ASSIST: A Suite of S functions Implementing Spline smoothing Techniques

Yuedong Wang[*]
University of California, Santa Barbara, USA
and
Chunlei Ke
St. Jude Medical, Sylmar, USA

May 18, 2004

Abstract

We present a suite of user friendly S functions for fitting various smoothing spline models including (a) non-parametric regression models for independent and correlated Gaussian data, and for independent binomial, Poisson and Gamma data; (b) semi-parametric linear mixed-effects models; (c) non-parametric nonlinear regression models; (d) semi-parametric nonlinear regression models; and (e) semi-parametric nonlinear mixed-effects models. The general form of smoothing splines based on reproducing kernel Hilbert spaces is used to model non-parametric functions. Thus these S functions deal with many different situations in a unified fashion. Some well known special cases are polynomial splines, periodic splines, spherical splines, thin-plate splines, l-splines, generalized additive models, smoothing spline ANOVA models, projection pursuit models, multiple index models, varying coefficient models, functional linear models, and self-modeling nonlinear regression models. These non-parametric/semi-parametric linear/nonlinear fixed/mixed models are widely used in practice to analyze data arising in many areas of investigation such as medicine, epidemiology, pharmacokinetics, econometrics and social science. This manual describes technical details behind these S functions and illustrate their applications using several examples.

1 Introduction

Smoothing spline models are widely used in practice as a tool to achieve flexibility. There has been intensive research on its theoretical properties and applications. For references on non-parametric regression using smoothing splines, see Eubank (1988), Wahba (1990), Hastie and Tibshirani (1990), Green and Silverman (1994), Simonoff (1996), and Gu (2002).

As the popularity of building models using splines increases, there is an increasing need for comprehensive and user friendly software. Existing software include `GCVPACK` (Bates, Lindstrom, Wahba and Yandell 1987) for fitting thin plate splines, `RKPACK` (Gu 1989) for fitting general smoothing spline regression models as described in Wahba (1990), and `GRKPACK` (Wang 1997) for fitting generalized smoothing spline regression models to data from exponential families. All three packages were written in Fortran which is inconvenient to use. Some user friendly S (S-plus and R) functions have

[*]This research was supported by NIH Grant R01 GM58533. The authors would like to express their deep appreciation and gratitude to José C. Pinheiro and Douglas M. Bates for answering many technical questions.

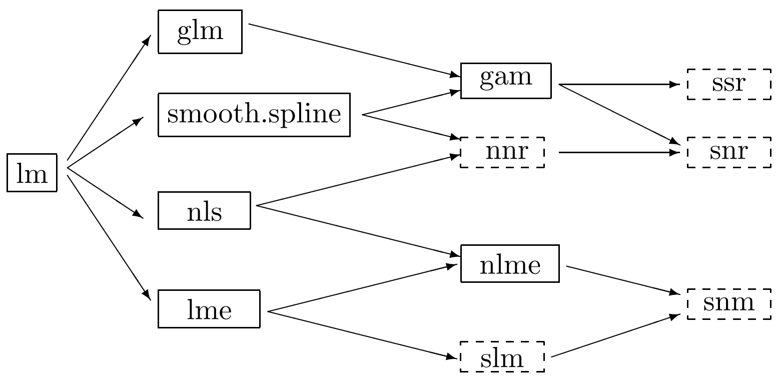


Figure 1: Functions in ASSIST (dashed boxes) and existing S-Plus functions (solid boxes). An arrow represents an extension to a more general model. LM: linear models. GLM: generalized linear models. NLS: nonlinear regression models. LME: linear mixed effects models. GAM: generalized additive models. NNR: nonlinear nonparametric regression models. NLME: nonlinear mixed effects models. SLM: semi-parametric linear mixed effects models. SSR: smoothing spline regression models. SNR: semi-parametric nonlinear regression models. SNM: semi-parametric nonlinear mixed effects models.

been developed recently. For example, the S function `smooth.spline` fits cubic splines; `FIELDS`, a suite of S-plus functions which can be downloaded from *http://www.cgd.ucar.edu/stats/software.shtml*, fits cubic and thin plate splines; `smooth.Lspline`, a S-plus function which can be downloaded from *ftp://ego.psych.mcgill.ca/pub/ramsay/Lspline*, fits L-splines; and `gss`, a suite of R functions which can be downloaded from *cran.r-project.org/src/contrib/PACKAGES.html*, fits general smoothing spline regression models to data from exponential families (Gu 2002).

In this document we describe a suite of S functions, `ASSIST`, with examples to show their usage. The purposes of the `ASSIST` package are to (a) provide a S complement of the `gss` package for fitting general smoothing spline non-parametric regression models to data from exponential families; (b) develop functions for fitting Gaussian data with certain variance and/or covariance structures; (c) develop functions for fitting more complicated models such as semi-parametric linear mixed-effects models, non-parametric nonlinear regression models, semi-parametric nonlinear regression models, and semi-parametric nonlinear mixed-effects models; and (d) provide inference tools for some simple models. We adopt notations in Wahba (1990).

Figure 1 shows how the functions in ASSIST generalize existing S-Plus functions.

Basic knowledge of reproducing kernel Hilbert spaces and general smoothing spline models as described in the first two chapters of Wahba (1990) is necessary to fully understand this article. However, this is not required for using our functions to fit simple smoothing spline models.

We review the general smoothing spline regression model and describe the corresponding S function `ssr` in Section 2. We review the semi-parametric linear mixed-effects model and describe the corresponding S function `slm` in Section 3. We review the non-parametric nonlinear regression model and describe the corresponding S function `nnr` in Section 4. We review the semi-parametric nonlinear regression model and describe the corresponding S function `snr` in Section 5. We review the semi-parametric nonlinear mixed-effects model and describe the corresponding S function `snm` in Section 6. We illustrate how to use these functions with several real data sets in Section 7. Computational concerns and tips are discussed in Section 8. Finally, in Section 9, we conclude with discussions on further work.

2 Smoothing Spline Regression Models

In Section 2.1, we introduce the `ssr` function for general smoothing spline regression models with one variable. In sections 2.2 to 2.4, we show how to use `ssr` to fit more complicated models such as partial spline models, smoothing spline ANOVA (SS ANOVA) models for correlated observations and/or observations with unequal variances, and SS ANOVA models for data from exponential families.

2.1 General Smoothing Spline Regression Models With One Variable

2.1.1 Model and Estimation

The general smoothing spline regression (SSR) model with one variable assumes that (Wahba 1990)

$$y_i = L_i f + \epsilon_i, \qquad i = 1, \cdots, n, \tag{1}$$

where y_i's are univariate responses; f is an unknown function of an independent variable t with t belonging to an arbitrary domain $\mathcal{T}$ and $f \in \mathcal{H}$, a given Reproducing Kernel Hilbert Space (RKHS); $L_i' s$ are bounded linear functionals on $\mathcal{H}$; and ϵ_i's are random errors with $\epsilon_i \overset{iid}{\sim} \mathrm{N}(0, \sigma^2)$. Note that t may be a vector. For most applications, L_i's are evaluation functionals at design points: $L_i f = f(t_i)$.

Suppose that

$$\mathcal{H} = \mathcal{H}_0 \oplus \mathcal{H}_1, \tag{2}$$

where $\mathcal{H}_0$ is a finite dimensional space with basis functions $\phi_1(t), \cdots, \phi_M(t)$, and $\mathcal{H}_1$ is a RKHS with reproducing kernel $R_1(s,t)$. See Aronszajn (1950) and Wahba (1990) for more information about RKHS. The estimate of f, $\hat{f}_\lambda$, is the minimizer of the following penalized least squares

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - L_i f)^2 + \lambda ||P_1 f||^2, \tag{3}$$

where P_1 is the orthogonal projection operator of f onto $\mathcal{H}_1$ in $\mathcal{H}$, and λ is a smoothing parameter controlling the balance between goodness-of-fit measured by the least squares and departure from the null space $\mathcal{H}_0$ measured by $||P_1 f||^2$. Note that functions in $\mathcal{H}_0$ are not penalized.

Let $\boldsymbol{y} = (y_1, \cdots, y_n)^T$. Define $\xi_i(t) = L_{i(\cdot)} R_1(t, \cdot)$, $T_{n\times M} = \{L_i \phi_v\}_{i=1}^{n}{}_{v=1}^{M}$ and $\Sigma = \{<\xi_i, \xi_j>\}_{i,j=1}^{n}$. Given λ, the solution to (3) has the form (Wahba, 1990)

$$\hat{f}_\lambda(t) = \sum_{i=1}^{M} d_i \phi_i(t) + \sum_{j=1}^{n} c_j \xi_j(t), \tag{4}$$

where the coefficients $\boldsymbol{d} = (d_1, \cdots, d_M)^T$ and $\boldsymbol{c} = (c_1, \cdots, c_n)^T$ are solutions to

$$\begin{aligned} (\Sigma + n\lambda I)\boldsymbol{c} + T\boldsymbol{d} &= \boldsymbol{y}, \\ T^T \boldsymbol{c} &= 0. \end{aligned} \tag{5}$$

The Fortran subroutine `dsidr.r` in `RKPACK` was developed to solve equations (5) (Gu 1989). In our `ASSIST` package, the S function `dsidr` serves as an intermediate interface between S and the driver `dsidr.r`.

2.1.2 The `ssr` Function

The S function for fitting a SSR model is `ssr`. A typical call is

```
    ssr(formula, rk, data)
```

Only the first two arguments, `formula` and `rk`, are required. Together they describe the response variable and the model space. `formula`, a statistical formula as in `lm`, lists the response variable on the left hand side and the bases $\phi_1(t), \cdots, \phi_M(t)$ of $\mathcal{H}_0$ on the right hand side of an operator `~`. `rk` is an expression that specifies the reproducing kernel R_1. Expressions of `rk` for commonly used reproducing kernels are available in our library. These expressions will be discussed in the following sections. Users can easily add their own expressions of reproducing kernels.

The optional argument `data` specifies the data frame. Other optional arguments will be discussed in the following sections.

`ssr` returns an object of class "ssr" with many variables representing the fit to the specified smoothing spline model. Detailed information about this object can be found in the help file on `ssr.object`.

2.1.3 Some Special Spline Models

Example 1. *Polynomial Spline.* For the polynomial spline of order m on $\mathcal{T} = [a, b]$, the model space is

$$\mathcal{H} = W_m([a,b]) = \{f : \ f, f', \cdots, f^{(m-1)} \text{absolutely continuous}, \int_a^b (f^{(m)})^2 dt < \infty\}. \tag{6}$$

Let $\mathcal{T} = [0, 1]$. Define an inner product on $W_m([0, 1])$ as

$$(f, g) = \sum_{i=0}^{m-1} \int_0^1 (f^{(i)} dt) \int_0^1 (g^{(i)} dt) + \int_0^1 f^{(m)} g^{(m)} dt.$$

Then we have

$$\begin{aligned} \mathcal{H}_0 &= span\{k_0(t), k_1(t), \cdots, k_{m-1}(t)\}, \\ R_1(s,t) &= k_m(s)k_m(t) + (-1)^{m-1} k_{2m}(s-t), \\ ||P_1 f||^2 &= \int_0^1 (f^{(m)})^2 dt, \end{aligned} \tag{7}$$

where $k_r(t) = B_r(t)/r!$ and B_r's are Bernoulli polynomials (Craven and Wahba 1979).

Let $\mathcal{T} = [a, b]$. Define inner product on $W_m([a, b])$ as

$$(f, g) = \sum_{i=0}^{m-1} f^{(i)}(a) g^{(i)}(a) + \int_a^b f^{(m)} g^{(m)} dt.$$

Then (Wahba 1990)

$$\begin{aligned} \mathcal{H}_0 &= span\{1, (t-a), \cdots, (t-a)^{m-1}\}, \\ R_1(s,t) &= \int_a^b \frac{(s-u)_+^{m-1}}{(m-1)!} \frac{(t-u)_+^{m-1}}{(m-1)!} du. \end{aligned} \tag{8}$$

Table 1 lists statements inside `ssr` for four simple polynomial splines. We assume variables $k1$, $k2$ and $k3$ have been calculated based on Bernoulli polynomials. All expressions of the reproducing kernels in this table are available in our library. Note that the domain under construction (7) is restricted to $[0,1]$ while the domain under construction (8) is an arbitrary interval $[a,b]$. Thus one needs to transform a variable into $[0,1]$ before using `rk` functions in the third column. The `rk` functions in the fourth column assume the domain $[0,T]$ for any fixed $T>0$. One can calculate the reproducing kernel on $[a,b]$ by a translation, for example $cubic2(t-a)$.

Table 1: Statements for fitting simple polynomial splines.

m	splines	under construction (7)	under construction (8)
1	linear	`y~1, rk=linear(t)`	`y~1, rk=linear2(t)`
2	cubic	`y~k1, rk=cubic(t)`	`y~t, rk=cubic2(t)`
3	quintic	`y~k1+k2, rk=quintic(t)`	`y~t+t**2, rk=quintic2(t)`
4	septic	`y~k1+k2+k3, rk=septic(t)`	`y~t+t**2+t**3, rk=septic2(t)`

Example 2. *Stein Estimate.* A James-Stein shrinkage estimate can be regarded as the solution to (3) with $\mathcal{T}=\{1,2,\cdots,K\}$, $\mathcal{H}=R^K$ and $L_i f = f(t_i)$ (Gu 2002).

For shrinkage toward a constant,

$$\begin{aligned} \mathcal{H}_0 &= \text{span}\{1\}, \\ R_1(s,t) &= I_{[s=t]} - 1/K, \\ ||P_1 f||^2 &= \sum_{i=1}^{K}[f(i) - \sum_{j=1}^{K} f(j)/K]^2. \end{aligned} \tag{9}$$

For shrinkage toward zero,

$$\begin{aligned} \mathcal{H}_0 &= \text{empty set}, \\ R_1(s,t) &= I_{[s=t]}, \\ ||P_1 f||^2 &= \sum_{i=1}^{K} f^2(i). \end{aligned} \tag{10}$$

The corresponding `rk` expressions for shrinkage toward a constant and shrinkage toward zero are `shrink1(t)` and `shrink0(t)` respectively. We use these shrinkage methods to model discrete covariates in smoothing spline ANOVA models as will be discussed later.

Example 3. *Periodic Spline.* For the m-th order periodic spline on $\mathcal{T}=[0,1]$ (Wahba 1990),

$$\begin{aligned} \mathcal{H} &= W_m(per) \\ &= \{f : f^{(j)} \text{ abs. cont.},\ f^{(j)}(0) = f^{(j)}(1),\ j=0,\cdots,m-1, \int_0^1 (f^{(m)})^2 dt < \infty\}, \\ \mathcal{H}_0 &= span\{1\}, \\ R_1(s,t) &= \sum_{v=1}^{\infty} \frac{2}{(2\pi v)^{2m}} \cos 2\pi v(s-t), \\ ||P_1 f||^2 &= \int_0^1 (f^{(m)})^2 dt. \end{aligned}$$

The function `periodic` in our library calculates R_1 evaluated at specified points. The order m is specified by the argument `order` with default `order=2`. For example, one may fit a cubic periodic spline ($m=2$) by

```
    ssr(y~1, rk=periodic(t))
```

Example 4. *Thin plate spline* (TPS) (Wahba 1990). For a TPS of order m on $\mathcal{T} = R^d$ with $2m-d>0$,

$$\begin{aligned} \mathcal{H} &= \{f: \ J_m^d(f) < \infty\}, \\ \mathcal{H}_0 &= \{f: \ J_m^d(f) = 0\}, \\ ||P_1 f|| &= J_m^d(f), \end{aligned} \tag{11}$$

where

$$J_m^d(f) = \sum_{\alpha_1+\cdots+\alpha_d=m} \frac{m!}{\alpha_1!\cdots\alpha_d!}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}\left(\frac{\partial^m f}{\partial t_1^{\alpha_1}\cdots\partial t_d^{\alpha_d}}\right)^2\prod_j dt_j.$$

A pseudo reproducing kernel (conditional positive definite rather than positive definite) for $\mathcal{H}_1$ is $R_1(s,t) = E(|s-t|)$, where $|s-t|$ is the Euclidean distance, and

$$E(u) = \begin{cases} \dfrac{(-1)^{d/2+1+m}}{2^{2m-1}\pi^{d/2}(m-1)!(m-d/2)!}|u|^{2m-d}ln|u|, & d \ \ even, \\ \dfrac{\Gamma(d/2-m)}{2^{2m}\pi^{d/2}(m-1)!}|u|^{2m-d}, & d \ \ odd. \end{cases}$$

The function `tp.psuedo` in our library calculates this pseudo kernel. The argument `order` of this function specifies m with default `order=2`. For example, for $d=2$ and $m=2$, one may fit the TSP model by

```
    ssr(y~t1+t2, rk=tp.pseudo(list(t1,t2)))
```

The true kernel discussed in Gu and Wahba (1993a) is calculated by the function `tp`. It takes longer to compute the true kernel and is only necessary for calculating posterior variances.

Example 5. *Spline on the sphere* is an extension of both the periodic spline defined on the unit circle and the TPS on R^2. Let the domain be $\mathcal{T} = \mathcal{S}$, where $\mathcal{S}$ is the unit sphere. Any point t on $\mathcal{S}$ can be represented as $t = (\theta, \phi)$, where θ ($0 \le \theta \le 2\pi$) is the longitude and ϕ ($-\pi/2 \le \phi \le \pi/2$) is the latitude. Define

$$J(f) = \int_{t\in\mathcal{S}} (\Delta^{m/2} f)^2 dt,$$

where Δf is the surface Laplacian on the unit sphere

$$\Delta f = \frac{1}{\cos^2\phi} f_{\theta\theta} + \frac{1}{\cos\phi}(\cos\phi f_\phi)_\phi.$$

The model space $\mathcal{H} = \mathcal{H}_0 \oplus \mathcal{H}_1$, where $\mathcal{H}_0 = \text{span}\{1\}$ and

$$\mathcal{H}_1 = \{f \in \mathcal{L}_2(\mathcal{S}): \ \ J(f) < \infty\}.$$

$\mathcal{H}$ is a RKHS with RK $R(s,t) = 1 + R_1(s,t)$, where

$$R_1(s,t) = \sum_{i=1}^{\infty} \frac{2i+1}{4\pi}\frac{1}{[i(i+1)]^m} L_i(\cos\gamma(s,t))$$

is the rk of $\mathcal{H}_1$, $\gamma(s,t)$ is the angle between s and t, and L_i's are the Legendre polynomials. $||P_1 f|| = J(f)$. R_1 is in the form of an infinite series which is inconvenient to compute. Closed form expressions are only available for $m = 2$ and $m = 3$. Wahba (1981) proposed replacing J by a topologically equivalent norm Q under which closed form of rk's can be derived. See Wahba (1981), Wahba (1982), Wahba (1990) and Wahba and Luo (1996) for more details. The function `sphere` in our library calculates R_1 under the norm Q for $2 \leq m \leq 6$. The argument `order` of this function specifies m with default `order=2`. For example, for $m = 3$, one may fit a spline on the sphere by

```
    ssr(y~t1+t2, rk=sphere(cbind(t1,t2),order=3))
```

where $t1$ and $t2$ are longitude and latitude respectively.

Example 6. *L-spline*. The penalty term, $||P_1 f||$, is usually used to penalize the roughness of the function f. However, sometimes it is advantageous to use other forms of penalty. For example, prior information may be incorporated or even estimated by a penalty to the departure of the non-parametric function f from a specific parametric model (Wahba 1990, Heckman and Ramsay 2000). Let L be a linear differential operator $L = D^m + \sum_{j=0}^{m-1} \omega_j D^j$, where D^j denotes the jth derivative operator and the ω's are continuous real-valued weight functions. The spline estimate with the penalty $||P_1 f||^2 = \int_a^b (Lf(t))^2 dt$ is called an L-spline. See Heckman and Ramsay (2000) and Gu (2002) for more details about the L-spline. The `lspline` function in our library calculates reproducing kernels for the following four L-spline models.

(a) Suppose that $\mathcal{T} = [0,1]$. If prior knowledge suggests that f is close to a linear combination of $\sin 2\pi t$ and $\cos 2\pi t$, one may use $\mathcal{H} = W_2(per) \ominus \{1\}$, and $L = D^2 + (2\pi)^2$. Then

$$\begin{aligned} \mathcal{H}_0 &= \text{span}\{\sin 2\pi t, \ \cos 2\pi t\}, \\ R_1(s,t) &= \sum_{\nu=2}^{\infty} \frac{2}{(2\pi)^4(1-\nu^2)^2} \cos 2\pi\nu(s-t). \end{aligned}$$

The statement for fitting such a model is

```
    ssr(y~sin(2*pi*t)+cos(2*pi*t)-1, rk=lspline(t,type="sine0"))
```

If we want to include the constant in the model space, then $\mathcal{H} = W_3(per)$, $L = D[D^2 + (2\pi)^2]$,

$$\begin{aligned} \mathcal{H}_0 &= \text{span}\{1, \ \sin 2\pi t, \ \cos 2\pi t\}, \\ R_1(s,t) &= \sum_{\nu=2}^{\infty} \frac{2}{(2\pi)^6 \nu^2 (1-\nu^2)^2} \cos 2\pi\nu(s-t). \end{aligned}$$

The statement for fitting such a model is

```
    ssr(y~sin(2*pi*t)+cos(2*pi*t), rk=lspline(t,type="sine1"))
```

(b) Suppose that $\mathcal{T} = [0,T]$. If prior knowledge suggests that f is close to a linear combination of 1 and $\exp(-t)$, one may use $\mathcal{H} = W_2([0,T])$, and $L = D^2 + D$. Then

$$\begin{aligned} \mathcal{H}_0 &= \text{span}\{1, \ \exp(-t)\}, \\ R_1(s,t) &= \min(s,t) + e^{-t} + e^{-s} - e^{\min(s,t)-s} - e^{\min(s,t)-t} - \frac{e^{-(s+t)}}{2} + \frac{e^{2\min(s,t)-s-t}}{2}. \end{aligned} \quad (12)$$

The statement for fitting such a model is

```
    ssr(y~exp(-t), rk=lspline(t,type="exp"))
```

(c) Suppose that $\mathcal{T} = [0,T]$. If prior knowledge suggests that f is close to the logistic function $\exp(t)/(1+\exp(t))$, one may use $\mathcal{H} = W_1([0,T])$, and $L = D - \frac{1}{1+e^t} I$. Then

$$\mathcal{H}_0 = \text{span}\{\exp(t)/(1+\exp(t))\},$$

$$R_1(s,t) \quad = \quad \frac{e^{s+t}}{(1+e^s)(1+e^t)}[\min(s,t) - 2e^{-\min(s,t)} - \frac{1}{2}e^{-2\min(s,t)} + \frac{5}{2}].$$

The statement for fitting such a model is

```
    ssr(y~I(exp(t)/(1+exp(t)))-1, rk=lspline(t,type="logit"))
```

(d) Suppose that $\mathcal{T} = [0, T]$. If prior knowledge suggests that f is close to a linear combination of 1, t, $\sin 2\pi t$ and $\cos 2\pi t$, one may use $\mathcal{H} = W_4([0,T])$, and $L = D^4 + D^2$. Then

$$\mathcal{H}_0 \quad = \quad \text{span}\{1,\ t,\ \sin(t),\ \cos(t)\},$$

$$R_1(s,t) \quad = \quad \begin{cases} - & \frac{t^3}{6} + \frac{st^2}{2} + (t-s) + s\cos(t) - \sin(t) + t\cos(s) - \sin(s) \\ & +\frac{t}{2}\cos(t-s) - \frac{5}{4}\sin(t-s) - \frac{1}{4}\sin(s+t), \qquad t < s, \\ - & \frac{s^3}{6} + \frac{s^2t}{2} + (s-t) + t\cos(s) - \sin(s) + s\cos(t) - \sin(t) \\ & +\frac{s}{2}\cos(s-t) - \frac{5}{4}\sin(s-t) - \frac{1}{4}\sin(s+t), \qquad s \le t. \end{cases}$$

The statement is

```
    ssr(y~t+cos(t)+sin(t), rk=lspline(t,type="linSinCos"))
```

2.1.4 The Smoothing Parameter

The choice of the smoothing parameter λ is critical to the performance of a spline estimate. Several data-adaptive methods have been successfully used in practice (Wahba 1990). The following three methods, Generalized Cross Validation (GCV), Generalized Maximum Likelihood (GML) and Unbiased Risk (UBR), were implemented in `RKPACK`, and are available in S functions `dsidr` and `ssr`. Denote $A(\lambda)$ as the hat matrix such that

$$(L_1\hat{f}_\lambda, \cdots, L_n\hat{f}_\lambda)^T = A(\lambda)\boldsymbol{y}.$$

The GCV, GML and UBR methods estimate λ as the minimizers of the following GCV function

$$\text{GCV}(\lambda) = \frac{\frac{1}{n}||(I - A(\lambda))\boldsymbol{y}||^2}{[\frac{1}{n}\text{tr}(I - A(\lambda))]^2},$$

GML function

$$\text{GML}(\lambda) = \frac{\boldsymbol{y}^T(I - A(\lambda))\boldsymbol{y}}{[\det^+((I - A(\lambda)))]^{1/(n-M)}},$$

where $\det^+$ represents the product of the nonzero eigenvalues, and UBR function

$$U(\lambda) = \frac{1}{n}||(I - A(\lambda))\boldsymbol{y}||^2 + \frac{2\sigma^2}{n}\text{tr}A(\lambda),$$

respectively.

The GCV method may lead to interpolation when the sample size is small (Wahba and Wang 1993). The GML method is very stable. For moderate sample sizes, the performance of the GCV and GML methods are comparable. For large sample sizes, the GCV method performs better then the GML method. In our S function `ssr`, an option `spar` is provided for specifying one of these three methods. `spar=‘‘v’’`, `spar=‘‘m’’` and `spar=‘‘u’’` correspond to the GCV, GML and UBR methods respectively with GCV as the default. For example, fitting a cubic spline with the GML choice of the smoothing parameter can be accomplished by

```
    ssr(y~t, rk=cubic(t), spar=‘‘m’’)
```
An estimate of σ^2 is needed for the UBR method. It can be specified with the argument `varht`. For example, the following statement uses UBR to choose the smoothing parameter with $\sigma^2 = 10$
```
    ssr(y~t, rk=cubic(t), spar=‘‘u’’, varht=10)
```
Several methods may be used to derive an estimate of σ^2 (Rice 1984, Gasser, Sroka and Jennen-Steinmetz 1986, Dette, Munk and Wagner 1998, Hall, Kay and Titterington 1990, Donoho and Johnston 1994).

2.1.5 Inferences

Consider the following Bayesian model

$$y_i = L_i F + \epsilon_i, \quad i = 1, \cdots, n,$$

with prior for F as

$$F(t) = \sum_{\nu=1}^{M} d_\nu \phi_\nu(t) + \tau^{1/2} X(t), \quad t \in \mathcal{T},$$

where $\boldsymbol{d} = (d_1, \cdots, d_M)^T \sim N(0, aI)$, a and τ are positive constants, and $X(t)$ is a zero mean Gaussian stochastic process independent of $\boldsymbol{d}$ with covariance $EX(s)X(t) = R_1(s,t)$. Wahba (1978) showed that $\lim_{a\to\infty} \mathrm{E}(F(t)|\boldsymbol{y}) = \hat{f}_\lambda(t)$ with $\lambda = \sigma^2/(n\tau)$. Formulae for computing posterior means and variances were provided in Gu and Wahba (1993b). Posterior variances can be used to construct confidence intervals for $\hat{f}_\lambda(t)$:

$$\hat{f}_\lambda(t) \pm z_{1-\alpha/2} \sqrt{\mathrm{Var}(\hat{f}_\lambda(t)|\boldsymbol{y})}, \tag{13}$$

where $z_{1-\alpha/2}$ is the $1-\alpha/2$ quantile of a standard normal distribution (Wahba 1990). The intervals defined in (13) are referred to as the Bayesian confidence intervals (Wahba 1983). These Bayesian confidence intervals are not point-wise confidence intervals. Rather, they provide across-the-function coverage (Nychka 1988, Wang and Wahba 1995).

Often one needs to test

$$H_0: \ f \in \mathcal{H}_0 \quad \text{against} \quad H_1: \ f \notin \mathcal{H}_0.$$

This hypothesis is equivalent to $P_1 f = 0$ or $\lambda = \infty$. Three tests were considered in Wahba (1990): locally most powerful (LMP), GCV and GML tests. Let

$$T = (Q_1 \ \ Q_2) \begin{pmatrix} R \\ 0 \end{pmatrix}$$

be the QR decomposition of T, and UDU^T be the eigenvalue decomposition of $Q_2^T \Sigma Q_2$ with eigenvalues $\lambda_{vn}, \ v = 1, \cdots, n-M$. Let $\boldsymbol{z} = (z_1, \cdots, z_{n-M})^T = U^T Q_2^T \boldsymbol{y}$. Then the test statistics for LMP, GML and GCV tests are

$$t_{\text{LMP approx}} = \sum_{v=1}^{n-M} \lambda_{vn} z_v^2 / \sum_{v=1}^{n-M} z_v^2,$$

$$t_{\text{GCV}} = \frac{\sum_{v=1}^{n-M} (z_v^2/(1+\hat{\gamma}\lambda_{vn})^2)}{\sum_{v=1}^{n-M} (1/(1+\hat{\gamma}\lambda_{vn})^2)} \times \frac{1}{\sum_{v=1}^{n-M} z_v^2},$$

and

$$t_{\rm GML} = \frac{\sum_{v=1}^{n-M}(z_v^2/(1+\hat{\gamma}\lambda_{vn})^2)}{\prod_{v=1}^{n-M}(1+\hat{\gamma}\lambda_{vn})^{-1/(n-M)}} \times \frac{1}{\sum_{v=1}^{n-M} z_v^2},$$

where $\hat{\gamma} = 1/n\hat{\lambda}$. It can be shown that under the corresponding Bayesian model, the LMP test is the score test and the GML test is the likelihood ratio test. Furthermore, the GCV test is closely related to the F-test based on the extra sum of squares principle (Liu and Wang 2004). Usually the p-values cannot be calculated analytically because the null distributions under H_0 are unknown. Standard theory for likelihood ratio tests does not apply because the parameter is on the boundary under the null hypothesis. The non-standard asymptotic theory developed by Self and Liang (1987) does not apply either because of the lack of replicated observations. Monte Carlo method can be used to approximate p-values. However, they are usually computational intensive since the smoothing parameter needs to be estimated for each Monte Carlo sample. In the current version, through the utility function `anova`, Monte Carlo p-values are calculated with fixed smoothing parameters. The Monte Carlo sample size is specified by the option `simu.size`. `tt anova` also provides the approximate p-values of the GML tests based on a mixture of two χ^2 distributions (Self and Liang 1987) even though they tend to be conservative. Methods developed in Liu and Wang (2004) and Liu, Meiring and Wang (2004) will be implemented in the future.

An alternative approach to visually check above hypothesis is to plot the projection of $\hat{f}$ onto $\mathcal{H}_0$ together with its Bayesian confidence intervals. When $\mathcal{H}_0$ is true, most parts of the zero function should be inside these confidence intervals. See Section 7 for examples. Two utility functions, `predict.ssr` and `plot.bCI`, are available to compute posterior means, standard deviations and plot fits with Bayesian confidence intervals. See help files of `predict.ssr` and `plot.bCI` for more details.

2.2 Partial Spline Models

The linear partial spline model assumes that (Wahba 1990)

$$y_i = \beta_1 x_{1i} + \cdots + \beta_d x_{di} + f(t_i) + \epsilon_i, \qquad i = 1, \cdots, n, \tag{14}$$

where the first part is a linear model of covariates $x_1, \cdots, x_d$, and $f \in \mathcal{H}$ as in (1). Note that an SS ANOVA model discussed in the next section can also be used for f when t is multivariate. Partial spline models provide a tool to model multiple covariates when the relationship is unknown for only a few variables. Note that some x_k's may be functions of t. For example, $x = (t - t_0)_+^q$ allows a jump in the qth derivative at t_0.

Let X be the design matrix of $x_1, \cdots, x_d$: $X^T = \{x_{ji}\}_{j=1}^{d}{}_{i=1}^{n}$, and $S = (X\ T)$. If S is of full column rank, the estimate of f has the same representation as in (4). Furthermore, coefficients $(\beta_1, \cdots, \beta_d, \boldsymbol{d}^T)^T$ and $\boldsymbol{c}$ are solutions to equations (5) with T replaced by S.

The linear model for $x_1, \cdots, x_d$ in (14) can be easily specified by adding these covariates to the right hand side of `formula`. For example, supposing $d = 3$ and a cubic spline for f, we can fit model (14) by

```
    ssr(y~x1+x2+x3+t, rk=cubic(t))
```

2.3 Smoothing Spline ANOVA Models

Consider model (1) with f being a function of multivariate variables $t_1, \cdots, t_d$. Each variable t_k itself could be a vector. Assume that $t_k \in \mathcal{T}_k$, where $\mathcal{H}_k$ is an arbitrary domain. Then f is a function of $\boldsymbol{t} = (t_1, \cdots, t_d) \in \mathcal{T} = \mathcal{T}_1 \otimes \cdots \otimes \mathcal{T}_d$.

Suppose that we want to use the RKHS $\mathcal{H}^{(k)} = \{1^{(k)}\} \oplus \mathcal{H}_1^{(k)}$ to model the effect of variable t_k. Denote P^k as the projection operator onto $\{1^{(k)}\}$ in $\mathcal{H}^{(k)}$. Then

$$\begin{aligned} f &= [\prod_{k=1}^{d}(P^k + I - P^k)]f \\ &= \sum_{B\subseteq\{1,\cdots,d\}} [\prod_{k\in B}(I-P^k)\prod_{k\in B^c}P^k f] \\ &= \mu + \sum_{k=1}^{d} f_k(t_k) + \sum_{k<l} f_{kl}(t_k,t_l) + \cdots + f_{1,\cdots,d}(t_1,\cdots,t_d), \end{aligned} \tag{15}$$

where elements f_k's are *main effects*, f_{jk}'s are *two factor interactions*, and so on.

(15) is just the simplest form of the so-called SS ANOVA decomposition. The classical ANOVA models are special cases with all variables being discrete. In general, suppose that we want to use the RKHS $\mathcal{H}^{(k)} = \mathcal{H}_0^{(k)} \oplus \mathcal{H}_1^{(k)} = \{\phi_1^{(k)}\} \oplus \cdots \oplus \{\phi_{m_k}^{(k)}\} \oplus \mathcal{H}_1^{(k)}$ to model the effect of variable t_k where $\phi_j^{(k)}$'s are basis functions for the null space $\mathcal{H}_0^{(k)}$. Denote P_j^k as the projection operator onto $\{\phi_j^{(k)}\}$ in $\mathcal{H}^{(k)}$. Then expansion of the following equation

$$f = [\prod_{k=1}^{d}(P_1^k + \cdots + P_{m_k}^k + I - \sum_{j=1}^{m_k} P_j^k)]f \tag{16}$$

provides the general form of SS ANOVA decomposition. (16) decomposes f in the tensor product space $\mathcal{H}^{(1)} \otimes \cdots \otimes \mathcal{H}^{(k)}$ into orthogonal and interpretable components. Which decomposition to use depends on prior knowledge and the purpose of a study. It is more precise to think of SS ANOVA decompositions as a powerful technique rather than as some specific models. See Wahba (1990), Gu and Wahba (1993a), Gu and Wahba (1993b), Wahba, Wang, Gu, Klein and Klein (1995), Wang, Wahba, Chappell and Gu (1995), Wang, Wahba, Gu, Klein and Klein (1997), Wang (1998a), Wang and Wahba (1998) and references there for more details. Similar to the classical ANOVA, usually a model space is a subspace containing lower order components. After a model is chosen, we can regroup and write the model space as

$$\mathcal{H} = \mathcal{H}_0 \oplus \sum_{k=1}^{p} \mathcal{H}_k, \tag{17}$$

where $\mathcal{H}_0 = \text{span}\{\phi_1(\boldsymbol{t}),\cdots,\phi_M(\boldsymbol{t})\}$ is a finite dimensional space containing functions which are not going to be penalized, and $\mathcal{H}_k$ is a RKHS with reproducing kernel $R_k(\boldsymbol{s},\boldsymbol{t})$. The estimate of f is the minimizer of

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - L_i f)^2 + \lambda \sum_{k=1}^{p} \theta_k^{-1}||P_k f||^2, \tag{18}$$

where P_k is the orthogonal projection of f onto $\mathcal{H}_k$ in $\mathcal{H}$. Let $\xi_{ki}(\boldsymbol{t}) = P_1 L_{i(.)} R_k(\boldsymbol{t},.)$ and $\Sigma_k = \{<\xi_{ki},\xi_{kj}>\}_{i,j=1}^{n}$. The solution to (18) is

$$\hat{f}(\boldsymbol{t}) = \sum_{i=1}^{M} d_i\phi_i(\boldsymbol{t}) + \sum_{j=1}^{n} c_j(\sum_{k=1}^{p}\theta_k\xi_{kj}(\boldsymbol{t})), \tag{19}$$

where $\boldsymbol{c}$ and $\boldsymbol{d}$ are solutions to (5) with Σ replaced by $\sum_{k=1}^{p}\theta_k\Sigma_k$. Smoothing parameters $\lambda/\theta_1,\cdots,\lambda/\theta_p$ can be estimated similarly using GCV, GML and UBR methods (Wahba 1990). The Fortran subroutine `dmudr.r` in `RKPACK` was developed to solve equations (5) and estimate the smoothing parameters for $p \geq 1$. In our `ASSIST` package, the function `dmudr` serves as an intermediate interface between S and the driver `dmudr.r`.

`ssr` can also be used to fit SS ANOVA models. Basis functions $\phi_1,\cdots,\phi_M$ can be specified as before using the `formula` argument. Reproducing kernels $R_1(\boldsymbol{s},\boldsymbol{t}),\cdots,R_p(\boldsymbol{s},\boldsymbol{t})$ can be specified using the argument `rk` as a *list* of expressions.

Example 7 Consider $d=2$, $t_1 \in \mathcal{T}_1 = \{1,\cdots,K\}$ and $t_2 \in \mathcal{T}_2 = [0,1]$. Functional data are a typical example of this case (Ramsay and Silverman 1997). Suppose that we want to model the t_1 effect using a one-way ANOVA effect model with $\mathcal{H}^{(1)} = R^K = \{1\} \oplus \{g: \sum_{t_1=1}^{K} g(t_1)=0\}$, and the t_2 effect using a linear spline $\mathcal{H}^{(2)} = W_1([0,1]) = \{1\} \oplus \{g \in W_1([0,1]): \int_0^1 g(t_2)dt_2 = 0\}$. Define two projection operators

$$\begin{aligned} P_1^1 f &= \sum_{t_1=1}^{K} f(t_1,t_2)/K, \\ P_1^2 f &= \int_0^1 f(t_1,t_2)dt_2. \end{aligned}$$

Then (15) leads to

$$f(t_1,t_2) = \mu + s_1(t_1) + s_2(t_2) + ss_{12}(t_1,t_2). \qquad (20)$$

We have

$$\begin{aligned} \mathcal{H}_0 &= \text{span}\{1\}, \\ R_1((s_1,s_2),(t_1,t_2)) &= I_{[s_1=t_1]} - 1/K, \\ R_2((s_1,s_2),(t_1,t_2)) &= k_1(s_2)k_1(t_2) + k_2(s_2-t_2), \\ R_3((s_1,s_2),(t_1,t_2)) &= [I_{[s_1=t_1]} - 1/K][(k_1(s_2)k_1(t_2) + k_2(s_2-t_2)]. \end{aligned}$$

Construction (7) of a polynomial spline is used. Construction (8) may also be used to derive a similar SS ANOVA decomposition. SS ANOVA model (20) can be fitted by

```
    ssr(y~1, rk=list(shrink1(t1),linear(t2),rk.prod(shrink1(t1),linear(t2))))
```

where `rk.prod` is a function in our library calculating the product of two reproducing kernels.

Suppose that instead of a linear spline, we want to model t_2 effect using a cubic spline

$$\mathcal{H}^{(2)} = W_2([0,1]) = \{1\} \oplus \{k_1\} \oplus \{g \in W_2([0,1]): \int_0^1 g(t_2)dt_2 = \int_0^1 g'(t_2)dt_2 = 0\}.$$

Define an additional projection operator

$$P_2^2 f = [\int_0^1 (\partial f/\partial t_2)dt_2]k_1.$$

Then (16) leads to

$$f = \mu + s_1(t_1) + \beta k_1(t_2) + s_2(t_2) + sl_{12}(t_1,t_2) + ss_{12}(t_1,t_2).$$

We have

$$\begin{aligned}
\mathcal{H}_0 &= \text{span}\{1, k_1(t_2)\}, \\
R_1((s_1,s_2),(t_1,t_2)) &= I_{[s_1=t_1]} - 1/K, \\
R_2((s_1,s_2),(t_1,t_2)) &= k_2(s_2)k_2(t_2) - k_4(s_2-t_2), \\
R_3((s_1,s_2),(t_1,t_2)) &= (I_{[s_1=t_1]} - 1/K)k_1(s_2)k_1(t_2), \\
R_4((s_1,s_2),(t_1,t_2)) &= (I_{[s_1=t_1]} - 1/K)[k_2(s_2)k_2(t_2) - k_4(s_2-t_2)].
\end{aligned}$$

Since $k_1(t_2) = t_2 - .5$, this SS ANOVA model can be fitted by

```
    ssr(y~I(t2-.5), rk=list(shrink1(t1),cubic(t2),
                            rk.prod(shrink1(t1),kron(t2-.5)),
                            rk.prod(shrink1(t1),cubic(t2))))
```

where the function `kron` in our library calculates the reproducing kernel for the space span$\{k_1(t_2)\}$.

Example 8 Consider $d = 2$, $t_1 \in \mathcal{T}_1 = [0,1]$ and $t_2 \in \mathcal{T}_2 = [0,1]$, a case with two continuous covariates. If we model both covariates using linear splines

$$\begin{aligned}
\mathcal{H}^{(1)} &= W_1([0,1]) = \{1\} \oplus \{g \in W_1([0,1]) : \int_0^1 g(t_1)dt_1 = 0\}, \\
\mathcal{H}^{(2)} &= W_1([0,1]) = \{1\} \oplus \{g \in W_1([0,1]) : \int_0^1 g(t_2)dt_2 = 0\},
\end{aligned}$$

then (15) leads to

$$f(t_1,t_2) = \mu + s_1(t_1) + s_2(t_2) + ss_{12}(t_1,t_2).$$

Thus

$$\begin{aligned}
\mathcal{H}_0 &= \text{span}\{1\}, \\
R_1((s_1,s_2),(t_1,t_2)) &= k_1(s_1)k_1(t_1) + k_2(s_1-t_1), \\
R_2((s_1,s_2),(t_1,t_2)) &= k_1(s_2)k_1(t_2) + k_2(s_2-t_2), \\
R_3((s_1,s_2),(t_1,t_2)) &= [k_1(s_1)k_1(t_1) + k_2(s_1-t_1)][k_1(s_2)k_1(t_2) + k_2(s_2-t_2)].
\end{aligned}$$

This SS ANOVA model can be fitted by

```
    ssr(y~1, rk=list(linear(t1),linear(t2),rk.prod(linear(t1),linear(t2))))
```

If we want to model both variables using cubic splines

$$\begin{aligned}
\mathcal{H}^{(1)} &= W_2([0,1]) = \{1\} \oplus \{k_1\} \oplus \{g \in W_2([0,1]) : \int_0^1 g(t_1)dt_1 = \int_0^1 g'(t_1)dt_1 = 0\}, \\
\mathcal{H}^{(2)} &= W_2([0,1]) = \{1\} \oplus \{k_1\} \oplus \{g \in W_2([0,1]) : \int_0^1 g(t_2)dt_2 = \int_0^1 g'(t_2)dt_2 = 0\},
\end{aligned}$$

then (16) leads to

$$f(t_1,t_2) = \mu + \alpha k_1(t_1) + \beta k_1(t_2) + s_1(t_1) + s_2(t_2) + ls_{12}(t_1,t_2) + sl_{12}(t_1,t_2) + ss_{12}(t_1,t_2). \tag{21}$$

We have

$$\mathcal{H}_0 = \text{span}\{1, k_1(t_1), k_1(t_2)\},$$

$$\begin{aligned}
R_1((s_1,s_2),(t_1,t_2)) &= k_2(s_1)k_2(t_1) - k_4(s_1-t_1),\\
R_2((s_1,s_2),(t_1,t_2)) &= k_2(s_2)k_2(t_2) - k_4(s_2-t_2),\\
R_3((s_1,s_2),(t_1,t_2)) &= k_1(s_1)k_1(t_1)[k_2(s_2)k_2(t_2) - k_4(s_2-t_2)],\\
R_4((s_1,s_2),(t_1,t_2)) &= [k_2(s_1)k_2(t_1) - k_4(s_1-t_1)]k_1(s_2)k(t_2),\\
R_5((s_1,s_2),(t_1,t_2)) &= [k_2(s_1)k_2(t_1) - k_4(s_1-t_1)][k_2(s_2)k_2(t_2) - k_4(s_2-t_2)]
\end{aligned}$$

SS ANOVA model (21) can be fitted by

```
ssr(y~I(t1-.5)+I(t2-.5), rk=list(cubic(t1),cubic(t2),
                                rk.prod(kron(t1-.5),cubic(t2)),
                                rk.prod(cubic(t1),kron(t2-.5)),
                                rk.prod(cubic(t1),cubic(t2))))
```

For the purpose of model building and inference, one may want to construct Bayesian confidence intervals for combinations of components in the model space (17). Gu and Wahba (1993b) provided formulae to calculate posterior covariances for any combination of components. Denote $f_{0v}(\boldsymbol{t}) = d_v\phi(\boldsymbol{t}),\ v = 1,\cdots,M$ as M components in the null space $\mathcal{H}_0$, and $f_k(\boldsymbol{t}) = \sum_{i=1}^n c_i\theta_k\xi_{ik}(\boldsymbol{t})$ as the component in space $\mathcal{H}_k,\ k = 1,\cdots,p$. Let $\delta_j,\ j = 1,\cdots,(M+p)$ be a sequence of 0's and 1's. The utility function `predict` calculates posterior means and standard deviations for the combination $\sum_{i=1}^M \delta_i f_{0i} + \sum_{k=1}^p \delta_{M+k} f_k$. Multiple combinations can be computed simultaneously. For example, after fitting the SS ANOVA model (21) and saving into an object, say `ssrfit`, then one may calculate the posterior mean and standard deviations for the smooth-smooth interaction ss_{12} and the total interaction $ls_{12}(t_1,t_2) + sl_{12}(t_1,t_2) + ss_{12}(t_1,t_2)$ by

```
predict(ssrfit,terms=c(0,0,0,0,0,0,0,1))
predict(ssrfit,terms=c(0,0,0,0,0,1,1,1))
```

These two statements can be combined into one

```
predict(ssrfit,terms=matrix(c(0,0,0,0,0,0,0,1,0,0,0,0,0,1,1,1),
                           ncol=2,byrow=T))
```

An object of class "bCI" is returned from this `predict` function, and the generic function `plot` can be used to plot these combinations with Bayesian confidence intervals. See help file of `plot.bCI` for details. `predict` function can also be used to calculate predicted values at any given points.

2.4 Spline Smoothing with Correlated Random Errors and/or Unequal Weights

So far we have assumed that random errors are independent with equal variances. In this section, we consider model (1) with model space for f given in (17), and $\boldsymbol{\epsilon} = (\epsilon_1,\cdots,\epsilon_n)^T \sim N(0,\sigma^2W^{-1})$. When W is completely known, we transform the data into iid case and fit as in previous sections. In the following we assume that W depends on a parsimonious set of parameters $\boldsymbol{\tau}$.

We estimate f as the minimizer of the following penalized weighted least squares

$$(\boldsymbol{y}-\boldsymbol{f})^TW(\boldsymbol{y}-\boldsymbol{f}) + n\lambda\sum_{k=1}^p \theta_k^{-1}||P_kf||^2. \tag{22}$$

Again, the solution has the form (19) (Wang 1998b).

Correlated random errors may have a profound effect on methods for selecting smoothing parameters such as GCV, GML and UBR (Wang 1998b). Extensions of GCV, GML and UBR methods with correlated random errors were developed in Wang (1998b). It was found that the extended GML method has the best overall performance. Therefore, we concentrate on the extended GML method here. Fixing $\lambda = 1$, the extended GML method estimates smoothing parameters $\boldsymbol{\theta}$ in (18) together with the variance-covariance parameters $\boldsymbol{\tau}$ as the minimizers of

$$GML(\boldsymbol{\theta}, \boldsymbol{\tau}) = \frac{\boldsymbol{y}^T W (I - A(\boldsymbol{\theta}))\boldsymbol{y}}{[\det^+(W(I - A(\boldsymbol{\theta})))]^{1/(n-M)}}. \tag{23}$$

To compute the minimizers of (23), we use the connection between smoothing spline models and linear mixed-effects models (LMM) (Wang 1998b, Opsomer, Wang and Yang 2001). Let $\Sigma_k = Z_k Z_k^T$, where Z_k is a $n \times m_k$ matrix with $m_k = \text{rank}(\Sigma_k)$. Consider the following LMM

$$\boldsymbol{y} = T\boldsymbol{d} + \sum_{k=1}^{p} Z_k \boldsymbol{b}_k + \boldsymbol{\epsilon}, \tag{24}$$

where $\boldsymbol{b}_k$'s are random effects, $\boldsymbol{b}_k \sim N(0, \sigma^2 \theta_k I_{m_k}/n)$ with I_{m_k} being the identity matrix of order m_k, and $\boldsymbol{b}_k$ are mutually independent and are independent of $\boldsymbol{\epsilon}$. It is easy to show that the restricted maximum likelihood (REML) estimate of the variance components $\boldsymbol{\theta}$ and $\boldsymbol{\tau}$ in model (24) are the minimizers of (23). Gu and Wahba (1991) noticed this connection and indicated that `RKPACK` can be used to fit LMMs (24) with independent random errors. Here the opposite is done: we use software for LMMs to fit smoothing spline models with correlated random errors. We first calculate Z_k through Choleski decomposition using the function `chol.new`. Then we calculate the GML (REML) estimates of $\boldsymbol{\theta}$ and $\boldsymbol{\tau}$ using the function `lme` in the `nlme` library (Pinheiro and Bates 2000). Finally we transform the data and call `dsidr.r` in `RKPACK` to calculate $\boldsymbol{c}$ and $\boldsymbol{d}$ in (19) with the smoothing parameters fixed at these GML estimates. All these steps are performed internally.

Two options in `ssr` can be used to specify the correlation and variance structures for Gaussian data. The first option, `correlation`, specifies the correlation structure of random errors. It inputs a `corMatrix` class as in `nlme`. Available correlation structures include `corSymm, corAR1, corCAR1, corARMA, corExp, corLin, corGaus, corSpher, corSpatial`. See the document of `nlme` (Pinheiro and Bates 2000) for a complete list. New structures can be added through facilities provided in `nlme`. The following statement fits a thin plate spline with an exponential correlation structure

```
ssr(y~t1+t2, rk=tp.pseudo(cbind(t1,t2)), corr=corExp(form=~t1+t2),
    spar=‘‘m’’)
```

The second option, `weights`, specifies the variance structure for the variance function of the random errors in the form of a `varFunc` class as in `nlme`. Available `varFunc` classes include `varFixed`, `varIdent`, `varPower`, `varExp` and `varComb`. See Pinheiro and Bates (2000) for details. New `varFunc` classes representing user-defined variance functions can be added. For example, the following statement fits a cubic spline with fixed weights (assuming a vector of `w` has been created)

```
ssr(y~t, rk=cubic(t), weights=w)
```

2.5 Generalized Smoothing Spline Models

Suppose that data have the form $(y_i, \boldsymbol{t}_i), i = 1, \cdots, n$, where y_i's are independent observations and $\boldsymbol{t}_i = (t_{1i}, \cdots, t_{di})$. The distribution of y_i is from an exponential family with density function

$$g(y_i; f_i, \phi) = exp\{y_i h(f_i) - b(f_i)/a(\phi) + c(y_i, \phi)\}, \tag{25}$$

where $f_i = f(\boldsymbol{t}_i)$, $h(f_i)$ is a monotone transformation of f_i known as the canonical link, and ϕ is a dispersion parameter. Assume that $f \in \mathcal{H}$ where $\mathcal{H}$ is given in (17). The penalized likelihood estimate of f is the minimizer of

$$-\sum_{i=1}^{n} l_i(f_i) + \frac{n\lambda}{2} \sum_{k=1}^{p} \theta_k^{-1} ||P_k f||^2, \tag{26}$$

where l_i is the log-likelihood of y_i. Again, the solution to (26) has the form (19) (Wahba et al. 1995), and $\boldsymbol{c}$ and $\boldsymbol{d}$ are solved by minimizing (26). Usually the coefficients cannot be solved directly. If all $l_i(f_i)$'s are strictly concave, the Newton-Raphson iterative procedure can be used to calculate $\boldsymbol{c}$ and $\boldsymbol{d}$ for fixed smoothing parameters. The smoothing parameters $\lambda/\theta_1, \cdots, \lambda/\theta_p$ can be estimated at each iteration using GCV, GML and UBR methods (Gu 1990, Gu 1992, Wahba et al. 1995). It was found that when the dispersion parameter is known, the UBR method works better than the GCV and GML methods (Wang et al. 1995). For binary, binomial, Poisson and gamma data, this procedure was implemented in `GRKPACK` (Wang 1997). In our `ASSIST` package, the functions `gdsidr` and `gdmudr` serve as intermediate interface between S and several drivers in `GRKPACK`.

The argument `family` in `ssr` specifies the distribution of y as in `glm`. Families supported are "binary", "binomial", "poisson", "gamma" and "gaussian" for Bernoulli, binomial, Poisson, gamma and Gaussian distributions respectively. The default is Gaussian.

Laplace approximations to the posterior mean and variance can be calculated by the `predict` function (Wahba et al. 1995). Then Bayesian confidence intervals can be constructed.

For example, one may fit a cubic spline to binary data with the UBR choice of the smoothing parameter and compute approximate posterior means and variances by

```
  a <- ssr(y~t, rk=cubic(t), family=‘‘binary’’, spar=‘‘u’’, varht=1)
  predict(a)
```

where `varht` specifies fixed variance (dispersion) parameter as 1 for the UBR function.

2.6 Other Options in `ssr` and Utility Functions

Additional arguments provided by `ssr` are `subset`, `scale`, `limnla` and `control`. `subset` selects a subset of the data for fitting. `scale`, if T (true), scales all covariates in the `rk` argument into $[0, 1]$. It is recommended that scaling be done before fitting. `limnla`, a vector of length 1 or 2, sets the searching limits for $n\lambda$ on log10 scale when fitting a univariate smoothing spline model (2). One may fix the smoothing parameter by setting the length of `limnla` to 1. For example, one may fit a cubic spline with $n\lambda = 0.01$ by

```
  ssr(y~t, rk=cubic(t), limnla=log10(0.01))
```

The `control` option specifies several control parameters used in `RKPACK` and `GRKPACK`. See `ssr.control` for details.

Generic utility functions supporting `ssr` include `summary`, `plot`, `deviance`, `residuals`, and `hat.ssr`, in addition to the `anova` and `predict` function discussed in previous sections. The `summary` function provides the basic description of a `ssr` fit. `plot` produces diagnostic plots for a `ssr` object. `hat.ssr` returns the hat matrix of a spline fit. Note that the full name `hat.ssr` should be used since the name "hat" has been utilized for another purpose. See the help files in the package for more detailed descriptions.

3 Semi-parametric Linear Mixed-Effects Models

3.1 Model and Estimation

Let f be a function of multivariate variables $t_1, \cdots, t_d$. Let $\boldsymbol{t} = (t_1, \cdots, t_d)$ and assume that $\boldsymbol{t} \in \mathcal{T} = \mathcal{T}_1 \otimes \cdots \otimes \mathcal{T}_d$. Suppose that we model f using an SS ANOVA decomposition. Specifically, we assume $f \in \mathcal{H}$ with $\mathcal{H}$ given in (17).

A semi-parametric linear mixed-effects (SLM) model assumes that

$$\boldsymbol{y} = \boldsymbol{f} + X\boldsymbol{\beta} + Z\boldsymbol{b} + \boldsymbol{\epsilon}, \tag{27}$$

where $\boldsymbol{y} = (y_1, \cdots, y_n)^T$, $\boldsymbol{f} = (f(\boldsymbol{t}_1), \cdots, f(\boldsymbol{t}_n))^T$, $\boldsymbol{t}_1, \cdots, \boldsymbol{t}_n$ are design points, X is the design matrix for some fixed effects with parameters $\boldsymbol{\beta}$, Z is the design matrix for the random effects $\boldsymbol{b}$, $\boldsymbol{b} \sim N(0, \sigma^2 D)$, $\boldsymbol{\epsilon}$ are random errors which are independent of $\boldsymbol{b}$ and $\boldsymbol{\epsilon} \sim N(0, \sigma^2 \Lambda)$. The covariance matrices D and Λ are assumed to depend on a parsimonious set of covariance parameters $\boldsymbol{\tau}$. Regarding the fixed effects part $\boldsymbol{f}+X\boldsymbol{\beta}$ as a partial spline, model (27) is essentially the same as the non-parametric mixed-effects model introduced in Wang (1998a).

For model (27), the marginal distribution of $\boldsymbol{y}$ is $\boldsymbol{y} \sim N(\boldsymbol{f}+X\boldsymbol{\beta}, \sigma^2 W^{-1})$ where $W^{-1} = ZDZ^T + \Lambda$. Given $\boldsymbol{\tau}$ and $\lambda/\theta_1, \cdots, \lambda/\theta_p$, we estimate fixed parameters f and $\boldsymbol{\beta}$ as the minimizers of the following penalized weighted least squares

$$(\boldsymbol{y} - \boldsymbol{f} - X\boldsymbol{\beta})^T W (\boldsymbol{y} - \boldsymbol{f} - X\boldsymbol{\beta}) + n\lambda \sum_{k=1}^{p} \theta_k^{-1} ||P_k f||^2. \tag{28}$$

Denote the estimates as $\hat{f}$ and $\hat{\boldsymbol{\beta}}$. We estimate $Z\boldsymbol{b}$ as the posterior mean $ZDZ^T W(\boldsymbol{y} - \hat{\boldsymbol{f}} - X\hat{\boldsymbol{\beta}})$, where $\hat{\boldsymbol{f}} = (\hat{f}(\boldsymbol{t}_1), \cdots, \hat{f}(\boldsymbol{t}_n))^T$.

Again, the solution of f has the form (19). Similar to Section 2.4, we use connections between a SLM and a LMM to estimate $\boldsymbol{\tau}$ and $\lambda/\theta_1, \cdots, \lambda/\theta_p$. Consider the following LMM

$$\boldsymbol{y} = T\boldsymbol{d} + \sum_{k=1}^{p} Z_k \boldsymbol{b}_k + X\boldsymbol{\beta} + Z\boldsymbol{b} + \boldsymbol{\epsilon} = (T, X)\begin{pmatrix} \boldsymbol{d} \\ \boldsymbol{\beta} \end{pmatrix} + (Z_1, \cdots, Z_p, Z)\begin{pmatrix} \boldsymbol{b}_1 \\ \vdots \\ \boldsymbol{b}_p \\ \boldsymbol{b} \end{pmatrix} + \boldsymbol{\epsilon}, \tag{29}$$

where Z_k's and $\boldsymbol{b}_k$'s are defined in (24), $\boldsymbol{\epsilon} \sim N(0, \sigma^2\Lambda)$, and $\boldsymbol{b}_1, \cdots, \boldsymbol{b}_p$, $\boldsymbol{b}$ and $\boldsymbol{\epsilon}$ are mutually independent. Then the GML estimates of $\boldsymbol{\tau}$ and $\lambda/\theta_1, \cdots, \lambda/\theta_p$ in (28) are the REML estimates of the variance components in (29) (Wang 1998a, Opsomer et al. 2001). As in Section 2.4, we use `lme` to calculate the GML (REML) estimates of $\boldsymbol{\tau}$ and $\lambda/\theta_1, \cdots, \lambda/\theta_p$. Then we transform the data and call `dsidr.r` to calculate $\boldsymbol{c}$, $\boldsymbol{d}$, and $\boldsymbol{\beta}$ with smoothing parameters fixed at the GML estimates. Formulae for calculating posterior variances were provided in Wang (1998a). Thus Bayesian confidence intervals can be constructed.

3.2 The `slm` Function

The S function for fitting a SLM is `slm`. A typical call is

```
    slm(formula, rk, random, data)
```

The first three arguments are required. `formula` and `rk` serve the same purposes as in `ssr`. `formula`, a two-sided formula separated by the operator `~`, lists the response variable on the left side, and the bases $\phi_1(t), \cdots, \phi_M(t)$ of $\mathcal{H}_0$ and covariates for the fixed effects in X on the right side. `rk` specifies the reproducing kernels of $\mathcal{H}_1, \cdots, \mathcal{H}_p$. `random` specifies the random effects the same way as in `nlme`. The syntax of `random` is in the form of a named list of formulae or some `pdMat` objects. See the help file of `lme` for more details.

Other options include `correlation`, `weights` and `control`. They all have the same functions as in `ssr`.

An object of `slm` class is returned. Generic functions `summary`, `predict` and `intervals` can be applied to extract further information. The `predict` function returns predictions at specified points. The `intervals` function returns the posterior means and variances of combinations of components in f as an object of class "bCI". Then the generic function `plot` can be used to construct plots. See help files for details.

As a simple example, consider repeated measures over time from multiple subjects. Suppose that we want to fit the following model

$$y_{ij} = f(t_{ij}) + b_i + \epsilon_{ij}, \quad i = 1, \cdots, m; \quad j = 1, \cdots, n_i; \quad t_{ij} \in [0,1],$$

where y_{ij} is the response at time t_{ij} from subject i, $f \in W_2([0,1])$, b_i is a random intercept for subject i and $b_i \overset{iid}{\sim} \mathrm{N}(0, \sigma_1^2)$, and ϵ_{ij}'s are random errors independent of b_i's. Suppose that random errors are independent between subjects, but correlated within a subject with a Gaussian correlation structure. Then we can fit such a model with

```
slm(y~t, rk=cubic(t), random=list(subject=~1),
    corr=corGaus(form=~t|subject))
```

4 Non-Parametric Nonlinear Regression Models

4.1 Model and Estimation

In model (1) we have assumed that the function f is observed through *linear* operators L_i's plus random errors. Sometimes the function is observed indirectly which involves *nonlinear* operators (O'Sullivan and Wahba 1985, Wahba 1987, Wahba 1990, O'Sullivan 1990, O'Sullivan 1991).

We consider the following non-parametric nonlinear regression (NNR) model

$$y_i = \eta(\boldsymbol{f}; \boldsymbol{t}_i) + \epsilon_i, \quad i = 1, \cdots, n, \tag{30}$$

where η is a known function of $\boldsymbol{t}_i = (t_{1i}, \cdots, t_{di})$ in an arbitrary domain $\mathcal{T}$, $\boldsymbol{f} = (f_1, \cdots, f_q)$ is a vector of unknown non-parametric functions which act nonlinearly as parameters of the function η, and $\boldsymbol{\epsilon} = (\epsilon_1, \cdots, \epsilon_n)^T$ are random errors distributed as $\mathrm{N}(\boldsymbol{0}, \sigma^2 W^{-1})$. The functions f_j's could have the same or different domains. We denote the model space of f_j as

$$\mathcal{H}_j = \mathcal{H}_{j0} \oplus \sum_{k=1}^{p_j} \mathcal{H}_{jk}. \tag{31}$$

Let $\boldsymbol{y} = (y_1, \cdots, y_n)^T$ and $\boldsymbol{\eta} = (\eta(\boldsymbol{f}; \boldsymbol{t}_1), \cdots, \eta(\boldsymbol{f}; \boldsymbol{t}_n))^T$. We estimate $\boldsymbol{f}$ as the minimizer of the following penalized weighted least squares

$$(\boldsymbol{y}-\boldsymbol{\eta})^T W(\boldsymbol{y}-\boldsymbol{\eta}) + n\lambda \sum_{j=1}^{q}\sum_{k=1}^{p_j} \theta_{jk}^{-1}||P_{jk}f_j||^2, \tag{32}$$

where P_{jk} is the orthogonal projection operator of f_j onto $\mathcal{H}_{jk}$ in $\mathcal{H}_j$.

In the following we consider the special case when

$$\eta(\boldsymbol{f};\boldsymbol{t}_i) = h(L_{1i}f_1,\cdots,L_{qi}f_q), \tag{33}$$

where h is a known nonlinear function, L_{ji}'s are linear operators. (33) holds for most applications and L_{ji}'s are usually the evaluational functionals. When (33) does not hold, using linearization method, we can approximate $\eta(\boldsymbol{f};\boldsymbol{t}_i)$ by a linear combination of linear operators.

When (33) holds, the solutions to (32) have the form (19). Specifically,

$$\hat{f}_j(\boldsymbol{t}) = \sum_{l=1}^{M_j} d_{jl}\phi_{jl}(\boldsymbol{t}) + \sum_{i=1}^{n} c_{ji}(\sum_{k=1}^{p_j}\theta_{jk}\xi_{kji}(\boldsymbol{t}), \tag{34}$$

where ϕ_{jl}, $l=1,\cdots,M_j$ are bases of $\mathcal{H}_{j0}$, $\xi_{kji}(\boldsymbol{t}) = L_{ji}R_{jk(\cdot)}(\boldsymbol{t},\cdot)$, and R_{jk} is the rk of $\mathcal{H}_{jk}$. We estimate coefficients d_{ji}'s and c_{jl}'s using (32) with f_j's being replaced by (34). Since h in (33) is nonlinear, an iterative method has to be used to solve these coefficients. Two methods are used: the Gauss-Newton and Newton-Raphson procedures. See Ke and Wang (2002) for more details.

4.2 The `nnr` Function

The S function for fitting a NNR model is `nnr`. A typical call is

```
    nnr(formula, func, start, data)
```

The first three arguments are required. `formula`, a two-sided formula separated by the operator `~`, lists the response variable on the left side, and an expression for the function η on the right side. `func` is a list of formulae specifying all components in $\boldsymbol{f}$. Each formula in this list has the form `f` $\sim$ `list(` $\sim \phi_1+\cdots+\phi_M,$`rk)`. For example, suppose that $\boldsymbol{f}=(f_1,f_2)$, and both f_1 and f_2 are modeled using cubic splines. Then

```
   func=list(f1(t)~list(~t, cubic(t)), f2(t)~list(~t, cubic(t)))
```

or

```
   func=list(f1(t)+f2(t)~list(~t, cubic(t)))
```

`start`, a vector or an expression, specifies the initial values for the iterative procedure.

Method for selecting smoothing parameters is specified by the argument `spar`. `spar=‘‘v’’`, `spar=‘‘m’’` and `spar=‘‘u’’` correspond to GCV, GML and UBR methods respectively, with GCV as the default. Other options include `correlation`, `weights`, `control`, and `subset`. They all have the same functions as in `ssr`. The option `method` in the argument `control` specifies the iterative method. `method=‘‘GN’’` and `method=‘‘NR’’` correspond the Gauss-Newton and Newton-Raphson methods respectively, with the Newton-Rahson method as the default.

An object of `nnr` class is returned. Generic functions `summary`, `predict` and `intervals` can be applied to extract further information. See help files for details.

As a simple example, consider the following common form of a non-parametric regression model

$$y_i = f(t_i)+\epsilon_i, \quad t_i\in[0,1], \ \ i=1,\cdots,n. \tag{35}$$

Suppose that we want to restrict f as a *positive* function. We may use the exponential transformation $f = \exp(g)$ or square transformation $f = g^2$ to enforce positivity. Then we can model the unconstrained function g by a spline model.

For the purpose of illustration, we generate $n = 100$ samples from model (35) with $f(t) = \exp(\sin 2\pi t)$ and $\epsilon_i \overset{iid}{\sim} \mathrm{N}(0, .5^2)$. We fit f using the exponential transformation in `nnrfit1` and square transformation in `nnrfit2`. We use a cubic periodic spline to model g. Figure 2 shows the fits.

```
t <- seq(0,1, len=100)
y <- exp(sin(2*pi*t))+0.5*rnorm(100)
nnrfit1 <- nnr(y~exp(g(t)), func=g(u)~list(~1, periodic(u)),
               start=log(abs(y)+0.001))
nnrfit2 <- nnr(y~g(t)**2, func=g(u)~list(~1, periodic(u)),
               start=sqrt(abs(y)))
```

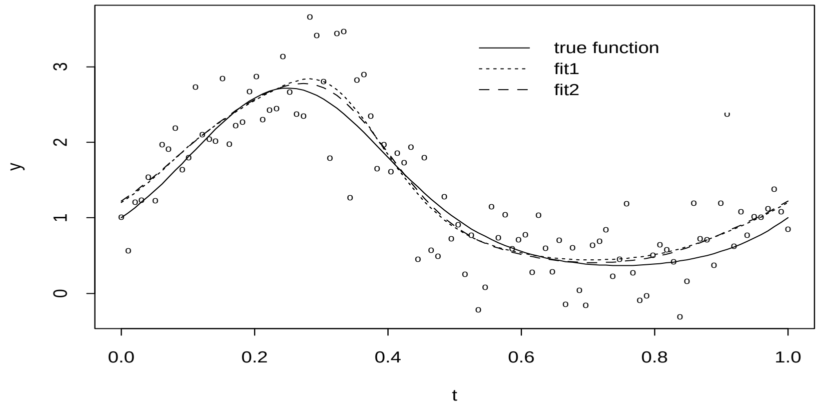


Figure 2: Points are observations; Lines are the true and estimated functions.

See Section 7 for more interesting examples.

5 Semi-parametric Nonlinear Regression Models

When building regression models, often we have enough knowledge to model some features of the mean response function parametrically, but only have vague knowledge about other features. Thus we want to leave these vague features unspecified and model them non-parametrically. A partial spline is an example where the mean function depends on both parameters and some non-parametric functions linearly. In this section we consider more general semi-parametric nonlinear regression (SNR) models where the mean function may depend on both parameters and non-parametric functions nonlinearly.

5.1 SNR Models for Non-grouped Data

We define a class of SNR models for non-grouped data as

$$y_i = \eta(\boldsymbol{\phi}, \boldsymbol{f}; \boldsymbol{t}_i) + \epsilon_i, \quad i = 1, \cdots, n, \tag{36}$$

where y_i's are responses; $\boldsymbol{t}_i = (t_{1i}, \cdots, t_{di})$ is a covariate in a general domain $\mathcal{T}$; η is a known function of $\boldsymbol{t}_i$ which depends on a vector of parameters $\boldsymbol{\phi} = (\phi_1, \cdots, \phi_r)^T$ and a vector of unknown non-parametric function $\boldsymbol{f} = (f_1, \cdots, f_q)^T$; and $\boldsymbol{\epsilon} = (\epsilon_1, \cdots, \epsilon_n)^T$ are random errors distributed as $\mathrm{N}(\mathbf{0}, \sigma^2 W^{-1})$. f_j's are modeled using SS ANOVA models as represented in (31). We assume that W depends on a parsimonious set of parameters $\boldsymbol{\tau}$.

For model (36), we denote $\boldsymbol{y} = (y_1, \cdots, y_n)^T$ and $\boldsymbol{\eta}(\boldsymbol{\phi}, \boldsymbol{f}) = (\eta(\boldsymbol{\phi}, \boldsymbol{f}; \boldsymbol{t}_1), \cdots, \eta(\boldsymbol{\phi}, \boldsymbol{f}; \boldsymbol{t}_n))^T$. Model (36) can then be written in the vector form

$$\boldsymbol{y} = \boldsymbol{\eta}(\boldsymbol{\phi}, \boldsymbol{f}) + \boldsymbol{\epsilon}. \tag{37}$$

5.2 SNR Models for Grouped Data

Grouped data include repeated measures data, longitudinal data, functional data and multilevel data as special cases. For such data, we define SNR models as

$$y_{ij} = \eta(\boldsymbol{\phi}_i, \boldsymbol{f}; \boldsymbol{t}_{ij}) + \epsilon_{ij}, \quad i = 1, \cdots, m, \quad j = 1, \cdots, n_i, \tag{38}$$

where y_{ij} is the response of subject i at design point $\boldsymbol{t}_{ij}$, $\boldsymbol{t}_{ij} = (t_{1ij}, \cdots, t_{dij})$ is a covariate in a general domain $\mathcal{T}$, η is a known function of $\boldsymbol{t}_{ij}$ which depends on a vector of parameter $\boldsymbol{\phi}_i = (\phi_{i1}, \cdots, \phi_{ir})^T$ and a vector of unknown non-parametric function $\boldsymbol{f} = (f_1, \cdots, f_q)^T$; and $\boldsymbol{\epsilon} = (\epsilon_{11}, \cdots, \epsilon_{1n_1}, \cdots, \epsilon_{m1}, \cdots \epsilon_{mn_m}) \sim \mathrm{N}(0, \sigma^2 W^{-1})$. For grouped data, usually observations are correlated within a subject but are independent between subjects. In this case W is in the form of block diagonal. Again, each function f_j is modeled using an SS ANOVA model (31). Again, assume that W depends on a parsimonious set of parameters $\boldsymbol{\tau}$.

For model (38), we denote $n = \sum_{i=1}^m n_i$, $\boldsymbol{y}_i = (y_{i1}, \cdots, y_{in_i})^T$, $\boldsymbol{y} = (\boldsymbol{y}_1^T, \cdots, \boldsymbol{y}_m^T)^T$, $\boldsymbol{\eta}_i(\boldsymbol{\phi}_i, \boldsymbol{f}) = (\eta(\boldsymbol{\phi}_i, \boldsymbol{f}; \boldsymbol{t}_{i1}), \cdots, \eta(\boldsymbol{\phi}_i, \boldsymbol{f}; \boldsymbol{t}_{in_i}))^T$, $\boldsymbol{\phi} = (\boldsymbol{\phi}_1^T, \cdots, \boldsymbol{\phi}_m^T)^T$, and $\boldsymbol{\eta}(\boldsymbol{\phi}, \boldsymbol{f}) = (\boldsymbol{\eta}_1(\boldsymbol{\phi}_1, \boldsymbol{f})^T, \cdots, \boldsymbol{\eta}_m(\boldsymbol{\phi}_m, \boldsymbol{f})^T)^T$. Then model (38) can be written in the same vector form as (37).

5.3 Estimation

Since both model (36) and (38) have the same vector form (37), we consider estimation of these two models simultaneously.

We estimate $\boldsymbol{\phi}$ and $\boldsymbol{f}$ as the minimizers of the following penalized weighted least squares

$$(\boldsymbol{y} - \boldsymbol{\eta}(\boldsymbol{\phi}, \boldsymbol{f}))^T W (\boldsymbol{y} - \boldsymbol{\eta}(\boldsymbol{\phi}, \boldsymbol{f})) + n\lambda \sum_{j=1}^{q} \sum_{k=1}^{p_j} \theta_{jk}^{-1} ||P_{jk} f_j||^2. \tag{39}$$

The following iterative procedure is used to solve (39).

Algorithm Estimate $\boldsymbol{f}$, $\boldsymbol{\phi}$ and $\boldsymbol{\tau}$ iteratively using the following two steps:
(a) Given the current estimates of $\boldsymbol{\phi}$ and $\boldsymbol{\tau}$, update $\boldsymbol{f}$;
(b) Given the current estimates of $\boldsymbol{f}$, update $\boldsymbol{\phi}$ and $\boldsymbol{\tau}$.

In step (a), if η is linear in $\boldsymbol{f}$, then model (37) is a SSR model. Thus the solutions have the form (19). After certain transformations, we can call `ssr` to update $\boldsymbol{f}$. If η is nonlinear in $\boldsymbol{f}$, then model (37) is a NNR model. Thus the closed form of solutions do not exist. We can approximate the solutions as in NNR models. After certain transformations, we can call `nnr` to update $\boldsymbol{f}$.

In step (b), (37) is a regular parametric nonlinear regression model when $\boldsymbol{f}$ is fixed. Thus we can update $\boldsymbol{\phi}$ and $\boldsymbol{\tau}$ using the S function `gnls`. We implemented the algorithm above by calling `ssr`/`nnr` and `gnls` alternately.

Conditional on $\boldsymbol{\phi}$, one can construct Bayesian confidence intervals as before. Adjustments need to be made to account for the loss of the degrees of freedom when $\boldsymbol{\phi}$ is estimated. See Ke and Wang (2002) for more detailed discussions.

5.4 The `snr` Function

The S function for fitting a SNR model is `snr`. A typical call is

```
    snr(formula, func, params, start, data)
```

The first four arguments are required. Arguments `formula` and `func` are the same as in `nnr`. `params` and `start` specify models for parameters $\boldsymbol{\phi}$ and their initial values for the iterative procedure.

Method for selecting smoothing parameters is specified by the argument `spar`. `spar=''v''`, `spar=''m''` and `spar=''u''` correspond to GCV, GML and UBR methods respectively, with GCV as the default. Other options include `correlation`, `weights`, and `control`. They all have the same function as in `ssr`.

An object of `snr` class is returned. Generic functions `summary`, `predict` and `intervals` can be applied to extract further information. See help files for details.

Example 9. *Projection Pursuit models* assume that $\eta(\boldsymbol{\phi}, \boldsymbol{f}; \boldsymbol{t}) = \sum_{j=1}^{q} f_j(\boldsymbol{\phi}_j^T \boldsymbol{t})$. They are also known as the multiple index models. Note that our estimation procedure is similar to that used in Roosen and Hastie (1994). For example, suppose that $q = 2$, $d = r = 3$, and both f_1 and f_2 are modeled using TPS on R with $m = 2$. Instead of polynomial splines, we use TPS's to avoid the limitation on the domain. For identifiability, we need the side condition $\phi_{j1}^2 + \phi_{j2}^2 + \phi_{j3}^2 = 1$. We can fit such a model by

```
   snr(y~f1(a11*t1+a12*t2+sqrt(1-a11**2-a12**2)*t3)
       + f2(a21*t1+a22*t2+sqrt(1-a21**2-a22**2)*t3),
       func=list(f1(z)+f2(z)~list(~z,rk=tp.pseudo(z)),
       params=list(a11+a12+a21+a22~1),
       start=c(a110,a120,a210,a220))
```

Example 10. *Nonlinear partial splines* assume that $\eta(\boldsymbol{\phi}, \boldsymbol{f}; \boldsymbol{t}) = f(\boldsymbol{t}) + g(\boldsymbol{\phi}; \boldsymbol{t})$, where g is a known function depends nonlinearly on parameters $\boldsymbol{\phi}$. For example, suppose that $\boldsymbol{t} = (t_1, t_2)$, f is a function of t_1 modeled using a periodic spline, and $g(\boldsymbol{\phi}; \boldsymbol{t}) = \phi_1 \exp(\phi_2 t_2)$. we can fit such a model by

```
    snr(y~f(t1)+a*exp(b*t2), func=list(f(z)~list(~1,rk=periodic(z)),
         params=list(a+b~1), start=c(a0,b0))
```

The function g can also be input as an outside S function as in `gnls`.

Example 11. *Monotone spline.* Consider model (35) where f is assumed to be a strictly increasing function. Nychka and Ruppert (1995) used the following transformation

$$f(t) = \eta(\boldsymbol{\phi}, g; t) = \phi_1 + \int_0^t \exp(g(s))ds,$$

where $g \in W_2([0, 1])$. We can fit such a model by

```
    snr(y~a+h(g), func=list(g(z)~list(~z,rk=cubic(z)),
         params=list(a~1), start=c(a0))
```

where h is an S function calculating the integral $\int_0^t \exp(g(s))ds$.

For monotone f (either increasing or decreasing), Ramsay (1998) suggested the following transformation

$$f(t) = \eta(\boldsymbol{\phi}, g; t) = \phi_1 + \phi_2 \int_0^t \exp(\int_0^s g(u)du)ds,$$

where $g \in W_2([0,1])$. We can fit such a model by

```
    snr(y~a+b*h(g), func=list(g(z)~list(~z,rk=cubic(z))),
        params=list(a~1,b~1), start=c(a0,b0))
```

where h is a S function calculating the integral $\int_0^t \exp(\int_0^s g(u)du)ds$.

Example 12. *Positive monotone spline.* Consider model (35) where f is assumed to be a strictly positive and increasing function. One may use the following transformation

$$f(t) = \eta(\boldsymbol{\phi}, g; t) = \exp(\phi_1 + \int_0^t \exp(g(s))ds),$$

where $g \in W_2([0,1])$. We can fit such a model by

```
    snr(y~exp(a+h(g)), func=list(g(z)~list(~z,rk=cubic(z)),
        params=list(a~1), start=c(a0))
```

where h is a S function calculating the integral $\int_0^t \exp(g(s))ds$.

Example 13. *Varying coefficient models.* Suppose that $\boldsymbol{t} = (\boldsymbol{x}, \boldsymbol{z})$. The varying coefficient models in Hastie and Tibshirani (1993) assume that

$$\eta(\boldsymbol{\phi}, \boldsymbol{f}; \boldsymbol{t}) = \phi_1 + \sum_{j=1}^{q} f_j(\boldsymbol{z}) x_j.$$

Suppose that $q = 2$, both f_1 and f_2 are univariate functions of a continuous variable z and are modeled using cubic splines. We can fit such a model by

```
    snr(y~a+f1(z)*x1+f2(z)*x2, func=list(f1(z)+f2(z)~list(~z,cubic(z))),
        params=list(a~1), start=c(a0))
```

Example 14. *Self-modeling nonlinear regression (SEMOR)* models were first proposed by Lawton, Sylvestre and Maggio (1972) to fit repeated measures data. They assumed that there exists a *common* curve f for all subjects and that a particular subject's response curve is some parametric transformation of the common curve. We consider a more general SEMOR model

$$y_{ij} = \alpha(\boldsymbol{\phi}_i; t_{ij}) + \delta(\boldsymbol{\phi}_i; t_{ij}) f(\gamma(\boldsymbol{\phi}_i; t_{ij})) + \epsilon_{ij}, \quad i = 1, \cdots, m, \quad j = 1, \cdots, n_i, \tag{40}$$

where α, δ and γ are known functions of t with unknown parameters $\boldsymbol{\phi}_i$. Usually t is a continuous variable such as time. f is a function of a univariate continuous variable. Depending on the range of γ, polynomial splines or univariate TPS may be used to model f. Most often, a SEMOR model is in the form of

$$y_{ij} = \phi_{i1} + \exp(\phi_{i2}) f((t_{ij} - \phi_{i3})/\exp(\phi_{i4})) + \epsilon_{ij}, \quad i = 1, \cdots, m, \quad j = 1, \cdots, n_i. \tag{41}$$

Model (41) is referred to as a shape invariant model (SIM). Suppose we model f with a TPS on R with $m = 2$ and random errors are iid normal. Note that we used exponential transformation to force the amplitude parameter and the scale parameter to be positive. It is clear that model (41) is not identifiable without side conditions. We now illustrate two approaches to make it identifiable. The first approach uses the following side conditions: $\phi_{11} = \phi_{12} = \phi_{13} = \phi_{14} = 0$. Correspondingly these set-to-zero conditions free f from confounding with a vertical shift, a vertical stretch, a horizontal shift and a horizontal stretch. Then model (41) can be fitted by

```
    snr(y~a1+exp(a2)*f((t-a3)/exp(a4)),
        func=list(f(z)~list(~z,tp.pseudo(z))),
        params=list(a1+a2+a3+a4~subject-1),
        start=c(a10,a20,a30,a40))
```

We may also impose side conditions on f. For example, we may remove the constant functions from the model space of f to make it identifiable with ϕ_{i1}'s. We may use the constraint $\sup|f(t)| = 1$ to make f identifiable with ϕ_{i2}'s. Under these alternative side conditions and $\phi_{13} = \phi_{14} = 0$, model (41) can be fitted by

```
  snr(y~a1+exp(a2)*f((t-a3)/exp(a4)),
      func=list(f(z)~list(~z-1,tp.pseudo(z))),
      params=list(a1+a2~subject,a3+a4~subject-1),
      start=c(a10,a20,a30,a40), constraint=list(maxValue=1))
```

Note that the "constraint" option is not available in the current version.

6 Semi-parametric Nonlinear Mixed-Effects Models

6.1 Model and Estimation

Semi-parametric nonlinear mixed-effects (SNM) models extend current statistical nonlinear models for grouped data in two directions: adding flexibility to a nonlinear mixed-effects model by allowing the mean function to depend on some non-parametric functions, and providing ways to model covariance structure and covariates effects in an SNR model. An SNM model assumes that

$$\begin{aligned} y_{ij} &= \eta(\boldsymbol{\phi}_i, \boldsymbol{f}; \boldsymbol{t}_{ij}) + \epsilon_{ij}, \quad j = 1, \cdots, n_i, \quad i = 1, \cdots, m, & (42) \\ \boldsymbol{\phi}_i &= A_i \boldsymbol{\beta} + B_i \boldsymbol{b}_i, \quad \boldsymbol{b}_i \overset{iid}{\sim} \mathrm{N}(\boldsymbol{0}, \sigma^2 D), & (43) \end{aligned}$$

where the first-stage model (42) is the same as a SNR model (38), and the second-stage model is the same as one for a nonlinear mixed-effect model. Specifically, y_{ij} is the response of subject i at design point $\boldsymbol{t}_{ij}$, $\boldsymbol{t}_{ij} = (t_{1ij}, \cdots, t_{dij})$ are independent variables in a general domain $\mathcal{T}$, η is a known function of $\boldsymbol{t}_{ij}$ which depends on a vector of parameter $\boldsymbol{\phi}_i = (\phi_{i1}, \cdots, \phi_{ir})^T$ and a vector of unknown non-parametric function $\boldsymbol{f} = (f_1, \cdots, f_q)^T$; random errors $\boldsymbol{\epsilon} = (\epsilon_{11}, \cdots, \epsilon_{1n_1}, \cdots, \epsilon_{m1}, \cdots, \epsilon_{mn_m}) \sim \mathrm{N}(0, \sigma^2\Lambda)$; $\boldsymbol{\beta}$ is a p-vector of fixed effects, $\boldsymbol{b}_i$ is k-vector of random effects associated with subject i; A_i and B_i are design matrices of sizes $r \times p$ and $r \times k$ for the fixed and random effects respectively. It is assumed that the random effects and random errors are mutually independent. Each function f_j is modeled using an SS ANOVA model (31).

Let $n = \sum_{i=1}^m n_i$, $\boldsymbol{y}_i = (y_{i1}, \cdots, y_{in_i})^T$, $\boldsymbol{y} = (\boldsymbol{y}_1^T, \cdots, \boldsymbol{y}_m^T)^T$, $\boldsymbol{\phi} = (\boldsymbol{\phi}_1^T, \cdots, \boldsymbol{\phi}_m^T)^T$, $\boldsymbol{\eta}_i(\boldsymbol{\phi}_i, \boldsymbol{f}) = (\eta(\boldsymbol{\phi}_i, \boldsymbol{f}; \boldsymbol{t}_{i1}), \cdots, \eta(\boldsymbol{\phi}_i, \boldsymbol{f}; \boldsymbol{t}_{in_i}))^T$, $\boldsymbol{\eta}(\boldsymbol{\phi}, \boldsymbol{f}) = (\boldsymbol{\eta}_1^T(\boldsymbol{\phi}_1, \boldsymbol{f}), \cdots, \boldsymbol{\eta}_m^T(\boldsymbol{\phi}_m, \boldsymbol{f}))^T$ and $\boldsymbol{b} = (\boldsymbol{b}_1^T, \cdots, \boldsymbol{b}_m^T)^T$. The SNM model (42) and (43) can then be written in a matrix form

$$\begin{array}{l} \boldsymbol{y}|\boldsymbol{b} \sim \mathrm{N}(\boldsymbol{\eta}(\boldsymbol{\phi}, \boldsymbol{f}), \sigma^2\Lambda), \\ \boldsymbol{\phi} = A\boldsymbol{\beta} + B\boldsymbol{b}, \ \boldsymbol{b} \sim \mathrm{N}(\boldsymbol{0}, \sigma^2\tilde{D}), \end{array} \tag{44}$$

where $A = (A_1^T, \cdots, A_m^T)^T$, $B = \mathrm{diag}(B_1, \cdots, B_m)$ and $\tilde{D} = \mathrm{diag}(D, \cdots, D)$. In the following we assume that Λ and D depend on an unknown parameter vector $\boldsymbol{\tau}$.

For fixed $\boldsymbol{\tau}$ and σ^2, we estimate $\boldsymbol{\beta}$, $\boldsymbol{f}$, $\boldsymbol{b}$, $\boldsymbol{\tau}$ as the minimizers of the following double penalized log-likelihood

$$(\boldsymbol{y} - \boldsymbol{\eta}(A\boldsymbol{\beta} + B\boldsymbol{b}, \boldsymbol{f}))^T \Lambda^{-1} (\boldsymbol{y} - \boldsymbol{\eta}(A\boldsymbol{\beta} + B\boldsymbol{b}, \boldsymbol{f})) + \boldsymbol{b}^T \tilde{D}^{-1} \boldsymbol{b} + n\lambda \sum_{j=1}^{q} \sum_{k=1}^{p_j} \theta_{jk}^{-1} ||P_{jk} f_j||^2. \tag{45}$$

Denote $\tilde{\boldsymbol{\beta}}$, $\tilde{\boldsymbol{f}}$ and $\tilde{\boldsymbol{b}}$ as solutions to (45). Let $\tilde{Z} = (\partial \boldsymbol{\eta}(A\boldsymbol{\beta} + B\boldsymbol{b}, \boldsymbol{f})/\partial \boldsymbol{b}^T)_{\boldsymbol{b}=\tilde{\boldsymbol{b}}}$, and $\tilde{V} = \Lambda + \tilde{Z}\tilde{D}\tilde{Z}^T$. We estimate $\boldsymbol{\tau}$ and σ^2 as minimizers of the approximate profile log-likelihood

$$\log|\sigma^2\tilde{V}| + \sigma^{-2}(\boldsymbol{y} - \boldsymbol{\eta}(A\tilde{\boldsymbol{\beta}} + B\tilde{\boldsymbol{b}}, \tilde{\boldsymbol{f}}) + \tilde{Z}\tilde{\boldsymbol{b}})^T\tilde{V}^{-1}(\boldsymbol{y} - \boldsymbol{\eta}(A\tilde{\boldsymbol{\beta}} + B\tilde{\boldsymbol{b}}, \tilde{\boldsymbol{f}}) + \tilde{Z}\tilde{\boldsymbol{b}}). \tag{46}$$

Since $\boldsymbol{f}$ may interact with $\boldsymbol{\beta}$ and $\boldsymbol{b}$ in a complicated way, we have to use iterative procedures to solve (45) and (46). We proposed two procedures in Ke and Wang (2001) for the case when η is linear in $\boldsymbol{f}$. It is not difficult to extend these procedures to the general case. In the following we describe the extension of Procedure 1 in Ke and Wang (2001).
Procedure 1: estimate $\boldsymbol{f}$, $\boldsymbol{\beta}$, $\boldsymbol{b}$, $\boldsymbol{\tau}$ and σ^2 iteratively using the following three steps:
(a) given the current estimates of $\boldsymbol{\beta}$, $\boldsymbol{b}$ and $\boldsymbol{\tau}$, update $\boldsymbol{f}$ by solving (45);
(b) given the current estimates of $\boldsymbol{f}$ and $\boldsymbol{\tau}$, update $\boldsymbol{\beta}$ and $\boldsymbol{b}$ by solving (45);
(c) given the current estimates of $\boldsymbol{f}$, $\boldsymbol{\beta}$ and $\boldsymbol{b}$, update $\boldsymbol{\tau}$ and σ^2 by solving (46).

Note that step (b) corresponds to the *pseudo-data* step and step (c) corresponds to part of the *LME* step in Lindstrom and Bates (1990). Thus the `nlme` can be used to accomplish (b) and (c). In step (a) (45) is reduced to (32) after certain transformations. Then depending on if η is linear in $\boldsymbol{f}$, the `ssr` or `nnr` function can be used to update $\boldsymbol{f}$. We choose smoothing parameters using a data-adaptive criterion such as GCV, GML or UBR at each iteration.

To minimize (45) we need to alternate between steps (a) and (b) until convergence. Our simulations indicate that one iteration is usually enough. Figure 3 shows the flow chart of Procedure 1 if we alternate (a) and (b) only once. Step (a) can be solved by `ssr` or `nnr`. It is easy to see that steps (b) and (c) are equivalent to fitting a NLMM with $\boldsymbol{f}$ fixed at the current estimate using the same methods proposed in Lindstrom and Bates (1990). Therefore these two steps can be combined and solved by S program `nlme` (Pinheiro and Bates 2000). Figure 3 suggests an obvious iterative algorithm by calling `nnr` and `nlme` alternately. It is not difficult to use other options in our implementation. For example, we may alternate steps (a) and (b) several times before proceeding to step (c). In our studies these approaches usually gave the same results. For details about the estimation methods and procedures, see Ke and Wang (2001).

$$\boldsymbol{f}^0\;\;\boldsymbol{\beta}^0\;\;\boldsymbol{b}^0\;\;\boldsymbol{\tau}^0\;\;(\sigma^2)^0 \longrightarrow \underbrace{\underset{a}{\boldsymbol{f}^1}}_{\texttt{ssr/nnr}}\;\;\underbrace{\underset{b}{\boldsymbol{\beta}^1}\;\;\underset{b}{\boldsymbol{b}^1}\;\;\underset{c}{\boldsymbol{\tau}^1}\;\;\underset{c}{(\sigma^2)^1}}_{\texttt{nlme}} \longrightarrow \underbrace{\underset{a}{\boldsymbol{f}^2}}_{\texttt{ssr/nnr}}\;\;\underbrace{\underset{b}{\boldsymbol{\beta}^2}\;\;\underset{b}{\boldsymbol{b}^2}\;\;\underset{c}{\boldsymbol{\tau}^2}\;\;\underset{c}{(\sigma^2)^2}}_{\texttt{nlme}} \longrightarrow \cdots$$

Figure 3: Flow chart of Procedure 1. The first row shows the order in which parameters are updated. The second row indicates the corresponding step for each parameter. The third row indicates that step (a) can be solved by `ssr` or `nnr` and steps (b) and (c) can be combined and solved by `nlme`.

Approximate Bayesian confidence intervals can be constructed for $\boldsymbol{f}$ (Ke and Wang 2001).

6.2 The `snm` Function

The S function for fitting a SNM model is `snm`. A typical call is

```
    snm(formula, func, fixed, start, random, data)
```

The first 4 arguments are required. Arguments `formula` and `func` are the same as in nnr. Following syntax in `nlme`, the `fixed` and `random` arguments specify the fixed and random effects models in the second stage model (43). The option `start` specifies initial values for all parameters in the fixed effects.

`snm` inherits most of the options in `nlme`. See documents of `nlme` and the help file of `snm` for details. Generic functions `summary`, `predict` and `intervals` can be applied to extract further information. `intervals` provides approximate posterior means and variances which can be used to construct Bayesian confidence intervals for the $\boldsymbol{f}$. Derivatives of η with respect to random effects are needed to compute these quantities (Ke and Wang 2001). In `interval.snm`, numerical derivatives are to be used.

Example 15. Mixed-effects SIMs. In the SIM (41) for repeated measure data, it is more appropriate to consider parameters as random variables (Ke and Wang 2001):

$$y_{ij} = \beta_1 + b_{1i} + \exp(b_{2i}) f((t_{ij} - b_{3i})/\exp(b_{4i})) + \epsilon_{ij}, \quad i = 1, \cdots, m; j = 1, \cdots, n_i, \tag{47}$$

where $\boldsymbol{b}_i = (b_{1i}, b_{2i}, b_{3i}, b_{4i})^T \sim \mathrm{N}(0, \Sigma)$ and Σ is an unstructured positive-definite matrix. Suppose that we want to model f using a TPS on R with $m = 2$. Note that no identifiability condition is necessary for f since $\mathrm{E}(b_{2i}) = 0$. We can fit model (47) with independent random errors by

```
snm(y~b1+exp(b2)*f((t-b3)/exp(b4)),
    func=list(f(u)~list(u,tp.pseudo(u))),
    fixed=list(b1~1), random=list(b1+b2+b3+b4~1), start=b10)
```

7 Vector Spline Models

In many applications two or more dependent variables are observed at several values of independent variables such as at multiple time points. Often observations from different variables are contemporaneously correlated. Observations from the same variable may also be correlated. The statistical problems are (i) to estimate functions that model their dependences on the independent variables and (ii) to investigate relationships between these functions. Wang, Guo and Brown (2000) proved that the joint estimates have smaller posterior variances than those of function-by-function estimates and are therefore more efficient. In this section we show how to use `ssr` and `nnr` to fit functions jointly.

Consider the following model

$$y_{ji} = f_j(t_{ji}) + \epsilon_{ji}, \quad j = 1, \cdots, J; \; i = 1, \cdots, n_j, \tag{48}$$

where the ith response of the jth variable y_{ji} is generated as the jth function f_j evaluated at the design point t_{ji} plus a random error ϵ_{ji}. We assume that f_j has the model space (31).

Denote $\boldsymbol{t}_j = (t_{j1}, \cdots, t_{jn_j})^T$, $\boldsymbol{f}_j = (f_j(t_{j1}), \cdots, f_j(t_{jn_j}))^T$, $\boldsymbol{y}_j = (y_{j1}, \cdots, y_{jn_j})^T$, $\boldsymbol{\epsilon}_j = (\epsilon_{j1}, \cdots, \epsilon_{jn_j})^T$, $\boldsymbol{f} = (\boldsymbol{f}_1^T, \cdots, \boldsymbol{f}_J^T)^T$, $\boldsymbol{y} = (\boldsymbol{y}_1^T, \cdots, \boldsymbol{y}_J^T)^T$, and $\boldsymbol{\epsilon} = (\boldsymbol{\epsilon}_1^T, \cdots, \boldsymbol{\epsilon}_J^T)^T$. Assume that $\boldsymbol{\epsilon} \sim \mathrm{N}(0, \sigma^2 W^{-1})$, where W depends on some parameters $\boldsymbol{\tau}$.

We estimate all functions f_j's jointly as the minimizer of the following penalized weighted least squares

$$(\boldsymbol{y} - \boldsymbol{f})^T W (\boldsymbol{y} - \boldsymbol{f}) + n\lambda \sum_{j=1}^{J} \sum_{k=1}^{p_j} \theta_{jk}^{-1} ||P_{jk} f_j||^2, \tag{49}$$

where P_{jk} is the orthogonal projection operator of f_j onto $\mathcal{H}_{jk}$ in $\mathcal{H}_j$. We use the GML method to estimate the variance-covariance parameters $\boldsymbol{\tau}$ and the smoothing parameters θ_{jk}'s as the minimizers of (23).

We now show how to trick `ssr` to fit model (48). For the simplicity of notations, we consider the case of $J = 2$. Situations with $J > 2$ can be fitted similarly. Note that the domains of t_{ji} may be different for different j. For most applications they are the same, which is assumed in the remaining of this section. Denote the common domain as $\mathcal{T}$. Rewrite $f_j(t)$ as $f(j,t)$, which is now considered as a function of both j and t variables on the tensor product domain $\{1,2\} \otimes \mathcal{T}$. Then we can represent the original functions as

$$f(j,t) = f_1(t) \times I_{[j=1]} + f_2(t) \times I_{[j=2]}. \tag{50}$$

Let

$$\mathcal{M}_j = \mathcal{H}_{j0} \oplus \mathcal{H}_{j1} \oplus \cdots \oplus \mathcal{H}_{jp_j}, \qquad j = 1,2 \tag{51}$$

be the model space for f_j, where $\mathcal{H}_{j0} = \text{span}\{\phi_{j1}(t), \cdots, \phi_{jm_j}\}$, and $\mathcal{H}_{jk}$'s are RKHS's with RK $R_{jk}(s,t)$ for $k \geq 1$. Then it is easy to check that

$$f(j,t) \in \mathcal{H}_0 \oplus \mathcal{H}_1 \oplus \cdots \oplus \mathcal{H}_{p_1} \oplus \mathcal{H}_{p_1+1} \oplus \cdots \mathcal{H}_{p_1+p_2}, \tag{52}$$

where $\mathcal{H}_0 = \text{span}\{\phi_1(j,t), \cdots, \phi_{m_1}(j,t), \phi_{m_1+1}(j,t), \cdots, \phi_{m_1+m_2}(j,t)\}$, $\phi_k(j,t) = \phi_{1k}(t) \times I_{[j=1]}$ for $1 \leq k \leq m_1$, and $\phi_k(j,t) = \phi_{2k}(t) \times I_{[j=2]}$ for $m_1 + 1 \leq k \leq m_1 + m_2$. $\mathcal{H}_k$ are RKHS's with RKs $R_k((l,s),(j,t)) = R_{1k}(s,t)) \times I_{[l=1]} \times I_{[j=1]}$ for $1 \leq k \leq p_1$ and $R_k((l,s),(j,t)) = R_{2k}(s,t)) \times I_{[l=2]} \times I_{[j=2]}$ for $p_1 + 1 \leq k \leq p_1 + p_2$. The model space (52) is similar to that of a SS ANOVA model. Thus we can use the function `ssr` to fit functions f_j's jointly.

We may reparametrize $f(j,t)$ as

$$\begin{aligned} f(j,t) &= f_1(t) + (f_2(t) - f_1(t)) \times I_{[j=2]} & (53) \\ &= (f_1(t) + f_2(t))/2 + (f_1(t) - f_2(t)) \times (I_{[j=1]} - I_{[j=2]})/2. & (54) \end{aligned}$$

(53) and (54) are SS ANOVA decompositions of $f(j,t)$ with the set-to-zero and sum-to-zero side conditions respectively. This kind of ANOVA decomposition can be carried out for general J.

For (53), let $g_1(t) = f_1(t)$ and $g_2(t) = f_2(t) - f_1(t)$. Let $\mathcal{M}_j$ in (51) be the model space of g_j. Then

$$f(j,t) \in \mathcal{H}_0 \oplus \mathcal{H}_1 \oplus \cdots \oplus \mathcal{H}_{p_1} \oplus \mathcal{H}_{p_1+1} \oplus \cdots \mathcal{H}_{p_1+p_2}, \tag{55}$$

where $\mathcal{H}_0 = \text{span}\{\phi_1(j,t), \cdots, \phi_{m_1}(j,t), \phi_{m_1+1}(j,t), \cdots, \phi_{m_1+m_2}(j,t)\}$, $\phi_k(j,t) = \phi_{1k}(t)$ for $1 \leq k \leq m_1$, and $\phi_k(j,t) = \phi_{2k}(t) \times I_{[j=2]}$ for $m_1+1 \leq k \leq m_1+m_2$. $\mathcal{H}_k$ are RKHS's with RKs $R_k((l,s),(j,t)) = R_{1k}(s,t)$ for $1 \leq k \leq p_1$ and $R_k((l,s),(j,t)) = R_{2k}(s,t) \times I_{[l=2]} \times I_{[j=2]}$ for $p_1 + 1 \leq k \leq p_1 + p_2$.

For (54), denote $g_1(t) = (f_1(t) + f_2(t))/2$ and $g_2(t) = (f_2(t) - f_1(t))/2$. Let $\mathcal{M}_j$ in (51) be the model space of g_j. Then

$$f(j,t) \in \mathcal{H}_0 \oplus \mathcal{H}_1 \oplus \cdots \oplus \mathcal{H}_{p_1} \oplus \mathcal{H}_{p_1+1} \oplus \cdots \mathcal{H}_{p_1+p_2}, \tag{56}$$

where $\mathcal{H}_0 = \text{span}\{\phi_1(j,t), \cdots, \phi_{m_1}(j,t), \phi_{m_1+1}(j,t), \cdots, \phi_{m_1+m_2}(j,t)\}$, $\phi_k(j,t) = \phi_{1k}(t)$ for $1 \leq k \leq m_1$, and $\phi_k(j,t) = \phi_{2k}(t) \times (I_{[j=1]} - I_{[j=2]})$ for $m_1 + 1 \leq k \leq m_1 + m_2$. $\mathcal{H}_k$ are RKHS's with RKs

$R_k((l,s),(j,t)) = R_{1k}(s,t)$ for $1 \le k \le p_1$ and $R_k((l,s),(j,t)) = R_{2k}(s,t) \times (I_{[l=1]} - I_{[l=2]}) \times (I_{[j=1]} - I_{[j=2]})$ for $p_1 + 1 \le k \le p_1 + p_2$.

Often we are interested in possible relationships, if any, between f_1 and f_2. For example, one may want to check if they are equal or parallel. Let P be a probability measure on $\mathcal{T}$. Consider the following SS ANOVA decomposition

$$f(j,t) = \mu + \alpha_j + g_1(t) + g_{12}(j,t), \tag{57}$$

where

$$\begin{aligned}
\mu &= \frac{1}{2}\int_{\mathcal{T}} (f_1(t) + f_2(t)) dP(t), \\
\alpha_j &= \int_{\mathcal{T}} f_j(t) dP(t) - \mu, \quad j = 1, 2, \\
g_1(t) &= \frac{1}{2}(f_1(t) + f_2(t)) - \mu, \\
g_{12}(j,t) &= f_j(t) - \mu - \alpha_j - g_1(t), \quad k = 1, 2.
\end{aligned}$$

We have $\sum_{j=1}^2 \alpha_j = \int_0^1 g_1(t)dt = \sum_{j=1}^2 g_{12}(j,t) = \int_0^1 g_{12}(j,t)dt = 0$. μ is the overall mean, α_j and $g_1(t)$ are the main effects and $g_{12}(j,t)$ is the interaction. (57), equivalent of (54), will make some hypotheses more transparent. It is easy to check that the following hypotheses are equivalent:

$$\begin{aligned}
\mathrm{H}_0: \ f_1(t) = f_2(t) &\iff \mathrm{H}_0: \ \alpha_j + g_{12}(j,t) = 0, \\
\mathrm{H}_0: \ f_1(t) - f_2(t) = \text{constant} &\iff \mathrm{H}_0: \ g_{12}(j,t) = 0, \\
\mathrm{H}_0: \ \int_0^1 f_1(t)dt = \int_0^1 f_2(t)dt &\iff \mathrm{H}_0: \ \alpha_j = 0, \\
\mathrm{H}_0: \ f_1(t) + f_2(t) = \text{constant} &\iff \mathrm{H}_0: \ g_1(t) = 0.
\end{aligned}$$

Furthermore, if $\alpha_j \ne 0$ and $g_1(t) \ne 0$,

$$\mathrm{H}_0: \ af_1(t) + bf_2(t) = c, \ \ |a| + |b| > 0 \iff \mathrm{H}_0: g_{12}(j,t) = \beta\alpha_j g_1(t).$$

Therefore the hypothesis that f_1 and f_2 are equal is equivalent to the j effect $\alpha_j + g_{12}(j,t) = 0$. The hypothesis that f_1 and f_2 are parallel is equivalent to the hypothesis that the interaction $g_{12} = 0$. The hypothesis that the integral of f_1 equals to that of f_2 is equivalent to the hypothesis that the main effect $\alpha_j = 0$. The hypothesis that the summation of f_1 and f_2 is a constant is equivalent to the hypothesis that the main effect $g_1 = 0$. The hypothesis that there exists a linear relationship between the functions f_1 and f_2 is equivalent to the hypothesis that the interaction is multiplicative. Thus, for these simple hypotheses we can fit the SS ANOVA model (57) and perform tests on the corresponding components.

For illustration, we generate a data set from the following model

$$\begin{aligned}
y_{1i} &= \sin(2\pi i/100) + \epsilon_{1i}, \\
y_{2i} &= \sin(2\pi i/100) + 2 \times (i/100) + \epsilon_{2i}, \quad i = 1, \cdots, 100,
\end{aligned}$$

where random errors $(\epsilon_{1i}, \epsilon_{2i})$'s follow bivariate normal with $\mathrm{Var}(\epsilon_{1i}) = .25$, $\mathrm{Var}(\epsilon_{2i}) = 1$ and $\mathrm{Cor}(\epsilon_{1i}, \epsilon_{2i}) = .5$. The variance-covariance structure can be specified with the combination of the `weights` and `correlation` options. We will fit with an arbitrary pairwise variance-covariance structure.

Suppose we want to use cubic splines to model f_1 and f_2 in (50), f_1 and $f_2 - f_1$ in (53), and $(f_1 + f_2)/2$ and $(f_1 - f_2)/2$ in (54). Then $m_1 = m_2 = 2$, $p_1 = p_2 = 1$, $\phi_{11}(t) = \phi_{21}(t) = 1$, $\phi_{12}(t) = \phi_{22}(t) = t - .5$, and R_{11} and R_{21} are the RK of a cubic spline. In the following we first fit f_1 and f_2 using marginal data as `bisp.fit1` and `bisp.fit2` respectively. Then we fit jointly using the formulation (50) as `bisp.fit3` the formulation (53) as `bisp.fit4`, and the formulation (54), or equivalently the formulation (57), as `bisp.fit5`.

```
> options(contrasts=rep("contr.treatment", 2))
> n <- 100
> s1 <- .5
> s2 <- 1
> r <- .5
> A <- diag(c(s1,s2))%*%matrix(c(sqrt(1-r**2),0,r,1),2,2)
> e <- NULL
> for (i in 1:n) e <- c(e,A%*%rnorm(2))
> t <- 1:n/n
> y1 <- sin(2*pi*t) + e[seq(1,2*n,by=2)]
> y2 <- sin(2*pi*t) + 2*t + e[seq(2,2*n,by=2)]
> bisp.dat <- data.frame(y=c(y1,y2),t=rep(t,2),id=rep(c(0,1),rep(n,2)),
                         pair=rep(1:n,2))

# fit separately
> bisp.fit1 <- ssr(y~I(t-.5),rk=cubic(t),spar="m",
                   data=bisp.dat[bisp.dat$id==0,])
> p.bisp.fit1 <- predict(bisp.fit1)
> bisp.fit2 <- ssr(y~I(t-.5),rk=cubic(t),spar="m",
                   data=bisp.dat[bisp.dat$id==1,])
> p.bisp.fit2 <- predict(bisp.fit2)

# fit jointly
> bisp.fit3 <- ssr(y~id*I(t-.5), rk=list(rk.prod(cubic(t),kron(id==0)),
                  rk.prod(cubic(t),kron(id==1))), spar="m",
                  weights=varIdent(form=~1|id),
                  cor=corSymm(form=~1|pair), data=bisp.dat)
> bisp.fit3
...
GML estimate(s) of smoothing parameter(s) : 0.2793703 0.3788567
Equivalent Degrees of Freedom (DF):  11.84544
Estimate of sigma:  0.4616973
Correlation structure of class corSymm representing
 Correlation:
     1
2 0.523
Variance function structure of class varIdent representing
 0        1
 1 2.270982
```

```
> p.bisp.fit3.1 <- predict(bisp.fit3,newdata=bisp.dat[bisp.dat$id==0,],
                           terms=c(1,0,1,0,1,0))
> p.bisp.fit3.2 <- predict(bisp.fit3,newdata=bisp.dat[bisp.dat$id==1,],
                           terms=c(1,1,1,1,0,1))
> p.bisp.fit3.1$pstd <- p.bisp.fit3.1$pstd*sqrt((2*n-bisp.fit3$df)
                        /(2*n-bisp.fit3$df-1))
> p.bisp.fit3.2$pstd <- p.bisp.fit3.2$pstd*sqrt((2*n-bisp.fit3$df)
                        /(2*n-bisp.fit3$df-1))

> bisp.fit4 <- ssr(y~id*I(t-.5), rk=list(cubic(t),
                   rk.prod(cubic(t),kron(id==1))),
                   spar="m", weights=varIdent(form=~1|id),
                   cor=corSymm(form=~1|pair), data=bisp.dat)
> p.bisp.fit4.1 <- predict(bisp.fit4,newdata=bisp.dat[bisp.dat$id==0,],
                           terms=c(1,0,1,0,1,0))
> p.bisp.fit4.2 <- predict(bisp.fit4,newdata=bisp.dat[bisp.dat$id==1,],
                           terms=c(1,1,1,1,1,1))
> p.bisp.fit4.3 <- predict(bisp.fit4,newdata=bisp.dat[bisp.dat$id==1,],
                           terms=c(0,1,0,1,0,1))
> p.bisp.fit4.4 <- predict(bisp.fit4,newdata=bisp.dat[bisp.dat$id==1,],
                           terms=c(0,0,0,1,0,1))
> p.bisp.fit4.1$pstd <- p.bisp.fit4.1$pstd*sqrt((2*n-bisp.fit4$df)
                           /(2*n-bisp.fit4$df-1))
> p.bisp.fit4.2$pstd <- p.bisp.fit4.2$pstd*sqrt((2*n-bisp.fit4$df)
                           /(2*n-bisp.fit4$df-1))
> p.bisp.fit4.3$pstd <- p.bisp.fit4.3$pstd*sqrt((2*n-bisp.fit4$df)
                           /(2*n-bisp.fit4$df-1))
> p.bisp.fit4.4$pstd <- p.bisp.fit4.4$pstd*sqrt((2*n-bisp.fit4$df)
                           /(2*n-bisp.fit4$df-1))
> bisp.fit5 <- ssr(y~id*I(t-.5),
                rk=list(cubic(t),rk.prod(cubic(t),kron((id==0)-(id==1)))),
                spar="m", weights=varIdent(form=~1|id),
                cor=corSymm(form=~1|pair), data=bisp.dat)
> p.bisp.fit5.1 <- predict(bisp.fit5,newdata=bisp.dat[bisp.dat$id==0,],
                   terms=c(1,0,1,0,1,1))
> p.bisp.fit5.2 <- predict(bisp.fit5,newdata=bisp.dat[bisp.dat$id==1,],
                   terms=c(1,1,1,1,1,1))
> p.bisp.fit5.3 <- predict(bisp.fit5,newdata=bisp.dat[bisp.dat$id==1,],
                   terms=c(0,1,0,1,0,1))
> p.bisp.fit5.4 <- predict(bisp.fit5,newdata=bisp.dat[bisp.dat$id==1,],
                   terms=c(0,0,0,1,0,1))
> p.bisp.fit5.1$pstd <- p.bisp.fit5.1$pstd*sqrt((2*n-bisp.fit5$df)
                   /(2*n-bisp.fit5$df-1))
> p.bisp.fit5.2$pstd <- p.bisp.fit5.2$pstd*sqrt((2*n-bisp.fit5$df)
                   /(2*n-bisp.fit5$df-1))
```

```
> p.bisp.fit5.3$pstd <- p.bisp.fit5.3$pstd*sqrt((2*n-bisp.fit5$df)
                     /(2*n-bisp.fit5$df-1))
> p.bisp.fit5.4$pstd <- p.bisp.fit5.4$pstd*sqrt((2*n-bisp.fit5$df)
                     /(2*n-bisp.fit5$df-1))
```

For each fit, we calculate the estimated functions and posterior variances. For fits `bisp.fit4` and `bisp.fit5`, we also calculate the estimates and posterior variances of $f_1 - f_2$ and g_{12} in (57). They are used to check if $f_1 = f_2$ and if they are parallel respectively. Note that we inflate the posterior variances by one more degree of freedom used for estimating the correlation parameter. These estimates are shown in Figures 4, 5 and 6. Even though not obvious in Figure 4, the Bayesian confidence intervals of the joint estimates are uniformly narrower than those of the function-by-function estimates. Obviously from Figures 5 and 6 that these two functions are not equal, nor parallel.

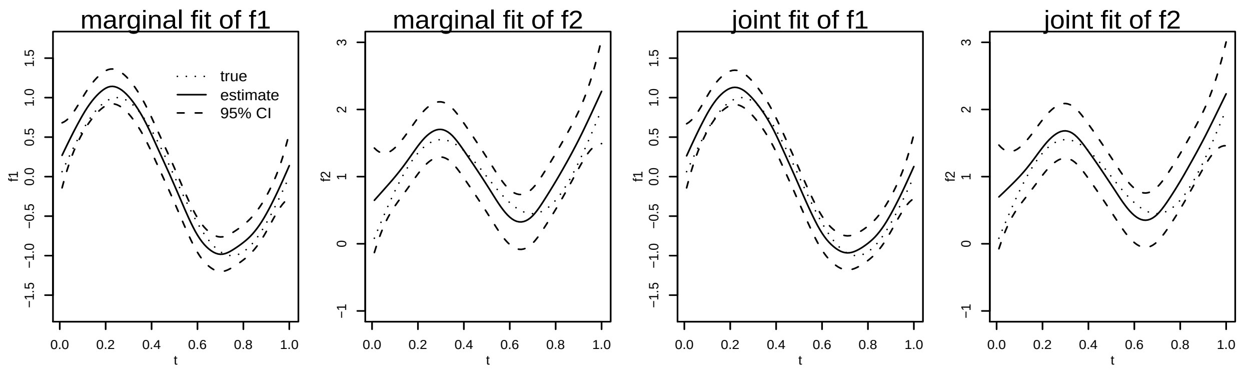


Figure 4: Upper left: estimate of f_1 from `bisp.fit1`. Upper right: estimate of f_2 from `bisp.fit2`. Lower left: estimate of f_1 from `bisp.fit3`. Lower right: estimate of f_2 from `bisp.fit3`. Dashed lines are the true function. Solid lines are the estimates. Dotted lines are 95% Bayesian confidence intervals.

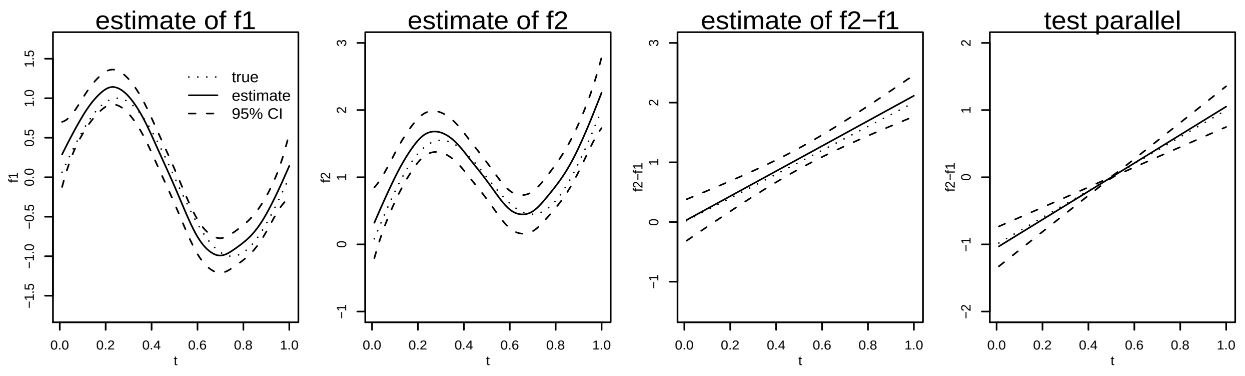


Figure 5: Upper left: estimate of f_1 from `bisp.fit4`. Upper right: estimate of f_2 from `bisp.fit4`. Lower left: estimate of $f_1 - f_2$ from `bisp.fit4`. Lower right: estimate of g_{12} from `bisp.fit4`. Dashed lines are the true function. Solid lines are the estimates. Dotted lines are 95% Bayesian confidence intervals.

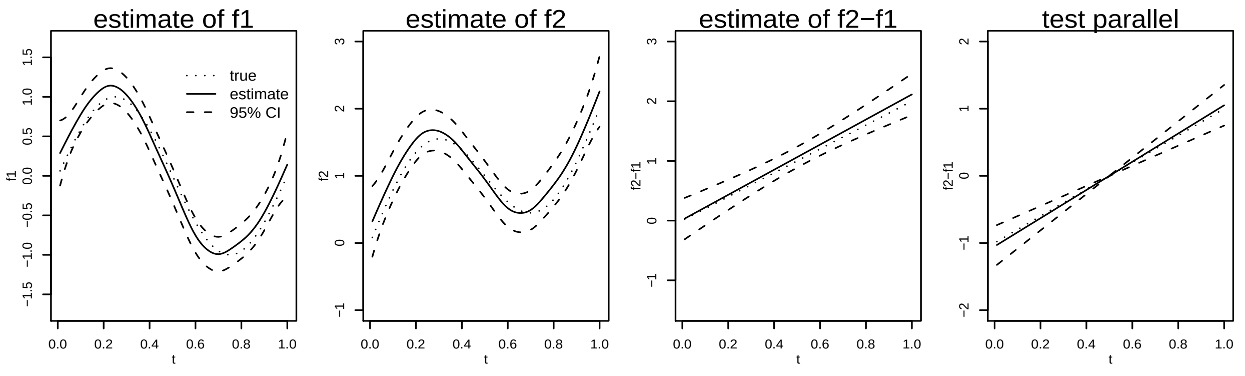


Figure 6: Upper left: estimate of f_1 from `bisp.fit5`. Upper right: estimate of f_2 from `bisp.fit5`. Lower left: estimate of $f_1 - f_2$ from `bisp.fit5`. Lower right: estimate of g_{12} from `bisp.fit5`. Dashed lines are the true function. Solid lines are the estimates. Dotted lines are 95% Bayesian confidence intervals.

8 Examples

Examples in this section are intended to show usage of the functions in `ASSIST`. They are not formal data analyses. All examples were run on R Version 1.7.1 for Linux. Minor differences may occur on different platforms.

8.1 Arosa Ozone Data

This is a data set in Andrews and Herzberg (1985). Monthly mean ozone thickness (Dobson units) in Arosa, Switzerland from 1926 to 1971 was recorded. The data is available as `Arosa`. We are interested in investigating how ozone `thickness` changes over `months` and `years`.

We use this data to illustrate how to fit a periodic spline, a partial spline, an L-spline, an L-spline with unequal variances, an L-spline spline with correlated random errors, a partial spline with both variables `month` and `year`, an SS ANOVA model, partial splines for the whole time series, a semi-parametric linear mixed effects model, and some varying coefficients models.

Let us ignore the `year` effect first and concentrate on the `month` effect on ozone `thickness`. Let `csmonth` $= ($`month` $- .5)/12$. We first fit a parametric sine and cosine model

$$\texttt{thickness}(\texttt{csmonth}) = \mu + \beta_1 \sin(2\pi\texttt{csmonth}) + \beta_2 \cos(2\pi\texttt{csmonth}) + \epsilon(\texttt{csmonth}), \tag{58}$$

and a periodic spline

$$\texttt{thickness}(\texttt{csmonth}) = f(\texttt{csmonth}) + \epsilon(\texttt{csmonth}), \tag{59}$$

where $f \in W_2(per)$.

```
> data(Arosa)
> Arosa$csmonth <- (Arosa$month-0.5)/12
> attach(Arosa)
> ozone.fit1 <- lm(thick~sin(2*pi*csmonth)+cos(2*pi*csmonth), data=Arosa)
> summary(ozone.fit1)
```

```
...
Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)           337.0986     0.7605 443.263  < 2e-16 ***
sin(2 * pi * csmonth) -47.3881     1.0719 -44.209  < 2e-16 ***
cos(2 * pi * csmonth)   7.7966     1.0790   7.226 1.81e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 17.31 on 515 degrees of freedom
Multiple R-Squared: 0.7958,Adjusted R-squared: 0.795
F-statistic:  1003 on 2 and 515 DF,  p-value: < 2.2e-16

> ozone.fit2 <- ssr(thick~1, rk=periodic(csmonth), spar="m", data=Arosa)
> summary(ozone.fit2)
...
Coefficients (d):
 (Intercept)
     337.091

GML estimate(s) of smoothing parameter(s) : 1.717233e-06
Equivalent Degrees of Freedom (DF):  9.0131
Estimate of sigma:  16.78767

> anova(ozone.fit2,simu.size=500)
Testing H_0: f in the NULL space

    test.value simu.size simu.p-value approximate.p-value
LMP  0.2663855       500            0
GML  0.2025431       500            0                   0
```

Figure 7 shows that parts of the parametric sine and cosine fit are outside the confidence intervals of the periodic spline fit. It suggests that the simple parametric model may not be sufficient. We can test the departure from the sine-cosine model using a partial spline model by adding the sine and cosine functions to the null space of a periodic spline

$$\begin{aligned} & \texttt{thickness(csmonth)} \\ = \ & \mu + \beta_1 \sin(2\pi\texttt{csmonth}) + \beta_2 \cos(2\pi\texttt{csmonth}) + f(\texttt{csmonth}) + \epsilon(\texttt{csmonth}), \end{aligned} \tag{60}$$

where $f \in W_2(per) \ominus \{1\}$.

```
> ozone.fit3 <- ssr(thick~sin(2*pi*csmonth)+cos(2*pi*csmonth),
                  rk=periodic(csmonth), spar="m", data=Arosa)
> summary(ozone.fit3)
...
Coefficients (d):
 (Intercept) sin(2 * pi * csmonth) cos(2 * pi * csmonth)
    337.0933             -47.42298              7.696095
```

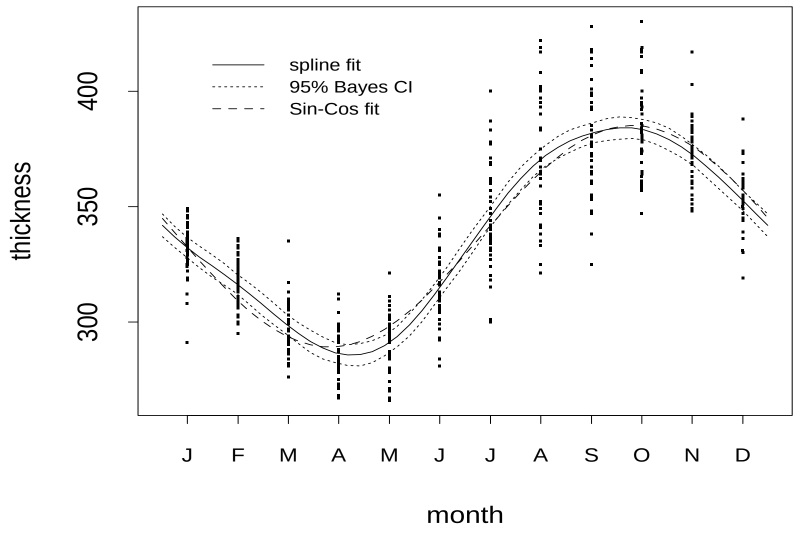


Figure 7: Parametric and periodic spline fits. Points are observations; Solid line is the periodic spline fit; Dotted lines are 95% Bayesian confidence intervals; Dashed line is the parametric fit.

```
GML estimate(s) of smoothing parameter(s) : 4.49682e-06
Equivalent Degrees of Freedom (DF):  7.46103
Estimate of sigma:  16.78036
> anova(ozone.fit3,simu.size=500)
Testing H_0: f in the NULL space

     test.value simu.size simu.p-value approximate.p-value
LMP 0.001262064       500            0
GML   0.9553638       500            0        3.490208e-12
```

The departure from the parametric model is significant. Note that with penalty $\int_0^1 (f'')^2 dt$, sine and cosine functions we added to the null space are also in the space $W_2(per) \ominus \{1\}$. Thus the parametric part of the partial spline model (60) is not orthogonal to $W_2(per) \ominus \{1\}$. Another approach to test the parametric model is to use a periodic L-spline model with $L = D[D^2 + (2\pi)^2]$:

$$\begin{aligned} & \texttt{thickness(csmonth)} \\ = \; & \mu + \beta_1 \sin(2\pi\texttt{csmonth}) + \beta_2 \cos(2\pi\texttt{csmonth}) + f(\texttt{csmonth}) + \epsilon(\texttt{csmonth}), \end{aligned} \tag{61}$$

where $f \in W_2(per) \ominus \{1,\ \sin(2\pi\texttt{csmonth}),\ \cos(2\pi\texttt{csmonth})\}$. Now the sine and cosine functions are orthogonal to f. Thus this approach will be more efficient. See Wang and Brown (1996) for detail.

```
> ozone.fit4 <- update(ozone.fit3, rk=lspline(csmonth,type="sine1"))
> summary(ozone.fit4,simu.size=500)
...
```

```
Coefficients (d):
 (Intercept) sin(2 * pi * csmonth) cos(2 * pi * csmonth)
    337.0937             -47.42247              7.695888

GML estimate(s) of smoothing parameter(s) : 5.404848e-09
Equivalent Degrees of Freedom (DF):  6.421139
Estimate of sigma:  16.77356
> anova(ozone.fit4,simu.size=500)
Testing H_0: f in the NULL space

      test.value simu.size simu.p-value approximate.p-value
LMP 2.539163e-06       500            0
GML    0.9533696       500            0        1.164513e-12
```

Figure 8 shows plots of the overall fits and their projections for three models: `ozone.fit2`, `ozone.fit3` and `ozone.fit4`. As we can see, all three models have similar overall fits. But their projections are quite different. As expected, the confidence intervals for the projections of the L-spline model (61) are narrower than those of the partial spline model (60).

Figure 7 suggests that the variances may not be a constant. We calculate residual variances for each month and plot them on the log scale (left panel of Figure 9). It is obvious that they are not equal. It seems that simple sine and cosine functions can be used to model the variance function.

```
> v <- sapply(split(ozone.fit4$resi,month),var)
> a <- unique(csmonth)
> b <- lm(log(v)~sin(2*pi*a)+cos(2*pi*a))
> summary(b)
...
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      5.43715    0.05763  94.341 8.57e-15 ***
sin(2 * pi * a) -0.71786    0.08151  -8.807 1.02e-05 ***
cos(2 * pi * a) -0.49854    0.08151  -6.117 0.000176 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1996 on 9 degrees of freedom
Multiple R-Squared: 0.9274,Adjusted R-squared: 0.9113
F-statistic: 57.49 on 2 and 9 DF,  p-value: 7.48e-06
```

Therefore we assume the following variance function for model (61)

$$\sigma^2(\texttt{csmonth}) = \mathrm{Var}(\epsilon(\texttt{csmonth})) = \sigma^2 \exp\{2[a\sin(2\pi\texttt{csmonth}) + b\cos(2\pi\texttt{csmonth})]\}.$$

```
> ozone.fit5 <- update(ozone.fit4, weights=varComb(varExp(form=
                ~sin(2*pi*csmonth)), varExp(form=~cos(2*pi*csmonth))))
> summary(ozone.fit5)
...
Coefficients (d):
 (Intercept) sin(2 * pi * csmonth) cos(2 * pi * csmonth)
```

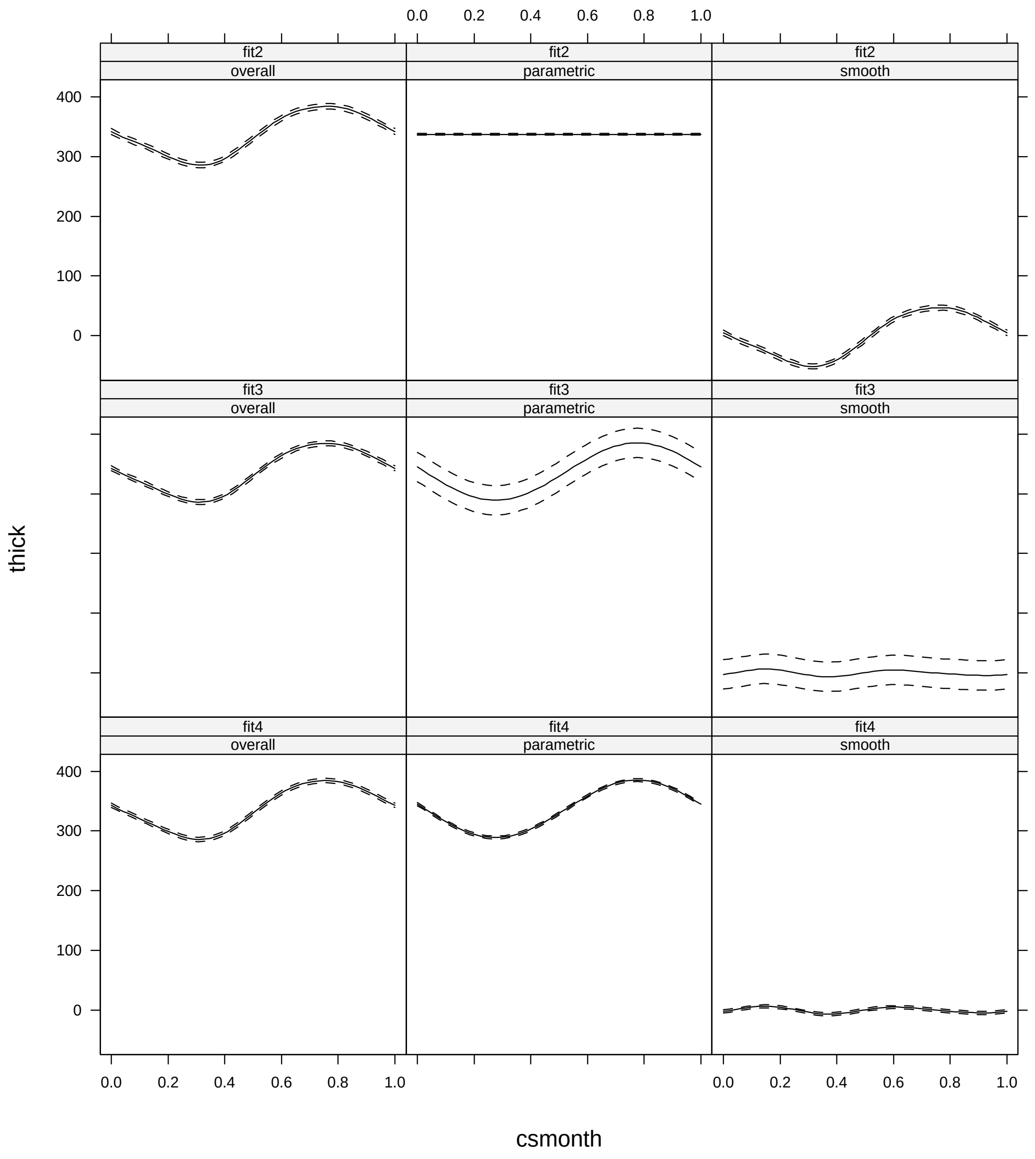


Figure 8: Overall fits and their projections. "Smooth" represents the component in the space $\mathcal{H}_1$. Dotted lines are 95% Bayesian confidence intervals.

```
    337.0753             -47.42951             7.773344

GML estimate(s) of smoothing parameter(s) : 3.676933e-09
Equivalent Degrees of Freedom (DF):  6.84808
Estimate of sigma:  15.22469

Combination of:
Variance function structure of class varExp representing
      expon
 -0.3555964
Variance function structure of class varExp representing
      expon
 -0.2496355
```

The estimated variance parameters, −0.3555964 and −0.2496355, are very close to (up to a scale of 2 by definition) those based on residual variances, −0.7178552 and −0.4985431. The fitted variance function is plotted on the left panel of Figure 9. As can be seen, it is almost identical to the fit based on residual variances. The right panel of Figure 9 plots the L-spline fit to the mean function with 95% Bayesian confidence intervals. Note that these confidence intervals are conditional on the estimated variance parameters. Thus they may have less coverage than the nominal value since the degrees of freedom for estimating the variance parameters are not counted. Nevertheless, we can see that unequal variances are reflected in the widths of these confidence intervals.

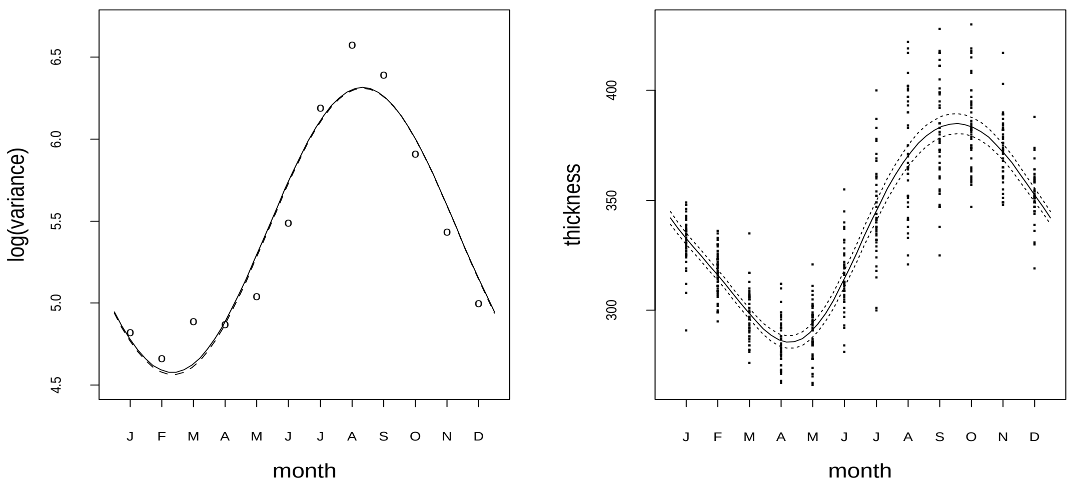


Figure 9: Left: circles are residual variances on the log scale, dotted line is the fit to residual variances, and solid line is the fit from `ozone.fit5`. Right: points are observations, solid line is the L-spline fit, and dotted lines are 95% Bayesian confidence intervals.

Observations close in time may be correlated. In the following we include a first-order autoregressive structure for random errors. Since some observations are missing, we use the continuous AR(1) correlation structure. That is, we assume the covariance between the random error of `month` i_1 in `year`

j_1 and the random error of month i_2 in year j_2 is $\sqrt{\sigma^2((i_1-.5)/12)\sigma^2((i_2-.5)/12)}\rho^{|i_1-i_2|+12|j_1-j_2|}$.

```
> Arosa$z <- month+12*(year-1)
> ozone.fit6 <- update(ozone.fit5,corr=corCAR1(form=~z))
> summary(ozone.fit6)
...
Coefficients (d):
 (Intercept) sin(2 * pi * csmonth) cos(2 * pi * csmonth)
    336.8969             -47.34584              7.788939

GML estimate(s) of smoothing parameter(s) : 3.573493e-09
Equivalent Degrees of Freedom (DF):  7.326922
Estimate of sigma:  15.31061

Correlation structure of class corCAR1 representing
       Phi
 0.3410923
Combination of:
Variance function structure of class varExp representing
      expon
 -0.3602399
Variance function structure of class varExp representing
      expon
 -0.3009847
```

Now let us consider the effects of both month and year variables. First, suppose that the simple sine and cosine functions are appropriate for modeling the month effect, and we do not want to assume a parametric model for the year effect. Then we may consider the following partial spline model

$$\begin{aligned}&\texttt{thickness}(\texttt{csmonth},\texttt{csyear})\\ =\ &\beta_1\sin(2\pi\texttt{csmonth})+\beta_2\cos(2\pi\texttt{csmonth})+f(\texttt{csyear})+\epsilon(\texttt{csmonth},\texttt{csyear}),\end{aligned}\qquad(62)$$

where $\texttt{csyear}=(\texttt{year}-1)/45$ and $f\in W_2([0,1])$.

```
> csyear <- (year-1)/45
> ozone.fit7 <- ssr(thick~sin(2*pi*csmonth)+cos(2*pi*csmonth)+I(csyear-0.5),
                   rk=cubic(csyear), spar="m", data=Arosa)
> summary(ozone.fit7)
...
Coefficients (d):
 (Intercept) sin(2 * pi * csmonth) cos(2 * pi * csmonth) I(csyear - 0.5)
    336.8798             -47.37047              7.776179        7.199672

GML estimate(s) of smoothing parameter(s) : 4.741079e-07
Equivalent Degrees of Freedom (DF):  16.49958
Estimate of sigma:  16.5235
> anova(ozone.fit7,simu.size=500)
Testing H_0: f in the NULL space
```

```
    test.value simu.size simu.p-value approximate.p-value
LMP 0.01157664       500        0.026
GML  0.9891654       500            0        0.0004092629
```

If we want to model both `month` and `year` non-parametrically, we can fit the following SS ANOVA model with a periodic spline for `month` and a cubic spline for `year`:

$$\begin{aligned} & \texttt{thickness}(\texttt{csmonth}, \texttt{csyear}) \\ = \; & \mu + \beta\texttt{csyear} + s_1(\texttt{csmonth}) + s_2(\texttt{csyear}) + sl(\texttt{csmonth}, \texttt{csyear}) \\ & + ss(\texttt{csmonth}, \texttt{csyear}) + \epsilon(\texttt{csmonth}, \texttt{csyear}). \end{aligned} \tag{63}$$

```
> ozone.fit8 <- ssr(thick~I(csyear-0.5),  spar="m", data=Arosa,
                    rk=list(periodic(csmonth), cubic(csyear),
                    rk.prod(periodic(csmonth), kron(csyear-.5)),
                    rk.prod(periodic(csmonth), cubic(csyear))))
> summary(ozone.fit8)
...
Coefficients (d):
 (Intercept) I(csyear - 0.5)
    336.8298        6.031531

GML estimate(s) of smoothing parameter(s) :
4.108999e-01 7.457006e-02 1.656897e+05 5.438436e+03
Equivalent Degrees of Freedom (DF):  24.59541
Estimate of sigma:  15.84154
> grid <- data.frame(csmonth=seq(0,1,len=50), csyear=seq(0,1,len=50))
> interaction.fit8 <- predict(ozone.fit8, grid, terms=c(0,0,0,0,1,1))
> max(abs(interaction.fit8$fit)/interaction.fit8$pstd)
[1] 0.009646192
```

The smoothing parameters for the interaction terms between `month` and `year` are large, which indicates that the interaction effects are small. Indeed the fitted values of interaction terms are very small (around 10^{-4}) compared to posterior standard deviations (around 0.01). Therefore, we delete these interaction terms in our final model.

```
> ozone.fit9 <- update(ozone.fit8, rk=list(periodic(csmonth), cubic(csyear)))
> summary(ozone.fit9)
...
Coefficients (d):
 (Intercept) I(csyear - 0.5)
    336.8287        5.989813

GML estimate(s) of smoothing parameter(s) : 0.41248034 0.07316391
Equivalent Degrees of Freedom (DF):  24.66142
Estimate of sigma:  15.8378
```

Figure 10 shows the estimated main effects of `month` and `year`.

We may also consider observations as a long time series. Note that many observations are missing, thus the state space approach in Kitagawa and Gersch (1984) cannot be used directly. Our approach

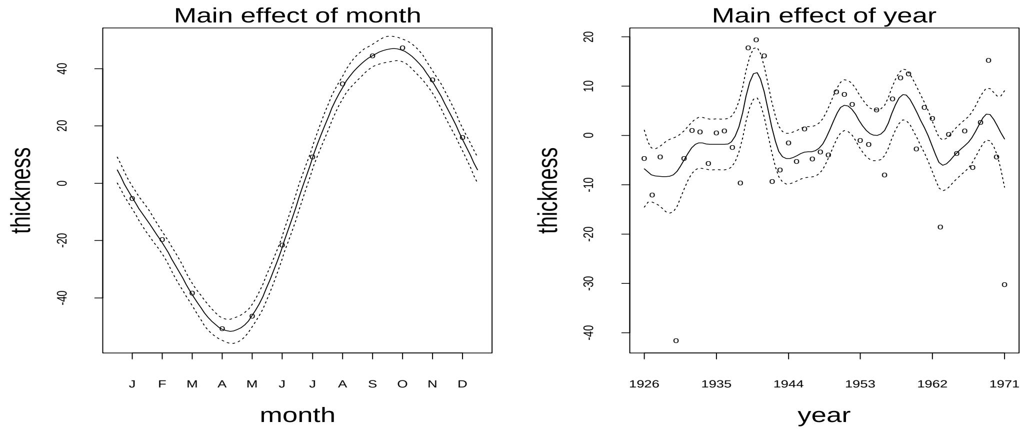


Figure 10: Estimated main effects with 95% Bayesian confidence intervals. A dot in the left panel is the average thickness for a particular month minus the overall mean. A dot in the right panel is the average thickness for a particular year minus the overall mean.

allows observations at unequally spaced points. Thus it can also be used to impute missing observations. Let us define a time variable as $t = \texttt{csmonth} + \texttt{year} - 1$. Similar to Kitagawa and Gersch (1984), we may consider the following model

$$\texttt{thickness}(t) = \mu + \beta_1 \sin(2\pi t) + \beta_2 \cos(2\pi t) + f(t) + \epsilon(t), \tag{64}$$

where $\sin(2\pi t)$ and $\cos(2\pi t)$ model seasonal trend, f models the long term trend and $f \in W_2([0,T]) \ominus \{1\}$, $T = 45.45833$ is the maximum value of t, and $\epsilon(t)$'s are random errors and $\epsilon(t) \overset{iid}{\sim} \mathrm{N}(0, \sigma^2)$. Model (64) is a partial spline. Since the domain is $[0,T]$ with $T \neq 1$, we use the `cubic2` kernel function.

```
> Arosa$t <- csmonth+Arosa$year-1
> ozone.fit10 <- ssr(thick~t+sin(2*pi*t)+cos(2*pi*t), rk=cubic2(t),
                     spar="m", data=Arosa)
> summary(ozone.fit10)
...
Coefficients (d):
    (Intercept)              t sin(2 * pi * t) cos(2 * pi * t)
    330.8107521     -0.9901175     -47.3246660       7.7786452

GML estimate(s) of smoothing parameter(s) : 0.01568941
Equivalent Degrees of Freedom (DF):  20.38134
Estimate of sigma:  16.25793
> anova(ozone.fit10, simu.size=500)
Testing H_0: f in the NULL space

    test.value simu.size simu.p-value approximate.p-value
LMP   1075.457       500         0.01
GML    0.98481       500            0         3.64301e-05
```

Estimate of the overall function and its two components, seasonal trend and long term trend, are shown in Figure 11. We can see that the long term trend is different from a constant.

Observations close in time are likely to be correlated. Suppose that we want to use the exponential correlation structure with nugget effect. Specifically, regard $\epsilon(t)$ as a zero mean stochastic process with

$$\mathrm{Cov}(\epsilon(s), \epsilon(t)) = \begin{cases} \sigma^2(1-c)\exp(-|s-t|/r), & s \neq t, \\ \sigma^2, & s = t, \end{cases} \tag{65}$$

where c is the nugget effect, and r is the range parameter.

```
> ozone.fit11 <- update(ozone.fit10, corr=corExp(form=~t,nugget=T))
> ozone.fit11
...
GML estimate(s) of smoothing parameter(s) : 469618845337
Equivalent Degrees of Freedom (DF):  4
Estimate of sigma:  17.36963
Correlation structure of class corExp representing
    range    nugget
0.3661093 0.6364710
```

New estimates are plotted in Figure 12. We can see that now the long term trend is almost a constant. The equivalent degrees of freedom is decreased to 4.008176.

The null space contains four components: the constant, linear, sine and cosine functions. Using the `cubic2` kernel penalizes the sine and cosine functions which may lead to biases. If this is to be avoided, we can use the `lspline` kernel with `type=''linSinCos''`. The period for the seasonal component is 1, while the period for the function `lspline` with `type=''linSinCos''` is 2π. Therefore we multiply the covariate t by a constant 2π to match the scale.

```
> ozone.fat12 <- update(ozone.fit11, rk=lspline(2*pi*t,type="linSinCos"))
> ozone.fit12
...
GML estimate(s) of smoothing parameter(s) : 9.562702e+13
Equivalent Degrees of Freedom (DF):  4
Estimate of sigma:  17.37282
Correlation structure of class corExp representing
    range    nugget
0.3665007 0.6360280
```

The estimates are plotted in Figure 13.

As in Kitagawa and Gersch (1984), sometimes we want to use a stochastic process to model the autocorrelation and regard this process as part of the signal. Then we need to separate this process with measurement errors and predict it at desired points. Specifically, assume

$$\texttt{thickness}(t) = \mu + \beta_1 \sin(2\pi t) + \beta_2 \cos(2\pi t) + f(t) + u(t) + \epsilon(t), \tag{66}$$

where f and ϵ are the same as in model (64). Assume that $u(t)$ is a stochastic process independent of $\epsilon(t)$ with mean zero and $\mathrm{Cov}(u(s), u(t)) = \sigma_1^2 \exp(-|s-t|/r)$, where r is the range parameter as in (65).

Denote $\boldsymbol{t}$ as the vector of design points for the variable t. Let $u(\boldsymbol{t})$ be the vector of the u process evaluated at design points. $u(\boldsymbol{t})$ are random effects and $u(\boldsymbol{t}) \sim \mathrm{N}(0, \sigma_1^2 D)$, where D depends the

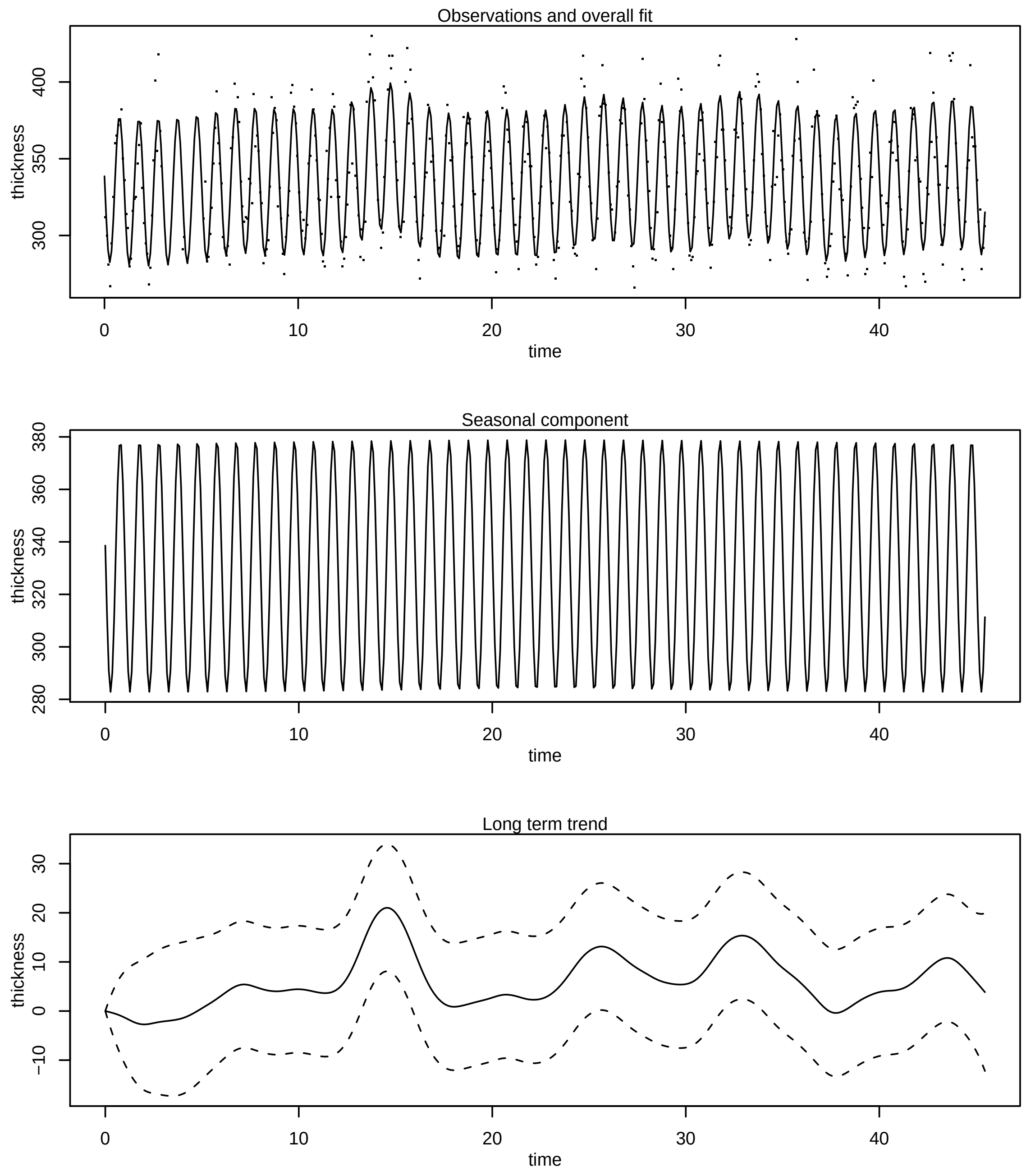


Figure 11: Plots of estimates from `ozone.fit10`. Above: observations as dots and estimated overall function as the solid line. Middle: estimated seasonal trend. Below: estimated long term trend as the solid line and 95% Bayesian confidence intervals as two dotted lines.

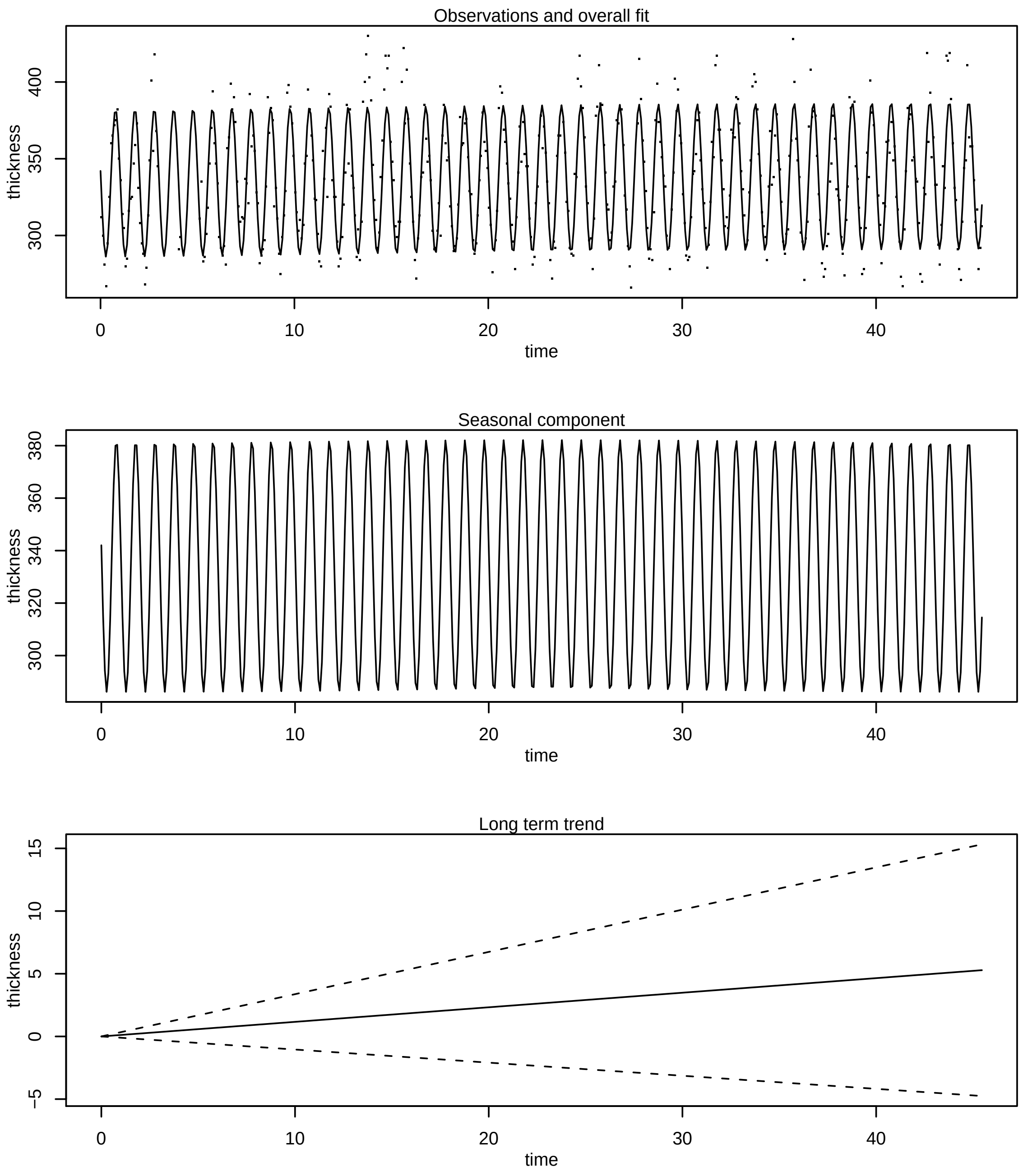
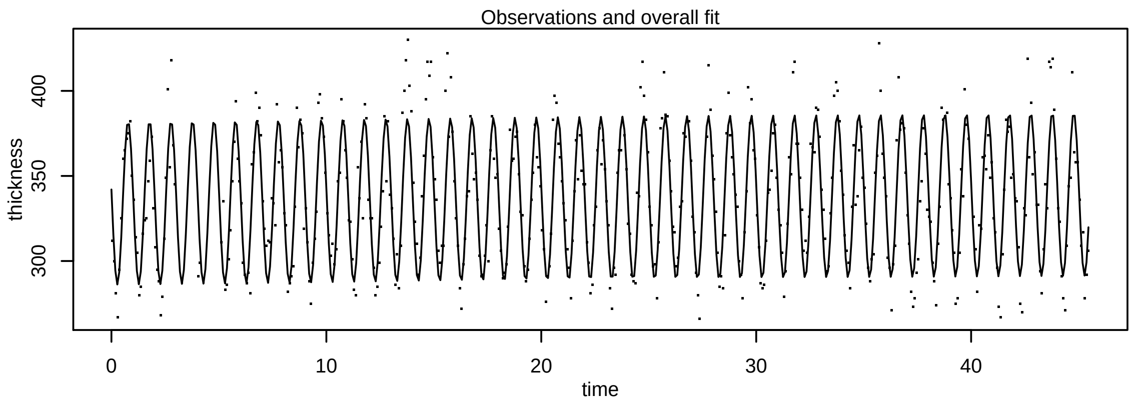
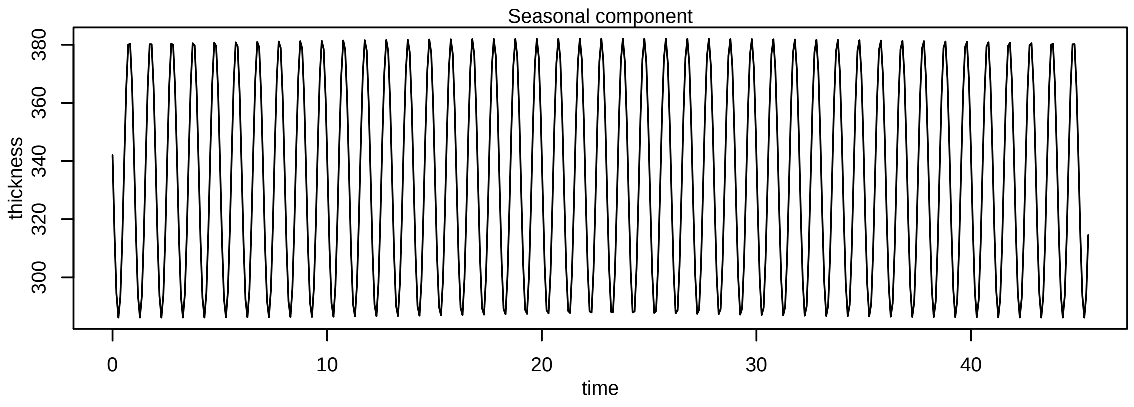


Figure 12: Plots of estimates from `ozone.fit11`. Above: observations as dots and estimated overall function as the solid line. Middle: estimated seasonal trend. Below: estimated long term trend as the solid line and 95% Bayesian confidence intervals as two dotted lines.





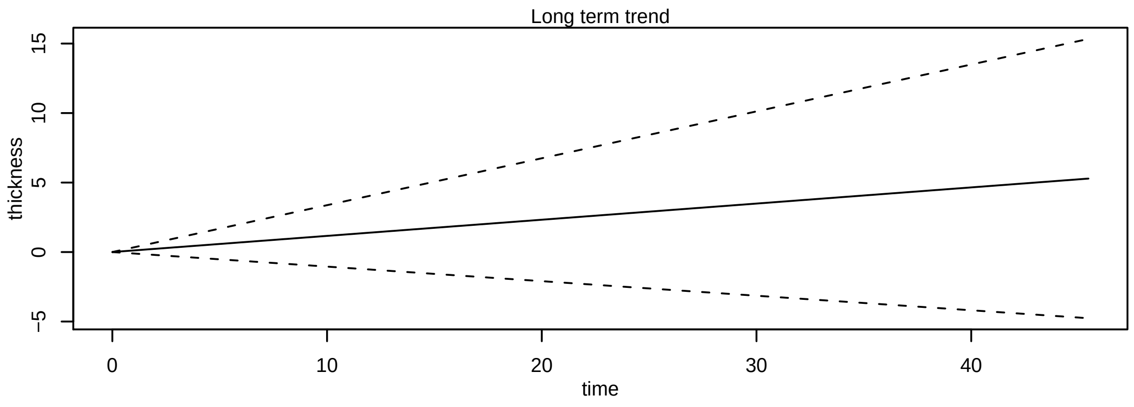


Figure 13: Plots of estimates from `ozone.fit12`. Above: observations as dots and estimated overall function as the solid line. Middle: estimated seasonal trend. Below: estimated long term trend as the solid line and 95% Bayesian confidence intervals as two dotted lines.

parameter r. (66) is a SLM model. However, it cannot be fitted directly using `slm` since D depends on the range parameter r nonlinearly. We fit model (66) in two steps. We first regard $u(\boldsymbol{t})$ as part of random error and estimate the range parameter. This is done in `ozone.fit11`. Then we calculate the estimate of D (without nugget effect) and regard it as the true covariance matrix. We calculate the Cholesky decomposition of D as $D = ZZ^T$, and transform the random effects $u(\boldsymbol{t}) = Z\boldsymbol{b}$, where $\boldsymbol{b} \sim \mathrm{N}(0, \sigma_1^2 I)$. Now we are ready to fit the transformed SLM:

```
> tau <- coef(ozone.fit11$cor.est, F)
> D <- corMatrix(initialize(corExp(tau[1],form=~t), data=Arosa))
> Z <- chol.new(D)
> ozone.fit13 <- slm(thick~t+sin(2*pi*t)+cos(2*pi*t), rk=cubic2(t),
                     random=list(pdIdent(~Z-1)), data=Arosa)
> summary(ozone.fit13)
...
Coefficients (d):
    (Intercept)               t sin(2 * pi * t) cos(2 * pi * t)
    332.2848808       0.3174285     -47.2930087       7.8994571

GML estimate(s) of smoothing parameter(s) : 50.62773
Equivalent Degrees of Freedom (DF):  4.517319
Estimate of sigma:  13.86935
```

Suppose that we want to predict u on grid points $\boldsymbol{s}$, $u(\boldsymbol{s})$. Let $R = \mathrm{Cov}(u(\boldsymbol{s}), u(\boldsymbol{t}))$. Then $\hat{u}(\boldsymbol{s}) = RD^{-1}\hat{u}(\boldsymbol{t}) = RZ^{-T}\hat{\boldsymbol{b}}$.

```
> grid3 <- data.frame(t=seq(0,max(Arosa$t)+0.001,len=500))
> newdata <- data.frame(t=c(Arosa$t,grid3$t))
> RD <- corMatrix(initialize(corExp(tau[1],form=~t), data=newdata))
> R <- RD[(length(Arosa$t)+1):length(newdata$t),1:length(Arosa$t)]
> u.new <- R%*%t(solve(Z))%*%as.vector(ozone.fit13$lme.obj$coef$random[[2]])
```

Estimates from `ozone.fit13` are shown in Figure 14.

We now show how to use `nnr` and `snr` to fit varying coefficients models. Suppose that the `thickness` in each year can be well approximated by a sinusoidal function of `month`. We want to investigate how two coefficients, the average `thickness` and amplitude, change over `years`. Specifically, we consider the following model:

$$\begin{aligned} & \texttt{thickness}(\texttt{csmonth}, \texttt{csyear}) \\ = \; & f_1(\texttt{csyear}) + f_2(\texttt{csyear}) \cos 2\pi \left(\texttt{csmonth} + \mathrm{alogit}(\alpha)\right) + \epsilon(\texttt{csmonth}, \texttt{csyear}), \end{aligned} \tag{67}$$

where $f_1(\texttt{csyear})$ and $f_2(\texttt{csyear})$ are yearly average `thickness` and amplitude respectively, $\mathrm{alogit}(\alpha) = \exp(\alpha)/(1+\exp(\alpha))$ guarantees the horizontal shift to be between 0 and 1. Assume that $f_1, f_2 \in W_2([0,1])$. Note that the constraint $f_2 \geq 0$ is not enforced here. We may replace f_2 by $\exp(f_2)$ as the amplitude function. The resulting model will be fitted later. Model (67) is a special case of the SNR model, and thus can be fitted using the function `snr`. We use the coefficients from `ozone.fit1` to calculate the initial value for α.

```
> tmp <- atan(-ozone.fit1$coef[3]/ozone.fit1$coef[2])/(2*pi)
> tmp <- log(tmp/(1-tmp))
> ozone.fit14 <- snr(thick~f1(csyear)+f2(csyear)*cos(2*pi*(csmonth+alogit(a))),
                    func=list(f1(x)+f2(x)~list(~x, cubic(x))), params=list(a~1),
```

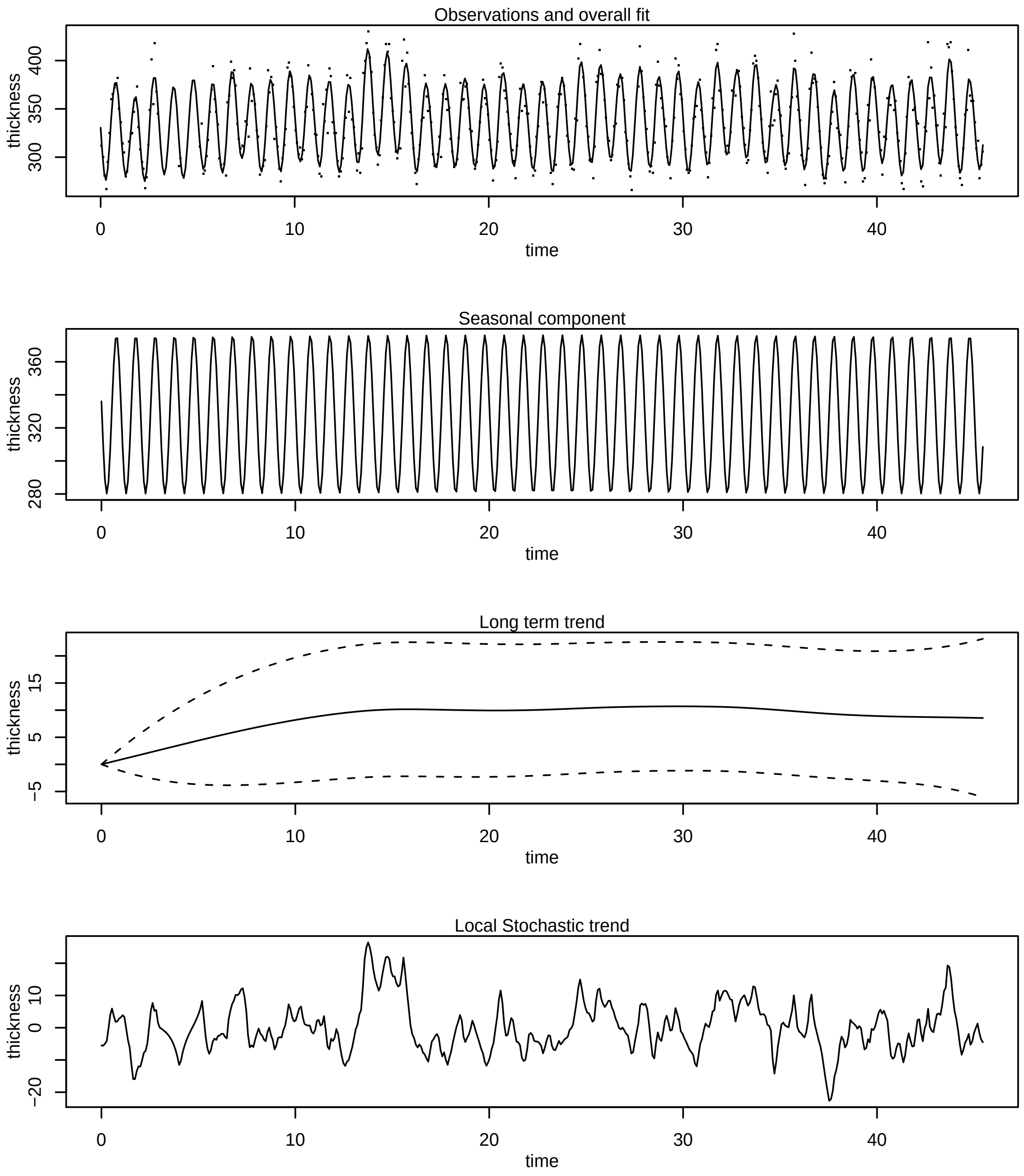


Figure 14: Plots of estimates from `ozone.fit13`. First row: observations as dots and estimated overall function as the solid line. Second row: estimated seasonal trend. Third row: estimated long term trend as the solid line and 95% Bayesian confidence intervals as two dotted lines. Fourth row: estimated local stochastic trend.

```
                   data=Arosa, start=list(params=c(tmp)), spar="m")
> summary(ozone.fit14)
Semi-parametric Nonlinear Regression Model Fit by Gauss-Newton Method
Model: thick ~ f1(csyear) + f2(csyear) * cos(2 * pi * (csmonth + alogit(a)))
Data: Arosa

       AIC      BIC    logLik
  4362.429 4370.929 -2179.214

Coefficients:
      Value  Std.Error   t-value p-value
a -1.243126 0.01933295 -64.30086       0

Standardized residuals:
        Min          Q1         Med          Q3         Max
-3.37533987 -0.60538971 -0.03263471  0.49885054  3.13869256

GML estimate(s) of smoothing spline parameter(s): 0.0001225915 0.4999998276
Equivalent Degrees of Freedom (DF) for spline function:  16.95087
Residual standard error: 16.81619

Converged after 3 iterations

> p.ozone.fit14 <- intervals(ozone.fit14,data.frame(x=seq(0,1,len=100)),
            terms=list(f1=matrix(c(1,1,1,1,1,0,0,0,1),ncol=3,byrow=T),
                       f2=matrix(c(1,1,1,1,1,0,0,0,1),ncol=3,byrow=T)))
```

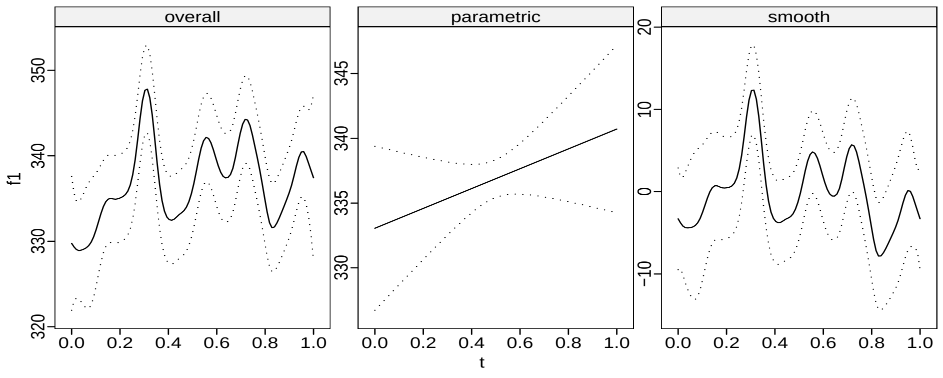


Figure 15: Estimated f_1 (overall), and its projections to $\mathcal{H}_0$ (parametric) and $\mathcal{H}_1$ (smooth). Dotted lines are 95% Bayesian confidence intervals.

Figures 15 and 16 show the estimated functions and their projections. The yearly average `thickness` has a similar trend as before, and the amplitude has a linear increasing, but non-significant, trend.

The sinusoidal function for monthly changes may be too restrictive. Then we can consider the

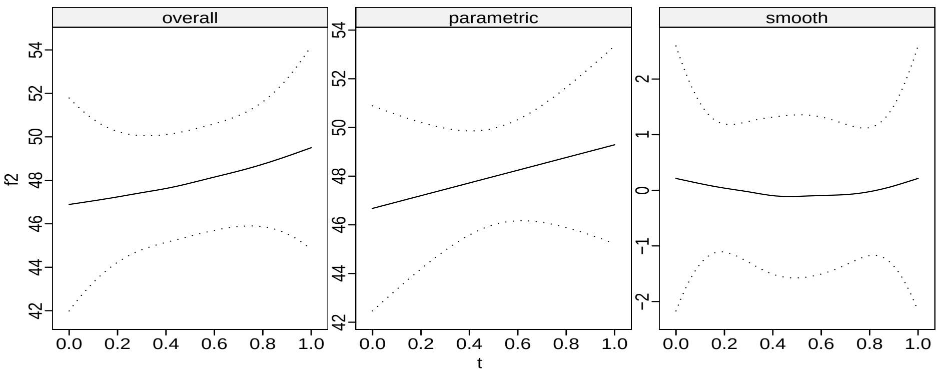


Figure 16: Estimated f_2 (overall), and its projections to $\mathcal{H}_0$ (parametric) and $\mathcal{H}_1$ (smooth). Dotted lines are 95% Bayesian confidence intervals.

following model

$$\begin{aligned} & \texttt{thickness}(\texttt{csmonth}, \texttt{csyear}) \\ = \; & f_1(\texttt{csyear}) + f_2(\texttt{csyear}) f_3(\texttt{csmonth}) + \epsilon(\texttt{csmonth}, \texttt{csyear}), \end{aligned} \tag{68}$$

where the cos function in model (67) is replaced by a nonparametric periodic function f_3. Assume that $f_3 \in W_2(per) \ominus \{1\}$. The constant functions were removed from the model space for identifiability. Thus we have $\int_0^1 f_3(t)dt = 0$. Also, for identifiability, we assume $\sup|f_3(t)| = 1$. Again, we did not enforce the constraint $f_2 \geq 0$. Model (68) is a special case of the NNR model, thus can be fitted using `nnr`. In the following, instead of using `nnr` directly, we write a simple program using backfitting methods to estimate f_1, f_2 and f_3. The purpose is to show that writing S codes for more complicated models is fairly easy.

```
# transform time variable into [0,1]
z <- (Arosa$t-min(Arosa$t))/(max(Arosa$t)-min(Arosa$t))

# create matrices for RK's
S1 <- S2 <- cubic(z)
S3 <- periodic((max(Arosa$t)-min(Arosa$t))*z)

# find initial values
f3.tmp <- ssr(thick~1,rk=S3,data=Arosa,spar="m")
f3.est <- as.vector(S3%*%f3.tmp$rkpk.obj$c/max(abs(S3%*%f3.tmp$rkpk.obj$c)))
f2.est <- rep(max(abs(S3%*%f3.tmp$rkpk.obj$c)),length(thick))
f1.est <- rep(f3.tmp$rkpk.obj$d[1],length(thick))

# backfitting
prec <- 1
while (prec>.0001) {
    # fix f2 and f3, fit f1
    y.tmp <- thick-f2.est*f3.est
    f1.tmp <- ssr(y.tmp~z,rk=S1,spar="m",limnla=c(-3,1))
    f1.est.old <- f1.est
    f1.est <- f1.tmp$fit

    # fix f1 and f3, fit f2. Note RK are multiplied by f3
    y.tmp <- thick-f1.tmp$fit
    f2.tmp <- ssr(y.tmp~f3.est+I(f3.est*z)-1,rk=rk.prod(S2,f3.est),
                  spar="m",limnla=c(-3,1))
    f2.est.old <- f2.est
    f2.est <- f2.tmp$rkpk.obj$d[1]+f2.tmp$rkpk.obj$d[2]*z
             +as.vector(S2%*%f2.tmp$rkpk.obj$c)

    # fix f1 and f2, fit f3, note the empty null space and weights
    f3.tmp <- ssr(I(y.tmp/f2.est)~-1,rk=S3,spar="m",weights=f2.est)
    f3.est.old <- f3.est
    # enforce \sup |f_3(t)|=1
    f3.est <- as.vector(S3%*%f3.tmp$rkpk.obj$c/
              max(abs(S3%*%f3.tmp$rkpk.obj$c)))

    # stopping criteria
    prec <- max(sum((f1.est-f1.est.old)**2),
                sum((f2.est-f2.est.old)**2),
                sum((f3.est-f3.est.old)**2))
}

# for prediction of f3 on new grid, we need to refit which specifies the
# rk function rk=periodic. rk= a exist matrix will not work
y.tmp <- thick-f1.tmp$fit
f3.tmp <- ssr(I(y.tmp/f2.est)~-1,
              rk=periodic((max(Arosa$t)-min(Arosa$t))*z),
              spar="m",weights=f2.est)

# calculate fits for the plot.bCI function.
# Note we inflate the posterior variances via degrees of freedom
p.ozone.fit15.f1 <- predict(f1.tmp,terms=matrix(c(1,1,1,1,1,0,0,0,1),ncol=3,
                            byrow=T))
p.ozone.fit15.f1$pstd <- p.ozone.fit15.f1$pstd*sqrt((length(thick)-
                f1.tmp$df)/(length(thick)-f1.tmp$df-f2.tmp$df-f3.tmp$df+1))
p.ozone.fit15.f2 <- predict(f2.tmp,terms=matrix(c(1,1,1,1,1,0,0,0,1),ncol=3,
                            byrow=T))
p.ozone.fit15.f2$fit <- p.ozone.fit15.f2$fit/f3.est
p.ozone.fit15.f2$pstd <- p.ozone.fit15.f2$pstd*sqrt((length(thick)-
     f2.tmp$df)/(length(thick)-f1.tmp$df-f2.tmp$df-f3.tmp$df+1))/abs(f3.est)
grid4 <- data.frame(z=seq(0,1/(max(Arosa$t)-min(Arosa$t)),len=100))
p.ozone.fit15.f3 <- predict(f3.tmp,grid4)
p.ozone.fit15.f3$pstd <- p.ozone.fit14.f3$pstd*sqrt((length(thick)-
f3.tmp$df)/(length(thick)-f1.tmp$df-f2.tmp$df-f3.tmp$df+1))
```

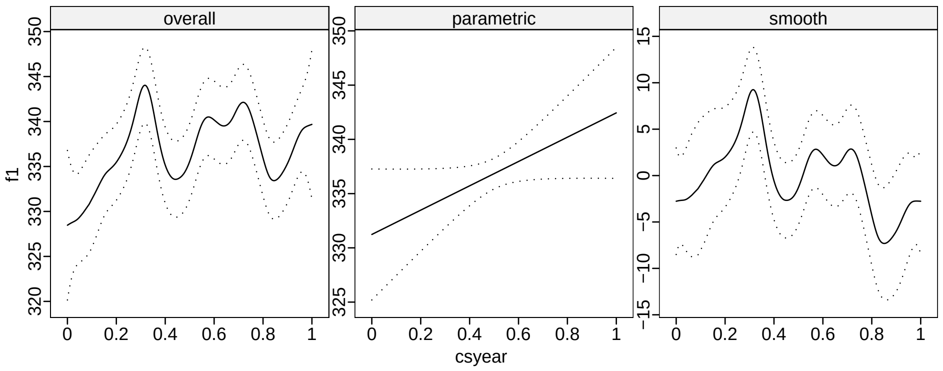


Figure 17: Estimated f_1 (overall), and its projections to $\mathcal{H}_0$ (parametric) and $\mathcal{H}_1$ (smooth). Dotted lines are 95% Bayesian confidence intervals.

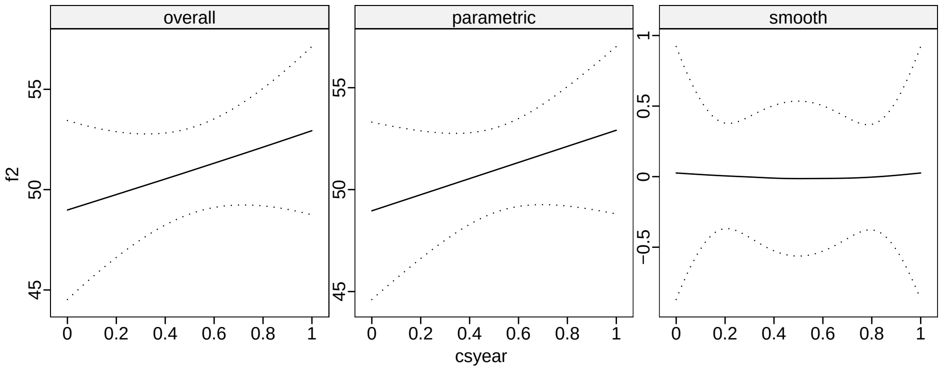


Figure 18: Estimated f_2 (overall), and its projections to $\mathcal{H}_0$ (parametric) and $\mathcal{H}_1$ (smooth). Dotted lines are 95% Bayesian confidence intervals.

Figures 17, 18 and 19 show the estimated functions and their projections. Yearly average `thickness` f_1 and amplitude f_2 have similar trends as before. f_3 is different from a sinusoidal function, even though the difference is small.

To enforce the constraint $f_2 \geq 0$ in model (68), we consider the following model

$$\begin{aligned} & \texttt{thickness}(\texttt{csmonth}, \texttt{csyear}) \\ = \; & f_1(\texttt{csyear}) + \exp(f_2(\texttt{csyear})) f_3(\texttt{csmonth}) + \epsilon(\texttt{csmonth}, \texttt{csyear}), \end{aligned} \tag{69}$$

where $f_1 \in W_2([0,1])$ and $f_3 \in W_2(per) \ominus \{1\}$. Since f_3 is close to a sinusoidal function, we use the L-spline with $L = D^2 + (2\pi)^2$. For identifiability, we remove the constant functions from the model space for f_2: $f_2 \in W_2([0,1]) \ominus \{1\}$.

```
> S3 <- cubic(z)
> f3.tmp <- ssr(thick~1,rk=S3,data=Arosa,spar="m")
> f3.ini <- as.vector(S3%*%f3.tmp$rkpk.obj$c)
```

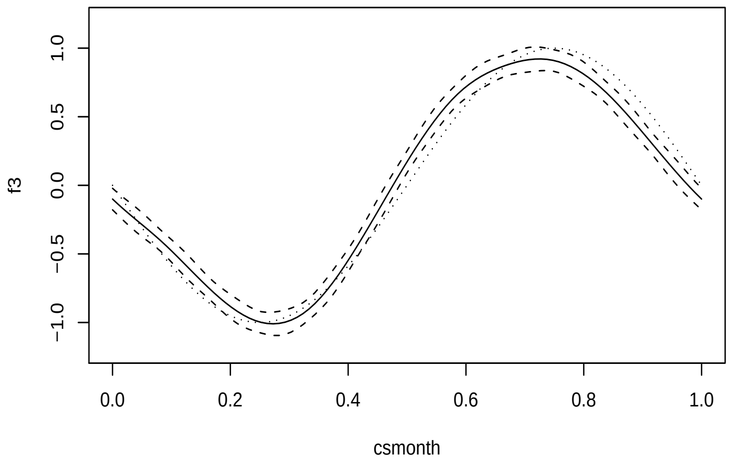


Figure 19: Solid line is the estimated f_3. Dotted lines are 95% Bayesian confidence intervals. Dashed line is the sine function.

```
> ozone.fit16 <- nnr(thick~f1(csyear)+exp(f2(csyear))*f3(csmonth),
                     func=list(f1(x)~list(~I(x-.5),cubic(x)),
                               f2(x)~list(~I(x-.5)-1,cubic(x)),
                               f3(x)~list(~sin(2*pi*x)+cos(2*pi*x)-1,
                               lspline(x,type="sine0"))),
                     data=Arosa,
                     start=list(f1=mean(thick),f2=0,f3=f3.ini),
                     control=list(converg="coef"))
> ozone.fit16
Nonlinear Nonparametric Regression Model Fit by Gauss-Newton Method
 Model: thick ~ f1(csyear) + exp(f2(csyear)) * f3(csmonth)
 Data: Arosa

 GML estimate(s) of smoothing parameter(s): 2.081637e-07 1.930501e-03 1.053201e-06
 Equivalent Degrees of Freedom (DF):  34.20186

 Residual standard error: 15.62308
 Number of Observations: 518
 Converged after 7 iterations

> x <- seq(0,1,len=50)
> u <- seq(0,1,len=50)
> p.ozone.fit16 <- intervals(ozone.fit16, newdata=list(csyear=x,csmonth=u),
               terms=list(f1=matrix(c(1,1,1,1,1,0,0,0,1),nrow=3,byrow=T),
                          f2=matrix(c(1,1,1,0,0,1),nrow=3,byrow=T),
                      f3=matrix(c(1,1,1,1,1,0,0,0,1),nrow=3,byrow=T)))
```

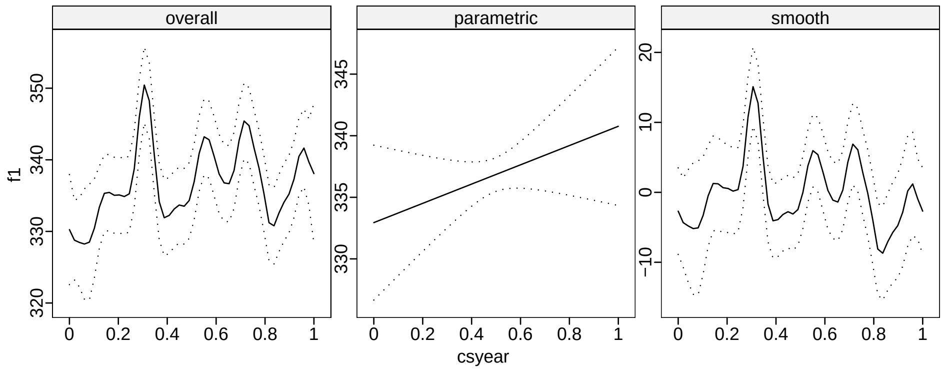


Figure 20: Estimated f_1 (overall), and its projections to $\mathcal{H}_0$ (parametric) and $\mathcal{H}_1$ (smooth). Dotted lines are 95% Bayesian confidence intervals.

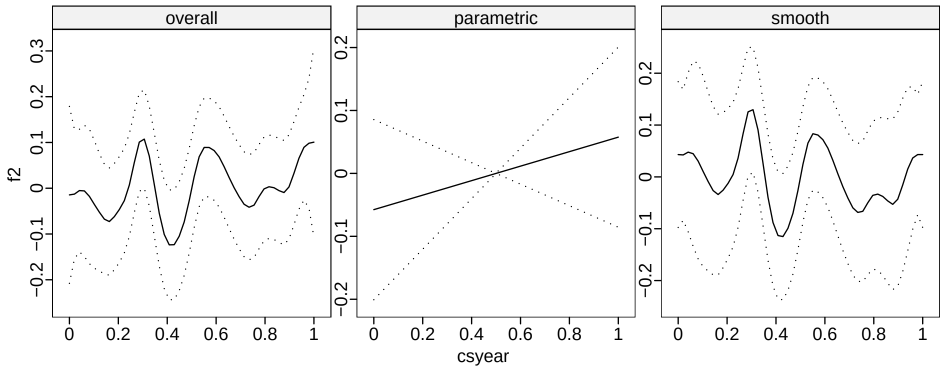


Figure 21: Estimated f_2 (overall), and its projections to $\mathcal{H}_0$ (parametric) and $\mathcal{H}_1$ (smooth). Dotted lines are 95% Bayesian confidence intervals.

Estimated functions and their projections are shown in Figures 20, 21 and 22. They are similar to previous estimates.

8.2 Global Climate Data

We downloaded this data set from the Carbon Dioxide Information Analysis Center at Oak Ridge National Laboratory (`http://cdiac.ESD.ORNL.GOV/ftp/ndp020`). As in Wahba and Luo (1996), we use the averages of winter (December, January and February) temperatures in 1981 from $n = 690$ stations (Wahba and Luo (1996) used $n = 725$ stations). The data also contains geological locations of these stations in terms of longitude (`long.degree`) and latitude (`lat.degree`). We use this data set to illustrate how to fit a spline on the sphere. We first made the following transformations: `long` $=$ `long.degree` $* \pi/180 + \pi$ and `lat` $=$ `lat.degree` $* \pi/180$. Then $0 \leq$ `long` $\leq 2\pi$ and $-\pi/2 \leq \text{lat} \leq \pi/2$.

```
> attach(climate)
```

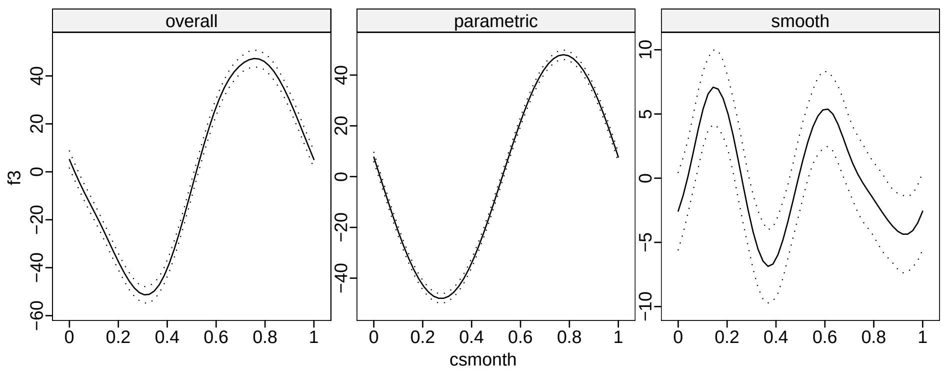


Figure 22: Estimated f_3 (overall), and its projections to $\mathcal{H}_0$ (parametric) and $\mathcal{H}_1$ (smooth). Dotted lines are 95% Bayesian confidence intervals.

```
> climate.fit <- ssr(temp~1, rk=sphere(cbind(long,lat)), data=climate)
> climate.fit
Smoothing spline regression fit by GCV method
Call: ssr(formula = temp ~ 1, rk = sphere(cbind(long, lat)), data = climate)

GCV estimate(s) of smoothing parameter(s) : 1.293621e-05
Equivalent Degrees of Freedom (DF):  210.993
Estimate of sigma:  2.270291

Number of Observations: 690

> long.grid <- seq(0,2*pi,len=60)
> lat.grid <- seq(-pi/2,pi/2,len=60)
> p.climate <- predict(climate.fit,expand.grid(long=long.grid,lat=lat.grid))
```

The contour plot of the predicted values is shown in Figure 23.

8.3 United States Historical Climate Data

We downloaded this data set from the Carbon Dioxide Information Analysis Center at Oak Ridge National Laboratory (`http://cdiac.ESD.ORNL.GOV/ftp/ndp019`). The data contains mean monthly temperature from 1890 to 1996 from 1221 stations in US (note that this data set has been updated on this site since we downloaded it five years ago), geological locations of these stations in terms of longitude (`long`) and latitude (`lat`). We use this data set to illustrate how to fit periodic spline, thin-plate spline, SS ANOVA and NNR models. We use data from 1961 to 1990 from 48 stations in Texas only. The data is saved as `TXtemp`.

We first fit a periodic spline, compute the estimates without the constant term (yearly mean) and its amplitude for each station.

```
> data(TXtemp)
> TXtemp$cm<- (TXtemp$month-0.5)/12
> amp <- NULL
```

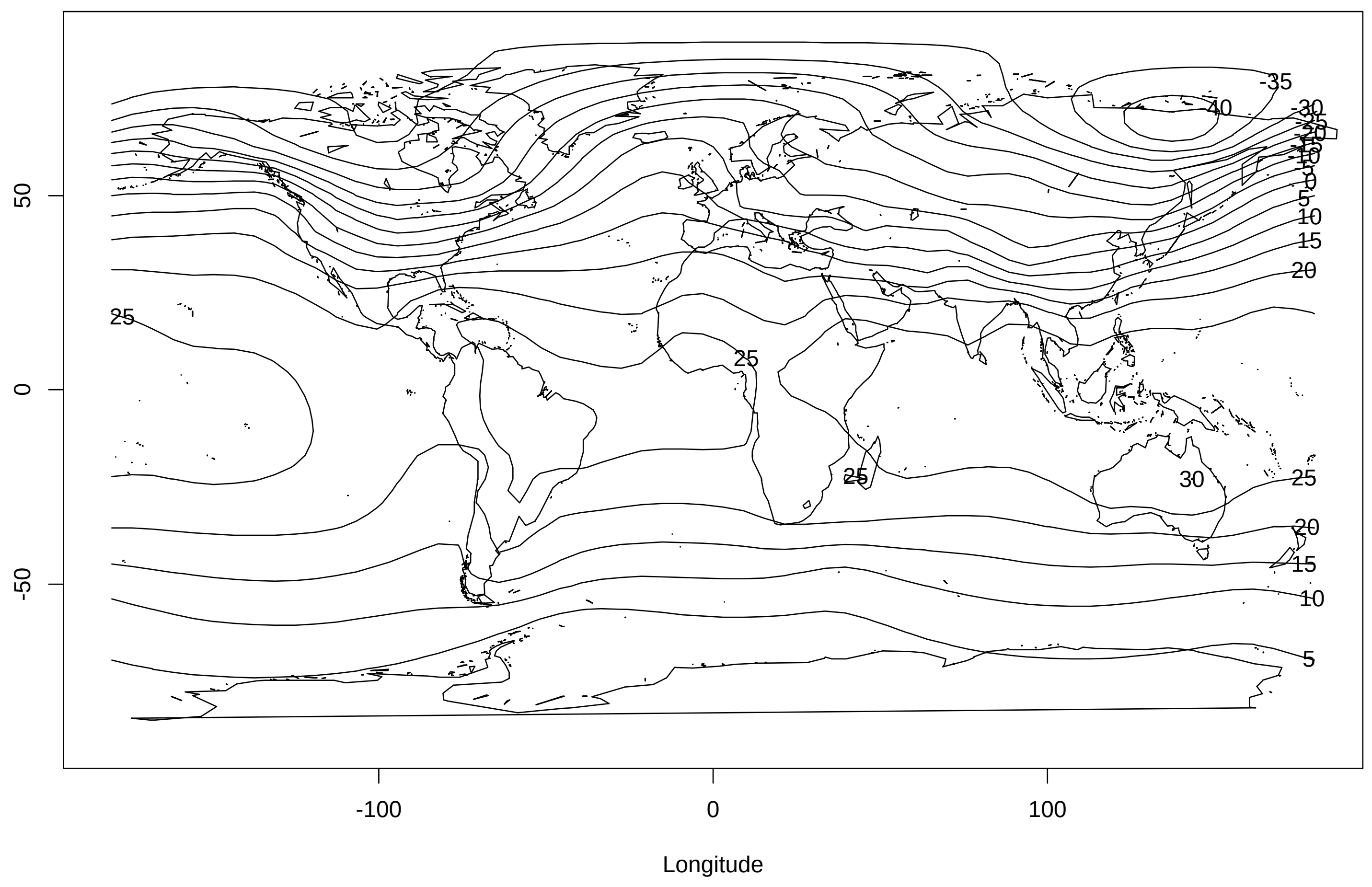


Figure 23: Contour plot of global average Winter temperature in 1981.

```
> for(i in 1:48){
  tmpfit<- ssr(mmtemp~1, rk=periodic(cm), spar="m",
               data=TXtemp[TXtemp$stacod==unique(TXtemp$stacod)[i]&
                           TXtemp$mmtemp!=-99.99,])
  p.tmpfit <- predict(tmpfit, terms=c(0,1), pstd=F,
                      newdata=data.frame(cm=seq(0,1,len=100)))$fit
  amp <- c(amp,max(abs(p.tmpfit)))
}
```

Figure 24 shows the estimated amplitudes on the logarithm scale plotted against longitude and latitude. It is clear that the middle and northern parts of Texas tend to have larger amplitudes (hotter Summer and colder Winter).

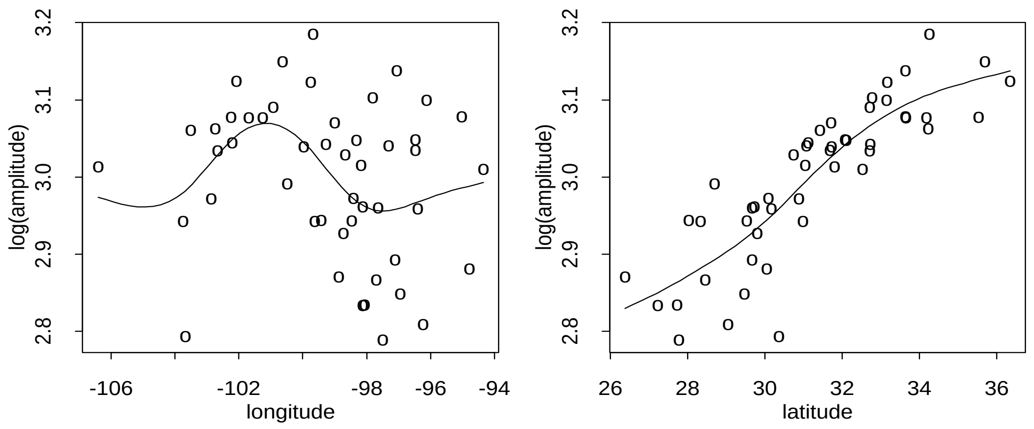


Figure 24: Plot of log(amplitude) vs longitude (left) and latitude (right). Circles are observations and solid lines are cubic spline fits.

We now fit a thin-plate spline with $d = 2$ and $m = 2$ to model the effect of `longitude` and `latitude` on the log(amplitude):

```
> loc <- TXtemp[TXtemp[,4]==1961&TXtemp[,6]==1,2:3]
> data <- data.frame(amp=log(amp),lat=loc[,1],long=loc[,2])
> tx.fit1 <- ssr(amp~long+lat,rk=tp.pseudo(list(long,lat)),data=data, spar="m")
> i <- interp(data$long,data$lat,data$amp)
> grid1 <- list(long=i$x,lat=i$y)
> p.tx.fit1 <- predict(tx.fit1,expand.grid(grid1),pstd=F)
```

The contour and 3-d plots of the fit are shown in Figure 25.

We can use SS ANOVA or NNR models to investigate seasonal trend (temporal) and location effect (spatial) together. For this purpose, we first compute average mean monthly temperature from 1996-1990 for each station.

```
> y <- gapply(TXtemp, which=5, FUN=function(x) mean(x[x!=-99.99]),
            group=TXtemp$stacod*TXtemp$month)
> tx.dat <- data.frame(y=as.vector(t(matrix(y,48,12,byrow=F))))
> tx.dat$month<-rep((1:12-0.5)/12, 48)
```

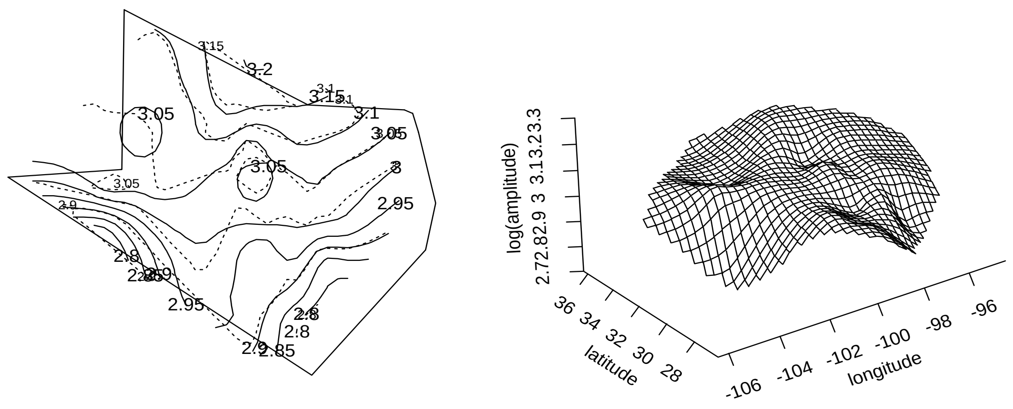


Figure 25: Left: contour plot of raw data (dotted lines) and TPS fit (solid lines). Right: 3-d plot of the TPS fit.

```
> tx.dat$lat<-rep(TXtemp$lat[seq(1, nrow(TXtemp),by=360)] ,rep(12,48))
> tx.dat$long<-rep(TXtemp$long[seq(1, nrow(TXtemp),by=360)] ,rep(12,48))
> tx.dat$stacod<-rep(TXtemp$stacod[seq(1, nrow(TXtemp),by=360)] ,rep(12,48))
```

Denote $t_1 =$ `month`, $x_1 =$ `longitude`, $x_2 =$ `latitude`, and $t_2 = (x_1, x_2)$. We model `month` (t_1) effect using a periodic spline, and spatial (t_2) effect using a TPS. Then we have the following SS ANOVA model:

$$\begin{aligned} y(t_1, t_2) &= \mu + \beta x_1 + \gamma x_2 + s_1(t_1) + s_2(t_2) + \\ &\quad sl^1_{12}(t_1, x_1) + sl^2_{12}(t_1, x_2) + ss_{12}(t_1, t_2) + \epsilon(t_1, t_2), \end{aligned} \tag{70}$$

where components on the right hand side are constant, linear main effect of x_1, linear main effect of x_2, smooth main effect of t_1, smooth main effect of t_2, smooth-linear interaction between t_1 and x_1, smooth-linear interaction between t_1 and x_2, and smooth-smooth interaction between t_1 and t_2.

```
> tx.fit3 <- ssr(y~long+lat, data=tx.dat, spar="m",
    rk=list(periodic(month), tp(list(long,lat)),
            rk.prod(periodic(month),kron(long)),
            rk.prod(periodic(month),kron(lat)),
            rk.prod(periodic(month),tp(list(long,lat)))))
> tx.fit3
...
GML estimate(s) of smoothing parameter(s) : 8.184394e-06 2.426733e-02
1.076039e-01 5.558457e-02 2.683768e-04
Equivalent Degrees of Freedom (DF):  345.5728
Estimate of sigma:  0.117142
```

If mean monthly temperature profiles from all stations have the same shape except a vertical shift and scale transformation, we may consider the following NNR model

$$y(t_1, t_2) = g_1(t_2) + \exp(g_2(t_2)) \times g_3(t_1) + \epsilon(t_1, t_2), \tag{71}$$

where $y(t_1, t_2)$ is the mean temperature in month t_1 of the station with longitude and latitude t_2. g_1, g_2 and g_3 are three unknown functions. g_3 represents seasonal trend. g_1 captures average climate differences between stations, and g_2 captures differences in the seasonal trend between stations. We will refer $\exp(g_2(t_2))$ as the amplitude. A bigger amplitude corresponds to a bigger seasonal variation. We model g_1 and g_2 using TPS's. Since g_3 is periodic and is close to a sinusoidal function, we use the L-spline with $L = D^2 + (2\pi)^2$. To make model (71) identifiable, we use the following side conditions: (a) $\int_0^1 g_2(t)dt = 0$, and (b) $\int_0^1 g_3(t)dt = 0$. Therefore, the model spaces for g_1, g_2 and g_3 are TPS, $TPS \ominus \{1\}$ and $W_2(per) \ominus \{1\}$ respectively.

```
> S3 <- periodic(tx.dat$month)
> f3.tmp <- ssr(y~1,rk=S3,data=tx.dat,spar="m")
> f3.ini <- as.vector(S3%*%f3.tmp$rkpk.obj$c)
> tx.fit4 <- nnr(y~f1(long,lat)+exp(f2(long,lat))*f3(month),
                 func=list(f1(x,z)~list(~x+z,tp(list(x,z))),
                           f2(x,z)~list(~x+z-1,tp(list(x,z))),
                           f3(x)~list(periodic(x))),
                 data=tx.dat,start=list(f1=mean(y),f2=0,f3=f3.ini))
> tx.fit4
Nonlinear Nonparametric Regression Model Fit by Gauss-Newton Method
 Model: y ~ f1(long, lat) + exp(f2(long, lat)) * f3(month)
 Data: tx.dat

 GCV estimate(s) of smoothing parameter(s): 2.576027e-06 1.736111e-03 4.471411e-07
 Equivalent Degrees of Freedom (DF):  95.47625

 Residual standard error: 0.9433764
 Number of Observations: 576
 Converged after 3 iterations
```

Contour plots of the estimates of g_1 and g_2 are shown in Figure 26. There is clear spatial effects to the mean temperature and amplitude. Southern part of the state is warmer and has less seasonal variation.

We now discuss two approaches two check the NNR model. We have fitted each station separately and saved the estimates without yearly mean in `p.tx.fit1`. One way to check if mean monthly temperature profiles from all stations have the same shape after removing yearly mean is to plot the estimates from one station against another to see if the points fall on a straight line. We rescale estimates from all stations such that all of them have amplitudes equal one. We then calculate Euclidean distances between stations. We select paired stations which have distances correspond to the 1%, 5%, 10%, 25%, 50%, 75%, 90%, 95% and 99% quantile of all possible paired stations. Figure 27 shows plots of these estimates for these selected stations. Some deviation from the straight line can be seen when distance becomes large.

```
> d <- dist(diag(1/amp)%*%t(p.tx.fit1))
> st <- NULL
```

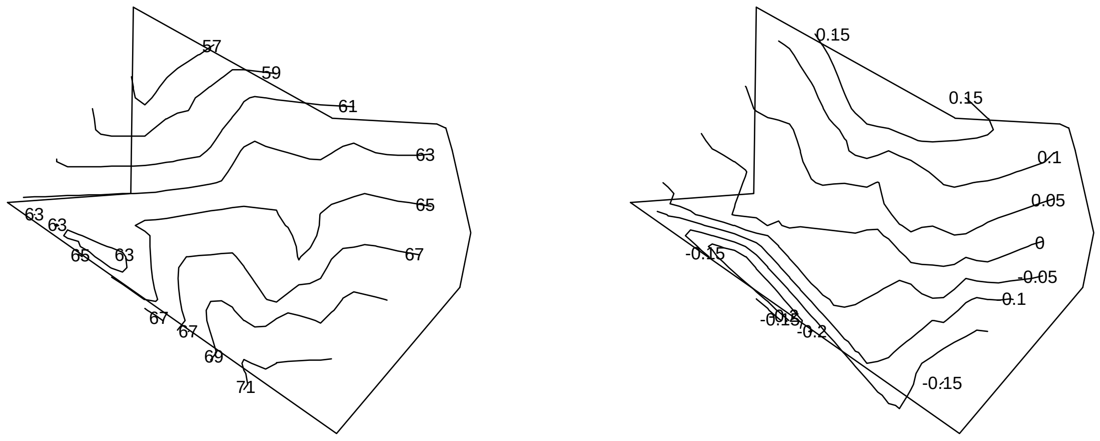


Figure 26: Contour plots of the estimates of g_1 (left) and g_2 (right).

```
> for (i in 1:47) { for (j in (i+1):48) st <- rbind(st,c(i,j))}
> ord <- order(d)
> tmp <- cbind(d[ord],st[ord,])
```

It is easy to see that model (70) reduces to the model (71) iff

$$f_{12} = [\exp(g_2)/\int \exp(g_2)dx_2 - 1]f_2. \tag{72}$$

Thus another way to check the NNR model (71), or equivalently condition (72), is to compute estimates of f_2 and f_{12} for all stations, and then plot f_2 against f_{12} to see if the points fall on a straight line.

```
> grid2 <- data.frame(month=rep(seq(0,1,len=40), 48),
     lat=rep(TXtemp$lat[seq(1,nrow(TXtemp),by=360)] ,rep(40,48)),
     long=rep(TXtemp$long[seq(1, nrow(TXtemp),by=360)] ,rep(40,48)))
> p.tx.fit3.f2 <- predict(tx.fit3,grid2[1:40,],
                          terms=c(0,0,0,1,0,0,0,0),pstd=F)$fit
> p.tx.fit3.f12 <- predict(tx.fit3,grid2,
                          terms=c(0,0,0,0,1,1,1,1),pstd=F)$fit
```

Such a plot is shown in Figure 28 which indicates certain deviation from the straight line.

For the purpose of illustration, we now fit the following additive model

$$y(t_1,t_2) = \mu + \beta x_1 + \gamma x_2 + s_1(t_1) + s_2(t_2) + \epsilon(t_1,t_2).$$

```
> tx.fit5 <- ssr(y~long+lat, rk=list(periodic(month), tp(list(long,lat))),
                 data=tx.dat, spar="m")
```

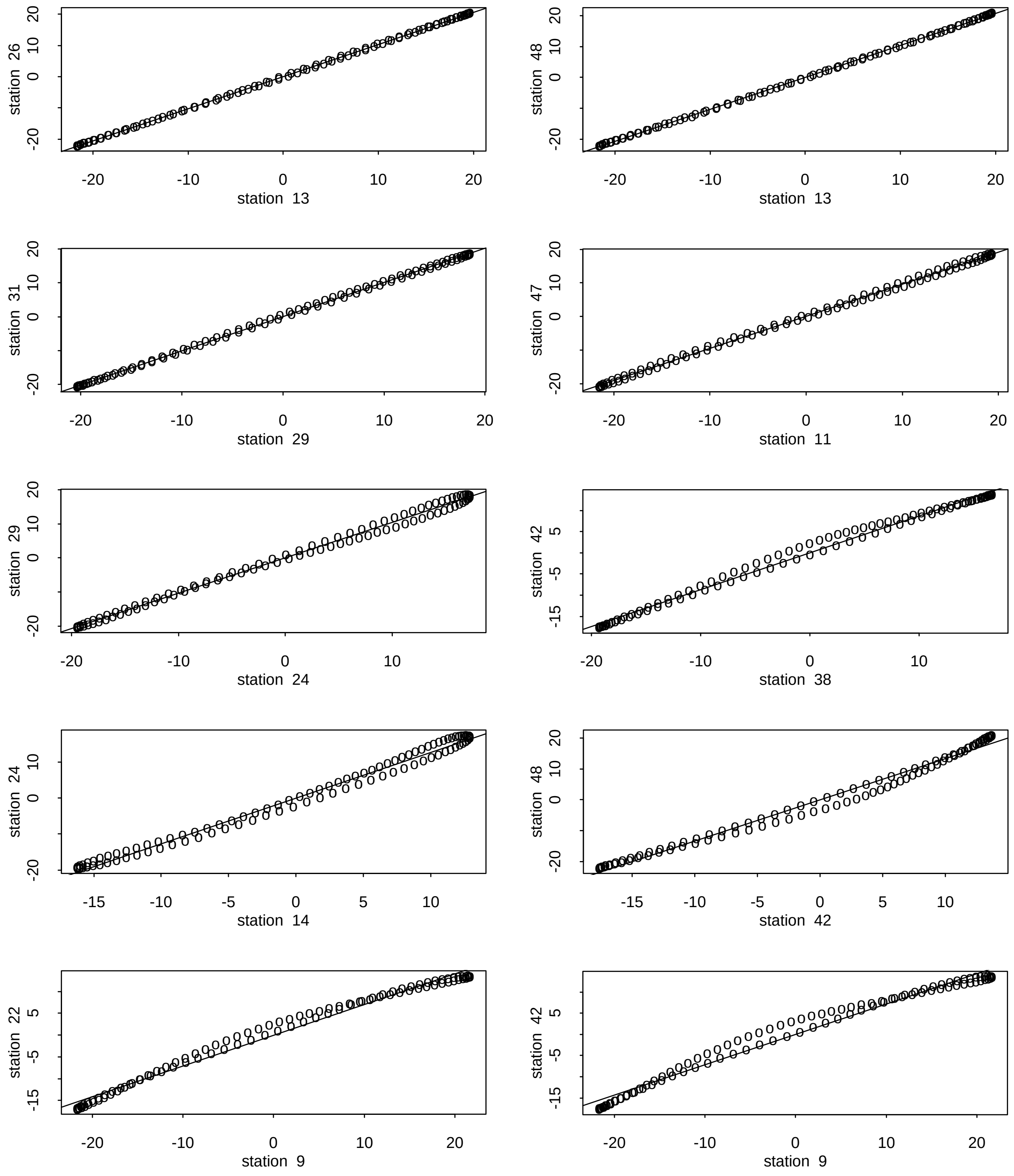


Figure 27: Plot of the centered and scaled estimates for the selected stations.

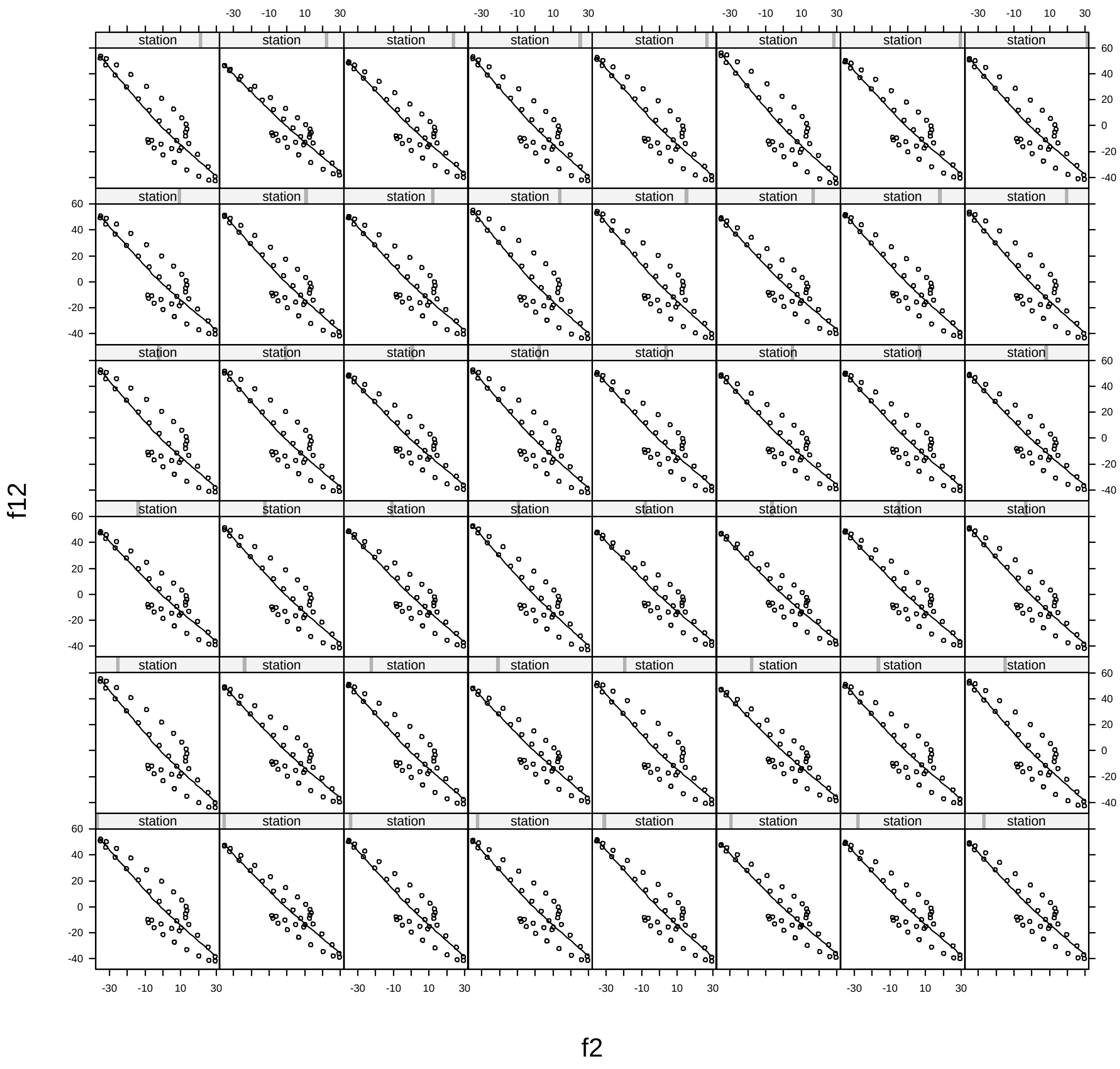


Figure 28: Plot of f_1 vs f_{12} for all stations.

```
...
GML estimate(s) of smoothing parameter(s) :  0.05823508 16.36685971
Equivalent Degrees of Freedom (DF):  41.59342
Estimate of sigma:  1.773092

Number of Observations: 576
```

It is obvious that above additive model is a special case of the NNR model (71) with $g_2 = 0$. We can compare three models using a model selection procedure such as GCV, AIC or BIC.

```
> n <- 576
> rss1 <- sum(tx.fit3$resi**2)
> rss2 <- sum(tx.fit4$resi**2)
> rss3 <- sum(tx.fit5$resi**2)
> gcv1 <- rss1/n/(1-tx.fit3$df/n)**2
> gcv2 <- rss2/n/(1-tx.fit4$df$f/n)**2
> gcv3 <- rss3/n/(1-tx.fit5$df/n)**2
> aic1 <- n*log(rss1/n)+2*tx.fit3$df
> aic2 <- n*log(rss2/n)+2*tx.fit4$df$f
> aic3 <- n*log(rss3/n)+2*tx.fit5$df
> bic1 <- n*log(rss1/n)+log(n)*tx.fit3$df
> bic2 <- n*log(rss2/n)+log(n)*tx.fit4$df$f
> bic3 <- n*log(rss3/n)+log(n)*tx.fit5$df
> print(c(rss1,rss2,rss3))
[1]    3.161981  428.052072 1680.096971
> print(c(gcv1,gcv2,gcv3))
[1] 0.0343016 1.0672903 3.3885449
> print(c(aic1,aic2,aic3))
[1] -2306.88183    19.73069   699.79434
> print(c(bic1,bic2,bic3))
[1] -801.5293  435.1371  880.9798
```

All model selection procedures choose the SS ANOVA model (70).

8.4 Chickenpox Epidemic

This data set, downloaded from `http://www-personal.buseco.monash.edu.au/~hyndman /TSDL/`, contains monthly number of reported cases of chickenpox in New York City from 1931 to the first six months of 1972. It has been analyzed by several authors to investigate dynamics in an epidemic (Yorke and London 1973, Schaffer and Kot 1985). Figure 29 shows time series plots of square root of monthly cases. We illustrate how to use the SS ANOVA and NNR models to investigate long term trend over years, seasonal trend and their interactions.

Denote y as the square root of reported cases in month t_1 of year t_2. Both t_1 and t_2 are transformed into the interval $[0, 1]$. We first consider an additive model

$$y(t_1, t_2) = \mu + \beta t_2 + s_1(t_1) + s_2(t_2) + \epsilon(t_1, t_2), \tag{73}$$

and a full SS ANOVA model

$$y(t_1, t_2) = \mu + \beta t_2 + s_1(t_1) + s_2(t_2) + sl^1_{12}(t_1, t_2) + ss_{12}(t_1, t_2), \tag{74}$$

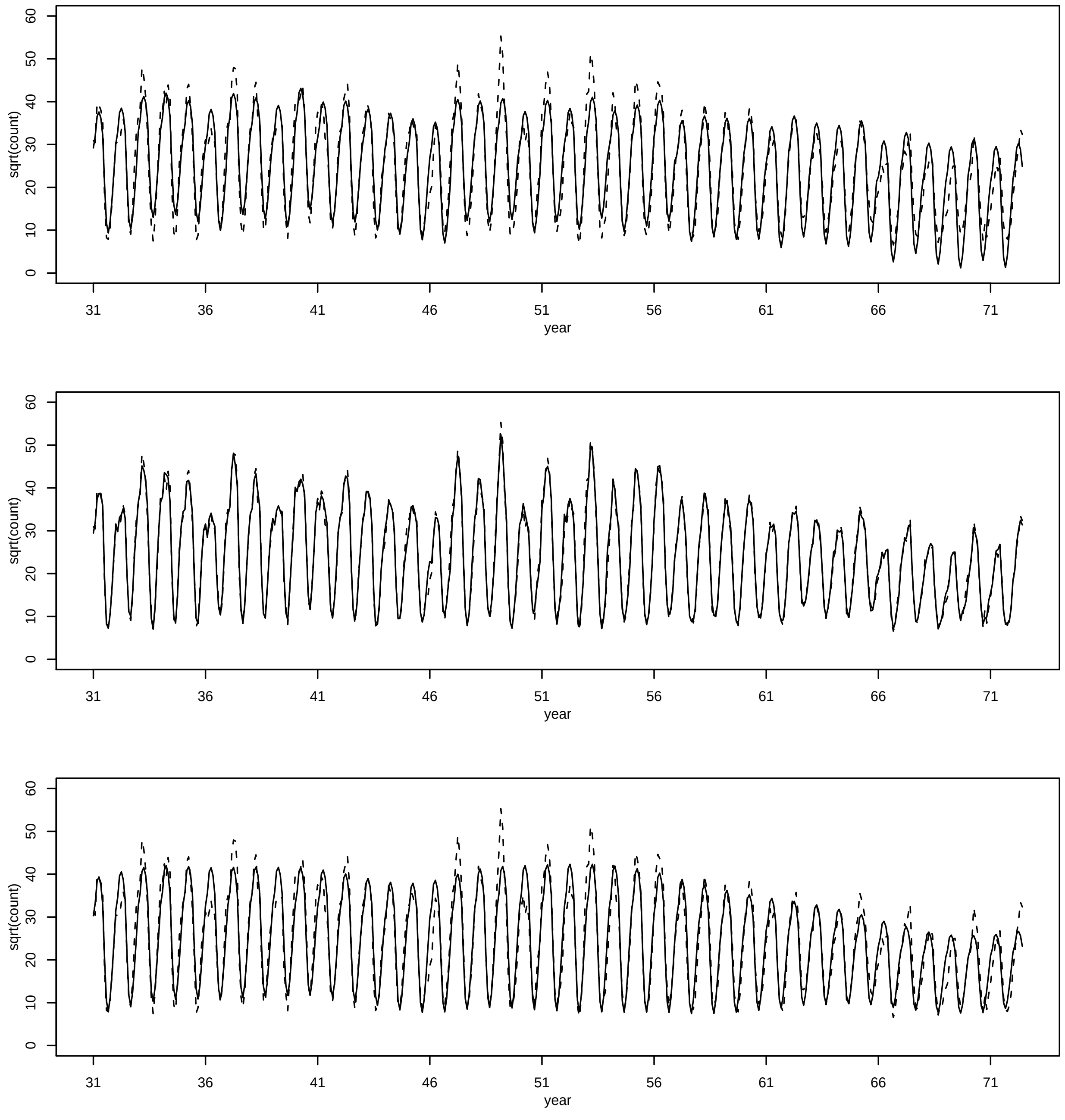


Figure 29: Plots of the square root of monthly cases (dotted lines) and fits (dotted lines) from models (73) (top), (74) (middle) and (75) (bottom).

where μ, βt_2, $s_1(t_1)$, $s_2(t_2)$, $sl^1_{12}(t_1, t_2)$ and $ss_{12}(t_1, t_2)$ are respectively constant, linear main effect of year, smooth main effect of month, smooth main effect of year, smooth-linear interaction between month and year, and smooth-smooth interaction between month and year. We model month (t_1) effect using a periodic spline and year (t_2) effect using a cubic spline.

```
> data(chickenpox)
> chickenpox$ct<- sqrt(chickenpox$count)
> chickenpox$csmonth<- (chickenpox$month-0.5)/12
> chickenpox$csyear<-  ident(chickenpox$year)
> chickenpox.fit1 <- ssr(ct~csyear, rk=list(periodic(csmonth),cubic(csyear)),
                        data=chickenpox)
> chickenpox.fit1
...
GCV estimate(s) of smoothing parameter(s) : 0.189468039 0.001004518
Equivalent Degrees of Freedom (DF):  46.32076
Estimate of sigma:  3.932079

Number of Observations: 498

> chickenpox.fit2 <- update(chickenpox.fit1,
          rk=list(periodic(csmonth),cubic(csyear),
                  rk.prod(periodic(csmonth),kron(csyear)),
                  rk.prod(periodic(csmonth),cubic(csyear))))

> chickenpox.fit2
...
GCV estimate(s) of smoothing parameter(s) : 3.175017e-02 1.703363e-04
                                            3.253160e-01 2.387995e-07
Equivalent Degrees of Freedom (DF):  278.5618
Estimate of sigma:  1.975239

Number of Observations: 498
```

The seasonal variation was mainly caused by two factors: (a) social behavior of children who made close contacts when school was in session; and (b) temperature and humidity which may affect the survival and transmission of dispersal stages (Yorke and London 1973, Schaffer and Kot 1985). Thus the seasonal variations were similar over the years. In the following NNR model we assume that the seasonal variation has the same shape after vertical shift and vertical scale transformations. Specifically we assume that

$$y(t_1, t_2) = g_1(t_2) + \exp(g_2(t_2)) \times g_3(t_1) + \epsilon(t_1, t_2), \tag{75}$$

where g_1, g_3 and g_2 are three unknown functions which represent respectively yearly mean cases, seasonal trend in a year and the magnitude of the seasonal variation for a particular year. We refer $\exp(g_2(t_2))$ as the amplitude. A bigger amplitude corresponds to a bigger seasonal variation. Thus in addition to being more parsimonious than the SS ANOVA model (74), the NNR model (75) has component functions with nice interpretations. We model g_1 and g_2 using cubic splines. It has been recognized that a simple sinusoidal model may be inappropriate (Earn, Rohani, Bolker and Gernfell 2000). Since g_3 is periodic and is close to a sinusoidal function, we use the L-spline with $L = D^2 + (2\pi)^2$. To make model (75) identifiable, we use the following side conditions: (a) $\int_0^1 g_2(t)dt = 0$, and (b) $\int_0^1 g_3(t)dt = 0$. Therefore, the model spaces for g_1, g_2 and g_3 are $W_2[0,1]$, $W_2[0,1] \ominus \{1\}$ and $W_2(per) \ominus \{1\}$ respectively.

```
> S3 <- periodic(chickenpox.data$month)
> f3.tmp <- ssr(y~1,rk=S3,data=chickenpox.data,spar=''m'')
> f3.ini <- as.vector(S3%*%f3.tmp$rkpk.obj$c)
> chickenpox.fit3 <- nnr(y~f1(year)+exp(f2(year))*f3(month),
           func=list(f1(x)~list(~I(x-.5),cubic(x)),
                     f2(x)~list(~I(x-.5)-1,cubic(x)),
                     f3(x)~list(~sin(2*pi*x)+cos(2*pi*x)-1,
                                lspline(x,type=''sine0''))),
           data=chickenpox.data,start=list(f1=mean(y),f2=0,f3=f3.ini),
           control=list(converg=''coef''),spar=''m'')
> chickenpox.fit3
...
 GML estimate(s) of smoothing parameter(s): 7.7261e-07 1.7787e-03 2.0090e-07
 Equivalent Degrees of Freedom (DF):  27.85445

 Residual standard error: 3.67597
 Number of Observations: 498
 Converged after 4 iterations
```

We can compare three models using a model selection procedure such as GCV, AIC or BIC.

```
> n <- 498
> rss1 <- sum(chickenpox.fit1$resi**2)
> rss2 <- sum(chickenpox.fit2$resi**2)
> rss3 <- sum(chickenpox.fit3$resi**2)
> gcv1 <- rss1/n/(1-chickenpox.fit1$df/n)**2
> gcv2 <- rss2/n/(1-chickenpox.fit2$df/n)**2
> gcv3 <- rss3/n/(1-chickenpox.fit3$df$f/n)**2
> aic1 <- n*log(rss1/n)+2*chickenpox.fit1$df
> aic2 <- n*log(rss2/n)+2*chickenpox.fit2$df
> aic3 <- n*log(rss3/n)+2*chickenpox.fit3$df$f
> bic1 <- n*log(rss1/n)+log(n)*chickenpox.fit1$df
> bic2 <- n*log(rss2/n)+log(n)*chickenpox.fit2$df
> bic3 <- n*log(rss3/n)+log(n)*chickenpox.fit3$df$f
> print(c(gcv1,gcv2,gcv3))
> [1] 17.04683  8.85435 14.31334
> print(c(aic1,aic2,aic3))
> [1] 1407.7146  826.9648 1323.6549
> print(c(bic1,bic2,bic3))
> [1] 1602.753 1999.877 1440.939
```

Thus the GCV and AIC criteria select the SS ANOVA model, and the BIC criteria selects the more parsimonious NNR model. The estimates of g_1 and g_2 and their 95% confidence intervals are shown in Figure 30. We also superimposed yearly averages on the plot of g_1 and the logarithm of scaled ranges on the plot of g_2. The scaled range of a specific year was calculated as the differences between the maximum and the minimum monthly number of cases divided by the range of the estimated seasonal trend g_3. It is clear that g_1 captures the long term trend in the mean and g_2 captures the long term trend in the range of seasonal variation. From Figure 29, the SS ANOVA model (74) captures local trend, particularly biennial pattern from 1945 to 1955, better than the NNR model (75). From the estimate of g_1, we can see that yearly averages peaked in the 1930s and 1950s, and gradually decreased in the 1960s after the introduction of mass vaccination. The amplitude reflects the seasonal variation in transmission rate. From the estimate of g_2 in Figure 30, we can see that magnitude of the seasonal variation peaked in the 1950s and then declined in the 1960s, possibly as a result of changing public health conditions including mass vaccination. Figure 31 shows the estimate of the seasonal trend g_3 and its projections onto the null space $\mathcal{H}_{30}$ (the simple sinusoidal model) and the orthogonal complement of the null space $\mathcal{H}_{31}$. Since the projection onto the complement space is significantly different from zero, we conclude that a simple sinusoidal model does not provide an accurate approximation.

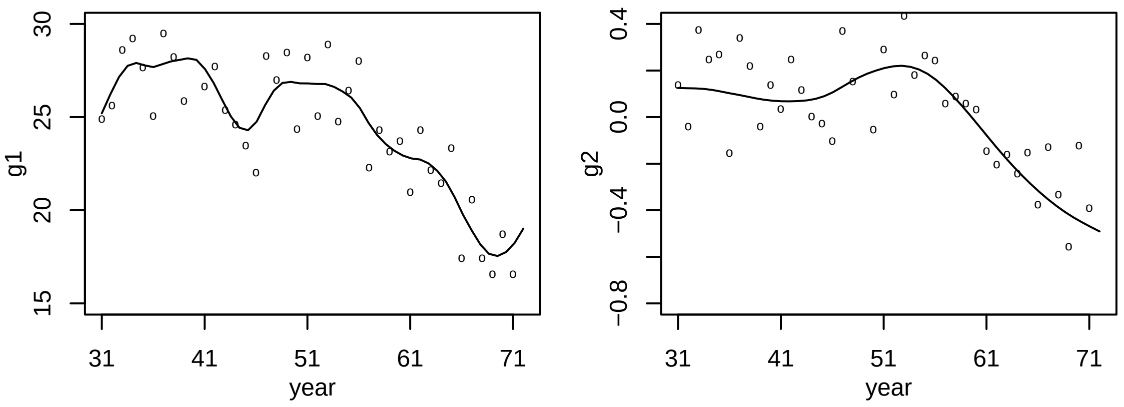


Figure 30: Left: plot of yearly averages (circles), estimate of g_1 (solid line) and its 95% confidence intervals (dotted lines). Right: plot of yearly scaled ranges on logarithm scale (circles), estimate of g_2 (solid line) and its 95% confidence intervals (dotted lines).

8.5 Lake Acidity Study

This data set was derived by Douglas and Delampady (1990) from the Eastern Lakes Survey of 1984. It contains measurements of 1789 lakes in three Eastern US regions: Northeast, Upper Midwest and Southeast. Of interest is the dependence of the water `pH` level (*ph*) on the `calcium` concentration in $\log_{10}$ milligrams per liter (t_1) and the geographical `location` ($t_2 = (x_1, x_2)$ where x_1=latitude and x_2=longitude). Gu and Wahba (1993a) analyzed this data set using SS ANOVA models. As in Gu and Wahba (1993a), we use a subset of 112 lakes in the southern Blue Ridge mountains area.

We use this data set to illustrate how to fit a cubic spline, a cubic spline with correlated random errors, a SLM and an SS ANOVA model.

First, we fit a cubic spline to *ph* using one variable `calcium` (t_1)

$$\texttt{pH}(t_1) = f(t_1) + \epsilon(t_1)$$

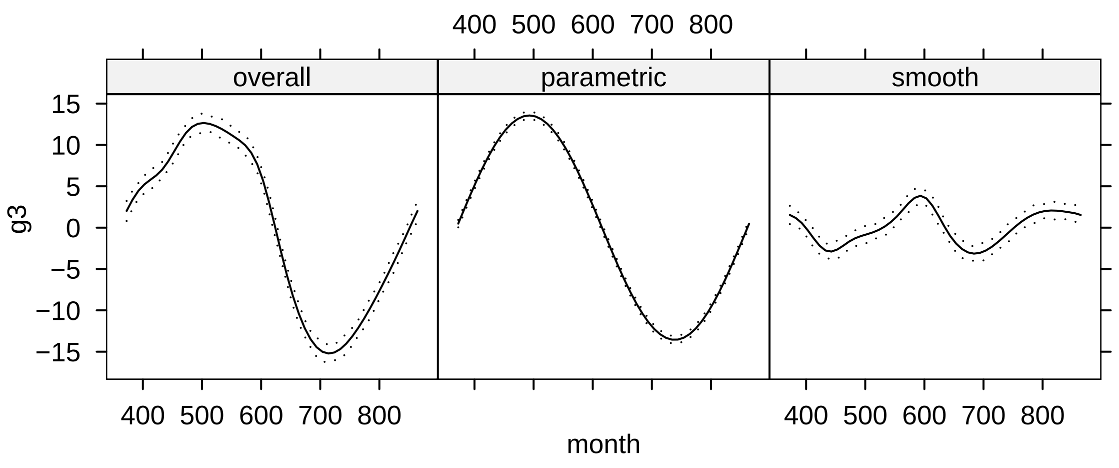


Figure 31: Estimated g_3 (overall), and its projections to $\mathcal{H}_0$ (parametric) and $\mathcal{H}_1$ (smooth). Dotted lines are 95% Bayesian confidence intervals.

where $f \in W_2([0,1])$.

```
> acid.fit1 <- ssr(ph~t1, rk=cubic(t1), data=acid, scale=T)
> summary(acid.fit1)
...
GCV estimate(s) of smoothing parameter(s) : 3.84e-06
Equivalent Degrees of Freedom (DF):  8.21
Estimate of sigma:  0.281
> anova(acid.fit1)

Testing H_0: f in the NULL space

     test.value simu.size simu.p-value
LMP 0.003634651       100         0.04
GCV 0.008239078       100         0.02
```

Both p-values from the LMP and GCV tests are small, indicating that a simple linear model is not sufficient to describe the relationship. This can be confirmed by looking at the fitted function and its projections onto $\mathcal{H}_0$ and $\mathcal{H}_1$. A linear model is equivalent to the projection onto $\mathcal{H}_1$ being zero. The following statements compute posterior means and standard deviations for the projections onto $\mathcal{H}_0$ with `terms=c(1,1,0)`, the projections onto $\mathcal{H}_1$ with `terms=c(0,0,1)`, and the overall fit with `terms=c(1,1,1)`.

```
> grid <- data.frame(t1=seq(min(acid$t1), max(acid$t1), len=100))
> tm <- matrix(c(1,1,0,0,0,1,1,1,1), ncol=3, byrow=T)
> p.acid.fit1 <- predict(acid.fit1, grid, terms=tm)
```

Figure 32 shows the fitted function and its projections. The nonlinear part is small, but different from zero.

`pH` observations close in geographic locations are often correlated. Suppose that we want to use spherical spatial correlation structure with nugget effect for the variable `location` $t_2 = (x_1, x_2)$. That

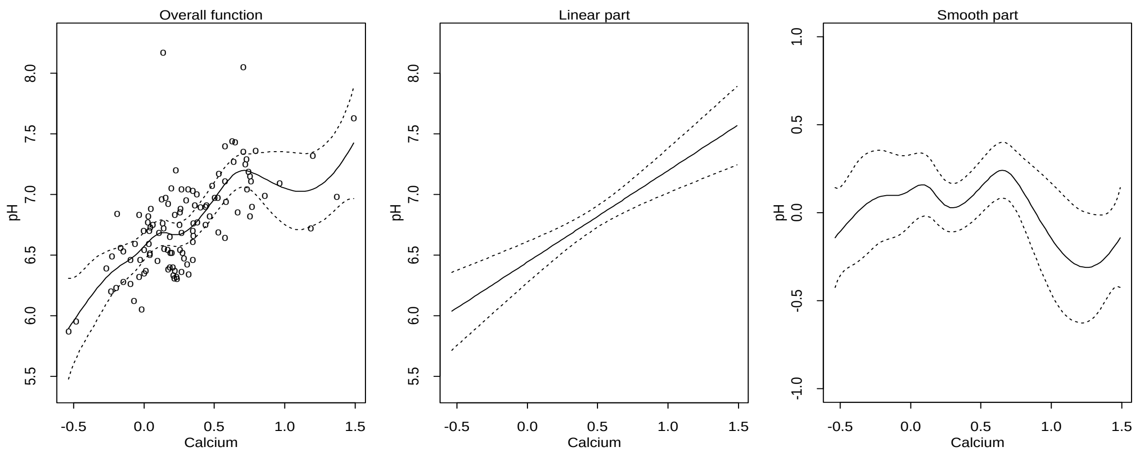


Figure 32: Left: circles are observations, solid line is the cubic spline fit. Middle: solid line is the projection of the overall function to the null space $\mathcal{H}_0$. Right: solid line is the projection of the overall function to the space $\mathcal{H}_1$. Dotted lines are 95% Bayesian confidence intervals.

is, regard random errors ϵ as a function of t_2, and assume

$$\text{Cov}(\epsilon(s_2), \epsilon(t_2)) = \begin{cases} \sigma^2(1-c)[1 - 1.5d(s_2,t_2)/r - .5d^2(s_2,t_2)/r^2], & 0 < d(s_2,t_2) \le r, \\ \sigma^2, & d(s_2,t_2) = 0, \end{cases}$$

where c is the nugget effect, $d(s_2, t_2)$ is the Euclidean distance between s_2 and t_2, and r is the range parameter.

```
> acid$s1 <- (acid$t1-min(acid$t1))/diff(range(acid$t1))
> acid.fit2 <- ssr(ph~s1, rk=cubic(s1), data=acid,
                   corr=corSpher(form=~x1+x2,nugget=T), spar="m")
> acid.fit2
...
Coefficients (d):
(Intercept)          s1
   6.221757    1.264431

GML estimate(s) of smoothing parameter(s) : 4.645508
Equivalent Degrees of Freedom (DF):  2.000284
Estimate of sigma:  0.2994075
Correlation structure of class corSpher representing
     range     nugget
0.03646616 0.68530949
```

We plot in Figure 33 fitted curves from `acid.fit1`, `acid.fit2` and `acid.fit5`. Notice the effect of the correlation on the smoothing parameter and the fit. Equivalent degrees of freedom for f decreased from 8.21 to 2.00. The new fit is linear. The smaller smoothing parameter in the first fit might be caused by the spatial correlation.

We can also model the effect of `location` directly as a covariate. We consider two approaches to

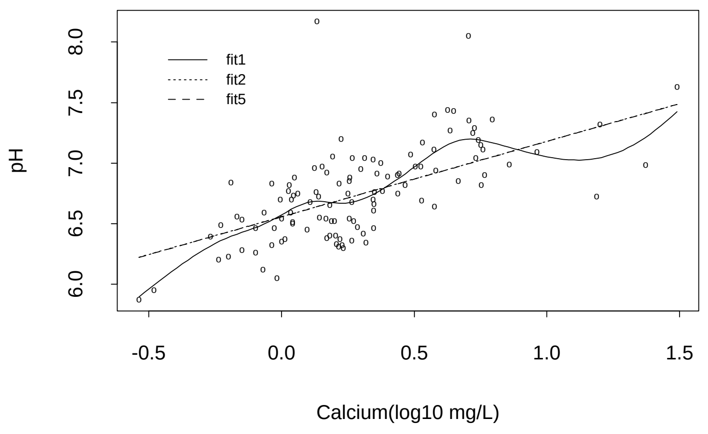


Figure 33: Points are observations. Lines are fitted curves from three models.

model `location` effect: use random effects with exponential spatial correlation structure (similar to Kriging), and use thin plate splines.

For the first approach, we consider

$$\texttt{pH}(t_1,t_2) = f(t_1) + \beta_1 x_1 + \beta_2 x_2 + u(t_2) + \epsilon(t_1,t_2), \tag{76}$$

where $f \in W_2([0,1])$, $u(t_2)$ is a spatial process and $\epsilon(t_1,t_2)$ are random errors independent of the spatial process. Model (76) separates the contribution of the spatial correlation to random errors in `acid.fit2` from other sources, and regard the spatial process as part of the signal. To show different options, we now assume an exponential correlation structure. Specifically, we assume $u(t_2)$ in (76) is a mean zero process with

$$\text{Cov}(u(s_2),u(t_2)) = \begin{cases} \sigma^2(1-c)\exp(-d(s_2,t_2)/r), & d(s_2,t_2) > 0 \\ \sigma^2, & d(s_2,t_2) = 0, \end{cases}$$

where c, $d(s_2,t_2)$ and r are the same as those defined in the spherical correlation structure. Note that we include the linear effects of `location` in the fixed effects. This will allow us to compare with the fit of a thin plate spline model later.

Denote $\boldsymbol{t}$ as the vector of design points for t_2. Let $u(\boldsymbol{t})$ be the vector of the u process evaluated at design points. $u(\boldsymbol{t})$ are random effects and $u(\boldsymbol{t}) \sim \text{N}(0, \sigma^2 D)$, where D depends parameters c and r. Again, the SLM model (76) cannot be fitted directly using `slm` since D depends on the range parameter r nonlinearly. We fit model (76) in two steps. We first regard $u(\boldsymbol{t})$ as part of random error, estimate the range parameter, and calculate the estimated covariance matrix without nugget effect:

```
> temp <- ssr(ph~t1+x1+x2, rk=tp(t1), data=acid,
              corr=corExp(form=~x1+x2, nugget=T), spar="m")
> tau <- coef(temp$cor.est, F)
> D <- corMatrix(initialize(corExp(tau[1],form=~x1+x2), data=acid))
```

Consider the estimated D as the true covariance matrix. Then we can calculate the Cholesky decomposition of D as $D = ZZ^T$, and transform the random effects $u(\boldsymbol{t}) = Z\boldsymbol{b}$, where $\boldsymbol{b} \sim \mathrm{N}(0, \sigma_1^2 I)$. Now we are ready to fit the transformed SLM:

```
> Z <- chol.new(D)
> acid.fit3 <- slm(ph~t1+x1+x2, rk=tp(t1), data=acid,
                  random=list(pdIdent(~Z-1)))
> summary(acid.fit3)
...
Coefficients (d):
(Intercept)          t1          x1          x2
  6.5483884   0.6795706  -8.3694493   2.1135023

GML estimate(s) of smoothing parameter(s) : 0.0005272899
Equivalent Degrees of Freedom (DF):  4.000179
Estimate of sigma:  0.002725062
```

We then calculate the estimated effect of `calcium`:

```
grid1 <- data.frame(t1=seq(min(acid$t1), max(acid$t1), len=100),
                    x1=min(acid$x1), x2=min(acid$x2))
p.acid.fit3.t1 <- intervals(acid.fit3, grid1, terms=c(0,1,0,0,1))
```

Suppose that we want to calculate the `location` effect on grid points $\boldsymbol{s}$. Let $u(\boldsymbol{s})$ be the vector of the u process evaluated at elements in $\boldsymbol{s}$. Let $R = \mathrm{Cov}(u(\boldsymbol{s}), u(\boldsymbol{t}))$. Then $\hat{u}(\boldsymbol{s}) = RD^{-1}\hat{u}(\boldsymbol{t}) = RZ^{-T}\hat{\boldsymbol{b}}$.

```
grid2 <- expand.grid(
               x1=seq(min(acid$x1)-.001,max(acid$x1)+.001, len=20),
               x2=seq(min(acid$x2)-.001,max(acid$x2)+.001, len=20))
newdata <- data.frame(y1=c(acid$x1,grid2$x1), y2=c(acid$x2,grid2$x2))
RD <- corMatrix(initialize(corExp(tau[1], form=~y1+y2), data=newdata))
R <- RD[(length(acid$x1)+1):length(newdata$y1),1:length(acid$x1)]
u.new <- R%*%t(solve(Z))%*%as.vector(acid.fit3$lme.obj$coef$random[[2]])
p.acid.fit3.t2 <- acid.fit3$lme.obj$coef$fixed[3]*grid2$x1+
                  acid.fit3$lme.obj$coef$fixed[4]*grid2$x2+u.new
```

Figure 34 plots the estimated main effects of t_1 and $t_2 = (x_1, x_2)$ on the left panel.

As the second approach, we consider the same thin plate spline model as in Gu and Wahba (1993a). Specifically, we use a TPS with $d = 1$ and $m = 2$ to model the effect of `calcium`, and a TPS with $d = 2$ and $m = 2$ to model the effect of `location`. Then we have the following SS ANOVA model

$$f(t_1, t_2) = \mu + \alpha t_1 + \beta x_1 + \gamma x_2 + s_1(t_1) + s_2(t_2) + ls_{12}(t_1, t_2) + sl_{12}^1(t_1, x_1) + sl_{12}^2(t_1, x_2) + ss_{12}(t_1, t_2).$$

Components on the right hand side are constant, linear main effect of t_1, linear main effect of x_1, linear main effect of x_2, smooth main effect of t_1, smooth main effect of t_2, linear-smooth interaction between t_1 and t_2, smooth-linear interaction between t_1 and x_1, smooth-linear interaction between t_1 and x_2, and smooth-smooth interaction between t_1 and t_2.

```
> acid.fit4 <- ssr(ph~t1+x1+x2, rk=list(tp(t1), tp(list(x1,x2)),
               rk.prod(kron(t1),tp(list(x1,x2))), rk.prod(kron(x1),tp(t1)),
               rk.prod(kron(x2),tp(t1)), rk.prod(tp(t1),tp(list(x1,x2)))),
               data=acid, spar="m")
> summary(acid.fit4)
```

```
...
Coefficients (d):
 (Intercept)        t1        x1       x2
    6.555506 0.6235892 -8.783944 1.974596

GML estimate(s) of smoothing parameter(s) : 1.816737e+04
3.600865e-03 2.710276e+01 2.759507e+00 1.992876e+01 1.440841e-01
Equivalent Degrees of Freedom (DF):  10.7211
Estimate of sigma:  0.2560693
```

Since the smoothing parameters corresponding to the interaction terms are large, it is easy to check that these interaction terms are small. Thus we reduce to the following additive model with main effects only

$$f(t_1, t_2) = \mu + \alpha t_1 + \beta x_1 + \gamma x_2 + s_1(t_1) + s_2(t_2).$$

```
> acid.fit5 <- update(acid.fit3,  rk=list(tp(t1), tp(list(x1,x2))))
> summary(acid.fit5)
... ...
Coefficients (d):
 (Intercept)        t1        x1       x2
    6.555502 0.6235885 -8.783569 1.974339

GML estimate(s) of smoothing parameter(s) : 6.045750e+03 3.600423e-03
Equivalent Degrees of Freedom (DF):  10.72108
Estimate of sigma:  0.2560684
> grid3 <- data.frame(t1=seq(min(acid$t1), max(acid$t1), len=100),
                      x1=min(acid$x1), x2=min(acid$x2))
> p.acid.fit5.t1 <- predict(acid.fit5, grid3, terms=c(0,1,0,0,1,0))
> grid4 <- expand.grid(t1=min(acid$t1),
                       x1=seq(min(acid$x1), max(acid$x1), len=20),
                       x2=seq(min(acid$x2), max(acid$x2), len=20))
> p.acid.fit5.t2 <- predict(acid.fit5, grid4, terms=c(0,0,1,1,0,1))
```

Figure 34 plots the estimated main effects of t_1 and $t_2 = (x_1, x_2)$ on the right panel. It is seen that the estimates of the `calcium` main effects are almost identical. The estimates of the `location` main effects have a similar pattern. The estimate from `acid.fit5` is smoother.

```
> grid3 <- data.frame(t1=seq(min(acid$t1), max(acid$t1), len=100),
                      x1=min(acid$x1), x2=min(acid$x2))
> p.acid.fit5.t1 <- predict(acid.fit5, grid3, terms=c(0,1,0,0,1,0))
> grid4 <- expand.grid(t1=min(acid$t1),
                       x1=seq(min(acid$x1), max(acid$x1), len=20),
                       x2=seq(min(acid$x2), max(acid$x2), len=20))
> p.acid.fit5.t2 <- predict(acid.fit5, grid4, terms=c(0,0,1,1,0,1))
```

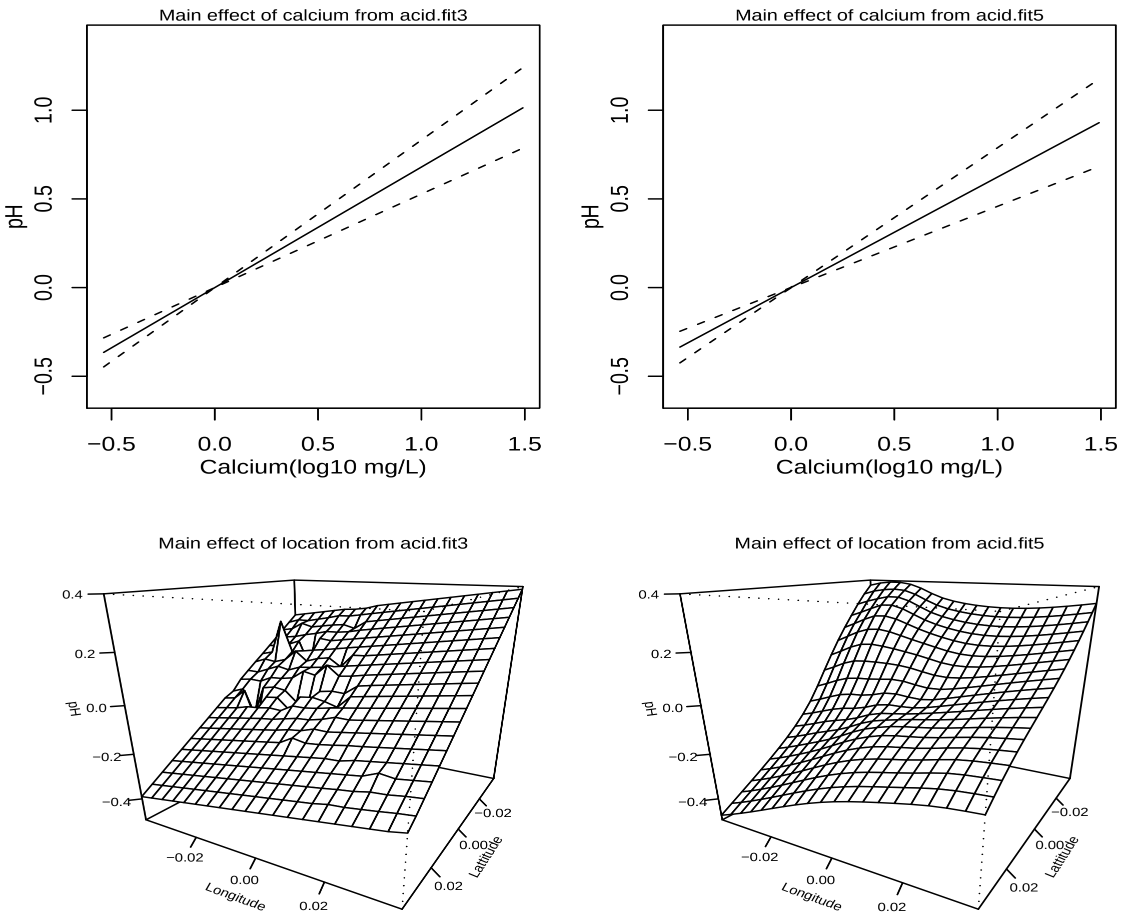


Figure 34: Plots of estimated main effects. Left: estimates from acid.fit3. Right: estimates from acid.fit5. First row: solid lines are the estimated main effects of `calcium`, and dotted lines are 95% confidence intervals. Second row: estimated main effects of `location`.

8.6 Wisconsin Epidemiological Study of Diabetic Retinopathy

Wisconsin Epidemiological Study of Diabetic Retinopathy (WESDR) is an epidemiological study of a cohort of diabetic patients receiving their medical care in an 11-county area in Southern Wisconsin. Detailed descriptions of the data can be found in Klein, Klein, Moss, Davis and DeMets (1988). A number of medical, demographic, ocular and other covariates were recorded at the baseline and later examinations along with a retinopathy score for each eye. As in Wahba et al. (1995), we investigate how progression of diabetic retinopathy at the first follow-up depends on the following covariates: `dur` (duration of diabetes at baseline), `gly` (glycosylated hemoglobin, a measure of hyperglycemia), and `bmi` (body mass index = weight in kg/(height in m)^2). As in Wahba et al (1994), we chose a subgroup of the younger onset consisting of 669 subjects with no or non-proliferative retinopathy at the baseline. See Wahba et al. (1995) for details of this data set.

We use this data set to illustrate how to fit smoothing spline models for non-Gaussian data. Firstly, we fit a simple cubic spline

$$\text{logit}P(\texttt{prg}=1 \mid \texttt{bmi}) = f(\texttt{bmi}),$$

where $f \in W_2([0,1])$.

```
> wesdr.fit1<- ssr(prg~bmi, rk=cubic(bmi), data=wesdr, family="binary",
                  scale=T, spar="u", varht=1)
> summary(wesdr.fit1)
...

Coefficients (d):
(Intercept)         bmi
  -1.286666    1.793480

UBR estimate(s) of smoothing parameter(s) : 8.793603e-06
Equivalent Degrees of Freedom (DF):  6.036309
Estimate of sigma:  1

> grid <- data.frame(bmi=seq(min(wesdr$bmi),max(wesdr$bmi),len=100))
> p.wesdr.fit1 <- predict(wesdr.fit1, grid)
```

Figure 35 shows the fitted probability function and its 95% Bayesian confidence intervals based on `wesdr.fit1`.

Wahba et al (1994) reached the following model

$$\begin{aligned}\text{logit}P(\texttt{prg}=1 \mid \texttt{dur},\texttt{gly},\texttt{bmi}) &= \mu + \alpha * \texttt{gly} + \beta * \texttt{dur} + \gamma * \texttt{bmi} + s_1(\texttt{dur}) + s_2(\texttt{bmi}) \\ &\quad + ls_{12}(\texttt{dur},\texttt{bmi}) + sl_{12}(\texttt{dur},\texttt{bmi}) + ss_{12}(\texttt{dur},\texttt{bmi}). \qquad (77)\end{aligned}$$

The `ssr` function and related utility functions can be used for model-building. We omit the details and fit (77) directly.

```
wesdr.fit2<- update(wesdr.fit1, prg~dur+gly+bmi+I(dur*bmi),
    rk=list(cubic(dur), cubic(bmi), rk.prod(kron(dur), cubic(bmi)),
    rk.prod(kron(bmi), cubic(dur)), rk.prod(cubic(dur), cubic(bmi))))
> summary(wesdr.fit2)
...
```

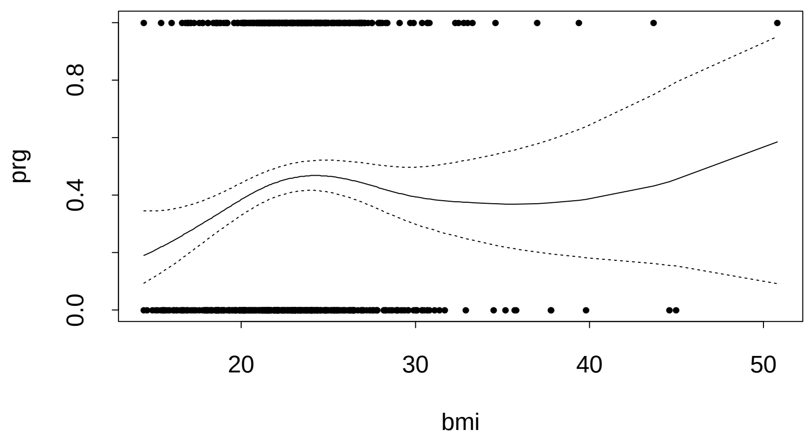


Figure 35: Points are observations. Solid line is fitted probability function. Dotted lines are 95% Bayesian confidence intervals.

```
Coefficients (d):
 (Intercept)          dur          gly          bmi I(dur * bmi)
  -6.1810142   -2.5728071    0.3864040    0.1401764   11.2939414

UBR estimate(s) of smoothing parameter(s) : 2.653578e+00 3.982479e+00 2.941409e+05 3.086313e+05
Equivalent Degrees of Freedom (DF):  11.19937
Estimate of sigma:  1
> grid <- expand.grid(dur=seq(min(wesdr$dur),max(wesdr$dur),len=40),
              bmi=seq(min(wesdr$bmi),max(wesdr$bmi),len=40),
              gly=median(wesdr$gly))
> p.wesdr.fit2 <- predict(wesdr.fit2, grid)
```

Figure 36 reproduce Figure 6.1 in Wahba et al (1994).

8.7 Potassium Measurements on Dogs

36 dogs were assigned to four groups: control, extrinsic cardiac denervation three weeks prior to coronary occlusion, extrinsic cardiac denervation immediately prior to coronary occlusion, and bilateral thoratic sympathectomy and stellectomy three weeks prior to coronary occlusion. Coronary sinus potassium concentrations were measured on each dog every two minutes from 1 to 13 minutes after occlusion (Grizzle and Allen 1969). Observations are shown in Figure 37.

We are interested in (i) estimating the group (treatment) effects; (ii) estimating the population mean concentration as functions of time; and (iii) predicting response over time for each dog. There are two categorical covariates `group` and `dog` and a continuous covariate `time`. We code the `group` factor as 1 to 4, and the observed `dog` factor as 1 to 36. We transform the `time` variable into [0,1]. We treat `group` and `time` are fixed factors. From the design, the `dog` factor is nested within the `group`

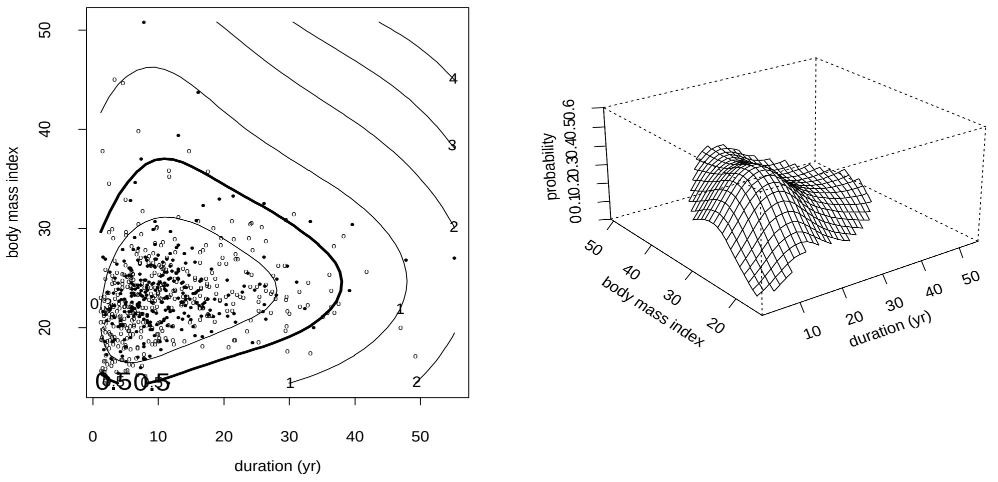


Figure 36: Left: data and contours of constant posterior standard deviation. Right: estimated probability of progression as a function of `dur`and `bmi` for `gly` fixed at its median.

factor. Therefore we treat `dog` as a random factor. For `group` k, denote $\mathcal{B}_k$ as the population from which the dogs in `group` k were drawn and $\mathcal{P}_k$ as the sampling distribution. Assume the following model

$$y_{kwj} = f(k, w, t_j) + \epsilon_{kwj}; \quad k = 1, \cdots, 4; \;\; w \in \mathcal{B}_k; \;\; t_j \in [0, 1],$$

where y_{kwj} is the observed potassium concentration at `time` t_j of `dog` w in the population $\mathcal{B}_k$, $f(k, w, t_j)$ is the "true" concentration at `time` t_j of `dog` w in the population $\mathcal{B}_k$, and ϵ_{kwj}'s are random errors. $f(k, w, t_j)$ is a function defined on $\{\{1\} \otimes \mathcal{B}_1, \{2\} \otimes \mathcal{B}_2, \{3\} \otimes \mathcal{B}_3, \{4\} \otimes \mathcal{B}_4\} \otimes [0, 1]$. Note that $f(k, w, j)$ is a random variable since w is a random sample from $\mathcal{B}_k$. What we observe are realizations of this "true" mean function plus random errors. We use label i to denote dogs we actually observe.

Suppose that we want to model the `time` factor using a cubic spline, and shrink both the `group` and `dog` factors toward constants. We define the following four projection operators:

$$\begin{aligned} P_2 f &= \int_{\mathcal{B}_k} f(k, w, t) d\mathcal{P}_k(w), \\ P_1 f &= \sum_{k=1}^{4} P_2 f(k, t)/4, \\ P_3 f &= \int_0^1 f(k, w, t) dt, \\ P_4 f &= [\int_0^1 (\partial f(k, w, t)/\partial t) dt](t - 0.5). \end{aligned}$$

Then we have the following SS ANOVA decomposition

$$\begin{aligned} & f \\ = \;& [P_1 + (P_2 - P_1) + (I - P_2)][P_3 + P_4 + (I - P_3 - P_4)]f \end{aligned}$$

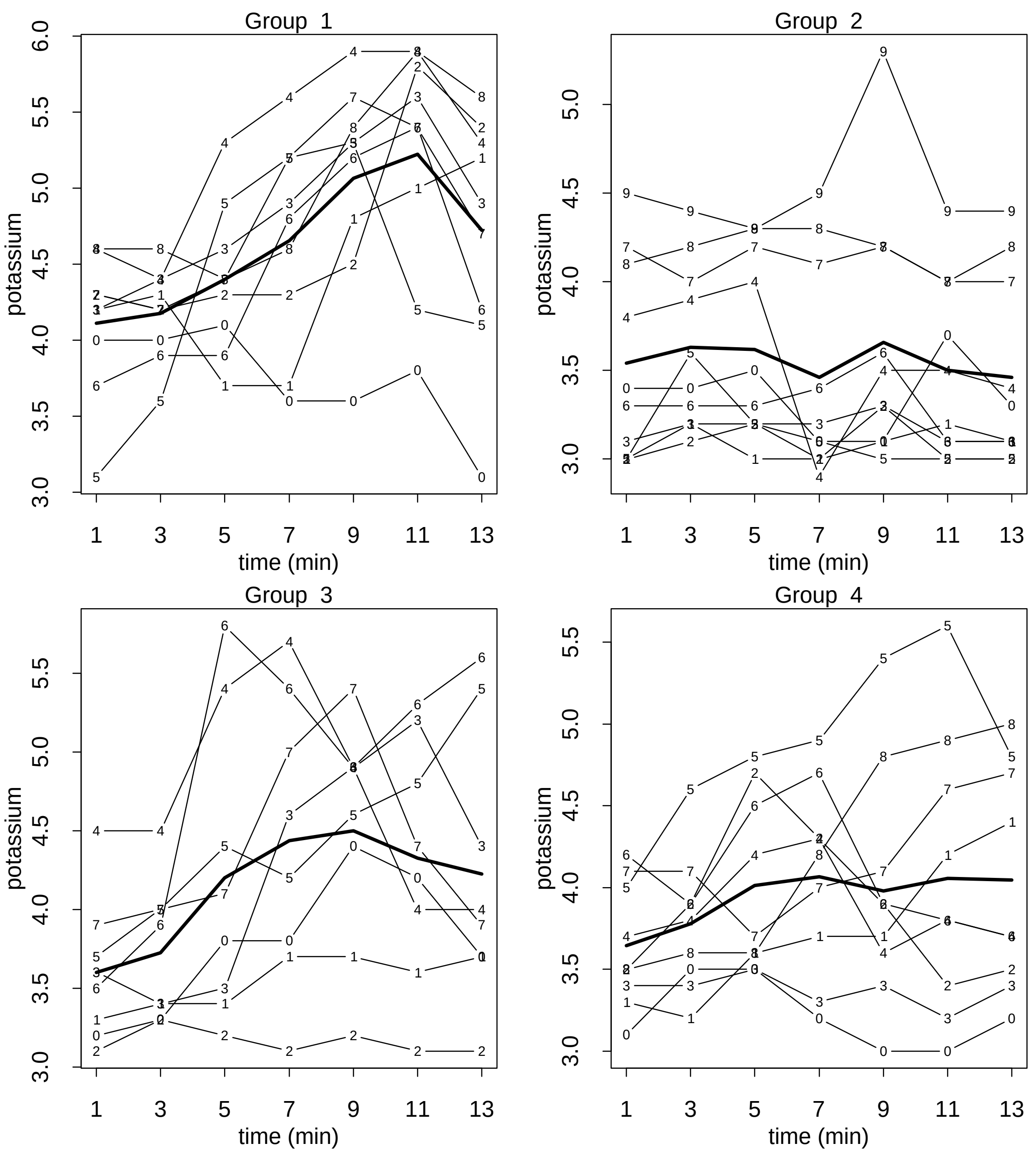


Figure 37: Plots of responses of each dog over time. Solid lines link within group average responses at each time point.

$$\begin{aligned} &= P_1P_3f + P_1P_4f + P_1(I-P_3-P_4)f \\ &\quad +(P_2-P_1)P_3f + (P_2-P_1)P_4f + (P_2-P_1)(I-P_3-P_4)f \\ &\quad +(I-P_2)P_3f + (I-P_2)P_4f + (I-P_2)(I-P_3-P_4)f \\ &= \mu + \beta(t-0.5) + s_1(t) \\ &\quad +\xi_k + \delta_k(t-0.5) + s_2(k,t) \\ &\quad +\alpha_{w(k)} + \gamma_{w(k)}(t-0.5) + s_3(k,w,t), \end{aligned} \tag{78}$$

where μ is a constant, $\beta(t-0.5)$ is the linear main effect of `time`, $s_1(t)$ is the smooth main effect of `time`, ξ_k is the main effect of `group`, $\delta_k(t-0.5)$ is the linear interaction between `time` and `group`, $s_2(k,t)$ is the smooth interaction between `time` and `group`, $\alpha_{w(k)}$ is the main effect of `dog`, $\gamma_{w(k)}(t-0.5)$ is the linear interaction between `time` and `dog`, and $s_3(k,w,t)$ is the smooth interaction between `time` and `dog`. We can calculate the main effect of `time` as $\beta(t-0.5)+s_1(t)$, the interaction between `time` and `group` as $\delta_k(t-0.5)+s_2(k,t)$, and the interaction between `time` and `dog` as $\gamma_{w(k)}(t-0.5)+s_3(k,w,t)$. The first six terms are fixed effects. The last three terms are random effects since they depend on the random variable w. Depending on `time` only, the first three terms represent the mean curve for all dogs. The middle three terms measure the departure of a particular group from the population mean curve. The last three terms measure the departure of a particular dog from the mean curve of a population from which the dog was chosen.

Based on the SS ANOVA decomposition (78), we will fit the following three models.

Model 1 includes the first seven terms in (78). It has a different population mean curve for each group plus a random intercept for each dog. We assume that $\alpha_i \overset{iid}{\sim} \mathrm{N}(0,\sigma_a^2)$, $\epsilon_{kij} \overset{iid}{\sim} \mathrm{N}(0,\sigma^2)$, and they are mutually independent.

Model 2 includes the first eight terms in (78). It has a different population mean curve for each group plus a random intercept and a random slope for each dog. We assume that $(\alpha_i,\gamma_i) \overset{iid}{\sim} \mathrm{N}((0,0),\mathrm{diag}(\sigma_1^2,\sigma_2^2))$, $\epsilon_{kij} \overset{iid}{\sim} \mathrm{N}(0,\sigma^2)$, and they are mutually independent.

Model 3 includes all nine terms in (78). It has a different population mean curve for each group plus a random intercept, a random slope and a smooth random effect for each dog. We assume that $(\alpha_i,\gamma_i) \overset{iid}{\sim} \mathrm{N}((0,0),\mathrm{diag}(\sigma_1^2,\sigma_2^2))$, $s_3(k,i,t)$'s are stochastic processes which are independent between dogs with mean zero and covariance function $\sigma_3^2[k_2(s)k_2(t)-k_4(s-t)]$, $\epsilon_{kij} \overset{iid}{\sim} \mathrm{N}(0,\sigma^2)$, and they are mutually independent.

Now we show how to fit these three models using `slm`. Notice that the fixed effects, $s_1(t)$, ξ_k, $\delta_k(t-0.5)$ and $s_2(k,t)$ are penalized. Model 1 and Model 2 can be fitted easily as follows.

```
> dog.fit1 <- slm(y~time, rk=list(cubic(time), shrink1(group),
                  rk.prod(kron(time-.5), shrink1(group)),
                  rk.prod(cubic(time), shrink1(group))),
                  random=list(dog=~1), data=dog.dat)
> dog.fit1
Semi-parametric linear mixed-effects model fit by REML
  Model: y ~ time
  Data: dog
  Log-restricted-likelihood: -180.4784

Fixed: y ~ time
```

```
(Intercept)        time
  3.8716387   0.4339031

Random effects:
 Formula: ~1 | dog
        (Intercept)  Residual
StdDev:   0.4980483 0.3924432


GML estimate(s) of smoothing parameter(s) : 0.0002334736 0.0034086209
                                            0.0038452499 0.0002049233
Equivalent Degrees of Freedom (DF):  13.00377
Estimate of sigma:  0.3924432


> dog.fit2 <- update(dog.fit1, random=list(dog=~time))
> dog.fit2
Semi-parametric linear mixed-effects model fit by REML
  Model: y ~ time
  Data: dog.dat
  Log-restricted-likelihood: -166.4478

Fixed: y ~ time
 (Intercept)      time
    3.876712 0.4196789

Random effects:
 Formula:  ~ time | dog
 Structure: General positive-definite
               StdDev   Corr
(Intercept) 0.4188372 (Inter
       time 0.5593078 0.025
   Residual 0.3403214

GML estimate(s) of smoothing parameter(s) :
1.674736e-04 3.286466e-03 5.778379e-03 8.944005e-05
Equivalent Degrees of Freedom (DF):  13.83273
Estimate of sigma:  0.3403214
```

To fit Model 3, we need to find a way to specify the smooth (non-parametric) random effect s_3. Let $\boldsymbol{t} = (t_1, \cdots, t_7)^T$, $u_i(\boldsymbol{t}) = (s_3(k, i, t_1), \cdots, s_3(k, i, t_7))^T$ and $\boldsymbol{u} = (\boldsymbol{u}_1^T, \cdots, \boldsymbol{u}_{36}^T)^T$. Then $u_i(\boldsymbol{t}) \overset{iid}{\sim} \text{N}(0, \sigma_3^2 D)$, where D is the RK of a cubic spline evaluated at the design points $\boldsymbol{t}$. Let $D = HH^T$ be the Cholesky decomposition of D, $\tilde{D} = \text{diag}(D, \cdots, D)$, and $G = \text{diag}(H, \cdots, H)$. Then $GG^T = \tilde{D}$. We can write $\boldsymbol{u} = G\boldsymbol{b}$, where $\boldsymbol{b} \sim \text{N}(0, \sigma_3^2 I)$. Then we can specify the random effects $\boldsymbol{u}$ using the matrix G.

```
> D <- cubic(dog.dat$time[1:7])
> H <- chol.new(D)
> G <- kronecker(diag(36), H)
> dog.dat$all <- rep(1,36*7)
> dog.fit3 <- update(dog.fit2, random=list(all=pdIdent(~G-1), dog=~time))
> dog.fit3
Semi-parametric linear mixed-effects model fit by REML
  Model: y ~ time
  Data: dog.dat
  Log-restricted-likelihood: -150.0885

Fixed: y ~ time
 (Intercept)      time
    3.885269 0.4046573

Random effects:
 Formula:  ~ G - 1 | all
 Structure: Multiple of an Identity
...

 Formula:  ~ time | dog %in% all
 Structure: General positive-definite
               StdDev   Corr
(Intercept) 0.4671536 (Inter
       time 0.5716811 -0.083
   Residual 0.2383432


GML estimate(s) of smoothing parameter(s) :
8.775590e-05 1.560998e-03 3.346731e-03 2.870384e-05
Equivalent Degrees of Freedom (DF):  15.37699
Estimate of sigma:  0.2383432
```

We could use the `anova` function for linear mixed-effects models to compare these three fits.

```
> anova(dog.fit1$lme.obj, dog.fit2$lme.obj, dog.fit3$lme.obj)
               Model df      AIC      BIC    logLik   Test  L.Ratio p-value
dog.fit1$lme.obj     1  8 376.9590 405.1307 -180.4795
dog.fit2$lme.obj     2 10 352.8955 388.1101 -166.4478 1 vs 2 28.06346  <.0001
dog.fit3$lme.obj     3 11 322.1771 360.9131 -150.0885 2 vs 3 32.71845  <.0001
```

So Model 3 is more favorable. We can calculate estimates of the population mean curves for four groups as follows.

```
> dog.grid <- data.frame(time=rep(seq(0,1,len=50),4),
                         group=as.factor(rep(1:4,rep(50,4))))
> e.dog.fit3 <- intervals(dog.fit3, newdata=dog.grid, terms=rep(1,6))
```

Figure 38 plots these mean curves and their 95% confidence intervals.

We have shrunk the effects of `group` factor toward constants. That is, we have penalized the `group`

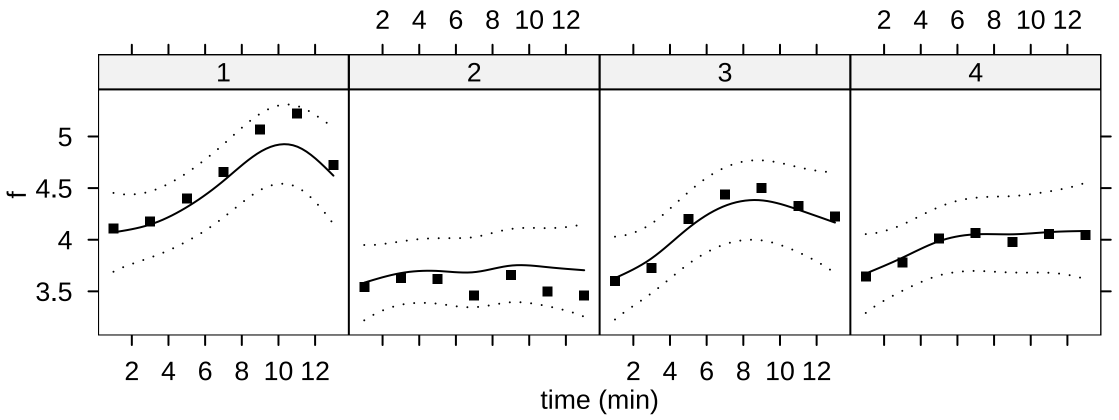


Figure 38: Estimates of the population mean response curve as a function of time with their 95% confidence intervals. Circles are within group average responses. Solid lines are estimates. Dotted lines are 95% confidence intervals.

main effect ξ_k and the linear `time-group` interaction $\delta_k(t-0.5)$ in the SS ANOVA decomposition (78). From Figure 38 we can see that the estimated population mean curve for group 2 are biased upward, while the estimated population mean curves for group 1 is biased downward. This is because responses from group 2 are smaller while responses from group 1 are larger than those from groups 3 and 4. Thus their estimates are pulled towards the overall mean. Shrinkage estimates in this case may not be advantageous since `group` only has four levels. One may want to leave ξ_k and $\delta_k(t-0.5)$ terms unpenalized to reduce biases. We can re-write the fixed effects in (78) as

$$\begin{aligned} f_k(t) &\overset{def}{=} \mu + \beta(t-0.5) + s_1(t) + \xi_k + \delta_k(t-0.5) + s_2(k,t) \\ &= [\mu + \xi_k] + [\beta(t-0.5) + \delta_k(t-0.5)] + [s_1(t) + s_2(k,t)] \\ &= \tilde{\xi}_k + \tilde{\delta}_k(t-0.5) + \tilde{s}_2(k,t). \end{aligned} \tag{79}$$

$f_k(t)$ is the population mean curve for group k. Assume $f_k \in W_2([0,1])$. Define penalty as $\int_0^1 (f_k''(t))^2 dt = ||\tilde{s}_2(k,t)||^2$. Then the constant term $\tilde{\xi}_k$ and the linear term $\tilde{\delta}_k(t-0.5)$ are not penalized. Note that this form of penalty was used in Wang (1998b). We can refit *Model 1*, *Model 2* and *Model 3* under this new form of penalty as follows.

```
> dog.fit4 <- slm(y~group*time, rk=list(rk.prod(cubic(time), kron(group==1)),
                                        rk.prod(cubic(time), kron(group==2)),
                                        rk.prod(cubic(time), kron(group==3)),
                                        rk.prod(cubic(time), kron(group==4))),
                 random=list(dog=~1), data=dog.dat)
> dog.fit4
Semi-parametric linear mixed-effects model fit by REML
  Model: y ~ group * time
  Data: dog
  Log-restricted-likelihood: -173.0538

Fixed: y ~ group * time
```

```
(Intercept)      group2      group3      group4        time group2:time
 4.30592658 -0.70377899 -0.45507981 -0.54411601  0.70494987 -0.80352151
group3:time group4:time
-0.02604081 -0.33005253

Random effects:
 Formula: ~1 | dog
        (Intercept)  Residual
StdDev:   0.4986658 0.3880457


GML estimate(s) of smoothing parameter(s) : 2.470589e-05 1.290422e+00
                                            7.737418e-05 8.136673e-04
Equivalent Degrees of Freedom (DF):  13.05131
Estimate of sigma:  0.3880457

> dog.fit5 <- update(dog.fit4, random=list(dog=~time))
> dog.fit5
Semi-parametric linear mixed-effects model fit by REML
  Model: y ~ group * time
  Data: dog
  Log-restricted-likelihood: -158.7129

Fixed: y ~ group * time
(Intercept)      group2      group3      group4        time group2:time
 4.31780496 -0.71566053 -0.46280282 -0.55513686  0.68418592 -0.78275742
group3:time group4:time
-0.01217737 -0.30620677

Random effects:
 Formula: ~time | dog
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev    Corr
(Intercept) 0.4225180 (Intr)
time        0.5536860 -0.005
Residual    0.3367037


GML estimate(s) of smoothing parameter(s) : 1.714698e-05 3.894053e+00
                                            5.640272e-05 4.988060e-04
Equivalent Degrees of Freedom (DF):  13.74553
Estimate of sigma:  0.3367037

> dog.fit6 <- update(dog.fit5, random=list(all=pdIdent(~G-1), dog=~time))
> dog.fit6
```

```
Semi-parametric linear mixed-effects model fit by REML
  Model: y ~ group * time
  Data: dog
  Log-restricted-likelihood: -142.3147

Fixed: y ~ group * time
(Intercept)      group2      group3      group4        time group2:time
 4.33679882 -0.72758015 -0.47372184 -0.57605073  0.65150749 -0.75063284
group3:time group4:time
 0.00427347 -0.26053905

Random effects:
 Formula:  ~ G - 1 | all
...

 Formula: ~time | dog %in% all
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev    Corr
(Intercept) 0.4688705 (Intr)
time        0.5665381 -0.104
Residual    0.2394261


GML estimate(s) of smoothing parameter(s) : 8.175602e-06 1.567092e+00
                                            3.081125e-05 1.289556e-01
Equivalent Degrees of Freedom (DF):  13.43655
Estimate of sigma:  0.2394261

> anova(dog.fit4$lme.obj, dog.fit5$lme.obj, dog.fit6$lme.obj)
                 Model df      AIC      BIC    logLik   Test  L.Ratio p-value
dog.fit4$lme.obj     1 14 374.1076 423.0680 -173.0538
dog.fit5$lme.obj     2 16 349.4257 405.3804 -158.7129 1 vs 2 28.68192  <.0001
dog.fit6$lme.obj     3 17 318.6295 378.0814 -142.3148 2 vs 3 32.79624  <.0001
```

We note that the estimates are similar to, but not exactly the same as, those calculated by SAS `proc mixed` in Wang (1998b). The differences may be caused by different starting values and/or numerical procedures. We calculate estimates of the population mean curves for four groups as before.

```
> e.dog.fit6 <- intervals(dog.fit6, newdata=dog.grid, terms=rep(1,12))
```

Figure 39 plots these mean curves and their 95% confidence intervals.

We now show how to calculate predictions for dogs. Prediction for dog i in group k on a point t can be computed as $\hat{\xi}_k+\hat{\delta}_k(t-0.5)+\hat{s}_2(k,t)+\hat{\alpha}_i+\hat{\gamma}_i(t-0.5)+\hat{s}_3(k,i,t)$. Prediction of the fixed effects can be computed using the `prediction.slm` function. $\hat{\alpha}_i$ and $\hat{\gamma}_i$ are provided in the estimates of the random effects. Thus we only need to compute $\hat{s}_3(k,i,t)$. Suppose that we want to predict s_3 for dog i in group k on a vector of points $\boldsymbol{z}_i=(z_{i1},\cdots,z_{ig_i})^T$. Let $u_i(\boldsymbol{z}_i)=(s_3(k,i,z_{i1}),\cdots,s_3(k,i,z_{ig_i}))^T$,

$$R_i=\mathrm{Cov}(u_i(\boldsymbol{z}_i),u_i(\boldsymbol{t}))=\{R_1(z_{ik},t_j)\}_{k=1}^{g_i}{}_{j=1}^{7},$$

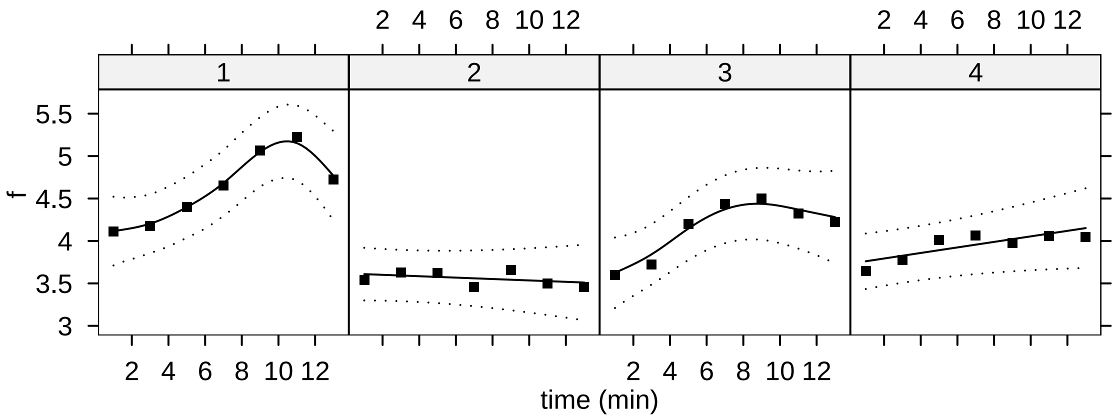


Figure 39: Estimates of the population mean response curve as a function of time with their 95% confidence intervals. Circles are within group average responses. Solid lines are estimates. Dotted lines are 95% confidence intervals.

where $R_1(z,t) = k_2(z)k_2(t) - k_4(z-t)$. Let $\boldsymbol{z} = (\boldsymbol{z}_1^T, \cdots, \boldsymbol{z}_{36}^T)^T$, $R = \text{diag}(R_1, \cdots, R_{36})$ and $\hat{\boldsymbol{u}}$ be the prediction of $\boldsymbol{u}$. We then can compute the prediction for all dogs as

$$\hat{u}(\boldsymbol{z}) = R\tilde{D}^{-1}\hat{\boldsymbol{u}}.$$

However the smallest eigen-value of D is close to zero, thus $\tilde{D}^{-1}$ cannot be calculated precisely. We will use an alternative approach which does not require inverting D. Denote the estimate of $\boldsymbol{b}$ as $\hat{\boldsymbol{b}}$. If we can find a vector $\boldsymbol{c}$ (need not to be unique) such that

$$G^T\boldsymbol{c} = \hat{\boldsymbol{b}}. \tag{80}$$

Then

$$\hat{u}(\boldsymbol{z}) = R\tilde{D}^{-1}\hat{\boldsymbol{u}} = R\tilde{D}^{-1}(G\hat{\boldsymbol{b}}) = R\tilde{D}^{-1}(GG^T\boldsymbol{c}) = R\tilde{D}^{-1}(GG^T)\boldsymbol{c} = R\boldsymbol{c}.$$

So the task now is to solve (80). Let

$$G = (Q_1, Q_2)\begin{pmatrix} V \\ \mathbf{0} \end{pmatrix}$$

be the QR decomposition of G. We consider $\boldsymbol{c}$ in the space spanned by Q_1: $\boldsymbol{c} = Q_1\boldsymbol{\alpha}$. Then from (80), $\boldsymbol{\alpha} = V^{-T}\hat{\boldsymbol{b}}$. Thus $\boldsymbol{c} = Q_1V^{-T}\hat{\boldsymbol{b}}$ is a solution to (80). This approach also applies to the situation when D is singular. In the following we calculate predictions for all 36 dogs on a set of grid points.

```
> dog.grid2 <- data.frame(time=rep(seq(0,1,len=50),36),
                          dog=rep(1:36,rep(50,36)))
> R <- kronecker(diag(36),cubic(dog.grid2$time[1:50],dog.dat$time[1:7]))
> b <- as.vector(dog.fit6$lme.obj$coef$random$all)
> G.qr <- qr(G)
> c.coef <- qr.Q(G.qr)%*%solve(t(qr.R(G.qr)))%*%b
> tmp1 <- c(rep(e.dog.fit6$fit[dog.grid$group==1],9),
            rep(e.dog.fit6$fit[dog.grid$group==2],10),
            rep(e.dog.fit6$fit[dog.grid$group==3],8),
            rep(e.dog.fit6$fit[dog.grid$group==4],9))
> tmp2 <- as.vector(rep(dog.fit6$lme.obj$coef$random$dog[,1],rep(50,36)))
> tmp3 <- as.vector(kronecker(dog.fit6$lme.obj$coef$random$dog[,2],
                              dog.grid2$time[1:50]))
> u.new <- as.vector(R%*%c.coef)
> p.dog.fit6 <- tmp1+tmp2+tmp3+u.new
```

Predictions for dogs 1, 2, 26 and 27 are shown in Figure 40.

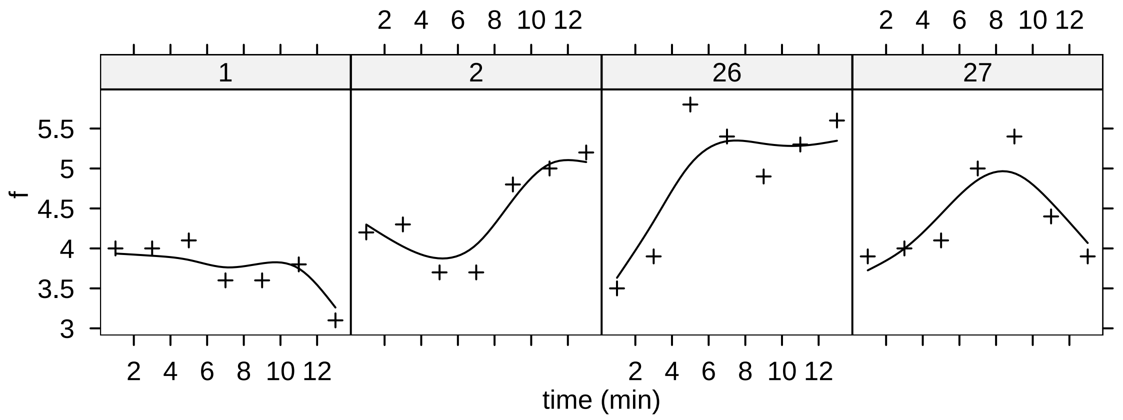


Figure 40: Plots of predictions for dogs 1, 2, 26 and 27. Circles are observations. Solid lines are predictions.

Observations close in time from the same dog may be correlated. In the following we fit a first-order autoregressive structure for random errors within each dog.

```
> dog.fit7 <- update(dog.fit6, cor=corAR1(form=~1|all/dog))
> dog.fit7
  Model: y ~ group * time
  Data: dog
  Log-restricted-likelihood: -132.6711

Fixed: y ~ group * time
 (Intercept)       group2       group3       group4         time  group2:time
 4.343695660 -0.760617887 -0.483550117 -0.620851406  0.629970047 -0.715780104
 group3:time  group4:time
 0.008927178 -0.239117441

Random effects:
 Formula:  ~ G - 1 | all
 Structure: Multiple of an Identity
...
```

```
 Formula: ~time | dog %in% all
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev    Corr
(Intercept) 0.2779298 (Intr)
time        0.2909205 0.991
Residual    0.4456182

Correlation structure of class corAR1 representing
      Phi
0.6359198

GML estimate(s) of smoothing parameter(s) : 2.921150e-05 1.954848e+06
                                            1.098781e-04 1.893688e+01
Equivalent Degrees of Freedom (DF):  11.63503
Estimate of sigma:  0.4030084
```

By convention in nlme, model dog.fit7 may also be fitted as

```
dog.fit8 <- update(dog.fit6, cor=corAR1(form=~1))
```

8.8 Rock Data

The rock data in Venables and Ripley (1998) contains measurements on four cross-sections of each of 12 oil-bearing rocks. The aim is to predict permeability (perm) from three other measurements: the total area (area), total perimeter (peri) and a measure of "roundness" of the pores in the rock cross-section (shape). Venables and Ripley (1998) fitted this data set with a projection pursuit (PP) regression model. Here we show how to use the snr function to fit the following PP regression model

$$\log(\texttt{perm}) = f(\alpha_1 \times \texttt{area} + \alpha_2 \times \texttt{peri} + \times \alpha_3 \texttt{shape}) + \epsilon, \tag{81}$$

where $\alpha_1^2 + \alpha_2^2 + \alpha_3^2 = 1$ and $\alpha_3 > 0$ for identifiability.

We first fit model (81) using the R function ppr as in Venables and Ripley (1998). Then we fit the same model using snr with initial values from the ppr fit. We use a TPS with $d = 1$ and $m = 2$ to model f. We made the following transformations: dividing area and peri by 10000, and taking the natural logarithm of perm.

```
> data(rock)
> attach(rock)
> area1 <- area/10000; peri1 <- peri/10000
> rock.ppr <- ppr(log(perm) ~ area1 + peri1 + shape,
                  data=rock, nterms=1, max.terms=5)
> summary(rock.ppr)
Call:
ppr(formula = log(perm) ~ area1 + peri1 + shape, rock.Rdata = rock,
    nterms = 1, max.terms = 5)

Goodness of fit:
  1 terms   2 terms   3 terms   4 terms   5 terms
19.590843  8.737806  5.289517  4.745799  4.490378
```

```
Projection direction vectors:
       area1        peri1        shape
 0.347565455 -0.937641311  0.005198698

Coefficients of ridge terms:
[1] 1.495419

> rock.snr <- snr(log(perm) ~ f(a1*area1+a2*peri1+sqrt(1-a1^2-a2^2)*shape),
                 func=f(u)~list(~u,tp(u)),
                 params=list(a1+a2~1),
                 start=list(params=c(.34,-.94)))
> rock.snr
Semi-parametric Nonlinear Regression Model Fit
 Model: log(perm) ~ f(a1 * area1 + a2 * peri1 + sqrt(1 - a1^2 - a2^2) *      shape)

 Log-likelihood: -50.04593

Coefficients:
        a1         a2
 0.3449282 -0.9386264

Smoothing spline:
 GCV estimate(s) of smoothing parameter(s): 1.249731e-06
 Equivalent Degrees of Freedom (DF):  4.144281

Residual standard error: 0.770572
Number of Observations: 48


Converged after 5 iterations

> a <- seq(min(z),max(z),len=50)
> rock.snr.ci <- intervals(rock.snr,newdata=data.frame(u=a))
> rock.ppr.p <- predict(rock.ppr)
```

`snr` converged after 5 iterations. Let $z = \hat{\alpha_1} \times \texttt{area1} + \hat{\alpha_2} \times \texttt{peri1} + \sqrt{1 - \hat{\alpha_1}^2 - \hat{\alpha_2}^2} \times \texttt{shape}$. In Figure 41, against the z values, we plot the observations as circles, the fitted curve f from `rock.snr` as the solid line with 95% Bayesian confidence intervals as the dotted lines, and the shape of the fitted curve f from `rock.ppr` as the dashed line. The fitted curves from `snr` and `ppr` have different shapes.

8.9 CO_2 Uptake Data

This dataset comes from a study of cold tolerance of a C_4 grass species, *Echinochloa crus-galli*, described in Potvin, Lechowicz and Tardif (1990) and Pinheiro and Bates (2000). A total of twelve four-week-old `plants` were used in the study. There were two `types` of plants: six from Quebec and six from Mississippi. Two `treatments`, nonchilling and chilling, were assigned to three plants of each

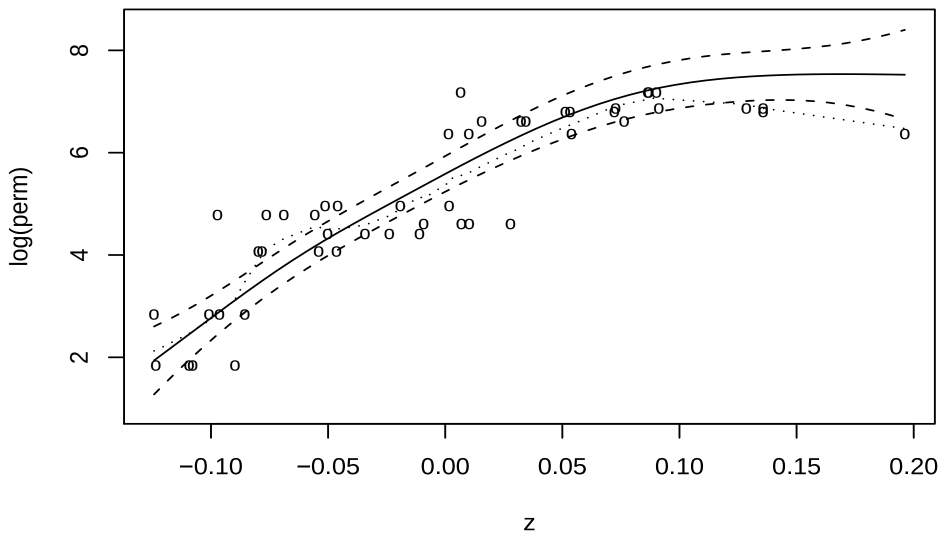


Figure 41: Plot of `perm` on logarithm scale vs z values as circles. The solid line is the fitted curve from `rock.snr`. Two dotted lines are its 95% Bayesian confidence intervals. The dashed line represents the shape of the fitted curve from `rock.ppr`.

type. Nonchilled plants were kept at 26^oC and chilled plants were subject to 14 hours of chilling at 7^oC. After 10 hours of recovery at 20^oC, CO_2 `uptake` rates (in $\mu mol/m^2s$) were measured for each plant at seven `concentrations` of ambient CO_2 in increasing, consecutive order. Plots of observations are shown in Figure 42 as circles. The objective of the experiment was to evaluate the effect of plant `type` and chilling `treatment` on the CO_2 `uptake`.

Pinheiro and Bates (2000) gave detailed analyses of this dataset based on NLMMs using their software `nlme`. They reached the following model:

$$\begin{aligned}
\texttt{uptake}_{ij} &= e^{\phi_{1i}}\{1-e^{-e^{\phi_{2i}}(\texttt{conc}_j-\phi_{3i})}\}+\epsilon_{ij}, \quad i=1,\cdots,12, \quad j=1,\cdots,7,\\
\phi_{1i} &= \beta_{11}+\beta_{12}\texttt{Type}+\beta_{13}\texttt{Treatment}+\beta_{14}\texttt{Treatment:Type}+b_i,\\
\phi_{2i} &= \beta_{21},\\
\phi_{3i} &= \beta_{31}+\beta_{32}\texttt{Type}+\beta_{33}\texttt{Treatment}+\beta_{34}\texttt{Treatment:Type}, \qquad (82)
\end{aligned}$$

where $\texttt{uptake}_{ij}$ denotes the CO_2 uptake rate of `plant` i at CO_2 ambient concentration $\texttt{conc}_j$; `Type` equals 0 for plants from Quebec and 1 for plants from Mississippi, `Treatment` equals 0 for chilled plants and 1 for control plants; $e^{\phi_{1i}}$, $e^{\phi_{2i}}$ and ϕ_{3i} denote respectively the asymptotic uptake rate, the uptake growth rate, and the maximum ambient CO_2 concentration at which no uptake is verified for plant i; random effects $b_i \overset{iid}{\sim} N(\mathbf{0},\sigma_b^2)$; and random errors $\epsilon_{ij} \overset{iid}{\sim} N(0,\sigma^2)$. Note that we used exponential transformations to enforce the positivity constraints.

```
> options(contrasts=rep("contr.treatment", 2))
> co2.fit1 <- nlme(uptake~exp(a1)*(1-exp(-exp(a2)*(conc-a3))),
```

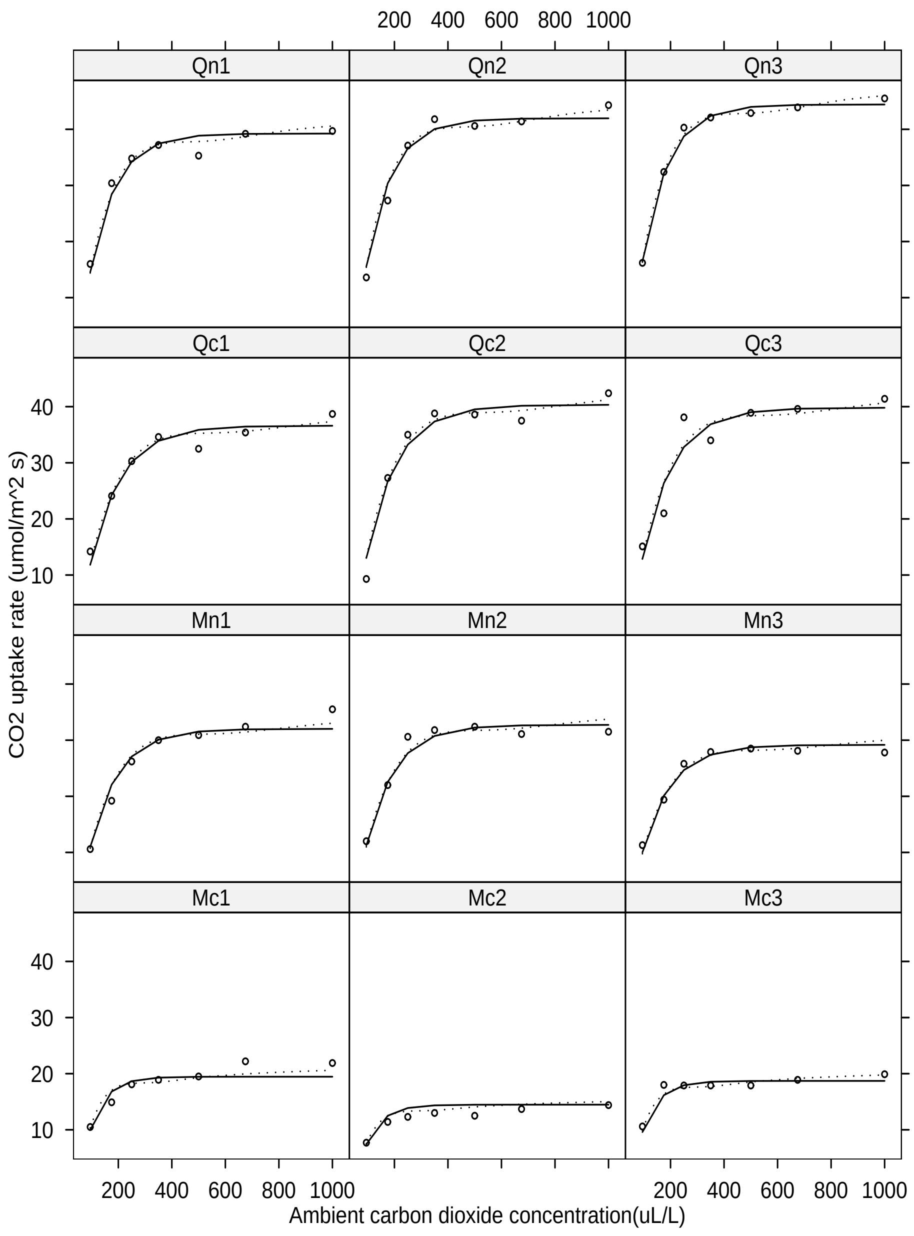


Figure 42: Plots of the data and fitted curves. Circles are observations. Solid lines represent SNM model fits from `co2.fit2`. Dotted lines represent NLMM fits from `co2.fit1`. In the strip name for each plot, "Q" indicates Quebec, "M" Mississippi, "c" chilled, "n" nonchilled, and "1", "2", "3" the replicate numbers.

```
                    fixed=list(a1+a2~Type*Treatment,a3~1),
                    random=a1~1, groups=~Plant, data=CO2,
                    start=c(log(30),0,0,0,log(0.01),0,0,0,50))
> co2.fit1
Nonlinear mixed-effects model fit by maximum likelihood
  Model: uptake ~ exp(a1) * (1 - exp(-exp(a2) * (conc - a3)))
 Data: CO2
       AIC      BIC    logLik
  393.2869 420.0259 -185.6434

Random effects:
 Formula: a1 ~ 1 | Plant
        a1.(Intercept) Residual
StdDev:     0.08221494 1.857658

Fixed effects: list(a1 + a2 ~ Type * Treatment, a3 ~ 1)
                                       Value Std.Error DF   t-value p-value
a1.(Intercept)                       3.73338  0.052536 64  71.06301  <.0001
a1.TypeMississippi                  -0.29080  0.075535 64  -3.84990  0.0003
a1.Treatmentchilled                 -0.07274  0.074633 64  -0.97459  0.3334
a1.TypeMississippi:Treatmentchilled -0.51321  0.109733 64  -4.67688  <.0001
a2.(Intercept)                      -4.57570  0.086116 64 -53.13387  <.0001
a2.TypeMississippi                  -0.09635  0.103950 64  -0.92687  0.3575
a2.Treatmentchilled                 -0.17048  0.091377 64  -1.86565  0.0667
a2.TypeMississippi:Treatmentchilled  0.70555  0.205851 64   3.42748  0.0011
a3                                  49.98833  4.576255 64  10.92341  <.0001
...
```

Fits of model (82) are plotted in Figure 42 as dotted lines. Based on model (82), one may conclude that the CO2 uptake is higher for plants from Quebec and that chilling, in general, results in lower uptake, and its effect on Mississippi plants is much larger than on Quebec plants. These conclusions are comparable to the results in Potvin et al. (1990).

We aim to use this dataset to demonstrate how to fit SNM models with covariates, and how to check if an NLMM is appropriate. As an extension of (82), we fit the following SNM model

$$\begin{aligned}
\texttt{uptake}_{ij} &= e^{\phi_{1i}} f(e^{\phi_{2i}}(\texttt{conc}_j - \phi_{3i})) + \epsilon_{ij}, \quad i = 1, \cdots, 12, \quad j = 1, \cdots, 7, \\
\phi_{1i} &= \beta_{12}\texttt{Type} + \beta_{13}\texttt{Treatment} + \beta_{14}\texttt{Treatment:Type} + b_i, \\
\phi_{2i} &= \beta_{21}, \\
\phi_{3i} &= \beta_{31} + \beta_{32}\texttt{Type} + \beta_{33}\texttt{Treatment} + \beta_{34}\texttt{Treatment:Type},
\end{aligned} \tag{83}$$

where $f \in W_2([0,T])$ for some fixed $T > 0$ and the second stage model is paralleled to (82). In order to test if the parametric model (82) is appropriate, we construct the following L-spline to model f. The hypothesis is H_0: $f \in span\{1 - e^{-t}\}$. Let $L = D + D^2$, where D^j denotes the jth derivative operator. The kernel space of L is $\mathcal{H}_1 = span\{1, e^{-t}\}$. Define a linear operator $\mathcal{B} : W_2[0,T] \to \mathcal{R}^2$ such that $\mathcal{B}f = (f(0), f'(0))$. Let $\mathcal{H}_2 = ker\mathcal{B}$. Define inner products on $\mathcal{H}_1, \mathcal{H}_2$, and $W_2[0,T]$ as $< f, g >_1 = (\mathcal{B}f)^T(\mathcal{B}g)$, $< f, g >_2 = \int_0^T (Lf)(Lg)dt$ and $< f, g > = < f, g >_1 + < f, g >_2$. Then it is easy to check that $W_2[0,T] = \mathcal{H}_1 \oplus \mathcal{H}_2$. The reproducing kernel of $\mathcal{H}_2$ is given in (12).

Note that β_{11} in (82) is excluded from (83) to make f free of constraint on the vertical scale. We need the side conditions that $f(0) = 0$ and $f(t) \neq 0$ for $t \neq 0$ to separate β_{31} from f. The first condition reduces $\mathcal{H}_1$ to $\mathcal{H}_1 = \text{span}\{1 - e^{-t}\}$ and is satisfied by all functions in $\mathcal{H}_2$. We do not enforce the second condition because it is satisfied by all reasonable estimates. Thus the null space for f becomes $\mathcal{H}_1^0 = \text{span}\{1 - e^{-t}\}$ and the model space is $\mathcal{H}_1^0 \oplus \mathcal{H}_2$. The penalty is still $J(f) = \int (Lf)^2$.

With the initial values chosen from the NLMM fit, our program converged after 5 iterations.

```
> M <- model.matrix(~Type*Treatment, data=CO2)[,-1]
> co2.fit2 <- snm(uptake~exp(a1)*f(exp(a2)*(conc-a3)),
                func=f(u)~list(~I(1-exp(-u))-1,lspline(u, type="exp")),
                fixed=list(a1~M-1,a3~1,a2~Type*Treatment),
                random=list(a1~1), group=~Plant, verbose=T,
                start=co2.fit1$coe$fixed[c(2:4,9,5:8)], data=CO2)
> summary(co2.fit2)
Semi-parametric Nonlinear Mixed Effects Model fit
  Model: uptake ~ exp(a1) * f(exp(a2) * (conc - a3))
  Data: CO2

       AIC     BIC    logLik
  406.4864 441.625 -188.3760

Random effects:
 Formula: a1 ~ 1 | Plant
        a1.(Intercept) Residual
StdDev:     0.09304172 1.816200

Fixed effects: list(a1 ~ M - 1, a3 ~ 1, a2 ~ Type * Treatment)
                                        Value Std.Error DF   t-value p-value
a1.MTypeMississippi                  -0.28569  0.055952 65  -5.10591  <.0001
a1.MTreatmentchilled                 -0.07212  0.054741 65  -1.31739  0.1923
a1.MTypeMississippi:Treatmentchilled -0.54879  0.099184 65  -5.53309  <.0001
a3                                   50.67565  4.226094 65  11.99113  <.0001
a2.(Intercept)                       -4.56698  0.085130 65 -53.64708  <.0001
a2.TypeMississippi                   -0.12162  0.098575 65  -1.23377  0.2217
a2.Treatmentchilled                  -0.16161  0.087009 65  -1.85740  0.0678
a2.TypeMississippi:Treatmentchilled   0.81924  0.211587 65   3.87187  0.0003
...
GCV estimate(s) of smoothing parameter(s): 1.864811
Equivalent Degrees of Freedom (DF):  4.867178

Converged after 5 iterations
```

Fits of model (83) are plotted in Figure 42 as solid lines. Since the data set is small, different initial values may lead to different estimates. However, the overall fits are similar. We also fitted models with AR(1) within-subject correlations and covariate effects on ϕ_3. None of these models improve fits significantly. The estimates are comparable to the nonlinear fits and the conclusion is similar to that based on (82).

To check if the parametric NLMM (82) is appropriate, we calculated approximate posterior means and variances using the function `intervals`. Then we plotted the estimated f (overall), its projection onto $\mathcal{H}_1^0$ (the parametric part) and $\mathcal{H}_2$ (the smooth part) in Figure 43.

```
> co2.grid2 <- data.frame(u=seq(0.3, 11, len=50))
> co2.ci <- intervals(co2.fit2, newdata=co2.grid2,
                   terms=matrix(c(1,1,1,0,0,1), ncol=2,byrow=T))
> plot.bCI(co2.ci,x=co2.grid2$u,layout=c(3,1),
          type.name=c("overall","parametric","smooth"))
```

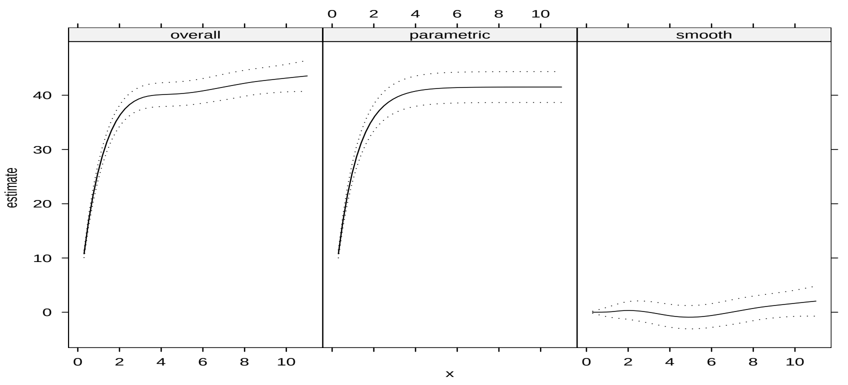


Figure 43: The overall fit of the common curve (left), its projection onto $\mathcal{H}_1^0$ (center) and $\mathcal{H}_2$ (right). Solid lines are fitted values. Dash lines are approximate 95% Bayesian confidence limits.

The zero line is inside the Bayesian confidence intervals for the projection onto $\mathcal{H}_2$ (smooth component) which suggests that the parametric NLME model (82 is adequate.

For L-splines, Heckman and Ramsay (2000) showed that selecting the right form of penalty function via a differential operator L can reduce the bias. An L-spline allows us to incorporate prior knowledge on the main features of f into the penalty. Usually the form of L is known with coefficients depending on some unknown parameters. Heckman and Ramsay (2000) proposed several methods to estimate these parameters. When the whole model can be written in the form of an SNM model as in this example, our methods can also be used to estimate the penalty. Our approach is the same as the PL (penalized likelihood) approach in Heckman and Ramsay (2000). They commented that the PL method is "philosophically appealing". However it is also "time-consuming" because a grid search was used. Our method estimates the coefficients in L and the functions f iteratively, thus making it less computationally intensive. In addition, we allow these coefficients to depend on covariates.

8.10 Core Body Temperature Data

Circadian rhythms have become a topic of intensive research during the last 40 years. Often the period is fixed as 24 hours due to the entrainment. However, for situations such as when individuals are denied exposure to zeitgebers or are living on irregular sleep/wake schedule, the period of the rhythm must be estimated from data. Several methods such as spectral analysis, autocorrelation, Enright's periodgram and linear regression of onset have been proposed in literature (Arendt, Mirors and Waterhouse 1989, Refinetti 1993). The first three methods require equidistant samples and the last method requires a marker for the onset of activity. All four methods requres several cycles of observations. Our method illustrated below, under the assumption that the shape of circadian rhythm remains unchanged, can be used for iregular design with few cycles.

This dataset is taken from a study on mood disorders. We thank Daniel Buysse, MD, and Hernando Ombao, PhD, from the University of Pittsburgh for permission to use part of the data. For several healthy and depressed subjects, core body temperature (`bt`) is measured over `time` every 5 minutes for 48 hours. Each subject is put into an isolated room free of time cues. We only use data from one subject and scale `time` variable into $[0, 1]$. Measurements of this subject are shown in Figure 44. The goal was to investigate possible effects of mode disorder on the biological rhythms. For other physiological measurements, it is known that biological rhythms exist. They are controlled by the internal clock with a period close to 25 hours. We will assume that biological rhythms exist for core body temperature with a similar period. Then the 48 hour observations contain at least one period. We need to estimate the period and the shape function for each subject. The common practice of fitting a sinusoidal function to each subject may not be appropriate (Wang and Brown 1996). We demonstrate how to use SNR models to fit such data for further analyses. See Hall, Reimann and Rice (2001) for another interesting example and discussions on identifiability.

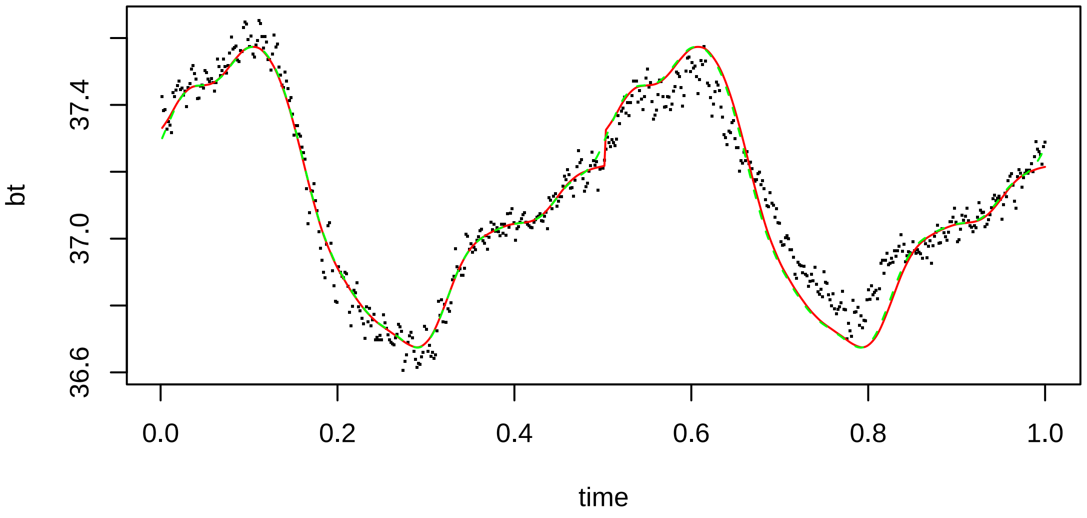


Figure 44: Points are observations. Solid red line is the fit from model (84). Dotted green line is the fit from model (85).

We consider the following model

$$\texttt{bt}_i = f(\texttt{time}_i - \alpha \times \text{int}(\frac{\texttt{time}_i}{\alpha})) + \epsilon_i, \tag{84}$$

where f is a periodic function on $[0, \alpha]$, $\alpha > 0$ is the unknown period, and the $\text{int}(x)$ returns the integer part of x. We assume an AR(1) correlation structure for random errors ϵ_i's. Because the period is not unique, we define α as the smallest period to make f identifiable. It is evident that the period is close to 24 hours. Thus we use $\alpha = 0.5$ as the initial value. To relax the positive constraint, we reparametrize α as a^2 in the following program. We used the GML method to select the smoothing parameter since the GCV under-smoothes in this case.

```
> cbt.fit1 <- snr(bt~f(time-a**2*floor(time/a**2)),
                func=f(u)~list(~1, periodic(u)),
                params=list(a~1), data=cbt, spar="m",
                start=list(params=c(sqrt(.5))), cor=corAR1())
> cbt.fit1
Semi-parametric Nonlinear Regression Model Fit
 Model: bt ~ f(time - a^2 * floor(time/a^2))
 Data: cbt
 Log-likelihood: 1141.090

Coefficients:
        a
0.7094294

Correlation Structure: AR(1)
 Formula: ~1
 Parameter estimate(s):
      Phi
0.8456162
Smoothing spline:
 GML estimate(s) of smoothing parameter(s): 6.161782e-08
 Equivalent Degrees of Freedom (DF):  21.72078

Residual standard error: 0.06496144
Number of Observations: 576

Converged after 4 iterations

> p.cbt.fit1 <- predict(cbt.fit1)
```

We can also fit an equivalent model

$$\texttt{bt}_i = g(\beta \times \texttt{time}_i) + \epsilon_i, \qquad (85)$$

where g is an unknown periodic function with period 1, and $\beta > 0$ is unknown scale parameter. Letting $f(t) = g(\beta t)$, it is easy to check that models (84) and (85) are equivalent with $\alpha = 1/\beta$. We can fit model (85) with initial value $\beta = 2$ by

```
> cbt.fit2 <- snr(bt~f(a**2*time), func=f(u)~list(~1, periodic(u)),
                params=list(a~1), data=cbt, spar="m",
                start=list(params=c(sqrt(2))), cor=corAR1())
> cbt.fit2
Semi-parametric Nonlinear Regression Model Fit
 Model: bt ~ f(a^2 * time)
 Data: cbt
 Log-likelihood: 1143.045

Coefficients:
       a
1.412535

Correlation Structure: AR(1)
 Formula: ~1
 Parameter estimate(s):
      Phi
0.8471175
Smoothing spline:
 GML estimate(s) of smoothing parameter(s): 4.377812e-07
 Equivalent Degrees of Freedom (DF):  21.38732

Residual standard error: 0.0649923
Number of Observations: 576

Converged after 4 iterations

> p.cbt.fit2 <- predict(cbt.fit2)
```
Fits from these two models are shown in Figure 44. They are very close. Both models suggest high auto-correlation. The estimate of β is $\hat{\beta} = 1.412535^2 = 1.995255$. $1/1.995255 = 0.501189$, close to $\hat{\alpha} = .709^2 = 0.5032901$. The estimated period based on models (84) and (85) are 24.15792 and 24.05707 hours respectively.

8.11 Canadian Temperature Data

The dataset, downloaded from the website `http://www.psych.mcgill.ca/faculty/ramsay/fda.html`, includes mean `monthly temperatures` at 35 Canadian weather `stations`. The left panel in Figure 45 shows the plots of mean `temperature` from 4 selected weather `stations`.

Ramsay and Silverman (1997) and Ramsay and Li (1998) used this dataset to illustrate the usefulness of Functional Data Analysis (FDA). Among others, they addressed the following two questions: how to measure the departure from a sinusoidal model and how to register the curves for further analyses. In the following, we approach these questions with SNM models.

We assume that mean `temperatures` over `month` at all weather `stations` share a common shape function. Variation between `stations` are modeled by vertical shift, vertical scale and horizontal shift parameters. That is, we assume that

$$\texttt{temp}_{ij} = \mu + b_{1i} + e^{b_{2i}} f(\texttt{month}_{ij} - \frac{e^{b_{3i}}}{1+e^{b_{3i}}}) + \epsilon_{ij}, \quad i = 1, \cdots, 35, \quad j = 1, \cdots, 12, \tag{86}$$

where $\texttt{temp}_{ij}$ is the `temperature` of the jth `month` at `station` i; $\texttt{month}_{ij}$ is the time point scaled to $[0, 1]$; $\boldsymbol{b}_i = (b_{1i}, b_{2i}, b_{3i})^T \sim \mathrm{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{D})$, where $\boldsymbol{D}$ is an unstructured covariance matrix; ϵ_{ij}'s are errors modeled by an AR(1) within-subject correlation structure with lag 1 autocorrelation coefficient ρ and

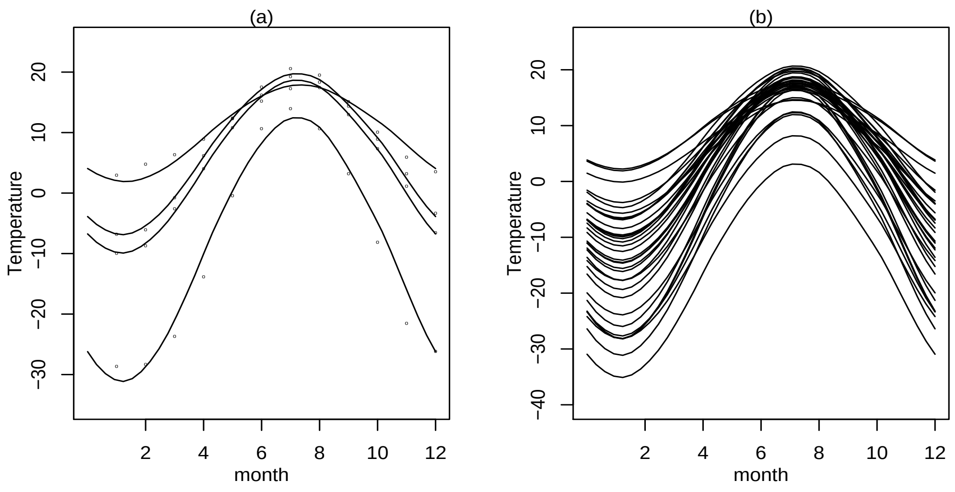


Figure 45: (a) Plot of `temperature` for four selected `stations`. With dots representing observations and lines representing SNMM fits. (b) Fitted curves after alignment for all 35 stations. Left: plot of temperatures for 4 selected stations.

variance σ^2; μ, fixed, is the annual mean temperature for all stations; b_{1i} is the vertical shift; $e^{b_{2i}}$ is the amplitude; and $e^{b_{3i}}/(1+e^{b_{3i}})$ is the horizontal shift of station i. Since the mean temperature is a periodic function with a period equal to one year, f is periodic with a period equal to 1. Thus we assume $f \in W_2(per)$. To make the constant μ and f identifiable, we need the side condition $\int_0^1 f = 0$. This is equivalent to assuming that $f \in W_2^0(per) = W_2(per) \ominus \text{span}\{1\}$, where span$\{1\}$ represents all constant functions. Since the sinusoidal form provides rough approximation to the temperature profile, we use the L-spline with $L = D^2 + (2\pi)^2$. The GML estimate of the smoothing parameter approaches to zero slowly as iteration increases, which leads to wiggly estimates. To get smoother estimates, we set a lower bound for the smoothing parameter. Specifically, $\log_{10}(n\lambda) \geq -4$. To save time, we also change the convergence criterion from the default .0005 to .005.

```
> canada.fit <- snm(temp~b1+exp(b2)*f(month-alogit(b3)),
   func=f(u)~list(~sin(2*pi*u)+cos(2*pi*u)-1,lspline(u,type=''sine0'')),
   fixed=list(b1~1), random=list(b1+b2+b3~1), cor=corAR1(),
   groups=~station, data=canadaTemp.dat,
   control=list(rkpk.control=list(limnla=c(-4,0)),prec.out=0.005))
> summary(canada.fit)
...
       AIC    BIC    logLik
  1529.772 1607.5 -745.6426

Random effects:
 Formula: list(b1 ~ 1, b2 ~ 1, b3 ~ 1)
 Level: station
 Structure: General positive-definite, Log-Cholesky parametrization
         StdDev     Corr
```

```
b1       5.94427125 b1     b2
b2       0.30031948 -0.745
b3       0.06308751 -0.013 -0.462
Residual 1.48044625

Correlation Structure: AR(1)
 Formula: ~1 | station
 Parameter estimate(s):
      Phi
0.7258356
Fixed effects: list(b1 ~ 1)
      Value Std.Error  DF  t-value p-value
b1 1.690498 0.6188728 385 2.731576  0.0066

GCV estimate(s) of smoothing parameter(s): 0.0001000000
Equivalent Degrees of Freedom (DF):  10.24364

Converged after 5 iterations
```

The resulting fits are plotted in the left panel of Figure 45 for the four selected stations. Fits for other stations are similar. The estimated correlation coefficient is $\hat{\rho} = 0.726$ with an approximate 95% confidence interval $(0.03, 1)$. Thus there is evidence that serial correlation does exist in the data. Figure 46 shows the overall fit of the common curve and its projections on subspaces $\mathcal{H}_1$ (sinusoidal part) and $\mathcal{H}_2$ (remaining part) together with their 95% Bayesian confidence intervals. The small departure from sinusoidal form is statistically significant, especially in Spring and Fall. Our conclusion here is comparable to that of Ramsay and Silverman (1997), but their approach is on individual stations and does not provide a formal test.

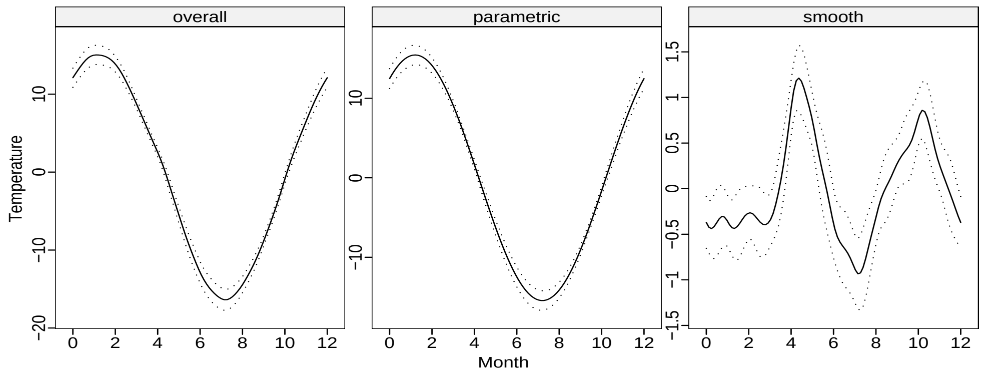


Figure 46: The overall fit of the common shape function (a), and its projection onto $\mathcal{H}_1$ (b) and $\mathcal{H}_2$ (c). Solid lines are fitted values. Dotted lines are 95% Bayesian confidence intervals.

By removing the horizontal shift in each curve, we can easily align all curves. Therefore our method can be used for curve registration.

```
grid <- NULL
for (i in 1:35) {
    grid <- c(grid,seq(0,1,len=40)+
              alogit(canada.fit$coef$random$station[70+i])+.5)
}
p.canada.2 <- predict(canada.fit, newdata=data.frame(month=grid,
                         station=rep(1:35,rep(40,35))))
```

The right panel in Figure 45 displays the fitted curves for all 35 stations after aligning them together by removing shifts.

Below is new

Ramsay and Silverman (1997) also fitted various functional linear models (FLM) to this data set. The functional data can appear in the FLM as (i) the dependent variable, (ii) the independent variable, or (iii) both. We now show that those FLM are special cases of general smoothing spline models. Therefore they can be fitted by the `ssr` function.

To investigate geographical differences in the pattern of the annual tempreture function, Ramsay and Silverman (1997) divided Canada into four meteorological zones: Atlantic, Pacific, Continental and Arctic. Then they considered the following FLM (Model (9.1) in Ramsay and Silverman (1997))

$$\texttt{temp}_{kg}(t) = \mu(t) + \alpha_g(t) + \epsilon_{kg}(t), \tag{87}$$

where $g = 1, 2, 3$ and 4 correspond to climate zones Atlantic, Pacific, Continental and Arctic respectively, $\texttt{temp}_{kg}(t)$ is the kth `temperature` function in the gth `zone`, $\mu(t)$ is the grand mean function, $\alpha_g(t)$ represent zone effect, and $\epsilon_{kg}(t)$ are random errors. Model (87) is an example of case (i).

Now consider expected temperature as a function of both `zone` g and `time` t: $f(g, t) = \mathrm{E}(\texttt{temp}_{kg}(t))$. Suppose that we want to model `zone` effect using an one-way ANOVA model and `time` effect using a periodic spline. Then we have the following SS ANOVA decomposition

$$f(g, t) = f_0 + f_1(g) + f_2(t) + f_{12}(g, t), \tag{88}$$

where f_0 is a constant, f_1 and f_2 are the main effects of `zone` and `time`, and f_{12} is the interaction between `zone` and `time`. Comparing (87) with (88), it is obvious that $\mu(t) = f_0 + f_2(t)$ and $\alpha_g(t) = f_1(g) + f_{12}(g, t)$. Therefore, model (87) can be regarded as an SS ANOVA model. Discretize the functional response by 12 months, we can fit the data as follows. Dense discretization such as by days could also be used.

```
> temps <- matrix(scan(‘‘monthtemp.dat’’,0), 35, 12, byrow=T)
> atlindex <- c(1,2,3,4,5,6,7,8,9,10,11,13,14,16)
> pacindex <- c(25,26,27,28,29)
> conindex <- c(12,15,17,18,19,20,21,22,23,24,30,31,35)
> artindex <- c(32,33,34)
> t <- seq(0.5, 11.5, 1)/12
> tgrid <- seq(0,1,len=50)
> x <- apply(temps,1,sum)
> n <- 35*12
> temp <- as.vector(t(temps[c(atlindex,pacindex,conindex,artindex),]))
> grp <- as.factor(rep(1:4,12*c(14,5,13,3)))
> tempdata <- data.frame(temp=temp,t=rep(t,35),grp=grp)

> canada4 <- ssr(temp ~ grp, spar="m",data=tempdata,
                 rk=list(periodic(t),rk.prod(periodic(t),shrink1(grp))))
> summary(canada4)
Smoothing spline regression fit by GML method
Call: ssr(formula = temp ~ grp, rk = list(periodic(t), rk.prod(periodic(t),
    shrink1(grp))), data = tempdata, spar = "m")

Coefficients (d):
(Intercept)        grp2        grp3        grp4
   4.787500    2.809167   -5.204167  -16.651389

GML estimate(s) of smoothing parameter(s) : 0.2622870 0.9971152
Equivalent Degrees of Freedom (DF):  23.74366
Estimate of sigma:  3.762337

Number of Observations:  420

> grid <- list(grp=as.factor(rep(1:4,rep(50,4))),t=rep(tgrid,4))
> p.canada4 <- predict(canada4,newdata=expand.grid(t=tgrid,grp=as.factor(1:4)),
                       terms=c(1,1,1,1,0,1))
```

Figures 47 and 48 display the estimated zone effects and the fits for four regions. They are very similar to Figures 9.1 and 9.2 in Ramsay and Silverman (1997). In addition to the estimates, we can provide confidence intervals.

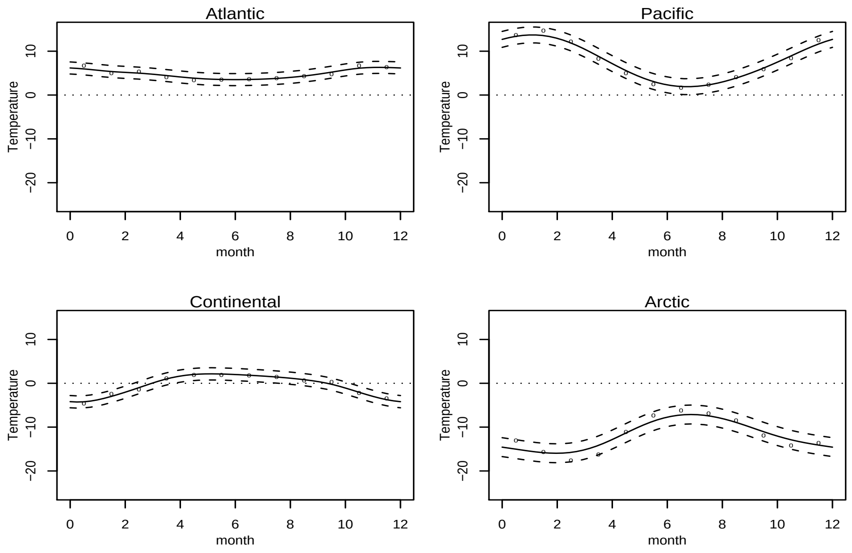


Figure 47: Solid lines are zone effects α_g for the temperature functions in model (46). Dashed lines are 95% Bayesian confidence intervals. Dotted lines are constant zero. Points are regional monthly averages minus monthly averages of the whole contry plus the average of all temperature observations.

To investigate the possible replationship between total annual precipitation and the temperature

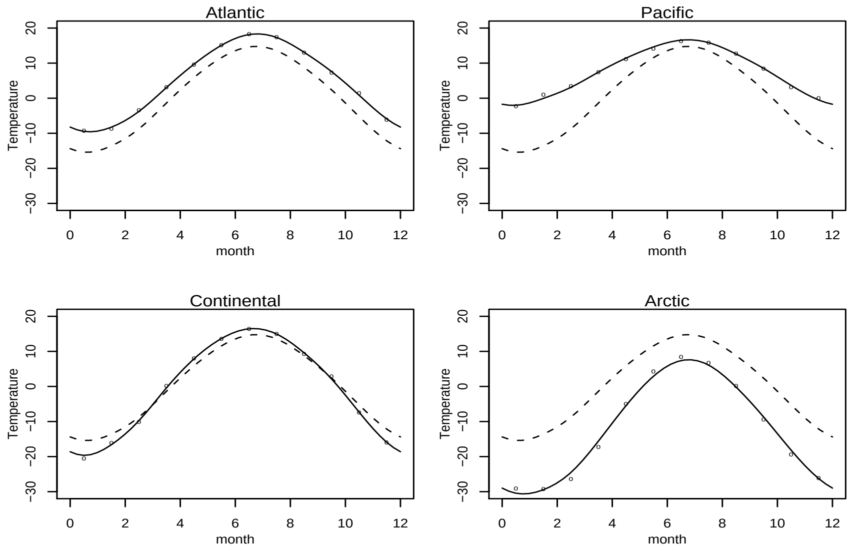


Figure 48: Solid lines are estimated region tempreture profiles. The dashed curve is the Canada mean function $\mu(t)$. Points are regional monthly averages.

function, Ramsay and Silverman (1997) considered the following FLM (Model (10.3))

$$y_i = \alpha + \int_0^T x_i(s)\beta(s)ds + \epsilon_i, \tag{89}$$

where y_i is the logarithm of total annual precipitation at station i, α is a constant parameter, $x_i(t)$ is the temperature function at station i, $\beta(t)$ is a unknown weight function, and ϵ_i are random errors. The goal is to estimate the weight function $\beta(t)$. Model (89) is an example of case (ii).

Without loss of the generality, suppose that the time interval has been transformed into $[0, 1]$. That it, $T = 1$. It is reasonable to assume that $\beta(t)$ is a smooth periodic function. Specifically, we model $\beta(t)$ using cubic periodic spline space $W_2(per)$. Define linear functionals $L_i\beta = \int_0^T x_i(s)\beta(s)ds$. L_i are bounded when $x_i \in \mathcal{L}_2$ which we assume to be true. Then model (89) is a partial spline model.

Denote $\beta(t) = \beta_0 + \beta_1(t)$ where $\beta_1 \in W_2^0(per)$. Let R be the reproducing kernel (RK) matrix for β_1. From the definition in Section 2.1.1, the ijth elements of the matrix R

$$\begin{aligned} R[i,j] &= \; < L_{i(\cdot)}R_1(t,\cdot), L_{j(\cdot)}R_1(t,\cdot) >= L_{i(\cdot)}L_{j(\cdot)}R_1(\cdot,\cdot) = \int_0^T\int_0^T x_i(s)x_j(t)R_1(s,t)dsdt \\ &\approx \sum_{k=1}^{12}\sum_{k=1}^{12} x_i(s_k)x_j(t_l)R_1(s_k,t_l) = \boldsymbol{x}_i^T\Sigma\boldsymbol{x}_j, \end{aligned} \tag{90}$$

where R_1 is the RK of $W_2^0(per)$, s_k and t_k represnet middle point of month k, $x_i(s_k)$ is the temperature of month k at station i, $\boldsymbol{x}_i = (x_i(s_1), \cdots, x_i(s_{12}))^T$, and $\Sigma = \{R_1(s_k, t_l)\}_{k,l=1}^{12}$. We used simple summations to approximate the integrals. More accurate approximations can be used. We also ignored a scaling constant (1/12) since it can be absorbed by the smoothing parameter.

```
> prec <- matrix(scan(‘‘monthprec.dat’’,0), 35, 12, byrow=T)
> y <- log(apply(prec,1,sum))
> R <- temps %*% periodic(t) %*% t(temps)
> canada5 <- ssr(y ~ x, rk=R)
> summary(canada5)
Smoothing spline regression fit by GML method
Call: ssr(formula = y ~ x, rk = R, spar = "m")

Coefficients (d):
(Intercept)           x
6.433069274 0.006122233

GML estimate(s) of smoothing parameter(s) : 0.006772994
Equivalent Degrees of Freedom (DF):  4.062615
Estimate of sigma:  0.3588121

Number of Observations:  35
```

The estimated weight function β is represented by

$$\hat{\beta}(t) = d_2 + \sum_{i=1}^{n} c_i L_{i(\cdot)} R_1(t, \cdot),$$

where d_1 is an estimate of α. To compute $\hat{\beta}(t)$ at grid points $\boldsymbol{t}_0$ of size n_0,

$$\begin{aligned}\hat{\beta}(\boldsymbol{t}_0) &= d_2 \mathbf{1}_{n_0} + \sum_{i=1}^{n} c_i \int_0^T R_1(\boldsymbol{t}_0, s) x_i(s) ds \approx d_2 \mathbf{1}_{n_0} + \sum_{i=1}^{n} c_i \sum_{j=1}^{12} R_1(\boldsymbol{t}_0, s_j) x_i(s_j) \\ &= d_2 \mathbf{1}_{n_0} + \sum_{i=1}^{n} c_i R_1(\boldsymbol{t}_0, \boldsymbol{s}) \boldsymbol{x}_i = d_2 \mathbf{1}_{n_0} + S\boldsymbol{c}\end{aligned}$$

where $\mathbf{1}_a$ is a vector of 1's of size a, $\boldsymbol{s} = (s_1, \cdots, s_{12})^T$, $S = R_1(\boldsymbol{t}_0, \boldsymbol{s}) X^T$, R_1 is the RK of $W_2^0(per)$, $R_1(\boldsymbol{t}_0, \boldsymbol{s})$ is a $n_0 \times 12$ matrix of R_1 evaluated at all combinations of $\boldsymbol{t}_0$ and $\boldsymbol{s}$, X is a $n \times 12$ matrix with each row consists monthly tempretures at each station, and $\boldsymbol{c} = (c_1, \cdots, c_n)^T$.

Bayesian confidence intervals are not available for non-evaluational functionals. Therefore we use the following bootstrap procedure to compute confidence intervals. We used simple percentile intervals. Other approaches may be used (Wang and Wahba 1995).

```
> S <- periodic(tgrid,t)%*%t(temps)
> f <- canada5$coef$d[2] + S%*%canada5$coef$c

> nboot <- 999
> fb <- NULL
> for (i in 1:nboot) {
  yb <- canada5$fit+sample(canada5$resi,35,replace=T)
  bfit <- ssr(yb ~ x, rk=R)
  fb <- cbind(fb,bfit$coef$d[2] + S%*%bfit$coef$c)
}
> lb <- apply(fb,1,quantile,prob=.025)
> ub <- apply(fb,1,quantile,prob=.975)
```

We used the default GCV method to select the smoothing parameter in the above computation. We repeated the computation with the GML method to select the smoothing parameter. Figure 49 displays the estimated regression weight functions. It is interesting to note that the estimates with GCV and GML choices of the smoothing parameter are similar to the upper left plot and the lower right plot in Figure 10.3 of Ramsay and Silverman (1997). As indicated in Figure 10.6 of Ramsay and Silverman (1997) that it is difficult to get a precise estimate of the smoothing parameter due to the small sample size. Interestingly that the GCV method picks an estimate of the smoothing parameter that is close to the lower bound and the GML method picks an estimate that is close to the upper bound. Wide confidence intervals are the consequence of the small sample size ($n = 35$).

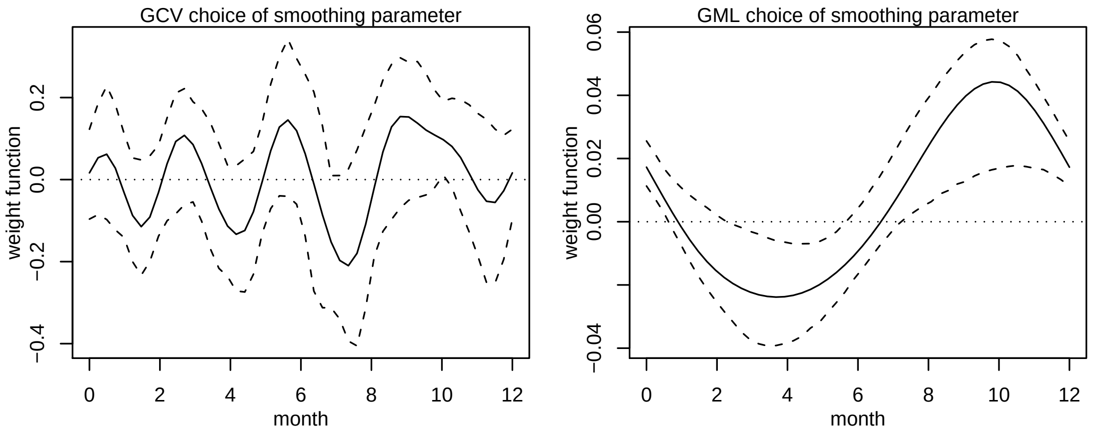


Figure 49: Estimated regression weight function β (solid lines) and its 95% bootstrap confidence intervals (dashed lines) with GCV (left) and GML (right) choices of the smoothing parameter.

To investigate the dependence of the complete log precipitation profile on the complete tempreture profile, Ramsay and Silverman (1997) considered the following FLM

$$y_i(t) = \alpha(t) + \int_0^T x_i(s)\beta(s,t)ds + \epsilon_i(t), \tag{91}$$

where $y_i(t)$ is the logarithm of annual precipitation profile at station i, $\alpha(t)$ plays the part of an intercept in the standard regression, $x_i(s)$ is the temperature function at station i, $\beta(s,t)$ is a unknown weight function at time t, and $\epsilon_i(t)$ are random errors processes. The goal is to estimate $\alpha(t)$ and $\beta(s,t)$. Model (91) is an example of case (iii).

The expected annual precipitation profile depends on two non-parametric functions: α and β. Again, we assume that $T = 1$, and $\alpha(t)$ and $\beta(s,t)$ are smooth periodic functions. Specifically, we assume that $\alpha(t) \in W_2(per) = \text{span}\{1\} \oplus W_2^0(per)$, and $\beta(s,t) \in W_2(per) \otimes W_2(per) = (\text{span}\{1\} \oplus W_2^0(per)) \otimes (\text{span}\{1\} \oplus W_2^0(per)) = \text{span}\{1\} \oplus (W_2^0(per) \otimes \text{span}\{1\}) \oplus (\text{span}\{1\} \otimes W_2^0(per)) \oplus (W_2^0(per) \otimes W_2^0(per))$. Equivalently, we use the following SS ANOVA decomposition for α and β:

$$\begin{aligned} \alpha(t) &= \alpha_0 + \alpha_1(t), \\ \beta(s,t) &= \beta_0 + \beta_1(s) + \beta_2(t) + \beta_{12}(s,t). \end{aligned} \tag{92}$$

Again, we discretize the annual precipitation profile by 12 months. Model (91) becomes

$$\begin{aligned} y_i(t_j) &= \alpha(t_j) + \int_0^T x_i(s)\beta(s,t_j)ds + \epsilon_i(t_j) \\ &= \alpha_0 + \alpha_1(t_j) + \beta_0 z_i + \int_0^T x_i(s)\beta_1(s)ds + z_i\beta_2(t_j) + \int_0^T x_i(s)\beta_{12}(s,t_j)ds + \epsilon_i(t_j), \quad (93) \\ & \quad i = 1,\cdots,n, \quad j = 1,\cdots,m, \end{aligned}$$

where $y_i(t_j)$ is the logarithm of precipitation in month j at station i, $n = 35$, $m = 12$, and $z_i = \int_0^T x_i(s)ds$. For simplicity, we assume random errors are independent. Models with correlated random errors can be fitted similarly.

Let $\boldsymbol{y} = (y_1(t_1),\cdots,y_1(t_m),\cdots,y_n(t_1),\cdots,y_n(t_m))^T$, $\boldsymbol{z} = (z_1,\cdots,z_n)^T$, $\boldsymbol{\alpha}_1 = (\alpha(t_1),\cdots,\alpha(t_m))^T$, $\boldsymbol{\beta}_1 = (\int_0^T x_1(s)\beta_1(s)ds,\cdots,\int_0^T x_n(s)\beta_1(s)ds)^T$, $\boldsymbol{\beta}_2 = (\beta_2(t_1),\cdots,\beta_2(t_m))^T$,
$\boldsymbol{\beta}_{12} = (\int_0^T x_1(s)\beta_{12}(s,t_1)ds,\cdots,\int_0^T x_1(s)\beta_{12}(s,t_m)ds,\cdots,\int_0^T x_n(s)\beta_{12}(s,t_1)ds,\cdots,\int_0^T x_n(s)\beta_{12}(s,t_m)ds)$, and $\boldsymbol{\epsilon} = (\epsilon_1(t_1),\cdots,\epsilon_1(t_m),\cdots,\epsilon_n(t_1),\cdots,\epsilon_n(t_m))^T$.

Then model (94) can be written in a matrix form

$$\boldsymbol{y} = \alpha_0\mathbf{1}_{mn} + \mathbf{1}_n \otimes \boldsymbol{\alpha}_1 + \beta_0\boldsymbol{z} \otimes \mathbf{1}_m + \boldsymbol{\beta}_1 \otimes \mathbf{1}_m + \boldsymbol{z} \otimes \boldsymbol{\beta}_2 + \boldsymbol{\beta}_{12} + \boldsymbol{\epsilon}. \quad (94)$$

We have two parametric terms, α_0 and β_0, and four non-parametric terms, α_1, β_1, β_2 and β_{12}, in model (94). To fit this model, we need to find RK matrices for four non-parametric terms. It is not difficult to see that the RK matrices for α_1, β_1 and β_2 are $\Sigma_1 = \mathbf{1}_n \otimes \mathbf{1}_n^T \otimes R_1(\boldsymbol{t},\boldsymbol{t})$, $\Sigma_2 = R \otimes \mathbf{1}_m \otimes \mathbf{1}_m^T$, $\Sigma_3 = \boldsymbol{z} \otimes \boldsymbol{z}^T \otimes R_1(\boldsymbol{t},\boldsymbol{t})$, where R_1 is the RK of $W_2^0(per)$, $\boldsymbol{t} = (t_1,\cdots,t_m)^T$, $R_1(\boldsymbol{t},\boldsymbol{t})$ represents a $m \times m$ matrix of R_1 evaluated at all combinations of (t_i,t_j), and R is a $n \times n$ matrix defined in (90). To compute the RK matrix for β_{12}, we define the linear functional $L_{ij}\beta_{12} = \int_0^T x_i(s)\beta_{12}(s,t_j)ds$ and assume that it is bounded. Since the RK of a tensor product space is the product of the two RK's, the RK of $W_2^0(per) \otimes W_2^0(per)$ is $R_1(s,u)R_1(t,v)$. Then elements of the repreducing kernel matrix for β_{12} is

$$L_{ij}L_{kl}R_1(s,u)R_1(t,v) = R_1(t_j,v_l)\int_0^T\int_0^T x_i(s)x_k(u)R_1(s,u)dsdu.$$

Thus the RK matrix for β_{12} is $\Sigma_4 = \text{rk.prod}(\Sigma_2,\Sigma_3)$, where rk.prod prepresents elementwise product of two matrices.

```
> prec <- matrix(scan(``monthprec.dat'',0), 35, 12, byrow=T)
> y <- as.vector(t(log(prec)))
> xx <- rep(x,rep(12,35))
> R1 <- kronecker(matrix(1,35,35),periodic(t))
> R2 <- kronecker(R,matrix(1,12,12))
> R3 <- kronecker(x%*%t(x),periodic(t))
> R4 <- rk.prod(R1,R2)
> canada6 <- ssr(y ~ xx, rk=list(R1,R2,R3,R4))
> summary(canada6)
Smoothing spline regression fit by GCV method
Call: ssr(formula = y ~ xx, rk = list(R1, R2, R3, R4))
```

```
Coefficients (d):
(Intercept)          xx
4.745601623 0.003035296

GCV estimate(s) of smoothing parameter(s) :
  1.727184e+03 3.630885e-02 1.416582e+07 4.597862e-04
Equivalent Degrees of Freedom (DF):  60.52553
Estimate of sigma:  0.3549322

Number of Observations:  420
```

We now show how to compute estimated functions evaluated at grid points. We stacked all observations as $\boldsymbol{y}$. Denote the corresponding months for $\boldsymbol{y}$ as $\tilde{\boldsymbol{t}} = (\boldsymbol{t}^T, \cdots, \boldsymbol{t}^T)^T$. Denote $N = mn$ as the total number of observations. Then the estimated functions are represented by

$$\begin{aligned}
\hat{\alpha}(t) &= d_1 + N\lambda_1 \sum_{i=1}^N c_i R_1(t, \tilde{t}_i), \\
\hat{\beta}_1(s) &= N\lambda_2 \sum_{i=1}^N c_i \int_0^T x_{[i/m]+1}(u) R_1(s,u)du, \\
\hat{\beta}_2(t) &= N\lambda_3 \sum_{i=1}^N c_i \left(\int_0^T x_{[i/m]+1}(u)du \right) R_1(t, \tilde{t}_i), \\
\hat{\beta}_{12}(s,t) &= N\lambda_4 \sum_{i=1}^N c_i \int_0^T x_{[i/m]+1}(u) R_1(s,u)du R_1(t, \tilde{t}_i),
\end{aligned}$$

where $[i/m]$ is the integer part of i/m. To compute $\hat{\alpha}$, $\hat{\beta}_1(s)$, $\hat{\beta}_2(t)$ at grid points $\boldsymbol{s}_0$ and $\boldsymbol{t}_0$ for s and t respectively,

$$\begin{aligned}
\hat{\alpha}(\boldsymbol{t}_0) &= d_1 \mathbf{1}_{n_0} + N\lambda_1 \sum_{i=1}^N c_i R_1(\boldsymbol{t}_0, \tilde{t}_i) = d_1 \mathbf{1}_{n_0} + N\lambda_1 S_1 \boldsymbol{c}, \\
\hat{\beta}_1(\boldsymbol{s}_0) &= N\lambda_2 \sum_{i=1}^N c_i \int_0^T x_{[i/m]+1}(u) R_1(\boldsymbol{s}_0, u)du \approx N\lambda_2 \sum_{i=1}^N c_i \sum_{j=1}^m R_1(\boldsymbol{s}_0, s_j) x_{[i/m]+1}(s_j) \\
&= N\lambda_2 \sum_{i=1}^N c_i R_1(\boldsymbol{s}_0, \boldsymbol{s}) \boldsymbol{x}_{[i/m]+1} = N\lambda_2 S_2 \boldsymbol{c}, \\
\hat{\beta}_2(\boldsymbol{t}_0) &= N\lambda_3 \sum_{i=1}^N c_i \left(\int_0^T x_{[i/m]+1}(s)ds \right) R_1(\boldsymbol{t}_0, \tilde{t}_i) = N\lambda_3 S_3 \boldsymbol{c}
\end{aligned}$$

where $S_1 = \mathbf{1}_n^T \otimes R_1(\boldsymbol{t}_0, \boldsymbol{t})$, $S_2 = S \otimes \mathbf{1}_m^T$, and $S_3 = \boldsymbol{z}^T \otimes R_1(\boldsymbol{t}_0, \boldsymbol{t})$. The interaction β_{12} is a bivariate function. Thus we evaluate it at a bivariate grid $\boldsymbol{s}_0 \otimes \boldsymbol{t}_0 = \{(s_{0k}, t_{0l}) : \ k, l = 1, \cdots, n_0\}$:

$$\hat{\beta}_{12}(s_{0k}, t_{0l}) = N\lambda_4 \sum_{i=1}^N c_i \int_0^T x_{[i/m]+1}(u) R_1(s_{0k}, u) du R_1(t_{0l}, \tilde{t}_i) \approx N\lambda_4 \sum_{i=1}^N c_i S_2[k,i] S_1[l,i] = N\lambda_4 S_4 \boldsymbol{c},$$

where S_4 is a $n_0^2 \times N$ matrix with $S_4[(k-1)n_0 + l, i] = S_2[k,i] S_1[l,i]$. We also compute bootstrap confidence intervals.

```
> S1 <- kronecker(t(rep(1,35)),periodic(tgrid,t))
> S2 <- kronecker(S,t(rep(1,12)))
> S3 <- kronecker(t(x),periodic(tgrid,t))
> S4 <- NULL
> for (i in 1:50) {for (j in 1:50) S4 <- rbind(S4,S1[i,]*S2[j,])}

> alpha <- canada6$coef$d[1] + n*canada6$lambda[1]*S1%*%canada6$coef$c
> beta0 <- canada6$coef$d[2]
> beta1 <- n*canada6$lambda[2]*S2%*%canada6$coef$c
> beta2 <- n*canada6$lambda[3]*S3%*%canada6$coef$c
> beta12 <- n*canada6$lambda[4]*S4%*%canada6$coef$c
> beta <- beta0 + rep(beta1,rep(50,50)) + rep(beta2,50) + beta12
> nboot <- 99
> alphab <- betab0 <- betab1 <- betab2 <- betab12 <- betab <- NULL
> for (i in 1:nboot) {
    yb <- canada6$fit+rnorm(n,sd=canada6$sig)
    bfit <- ssr(yb ~ xx, rk=list(R1,R2,R3,R4))
    alphab <- cbind(alphab,bfit$coef$d[1] + n*bfit$lambda[1]*S1%*%bfit$coef$c)
    bb0 <- bfit$coef$d[2]
    bb1 <- n*bfit$lambda[2]*S2%*%bfit$coef$c
    bb2 <- n*bfit$lambda[3]*S3%*%bfit$coef$c
    bb12 <- n*bfit$lambda[4]*S4%*%bfit$coef$c
    betab0 <- c(betab0,bb0)
    betab1 <- cbind(betab1,bb1)
    betab2 <- cbind(betab2,bb2)
    betab12 <- cbind(betab12,bb12)
    betab <- cbind(betab, bb0 + rep(bb1,rep(50,50)) + rep(bb2,50) + bb12)
}
> lba <- apply(alphab,1,quantile,prob=.025)
> uba <- apply(alphab,1,quantile,prob=.975)
> lbb1 <- apply(betab1,1,quantile,prob=.025)
> ubb1 <- apply(betab1,1,quantile,prob=.975)
> lbb2 <- apply(betab2,1,quantile,prob=.025)
> ubb2 <- apply(betab2,1,quantile,prob=.975)
> lbb12 <- apply(betab12,1,quantile,prob=.025)
> ubb12 <- apply(betab12,1,quantile,prob=.975)
> lbb <- apply(betab,1,quantile,prob=.025)
> ubb <- apply(betab,1,quantile,prob=.975)
```

Figure 50 displays the estimates of $\beta_{12}(s,t)$ and $\beta(s,t)$. The scale, as well as the GCV estimates of the smoothing parameters ($\lambda_4 = 4.597862e-04$), indicate that the interaction β_{12} is not small. Figures 51 and 52 display splices of the estimated interaction function β_{12} and their 95% bootstrap confidence intervals with one variable fixed approximately at the middle points of 12 months. Figures 53 and 54 display splices of the estimated function β and their 95% bootstrap confidence intervals with one variable fixed approximately at the middle points of 12 months. Figure 55 displays estimates of α, β_1 and β_2 functions.

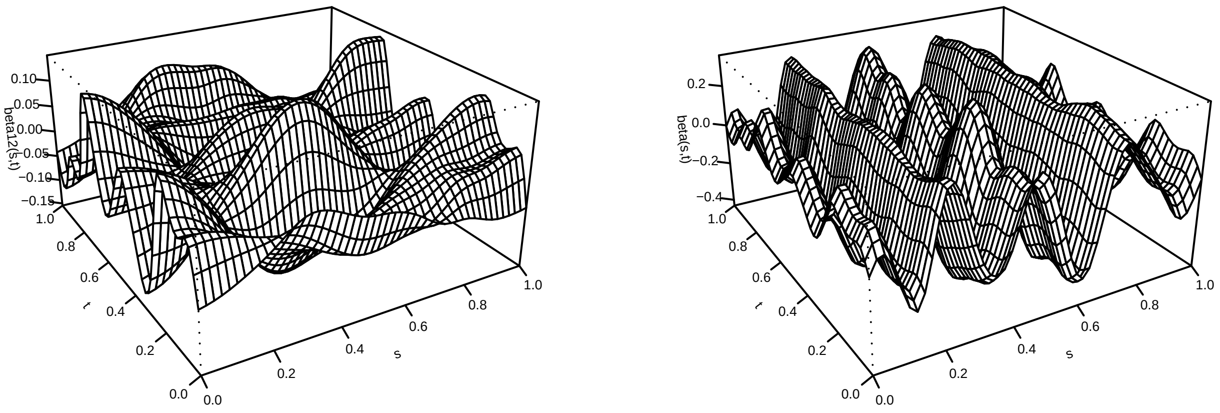


Figure 50: Estimated β_{12} (left) and β (right) functions with GCV choice of the smoothing parameter.

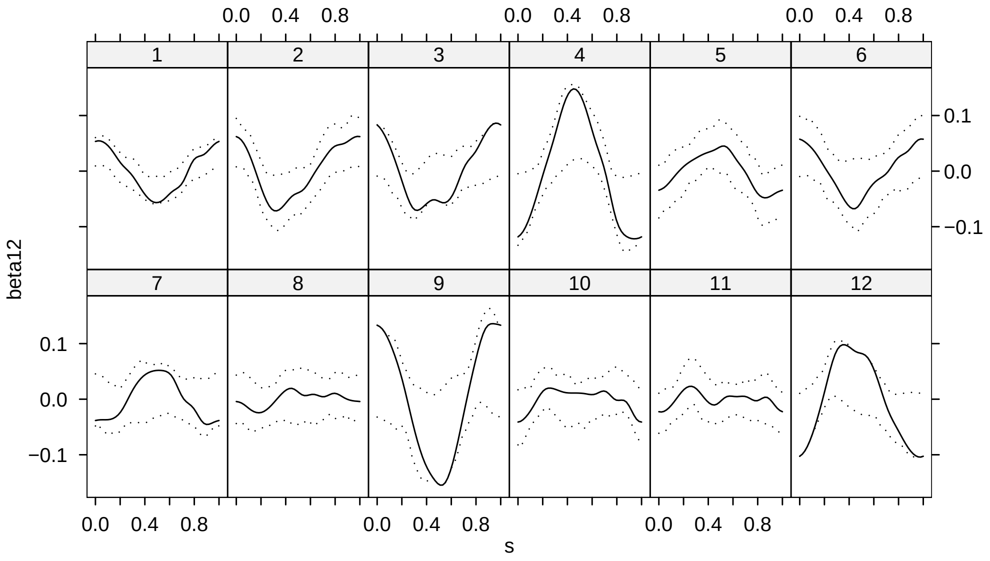


Figure 51: Estimated $\beta_{12}(s,t)$ as a function of s with t fixed approximately at the middle points of 12 months.

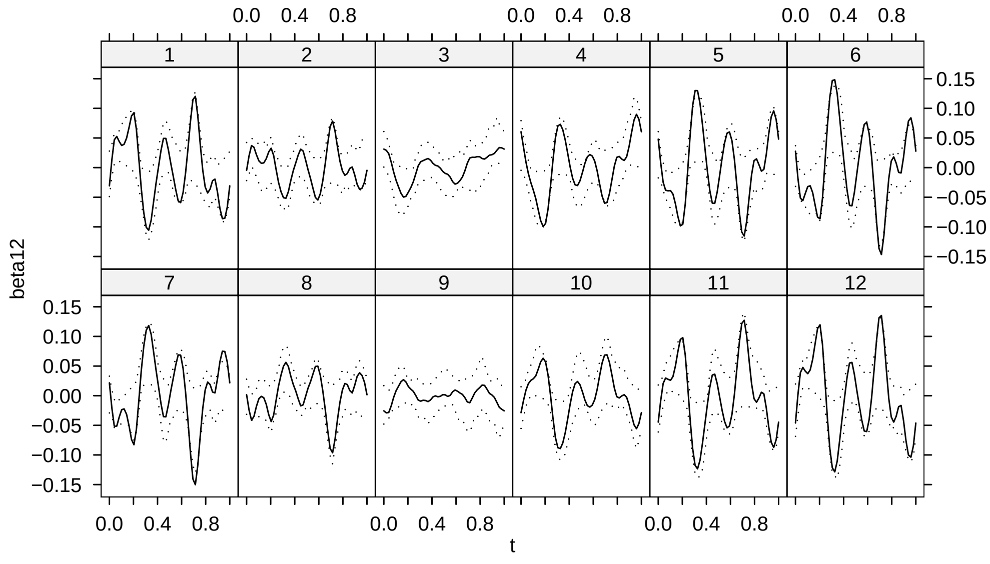


Figure 52: Estimated $\beta_{12}(s,t)$ as a function of t with s fixed approximately at the middle points of 12 months.

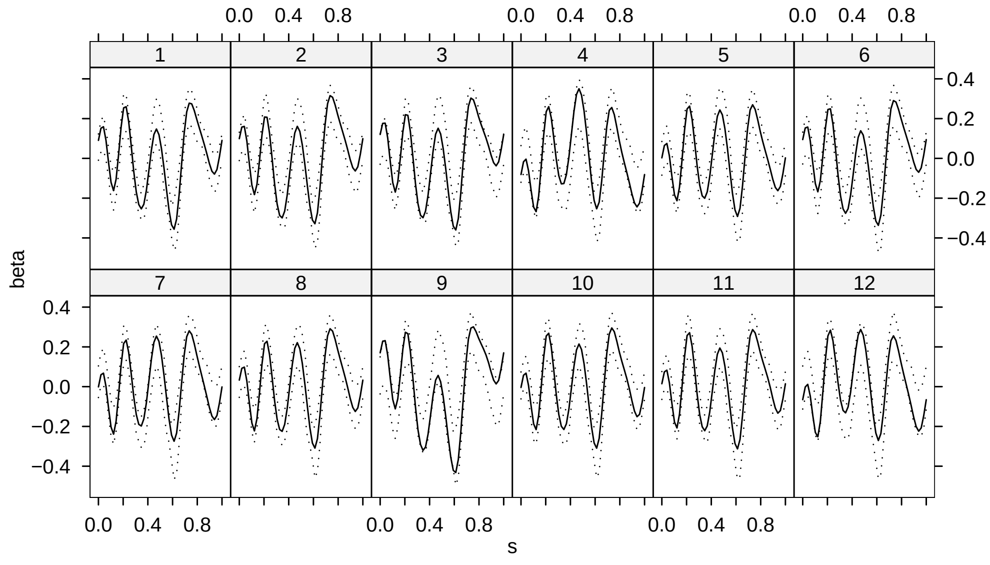


Figure 53: Estimated $\beta(s,t)$ as a function of s with t fixed approximately at the middle points of 12 months.

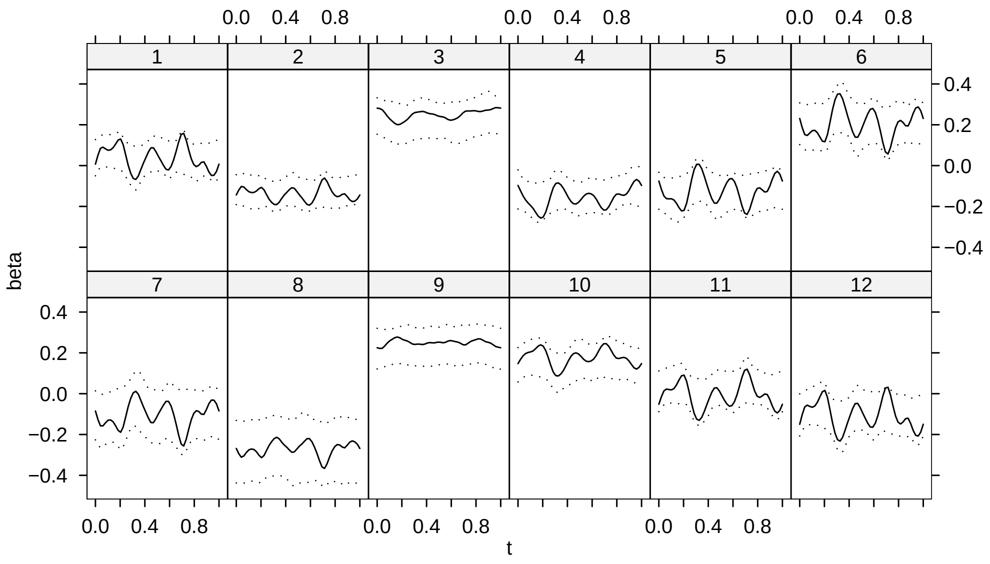


Figure 54: Estimated $\beta(s,t)$ as a function of t with s fixed approximately at the middle points of 12 months.

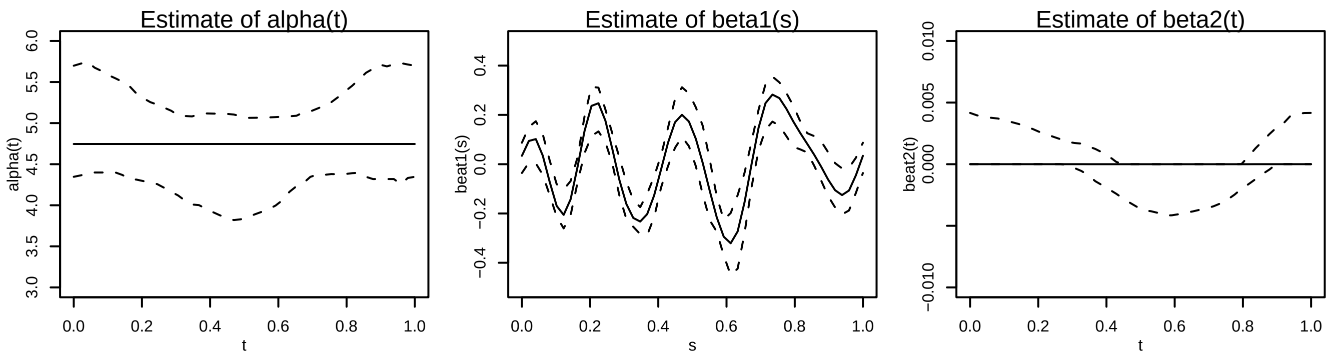


Figure 55: Estimates of $\alpha(t)$ (left), $\beta_1(s)$ (middle) and β_2 (right) functions.

We also fit the data using the GML choice of the smoothing parameter.

```
> canada6.1 <- ssr(y ~ xx, rk=list(R1,R2,R3,R4),spar="m")
> summary(canada6.1)
Smoothing spline regression fit by GML method
Call: ssr(formula = y ~ xx, rk = list(R1, R2, R3, R4), spar = "m")

Coefficients (d):
(Intercept)          xx
  4.7327087   0.0030902

GML estimate(s) of smoothing parameter(s) :
7.195469e-01 4.852421e-02 3.633548e+04 3.613019e+00
Equivalent Degrees of Freedom (DF):  27.24519
Estimate of sigma:  0.3733665

Number of Observations:  420
```

The GML estimates of smoothing parameters indicates that both the interaction β_{12} and the main effect β_2 are small. Figure 56 displays the estimates of $\beta_{12}(s,t)$ and $\beta(s,t)$. Figure 57 displays estimates of α, β_1 and β_2 functions.

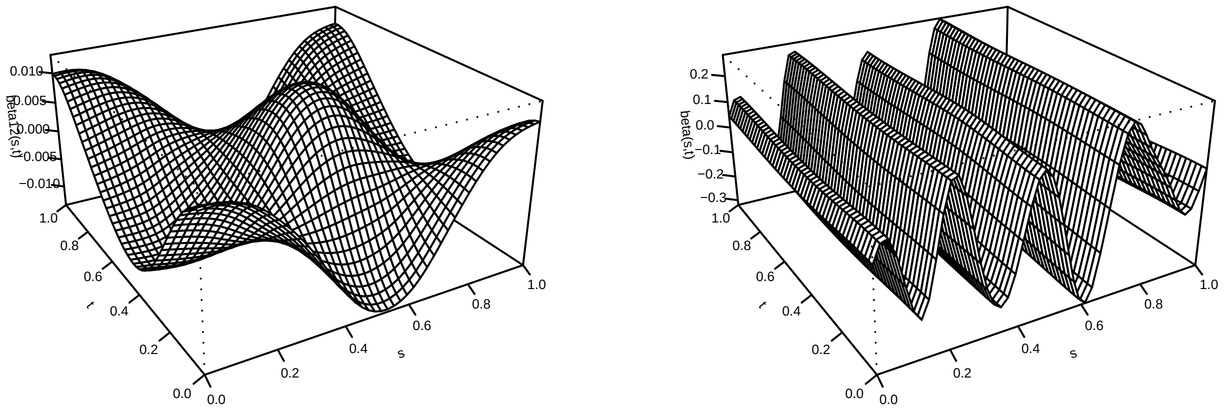


Figure 56: Estimated β_{12} (left) and β (right) functions with GML choice of the smoothing parameter.

8.12 Human Circadian Rhythms

Medical studies often collect physiological and/or psychological measurements over time from multiple subjects in order to study dynamics such as circadian rhythms and interactions between variables. Experiments are typically conducted in such a way that variables of interest are measured several times during a time period, e.g. 24 hours, from a group of normal (or ill) human subjects. The problems

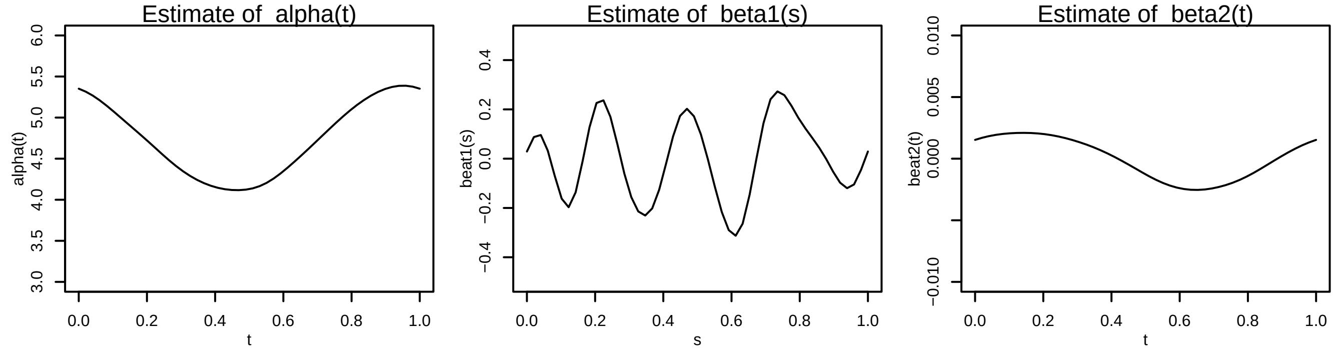


Figure 57: Estimates of $\alpha(t)$ (left), $\beta_1(s)$ (middle) and β_2 (right) functions with GML choice of the smoothing parameter.

of interest are: (1) do circadian rhythms exist? (2) do demographic, environmental and psychological variables affect circadian rhythms, and if so, how? and (3) how are variables associated with each other?

In an experiment to study immunological responses in humans, blood samples were collected every two hours for 24 hours from 9 healthy normal volunteers, 11 patients with major depression and 16 patients with Cushing's disease. These blood samples were analyzed for parameters that measure immune functions and hormones of the HPA axis (Kronfol, Nair, Zhang, Hill and Brown 1997). We will concentrate on one hormone: cortisol (`cort`).The data frame `horm.cort` contains the following variables: `ID` of subjects, `time` when measurements are taken, hormone measurements `conc`, and subject `type` (normal, depression, cushing). For simplicity, the `time` variable of 24-hour period is transformed into [0,1].

Observations are shown in Figures 59, 60 and 61. The data are noisy and it is difficult to identify patterns among subjects. The investigator hypothesizes that there is a common curve for all individuals. However, the time axis may be shifted and the magnitude of the values may differ between subjects; i.e., there may be phase and amplitude differences between subjects. Since the 24-hour periodicity is entrained, the cycle length is fixed. The common practice is to fit a sinusoidal function to each subject. Problems with this approach are: (1) the pattern over time may not be symmetric. That is, the peak and nadir may not be separated by 12 hours and/or the amplitude and width of the peak may differ from those of the nadir; (2) sometimes there are local minimum and maximum points (Wang and Brown 1996). Although adding harmonics can improve the fit, it is difficult to decide how many harmonics to include in the model and the results are difficult to interpret.

We first show how to fit a SIM to this kind of data. Consider the following model (Wang and Brown 1996)

$$\begin{aligned} y_{ij} &= \phi_{1i} + \exp(\phi_{2i}) f(t_{ij} - \exp(\phi_{3i})/(1+\exp(\phi_{3i}))) + \epsilon_{ij}, \\ & \quad i = 1, \cdots, m, \quad j = 1, \cdots, n_i, \end{aligned} \tag{95}$$

where m is the total number of subjects, n_i is the number of observations for subject i, y_{ij} is the response of ith individual at the jth time point t_{ij}, ϕ_{1i} is the 24-hour mean of the ith individual, $\exp(\phi_{2i})$ is the amplitude of the ith individual where exponential transformation is used to enforce positive constraint, $\exp(\phi_{3i})/(1+\exp(\phi_{3i}))$ is the phase of the ith individual where the inverse logistic transformation forces the phase inside the interval [0,1]. ϵ_{ij} are random errors and $\epsilon_{ij} \overset{iid}{\sim} \mathrm{N}(0, \sigma^2)$. The function f is the common curve. Since it is a periodic function, we use periodic spline to model f. In order to make f identifiable with ϕ_{1i}, we use the side condition $\int_0^1 f = 0$ which is equivalent to removing the constant from the model space. Thus we assume that $f \in W_2^0(per)$. In order to make ϕ_{2i} and ϕ_{3i} identifiable with f, we add constraints: $\phi_{21} = \phi_{31} = 0$.

We now fit model (95) to cortisol measurements from normal subjects.

```
> options(contrasts=c("contr.treatment", "contr.poly"))
> data(horm.cort)
> cort.nor <- horm.cort[horm.cort$type=="normal",]
> M <- model.matrix(~as.factor(ID), data=cort.nor)[,-1]
> cort.nor.snr.fit1 <- snr(conc~b1+exp(b2)*f(time-alogit(b3)),
    func=f(u)~list(periodic(u)),params=list(b1~as.factor(ID), b2+b3~M-1),
    start=list(params=c(mean(cort.nor$conc),rep(0,24))), data=cort.nor,
    spar="m", control=list(prec.out=0.001,converg="PRSS"))
> summary(cort.nor.snr.fit1)

Semi-parametric Nonlinear Regression Model Fit
  Model: horm ~ b1 + exp(b2) * f(time - alogit(b3))
  Data: cort.nor.dat

       AIC     BIC   logLik
  113.9554 183.449 -30.9777

Coefficients:
...

Standardized residuals:
      Min          Q1        Med        Q3      Max
 -2.11257 -0.5740086 0.04326369 0.5185883 2.521769

GML estimate(s) of smoothing spline parameter(s): 1.505551e-06
Equivalent Degrees of Freedom (DF) for spline function:  10.13926
Degrees of freedom: 107 total; 71.86074 residual
Residual standard error: 0.4213113

Converged after 5 iterations
```

Notice that we used the option *converg="PRSS"* instead of the default *converg="COEF"* because this option usually requires fewer number of iterations (the same is true for the `snm` function). We also relaxed the tolerance for convergence criterion. The potential risk of these options is that the algorithm may stop too early. The fits shown in Figure 58 seems adequate.

Random errors in model (95) may be correlated. In the following we fit with an AR(1) within-subject correlation structure.

```
> cort.nor.snr.fit2 <- update(cort.nor.snr.fit1, cor=corAR1(form=~1|ID))
> summary(cort.nor.snr.fit2)
Semi-parametric Nonlinear Regression Model Fit
```

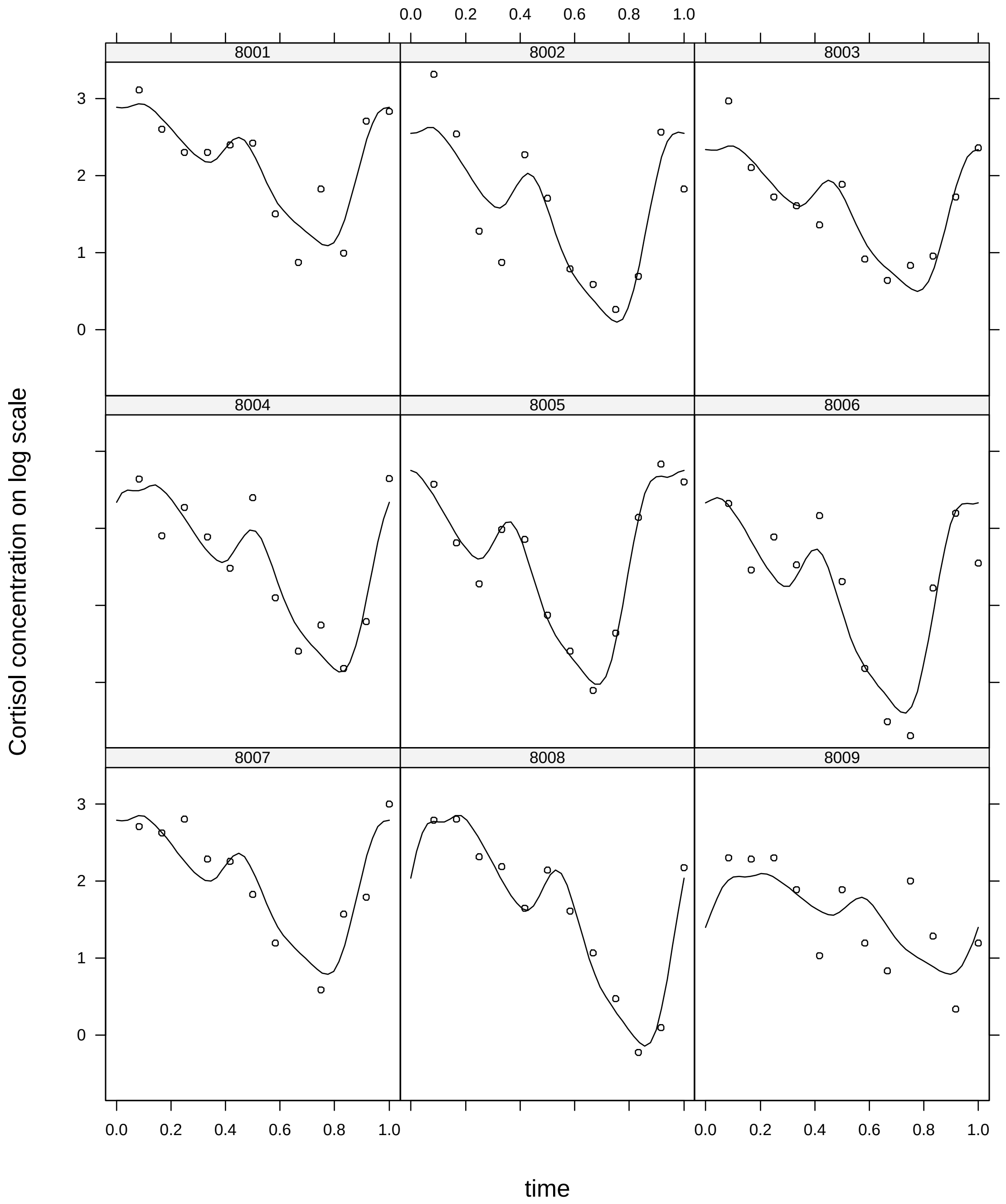


Figure 58: Plots of the data and fitted curves. Circles are observations. Solid lines represent fits from `cort.nor.snr.fit.1`. Subjects' ID are shown in the strip.

```
Model: conc ~ b1 + exp(b2) * f(time - alogit(b3))
Data: cort.nor

       AIC      BIC    logLik
  117.6520 189.8184 -31.82602

Correlation Structure: AR(1)
 Formula: ~1 | ID
 Parameter estimate(s):
       Phi
-0.1641137

Coefficients:
...

GML estimate(s) of smoothing spline parameter(s): 0.0001425255
Equivalent Degrees of Freedom (DF) for spline function:  10.33217
Residual standard error: 0.4311652

Converged after 5 iterations
```

The lag 1 autocorrelation coefficient is small.

Parameters $\boldsymbol{\phi}_{1i}$, $\boldsymbol{\phi}_{2i}$ and $\boldsymbol{\phi}_{3i}$ in model (95) are deterministic. Thus model (58) has following drawbacks: (1) they ignore correlations among observations; (2) they have the number of parameters proportional to the number of subjects; and (3) it is difficult to investigate covariate effects on parameters and/or the common curve. In the remaining of this section we show how to fit hormone data using SNM models.

We first show how to fit individual hormone for each group. Consider the following model

$$\begin{aligned} y_{ij} &= \mu + b_{1i} + \exp(b_{2i}) f(t_{ij} - \exp(b_{3i})/(1+\exp(b_{3i}))) + \epsilon_{ij}, \\ & \quad i = 1, \cdots, m, \quad j = 1, \cdots, n_i, \end{aligned} \tag{96}$$

where the fixed effect μ represents 24-hour mean of the population, the random effects b_{1i}, b_{2i} and b_{3i} represent the ith subject's deviation of 24-hour mean, amplitude and phase. We assume that $f \in W_2^0(per)$ and $\boldsymbol{b}_i = (b_{1i}, b_{2i}, b_{3i})^T \overset{iid}{\sim} N(0, \sigma^2 \boldsymbol{D})$, where $\boldsymbol{D}$ is an unstructured positive-definite matrix. The assumption of zero population mean for amplitude and phase takes care of potential confounding between amplitude, phase and the nonparametric common function f in a natural way.

We now fit model (96) to cortisol measurements from normal subjects.

```
> cort.nor.fit1 <- snm(conc~b1+exp(b2)*f(time-alogit(b3)),
    func=f(u)~list(periodic(u)), data=cort.nor, fixed=list(b1~1),
    random=list(b1+b2+b3~1), start=c(mean(cort.nor$conc)), groups=~ID,
    spar="m", control=list(prec.out=0.005,converg="PRSS"))
> summary(cort.nor.fit1)
Semi-parametric Nonlinear Mixed Effects Model fit
  Model: conc ~ b1 + exp(b2) * f(time - alogit(b3))
  Data: cort.nor
```

```
       AIC      BIC   logLik
  176.4104 224.4590 -70.2004

Random effects:
 Formula: list(b1 ~ 1, b2 ~ 1, b3 ~ 1)
 Level: ID
 Structure: General positive-definite, Log-Cholesky parametrization
         StdDev    Corr
b1       0.2475622 b1     b2
b2       0.1889174 -0.629
b3       0.2483884  0.035 -0.479
Residual 0.3948469

Fixed effects: list(b1 ~ 1)
      Value  Std.Error DF  t-value p-value
b1 1.670316 0.07686368 98 21.73089  <.0001

GML estimate(s) of smoothing parameter(s): 0.0001218930
Equivalent Degrees of Freedom (DF):  10.00480

Converged after 4 iterations

> cort.dep <- horm.cort[horm.cort$type=="depression",]
> cort.dep.fit1 <- snm(conc~b1+exp(b2)*f(time-alogit(b3)),
    func=f(u)~list(periodic(u)), data=cort.dep, fixed=list(b1~1),
    random=list(b1+b2+b3~1), start=c(mean(cort.dep$conc)), groups=~ID,
    spar="m", control=list(prec.out=0.005,converg="PRSS"))
> summary(cort.dep.fit1)

Semi-parametric Nonlinear Mixed Effects Model fit
 Model: conc ~ b1 + exp(b2) * f(time - alogit(b3))
  Data: cort.dep

       AIC      BIC    logLik
  248.2489 293.5048 -108.4047

Random effects:
 Formula: list(b1 ~ 1, b2 ~ 1, b3 ~ 1)
 Level: ID
 Structure: General positive-definite, Log-Cholesky parametrization
         StdDev    Corr
b1       0.4139559 b1     b2
b2       0.3883549 -0.815
b3       0.3806670  0.195 -0.099
Residual 0.4390506
```

```
Fixed effects: list(b1 ~ 1)
      Value  Std.Error  DF  t-value p-value
b1 1.956112 0.09046153 121 21.62369  <.0001

GML estimate(s) of smoothing parameter(s): 0.0005380155
Equivalent Degrees of Freedom (DF):  7.719702

Converged after 5 iterations

> cort.cush <- horm.cort[horm.cort$type=="cushing",]
> cort.cush.fit1 <- snm(conc~b1+exp(b2)*f(time-alogit(b3)),
    func=f(u)~list(periodic(u)), data=cort.cush, fixed=list(b1~1),
    random=list(b1+b2+b3~1), start=c(mean(cort.cush$conc)), groups=~ID,
    spar="m", control=list(prec.out=0.005,converg="PRSS"))
> summary(cort.cush.fit1)
  Model: conc ~ b1 + exp(b2) * f(time - alogit(b3))
  Data: cort.cush

        AIC       BIC   logLik
  -38.94984 -13.18616 27.47518

Random effects:
 Formula: list(b1 ~ 1, b2 ~ 1, b3 ~ 1)
 Level: ID
 Structure: General positive-definite, Log-Cholesky parametrization
         StdDev       Corr
b1       3.429996e-01 b1     b2
b2       1.028451e+04 -0.690
b3       6.520918e+03 -0.107  0.593
Residual 1.685278e-01

Fixed effects: list(b1 ~ 1)
      Value  Std.Error  DF t-value p-value
b1 3.034334 0.07610914 170 39.8682  <.0001

GML estimate(s) of smoothing parameter(s): 999.9996
Equivalent Degrees of Freedom (DF):  0.0002583048

Converged after 2 iterations
```

The fits are shown in Figures 59, 60 and 61.

We calculate the posterior means and variances of the common function f for all three `groups`:

```
u <- seq(0,1,len=50)
cort.nor.ci <- intervals(cort.nor.fit.1, newdata=data.frame(u=u))
cort.dep.ci <- intervals(cort.dep.fit.1, newdata=data.frame(u=u))
cort.cush.ci <- intervals(cort.cush.fit.1, newdata=data.frame(u=u))
```

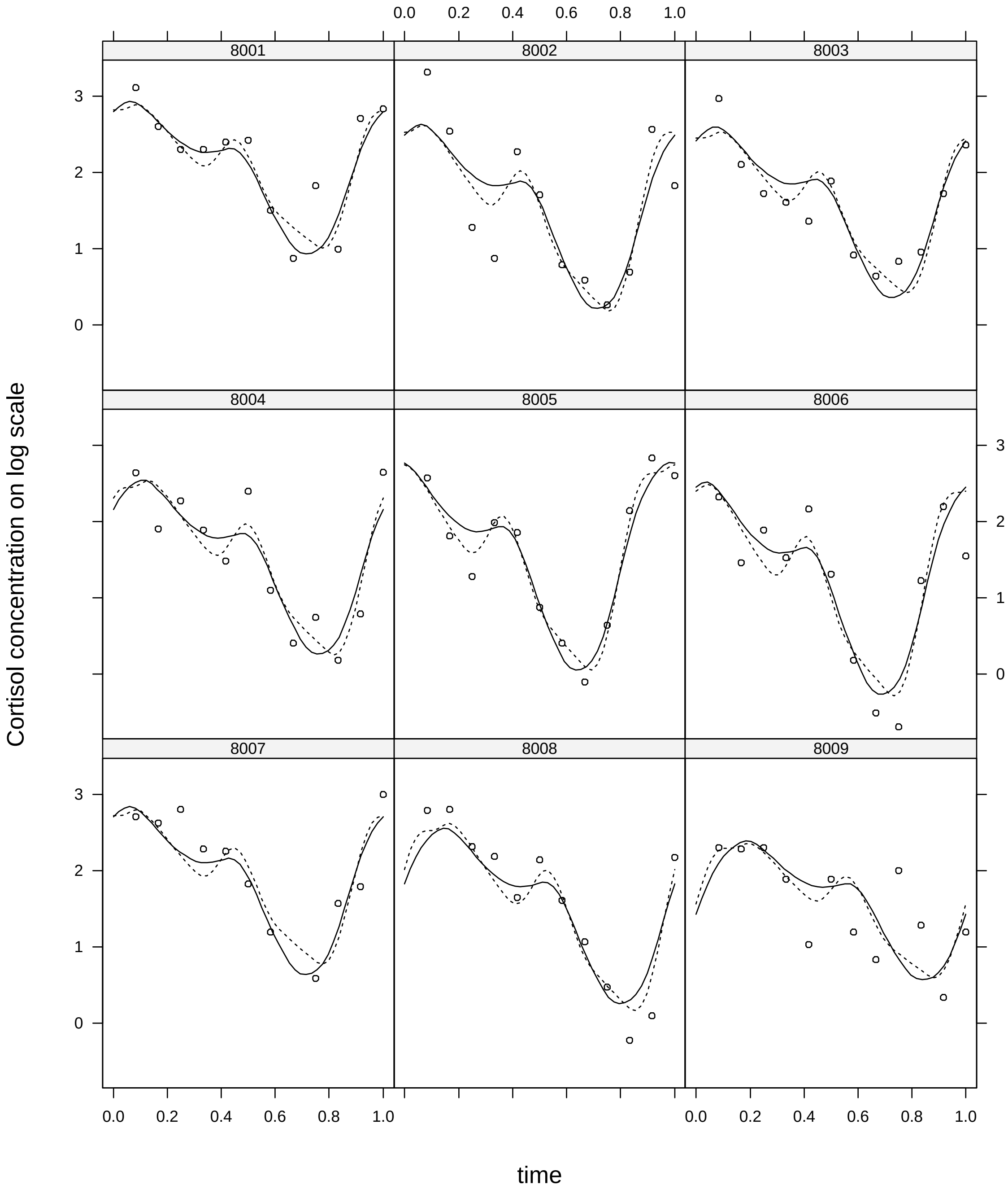


Figure 59: Plots of the data and fitted curves. Circles are observations. Solid lines represent fits from `cort.nordep.fit.3`. Dotted lines represent fits from `cort.nor.fit.1`. Subjects' ID are shown in the strip.

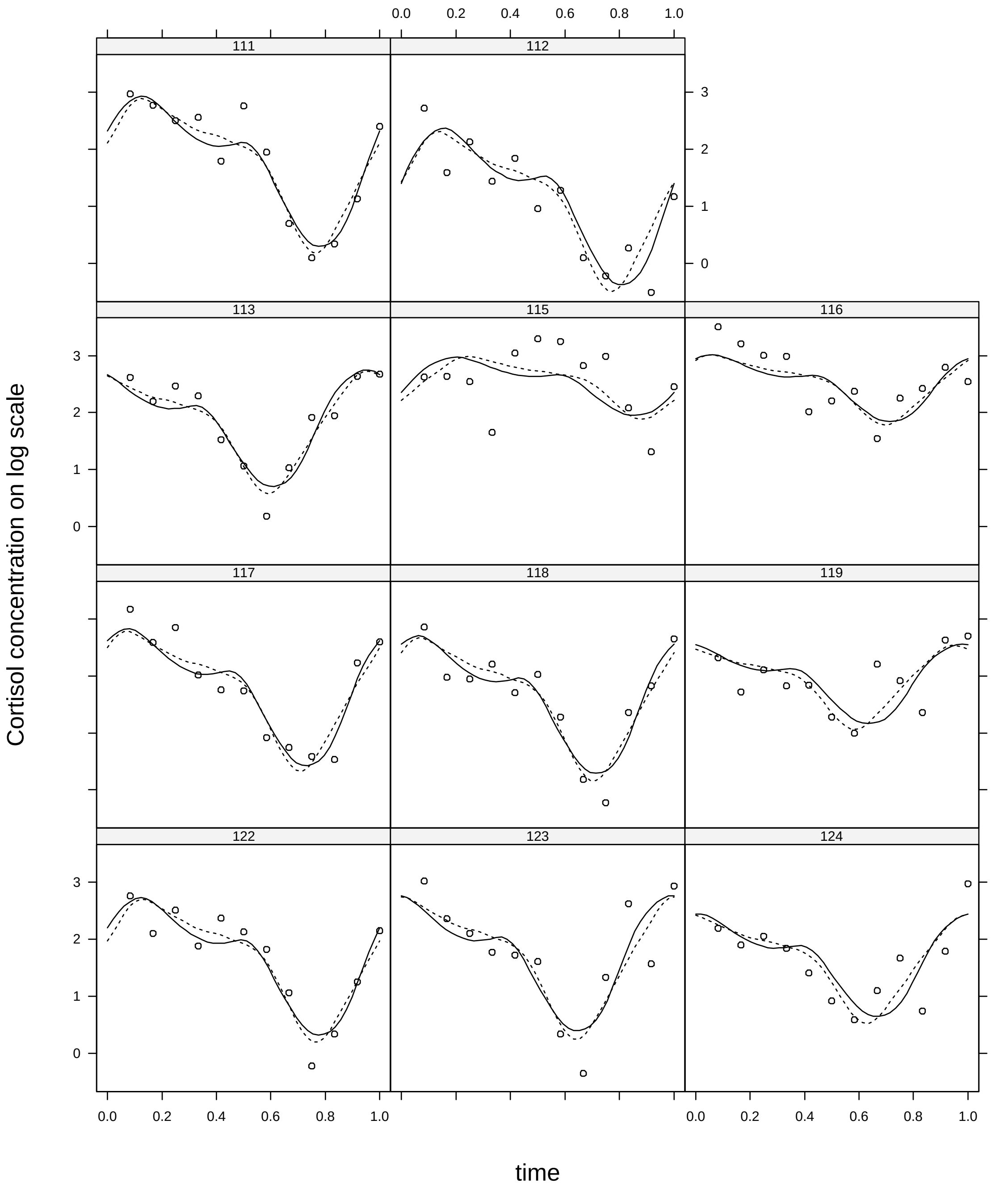


Figure 60: Plots of the data and fitted curves. Circles are observations. Solid lines represent fits from `cort.nordep.fit.3`. Dotted lines represent fits from `cort.dep.fit.1`. Subjects' ID are shown in the strip.

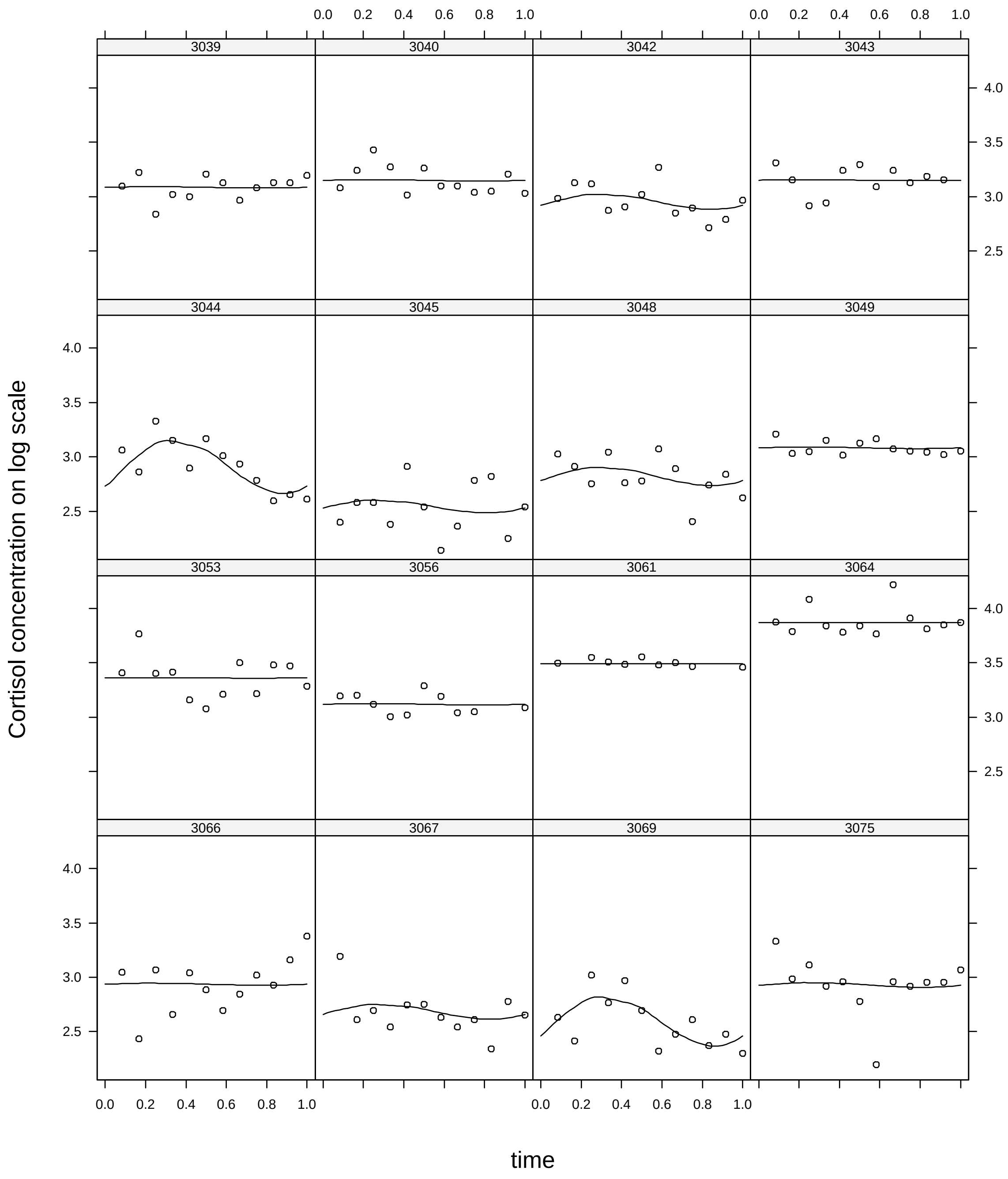


Figure 61: Plots of the data and fitted curves. Circles are observations. Solid lines represent fits from `cort.cush.fit.1`. Subjects' ID are shown in the strip.

The estimated common functions are shown in Figure 97 together with their 95% Bayesian confidence intervals.

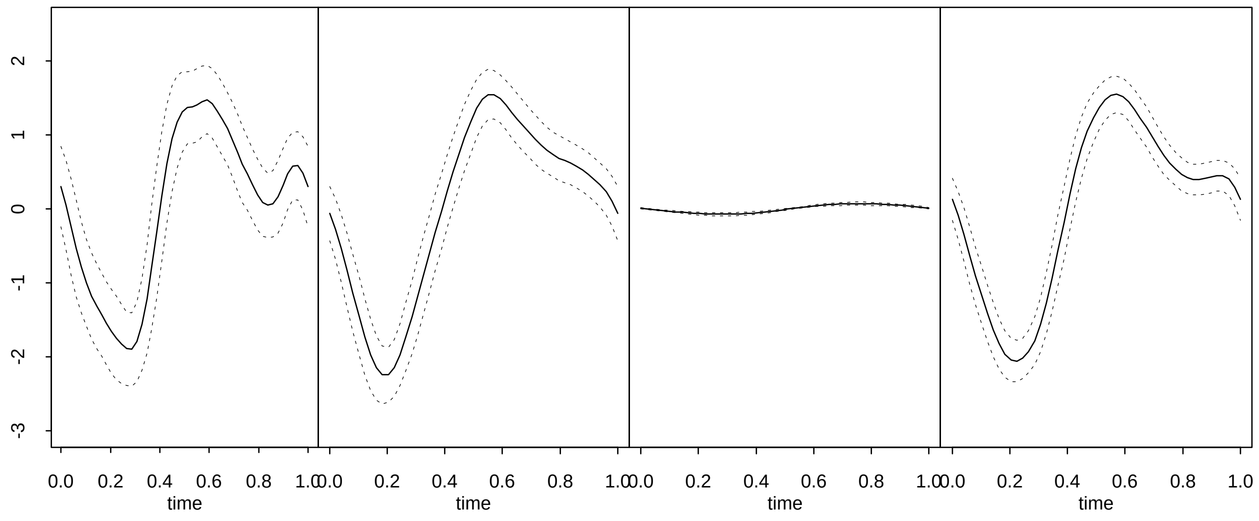


Figure 62: Solid lines are estimates of the common functions and dotted lines are 95% Bayesian confidence intervals. The left three panels are estimates from `cort.nor.fit.1`, `cort.dep.fit.1` and `cort.cush.fit.1` respectively. The right panel is the estimate from `cort.nordep.fit.3`.

It is obvious that the common function for Cushing group is almost zero, which suggests that circadian rhythms are lost for Cushing patients except subjects 3044 and 3069. It seems that the shape functions for `normal` and `depression` groups are similar. We now test this hypothesis by fitting data from these two groups simultaneous. Consider the following model

$$\begin{aligned} y_{ijk} &= \mu_k + b_{1ik} + \exp(b_{2ik}) f(k, t_{ijk} - \exp(b_{3ik})/(1+\exp(b_{3ik}))) + \epsilon_{ijk}, \\ & i = 1, \cdots, m, \quad j = 1, \cdots, n_i, \quad k = 1, 2, \end{aligned} \tag{97}$$

where k represents `group` with $k = 1$ and $k = 2$ correspond to `depression` and `normal` groups respectively, fixed effects μ_k is the population 24-hour mean of `group` k, random effects b_{1ik}, b_{2ik} and b_{3ik} represent the ith subject's deviation of 24-hour mean, amplitude and phase. Note that subjects are nested within `group`, even though this is not made explicit in out notations. We allow different correlation structures for the random effects in each `group`. That is, we assume that $\boldsymbol{b}_{ik} = (b_{1ik}, b_{2ik}, b_{3ik})^T \overset{iid}{\sim} N(0, \sigma^2 \boldsymbol{D}_k)$, where $\boldsymbol{D}_k$'s are unstructured positive-definite matrices. We assume different common functions for each `group`. Thus f is a function of both `group` (denoted as k) and `time` (denoted as t). We model `group` effect using an one-way ANOVA effect model and `time` effect using a periodic spline model without the constant term. That is, we assume that $f \in R^2 \otimes W_2^0(per)$. Writing $R^2 = \{1\} \oplus \{g : \sum_{k=1}^2 g(k) = 0\}$, we have the following SS ANOVA decomposition

$$R^2 \otimes W_2^0(per) = W_2^0(per) \oplus \{g : \sum_{k=1}^2 g(k) = 0\} \otimes W_2^0(per),$$

or equivalently,

$$f(k,t) = s(t) + ss(k,t), \tag{98}$$

where $s(t)$ is the main effect of `time`, and $ss(k,t)$ is the interaction between `group` and `time`. The hypothesis $H_0: \ f(1,t) = f(2,t)$ is equivalent to $H_0: \ ss(k,t) = 0$. We now fit model (97). Note that `pdStrat` is not available currently for the R version of `nlme`. The following results were based on the old S-Plus version.

```
> cort.nordep.dat <- horm[horm$type=="cort"&horm$group!="cushing",]
> cort.nordep.dat$group <- as.vector(cort.nordep.dat$group)
> cort.nordep.fit.1 <- snm(horm~b1+exp(b2)*f(group,time-alogit(b3)),
                        func=f(g,u)~list(list(periodic(u),
                        rk.prod(shrink1(g),periodic(u)))),
                        data=cort.nordep.dat, fixed=list(b1~group),
                        random=pdStrat(b1+b2+b3~1,strata=~as.factor(group)),
                        weights=varIdent(form=~1|group),
                        control=list(prec.out=0.005,converg="PRSS"),
                        spar="m", start=c(1.8,-.2), groups=~ID)
> summary(cort.nordep.fit.1)
Semi-parametric Nonlinear Mixed Effects Model fit
  Model: cort ~ b1 + exp(b2) * f(group, time - alogit(b3))
  Data: cort.nordep.dat

       AIC      BIC    logLik
  430.7103 516.9962 -190.4965

Random effects:
 Formula: list(b1 ~ 1, b2 ~ 1, b3 ~ 1)
 Level: ID
 Structure: General positive-definite stratified by as.factor(group)
Stratum: depression
                  StdDev   Corr
b1.(Intercept) 0.4025420 b1.(I) b2
            b2 0.3441774 -0.779
            b3 0.3352246  0.080 -0.013
Stratum: normal
                  StdDev   Corr
b1.(Intercept) 0.2526908 b1.(I) b2
            b2 0.2352516 -0.535
            b3 0.2431394 -0.050 -0.384
 Within-group standard deviation: 0.4516581

Variance function:
 Structure: Different standard deviations per stratum
 Formula:  ~ 1 | group
 Parameter estimates:
```

```
 depression    normal
          1 0.9344799
Fixed effects: list(b1 ~ group)
                    Value Std.Error  DF   t-value p-value
b1.(Intercept)  1.860474 0.0962724 218  19.32511  <.0001
      b1.group -0.160235 0.1274092 218  -1.25764  0.2099
 Correlation:
         b1.(I)
b1.group -0.756

GML estimate(s) of smoothing parameter(s): 5.037821e-04 1.487554e+02
Equivalent Degrees of Freedom (DF):  8.858698

Converged after 4 iterations

> u <- seq(0,1,len=50)
> cort.nordep.ci <- intervals(cort.nordep.fit.1, newdata=data.frame(
                 g=rep(c("normal","depression"),c(50,50)),u=rep(u,2)),
                 terms=c(0,1))
```

The smoothing parameter for the interaction term $ss(k,t)$ is large, which means that the interaction is small. In fact, $ss(k,t)$ is essentially zero: the posterior means are on the magnitude of 10^{-6} while the posterior standard deviations are on the magnitude of 10^{-4}. Therefore `normal` and `depression` groups have the same shape function.

Under the assumption of one shape function for both groups, we now can investigate differences of 24-hour mean, amplitude, and phase between two groups. For this purpose, consider the following model

$$\begin{aligned} y_{ijk} &= \mu_k + b_{1ik} + \exp(b_{2ik} + d_1 \times I(k=2)) \times \\ &\quad f(t_{ijk} - \exp(b_{3ik} + d_2 \times I(k=2))/(1+\exp(b_{3ik} + d_2 \times I(k=2)))) + \epsilon_{ijk}, \\ &\quad i = 1,\cdots,m, \quad j = 1,\cdots,n_i, \quad k = 1,2, \end{aligned} \tag{99}$$

where d_1 and d_2 measures the differences of amplitude and phase between `normal` and `depression` groups.

```
> cort.nordep.fit.2 <- snm(horm~b1+exp(b2+d1*I(group=="normal"))*
                        f(time-alogit(b3+d2*I(group=="normal"))),
                        func=f(u)~list(periodic(u)),
                        data=cort.nordep.dat,
                        fixed=list(b1~group,d1+d2~1),
                        random=pdStrat(b1+b2+b3~1,strata=~as.factor(group)),
                        weights=varIdent(form=~1|group),
                        control=list(prec.out=0.005,converg="PRSS"),
                        spar="m", start=c(1.9,-0.3,0,0), groups=~ID)
> summary(cort.nordep.fit.2)
Semi-parametric Nonlinear Mixed Effects Model fit
```

```
  Model: cort ~ b1 + exp(b2 + d1 * I(group == "normal")) *
            f(time - alogit(b3 + d2 * I(group == "normal")))
  Data: cort.nordep.dat

       AIC      BIC    logLik
  438.1875 531.3631 -192.2045

Random effects:
 Formula: list(b1 ~ 1, b2 ~ 1, b3 ~ 1)
 Level: ID
 Structure: General positive-definite stratified by as.factor(group)
Stratum: depression
                  StdDev   Corr
b1.(Intercept) 0.4072866 b1.(I) b2
            b2 0.3789050 -0.784
            b3 0.3324878  0.086 -0.046
Stratum: normal
                  StdDev   Corr
b1.(Intercept) 0.2448359 b1.(I) b2
            b2 0.1626245 -0.522
            b3 0.2661856 -0.032 -0.640
 Within-group standard deviation: 0.4515871

Variance function:
 Structure: Different standard deviations per stratum
 Formula:  ~ 1 | group
 Parameter estimates:
 depression    normal
          1 0.9358295
Fixed effects: list(b1 ~ group, d1 + d2 ~ 1)
                   Value Std.Error  DF   t-value p-value
b1.(Intercept)  1.907384 0.0956961 216  19.93168  <.0001
      b1.group -0.272381 0.1310609 216  -2.07828  0.0389
            d1  0.234996 0.0766247 216   3.06684  0.0024
            d2  0.029918 0.0915166 216   0.32691  0.7441
 Correlation:
        b1.(I) b1.typ     d1
b1.group -0.730
     d1   0.000 -0.225
     d2   0.000 -0.017 -0.428

GML estimate(s) of smoothing parameter(s): 0.0005858115
Equivalent Degrees of Freedom (DF):  8.889222

Converged after 4 iterations
```

The differences of 24-hour mean and amplitude are significant, while the difference of phase is not. We refit without the d_2 term.

```
> acth.nordep.fit.3 <- snm(horm~b1+exp(b2+d1*I(group=="normal"))*
                        f(time-alogit(b3)),
                        func=f(u)~list(periodic(u)),
                        data=acth.nordep.dat,
                        fixed=list(b1~group,d1~1),
                        random=pdStrat(b1+b2+b3~1,strata=~as.factor(group)),
                        weights=varIdent(form=~1|group),
                        control=list(prec.out=0.005,converg="PRSS"),
                        spar="m", start=c(2.8,-0.4,0), groups=~ID)
> summary(acth.nordep.fit.3)
Semi-parametric Nonlinear Mixed Effects Model fit
  Model: acth ~ b1 + exp(b2 + d1 * I(group == "normal")) *
              f(time - alogit(b3))
  Data: acth.nordep.dat

       AIC      BIC    logLik
  417.8258 518.0793 -180.2195

Random effects:
 Formula: list(b1 ~ 1, b2 ~ 1, b3 ~ 1)
 Level: ID
 Structure: General positive-definite stratified by as.factor(group)
Stratum: depression
                  StdDev   Corr
b1.(Intercept) 0.4414536 b1.(I) b2
            b2 0.3572042 -0.606
            b3 0.3117688  0.212  0.238
Stratum: normal
                  StdDev   Corr
b1.(Intercept) 0.3941497 b1.(I) b2
            b2 0.3102877 -0.532
            b3 0.1961919  0.067  0.113
 Within-group standard deviation: 0.4178081

Variance function:
 Structure: Different standard deviations per stratum
 Formula:  ~ 1 | group
 Parameter estimates:
 depression    normal
         1 0.9706348
Fixed effects: list(b1 ~ group, d1 ~ 1)
                   Value Std.Error  DF   t-value p-value
b1.(Intercept)  2.913696 0.1101288 222  26.45717  <.0001
```

```
      b1.group -0.509698 0.1732330 222  -2.94227  0.0036
            d1  0.340090 0.1283891 222   2.64890  0.0087
 Correlation:
        b1.(I)  b1.group
b1.group -0.636
     d1   0.000 -0.314

GML estimate(s) of smoothing parameter(s): 0.0002518025
Equivalent Degrees of Freedom (DF):  11.69337

Converged after 6 iterations
```

The fits are shown in Figures 59 and 60. The right panel of Figure 62 shows the estimated common function and its 95% Bayesian confidence intervals. Data from two groups are pooled to estimate the common function which has narrower confidence intervals. We conclude that the depressed subjects have their mean cortisol level elevated and less profound circadian rhythm than normal subjects.

9 Computational Concerns

There is a trade-off between generality and speed. Our goal is to develop software for general spline models. Thus all functions in `ASSIST` require at least $O(n^3)$ flops. For some special spline models such as polynomial splines, software exist which only require $O(n)$ flops. For example, the `smooth.spline` function in S should be used to fit simple cubic spline models.

Fitting complicated smoothing spline models such as SS ANOVA models with multiple smoothing parameters and SNMMs are computationally intensive. For large data sets, it may take hours and/or exhaust memory. Certain tricks in S are helpful. For example, for kernels not available in our library, it is more efficient to write functions in lower languages such as C or Fortran and then load them into S. Vectorization is also important. For large data sets, one may re-set `memory` and `object.size` arguments in the S function `options()` to increase memory and object size. Control parameters such as the number of iterations and convergence criteria can be reset to save time.

Functions using iterative procedures such as `nnr`, `snr` and `snm` may fail to converge within the prespecified number of iterations. In these cases convergence may be achieved with modifications of some of the following arguments: initial values, convergence criteria, method for estimating smoothing parameters and number of iterations.

Some reproducing kernels in `ASSIST` library have restricted domains (see Section 2). Thus it is important to check whether the domain of a `rk` function coincides with the range of a covariate. Otherwise, transformations should be made or the `scale=T` should be used. We recommend the first option.

The `nlme` library (version 3.0 or later) is required for the `ssr` and `nnr` functions when a unknown variance-covariance structure is involved, and for the `slm`, `snr` and `snm` functions. The `nlme` library in the standard libraries, if available, may be loaded using

```
    library(NLME,first=T)
```

The `nlme` library in a directory, say /home/nlme, may be loaded using

```
    dyn.load("/home/nlme/NLME_l.o")
```

We have used the same methods and notations as in `nlme` to model correlation, heteroscedasticity and random effects. Therefore users need to learn this part in `nlme` (Pinheiro and Bates 2000) when specifying the `correlation`, `weights` and `random` options.

10 Future Research

To reduce the computational burden, we will add options for selecting a subset of representers among all n representers ξ_i's in the solution (4). Inferences on nonparametric functions are usually accomplished using Bayesian confidence intervals. However, they do not provide pointwise coverage, nor a single p-value. Hypothesis tests are only available for simple spline models (Liu and Wang 2004, Liu et al. 2004). Further research on model selection is also needed. One of our future task is to extend the `anova` function to perform hypothesis tests for more complicated models, and to compare different models.

Karcher and Wang (2002) proposed the SLM models for correlated non-Gaussian data. Thus we can extend the `slm` function for non-Gaussian families.

References

Andrews, D. F. and Herzberg, A. M. (1985). *Data: A Collection of Problems From Many Fields for the Student and Research Worker*, Springer:Brln:NY.

Arendt, J., Mirors, D. S. and Waterhouse, J. M. (1989). *Biological rhythms in clinical practice*, Wright, London.

Aronszajn, N. (1950). Theory of reproducing kernels, *Trans. Amer. Math. Soc.* **68**: 337–404.

Bates, D. M., Lindstrom, M. J., Wahba, G. and Yandell, B. S. (1987). GCVPACK: Routines for generalized cross validation, *CommStB* **16**: 263–297.

Craven, P. and Wahba, G. (1979). Smoothing noisy data with spline functions, *Numer. Math.* **31**: 377–403.

Dette, H., Munk, A. and Wagner, T. (1998). Estimating the variance in nonparametric regression - what is a reasonable choice?, *Journal of the Royal Statistical Society B* **60**: 751–764.

Donoho, D. L. and Johnston, I. M. (1994). Ideal spatial adaption by wavelet shrinkage, *Biometrika* **81**: 425–456.

Earn, D. J. D., Rohani, P., Bolker, B. M. and Gernfell, B. T. (2000). A simple model for complex dynamical transitions in epidemics, *Science* **287**: 667–670.

Eubank, R. (1988). *Spline Smoothing and Nonparametric Regression*, New York: Dekker.

Gasser, T., Sroka, L. and Jennen-Steinmetz, C. (1986). Residual variance and residual pattern in nonlinear regression, *Biometrika* **73**: 625–633.

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

Grizzle, J. E. and Allen, D. M. (1969). Analysis of growth and dose response curves, *Biometrics* **25**: 357–381.

Gu, C. (1989). RKPACK and its applications: Fitting smoothing spline models, *Proceedings of the Statistical Computing Section, ASA:* pp. 42–51.

Gu, C. (1990). Adaptive spline smoothing in non-Gaussian regression models, *Journal of the American Statistical Association* **85**: 801–807.

Gu, C. (1992). Cross-validating non Gaussian data, *Journal of Computational and Graphical Statistics* **2**: 169–179.

Gu, C. (2002). *Smoothing Spline ANOVA Models*, Springer-Verlag, New York.

Gu, C. and Wahba, G. (1991). Minimizing GCV/GML scores with multiple smoothing parameters via the Newton method, *SIAM J. Sci. Stat. Comput.* **12**: 383–398.

Gu, C. and Wahba, G. (1993a). Semiparametric ANOVA with tensor product thin plate spline, *Journal of the Royal Statistical Society B* **55**: 353–368.

Gu, C. and Wahba, G. (1993b). Smoothing spline ANOVA with component-wise Bayesian confidence intervals, *Journal of Computational and Graphical Statistics* **2**: 97–117.

Hall, P., Kay, J. W. and Titterington, D. M. (1990). Asymptotically optimal difference-based estimation of variance in nonparametric regression, *Biometrika* **77**: 521–528.

Hall, P., Reimann, J. and Rice, J. (2001). Nonparametric estimation of a periodic function, *Biometrika* **87**: 545–557.

Hastie, T. and Tibshirani, R. (1990). *Generalized Additive Models*, Chapman and Hall.

Hastie, T. and Tibshirani, R. (1993). Varying coefficient model, *Journal of the Royal Statistical Society B* **55**: 757–796.

Heckman, N. and Ramsay, J. O. (2000). Penalized regression with model-based penalties, *Canadian Journal of Statistics* **28**: 241–258.

Karcher, P. and Wang, Y. (2002). Generalized nonparametric mixed effects models, *Journal of Computational and Graphical Statistics* **10**: 641–655.

Ke, C. and Wang, Y. (2001). Semi-parametric nonlinear mixed effects models and their applications (with discussion), *Journal of the American Statistical Association* **96**: 1272–1298.

Ke, C. and Wang, Y. (2002). Nonparametric nonlinear regression models, Technical Report # 385, Department of Statistics and Applied Probability, University of California - Santa Barbara.

Kitagawa, G. and Gersch, W. (1984). A smoothness priors-state space modeling of time series with trend and seasonality, *Journal of the American Statistical Association* **79**: 378–389.

Klein, R., Klein, B. E. K., Moss, S. E., Davis, M. D. and DeMets, D. L. (1988). Glycosylated hemoglobin predicts the incidence and progression of diabetic retinopathy, *Journal of the American Medical Association* **260**: 2864–2871.

Kronfol, Z., Nair, M., Zhang, Q., Hill, E. and Brown, M. (1997). Circadian immune measures in healthy volunteers: Relationship to hypothalamic-pituitary-adrenal axis hormones and sympathetic neurotransmitters, *Psychosomatic Medicine* **59**: 42–50.

Lawton, W. H., Sylvestre, E. A. and Maggio, M. S. (1972). Self-modeling nonlinear regression, *Technometrics* **13**: 513–532.

Liu, A. and Wang, Y. (2004). Hypothesis testing in smoothing spline models, *Journal of Statistical Computation and Simulation*.

Liu, A., Meiring, W. and Wang, Y. (2004). Testing generalized linear models using smoothing spline methods, *Statistica Sinica* **14**: 000–000.

Nychka, D. (1988). Bayesian confidence intervals for smoothing splines, *Journal of the American Statistical Association* **83**: 1134–1143.

Nychka, D. and Ruppert, D. (1995). A nonparametric transformation applied to both sides of a regression model, *Journal of the Royal Statistical Society B* **57**: 519–532.

Opsomer, J., Wang, Y. and Yang, Y. (2001). Nonparametric regression with correlated errors, *Statistical Science* **16**: 134–153.

O'Sullivan, F. (1990). Convergence characteristics of methods of regularization estimators for nonlinear operator equations, *SIAM Journal on Numerical Analysis* **27**: 1635–1649.

O'Sullivan, F. (1991). Sensitivity analysis for regularized estimation in some system identification problems, *SIAM J. Sci. Stat. Comput.* **12**: 1266–1283.

O'Sullivan, F. and Wahba, G. (1985). A cross validated Bayesian retrieval algorithm for non-linear remote sensing, *J. Comput. Phys.* **59**: 441–455.

Pinheiro, J. and Bates, D. M. (2000). *Mixed-effects Models in S and S-plus*, Springer, New York.

Potvin, C., Lechowicz, M. J. and Tardif, S. (1990). The statistical analysis of ecophysiological response curves obtained from experiments involving repeated measures, *Ecology* **71**: 1389–1400.

Ramsay, J. O. (1998). Estimating smooth monotone functions, *Journal of the Royal Statistical Society B* **60**: 365–375.

Ramsay, J. O. and Li, X. (1998). Curve registration, *Journal of the Royal Statistical Society B* **60**: 351–363.

Ramsay, J. O. and Silverman, B. W. (1997). *Functional Data Analysis*, Springer, New York.

Refinetti, R. (1993). Laboratory instrumentation and computing: Comparison of six methods for the determination of the period of circadian rhythms, *Physiology and Behavior* **54**: 869–875.

Rice, J. A. (1984). Bandwidth choice for nonparametric regression, *Annals of Statistics* **12**: 1215–1230.

Roosen, C. and Hastie, T. (1994). Automatic smoothing spline projection pursuit, *Journal of Computational and Graphical Statistics* **3**: 235–248.

Schaffer, W. and Kot, M. (1985). Nearly one dimensional dynamics in an epidemic, *Journal of Theoretical Biology* **112**: 403–427.

Self, S. G. and Liang, K.-Y. (1987). Asymptotic properties of maximum likelihood estimators and likelihood ratio tests under nonstandard conditions, *Journal of the American Statistical Association* **82**: 605–610.

Simonoff, J. (1996). *Smoothing Methods in Statistics*, Springer-Verlag, New York.

Venables, W. N. and Ripley, B. D. (1998). *Modern Applied Statistics With S-Plus*, Springer, New York.

Wahba, G. (1978). Improper priors, spline smoothing, and the problem of guarding against model errors in regression, *Journal of the Royal Statistical Society B* **40**: 364–372.

Wahba, G. (1981). Spline interpolation and smoothing on the sphere, *SIAM J. Sci. Stat. Comput.* **2**: 5–16.

Wahba, G. (1982). Erratum: Spline interpolation and smoothing on the sphere, *SIAM J. Sci. Stat. Comput.* **3**: 385–386.

Wahba, G. (1983). Bayesian confidence intervals for the cross-validated smoothing spline, *Journal of the Royal Statistical Society B* **45**: 133–150.

Wahba, G. (1987). Three topics in ill posed inverse problems, *Inverse and Ill-Posed Problems*, M. Engl and G. Groetsch, eds., Academic Press.

Wahba, G. (1990). *Spline Models for Observational Data*, SIAM, Philadelphia. CBMS-NSF Regional Conference Series in Applied Mathematics, Vol. 59.

Wahba, G. and Luo, Z. (1996). Smoothing spline ANOVA fits for vary large, nearly regular data sets, with application to historical global climate data, *Festschrift in Honor of Ted Rivlin*, C. Micchelli, Ed.

Wahba, G. and Wang, Y. (1993). Behavior near zero of the distribution of GCV smoothing parameter estimates for splines, *Statistics and Probability Letters* **25**: 105–111.

Wahba, G., Wang, Y., Gu, C., Klein, R. and Klein, B. (1995). Smoothing spline ANOVA for exponential families, with application to the Wisconsin Epidemiological Study of Diabetic Retinopathy, *Annals of Statistics* **23**: 1865–1895.

Wang, Y. (1997). GRKPACK: fitting smoothing spline analysis of variance models to data from exponential families, *Communications in Statistics: Simulation and Computation* **26**: 765–782.

Wang, Y. (1998a). Mixed-effects smoothing spline ANOVA, *Journal of the Royal Statistical Society B* **60**: 159–174.

Wang, Y. (1998b). Smoothing spline models with correlated random errors, *Journal of the American Statistical Association* **93**: 341–348.

Wang, Y. and Brown, M. B. (1996). A flexible model for human circadian rhythms, *Biometrics* **52**: 588–596.

Wang, Y. and Wahba, G. (1995). Bootstrap confidence intervals for smoothing splines and their comparison to Bayesian confidence intervals, *J. Statist. Comput. Simul.* **51**: 263–279.

Wang, Y. and Wahba, G. (1998). Discussion of "Smoothing Spline Models for the Analysis of Nested and Crossed Samples of Curves" by Brumback and Rice, *Journal of the American Statistical Association* **93**: 976–980.

Wang, Y., Guo, W. and Brown, M. B. (2000). Spline smoothing for bivariate data with applications to association between hormones, *Statistica Sinica* **10**: 377–397.

Wang, Y., Wahba, G., Chappell, R. and Gu, C. (1995). Simulation studies of smoothing parameter estimates and Bayesian confidence intervals in Bernoulli SS ANOVA models, *Communications in Statistics: Simulation and Computation* **24**: 1037–1059.

Wang, Y., Wahba, G., Gu, C., Klein, R. and Klein, B. (1997). Using smoothing spline ANOVA to examine the relation of risk factors to the incidence and progression of diabetic retinopathy, *Statistics in Medicine* **16**: 1357–1376.

Yorke, J. and London, W. (1973). Recurrent outbreaks of measles, chickenpox and mumps, *American Journal of Epidemiology* **98**: 453–482.