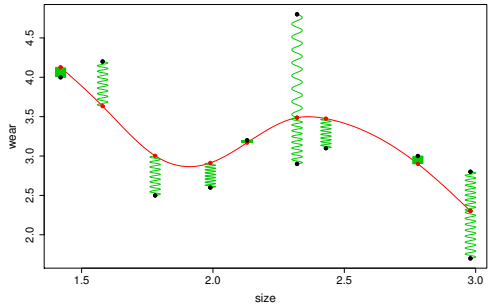
More smooth model components

Simon Wood
School of Mathematics, University of Edinburgh, U.K.

Splines

- Most smooths covered here are based on *smoothing splines*. We met these already, but here's a more visual take on the idea ...



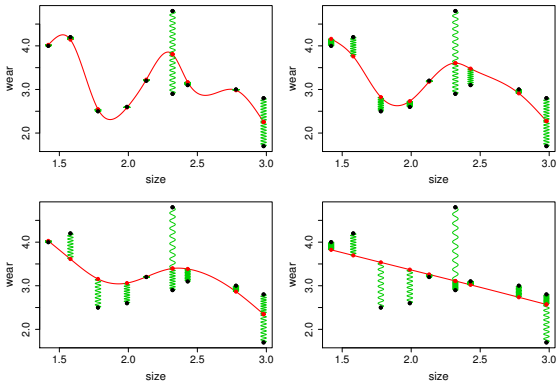
- Mathematically the red curve is the *function* minimizing

$$\sum_i (y_i - f(x_i))^2 + \lambda \int f''(x)^2 dx.$$

- We don't pre-specify a basis for f – it arises naturally from the minimization.

Splines have variable stiffness

- Varying the flexibility of the strip (i.e. varying λ) changes the *spline function* curve.



- Irrespective of λ the spline functions always have the same basis.
- But there is one basis function per data point.

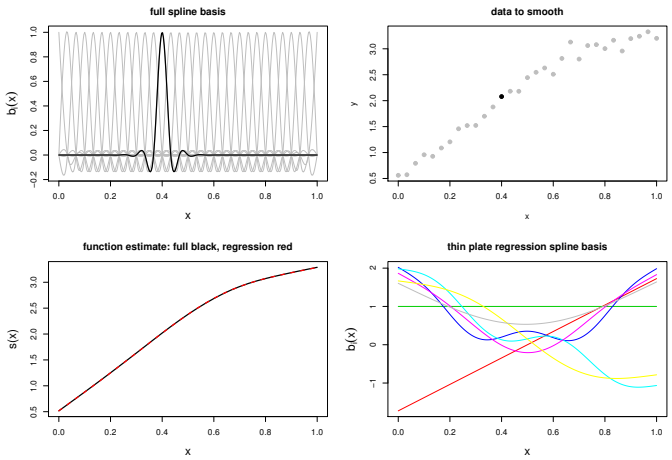
Penalized regression splines

- A basis function per data point is computationally wasteful, when penalization ensures that the *effective* degrees of freedom will be much smaller than this.
- Penalized regression splines simply use fewer spline basis functions. There are two alternatives:
  1. Choose a representative subset of your data (the 'knots'), and create the spline basis as if smoothing only those data. Once you have the basis, use it to smooth all the data.
  2. Choose how many basis functions are to be used and then solve the problem of finding the set of this many basis functions that will optimally approximate a full spline.

  I'll refer to 1 as *knot based* and 2 as *eigen based*.

Eigen basis example

- We saw knot based regression spline bases already. Here is an illustration of an eigen-basis (`"tp"` in `mgcv`).



Why is this rank reduction OK?

- Consider a spline, f, with k equally spaced knots.
- If used for interpolation the approximation error will be $O(k^{-4})$.
- If we use the spline basis for unpenalized regression with n data, the standard deviation of f will be $O(\sqrt{k/n})$. Its bias will be inherited from the approximation error: $O(k^{-4})$.
- So if we want neither bias nor sampling error to dominate then we need to set $k = O(n^{1/9})$, so that both become $O(n^{-4/9})$.
- With penalization $k = O(n^{1/5})$ is better.
- The point is that $n = O(n)$ is pointlessly wasteful. We might as well use penalized regression splines.

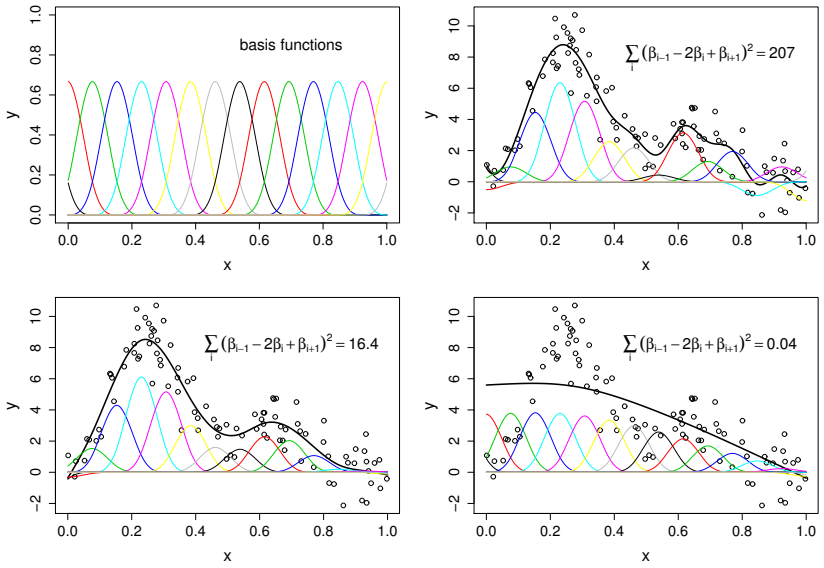
P-splines: `"ps"`, `"cp"` & `"bs"`

- These are popular in the literature (very easy to code and sparse, for example). `"ps"` and `"cp"` in `mgcv`.
- Take an evenly spaced B-spline basis to represent $f(x)$.
- Instead of penalizing something like $\int f''(x)^2 dx$, use a discrete penalty directly on the basis coefficients. e.g.

$$\sum_{j=2}^{k-1} (\beta_{j+1} - 2\beta_j + \beta_{j-1})^2.$$

- Can mix-and-match order of penalty and basis.
- Actually sparsity and mix-and-match almost as easy with derivative based penalties (`"bs"` in `mgcv`).

P-spline illustration



An adaptive smoother

- Can let the p-spline penalty vary with the predictor. e.g.

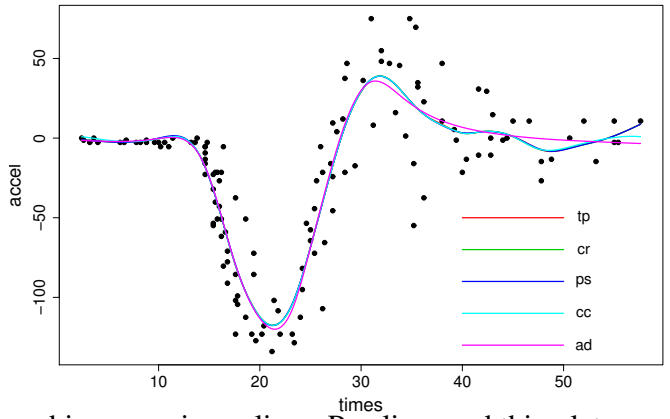
$$\mathcal{P}_a = \sum_{k=2}^{K-1} \omega_k(\beta_{k-1} - 2\beta_k + \beta_{k+1})^2 = \boldsymbol{\beta}^{\mathrm{T}}\mathbf{D}^{\mathrm{T}}\mathrm{diag}(\boldsymbol{\omega})\mathbf{D}\boldsymbol{\beta}$$

where $\mathbf{D} = \begin{bmatrix} 1 & -2 & 1 & 0 & \cdot \\ 0 & 1 & -2 & 1 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}$.

- Now let ω_k vary smoothly with k, using a B-spline basis, so that $\boldsymbol{\omega} = \mathbf{B}\boldsymbol{\lambda}$, where $\boldsymbol{\lambda}$ is the vector of basis coefficients.
- So, writing $\mathbf{B}_{\cdot k}$ for the k^{th} column of $\mathbf{B}$ we have

$$\boldsymbol{\beta}^{\mathrm{T}}\mathbf{D}^{\mathrm{T}}\mathrm{diag}(\boldsymbol{\omega})\mathbf{D}\boldsymbol{\beta} = \sum_k \lambda_k \boldsymbol{\beta}^{\mathrm{T}}\mathbf{D}^{\mathrm{T}}\mathrm{diag}(\mathbf{B}_{\cdot k})\mathbf{D}\boldsymbol{\beta} = \sum_k \lambda_k \boldsymbol{\beta}^{\mathrm{T}}\mathbf{S}_k\boldsymbol{\beta}.$$

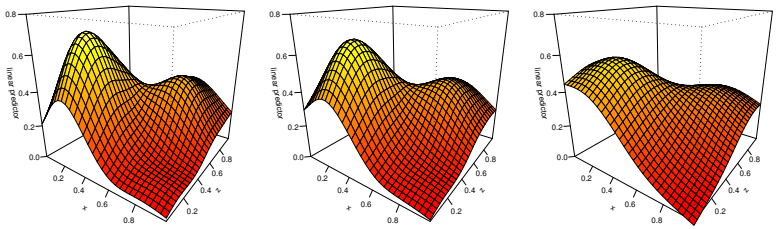
1D smooths compared



- So cubic regression splines, P-splines and thin plate regression splines give very similar results.
- A cyclic smoother matches at both ends (up to second derivative) and is a little different, of course.
- An adaptive smoother can look very different.

Beyond 1D: Isotropic smooths

- One way of generalizing splines from 1D to several D is to turn the flexible strip into a flexible sheet (hyper sheet).
- This results in a *thin plate spline*. It is an *isotropic* smooth.
- Isotropy may be appropriate when different covariates are naturally on the same scale.
- In `mgcv` terms like `s(x,z)` generate such smooths.



Thin plate splines

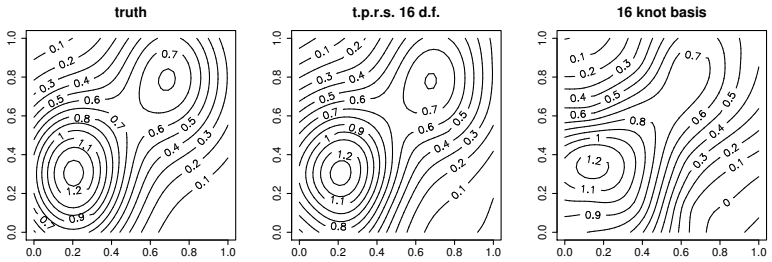
- In 2 dimensions a thin plate spline is the function minimizing

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

- This generalizes to any number of dimensions, d, and any order of differential, m, such that $2m > d + 1$.
- Actually thin plate splines are a special case of Duchon splines (Duchon, 1977). The general case allows further penalization of higher frequencies of the derivative field and relaxation of the restriction on the derivative order, m.
- Duchon splines have a basis – quadratic penalty representation, and eigen based rank versions can be produced. `mgcv` implements them as bases `"ds"` and `"tp"`.

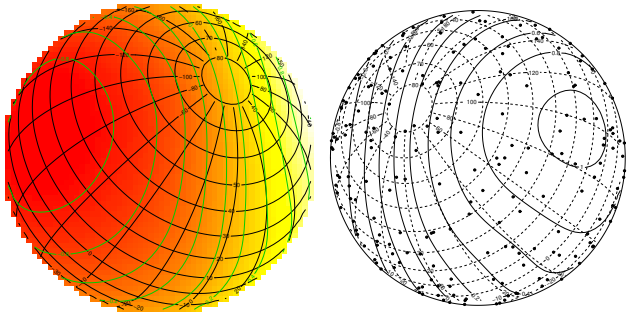
Thin plate regression spline illustration

- The eigen approximation is quite effective. The following figure compares reconstructions of of the true function on the left, using and eigen based thin plate regression spline (middle), and one based on choosing knots. Both are rank 16 approximations.



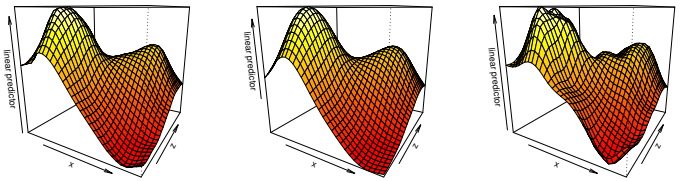
Smoothing on the globe

- Thin plate spline like smoothers can also be constructed for the sphere [`s(la,lo,bs="sos")`]...



Gaussian process smoothers

- Suppose $f(\mathbf{x})$ is a random field with correlation between $f(\mathbf{x}_i)$ and $f(\mathbf{x}_i)$ of $c(\|\mathbf{x}_i - \mathbf{x}_j\|)$. To produce this correlation, let $f(\mathbf{x}) = (1, \mathbf{x}^{\mathrm{T}})\boldsymbol{\beta} + \sum_i b_i c(\|\mathbf{x} - \mathbf{x}_i\|)$ where $\mathbf{b} \sim N(\mathbf{0}, (\lambda \mathbf{C})^{-1})$, and $C_{ij} = c(\|\mathbf{x}_i - \mathbf{x}_j\|)$.
- So f has a basis penalty representation, and eigen based rank reduction is possible. (`"gp"` in `mgcv`.)



Discrete spatial smoothing: Markov random fields

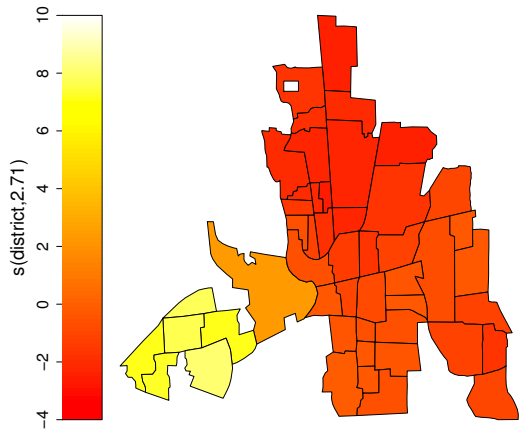
- Sometimes data come allocated to irregular partitions of space (e.g. administrative regions).
- Markov random fields are popular for smoothing such data.
- The smooth has a coefficient, γ_i, for each region.
- The neighbouring regions of each region are found, and a quadratic penalty constructed. If N_i is the set of indices of the neighbours of region i, then the simplest penalty is

$$\sum_i (\sum_{j \in N_i} (\gamma_i - \gamma_j))^2$$

- Eigen based rank reduction is also effective here.

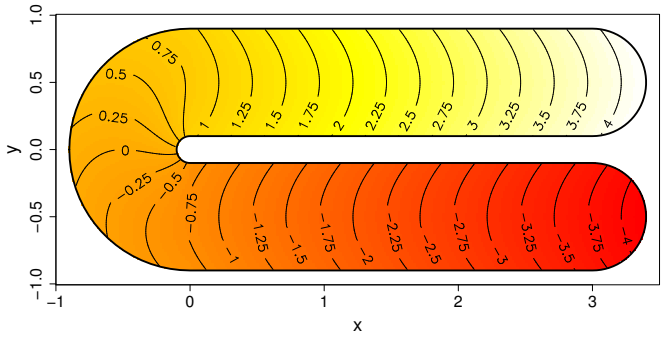
Markov random field illustration

```
data(columb.polys) ## district shapes list
xt <- list(polys=columb.polys)
gam(crime ~ s(district,bs="mrf",xt=xt),data=columb)
```



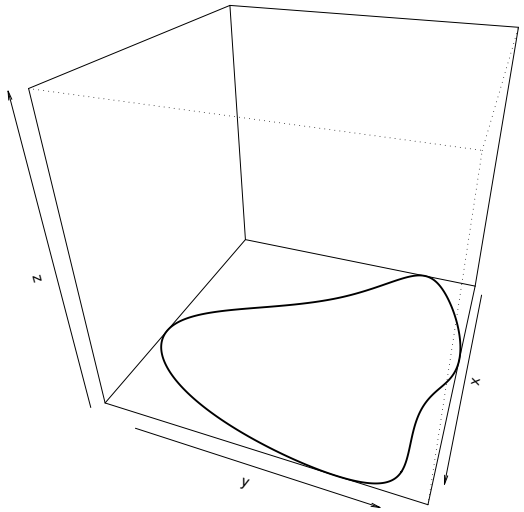
Finite area smoothing

- Suppose we want to smooth samples from this function

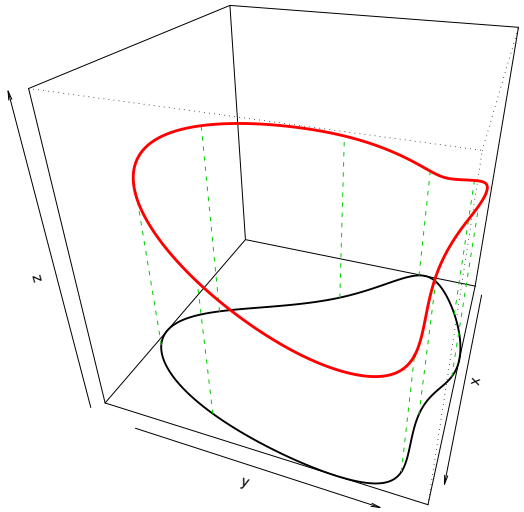


- . . . without 'smoothing across' the gap in the middle?
- Let's use a soap film . . .

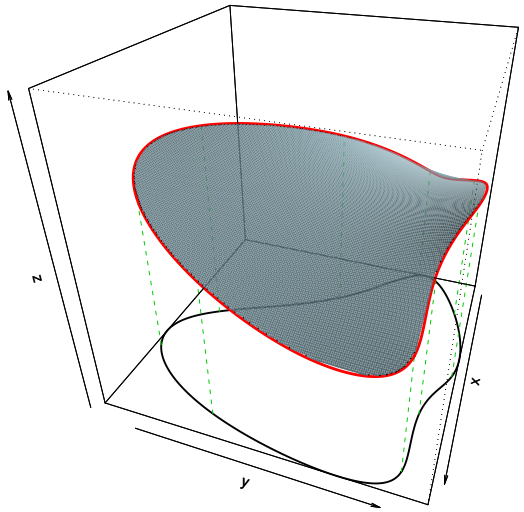
The domain



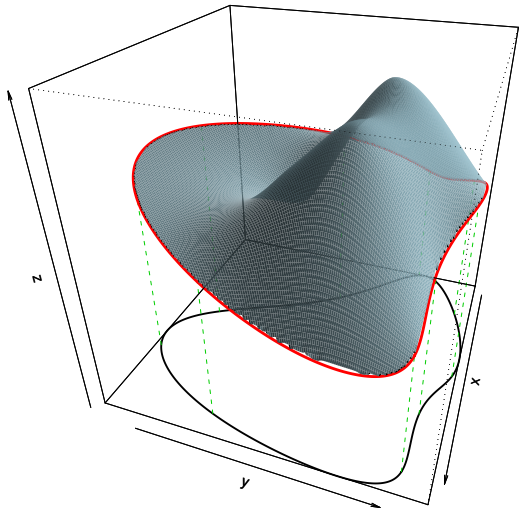
The boundary condition



The boundary interpolating film

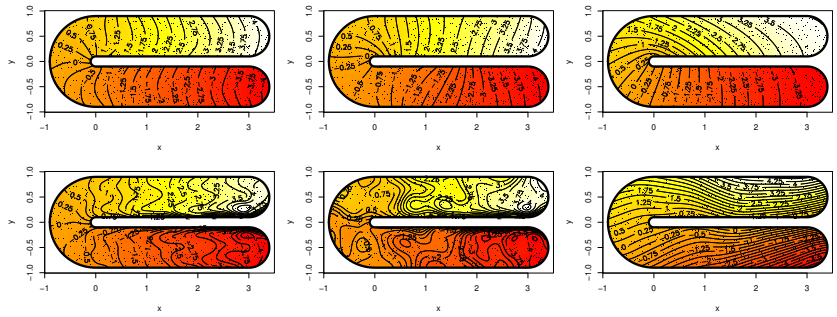


Distorted to approximate data



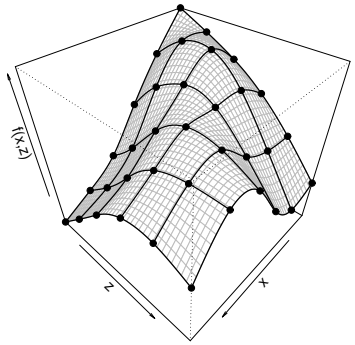
Soap film smoothers `s(...,bs="so")`

- Mathematically this smoother turns out to have a basis-penalty representation.
- It also turns out to work...



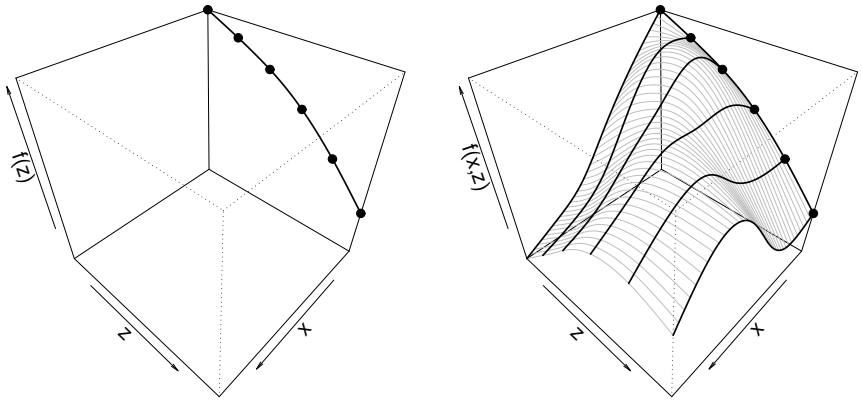
Scale invariant smoothing: tensor product smooths

- Isotropic smooths assume that a unit change in one variable is equivalent to a unit change in another variable, in terms of function variability.
- When this is not the case, isotropic smooths can be poor.
- *Tensor product smooths* generalize to multiple dimensions using a lattice of bendy strips, *with different flexibility in different directions*.



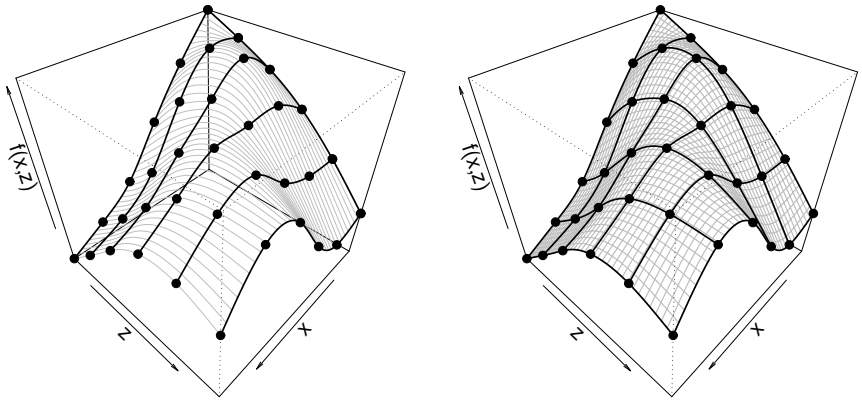
Tensor product basis construction

- Make a spline $f_z(z)$ (parameterized in terms of function values) a function of x by letting its coefficients vary smoothly with x



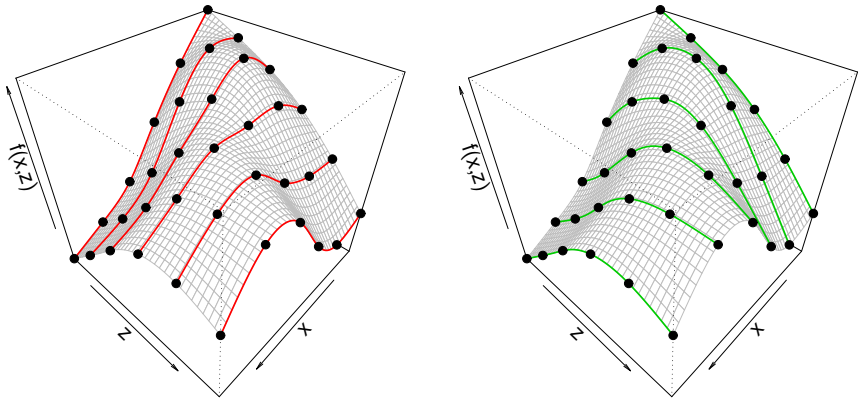
The complete tensor product smooth

- Use a spline $f_x(x)$ for each f_z coefficient (left).
- Symmetric construction: can exchange roles of f_x and f_z (right).



Tensor product penalties - one per dimension

- x-wiggliness: sum f_x spline penalties over red curves.
- z-wiggliness: sum f_z spline penalties over green curves.



Tensor product expressions

- Let $b_{zj}(z)$ and $b_{xi}(x)$ be the basis functions for f_z and f_x with penalty matrices $\mathbf{S}_x$ and $\mathbf{S}_z$. The *marginal* smoothers.
- The tensor product basis construction shown above gives:

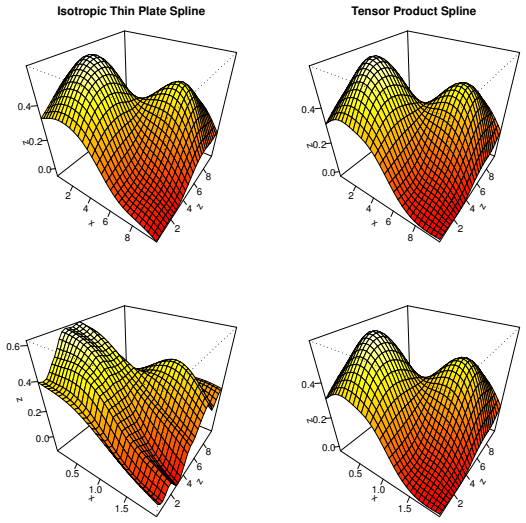
$$f(x,z) = \sum_i \sum_j \beta_{ij} b_{zj}(z) b_{xi}(x)$$

- With double penalties

$$\boldsymbol{\beta}^{\mathrm{T}} \mathbf{I} \otimes \mathbf{S}_z \boldsymbol{\beta} \text{ and } \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S}_x \otimes \mathbf{I} \boldsymbol{\beta}$$

- The construction generalizes to any number of marginals and multi-dimensional marginals.
- Can start from any marginal bases & penalties (including mixtures of types).

Isotropic vs. tensor product comparison

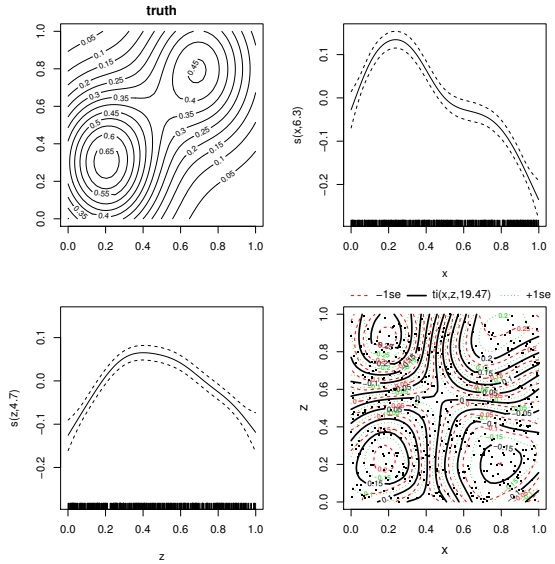


. . . each figure smooths the same data. The only modification is that x has been divided by 5 in the bottom row.

Functional ANOVA and `ti` terms

- The basis for a `te` tensor product smooth, $f(x,z)$, contains a subspace of functions of the form $f(x)+f(z)$ (similar applies in higher dimensions).
- This is because the marginal bases include the constant function in their span.
- Applying sum-to-zero constraints to the marginal bases *before* forming the tensor product smooth removes this $f(x)+f(z)$ component. See `ti()` terms in `mgcv`.
- Such a construction facilitates a 'mean effect plus interactions' smooth model.
- e.g. $f(x)+f(z)+f(x,z)$ can be fitted via `ti(x) + ti(z) + ti(x,z)`, in a stably interpretable manner. `t2` smooths are an alternative. See also Chong Gu's book 'Smoothing Spline ANOVA'.

`ti` example



Interactions with parametric effects

- Tensor product terms are smooth interactions. Sometimes an interaction between a smooth and a parametric term is required. The `by` mechanism facilitates this.
- Suppose you want a term $f(x)z$ where x and z are numeric variables. `s(x,by=z)` creates such terms (varying coefficient models/ geographic regression).
- What about interactions with a factor, g[1]?
  1. `s(x,by=g)` creates a smooth of x for each level of g.
  2. If g is an *ordered factor*, no smooth is created for its first level.
  3. `s(x,by=g,id="foo")` constrains all the smoothing parameters to the same value.
  4. `s(x,g,bs="sz")` is similar, but bases for g level smooths exclude main effect of x, allowing separate main effect smooth.
  5. `s(x,g,bs="fs")` is similar, but smooths have no un-penalized component, and are set up for efficient `gamm` computation.

[1] see `mgcv` help file `?factor.smooth` for more detail

Random effect `s(...,bs="re")`

- Statistically, smooths consist of a basis and a quadratic penalty, where the penalty matrix can be treated as the generalized inverse of a covariance matrix.
- They can therefore be estimated as random effects.
- Reversing this, we can treat simple random effects as (zero dimensional) smooths.
- `s(a,b,bs="re")` creates a terms with model matrix `model.matrix(~a:b-1)` and a scaled identity penalty/covariance matrix.
- Any number of covariates are possible.
- Function `gam.vcomp` helps later interpretation by converting smoothing parameters to variance components.

Summary

- We can treat simple random effects as 0 dimensional smooths.
- In 1 dimension, the choice of basis is not critical. The main decisions are whether it should by cyclic or not and whether or not it should be adaptive.
- In 2 dimensions and above the key decision is whether an isotropic smooth, `s`, or a scale invariant smooth, `ti`/`te`/`t2`, is appropriate. (`te`/`i`/`2` terms may be isotropic in some marginals.)
- Spatial smoothing may sometimes require more specialized smoothers (Markov random fields, spherical splines, finite area smooths).
- The basis dimension is a modelling decision that should be checked.

Further reading