lmeSplines

An R package for fitting smoothing spline terms in LME models

by Rod Ball

Smoothing splines can be formulated in terms of linear mixed models. Corresponding to any smoothing spline term there is a mixed model with a set of random effects, a covariance structure, and a Z-matrix linking the random effects back to observations in the users dataframe. For a smoothing spline model a single variance component is estimated which is inversely proportional to the smoothing parameter for the smoothing spline. The `lmeSplines` package provides functions for generating and manipulating the necessary Z-matrices corresponding to sets of random effects. Using the Choleski decomposition of the covariance matrix, the random effects are transformed to an independent set of random effects, enabling the model to be fitted in `lme` with existing `pdMat` classes. Model predictions can be obtained at the data points and, by interpolation in the Z-matrices, at alternate points using `predict.lme`.

Smoothing splines

Smoothing splines are a parsimonious representation (only one variance parameter is fitted) of a non-parametric curve. Compared with non-linear models, smoothing splines offer a number of advantages:

- Avoiding the need to find a parametric model.
- Avoiding strong model assumptions about the form of the curve.
- Avoiding problems e.g. in longitudinal data where some individuals don't follow the standard curves.
- Can be used to test for smooth departures from a given model.
- A non-linear mixed model can be replaced with a linear model giving more rapid convergence of model fits, and hence enabling more complex and realistic variance structures to be fitted.

Smoothing splines as mixed models

Our package is based on the formulation given in Verbyla *et al* (1999), and extends the capabilities of the NLME package for fitting linear and non-linear mixed models. Readers interested in trying NLME are recommended to read the excellent book (Pinheiro and Bates 2000). The NLME package is substantial, with comprehensive on-line documentation, however it helps to have worked through the examples, and have an overview understanding of the system to fully appreciate its potential.

In the one dimensional case, the smoothing spline for fitting a function of the form $y = g(t) + \epsilon$, is found by maximising the penalised likelihood:

$$\text{penalised likelihood} = \log \text{likelihood} - \lambda \int g''(t)^2 dx \tag{1}$$

Restricting to the observed data points, the choice of g from an infinite dimensional space reduces to a finite dimensional problem, which can be expressed as a linear mixed model. The mixed model has the form:

$$y = X_s\beta_s + Z_s u_s + \epsilon \tag{2}$$

where β are fixed effects with intercept and slope coefficients, u_s is a set of random effects with $u_s \sim N(0, G_s\sigma_s^2)$, and $\epsilon \sim N(0, \sigma^2)$. The matrices Q and G_s depend only on the the spacings between the time points (see Verbyla *et al* p. 278), where Q is denoted by Δ), Z_s is given by

$$Z_s = Q(Q^tQ)^{-1}, \tag{3}$$

and the smoothing spline parameter λ is related to the mixed model parameters by

$$\lambda = \frac{\sigma^2}{\sigma_s^2}. \tag{4}$$

We transform to a set of independent random effects u_s' with

$$u_s = Lu_s',\ Z_s' = Z_sL, \tag{5}$$

where L is the Choleski decomposition of G_s, $LL' = G_s$, giving

$$y = X_s\beta_s + Z_s'u_s' + \epsilon \tag{6}$$

with $u_s' \sim N(0, I\sigma_s^2)$.

For further information on the historical context of representing smoothing splines as mixed models and related approaches to smoothing see the discussion (Verbyla *et al* p. 276), and the other references.

Fitting smoothing spline models

The package provides 3 functions: the function `smspline()` calculates the matrix Z_s' in (6), (henceforth referred to as the Z-matrix, or simply Z), with columns corresponding to the unique values of the 'time' covariate, except for the two endpoints. The function `smspline.v()` returns a list containing the matrices X, Q, Z, G_s which can be used for further calculations in the smoothing spline framework. A

third function `approx.Z` is used for transforming the spline basis (columns of the Z-matrix).

The function `smspline()` is used to calculate the Z-matrix. A smoothing spline is fitted by including a term (or block) of the form `pdIdent(~ Z - 1)` in the LME random effects structure.

In the first example a smoothing spline is fit to a curve with measurements made on 100 time points. This takes less than a second to fit on a 2 Ghz Linux PC.

Example 1.

```
> library{lmeSplines}
> data(smSplineEx1)
> # variable 'all' for top level grouping
> smSplineEx1$all <- rep(1,nrow(smSplineEx1))
> # setup spline Z-matrix
> smSplineEx1$Zt <- smspline(~ time,
+    data=smSplineEx1)
> fit1s <- lme(y ~ time, data=smSplineEx1,
+    random=list(all=pdIdent(~Zt - 1)))
```

Note:

1. LME has several methods for specifying the covariance structure for random effects which can be confusing to new users. We will use only one: consisting of a named list with each element corresponding to a level of grouping, specified by a formula, such as `~ time`, or a 'pdMat' object such as `pdIdent(~Zt - 1)`, or a list of such objects wrapped up with `pdBlocked` e.g. `pdBlocked(list(~time,pdIdent(~Zt - 1)))`.

2. Where there is no grouping structure, or we wish to use the 'top-level' grouping, consisting of the whole dataset, we introduce the variable `all` consisting of a vector of ones.

3. The structure `smSplineEx1` is a dataframe with 100 rows. We have added the matrix `Zt` which is a matrix with 100 rows.

   ```
   > str(smSplineEx1)
   'data.frame':   100 obs. of  5 variables:
    $ time  : num  1 2 3 4 5 6 7 8 9 10 ...
    $ y     : num  5.80 5.47 4.57 3.65 ...
    $ y.true: num  4.24 4.46 4.68 4.89 ...
    $ all   : num  1 1 1 1 1 1 1 1 1 1 ...
    $ Zt    : num [1:100, 1:98] 1.169 ...
   ```

   Fortunately the R dataframes and model formulae can contain matrix terms. Using `~ Zt` in a model formula is equivalent to specifying every column of `Zt` as a term.

4. The ' -1' in `pdIdent(~Zt - 1)` stops R from adding an intercept term. Thus there are 98 random effects specified, one for every column of `Zt`, i.e. one for every unique time point except the ends.

The LME fitted model is summarised as:-

```
> summary(fit1s)
Linear mixed-effects model fit by REML
 Data: smSplineEx1
  AIC BIC logLik
  282 292   -137
Random effects:
 Formula: ~Zt - 1 | all
 Structure: Multiple of an Identity
           Zt1    Zt2    Zt3    Zt4
StdDev: 0.0163 0.0163 0.0163 0.0163
. . .
          Zt97   Zt98 Residual
StdDev: 0.0163 0.0163     0.86

Fixed effects: y ~ time
            Value Std.Error DF t-value
(Intercept)  6.50     0.173 98    37.5
time         0.04     0.003 98    13.0
            p-value
(Intercept)  <.0001
time         <.0001
 Correlation:
     (Intr)
time -0.868

Standardized Within-Group Residuals:
   Min     Q1    Med     Q3    Max
-2.889 -0.581  0.116  0.659  1.784

Number of Observations: 100
Number of Groups: 1
```

We read off the estimate of $\hat{\sigma_s} = 0.0163$ (standard deviation parameters for `Zt1, Zt2,...`), and the estimate of $\hat{\sigma} = 0.86$ (residual standard deviation), and hence the smoothing parameter is estimated as $\hat{\lambda} = (\hat{\sigma}/\hat{\sigma_s})^2 = (0.0163/0.86)^2 = 3.6 \times 10^{-4}$ representing quite a smooth curve. (Cf Figure 1).

Comparison with B-splines and Flexi

We compare the smoothing spline fit to B-spline models with 3 and 5 knots (`fit1bs3, fit1bs5` respectively) viz:-

```
> fit1bs3 <- update(fit1s, random=list(all=
+    ~bs(time,3)-1))
> fit1bs5 <- update(fit1s, random=list(all=
+    ~bs(time,5)-1))
> anova(ex1.fit1s,ex1.fit1bs3,ex1.fit1bs5)
        Model df AIC BIC logLik L.Rat Pval
fit1s       1  4 282 292   -137
fit1bs3     2 13 298 332   -136  1.86 0.99
fit1bs5     3 24 316 378   -134  4.54 0.95
```

A plot of the smoothing spline fit is shown in Figure 1, with B-spline fits with 3 and 5 knots are shown for comparison. Output from Flexi, a Bayesian smoother (Upsdell 1994,1996; Wheeler and Upsdell 1997) is also shown. The Flexi and smoothing spline curves appear similar with Flexi giving slightly less of a 'hump' near the end. The B-spline models used many more degrees of freedom, took longer to fit (2 sec and 10 sec for 3 and 5 knot points respectively) and gave poorer fits by either AIC of BIC criteria—note the polynomial type wobbles.

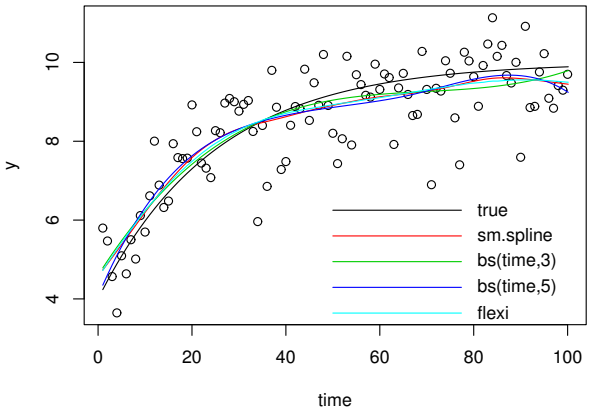

Figure 1: Spline fits. Curves fitted include: the 'true' curve, from which deviations were simulated, a smoothing spline, B-splines with 3,5 knots, and Flexi, a Bayesian smoother.

Aside: Discussion of mixed model formulation in LME, GENSTAT and SAS

By using the variable 'all' we see that models with no grouping structure are a special case of models with a grouping structure. This makes it possible to fit any models, such as commonly fitted in GENSTAT or SAS, where there is no grouping structure (unless specifically requested using a 'repeated' or 'random' statement in SAS), albeit without taking advantage of the efficiency gains possible from LME's treatment of grouping. Model formulae in LME are often more cumbersome than the corresponding GENSTAT REML or SAS PROC MIXED formulae, because LME assumes a general positive definite symmetric covariance matrix for all effects in a formula, while GENSTAT and SAS assume independent sets of random effects for each term in a formula. Suppose that a,b are factors, and we want random terms for a, b, and the interaction, with different variances for different levels of b. In GENSTAT this would be fitted with:

```
vcomp random = a + b + a.b
vstruct[term=b]factor=b; model=diag
vstruct[term=a.b]factor=a,b; model=id,diag
```

In LME, using the levels of a as groups, this would become

```
random=list(
  all=pdBlocked(list(
    pdIdent(~1),pdIdent(~a-1),pdDiag(~b-1))),
  a = pdDiag(~b-1))
```

LME is still lacking some 'pdMat' classes corresponding to the separable (i.e. tensor product) structures in GENSTAT. In many cases, one of the covariance matrices in a tensor product is the identity, and can be modelled using the factor with the identity structure as a grouping factor, as above. If, however, a in the GENSTAT model had a general symmetric covariance matrix V_a and b had a diagonal matrix D_b, the tensor product covariance matrix $V_a \otimes D_b$ for the interaction between a and b would be fitted in GENSTAT with:

```
vcomp random = a + b + a.b
vstruct a; symm
vstruct b; diag
vstruct a.b; symm,diag
```

which currently has no equivalent in LME.

Of course LME has the major advantage that anyone can write a new pdMat class.

Fitting with alternate sets of time points

The fitted curve in Fig. 1 was obtained using fitted values from the NLME fitted model object viz:-

```
lines(smSplineEx1$time,fitted(fit1s),col=2)
```

If the observed data are irregular or more points are needed, predictions at other points can be obtained in principle from the spline model using the matrices X_s, Q, G_s. We have not implemented these calculations, rather, we use a third function approx.Z() for transforming the spline basis to alternate sets of points (e.g. finer or coarser grids) for modelling and/or prediction. For example to fit the model on a coarser grid:-

```
# fit model on coarser grid
times20 <- seq(1,100,length=20)
Zt20 <- smspline(times20)
smSplineEx1$Zt20 <- approx.Z(Zt20,times20,
    smSplineEx1$time)
fit1s20 <- lme(y ~ time, data=smSplineEx1,
    random=list(all=pdIdent(~Zt20 - 1)))
```

Predictions with alternate sets of time points

For model predictions we need to generate a newdata argument for predict.lme() which contains all the terms in the model including the Z-matrix (here Zt20). This means we need a replacement for

Zt20 with values for each time point in the prediction dataset. This is obtained by calling the function approx.Z which uses linear interpolation in the columns of the Z-matrix.

```
# get model predictions on a finer grid
times200 <- seq(1,100,by=0.5)
pred.df <- data.frame(all=rep(1,
    length(times200)),time=times200)
pred.df$Zt20 <- approx.Z(Zt20,times20,
    times200)
yp20.200 <- predict(fit1s20,newdata=pred.df)
```

Note: Linear interpolation is a good approximation here because the spline basis functions (columns of the Z-matrix, i.e. the matrix Z_s' after the transformation (5)) are approximately piecewise linear.

Models with multiple levels of grouping

More general models can contain multiple smoothing spline terms at different levels of grouping and/or with different time covariates.

This is illustrated in the second example. The Spruce dataset contains size measurements taken over time on 13 different trees from each of 4 different plots.

Example 2.

```
data(Spruce)
Spruce$Zday <- smspline(~days, data=Spruce)
spruce.fit2 <- lme(logSize ~ days,
  data=Spruce, random=list(
    all=pdIdent(~Zday - 1),
    plot=pdBlocked(list(
             ~ days, pdIdent(~Zday - 1))),
    Tree = ~1))
```

Fixed or random? Note the inclusion of intercept and slope terms as *random* effects at the plot level. The smoothing spline model (6) specifies *fixed* effects for the slope and intercept. We recommend that the choice of random or fixed effects should be made prior to consideration of fitting a smoothing spline term—whether these should be fixed or random follows the same logic regardless of whether a smoothing spline term is added. We specify random effects since plot effects come from a population of plots, which gives rise to a population of spline curves.

The pdBlocked construction at the plot level gives a variance structure in two mutually independent blocks. The first block contains two random effects (intercept and slope) for each plot with a general symmetric 2×2 covariance matrix, and the second contains 11 independent random effects for the smoothing spline.

Alternate possible models include adding linear terms at the tree level, or linear and spline terms. However, for these data a single overall spline term gave the best fit.

Much more complex models can be fitted. Recently we have fitted models for the spatial distribution of wood density in tree stems. There were measurements made for each annual ring of each of a number of discs taken from log ends (at approximately 5m intervals) on 2 trees per clone for each of 10 clones, for a total sample size of around 2650. There were four levels of grouping: all/clone/tree/disc. The model contained linear terms, smooth trends represented by pdIdent(~Zr-1), pdIdent(~Zh-1), in ring number and height, and deviations represented by e.g. pdIdent(~factor(ring)-1), at each level, as appropriate, additionally with an AR(1) correlation structure for within group errors. The LME model fitted was:

Example 3.

```
lme(density ~ ring + height, random=list(
  all = pdBlocked(list(
          pdIdent(~Zr-1),
          pdIdent(~factor(ring) -1),
          pdIdent(~Zh-1),
          pdIdent(~factor(height)-1))),
  clone = pdBlocked(list(
          ~ ring + height,
          pdIdent(~factor(ring) -1),
          pdIdent(~Zh-1),
          pdIdent(~factor(height)-1))),
  tree  = pdBlocked(list(
          ~ ring + height,
          pdIdent(~factor(ring) -1),
          pdIdent(~Zh-1),
          pdIdent(~factor(height)-1))),
  disc = pdBlocked(~ ring, pdIdent(~Zr -1)),
  cor=corAR1(form= ~ring))
```

The smoothing spline models fit relatively quickly, but with additional within group correlation structures the model fits take a number of hours. Further details will be reported elsewhere.

References

Ball, R. D. (2003) Experiences with LME, smoothing splines, and application to Pinus radiata wood density. New Zealand Statistical Association Annual Conference, Massey University, Palmerston North.

Green, P. J. and Silverman, B. W. (1994) *Nonparametric Regression and Generalized Linear Models*. Chapman and Hall, London.

Guo, W. (2002) Inference in smoothing spline analysis of variance. *J. Royal Stat. Soc. B* 64(4) 887–898.

Kimeldorf, G.S. and Wahba, G. (1970) A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *Ann. Math. Statist.*, 41, 495–502.

Pinheiro, J. C. and Bates, D. M. (2000) *Mixed-Effects Models in S and S-PLUS* Springer-Verlag, New York.

Silverman, B. W. (1985) Some aspects of the spline smoothing approach to non parametric regression curve fitting (with discussion). *J.R. Statist. Soc. B*, 47, 1–52.

Speed, T. P. (1991) Discussion of "That BLUP is a good thing: the estimation of random effects" by G. K. Robinson. *Statist. Sci.*, 6, 42–44.

Upsdell, M. P. (1994) Bayesian smoothers as an extension of nonlinear regression. The New Zealand Statistician, 29, pp.66–81

Upsdell, M. P. (1996) Choosing an Appropriate Covariance Function in Bayesian Smoothing, *Bayesian Statistics 5*, pp.747-756, Oxford University Press, Oxford.

Verbyla, A., Cullis, B. R., Kenward, M. G., and Welham, S. J. (1999) The analysis of designed experiments and longitudinal data by using smoothing splines. Appl. Statist. 48(3) 269–311.

Wahba, G. (1978) Improper priors, spline smoothing and the problem of guarding against model errors in regression. *J.R. Statist. Soc. B*, 40, 364–372.

Wahba, G. (1983) Bayesian "confidence intervals" for the cross-validated smoothing spline. *J.R. Statist. Soc.B* 45, 2133–150.

Wecker, W. P. and Ansley, C. F. (1983) The signal extraction approach to nonlinear regression and spline smoothing. *J. Am. Statist. Ass.*, 78, 81–89.

Wheeler, D. M. and Upsdell, M. P (1997) *Flexi 2.4 reference manual*. New Zealand Pastoral Agriculture Research Institute Ltd, Ruakura, Hamilton, New Zealand. martin.upsdell@agresearch.co.nz www.agresearch.co.nz

Rod Ball,
Forest Research, Rotorua, New Zealand
Rod.Ball@forestresearch.co.nz