Review of Functional Partial Least Squares

Application to Spectral Misalignment

Slides by Rory Samuels

Multiple Linear Regression Model

Suppose we have a sample of n scalar valued response variables $y_i \in \mathbb{R}$ and p corresponding predictor variables $\mathbf{x}_i \in \mathbb{R}^p$. The multiple linear regression model is given by

$$y_i = \beta_0 + \boldsymbol{\beta}'\mathbf{x}_i + \epsilon_i,$$

where β_0 is the intercept, $\boldsymbol{\beta}$ is a a vector of coefficients corresponding to the variables in $\mathbf{x}$, and $\epsilon_i \sim N(0, \sigma_\epsilon^2)$.

- Note: for simplicity we assume $\beta_0 = 0$.

Clasical Partial Least Squares (PLS)

Let $\mathbf{y}$ be the $n \times 1$ vector of responses and $\mathbf{X}$ be the $n \times p$ matrix of measured predictor variables. Given weight vectors $\mathbf{r}_1, ..., \mathbf{r}_{k-1}$, the kth PLS weight vector is obtained via

$$\arg\max_{\mathbf{r}} \mathsf{Cov}^2\left(\mathbf{y}, \mathbf{X}\mathbf{r}\right), \quad \text{subject to:}$$

$$\mathsf{Cov}\left(\mathbf{X}\mathbf{r}_m, \mathbf{X}\mathbf{r}\right) = 0 \;\; \text{for} \;\; m = 1, ..., k-1, \quad \text{and}$$

$$||\mathbf{r}|||_2^2 = 1.$$

- Many algorithms for solving efficiently (e.g. SIMPLS/NIPALS)

PLS Coefficients

Let $\mathbf{R}$ be the $p \times K$ matrix whose columns are the first K PLS empirical weight vectors $\hat{\mathbf{r}}_1, ...\hat{\mathbf{r}}_K$.

- X scores: $\mathbf{T} = \mathbf{XR}$
- Y loadings: $\boldsymbol{\alpha} = (\mathbf{T}'\mathbf{T})^{-1}\mathbf{T}'\mathbf{y}$

$$\hat{\mathbf{y}} = \mathbf{T}\boldsymbol{\alpha} = \mathbf{X}(\mathbf{R}\boldsymbol{\alpha})$$

- Coefficient: $\hat{\boldsymbol{\beta}}_{PLS} = \mathbf{R}\boldsymbol{\alpha}$

Functional Linear Regression Model

For functional valued predictors $x_i(w) \in L^2([a, b])$, the functional linear regression model (FLM) is given by

$$y_i = \beta_0 + \int_a^b x_i(w)\beta(w)dw + \epsilon_i,$$

where $\beta(w)$ is a functional valued coefficient.

- For now, we assume $x_i(w)$ are known functions
- Again assuming $\beta_0 = 0$ for notational simplicity

Functional Partial Least Squares (FPLS)

Given weight functions $r_1(w), ..., r_{k-1}(w)$, the kth FPLS weight function is given by

$$\arg\max_{r(w)} \text{Cov}^2\left(y, \int_a^b x(w)r(w)dw\right), \quad \text{subject to:}$$

$$\text{Cov}\left(\int_a^b x(w)r_m(w)dw, \int_a^b x(w)r(w)dw\right) = 0 \text{ for } m = 1, ..., k-1, \quad \text{and}$$

$$||r(w)|||_2^2 = 1.$$

Basis Expansion for Weight Functions

Let $\mathbf{B}(w) = (B_1(w), ..., B_{M+d}(w))'$ be a vector of $M + d$ B-spline basis functions of degree d defined over $M - 1$ equally spaced knots on $[a, b]$.

We can approximate the mth weight function by

$$r_m(w) \approx \mathbf{b}_m' \mathbf{B}(w),$$

where $\mathbf{b}_m$ is a vector of $M + d$ basis coefficients.

Defining the $\mathbf{U}$ Matrix

Define $u_{ij} = \int_a^b x_i(w)B_j(w)dw$ and $\mathbf{u}_i = (u_{i1}, ..., u_{i(M+d)})'$. We can approximate the needed inner-products with:

$$\int_a^b x_i(w)r(w)dw \approx \mathbf{b}'\mathbf{u}_i.$$

- For all n observations, we can define an $n \times (M+d)$ matrix $\mathbf{U}$ with elements $\mathbf{U}_{(ij)} = u_{ij}$.

Empirical FPLS Task

Given weight vectors $\mathbf{b}_1, ..., \mathbf{b}_{k-1}$, the kth FPLS weight vector is obtained via

$$\arg\max_{\mathbf{b}} \mathsf{Cov}^2\left(\mathbf{y}, \mathbf{U}\mathbf{b}\right), \quad \text{subject to:}$$

$$\mathsf{Cov}\left(\mathbf{U}\mathbf{b}_m, \mathbf{U}\mathbf{b}\right) = 0 \;\; \text{for} \;\; m = 1, ..., k-1, \quad \text{and}$$

$$||\mathbf{b}'\mathbf{V}\mathbf{b}||_2^2 = 1.^{1}$$

- Equivalent to classical PLS with response vector $\mathbf{y}$ and data-matrix $\mathbf{U}$

[1] $\mathbf{V}$ is the pos. def. matrix of inner products between all pairs of basis functions.

FPLS Coefficient

Let $\mathbf{R}$ be the $p \times K$ matrix whose columns are the first K PLS empirical weight vectors $\hat{\mathbf{b}}_1, ...\hat{\mathbf{b}}_K$.

- U scores: $\mathbf{T} = \mathbf{U}\mathbf{R}$
- Y loadings: $\boldsymbol{\alpha} = (\mathbf{T}'\mathbf{T})^{-1}\mathbf{T}'\mathbf{y}$

The estimated functional coefficient is then

$$\hat{\beta}_{FPLS}(w) = (\mathbf{R}\boldsymbol{\alpha})'\mathbf{B}(w)$$

Starting from Discrete Observations

The key to functional partial least squares is obtaining

$$\mathbf{U}_{(ij)} = \int_a^b x_i(w) B_j(w) dw, \quad i = 1, ..., n, \quad j = 1, ..., M + d.$$

- In practice, we observe p discrete points along each $x_i(w)$
- We have options for how we approximate $\mathbf{U}_{(ij)}$

Numerical Approximation

- Simple option: we can approximate $\mathbf{U}_{(ij)}$ by

$$\mathbf{U}_{(ij)} \approx \frac{b-a}{p} \sum_{k=1}^{p} x_i(w_k) B_j(w_k).$$

- Assumes noise-free observations
- Good if we have a dense observation grid

Basis Expansion for Data

- Alternatively, we can expand each observation onto a set of suitable basis functions:

$$x_i(w) \approx \mathbf{c}_i' \mathbf{B}^x(w),$$

where $\mathbf{B}^x(w)$ is a vector of $M_x + d$ B-spline basis functions and $\mathbf{c}_i$ is a vector of $M_x + d$ basis coefficients. If we define

$$\mathbf{\Theta}_{(ij)} = \int_a^b B_i^x(w) B_j(w) dw,$$

then we can express $\mathbf{U}$ as

$$\mathbf{U} \approx \mathbf{C}\mathbf{\Theta},$$

where $\mathbf{C}$ is an $n \times (M_x + d)$ matrix of basis coefficients.

Example I: Generated Responses

We generated $n = 500$ scalar responses from

$$y_i = \int_0^1 x_i(w)\beta(w)dw + \epsilon_i$$

- $x_i(w)$: random linear combinations of cubic B-spline basis functions[2]
- $\beta(w) = 10(w-1)^2 + 30\cos(4\pi w^3)$
- $\epsilon_i \sim N(0, \sigma_\epsilon^2)$[3]

[2] The basis functions were defined over 50 knots and all coefficients were generated from a standard normal distribution.

[3] The error variance σ_ϵ^2 was chosen such that the signal-to-noise ratio was 5.

Example I: Generated Predictors

To simulate misalignment, we sampled each $x_i(w)$ along two observation grids, G_A and G_B, of length 425 and 150 respectively.

- G_A: $w = 0, .0024, .0048, ..., 1$
- G_B: $w = 0, .0068, .0136, ..., 1$

The final data-set consisted of y_i and corresponding discrete observations of $x_i(w)$ on both G_A and G_B, for $i = 1, ..., 500$.

Example I: Generated Predictors

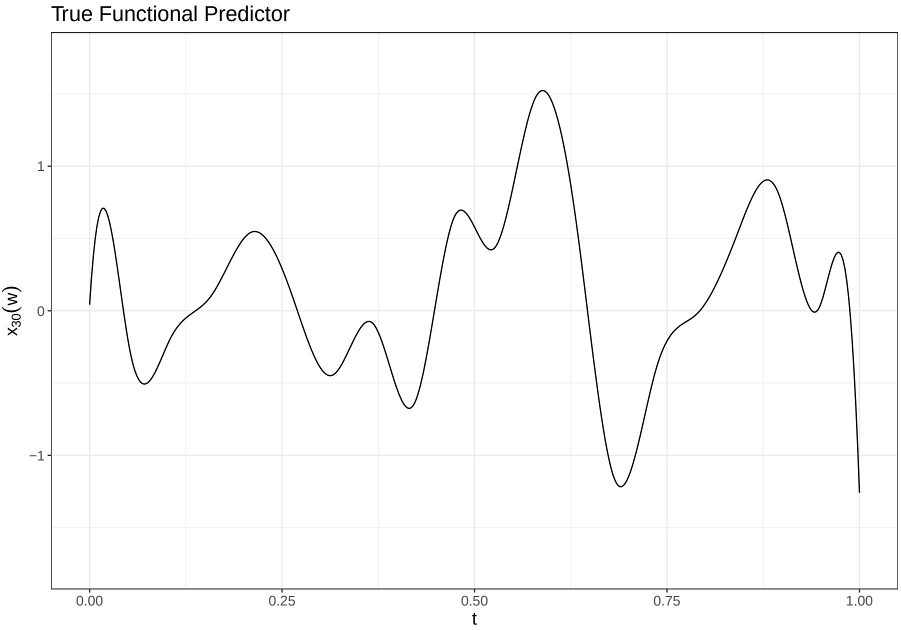
To simulate misalignment, we sampled each $x_i(w)$ along two observation grids, G_A and G_B, of length 425 and 150 respectively.

- G_A: $w = 0, .0024, .0048, ..., 1$
- G_B: $w = 0, .0068, .0136, ..., 1$

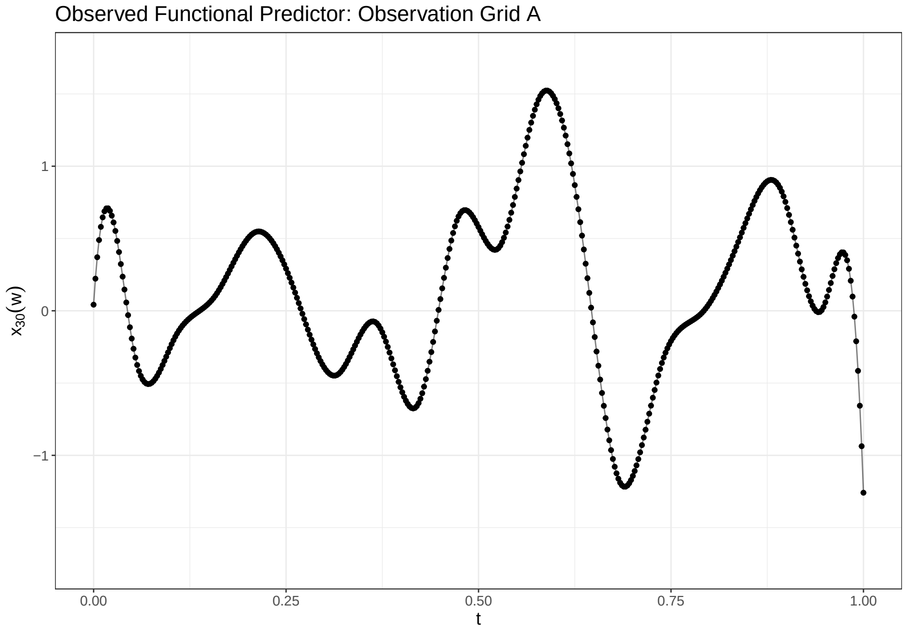
The final data-set consisted of y_i and corresponding discrete observations of $x_i(w)$ on both G_A and G_B, for $i = 1, ..., 500$.

- Goal: predict y from $x(w)$ observed on G_B, using a model trained with $x(w)$ observed on G_A.
  - $80/20$ train/test split.

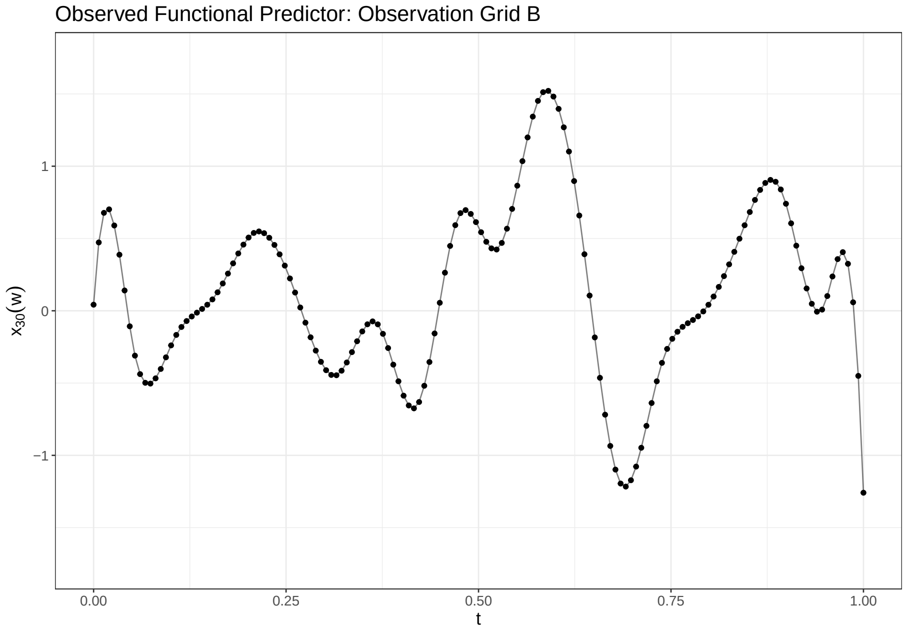
Example I: Misaligned Grids



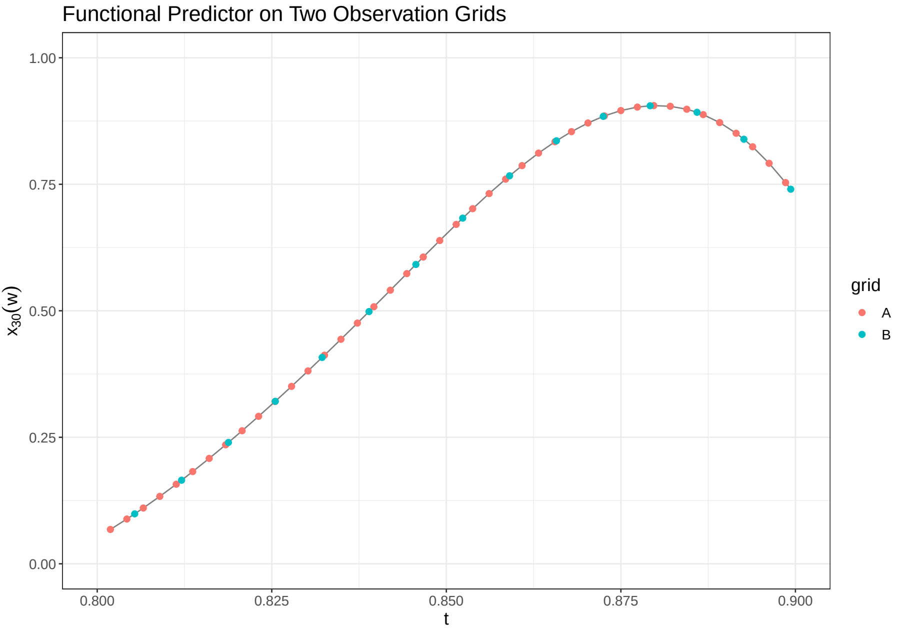
Example I: Misaligned Grids



Example I: Misaligned Grids



Example I: Misaligned Grids



Example I: Two Approaches

- Goal: predict y from $x(w)$ observed on G_B, using a model trained with $x(w)$ observed on G_A.
  - $80/20$ train/test split.

Classical PLS Approach:

- Obtain PLS coefficients $\hat{\boldsymbol{\beta}}_A$ using y^{train} and $x^{train}(w)$ on G_A
- Select PLS coefficients closest to points on G_B, $\hat{\boldsymbol{\beta}}_B$
- Predict y^{test} using observations of $x^{test}(w)$ on G_B and $\hat{\boldsymbol{\beta}}_B$

Example I: Two Approaches

- Goal: predict y from $x(w)$ observed on G_B, using a model trained with $x(w)$ observed on G_A.
  - $80/20$ train/test split.

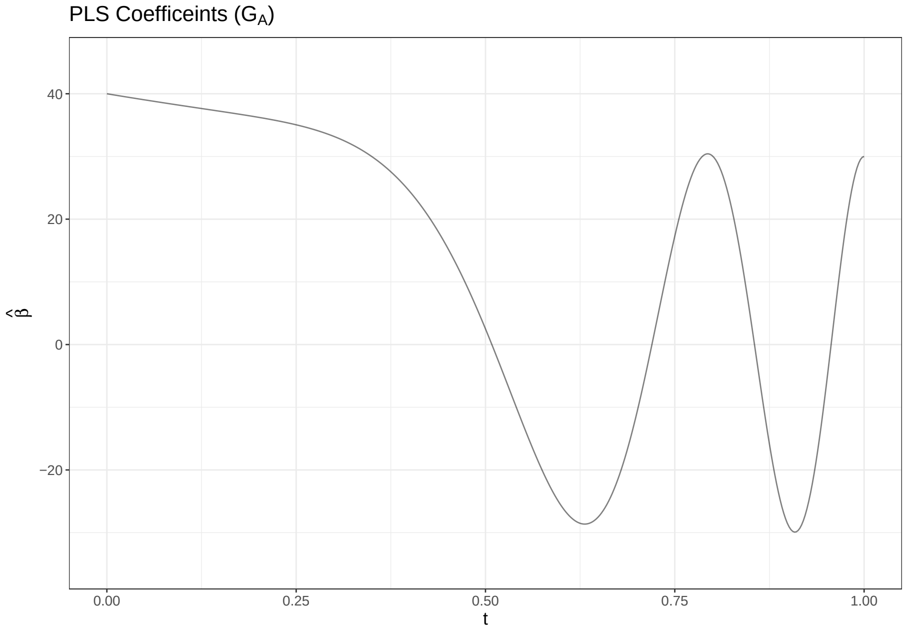
Classical PLS Approach:

- Obtain PLS coefficients $\hat{\boldsymbol{\beta}}_A$ using y^{train} and $x^{train}(w)$ on G_A
- Select PLS coefficients closest to points on G_B, $\hat{\boldsymbol{\beta}}_B$
- Predict y^{test} using observations of $x^{test}(w)$ on G_B and $\hat{\boldsymbol{\beta}}_B$

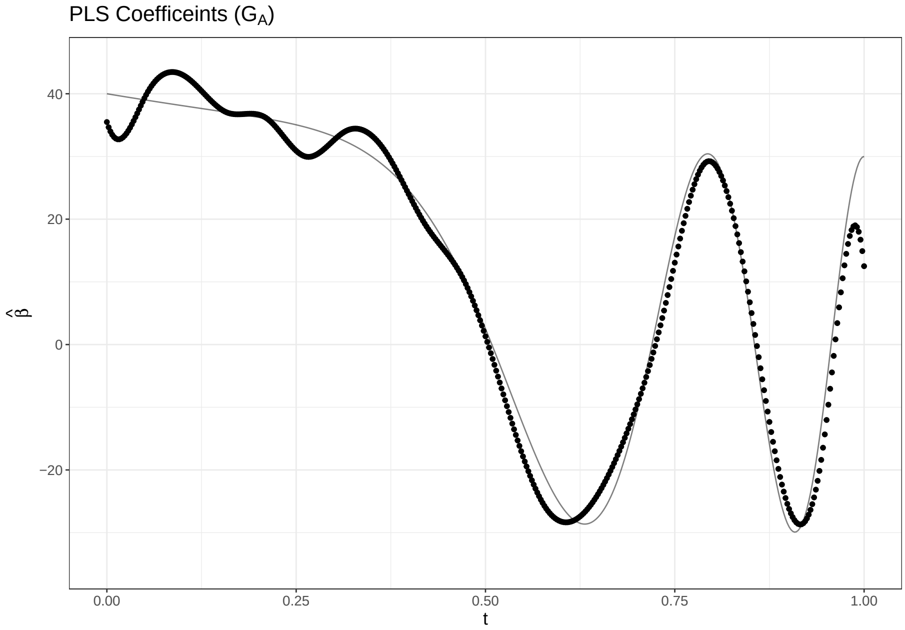
Functional PLS approach:

- Obtain $\hat{\beta}_{FPLS}(w)$ using observations of $x^{train}(w)$ on G_A
- Predict y^{test} using observations of $x^{test}(w)$ on G_B and $\hat{\beta}_{FPLS}(w)$.

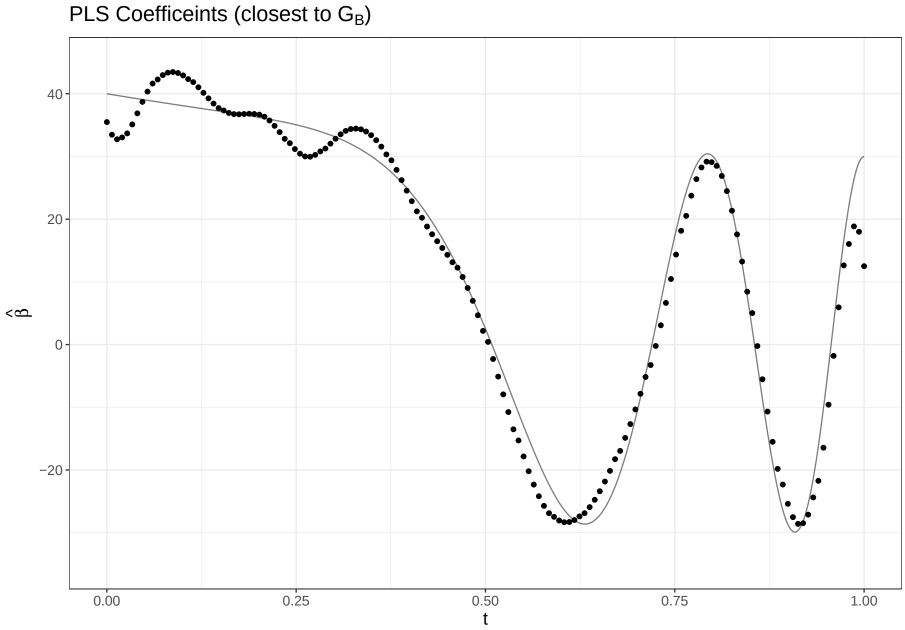
Example I: Classical PLS



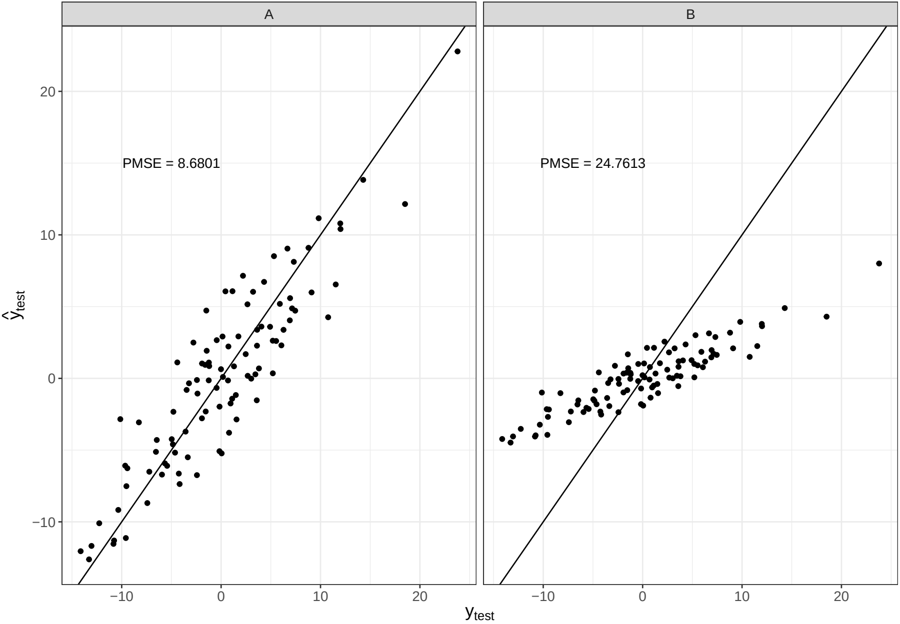
Example I: Classical PLS



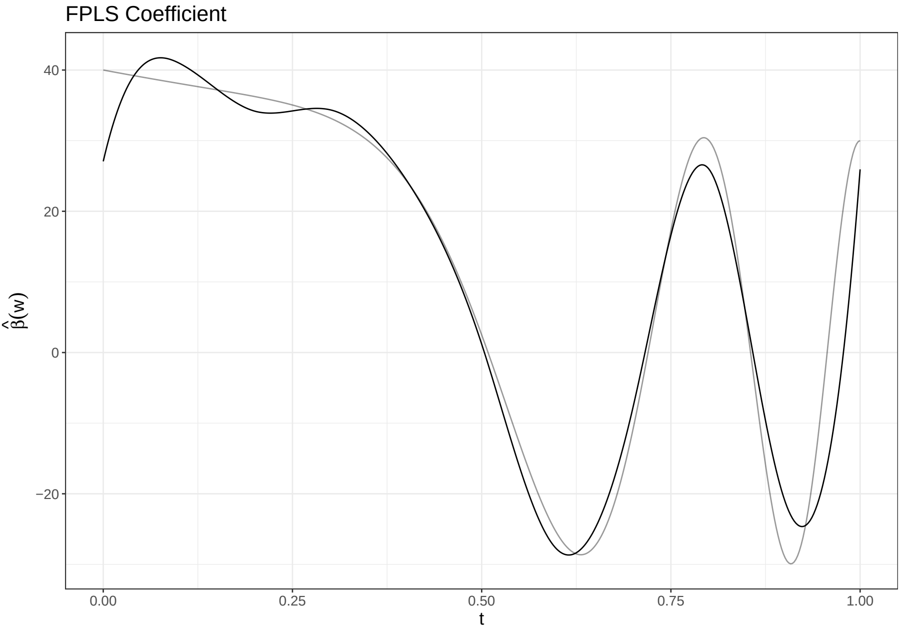
Example I: Classical PLS



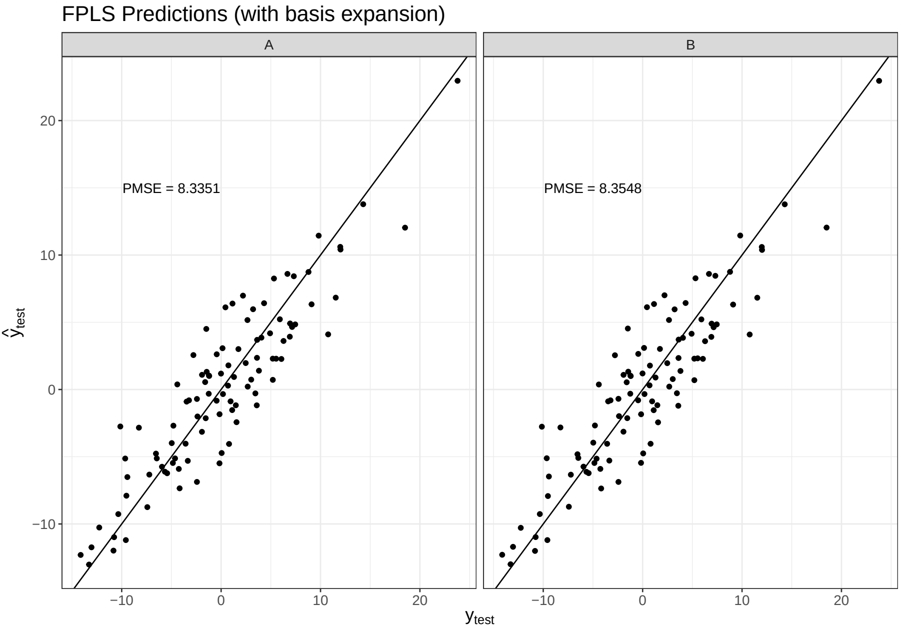
Example I: Classical PLS



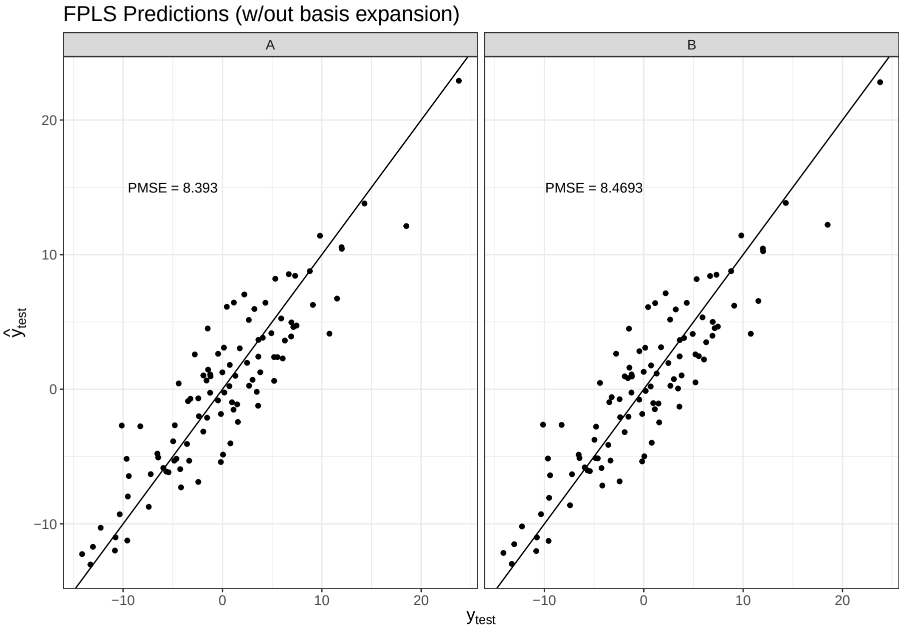
Example I: Functional PLS



Example I: Functional PLS



Example I: Functional PLS



Example II: AOP Crown Data

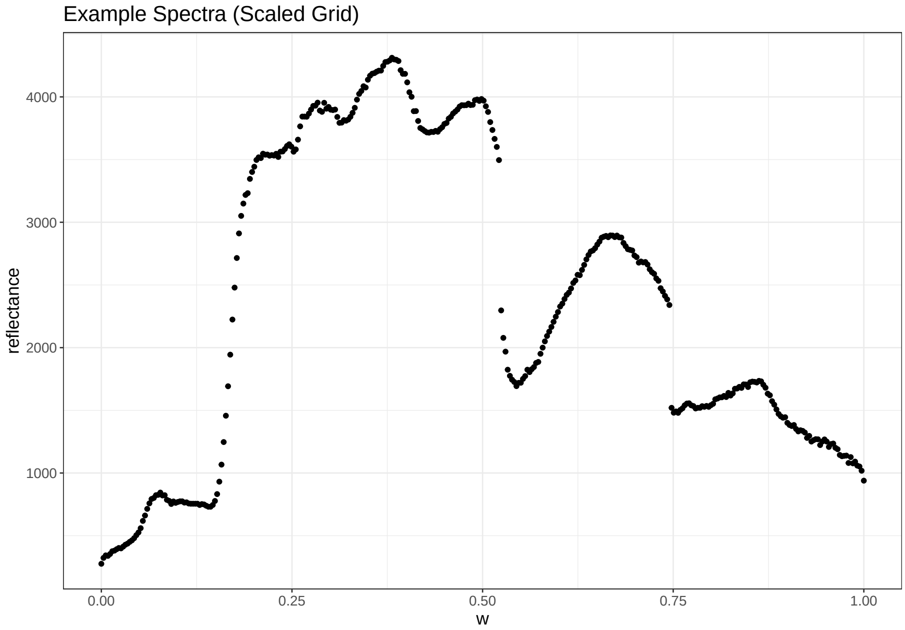
We applied the same method to the AOP Crown data-set to predict `d15N` from observed spectra.

- $n = 2515$ observations
- $p_A = 350$ spectral points per spectra.

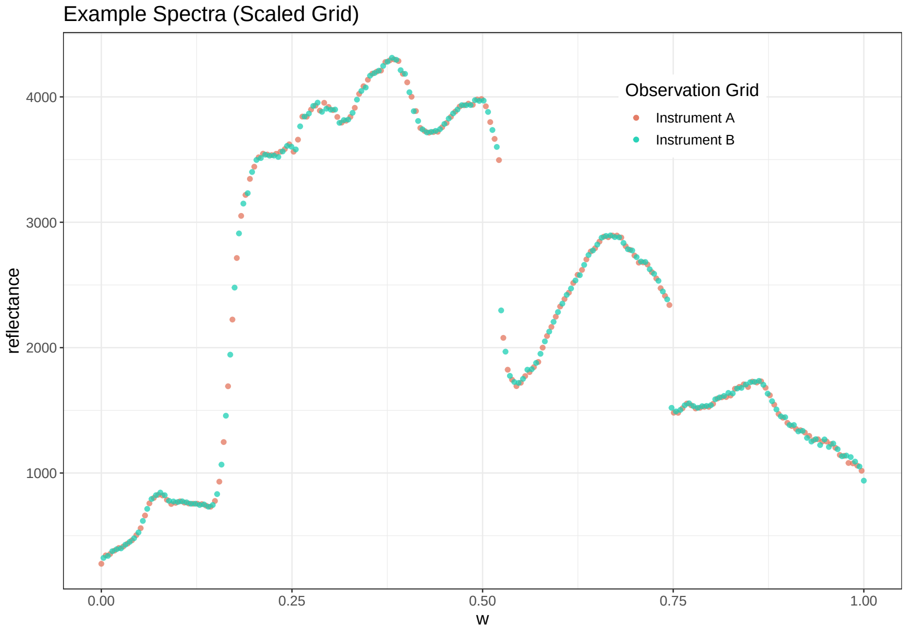
To simulate misalignment, we sampled every other spectral point for observation grid A and assigned the remaining spectral points to grid B.

- Grid A (G_A): odd indices
- Grid B (G_B): even indices.

Example II: Spectra



Example II: Two Observation Grids



Example II: Two Approaches

- Goal: predict `d15N` from spectra observed on G_B, using a model trained with spectra observed on G_A.
  - $80/20$ train/test split.

Example II: Two Approaches

- Goal: predict `d15N` from spectra observed on G_B, using a model trained with spectra observed on G_A.
  - $80/20$ train/test split.

Classical PLS Approach + Linear Approximation:

- Obtain PLS coefficients $\hat{\boldsymbol{\beta}}_A$ using y^{train} and $x^{train}(w)$ on G_A
- Approximate $x^{test}(w)$ on G_A using $x^{test}(w)$ on G_B
- Predict y^{test} using approx. observations of $x^{test}(w)$ on G_A and $\hat{\boldsymbol{\beta}}_A$.

Example II: Two Approaches

- Goal: predict `d15N` from spectra observed on G_B, using a model trained with spectra observed on G_A.
  - $80/20$ train/test split.

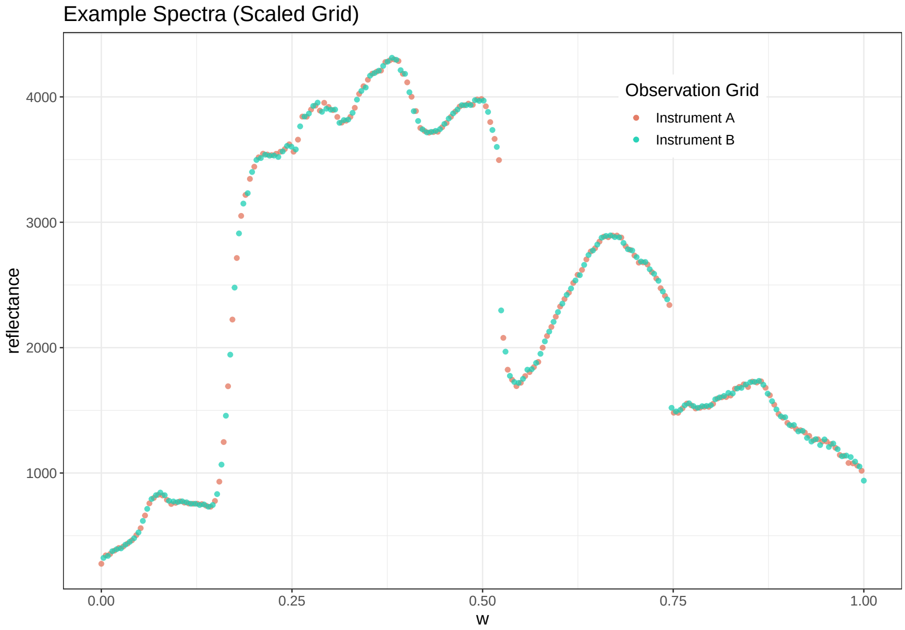
Classical PLS Approach + Linear Approximation:

- Obtain PLS coefficients $\hat{\boldsymbol{\beta}}_A$ using y^{train} and $x^{train}(w)$ on G_A
- Approximate $x^{test}(w)$ on G_A using $x^{test}(w)$ on G_B
- Predict y^{test} using approx. observations of $x^{test}(w)$ on G_A and $\hat{\boldsymbol{\beta}}_A$.

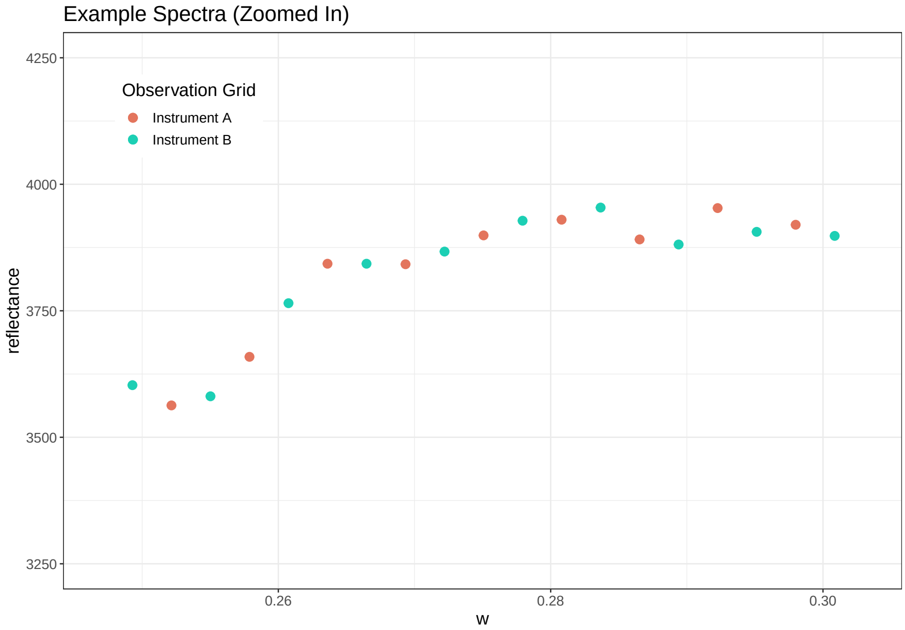
Functional PLS approach:

- Obtain $\hat{\beta}_{FPLS}(w)$ using observations of $x^{train}(w)$ on G_A
- Predict y^{test} using observations of $x^{test}(w)$ on G_B and $\hat{\beta}_{FPLS}(w)$.

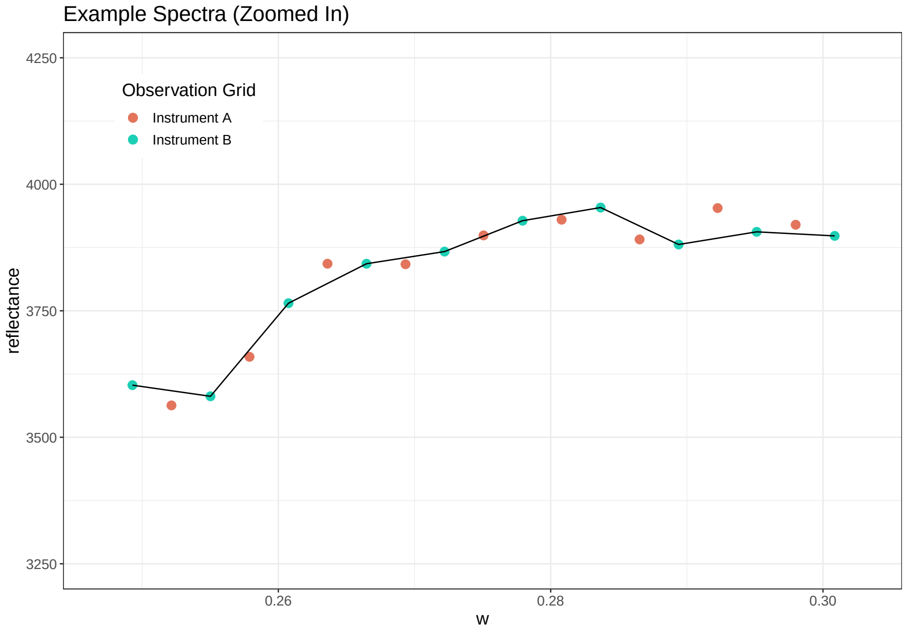
Example II: Linear Approximation



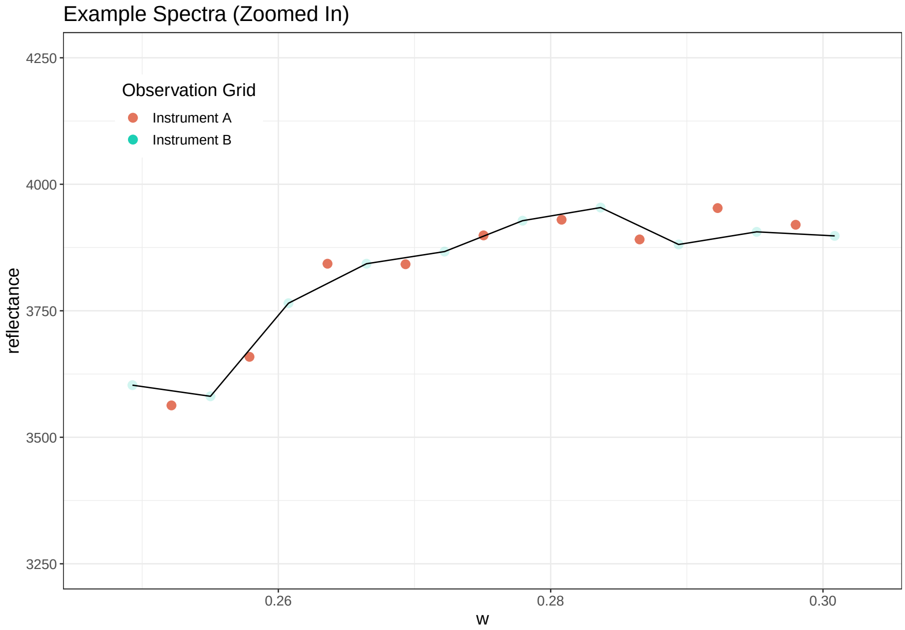
Example II: Linear Approximation



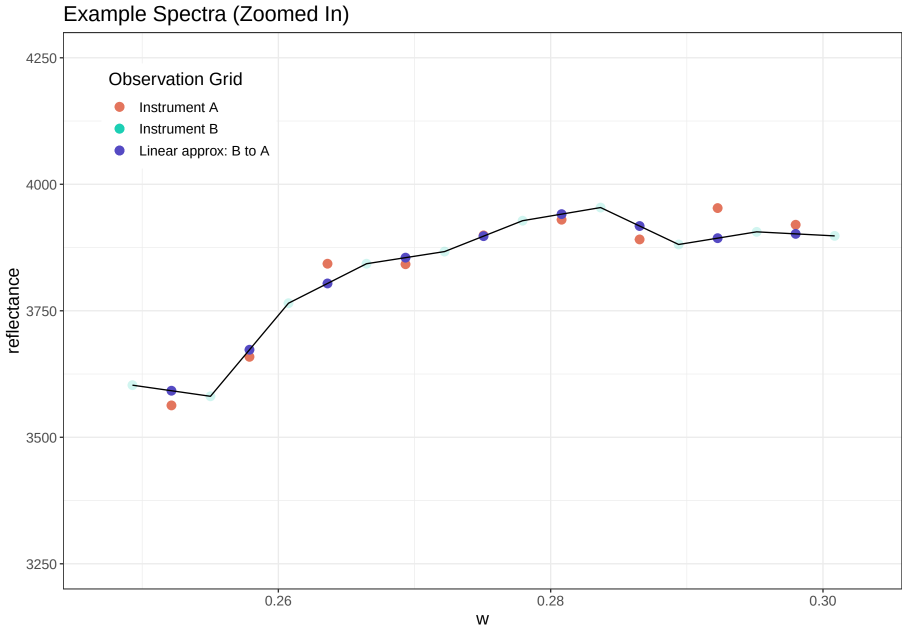
Example II: Linear Approximation



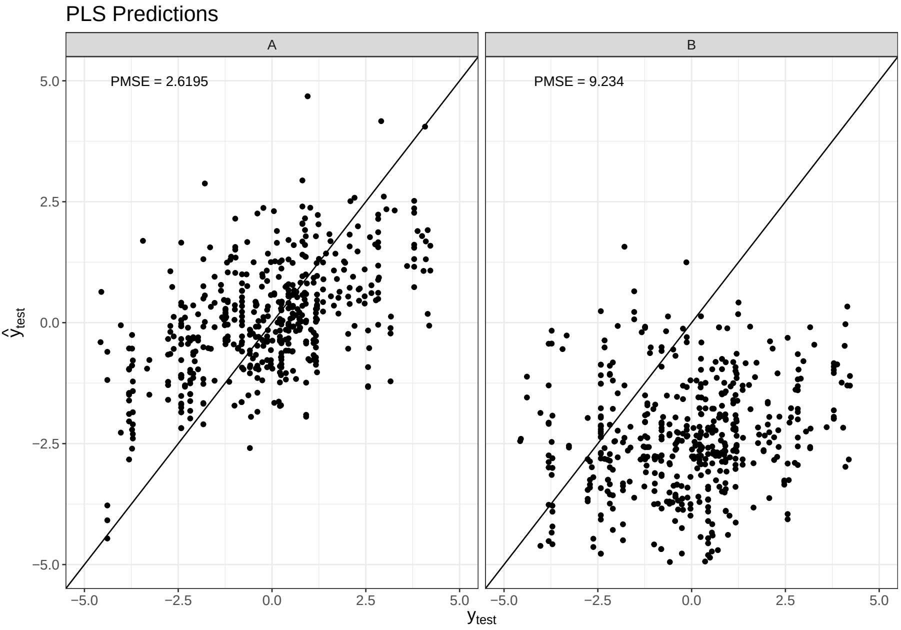
Example II: Linear Approximation



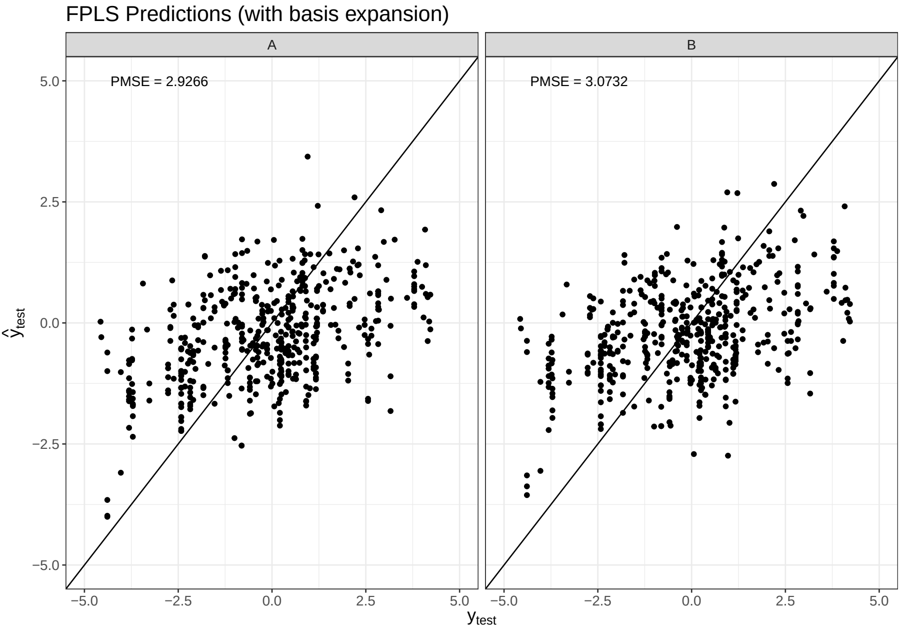
Example II: Linear Approximation



Example II: Classical PLS + Linear Approximation



Example II: Functional PLS



Example II: Results for Other Responses

We repeated the example for two other responses in the AOP Crown data-set, `LWC_per` and `LMA_gm2`. Using functional PLS resulted in reduced root mean squared prediction error (RMSPE) for all three responses.

	`d15N`	`LWC_per`	`LMA_gm2`
PLS	3.04	45.42	218.89
FPLS	1.75	38.21	150.10

Table 1: RMSPE for three site trait variables in the AOP Crown data-set.

Appendix I: Intuition Behind FPLS Coefficient

Recall that the (0-intercept) FLM:

$$y_i = \int_a^b x_i(w)\beta(w)dw + \epsilon_i. \tag{1}$$

When we approximate $r(w)$ as $r(w) \approx \mathbf{b}'\mathbf{B}(w)$, we implicitly assume

$$\beta(w) \approx \boldsymbol{\gamma}'\mathbf{B}(w), \tag{2}$$

allowing us to re-write (1) as

$$\mathbf{y} = \mathbf{U}\boldsymbol{\gamma} + \epsilon.$$

Performing PLS of $\mathbf{U}$ on $\mathbf{y}$ yields $\hat{\boldsymbol{\gamma}} = \mathbf{R}\boldsymbol{\alpha}$. Hence, from (2),

$$\hat{\beta}(w) = (\mathbf{R}\boldsymbol{\alpha})'\mathbf{B}(w).$$