Optimization for Data Science Lecture 07: Total Variation Image Denoising

Kimon Fountoulakis

School of Computer Science
University of Waterloo

01/10/2019

Laplacian Denoising

$$\text{minimize} \frac{\lambda}{2}\|Dx\|_2^2 + \frac{1}{2}\|x - z_{noisy}\|_2^2$$

- User defined parameter that controls the balance between regularization and fitting to the noisy image.

noisy image

$\lambda = 3$

$\lambda = 10$

Issues with Laplacian Denoising

- The output is “smoother” than the input, but it looks blurry.

noisy image

$\lambda = 3$

$\lambda = 10$

Outline

- Total-variation denoising

Laplacian denoising

Total-variation denoising

Outline

- Iteration complexity of gradient descent for non-smooth functions.
- Smoothing non-smooth functions

Absolute value of complex numbers

$$|x + i \cdot y| = \left(x^2 + y^2\right)^{1/2}$$

Real part **Imaginary part**

L1-norm for vectors of complex numbers

$$\|x\|_1 = \sum_{i=1}^{n} |x_i| = \sum_{i=1}^{n} \left(Re(x_i)^2 + Im(x_i)^2 \right)^{1/2}$$

$$x_i = Re(x_i) + i \cdot Im(x_i)$$

Total variation norm

$$\|x\|_{TV} = \|Dx\|_1$$

- where

$$D = D_h + iD_v$$

Complex matrix

Horizontal differences Operator

Vertical differences Operator

- And x is the vectorized image, which is a vector of real numbers

Total-variation optimization problem

$$\text{minimize } \lambda \|Dx\|_1 + \frac{1}{2}\|x - z_{noisy}\|_2^2$$

- The L1-norm is a major difference compared to Laplacian denoising for which we used the L2-norm to measure differences among pixels

- In particular, the effects of the L1-norm are visible both in the output of the denoising problem as well as the performance of gradient descent

Parameter tuning

$$\text{minimize} \lambda \|Dx\|_1 + \frac{1}{2}\|x - z_{noisy}\|_2^2$$

- User defined parameter that controls the balance between regularization and fitting to the noisy image.

$\lambda = 5$

$\lambda = 25$

$\lambda = 50$

Why L1-norm? What is wrong with the Euclidean norm?

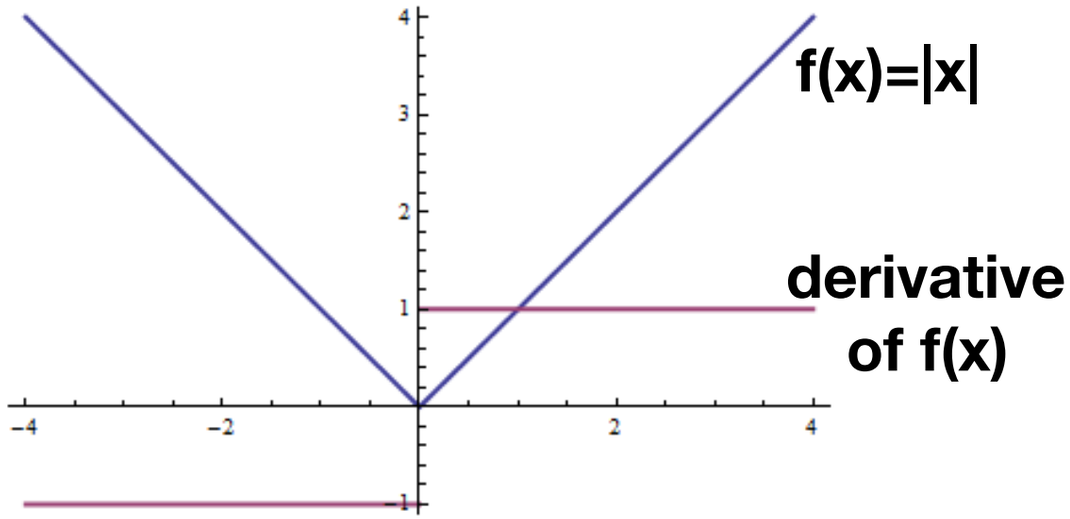
- Less blurry output. It preserves the edges of segments in the image better.

Laplacian denoising

Total variation denoising

Non-differentiability

- Unfortunately, the absolute value function is non-smooth at $x = 0$.



- Non-smoothness of the absolute value makes the L1-norm in the objective function non-differentiable.

- Therefore, direct application of gradient descent is not possible. We will have to make a few modifications to the problem.

Can we ignore non-differentiability?

- The absolute value function $|x|$ is non-smooth at $x = 0$.
- Since this is only one point out of infinitely many, maybe we could pretend it's not a problem?

No, this is not a good idea!!!

Can we ignore non-differentiability?

- The L1-norm is a **sparsity inducing** norm.

- Let x^* denote the minimizer of the TV-denoising problem

$$\text{minimize } \lambda \|Dx\|_1 + \frac{1}{2}\|x - z_{noisy}\|_2^2$$

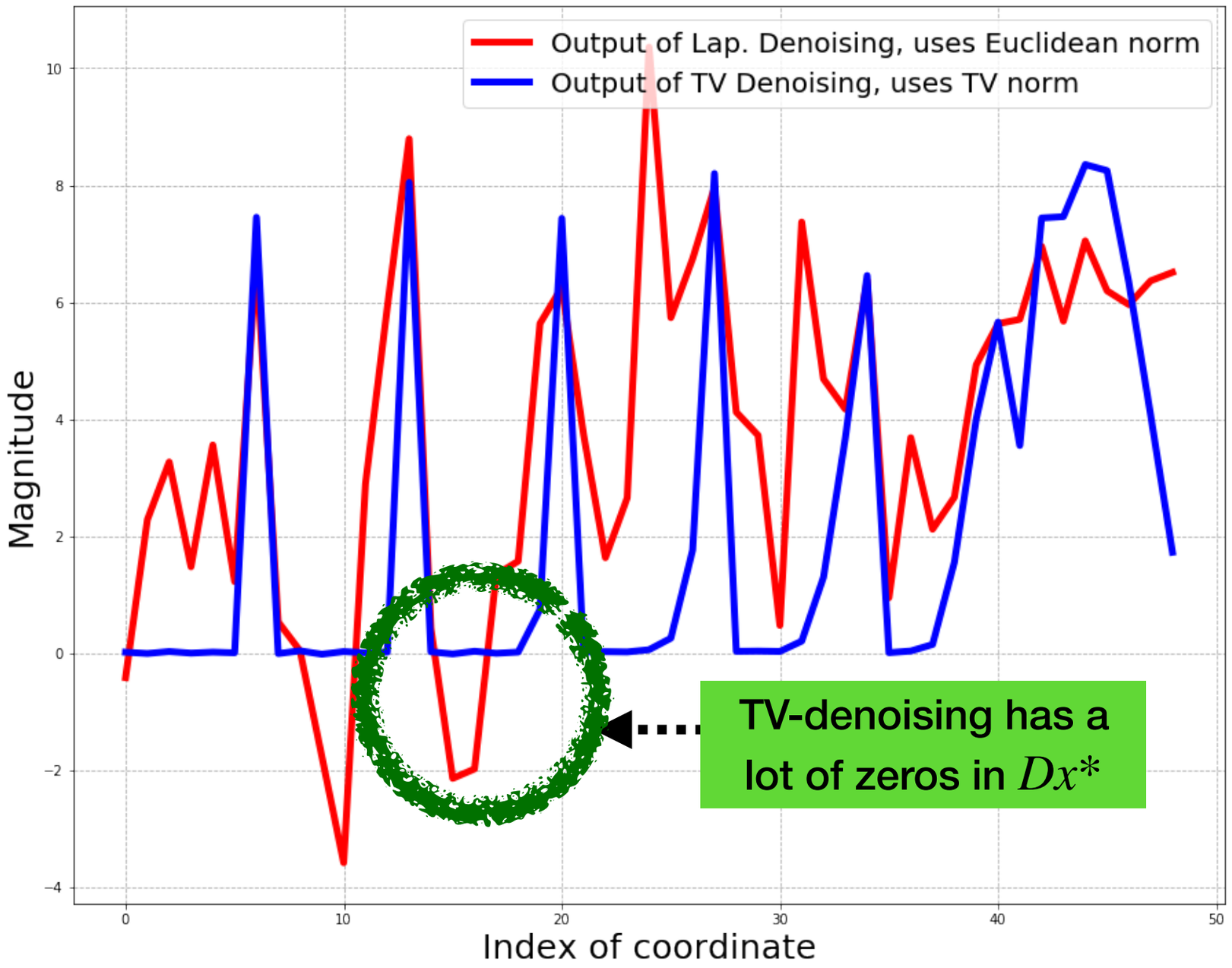
- then, this means that the vector Dx^* will have a many components that are zero.

Can we ignore non-differentiability?

- Since at optimality we have that Dx^* has a lot of zeros, then as gradient descent converges, i.e., $x_k \to x^*$, then Dx_k will have a lot of zeros as well.

- Therefore, non-differentiability of $\|Dx\|_1$, will occur during execution of gradient descent as $x_k \to x^*$.

Can we ignore non-differentiability?



- Let x^* be the minimizer of total-variation denoising or Laplacian denoising.
- The blue line corresponds to total-variation denoising and the red line corresponds to Laplacian denoising.
- The x-axis is the index of coordinates of Dx^* and $D\tilde{x}$, and the y-axis is the magnitude of the coordinates.

Smoothing

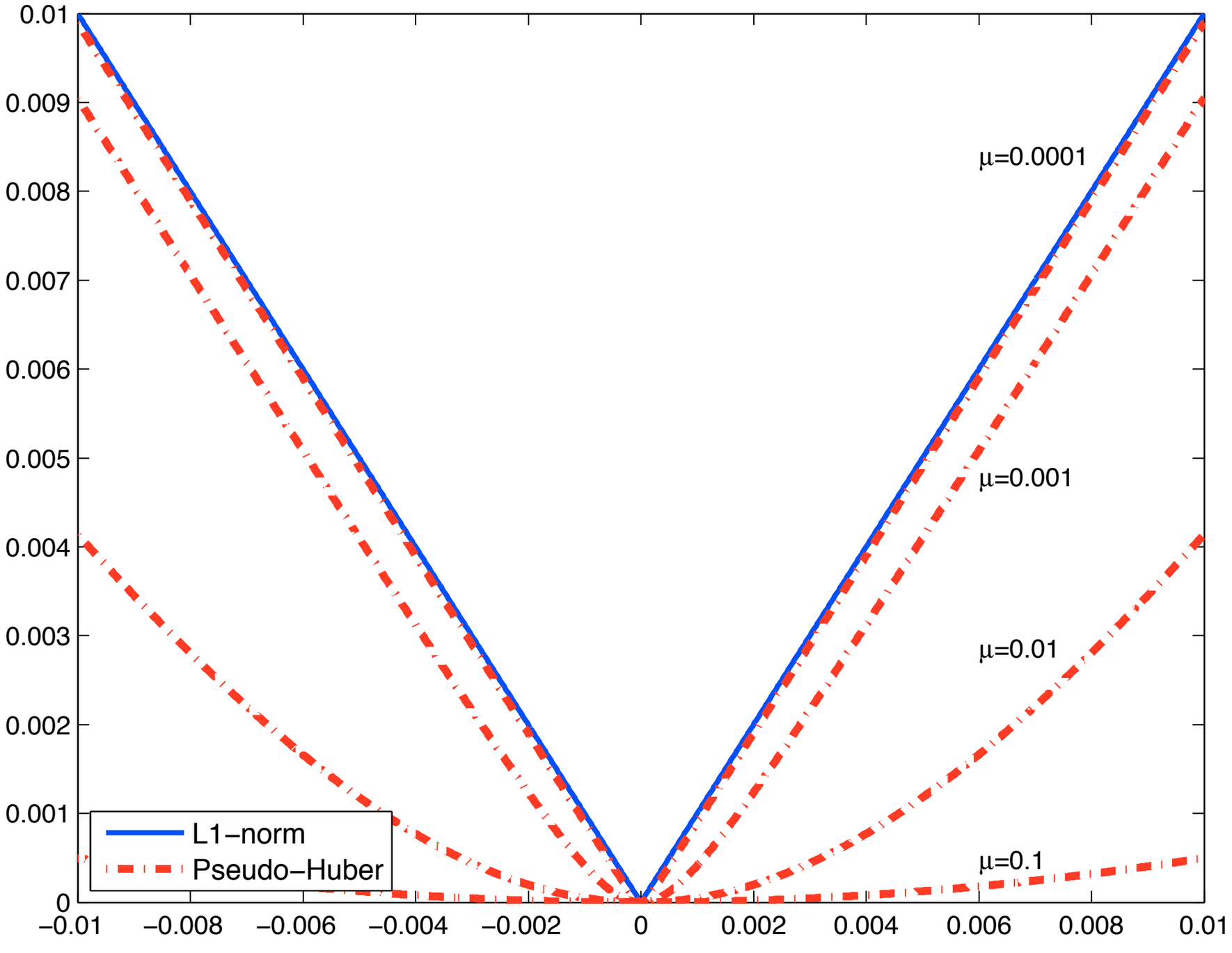
- Replace the L1-norm with the pseudo-Huber function

$$\psi_\mu(x) = \sum_{i=1}^{n} \left((\mu^2 + |x_i|^2)^{1/2} - \mu \right)$$

- The pseudo-Huber function approximates the L1-norm and it is differentiable.

- where μ is a user-defined parameter that controls the quality of the approximation. The smaller μ is the better the approximation.

Smoothing


0.01
0.009
0.008
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0
−0.01
−0.008
−0.006
−0.004
−0.002
0
0.002
0.004
0.006
0.008
0.01
μ=0.0001
μ=0.001
μ=0.01
μ=0.1
L1−norm
Pseudo−Huber

Smoothed total variation

- The smoothed optimization problem is

$$\text{minimize } \lambda\psi_\mu(Dx) + \frac{1}{2}\|x - z_{noisy}\|_2^2$$

- This problem is differentiable for any x . This means that we can apply gradient descent to solve it.

Trade-off

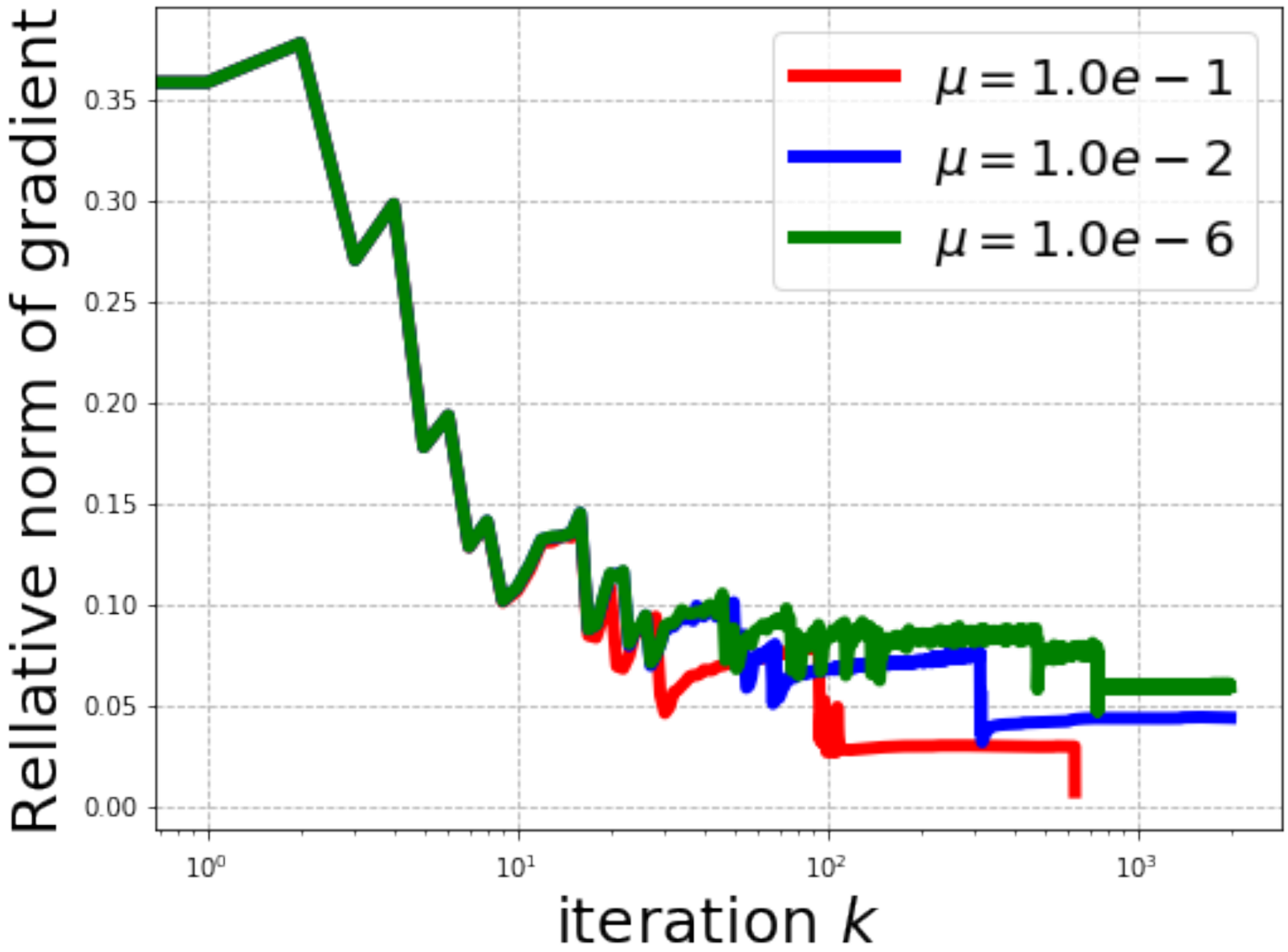
- **Parameter μ affects the Lipschitz constant of the gradient of the smoothed problem and accuracy (how similar the smoothed problem is compared to the original problem).**

- The smaller μ is the larger the Lipschitz constant of the smoothed problem, the more iterations gradient descent requires to converge.

- The smaller μ is the more similar the smoothed problem is the original problem.

- **Summary: small μ results in a lot of iterations and high accuracy, large μ results in small number of iterations but low accuracy.**

Performance of gradient descent w.r.t the pseudo-Huber parameter



- Smaller μ makes for a better approximation.
- But it also makes convergence of gradient descent slower.
- If μ is very small then the method might stagnate, see green line.

Lipschitz constant of the pseudo-Huber function

- The pseudo-Huber function is

$$\psi_\mu(x) = \sum_{i=1}^{n} \left((\mu^2 + |x_i|^2)^{1/2} - \mu \right)$$

- The gradient $\nabla \psi_\mu(x)$ is Lipschitz continuous with constant :

$$L_\mu := \frac{1}{\mu}$$

- which clearly shows that the smaller μ is the worse the Lipschitz constant.

How do the L1-norm and pseudo-Huber functions compare?

- We can lower and upper bound the pseudo-Huber function:

$$\|x\|_1 - \frac{n\mu}{2} \leq \psi_\mu(x) \leq \|x\|_1$$

- The smaller μ is the closer the pseudo-Huber function is to the L1-norm.

- $\psi_\mu(x)$ is a uniform smooth approximation of $\|x\|_1$.

How do the original and the smoothed TV denoising problems compare?

- Let $f(x) := \lambda \|Dx\|_1 + \frac{1}{2}\|x - z_{noisy}\|_2^2$ and

  $f_\mu(x) := \lambda \psi_\mu(Dx) + \frac{1}{2}\|x - z_{noisy}\|_2^2$

- Then we have that

$$f(x) - \frac{n\mu}{2} \leq f_\mu(x) \leq f(x)$$

Iteration complexity

- The smoothed denoising problem is strongly convex, differentiable and the Lipschitz constant of its gradient is $L = 1/\mu + 1.$

- Using the iteration complexity result of gradient descent for strongly convex functions we have that gradient descent requires:

$$\mathcal{O}\left(\frac{1}{\mu}\frac{1}{\delta}\log\frac{1}{\epsilon_\mu}\right)$$

- where δ is the strong convexity parameter of the smoothed TV denoising problem, and $\epsilon_\mu > 0$ is the termination tolerance parameter for the same problem.

Iteration complexity

- Using the iteration complexity result of gradient descent for strongly convex functions we have that gradient descent requires:

$$\mathcal{O}\left(\frac{1}{\mu}\frac{1}{\delta}\log\frac{1}{\epsilon_\mu}\right)$$

- The smaller μ becomes the more iterations gradient descent will need to converge to a solution that satisfies $f_\mu(x_k) - f_\mu^* \leq \epsilon_\mu$, where $f_\mu(x)$ is the smoothed TV denoising objective function.

Iteration complexity

- After $\mathcal{O}((\mu\delta)^{-1}\log(\epsilon_\mu)^{-1})$ iterations we are ϵ_μ close to the optimal solution of the smoothed problem, i.e., $f_\mu(x_k) - f_\mu^* \leq \epsilon_\mu$.

- What about the original problem? Do we also have $f(x_k) - f^* \leq \epsilon_\mu$? **No!**

- From $f(x) - n\mu/2 \leq f_\mu(x) \leq f(x)$ we have that:

$$f(x) - f^* \leq f_\mu(x) - f_\mu^* + \frac{n\mu}{2}$$

Iteration complexity

- Using $f(x) - f^* \leq f_\mu(x) - f_\mu^* + \frac{n\mu}{2}$, we get that to guarantee $f(x_k) - f^* \leq \epsilon$, we have to set the tolerance of gradient descent for the smooth problem as:

$$\epsilon_\mu := \epsilon - \frac{n\mu}{2}$$

- which gives the following iteration complexity result for gradient descent:

$$\mathcal{O}\left(\frac{1}{\mu}\frac{1}{\delta}\log\left(\epsilon - \frac{n\mu}{2}\right)^{-1}\right)$$

Iteration complexity

- Let's set $\mu := \epsilon/n$. Then we get that gradient descent requires

$$\mathcal{O}\left(\frac{n}{\epsilon}\frac{1}{\delta}\log\frac{1}{\epsilon}\right)$$

- iterations to converge to a solution that satisfies: $f(x_k) - f^* \leq \epsilon$.

- The number of iterations depend on the dimensions n and scale w.r.t $1/\epsilon$. So we are back to convex-type iteration complexity. We can improve this, but we need a method called accelerated gradient descent (next lectures).

Iteration complexity

- If the function was convex instead of strongly-convex, then the iteration complexity is

$$\mathcal{O}\left(\frac{n}{\epsilon^2}\right)$$

- iterations to converge to a solution that satisfies: $f(x_k) - f^* \leq \epsilon$.

- which is **terribly slow**, but can be also improved using accelerated gradient descent.

Iteration complexity of gradient descent for TV denoising

	Non-convex	**Convex**	$\delta-$**strongly-convex**
Smooth $f(x)$	$\mathcal{O}\left(\frac{L}{\epsilon}\right)$	$\mathcal{O}\left(\frac{L}{\epsilon}\right)$	$\mathcal{O}\left(\frac{L}{\delta}\log\frac{1}{\epsilon}\right)$
Non-smooth $f(x)$	$\mathcal{O}\left(\frac{n}{\epsilon^2}\right)$	$\mathcal{O}\left(\frac{n}{\epsilon^2}\right)$	$\mathcal{O}\left(\frac{n}{\delta}\frac{1}{\epsilon}\log\frac{1}{\epsilon}\right)$

Short outline

- We will learn how to smooth non-smooth functions
- We will mainly work on the example of L1-norm
- **Our goal**, in subsequent lectures, is to improve the iteration complexity of gradient-based methods for non-smooth functions using the so called accelerated method. Keep this goal in mind whenever you are questioning yourself why do you need all these tools.

Smoothing of non-smooth functions

- We will think a function f in terms of its supporting hyperplanes (linear functions that support f below)

- We will use the new representation to define a new function that is called **convex conjugate**.

- We will use the convex conjugate to smooth our non-smooth convex function. Keep this goal in mind whenever you are questioning yourself why do you need all these details.

Some definitions

- The **argmin** operator returns the points at which a function is minimized. For example:

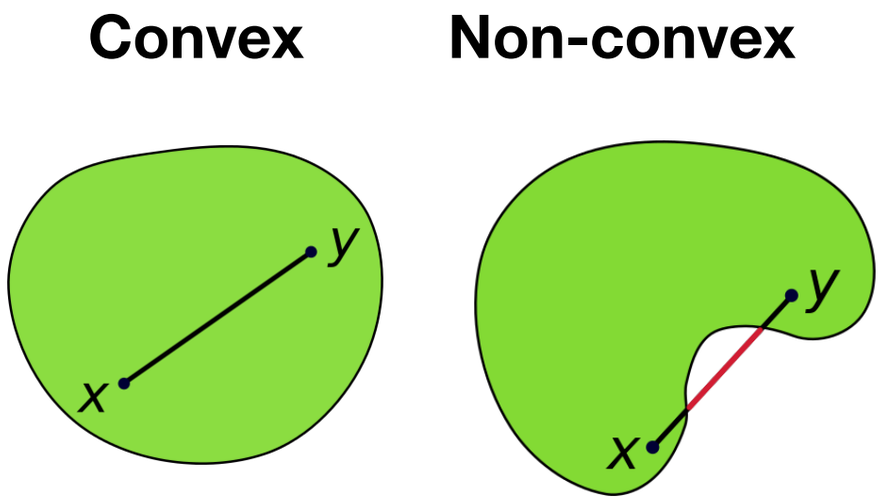
$$x^* := \text{argmin}_x \ f(x)$$

- where x^* represents a minimizer of f.

- We can also define **argmax**, which returns the points at which a function is maximized.

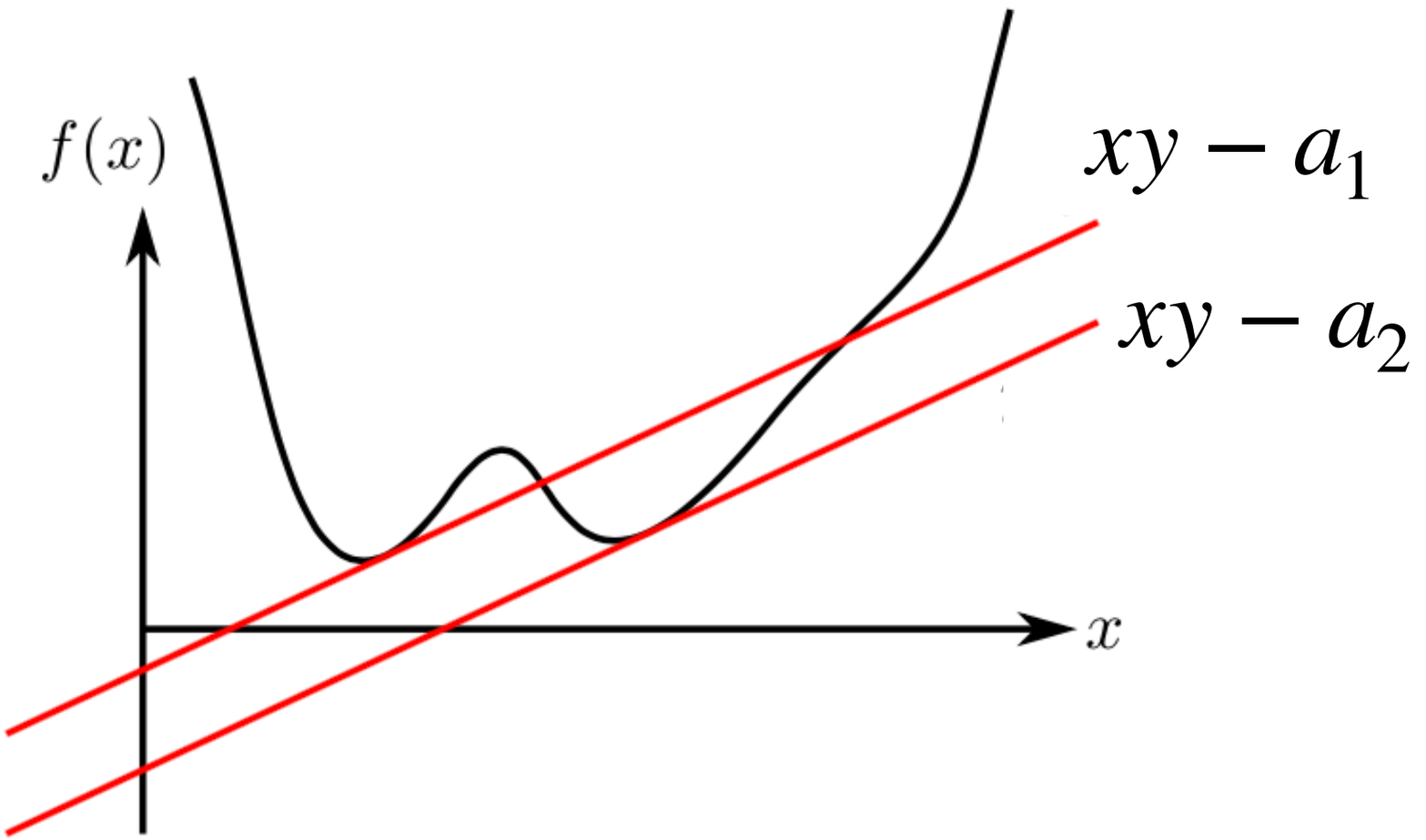
Some definitions

- A function has bounded domain if its domain is a bounded set, i.e., the absolute value of every element in the domain is bounded by a constant.
- A set is closed iff it contains all of its limits points.
- A subset of $\mathbb{R}^n$ is compact if it is closed and bounded.
- A set C is convex if $x, y \in C$ and $tx + (1-t)y \in C \ \forall t \in [0,1]$



Represent a function in terms of linear functions

- Given a slope y, I can find many lines that cross the function. I am interested in the line that "touches" or "supports" f.



Represent a function in terms of linear functions

- How do I find the hyperplane (multi-dimensional line) that "supports" f?
- When a hyperplane with slope y crosses f we have:

$$y^T x + a = f(x)$$

- Let's re-arrange

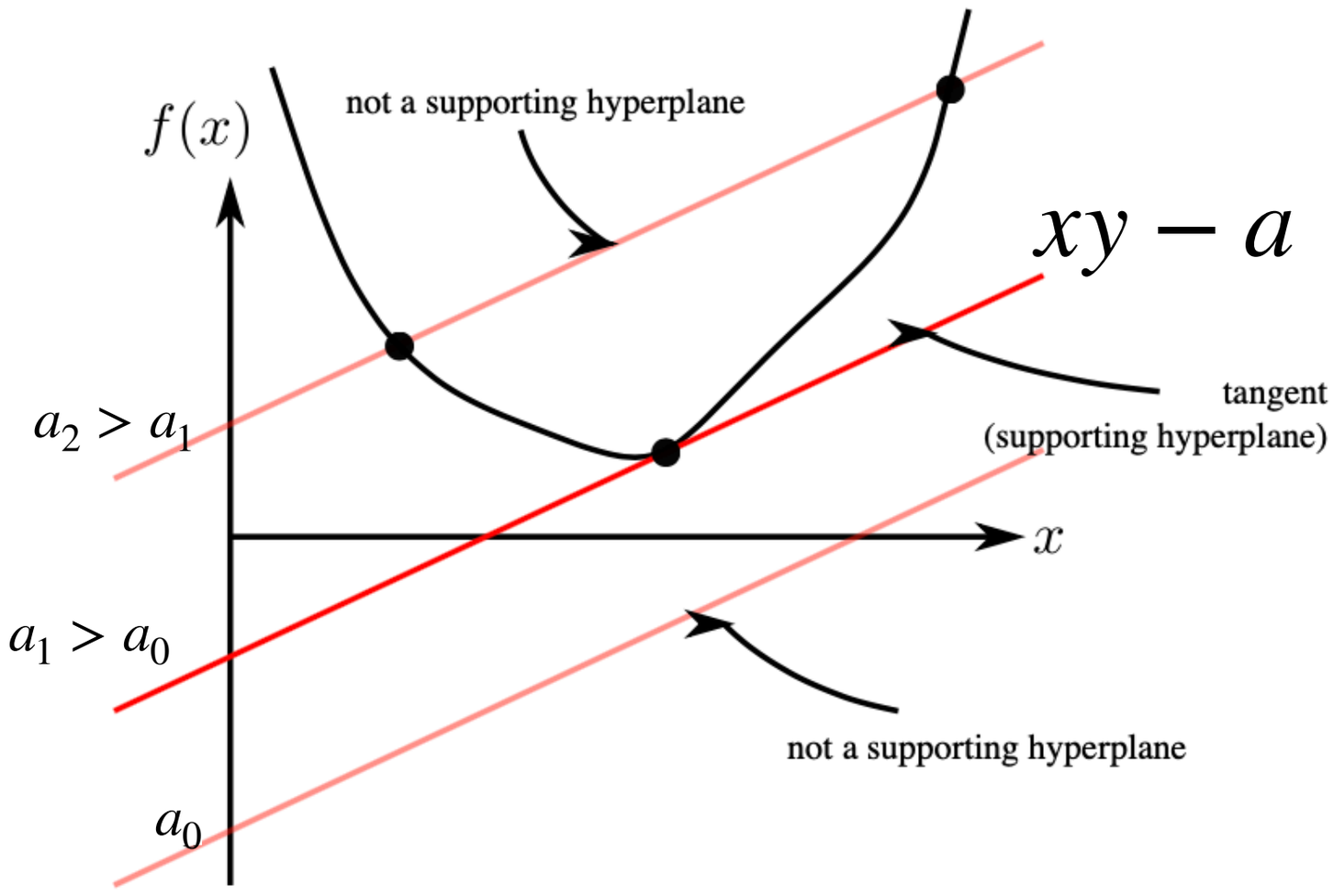
$$a = f(x) - y^T x$$

- The hyperplane that "supports" f is given for the minimum a:

$$a^* = \min_{x \in \mathbb{R}^n} \; f(x) - y^T x$$

Represent a function in terms of linear functions

- Example



Represent a function in terms of linear functions

- The hyperplane that "supports" f is given for the minimum a:

$$\begin{aligned} a^*(y) &= \min_{x \in \mathbb{R}^n} \; f(x) - y^T x \\ &= \min_{x \in \mathbb{R}^n} \; -(y^T x - f(x)) \\ &= -\max_{x \in \mathbb{R}^n} \; y^T x - f(x) \end{aligned}$$

- In the last equality we used $\min_x g(x) = -\max_x -g(x)$.
- Let's define $f^*(y) := -a^*(y)$ to get

$$f^*(y) = \max_{x \in \mathbb{R}^n} \; x^T y - f(x)$$

Convex conjugate

- where $f^*(y)$ is called the **convex conjugate** of f.

$$f^*(y) := \max_{x \in \mathbb{R}^n} \; y^T x - f(x)$$

- We are interested in convex conjugate functions $f^*(y)$ where $f^*(y)$ has **closed and bounded domain**. We will denote the domain of f^* by dom f^*

Convex conjugate

$$f^*(y) := \max_{x \in \mathbb{R}^n} \; y^T x - f(x)$$

- $f^*(y)$ is always convex because it is the point-wise maximum of affine functions.
- $f^{**}(x) := (f^*(y))^* \leq f(x)$
- If f is convex, then $f^{**}(x) = f(x)$.

Conjugate of the convex conjugate

- The conjugate of the convex conjugate is defined as

$$f^{**}(x) := \max_{y \in \mathrm{dom}\ f^*} \quad x^T y - f^*(y)$$

- If f is convex then we have $f^{**}(x) = f(x)$. Thus, we get

$$\boxed{f(x) := \max_{y \in \mathrm{dom}\ f^*} \quad x^T y - f^*(y)}$$

Represent a function in terms of linear functions

- Why would I need the convex conjugate $f^*(x)$?

- Sometimes it is easier to work with f^* than f (depending on the problem). It also provides a lower bound $f^{**}(x) \leq f(x)$ ($f^{**}(x) = f(x)$ if f is convex).

- Understanding properties of f^* allows us to smooth f (this lecture).

Example: conjugate of the L1-norm

$$f^*(y) = \begin{cases} 0 & \text{if } \|y\|_\infty \leq 1 \\ \infty & \text{otherwise} \end{cases}$$

- where $\|y\|_\infty := \max_{1 \leq i \leq n} |y_i|$.

- $f^*(y)$ is the indicator function for the set $\{y \in \mathbb{R}^n \mid \|y\|_\infty \leq 1\}$

The gradient and Lipschitz continuity w.r.t the conjugate

- Suppose the convex conjugate f^* of f is continuous with compact domain and strongly convex with strong convexity parameter δ, then function f is differentiable with gradient:

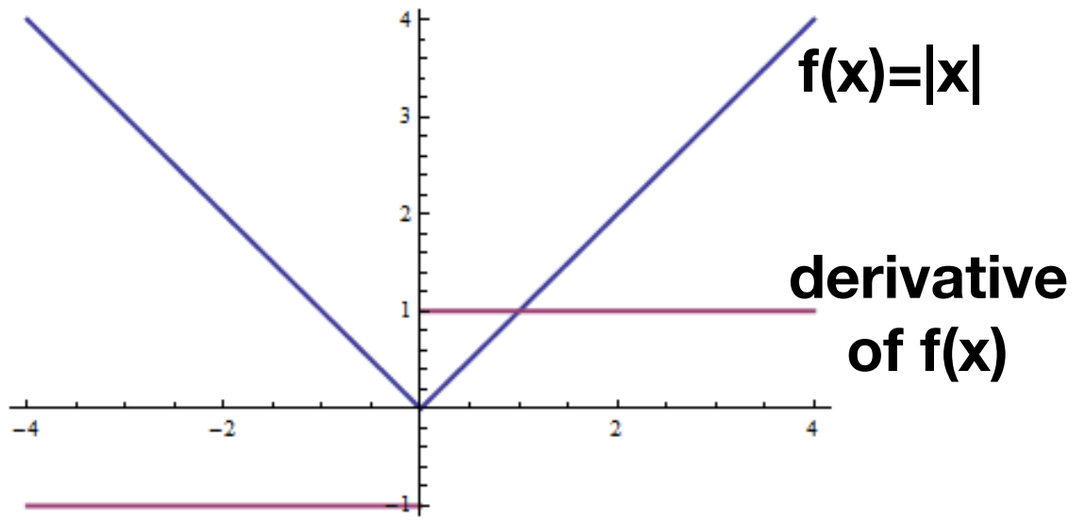
$$\nabla f(x) = \text{argmax}_{y \in \text{dom } f^*} x^T y - f^*(y)$$

- The gradient of $f(x)$ is Lipschitz continuous with $L = 1/\delta$:

$$\|\nabla f(z) - \nabla f(s)\|_2 \leq \frac{1}{\delta}\|z - s\|_2$$

Understanding non-differentiability through the conjugate

- One way to see that $|x|$ is non-smooth is by simply attempting to compute the derivative. We observe that at zero it is non-differentiable.



Understanding non-differentiability through the conjugate

- Having smoothing in mind, a better way to understand why the L1-norm is non-differentiable is by using the convex conjugate.

- We mentioned earlier that the gradient of a function f is equivalent to:

$$\nabla f(x) = \text{argmax}_{y \in \text{dom } f^*} x^T y - f^*(y)$$

- The gradient is unique if the convex conjugate f^* is strongly convex. That's because this guarantees that the RHS has a unique maximizer.

Understanding non-differentiability through the conjugate

- What if f^* is simply convex? Then the RHS might have multiple minimizers!! This makes the gradient non-unique!

Understanding non-differentiability through the conjugate

- Let's look into the L1-norm example. Using the convex conjugate of the L1-norm:

$$f^*(y) = \begin{cases} 0 & \text{if } \|y\|_\infty \leq 1 \\ \infty & \text{otherwise} \end{cases}$$

- The gradient of the L1-norm should be equal to

$$\text{argmax}_{\|y\|_\infty \leq 1} \; x^T y$$

- But for $x = 0$ we have that

$$(\text{any } y \text{ that satisfies } \|y\|_\infty \leq 1) = \text{argmax}_{\|y\|_\infty \leq 1} \; 0$$

- Thus at $x = 0$ the gradient of the L1-norm is not uniquely defined!

Smoothing the L1-norm

- We will smooth the absolute value function first, smoothing of the L1-norm is obtained by applying the same result n times, since $\|x\|_1 = \sum_{i=1}^{n} |x_i|$.

- The convex conjugate of the absolute value function is

$$f^*(y) = \begin{cases} 0 & \text{if } |y| \leq 1 \\ \infty & \text{otherwise} \end{cases}$$

Smoothing the L1-norm

- Normally we could compute the absolute value function by taking the conjugate of the conjugate:

$$|x| = \max_{y \in \mathbb{R}} \; xy - f^*(y) = \max_{|y| \leq 1} \; xy$$

- But now we want to make sure that the RHS has a unique maximizer for any x. Let's add a **strongly convex** function $0.5\delta y^2$ in the computation of the conjugate of the conjugate

$$\begin{aligned} h(x) &= \max_{y \in \mathbb{R}} \; xy - f^*(y) - \frac{\delta}{2} y^2 \\ &= \max_{|y| \leq 1} \; xy - \frac{\delta}{2} y^2 \quad (\text{using the definition of } f^* \text{ for } f(x) = |x|) \\ &= \begin{cases} \frac{x^2}{2\delta} & \text{if } |x| \leq \delta \\ |x| - \frac{\delta}{2} & \text{if } |x| > \delta \end{cases} \end{aligned}$$

Smoothing the L1-norm

- Function

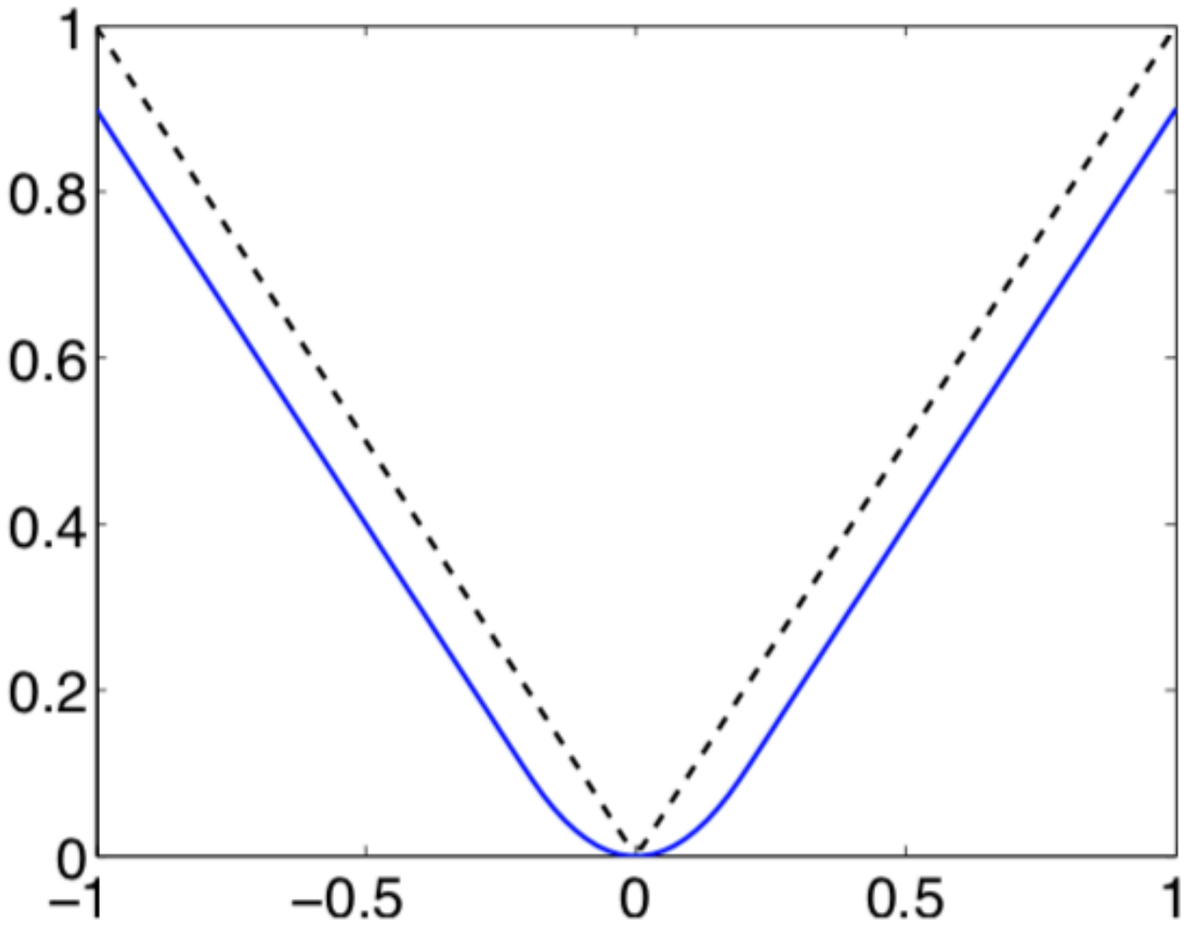
$$h(x) = \begin{cases} \frac{x^2}{2\delta} & \text{if } |x| \leq \delta \\ |x| - \frac{\delta}{2} & \text{if } |x| > \delta \end{cases}$$

- Is an approximation to $f(x) = |x|$ and it is differentiable with derivative

$$h'(x) = \begin{cases} \frac{x}{\delta} & \text{if } |x| \leq \delta \\ \text{sign}(x) & \text{if } |x| > \delta \end{cases}$$

Smoothing the L1-norm

$$\delta = 0.2$$



Black dashed line is the l1-norm

Blue solid line is the smooth approximation

Smoothing non-smooth functions

- To smooth a non-smooth function f we will need the so-called proximity functions.

- $d(y)$ is a proximity function for a closed convex set $C \subseteq$ dom d, if it satisfies:

  - $d(y)$ is continuous and strongly convex.

Smoothing non-smooth functions

- We will assume that the proximity function is normalized:

$$\min_{y \in C} \; d(y) = 0$$

- and it has strong-convexity constant equal to 1.

Examples of proximity functions

- $d(y) = \frac{1}{2}\|y - y_c\|_2^2$, where $y_c := \text{argmin}_{y \in C}\ d(y)$

- $d(y) = \frac{1}{2}\sum_{i=1}^{n} w_i (y_i - [y_c]_i)^2$, where $w_i \geq 0$ and , where $y_c := \text{argmin}_{y \in C}\ d(y)$.

Smoothing non-smooth functions

- Let's define the function

$$f_\mu(x) := \max_{y \in \mathbb{R}^n} \; x^T y - f^*(y) - \mu d(y)$$

- It can be shown that if $C :=$ dom $f^* \subseteq$ dom d.

$$f(x) - \mu D \leq f_\mu(x) \leq f(x)$$

- where $D := \max_{y \in C} \; d(y)$, which is bounded $D < \infty$.

Smoothing non-smooth functions

- The gradient of $f_\mu(x)$ is Lipschitz continuous

$$\|\nabla f_\mu(z) - \nabla f_\mu(s)\|_2 \leq \frac{1}{\mu}\|z - s\|_2$$

Example: pseudo-huber function

- $d(y) = \sum_{i=1}^{n} 1 - \sqrt{1 - y_i^2}$, with domain $|y_i| \leq 1$.

- Using the above proximity function to perturb f gives the pseudo-Huber function:

$$\psi_\mu(x) = \sum_{i=1}^{n} \left((\mu^2 + |x_i|^2)^{1/2} - \mu \right)$$