# Supplementary Material for "Functional Additive Mixed Models"

Fabian Scheipl,[*] Ana-Maria Staicu, Sonja Greven

November 18, 2013

[*]fabian.scheipl@stat.uni-muenchen.de

# Contents

# Overview

This supplement is divided into three parts. Section A gives details about suitable identifiability constraints for the additive models for functional responses and an illustrative example in Section A.2.

Section B provides more details about the simulation study in the main article: Section B.1 offers a detailed description of the data generating processes used for the simuation, Sections B.2 to Sections B.4 give the unabridged simulation results, and Section B.5 shows examples of the generated data and fitted models for the various settings. Section B.6 gives the full results for the comparison to `WFMM` (Herrick, 2013).

Section C is a fully reproducible and extended treatment of the Canadian Weather Data example.

# A Identifiability constraints for additive models for functional responses

## A.1 Deriving and imposing suitable constraints

The issue is that

$$y_{ij}(t) = g_0(t) + g(z_{ij}, t) + b_{0i}(t) + \epsilon_{ij}(t),$$

is not identifiable in the sense that

$$y_{ij}(t) = g_0(t) + \bar{g}_z(t) + (g(z_{ij}, t) - \bar{g}_z(t)) + \bar{b}_0(t) + (b_{0i}(t) - \bar{b}_0(t)) + \epsilon_{ij}(t)$$

with $\bar{b}_0(t) = (\sum_i n_i)^{-1} \sum_i n_i b_{0i}(t)$, $\bar{g}_z(t) = n^{-1} \sum_{i,j} g(z_{ij}, t)$ etc., with $n = \sum_i n_i$, yields exactly the same fit.

If we fit the model above with the constraints that $\bar{g}_z(t), \bar{b}_0(t)$ are constant zero across $\mathcal{T}$, i.e., with the constraint $(\sum_i n_i)^{-1} \sum_i n_i b_{0i}(t) = n^{-1} \sum_{i,j} g(z_{ij}, t) = 0\, \forall t$, we get a model with interpretable effects in the sense that:

- $g_0(t)$ is the (smoothed) sample mean of $Y(t)$,
- effects that vary over the index of $Y(t)$ are directly interpretable as deviations from this sample mean trajectory.

The default sum-to-zero constraints $\sum_{i,t} f(z_i, t) = 0$ implemented in `mgcv` do not yield effects that are interpretable like this, see Section A.2 below for an illustrative example.

Consider a function

$$f(\boldsymbol{z}, \boldsymbol{t}) = \begin{pmatrix} f(z_1, t_1) \\ f(z_1, t_2) \\ \vdots \\ f(z_1, t_G) \\ f(z_2, t_1) \\ \vdots \\ f(z_n, t_G) \end{pmatrix} \approx \boldsymbol{B\theta}$$

with a $n \times K = K_z K_t$ tensor product basis function matrix $\boldsymbol{B}$, where each row of $\boldsymbol{B}$ is the tensor product of the associated marginal basis functions $\boldsymbol{B}'_z|_{z=z_i} \otimes \boldsymbol{B}'_t|_{t=t_G}$, with $K_z$ marginal basis functions in $z$ and $K_t$ marginal basis functions in $t$. For this representation, the constraint above can be written as a linear constraint on the associated spline coefficients $\boldsymbol{\theta}$:

$$\boldsymbol{C}\boldsymbol{\theta} = \boldsymbol{0} \text{ with } \boldsymbol{C} = ((\underbrace{1, \dots, 1}_{1\times n}) \otimes \boldsymbol{I}_G)\boldsymbol{B}$$

(i.e.,

$$(\boldsymbol{1}_n' \otimes \boldsymbol{I}_G) = \begin{pmatrix} 1 & & & 1 & & & & 1 & \\ & \ddots & & & \ddots & & \dots & & \ddots & \\ & & 1 & & & 1 & & & & 1 \end{pmatrix}$$

does a summation of basis function values for every observed timepoint). Constraints are absorbed into a modified design matrix by post-multiplying $\boldsymbol{B}$ with the Q-factor of the QR-decomposition of $\boldsymbol{C}^\top$ (Wood, 2006, ch. 1.8.1). Since we can write $\boldsymbol{C}$ as

$$\boldsymbol{C} = (B_1^z(\bar{z}), \dots, B_{K_z}^z(\bar{z})) \otimes \begin{pmatrix} B_1^t(t_1) & \dots & B_{K_t}^t(t_1) \\ \vdots & & \vdots \\ B_1^t(t_G) & \dots & B_{K_t}^t(t_G) \end{pmatrix},$$

with $B_l^t(\bar{z}) = \sum_{i=1}^n B_l^t(z_i)$, the rank of $\boldsymbol{C}$ is $K_t$. In practice it is sufficient to enforce the constraint for $K_t$ of the timepoints spread across $t_1, \dots, t_G$ and rely on the smoothness of the function estimates to make sure that the constraint is fulfilled (approximately) in between. This allows us to avoid numerical issues that occur for the otherwise rank-deficient $\boldsymbol{C}$.

Our `pffr()` function overrides the inappropriate default constraint implemented in `mgcv`'s `gam()`-function and instead imposes the constraints given above for terms that vary smoothly over the index of the response.

## A.2 Data Example

The code for the following example is included in the accompanying R-script file `lfpr_supplement.R`.

We generate an artificial data set $Y_i(t) = \beta_0(t) + f(z_i, t) + \beta x_i + \epsilon_{it}$ with $n$ =80 observations on $G$ =60 gridpoints, with a global functional intercept $\beta_0(t)$, a smooth functional effect $f(z, t)$ (see top row in Figure 1 for the shapes) and a constant linear effect $x\beta$ with $\beta = 3$. Covariates $x_i$ and $z_i$ are standard uniform variables, $\epsilon_{it}$ are i. i. d. Gaussian with negligible variance for almost noiseless data with signal-to-noise ratio $10^5$. The code below fits an overspecified model $E(Y_i(t)) = \beta_0(t) + f(z_i, t) + f(x_i, t) + \epsilon_{it}$, once with the standard sum-to-zero constraints $\sum_{i,t} f(z_i, t) = \sum_{i,t} f(x_i, t) = 0$ (fit with a deprecated version `pffrOld` of `pffr`), and once with the modified constraints $\sum_i f(z_i, t) = \sum_i f(x_i, t) = 0 \,\forall\, t$, both with 10 marginal cubic P-spline basis functions and first order difference penalties.

```
m.OldConstr <- pffrOld(Y ~ s(z) + s(x),
        yind=t, data=d, method="REML",
        bsy.default = list(bs="ps", m=c(2, 1), k=10))
m <- pffr(Y ~ s(z) + s(x),
        yind=t, data=d, method="REML",
        bs.int = list(bs="ps", m=c(2, 1), k=10),
        bs.yind= list(bs="ps", m=c(2, 1), k=10))
```

The fits for the two models are almost equivalent, the correlation between their fitted values is about 0.996. Figure 1, however, shows that the model without the appropriate centering constraints completely misses the true shape of the effects, that the functional intercept in the model with sum-to-zero-$\forall$-t constraints is directly interpretable as the mean trajectory in the data, and that estimation uncertainty is reduced by using the appropriate constraints (c.f. the widths of the confidence intervals around $\hat{\beta}_0(t)$).

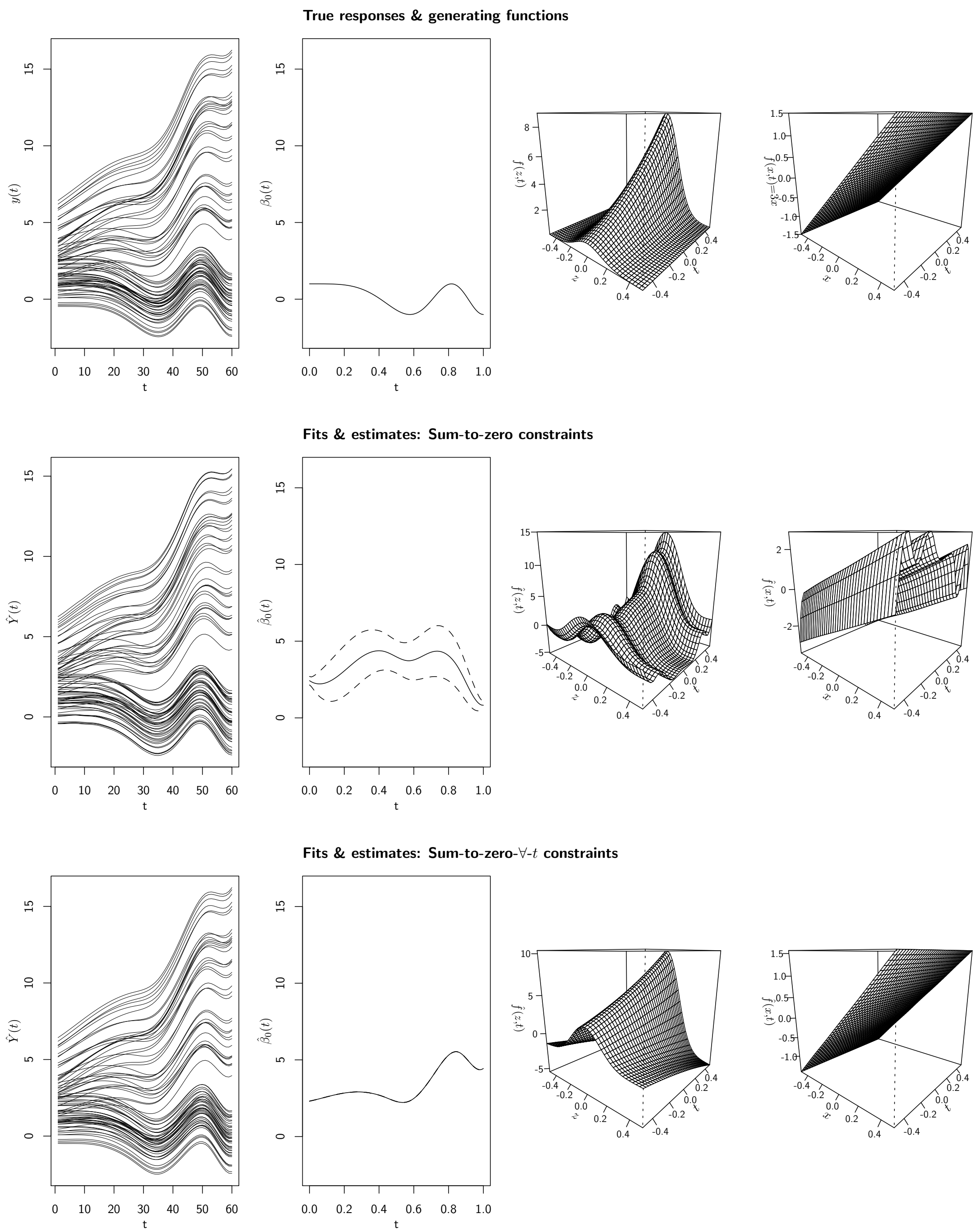


**Figure 1:** True (top row) and estimated effects for the data described in section A.2. Middle row shows effects estimated with the default sum-to-zero constraints, while the bottom row shows effects estimated under the sum-to-zero-∀-t constraints.

# B Simulation study

## B.1 Data generating process

This subsection describes in detail the data generating process used for the simulation study in the article. We use

$$\alpha(t) = t^2 - \sqrt{t} + \phi(t, 0.2, 0.1) + \phi(t, 0.6, 0.4) - f_B(t, 7, 4) + f_B(t, 9, 9),$$

where $\phi(\cdot, \mu, \sigma)$ is a $N(\mu, \sigma^2)$-density and $f_B(\cdot, a, b)$ is a Beta$(a, b)$-density,

$$\beta_1(s,t) = s\cos(\pi|s-t|) - 0.19$$
$$\beta_2(s,t) = \cos(\pi s)\sin(\pi t) + (st)^2 - 0.11,$$

Functional random effects $b_{i0}(t), b_{i1}(t)$ and functional covariates $x_{1i}(s), x_{2i}(s)$ are generated from a cubic B-spline basis with 5 basis functions whose coefficients are i. i. d. $N(0,1)$ and centered so that $\sum_i b_i(t) = \sum_i x_i(t) = 0 \,\forall\, t$. For the scalar covariates, $z_{1,ij} \overset{\text{i.i.d.}}{\sim} U[-0.5, 0.5]$ and $z_{2,ij} \overset{\text{i.i.d.}}{\sim} U[0,1]$. Residuals $\varepsilon_{ij}(t)$ are i. i. d. Gaussian, with variance $\sigma_\varepsilon^2$ determined by the signal-to-noise ratio $\text{SNR}_\varepsilon$ of the setting.

## B.2 Relative integrated mean square errors

The following graphs display the relative integrated mean squared error $\text{rIMSE}(\hat{f}(t)) = \frac{\int(\hat{f}(t)-f(t))^2dt}{\int f(t)^2dt}$ for the various scenarios and settings. Tables give estimated coefficients for main effect models for $\log_2(\text{rIMSE})$ with the various settings under the different scenarios. Note that we are evaluating the estimation accuracy of the effects on the scale of the response, not on the scale of the coefficient function itself, e.g. for the effect of a functional covariate $x(s)$ we consider the error $\int_{\mathcal{T}}\int_{\mathcal{S}}(x(s)\hat{\beta}(s,t) - x(s)\beta(s,t))^2dsdt$ (and not $\int_{\mathcal{T}}\int_{\mathcal{S}}(\hat{\beta}(s,t) - \beta(s,t))^2dsdt$).

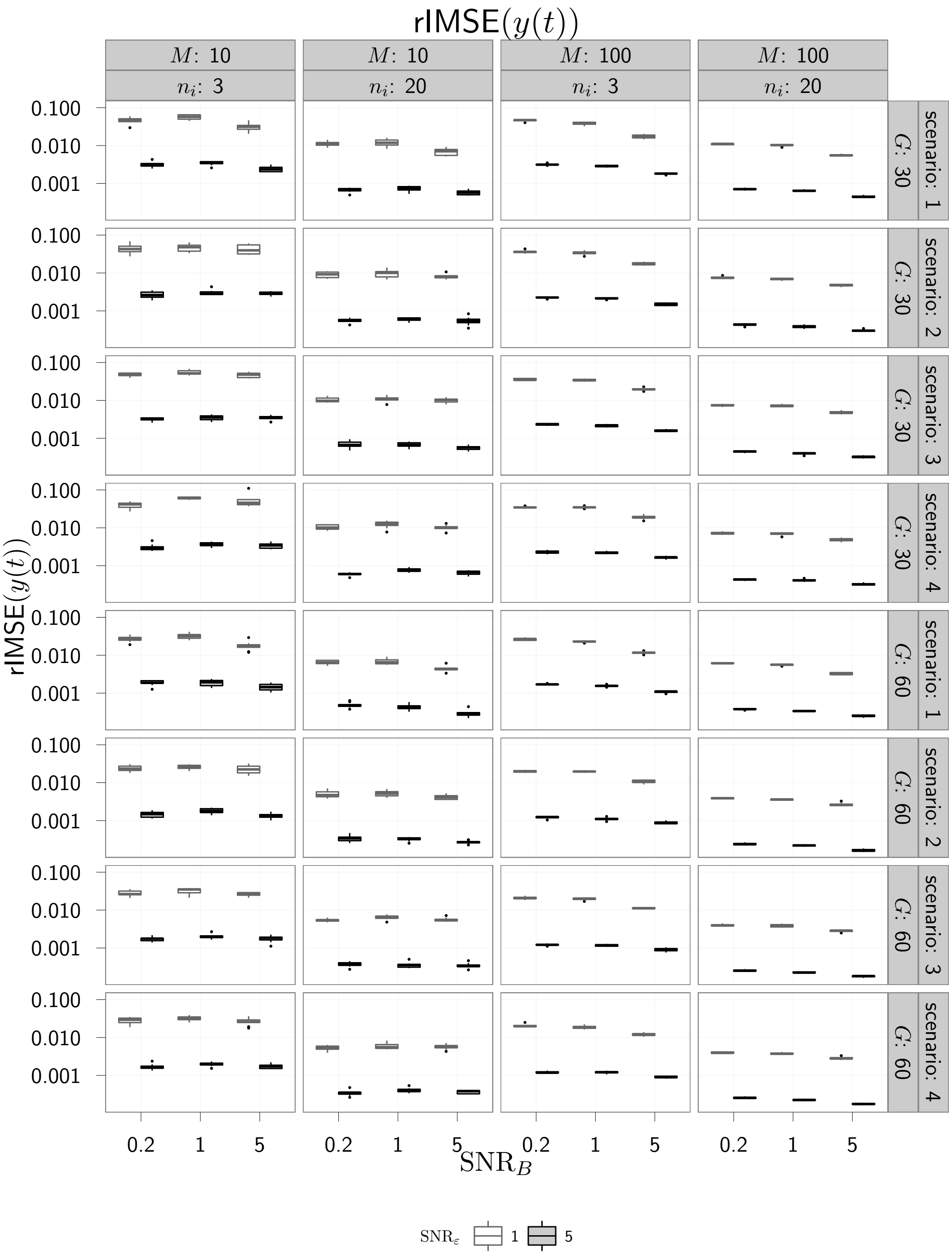


**Figure 2:** rIMSE for $\hat{y}(t)$ for all combinations of the various settings.

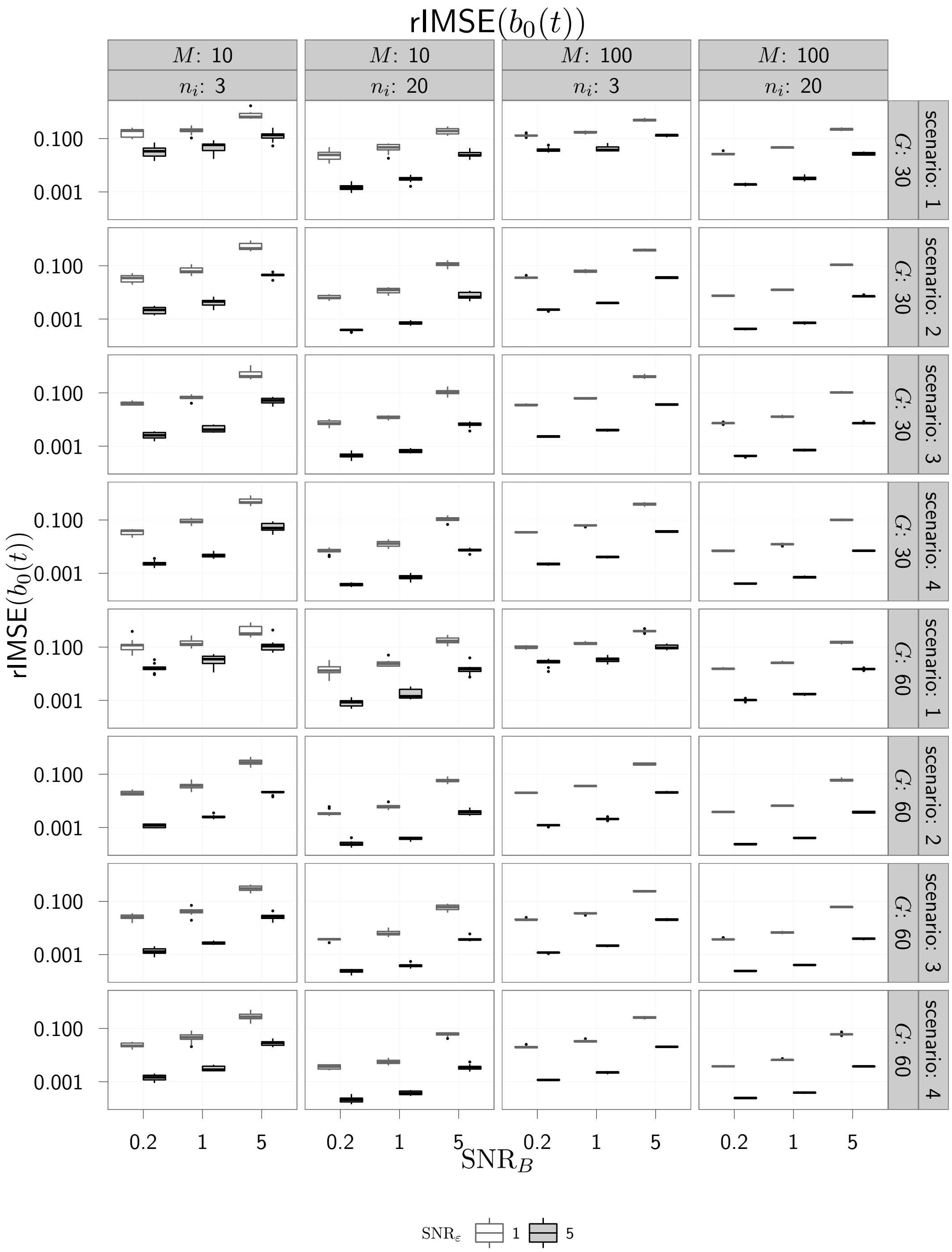


**Figure 3:** rIMSE for $\hat{b}_0(t)$ for all combinations of the various settings.

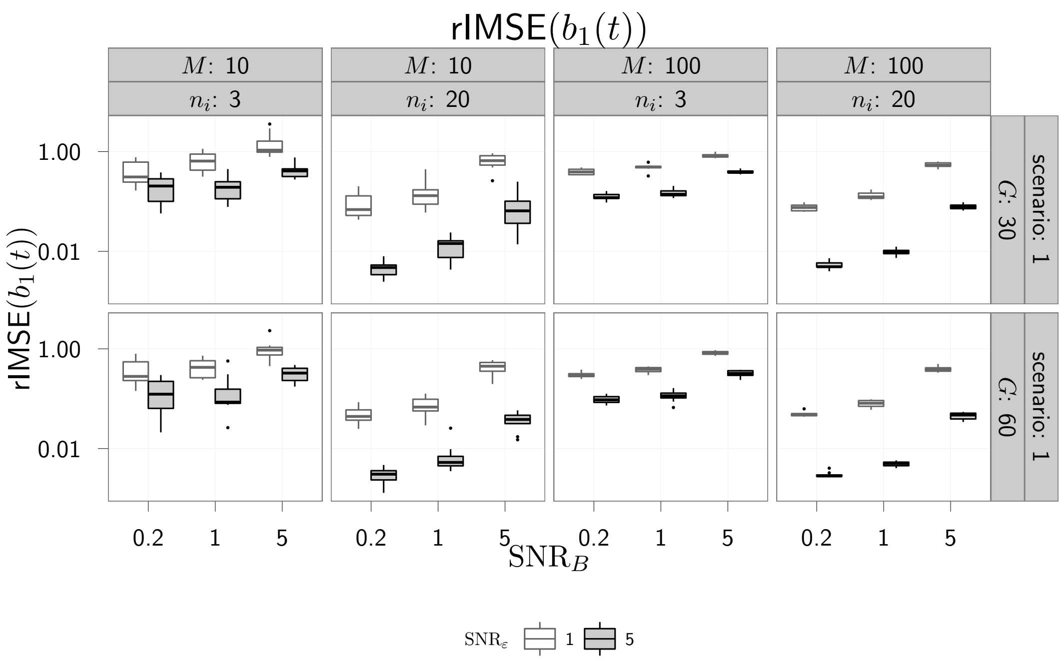


**Figure 4:** rIMSE for $\hat{b}_1(t)$ for all combinations of the various settings.

| | scenario | setting | high | low | estimate |
|---|---|---|---|---|---|
| rIMSE($y(t)$) | 1 | baseline | 0.05 | 0.05 | 0.05 |
| rIMSE($y(t)$) | 2 | baseline | 0.05 | 0.05 | 0.05 |
| rIMSE($y(t)$) | 3 | baseline | 0.06 | 0.05 | 0.05 |
| rIMSE($y(t)$) | 4 | baseline | 0.05 | 0.05 | 0.05 |
| rIMSE($y(t)$) | 1 | $M : 10 \to 100$ | 0.85 | 0.80 | 0.83 |
| rIMSE($y(t)$) | 2 | $M : 10 \to 100$ | 0.70 | 0.66 | 0.68 |
| rIMSE($y(t)$) | 3 | $M : 10 \to 100$ | 0.62 | 0.58 | 0.60 |
| rIMSE($y(t)$) | 4 | $M : 10 \to 100$ | 0.61 | 0.57 | 0.59 |
| rIMSE($y(t)$) | 1 | $n_i : 3 \to 20$ | 0.24 | 0.23 | 0.23 |
| rIMSE($y(t)$) | 2 | $n_i : 3 \to 20$ | 0.21 | 0.20 | 0.20 |
| rIMSE($y(t)$) | 3 | $n_i : 3 \to 20$ | 0.21 | 0.20 | 0.20 |
| rIMSE($y(t)$) | 4 | $n_i : 3 \to 20$ | 0.21 | 0.20 | 0.21 |
| rIMSE($y(t)$) | 1 | $G : 30 \to 60$ | 0.59 | 0.56 | 0.58 |
| rIMSE($y(t)$) | 2 | $G : 30 \to 60$ | 0.56 | 0.53 | 0.54 |
| rIMSE($y(t)$) | 3 | $G : 30 \to 60$ | 0.57 | 0.54 | 0.56 |
| rIMSE($y(t)$) | 4 | $G : 30 \to 60$ | 0.57 | 0.54 | 0.55 |
| rIMSE($y(t)$) | 1 | $\mathrm{SNR}_B : 0.2 \to 1$ | 1.02 | 0.95 | 0.98 |
| rIMSE($y(t)$) | 2 | $\mathrm{SNR}_B : 0.2 \to 1$ | 1.04 | 0.96 | 1.00 |
| rIMSE($y(t)$) | 3 | $\mathrm{SNR}_B : 0.2 \to 1$ | 1.05 | 0.98 | 1.01 |
| rIMSE($y(t)$) | 4 | $\mathrm{SNR}_B : 0.2 \to 1$ | 1.12 | 1.04 | 1.08 |
| rIMSE($y(t)$) | 1 | $\mathrm{SNR}_B : 0.2 \to 5$ | 0.63 | 0.59 | 0.61 |
| rIMSE($y(t)$) | 2 | $\mathrm{SNR}_B : 0.2 \to 5$ | 0.79 | 0.74 | 0.76 |
| rIMSE($y(t)$) | 3 | $\mathrm{SNR}_B : 0.2 \to 5$ | 0.83 | 0.77 | 0.80 |
| rIMSE($y(t)$) | 4 | $\mathrm{SNR}_B : 0.2 \to 5$ | 0.88 | 0.82 | 0.85 |
| rIMSE($y(t)$) | 1 | $\mathrm{SNR}_\varepsilon : 1 \to 5$ | 0.07 | 0.07 | 0.07 |
| rIMSE($y(t)$) | 2 | $\mathrm{SNR}_\varepsilon : 1 \to 5$ | 0.07 | 0.06 | 0.06 |
| rIMSE($y(t)$) | 3 | $\mathrm{SNR}_\varepsilon : 1 \to 5$ | 0.07 | 0.06 | 0.06 |
| rIMSE($y(t)$) | 4 | $\mathrm{SNR}_\varepsilon : 1 \to 5$ | 0.07 | 0.06 | 0.06 |

**Table 1:** Exponeniated coefficient estimates (i.e., multiplication factors) for the $\log_2$-linear model for rIMSE($y(t)$). "High" and "low" estimates are estimated coefficients $\pm$ 2 standard deviations, exponentiated.

| | scenario | setting | high | low | estimate |
|---|---|---|---|---|---|
| rIMSE($b_0(t)$) | 1 | baseline | 0.16 | 0.14 | 0.15 |
| rIMSE($b_0(t)$) | 2 | baseline | 0.04 | 0.03 | 0.03 |
| rIMSE($b_0(t)$) | 3 | baseline | 0.04 | 0.04 | 0.04 |
| rIMSE($b_0(t)$) | 4 | baseline | 0.04 | 0.03 | 0.04 |
| rIMSE($b_0(t)$) | 1 | $M : 10 \to 100$ | 1.09 | 0.98 | 1.03 |
| rIMSE($b_0(t)$) | 2 | $M : 10 \to 100$ | 1.04 | 0.94 | 0.99 |
| rIMSE($b_0(t)$) | 3 | $M : 10 \to 100$ | 0.99 | 0.89 | 0.94 |
| rIMSE($b_0(t)$) | 4 | $M : 10 \to 100$ | 0.97 | 0.87 | 0.92 |
| rIMSE($b_0(t)$) | 1 | $n_i : 3 \to 20$ | 0.14 | 0.13 | 0.13 |
| rIMSE($b_0(t)$) | 2 | $n_i : 3 \to 20$ | 0.20 | 0.18 | 0.19 |
| rIMSE($b_0(t)$) | 3 | $n_i : 3 \to 20$ | 0.19 | 0.17 | 0.18 |
| rIMSE($b_0(t)$) | 4 | $n_i : 3 \to 20$ | 0.19 | 0.17 | 0.18 |
| rIMSE($b_0(t)$) | 1 | $G : 30 \to 60$ | 0.69 | 0.61 | 0.65 |
| rIMSE($b_0(t)$) | 2 | $G : 30 \to 60$ | 0.58 | 0.52 | 0.55 |
| rIMSE($b_0(t)$) | 3 | $G : 30 \to 60$ | 0.60 | 0.53 | 0.56 |
| rIMSE($b_0(t)$) | 4 | $G : 30 \to 60$ | 0.59 | 0.53 | 0.56 |
| rIMSE($b_0(t)$) | 1 | $\text{SNR}_B : 0.2 \to 1$ | 1.67 | 1.46 | 1.56 |
| rIMSE($b_0(t)$) | 2 | $\text{SNR}_B : 0.2 \to 1$ | 1.92 | 1.67 | 1.79 |
| rIMSE($b_0(t)$) | 3 | $\text{SNR}_B : 0.2 \to 1$ | 1.83 | 1.60 | 1.71 |
| rIMSE($b_0(t)$) | 4 | $\text{SNR}_B : 0.2 \to 1$ | 1.98 | 1.73 | 1.85 |
| rIMSE($b_0(t)$) | 1 | $\text{SNR}_B : 0.2 \to 5$ | 7.67 | 6.69 | 7.16 |
| rIMSE($b_0(t)$) | 2 | $\text{SNR}_B : 0.2 \to 5$ | 16.92 | 14.78 | 15.81 |
| rIMSE($b_0(t)$) | 3 | $\text{SNR}_B : 0.2 \to 5$ | 16.16 | 14.11 | 15.10 |
| rIMSE($b_0(t)$) | 4 | $\text{SNR}_B : 0.2 \to 5$ | 17.04 | 14.88 | 15.92 |
| rIMSE($b_0(t)$) | 1 | $\text{SNR}_\varepsilon : 1 \to 5$ | 0.14 | 0.13 | 0.14 |
| rIMSE($b_0(t)$) | 2 | $\text{SNR}_\varepsilon : 1 \to 5$ | 0.07 | 0.06 | 0.07 |
| rIMSE($b_0(t)$) | 3 | $\text{SNR}_\varepsilon : 1 \to 5$ | 0.07 | 0.06 | 0.07 |
| rIMSE($b_0(t)$) | 4 | $\text{SNR}_\varepsilon : 1 \to 5$ | 0.07 | 0.06 | 0.07 |

**Table 2:** Exponeniated coefficient estimates (i.e., multiplication factors) for the $\log_2$-linear model for rIMSE($b_0(t)$). "High" and "low" estimates are estimated coefficients $\pm$ 2 standard deviations, exponentiated.

| | scenario | setting | high | low | estimate |
|---|---|---|---|---|---|
| rIMSE($b_1(t)$) | 1 | baseline | 0.56 | 0.41 | 0.48 |
| rIMSE($b_1(t)$) | 1 | $M : 10 \to 100$ | 1.05 | 0.84 | 0.94 |
| rIMSE($b_1(t)$) | 1 | $G : 30 \to 60$ | 0.74 | 0.60 | 0.67 |
| rIMSE($b_1(t)$) | 1 | $\text{SNR}_\varepsilon : 1 \to 5$ | 0.19 | 0.15 | 0.17 |
| rIMSE($b_1(t)$) | 1 | $\text{SNR}_B : 0.2 \to 1$ | 1.70 | 1.29 | 1.48 |
| rIMSE($b_1(t)$) | 1 | $\text{SNR}_B : 0.2 \to 5$ | 6.28 | 4.77 | 5.48 |
| rIMSE($b_1(t)$) | 1 | $n_i : 3 \to 20$ | 0.15 | 0.12 | 0.13 |

**Table 3:** Exponeniated coefficient estimates (i.e., multiplication factors) for the $\log_2$-linear models for rIMSE($b_1(t)$). "High" and "low" estimates are estimated coefficients $\pm$ 2 standard deviations, exponentiated.

| | scenario | setting | high | low | estimate |
|---|---|---|---|---|---|
| rIMSE($\beta_1(s,t)$) | 2 | baseline | 0.62 | 0.51 | 0.56 |
| rIMSE($\beta_1(s,t)$) | 3 | baseline | 0.76 | 0.62 | 0.69 |
| rIMSE($\beta_1(s,t)$) | 4 | baseline | 1.31 | 1.08 | 1.19 |
| rIMSE($\beta_1(s,t)$) | 2 | $M : 10 \to 100$ | 0.14 | 0.12 | 0.13 |
| rIMSE($\beta_1(s,t)$) | 3 | $M : 10 \to 100$ | 0.14 | 0.12 | 0.13 |
| rIMSE($\beta_1(s,t)$) | 4 | $M : 10 \to 100$ | 0.13 | 0.12 | 0.12 |
| rIMSE($\beta_1(s,t)$) | 2 | $n_i : 3 \to 20$ | 0.15 | 0.13 | 0.14 |
| rIMSE($\beta_1(s,t)$) | 3 | $n_i : 3 \to 20$ | 0.14 | 0.12 | 0.13 |
| rIMSE($\beta_1(s,t)$) | 4 | $n_i : 3 \to 20$ | 0.14 | 0.13 | 0.13 |
| rIMSE($\beta_1(s,t)$) | 2 | $G : 30 \to 60$ | 0.60 | 0.52 | 0.55 |
| rIMSE($\beta_1(s,t)$) | 3 | $G : 30 \to 60$ | 0.62 | 0.54 | 0.58 |
| rIMSE($\beta_1(s,t)$) | 4 | $G : 30 \to 60$ | 0.58 | 0.50 | 0.54 |
| rIMSE($\beta_1(s,t)$) | 2 | $\text{SNR}_B : 0.2 \to 1$ | 0.12 | 0.10 | 0.11 |
| rIMSE($\beta_1(s,t)$) | 3 | $\text{SNR}_B : 0.2 \to 1$ | 0.12 | 0.10 | 0.11 |
| rIMSE($\beta_1(s,t)$) | 4 | $\text{SNR}_B : 0.2 \to 1$ | 0.14 | 0.12 | 0.13 |
| rIMSE($\beta_1(s,t)$) | 2 | $\text{SNR}_B : 0.2 \to 5$ | 0.06 | 0.05 | 0.05 |
| rIMSE($\beta_1(s,t)$) | 3 | $\text{SNR}_B : 0.2 \to 5$ | 0.06 | 0.05 | 0.06 |
| rIMSE($\beta_1(s,t)$) | 4 | $\text{SNR}_B : 0.2 \to 5$ | 0.07 | 0.06 | 0.07 |
| rIMSE($\beta_1(s,t)$) | 2 | $\text{SNR}_\varepsilon : 1 \to 5$ | 0.07 | 0.06 | 0.06 |
| rIMSE($\beta_1(s,t)$) | 3 | $\text{SNR}_\varepsilon : 1 \to 5$ | 0.07 | 0.06 | 0.07 |
| rIMSE($\beta_1(s,t)$) | 4 | $\text{SNR}_\varepsilon : 1 \to 5$ | 0.07 | 0.06 | 0.07 |

**Table 4:** Exponeniated coefficient estimates (i.e., multiplication factors) for the $\log_2$-linear models for rIMSE($\int x_1(s)\beta_1(s,t)ds$). "High" and "low" estimates are estimated coefficients $\pm$ 2 standard deviations, exponentiated.

| | scenario | setting | high | low | estimate |
|---|---|---|---|---|---|
| rIMSE($\beta_2(s,t)$) | 3 | baseline | 1.07 | 0.89 | 0.97 |
| rIMSE($\beta_2(s,t)$) | 3 | $M : 10 \to 100$ | 0.14 | 0.12 | 0.13 |
| rIMSE($\beta_2(s,t)$) | 3 | $n_i : 3 \to 20$ | 0.16 | 0.14 | 0.15 |
| rIMSE($\beta_2(s,t)$) | 3 | $G : 30 \to 60$ | 0.62 | 0.54 | 0.58 |
| rIMSE($\beta_2(s,t)$) | 3 | $\text{SNR}_B : 0.2 \to 1$ | 0.12 | 0.10 | 0.11 |
| rIMSE($\beta_2(s,t)$) | 3 | $\text{SNR}_B : 0.2 \to 5$ | 0.07 | 0.06 | 0.06 |
| rIMSE($\beta_2(s,t)$) | 3 | $\text{SNR}_\varepsilon : 1 \to 5$ | 0.07 | 0.06 | 0.07 |

**Table 5:** Exponeniated coefficient estimates (i.e., multiplication factors) for the $\log_2$-linear models for rIMSE($\int x_2(s)\beta_2(s,t)ds$). "High" and "low" estimates are estimated coefficients $\pm$ 2 standard deviations, exponentiated.

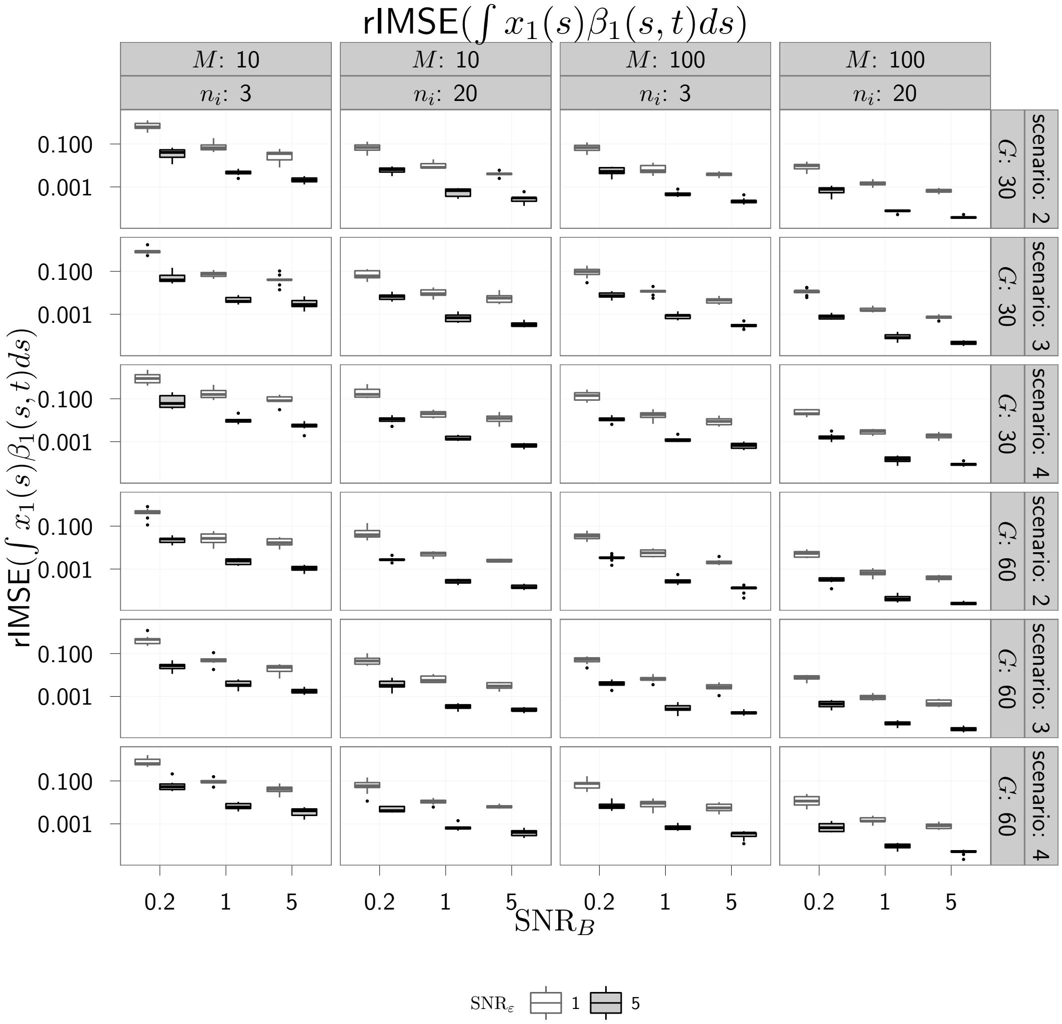


**Figure 5:** rIMSE for $\int x_1(s)\hat{\beta}_1(s,t)ds$ for all combinations of the various settings.

| | scenario | setting | high | low | estimate |
|---|---|---|---|---|---|
| rIMSE($\gamma_1(z_1,t)$) | 4 | baseline | 1.00 | 0.82 | 0.91 |
| rIMSE($\gamma_1(z_1,t)$) | 4 | $M : 10 \to 100$ | 0.16 | 0.14 | 0.15 |
| rIMSE($\gamma_1(z_1,t)$) | 4 | $n_i : 3 \to 20$ | 0.18 | 0.16 | 0.17 |
| rIMSE($\gamma_1(z_1,t)$) | 4 | $G : 30 \to 60$ | 0.66 | 0.56 | 0.61 |
| rIMSE($\gamma_1(z_1,t)$) | 4 | $\text{SNR}_B : 0.2 \to 1$ | 0.16 | 0.13 | 0.14 |
| rIMSE($\gamma_1(z_1,t)$) | 4 | $\text{SNR}_B : 0.2 \to 5$ | 0.09 | 0.07 | 0.08 |
| rIMSE($\gamma_1(z_1,t)$) | 4 | $\text{SNR}_\varepsilon : 1 \to 5$ | 0.10 | 0.08 | 0.09 |

**Table 6:** Exponeniated coefficient estimates (i.e., multiplication factors) for the $\log_2$-linear models for rIMSE($\gamma_1(z_1,t)$). "High" and "low" estimates are estimated coefficients ± 2 standard deviations, exponentiated.

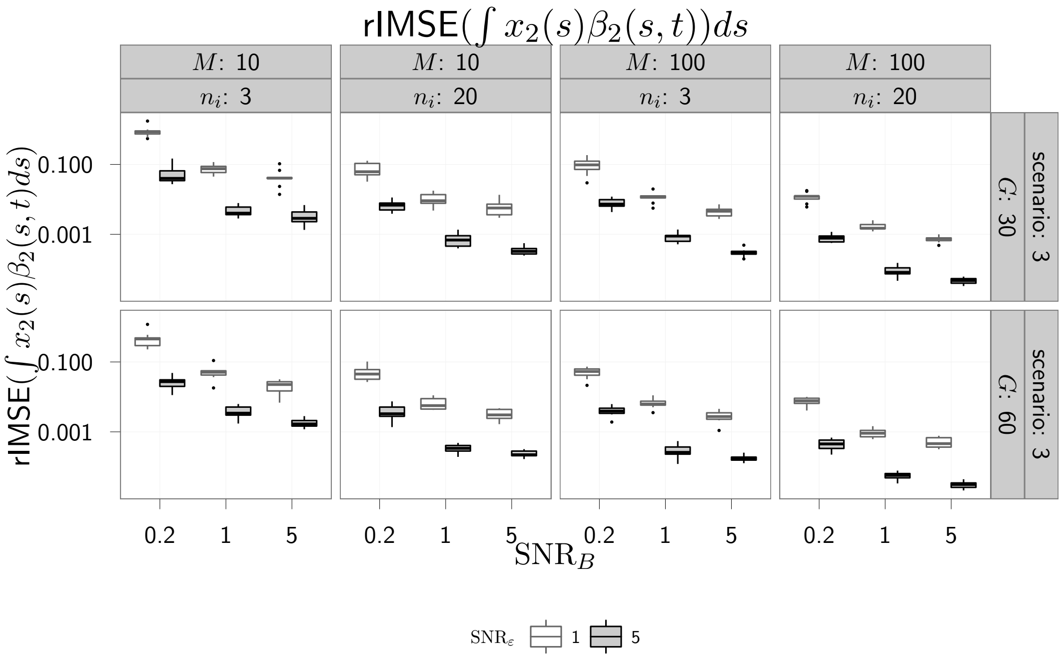


**Figure 6:** rIMSE for $\int x_2(s)\hat{\beta}_2(s,t)ds$ for all combinations of the various settings.

| | scenario | setting | high | low | estimate |
|---|---|---|---|---|---|
| rIMSE($\delta_2(t)z_2$) | 4 | baseline | 2.23 | 1.57 | 1.87 |
| rIMSE($\delta_2(t)z_2$) | 4 | $M: 10 \rightarrow 100$ | 0.12 | 0.09 | 0.11 |
| rIMSE($\delta_2(t)z_2$) | 4 | $n_i: 3 \rightarrow 20$ | 0.16 | 0.12 | 0.14 |
| rIMSE($\delta_2(t)z_2$) | 4 | $G: 30 \rightarrow 60$ | 0.63 | 0.49 | 0.55 |
| rIMSE($\delta_2(t)z_2$) | 4 | $\mathrm{SNR}_B: 0.2 \rightarrow 1$ | 0.13 | 0.09 | 0.11 |
| rIMSE($\delta_2(t)z_2$) | 4 | $\mathrm{SNR}_B: 0.2 \rightarrow 5$ | 0.07 | 0.05 | 0.06 |
| rIMSE($\delta_2(t)z_2$) | 4 | $\mathrm{SNR}_\varepsilon: 1 \rightarrow 5$ | 0.08 | 0.06 | 0.07 |

**Table 7:** Exponeniated coefficient estimates (i.e., multiplication factors) for the $\log_2$-linear models for rIMSE($\delta_2(t)z_2$). "High" and "low" estimates are estimated coefficients $\pm$ 2 standard deviations, exponentiated.

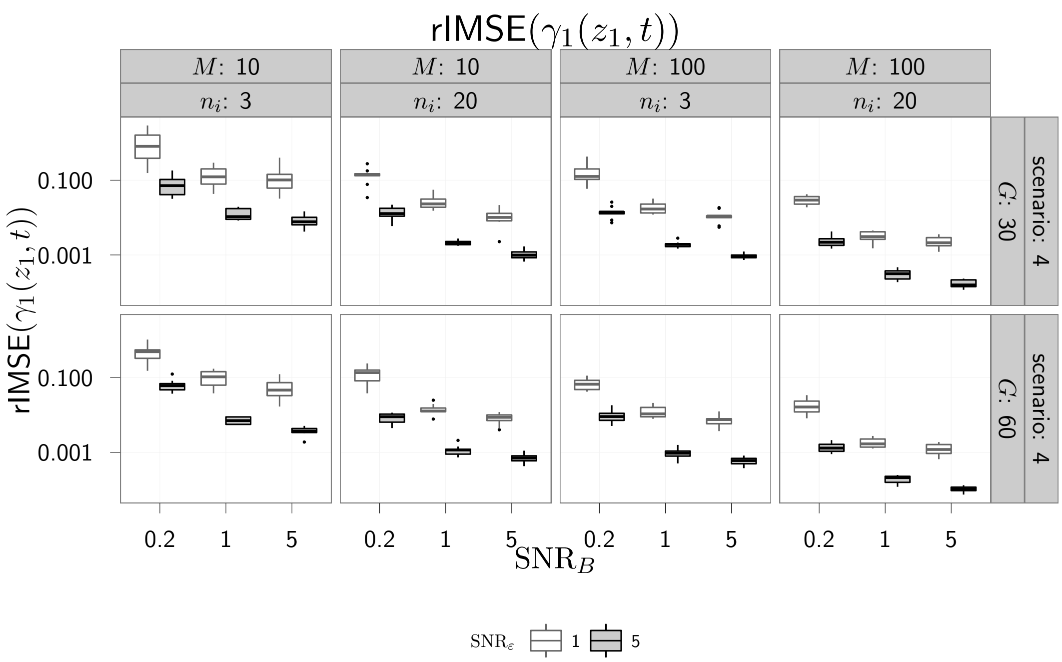


**Figure 7:** rIMSE for $\hat{\gamma}_1(z_1, t)$ for all combinations of the various settings.

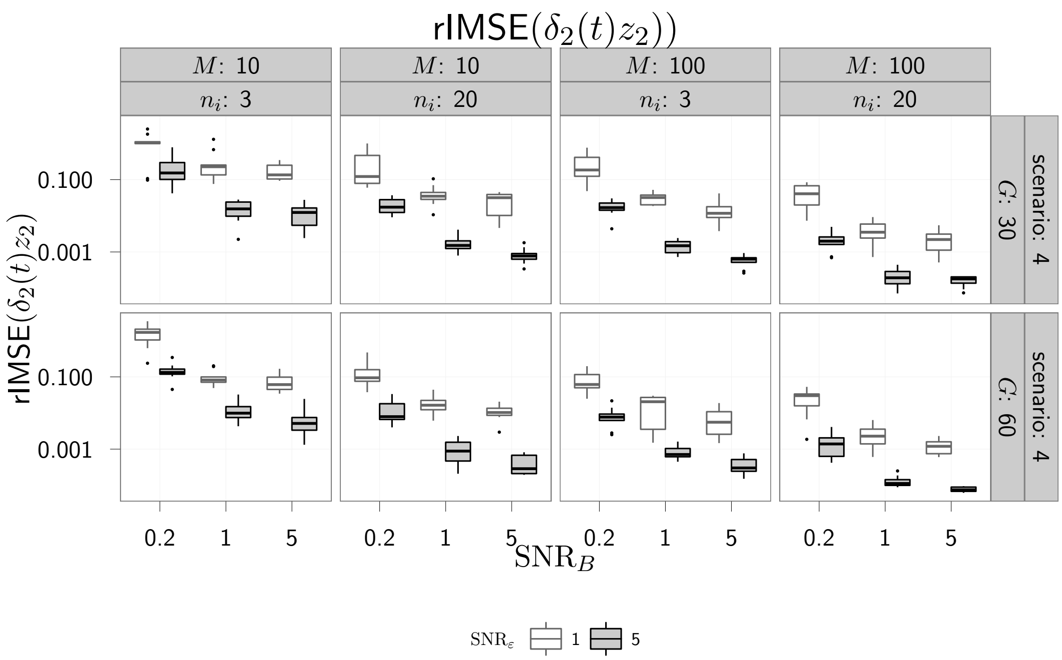


**Figure 8:** rIMSE for $\hat{\delta}_2(t)z_2$ for all combinations of the various settings.

## B.3 Coverage

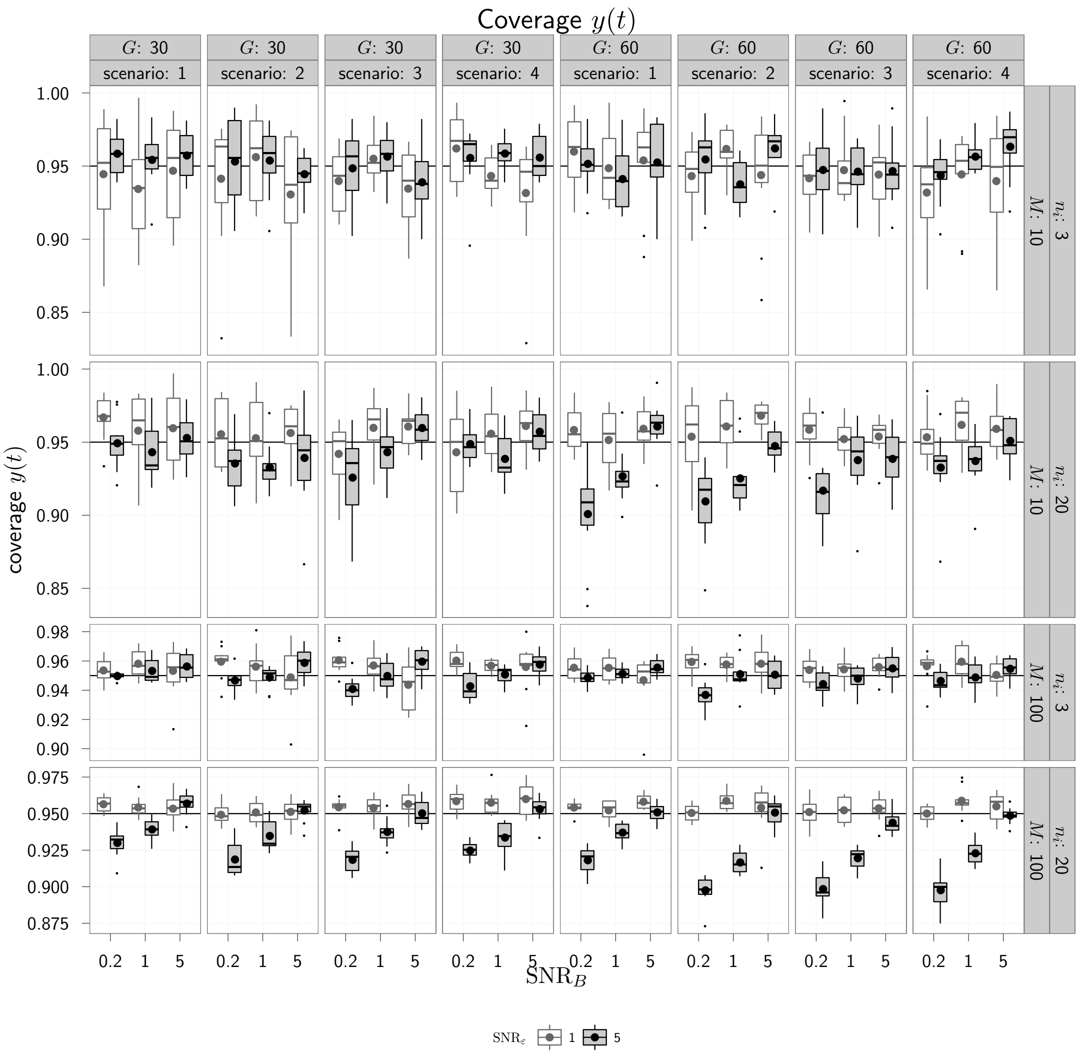


**Figure 9:** Coverage for $y(t)$ for all combinations of the various settings for nominal approximate 95% CIs. Dots denote mean coverage.

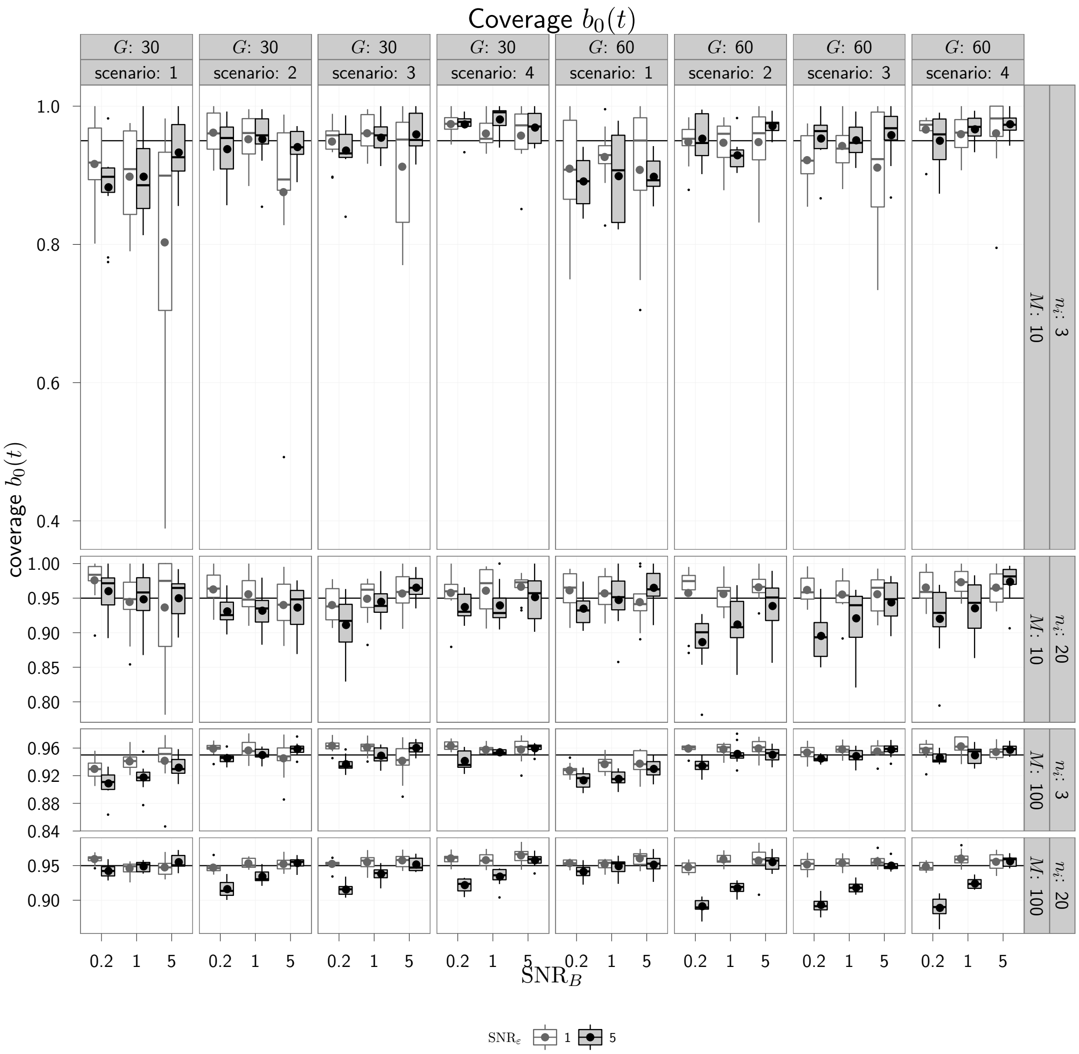


**Figure 10:** Coverage for $b_0(t)$ for all combinations of the various settings for nominal approximate 95% CIs. Dots denote mean coverage.

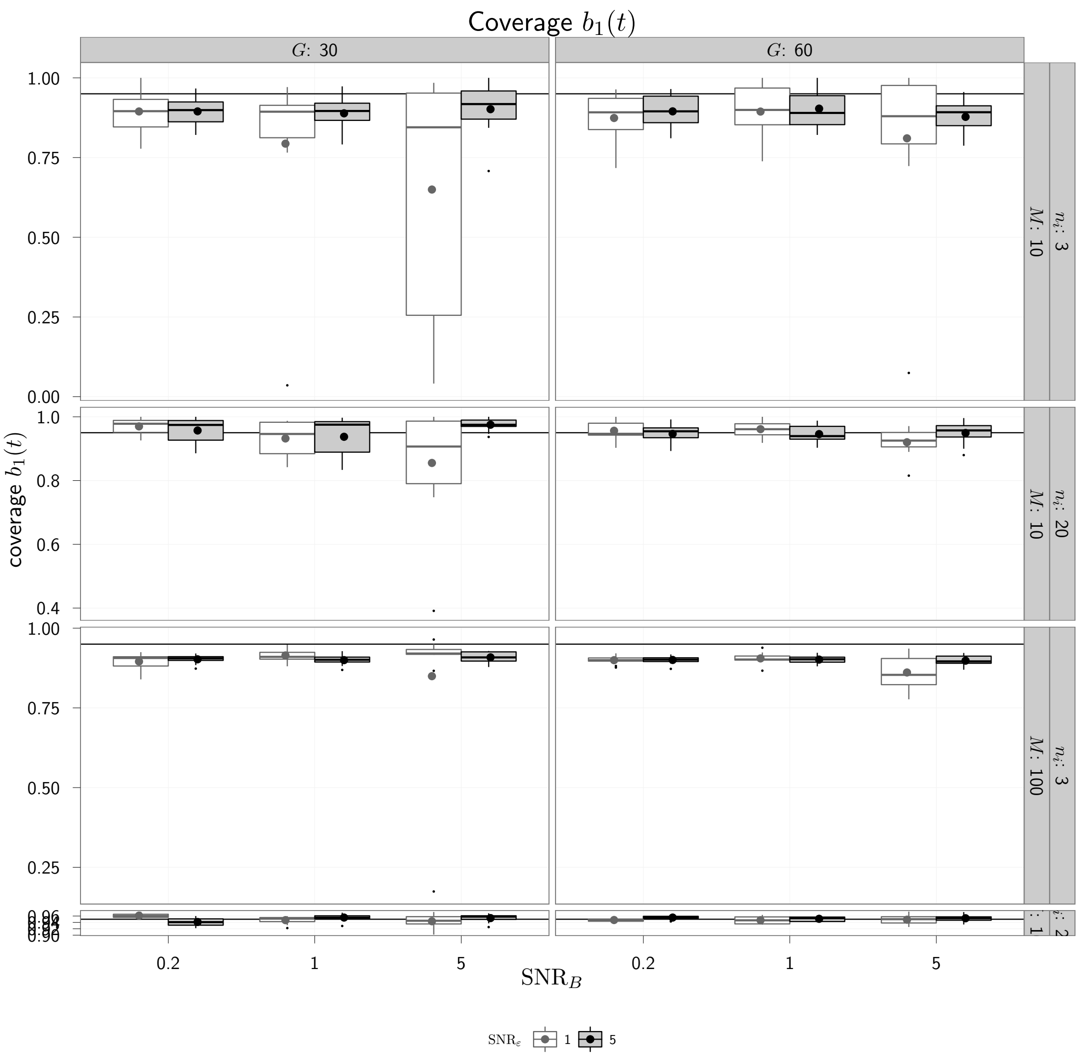


**Figure 11:** Coverage for $b_1(t)$ for all combinations of the various settings for nominal approximate 95% CIs. Dots denote mean coverage.

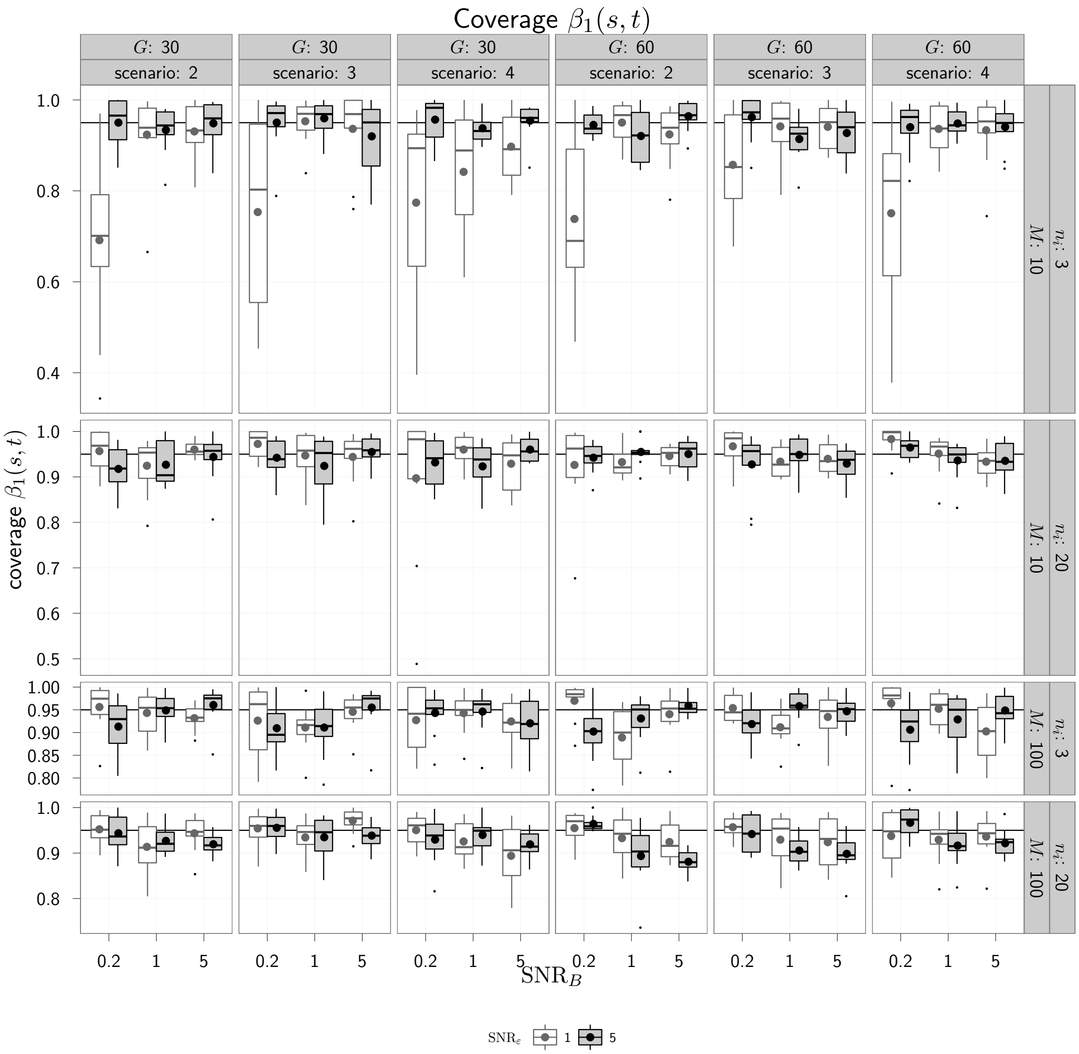


**Figure 12:** Coverage for effect of $x_1(s)$ for all combinations of the various settings for nominal approximate 95% CIs. Dots denote mean coverage.

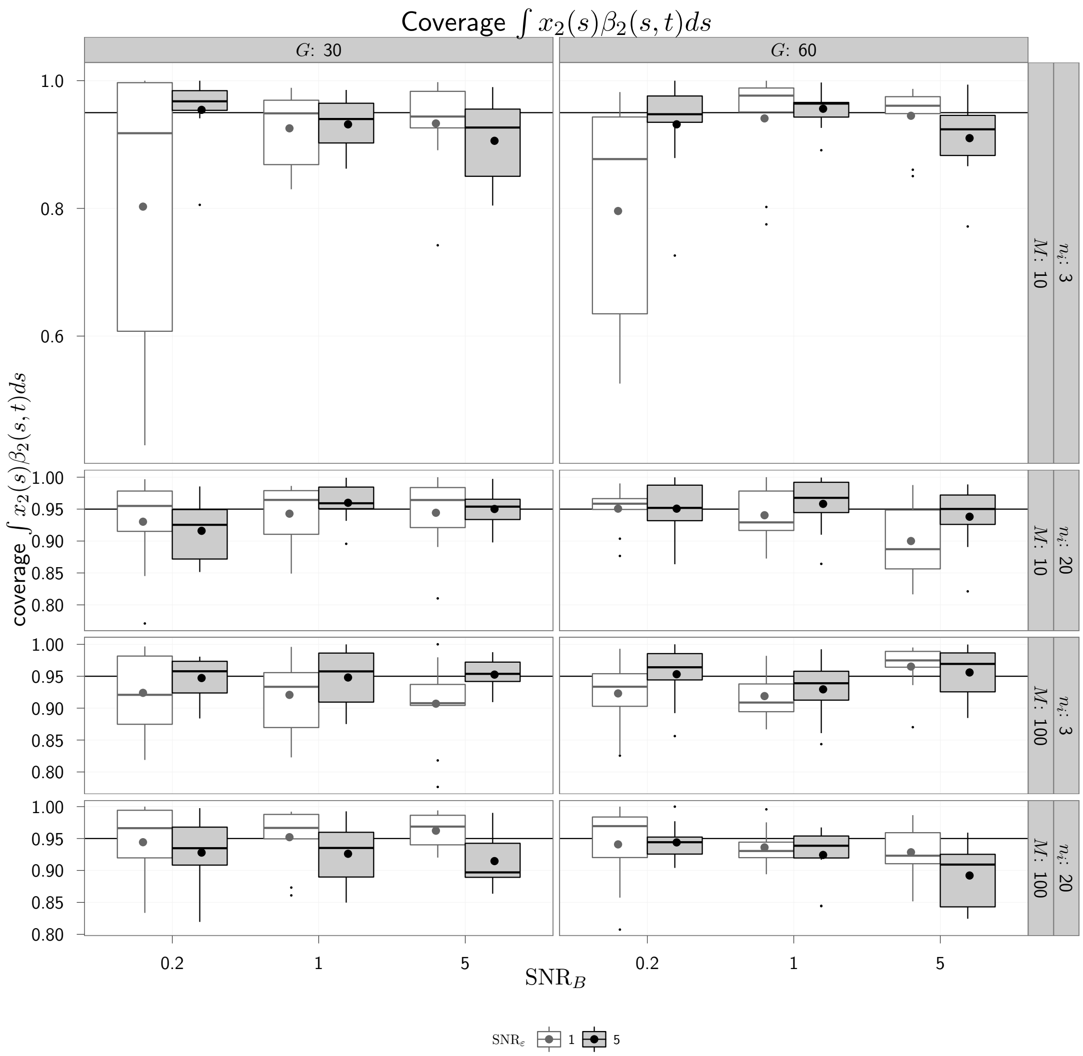


**Figure 13:** Coverage for effect of $x_2(s)$ for all combinations of the various settings for nominal approximate 95% CIs. Dots denote mean coverage.

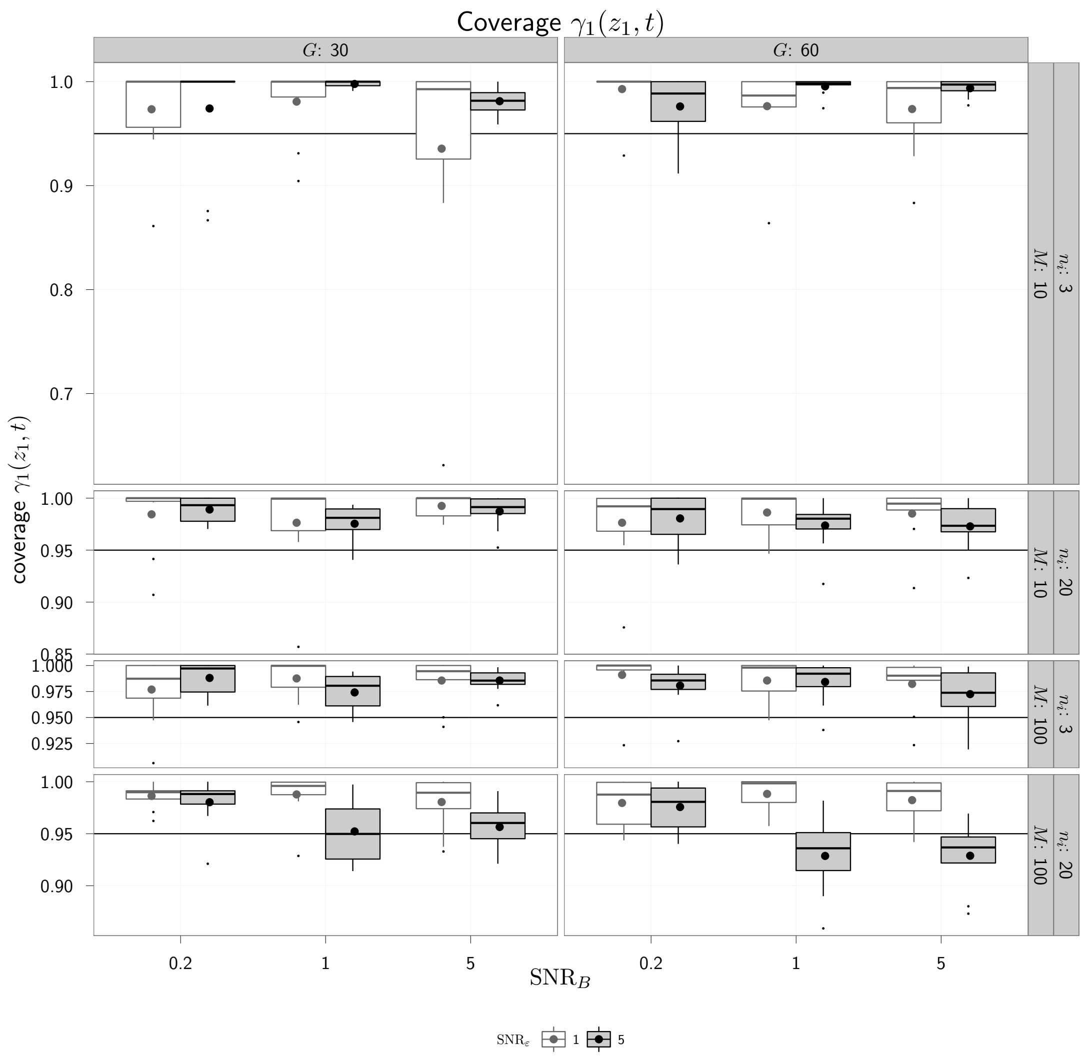


**Figure 14:** Coverage for $\gamma_1(z_1, t)$ for all combinations of the various settings for nominal approximate 95% CIs. Dots denote mean coverage.

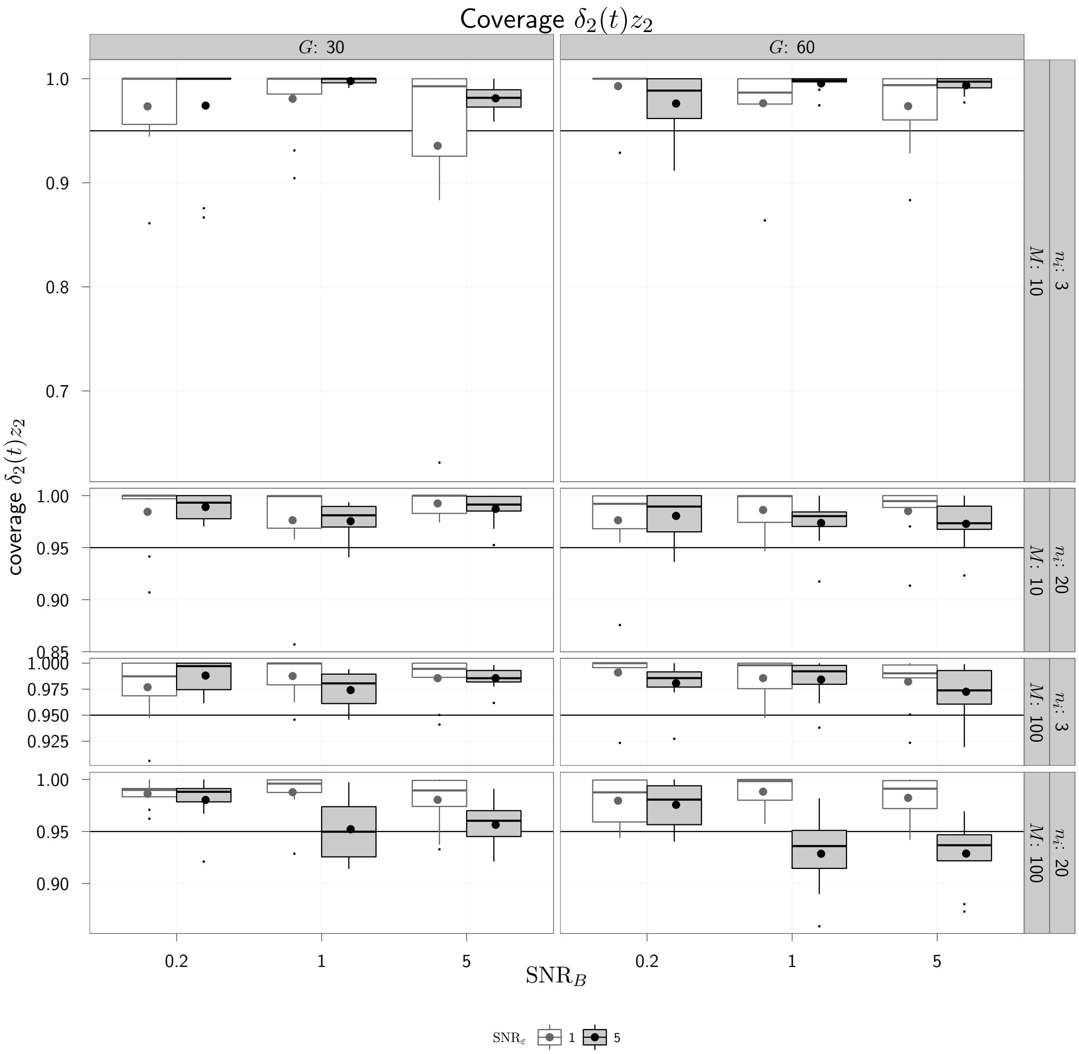


**Figure 15:** Coverage for $z_2\delta_2(t)$ for all combinations of the various settings for nominal approximate 95% CIs. Dots denote mean coverage.

## B.4 Comparison to FPC-based approaches

The following graphs show the IMSEs for the PCA-based approaches tested for scenario 2, i.e. a model with an FPC-based function-on-function term and a model with FPC-based functional random intercepts.

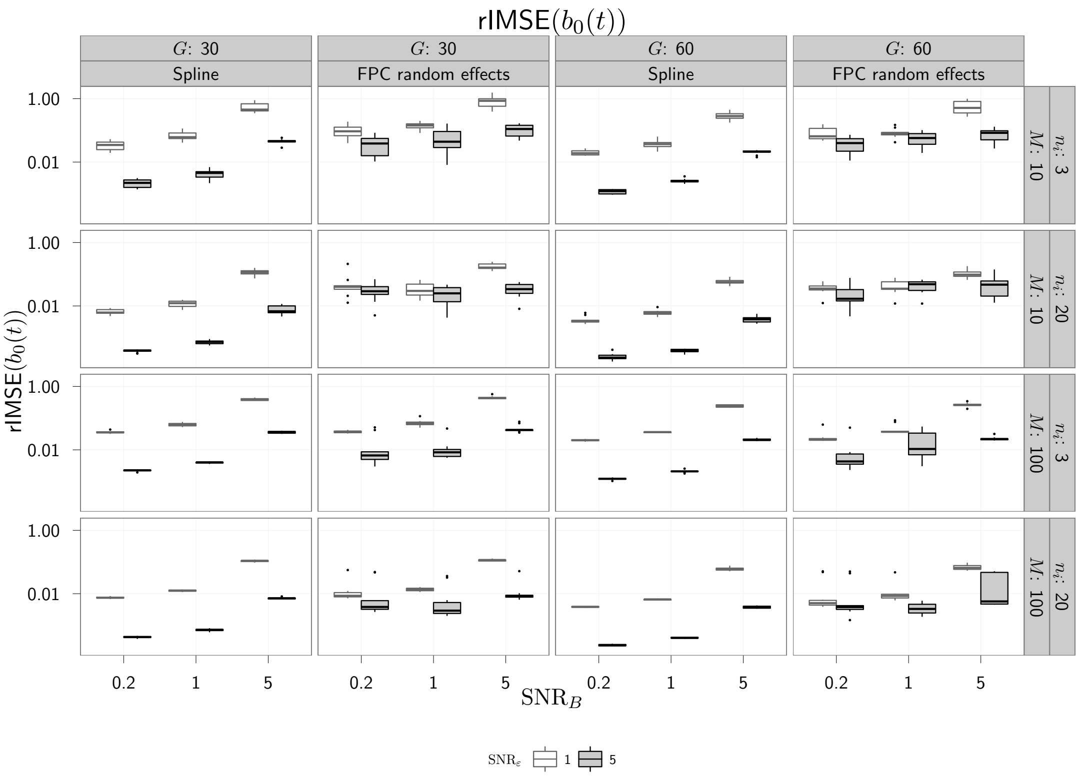


**Figure 16:** rIMSE for $b_0(t)$ for spline-based and FPC-based random effects in scenario 2.

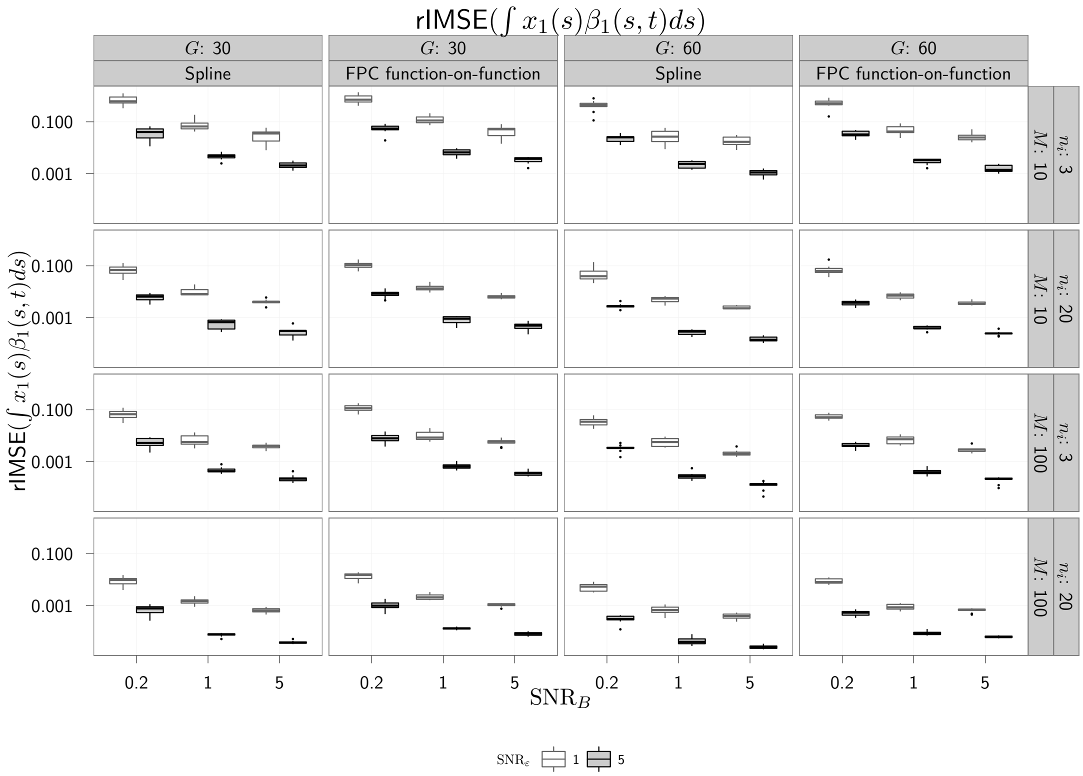


**Figure 17:** rIMSE for $\int x_1(s)\beta_1(s,t)ds$ for spline-based and FPC-based estimates in scenario 2.

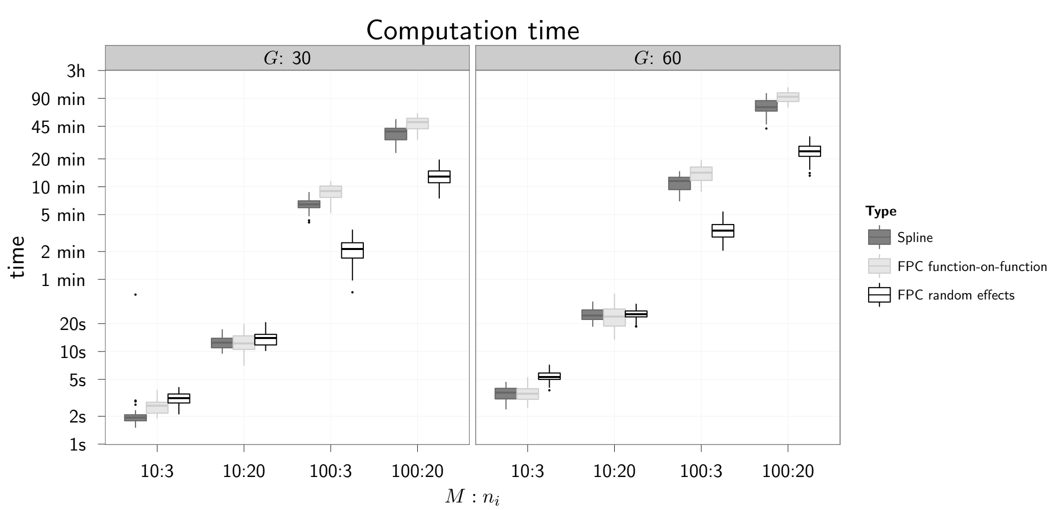


**Figure 18:** Computation times for spline-based and FPC-based fits in scenario 2.

## B.5 Exemplary data sets and fits

For each scenario, we present the setting and replication with rIMSE values closest to the median rIMSE values, followed by those with minimal and maximal rIMSE values across the various combinations of $n_i$, $M$, $G$, $\text{SNR}_B$ and $\text{SNR}_\varepsilon$. For all plots, the left column shows the observed or true quantities, while the right column shows their estimates. The bottom row displays the observed functional responses on the left and the estimated residual curves on the right, note that they are on different vertical scales. Trajectories are colour-coded for subject. For larger data sets, only a sample of at most 300 observations is plotted.

### B.5.1 Scenario 1

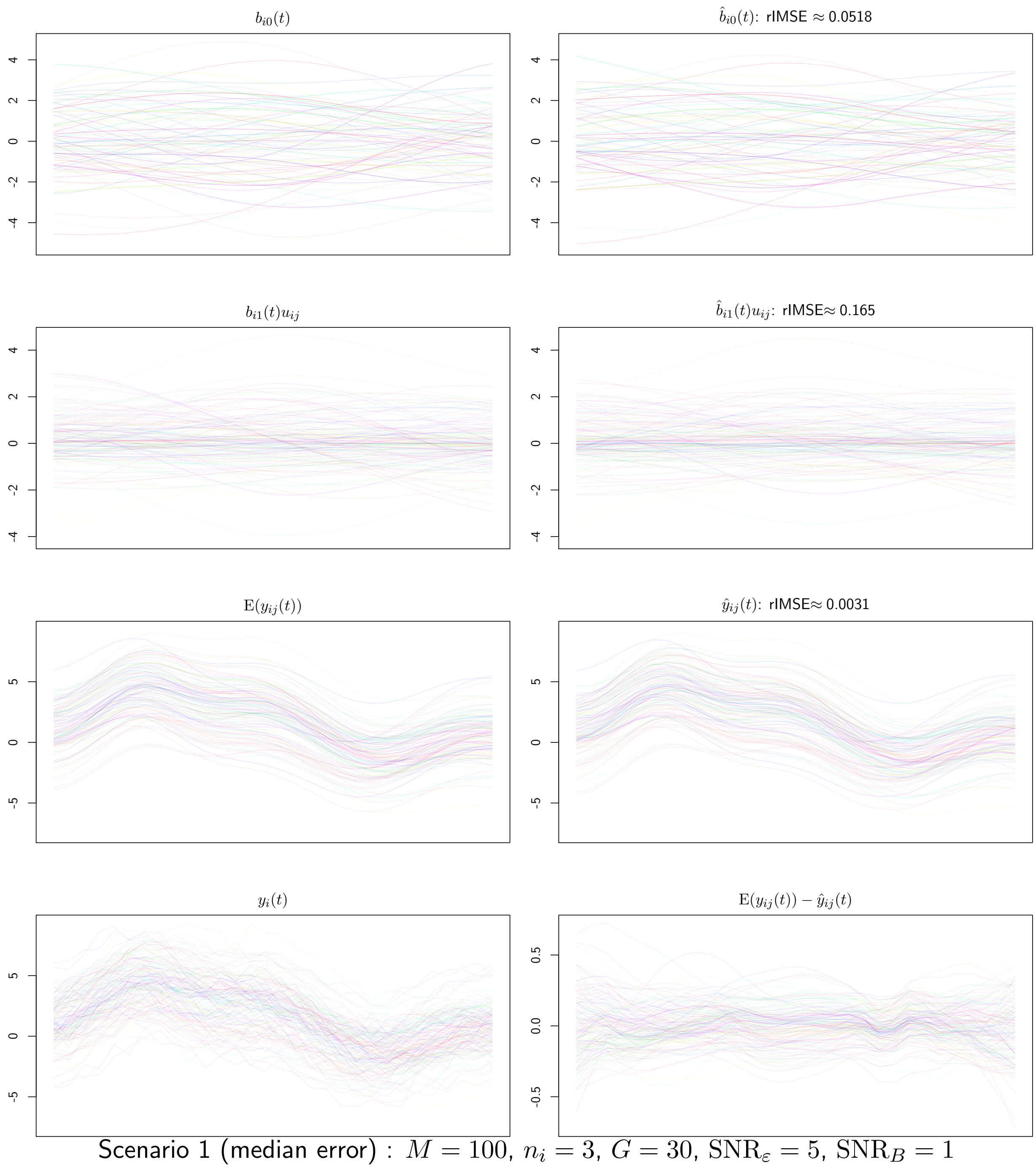


**Figure 19:** Example of data and fit for scenario 1 with median error.

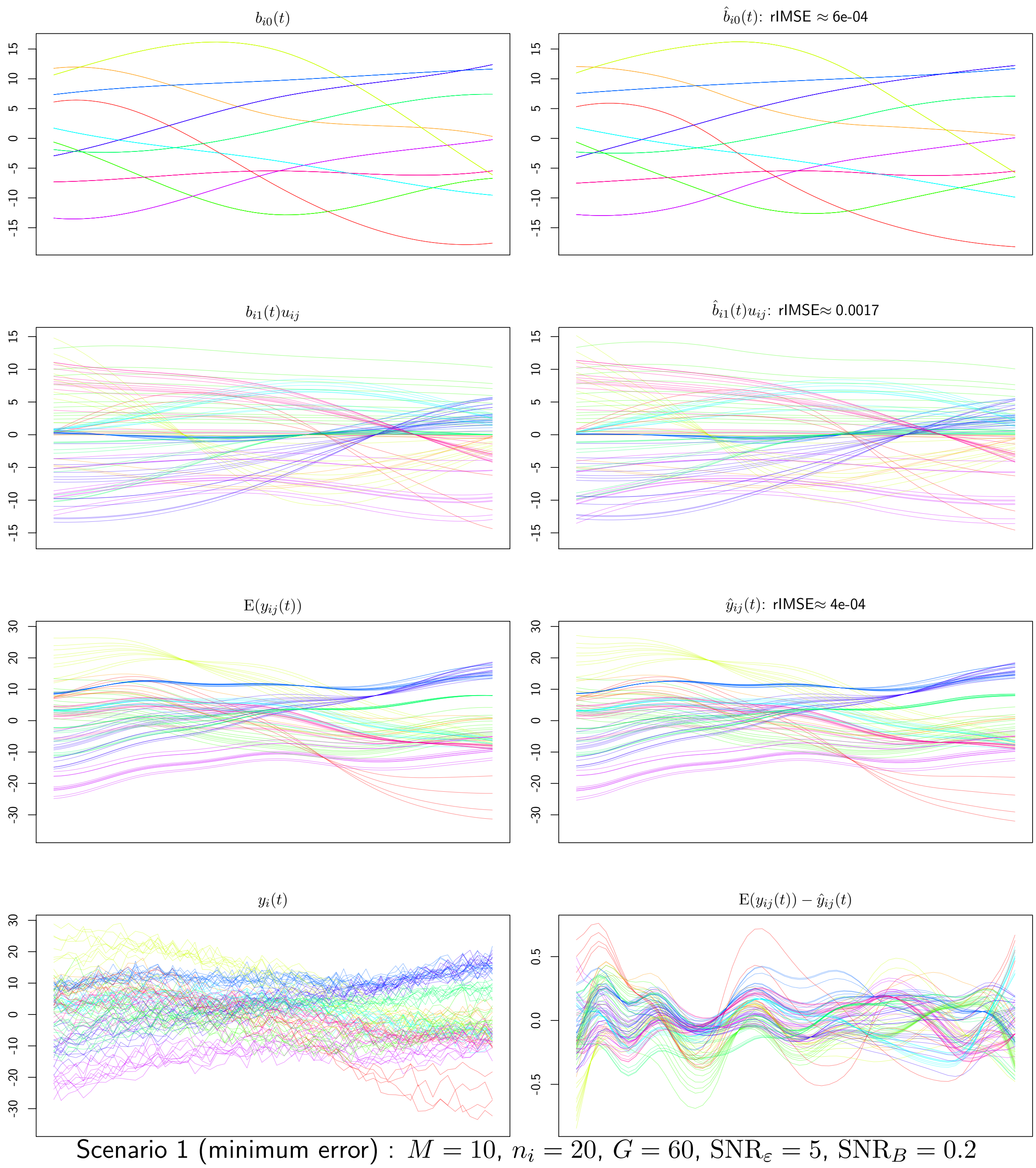


**Figure 20:** Example of data and fit for scenario 1 with minimum error.

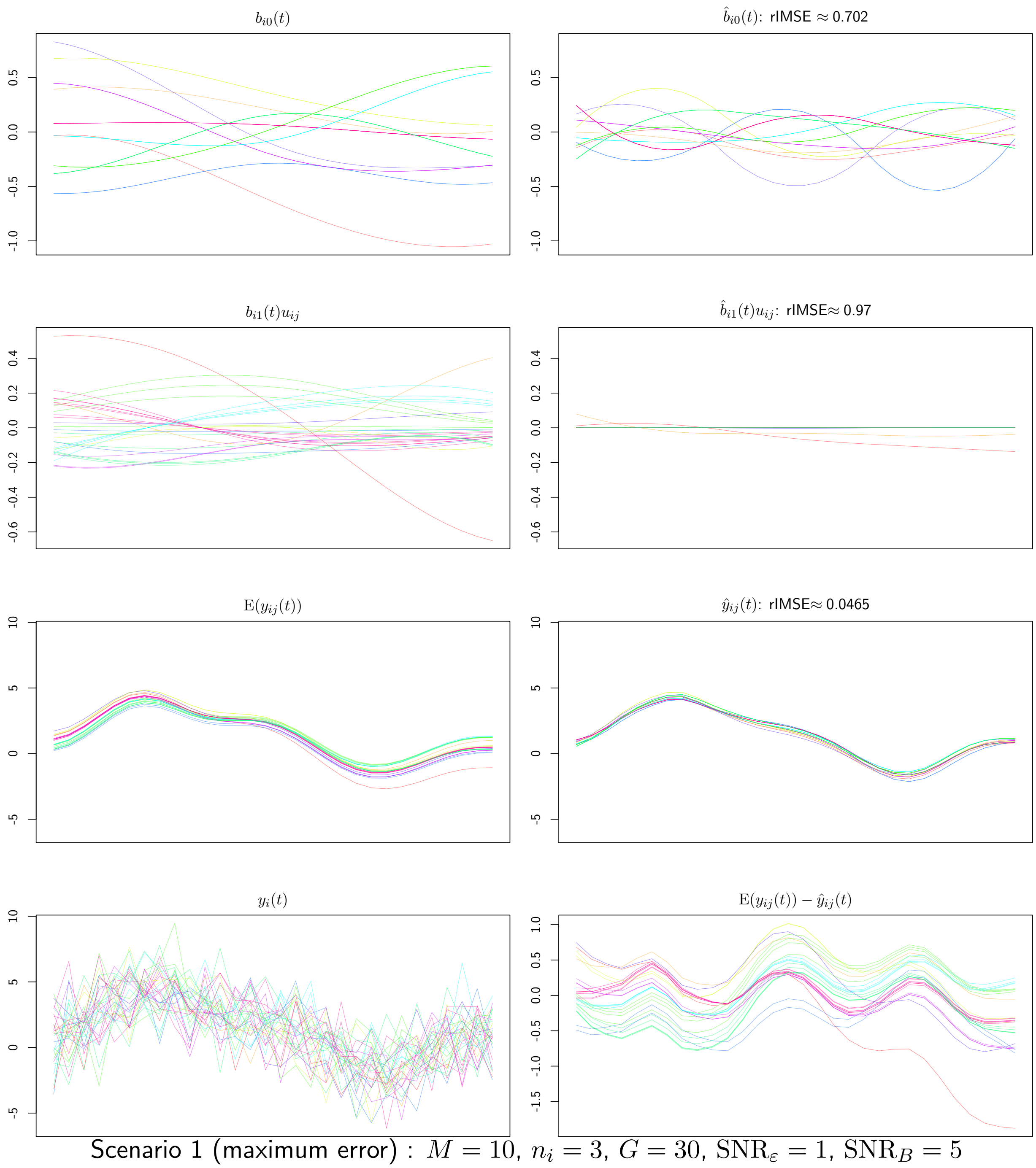


**Figure 21:** Example of data and fit for scenario 1 with maximum error.

### B.5.2 Scenario 2

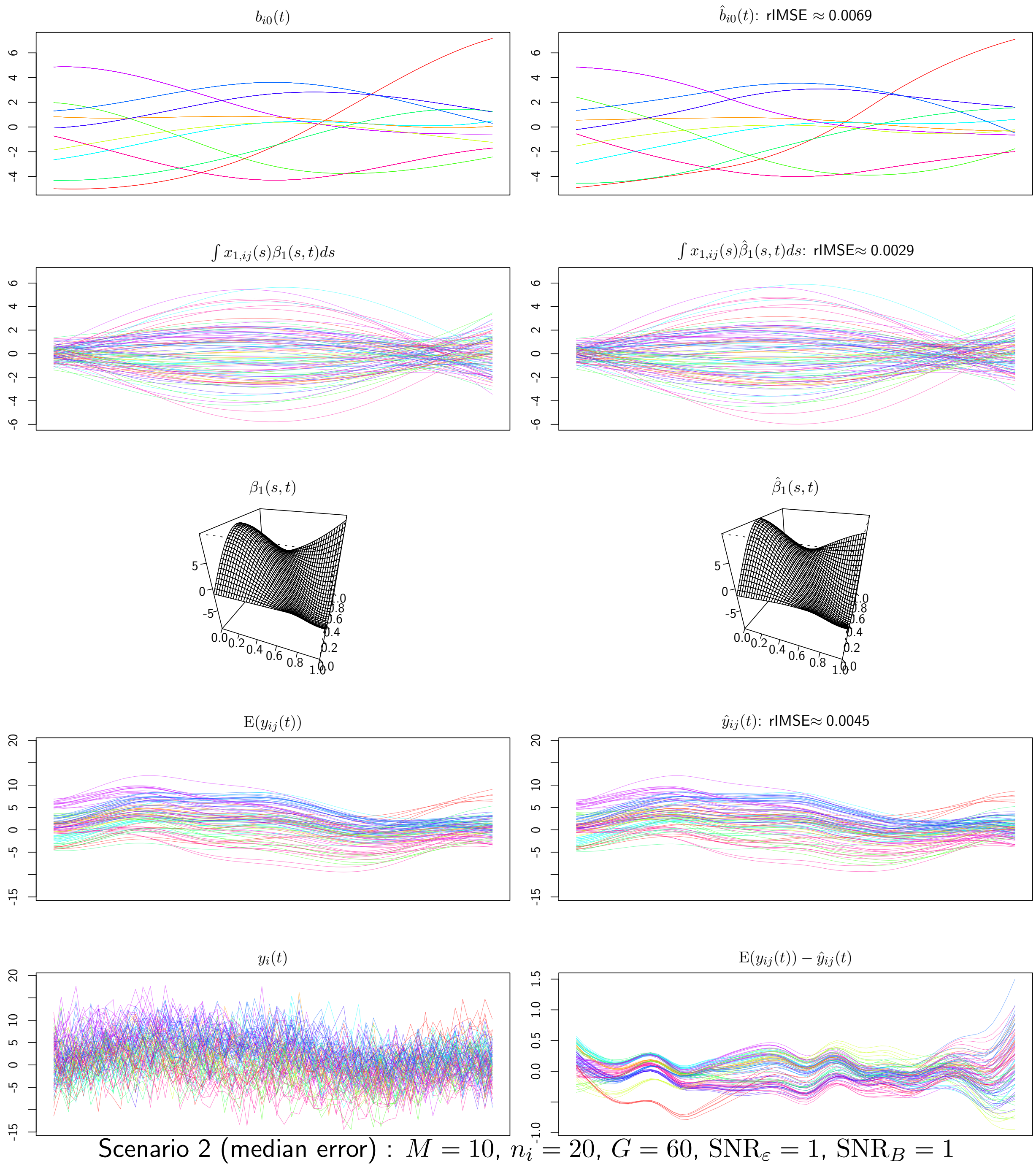


**Figure 22:** Example of data and fit for scenario 2 with median error.

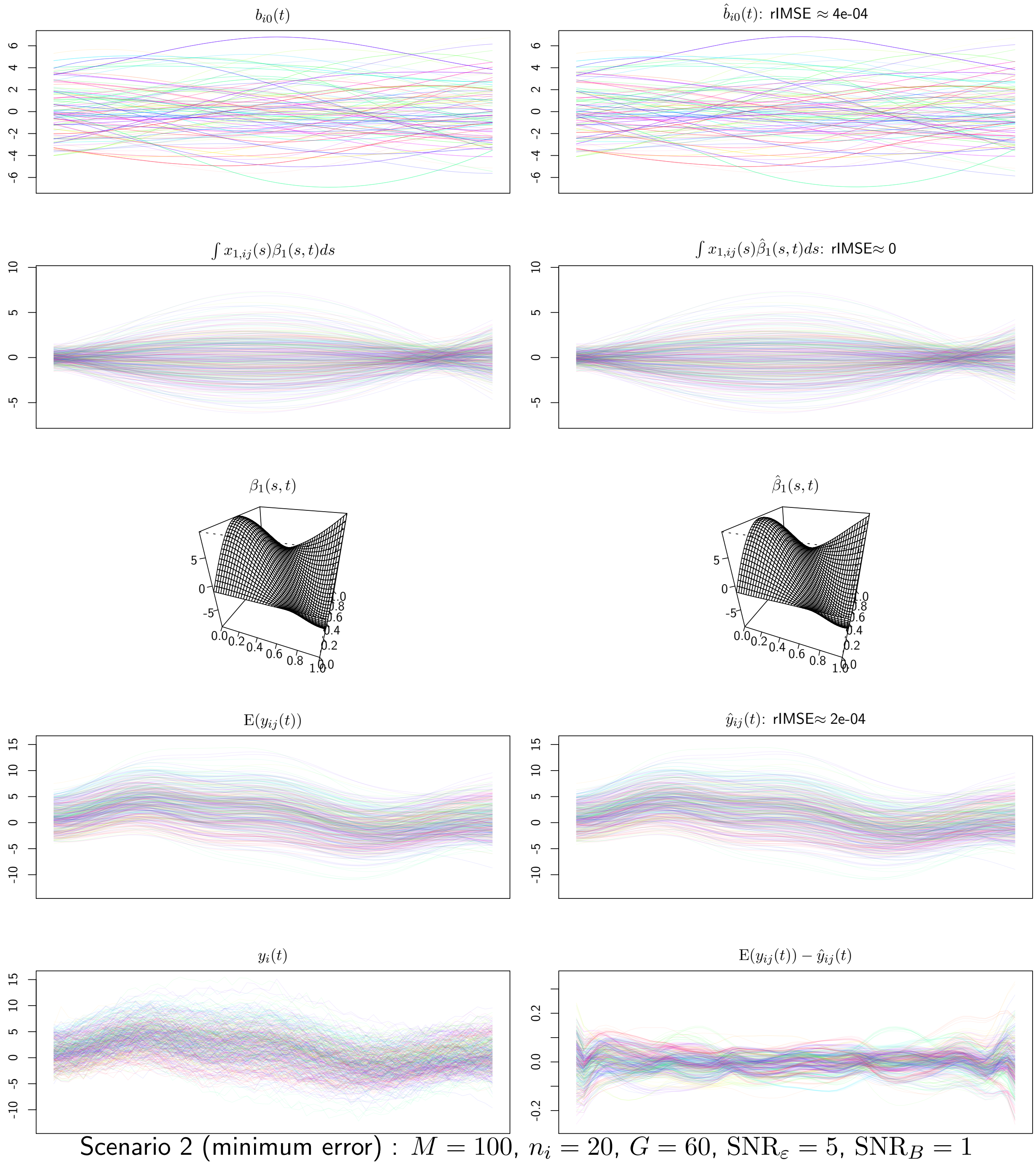


**Figure 23:** Example of data and fit for scenario 2 with minimum error.

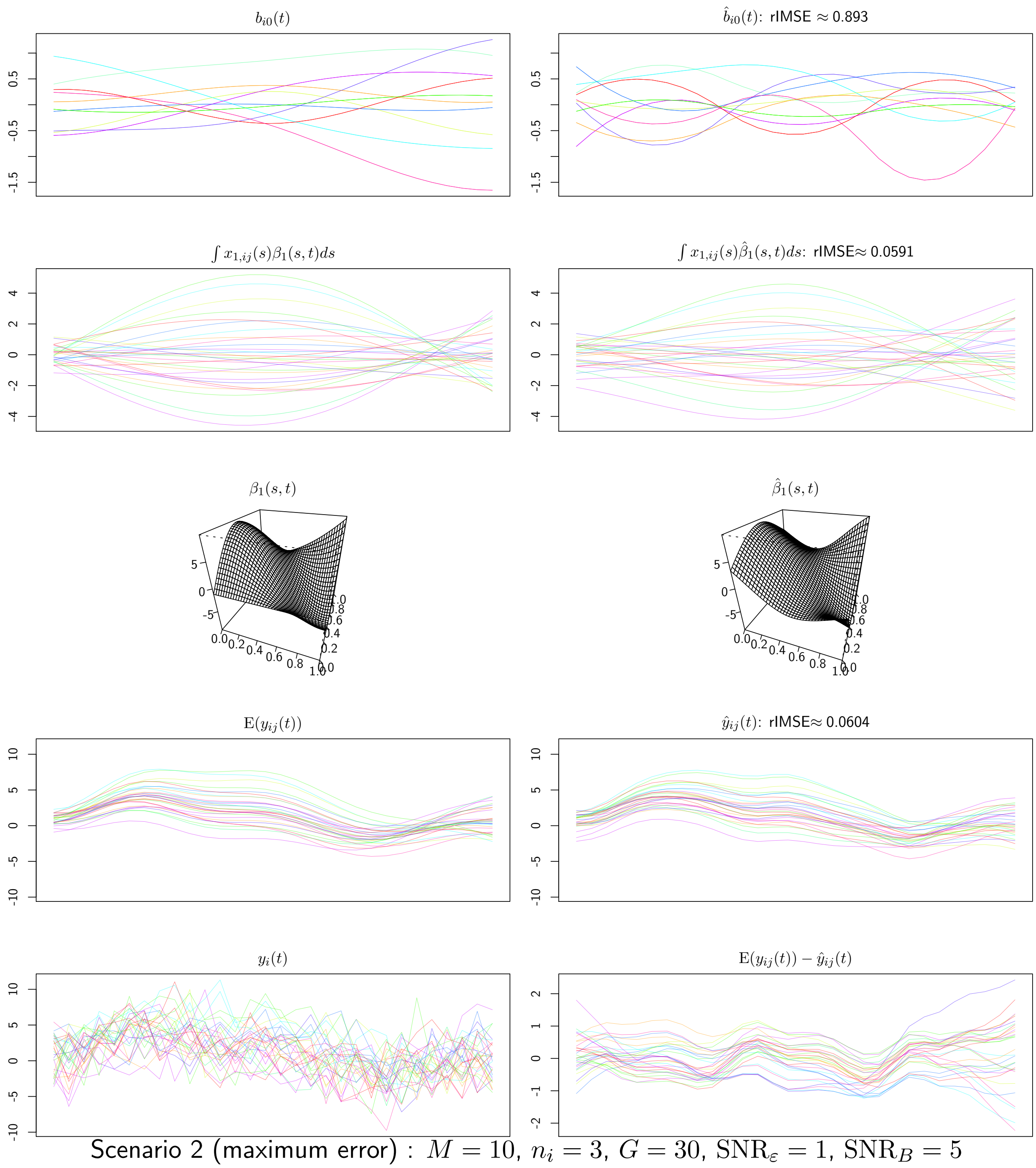


**Figure 24:** Example of data and fit for scenario 2 with maximum error.

### B.5.3 Scenario 3

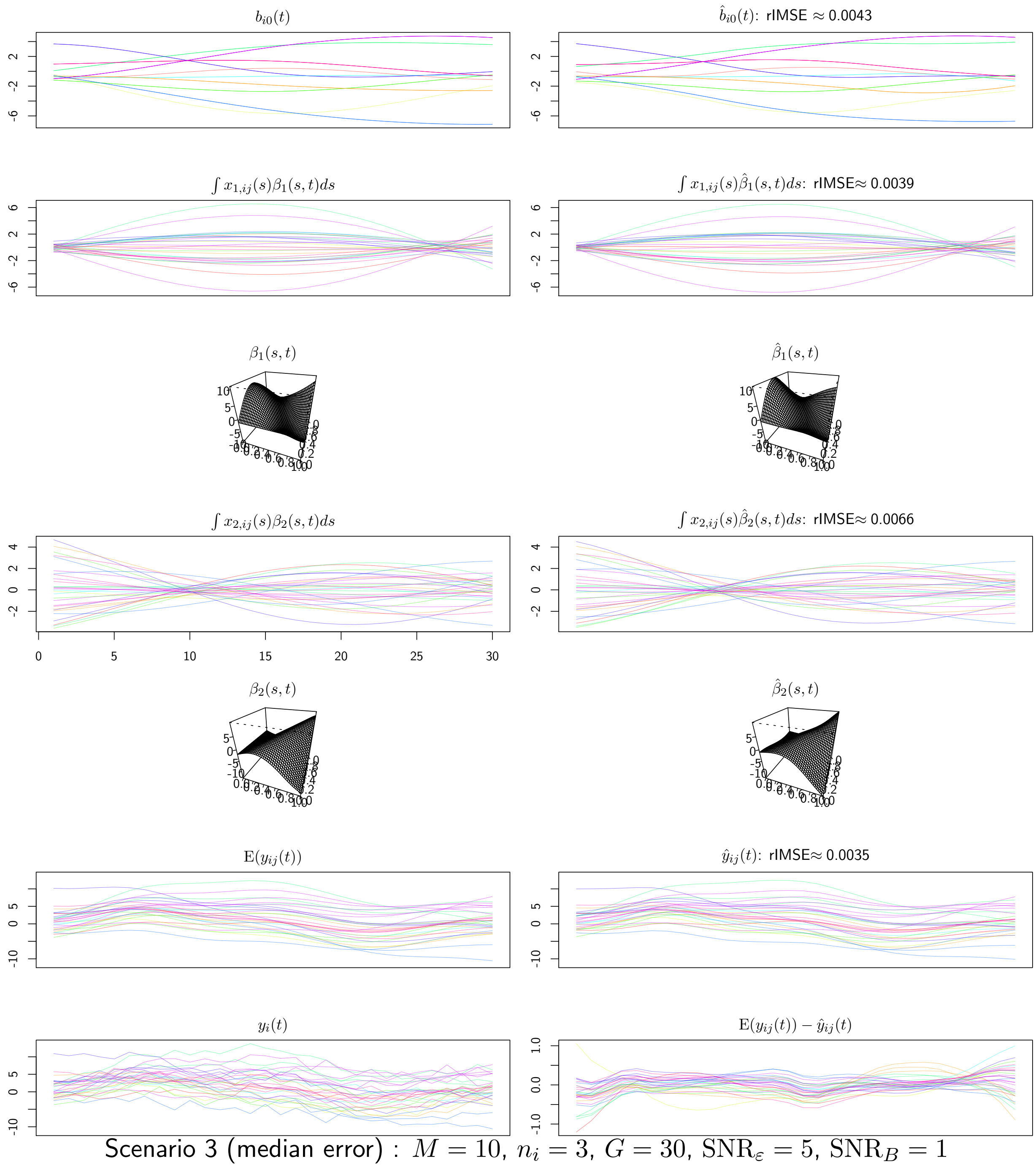


**Figure 25:** Example of data and fit for scenario 3 with median error.

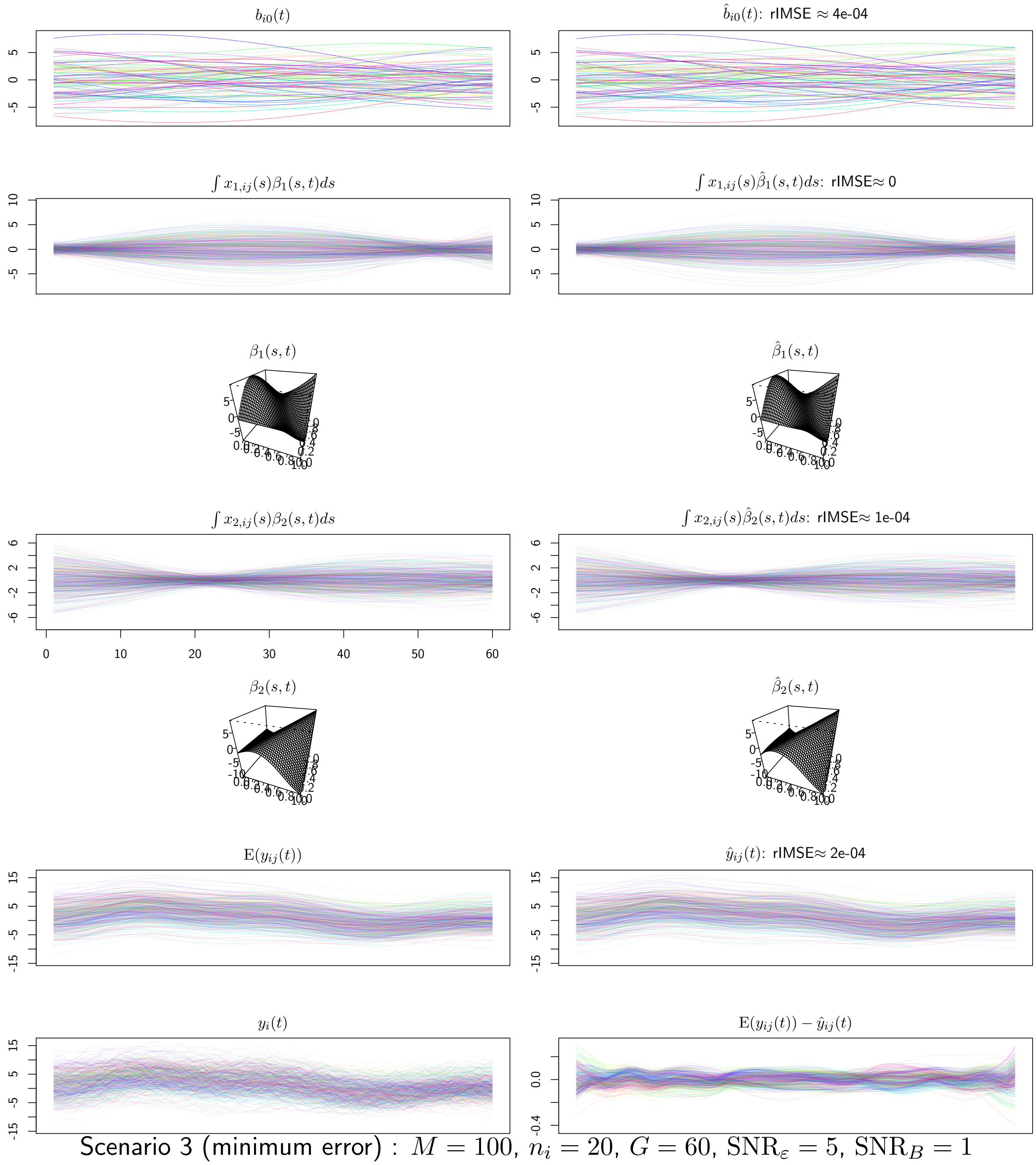


**Figure 26:** Example of data and fit for scenario 3 with minimum error.

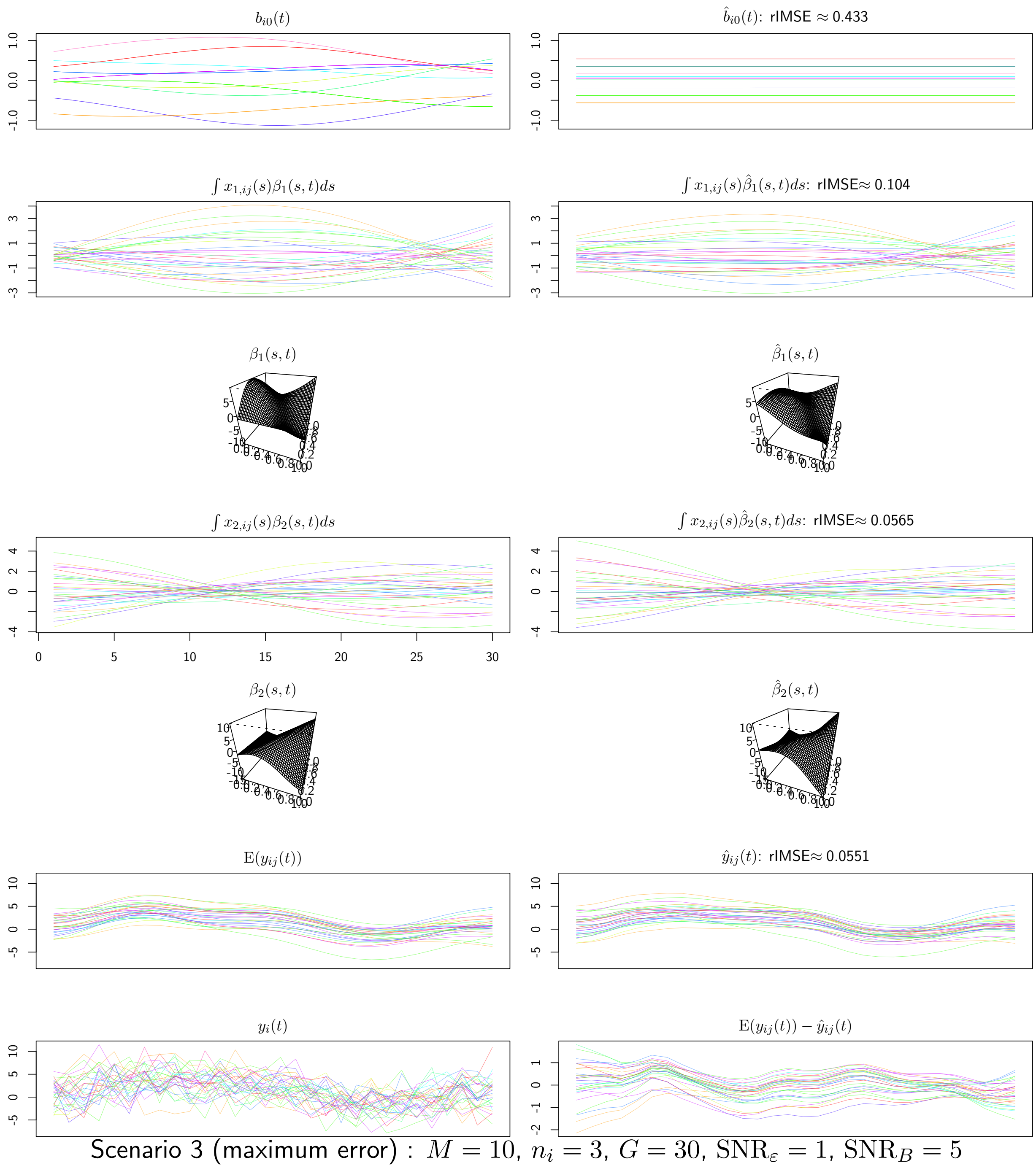


**Figure 27:** Example of data and fit for scenario 3 with maximum error.

### B.5.4 Scenario 4

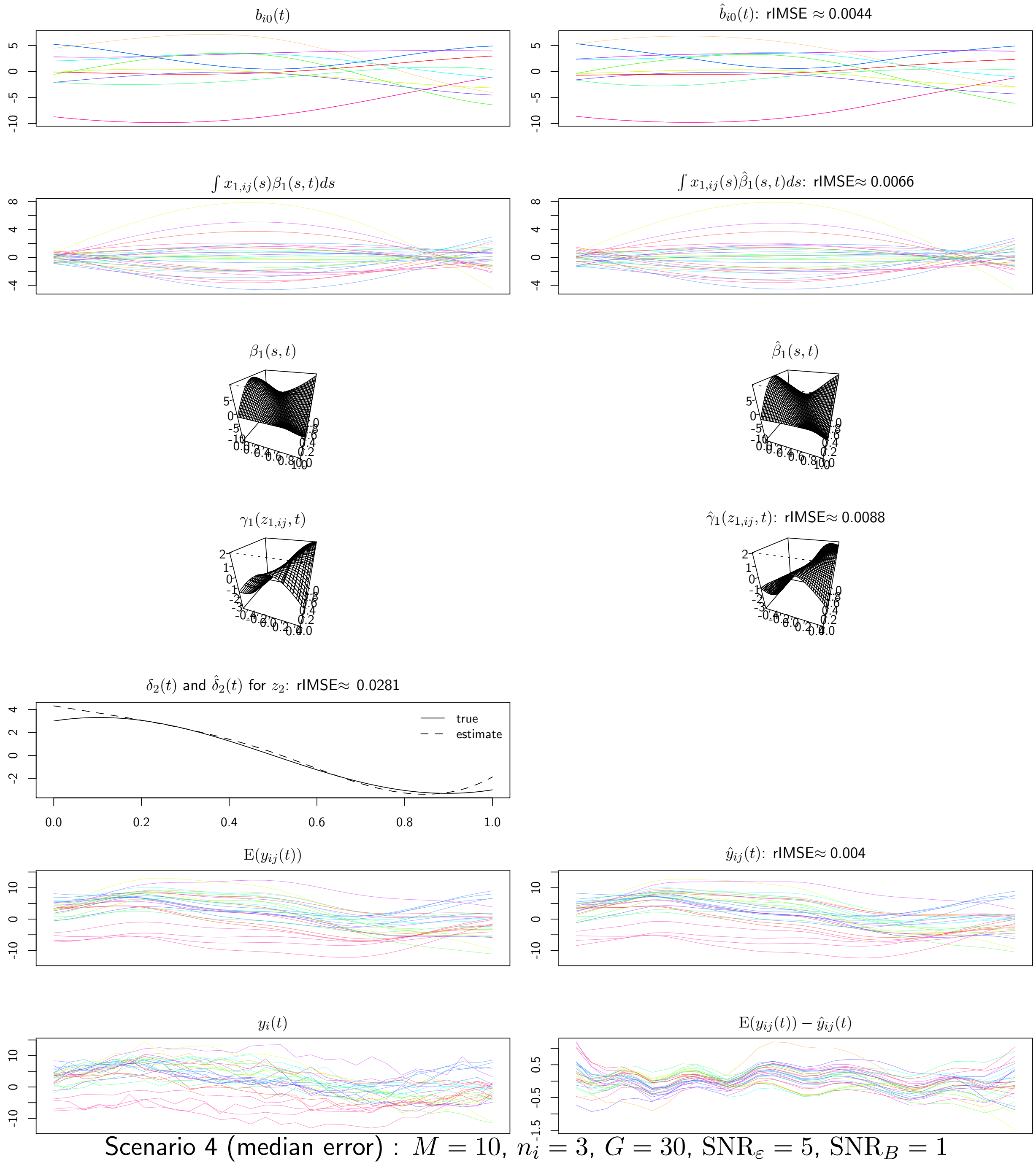


**Figure 28:** Example of data and fit for scenario 4 with median error.

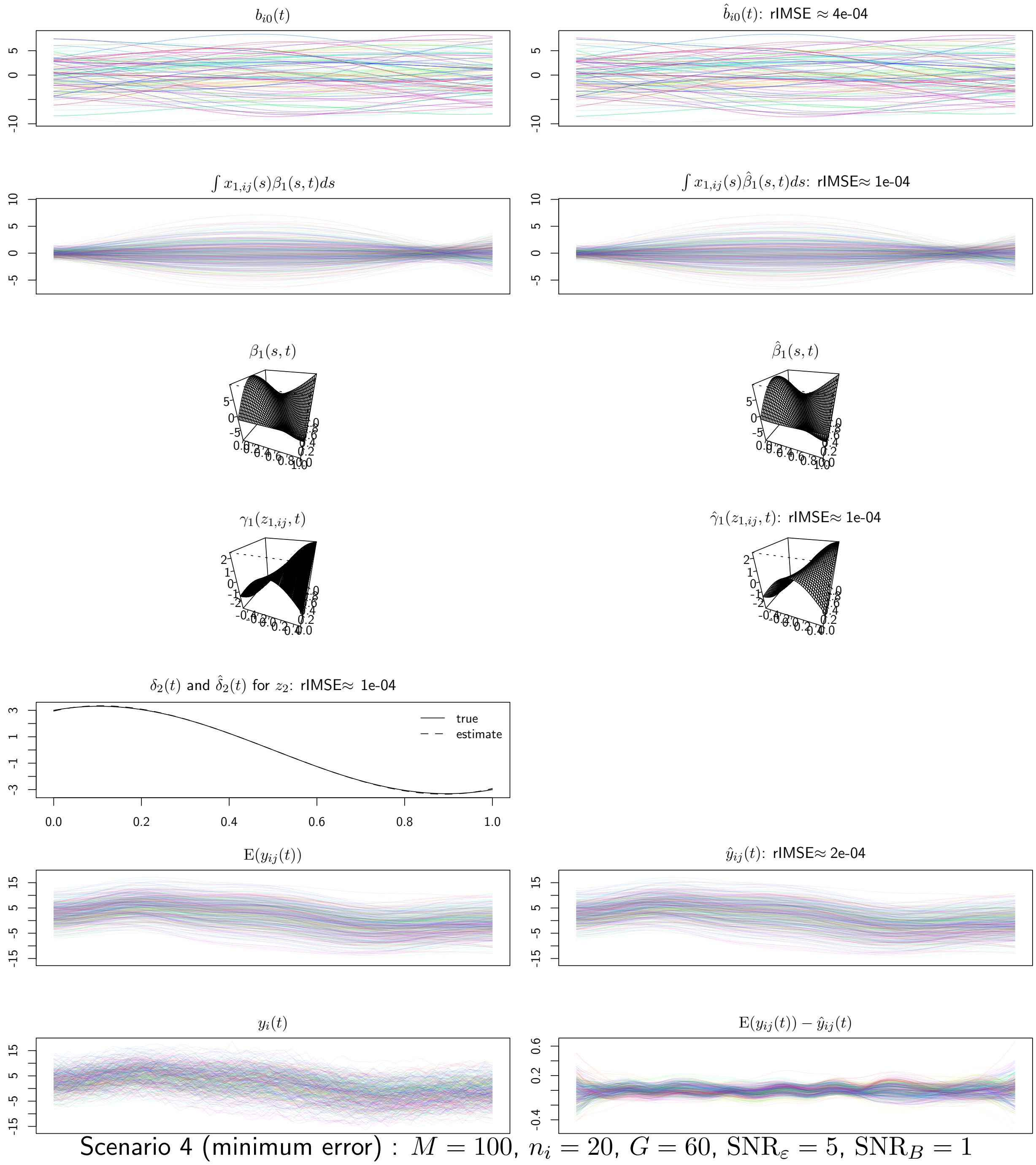


**Figure 29:** Example of data and fit for scenario 4 with minimum error.

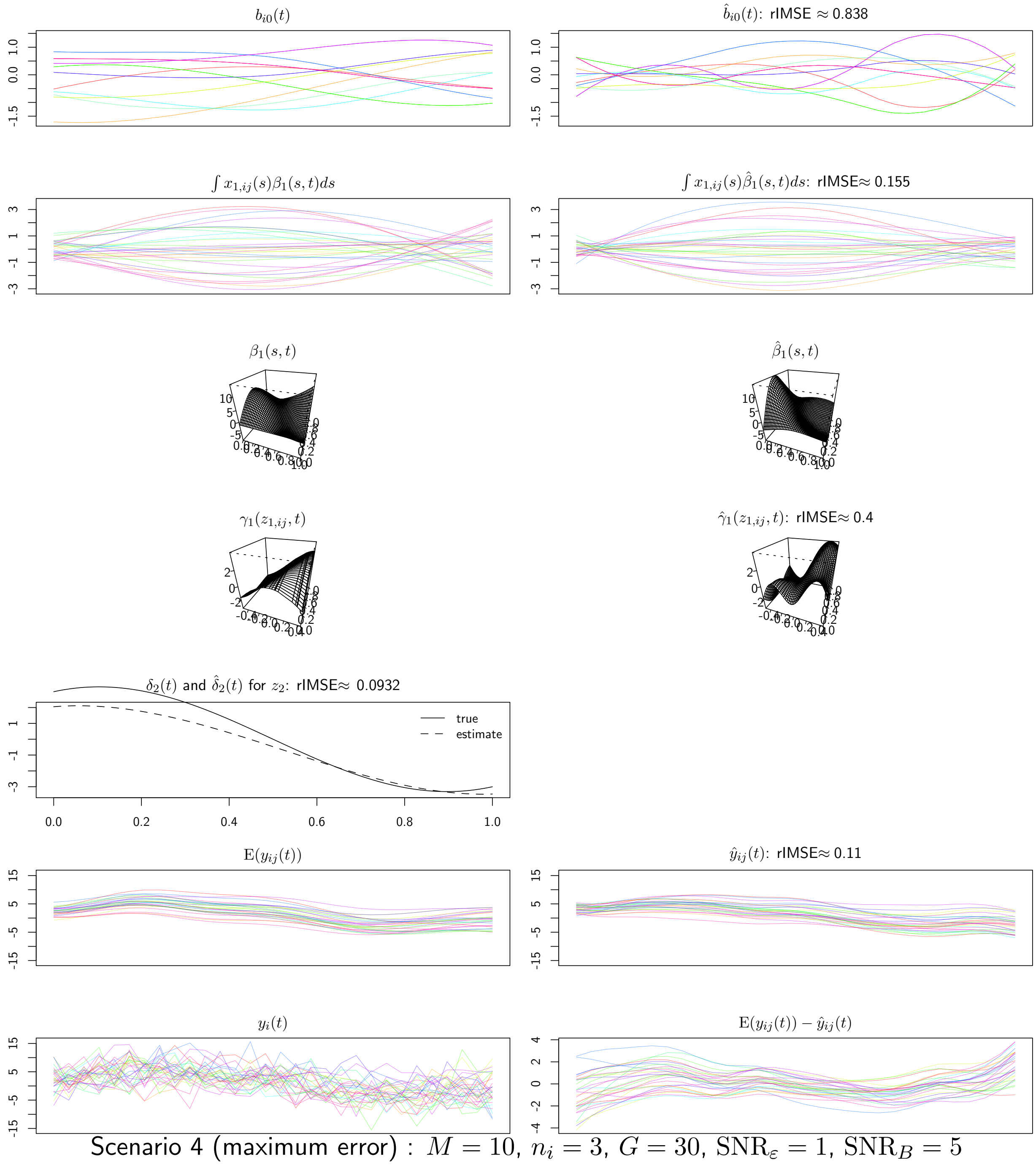


**Figure 30:** Example of data and fit for scenario 4 with maximum error.

## B.6 Comparison to WFMM

The following graphs show the IMSE ratios for the the WFMM-based results (Herrick, 2013) tested for scenario 1, i.e., the WFMM errors (and computation times) divided by those for pffr. We can only provide this comparison for scenario 1 as the other scenarios feature terms that are not possible to include in WFMM, which can only fit random effect curves and functional linear effects $z_{ij}\beta(t)$ of scalar covariates $z$. Note that, differing from the results for pffr in the main article, these results are for balanced data, as the WFMM algorithm seems to fail whenever there are any subjects with 1 or 2 observations only, and 10 replicates per setting. In general, the IMSEs for WFMM are about double to three times those of pffr. Note, however, that this comparison is not entirely fair to WFMM, as it was designed for spiky data from spectrometry (i.e., it assumes sparsity in a suitable wavelet domain), not the very smooth functional data we simulate here. Results for WFMM are based on the default settings with Daubechies tap 4 wavelets, with the exception of an increased burn-in time of 2000 iterations and a longer sampling phase (10000 iterations keeping every tenth, not 5000 keeping every fifth.). Results for other wavelet bases (Haar, Symmlet) were very similar.

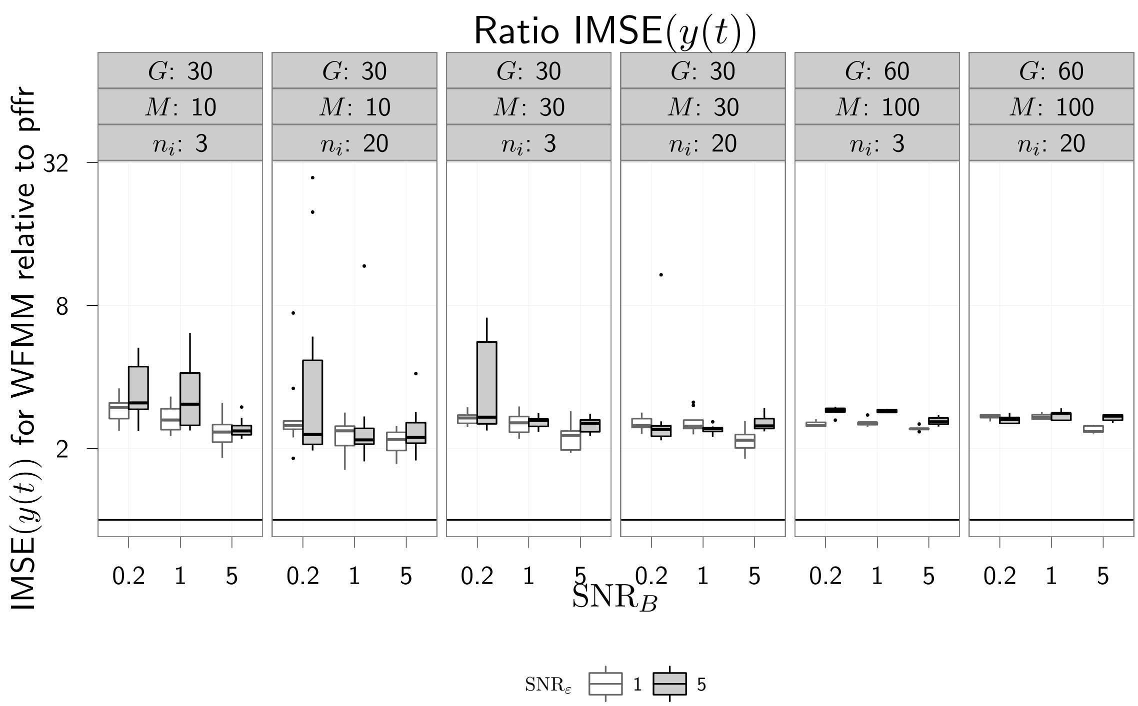


**Figure 31:** rIMSE for $y_{ij}(t)$ for WFMM divided by pffr. Vertical axis on $\log_2$-scale.

Figure 34 shows that, although WFMM is much slower (4- to 16-fold) than pffr for small and intermediate data sizes, it scales much better than pffr to large data sets due to its efficient data representation in the wavelet domain and its very fast C++ implementation.

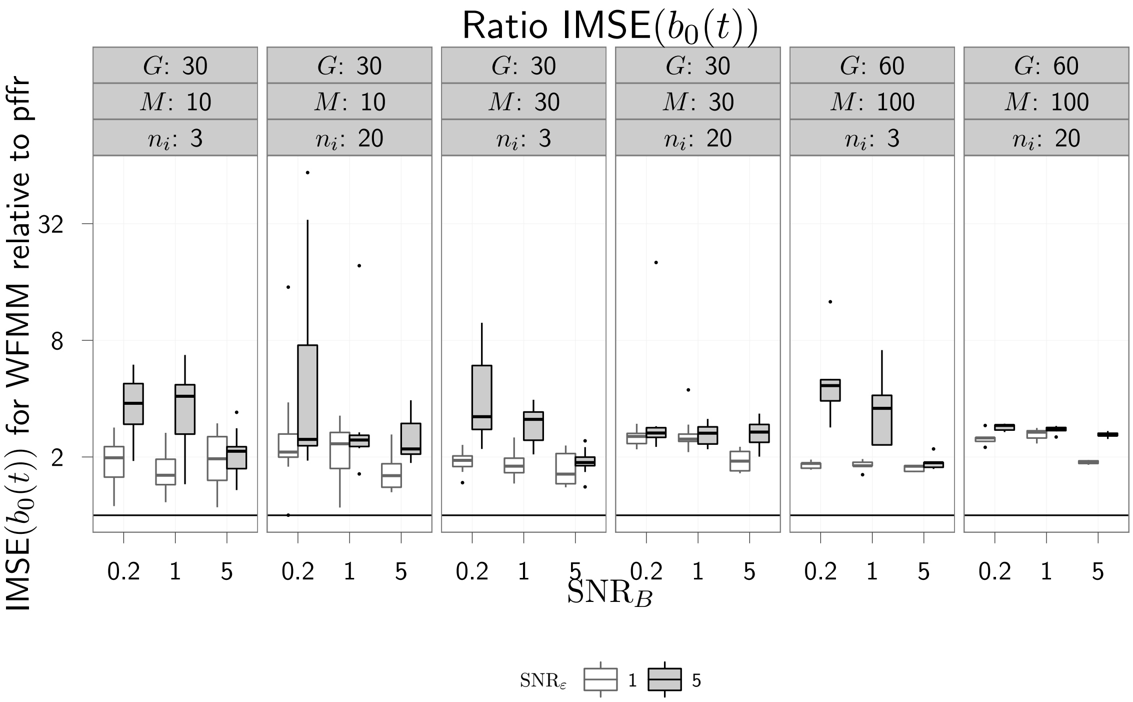


**Figure 32:** rIMSE for $b_{0i}(t)$ for `WFMM` divided by `pffr`. Vertical axis on $\log_2$-scale.

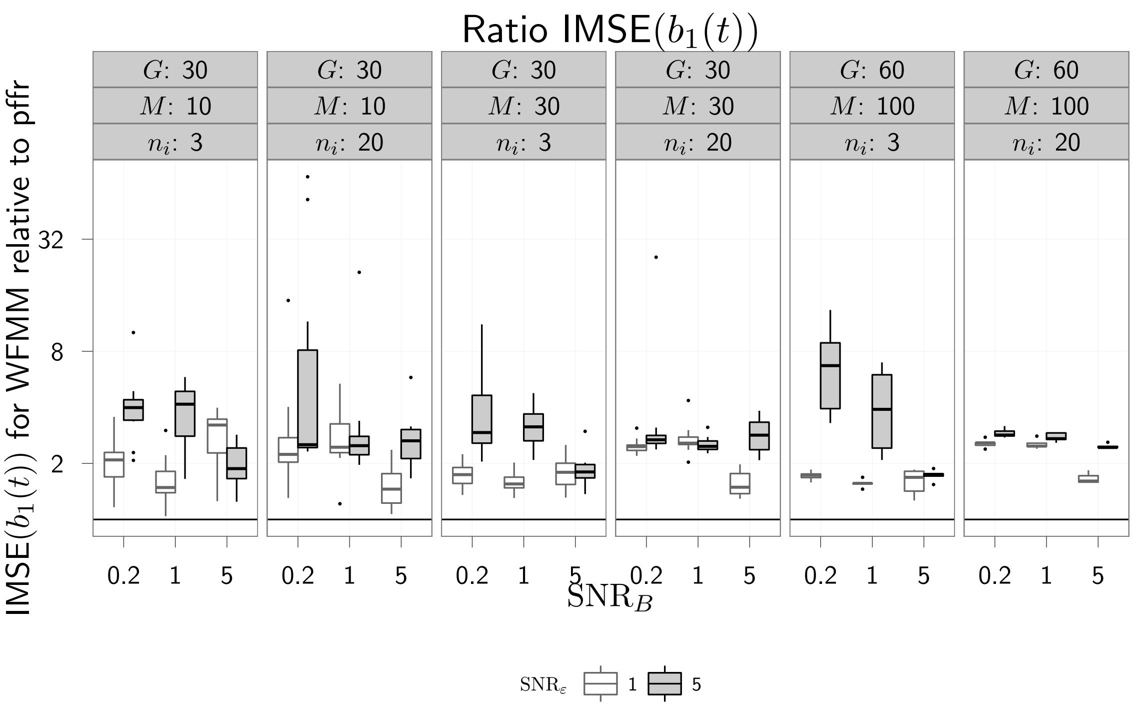


**Figure 33:** rIMSE for $b_{1i}(t)$ for `WFMM` divided by `pffr`. Vertical axis on $\log_2$-scale.

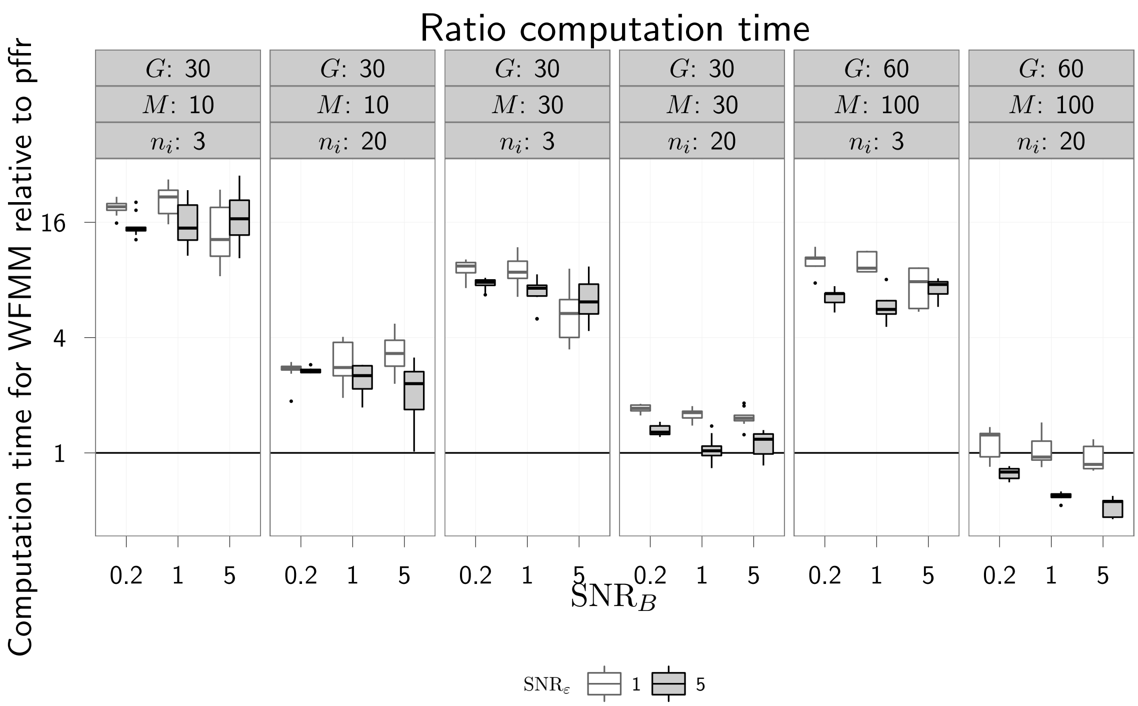


**Figure 34:** Computation times for `WFMM` divided by `pffr`. Vertical axis on $\log_2$-scale.

# C Predicting Precipitation Profiles from Temperature Curves for the Canadian Weather Data

This section is primarily intended to show the flexibility and performance of `pffr` on this small toy data set with some example code and graphical summaries, and not to attempt a stringent model criticism or model comparisons. R-Code used to perform the analysis is set in typewriter font and put in light-grey boxes, output returned by the R-console is indicated by `##`. Comments in the code are indicated by `###`.

```
###load data:
data(CanadianWeather)
dataM <- with(CanadianWeather,
              list(
                temp = t(monthlyTemp),
                l10precip = t(log10(monthlyPrecip)),
                lat = coordinates[,"N.latitude"],
                lon = coordinates[,"W.longitude"],
                region = factor(region),
                place = factor(place)
                ))
### correct Prince George location
### (wrong at least until fda_2.2.7):
dataM$lon["Pr. George"] <- 122.75
dataM$lat["Pr. George"] <- 53.9
### center temperature curves:
dataM$tempRaw <- dataM$temp
dataM$temp <- sweep(dataM$temp, 2, colMeans(dataM$temp))
### define function indices
month.t <- 1:12
month.s <- 1:12
```

The Canadian weather data consists of temperature and precipitation curves, measured as the monthly average over several years at 35 Canadian weather stations (see Figure 35, top). The data has been used extensively in the functional data analysis literature. As it is available as part of the R-package fda (Ramsay et al., 2011), we can make our analysis fully reproducible, the full source code for this section is in the `CanadianWeather_Long.R` file included in this supplement.

We will here focus on both the functional relationship between temperature and precipitation profiles as well as on the spatial nature of the data, clearly visible from the locations of the weather stations depicted in Figure 35 (middle left). Ramsay and Silverman (2005) propose a concurrent model to predict precipitation profiles from temperature curves, where temperature is allowed to influence log-precipitation linearly at the same time point $t$. Within the framework of our model, we can investigate more flexible functional regression models that allow for cumulative and lagged temperature effects. Additionally, we can take into account the spatial correlation structure between weather stations.

## C.1 Model 1: Time-varying smooth effect for smooth spatially correlated residual curves.

We consider the model

$$y_i(t) = g_0(t) + \gamma(c_i, d_i, t) + b_{0i} + \int x_i(s)\beta(s,t)ds + \varepsilon_{it}, \quad \varepsilon_{it} \sim N(0, \sigma^2_\varepsilon),$$

where $y_i(t)$ and $X_i(t)$ denote the log-precipitation and the (centered) temperature at location $i$ and time $t$, and $c_i$ and $d_i$ denote longitude and latitude of location $i$. As the temperature curves were centered pointwise across stations and $\gamma(c_i, d_i, t)$ is constrained to sum to zero for each $t$, $g_0(t)$ indicates the mean log-precipitation curve for a station with average temperature profile. The spatio-temporal term $\gamma(c_i, d_i, t)$ yields a smooth cyclic residual curve for each station, with a spatial covariance structure induced by the bivariate spline basis for longitude and latitude and its smoothness penalty. Alternatively, it can be viewed as a smooth spatial effect that varies cyclically throughout the year. Small scale local differences in the levels of precipitation not captured by these spatially correlated residuals are modelled with a scalar random intercept $b_{0i} \overset{\text{i.i.d.}}{\sim} N(0, \sigma_b^2)$. It is important to note that the temperature and precipitation profiles considered here are averaged over several years. Our model thus does not have the problem of 'the future influencing the past', but relates general weather patterns to one another.

The temperature data is of very low rank, which can cause identifiability issues (c.f. Scheipl and Greven, 2012), and we use a small number of basis functions to reflect the low information content of the data:

```
### check effective rank of covariance of temperature deviations:

cov.temp <- crossprod(dataM$temp)
ev.cov.temp <- eigen(cov.temp)$values
cumsum(ev.cov.temp)/sum(ev.cov.temp)
##  [1] 0.89238 0.97506 0.99335 0.99800 0.99892 0.99949 0.99976
##  [8] 0.99984 0.99991 0.99996 0.99999 1.00000
### first 4 eigenfunctions represent >.995 of total variability
```

```
 B <- smooth.construct.cc.smooth.spec(
        object=list(term="month.t", bs.dim=4, fixed=FALSE, dim=1,
                    p.order=NA, by=NA),
        data=list(month.t=1:12),knots=list())
 N.P <- B$X %*% Null(B$S[[1]])
 N.X <- svd(t(dataM$temp))$u[,-(1:4)]
 getSpandDist(svd(N.P)$u, N.X)
 # 0.99676
```

To take into account the cyclic nature of both the response and the predictor curves, we use cyclic basis functions in both $s$ and $t$ direction. The model can then be fit using the `pffr()` function in the refund package (Crainiceanu et al., 2011) as

```
mM <- pffr(l10precip ~ s(lat,lon) + c(s(place, bs="re")) +
```

```
  ff(temp, yind=month.t, xind=month.s,
     splinepars=list(bs=c("cc", "cc"), k=c(4, 4)),
     check.ident=FALSE),
           bs.int = list(bs = "cc", k=10), bs.yindex = list(bs="cc"),
           data=dataM, yind = month.t,
           knots=list(month.t.vec=c(0.5,12.5), temp.tmat=c(0.5,12.5),
                      temp.smat=c(0.5,12.5)))
summary(mM)
##
## Family: gaussian
## Link function: identity
##
## Formula:
## l10precip ~ s(lat, lon) + c(s(place, bs = "re")) + ff(temp, yind = month.t,
##     xind = month.s, splinepars = list(bs = c("cc", "cc"), k = c(4,
##         4)), check.ident = FALSE)
##
## Constant coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.20852    0.00228    91.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Smooth terms & functional coefficients:
##                              edf Ref.df      F p-value
## Intercept(month.t)          6.87   8.00 196.94 < 2e-16 ***
## s(lat,lon)                113.88 167.00 590.34 0.00043 ***
## c(s(place))                23.08  35.00   9.87 < 2e-16 ***
## ff(temp,month.t,month.s)    5.29   5.98   1.88 0.08539 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.981   Deviance explained = 98.8%
## REML score = -416.05  Scale est. = 0.0023164  n = 420(35 x 12)

### get estimated coefficients, predictor components, and responses:
coefs <- coef(mM, n1=100, n2=80)
## using seWithMean for  s(month.t.vec) .
## using seWithMean for  te(lat,lon,month.t.vec) .
terms <- predict(mM, type="terms")
fit <- fitted(mM)
```

Here, `l10precip` and `temp` are $35 \times 12$ matrices containing the log-precipitation and centered temperature profiles, respectively; `lat` and `lon` are vectors containing the latitude and longitude of each location, respectively, and `month.s` and `month.t` are vectors containing the months 1 to 12. A smooth overall mean function $\alpha(t)$ (i.e, `Intercept(month.t)`) is added by default. `s(lat,lon)` yields a smooth spatio-temporal surface $\gamma(c_i, d_i, t)$. `c(s(place, bs="re"))` yields a scalar random intercept $b_{0i}$ for the stations. `ff(temp, ..., splinepars=list(bs=c("cc", "cc"), k=c(4, 4)))` fits the linear functional regression term $\int x_i(s)\beta(s,t)ds$, with `bs=c("cc", "cc")` specifying a cyclic cubic regression spline basis with wrapped around penalty in both $s$ and $t$ direction and 4 marginal basis functions each (`k=c(4,4)`). We specify `check.ident=FALSE` in this case to switch off the identifiability check included in `ff()` that would warn us of the temperature data's low rank. `bs.int = list(bs = "cc", k=10)` and `bs.yindex = list(bs="cc")` specify a cyclic spline basis $\boldsymbol{\Phi_t}$ for the intercept curve and all other terms that are functions in $t$. The `knots`-statement specifies the timepoints at which cyclic basis functions are "wrapped around", i.e. timepoint 0.5 is equivalent to timepoint 12.5 in this case.

Most of the variation in the data is explained by the effect of temperature, followed by the spatially varying functional random intercept $\gamma(c_i, d_i, t)$ and the scalar random intercept. The estimated overall mean function (Figure 35, bottom left) shows a seasonal pattern, with slightly lower precipitation in the spring and higher precipitation in the fall, above and beyond what is explained by the effect of temperature.

The effect of temperature on log-precipitation is shown in Figure 35 (bottom right). In $s$ direction, a clear seasonal pattern is visible, with higher temperatures in winter and especially autumn associated with an increase of precipitation, and higher temperatures in the spring and and summer asssociated with a decrease of precipitation. In $t$ direction, effects are much stronger for the winter months, and much weaker for the summer months, especially the effect of winter temperatures on precipitation in the middle of the year; a plausible result.

The spatially correlated smooth residual plus the scalar random intercept for each weather station is depicted in Figure 35 (middle right – see Figure 36 for a version without overlapping). It shows some interesting local features, which clearly illustrate that regional effects cannot capture the spatially varying structure of precipitation curves adequately. For example, the Arctic station Inuvik in the very north-west shows a similar error pattern to Continental stations in the north-west (precipitation higher in winter and lower in the summer and especially in spring). These northern Continental stations, in turn, exhibit a pattern which is completely different than that of the more southern Continental stations (higher in the summer and lower in the winter). Another perspective on $\gamma(c_i, d_i, t)$ is to view it as a smoothly time-varying surface estimate for a spatial effect. Figure 37 displays the temporal evolution of $\gamma(c_i, d_i, t)$ over the year. We can distinguish essentially two phases, an autumn/winter phase (top row, left three panels of bottom row) with higher precipitation in the coastal regions and lower precipitation in the interior than what can be explained by the mean temperature deviations, followed by a short transition phase and then a late spring/summer phase with increased precipitation in the interior, especially in the south and relative to the Atlantic Coast region. The right panel in the bottom row shows BLUPS for the estimated uncorrelated scalar random intercepts $b_{i0}$ on the same vertical scale as the remainder of the panels. Note the fairly high small-scale variability in mean precipitation levels that has about the same magnitude as the smooth time-varying spatial effect.

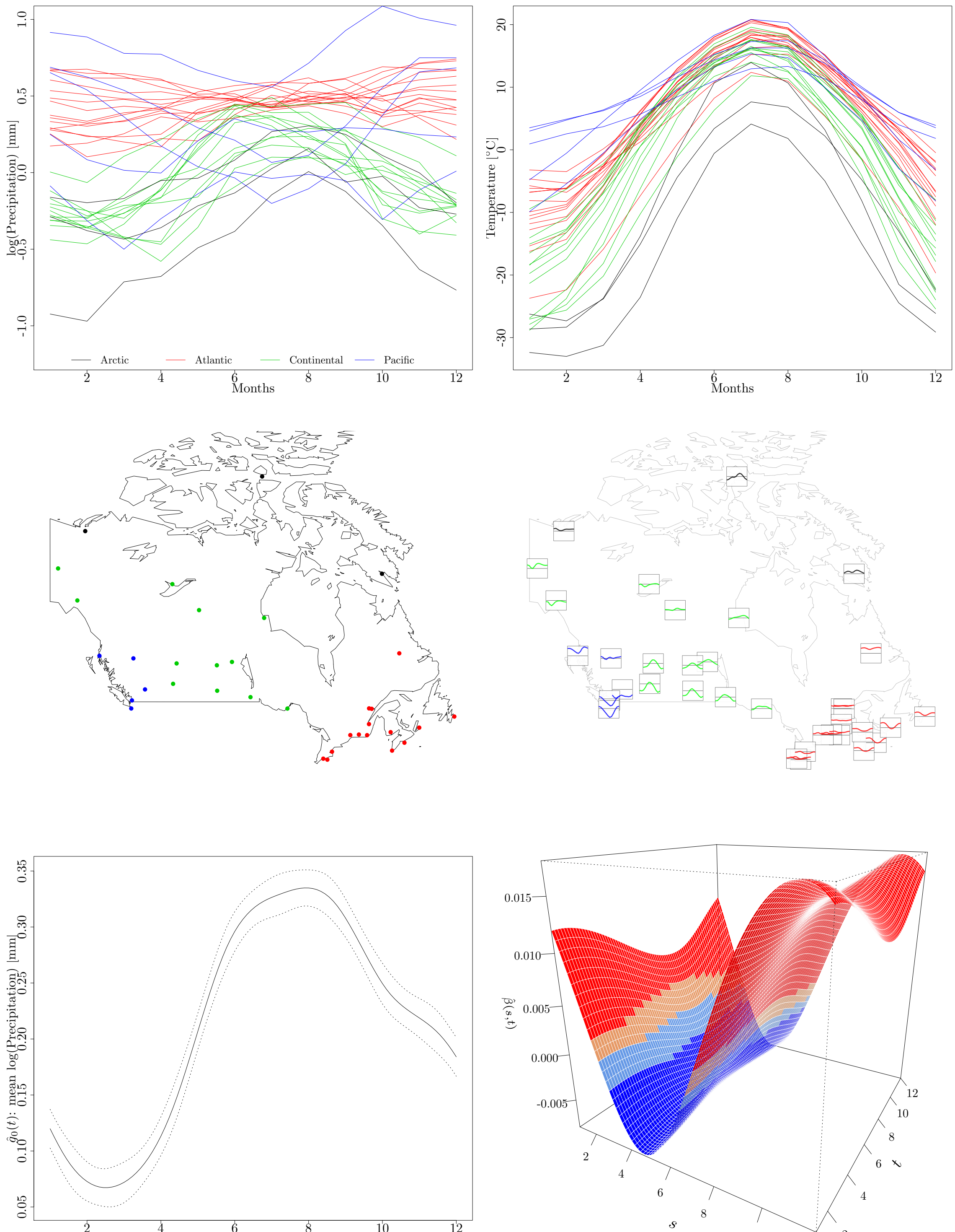


**Figure 35:** Log-precipitation (top left) and temperature (top right) at 35 Canadian weather stations. Middle left: Stations' locations. Middle right: Estimated spatially correlated smooth residual for each weather station. Bottom left: Estimated overall mean effect. Bottom right: Estimated functional effect $\hat{\beta}(s,t)$ of temperature in month $s$ on log-precipitation in month $t$, color-coded for sign and pointwise significance (95%): blue if significantly $< 0$, lightblue if $< 0$, lightred if $> 0$, red if sig. $> 0$.

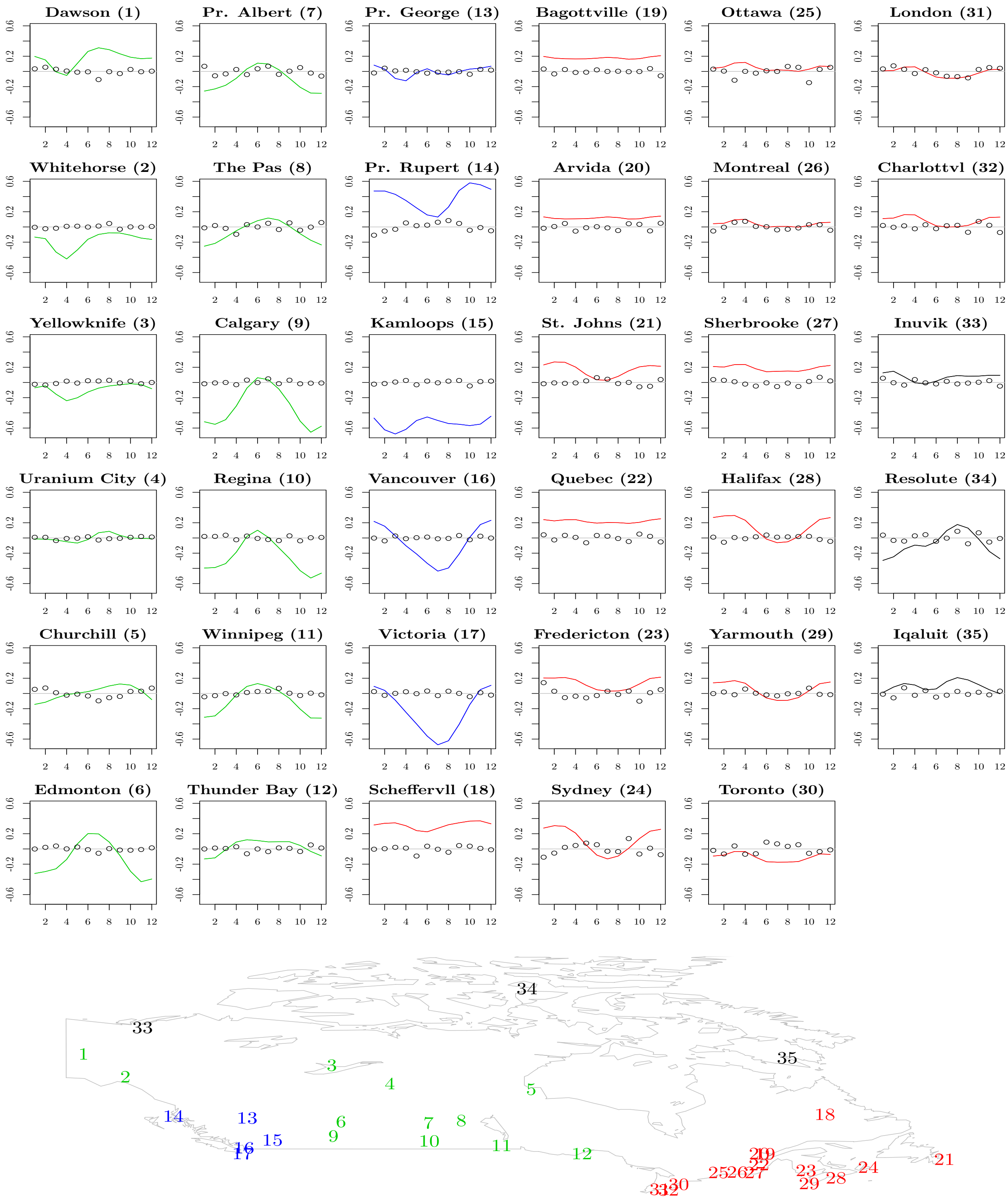


**Figure 36:** Solid lines: Spatially correlated smooth residual curves plus scalar random intercept for each weather station. Points: Observed errors $y_i(t) - \hat{y}_i(t)$. Stations are roughly ordered from north-west to south-east within regions. Color coding is as in Figure 35.

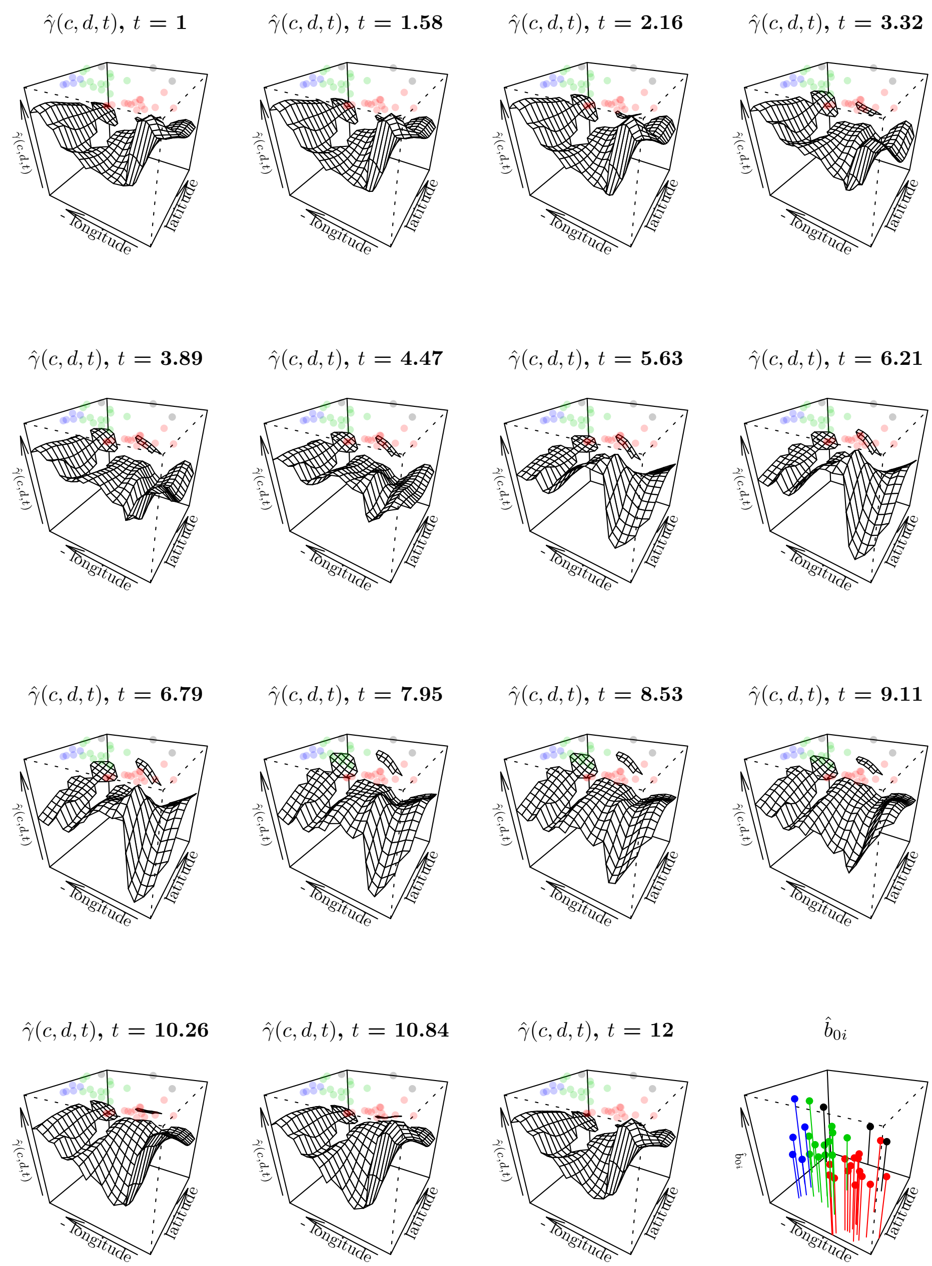


**Figure 37:** Time-varying spatial effect $\hat{\gamma}(c_i, d_i, t)$ over the course of the year and scalar random intercepts $\hat{b}_{0i}$. Station locations given by the dots at the top of the plots.

## C.2 Model 2: Time-varying regional effects and spatially un-correlated residual curves

For comparison, a simpler model ignoring the spatial correlation

$$y_i(t) = \alpha_{g_i}(t) + \int x_i(s)\beta(s,t)ds + e_i(t) + \varepsilon_{it}, \quad \varepsilon_{it} \sim N(0, \sigma_\varepsilon^2)$$

could be fit. Here, $g_i$ indicates which of the four climate regions (Atlantic, Continental, Pacific and Arctic) $i$ belongs to and $\alpha_{g_i}(t)$ thus denotes a region-specific intercept curve. $e_i(t)$ is a location-specific smooth residual centered at zero for each $t$. This model can be fit via

```
mR <- pffr(l10precip ~  0 + c(0) + region + c(region) +
  s(place, bs="re") +
  ff(temp, yind=month.t, xind=month.s,
     splinepars=list(bs=c("cc", "cc"), k=c(4, 4)),
     check.ident=FALSE),
           bs.int = list(bs = "cc", k=10), bs.yindex = list(bs="cc"),
           data=dataM, yind = month.t,
           knots=list(month.t.vec=c(0.5,12.5), temp.tmat=c(0.5,12.5),
                      temp.smat=c(0.5,12.5)))
### get estimated coefficients, predictor components, and responses:
coefsR <- coef(mR, n1=100, n2=80)
termsR <- predict(mR, type="terms")
fitR <- fitted(mR)
```

Specifically, `region + c(region)` yields time-varying region effects not centered at zero, made all estimable by dropping the constant and time-varying intercepts via `0 + c(0)`, and `s(place, bs = "re")` is used to obtain location-specific smooth residuals $e_i(t)$. In the notation of section 2.4 in the main article, this corresponds to estimating a functional random intercept for each observation $i$, with no inter-subject correlation ($\boldsymbol{P}_x = \boldsymbol{I}_{35}$).

Estimated regional effects and the coefficient surface for temperature deviations are displayed in Figure 38. Estimated temperature effects are very similar for the two models, indicating that either model formulation captures the spatial component of the responses' variability well enough to allow for a reliable estimate of the temperature effect. The spatially uncorrelated smooth residuals are shown in Figure 39. Note that they are fairly similar for closely neighboring stations despite the fact that no spatial correlation structure was assumed here.

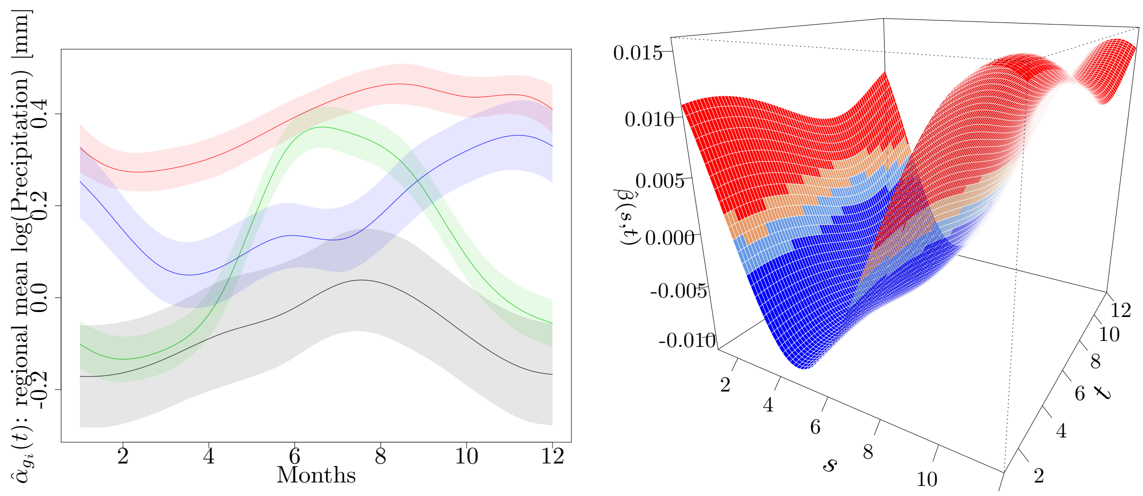


**Figure 38:** Estimated effects for a model with region-specific intercepts and spatially uncorrelated smooth residuals. Left: Estimated region effects and approximate pointwise 95% confidence intervals (Arctic, Atlantic, Continental, Pacific). Right: Estimated functional effect $\hat{\beta}(s, t)$ of temperature in month $s$ on log-precipitation in month $t$, color-coded for sign and pointwise significance (95%): blue if significantly $< 0$, lightblue if $< 0$, lightred if $> 0$, red if sig. $> 0$.

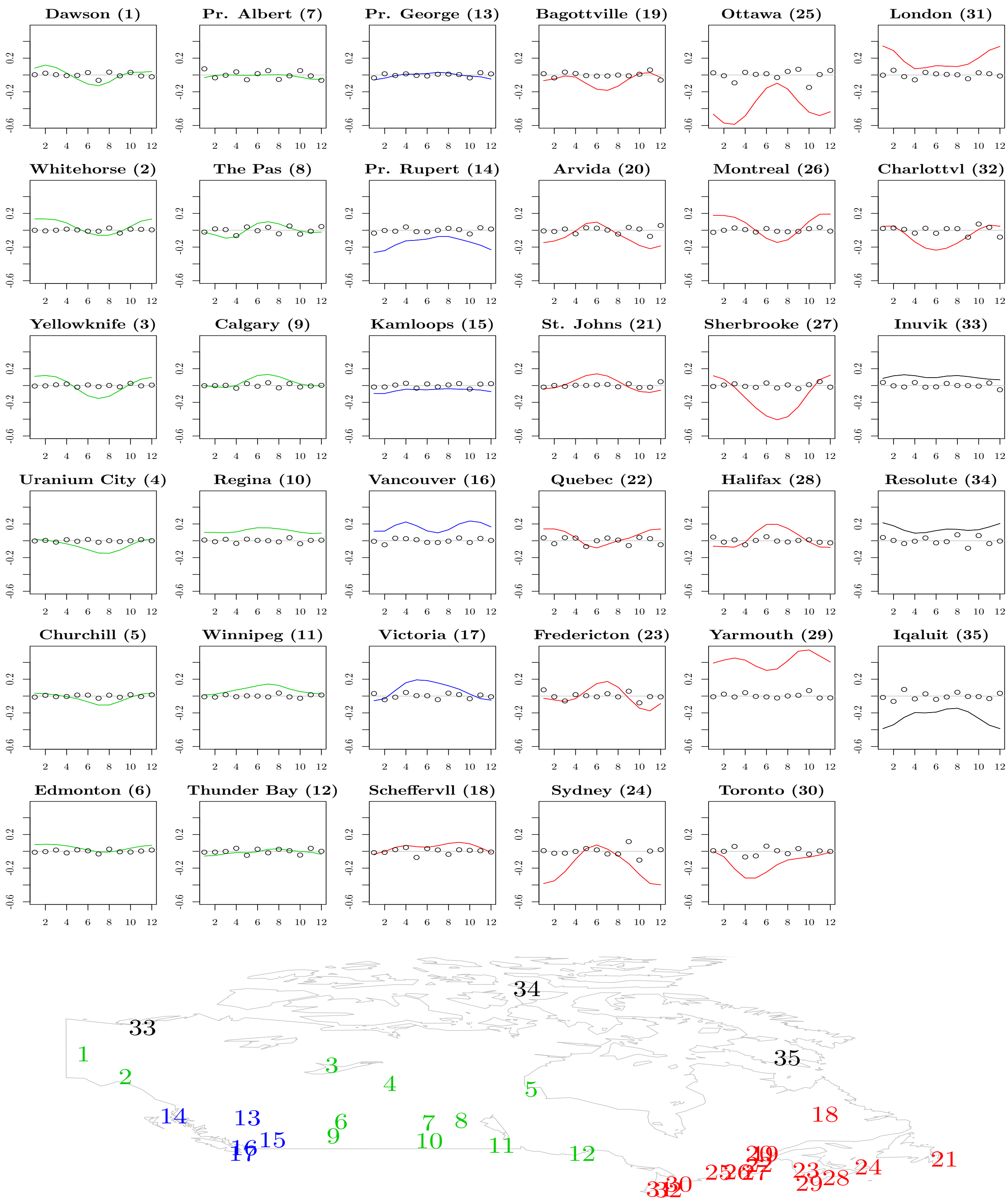


**Figure 39:** Solid lines: Spatially uncorrelated smooth residual curves for each weather station for the model with regional effects. Points: Observed errors $y_i(t) - \hat{y}_i(t)$. Stations are roughly ordered from north-west to south-east within regions. Color coding as in previous figures.

## C.3 Models 3-6: Time-varying regional effects and spatially correlated residual curves with fixed correlation structures

We fit a model

$$y_i(t) = \alpha_{g_i}(t) + \int x_i(s)\beta(s,t)ds + e_i(t) + \varepsilon_{it}, \quad \varepsilon_{it} \sim N(0, \sigma^2_\varepsilon)$$

with a marginal spatial correlation structure for the smooth residual curves $e_i(t)$ to encourage similarity of residual curves for stations that are in close proximity. In `pffr`, we can achieve this by using the inverse of a given correlation matrix as the marginal precision for a Gaussian random field across the different stations. In the notation of section 2.4 in the main article, we now use the inverse of a spatial correlation matrix of the stations as $\boldsymbol{P}_x$.

```
locations <- cbind(dataM$lon, dataM$lat)
### fix location names s.t. they correspond to levels in places
rownames(locations) <- as.character(dataM$place)
### get great circle distances between locations:
dist <- rdist.earth(locations, miles=FALSE, R=6371)
```

For example, we could choose smoothness parameters $\nu = .5$ (i.e, the exponential correlation function), $\nu = 1$ and $\nu = 10$ and range parameters chosen so that the correlation drops to about 0.2 for great circle distances of 500 km, 1500 km and 3000 km, respectively:

```
### construct Matern correlation matrices as
### marginal penalty for a GRF over the locations:

### find ranges for nu = .5, 1 and 10
### where the correlation drops to .2 at a distance of 500/1500/3000 km
### (about the 10%/40%/70% quantiles of distances here)
r.5 <- Matern.cor.to.range(500, nu=0.5, cor.target=.2)
r1 <- Matern.cor.to.range(1500, nu=1.0, cor.target=.2)
r10 <- Matern.cor.to.range(3000, nu=10.0, cor.target=.2)
### compute correlation matrices
corr_nu.5 <- apply(dist, 1, Matern, nu=.5, range=r.5)
corr_nu1 <- apply(dist, 1, Matern, nu=1, range=r1)
corr_nu10 <- apply(dist, 1, Matern, nu=10, range=r10)

### invert to get precisions
P_nu.5 <- solve(corr_nu.5)
P_nu1 <- solve(corr_nu1)
P_nu10 <- solve(corr_nu10)
```

To fit models where entry $(i,j)$ in $(\boldsymbol{P}_x)^{-1}$ is $\rho(d((c_i,d_i),(c_j,d_j));\nu,\text{range})$, with latitudes $c_i$ and longitudes $d_i$, great circle distance function $d()$, and Matèrn correlation function $\rho(d;\nu,\text{range})$ we specify an `mrf`-term ("Markov" random field, actually a conventional Gaussian random field in this case) for the different stations, and supply the inverse of the Matèrn correlation matrix as the precision of the Markov random field, i.e. we specify `s(place, bs="mrf", k=35, xt=list(list(penalty=P.nu1)))`. Specifying the `k`-argument to equal the number of locations avoids the low-rank approximation of the MRF used by default in mgcv, because this approximation only worked for positive *semi*-definite precision matrices at the time of writing.

```
mR_nu.5 <- pffr(l10precip ~  0 + c(0) + region + c(region) +
```

```
  s(place, bs="mrf", k=35, xt=list(list(penalty=P_nu.5))) +
  ff(temp, yind=month.t, xind=month.s,
     splinepars=list(bs=c("cc", "cc"), k=c(4, 4)),
     check.ident=FALSE),
                bs.int = list(bs = "cc", k=10), bs.yindex = list(bs="cc"),
                data=dataM, yind = month.t,
                knots=list(month.t.vec=c(0.5,12.5), temp.tmat=c(0.5,12.5),
                           temp.smat=c(0.5,12.5)))
summary(mR_nu.5)
##
## Family: gaussian
## Link function: identity
##
## Formula:
## l10precip ~ 0 + c(0) + region + c(region) + s(place, bs = "mrf",
##     k = 35, xt = list(list(penalty = P_nu.5))) + ff(temp, yind = month.t,
##     xind = month.s, splinepars = list(bs = c("cc", "cc"), k = c(4,
##         4)), check.ident = FALSE)
##
## Constant coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## regionArctic      -0.06775    0.01134   -5.97  8.7e-09 ***
## regionAtlantic     0.37988    0.00400   94.90  < 2e-16 ***
## regionContinental  0.09383    0.00444   21.11  < 2e-16 ***
## regionPacific      0.17714    0.00945   18.75  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Smooth terms & functional coefficients:
##                               edf Ref.df     F p-value
## regionArctic(month.t)        2.66   8.00  0.96 7.4e-05 ***
## regionAtlantic(month.t)      5.84   8.00 11.52 5.3e-12 ***
## regionContinental(month.t)   7.17   8.00 14.27 < 2e-16 ***
## regionPacific(month.t)       5.18   8.00  2.30 4.8e-07 ***
## s(place)                   160.66 237.00 26.63 < 2e-16 ***
## ff(temp,month.t,month.s)     5.45   5.48 71.46 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.986   Deviance explained = 99.3%
## REML score = -427.57  Scale est. = 0.0016962  n = 420(35 x 12)
```

```
mR_nu1 <- pffr(l10precip ~  0 + c(0) + region + c(region) +
```

```
  s(place, bs="mrf", k=35, xt=list(list(penalty=P_nu1))) +
  ff(temp, yind=month.t, xind=month.s,
     splinepars=list(bs=c("cc", "cc"), k=c(4, 4)),
     check.ident=FALSE),
               bs.int = list(bs = "cc", k=10), bs.yindex = list(bs="cc"),
               data=dataM, yind = month.t,
               knots=list(month.t.vec=c(0.5,12.5), temp.tmat=c(0.5,12.5),
                          temp.smat=c(0.5,12.5)))
summary(mR_nu1)
##
## Family: gaussian
## Link function: identity
##
## Formula:
## l10precip ~ 0 + c(0) + region + c(region) + s(place, bs = "mrf",
##     k = 35, xt = list(list(penalty = P_nu1))) + ff(temp, yind = month.t,
##     xind = month.s, splinepars = list(bs = c("cc", "cc"), k = c(4,
##         4)), check.ident = FALSE)
##
## Constant coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## regionArctic      -0.06184    0.01158   -5.34  2.2e-07 ***
## regionAtlantic     0.37963    0.00397   95.56  < 2e-16 ***
## regionContinental  0.09528    0.00446   21.35  < 2e-16 ***
## regionPacific      0.17087    0.00981   17.41  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Smooth terms & functional coefficients:
##                               edf Ref.df      F p-value
## regionArctic(month.t)        2.95   8.00   1.39 6.1e-06 ***
## regionAtlantic(month.t)      5.82   8.00  15.71 1.3e-10 ***
## regionContinental(month.t)   7.16   8.00  13.25 < 2e-16 ***
## regionPacific(month.t)       5.19   8.00   2.48 6.2e-07 ***
## s(place)                   160.82 237.00  27.39 < 2e-16 ***
## ff(temp,month.t,month.s)     5.24   5.29 102.34 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.987   Deviance explained = 99.3%
## REML score = -432.11  Scale est. = 0.0016658  n = 420(35 x 12)
```

```
mR_nu10 <- pffr(l10precip ~  0 + c(0) + region + c(region) +
```

```
  s(place, bs="mrf", k=35, xt=list(list(penalty=P_nu10))) +
  ff(temp, yind=month.t, xind=month.s,
     splinepars=list(bs=c("cc", "cc"), k=c(4, 4)),
     check.ident=FALSE),
                bs.int = list(bs = "cc", k=10), bs.yindex = list(bs="cc"),
                data=dataM, yind = month.t,
                knots=list(month.t.vec=c(0.5,12.5), temp.tmat=c(0.5,12.5),
                           temp.smat=c(0.5,12.5)))
summary(mR_nu10)
##
## Family: gaussian
## Link function: identity
##
## Formula:
## l10precip ~ 0 + c(0) + region + c(region) + s(place, bs = "mrf",
##     k = 35, xt = list(list(penalty = P_nu10))) + ff(temp, yind = month.t,
##     xind = month.s, splinepars = list(bs = c("cc", "cc"), k = c(4,
##         4)), check.ident = FALSE)
##
## Constant coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## regionArctic      -0.06119    0.01182   -5.18  4.6e-07 ***
## regionAtlantic     0.37996    0.00406   93.64  < 2e-16 ***
## regionContinental  0.09544    0.00456   20.93  < 2e-16 ***
## regionPacific      0.16910    0.01004   16.84  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Smooth terms & functional coefficients:
##                              edf Ref.df     F p-value
## regionArctic(month.t)        2.79   8.00  1.53 3.9e-06 ***
## regionAtlantic(month.t)      5.77   8.00 16.06 3.5e-10 ***
## regionContinental(month.t)   7.08   8.00 13.63 < 2e-16 ***
## regionPacific(month.t)       5.54   8.00  2.97 6.0e-08 ***
## s(place)                   140.22 235.00 67.32 1.2e-11 ***
## ff(temp,month.t,month.s)     5.99   6.28 78.56 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.986   Deviance explained = 99.2%
## REML score = -417.28  Scale est. = 0.0017421  n = 420(35 x 12)
```

As the summaries show, the fits are very similar and robust against different specifications of the spatial correlation structure.

For a less arbitrary approach, we can try numerical optimization to find Matèrn parameters for the correlation function that maximize the model's likelihood:

```
### define optimization criterion:
```

```
optll <- function(par){
  nu <- par[1]
  range <- par[2]
  ### construct Matern correlation matrix as
  ### marginal penalty for a GRF over the locations:
  corr <- apply(dist, 1, Matern, nu=nu, range=range)
  ### invert to get precisions
  P <<- try(solve(corr))
  ### fit model
  m <- try(pffr(l10precip ~  0 + c(0) + region + c(region) +
    s(place, bs="mrf", k=35, xt=list(list(penalty=P))) +
    ff(temp, yind=month.t, xind=month.s,
       splinepars=list(bs=c("cc", "cc"), k=c(4, 4)),
       check.ident=FALSE),
                bs.int = list(bs = "cc", k=10), bs.yindex = list(bs="cc"),
                data=dataM, yind = month.t,
                knots=list(month.t.vec=c(0.5,12.5), temp.tmat=c(0.5,12.5),
                           temp.smat=c(0.5,12.5))))
  ### return likelihood
  if(class(m)[1] != "try-error"){
    cat("nu:", nu, "--range:", range, "--ll:", logLik(m), "\n")
    return(as.numeric(logLik(m)))
  } else {
    cat("nu:", nu, "--range:", range, "--nope.\n")
    return(1500)
  }
}
### find optimal values (takes quite a long run time,
### results not entirely stable due to many local maxima.)
nurange <- optim(c(1, 3000), optll,
                 lower = c(0.1, 200), upper=c(10, 4000),
                 method = "L-BFGS-B",
                 control=list(fnscale=-1, parscale=c(1, 100),
                              factr=1e4, ndeps=c(2e-1, 2)))
# ....
#nu: 0.71885 --range: 3000.2 --ll: 876.35

### use optimal nu=0.7, range=3000
corr_nu.7 <- apply(dist, 1, Matern, nu=.7, range=3000)
### invert to get precisions
P_nu.7 <- solve(corr_nu.7)
```

```
mR_nu.7 <- pffr(l10precip ~  0 + c(0) + region + c(region) +
  s(place, bs="mrf", k=35, xt=list(list(penalty=P_nu.7))) +
  ff(temp, yind=month.t, xind=month.s,
     splinepars=list(bs=c("cc", "cc"), k=c(4, 4)),
     check.ident=FALSE),
                bs.int = list(bs = "cc", k=10), bs.yindex = list(bs="cc"),
                data=dataM, yind = month.t,
                knots=list(month.t.vec=c(0.5,12.5), temp.tmat=c(0.5,12.5),
                           temp.smat=c(0.5,12.5)))

summary(mR_nu.7)
```

It should be noted, however, that the likelihood surface is very flat in these parameters and that many other Matèrn parameters yield practically identical fits and likelihoods. Figure 40 shows the shapes of the four correlation functions – we fit one model with a very quickly decreasing correlation structure ($\nu = 0.5$), one intermediate correlation structure ($\nu = 1$), one model with a very persistent spatial correlation structure ($\nu = 10$), and one with parameters ($\nu = 0.7$, range= 3000) determined by numerical optimization yielding a very long-range correlation structure.

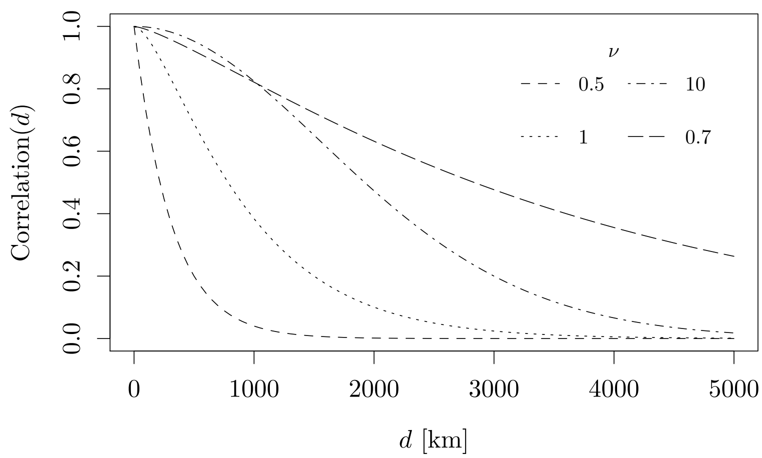


**Figure 40:** Four different Matérn correlation functions. Range parameters are 311, 624, 377, and 3000, respectively, for $\nu = 0.5$, $\nu = 1$, $\nu = 10$ and $\nu = 0.7$

Figures 41 to 43 compare the estimated effects for the model with region effects and spatially correlated random effects. As seen in figure 41, region effects are very robust against the different specifications for the smooth residual terms. Figure 42 shows that, as the spatial correlation of the smooth residuals increases (and thus the flexibility of the residual terms decreases), the temperature effect becomes somewhat larger, while retaining its shape fairly exactly. Our interpretation of this phenomenon is that, since temperature curves of course correlate strongly with the spatial locations, including more flexible spatial effects tends to attenuate the effect of temperature. The estimated spatial effects (see Figure 43) are very similar for the four different model specifications because residual curves of neighboring locations are very similar even for the model with independence assumption in this case (not shown) so that the specification of a spatial correlation structure seems to make fairly little difference in terms of the BLUPs. As expected, observed residuals $\hat{\epsilon}_{it}$ are occasionally somewhat larger for the models with stronger spatial autocorrelation (see Figure 44). The advantage of specifying a correlation structure for the random effects in this case is that it allows interpolation or prediction for previously unobserved locations and improves precision of the estimates if the specified correlation structure approximates the true one.

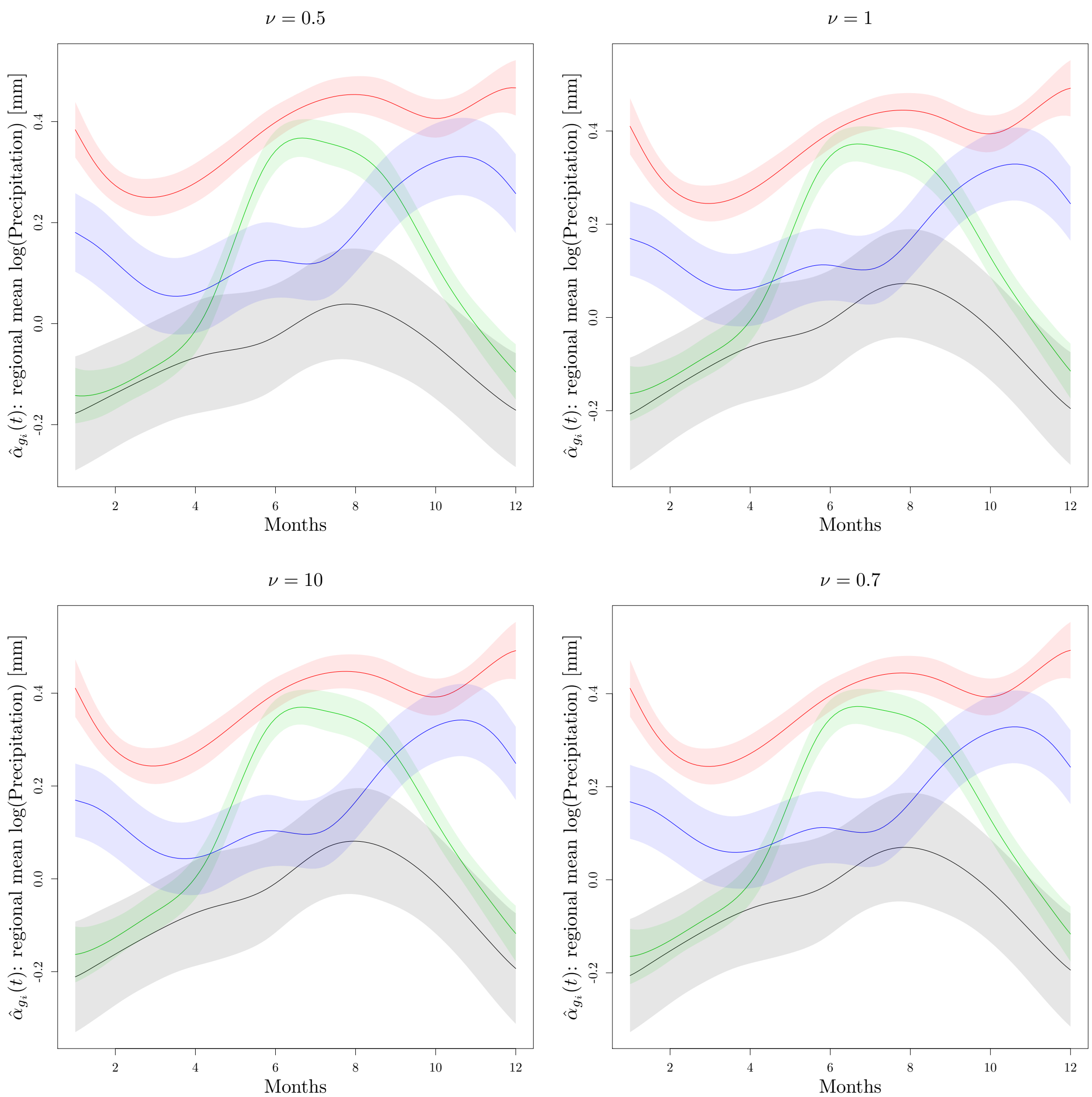


**Figure 41:** Estimated region effects for the models with spatially correlated smooth residuals.

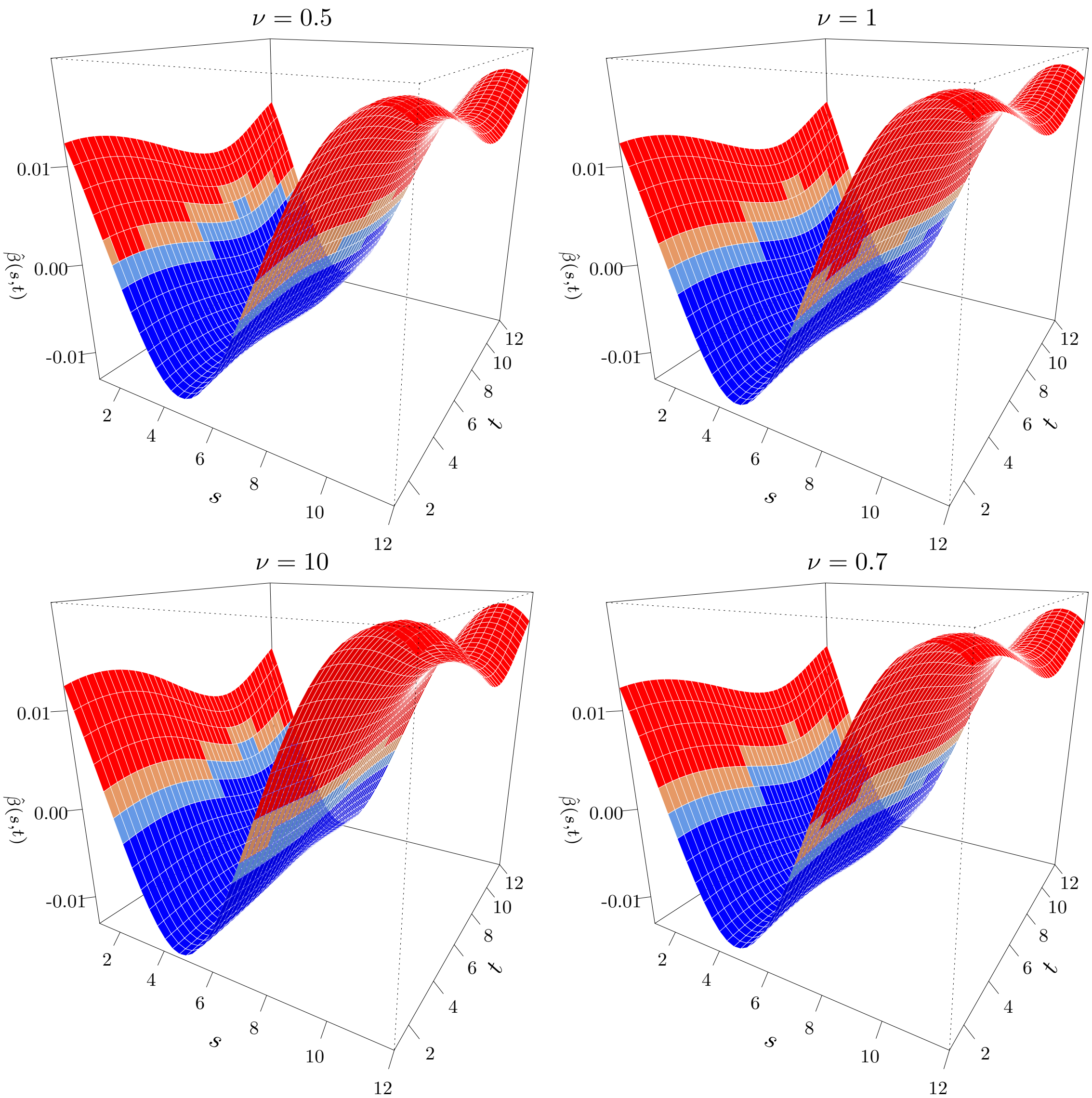


**Figure 42:** Estimated functional effect $\hat{\beta}(s,t)$ of temperature in month $s$ on log-precipitation in month $t$, color-coded for sign and pointwise significance (95%): blue if sig. $< 0$, lightblue if $< 0$, lightred if $> 0$, red if sig. $> 0$.

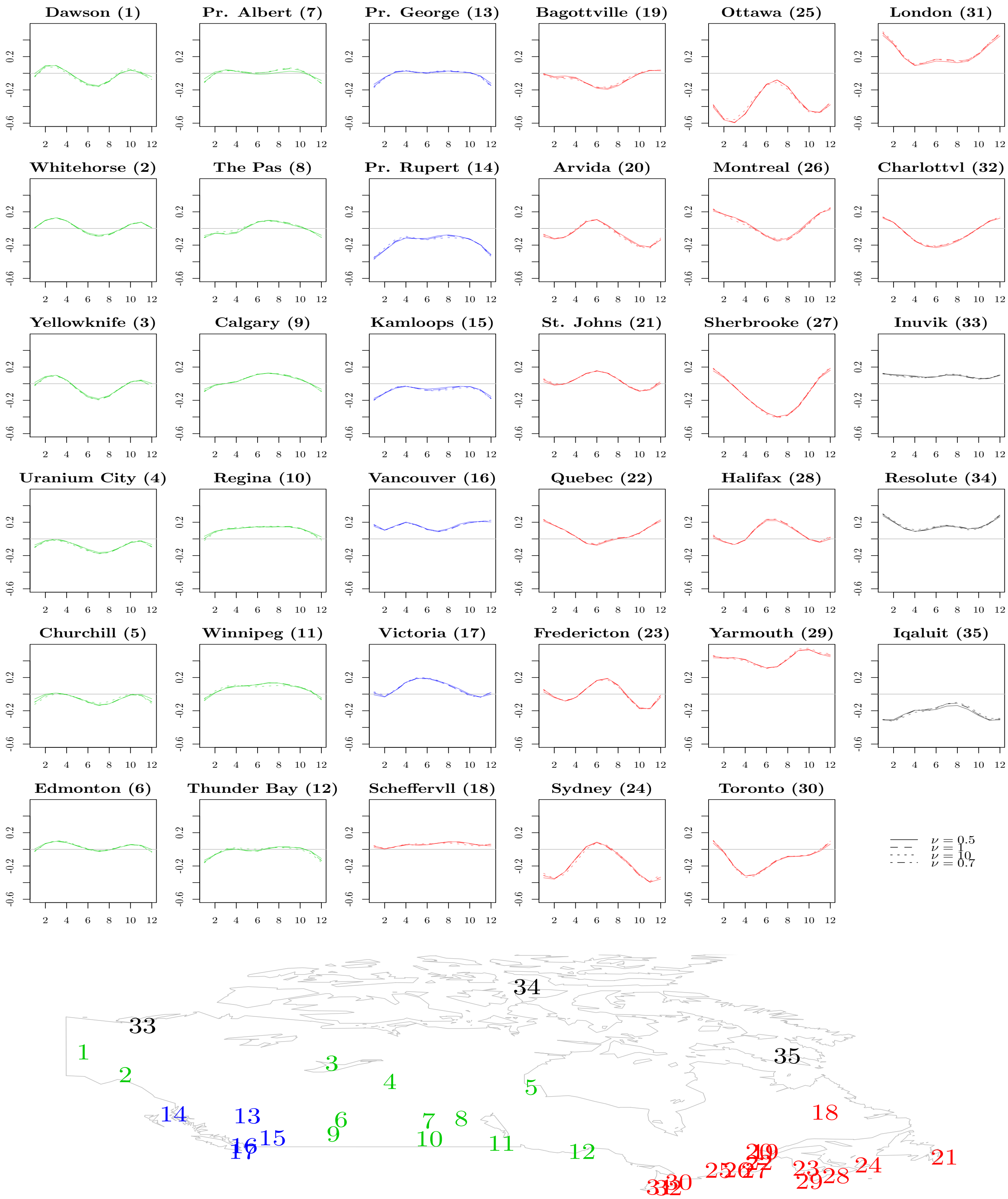


**Figure 43:** Spatially correlated random effects for each weather station for the model with regional effects. Stations are roughly ordered from north-west to south-east within regions. Color coding for regions as in previous figures, line types code for the 4 different models.

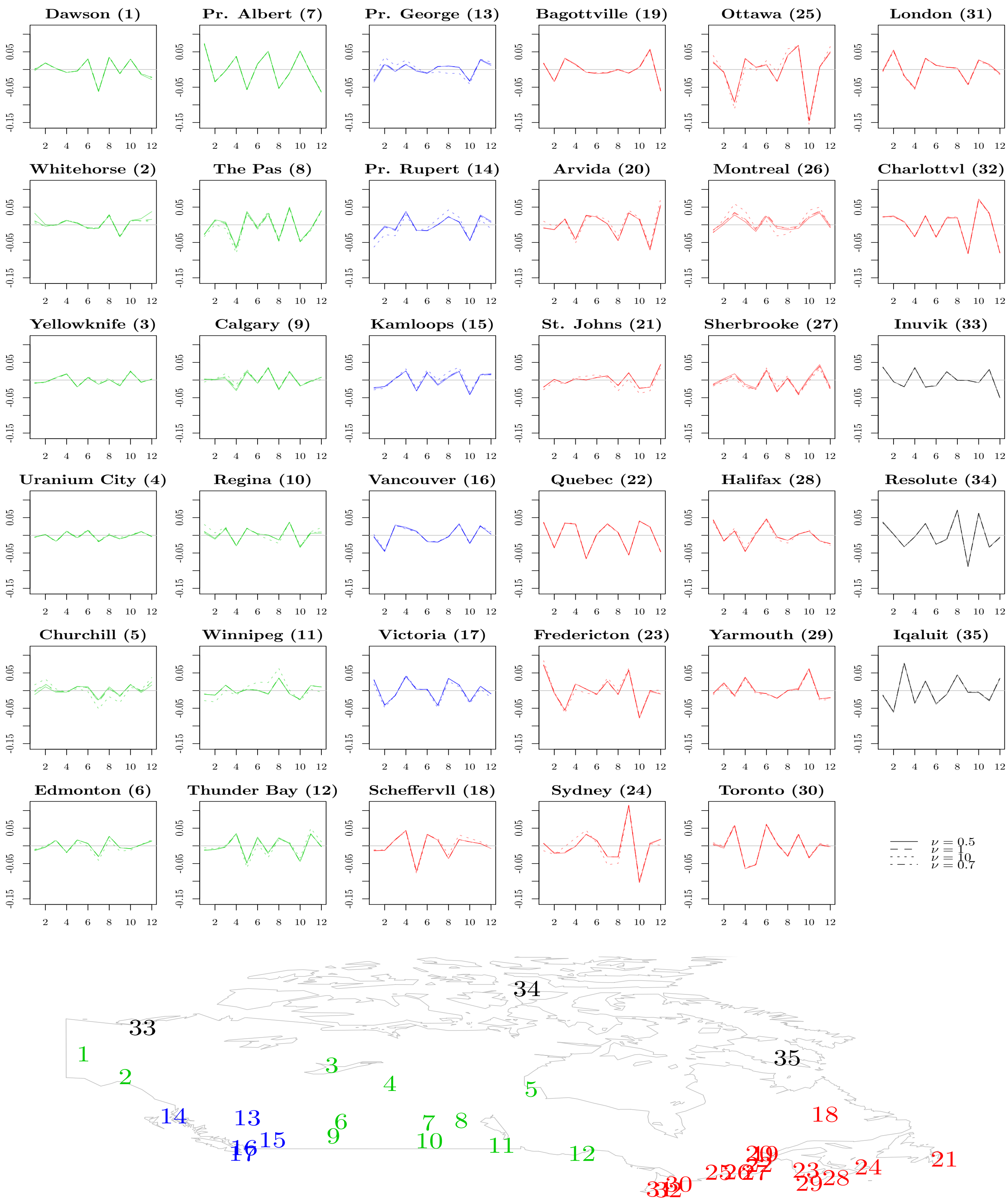


**Figure 44:** Observed residuals $y_i(t) - \hat{y}_i(t)$ for each weather station for the model with regional effects. Stations are roughly ordered from north-west to south-east within regions. Color coding for regions as in previous figures, line types code for the 4 different models.

### C.4 Model 6b: FPC-based function-on-function effect

Instead of relying on a spline-based representation of the temperature effect, we can also fit a model based on an estimated FPC decomposition $x_i(s) \approx \sum_{k=1}^{K_x} \hat{\psi}_k(s)\hat{\xi}_{ik}$ of the (centered) temperature curves, i.e. we re-parameterize $\int_{\mathcal{S}} x_i(s)\beta(s,t)ds \approx \int_{\mathcal{S}} \sum_{k=1}^{K_x} \psi_k(s)\xi_{ik}\beta(s,t)ds = \sum_{k=1}^{K_x} \xi_{ik}\tilde{\beta}_k(t)$ with $\tilde{\beta}_k(t) = \int_{\mathcal{S}} \psi_k(s)\beta(s,t)ds$. In other words, a linear function-on-function effect can be represented as a sum of varying coefficient terms for the FPC loading vectors $\hat{\boldsymbol{\xi}}_k$. In `pffr`, we can specify such an effect as an `ffpc` term. In this case, we set `npc`$= K_x = 4$. Specifications for the region effects and spatially correlated random effects are the same as in the previous subsection.

```
mR_nu.7.ffpc <- pffr(l10precip ~  0 + c(0) + region + c(region) +
  s(place, bs="mrf", k=35, xt=list(list(penalty=P_nu.7))) +
  ffpc(temp, yind=month.t, xind=month.s,
       splinepars=list(bs="cc", k=4),
       decomppars=list(npc=4)),
                     bs.int = list(bs = "cc", k=10), bs.yindex = list(bs="cc"),
                     data=dataM, yind = month.t,
                     knots=list(month.t.vec=c(0.5,12.5), temp.tmat=c(0.5,12.5),
                                   temp.smat=c(0.5,12.5)))
summary(mR_nu.7.ffpc)
```

Figure 45 displays the estimated precipitation effect from this model specification and the corresponding spline-based specification for comparison. Region effects and spatially correlated random effects are not shown, they are visually indistinguishable from those in the spline-based specification. The top row shows the first four estimated FPCs $\hat{\psi}_k(s)$ of the temperature profiles on the left and the associated coefficient function estimates $\tilde{\beta}_k(t)$ for the first four FPC loading vectors with pointwise approximate 95 % CIs. As for the coefficient surface representation in the spline-based models, it is immediately obvious that the effect of temperature on precipitation is fairly constant over $t$, as all the $\tilde{\beta}_k(t)$ are almost constant over $t$. Since the estimated FPCs are not cyclic over $s$, as would be appropriate in this setting, interpretation is a little iffy: For example, the extreme sign changes between December and January for FPCs 3 and 4 are an unavoidable artefact of the orthonormality constraint on the FPCs. FPC number 4 (blue) loads strongly on above-average temperatures in late summer and autumn and below average temperature in late winter and spring, and this seems to have a strong positive association ($\tilde{\beta}_4(t) \approx 0.04$) with increased precipitation throughout the year. FPC number 1 (orange) loads strongly on below-average temperatures in the winter months. The shape of FPC number 2 is almost a mirror image of that of FPC number 1, as is the shape of the associated $\tilde{\beta}_2(t)$, so it's hard to separate the effect of these two — both seem to be associated with less precipitation in winter and spring ($\tilde{\beta}_1(t) \approx -0.01$ for winter and spring, $\tilde{\beta}_2(t) \approx 0.01$), but not associated quite as strongly with precipitation in the summer and autumn. The bottom row shows the coefficient surface implied by $\hat{\psi}_k(s)$ and the associated $\tilde{\beta}_k(t)$ on the left and, for comparison, the coefficient surface estimate from the spline-based model. It is easy to see that both model specifications result in qualitatively very similar effect shapes. In this setting, the spline-based estimate seems more appropriate as it is able to accomodate the cyclic nature of the data at hand. That the effects of temperature in December and January on precipitation profiles have opposing signs and similar magnitude in the FPC-based estimate is a consequence of the (not quite interpretable) shapes of FPCs 3 and 4. Nevertheless, FPC-based specifications could conceivably increase interpretability in many settings.

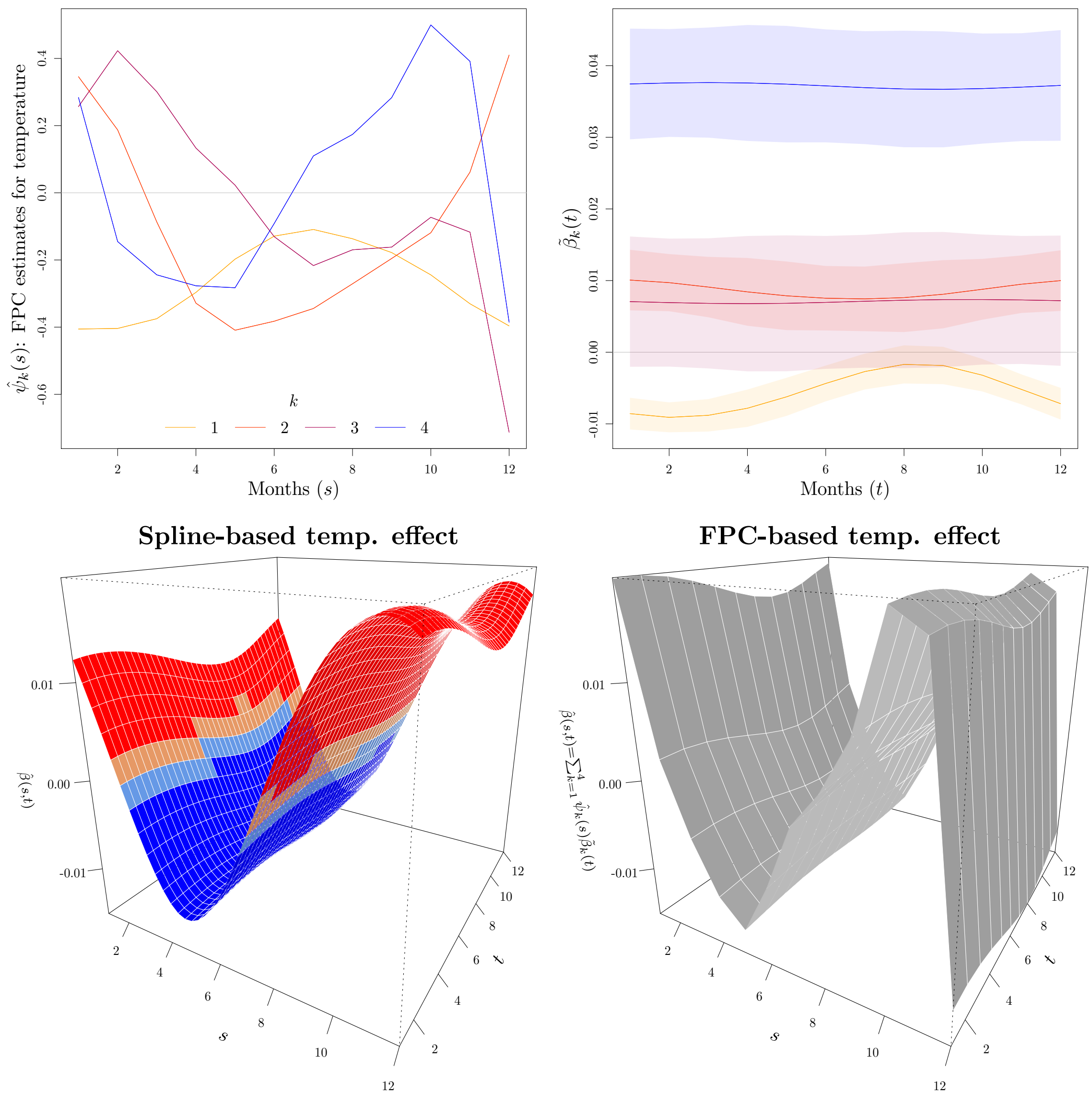


**Figure 45:** Clockwise from top left: First four estimated FPCs $\hat{\psi}_k(s)$ of the temperature profiles; Coefficient function estimates $\tilde{\beta}_k(t)$ (color-coded) for the first four FPC loading vectors (with pointwise approximate 95 % CIs); Coefficient surface implied by $\tilde{\beta}_k(t)$ and $\hat{\psi}_k(s)$; Estimated coefficient surface from the corresponding spline-based specification.

# Computational Details

This section was compiled with knitr (Xie, 2012), under the following setup:

```
sessionInfo()
## R version 3.0.1 (2013-05-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=en_US.UTF-8
##  [4] LC_COLLATE=en_US.UTF-8     LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=C                 LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] splines   stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] refund_0.1-7     boot_1.3-9       gamm4_0.1-6      lme4_0.999999-2  wavethresh_4.6.5
##  [6] nlme_3.1-110     magic_1.5-4      abind_1.4-0      glmnet_1.9-3     MASS_7.3-27
## [11] fields_6.7.6     spam_0.29-3      mapdata_2.2-2    maps_2.3-2       fda_2.3.6
## [16] Matrix_1.0-12    lattice_0.20-15  zoo_1.7-10       mgcv_1.7-24      tikzDevice_0.6.3
## [21] filehash_2.2-1   ggplot2_0.9.3.1  knitr_1.2
##
## loaded via a namespace (and not attached):
##  [1] codetools_0.2-8    colorspace_1.2-2   dichromat_2.0-0    digest_0.6.3
##  [5] evaluate_0.4.4     formatR_0.8        grid_3.0.1         gtable_0.1.2
##  [9] labeling_0.2       munsell_0.4        plyr_1.8           proto_0.3-10
## [13] RColorBrewer_1.0-5 reshape2_1.2.2     scales_0.2.3       stats4_3.0.1
## [17] stringr_0.6.2      tcltk_3.0.1        tools_3.0.1
```

The version of `pffr` used for this supplement is included in the code folder.

# References

Crainiceanu, C. M., P. T. Reiss (Coordinating authors), J. Goldsmith, S. Greven, L. Huang, and F. Scheipl (Contributors) (2011). *refund: Regression with Functional Data*. R package version 0.1-5.

Herrick, R. (2013). *WFMM* (Version 3.0 ed.). The University of Texas M.D. Anderson Cancer Center.

Ramsay, J. O. and B. W. Silverman (2005). *Functional Data Analysis*. Springer.

Ramsay, J. O., H. Wickham, S. Graves, and G. Hooker (2011). *fda: Functional Data Analysis*. R package version 2.2.7.

Scheipl, F. and S. Greven (2012). Identifiability in penalized function-on-function regression models. Technical Report 125, LMU München.

Wood, S. N. (2006). *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC.

Xie, Y. (2012). *knitr: A general-purpose package for dynamic report generation in R*. R package version 0.4.