when isotropic smoothness is appropriate.
- But thin plate (regression) splines are sensitive to arbitrary independent linear rescaling of covariates.



truth

t.p.r.s − well scaled

t.p.r.s − badly scaled

Tensor product smooths

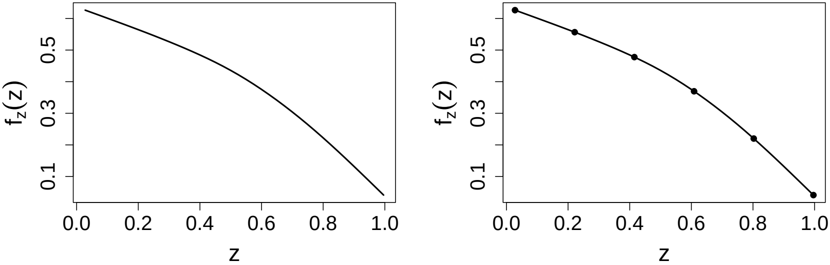
- Carefully constructed tensor product smooths are scale invariant.
- Consider constructing a smooth of x, z.
- Start by choosing *marginal* bases and penalties, as if constructing 1-D smooths of x and z. e.g.

$$f_x(x) = \sum \alpha_i a_i(x), \quad f_z(z) = \sum \beta_j b_j(z),$$

$$J_x(f_x) = \int f_x''(x)^2 dx = \boldsymbol{\alpha}^{\mathrm{T}} \mathbf{S}_x \boldsymbol{\alpha} \;\; \& \;\; J_z(f_z) = \mathcal{B}^{\mathrm{T}} \mathbf{S}_z \mathcal{B}$$

Marginal reparameterization

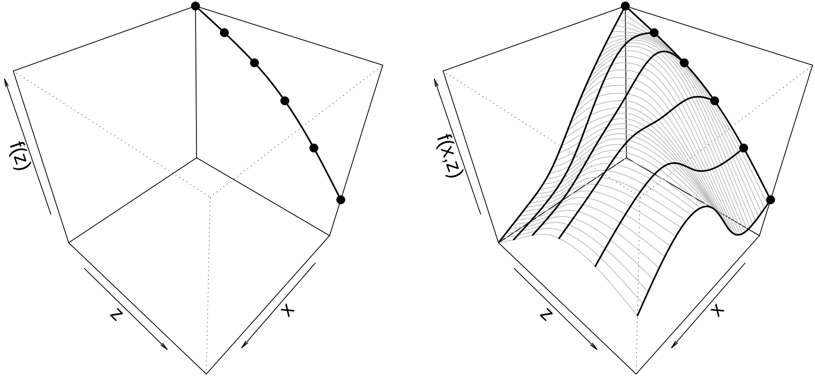
- Suppose we start with $f_z(z) = \sum_{i=1}^{6} \beta_j b_j(z)$, on the left.



- We can always re-parameterize so that its coefficients are functions heights, at knots (right). Do same for f_x.

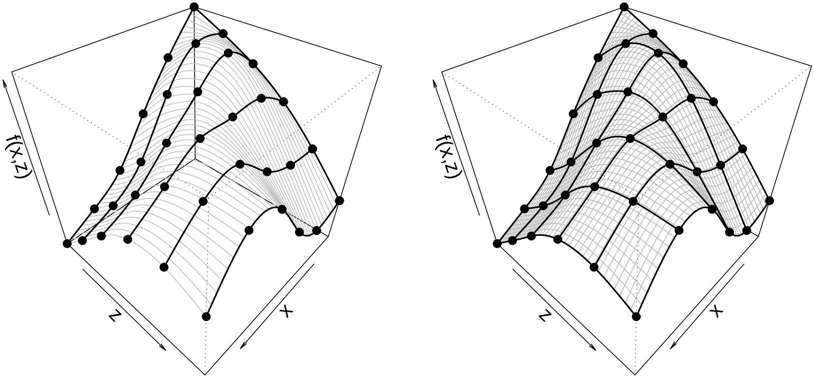
Making f_z depend on x

- Can make f_z a function of x by letting its coefficients vary smoothly with x



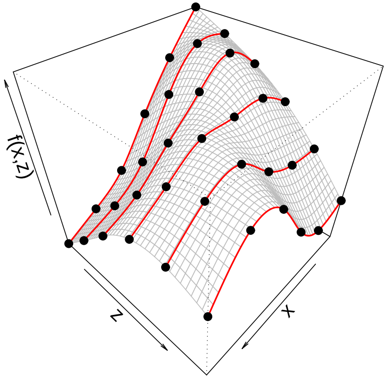
The complete tensor product smooth

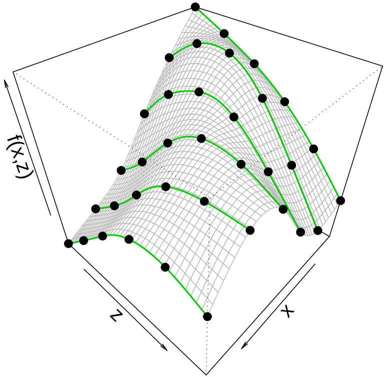
- Use f_x basis to let f_z coefficients vary smoothly (left).
- Construct in symmetric (see right).



Tensor product penalties - one per dimension

- *x*-wiggliness: sum marginal *x* penalties over red curves.
- *z*-wiggliness: sum marginal *z* penalties over green curves.





Tensor product expressions

- So the tensor product basis construction gives:

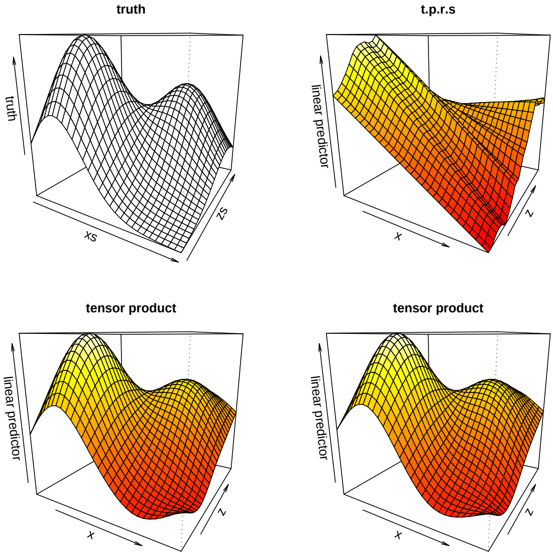
$$f(x,z) = \sum \sum \beta_{ij} b_j(z) a_i(x)$$

- With double penalties

$$J_z^*(f) = \boldsymbol{\beta}^{\mathrm{T}} \mathbf{I}_I \otimes \mathbf{S}_z \boldsymbol{\beta} \text{ and } J_x^*(f) = \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S}_x \otimes \mathbf{I}_J \boldsymbol{\beta}$$

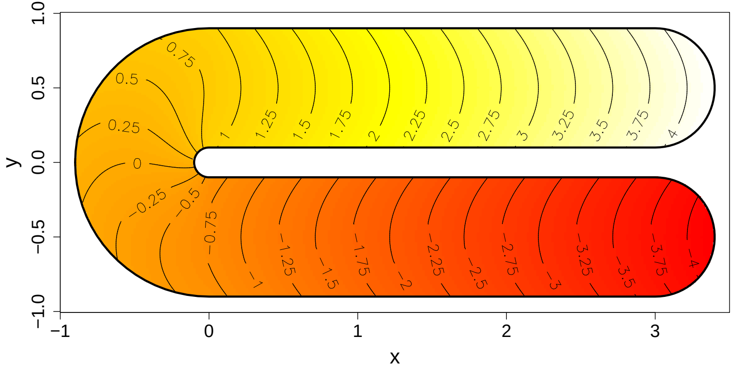
- The construction generalizes to any number of marginals and multi-dimensional marginals.
- Can start from any marginal bases & penalties (including mixtures of types).
- Note that the penalties maintain the basic meaning inherited from the marginals.

It works!


truth
truth
xs
zs
t.p.r.s
linear predictor
x
z
tensor product
linear predictor
x
z
tensor product
linear predictor
x
z

Smooths for difficult domains

- Conventional smooths do very badly at reconstructing this function…



Soap film smooths

- Consider estimating a function $f(x,z)$ from y,x,z data, over an awkward x,z domain, Ω, with boundary B.
- Can construct a smoother based on the notion of a soap film, matching f on B, and smoothly distorting towards the y values.
- This soap film smoother is function, f, minimizing

$$\sum_{i=1}^{n}\{y_i - f(x_i, z_i)\}^2 + \lambda \int_\Omega (f_{xx} + f_{zz})^2 dxdz$$

Subject to a known, or unknown but parameterized, boundary condition.

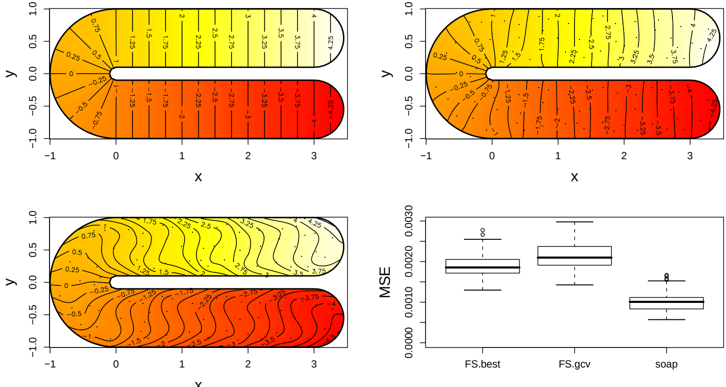
Soap film smooth construction

- The solution to the soap film smoothing problem can be shown to satisfy the PDEs

$$f_{xx} + f_{zz} = \rho(x, z) \ \text{and} \ \rho_{xx} + \rho_{yy} = 0$$

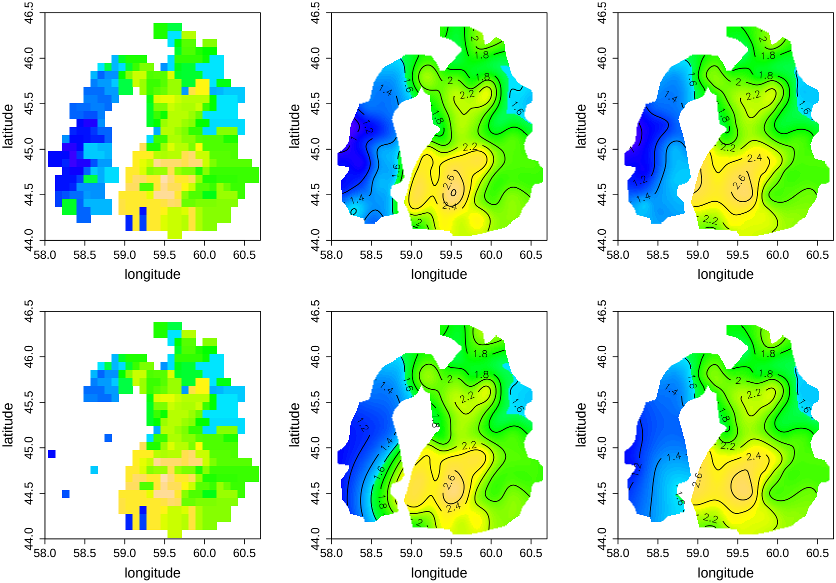
  except for ρ discontinuities at observation points x_i, z_i.
- This result allows straightforward numerical evaluation of a basis for the soap film smoother, as well as representation of the soap film penalty as a quadratic form in the basis coefficients.
- In practice the boundary of f is parameterized using a cyclic penalized regression spline.

Soap films work!



- Comparison with Ramsay’s (2002, JRSSB) approach. Soap does better, and is easier to compute and incorporate into a GAM or other model.

Aral sea chlorophyll



Efficiency and stability issues

- Now we have a toolbox of smoothers for constructing all sorts of wonderful models, but there is a problem.
- Models with this degree of flexibility can exhibit numerical stability problems if not handled carefully.
- Smoothness selection is not easy to achieve efficiently and demands very good numerical stability.
- As an illustration, consider a very simple case.

Simple additive model

- Consider data that can be modelled as

$$\mathbb{E}(y) = f_1(x, z) + f_2(v)$$

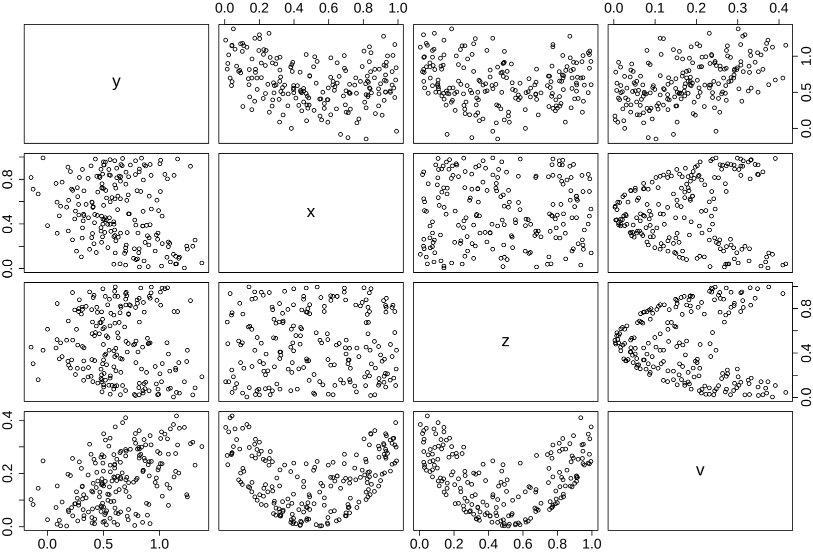
  where y is independent Gaussian.
- In fact v is a smooth function of x and z plus some additive noise. Such *Concurvity* is very common in practice.
- Given a basis expansion and penalties, the coefficient estimates for this model are

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \mathbf{S}_\lambda)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y} \text{ where } \mathbf{S}_\lambda = \sum \lambda_j \mathbf{S}_j$$

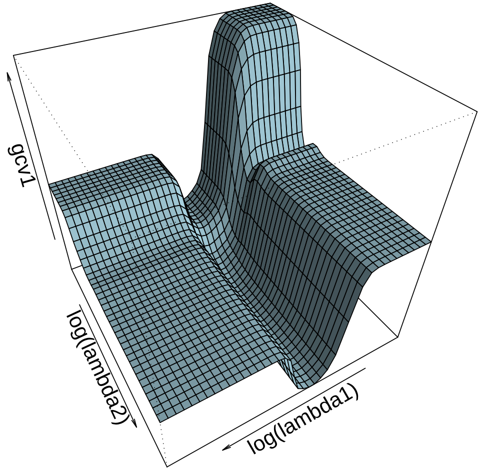
- If $\mathbf{F} = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \mathbf{S}_\lambda)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{X}$, the GCV score for $\boldsymbol{\lambda}$ selection is

$$\mathcal{V}(\boldsymbol{\lambda}) = \frac{n\|\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\|^2}{[n - \mathrm{tr}(\mathbf{F})]^2}$$

Example data with concurvity problem

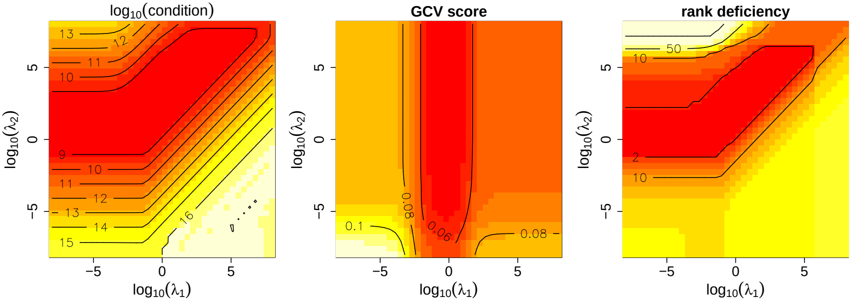


$y = f_1(x, z) + f_2(v)$ GCV score



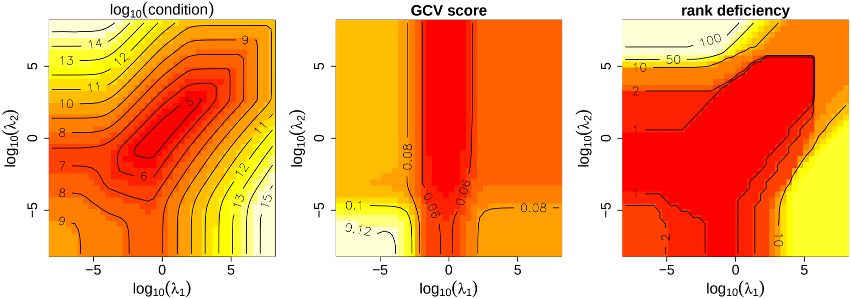
$y = f_1(x, z) + f_2(v)$ with TPS ‘selected knots’ basis

- Used knot based thin plate spline bases of rank 100 & 40.
- The condition number of $(\mathbf{X}^{\mathrm{T}}\mathbf{X} + \mathbf{S}_\lambda)$ was plotted vs $\boldsymbol{\lambda}$.
- Condition number is the ratio of largest to smallest singular values. Small is good. 10^{16} is disaster in double precision.
- Loss of rank (single precision) was also plotted.



$y = f_1(x, z) + f_2(v)$ with TPRS eigen-basis

- Eigen based TPRS bases of rank 100 & 40, were substituted. These bases are also optimally stable.



- Matters are much improved, but still not perfect.

Stable computation

- So, as in linear regression, the 'normal equations'

  $$\hat{\beta} = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \mathbf{S}_\lambda)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y} \text{ where } \mathbf{S}_\lambda = \sum \lambda_j \mathbf{S}_j$$

  do not have good stability.
- Instead note that, if $\mathbf{E}^{\mathrm{T}}\mathbf{E} = \mathbf{S}_\lambda$:

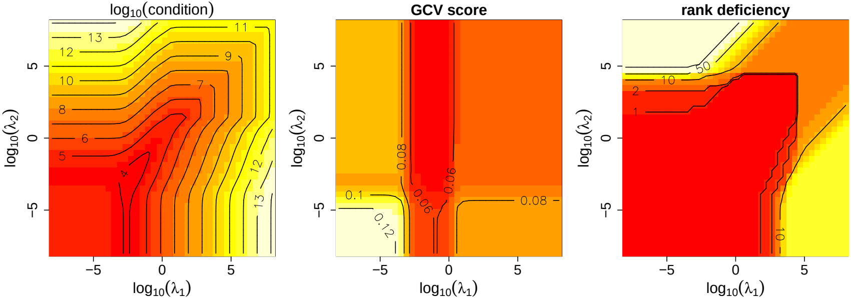
  $$\|\mathbf{y} - \mathbf{X}\beta\|^2 + \beta^{\mathrm{T}}\mathbf{S}_\lambda\beta = \left\| \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix} - \begin{bmatrix} \mathbf{X} \\ \mathbf{E} \end{bmatrix} \beta \right\|^2 .$$

- Using the usual QR method on the right hand side form

  $$\mathbf{QR} = \begin{bmatrix} \mathbf{X} \\ \mathbf{E} \end{bmatrix} \quad \Rightarrow \quad \hat{\beta} = \mathbf{R}^{-1}\mathbf{Q}^{\mathrm{T}}_{1:n;}\mathbf{y}$$

$y = f_1(x, z) + f_2(v)$ with TPRS & stable computation

- Stability is much improved by the QR approach.



- But notice that the high condition number regions correspond to sensible models, with some terms completely smooth and others more variable.
- What if the model had been more complicated, with highly variable weights (e.g. logistic regression)?

Truncation

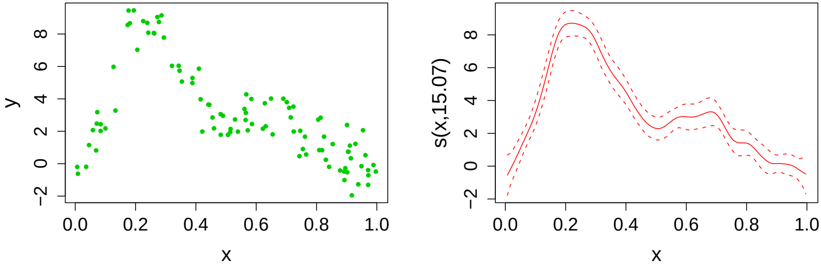
- In practice it's easy to obtain condition numbers which will defeat the QR approach.
- With care the GCV score breaks down later than the parameters, but it breaks eventually.
- The derivatives of the GCV score break down at the same point as the coefficient estimates.
- The solution is to perform

  $$\mathbf{QR} = \begin{bmatrix} \mathbf{X} \\ \mathbf{E} \end{bmatrix}$$

  with pivoting, and to then test **R** for rank deficiency.
- If rank deficiency is detected then truncate the coefficient space accordingly. This is safe — the data contain no information on the deleted space.

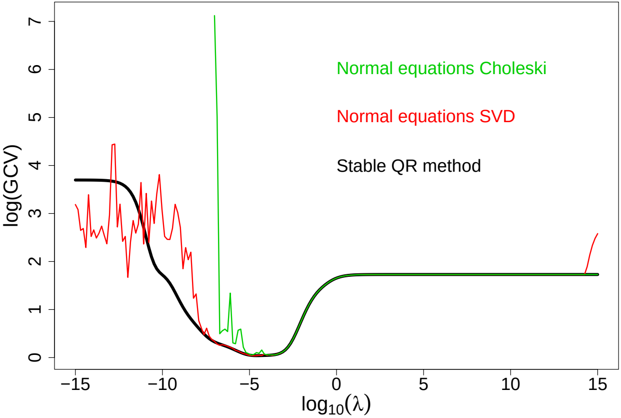
A 1D stability breakdown example

The x,y, data on the left were modelled using the cubic spline on the right, with λ chosen by GCV.



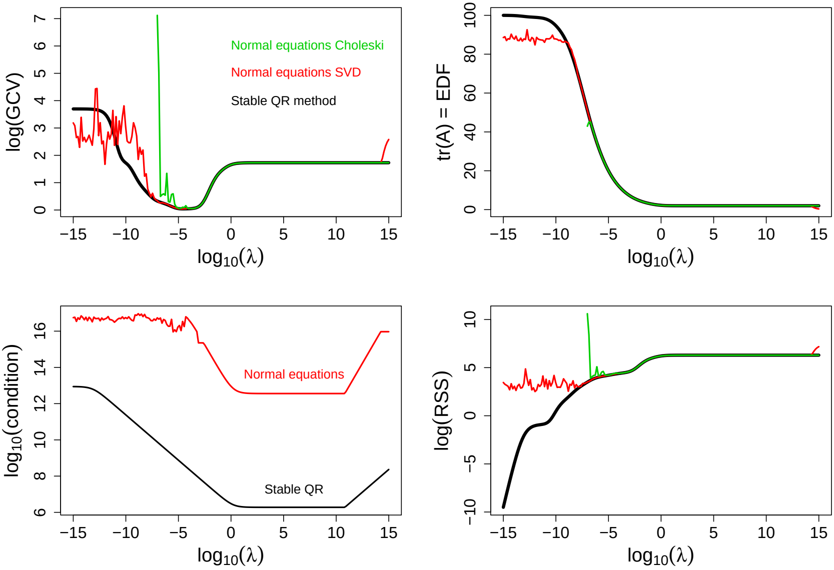
The spline basis used was a full rank 1D thin plate spline basis. The next slide compares naïve and stable GCV score calculations . . .

Stability matters for λ selection!

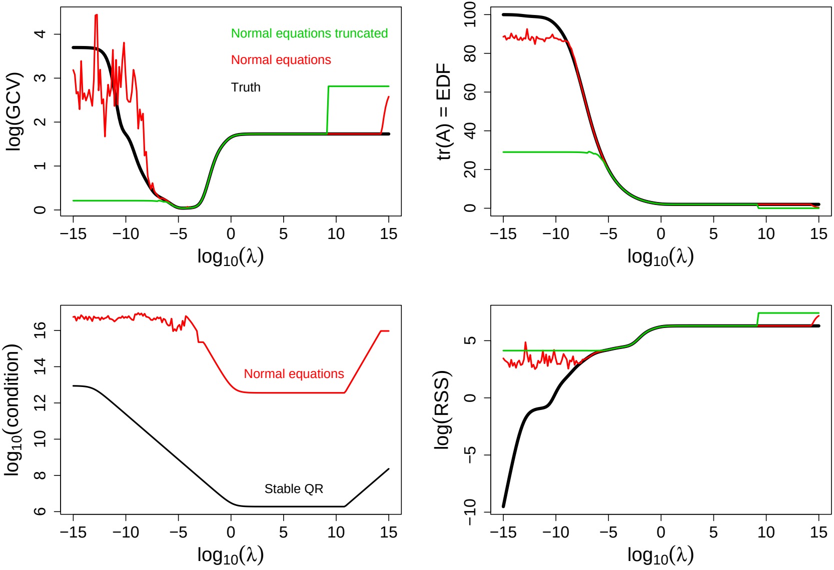


…automatic minimization of the red or green versions of GCV is not a good idea.

What goes wrong with the naïve calculations?



Example of stabilizing truncation of normal equations



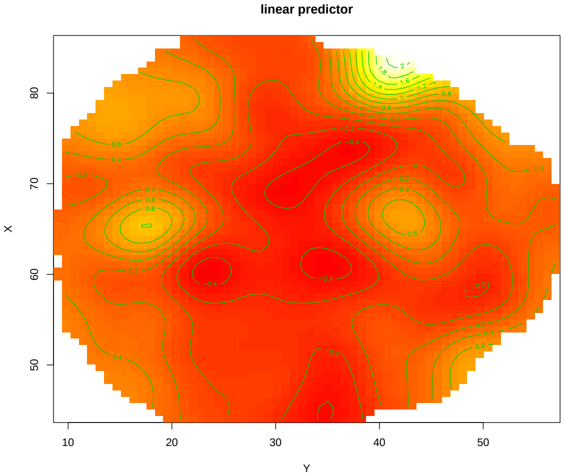
GAM computation

- For full GAMs we need to compute $\hat{\beta}$ and evaluate, e.g.

  $$\mathcal{V}(\boldsymbol{\lambda}) = \frac{nD}{[n - \text{tr}(\mathbf{F})]^2}$$

  plus its first and second derivatives w.r.t. all $\log(\lambda_j)$.
- Efficient and stable computation of the derivatives is the hard part.
- It is necessary to differentiate the P-IRLS, and to find efficient stable ways to compute $\text{tr}(\mathbf{F})$.
- The stability part employs the ideas discussed already. Efficiency is tedious to achieve, but possible.
- R package `mgcv` does it for you.

Example: Smoothed brain scan



- Model: $\log\{\mathbb{E}(\texttt{FPQ})\} = f(X, Y)$, f smooth, $\texttt{FPQ} \sim$ Gamma.
- $f(X, Y)$ shown, represented using a penalized rank 100 TPRS.

Retinopathy model

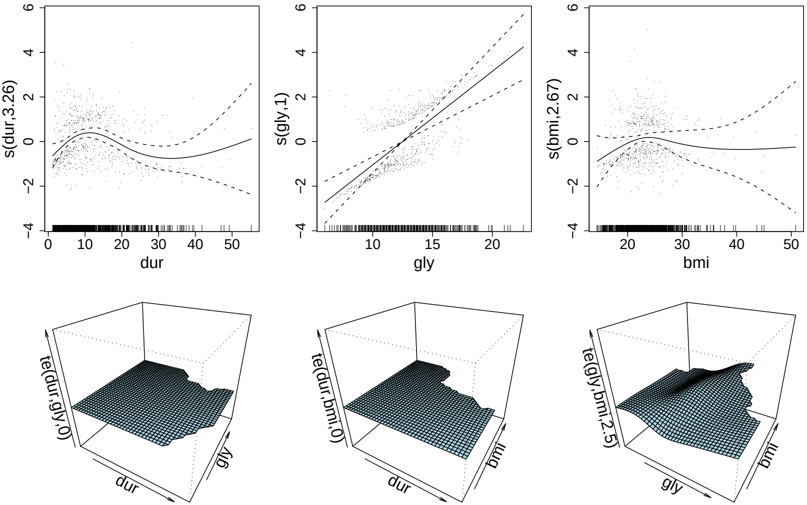
- The question was how retinopathy is related to body mass index, duration of disease at baseline and percentage glycosylated haemoglobin in blood.
- A possible model is

  $$\begin{aligned}\text{logit}\{\mathbb{E}(\texttt{ret})\} &= f_1(\texttt{dur}) + f_2(\texttt{bmi}) + f_3(\texttt{gly}) \\ &\quad + f_1(\texttt{dur}, \texttt{bmi}) + f_2(\texttt{dur}, \texttt{gly}) + f_3(\texttt{gly}, \texttt{bmi})\end{aligned}$$

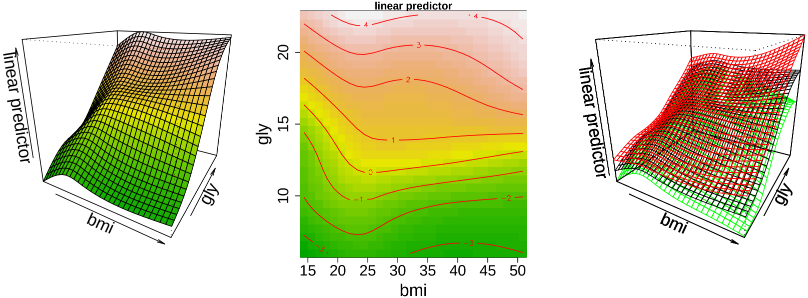
  where $\texttt{ret} \sim$ Bernoulli.
- In R, model is fit with something like

  gam(ret ∼ s(dur) + s(gly) + s(bmi) + te(dur,gly) + te(dur,bmi) + te(gly,bmi),family=binomial)

Retinopathy Estimated effects



Retinopathy GLY-BMI interaction



red/green are +/− TRUE s.e.

References

- Hastie and Tibshirani (1986) invented GAMs. The work of Wahba (e.g. 1990) and Gu (e.g. 2002) heavily influenced the work presented here. Duchon (1977) invented thin plate splines. The Retinopathy data are from Gu.
- Penalized regression splines go back to Wahba (1980), but were given real impetus by Eilers and Marx (1996) and in a GAM context by Marx and Eilers (1998).
- See Wood (2006) *Generalized Additive Models: An Introduction with R*, CRC for more information (including the air pollution modelling), or the `mgcv` package in R for implementation.