Inductive bias and PCA

COMS 4771 Fall 2019

Overview

- Inductive biases and regularization
- Model averaging and Bayesian perspectives
- Principal component analysis
- Gradient descent

Inductive bias

- What if ERM solution is not unique?
- Infinitely-many solutions to normal equations.
- Which one should we pick?
  - Possible answer: Pick shortest solution, i.e., of minimum (squared) Euclidean norm $\|\boldsymbol{w}\|_2^2$.
  - Smaller norm $\Rightarrow$ slower variations (Cauchy-Schwarz):

$$|\boldsymbol{w}^\mathsf{T}\boldsymbol{x} - \boldsymbol{w}^\mathsf{T}\boldsymbol{x}'| \leq \|\boldsymbol{w}\|_2 \cdot \|\boldsymbol{x} - \boldsymbol{x}'\|_2$$

  - But data does not give reason to choose shorter $\boldsymbol{w}$ over longer $\boldsymbol{w}$.
  - Preference for short $\boldsymbol{w}$ is an example of an *inductive bias*.
- All learning algorithms encode some form of inductive bias.

Example of minimum norm inductive bias I

- Trigonometric feature expansion with particular weighting

$$\boldsymbol{\varphi}(x) = (1, \sin(x), \cos(x), \frac{1}{2}\sin(2x), \frac{1}{2}\cos(2x), \frac{1}{3}\sin(3x), \frac{1}{3}\cos(3x), \dots)$$

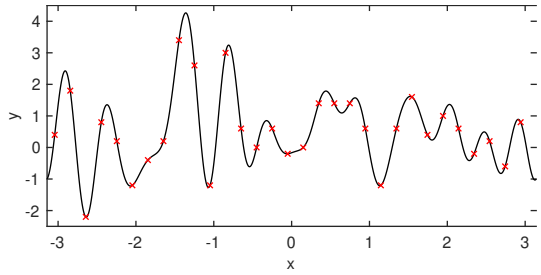


Figure 1: Arbitrary solutions to normal equations can be arbitrarily "wiggly"

Example of minimum norm inductive bias II

- Trigonometric feature expansion with particular weighting

$$\boldsymbol{\varphi}(x) = (1, \sin(x), \cos(x), \frac{1}{2}\sin(2x), \frac{1}{2}\cos(2x), \frac{1}{3}\sin(3x), \frac{1}{3}\cos(3x), \dots)$$

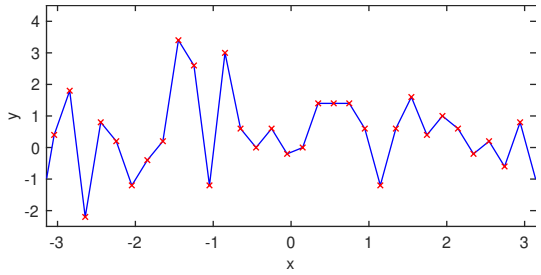


Figure 2: Least norm solution is a very particular interpolation

Representation of minimum norm solution

- **Claim**: The minimum (Euclidean) norm solution to normal equations lives in span of the $\boldsymbol{x}_i$'s (i.e., in $\mathrm{range}(\boldsymbol{A}^T)$).
  - I.e., can write

$$\boldsymbol{w} = \boldsymbol{A}^T\boldsymbol{\alpha} = \sum_{i=1}^{n} \alpha_i \boldsymbol{x}_i$$

  for some $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_n) \in \mathbb{R}^n$.

- In fact, the solution in $\mathrm{range}(\boldsymbol{A}^T)$ is unique!

Regularized ERM

- Combine two concerns: making both $\widehat{\mathcal{R}}(\boldsymbol{w})$ and $\|\boldsymbol{w}\|_2^2$ small
  - Pick $\lambda \geq 0$, and minimize $\widehat{\mathcal{R}}(\boldsymbol{w}) + \lambda\|\boldsymbol{w}\|_2^2$
- If $\lambda > 0$, solution is always unique (even if $n < d$).
  - Called *ridge regression*.
  - $\lambda = 0$ is ERM/OLS.
  - λ controls how much to pay attention to *regularizer* $\|\boldsymbol{w}\|_2^2$ relative to *data fitting term* $\widehat{\mathcal{R}}(\boldsymbol{w})$
  - λ is hyperparameter to tune (e.g., using cross-validation)

Data augmentation I

- Let $\widetilde{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A} \\ \sqrt{\lambda}\boldsymbol{I} \end{bmatrix}$ and $\tilde{\boldsymbol{b}} = \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix} \in \mathbb{R}^{n+d}$
- Then $\|\widetilde{\boldsymbol{A}}\boldsymbol{w} - \tilde{\boldsymbol{b}}\|_2^2 = \widehat{\mathcal{R}}(\boldsymbol{w}) + \lambda\|\boldsymbol{w}\|_2^2$ (ridge regression objective)
- Interpretation:
  - d "fake" data points, ensures augmented data matrix $\widetilde{\boldsymbol{A}}$ has rank d
  - All corresponding labels are zero.
- So ridge regression solution is $\hat{\boldsymbol{w}} =$

Data augmentation II

- Domain-specific data augmentation: e.g., image transformations

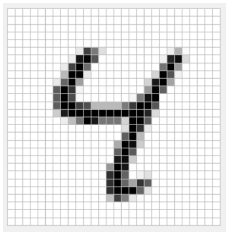


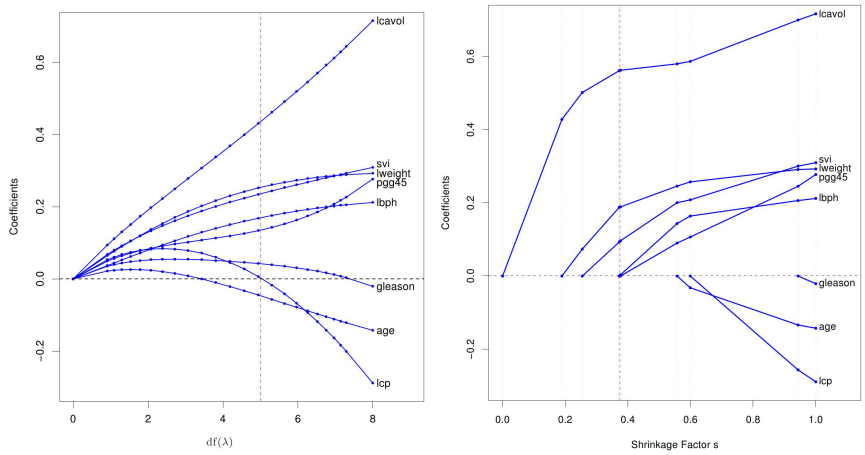
Figure 3: Pixels of OCR image

Lasso

- *Lasso*: minimize $\widehat{\mathcal{R}}(\boldsymbol{w}) + \lambda \|\boldsymbol{w}\|_1$
  - Here, $\|\boldsymbol{v}\|_1 = \sum_{i=1}^n |v_i|$, sum of absolute values of vector entries
  - Prefers short $\boldsymbol{w}$, where length is measured using different norm
  - Tends to produce $\boldsymbol{w}$ that are *sparse* (i.e., have few non-zero entries), or at least are well-approximated by sparse vectors.
  - A different inductive bias

Lasso vs ridge regression

- Example: coefficient profile of Lasso vs ridge
- $\boldsymbol{X}$ = clinical measurements, Y = level of prostate cancer antigen
- Horizontal axis: varying λ (large λ to left, small λ to right).
- Vertical axis: coefficient value in ridge and Lasso solutions, for eight different features



Inductive bias from minimum ℓ_1 norm

- **Theorem**: Pick any $\boldsymbol{w} \in \mathbb{R}^d$ and any $\varepsilon \in (0, 1)$. Form $\tilde{\boldsymbol{w}} \in \mathbb{R}^d$ by including the $\lceil 1/\varepsilon^2 \rceil$ largest (by magnitude) coefficients of $\boldsymbol{w}$, and setting remaining entries to zero. Then

$$\|\tilde{\boldsymbol{w}} - \boldsymbol{w}\|_2 \leq \varepsilon \|\boldsymbol{w}\|_1.$$

  - If $\|\boldsymbol{w}\|_1$ is small (compared to $\|\boldsymbol{w}\|_2$), then theorem says $\boldsymbol{w}$ is well-approximated by sparse vector.

Sparsity

- Lasso also tries to make coefficients small. What if we only care about sparsity?
- *Subset selection*: minimize empirical risk among all k-sparse solutions
- Greedy algorithms: repeatedly choose new variables to "include" in support of w until k variables are included.
  - *Forward stepwise regression* / *orthogonal matching pursuit*: Each time you "include" a new variable, re-fit all coefficients for included variables.
  - Often works as well as Lasso
- Why do we care about sparsity?

Detour: Model averaging

- Suppose we have M real-valued predictors, $\hat{f}_1, \ldots, \hat{f}_M$
  - E.g., nearest neighbor regression, regression trees, linear models with different feature expansions, ...
- How to take advantage of all of them?
- *Model selection*: pick the best one, e.g., using hold-out method or K-fold cross-validation
- *Model averaging*: form "ensemble" predictor $\hat{f}_{\text{avg}}$, where for any x,

$$\hat{f}_{\text{avg}}(x) := \frac{1}{M} \sum_{i=1}^{M} \hat{f}_i(x).$$

Risk of model averaging

- **Theorem**: Risk of $\hat{f}_{\text{avg}}$:

$$\mathcal{R}(\hat{f}_{\text{avg}}) = \frac{1}{M} \sum_{i=1}^{M} \mathcal{R}(\hat{f}_i) - \frac{1}{M} \sum_{i=1}^{M} \mathbb{E}\left[(\hat{f}_{\text{avg}}(X) - \hat{f}_i(X))^2 \right].$$

- Better than model selection when:
  - all $\hat{f}_i$ have similar risks, and
  - all $\hat{f}_i$ predict very differently from each other

Stacking and features

- In model averaging, "weights" of $1/M$ for all $\hat{f}_i$ seems arbitrary
- Can "learn" weights using linear regression!
  - Use feature expansion $\varphi(x) = (\hat{f}_1(x), \ldots, \hat{f}_M(x))$
  - Called *stacking*
  - Use additional data (independent of $\hat{f}_1, \ldots, \hat{f}_M$)
- Upshot: Any function (even learned functions) can be a feature
- Conversely: Behind every feature is a deliberate modeling choice

Detour: Bayesian statistics

- *Bayesian inference*: probabilistic approach to updating beliefs
  - Posit a (parametric) statistical model for data (*likelihood*)
  - Start with some beliefs about the parameters of model (*prior*)
  - Update beliefs after seeing data (*posterior*)

$$\underbrace{\Pr(\boldsymbol{w} \mid \text{data})}_{\text{posterior}(\boldsymbol{w})} \propto \underbrace{\Pr(\boldsymbol{w})}_{\text{prior}(\boldsymbol{w})} \cdot \underbrace{\Pr(\text{data} \mid \boldsymbol{w})}_{\text{likelihood}(\boldsymbol{w})}$$

  - (Finding proportionality constant is often the computationally challenging part of belief updating.)
  - Basis for reasoning in humans (maybe?), robots, etc.

Beyond Bayesian inference

- Can use Bayesian inference framework for designing estimation/learning algorithms (even if you aren't a Bayesian!)
  - E.g., instead of computing entire posterior distribution, find the $\boldsymbol{w}$ with highest posterior probability
  - Called *maximum a posteriori (MAP)* estimator
  - Just find $\boldsymbol{w}$ to maximize

$$\text{prior}(\boldsymbol{w}) \times \text{likelihood}(\boldsymbol{w}).$$

  - (Avoids issue with finding proportionality constant.)

Bayesian approach to linear regression

- In linear regression model, express prior belief about $\boldsymbol{w} = (w_1, \dots, w_d)$ using a probability distribution with density function π
  - Simple choice: $\text{prior}(w_1, \dots, w_d) = \prod_{j=1}^{d} \sqrt{\frac{\tau}{2\pi}} \exp(-\tau w_j^2/2)$
  - I.e., treat $w_1, \dots, w_d$ as independent $\mathrm{N}(0, 1/\tau)$ random variables
- Likelihood model: $(\boldsymbol{X}_1, Y_1), \dots, (\boldsymbol{X}_n, Y_n)$ are conditionally independent given $\boldsymbol{w}$, and $Y_i \mid (\boldsymbol{X}_i, \boldsymbol{w}) \sim \mathrm{N}(\boldsymbol{X}_i^\mathsf{T} \boldsymbol{w}, 1)$.
- What is the MAP?

MAP for Bayesian linear regression

- Find $\boldsymbol{w}$ to maximize

$$\underbrace{\prod_{j=1}^{d} \sqrt{\frac{\tau}{2\pi}} \exp(-\tau w_j^2/2)}_{\text{prior}(\boldsymbol{w})} \cdot \underbrace{\prod_{i=1}^{n} p(x_i) \cdot \frac{1}{\sqrt{2\pi}} \exp(-(y_i - \boldsymbol{x}_i^\mathsf{T} \boldsymbol{w})^2/2)}_{\text{likelihood}(\boldsymbol{w})}.$$

  (Here, p is marginal density of $\boldsymbol{X}$; unimportant.)
- Take logarithm and omit terms not involving $\boldsymbol{w}$:

$$-\frac{\tau}{2} \sum_{i=1}^{d} w_j^2 - \frac{1}{2} \sum_{i=1}^{n} (y_i - \boldsymbol{x}_i^\mathsf{T} \boldsymbol{w})^2.$$

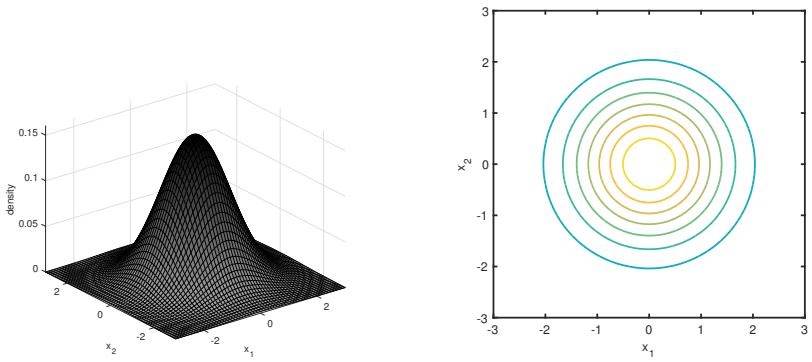
- For $\tau = n\lambda$, same as minimizing

$$\frac{1}{n} \sum_{i=1}^{n} (\boldsymbol{x}_i^\mathsf{T} \boldsymbol{w} - y_i)^2 + \lambda \|\boldsymbol{w}\|_2^2,$$

  which is the ridge regression objective!
- What about different Gaussian prior?

Multivariate Gaussians I: Isotropic Gaussians

- Start with $\boldsymbol{X} = (X_1, \ldots, X_d) \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{I})$, i.e., $X_1, \ldots, X_d$ are iid $\mathrm{N}(0, 1)$ random variables.
  - Probability density function is product of (univariate) Gaussian densities
  - $\mathbb{E}(X_i) = 0$
  - $\mathrm{var}(X_i) = \mathrm{cov}(X_i, X_i) = 1$, $\mathrm{cov}(X_i, X_j) = 0$ for $i \neq j$
  - Arrange in mean vector $\mathbb{E}(\boldsymbol{X}) = 0$, covariance matrix $\mathrm{cov}(\boldsymbol{X}) = \boldsymbol{I}$



Affine transformations of random vectors

- Start with any random vector $\boldsymbol{X}$, then apply linear transformation, followed by translation
- $\boldsymbol{Y} := \boldsymbol{M}\boldsymbol{X} + \boldsymbol{\mu}$, for $\boldsymbol{M} \in \mathbb{R}^{k \times d}$ and $\boldsymbol{\mu} \in \mathbb{R}^k$
- $\mathbb{E}(\boldsymbol{Y}) =$
- $\mathrm{cov}(\boldsymbol{Y}) =$
- Let $\boldsymbol{u} \in \mathbb{R}^d$ be a unit vector ($\|\boldsymbol{u}\|_2 = 1$), and $\boldsymbol{Y} := \boldsymbol{u}^\mathsf{T}\boldsymbol{X}$ (projection of $\boldsymbol{X}$ along direction $\boldsymbol{u}$).
- $\mathbb{E}(\boldsymbol{Y}) =$
- $\mathrm{var}(\boldsymbol{Y}) =$

Multivariate Gaussians II: General Gaussians

- If $\boldsymbol{X} \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{I})$ and $\boldsymbol{Y} = \boldsymbol{M}\boldsymbol{X} + \boldsymbol{\mu}$, we have $\mathbb{E}(\boldsymbol{Y}) = \boldsymbol{\mu}$ and $\mathrm{cov}(\boldsymbol{Y}) = \boldsymbol{M}\boldsymbol{M}^\mathsf{T}$
  - Assume $\boldsymbol{M} \in \mathbb{R}^{d \times d}$ is invertible (else we get a degenerate Gaussian distribution).
  - We say $\boldsymbol{Y} \sim \mathrm{N}(\boldsymbol{\mu}, \boldsymbol{M}\boldsymbol{M}^\mathsf{T})$
  - Density function given by

$$\frac{1}{(2\pi)^{d/2}|\boldsymbol{M}\boldsymbol{M}^\mathsf{T}|^{1/2}} \exp\left(-\frac{1}{2}\|\boldsymbol{M}^{-1}(\boldsymbol{y} - \boldsymbol{\mu})\|_2^2\right).$$

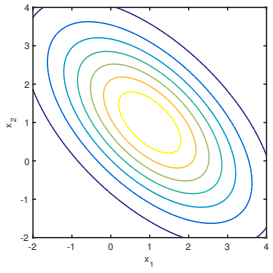


Figure 4: Contour lines of a bivariate Gaussian density

MAP with general Gaussian priors

- Prior: multivariate Gaussian, written $\boldsymbol{w} \sim \mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
  - Probability density is $\mathrm{prior}(\boldsymbol{w}) \propto \exp\left(-\frac{1}{2}(\boldsymbol{w} - \boldsymbol{\mu})^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\boldsymbol{w} - \boldsymbol{\mu})\right)$
- Find $\boldsymbol{w}$ to maximize

$$\underbrace{\exp\left(-\frac{1}{2}(\boldsymbol{w} - \boldsymbol{\mu})^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\boldsymbol{w} - \boldsymbol{\mu})\right)}_{\mathrm{prior}(\boldsymbol{w})} \cdot \underbrace{\prod_{i=1}^n p(\boldsymbol{x}_i) \cdot \frac{1}{\sqrt{2\pi}} \exp(-(y_i - \boldsymbol{x}_i^\mathsf{T}\boldsymbol{w})^2/2)}_{\mathrm{likelihood}(\boldsymbol{w})}.$$

- Take logarithm and omit terms not involving $\boldsymbol{w}$:

$$-\frac{1}{2}(\boldsymbol{w} - \boldsymbol{\mu})^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\boldsymbol{w} - \boldsymbol{\mu}) - \frac{1}{2}\sum_{i=1}^n (y_i - \boldsymbol{x}_i^\mathsf{T}\boldsymbol{w})^2.$$

- For $\boldsymbol{C} := \boldsymbol{\Sigma}^{-1}/n$, same as minimizing

$$\frac{1}{n}\sum_{i=1}^n (\boldsymbol{x}_i^\mathsf{T}\boldsymbol{w} - y_i)^2 + (\boldsymbol{w} - \boldsymbol{\mu})^\mathsf{T}\boldsymbol{C}(\boldsymbol{w} - \boldsymbol{\mu}),$$

a different regularizer!

Eigendecomposition

- Every symmetric matrix $\boldsymbol{M} \in \mathbb{R}^{d \times d}$ has d real *eigenvalues*, which we arrange as $\lambda_1 \geq \cdots \geq \lambda_d$
  - Can choose corresponding *eigenvectors* $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d \in \mathbb{R}^d$ to be orthonormal
  - This means $\boldsymbol{M}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$ for each $i = 1, \ldots, d$, and $\boldsymbol{v}_i^\mathsf{T} \boldsymbol{v}_j = \mathbb{1}_{\{i=j\}}$
- Often arrange $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d$ in an *orthogonal matrix* $\boldsymbol{V} := [\boldsymbol{v}_1 | \cdots | \boldsymbol{v}_d]$
  - $\boldsymbol{V}^\mathsf{T}\boldsymbol{V} = \boldsymbol{I}$ and $\boldsymbol{V}\boldsymbol{V}^\mathsf{T} = \sum_{i=1}^d \boldsymbol{v}_i \boldsymbol{v}_i^\mathsf{T} = \boldsymbol{I}$
- *Eigendecomposition* (*spectral decomposition*):

- *Diagonalization*:

Covariance matrix

- $\boldsymbol{A} \in \mathbb{R}^{n \times d}$ is data matrix
- $\boldsymbol{\Sigma} := \boldsymbol{A}^\mathsf{T}\boldsymbol{A} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i \boldsymbol{x}_i^\mathsf{T}$ is *(empirical) second-moment matrix*
  - Assuming $\frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i = \boldsymbol{0}$, this is the *(empirical) covariance matrix*
- For any unit vector $\boldsymbol{u} \in \mathbb{R}^d$,

$$\boldsymbol{u}^\mathsf{T}\boldsymbol{\Sigma}\boldsymbol{u} = \frac{1}{n}\sum_{i=1}^n (\boldsymbol{u}^\mathsf{T}\boldsymbol{x}_i)^2$$

  is variance of data along direction $\boldsymbol{u}$
- Note: some pixels in OCR data have very little (or zero!) variation

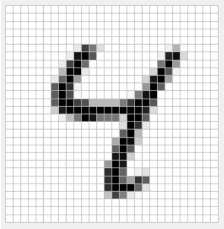


Figure 5: Pixels of OCR image

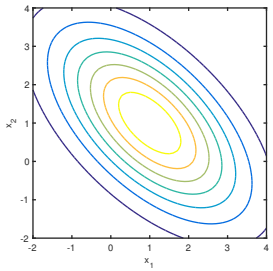
Top eigenvector

- $\boldsymbol{\Sigma}$ is symmetric, so can write eigendecomposition

$$\boldsymbol{\Sigma} = \sum_{i=1}^n \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i^\mathsf{T}$$

- In which direction is variance maximized?
- Answer: $\boldsymbol{v}_1$, corresponding to largest eigenvalue λ_1
  - Called the *top eigenvector*
  - This follows from the following characterization of $\boldsymbol{v}_1$:

$$\boldsymbol{v}_1^\mathsf{T}\boldsymbol{\Sigma}\boldsymbol{v}_1 = \max_{\boldsymbol{u} \in \mathbb{R}^d : \|\boldsymbol{u}\|_2 = 1} \boldsymbol{u}^\mathsf{T}\boldsymbol{\Sigma}\boldsymbol{u} = \lambda_1.$$



Top k eigenvectors

- What about among directions orthogonal to $\boldsymbol{v}_1$?
  - Answer: $\boldsymbol{v}_2$, corresponding to second largest eigenvalue λ_2
- Etc.
- For any k, $\boldsymbol{V}_k := [\boldsymbol{v}_1 | \cdots | \boldsymbol{v}_k]$ satisfies

$$\sum_{i=1}^k \boldsymbol{v}_i^\mathsf{T}\boldsymbol{\Sigma}\boldsymbol{v}_i = \operatorname{tr}(\boldsymbol{V}_k^\mathsf{T}\boldsymbol{\Sigma}\boldsymbol{V}_k) = \max_{\boldsymbol{U} \in \mathbb{R}^{d \times k} : \boldsymbol{U}^\mathsf{T}\boldsymbol{U} = \boldsymbol{I}} \operatorname{tr}(\boldsymbol{U}^\mathsf{T}\boldsymbol{\Sigma}\boldsymbol{U}) = \sum_{i=1}^k \lambda_i$$

(the *top k eigenvectors*)

Principal component analysis

- k-dimensional *principal components analysis (PCA)* mapping:

$$\varphi(\boldsymbol{x}) = (\boldsymbol{x}^\top \boldsymbol{v}_1, \ldots, \boldsymbol{x}^\top \boldsymbol{v}_k) = \boldsymbol{V}_k^\top \boldsymbol{x} \in \mathbb{R}^k$$

  where $\boldsymbol{V}_k = [\boldsymbol{v}_1 | \cdots | \boldsymbol{v}_k] \in \mathbb{R}^{d \times k}$
- (Only really makes sense when $\lambda_k > 0$.)
- This is a form of *dimensionality reduction* when $k < d$.

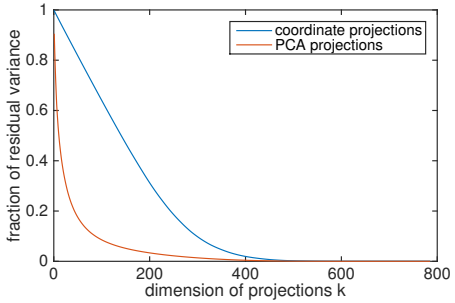


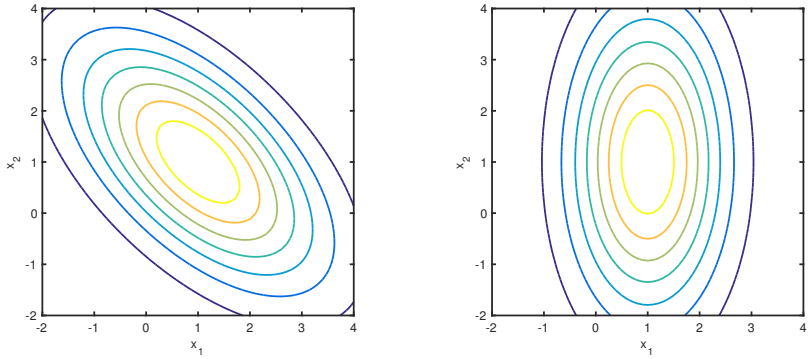
Figure 7: Fraction of residual variance with PCA and coordinate projections

Covariance of data upon PCA mapping

- Covariance of data upon PCA mapping:

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^n \varphi(\boldsymbol{x}_i)\varphi(\boldsymbol{x}_i)^\top &= \frac{1}{n}\sum_{i=1}^n \boldsymbol{V}_k^\top \boldsymbol{x}_i \boldsymbol{x}_i^\top \boldsymbol{V}_k \\ &= \boldsymbol{V}_k^\top \boldsymbol{\Sigma} \boldsymbol{V}_k \\ &= \boldsymbol{\Lambda}_k\end{aligned}$$

  where $\boldsymbol{\Lambda}_k$ is diagonal matrix with $\lambda_1, \ldots, \lambda_k$ along diagonal.
  - In particular, coordinates in $\varphi(\boldsymbol{x})$-representation are uncorrelated.



PCA and linear regression

- Use k-dimensional PCA mapping $\varphi(\boldsymbol{x}) = \boldsymbol{V}_k^\top \boldsymbol{x}$ with OLS
- (Assume rank of $\boldsymbol{A}$ is at least k, so $\boldsymbol{A}^\top \boldsymbol{A}$ has $\lambda_k > 0$)
- Data matrix is

$$\frac{1}{\sqrt{n}} \begin{bmatrix} \leftarrow & \varphi(\boldsymbol{x}_1)^\top & \rightarrow \\ & \vdots & \\ \leftarrow & \varphi(\boldsymbol{x}_n)^\top & \rightarrow \end{bmatrix} = \frac{1}{\sqrt{n}} \begin{bmatrix} \leftarrow & \boldsymbol{x}_1^\top \boldsymbol{V}_k & \rightarrow \\ & \vdots & \\ \leftarrow & \boldsymbol{x}_n^\top \boldsymbol{V}_k & \rightarrow \end{bmatrix} = \boldsymbol{A}\boldsymbol{V}_k \in \mathbb{R}^{n \times k}$$

- Therefore, OLS solution is

$$\begin{aligned}\hat{\boldsymbol{\beta}} &= (\boldsymbol{V}_k^\top \boldsymbol{A}^\top \boldsymbol{A} \boldsymbol{V}_k)^{-1} (\boldsymbol{A}\boldsymbol{V}_k)^\top \boldsymbol{b} \\ &= \boldsymbol{\Lambda}_k^{-1} \boldsymbol{V}_k^\top \boldsymbol{A}^\top \boldsymbol{b}\end{aligned}$$

(Note: here $\hat{\boldsymbol{\beta}} \in \mathbb{R}^k$.)

Principal component regression

- Use $\hat{\boldsymbol{\beta}} = \boldsymbol{\Lambda}_k^{-1} \boldsymbol{V}_k^\top \boldsymbol{A}^\top \boldsymbol{b}$ to predict on new $\boldsymbol{x} \in \mathbb{R}^d$:

$$\begin{aligned}\varphi(\boldsymbol{x})^\top \hat{\boldsymbol{\beta}} &= (\boldsymbol{V}_k^\top \boldsymbol{x})^\top \boldsymbol{\Lambda}_k^{-1} \boldsymbol{V}_k^\top \boldsymbol{A}^\top \boldsymbol{b} \\ &= \boldsymbol{x}^\top (\boldsymbol{V}_k \boldsymbol{\Lambda}_k^{-1} \boldsymbol{V}_k^\top)(\boldsymbol{A}^\top \boldsymbol{b})\end{aligned}$$

- So "effective" weight vector (that acts directly on $\boldsymbol{x}$ rather than $\varphi(\boldsymbol{x})$) is given by

$$\hat{\boldsymbol{w}} := (\boldsymbol{V}_k \boldsymbol{\Lambda}_k^{-1} \boldsymbol{V}_k^\top)(\boldsymbol{A}^\top \boldsymbol{b}) = \left(\sum_{i=1}^k \frac{1}{\lambda_i} \boldsymbol{v}_i \boldsymbol{v}_i^\top \right) (\boldsymbol{A}^\top \boldsymbol{b}).$$

- This is called *principal component regression (PCR)* (here, k is hyperparameter)
- Alternative hyper-parameterization: $\lambda > 0$; same as before but using the largest k such that $\lambda_k \geq \lambda$.

Spectral regularization

- PCR and ridge regression are examples of *spectral regularization*.
- For a function $g\colon \mathbb{R} \to \mathbb{R}$, write $g(\boldsymbol{M})$ to mean

$$g(\boldsymbol{M}) = \sum_{i=1}^{d} g(\lambda_i)\boldsymbol{v}_i\boldsymbol{v}_i^\mathsf{T}$$

  where $\boldsymbol{M}$ has eigendecomposition $\boldsymbol{M} = \sum_{i=1}^{d} \lambda_i \boldsymbol{v}_i\boldsymbol{v}_i^\mathsf{T}$.

- **Claim**: Can write each of PCR and ridge regression as

$$\hat{\boldsymbol{w}} = g(\boldsymbol{A}^\mathsf{T}\boldsymbol{A})\boldsymbol{A}^\mathsf{T}\boldsymbol{b}$$

  for appropriate function g (depending on λ).

Comparing ridge regression and PCR

Ridge: $g(z) = \dfrac{1}{z+\lambda}$; PCR: $g(z) = \mathbb{1}_{\{z \geq \lambda\}} \cdot \dfrac{1}{z}$

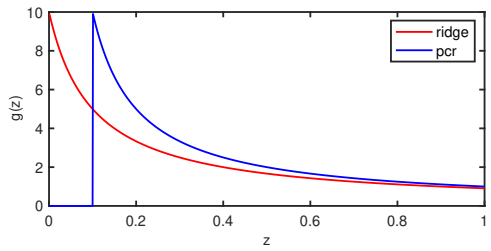


Figure 8: Spectral regularization function g for ridge and PCR ($\lambda = 0.1$)

- Interpretation:
  - PCR only uses directions with sufficient variability; ignores the rest
  - Ridge artificially inflates the variance in all directions

Optimization for linear regression

- Back to considering ordinary least squares.
- Gaussian elimination to solve normal equations can be slow when d is large (time is $O(nd^2)$).
- Alternative: find approximate solution using *gradient descent*

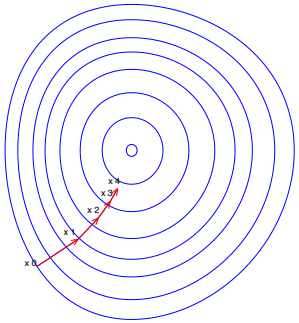


Figure 9: Gradient descent

Gradient descent for linear regression

- Algorithm: start with some $\boldsymbol{w}^{(0)} \in \mathbb{R}^d$ and $\eta > 0$.
  - For $t = 1, 2, \ldots$:

$$\begin{aligned}\boldsymbol{w}^{(t)} &:= \boldsymbol{w}^{(t-1)} - 2\eta\boldsymbol{A}^\mathsf{T}(\boldsymbol{A}\boldsymbol{w}^{(t-1)} - \boldsymbol{b}) \\ &= \boldsymbol{w}^{(t-1)} - 2\eta \cdot \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i^\mathsf{T}\boldsymbol{w}^{(t-1)} - y_i)\boldsymbol{x}_i\end{aligned}$$

- Time to multiply matrix by vector is linear in matrix size.
- So each iteration takes time $O(nd)$.
- η is called *step size* (somewhat of a misnomer)

Motivation for gradient descent

- Why move in direction of (negative) gradient?
- Affine approximation of $\widehat{\mathcal{R}}(\boldsymbol{w} + \boldsymbol{\delta})$ around $\boldsymbol{w}$:

- Use $\boldsymbol{\delta} := -\eta \nabla \widehat{\mathcal{R}}(\boldsymbol{w})$ for some $\eta > 0$:

Interpretation of gradient descent for linear regression

- Interpretation (specific to least squares objective):

$$\nabla (\boldsymbol{x}_i^\top \boldsymbol{w} - y_i)^2 = 2(\boldsymbol{x}_i^\top \boldsymbol{w} - y_i)\boldsymbol{x}_i.$$

  - If $\boldsymbol{x}_i^\top \boldsymbol{w} > y_i$, subtract a little bit of $\boldsymbol{x}_i$ from $\boldsymbol{w}$
  - If $\boldsymbol{x}_i^\top \boldsymbol{w} < y_i$, add a little bit of $\boldsymbol{x}_i$ from $\boldsymbol{w}$
  - If $\boldsymbol{x}_i^\top \boldsymbol{w} = y_i$, i-th term has no contribution

Behavior of gradient descent for linear regression

- **Theorem**: Let $\hat{\boldsymbol{w}}$ be the minimum Euclidean norm solution to normal equations. Assume $\boldsymbol{w}^{(0)} = \boldsymbol{0}$. Write eigendecomposition $\boldsymbol{A}^\top \boldsymbol{A} = \sum_{i=1}^r \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i^\top$ with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r > 0$. Then $\boldsymbol{w}^{(t)} \in \mathrm{range}(\boldsymbol{A}^\top)$ and

$$\boldsymbol{v}_i^\top \boldsymbol{w}^{(t)} = \left(2\eta\lambda_i \sum_{k=0}^{t-1} (1 - 2\eta\lambda_i)^k \right) \boldsymbol{v}_i^\top \hat{\boldsymbol{w}}, \quad i = 1, \ldots, r.$$

- Implications:
  - If we choose η such that $2\eta\lambda_i < 1$, then

$$2\eta\lambda_i \sum_{k=0}^{t-1} (1 - 2\eta\lambda_i)^k = 1 - (1 - 2\eta\lambda_i)^t,$$

    which converges to 1 as $t \to \infty$.
  - So, when $2\eta\lambda_1 < 1$, we have $\boldsymbol{w}^{(t)} \to \hat{\boldsymbol{w}}$ as $t \to \infty$.
  - *Rate of convergence* is geometric, i.e., "exponentially fast convergence".

Inductive bias of gradient descent

- Gradient descent for linear regression has an inductive bias—converges to the minimum norm solution.
- Also a form of spectral regularization, with function

$$g(z) = \mathbb{1}_{\{z>0\}} \cdot \frac{1 - (1 - 2\eta z)^t}{z}.$$

- Minimum norm solution uses

$$g(z) = \mathbb{1}_{\{z>0\}} \cdot \frac{1}{z}.$$