Machine Learning: Optimization 4

Hiroshi Shimodaira and Hao Tang

2025

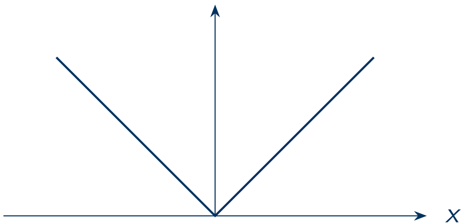
Ver. 1.0

Topics

- Subgraident
- Hinge loss
- Constrained optimisation problems
- Feasible solutions
- Lagrangian and Lagrange multiplier

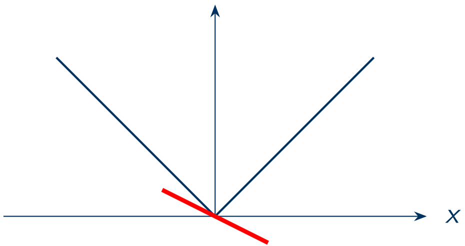
Subgradients for absolute values

$f(x) = |x|$



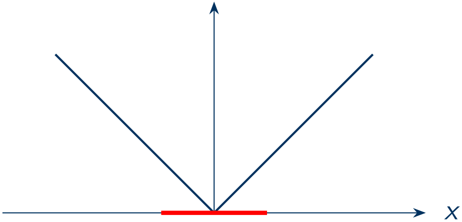
Subgradients for absolute values

$$f(x) = |x|$$



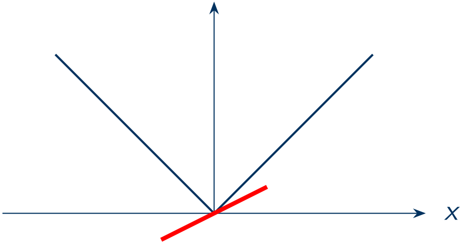
Subgradients for absolute values

$f(x) = |x|$



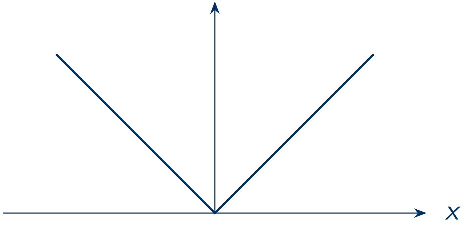
Subgradients for absolute values

$f(x) = |x|$

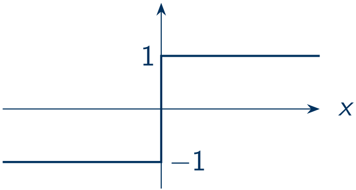


Subgradients for absolute values

$f(x) = |x|$



$\partial\,|x|$



$$\partial\,|x| = \begin{cases} \{-1\} & \text{if } x < 0 \\ [-1, 1] & \text{if } x = 0 \\ \{+1\} & \text{if } x > 0 \end{cases}$$

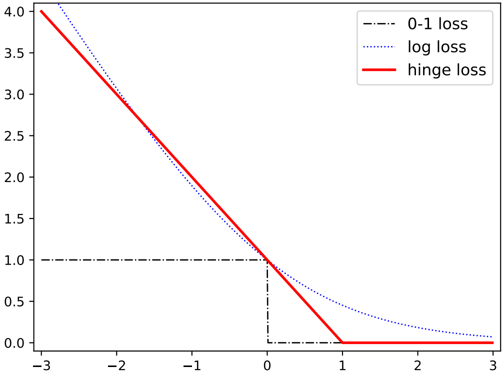
Subgradient

- A subgradient at $\boldsymbol{x}$ is a vector $\boldsymbol{g}$ that satisfies

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \boldsymbol{g}^\top(\boldsymbol{y} - \boldsymbol{x}) \tag{1}$$

  for any $\boldsymbol{y}$, and the set of subgradients at $\boldsymbol{x}$ is denoted as $\partial f(\boldsymbol{x})$.
- Obviously, $\nabla f(\boldsymbol{x}) \in \partial f(\boldsymbol{x})$, if $\nabla f(\boldsymbol{x})$ exists.
- Convergence theorems can be ported to subgradient descent.

Hinge loss



Hinge loss *(cont.)*

- The hinge loss is defined as ($\hat{y}$: the raw output of classifier)

$$\ell_{\text{hinge}}(\hat{y}, y) = \max(0,\, 1 - \hat{y}y) \tag{2}$$

for a linear classifier

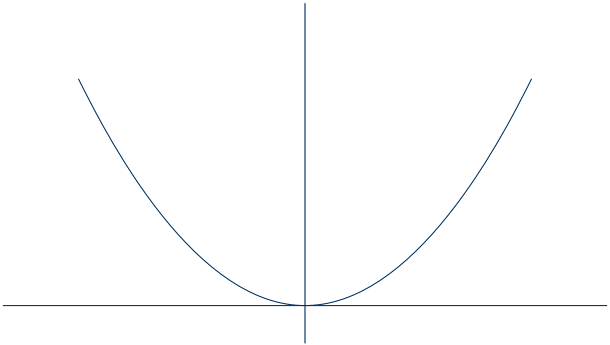
$$\ell_{\text{hinge}}(\boldsymbol{w}; \boldsymbol{x}, y) = \max(0,\, 1 - y\boldsymbol{w}^\top \boldsymbol{x}). \tag{3}$$

- Just like the absolute value, the hinge loss is continuous and convex, but it is not differentiable.

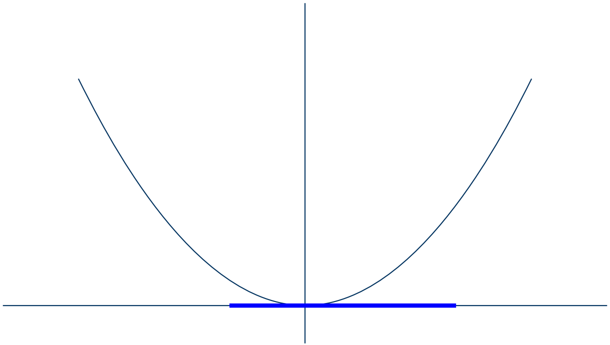
$$\nabla_{\boldsymbol{w}}\, \ell_{\text{hinge}} = \begin{cases} \mathbf{0} & \text{if } y\boldsymbol{w}^\top \boldsymbol{x} \geq 1 \\ -y\boldsymbol{x} & \text{if } y\boldsymbol{w}^\top \boldsymbol{x} < 1 \end{cases} \tag{4}$$

- When $y\boldsymbol{w}^\top \boldsymbol{x} = 1$, we can pick and choose any vector that supports the loss function from below as the subgradient. In fact, $\mathbf{0}$ and $-y\boldsymbol{x}$ both work.

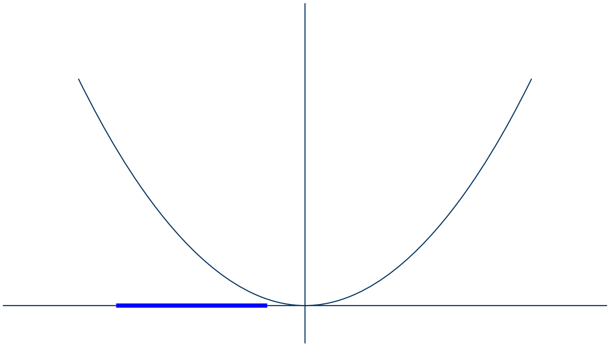
Constrained optimisation



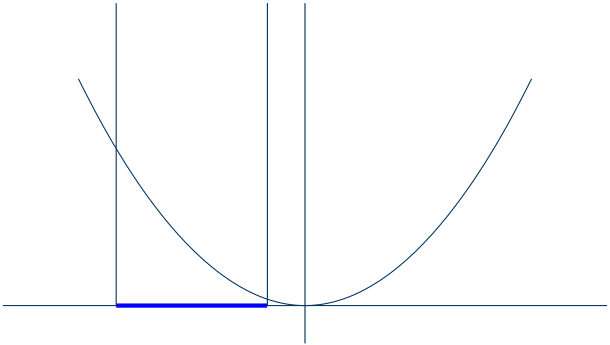
Constrained optimisation



Constrained optimisation



Setting up a barrier



An example optimisation-problem with constraints

- The problem

$$\begin{aligned} \min_{x} \quad & x^2 \\ \text{s.t.} \quad & -2.5 \leq x \leq -0.5 \end{aligned} \tag{5}$$

  is an example of a constrained optimisation problem.

- The inequality $-2.5 \leq x \leq -0.5$ is called a *constraint*.

- Solutions that satisfy the constraints are called **feasible** solutions.

Setting up a barrier

- The problem

$$\begin{aligned} \min_{x} \quad & x^2 \\ \text{s.t.} \quad & -2.5 \leq x \leq -0.5 \end{aligned} \tag{6}$$

is equivalent to

$$\min_{x} x^2 + V_{-}(x) \tag{7}$$

where

$$V_{-}(x) = \begin{cases} 0 & \text{if } -2.5 \leq x \leq -0.5 \\ \infty & \text{otherwise} \end{cases} \tag{8}$$

An example optimisation-problem with constraints

- The problem

$$\begin{aligned} \min_{\boldsymbol{w}} \quad & L(\boldsymbol{w}) \\ \text{s.t.} \quad & \|\boldsymbol{w}\|_2^2 \leq 1 \end{aligned} \tag{9}$$

  is an example of a constrained optimisation problem.

- The inequality $\|\boldsymbol{w}\|_2^2 \leq 1$ is called a *constraint*.

- Solutions that satisfy the constraints are called **feasible** solutions.

Setting up a barrier

- We can write the optimisation problem as

$$\min_{\boldsymbol{w}} \qquad L(\boldsymbol{w}) + V_{-}(\|\boldsymbol{w}\|_2^2 - 1), \tag{10}$$

where

$$V_{-}(s) = \begin{cases} 0 & \text{if } s \leq 0 \\ \infty & \text{if } s > 0 \end{cases}. \tag{11}$$

- This does not change anything; both problems are equally hard (or easy) to solve.

Soften the constraints

- We can approximate

$$\min_{w} \quad L(\boldsymbol{w}) + V_{-}(\|\boldsymbol{w}\|_2^2 - 1) \tag{12}$$

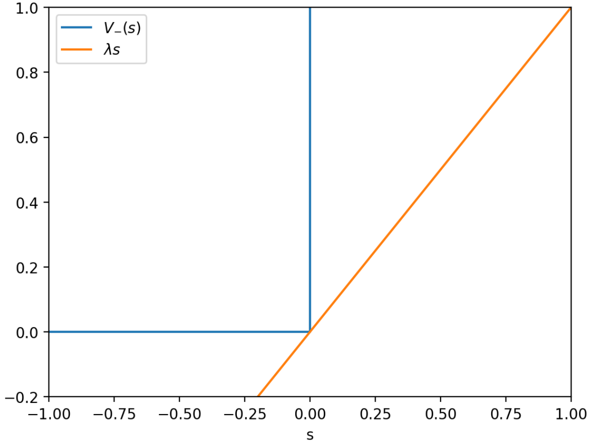
with

$$\min_{w} \quad L(\boldsymbol{w}) + \lambda(\|\boldsymbol{w}\|_2^2 - 1), \tag{13}$$

for some $\lambda \geq 0$.

- Note that $\lambda s \leq V_{-}(s)$ for all s.

Soften the constraints *(cont.)*



Lagrangian

- In general, if you have a optimisation problem

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & h(\boldsymbol{x}) \leq 0 \end{aligned} \tag{14}$$

  the **Lagrangian** is defined as

$$f(\boldsymbol{x}) + \lambda h(\boldsymbol{x}) \tag{15}$$

  for $\lambda \geq 0$.

- The value λ is called the **Lagrange multiplier**.

Solving the Lagrangian

- Solve $g(\lambda) = \min_{\boldsymbol{x}}[f(\boldsymbol{x}) + \lambda h(\boldsymbol{x})]$ for a particular λ.
- Find $\hat{\lambda}$ such that $\min_{\boldsymbol{x}}[f(\boldsymbol{x}) + \hat{\lambda} h(\boldsymbol{x})]$ gives a feasible solution.
- Suppose $\hat{\boldsymbol{x}} = \operatorname{argmin}_{\boldsymbol{x}}[f(\boldsymbol{x}) + \hat{\lambda} h(\boldsymbol{x})]$ and $\boldsymbol{x}^* = \operatorname{argmin}_{\boldsymbol{x}:h(\boldsymbol{x})\leq 0} f(\boldsymbol{x})$.

$$f(\hat{\boldsymbol{x}}) + \hat{\lambda} h(\hat{\boldsymbol{x}}) \;\leq\; f(\boldsymbol{x}^*) + \hat{\lambda} h(\boldsymbol{x}^*) \;\leq\; f(\boldsymbol{x}^*) \tag{16}$$

Solving the Lagrangian *(cont.)*

- We want $f(\hat{\boldsymbol{x}}) = f(\hat{\boldsymbol{x}}) + \hat{\lambda} h(\hat{\boldsymbol{x}})$ leading to $f(\hat{\boldsymbol{x}}) \leq f(\boldsymbol{x}^*)$, so that we can conclude $f(\hat{\boldsymbol{x}}) = f(\boldsymbol{x}^*)$.
- If we want $\hat{\lambda} h(\hat{\boldsymbol{x}}) = 0$, then either $\hat{\lambda} = 0$ or $h(\hat{\boldsymbol{x}}) = 0$.
  - When $\hat{\lambda} = 0$, the minimiser of f is a feasible solution already.
  - When $h(\hat{\boldsymbol{x}}) = 0$, the minimiser of f is not necessarily a feasible solution, and we are on the edge of a constraint.

Example 1 - training of a word unigram model

Row, row, row your boat, gently down the stream
Merrily, merrily, merrily, merrily, life is but a dream

Example 1 - training of a word unigram model

Row, row, row your boat, gently down the stream
Merrily, merrily, merrily, merrily, life is but a dream

- There are 18 words.
- Intuitively,

$$p(\text{row}) = \frac{3}{18} \qquad p(\text{merrily}) = \frac{4}{18} \qquad p(\text{is}) = \frac{1}{18} \tag{17}$$

Example 1 - training of a word unigram model *(cont.)*

- There are 13 unique words.

- We refer to the set of unique words $V = \{\text{row}, \text{your}, \text{boat}, \text{gently}, \text{down}, \text{the}, \text{stream}, \text{merrily}, \text{life}, \text{is}, \text{but}, \text{a}, \text{dream}\}$ as the vocabulary.

- We assign each word v a probability β_v.

- The probability of a word is

$$p(w) = \prod_{v \in V} \beta_v^{\mathbb{1}_{v=w}}. \tag{18}$$

Example 1 - training of a word unigram model *(cont.)*

- We assume that each word is independent of others.
- This assumption is obviously wrong, but can go really far.
- The likelihood of β given the data is

$$\log p(w_1, \dots, w_N) = \log \prod_{i=1}^{N} p(w_i) = \log \prod_{i=1}^{N} \prod_{v \in V} \beta_v^{\mathbb{1}_{v=w_i}}. \tag{19}$$

- Since β is a probability vector, we have the assumption

$$\sum_{v \in V} \beta_v = 1. \tag{20}$$

Example 1 - training of a word unigram model *(cont.)*

- We arrive at the optimisation problem

$$\begin{aligned} \min_{\beta} \quad & -\sum_{i=1}^{N} \sum_{v \in V} \mathbb{1}_{v=w_i} \log \beta_v \\ \text{s.t.} \quad & \sum_{v \in V} \beta_v = 1 \end{aligned} \tag{21}$$

- Its Lagrangian is

$$F = -\sum_{i=1}^{N} \sum_{v \in V} \mathbb{1}_{v=w_i} \log \beta_v + \lambda \left(\sum_{v \in V} \beta_v - 1 \right). \tag{22}$$

Example 1 - training of a word unigram model (*cont.*)

- Solving the optimality condition gives

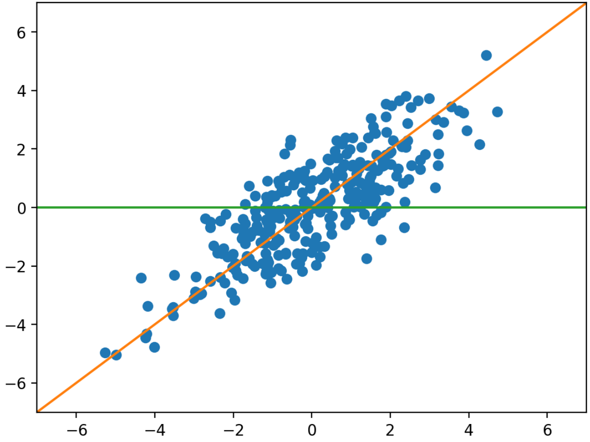
$$\frac{\partial F}{\partial \beta_k} = \sum_{i=1}^{N} \mathbb{1}_{k=w_i} \frac{1}{\beta_k} - \lambda = 0 \implies \beta_k = \frac{1}{\lambda} \sum_{i=1}^{N} \mathbb{1}_{k=w_i}. \tag{23}$$

Example 1 - training of a word unigram model *(cont.)*

$$\sum_{v \in V} \beta_v = \sum_{v \in V} \frac{1}{\lambda} \sum_{i=1}^{N} \mathbb{1}_{v=w_i} = 1 \implies \lambda = \sum_{v \in V} \sum_{i=1}^{N} \mathbb{1}_{v=w_i} = N \tag{24}$$

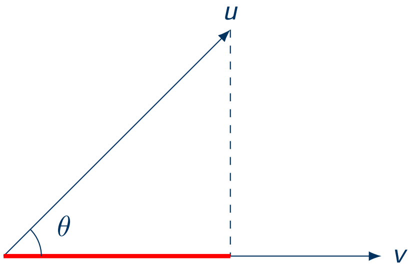
$$\beta_k = \frac{\sum_{i=1}^{N} \mathbb{1}_{k=w_i}}{\sum_{v \in V} \sum_{i=1}^{N} \mathbb{1}_{v=w_i}} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{k=w_i} \tag{25}$$

Example 2 - finding the best projection line/hyperplane



Projection of a vector

Projection of $\boldsymbol{u}$ onto/from $\boldsymbol{v}$



$$\|\boldsymbol{u}\|_2 \cos\theta = \|\boldsymbol{u}\|_2 \frac{\boldsymbol{u}^\top \boldsymbol{v}}{\|\boldsymbol{u}\|_2 \|\boldsymbol{v}\|_2} = \frac{\boldsymbol{u}^\top \boldsymbol{v}}{\|\boldsymbol{v}\|_2} \tag{26}$$

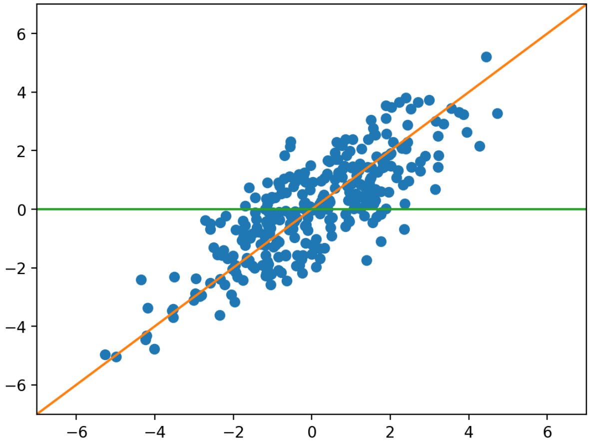
Example 2 - finding the best projection line/hyperplane (*cont.*)

- The projection of $\boldsymbol{x}$ onto $\boldsymbol{w}$ is $\frac{\boldsymbol{x}^\top \boldsymbol{w}}{\|\boldsymbol{w}\|_2}$.
- If we have N data points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, then the sum of the (squared) projection is

$$\sum_{i=1}^{N} \left(\frac{|\boldsymbol{x}_i^\top \boldsymbol{w}|}{\|\boldsymbol{w}\|_2} \right)^2 = \frac{\boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X} \boldsymbol{w}}{\boldsymbol{w}^\top \boldsymbol{w}}. \tag{27}$$

- The sum of squared projection can be seen as the spread of the data.

Maximal projection



Maximal projection *(cont.)*

- We want to find the maximum direction to project.
- The optimisation problem is

$$\max_{\boldsymbol{w}} \frac{\boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X} \boldsymbol{w}}{\boldsymbol{w}^\top \boldsymbol{w}}. \tag{28}$$

Maximal projection *(cont.)*

- The problem is scale invariant.

$$\frac{(a\boldsymbol{w})^\top \boldsymbol{X}^\top \boldsymbol{X}(a\boldsymbol{w})}{(a\boldsymbol{w})^\top(a\boldsymbol{w})} = \frac{\boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w}}{\boldsymbol{w}^\top \boldsymbol{w}}. \tag{29}$$

- The problem is equivalent to

$$\max_{\boldsymbol{w}} \boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w} \qquad \text{s.t. } \|\boldsymbol{w}\|_2^2 = 1. \tag{30}$$

Maximal projection *(cont.)*

- The Lagrangian is

$$F = \boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w} + \lambda(1 - \|\boldsymbol{w}\|_2^2). \tag{31}$$

- Finding the optimal solution gives

$$\frac{\partial F}{\partial \boldsymbol{w}} = (\boldsymbol{X}^\top \boldsymbol{X} + \boldsymbol{X}^\top \boldsymbol{X})\boldsymbol{w} - 2\lambda \boldsymbol{w} = 0 \implies \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w} = \lambda \boldsymbol{w}. \tag{32}$$

- It turns out that λ is an eigenvalue, and $\boldsymbol{w}$ an eigenvector of $\boldsymbol{X}^\top \boldsymbol{X}$.

Maximal projection (*cont.*)

- Plugging the solution back to the objective,

$$\frac{\boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w}}{\boldsymbol{w}^\top \boldsymbol{w}} = \frac{\lambda \boldsymbol{w}^\top \boldsymbol{w}}{\boldsymbol{w}^\top \boldsymbol{w}} = \lambda \tag{33}$$

- Since the goal is to find the maximal projection, this is now equivalent to finding the largest eigenvalue of $\boldsymbol{X}^\top \boldsymbol{X}$.

Maximal projection *(cont.)*

- The term

$$\frac{\boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w}}{\boldsymbol{w}^\top \boldsymbol{w}} \tag{34}$$

  is called the Rayleigh quotient.

- The optimal $\boldsymbol{w}$ is called the first principal component.

- We will learn more about this when we talk about principal component analysis.

Quizzes

- Consider a set of two-dimensional data $\{\boldsymbol{x}_i\}_{i=1}^N$, where $\boldsymbol{x}_i = (x_{i1}, x_{i2})^\top$. Explain the difference between the best projection line (defined in the slides) and linear regression line from x_1 to x_2 (or from from x_2 to x_1).