Distributed and Stochastic Optimization for Machine Learning

Marco Cuturi

ENSAE ParisTech
École nationale de la statistique et de l'administration économique
université PARIS-SACLAY

Machine Learning as Optimization

1. Machine Learning often boils down to minimizing
   - The variable to minimize: *a parameter* which describes the machine.
   - The objective: *fitting error* with respect to data sample + *regularization*
   - This can be interpreted as *likelihood* + *prior* of the parameter.

2. The structure of that minimization is peculiar
   - The *fitting error* is either an integral or a sum.
   - The *regularization* term is usually a simple function.

3. Dimensions are a problem (>2000's)
   - The parameter space is usually very large. Sometimes even the **parameter** hardly fits in a single machine (NN).
   - The space required to store a single data point might be large.
   - If evaluated on a finite *sum*, the number of points is usually **huge**. Data cannot fit on a single machine either.

Machine Learning as Optimization

1. Machine Learning often boils down to minimizing
   - The variable to minimize: *a parameter* which describes the machine.
   - The objective: *fitting error* with respect to data sample + *regularization*

Only tractable computer implementation is to randomize and/or to distribute computations. This is the topic addressed in these lectures.

2.

3.

parameter hardly fits in a single machine (NN).
- The space required to store a single data point might be large.
- If evaluated on a finite *sum*, the number of points is usually **huge**. Data cannot fit on a single machine either.

Self-introduction

- ENSAE ('01) / MVA / Phd. ENSMP / Japan & US
  - post-doc then hedge-fund in Japan ('05~'08)
  - Lecturer @ Princeton University ('09~'10)
  - Assoc. Prof. @ Kyoto University ('10~'16)
  - Prof @ ENSAE since 9/'16.
- Active in ML community, stats/optim flavor.
  - Attend & publish regularly in *NIPS & ICML.*
- Interests
  - Optimal transport, kernel methods, time series.

Practical Aspects

- Reach me:
  - Bureau: E02, entresol.
  - email: `marco.cuturi@ensae.fr`
  - page web: `http://marcocuturi.net`
- Lectures: structure
  - 6 x 2h class.
  - 3 x 2h python hands-on sessions, Fabian Pedregosa.
  - Last class (intro to text data) Stéphanie Combes.
  - Validation through memoir.
- No notes yet. pointers to relevant material:
  - 1606.04838v1.pdf
  - Taiji Suzuki's slides: http://bit.ly/taiji_slides

Schedule

1. Introduction
    - Link between ML - Optimisation. (R)(E)RM problems
    - Convexity, Fenchel duality
2. Stochastic gradient (SG) method
3. Incremental gradient methods
4. Curvature: second order methods for SG
5. Asynchronous optimization
6. Distributed optimization

Tentative list of ingredients in *batch* ML

$$\{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathcal{X} \times \mathcal{Y})^n$$

samples from $p \in \mathcal{P}(\mathcal{X} \times \mathcal{Y})$

loss function $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$

function class $\mathcal{F} = \{f_\theta : \mathcal{X} \to \mathcal{Y}, \theta \in \Theta\}$

regularizer $\psi : \Theta \to \mathbb{R}_+$

Tentative list of ingredients in *batch* ML

$$\{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathcal{X} \times \mathcal{Y})^n$$

$n \approx \infty$

$\dim(\mathcal{X}) \approx \infty$

samples from $p \in \mathcal{P}(\mathcal{X} \times \mathcal{Y})$

loss function $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$

function class $\mathcal{F} = \{f_\theta : \mathcal{X} \to \mathcal{Y}, \theta \in \Theta\}$

regularizer $\psi : \Theta \to \mathbb{R}_+$

Goal of Batch ML

1. The elusive golden standard: Risk Minimization

$$\boxed{\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mathbb{E}_p[\boldsymbol{l}(f_{\boldsymbol{\theta}}(X), Y)]}$$

2. The naive alternative: Empirical Risk Minimization

$$\boxed{\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{l}(f_{\boldsymbol{\theta}}(x_i), y_i)}$$

Supervised ML

3. The reasonable compromise

$$\boxed{\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{l}(f_{\boldsymbol{\theta}}(x_i), y_i)}$$

From an optimization point of view:

- parameter size is huge.
- loss and regularizer functions might be ugly.
- *n* points might be too much for a single RAM machine (~256Gb *vs.* a few terabytes of more for modern datasets).

Supervised ML

3. The reasonable compromise

$$\boxed{\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{l}(f_{\boldsymbol{\theta}}(x_i), y_i) + \boldsymbol{\psi}(\boldsymbol{\theta})}$$

From an optimization point of view:

- parameter size is huge.
- loss and regularizer functions might be ugly.
- *n* points might be too much for a single RAM machine (~256Gb *vs.* a few terabytes of more for modern datasets).

Tentative list of ingredients in online ML

$$(x_t, y_t) \in \mathcal{X} \times \mathcal{Y}, t \geq 0.$$

each sampled from $p \in \mathcal{P}(\mathcal{X} \times \mathcal{Y})$

loss function $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$

function class $\mathcal{F} = \{f_\theta : \mathcal{X} \to \mathcal{Y}, \theta \in \Theta\}$

regularizer $\psi : \Theta \to \mathbb{R}_+$

Online ML

1. Same risk minimization ideal

$$\boxed{\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathbb{E}_p[\boldsymbol{l}(f_{\boldsymbol{\theta}}(X), Y)]}$$

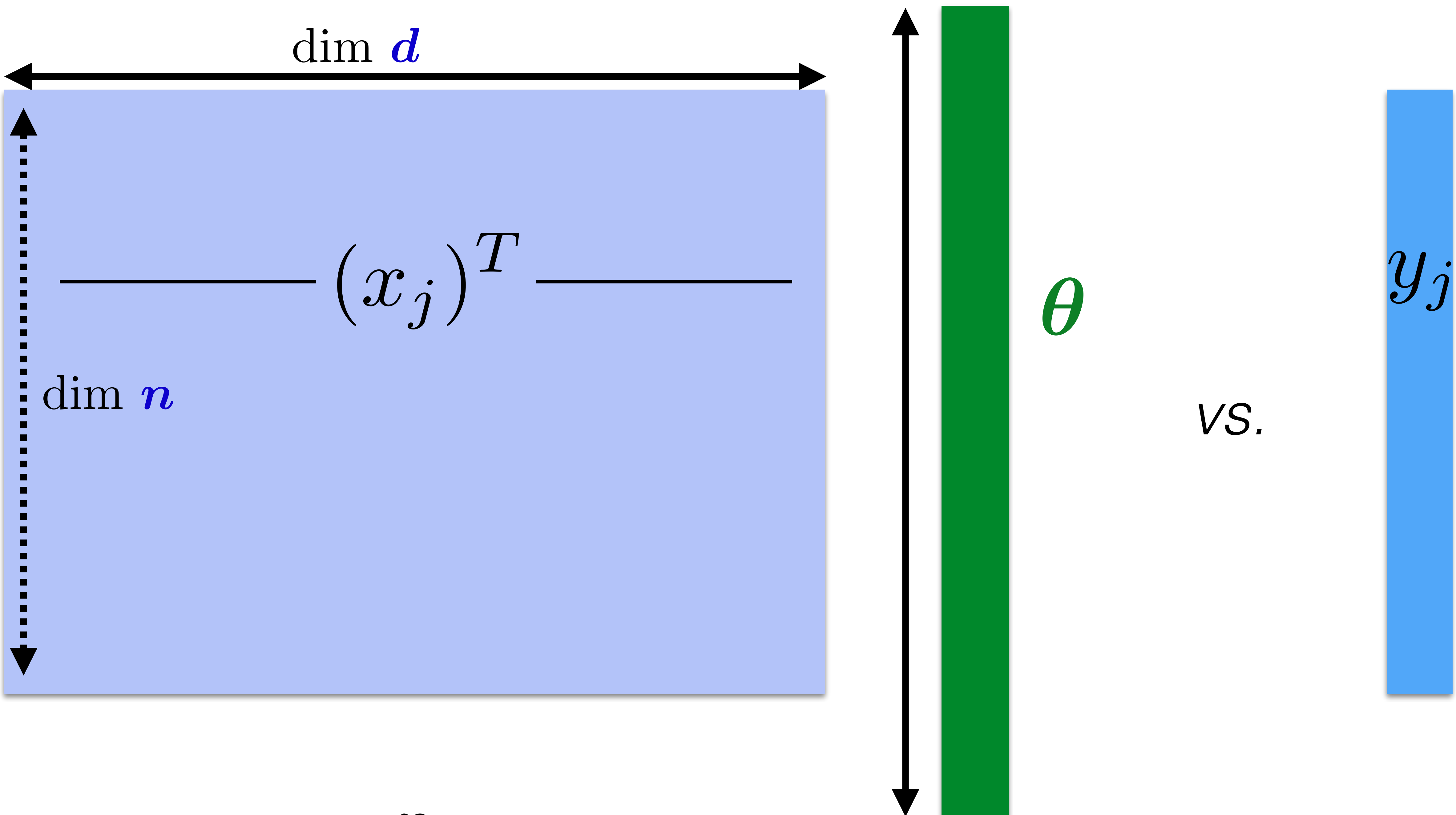
2. Due to practical constraints, samples only come **one by one**, each at a time t, and **cannot be stored**. Only previous parameter is stored.

$$\boxed{\boldsymbol{\theta_t} = F(\boldsymbol{l}(f_{\boldsymbol{\theta_{t-1}}}(x_t), y_t), \boldsymbol{\psi}, \boldsymbol{\theta_{t-1}})}$$

From an optimization point of view:

- size problem is gone.
- refresh speed might be *very* fast.
- What update rule can we consider to guarantee good approx.?

Example: Regression (Regularized)



$$\min_{\theta, b} \frac{1}{n} \sum_{j=1}^{n} (x_j^T \theta + b - y_j)^2 + \lambda \|\theta\|_q^q$$

Example: Binary Classification *(linear)*

$$\{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathbb{R}^p \times \{-1, 1\})^n$$

$$\text{0-1 loss} : l(a, b) = \mathbf{1}_{a \neq b}$$

$$\mathcal{F} = \{f_\theta : x \mapsto \mathrm{sign}(w^T x + b), \theta = (\omega, b) \in \mathbb{R}^{p+1}\}$$

$$\psi(\omega, b) = \frac{1}{2}\|w\|^2$$

$$\boxed{\min_{\boldsymbol{w}, \boldsymbol{b}} \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{(f_{\boldsymbol{w}, \boldsymbol{b}}(x_i) \neq y_i)} + \frac{1}{2}\|\boldsymbol{w}\|^2}$$

Cleaner optimization setup for ML

$$\{z_1, \ldots, z_n\} \in (\mathcal{X} \times \mathcal{Y})^n$$

$$\{l_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+\}_{\theta \in \Theta}$$

$$\psi : \Theta \to \mathbb{R}_+$$

$$\boxed{\min_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^{n} l_{\boldsymbol{\theta}}(z_i) + \psi(\boldsymbol{\theta})}$$

Examples

1. Support Vector Machine.
2. Logistic regression.
3. Multiclass logistic regression with a KL loss.
4. Multiclass logistic regression with a Wasserstein loss.

Reminders on Convexity: *Sets*

- Line segment between two points in Hilbert space:

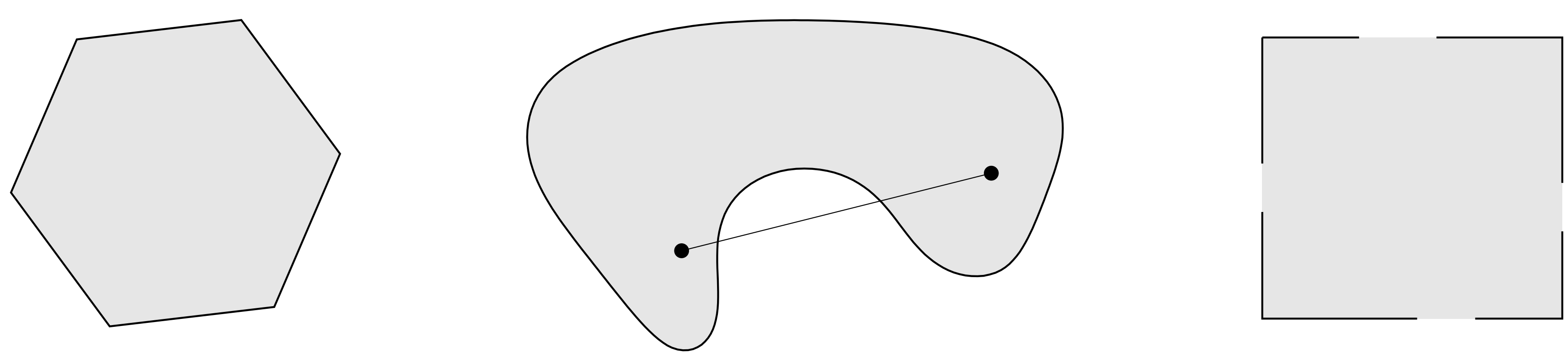
$$\{x = \lambda x_1 + (1-\lambda)x_2, \quad 0 \leq \lambda \leq 1\}$$

- A convex set contains all segments of all its points

Def

C is convex $\Leftrightarrow \forall x_1, x_2 \in C, 0 \leq \lambda \leq 1; \quad \lambda x_1 + (1-\lambda)x_2 \in C$

- Examples

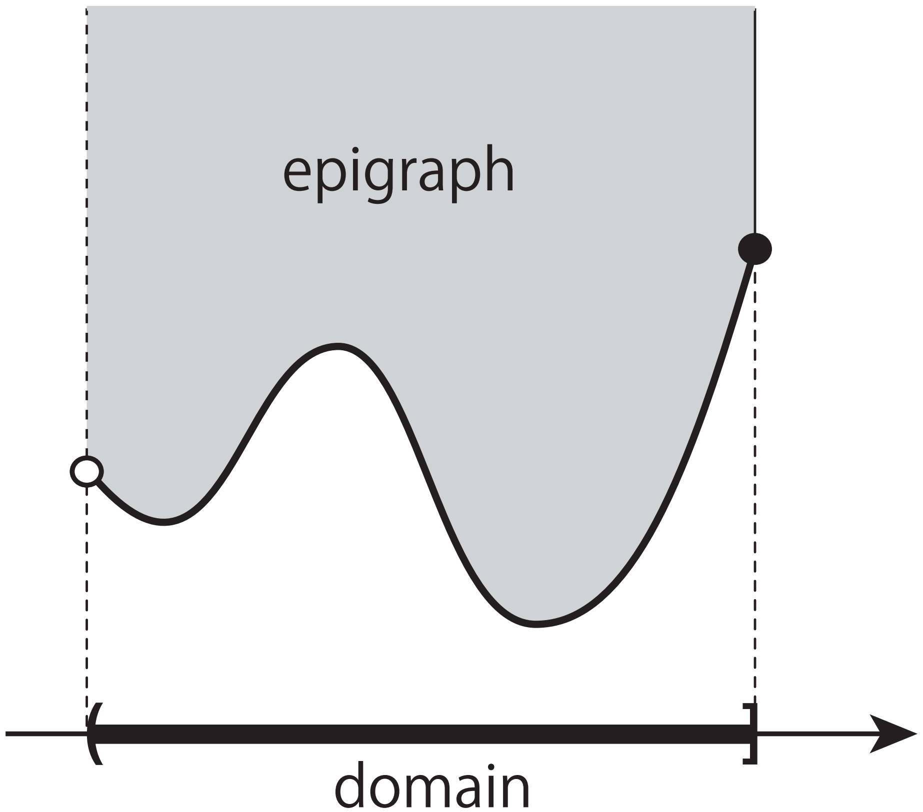


Reminders on Convexity: *Epigraph*

- Epigraphs and domain

Def

$$\mathrm{epi}(f) = \{(x,t) \in \mathbb{R}^p \times \mathbb{R} : f(x) \leq t\}$$
$$\mathrm{dom}(f) = \{x \in \mathbb{R}^p : f(x) < \infty\}$$



Reminders on Convexity: *Functions*

- Convex function

$$\bar{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$$

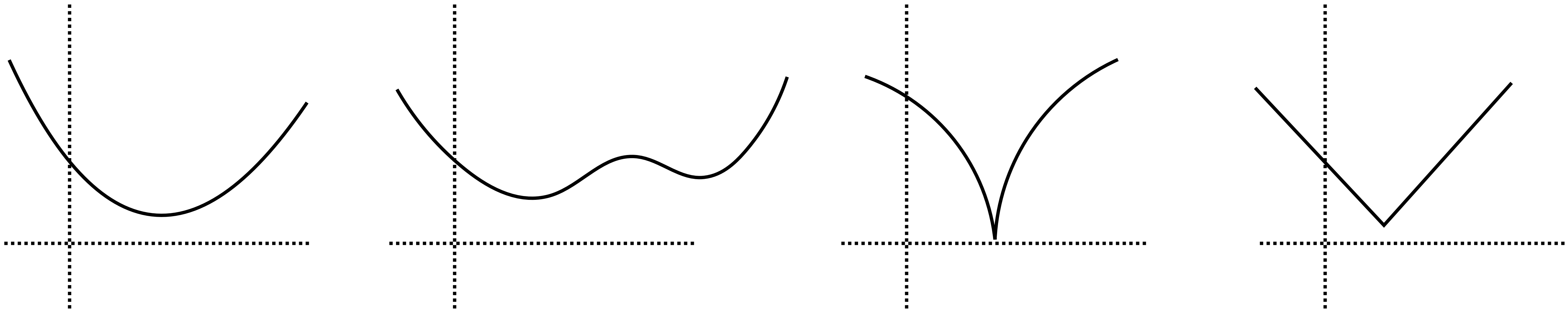
Def

$$f : \mathbb{R}^p \to \bar{\mathbb{R}} \text{ convex}$$

$$\Updownarrow$$

$$\forall x_1, x_2 \in \mathbb{R}^p, 0 \leq \lambda \leq 1,$$

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$$



Reminders on Convexity: *Functions*

- Convex function

$$\bar{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$$

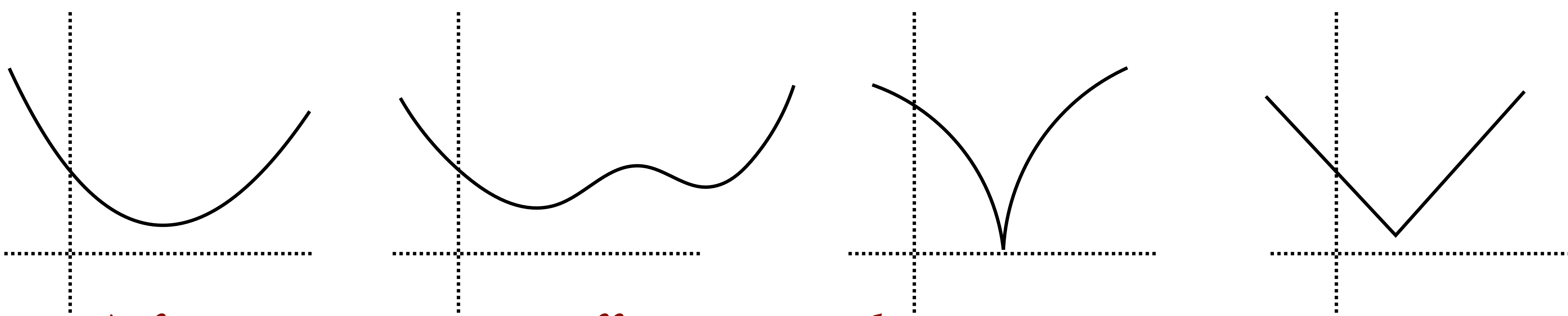
Def

$$f : \mathbb{R}^p \to \bar{\mathbb{R}} \text{ convex}$$

$$\Updownarrow$$

$$\forall x_1, x_2 \in \mathbb{R}^p, 0 \leq \lambda \leq 1,$$

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$$



- *A function is convex iff its epigraph is.*

convex loss functions for regression

- Label is a real number (regression)

$$l(u,y) = \frac{1}{2}(u-y)^2,$$

quadratic

$$l_\tau(u,y) = (1-\tau)\max(u-y,0) + \tau\max(y-u,0), \tau \in [0,1]$$

$$l_\varepsilon(u,y) = \max(|y-u| - \varepsilon), \varepsilon > 0$$

tau-quantile

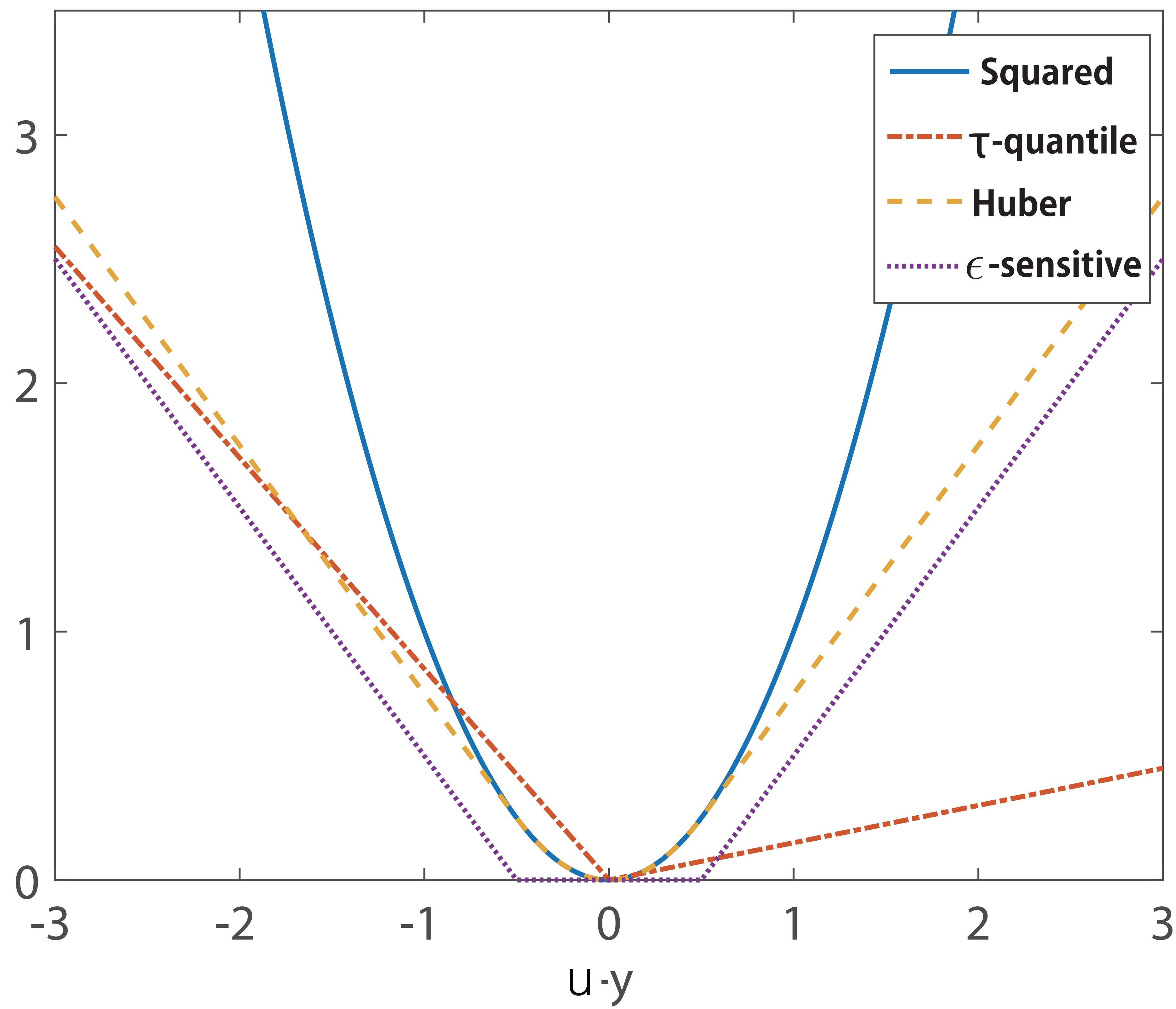
eps-sensitive

$$l_\delta(u,y) = \begin{cases} \frac{1}{2}(y-u)^2 & \text{for } |y-u| \le \delta, \\ \delta\,|y-u| - \frac{1}{2}\delta^2 & \text{otherwise.} \end{cases}$$

huber

u

convex loss functions for regression



convex loss functions for classification

- Label is a binary, prediction is a number

$$l(u, y) = \log(1 + \exp(-yu)),$$ logistic

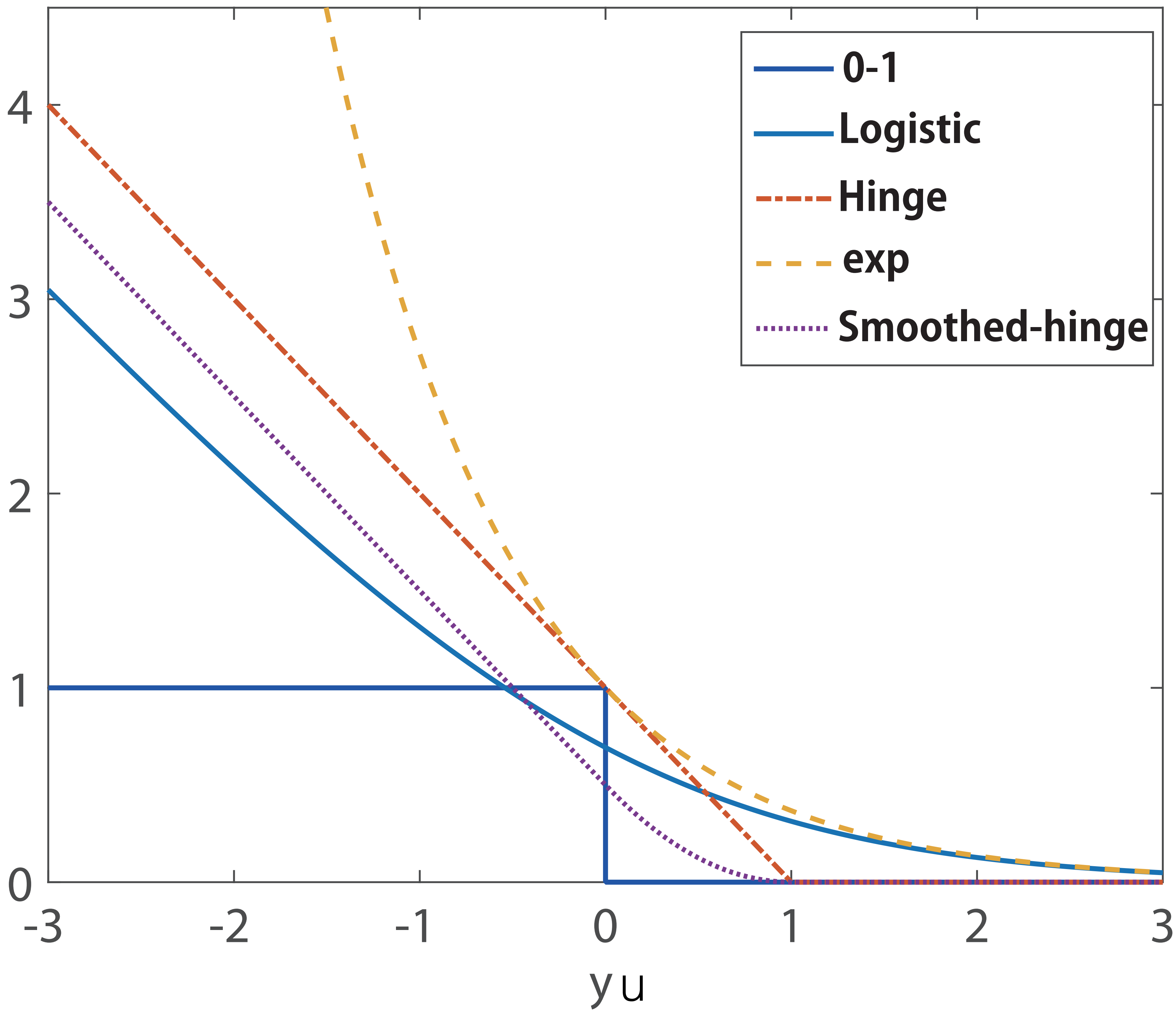
$$l(u, y) = |1 - yu|_+ = \max(1 - yu, 0),$$ hinge

$$l(u, y) = \exp(-yu),$$ exponential

$$l(u, y) = \begin{cases} 0, & yu \geq 1, \\ \frac{1}{2} - yu, & yu < 0, \\ \frac{1}{2}(1 - yu)^2, & \text{otherwise.} \end{cases}$$ smoothed hinge

u

convex loss functions for regression



convex regularizers

$$\psi(\theta) = \|\theta\|_2^2 = \theta^T\theta,$$ ridge

$$\psi(\theta) = \|\theta\|_1 = \sum_i |\theta_i|,$$ L-1

$$\psi(\theta) = a\|\theta\|_1 + b\|\theta\|_2^2,$$ elastic net

$$\psi(\theta) = \|\theta\|_{\mathrm{tr}} = \sum_i^{\min(q,r)} \sigma_j(\theta)$$

trace norm (for matrices)

convex loss functions for regression