Lecture 8
Basics of Optimization

- Gradient Descent, Stochastic Gradient Descent, Newton's Method

- Stochastic Gradient Descent

- Newton's Method

- Example: Solving maximum likelihood estimator for CT imaging

Contents

Continuous Optimization.

input, regressor

$(y_i, \; x_{i1}, \; x_{i2}, \; \cdots, \; x_{ip})$

response

Multiple linear regression

p: dimension

n: sample size.

- set-up: p variables, n observations:

$$y_i = \beta_0 + \beta_1 x_{i1} + \ldots \beta_p x_{ip} + \epsilon_i, \quad i = 1, \ldots, n$$

intercept. i.i.d.

coefficients $\underbrace{\beta = [\beta_0, \beta_1, \cdots, \beta_p]^\mathsf{T}}_{\text{column vector.}}$

$$\min_\beta \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_{i1} + \ldots \beta_p x_{ip}))^2$$

least square

MLE.

- matrix-vector form

$y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \in R^n$ $\quad \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}$

$$y = X\beta + \epsilon, \quad X = \begin{bmatrix} 1 & x_{11} & \cdots & x_{p1} \\ \vdots & \vdots & & \vdots \\ 1 & x_{1n} & \cdots & x_{pn} \end{bmatrix} \in \mathbb{R}^{n \times (p+1)}$$

- parameter estimation

$$\min_\beta \|y - X\beta\|_2^2$$

$\|z\|_2$: ℓ_2 norm.

$\|z\|_2 = \sqrt{z^\mathsf{T} z}$

Simple linear regression

- Linear regression model

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i, \quad i = 1, 2, \ldots, n$$

- To estimate (β_0, β_1), we find values that minimize the sum-of-squares error

intercept slope

$$\min_{\beta_0,\beta_1} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2 \triangleq J(\beta_0, \beta_1)$$

$$\hat{\beta}_1 = S_{xy}/S_{xx}$$

$$\frac{\partial J}{\partial \beta_0} = 0 \qquad \frac{\partial J}{\partial \beta_1} = 0$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

cross covariance

$$S_{xy} = \sum_{i=1}^{n} (y_i - \bar{y})(x_i - \bar{x}), \; S_{xx} = \sum_{i=1}^{n} (x_i - \bar{x})^2$$

sample covariance,

$$\frac{\partial J}{\partial \beta_0} = \sum_i \frac{\partial}{\partial \beta_0} (\qquad)^2$$

$$= \sum_i 2(y_i - \beta_0 - \beta_1 x_i)(-1) = 0$$

$$\frac{\partial J}{\partial \beta_1} = \sum_i 2(y_i - \beta_0 - \beta_1 x_i)(-x_i) = 0$$

$$\Rightarrow \begin{cases} n\beta_0 + \beta_1 \left(\sum_i x_i\right) = \sum_i y_i \\ \left(\sum x_i\right)\beta_0 + \beta_1 \left(\sum_i x_i^2\right) = \sum_i x_i y_i . \end{cases}$$

Solving multiple linear regression

$\frac{\partial x^\top a}{\partial x} = a \qquad \frac{\partial a^\top x}{\partial x} = a.$

$y^\top$

$\|z\|_2^2 = z^\top z.$

$\|y - X\beta\|_2^2$

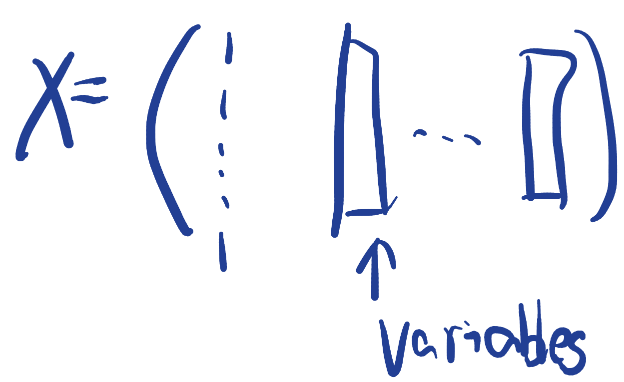
$= y^\top y - \beta^\top X^\top y - y^\top X\beta + \beta^\top X^\top X\beta$

$= y^\top y - 2\beta^\top X^\top y + \beta^\top X^\top X\beta,$

$\nabla f(\beta) = -X^\top y - X^\top y + 2X^\top X\beta.$

$$\min_{\beta}\ f(\beta) := \|y - X\beta\|_2^2, \quad X \in \mathbb{R}^{n\times(p+1)}$$

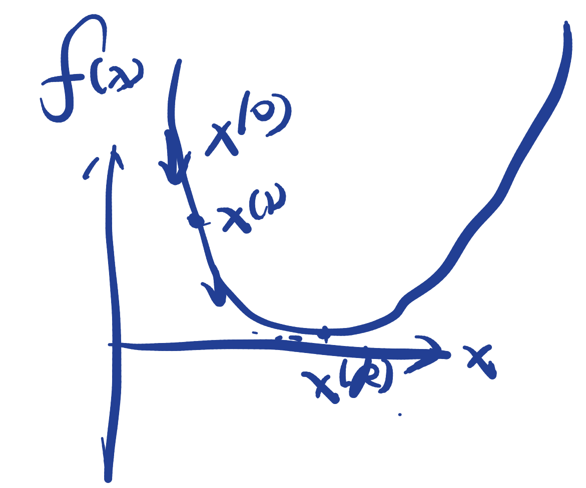
- Gradient $\nabla f(\beta) = 2X^\top(X\beta - y)$
- Exact solution $\hat\beta = (X^\top X)^{-1}X^\top y$
- issue: complexity $\mathcal{O}(p^3)$ $\quad X^\top X \in \mathbb{R}^{(p+1)\times(p+1)}$
- issue: $(X^\top X)^{-1}$ may not be a good idea



If they're correlated (e.g. predict house price)

$X^\top X$ is rank-deficient

Solving optimization problem



- solve optimization problem

$$\min_x f(x)$$

- produce sequence of points $x^{(k)}$, $k = 0, 1, 2, \ldots$ with

initialize $x^{(0)} \to x^{(1)} \to x^{(2)} \to \cdots$

$$f(x^{(k)}) \to p^*$$

approximation algorithm,

- iterative methods for solving optimality condition

$$\nabla f(x^*) = 0$$

First-order method: Gradient decent

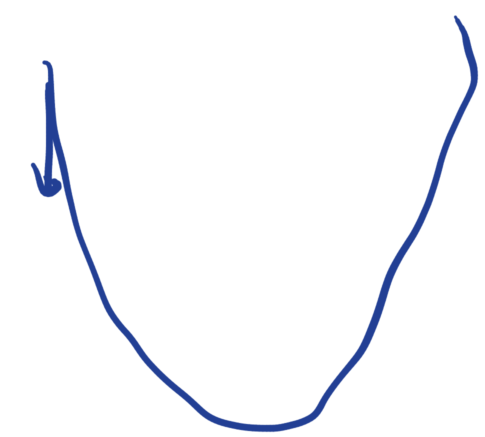
most popular approach

[MLE, training NN]

$$x^{(k+1)} = x^{(k)} - t_k \nabla f(x^{(k)})$$

t_k: step-size for the kth iteration
$\nabla f(x)$: gradient vector

min f(x)



- for **convex** optimization it gives the global optimum under fairly general conditions.

- for **nonconvex** optimization it may achieve a local optimum

Example: solving multiple linear regression

$$\min_{\beta} \; f(\beta) := \|y - X\beta\|_2^2, \quad X \in \mathbb{R}^{n\times(p+1)}$$

- Gradient $\nabla f(\beta) = 2X^\intercal(X\beta - y)$
- Exact solution $\hat{\beta} = (X^\intercal X)^{-1}X^\intercal y$, issue: complexity $\mathcal{O}(p^3)$
- Gradient descent

$$\beta^{(k+1)} = \beta^{(k)} - 2t_k X^\intercal(X\beta^{(k)} - y)$$

no matrix inversion involved

$$= (I - 2t_k X^\intercal X)\beta^{(k)} + 2t_k X^\intercal y$$

complexity $\mathcal{O}(np)$

- Question: does this converges to the desired result?

matrix

$n_1 \times n_2$ $\quad$ $n_2 \times n_3$

complexity $= O(n_1 n_2 n_3)$

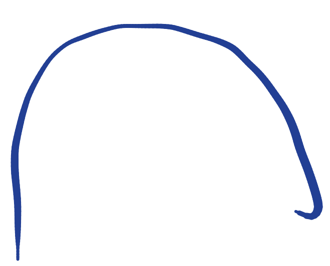
Convex function

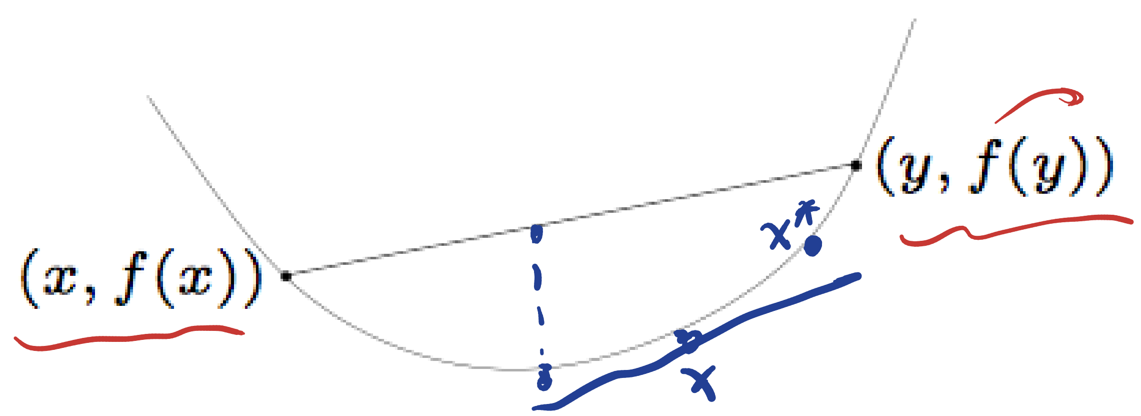
A function f is convex if "holds water"

linear interpolation between (x, f(x)) (y, f(y))

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y)$$

• Concave:





draw a line

min f(x)
↑
convex

⇒ convex optimization

s.t. x∈S

S: convex set

Property (first-order):

$$f(x^*) \geq f(x) + g(x)^T (x^* - x)$$

A easy to use way to check: Univariate $f(x)$ is convex if and only if

$$\frac{\partial^2 f(x)}{\partial x^2} \geq 0$$

Convex function

Multivariate $f(x) : \mathbb{R}^d \to \mathbb{R}$ is convex if and only if the Hessian matrix is positive semi-definite (PSD)

symmetric

$H = H^T$

$$H := H[f(x)] = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \dots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_d} \\ \dots & \dots & \dots \\ \frac{\partial^2 f(x)}{\partial x_1 \partial x_d} & \dots & \frac{\partial^2 f(x)}{\partial x_d^2} \end{bmatrix}_{d \times d}$$

and this matrix H is PSD means either one of the following is true

1. ✓ H can be written as $H = SS^T$ for some matrix S
2. ✓ All eigenvalues of H are non-negative
3. All the principal sub-matrices of H, denoted as H_i, satisfy $\det(H_i) \geq 0$

Example: solving multiple linear regression

- Convex.
- gradient descent.

$$\min_\beta \|y - X\beta\|_2^2, \quad X \in \mathbb{R}^{n\times(p+1)}$$

- $f(\beta) = \|y - X\beta\|_2^2$
- Gradient $\nabla f(\beta) = 2X^\intercal(X\beta - y)$
- Hessian $H[f](\beta) = 2X^\intercal X$ $= (\sqrt{2}X)^T(\sqrt{2}X)$.
  (using basic multivariate calculus)

$H[f](\beta) = \nabla(\nabla f(\beta))$

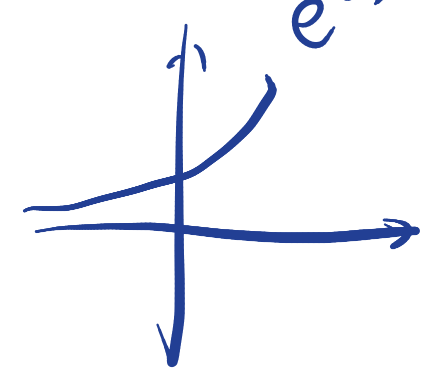
$= 2X^T X$.

Examples

convex functions

- affine: $ax + b$
- exponential e^{ax}
- powers $|x|^\alpha$ for $\cancel{p}\,\alpha \ge 1$

$f(x) = ax+b$ $\quad \frac{\partial^2}{\partial x^2} f(x) = 0$.



$f''(x) = \frac{\partial}{\partial x}(a e^{ax}) = a^2 e^{ax} \ge 0$.

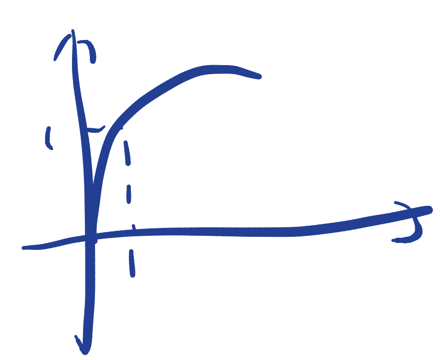
strictly convex.

$\alpha = 2, \ \alpha = 3$.

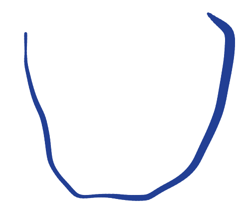
concave:

- affine: $ax + b$
- log: $\log x$ $\quad x > 0$.
- powers x^α for $0 \le \alpha \le 1$

$\alpha = \frac{1}{2}$:



Convergence results



$x^* = \arg\min f(x)$

- **Gradient descent**: for strongly convex f with constant m

$$0 \le f(x^{(k)}) - f(x^*) \le c^k(f(x^{(0)}) - f(x^*))$$

k-th iteration   true solution

$c = 1/2$
$c^2 = 1/4$
$c^3 = 1/8$
$\vdots$

$c \in (0, 1)$ is a constant depends on $x^{(0)}$, step-size, m etc.
Very simple, but converges very slow.
Number of iterations until $f(x) - f(x^*) \le \epsilon$ is $\mathcal{O}(\log(1/\epsilon))$

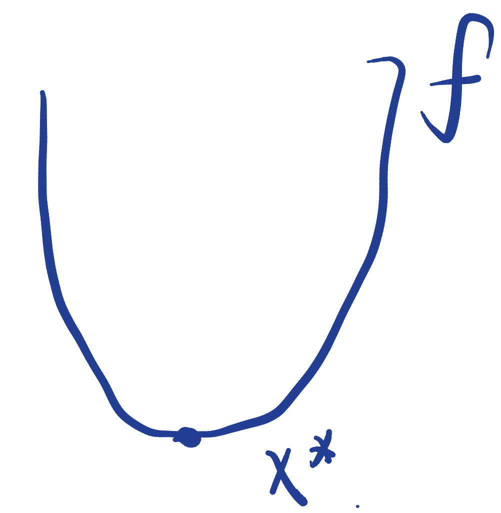
if $c^k(f(x^{(0)}) - f(x^*)) < \varepsilon \Rightarrow k = O(\log \frac{1}{\varepsilon})$

- **Newton's method**: for strongly convex f with constant m
number of iterations until $f(x) - p^* \le \epsilon$ is $\mathcal{O}(\log \log(1/\epsilon))$

$\log(10^{10}) = 10$

$\log \log(10^{10}) = 1$

Convergence proof



key quantity: Euclidean distance to the optimal set

Let x^* be any minimizer of f

$x^{(k+1)} = x^{(k)} - t_k g^{(k)}$.

$\|x-y\|_2^2 = \|x\|_2^2 + \|y\|_2^2 - 2x^T y$.

$$\begin{aligned}
\|x^{(k+1)} - x^*\|_2^2 &= \|x^{(k)} - t_k g^{(k)} - x^*\|_2^2 \\
&= \|x^{(k)} - x^*\|_2^2 - 2t_k g^{(k)T}(x^{(k)} - x^*) + t_k^2 \|g^{(k)}\|_2^2 \\
&\le \|x^{(k)} - x^*\|_2^2 - 2t_k (f(x^{(k)}) - f^*) + t_k^2 \|g^{(k)}\|_2^2
\end{aligned}$$

where $f^* = f(x^*) \ge f(x^{(k)}) + g^{(k)T}(x^* - x^{(k)})$

Basic inequality: for convex differentiable f:

$$f(y) \ge f(x) + \nabla f(x)^T (y - x)$$

apply recursively to get

$$
\begin{aligned}
&\|x^{(k+1)} - x^*\|_2^2 \\
&\le \underbrace{\|x^{(1)} - x^*\|_2^2}_{\le R^2} - 2\sum_{i=1}^{k} t_i(f(x^{(i)}) - f^*) + \sum_{i=1}^{k} t_i^2 \|g^{(i)}\|_2^2 \\
&\le R^2 - 2\sum_{i=1}^{k} t_i(f(x^{(i)}) - f^*) + G^2 \sum_{i=1}^{k} t_i^2
\end{aligned}
$$

$\nabla f(x) = 0.$

Now use

$$f(x^{(i)}) - f^* \ge f_{\text{best}}^{(k)} - f^*$$

we have

$$f_{\text{best}}^{(k)} - f^* \le \frac{R^2 + G^2 \sum_{i=1}^{k} t_i^2}{2\sum_{i=1}^{k} t_i}$$

$\sum_{i=1}^{k} t_i^2 \to$ converge $\to c$

$\sum_{i=1}^{k} t_i \to$ diverge $\to +\infty$

$f_{\text{best}}^{(k)} = \min_{i=1,\ldots,k} f(x^{(i)})$, $\|g\|_2 \le G$ for all gradient g

Strong convexity and implications

- f is **strongly convex** on domain S if there exists an $m > 0$ such that

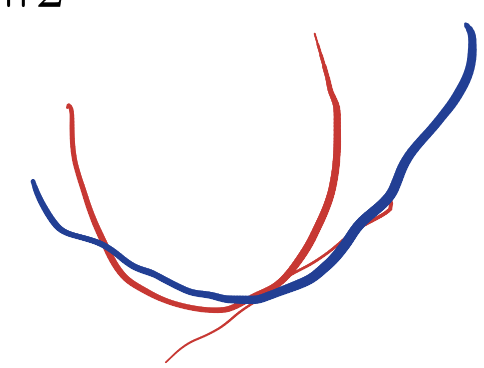
$$H[f(x)] \geq m\underbrace{I}_{\text{identity}}, \quad \text{for all } x \in S.$$

iff $H[f(x)] - mI$ is PSD

- **implications**
  - for $x, y \in S$

$$f(y) \geq f(x) + \nabla f(x)^\intercal (y - x) + \frac{m}{2}\|x - y\|_2^2$$

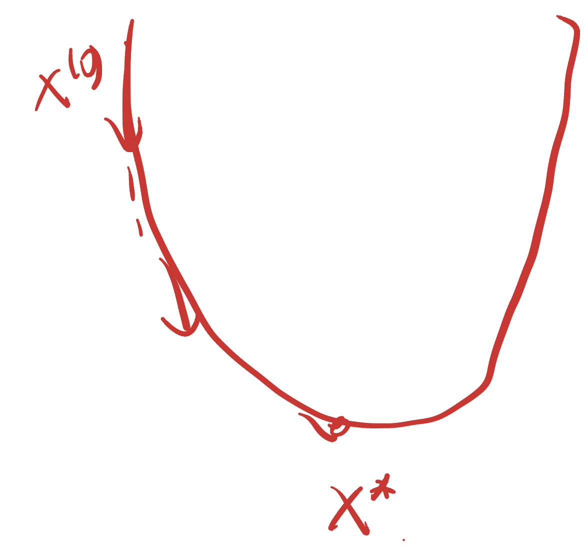
  - for $x \in S$, and x^* being the minimizer



$$f(x) - f(x^*) \leq \frac{1}{2m}\|\nabla f(x)\|_2^2$$

useful as a stoping criterion

Stopping criterion



- Stop when $\dfrac{R^2 + G^2 \sum_{i=1}^{k} t_i^2}{2 \sum_{i=1}^{k} t_i}$ is small

- Stop when $\|\nabla f(x)\|_2^2$ is sufficiently small

- Stop when $\|x^{k+1} - x^k\|_2$ or $|f(x^{k+1}) - f(x^k)|$ is small

- Reality: there isn't a universally good stopping criterion

Logistic regression

classification.

- random variable $y \in \{0,1\}$ with distribution

link function

$$h(x;a,b) = \mathbb{P}(y=1) = \sigma(a^T x + b)$$

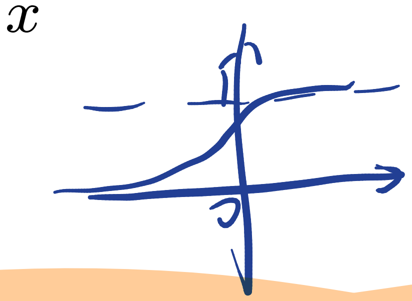
variable

Sigmoid function

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

$x \to -\infty$: $\sigma(x) \to 0$

$x \to +\infty$: $\sigma(x) \to 1$



- maximum likelihood

$$\max_{a,b} \sum_{i=1}^{n} \{y_i \log h(x_i;a,b) + (1-y_i)\log(1-h(x_i;a,b))\}$$

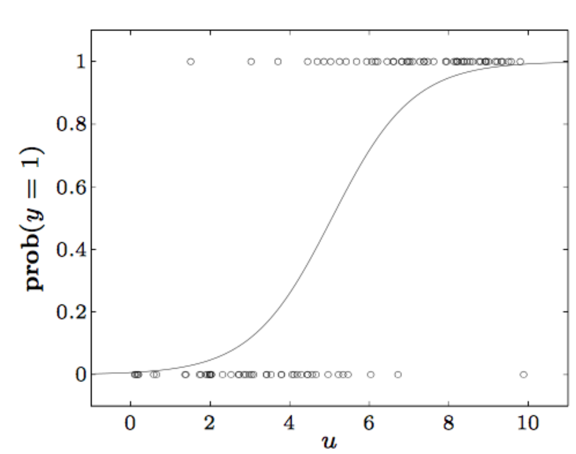
(LHW)

$P(y=1) = p$

$P(y=0) = 1-p$

$\Rightarrow P(y=\mu) = p^{\mu}(1-p)^{1-\mu}$

$\mu \in \{0,1\}$.



likelihood function:

i.i.d. data: (x_i, y_i) $i=1,\dots,n$

$x_i \in \mathbb{R}^p$, $y_i \in \{0,1\}$

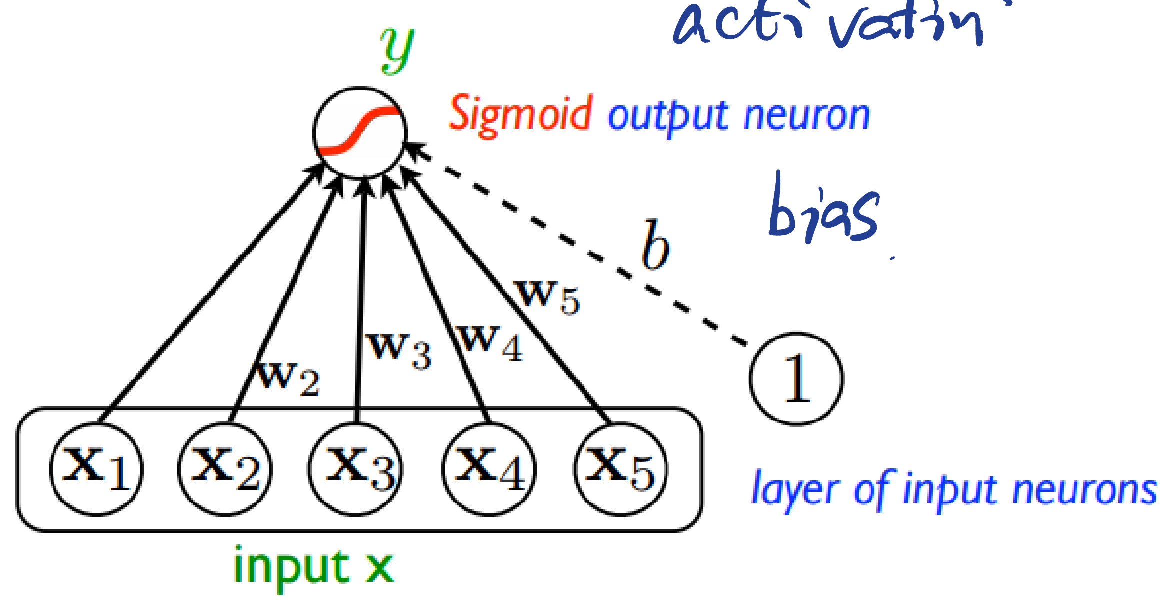
$\log \prod_{i=1}^{n} p^{y_i}(1-p)^{1-y_i}$

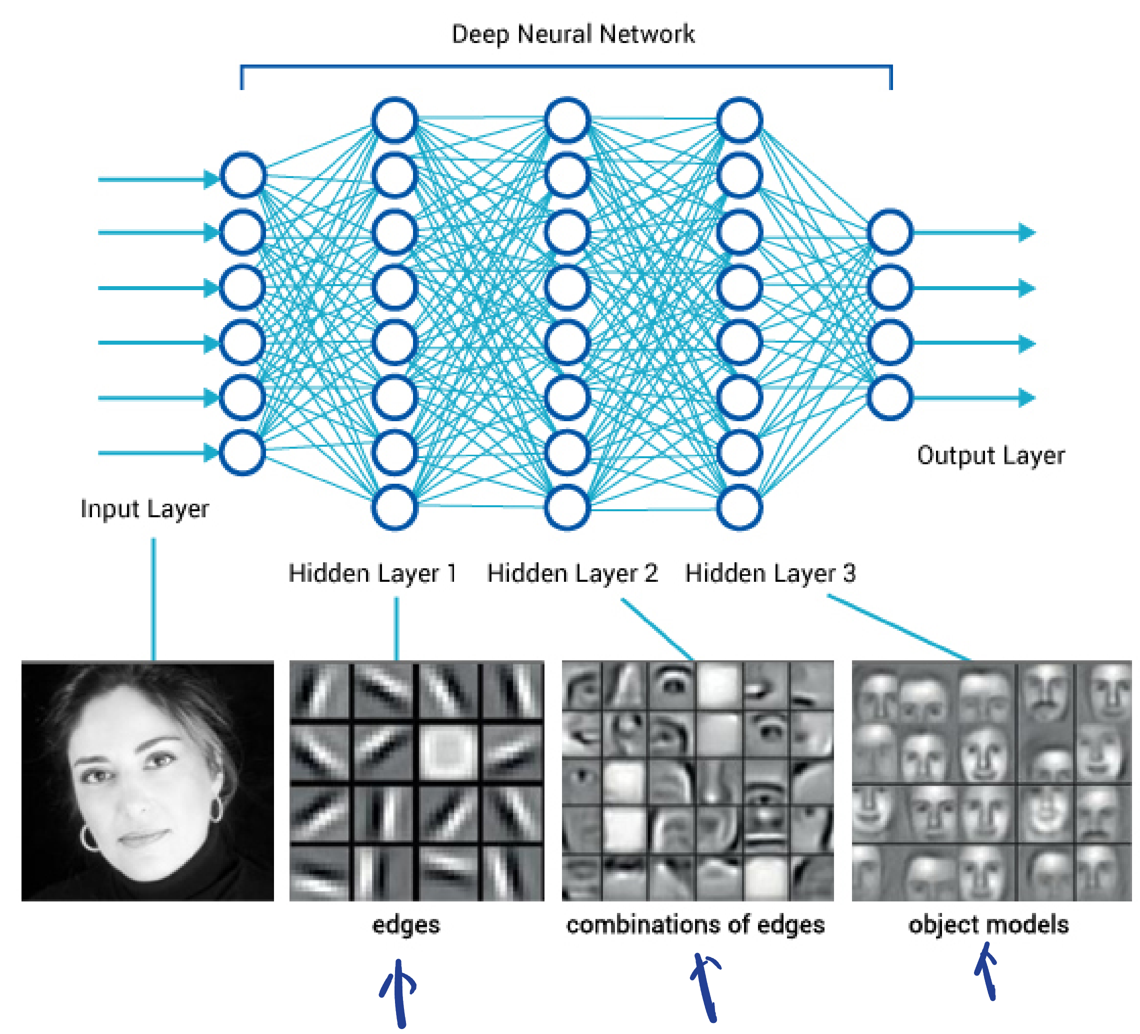
$= \sum_{i=1}^{n} \log p^{y_i}(1-p)^{1-y_i} = \sum y_i \log p + (1-y_i)\log(1-p)$

Deep learning and neural networks

activation

bias





Example: Optimization in training neural networks

Data: (x_i, y_i), $i = 1, \dots, n$.

Loss function: $\ell(w, \alpha, \beta) = \sum_{i=1}^{n} (y_i - \sigma(w^T z_i))^2$

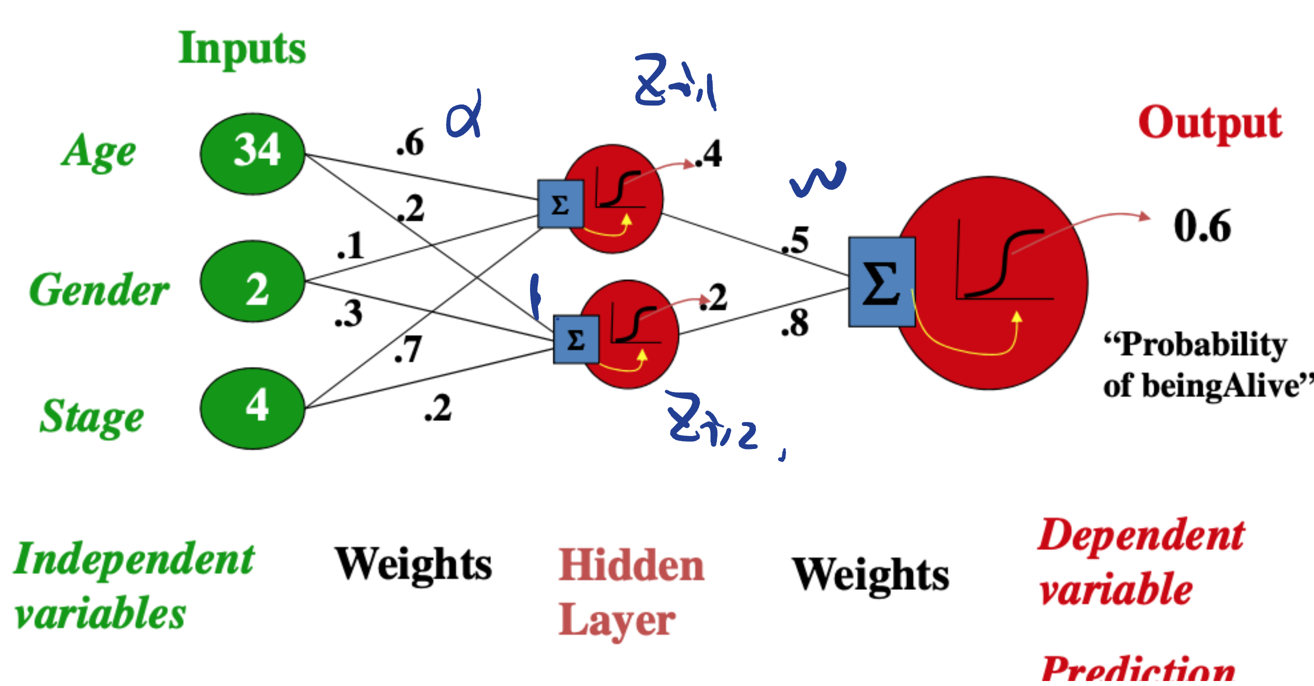
$$\min_{w,\alpha,\beta} \ell(w, \alpha, \beta)$$

where

$$z_{i,1} = \sigma(\alpha^T x_i), \quad z_{i,2} = \sigma(\beta^T x_i)$$

Sigmoid function $\sigma(x) = \frac{1}{1+e^{-u}}$

- Not a convex objective function
- Use gradient descent to find a local optimum solution



Gradient descent: backpropagation

[Pytorch, Tensorflow].

Backpropagation computes the gradient in weight space of a feedforward neural network, with respect to a loss function.

- Loss function: $\ell(w, \alpha, \beta) = \sum_{i=1}^{n} (y_i - \sigma(w^T z_i))^2$
- Gradient with respect to the weights w in the last layer (HW).

$$\frac{\partial \ell(w, \alpha, \beta)}{\partial w} = -\sum_{i=1}^{n} 2(y_i - \sigma(u_i))\sigma(u_i)(1 - \sigma(u_i))z_i$$

  where $u_i = w^T z_i$, $z_{i,1} = \sigma(\alpha^T x_i)$, $z_{i,2} = \sigma(\beta^T x_i)$
- Use chain rule, gradient with respect to weights in previous layer

$$\begin{aligned}\frac{\partial \ell(w, \alpha, \beta)}{\partial \alpha} &= \frac{\partial \ell(w, \alpha, \beta)}{\partial z_{i,1}} \frac{\partial z_{i,1}}{\partial \alpha} \\ &= -\sum_{i=1}^{n} 2(y_i - \sigma(u_i))\sigma(u_i)(1 - \sigma(u_i))w_1 \sigma(v_i)(1 - \sigma(v_i))x_i\end{aligned}$$

  where $v_i = \alpha^T x_i$

Example: prostate cancer

The data for this example come from a study by Stamey et al. (1989). They examined the correlation between the level of prostate-specific antigen and a number of clinical measures in men who were about to receive a radical prostatectomy. The variables are log cancer volume (`lcavol`), log prostate weight (`lweight`), `age`, log of the amount of benign prostatic hyperplasia (`lbph`), seminal vesicle invasion (`svi`), log of capsular penetration (`lcp`), Gleason score (`gleason`), and percent of Gleason scores 4 or 5 (`pgg45`). The correlation matrix of the predictors given in Table 3.1 shows many strong correlations. Figure 1.1 (page 3) of Chapter 1 is a scatterplot matrix showing every pairwise plot between the variables. We see that `svi` is a binary variable, and `gleason` is an ordered categorical variable. We see, for example, that both `lcavol` and `lcp` show a strong relationship with the response `lpsa`, and with each other. We need to fit the effects jointly to untangle the relationships between the predictors and the response.

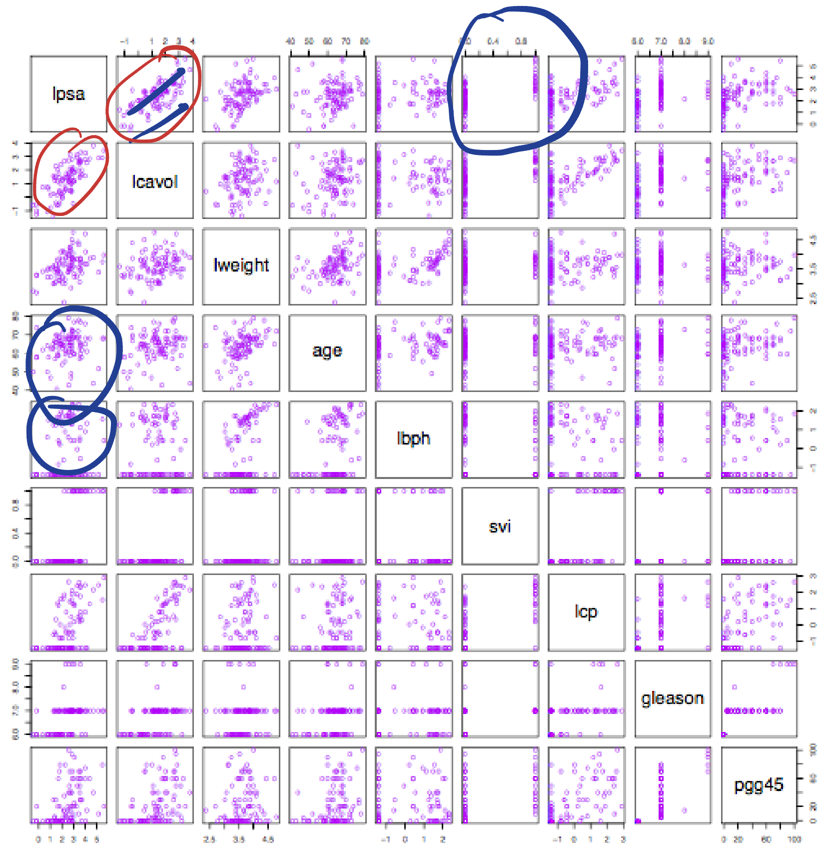


FIGURE 1.1. *Scatterplot matrix of the prostate cancer data. The first row shows the response against each of the predictors in turn. Two of the predictors,* `svi` *and* `gleason`*, are categorical.*

Scatter plot.

<u>Variable selection</u>: for multiple linear regression, select the "most important" variables that are responsible for the output:

Lasso (1996).

$$\min_{\beta} \|y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

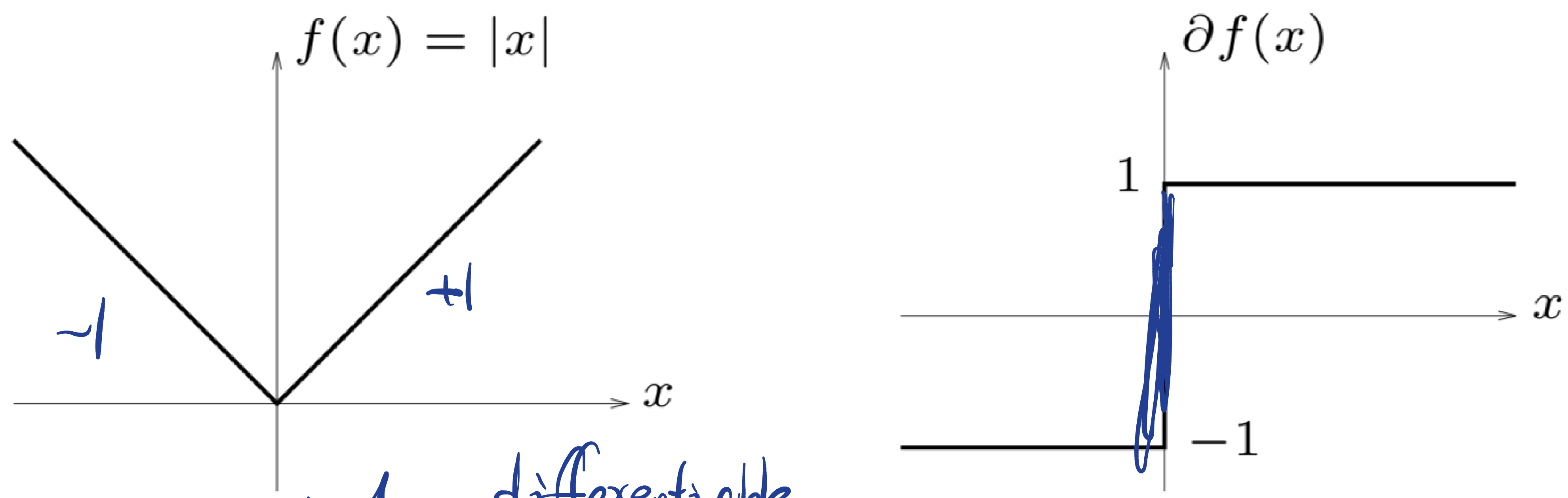
where $\|\beta\|_1 = \sum_{i=1}^{p} |\beta_i|$

regularizer,

$\lambda > 0$: regularize parameter

Example of subgradient

$f(x) = |x|$



not differentiable.

righthand plot shows $\bigcup \{(x,g) \mid x \in \mathbf{R},\ g \in \partial f(x)\}$

We need this to solve lasso:

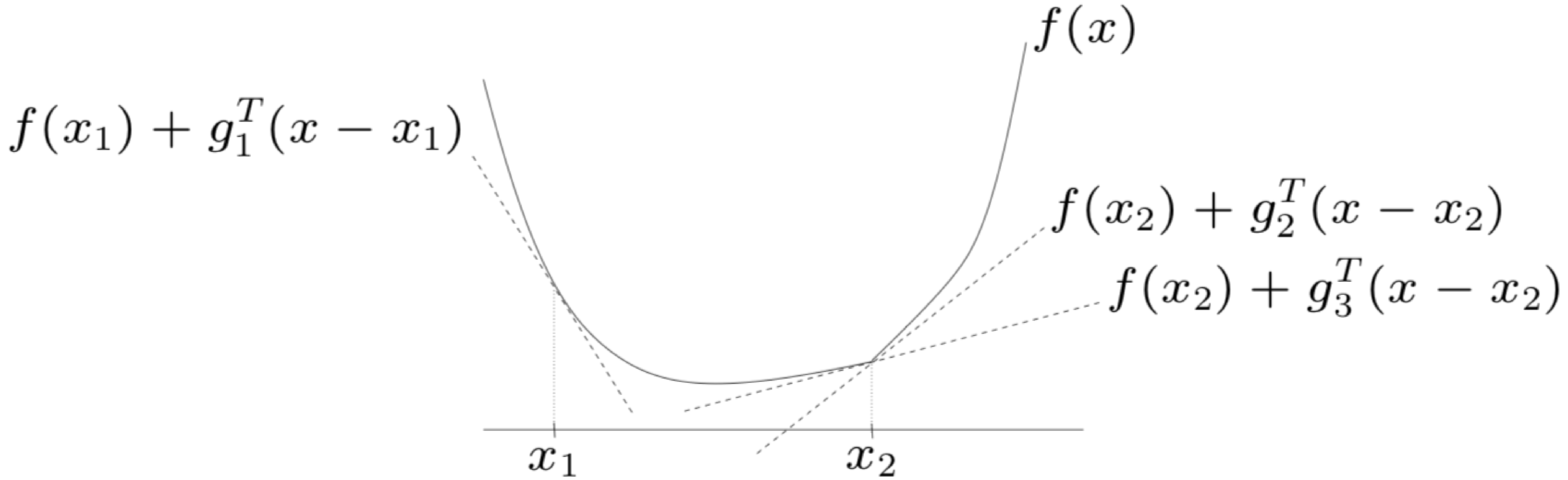
$$\min_{\beta} \|y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

where $\|\beta\|_1 = \sum_{i=1}^{p} |\beta_i|$

Extension: Subgradient

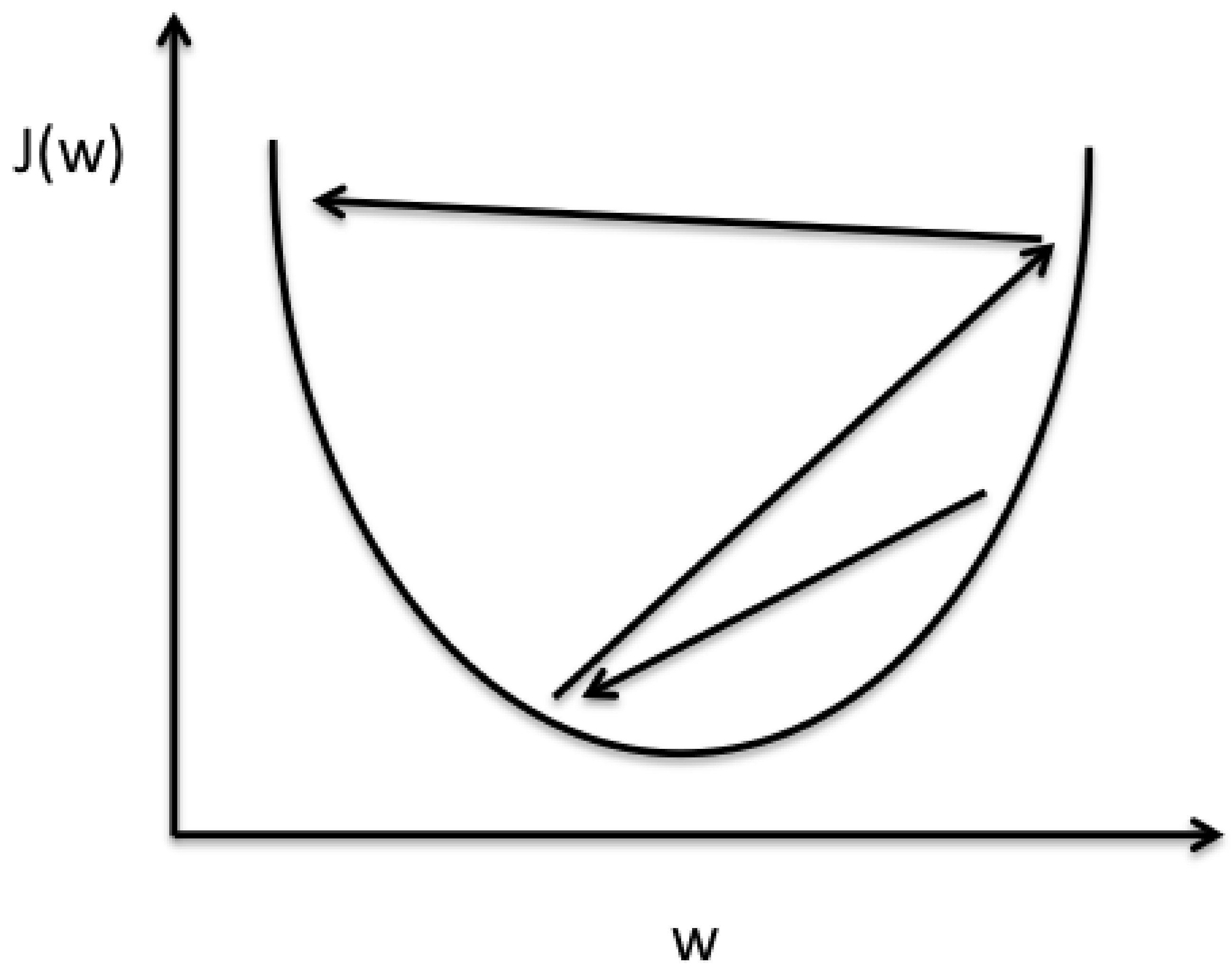
g is a subgradient of f (not necessarily convex) at x if

$$f(y) \geq f(x) + g^T(y - x), \quad \forall y$$

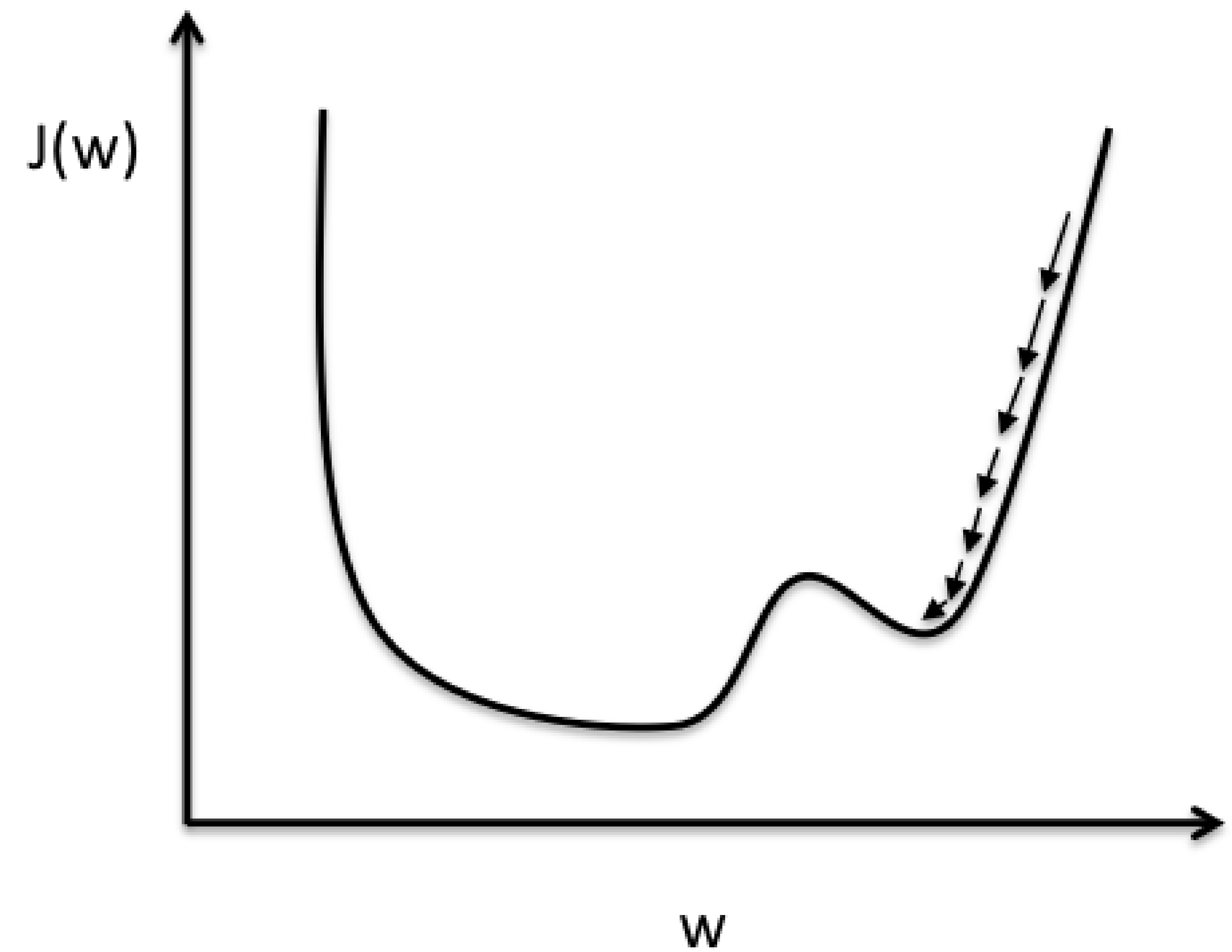


g_2, g_3 are subgradients at x_2; g_1 is a subgradient at x_1

Choice of step-size



Large learning rate: Overshooting.



Small learning rate: Many iterations until convergence and trapping in local minima.

Step size rules

- Step sizes are fixed ahead of time [data adaptive stepsize, AdaM polyak]
- Constant step size: $t_k = t$ (constant) 0.1 , 0.2
- Constant step length: $t_k = \gamma / \|\nabla f(x^{(k)})\|_2$ (so $\|x^{(k+1)} - x^{(k)}\|_2 = \gamma$)
- Square summable but not summable: step sizes satisfy

Recall page 15.

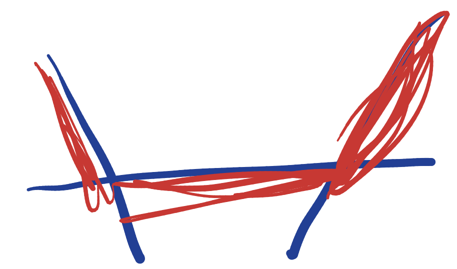
$$\sum_{k=1}^{\infty} t_k^2 < \infty, \quad \sum_{k=1}^{\infty} t_k = \infty$$

  e.g., $t_k = 1/k$
- Nonsummable diminishing: step sizes satisfy

$$\lim_{k \to \infty} t_k = 0, \quad \sum_{k=1}^{\infty} t_k = \infty.$$

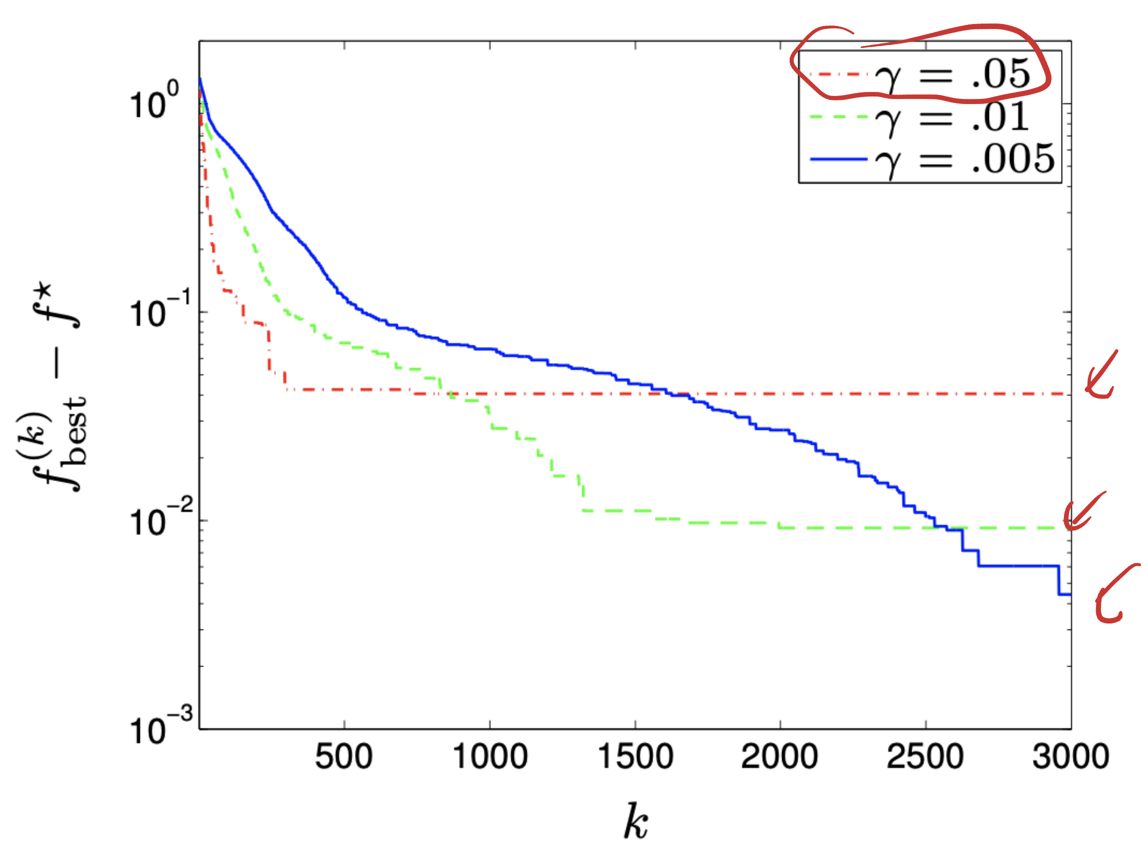
Example

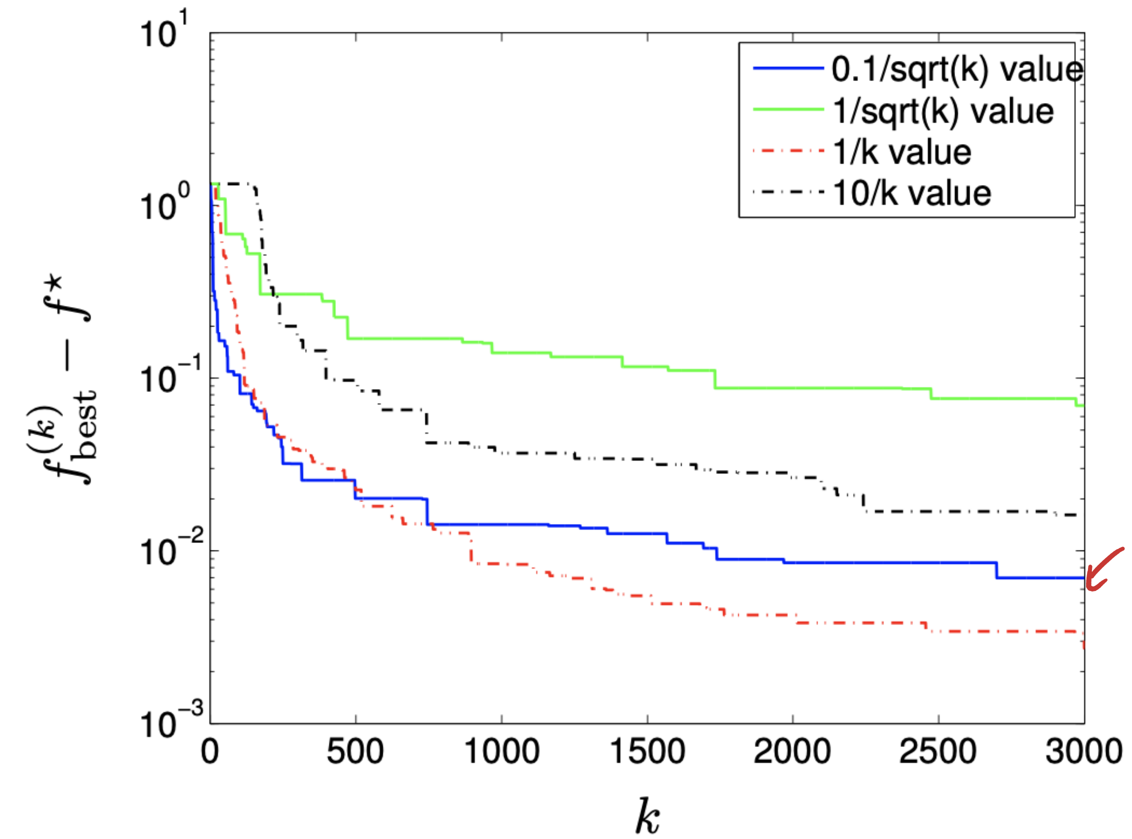
Minimizing piecewise linear function



$$\text{minimize}_{a,b} \; \underbrace{\max_{i=1,\ldots,m} (a_i^T x + b_i)}_{\text{convex}}$$

Problem instance with 20 variables.

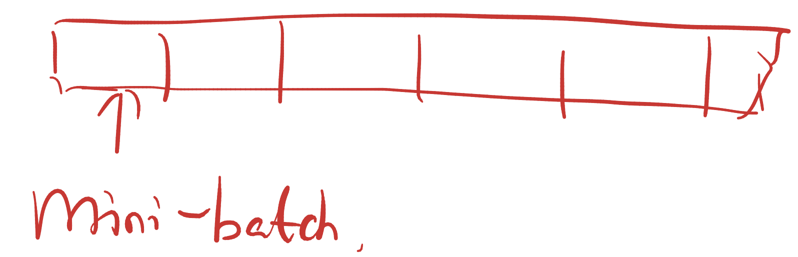




Contents

Stochastic gradient descent (SGD)

$O(np)$.

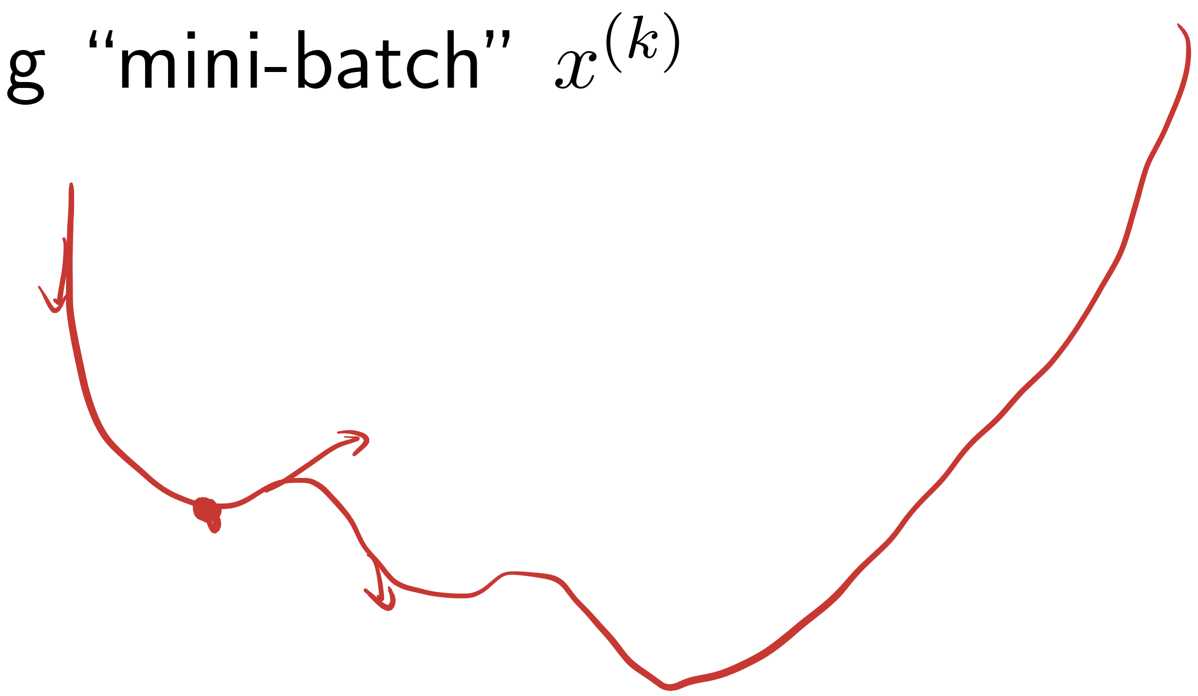


- Sequentially "load" part of data; use gradient using "mini-batches" of data
- Save memory; sometimes has better performance for non-convex problems
- Uses noisy unbiased subgradients

$$x^{(k+1)} = x^{(k)} - t_k \tilde{g}^{(k)}$$

- $\tilde{g}^{(k)}$ is any noisy unbiased estimate of gradient using "mini-batch" $x^{(k)}$

$$\mathbb{E}[\tilde{g}^{(k)}] = g^{(k)}$$



Stochastic gradient descent for linear regression

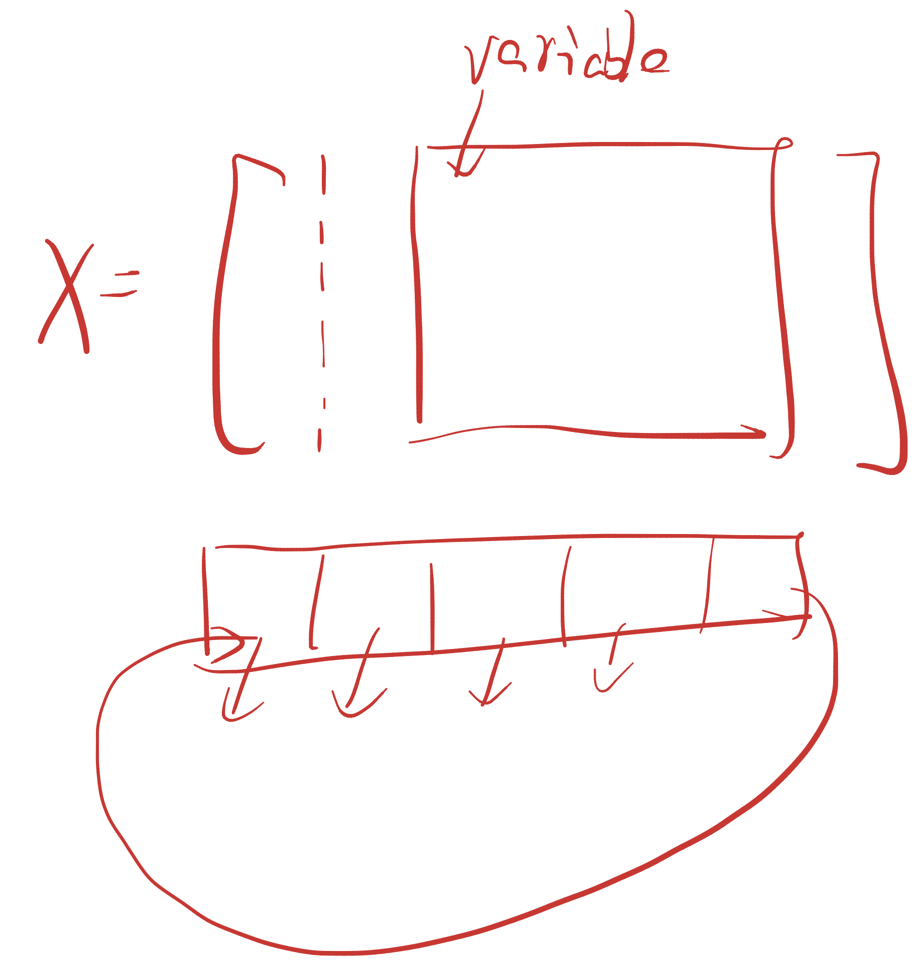
Loss function

$$\min_{\beta} \|y - X\beta\|_2^2$$

Gradient: $f(\beta) = 2X^T(X\beta - y)$

Partition the **data** into M parts

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_M \end{bmatrix}, X = \begin{bmatrix} X_1 \\ \vdots \\ X_M \end{bmatrix}$$



Stochastic gradient descent:

$$\beta^{(k+1)} = \beta^{(k)} - t_k X_k^T (X_k \beta^{(k)} - y_k)$$

$O(\frac{n}{M} p)$

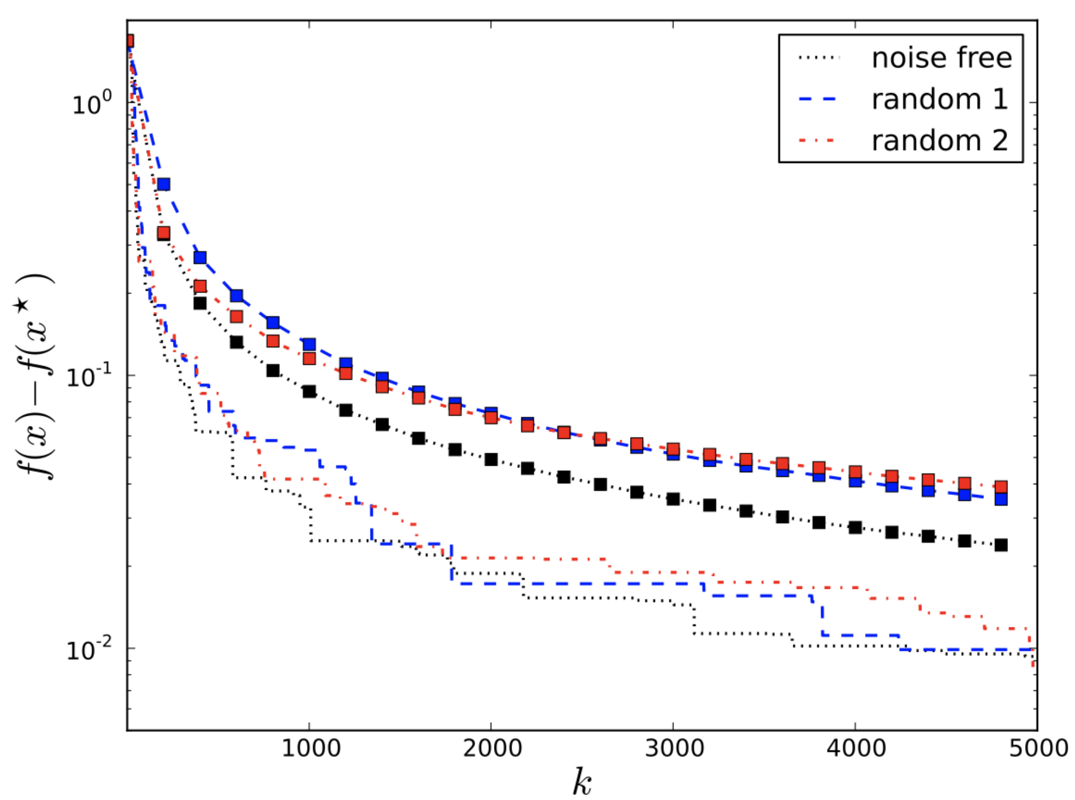
Compare with Gradient descent

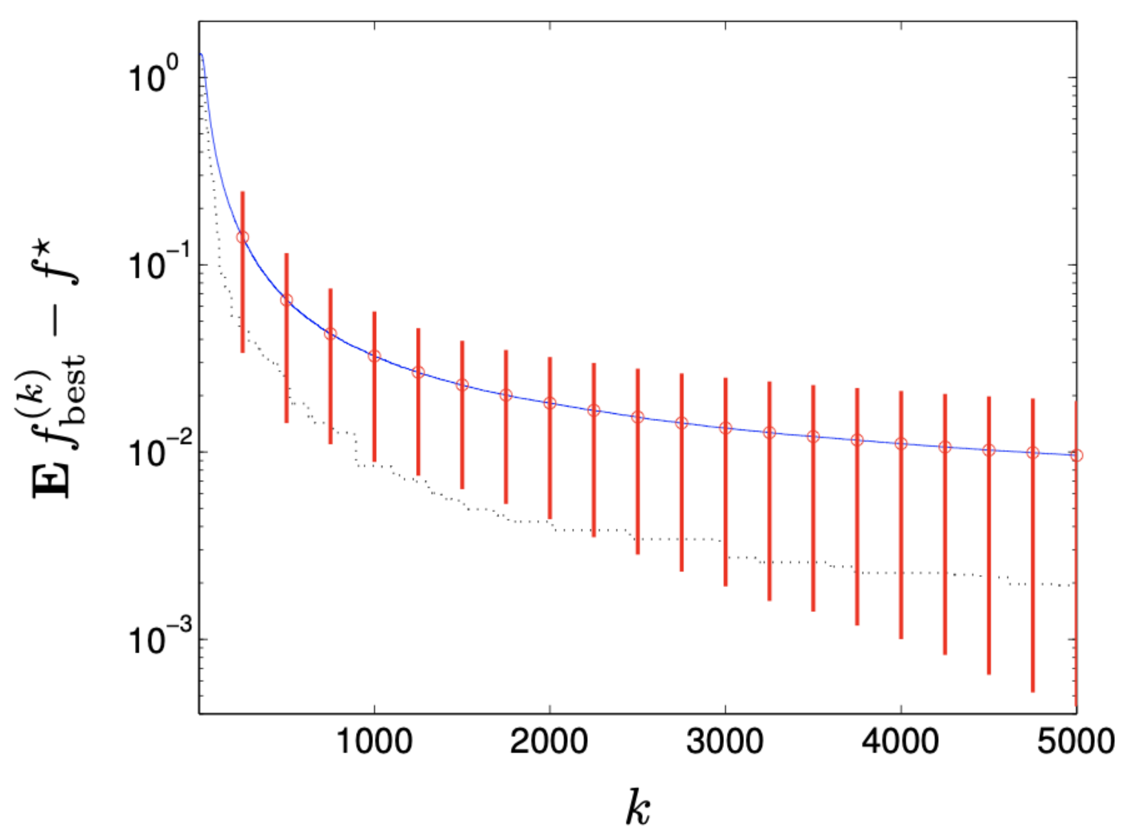
$$\beta^{(k+1)} = \beta^{(k)} - t_k X^{\mathsf{T}} (X\beta^{(k)} - y)$$

$O(np)$

Example

Minimizing piecewise linear function with SGD (solid lines are averaged over 100 instances)





Contents

Second-order Method: Newton's method

$$x^{(k+1)} = x^{(k)} - t_k[H\{f(x^{(k)})\}]^{-1}\nabla f(x^{(k)})$$

t_k: step-size for the kth iteration

- interpretation $x + v$ minimizes the second order approximation of the function

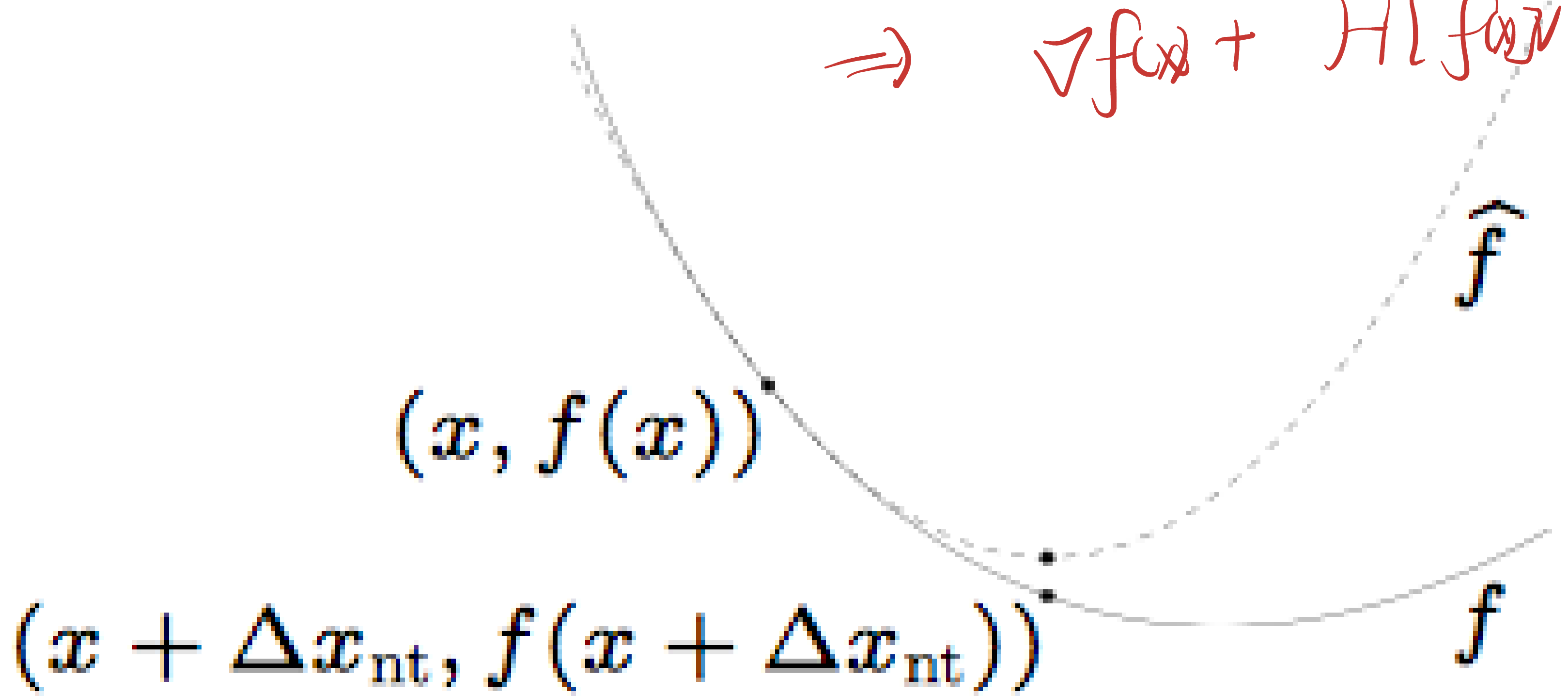
$$f(x+v) \approx f(x) + \nabla f(x)^\intercal v + \frac{1}{2} v^\intercal H\{f(x)\} v$$

quadratic approximation

Find v to minimize

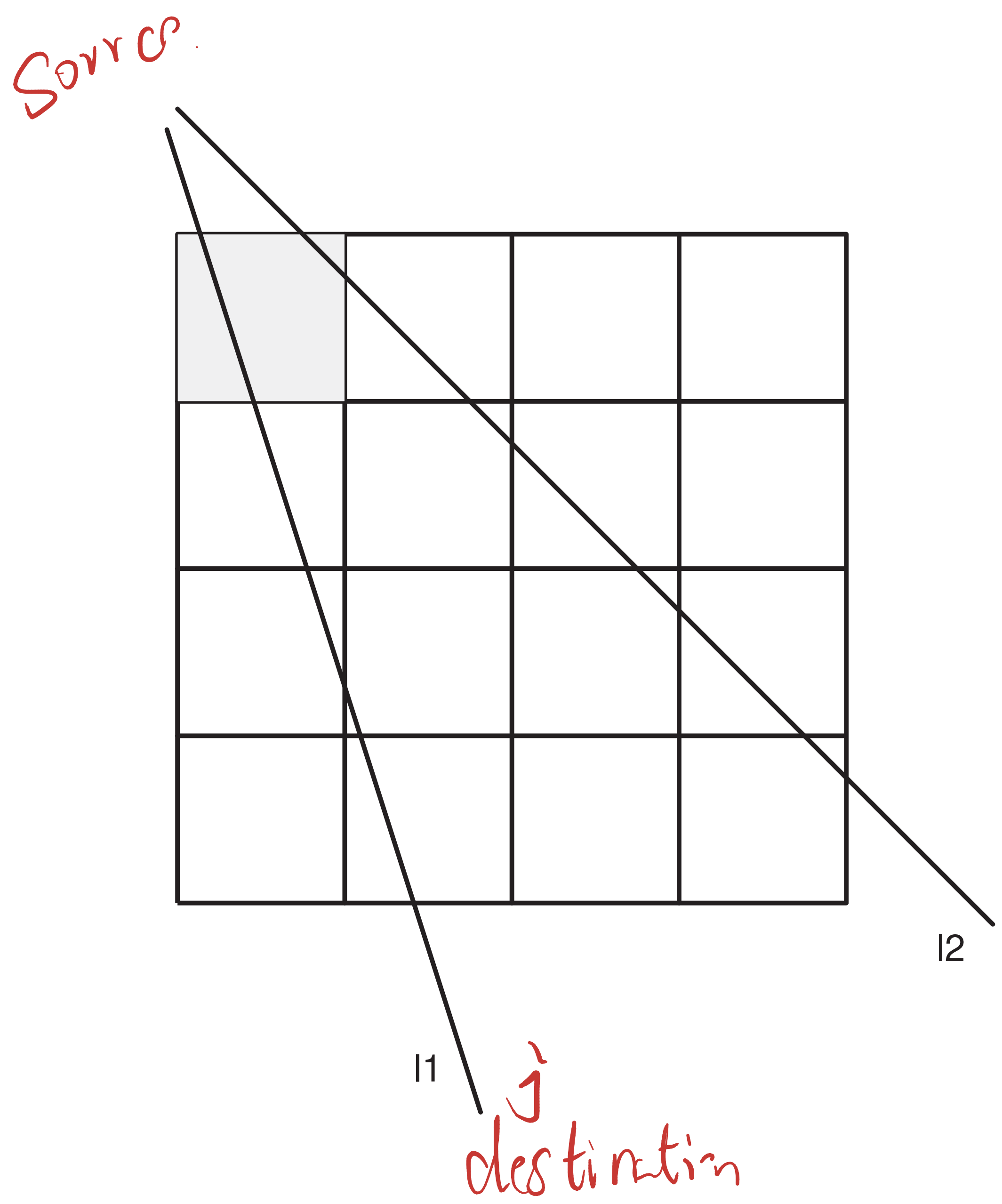
$\Rightarrow \nabla f(x) + H[f(x)]v = 0$

$v = -H^{-1}[f(x)]\nabla f(x)$



Contents

CT image reconstruction using MLE



- n line integral measurements.
- image of size $p \times p$
- jth line is characterized by $\{l_{ij}\}$, where l_{ij} is the length of the intersection of jth line with ith pixel (or zero if they don't intersect)

low-dose CT

- Measurements forms a vector $y \in \mathbb{R}^n$

$$y_j \sim \text{Poisson}(\lambda_j), \quad j = 1, \ldots, n.$$

- The parameters $\{\lambda_j\}$ are determined according to Beer's law:

$$\lambda_j = I_j e^{-\sum_{i=1}^{p^2} l_{ij} x_j},$$

where I_j is the intensity of the ith X-ray before passing through the object.

- The problem is to reconstruct the pixel densities $x \in \mathbb{R}^{p^2}$ from the line integral measurements y.

Maximum likelihood estimate

- The likelihood function is given by

$$p_x(y) = \prod_{j=1}^{n} \frac{\lambda_j^{y_j}}{y_j!} e^{-\lambda_j},$$

- Log-likelihood function

$$\ell(x) = \log p_x(y) = \sum_{j=1}^{n} (y_j \log \lambda_j - \lambda_j - \log(y_j!)).$$

- MLE estimate

$$\text{minimize}_x \quad -\sum_{j=1}^{n} (y_j \log \lambda_j - \lambda_j)$$

- To prevent overfitting the noisy data, we also add a regularization term $\phi(x)$ in the cost function.

$$\text{minimize}_x \quad -\sum_{j=1}^{n}(y_j \log \lambda_j + \lambda_j) + \lambda\phi(x).$$

Tiknono Lasso.

e.g. $\phi(x) = \|x\|_2^2$, $\phi(x) = \|x\|_1$.

- Matrix-vector representation

$$\text{minimize}_x \quad f(x) = y^T Lx + I^T e^{-Lx} + \lambda\phi(x).$$

Forward projection matrix $L = [l_1, \cdots, l_n] \in \mathbb{R}^{n\times p^2}$.
$I = [I_1, \cdots, I_n]^T \in \mathbb{R}^n$.
Functions e^x are overloaded to operate on each element of the input vector.

- Since $f(x)$ is differentiable and convex, a necessary and sufficient condition for a solution x^* to be optimal is

$$\nabla f(x^*) = L^T \left(y - \mathbf{diag}\{I\} e^{-Lx^*} \right) + \lambda \nabla \phi(x^*) = 0.$$

- Hessian matrix

$$H = L^T \mathbf{diag}\{\hat{y}\} L + \lambda H[\phi(x)] > 0,$$

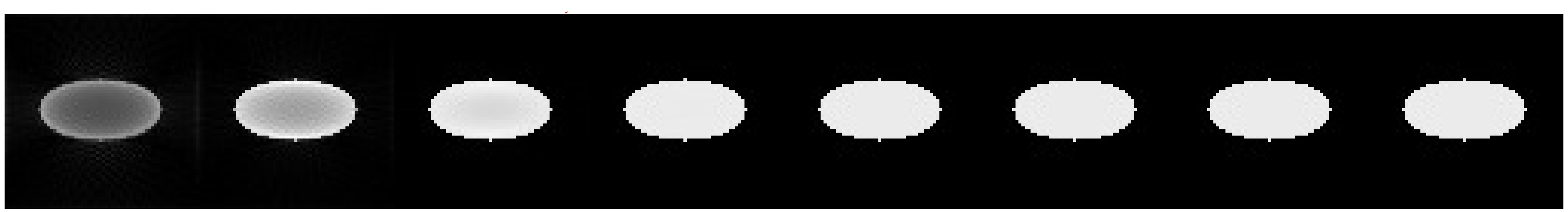
where

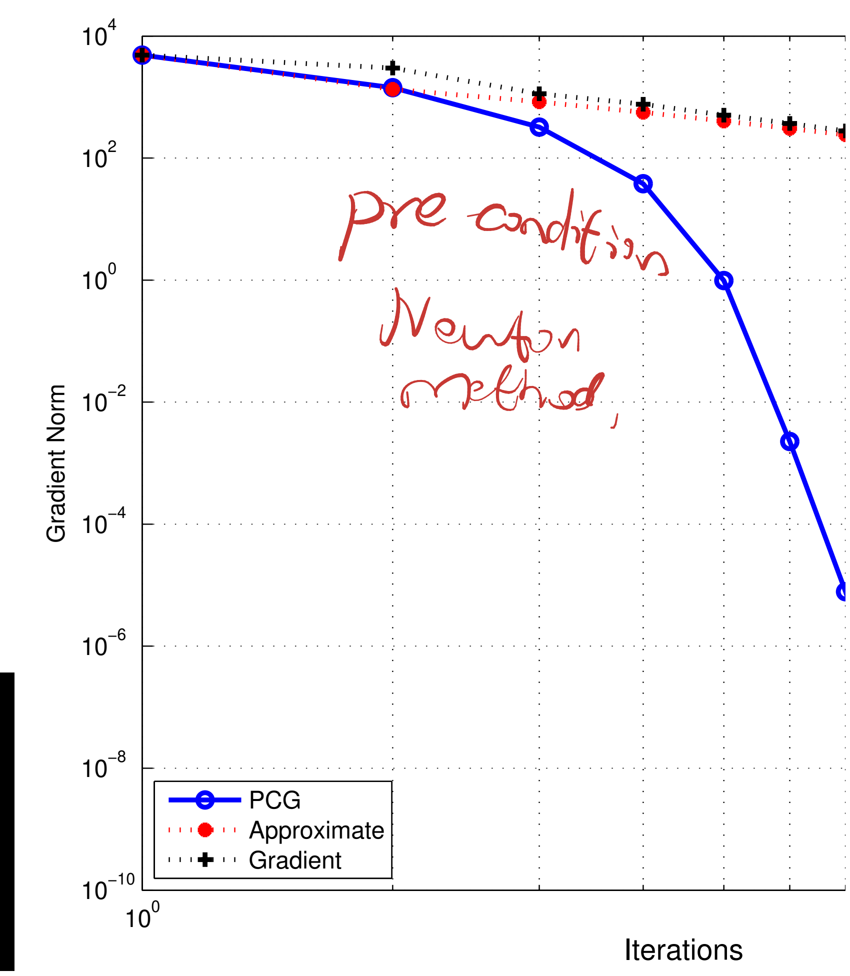
$$\hat{y} = \mathbf{diag}\{I\} e^{-Lx}$$

Results

We simulated a parallel beam CT geometry, with 100 detectors, and 180 uniform angular sampling, so $m = 18000$. The rays spread out wide enough to cover the entire image, with uniform intensities $I_j = 10^6$. The image has 64×64 $(= 4096)$ pixels.
Use $\|\nabla f\|_2 < 10^{-8}$ as a stopping criterion.

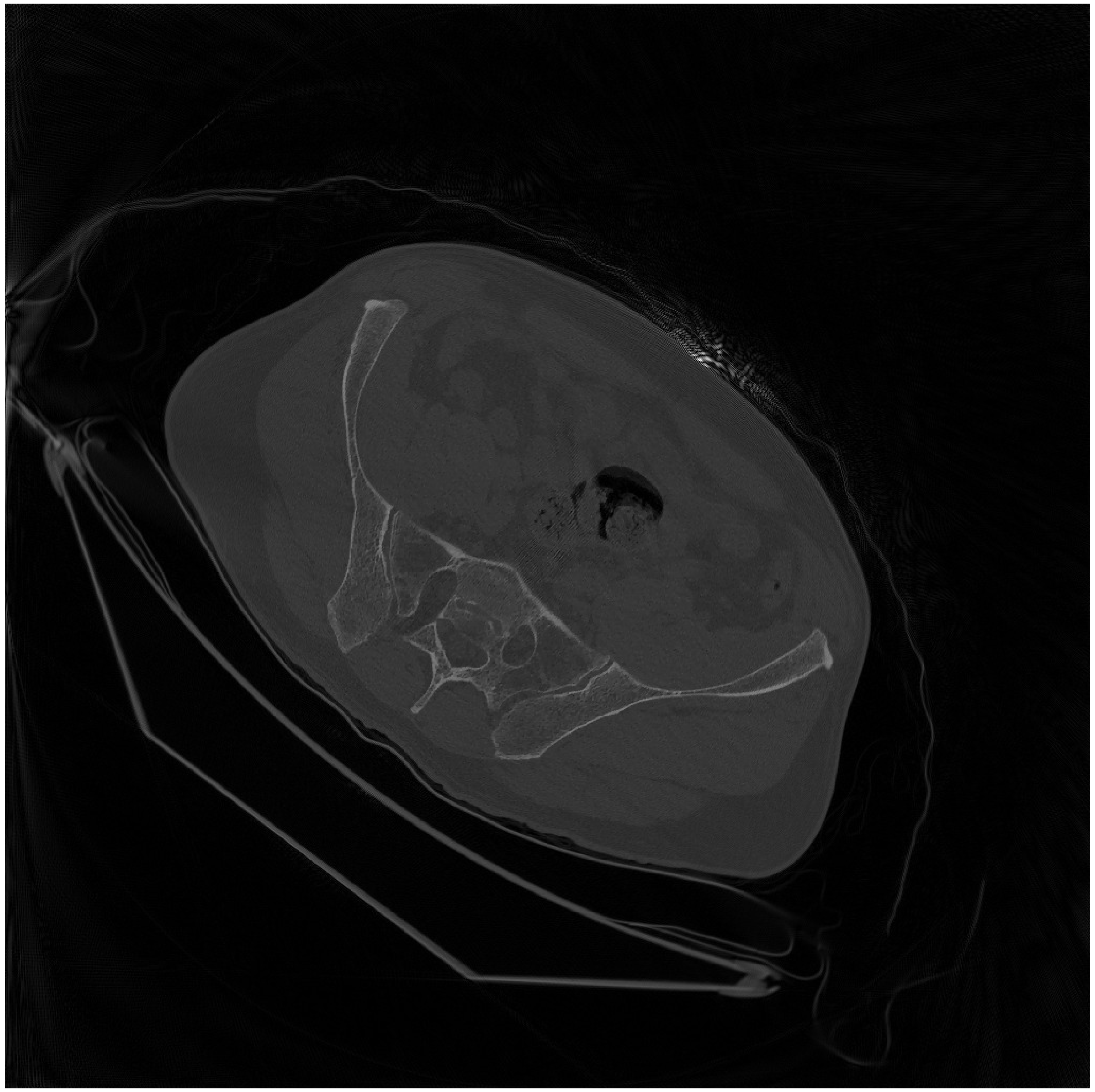
k=1 2 →





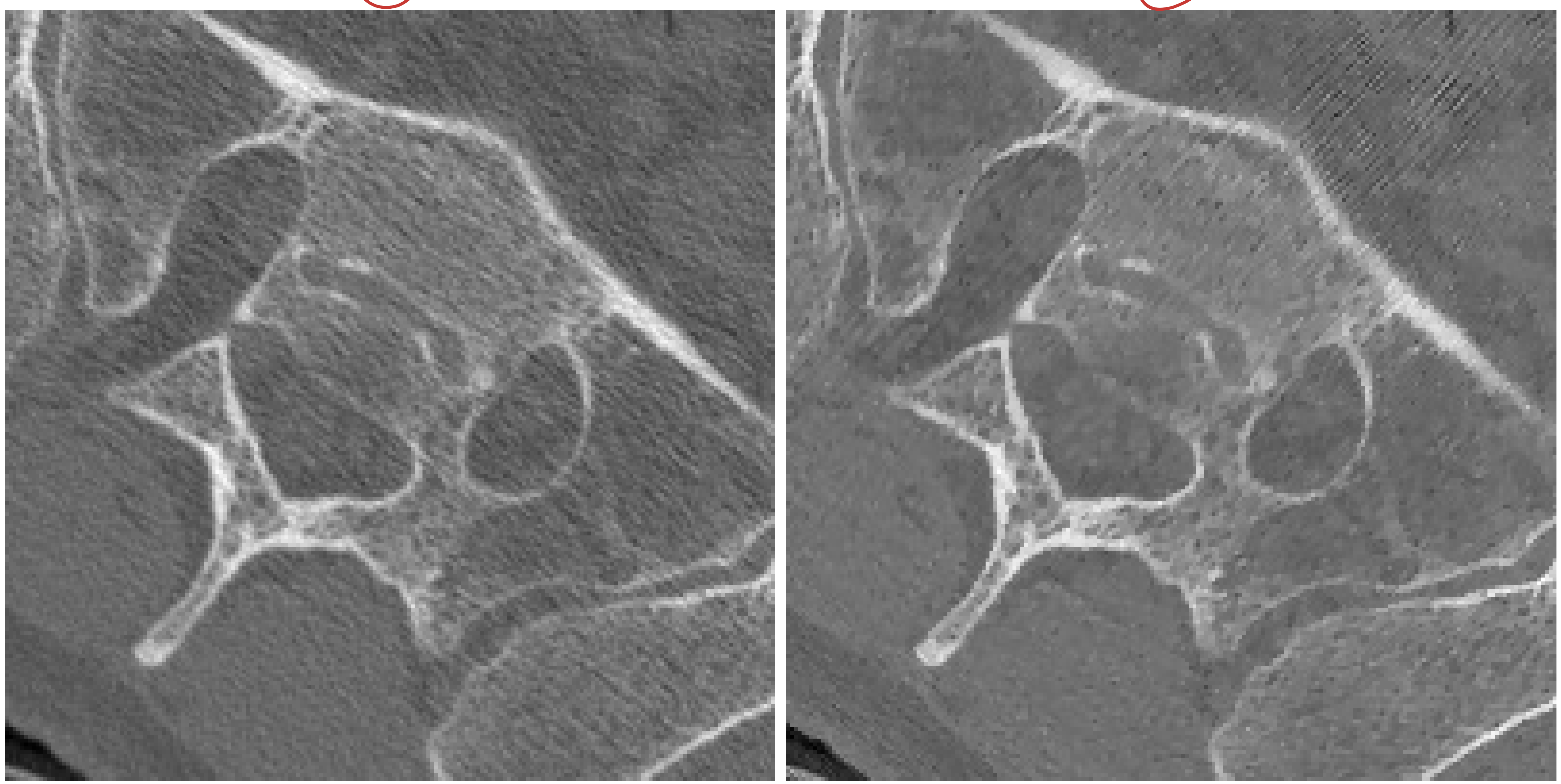
pre condition
Newton method,

Using real data measured on a GE fan beam geometry CT scanner: 1024×1024.



Comparison with a deterministic inverse algorithm

FBP ML



Summary

- Gradient descent and convergence
  - Example: solving multiple linear regression, logistic regression, neural networks
  - Subgradient
  - Step-size
  - Stochastic gradient descent
- Newton's method