Splines and basis expansion (week 3?)

15 Feb 2023

Table of contents

```
## use help("image-methods", "Matrix")
## lattice graphics: ?lattice:xyplot for details on scales
ifun <- function(x, title = "", ck = FALSE, raster = TRUE) {
    image(Matrix(x),
          sub = "", xlab = "", ylab = "",
          colorkey = ck,
          aspect = "fill",
          scales = list(x = list(draw = FALSE),
                        y = list(draw = FALSE)),
          main = title,
          useRaster = raster
          )
}
```

linear basis expansion

- transformations of various kinds
- quadratic expansion
- nonlinear transformations
- indicator variables

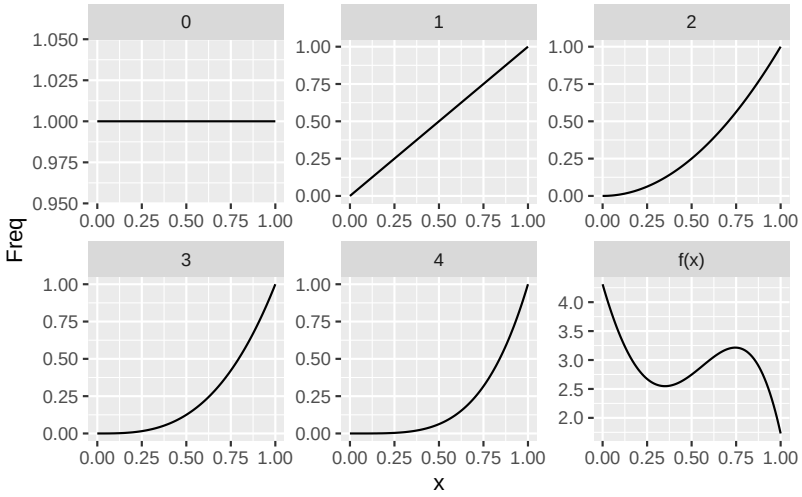
Select or regularize from the expanded set.

polynomial basis

- polynomial basis: $y_i = \sum_{j=0}^{n} \beta_j x_i^j$

```
## replicate figure 3.2 from Wood
library(ggplot2)
```

```r
x <- seq(0, 1, length = 101)
n <- 4
y <- sapply(0:n, \(j) x^j)
beta <- c(4.31, -10.72, 16.8, 2.22, -10.88)
y <- cbind(y, fx = y %*% beta)
dimnames(y) <- list(x = x, j = c(0:n, "f(x)"))
yy <- as.data.frame(as.table(y))
yy$x <- as.numeric(as.character(yy$x))
ggplot(yy, aes(x, Freq)) + geom_line() + facet_wrap(~j, scale = "free")
```



piecewise polynomial bases

- constant, linear, continuous
- basis functions
- translate from x_i to columns of $\mathbf{X}$

splines

- **piecewise** polynomials with continuity/smoothness constraints
- very useful for function approximation
- convert a single numeric predictor into a flexible basis
- efficient
- with multiple predictors, consider **additive models**

- handle interactions (multidim smooth surfaces) *if reasonably low-dimensional*: tensor products etc.

spline terminology

- **knots**: breakpoints (boundary, interior)
- order-M (ESL): continuous derivatives up to order $M-2$ (cubic, $M=4$)
- typically $M = 1, 2, 4$
- number of knots = df (degrees of freedom) -1 -intercept

truncated power basis

- $X^0 \ldots X^n$
- remaining columns are $(x-\xi_\ell)+^{M-1}$ where ℓ are the *interior knots*

truncated power basis

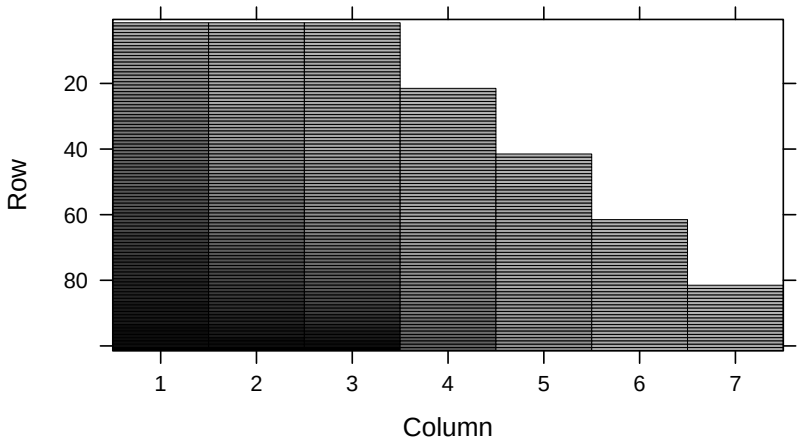
- **Kronecker product**: blockwise multiplication ($\mathbf{A} \bigotimes \mathbf{B}$ multiplies $\mathbf{B}$ by each a_{ij})
- **Khatri-Rao product**: columnwise Kronecker product
  - super-handy for combining indicator variables with

```
truncpolyspline <- function(x, df) {
    if (!require("Matrix")) stop("need Matrix package")
    knots <- quantile(x, seq(0, 1, length = df - 1))
    ## should probably use seq() instead of `:`
    ## dim: n x (df-2)
    trunc_fun <- function(k) (x>=k)*(x-k)^3
    S <- sapply(knots[1:(df-2)], trunc_fun)
    S <- as(S, "CsparseMatrix")
    ## dim: n x df
    S <- cbind(x, x^2, S)
    return(S)
}
xvec <- seq(0, 1, length = 101)
```

```
tS <- truncpolyspline(xvec, df = 7)
```

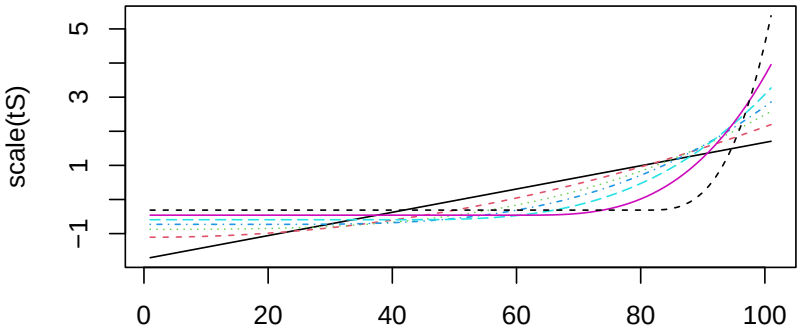
```
Loading required package: Matrix
```

```
image(tS, aspect = "fill")
```

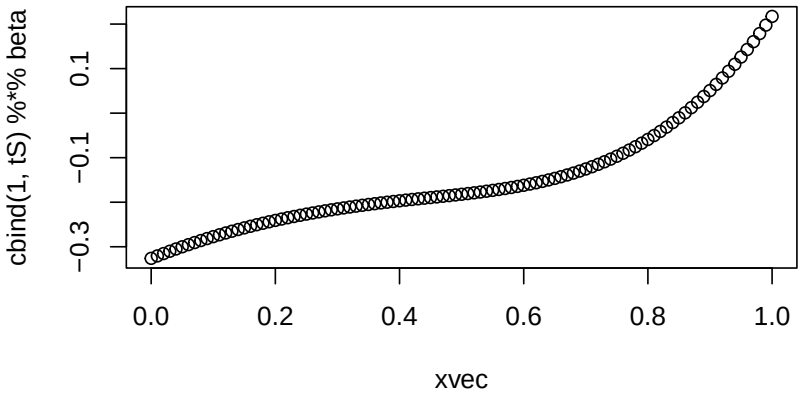


Dimensions: 101 x 7

```
matplot(scale(tS), type = "l")
```




```
set.seed(101)
beta <- rnorm(8)
plot(xvec, cbind(1, tS) %*% beta)
```



Alternatively: create directly in sparse format.

- `d`: numeric (double-precision), vs `l` (logical), `n` (position)
- `g`: general, vs. `t` (triangular), `s` (symmetric)
- `C`: compressed sparse column form, vs. `R` (row form); `T` (triplet form)

compressed sparse column form

- `@i`: vector of row-**i**ndices (0-indexed??)
- `@p`: (0-indexed) vector of **p**ointers to starting elements of each column
- `@x`: values

e.g. `M@i[M@p[4]+1]` is the row-index of the first non-zero element in the fourth column; `M@x[M@p[4]+1]` is the value of the first non-zero element in the fourth column

```
truncpolyspline2 <- function(x, df) {
    knots <- quantile(x, seq(0, 1, length = df - 1))
    nx <- length(x)
    iL <- list()
    pL <- list()
    xL <- list()
    n <- 3
    j <- 0
    pL[[1]] <- 0L
    for (i in 1:(n-1)) {
```

```
        j <- j+1
        xL[[j]] <- x^i
        iL[[j]] <- seq(nx)-1L
        pL[[j+1]] <- i*nx
    }
    for  (i in 1:(df-2)) {
        j <- j+1
        ## figure out number of non-zero elements
        ## (could squeeze out a bit more here by counting up)
        nzk <- sum(x < knots[i])
        pos <- (nzk+1):nx
        xL[[j]] <- (x[pos]-knots[i])^3
        iL[[j]] <- pos-1L
        pL[[j+1]] <- pL[[length(pL)]]+(nx-nzk)
    }
    new("dgCMatrix", i = unlist(iL), p = unlist(pL), x = unlist(xL),
        Dim = c(nx, as.integer(df)))
}
tS2 <- truncpolyspline2(xvec, df = 7)
all.equal(unname(tS), tS2) ## TRUE
```

```
[1] TRUE
```

```
identical(unname(matrix(tS)), matrix(tS2))  ## TRUE
```

```
[1] TRUE
```

B-spline basis

- splines of a given order with *minimal support* (i.e., local)
- basis functions defined by recursion (not pretty)
- convenient for regression splines (see below)

natural cubic splines

- linear constraints beyond boundary knots (so 2d and 3d derivatives are 0 at the boundaries)

```
library(splines)
bb <- bs(1:20, df = 5)
attributes(bb)[c("degree", "knots", "Boundary.knots")]
```

```
$degree
[1] 3

$knots
33.33333% 66.66667%
 7.333333 13.666667

$Boundary.knots
[1]  1 20
```

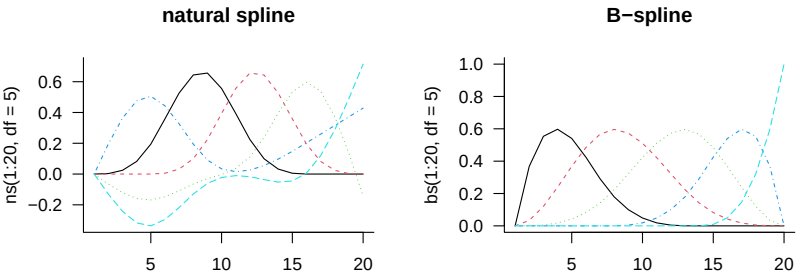
```
nn <- ns(1:20, df = 7)
attributes(nn)[c("degree", "knots", "Boundary.knots")]
```

```
$degree
[1] 3

$knots
14.28571% 28.57143% 42.85714% 57.14286% 71.42857% 85.71429%
 3.714286  6.428571  9.142857 11.857143 14.571429 17.285714

$Boundary.knots
[1]  1 20
```

```
par(mfrow = c(1,2),las =1, bty ="l")
matplot(ns(1:20, df = 5), type = "l", main = "natural spline")
matplot(bs(1:20, df = 5), type = "l", main = "B-spline")
```

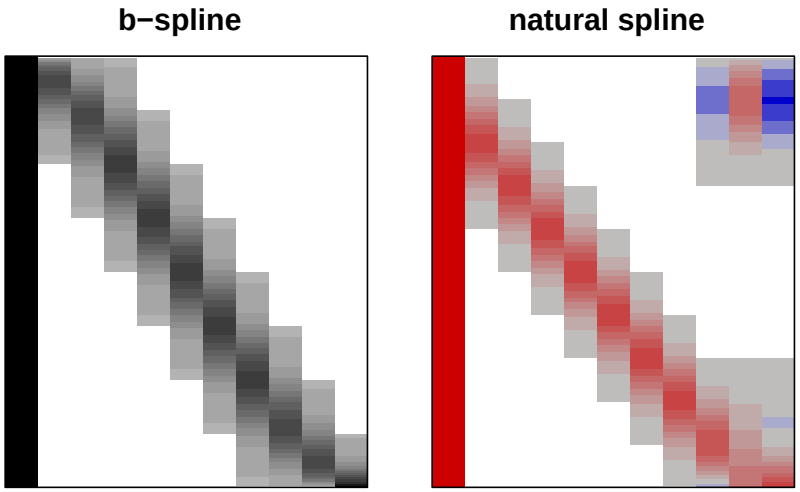


variance of predictions

- suppose $\mathbf{V}$ = variance of coefficient β
- then covariance matrix of predictions $\mathbf{X}\beta$ is $\mathbf{XVX}^\top$
- Variance of predictions, brute force: Diag($\mathbf{XVX}^\top$)
- Clever: compute diagonal directly
  - `emulator::quad.diag` (`colSums(crossprod(M, Conj(tx)) * tx)`)

sparsity patterns

```
library(Matrix)
dd <- data.frame(x=1:200)
Xb <- model.matrix(~splines::bs(x, df = 10), data = dd)
Xn <- model.matrix(~splines::ns(x, df = 10), data = dd)
gridExtra::grid.arrange(
               ncol = 2,
               ifun(Xb, "b-spline"),
               ifun(Xn, "natural spline")
               )
```



examples: South African heart disease

- use splines in a GLM with no additional effort
- fit splines to all continuous variables
- ESL says "use four natural spline bases" (… elements??)
- i.e. `df = 4` (no intercept)
- stepwise deletion via AIC
- why??
- showing p-values (why???)
- `stepAIC(..., direction = "backward")`

phoneme example

- combination of feature transformation (time to Fourier domain) and regularization
- smooth first, regress afterwards

smoothing splines

- as many knots as data points
- plus squared-second-derivative ("wiggliness") penalty

$$\text{RSS} + \lambda \int (f''(t))^2 \, dt$$

* defined on an infinite-dimensional space * minimizer is a natural cubic spline with knots at x_i

$$(\mathbf{y} - \mathbf{N}\theta)^\top(\mathbf{y} - \mathbf{N}\theta) + \lambda\theta^\top\Omega_N\theta$$

with $\{\Omega_N\}_{jk} = \int N_j''(t)N_k''(t)\,dt$ $$ **generalized** ridge regression: penalize by $\lambda\Omega_N$ rather than λI * same data augmentation methods as before except that now we use $\sqrt{\lambda}C$ where C is a matrix, and the "square root" of Ω_N

See Wood (2017a), Perperoglou et al. (2019)

constructing smoothing penalties

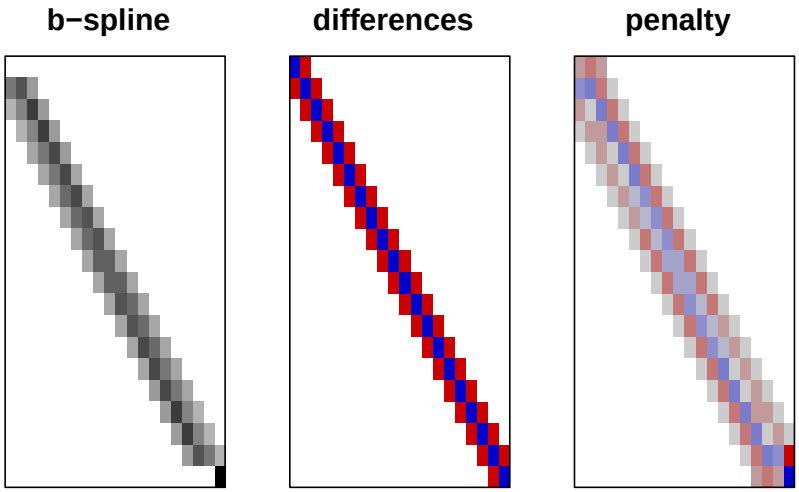
- Eilers and Marx (1996): use (products of) finite differences as approximation to squared second derivative penalty

```
library(splines)
b <- bs(1:20, df = 20)
m <- diag(-2, nrow = 20)
m[row(m) == col(m) -1 ] <- 1
m[row(m) == col(m) +1 ] <- 1
gridExtra::grid.arrange(
              nrow = 1,
              ifun(b, "b-spline"),
              ifun(m, "differences"),
              ifun(m %*% b, "penalty")
            )
```


———. 2017a. *Generalized Additive Models: An Introduction with R*. CRC Texts in Statistical Science. Chapman & Hall. https://www.amazon.com/Generalized-Additive-Models-Introduction-Statistical-ebook/dp/B071Z9L5D5/ref=sr_1_1?ie=UTF8&qid=1511887995&sr=8-1&keywords=wood+additive+models.

Perperoglou, Aris, Willi Sauerbrei, Michal Abrahamowicz, and Matthias Schmid. 2019. "A Review of Spline Function Procedures in R." *BMC Medical Research Methodology* 19 (1): 46. https://doi.org/10.1186/s12874-019-0666-3.

Eilers, Paul H. C., and Brian D. Marx. 1996. "Flexible Smoothing with B-Splines and Penalties." *Statistical Science* 11 (2): 89–121. https://doi.org/10.1214/ss/1038425655.



Eilers and Marx:

> This system has a banded structure because of the limited overlap of the B-splines. It is seldom worth the trouble to exploit this special structure, as the number of equations is equal to the number of splines, which is generally moderate (10–20).

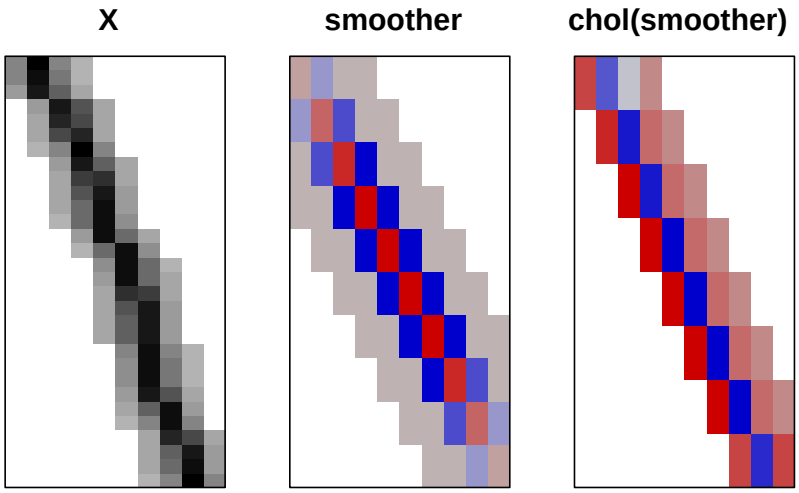
Wood (2017b): it's not that hard (if you're Simon Wood!) to generate the banded matrix Ω from the derivatives of the B-spline basis

———. 2017b. "P-Splines with Derivative Based Penalties and Tensor Product Smoothing of Unevenly Distributed Data." *Statistics and Computing* 27 (4): 985–89. https://doi.org/10.1007/s11222-016-9666-x.

```
library(mgcv)
set.seed(101)
x <- sort(runif(30))
sm <- smoothCon(s(x, bs = "bs", k = 10), data.frame(x=x))
names(sm[[1]])
```

```
 [1] "term"           "bs.dim"         "fixed"          "dim"
 [5] "p.order"        "by"             "label"          "xt"
 [9] "id"             "sp"             "m"              "X"
[13] "knots"          "S"              "D"              "rank"
[17] "null.space.dim" "plot.me"        "side.constrain" "repara"
[21] "C"              "df"             "S.scale"
```

```
gridExtra::grid.arrange(
                nrow = 1,
                ifun(sm[[1]]$X, "X"),
                ifun(sm[[1]]$S[[1]], "smoother"),
                ifun(sm[[1]]$D[[1]], "chol(smoother)")
                )
```



degrees of freedom and smoother matrix

- The equivalent of the hat matrix is

$$\mathbf{N}(\mathbf{N}^\top\mathbf{N} + \lambda\Omega_N)^{-1}\mathbf{N}^\top$$

- Also called the **smoother matrix**
- hat matrix is *idempotent* (why?), smoother matrix is *shrinking*
- smoother matrix has lower rank
- effective degrees of freedom = trace of hat matrix (again)
- can write as $(\mathbf{I} + \lambda\mathbf{K})^{-1}$
- similarly to ridge regression, eigenvectors are shrunk by a factor $1/(1 + \lambda d_k)$ where d_k s an eigenvector of $\mathbf{K}$.

reduced-rank splines

- We *can* use as many knots as observations, but do we really need to?
- From `?mgcv::s`:

> … exact choice of 'k' is not generally critical: it should be chosen to be large enough that you are reasonably sure of having enough degrees of freedom to represent the underlying 'truth' reasonably well, but small enough to maintain reasonable computational efficiency. Clearly 'large' and 'small' are dependent on the particular problem being addressed.
>
> The default basis dimension, 'k', is the larger of 10 and 'm[1]' (spline order)

fitting additive models (more than one smooth term): backfitting

Hastie and Tibshirani (1987)

Hastie, Trevor, and Robert Tibshirani. 1987. "Generalized Additive Models: Some Applications." *Journal of the American Statistical Association* 82 (398): 371–86. https://doi.org/10.1080/01621459.1987.10478440.

- scatterplot smoother ($S()$: **any** smoothing method, e.g. local linear or kernel estimation)
- **backfitting**
  - take partial residuals: $r_{ij} = y_i - \hat{\alpha} - \sum_{k \neq j} \hat{f}_k(X_{ki})$
  - smooth them ($\hat{f}_j(x_{ji}) = S(r_j | x_{ji})$)
- either do a *cyclic* fit (backfitting), fitting on partial residuals each time
- can do *semiparametric* fitting (some regular linear terms, some smoothed terms)
- **local scoring** (the same as an IRLS step)

$$
\begin{aligned}
\hat{\eta} &= \hat{\alpha} + \sum \hat{f}_j \\
\hat{\mu} &= g^{-1}(\hat{\eta}) \\
\mathbf{z} &= \hat{\eta} + (y - \mu)/V(\mu) \\
\mathbf{w} &= \mathrm{Diag}(1/V(\mu))
\end{aligned}
$$

Now do back-fitting (instead of weighted least squares) on $\mathbf{z}$ with weights $\mathbf{w}$

historical note

- Backfitting was developed in the context of **alternating conditional expectations** (Breiman and Friedman 1985); finding optimal transformations for the response and each of the variables in a multivariate regression.
- Univariate ACE is like backfitting but alternating between transformations for y and x rather than among the different predictor variables
- `acepack` package for R

Breiman, Leo, and Jerome H. Friedman. 1985. "Estimating Optimal Transformations for Multiple Regression and Correlation." *Journal of the American Statistical Association* 80 (391): 580–98. https://doi.org/10.1080/01621459.1985.10478157.

fitting additive models: alternative

- stuff the whole thing into one giant GLM with appropriate penalization

generalized cross-validation

Larsen (2015), Golub, Heath, and Wahba (1979)

- minimize $\text{RSS}/(\text{Tr}(\mathbf{I}-\mathbf{S}(\lambda)))^2$
- "a rotation-invariant version of PRESS" $(\sum(e_i/(1-h_i i))^2)$
- replace RSS with approximation of deviance,

$$||\sqrt{\mathbf{W}}(\mathbf{z}-\mathbf{X}\beta)||^2$$

  for generalized (non-Gaussian) models
- also very close to AIC
- minimize in **outer loop**
  - can find $\partial\beta/\partial(\log\lambda)$ and the derivative of the trace of the hat matrix by algebra (Wood 1st ed. §§4.7.1) and use Newton or quasi-Newton to optimize GCV

Larsen, Kim. 2015. "GAM: The Predictive Modeling Silver Bullet | Stitch Fix Technology – Multithreaded." *MultiThreaded (StitchFix)*. https://multithreaded.stitchfix.com/blog/2015/07/30/gam/.

Golub, Gene H., Michael Heath, and Grace Wahba. 1979. "Generalized Cross-Validation as a Method for Choosing a Good Ridge Parameter." *Technometrics* 21 (2): 215–23. https://doi.org/10.1080/00401706.1979.10489751.

REML criterion

- Reiss and Ogden (2009), Wood (2011)
- less likely to overfit than GCV
- reduced tendency to multiple minima

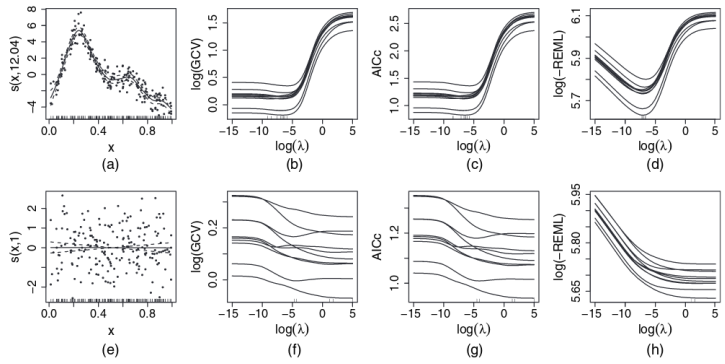


Fig. 1. Example comparison of GCV, AICc and REML criteria: (a) some (x,y)-data modelled as $y_i = f(x_i) + \varepsilon_i$, ε_i independent and identically distributed $N(0,\sigma^2)$ where smooth function f was represented by using a rank 20 thin plate regression spline (Wood, 2003); (b)–(d) various smoothness selection criteria plotted against logarithmic smoothing parameters, for 10 replicates of the data (each generated from the same 'truth') (note how shallow the GCV and AICc minima are relative to the sampling variability, resulting in rather variable optimal λ-values (which are shown as a rug plot), and a propensity to undersmooth; in contrast the REML optima are much better defined, relative to the sampling variability, resulting in a smaller range of λ-estimates); (e)–(h) are equivalent to (a)–(d), but for data with no signal, so that the appropriate smoothing parameter should tend to ∞ (note GCV's and AICc's occasional multiple minima and undersmoothing in this case, compared with the excellent behaviour of REML; ML (which is not shown) has a similar shape to REML)

computing the REML criterion

- treat spline smoothing as a *mixed model* problem
- spline (penalized) parameters are $\mathbf{u}$
- $y|u \sim N(\mathbf{X}\beta + \mathbf{Z}, \sigma^2\mathbf{I})$; $\sim N(0, (\sigma^2/\lambda)\mathbf{W}^{-1})$
- where the $\mathbf{W}$ is the penalty matrix
- corresponds to minimizing $||\mathbf{y} - \mathbf{X}\beta - \mathbf{Z}||^2 + \lambda\top\mathbf{W}$
- "fixed effects are viewed as random effects with improper uniform priors and are integrated out" (Wood 2011)
- **Laplace approximation**: may be imprecise for binary or low-mean Poisson data …

Reiss, Philip T., and R. Todd Ogden. 2009. "Smoothing Parameter Selection for a Class of Semiparametric Linear Models." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 71 (2): 505–23. https://doi.org/10.1111/j.1467-9868.2008.00695.x.

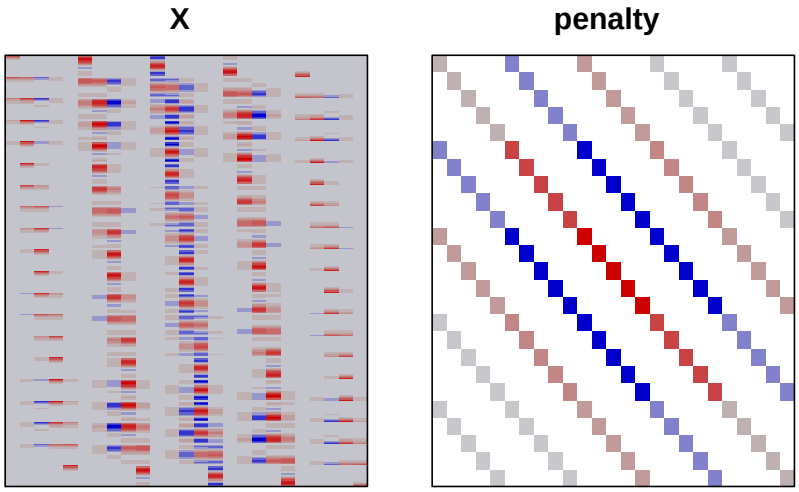
———. 2011. "Fast Stable Restricted Maximum Likelihood and Marginal Likelihood Estimation of Semiparametric Generalized Linear Models." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 73 (1): 3–36. https://doi.org/10.1111/j.1467-9868.2010.00749.x.

multidimensional splines

tensor product

- $g_{jk}(X) = \sum jh_{1j}(X_1)h_{2k}(X_2)$
- if we have a model matrix written out as `expand.grid(xvec, yvec)`

```
dd <- expand.grid(x = 1:20, y = 1:20)
sm <- smoothCon(te(x, y, k = c(5,5), bs = "bs"), data = dd)
gridExtra::grid.arrange(
                nrow = 1,
                ifun(sm[[1]]$X, "X"),
                ifun(sm[[1]]$S[[1]], "penalty")
                )
```



- Tensor product $\mathbf{X}$ is the Kronecker product of the two bases, unpacked properly (ugh)

thin-plate splines

- Wood (2003)
- tensor product smooth
- Impose a wiggliness penalty on the **cross-second-derivatives**

Wood, Simon N. 2003. "Thin Plate Regression Splines." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 65 (1): 95–114. https://doi.org/10.1111/1467-9868.00374.

- $\int \partial^2 f/\partial x_1^2 + \partial^2 f/\partial x_2^2 + \partial^2 f/\partial x_1 x_2$
- results in an **isotropic** smooth (may or may not be desirable)
- full thin-plate spline gets big fast
- **reduced-rank** TPRS: **Lanczos iteration** to find the rank-k truncated eigendecomposition of a symmetric matrix in $O(kn^2)$ operations

  The default basis dimension for this class is 'k=M+k.def' where 'M' is the null space dimension (dimension of unpenalized function space) and 'k.def' is 8 for dimension 1, 27 for dimension 2 and 100 for higher dimensions. This is essentially arbitrary, and should be checked, but as with all penalized regression smoothers, results are statistically insensitive to the exact choi[c]e

null space

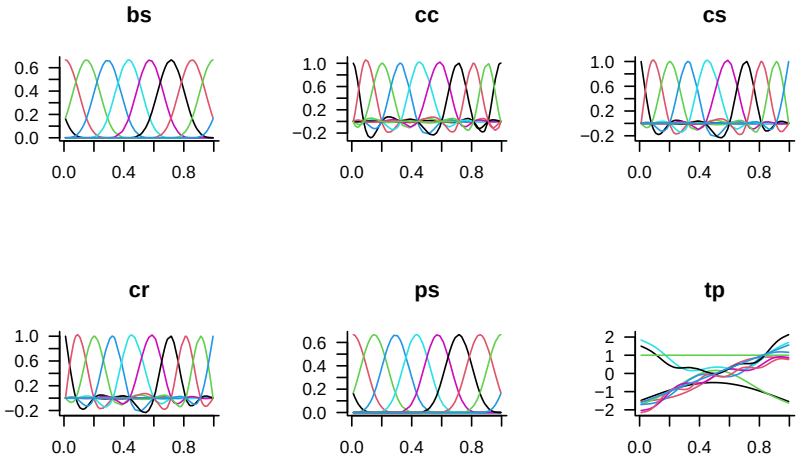
- Space of functions that are **not** exposed to shrinkage
- Typically functions linear in x (i.e., spline shrinks to a linear term as $\lambda \to \infty$.
- e.g. for ridge the null space is the intercept term.
- From `?mgcv::smooth.construct.cs.smooth.spec`:

  The shrinkage version of the smooth [i.e. bs = "cs"], eigen-decomposes the wiggliness penalty matrix, and sets its 2 zero eigenvalues to small multiples of the smallest strictly positive eigenvalue. The penalty is then set to the matrix with eigenvectors corresponding to those of the original penalty, but eigenvalues set to the pe[r]turbed versions. This penalty matrix has full rank and shrinks the curve to zero at high enough smoothing parameters.

```
apropos("smooth.construct") |>
    gsub(pattern = "(smooth.construct.)|(.smooth.spec)", replacement = "")
```

```
 [1] "smooth.construct" "ad"               "bs"               "cc"
 [5] "cp"               "cr"               "cs"               "ds"
 [9] "gp"               "mrf"              "ps"               "re"
[13] "sf"               "so"               "sos"              "sw"
[17] "t2"               "tensor"           "tp"               "ts"
[21] ""
```

```
xvec <- sort(runif(101))
mfun <- function(bs) {
    sm <- smoothCon(s(x, bs = bs), data.frame(x=xvec))
    matplot(xvec, sm[[1]]$X, type = "l", lty = 1, main = bs, xlab = "", ylab = "")
}
par(mfrow = c(2,3), las = 1, bty = "l")
svec <- c("bs", "cc", "cs", "cr", "ps", "tp")
invisible(sapply(svec, mfun))
```



Consider:

- cost of setting up the basis in the first place

- sparsity (¿ not that important ?)
- mean-squared error for a given k

other languages

- Julia doesn't seem to have much (discussion here, beginning of a package

```
using Pkg
Pkg.add(url="https://github.com/hendersontrent/GAM.jl.git")
using Random, RDatasets, GAM

mtcars = dataset("datasets", "mtcars");
X = Matrix(mtcars[:, [:AM, :Cyl, :WT, :HP]]);
y = mtcars[:, :MPG];
model = fit_gam(X, y, :gaussian)
```

(fails almost immediately!)

Python: `pyGAM` here

project possibilities

- grouped lasso with spline penalties for spline selection?
- other combinations of spline/elastic net machinery? (Marra and Wood 2011)

Marra, Giampiero, and Simon N. Wood. 2011. "Practical Variable Selection for Generalized Additive Models." *Computational Statistics & Data Analysis* 55 (7): 2372–87. https://doi.org/10.1016/j.csda.2011.02.004.

wavelets

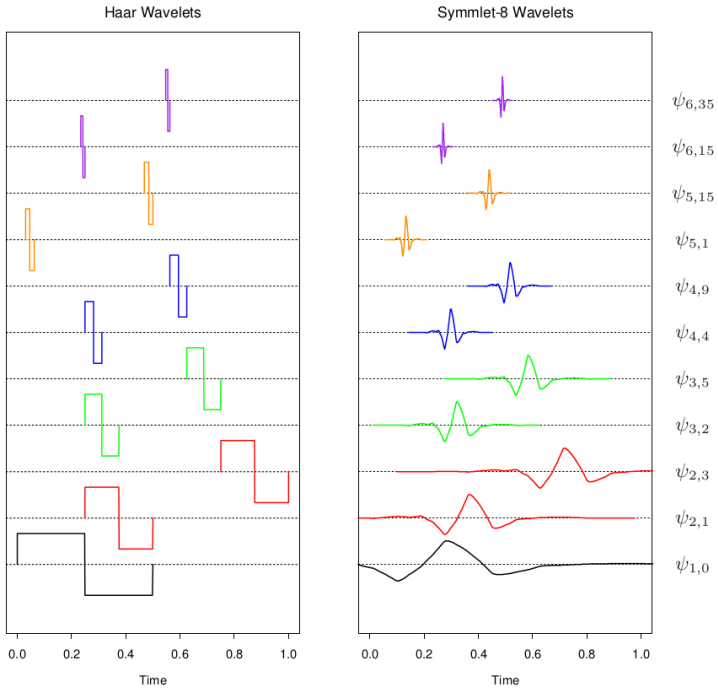


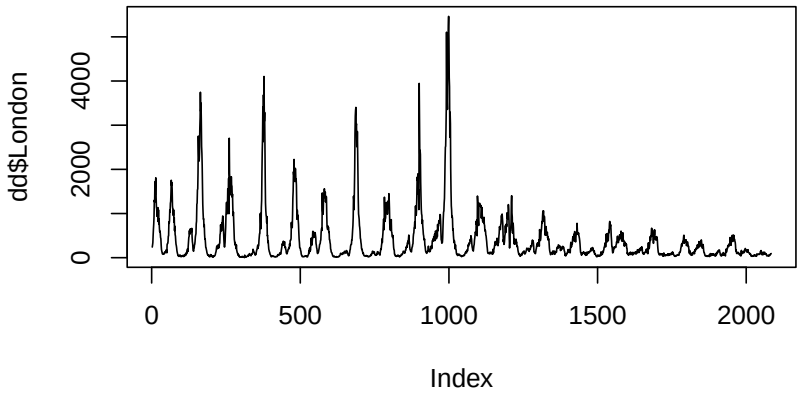
FIGURE 5.16. *Some selected wavelets at different translations and dilations for the Haar and symmlet families. The functions have been scaled to suit the display.*

* time and frequency localization * Haar, symlet * **fast wavelet transform**: $O(n)$ or $O(n \log n)$ depending on details/version

Donoho et al. (1995)

```
url <- "http://ms.mcmaster.ca/~bolker/measdata/ewcitmeas.dat"
download.file(url, dest = "meas.dat")
dd <- read.table("meas.dat", na.strings = "*", header = TRUE, comment = "")
plot(dd$London, type = "l")
```

Donoho, David L., Iain M. Johnstone, Gérard Kerkyacharian, and Dominique Picard. 1995. "Wavelet Shrinkage: Asymptopia?" *Journal of the Royal Statistical Society: Series B (Methodological)* 57 (2): 301–37. https://doi.org/10.1111/j.2517-6161.1995.tb02032.x.




```
library(wavethresh)
```

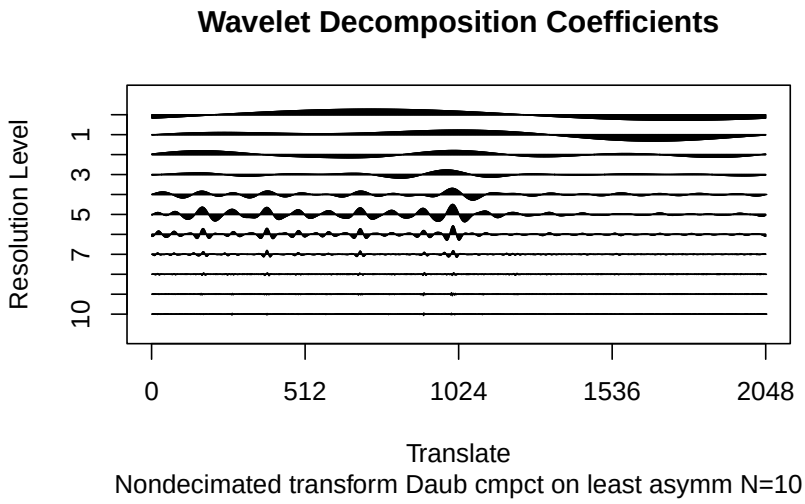
```
Loading required package: MASS
```

```
WaveThresh: R wavelet software, release 4.7.2, installed
```

```
Copyright Guy Nason and others 1993-2022
```

```
Note: nlevels has been renamed to nlevelsWT
```

```
dd$date <- with(dd, as.Date(sprintf("19%d-%d-%d", YY, MM, X.DD)))
## skip (first) NA value: power-of-2 length
w <- wd(dd$London[2:2049], type = "station")
plot(w)
```

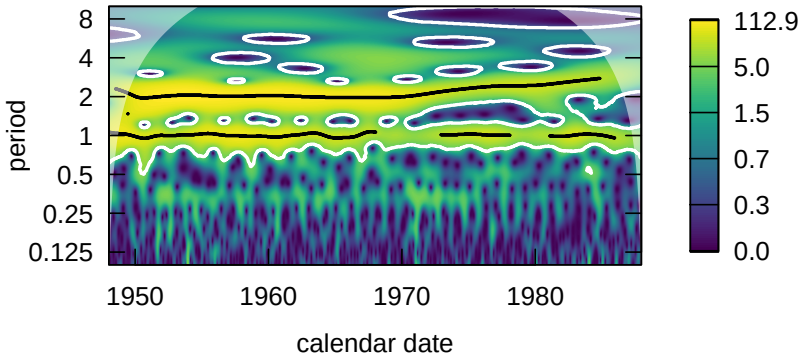



```
[1] 11286.62 11286.62 11286.62 11286.62 11286.62 11286.62 11286.62 11286.62
[9] 11286.62 11286.62 11286.62
```

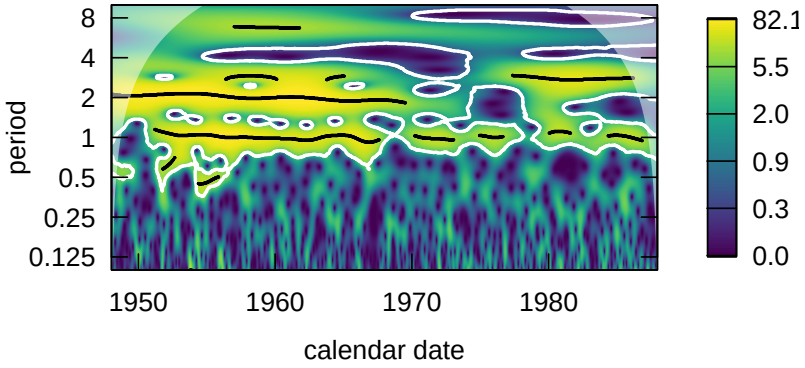
```
library(WaveletComp)
library(viridisLite)
dd$log_London <- log10(dd$London)
dd$log_Bristol <- log10(dd$Bristol+1)
dd$log_Bristol[1290]  <- 0  ## replace NA
any(is.na(dd$log_Bristol))
ww <- analyze.wavelet(dd[-1,], "log_London",
                loess.span = 0,
                dt = 1/52,
                dj = 1/100,
                lowerPeriod = 0.1,
                upperPeriod = 10)
ww_b <- analyze.wavelet(dd[-1,], "log_Bristol",
                loess.span = 0,
                dt = 1/52,
                dj = 1/100,
                lowerPeriod = 0.1,
                upperPeriod = 10)
```

```
library(WaveletComp)
library(viridisLite)
```

```
wt.image(ww, show.date = TRUE,
         color.palette = "viridis(n.levels, direction = -1)")
```

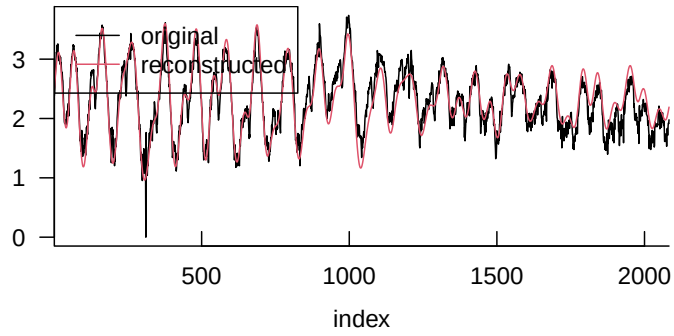



```
wt.image(ww_b, show.date = TRUE,
         color.palette = "viridis(n.levels, direction = -1)")
```




```
par(las=1, bty = "l")
reconstruct(ww)
```

```
Your input object class is 'analyze.wavelet'...
Your time series 'log_London' will be reconstructed...
Starting the reconstruction process...
Original (detrended) and reconstructed series are being plotted...
```



ower level: 0, significance level: 0.05, only coi: FALSE, only ridge: FALSE, p

```
Class attributes are accessible through following names:
series rec.waves loess.span lvl only.coi only.sig siglvl only.ridge rnum.used rescale dt dj Per
```

(R wavelet packages: `wavethresh`, `wsyn`, `WaveletComp`, `Rwave`, ...)

- compression: drop smallest components
- **or** soft-threshold as for lasso (SURE)
- as penalized component: Wand and Ormerod (2011)

Wand, M. P., and J. T. Ormerod. 2011. "Penalized Wavelets: Embedding Wavelets into Semiparametric Regression." *Electronic Journal of Statistics* 5 (none). https://doi.org/10.1214/11-EJS652.