Praktikum z ekonometrie

VŠE Praha

Tomáš Formánek

Block 2 – Model selection – Outline

1. Model selection
   - Theoretical approach: specific-to-general, general-to-specific
   - Stepwise model selection
2. Model selection & regularization
   - Penalized regression
   - Dimension reduction: PCR & PLS
3. Moving beyond linearity
   - QREG - quick repetition
   - Polynomial and step regression
   - Regression splines and smoothing splines
   - Local regression
   - Generalized Additive Models (GAM)

Theoretical basis for model selection algorithms

Alternative approaches to econometric modeling

Simple-to-general approach

- Traditional approach to econometric modeling
- Starts with formulation of the simplest model consistent with the relevant economic theory.
- If this initial model proves unsatisfactory, it is improved in some way – adding or changing variables, using different estimators etc.

Alternative approaches to econometric modeling

Criticism of the simple-to-general approach

- Revisions to the simple model are carried out arbitrarily and simply reflect investigator's prior beliefs: danger of always finding what you want to find.
- It is open to accusation of data mining: researchers usually presents just the final model (true significance level is problematic).

Alternative approaches to econometric modeling

General-to-specific approach

- Professor Hendry, London School of Economics started this approach in the 80ies.
- It starts with formulation of a very general and maybe quite complicated model.
- Starting model contains a series of simpler models, nested within it as special cases.
- These simpler models should represent all the alternative economic hypotheses that require consideration.

Alternative approaches to econometric modeling

General-to-specific approach

- General model must be able to explain existing data and be able to satisfy various tests of misspecification.
- What follows is simplification search (testing-down procedure). Through parameter restrictions, we test nested models against the containing model. If the nested model does not pass the tests, we can reject the whole branch of sub-nested models.
- If we find more non-nested models satisfying tests, we can compare those (using information criteria, etc.).

Alternative approaches to econometric modeling

Advantages of the general-to-specific approach

- "Data mining" present in this approach is transparent (for all to see) and it is carried out in a systematic manner that avoids worst data mining problems.
- Researcher usually reports both the initial general model and all steps involved so it is possible to get some idea about the true significance levels.
- Supporters of this approach stress the importance of both testing final models against new data and the ability of the model to provide adequate out-of-sample forecasts.

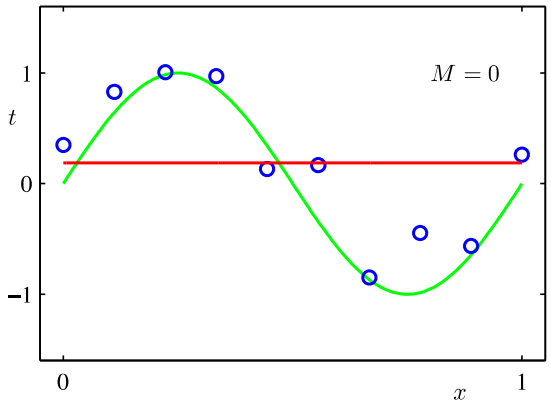
Model selection basics - repetition from previous courses

Variance vs. Bias trade-off - repetition

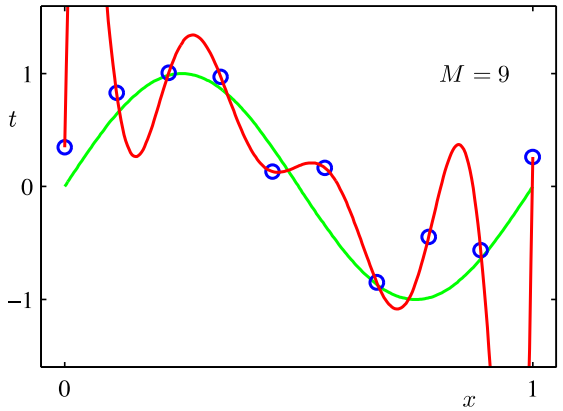
Population equation example: $y = \sin(x) + u$

Bias-Variance tradeoff – Intuition

- Model too simple: does not fit the data well
  - A *biased* solution



- Model too complex: small changes to the data, solution changes a lot
  - A *high-variance* solution



Train sample & Test sample - repetition

Suppose we fit a model $\hat{f}(\boldsymbol{x})$ to some training data $\text{Tr} = \{y_i, \boldsymbol{x}_i\}_1^n$ and we wish to see how well it performs.

- We could compute *MSE* over Tr:

$$MSE_{\text{Tr}} = \frac{1}{n} \sum_{i \in \text{Tr}} \left[y_i - \hat{f}(\boldsymbol{x}_i) \right]^2$$

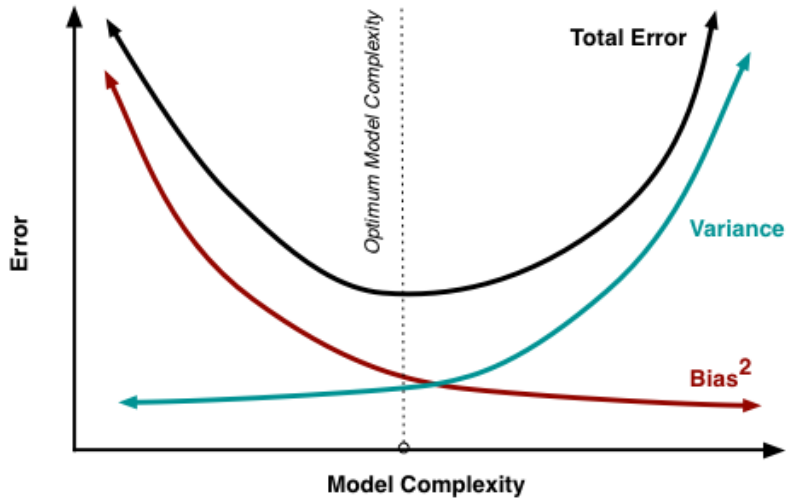
When searching for the "best" model by minimizing *MSE*, the above statistic would lead to over-fit models.

- Instead, we should (if possible) compute the *MSE* using fresh test data $\text{Te} = \{y_i, \boldsymbol{x}_i\}_1^m$:

$$MSE_{\text{Te}} = \frac{1}{m} \sum_{i \in \text{Te}} \left[y_i - \hat{f}(\boldsymbol{x}_i) \right]^2$$

Variance vs. Bias trade-off - repetition

$$\mathrm{E}(MSE_0) = var(\hat{f}(\boldsymbol{x}_0)) + [Bias\,(\hat{f}(\boldsymbol{x}_0))]^2 + \mathrm{var}(\varepsilon_0)$$



This is an illustration, $\mathrm{var}(\varepsilon_0)$ not shown explicitly.
(lies at the /asymptotic/ minima of Variance and Bias^2)

- Variance refers to the amount by which $\hat{f}(\boldsymbol{x}_0)$ would change if we estimate it using different training data sets.
- Bias is introduced by approximating real-life DGP by simple model.
- Derivation of the formula is complex, see eg. information here

k-Fold Cross Validation - repetition

- Training error (MSE_{Tr}) can be calculated easily.
- However, MSE_{Tr} is not a good approximation for the MSE_{Te} (out-of sample predictive properties of the model).
- Usually, MSE_{Tr} dramatically underestimates MSE_{Te}.

Cross-validation is based on re-sampling (similar to bootstrap).

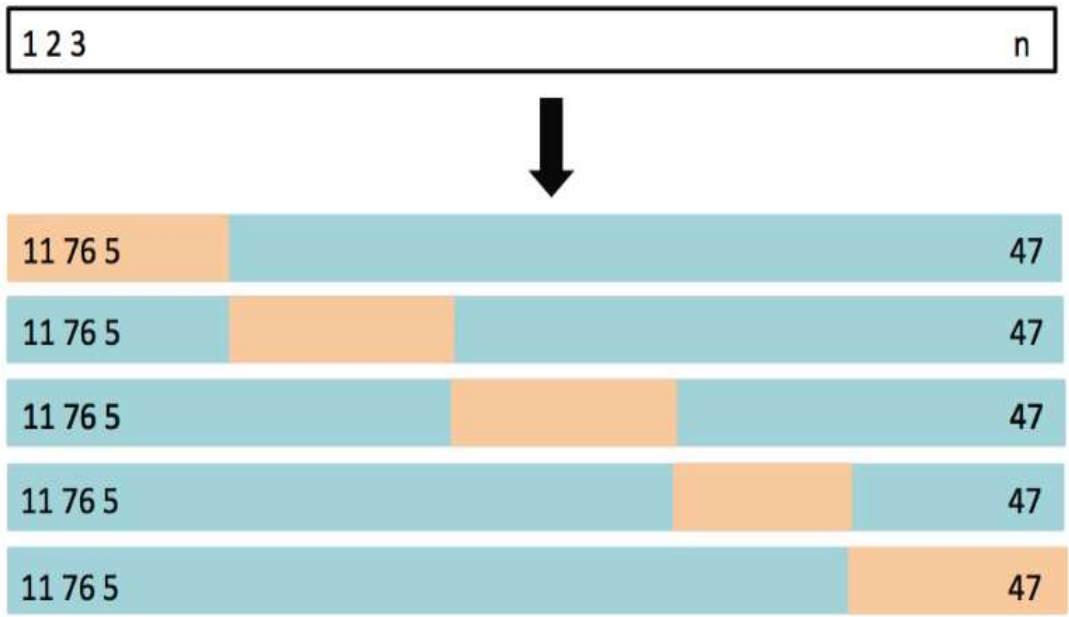
Repeatedly fit a model of interest to samples formed from the training set & make "test sample" predictions, in order to obtain additional information about predictive properties of the model.

k-Fold Cross Validation - repetition

- In k-Fold Cross-Validation (kFCV), the original sample is randomly partitioned into k roughly equal subsamples (divisibility).
- Of the k subsamples, a single subsample is retained as the test sample, and the remaining $(k-1)$ subsamples are used as training data.
- The cross-validation process is then repeated k times (the k folds), with each of the k subsamples used exactly once as the test sample.
- The k results from the folds can then be averaged to produce a single estimation – the cross-validated error.
- $k = 5$ or $k = 10$ is commonly used.
- Sometimes, kFCV process is repeated (R-times – say, 100) to get distribution of the cross-validated error term.

k-Fold Cross Validation - repetition

kFCV example for CS data & $k = 5$:
(random sampling, no replacement)



In TS, a similar "Walk forward" test procedure may be applied.

k-Fold Cross Validation - repetition

$$CV_{(k)} = \frac{1}{k} \sum_{s=1}^{k} MSE_s \,,$$

where:

$CV_{(k)}$ is the k-fold CV estimate,

k is the number of folds used (e.g. 5 or 10),

$MSE_s = \frac{1}{m_s} \sum_{i \in C_s} (y_i - \widehat{y}_i)^2$

m_s and C_s refer to test sample observations for each of the kFCV steps.

As we evaluate predictions from two or more models, we look for the lowest $CV_{(k)}$.

Comparison of estimation methods / models

Comparison of models/methods (besides *k*FCV methods):

$$\text{Mallow's } C_p = \frac{1}{n}(RSS + 2d\widehat{\sigma}^2),$$

$$AIC = \frac{1}{n\widehat{\sigma}^2}(RSS + 2d\widehat{\sigma}^2),$$

$$BIC = \frac{1}{n}(RSS + \log(n)d\widehat{\sigma}^2),$$

- where d is the number of regressors and n is the sample size.
- Model selection: find a model where a statistic is minimized.
- $\log(n) > 2$ $(n > 7) \Rightarrow$ generally, BIC penalizes complexity more.
- When comparing models, $AIC \propto C_p$;
  AIC and BIC may contradict
- If $\widehat{\sigma}^2$ is an unbiased estimate of σ^2, then C_p is an unbiased estimate of test MSE (training error is adjusted by a factor proportional to the number of basis functions used).
- Sometimes, models are selected using C_p (AIC) instead kFCV.

Model selection algorithms

Model selection algorithms - Introduction

- **Subset Selection:** We identify a subset of the p predictors that we believe to be related to the response. We then fit a model using least squares on the reduced set of variables.
- **Shrinkage:** We fit a model involving all p predictors, but the estimated coefficients are shrunken towards zero relative to the least squares estimates. This shrinkage (also known as regularization) has the effect of reducing variance and can also perform variable selection.
- **Dimension Reduction:** We project the p predictors into a M-dimensional subspace, where $M < p$. This is achieved by computing M different linear combinations, or projections, of the variables. Then these M projections are used as predictors to fit a linear regression model by least squares.

Model selection algorithms - Subset selection

1. Best subset selection
2. Forward stepwise selection
3. Backward stepwise selection
4. Algorithms combining Forward and Backward stepwise selection
5. Comparison & computational complexity of methods

Best subset selection

1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors.
   Say, $y_i = \beta_0 + u_i$
   This model simply predicts the sample mean for y.
2. For $k = 1, 2, \ldots, p$:
   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.
   (b) Choose the best among these $\binom{p}{k}$ models and call it $\mathcal{M}_k$.
   Here, best is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$, using crossvalidated prediction error, C_p, AIC, BIC or adj. R^2.

Note: $\binom{p}{k} = \frac{p!}{k!(p-k)!}$; $\sum_{k=1}^{p} \binom{p}{k} = 2^p$

Forward stepwise selection

1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. Say, $y_i = \beta_0 + u_i$
2. For $k = 0, 1, \ldots, (p-1)$:
   (a) Consider all $(p-k)$ models that augment the predictors in $\mathcal{M}_k$ with one additional predictor.
   (b) Choose the best among these $(p-k)$ models, and call it $\mathcal{M}_{k+1}$. Here, best is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$, using crossvalidated prediction error, C_p, AIC, BIC or adj. R^2.

Backward stepwise selection

1. Let $\mathcal{M}_p$ denote the *full model*, which contains all p predictors.
   Say, $y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + u_i$
2. For $k = p, (p-1), \ldots, 1$:
   (a) Consider all k models that contain all but one of the predictors in $\mathcal{M}_k$, for a total of $(k-1)$ predictors.
   (b) Choose the best among these k models, and call it $\mathcal{M}_{k-1}$.
   Here, best is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$, using crossvalidated prediction error, C_p, AIC, BIC or adj. R^2.

Computational complexity of methods

Computational complexity:

- Forward stepwise and Backward stepwise selection:
  Greedy algorithms.
  $[1 + p(p+1)/2] \approx p^2$ models need to be estimated and evaluated.
  Computationally feasible even for high p values (large sets of potential regressors).

- Best subset selection
  2^p models to be estimated and evaluated.
  For large p, enormous search space can lead to over-fitting and high variance of the coefficient estimates.

Forward & Backward stepwise [and their hybrid combinations] tends to do well in practice (are efficient algorithms), yet they do not guarantee finding the best possible model out of all 2^p possible models.

Parameter shrinkage methods

Parameter shrinkage methods

Subset selection:

- Subset of predictors is retained, the rest is discarded.
- Generates interpretable models.
- Selection is a discrete process: variables are either retained or discarded.
- Predictions based on models with different regressor-sets often exhibits high variance. Shrinkage methods are more continuous, and don't suffer as much from high variability.

Shrinkage methods

- More continuous – do not suffer as much from high variability.

Parameter shrinkage methods

Ridge regression and lasso regression

- As an alternative to stepwise selection, we can fit a model containing all p predictors using a shrinkage method that constrains or regularizes the coefficient estimates and/or that shrinks the coefficient estimates towards zero.
- It may not be immediately obvious why such constraints or shrinkage should improve the fit – details discussed next.

Ridge regression

Consider a LRM: $y = f(x_1, x_2, \ldots, x_p)$

- **OLS** can be used to estimate $\hat{\boldsymbol{\beta}} = (\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p)'$ by minimizing the RSS:

$$\min_{\boldsymbol{\beta}}: RSS = \sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right)^2$$

- **Ridge regression** $\hat{\boldsymbol{\beta}}$ estimates are the values that minimize:

$$\min_{\boldsymbol{\beta}}: \left[\sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \hat{\beta}_j^2 \right] = \left(RSS + \lambda \sum_{j=1}^{p} \hat{\beta}_j^2 \right)$$

where $\lambda > 0$ is a tuning parameter, determined separately.

Ridge regression

$$\min_{\boldsymbol{\beta}}: \left[\sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \hat{\beta}_j^2 \right]$$

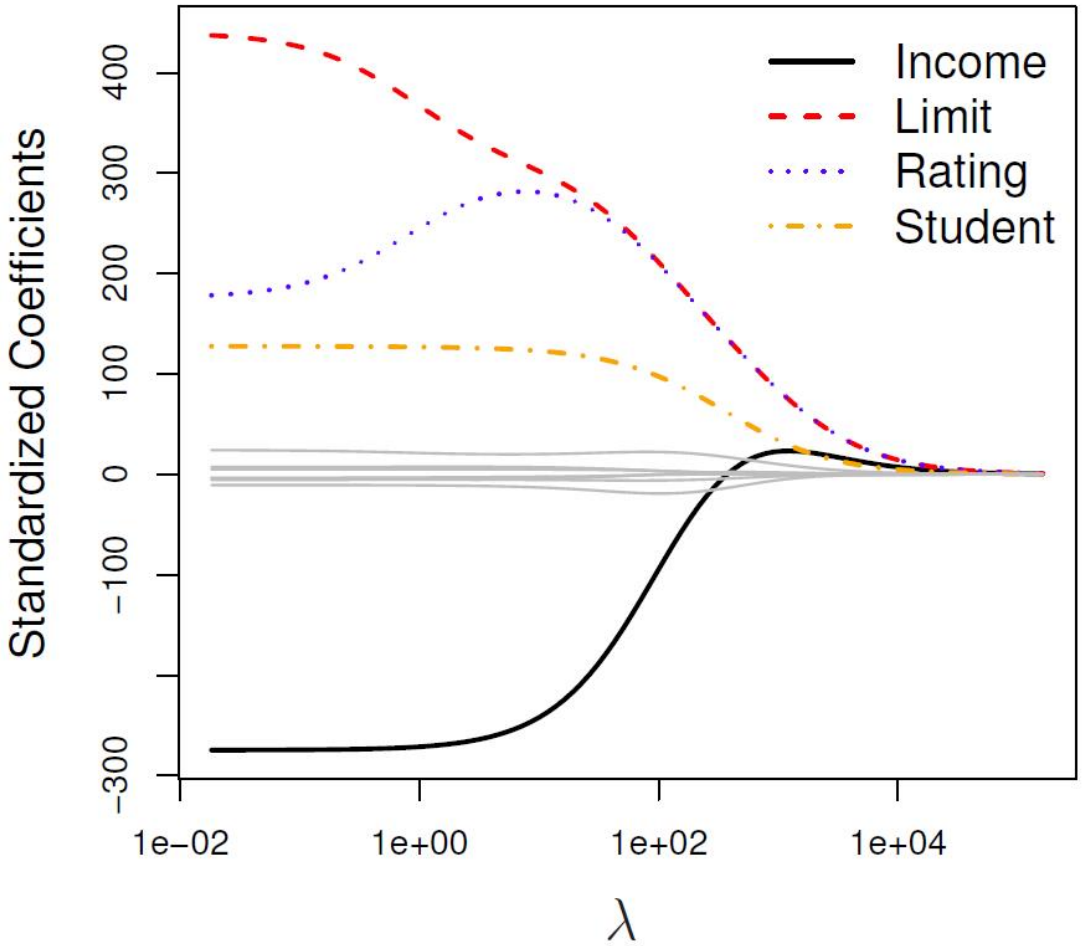
- Seeks β_j estimates that fit the data well, by making the RSS small.
- Shrinks regression coefficients by imposing penalty on their size. The ridge coefficients minimize a penalized RSS
- $\lambda \geq 0$ is a complexity parameter – controls the amount of shrinkage: larger value of $\lambda \ \rightarrow$ greater amount of shrinkage.
- $(\lambda \sum_{j=1}^{p} \hat{\beta}_j^2)$ is a shrinkage penalty.
  It is small when $\hat{\beta}_1, \ldots, \hat{\beta}_p$ are close to zero and/or λ is small.
  High λ shrinks $\hat{\beta}_j$ towards zero and towards each other.

Ridge regression

$$\min_{\boldsymbol{\beta}}: \left[\sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \hat{\beta}_j^2 \right]$$

- With many correlated variables in a LRM (i.e. under multicollinearity), OLS-estimated coefficients can become poorly determined and exhibit high variance.
  - A wildly large positive coefficient on one variable can be canceled by a similarly large negative coefficient on the correlated regressor(s).
  - Even with small sampling changes, such coefficients may change dramatically (even in sign).
- By imposing a ridge penalty (size constraint on the coefficients), this problem is alleviated.
- For predictive properties, selecting a good value for λ is critical; cross-validation is used.

Ridge regression - example



Ridge regression example output:
coefficient estimates are plotted as a function of λ.

Ridge regression

- The standard OLS coefficient estimates are scale equivariant: multiplying x_j by a constant c simply leads to a scaling of the least squares coefficient estimates by a factor of $1/c$.

  Regardless of predictor scaling, $(\hat{\beta}_j x_{ij})$ will remain the same.

- In contrast, the ridge regression coefficient estimates can change substantially when multiplying a given predictor (or other predictors!) by a constant, due to the sum of squared coefficients term in the penalty part of the ridge regression objective function.

- Therefore, it is best to apply ridge regression after standardizing the predictors, using the formula:

$$\widetilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \overline{x}_j)^2}} \quad \text{hence} \quad \text{s.d.}(\widetilde{x}_j) = 1\,;\; j = 1, 2, \ldots$$

Ridge regression - final remarks

$$\min_{\boldsymbol{\beta}}: \left[\sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \hat{\beta}_j^2 \right]$$

- Ridge solutions are not equivariant under scaling of the inputs, so we standardize the inputs before estimation (this just recaps previous page).
- The intercept β_0 has been left out of the penalty term. Penalization of the intercept would make the procedure depend on the origin chosen for y.
- Ridge penalty shrinks coefficients towards zero (except $\hat{\beta}_0$). Coefficients of correlated variables are shrunk toward each other. (See chapter 3 of ESLII for detailed technical discussion.)

Ridge regression - final remarks

For LRM, RSS and OLS may be easily written in matrix form as:

- $RSS(\text{OLS}) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})'(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})$
- $\hat{\boldsymbol{\beta}}_{\text{OLS}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$

For ridge regression, this may be re-written as

- $RSS(\lambda) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})'(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) + \lambda\,\boldsymbol{\beta}'\boldsymbol{\beta}$
- $\hat{\boldsymbol{\beta}}_{\text{ridge}} = (\boldsymbol{X}'\boldsymbol{X} + \lambda\boldsymbol{I}_p)^{-1}\boldsymbol{X}'\boldsymbol{y}$

With the choice of quadratic penalty $\boldsymbol{\beta}'\boldsymbol{\beta}$, the ridge regression solution is again a linear function of y.

Ridge method adds a positive constant to the diagonal of $(\boldsymbol{X}'\boldsymbol{X})$ before inversion. This makes the problem non-singular, even if $(\boldsymbol{X}'\boldsymbol{X})$ is not of full rank (perfect multicollinearity, $p > n$, $p \gg n$).

This was the main motivation for ridge regression when it was first introduced in statistics (Hoerl and Kennard, 1970)

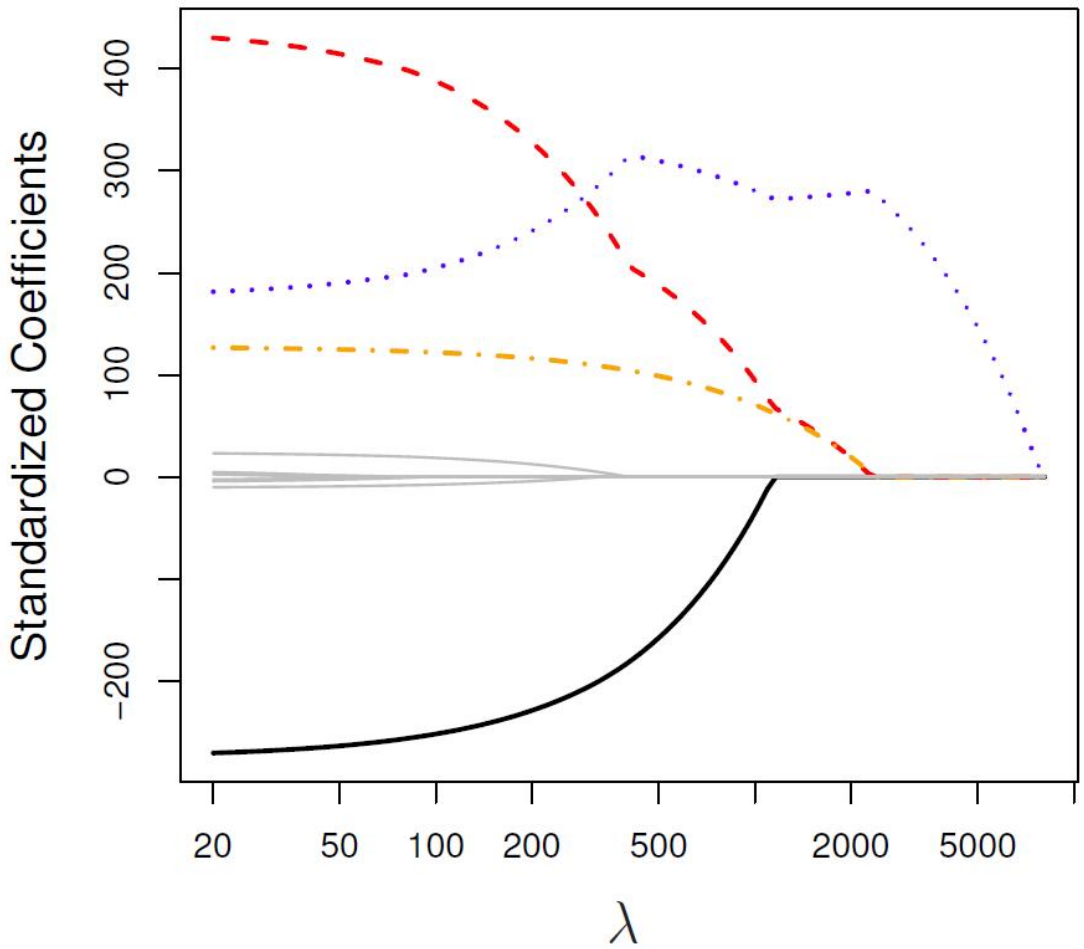
Lasso regression

- Ridge regression has one empirical disadvantage: unlike subset selection, which will generally select models that involve just a subset of the variables, ridge regression will include all p predictors in the final model (with potentially small but nonzero values).
- The Lasso is a relatively recent alternative to ridge regression that overcomes this disadvantage. The lasso coefficients, $\hat{\boldsymbol{\beta}}_L$ estimates are the values that minimize the penalized RSS:

$$\min_{\boldsymbol{\beta}}: \left[\sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\hat{\beta}_j| \right]$$

  again, $\lambda > 0$ is a tuning parameter, determined separately (kFCV).
- In statistical parlance, the lasso uses an ℓ_1 (pronounced "ell 1") penalty instead of an ℓ_2 penalty. The ℓ_1 norm of a coefficient vector is given by $\|\beta\|_1 = \sum |\beta|$.

Lasso regression - example



Lasso regression example output:
coefficient estimates are plotted as a function of λ.

Lasso regression

- LASSO: Least Absolute Shrinkage and Selection Operator
- As with ridge regression, the lasso shrinks the coefficient estimates towards zero.
- In the case of the lasso, the ℓ_1 penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large (see ISLR textbook).
- Much like stepwise model selection, the lasso regression performs variable selection.
- Lasso yields sparse models - that is, models that involve only a subset of the variables.
- As in ridge regression, selecting a good value of λ for the lasso is critical; cross-validation is used.

Ridge & Lasso - discussion

- Neither ridge regression nor the lasso will universally dominate the other.
- In general, one might expect the lasso to perform better when the response is a function of only a relatively small number of predictors.
- However, the number of predictors that is related to the response is never known a priori for real data sets.
- CV can be used in order to determine which approach is better on a particular data set.

Ridge & Lasso - λ selection

Cross-validation is used to determine λ, as follows:

1. We choose a grid of λ values and compute the cross-validation error rate for each value of λ.
2. We select the tuning parameter λ, for which the cross-validation error is smallest.
3. Finally, the model is re-fit using all of the available observations and the selected value of the tuning parameter λ.

The above steps 1 and 2 can be performed for both ridge and lasso.
. . . cross-validation errors are compared to select "best" λ
. . . and to choose between ridge and lasso.

Elastic net regression (penalty)

$$\min \left[\sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \left(\alpha |\beta_j| + (1-\alpha) \hat{\beta}_j^2 \right) \right]$$

- **Lasso penalty** encourages sparse solutions (in terms of coefficients), yet it is somewhat indifferent to the choice among a set of strong but correlated regressors.
- **Ridge penalty** shrinks coefficients of correlated variables toward each other, no spare solution effect.
- **Elastic net penalty** is a compromise (combined method). The second term of the penalization element encourages highly correlated features to be averaged, while the first term encourages a sparse solution in the coefficients of these averaged features.
- Minimizer follows from Zou and Hastie, (2005). Other penalty specifications are possible (e.g. general λ_1 and λ_2 can be applied).

Elastic net regression (penalty)

$$\min \left[\sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \left(\alpha |\beta_j| + (1-\alpha) \hat{\beta}_j^2 \right) \right]$$

- The elastic net penalty can be used with any linear model (LM, GLM), in particular for regression or classification.
  Logit (GLM/MLE) example of elastic net penalty generalization:
  $\max_{\boldsymbol{\beta}} \left[\sum_{i=1}^{n} (y_i \log[G(\boldsymbol{x}_i \boldsymbol{\beta})] + (1-y_i) \log[1-G(\boldsymbol{x}_i \boldsymbol{\beta})]) - \lambda \sum_{j=1}^{p} \left(\alpha |\beta_j| + (1-\alpha) \hat{\beta}_j^2 \right) \right]$

- Parameter α determines the relative mix of ridge and lasso penalties. It is set prior to model estimation.

- CV can be used to chose α and λ.

High dimensionality & dimension reduction methods

Principal component vs. Factor analysis

PCA vs FA – quick overview:

- **Principal component analysis** involves extracting linear composites of observed variables. We use PCA to reduce a dataset of correlated observed variables to a smaller set of important independent composite variables.

- **Factor analysis** is based on a formal model predicting observed variables from theoretical latent factors. We use FA for testing/estimating a theoretical model of latent factors causing observed variables.

The following discussion uses PCA-based approach.

Dimension reduction methods

- Stepwise regression, ridge and lasso involve fitting linear regression models (by OLS or by parameter shrinkage) using the original predictors: $x_1, x_2, \ldots, x_p$.
- **Dimension reduction methods** transform the predictors and then fit a least squares model using the transformed variables:

1. **Principal components analysis (PCA)**: data (pre)processing, **feature extraction – dimension reduction with minimized information loss.** PCA output can be used in supervised methods of analysis (OLS).

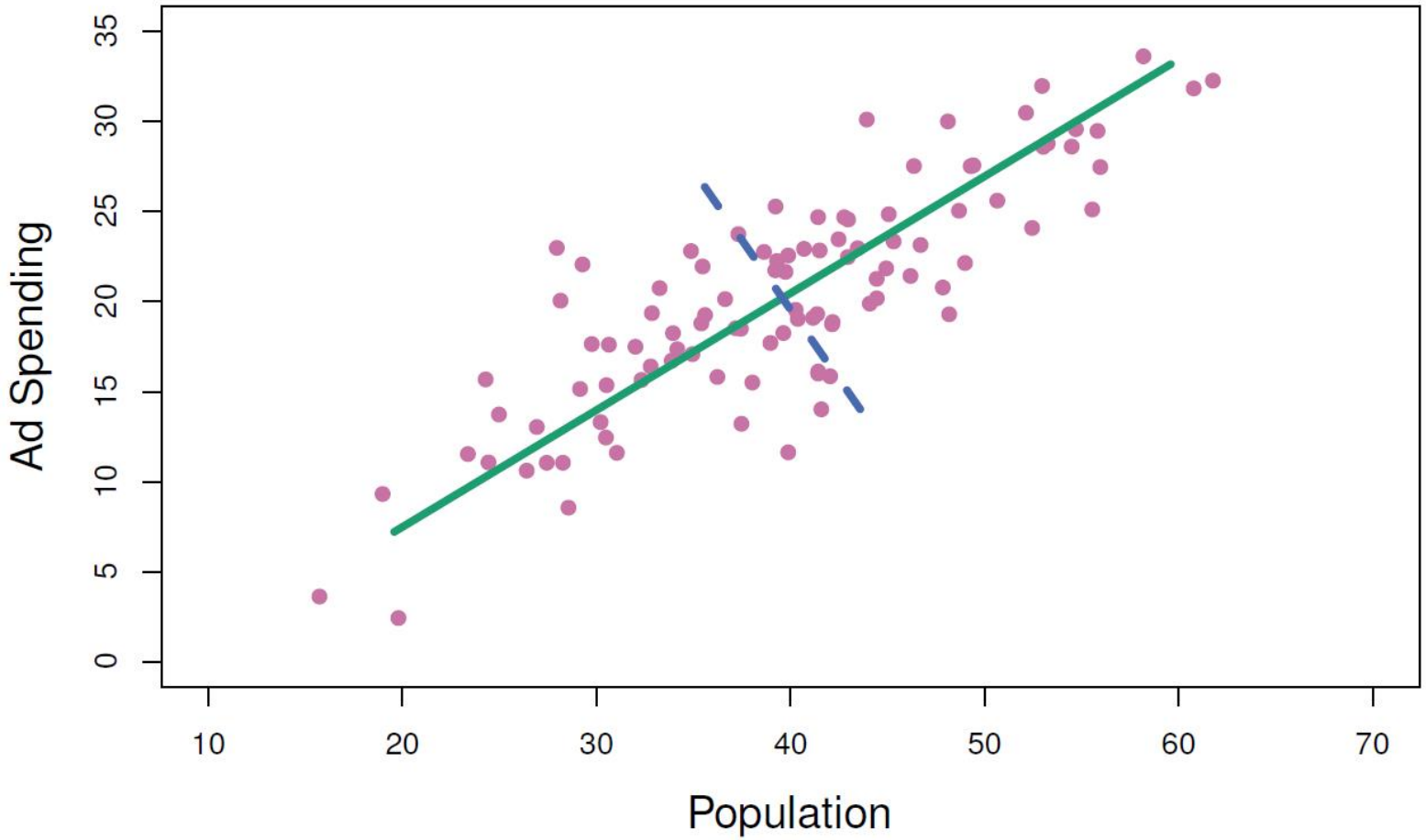
2. **Principal component regression (PCR)**: In the LRM, the potentially many correlated original variables are replaced with a small set of principal components that capture their joint variation.

Principal component analysis (PCA)

- PCA produces a low-dimensional representation of a dataset. It finds a sequence of linear combinations of the variables that have maximal variance and are mutually uncorrelated.
- Apart from producing derived variables for use in supervised learning problems, PCA also serves as a tool for data visualization.

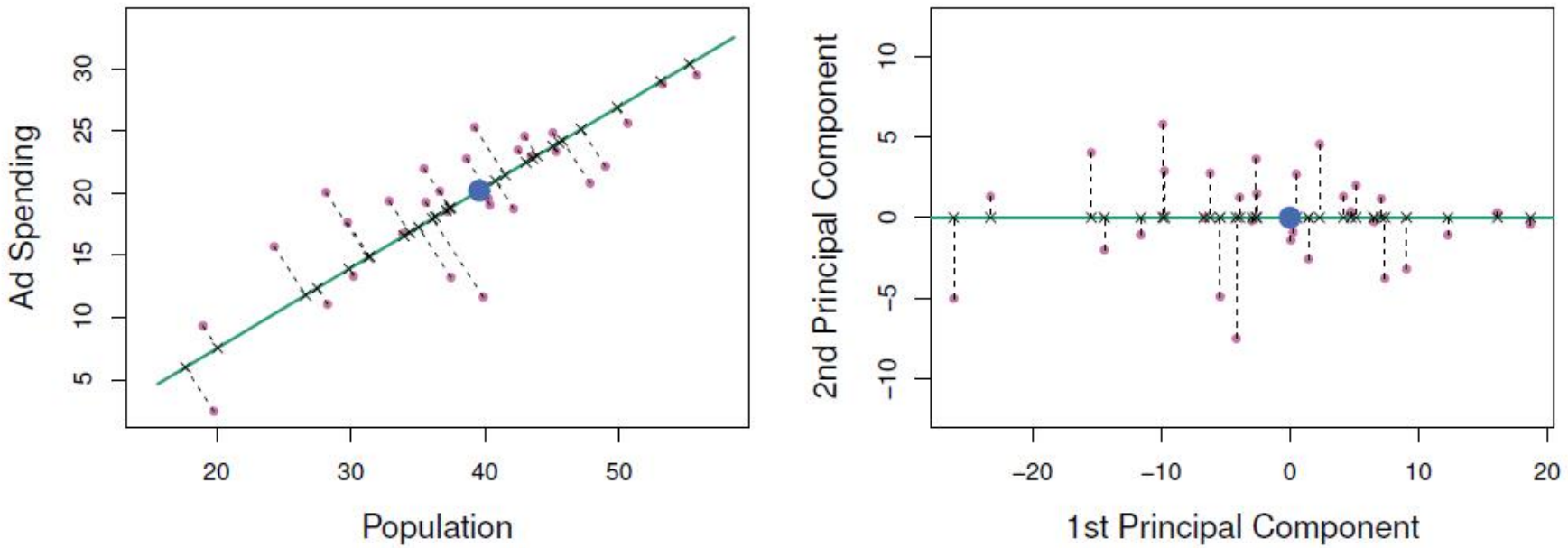
- Suppose we have a $(n \times p)$ dataset $\boldsymbol{X}$. Since we are mainly interested in variance here, we can assume that each of the variables in $\boldsymbol{X}$ has been **centered** to have mean zero (all column-means of $\boldsymbol{X}$ are zero). If necessary, the transformation (centering of $\boldsymbol{X}$) is straight-forward.
- Typically, empirical analyses (in `R` and elsewhere) would involve variable standardizing/scaling to $\text{var}(\boldsymbol{x}_j) = 1$.

PCA motivation & example



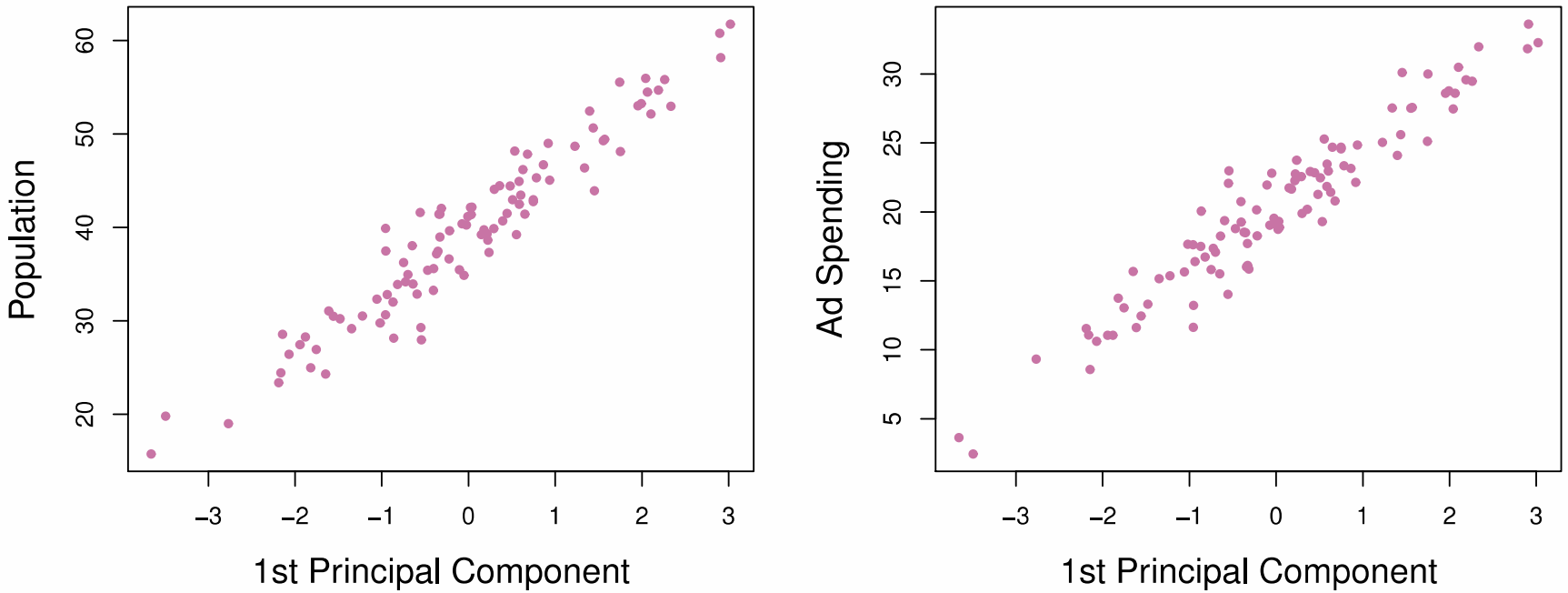
Sample dataset with 2 variables. Green line indicates the first principal component, Z_1. Along Z_1, data varies the most (out of all directions possible – in 2D). Blue dashed line indicates Z_2 (most variability orthogonal to Z_1).

PCA motivation & example



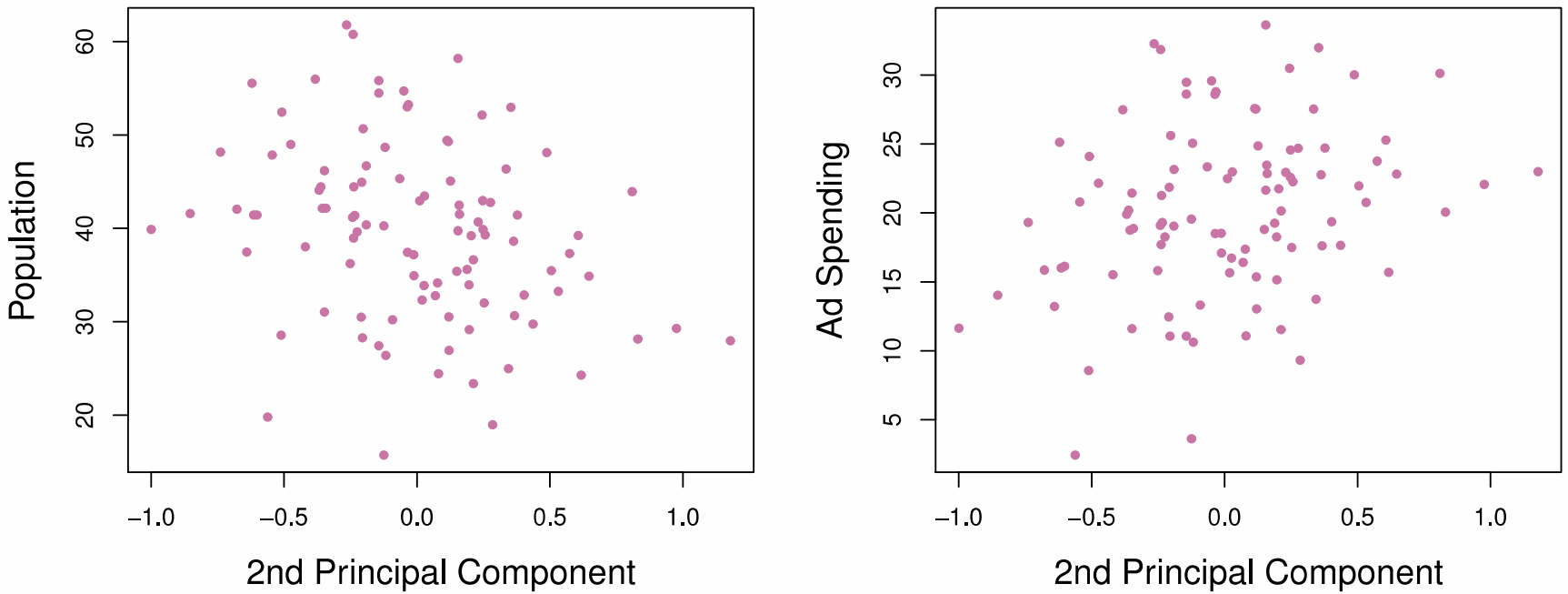
Sample dataset with 2 variables.
Z_1 minimizes the squared perpendicular distances to observed data.
Data vary most along Z_1 (data most spread-out along Z_1).
Values of $z_{i1} \in Z_1$ and $z_{i2} \in Z_2$ are shown as distances from "zero" (blue dot). Right panel/plot is rotated for readability.

PCA motivation & example



Sample dataset with 2 variables. First principal component is shown (on the x-axis) against `Population` and `Ad Spending` variables. Strong correlation is apparent in both plots, $\rightarrow$ Z_1 summarizes both series well and can be used as a single composite predictor for `Sales` (instead of the two observed regressors).

PCA motivation & example



Sample dataset with 2 variables. Second principal component is shown (on the x-axis) against `Population` and `Ad Spending` variables. There is little relationship between Z_2 and the two regressors. Hence, Z_1 apparently summarizes both (strongly correlated) regressors well enough.

Principal component analysis (PCA)

- 1^{st} principal component vector $\boldsymbol{z}_1$ of a set of centered variables $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_p$ (all $n \times 1$) is the normalized linear combination:

$$\boldsymbol{z}_1 = \phi_{11}\boldsymbol{x}_1 + \phi_{21}\boldsymbol{x}_2 + \cdots + \phi_{p1}\boldsymbol{x}_p$$

  that has the largest variance. Hence, we solve:

$$\underset{\phi_{11},\ldots,\phi_{p1}}{\text{maximize}} \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{j1}x_{ij}\right)^2 \quad s.t. \quad \sum_{j=1}^{p}\phi_{j1}^2 = 1 \tag{1}$$

- The ϕ_{j1} elements $\phi_{11}, \ldots, \phi_{p1}$ are *loadings* of the first principal component and they make up the first principal component loading vector, $\boldsymbol{\phi}_1 = (\phi_{11}, \phi_{21}, \ldots, \phi_{p1})'$.
- $\sum_{j=1}^{p}\phi_{j1}^2 = 1$ is the normalization condition: sum of squares of loadings is equal to one. Otherwise, setting $|\phi_{j1}|$ arbitrarily large leads to arbitrarily large variance.
- (1) is solvable by linear algebra (singular-value decomposition)

Principal component analysis (PCA)

- By solving (1), we obtain the linear combination of the sample variables of the form:

$$z_{i1} = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \cdots + \phi_{p1} x_{ip} \quad ; \quad i = 1, \ldots, n.$$

- $\boldsymbol{z}_1 = (z_{11}, z_{21}, \ldots, z_{n1})'$ is the first principal component.
- Since each of the $\boldsymbol{x}_j$ variables has mean zero, so does $\boldsymbol{z}_1$

  Hence, the sample variance of $\boldsymbol{z}_1$ can be calculated as $\frac{1}{n} \sum_{i=1}^{n} z_{i1}^2$.

Principal component analysis (PCA)

- The loading vector $\boldsymbol{\phi}_1$ with elements $\phi_{11}, \phi_{21}, \ldots, \phi_{p1}$ defines a direction in variable space (column space of $\boldsymbol{X}$), along which the data vary the most.
- **The second principal component** is the linear combination of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p$ that maximizes variance among all linear combinations that are **uncorrelated** with $\boldsymbol{z}_1$. Hence, we add orthogonality condition to (1) and repeat the optimization.
- The second principal component $\boldsymbol{z}_2$ and its elements $z_{12}, z_{22}, \ldots z_{n2}$ take the form:

$$z_{i2} = \phi_{12} x_{i1} + \phi_{22} x_{i2} + \cdots + \phi_{p2} x_{ip} \quad ; \quad i = 1, \ldots, n.$$

where $\boldsymbol{\phi}_2 = (\phi_{12}, \phi_{22}, \ldots, \phi_{p2})'$ is the second principal component loading vector.

Principal component analysis (PCA)

- Constraining $\boldsymbol{z}_2$ to be uncorrelated with $\boldsymbol{z}_1$ is equivalent to constraining the direction $\boldsymbol{\phi}_2$ to be orthogonal (perpendicular) to the direction $\boldsymbol{\phi}_1$.
- Subsequent principal components:

  For a sequence of additional $\boldsymbol{z}_2, \boldsymbol{z}_3, \ldots$ principal components, we solve (1) while adding orthogonality condition with respect to all preceding principal components.
- Important geometrical interpretations to principal components apply (see ISLR textbook).

Principal component analysis (PCA)

Proportion of variance explained by principal components

- To understand the "strength" of each principal component, we calculate the proportion of variance explained by each component.
- **Total variance present in a data set** (assuming $\boldsymbol{X}$ matrix $(n \times p)$ of centered variables $\boldsymbol{x}_j$ with mean zero) is defined as:

$$\sum_{j=1}^{p} \text{var}\left(\boldsymbol{x}_j\right) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2$$

- **Variance explained** by the m-th principal component is:

$$\text{var}\left(\boldsymbol{z}_m\right) = \frac{1}{n} \sum_{i=1}^{n} z_{im}^2$$

- $\sum_{j=1}^{p} \text{var}\left(\boldsymbol{x}_j\right) = \sum_{m=1}^{M} \text{var}\left(\boldsymbol{z}_m\right)$, where $M = \min(n-1, p)$.
  i.e. if all PC are used, they explain 100 % of variance in $\boldsymbol{X}$.

Principal component analysis (PCA)

Proportion of variance explained (PVE)

- PVE of the m-th principal component $\boldsymbol{z}_m$ lies between 0 and 1 and it is defined as:

$$\text{PVE}_m = \frac{\sum_{i=1}^{n} z_{im}^2}{\sum_{j=1}^{p} \sum_{i=1}^{n} x_{ij}^2} \, .$$

- Also,

$$\sum_{m=1}^{M} \text{PVE}_m = 1 \, ,$$

  i.e. PVEs sum to 1 and we can display & interpret cumulative PVEs.

Principal component analysis (PCA)

R example:

```
pca1 = princomp(x, scores=TRUE, cor=TRUE) # x has 7 columns
summary(pca1)
```

```
## Importance of components:
##                           Comp.1     Comp.2     Comp.3      Comp.4
## Standard deviation     1.9036937  1.0423367  0.81837919  0.75632747
## Proportion of Variance 0.5177214  0.1552094  0.09567779  0.08171875
## Cumulative Proportion  0.5177214  0.6729308  0.76860854  0.85032729
##                           Comp.5     Comp.6     Comp.7
## Standard deviation     0.64958592 0.56978592 0.54871770
## Proportion of Variance 0.06028027 0.04637943 0.04301302
## Cumulative Proportion  0.91060756 0.95698698 1.00000000
```

- The number of components is also the number of variables (if $n > p$).
- Proportion of variance: Eg. if $\text{PVE}_1 = .52$, $\boldsymbol{z}_1$ explains 52% of variance in $\boldsymbol{X}$.
- Cumulative Proportion: PVE by $\boldsymbol{z}_m$ and previous components.
- Standard deviation = eigenvalues
- How many components to use in PCR? Choose the components with eigenvalues equal or higher than 1. (or use cross-validation)

Kaiser-Meyer-Olkin (KMO) statistic

PCA can perform a compression of the available information (reduce dimension) only if we can "reject" independence (orthogonality) among variables $\boldsymbol{x}_j$ in $\boldsymbol{X}$. Individual KMO (for j-th variable):

$$KMO_j = \frac{\sum_{i \neq j} r_{ij}^2}{\sum_{i \neq j} r_{ij}^2 + \sum_{i \neq j} a_{ij}^2}; \qquad KMO_j \in \langle 0, 1 \rangle$$

Overall KMO:

$$KMO = \frac{\sum_j \sum_{i \neq j} r_{ij}^2}{\sum_j \sum_{i \neq j} r_{ij}^2 + \sum_j \sum_{i \neq j} a_{ij}^2}; \qquad KMO \in \langle 0, 1 \rangle$$

where:
$\{r_{ij}\} = \boldsymbol{R}$, which is a correlation matrix (here, i, j denote variables),
$\{a_{ij}\} = \boldsymbol{A}$, which is a partial correlation matrix (partial correlations represent the direct interactions between two variables, with the indirect effects of all remaining variables removed)
$a_{ij} = -\frac{v_{ij}}{\sqrt{v_{ii} \cdot v_{jj}}}$ where $\{v_{ij}\} = \boldsymbol{V} = \boldsymbol{R}^{-1}$

Kaiser-Meyer-Olkin (KMO) statistic

KMO description

- *KMO* compares correlations between variables against their partial correlations.
- If partial correlations a_{ij} are near zero, PCA can perform efficiently, because the variables are highly related and $KMO \approx 1$.
- If *KMO* is low (KMO ≈ 0), PCA is not relevant.
  In empirical applications, PCA is gerally not usefull if $KMO < 0.5$.

KMO-based variable selection:

- Overall *KMO* should be .60 or higher (ideally over 0.90).
- If it is not, drop the variables with the lowest individual KMO_j values, until overall *KMO* rises above .60.
- This approach requires that we start with multiple variables/regressors in our dataset; at least $p > 5$.
- Alternative: Bartlett's test in `R`: cortest.bartlett() in {psych}.

Principal component regression (PCR)

PCR motivation:

If we have many correlated original variables as regressors in a LRM, we can replace them with a small set of principal components that capture their joint variation.

- Variance-Bias tradeoff benefits
- Models unsuitable for LRM-like parameter interpretation

Principal component regression (PCR)

- Using PCA, we linearly transform our dataset of predictors $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_p$ into $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_M$ variables where $M < p$. The PCA transformation can be outlined as follows:

$$\boldsymbol{z}_m = \boldsymbol{X}\boldsymbol{\phi}_m \qquad \text{where} \qquad z_{im} = \sum_{j=1}^{p} \phi_{jm}\, x_{ij}, \tag{2}$$

  for some constant parameters $\phi_{m1}, \ldots, \phi_{mp}$.
- Now, we can use OLS to fit a LRM:

$$y_i = \theta_0 + \sum_{j=1}^{M} \theta_m z_{im} + \varepsilon_i, \tag{3}$$

- Note that in model (3), the regression coefficients are given as $\theta_0, \ldots, \theta_M$. If the constants $\phi_{1m}, \phi_{2m}, \ldots, \phi_{pm}$ are chosen wisely (PCA), then such dimension reduction approaches can often outperform OLS regression in terms of CV errors, etc.

Principal component regression (PCR)

From equation/definition (2), we can write

$$\sum_{m=1}^{M} \theta_m z_{im} = \sum_{m=1}^{M} \theta_m \sum_{j=1}^{p} \phi_{jm} x_{ij} = \sum_{j=1}^{p} \sum_{m=1}^{M} \theta_m \phi_{jm} x_{ij} = \sum_{j=1}^{p} \beta_j x_{ij}$$

where

$$\beta_j = \sum_{m=1}^{M} \theta_m \phi_{jm}. \tag{4}$$

- Therefore, model (3) can be thought of as a special case of the original linear regression model.
- Dimension reduction serves to constrain the estimated β_j coefficients, since now they must take the form (4).
- This approach can have significant benefits in terms of bias-variance tradeoff.

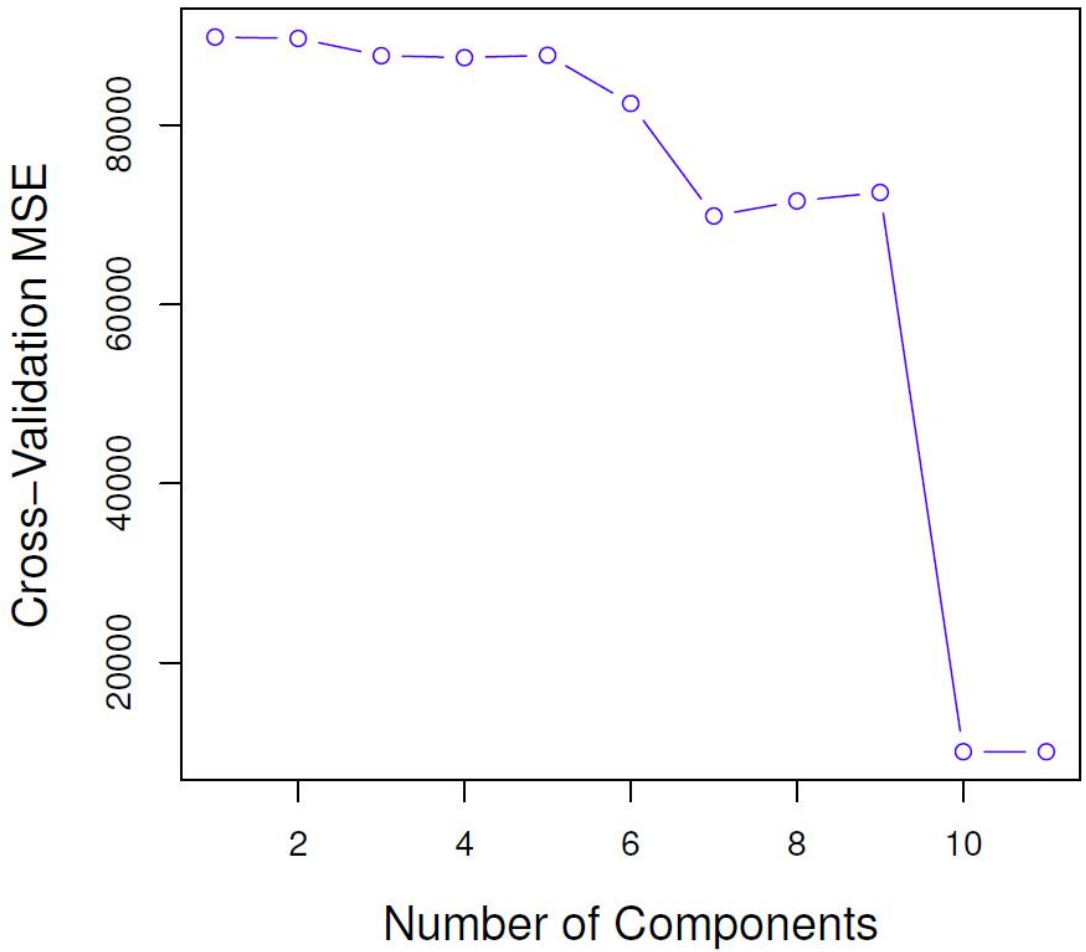
Principal component regression (PCR)

PCR: algorithm

- First, we apply principal components analysis (PCA) to find suitable linear combinations of predictors for use in our regression.
  - The first principal component is the (normalized) linear combination of the regressors that has the largest variance.
  - The second principal component has largest variance, subject to being uncorrelated with the first.
  - And so on.
- The dependent variable is then regressed on few principal components, rather than many original regressors.

  The optimal number of principal components can be assessed using cross validation.

Principal component regression (PCR)



Sample data, selection of the number of components.
In this particular illustration, PCR would provide little improvement over OLS (this may happen often for $n \gg p$ datasets).

Principal component regression (PCR)

PCR: final discussion

- PCA identifies linear combinations (directions) that best represent the predictors $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_p$.
- These directions are identified in an **unsupervised** way, since the response y is not used to help determine the principal component directions. i.e. the response does not supervise the identification of the principal components.
- PCR suffers from a potentially serious drawback: there is no guarantee that the directions that best explain the predictors will also be the best directions to use for predicting the response.

Potential solutions to the problem:

- Partial least squares (ISLR, ch. 6.3.2)

Partial least squares (PLS)

Partial least squares (PLS)

- Much like with the PCR method, in PLS we also search for convenient (aggregating) linear combinations of regressors in matrix $\boldsymbol{X}$.
- PLS is not scale-invariant, so we assume each $\boldsymbol{x}_j$ regressor is standardized – much the same way as in PCR.
- PLS, unlike PCR, uses *supervised* identification of the components: both $\boldsymbol{X}$ and $\boldsymbol{y}$ are used when searching for linear combinations of regressors.
- PLS-based linear combinations of $\boldsymbol{x}_j$ ("directions") not only approximate the original (correlated) data in $\boldsymbol{X}$, but are also related to the response $\boldsymbol{y}$.

Partial least squares (PLS)

First component for PLS:

$$\boldsymbol{z}_1^{\text{PLS}} = \boldsymbol{X}\boldsymbol{\psi}_1 \qquad \text{where} \qquad z_{i1}^{\text{PLS}} \sum_{j=1}^{p} \psi_{j1} x_{ij}$$

and coefficients ψ_{j1} are calculated in two steps:

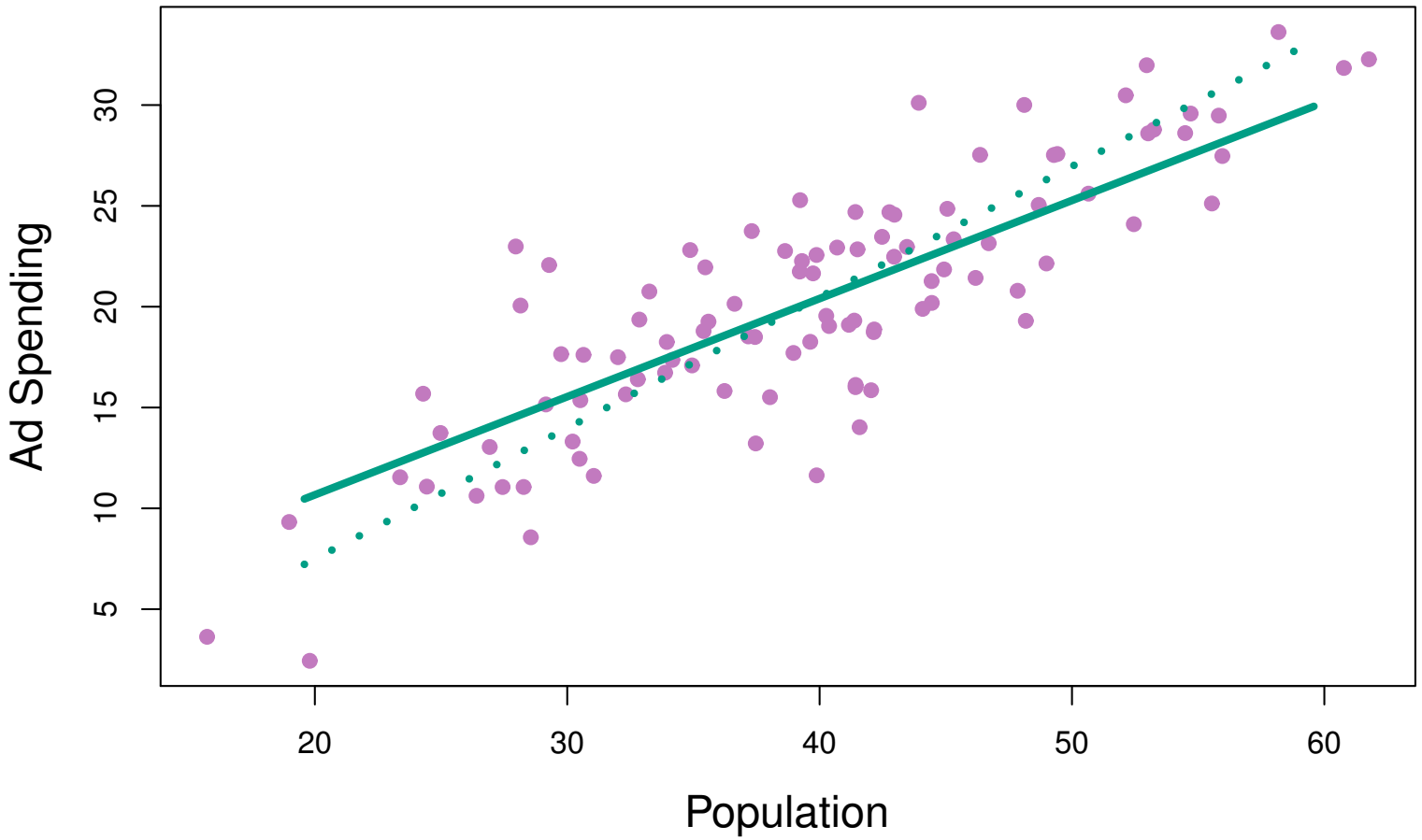
1. Use OLS to estimate slope-coefficients ψ_{j1} of the "simple" linear regressions $\boldsymbol{y} \leftarrow \boldsymbol{x}_j$.

   Here, $j = 1, \ldots, p$ separate SLRM are estimated and corresponding ψ slope-coefficients are recorded.

2. Standardize the ψ_{j1} coefficients so that $\sum_{j=1}^{p} \psi_{j1}^2 = 1$.

Note that with PLS, highest weights are on variables ($\boldsymbol{x}_j$) that are most related to the response.

Partial least squares (PLS) – illustration



Sample dataset with 2 variables. The first PLS direction (component) – solid line – is shown. Compare to the PCA/PCR first direction (component) – dotted line (shown previously).

Partial least squares (PLS)

Second component for PLS:

1. We regress each variable on $\boldsymbol{z}_1^{\text{PLS}}$ and take residuals
   ($\boldsymbol{x}_1 \leftarrow \boldsymbol{z}_1^{\text{PLS}}$ and save OLS residuals $\ddot{\boldsymbol{x}}_1$; repeat for $\boldsymbol{x}_2$, etc.)

   Individual $\ddot{\boldsymbol{x}}_j$ residuals can be interpreted as the "remaining" information of $\boldsymbol{x}_j$ that is not explained by the first PLS direction (component).

2. Compute $\boldsymbol{z}_2^{\text{PLS}}$ using the orthogonalized data ($\ddot{\boldsymbol{X}}$),
   the same way as the first component.
   (Run all $\boldsymbol{y} \leftarrow \ddot{\boldsymbol{x}}_j$ regressions and standardize coefficients).

By analogy, this procedure can be repeated for all subsequent components.

Partial least squares (PLS)

The supervised dimension reduction in PLS can reduce bias (compared to PCR).

However, it can also increase variance of predictions. Hence, the benefits of using PLS over PCR can be outweighted by drawbacks (kFCV may be used to compare the two methods).

PCR/PLS – detailed technical description and estimation algorithm:

- ({pls} package manual)
- (The Elements of Statistical Learning, ch. 3.5.1—3.5.2)

Moving beyond linearity

Moving beyond linearity

Different approaches are discussed (from simple to more complex), QREG is included mostly for reference.

- Quantile regression (quick repetition)
- Polynomial and step (piecewise-constant) regression
- Regression splines
- Smoothing splines
- Generalized Additive Models (GAM)

QREG - quick repetition

- Quantile regression estimates the relationship between regressors and a specified quantile of dependent variable.
- One important special case of quantile regression is the least absolute deviations (LAD) estimator, which corresponds to fitting the conditional median of the response variable ($q = \frac{1}{2}$).
- QREG (LAD) estimator can be motivated as a robust alternative to OLS (with respect to outliers).
- Linear programming can be used for finding QREG estimates (Koenkerr and Bassett (around 1980).
- QREG-based predictions: Conditional quantiles (expected values) can be produced.
- To optimize prediction properties, QREG may be combined with parameter-shrinkage methods (lasso, ridge).

Quantile regression (QREG)

For LRMs, the q-th quantile regression estimator $\boldsymbol{\beta}_q$ minimizes:

$$\min_{\hat{\boldsymbol{\beta}}_q}: \quad Q_n(\hat{\boldsymbol{\beta}}_q) = \sum_{i:\, e_i \geq 0}^{n} q|y_i - \boldsymbol{x}_i\hat{\boldsymbol{\beta}}_q| \; + \sum_{i:\, e_i < 0}^{n} (1-q)|y_i - \boldsymbol{x}_i\hat{\boldsymbol{\beta}}_q|,$$

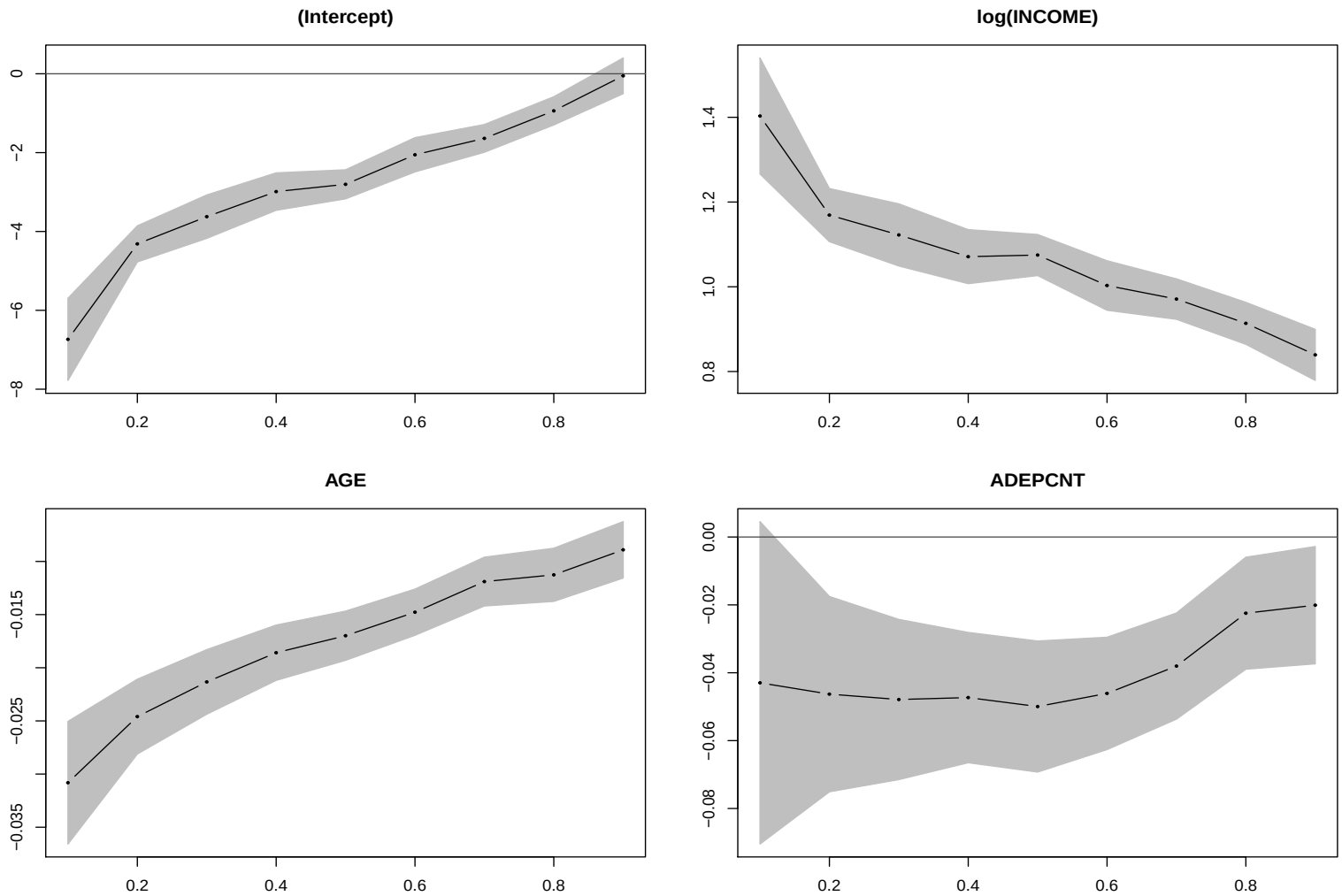
where $e_i = (y_i - \boldsymbol{x}_i\hat{\boldsymbol{\beta}}_q)$.

- We use the notation $\hat{\boldsymbol{\beta}}_q$ to make clear that different choices of q lead to different $\hat{\boldsymbol{\beta}}$.
- Slope of the loss function Q_n is asymmetrical (around $e_i = 0$).
- The loss function is not differentiable (at $e_i = 0$) $\rightarrow$ gradient methods are not applicable (linear programming can be used).

Quantile regression example

Example 7.10 (Greene): Income Elasticity of Credit Cards Expenditure

```
CCexpend ← log(INCOME) + AGE + #DEPENDANTS
```



Polynomial regression

Standard way to apply LRMs in situations where the relationship between a regressor and a dependent variable is non-linear.

- Simple linear regression model

$$y_i = \beta_0 + \beta_1 x_i + u_i,$$

- is replaced by a polynomial function (linear in parameters):

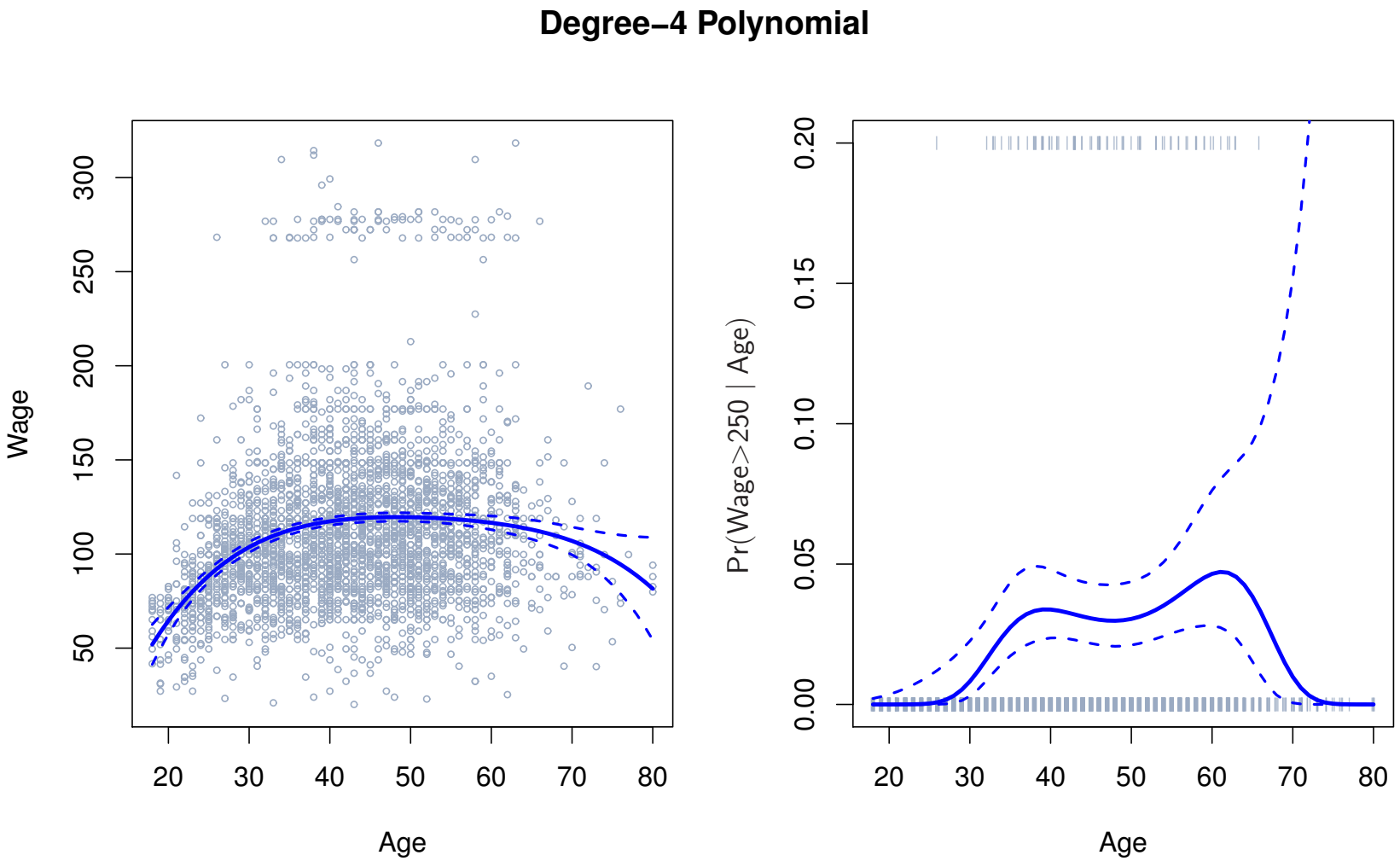
$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots + \beta_d x_i^d + u_i.$$

- This approach extends easily to logistic regression models if y_i is binary (applies to count and similar LDVs by analogy):

$$\Pr(y_i = 1|x_i) = \frac{\exp(\beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots + \beta_d x_i^d)}{1 + \exp(\beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots + \beta_d x_i^d)}$$

Polynomial regression (LRM & logit examples)

`Wage` $\leftarrow$ `Age` `1[Wage>250]` $\leftarrow$ `Age`



Polynomial regression

- Mostly, we are not interested in exact values of coeffcients. Usually, we seek to obtain "good" fitted values for some x_0:
  $\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0 + \hat{\beta}_2 x_0^2 + \cdots + \hat{\beta}_d x_0^d$
  and we study the general prediction properties (variance, bias) of $\hat{f}(x_0)$.
- Either fix the d at some low value (4 at most), or use cross-validation to choose d.
- Caveat: polynomials have notorious tail behavior – bad for extrapolation.
- We can use polynomials for several separate regressors in a LRM: we just stack the variables into one $\boldsymbol{X}$ matrix (see GAMs next).

Step (piecewise-constant) regression

- Polynomial regression function imposes a *global* structure on the non-linear function.
- Instead, we can use a step function where regressor x is broken into different bins and we fit a different constant for each bin (our regressor is transformed into a ordered categorical variable).
- Alternatively, we select adequate (ad-hoc) cutpoints $c_1, c_2, \ldots, c_K$ for a continuous regressor x and construct $K+1$ dummy variables:

$$\begin{aligned} C_0(X) &= 1[X < c_1], \\ C_1(X) &= 1[c_1 \leq X < c_2], \\ &\vdots \\ C_{K-1}(X) &= 1[c_{K-1} \leq X < c_K], \\ C_K(X) &= 1[c_K \leq X] \end{aligned}$$

Step (piecewise-constant) regression

- Next, we use $C_0(X), C_1(X), C_2(X), \ldots, C_K(X)$ to fit LRM:

$$y_i = \beta_0 + \beta_1 C_1(x_i) + \beta_2 C_2(x_i) + \cdots + \beta_K C_K(x_i) + u_i.$$

  C_0 is excluded from the model to avoid perfect multicolinearity. Exclusion is arbitrary – any of the dummies may be excluded. Alternatively, we may include all dummies and drop β_0.

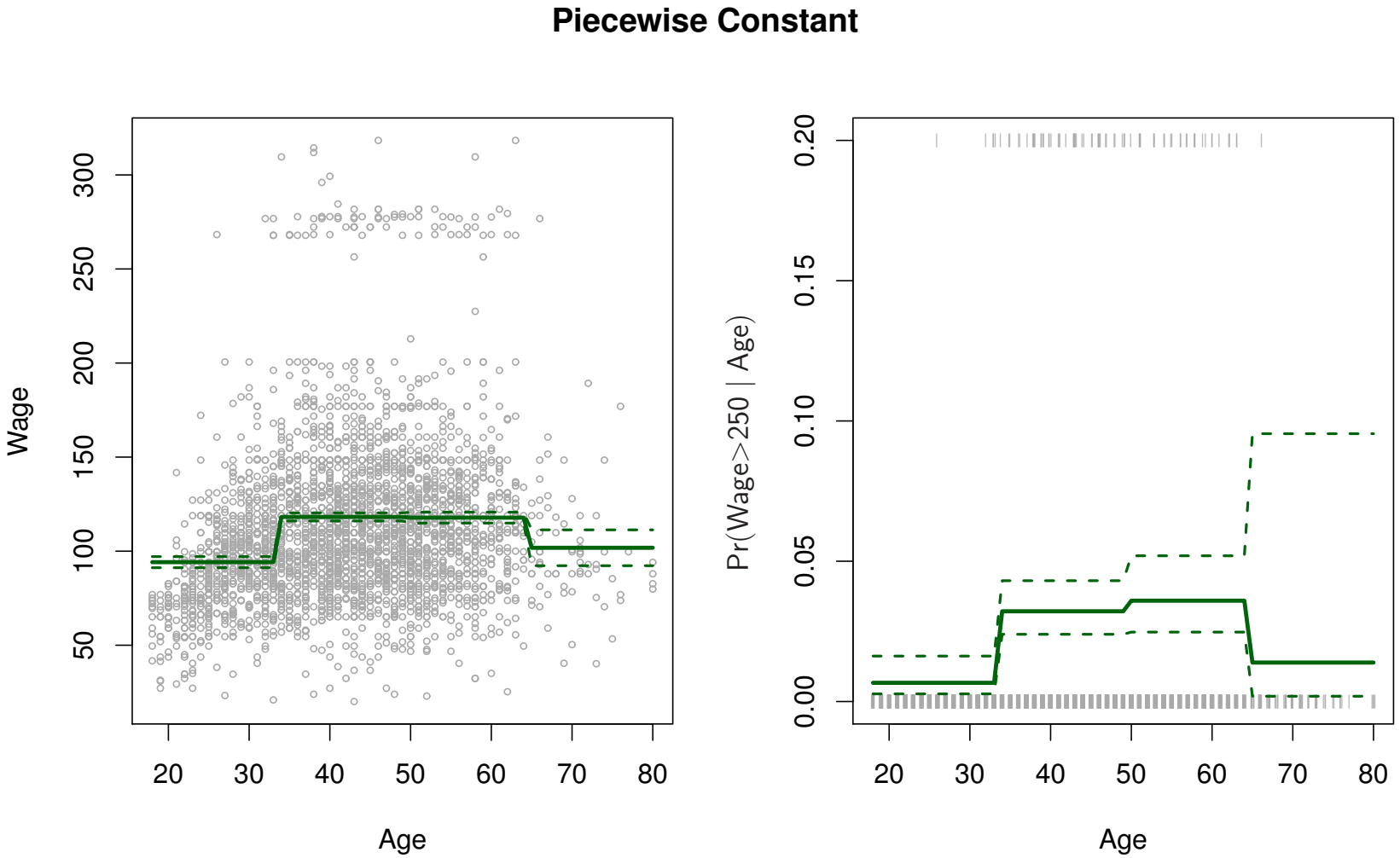
- Again, this approach extends easily to logistic regression models if y_i is binary (applies to count and similar LDVs by analogy):

$$\Pr(y_i = 1|x_i) = \frac{\exp[\beta_0 + \beta_1 C_1(x_i) + \beta_2 C_2(x_i) + \cdots + \beta_K C_K(x_i)]}{1 + \exp[\beta_0 + \beta_1 C_1(x_i) + \beta_2 C_2(x_i) + \cdots + \beta_K C_K(x_i)]}.$$

Piecewise-constant regression (LRM & logit examples)

`Wage` ← `Age`

`1[Wage>250]` ← `Age`



Step (piecewise-polynomial) regression

Combines polynomial regression with step (piecewise) regression.

Piecewise-polynomial regression:
Fit separate (low-degree) polynomials over different regions of X (instead of one high-degree polynomial over the entire range of X).

Example: piecewise cubic polynomial with a single knot (cutpoint) c_1:

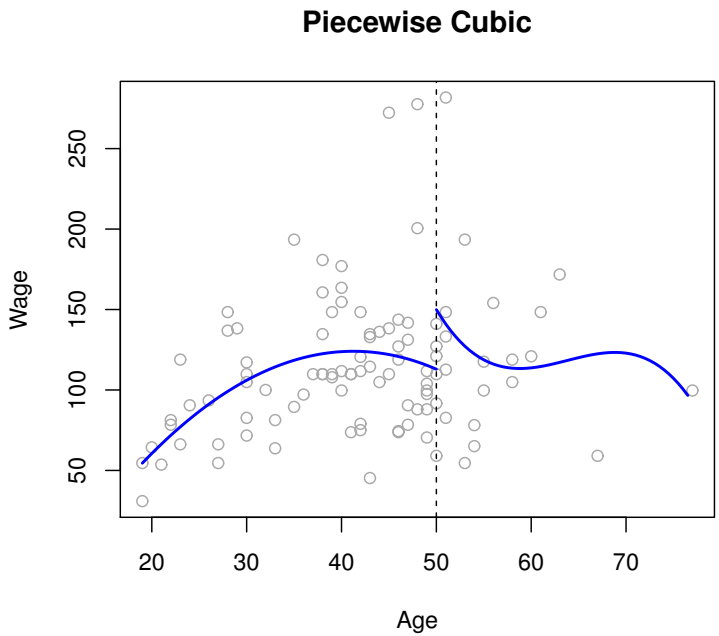
$$y_i = \begin{cases} \beta_{01} + \beta_{11} x_i + \beta_{21} x_i^2 + \beta_{31} x_i^3 + u_i & \text{if } x_i < c_1; \\ \beta_{02} + \beta_{12} x_i + \beta_{22} x_i^2 + \beta_{32} x_i^3 + u_i & \text{if } x_i \geq c_1. \end{cases}$$

- Here, we fit two different polynomials to the data.
- For K knots (cutpoints), we fit $K + 1$ different polynomials.
  (i.e. with one knot, twice as much parameters are estimated)
- Different d are possible: $d = 1 \rightarrow$ piecewise linear function.

Step (piecewise-polynomial) regression

Example: piecewise cubic polynomial with one knot:

$$\texttt{Wage} \leftarrow \texttt{Age}$$

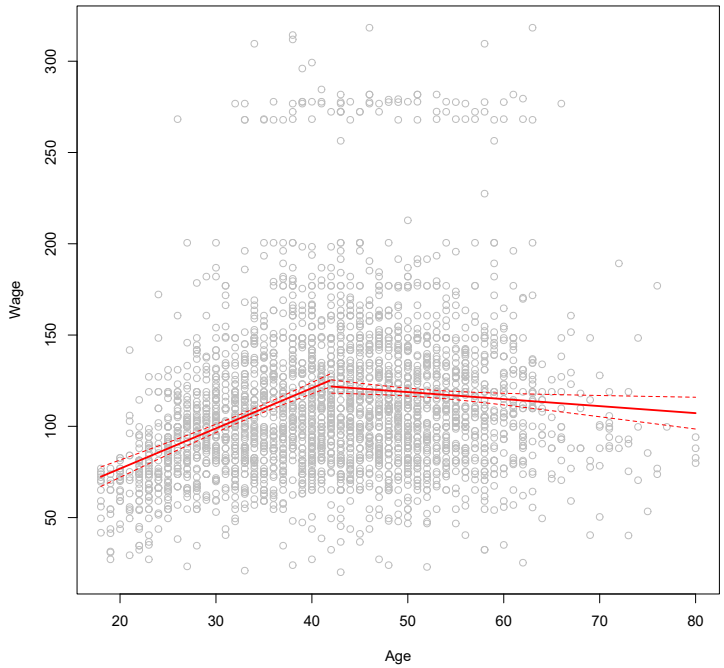


✓ Improved flexibility

✗ Fitted curve non-continuous

Regression splines - motivation

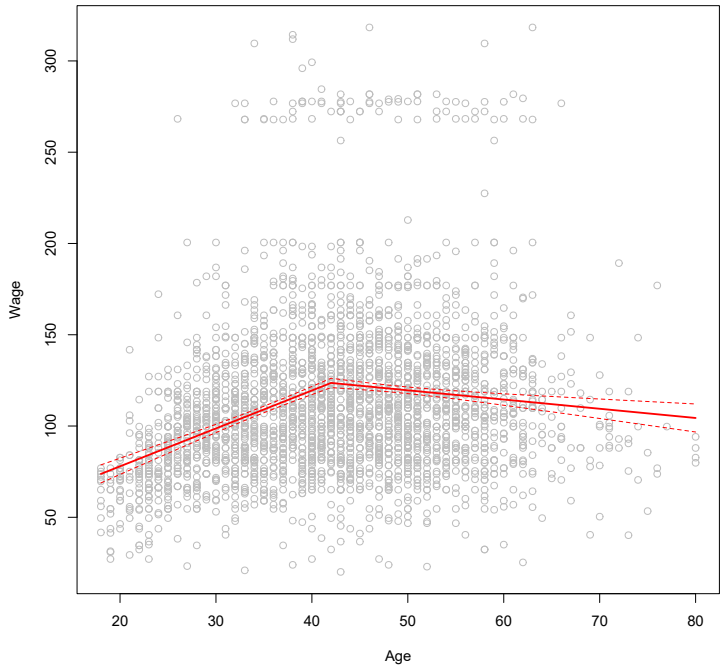
Piecewise linear regression with one knot at c_1:



$$y_i = \begin{cases} \beta_{01} + \beta_{11} x_i + u_i & \text{if} \quad x_i < c_1; \\ \\ \beta_{02} + \beta_{12} x_i + u_i & \text{if} \quad x_i \geq c_1. \end{cases}$$

Regression splines

Piecewise linear "continuous" regression with one knot at c_1:



"Continuity" can be easily imposed as follows:

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 \{1[x_i > c_1](x_i - c_1)\} + u_i$$

Regression splines (linear spline)

Linear spline: fit regression line in each region of the predictor space, requiring continuity in each knot. Regression model with K knots $\xi_k, k = 1, \ldots, K$ can be represented using basis functions as follows:

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \cdots + \beta_{K+1} b_{K+1}(x_i) + u_i$$

where b_k are the basis functions:

$$b_1(x_i) = x_i\,,$$
$$b_2(x_i) = (x_i - \xi_1)_+ = 1[x_i > \xi_1](x_i - \xi_1)\},$$
$$b_3(x_i) = (x_i - \xi_2)_+\,,$$
etc.

- The $(x_i - \xi_k)_+$ means the positive part of the difference.
- With linear splines, each knot only adds one estimated parameter to the estimation, i.e. $K+2$ d.f. are used (compare to non-continuous piecewise linear regression: $2K$ d.f. used).
- Basis functions (notation) will be useful in subsequent discussion.

Regression splines (cubic spline)

Cubic spline with knots $\xi_k,\ k = 1, \ldots, K$ is a piecewise cubic polynomial with continuous derivatives up to order 2 at each knot.

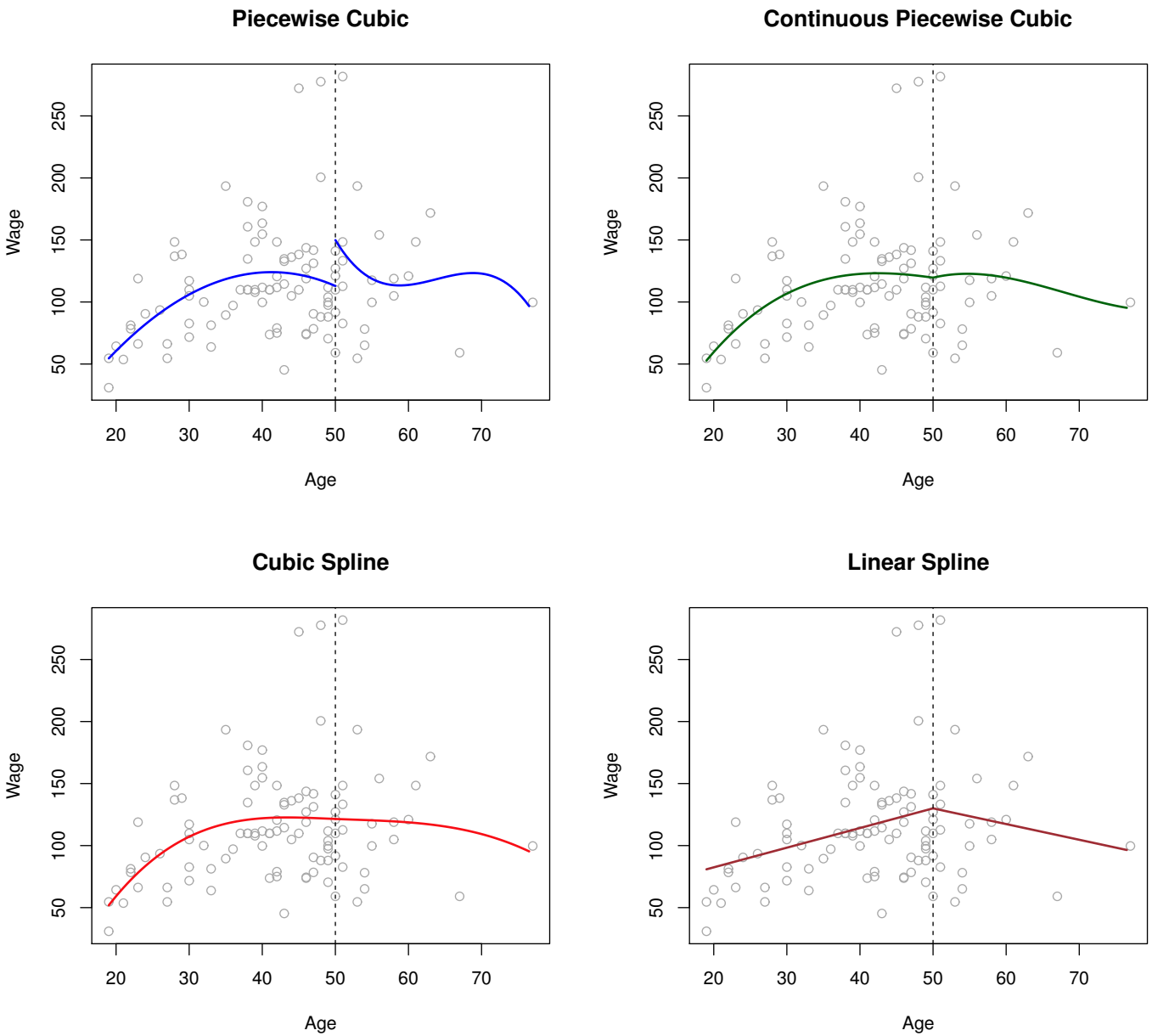
Again we can represent this model with truncated power basis functions:

$$y_i = \beta_0 + \beta_1\, h_1(x_i) + \beta_2\, h_2(x_i) + \cdots + \beta_{K+3}\, h_{K+3}(x_i) + u_i$$

where:

$$\begin{aligned}
&h_1(x_i) = x_i\,, \\
&h_2(x_i) = x_i^2\,, \\
&h_3(x_i) = x_i^3\,, \\
&h_4(x_i) = (x_i - \xi_1)_+^3 \qquad \text{(note that } h_4 = h_{k+3})\,, \\
&h_5(x_i) = (x_i - \xi_2)_+^3 \ \ , \\
&\ldots \\
&h_{K+3}(x_i) = (x_i - \xi_K)_+^3\,.
\end{aligned}$$

Regression splines (cubic spline)



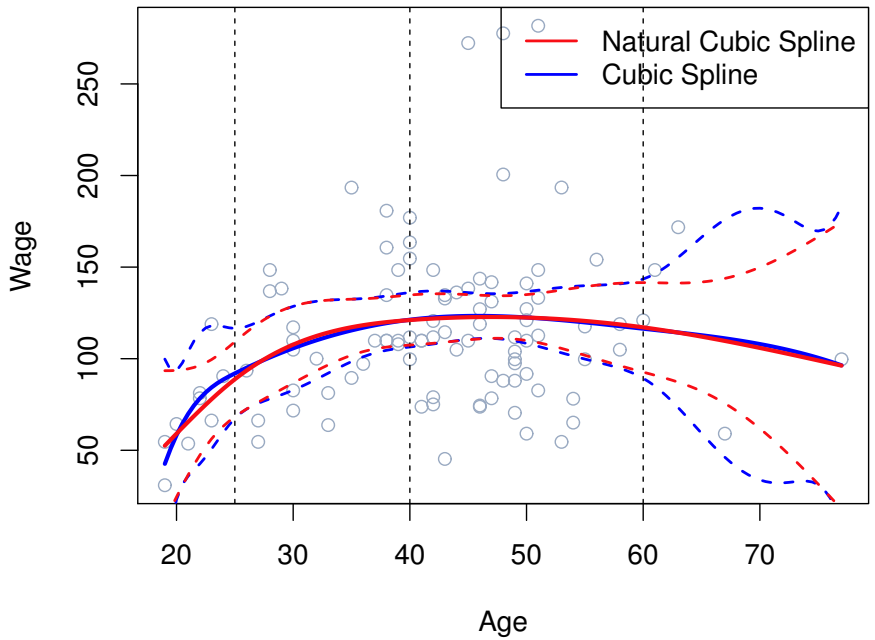
Regression splines (cubic spline)

- Cubic spline uses $4 + K$ d.f.
- Cubic spline has continuous derivatives up to order 2 at each knot. d-spline (degree-d spline) has continuous derivatives up to degree $d - 1$ at each knot.
- Cubic spline popularity is based on human perception: this is the lowest-order spline where knot-discontinuity is not visible to the human eye.
- Unfortunately, d-splines (incl. cubic splines) tend to have high variance at the outer range of the predictor (i.e. when X is very low or very high). Possible solution: use "natural splines" with additional boundary constraints.

Regression splines (natural spline)

Natural cubic spline: a cubic regression spline with additional boundary constraints:

- For a cubic spline regression model $y \leftarrow x$ with K knots, $f(X)$ is required to be linear for $X \leq \xi_1$ and for $X > \xi_K$.
- Additional restrictions free 4 d.f. (compared to cubic spline), so we use K d.f. for estimation.



Regression splines (natural spline)

Example: For $K=4$ and using the truncated power bases of cubic splines $(X-\xi_k)_+^3$, we can write:

$$y_i = \beta_0\, N_1(x_i) + \beta_1\, N_2(x_i) + \beta_2\, N_3(x_i) + \beta_3\, N_4(x_i) + u_i$$

where:

$$N_1(x_i) = 1 \qquad (\beta_0 \text{ is the intercept}),$$
$$N_2(x_i) = x_i\,,$$
$$N_3(x_i) = d_1(X) - d_3(X) \qquad (\text{note that } N_3 = N_{k+2}),$$
$$N_4(x_i) = d_2(X) - d_3(X)$$

and $d_k(X) = \dfrac{(X-\xi_k)_+^3 - (X-\xi_K)_+^3}{\xi_K - \xi_k}$.

For basis functions, we only use knots up to $K-2$ (i.e. ξ_1, ξ_2 for $K=4$).

Regression splines (natural spline)

General notation for natural cubic splines with K knots:

$$y_i = \beta_0 N_1(x_i) + \beta_1 N_2(x_i) + \beta_2 N_{k+2}(x_i) + \cdots + \beta_{K-3} N_{K-2}(x_i) + u_i$$

where:

$$N_1(x_i) = 1\,,$$
$$N_2(x_i) = x_i\,,$$
$$N_{k+2}(x_i) = d_k(X) - d_{K-1}(X)\,,$$
$$\cdots$$
$$N_{K-2}(x_i) = d_{K-2}(X) - d_{K-1}(X)\,,$$

and $d_k(X) = \dfrac{(X-\xi_k)_+^3 - (X-\xi_K)_+^3}{\xi_K - \xi_k}$.

All basis functions $N_1, N_2, N_{k+2}, \ldots, N_{K-2}$ have 2^{nd} and 3^{rd} derivatives equal to zero for $X < \xi_1$ and for $X \geq \xi_K$.

Regression splines (knot selection)

Where should we place the knots?

- Specify the number of knots and use uniformly distributed quantiles.
  - 1 knot: at median, 2 knots: at $q_{0.33}$ and $q_{0.66}$, etc.
  - Optimum number of knots can be determined by kFCV.
- Spline function is most flexible in regions that contain a lot of knots. Hence:
  - Place more knots in regions where we assume the function may vary most rapidly.
  - Place fewer knots where the function seems more stable.
  - Ad-hoc and potentially misleading approach.

Smoothing splines (quick overview)

- Smoothing splines – alternative approach to splines.
- Search for a function $g(x)$ that minimizes RSS and is smooth.

$$\min: \quad \text{RSS}(g, \lambda) = \sum_{i=1}^{n} [y_i - g(x_i)]^2 + \lambda \int g''(t)^2 \, dt$$

- $\sum_{i=1}^{n} [y_i - g(x_i)]^2$ is a loss function, makes $g(x)$ fit the data.
  $g(x)$ can be any function for which the second term is defined.
- $\lambda \int g''(t)^2 \, dt$ penalizes high variability in $g(x)$.
  λ is a penalty term. For $\lambda = \infty$, smoothing splines lead to OLS fit.

Smoothing splines (quick overview)

$$\min: \quad \text{RSS}(g, \lambda) = \sum_{i=1}^{n} [y_i - g(x_i)]^2 + \lambda \int g''(t)^2 \, dt$$

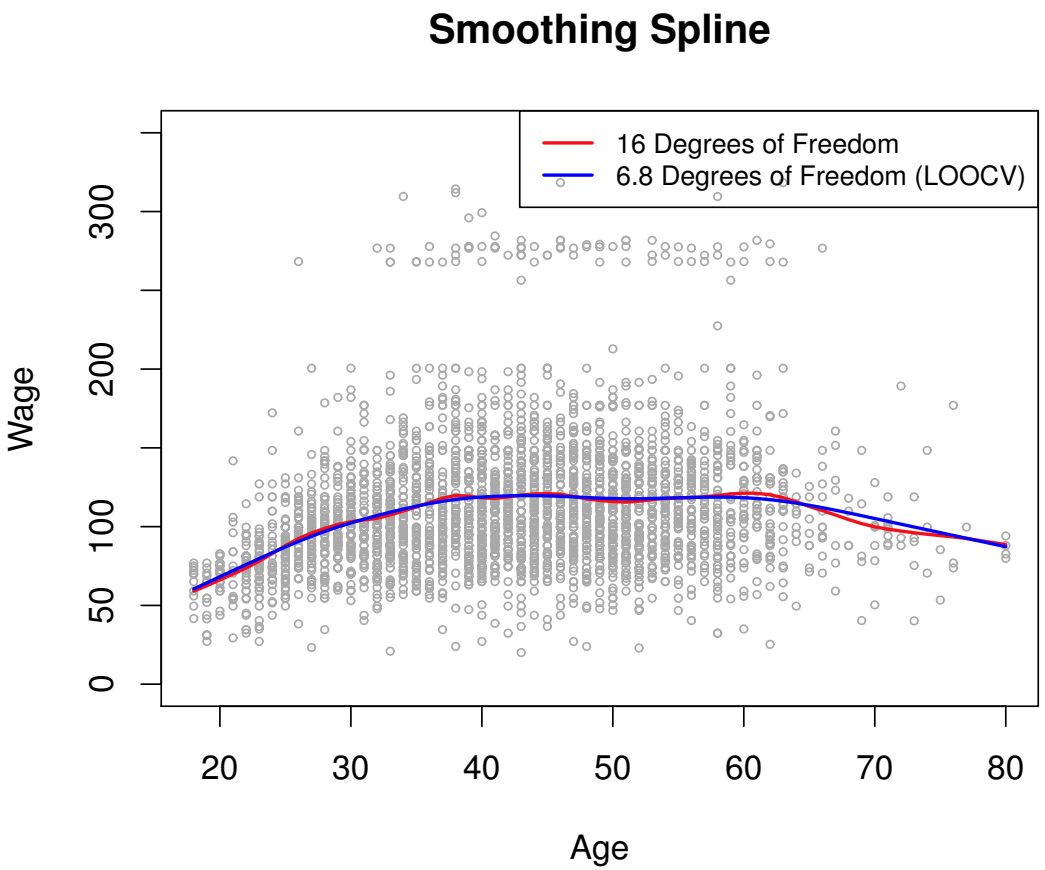
- $g'(t)$ is the first derivative, measures slope of $g(t)$ at t.
  $g''(t)$ indicates 2^{nd} derivative and corresponds to the amount by which the slope is changing.
- $g''(t)$ is a measure of "*roughness*" of $g(t)$ around t.
  it is large (in absolute value) if $g(t)$ is "wiggly" near t,
  it is zero if $g(t)$ is a straight line around t.
- $\int g''(t)^2 \, dt$ is a (specific) measure of the total change in $g'(t)$ over the entire range.

Smoothing splines (quick overview)

$$\min: \quad \mathrm{RSS}(g,\lambda) = \sum_{i=1}^{n}[y_i - g(x_i)]^2 + \lambda \int g''(t)^2\,dt$$

- While g can be any function, it may be shown that $\mathrm{RSS}(g,\lambda)$ has an explicit & unique minimizer: natural cubic spline with N knots at the unique values of x_i, $i = 1, 2, \ldots, N$.
  (more precisely, a penalized natural cubic spline with N knots)
- With a suitable λ parameter, the model is not overparameterized: effective degrees of freedom used for estimation (df_λ) decrease from N to 2 as λ increases from 0 to ∞.
- Cross-validation can be used to find λ with the lowest cross-validated RSS (LOOC is computationally very efficient here).
- For technical discussion of smoothing splines, see (ESL, ch. 5.4).

Smoothing splines (quick overview)

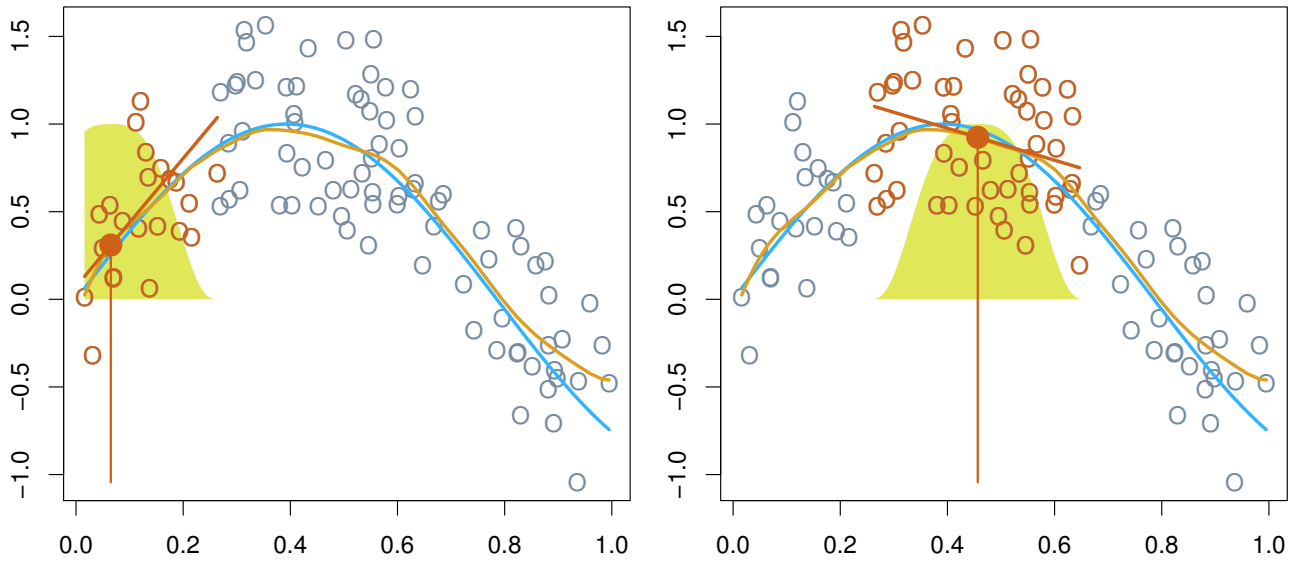


Note that df in the legend refer to (df_λ) used by the estimation.

Local regression

- Another approach for fitting flexible (non-linear) functions.
- For a given target point x_0, regression is fit using nearby observations only.

Local regression example (artificial data)



Legend: orange: local regression, blue: DGP, yellow bell: weight of observations used in regression.

Local regression algorithm

Local regression at $X = x_0$

1. Gather the fraction $s = k/n$ of training points whose x_i are closest to x_0.
2. Assign a weight $K_{i0} = K(x_i, x_0)$ to each point in this neighborhood, so that the closest point has the highest weight and vice versa. All but the nearest k neighbors get zero weights.
3. Fit *weighted least squares regression* of y_i on x_i by finding β parameters that minimize the following expression:

$$\min : \quad \sum_{i=1}^{n} K_{i0}(y_i - \beta_0 - \beta_1 x_i)^2 \,.$$

4. Fitted value at x_0 is given by $\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0 \,.$

Local regression

Local regression: choices to be made

- Weighting function K must be defined (e.g.: $\frac{1}{(x_0 - x_i)^2}, x_0 \neq x_i$).
- Functional form in Step 3 (constant, linear, quadratic, etc.).
- Span s controls flexibility of the fit (can choose using CV).

Local regression: notes

- Local regression can be generalized to models with multiple regressors: the model is global in some regressors and local in others (time). Such model adapts to the most recently gathered observations.
- Local regression can be generalized to neighborhoods of higher dimensions (say, over X_1 and X_2). However, this approach does not extend easily to higher dimensions (beyond 3 or 4).

Generalized Additive Models (GAM)

So far, polynomial and step regression, splines and local regression have been discussed as univariate regression models with $y_i \leftarrow f(x_i)$.

GAMs extend multivariate LRMs by allowing non-linear functions & smoothers in each variable, while maintaining additivity (and easy interpretation for each regressor).

Multiple-regressor LRM:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + u_i$$

can be generalized to a GAM:

$$y_i = \alpha + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_p(x_{ip}) + u_i$$

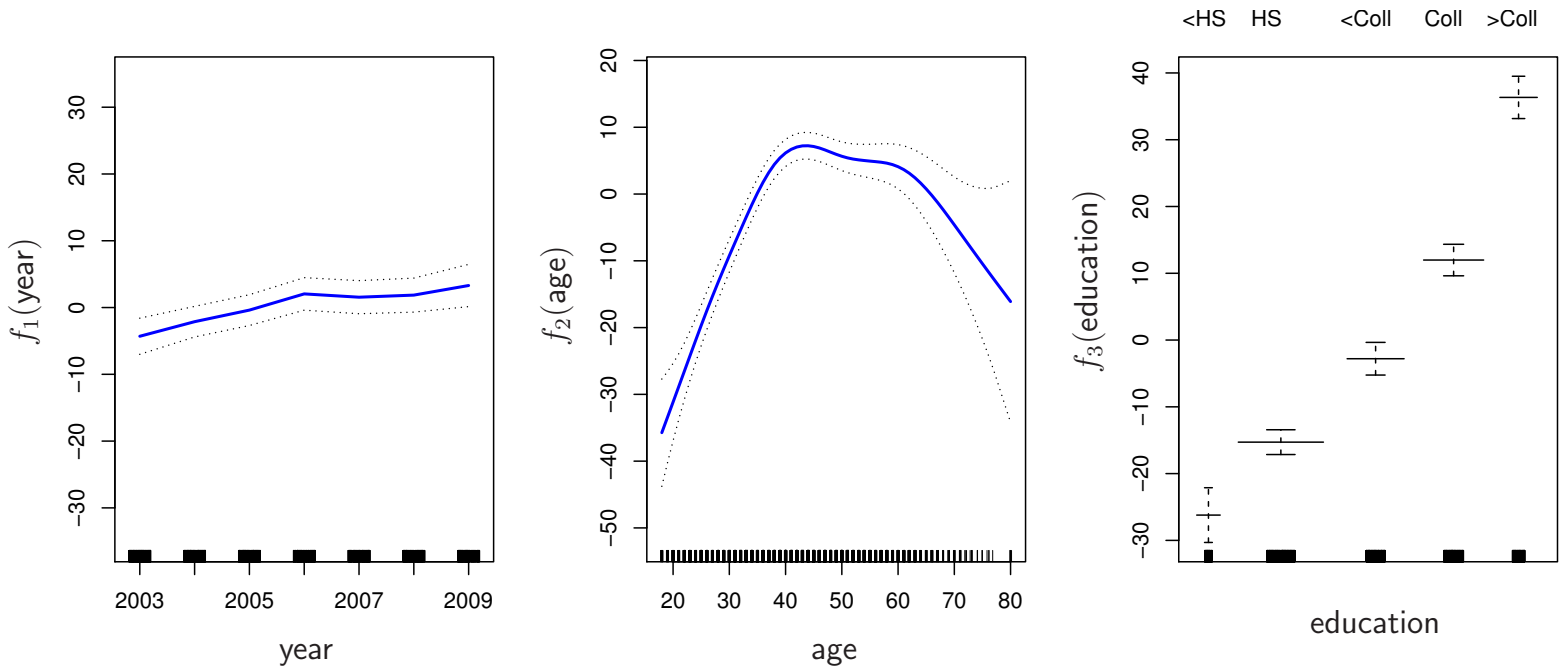
where each $f_j(x_{ij})$ represents any convenient function of x: piecewise constant, linear, polynomial, spline, etc.

GAM example

$$\texttt{wage} = \alpha + f_1(\texttt{year}) + f_2(\texttt{age}) + f_3(\texttt{education}) + u_i$$

where f_1 and f_2 are smoothing splines and f_3 is a step function

(`education` is a qualitative variable with five levels).



As we hold `age` and `education` fixed, `wage` increases slightly with `year`.
If we fix `year` and `education`, than `wage` tends to be highest for intermediate values of `age` and lowest for the very young and very old workers.

GAM estimation (backfitting)

$$y_i = \alpha + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_p(x_{ip}) + u_i$$

- f_j functions are unspecified smooth ("nonparametric") functions.
- With smoothing splines, we do not use an expansion of basis functions for estimation (i.e. constituent elements used in some matrix of regressors $\boldsymbol{X}$).
- Instead, we fit all p functions "simultaneously", using a scatterplot smoother (e.g. smoothing spline, local regression, etc.) along with a **backfitting algorithm**.
- Say, we use cubic smoothing splines for all regressors. The penalized RSS (PRSS) can be specified as:

$$\text{PRSS}(\alpha, f_1, \ldots, f_p, \boldsymbol{\lambda}) = \sum_{i=1}^{n} \left[y_i - \alpha - \sum_{j=1}^{p} f_j(x_{ij}) \right]^2 + \sum_{j=1}^{p} \lambda_j \int f_j''(t_j)^2 \, dt_j \,,$$

where $\lambda_j \geq 0$ and PRSS is used for estimaton (described next).

GAM estimation (backfitting)

$$y_i = \alpha + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_p(x_{ip}) + u_i$$

Backfitting algorithm for GAMs

1. Initialize: $\hat{\alpha} = \frac{1}{n}\sum_{i=1}^{n} y_i$ and $\hat{f}_j \equiv 0, \quad \forall i, j.$
2. Cycle: $j = 1, 2, \ldots, p, 1, 2, \ldots, p, \ldots$

   (a) $\hat{f}_j \leftarrow \mathcal{S}_j \left[\left\{ y_i - \hat{\alpha} - \sum_{k \neq j} \hat{f}_k(x_{ik}) \right\}_1^n \right]$ (backfitting step)

   (b) $\hat{f}_j \leftarrow \hat{f}_j - \frac{1}{n}\sum_{i-1}^{n} \hat{f}_j(x_{ij})$ (mean centering of estimated function)

   until functions $\hat{f}_j$ change less than a prespecified threshold.

GAM estimation (backfitting)

Backfitting algorithm for GAMs

- Each of the f_j functions is a cubic spline (with knots at each unique value of x_j and df_λ).
- Without further restrictions, solution is non-unique – constant α is non-identifiable as we can add/subtract any constant to each of the f_j functions.
- This is solved in Step 1 (i.e. by setting $\hat{\alpha} = \text{mean}(y_i)$ and $\hat{f}_j \equiv 0$). Note that this setting does not change troughout the backfitting estimation.
- Step 2 (a) applies a cubic smoothing spline $\mathcal{S}_j$ to the X_j regressor while fixing all other $\hat{f}_k$ at their current estimates when computing the $\left\{y_i - \hat{\alpha} - \sum_{k \neq j} \hat{f}_k(x_{ik})\right\}_1^n$ term.
- Step 2 (b) is technical – it adjusts for rounding errors in the algorithm (in theory, fit to a mean-zero response has mean zero).

Generalized Additive Models (GAM)

- ✓ GAM backfitting can accommodate regression/natural splines, piecewise and local regression and even interaction terms in the f_j functions.

- ✓ GAMs extend easily to logistic regression models if y_i is binary (applies to count and similar LDVs by analogy):

$$\log\left(\frac{p_i}{1+p_i}\right) = \alpha + f_1(\texttt{year}) + f_2(\texttt{age}) + f_3(\texttt{education}) + u_i$$

  where $p_i = P(\texttt{wage}_i > 250 \mid \texttt{year}_i, \texttt{age}_i, \texttt{education}_i)$.

- ✓ Individual f_j in GAMs can be very flexible – we do not have to manually try different transformations for each regressor.

- ✓ Non-linear fits can produce more accurate predictions.

Generalized Additive Models (GAM)

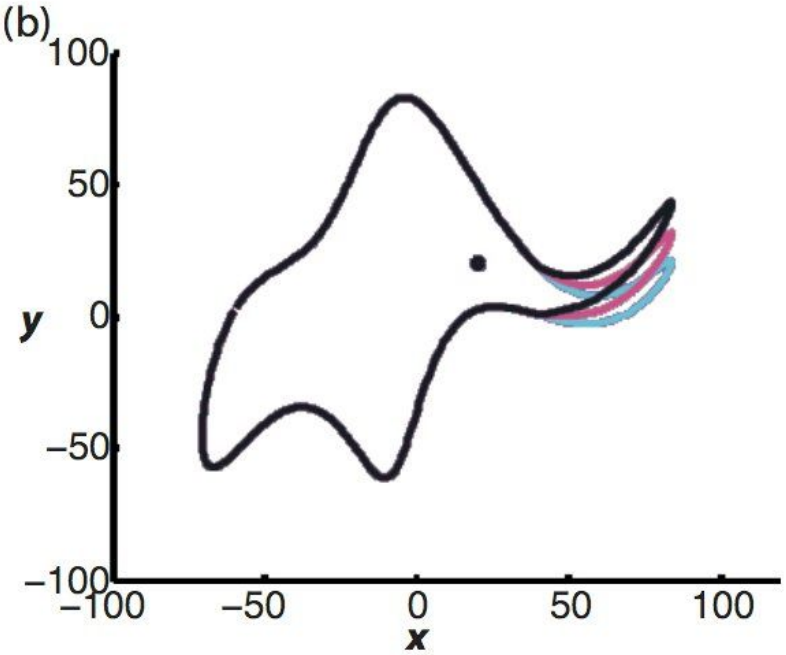
- ✓ As GAMs are additive, they provide intuitive interpretation.
- ✓ Also, GAMs provide useful representation for statistical inference.
- ✓ For each regressor, smoothness of f_j can be represented by (effective) *df* used.
- ✗ Additive nature of GAMs may lead to missing important interactions.
- ✓ However, low-dimensional interactions can be included manually; e.g. by adding some $f_{jm}(X_j, X_m)$ term into the model.

Final note

For a detailed technical discussion of the topics covered in Block 2, you may consult e.g.: The Elements of Statistical Learning

Final note

John von Neumann: “With 4 parameters, I can fit an elephant and with 5, I can make him wiggle his trunk.”



Drawing an elephant with four complex parameters