Pattern Information Processing: Sparse Methods

Masashi Sugiyama
(Department of Computer Science)

Contact: W8E-505
sugi@cs.titech.ac.jp
http://sugiyama-www.cs.titech.ac.jp/~sugi/

Sparseness and Continuous Model Choice

- Two approaches to avoiding over-fitting:

	Sparseness	Model parameter
Subspace LS	Yes	Discrete
Quadratically constrained LS	No	Continuous

- We want to have sparseness and continuous model choice at the same time.

Today's Plan

- Sparse learning method
- How to deal with absolute values in optimization
- Standard form of quadratic programs

Non-Linear Learning for Linear / Kernel Models

- Linear / kernel models

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{b} \alpha_i \varphi_i(\boldsymbol{x})$$

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i)$$

- Non-linear learning

$$\hat{\boldsymbol{\alpha}} = \boldsymbol{L}(\boldsymbol{y})$$

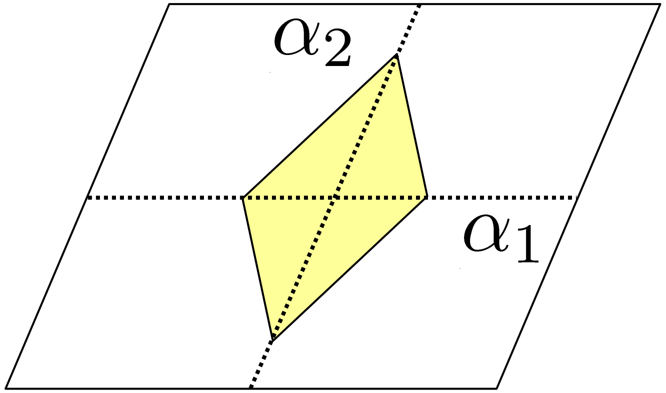
$\boldsymbol{L}(\cdot)$:Non-linear function

l1-Constrained LS

- Restrict the search space within a (rotated) hyper-cube.

$$\hat{\boldsymbol{\alpha}}_{\ell_1 CLS} = \mathop{\mathrm{argmin}}_{\boldsymbol{\alpha} \in \mathbb{R}^b} J_{LS}(\boldsymbol{\alpha})$$

$$\text{subject to } \|\boldsymbol{\alpha}\|_1 \leq C$$



$\ell_1 - \mathrm{norm}$

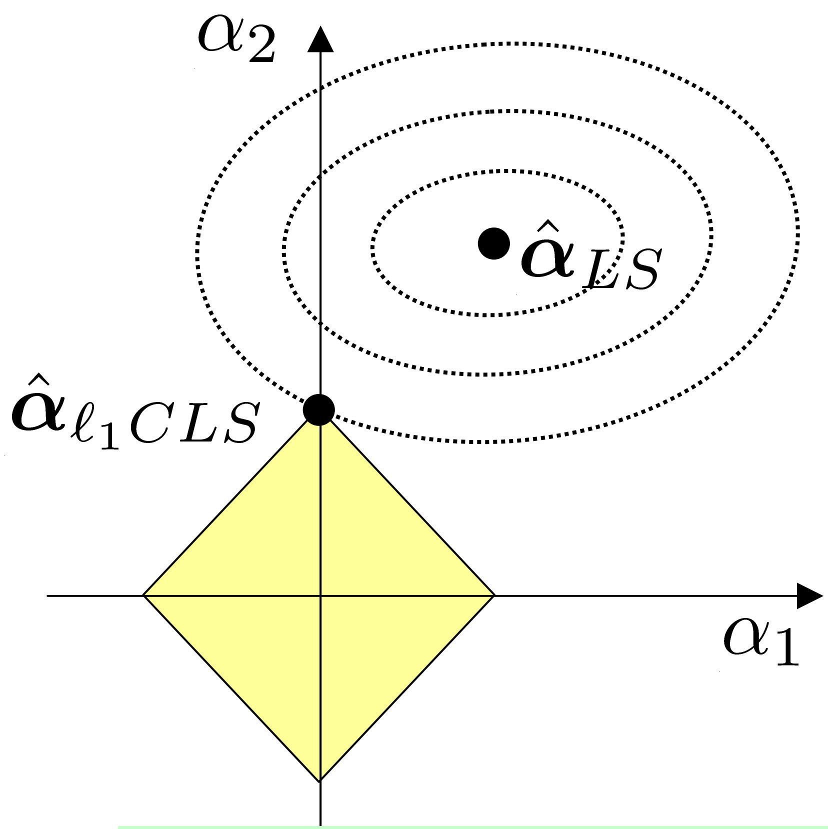
$$\|\boldsymbol{\alpha}\|_1 = \sum_{i=1}^{b} |\alpha_i|$$

See:
Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society, Series B, 58(1), 267-288,1996.

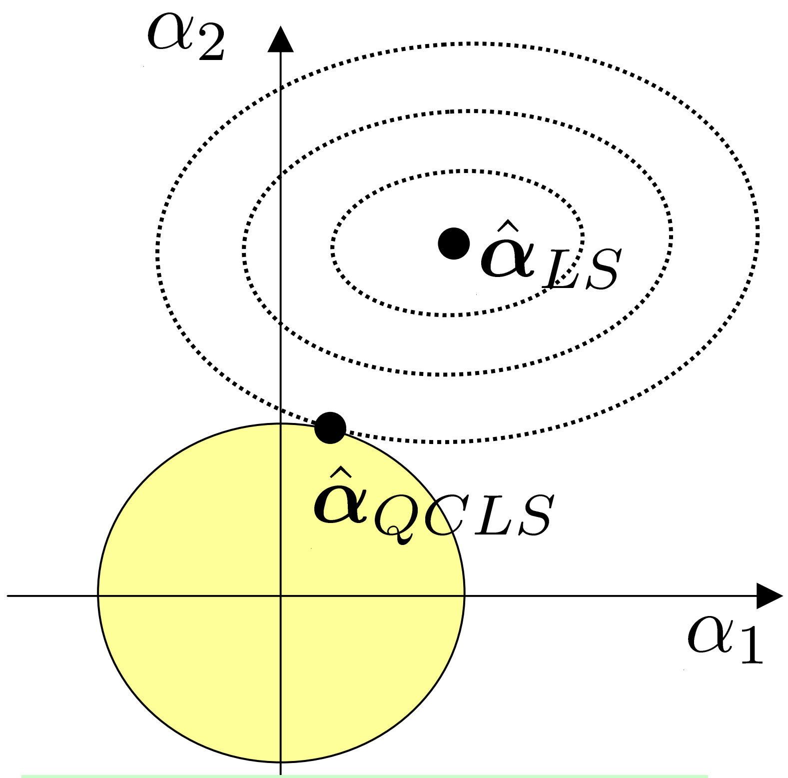
Chen, Donoho & Saunders, Atomic decomposition by basis pursuit, SIAM Journal on Scientific Computing, 20(1), 33-61, 1998.

Why Sparse?

- The solution is often exactly on an axis.



ℓ_1 constrained LS



Quadratically constrained LS

How to Obtain Solutions

- Lagrangian:

$$J_{\ell_1 CLS}(\boldsymbol{\alpha}) = J_{LS}(\boldsymbol{\alpha}) + \lambda(\|\boldsymbol{\alpha}\|_1 - C)$$

- λ :Lagrange multiplier
- Similar to QCLS, we practically start from $\lambda\ (\geq 0)$ and solve

$$\hat{\boldsymbol{\alpha}}_{\ell_1 CLS} = \mathop{\mathrm{argmin}}_{\boldsymbol{\alpha} \in \mathbb{R}^b} J_{\ell_1 CLS}(\boldsymbol{\alpha})$$

- It is often called ℓ_1 regularized LS.

How to Obtain Solutions (cont.)

- How to deal with ℓ_1-norm?
- Use the following lemma:

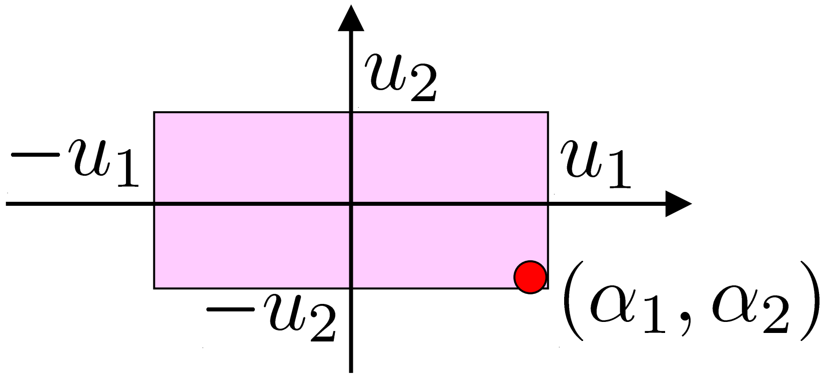
Lemma

$$\|\boldsymbol{\alpha}\|_1 = \min_{\boldsymbol{u}\in\mathbb{R}^b} \sum_{i=1}^{b} u_i$$

$$\text{subject to } -\boldsymbol{u} \leq \boldsymbol{\alpha} \leq \boldsymbol{u},$$

Note: Inequality in constraint is component-wise

Intuition: Obtain smallest box that includes $\boldsymbol{\alpha}$



Proof of Lemma

- Proof: Let $$\hat{\boldsymbol{u}} = \mathop{\mathrm{argmin}}_{\boldsymbol{u} \in \mathbb{R}^b} \sum_{i=1}^{b} u_i$$ $$\text{subject to } -\boldsymbol{u} \leq \boldsymbol{\alpha} \leq \boldsymbol{u},$$

  The constraint implies $\hat{u}_i \geq |\alpha_i|$.
  Suppose $\hat{u}_i > |\alpha_i|$. Then such $\hat{u}_i$ is not a solution since $\tilde{u}_i = |\alpha_i|$ gives a smaller value:

$$\sum_{i=1}^{b} \tilde{u}_i < \sum_{i=1}^{b} \hat{u}_i$$

  This implies that the solution satisfies $\hat{u}_i = |\alpha_i|$, which yields

$$\sum_{i=1}^{b} \hat{u}_i = \sum_{i=1}^{b} |\alpha_i| = \|\boldsymbol{\alpha}\|_1$$

(Q.E.D.)

How to Obtain Solutions (cont.)

$$\hat{\boldsymbol{\alpha}}_{\ell_1 CLS} = \mathop{\mathrm{argmin}}_{\boldsymbol{\alpha} \in \mathbb{R}^b} J_{\ell_1 CLS}(\boldsymbol{\alpha})$$

$$J_{\ell_1 CLS}(\boldsymbol{\alpha}) = J_{LS}(\boldsymbol{\alpha}) + \lambda \|\boldsymbol{\alpha}\|_1$$

- $\hat{\boldsymbol{\alpha}}_{\ell_1 CLS}$ is given as the solution of

$$\min_{\boldsymbol{\alpha}, \boldsymbol{u} \in \mathbb{R}^b} \left[J_{LS}(\boldsymbol{\alpha}) + \lambda \sum_{i=1}^{b} u_i \right]$$

$$\text{subject to } -\boldsymbol{u} \leq \boldsymbol{\alpha} \leq \boldsymbol{u},$$

$$J_{LS}(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \left(\hat{f}(\boldsymbol{x}_i) - y_i \right)^2$$

$$= \|\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{y}\|^2$$

Linearly Constrained Quadratic Programming Problem

- Standard optimization software can solve the following form of linearly constrained quadratic programming problems.

$$\min_{\boldsymbol{\beta}} \left[\frac{1}{2} \langle \boldsymbol{Q}\boldsymbol{\beta}, \boldsymbol{\beta} \rangle + \langle \boldsymbol{\beta}, \boldsymbol{q} \rangle \right]$$

$$\text{subject to } \boldsymbol{V}\boldsymbol{\beta} \le \boldsymbol{v}$$

$$\boldsymbol{G}\boldsymbol{\beta} = \boldsymbol{g}$$

Transforming into Standard Form

- Let

$$\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{\alpha} \\ \boldsymbol{u} \end{pmatrix}$$

$$\begin{aligned} \boldsymbol{\Gamma}_{\boldsymbol{\alpha}} &= (\boldsymbol{I}_b, \boldsymbol{O}_b) \\ \boldsymbol{\Gamma}_{\boldsymbol{u}} &= (\boldsymbol{O}_b, \boldsymbol{I}_b) \end{aligned}$$

- Then

$$\begin{aligned} \boldsymbol{\alpha} &= \boldsymbol{\Gamma}_{\boldsymbol{\alpha}}\boldsymbol{\beta} \\ \boldsymbol{u} &= \boldsymbol{\Gamma}_{\boldsymbol{u}}\boldsymbol{\beta} \end{aligned}$$

- Use these expressions and replace all $\boldsymbol{\alpha}, \boldsymbol{u}$ with $\boldsymbol{\beta}$.

Standard Form

$$\min_{\boldsymbol{\beta}} \left[\frac{1}{2} \langle \boldsymbol{Q}\boldsymbol{\beta}, \boldsymbol{\beta} \rangle + \langle \boldsymbol{\beta}, \boldsymbol{q} \rangle \right]$$

$$\text{subject to } \boldsymbol{V}\boldsymbol{\beta} \leq \boldsymbol{v}$$
$$\boldsymbol{G}\boldsymbol{\beta} = \boldsymbol{g}$$

- ℓ_1-constrained LS can be expressed as

$$\begin{aligned}
\boldsymbol{Q} &= 2\boldsymbol{\Gamma}_{\boldsymbol{\alpha}}^\top \boldsymbol{X}^\top \boldsymbol{X} \boldsymbol{\Gamma}_{\boldsymbol{\alpha}} \\
\boldsymbol{q} &= -2\boldsymbol{\Gamma}_{\boldsymbol{\alpha}}^\top \boldsymbol{X}^\top \boldsymbol{y} + \lambda \boldsymbol{\Gamma}_{\boldsymbol{u}}^\top \mathbf{1}_b \\
\boldsymbol{V} &= \begin{pmatrix} -\boldsymbol{\Gamma}_{\boldsymbol{\alpha}} - \boldsymbol{\Gamma}_{\boldsymbol{u}} \\ \boldsymbol{\Gamma}_{\boldsymbol{\alpha}} - \boldsymbol{\Gamma}_{\boldsymbol{u}} \end{pmatrix} \\
\boldsymbol{v} &= \mathbf{0}_{2b} \\
\boldsymbol{G} &= \boldsymbol{O}_{2b} \\
\boldsymbol{g} &= \mathbf{0}_{2b}
\end{aligned}$$

$$\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{\alpha} \\ \boldsymbol{u} \end{pmatrix}$$

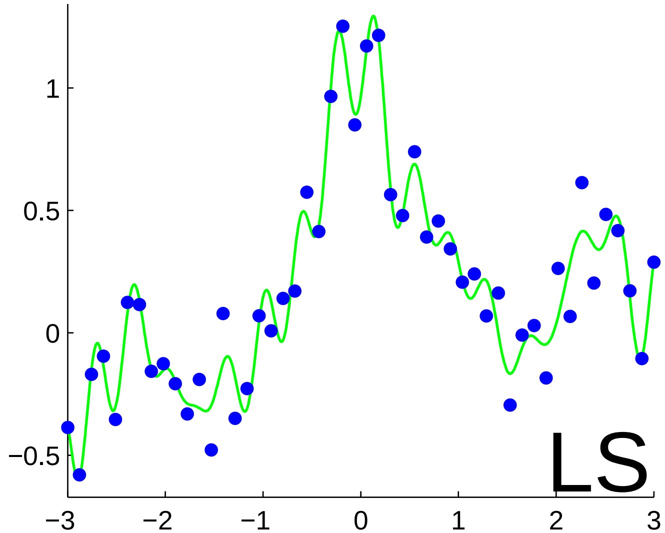
$$\begin{aligned}
\boldsymbol{\Gamma}_{\boldsymbol{\alpha}} &= (\boldsymbol{I}_b, \boldsymbol{O}_b) \\
\boldsymbol{\Gamma}_{\boldsymbol{u}} &= (\boldsymbol{O}_b, \boldsymbol{I}_b)
\end{aligned}$$

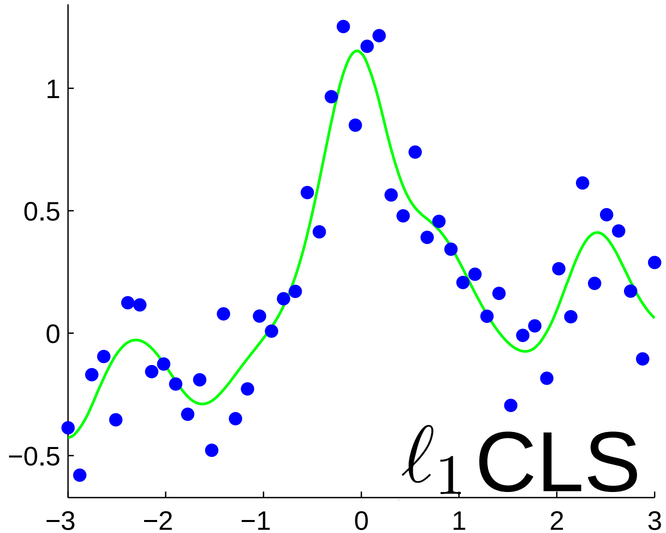
Proof: Homework!

Example of Sparse Learning

- Gaussian kernel model:

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i) \qquad K(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\|\boldsymbol{x} - \boldsymbol{x}'\|^2/2\right)$$





- Over-fit can be avoided by properly choosing the regularization factor λ .
- 27 out of 50 parameters are exactly zero.

Feature Selection

- If ℓ_1 CLS is combined with linear model with respect to input,

$$\hat{f}(\boldsymbol{x}) = \boldsymbol{\alpha}^\top \boldsymbol{x} \qquad \boldsymbol{x} = (x^{(1)}, x^{(2)}, \ldots, x^{(d)})^\top$$

some of the input variables are not used for prediction.

→ Important features are automatically selected

- Example: Gene selection
- Generally, 2^d combinations need to be tested for feature selection (cf. SLS).
- On the other hand, ℓ_1 CLS only involves a continuous model parameter λ.

Constrained LS

	Sparseness	Model parameter	Parameter learning
Subspace LS	Yes	Discrete	Analytic (Linear)
Quadratically constrained LS	No	Continuous	Analytic (Linear)
ℓ_1 constrained LS	Yes	Continuous	Iterative (Non-linear)

Homework

1. Derive the standard quadratic programming form of ℓ_1 -constrained LS.

$$\min_{\boldsymbol{\beta}} \left[\frac{1}{2} \langle \boldsymbol{Q}\boldsymbol{\beta}, \boldsymbol{\beta} \rangle + \langle \boldsymbol{\beta}, \boldsymbol{q} \rangle \right]$$

$$\text{subject to } \boldsymbol{V}\boldsymbol{\beta} \leq \boldsymbol{v}$$

$$\boldsymbol{G}\boldsymbol{\beta} = \boldsymbol{g}$$

$$\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{\alpha} \\ \boldsymbol{u} \end{pmatrix}$$

$$\begin{aligned} \boldsymbol{\Gamma}_{\boldsymbol{\alpha}} &= (\boldsymbol{I}_b, \boldsymbol{O}_b) \\ \boldsymbol{\Gamma}_{\boldsymbol{u}} &= (\boldsymbol{O}_b, \boldsymbol{I}_b) \end{aligned}$$

$$\begin{aligned} \boldsymbol{Q} &= 2\boldsymbol{\Gamma}_{\boldsymbol{\alpha}}^\top \boldsymbol{X}^\top \boldsymbol{X} \boldsymbol{\Gamma}_{\boldsymbol{\alpha}} \\ \boldsymbol{q} &= -2\boldsymbol{\Gamma}_{\boldsymbol{\alpha}}^\top \boldsymbol{X}^\top \boldsymbol{y} + \lambda \boldsymbol{\Gamma}_{\boldsymbol{u}}^\top \mathbf{1}_b \\ \boldsymbol{V} &= \begin{pmatrix} -\boldsymbol{\Gamma}_{\boldsymbol{\alpha}} - \boldsymbol{\Gamma}_{\boldsymbol{u}} \\ \boldsymbol{\Gamma}_{\boldsymbol{\alpha}} - \boldsymbol{\Gamma}_{\boldsymbol{u}} \end{pmatrix} \\ \boldsymbol{v} &= \mathbf{0}_{2b} \\ \boldsymbol{G} &= \boldsymbol{O}_{2b} \\ \boldsymbol{g} &= \mathbf{0}_{2b} \end{aligned}$$

Homework (cont.)

2. For your own toy 1-dimensional data, perform simulations using
   - Gaussian kernel models
   - ℓ_1-constraint least-squares learning

   and analyze the results, e.g., by changing
   - Target functions
   - Number of samples
   - Noise level

   Use 5-fold cross-validation for choosing
   - Width of Gaussian kernel
   - Regularization parameter

   Compare the results of QCLS and ℓ_1CLS, e.g., in terms of sparseness and accuracy.