Convexity

- When is an optimization (or learning) easy/fast to solve?

Recap: Ridge vs. Lasso

- **Ridge**

$$\text{minimize}_w \quad \sum_{i=1}^{n} (w^T x_i - y_i)^2 + \lambda \|w\|_2^2$$

  - Very fast:
    - Closed form solution if used with linear models
    - Even with other loss functions, optimization is fast for squared ℓ_2 regularization, because $\|w\|_2^2$ is **convex and smooth**

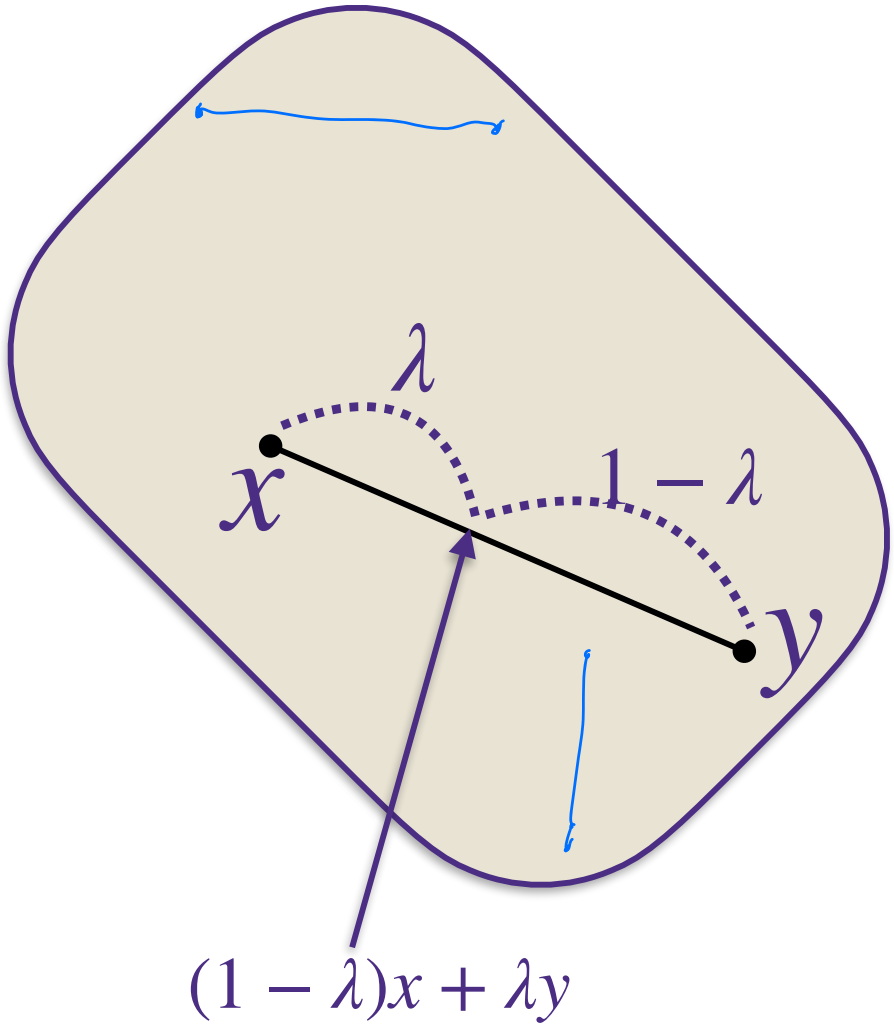
- **Lasso**

$$\text{minimize}_w \quad \sum_{i=1}^{n} (w^T x_i - y_i)^2 + \lambda \|w\|_1$$

  - Slower than Ridge:
    - Requires iterative optimization algorithm like sub-gradient descent
    - In particular, it is slower because $\|w\|_1$ is **convex but non-smooth**

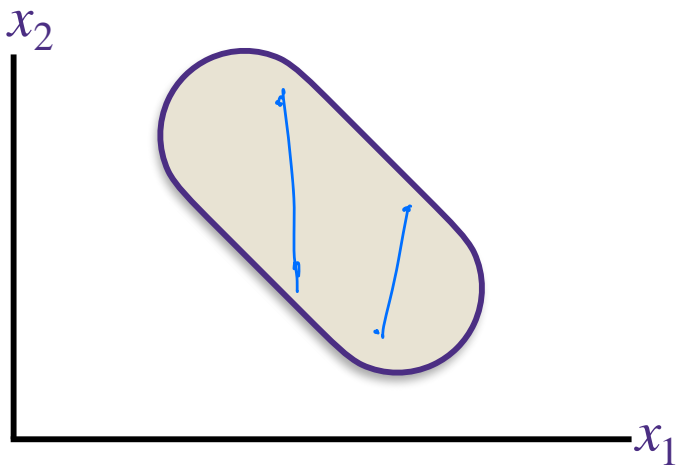
What is a convex set?

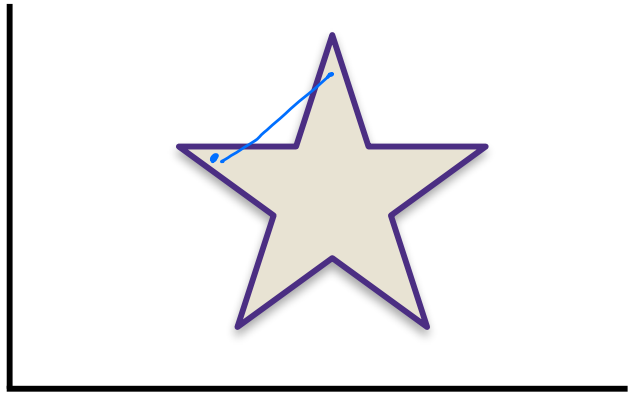
A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

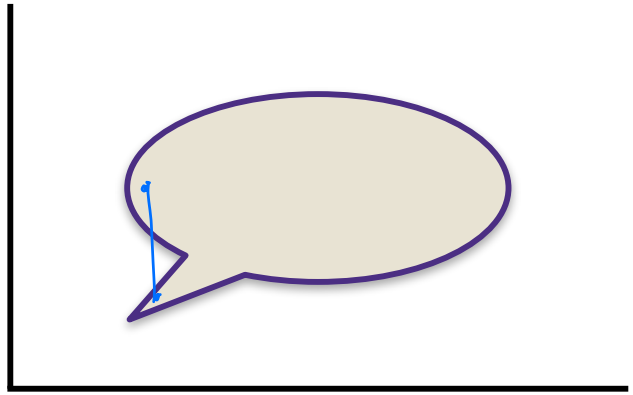


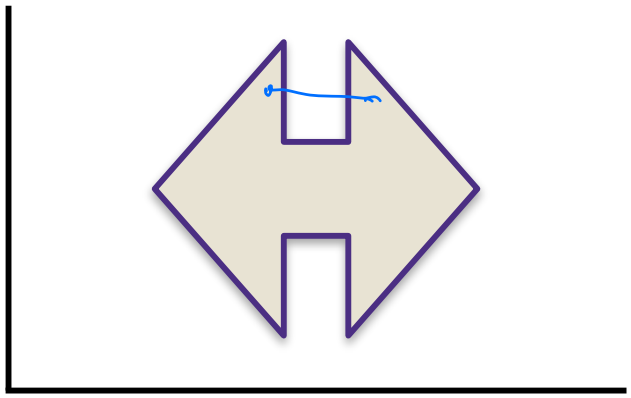
What is a convex set?

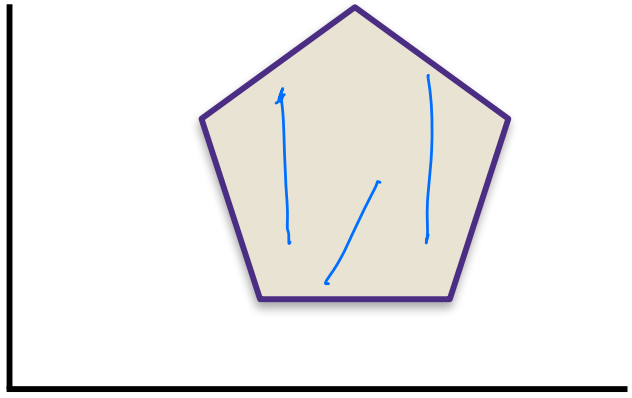
A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

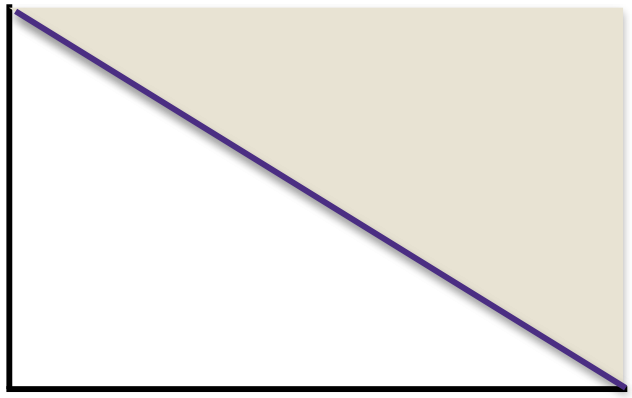






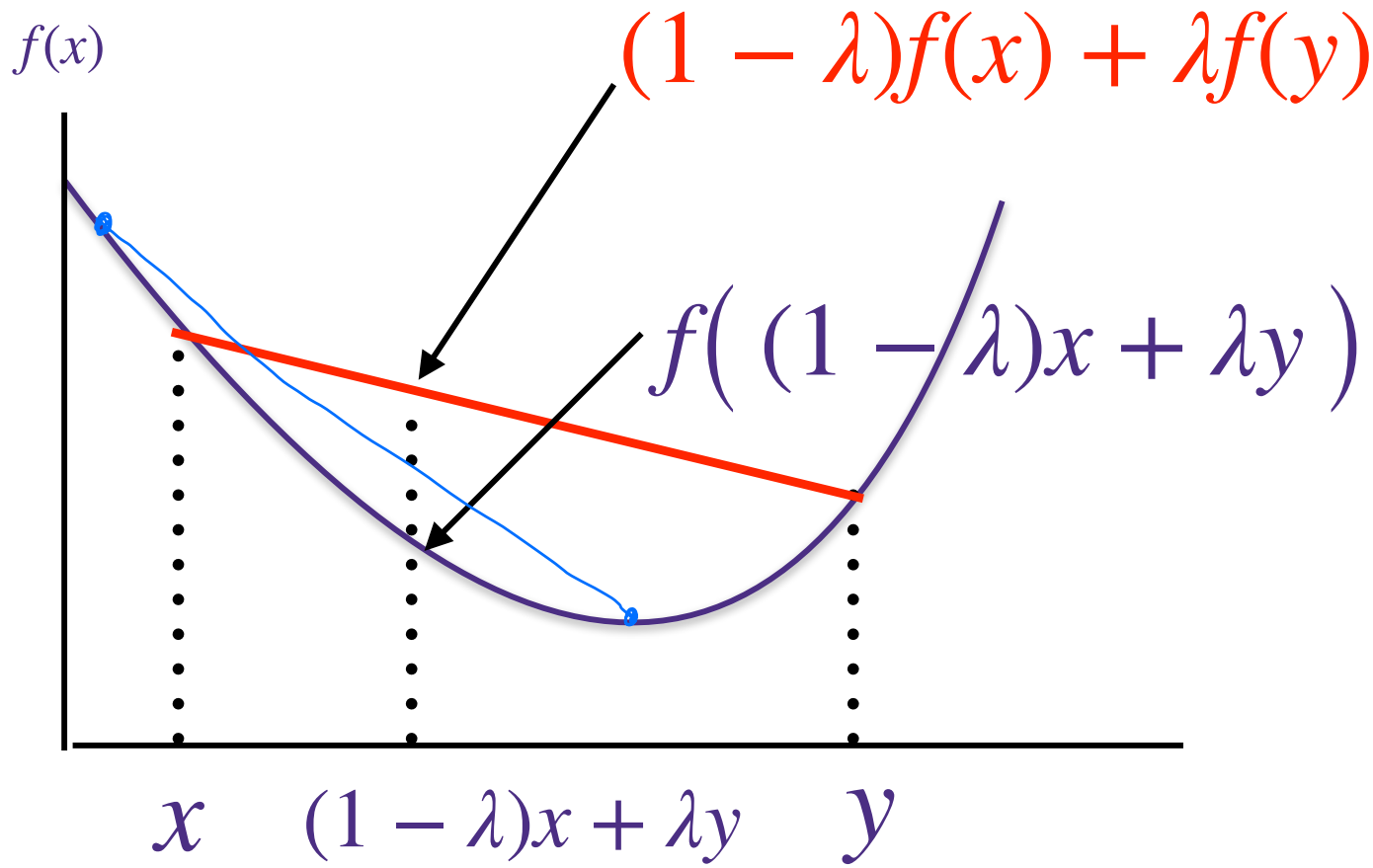






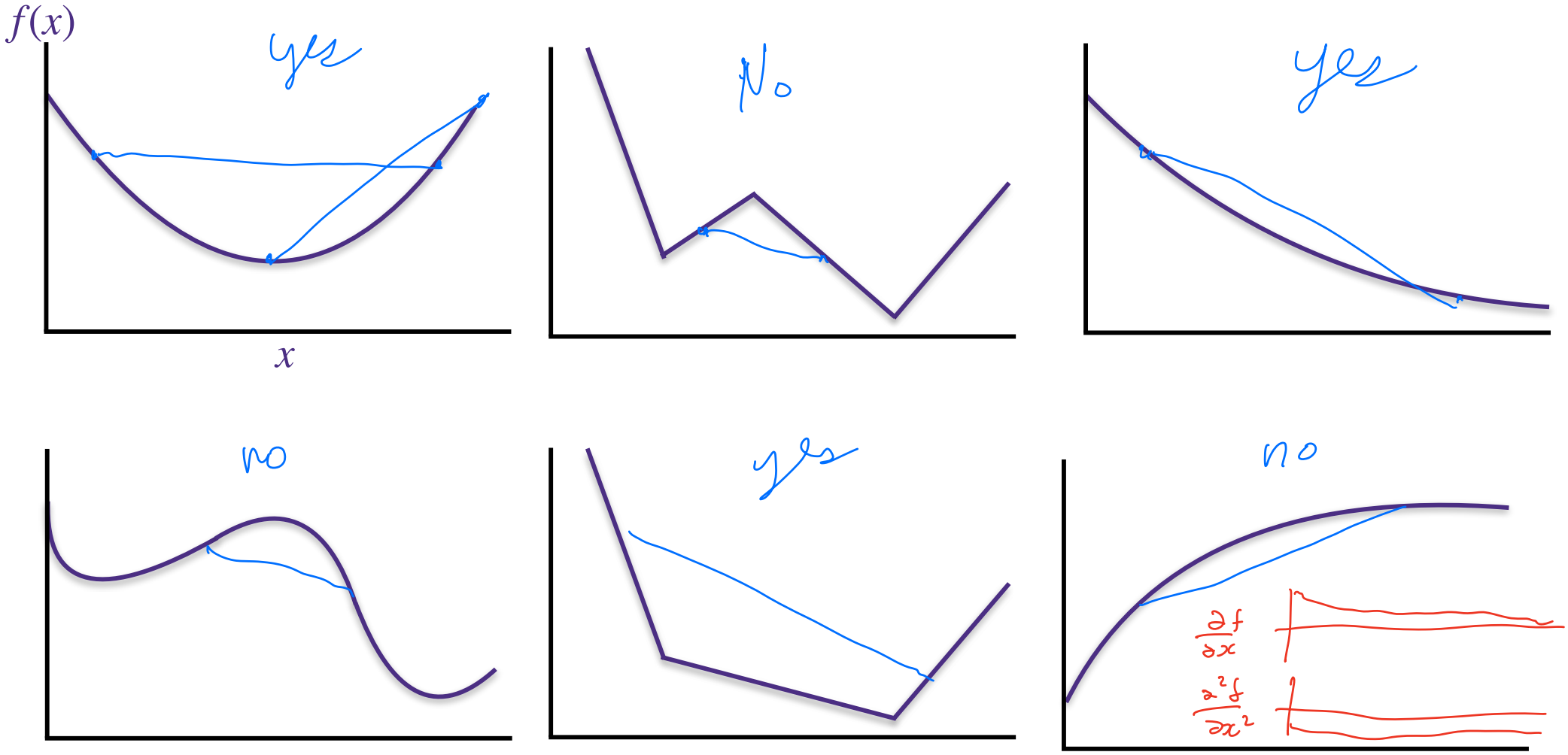
What is a convex function?

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in \mathbb{R}^d$ and $\lambda \in [0, 1]$



What is a convex function?

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in \mathbb{R}^d$ and $\lambda \in [0, 1]$



Convex functions and convex sets?

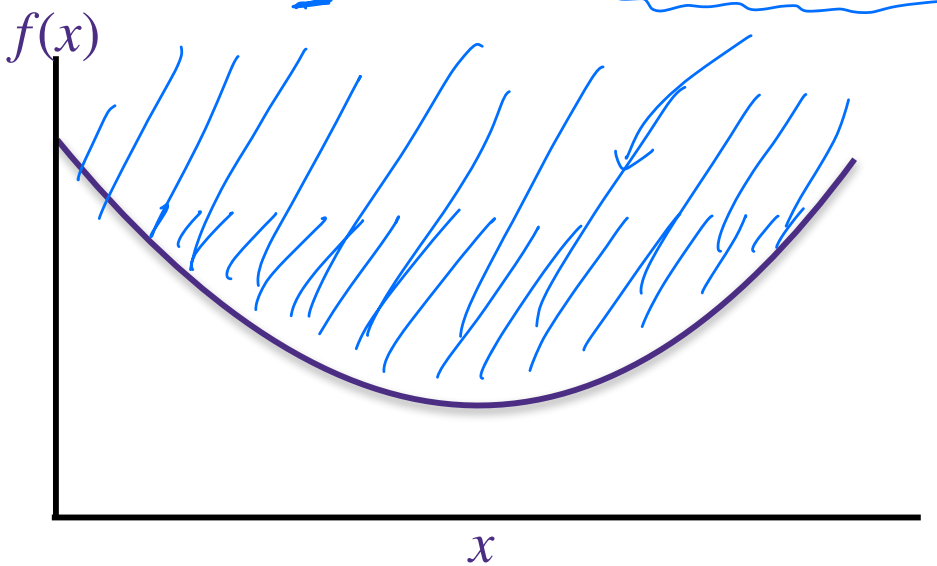
A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

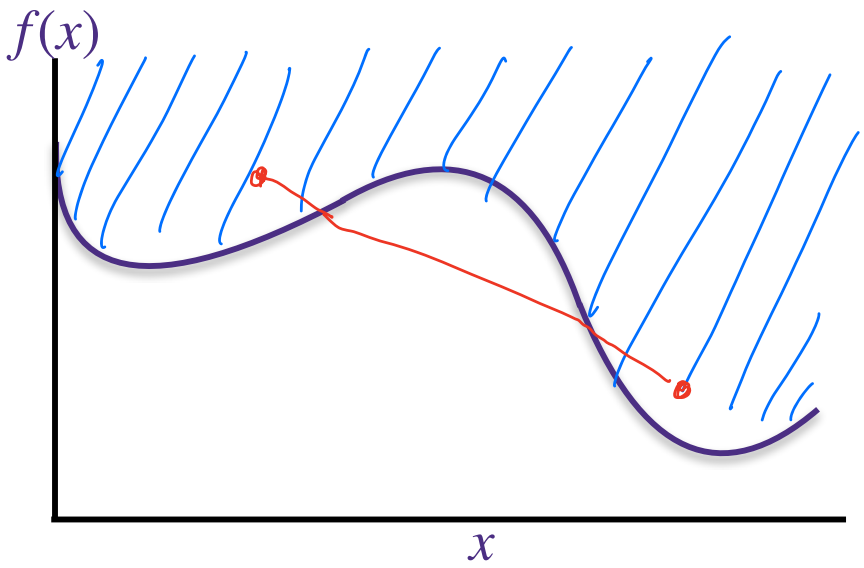
A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in \mathbb{R}^d$ and $\lambda \in [0,1]$

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if the set $\{(x,t) \in \mathbb{R}^{d+1} : f(x) \leq t\}$ is convex

Graph of f id defined as $\{(x,t) : f(x) = t\}$

Epigraph of f is defined as $\{(x,t) : f(x) \leq t\} \subset \mathbb{R}^{d+1}$



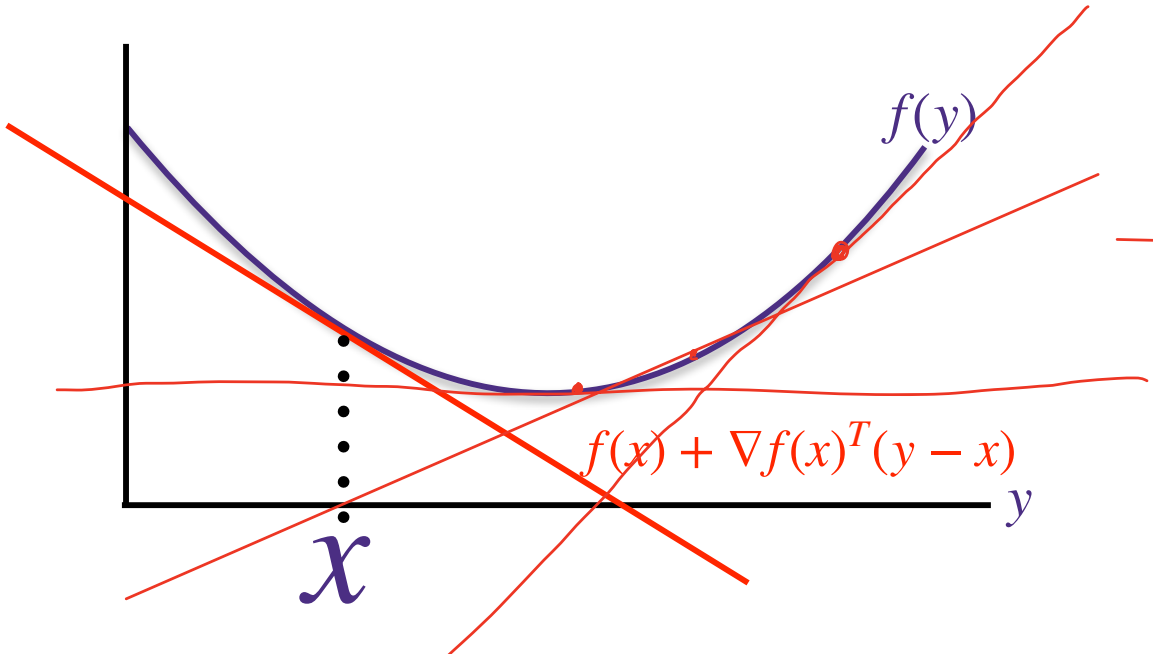


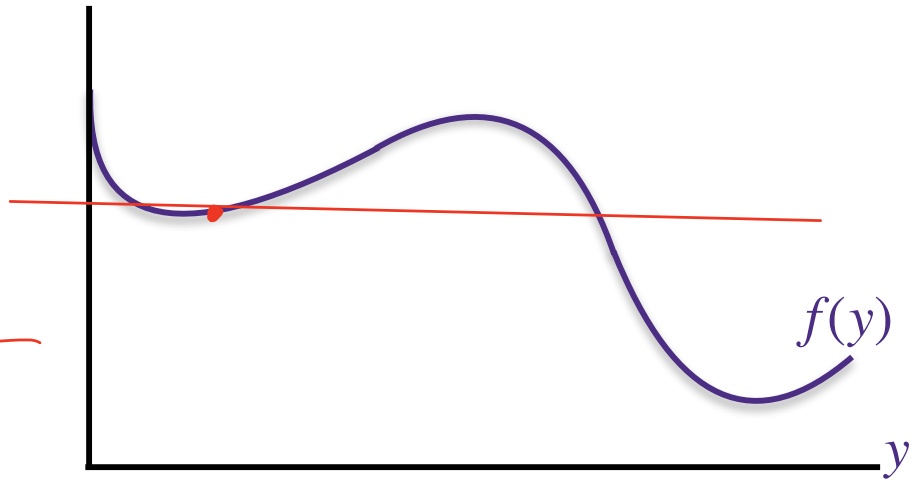
More definitions of convexity

A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if the set $\{(x,t) \in \mathbb{R}^{d+1} : f(x) \leq t\}$ is convex

A function $f : \mathbb{R}^d \to \mathbb{R}$ that is differentiable everywhere is convex if $f(y) \geq f(x) + \nabla f(x)^\top (y - x)$ for all $x, y \in dom(f)$

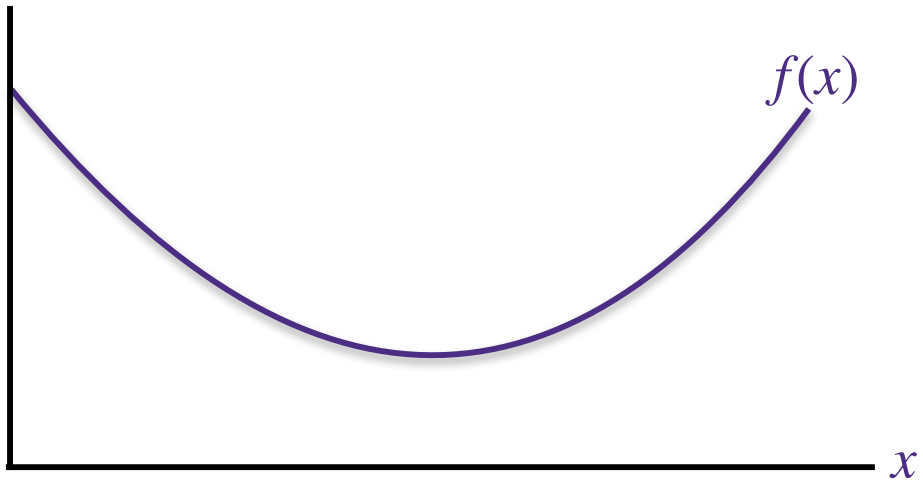


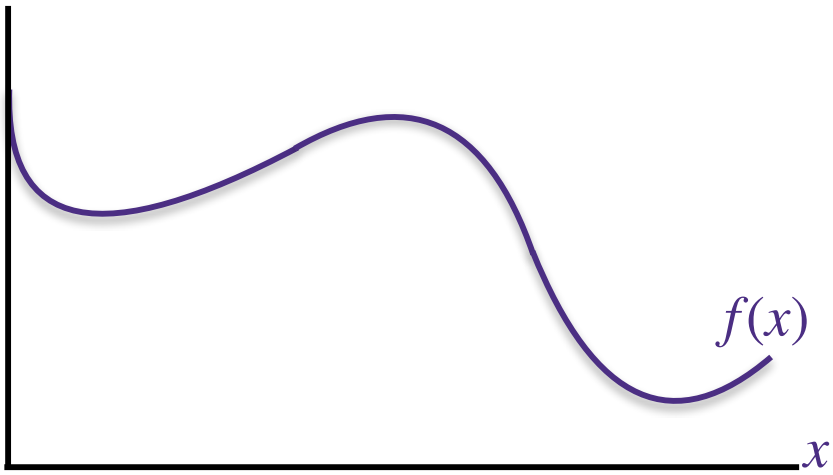


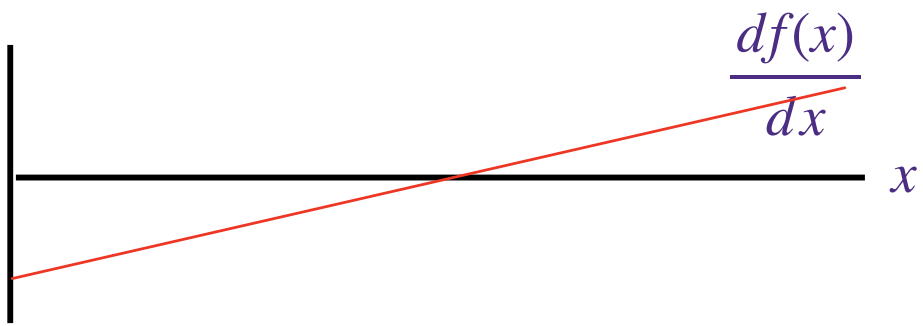
More definitions of convexity

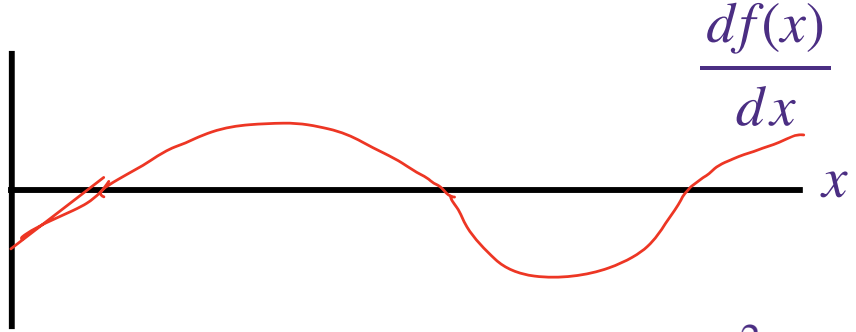
$$[\nabla^2 f(x)]_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}$$

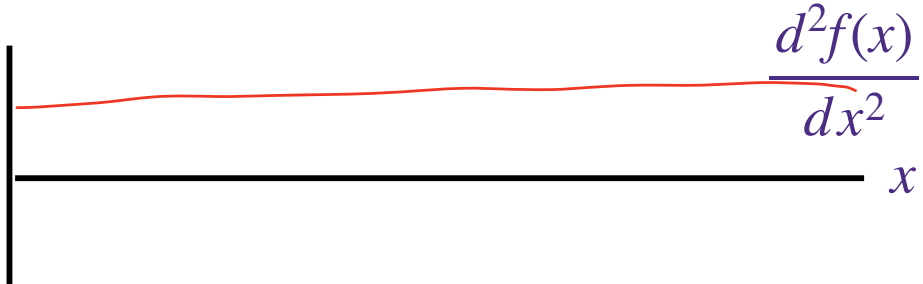
A function $f : \mathbb{R}^d \to \mathbb{R}$ that is twice-differentiable everywhere is convex if $\nabla^2 f(x) \succeq 0$ for all $x \in dom(f)$

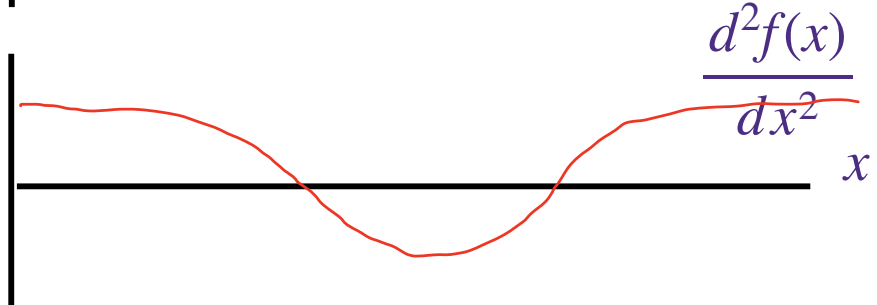










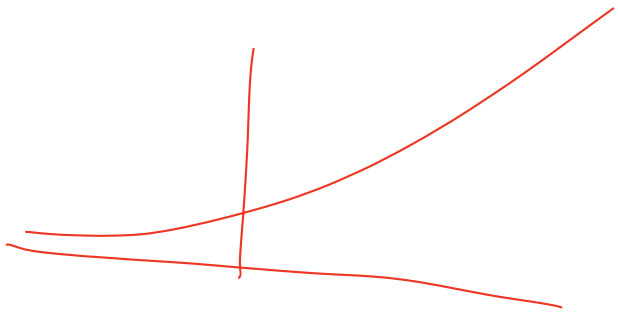


We say a matrix A is positive semidefinite (PSD) if

$x^T A x \geq 0 \quad \forall x$ and is notated $A \succeq 0$.

ex.

$$f(x) = e^{3x^T w}$$



$$\nabla f(x) = e^{3x^T w} \cdot 3w$$

$$\nabla^2 f(x) = e^{3x^T w} \cdot 9 w w^T$$

$$[\nabla f^2(w)]_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j} = \frac{\partial}{\partial x_j} [\nabla f(x)]_i$$

If $\nabla^2 f(x) \succeq 0$ then $z^T \nabla^2 f(x) z \geq 0 \quad \forall z$

$$z^T \nabla^2 f(x) z = 9 e^{3x^T w} \cdot z^T w \cdot w^T z$$

$$= 9 e^{3x^T w} (z^T w)^2$$

$$\geq 0 \quad \forall z.$$

$$f(x) = e^{3x^T w} \qquad g(y) = e^y$$

$$= g(3x^T w)$$

$$\nabla f(x) = \left. \frac{\partial g(y)}{\partial y} \right|_{y = 3x^T w} \cdot \nabla (3x^T w) = e^{3x^T w} 3w$$

More definitions of convexity

A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in \mathbb{R}^d$ and $\lambda \in [0,1]$

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if the set $\{(x,t) \in \mathbb{R}^{d+1} : f(x) \leq t\}$ is convex

A function $f : \mathbb{R}^d \to \mathbb{R}$ that is differentiable everywhere is convex if $f(y) \geq f(x) + \nabla f(x)^\top (y - x)$ for all $x, y \in dom(f)$

A function $f : \mathbb{R}^d \to \mathbb{R}$ that is twice-differentiable everywhere is convex if $\nabla^2 f(x) \succeq 0$ for all $x \in dom(f)$

Why do we care about convexity?

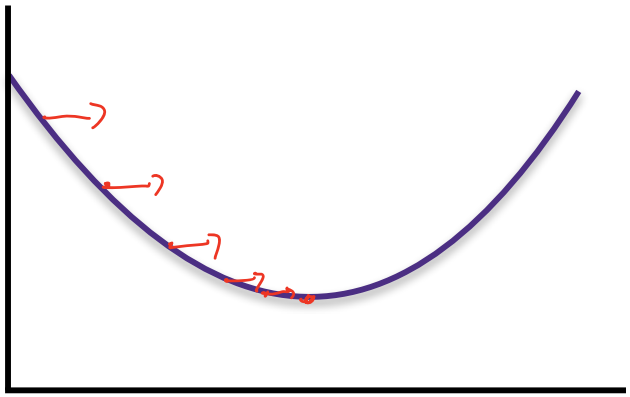
x_* is a local minima
$\nabla f(x_*) = 0$

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

Convex functions

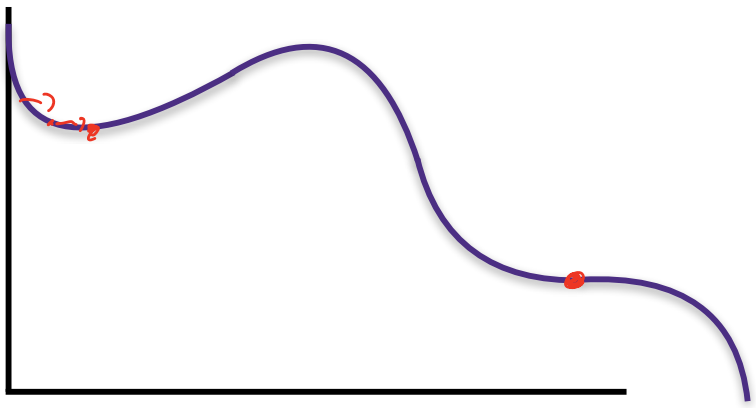
- All local minima are global minima
- Efficient to optimize (e.g., gradient descent)

Convex Function



We only need to find a point with $\nabla f(x) = 0$, which for convex functions implies that it is a local minima and a global minima

Non-convex Function



For non-convex functions, a stationary point with $\nabla f(x) = 0$ could be a local minima, a local maxima, or a saddle point

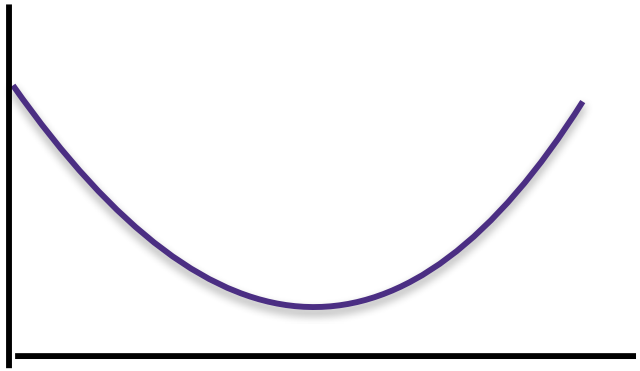
Gradient Descent on $\min_w f(w)$

Initialize: $w_0 = 0$

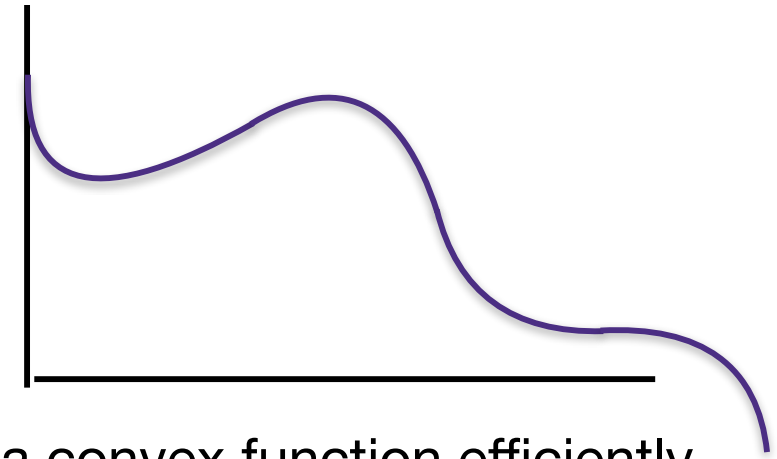
for $t = 1, 2, \ldots$

$$w_{t+1} = w_t - \eta \nabla f(w_t)$$

Convex Function

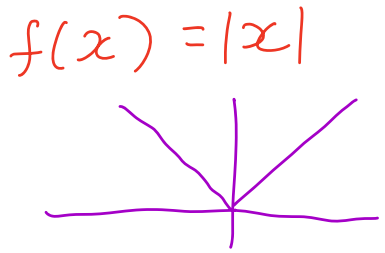


Non-convex Function



- Strength: Can find global minima of a convex function efficiently
- Weakness: Can only be applied to smooth functions
  - i.e., functions that is differentiable everywhere,
  - otherwise $\nabla f(x)$ is not defined and gradient descent cannot be applied

Sub-Gradient



at $x=0$

$\{ g : |y| = f(y) \geq f(0) + g^T y \} = [-1, 1]$

$= g \cdot y$

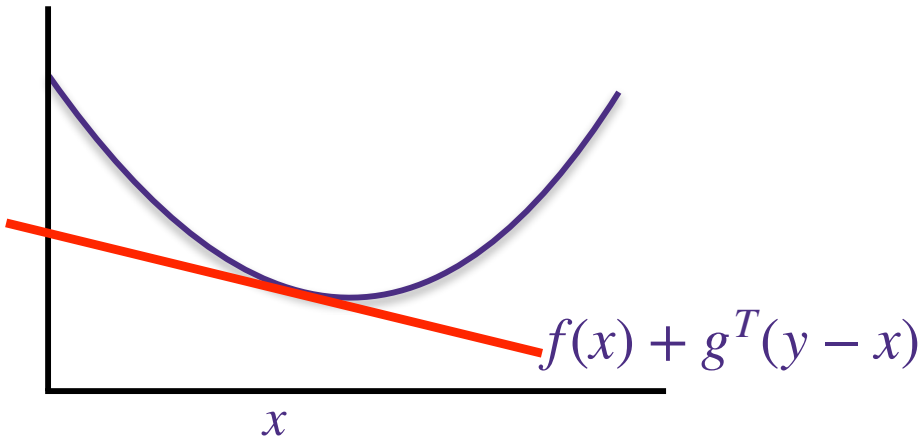
$|y| \geq f(0) + g\,y$

Definition: a function is **non-smooth** if it is not differentiable everywhere

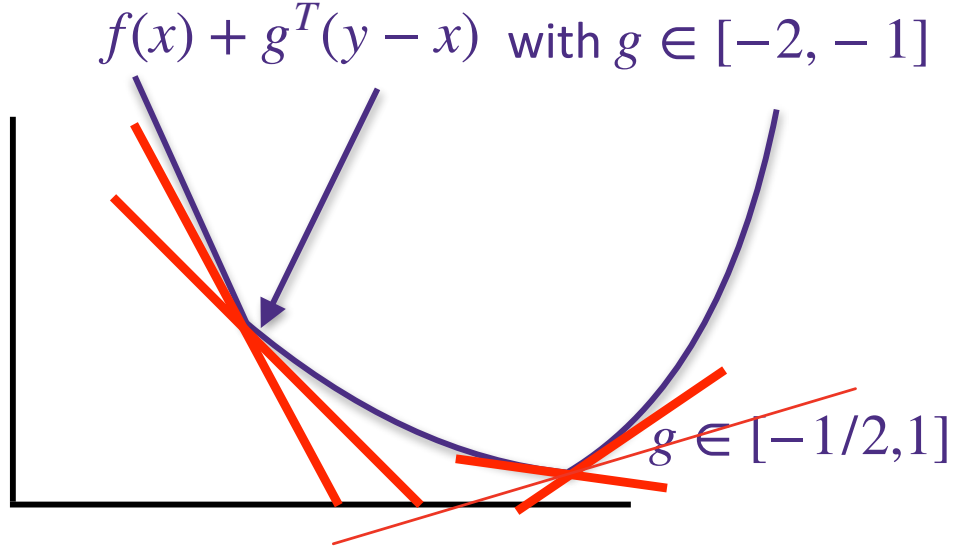
Definition: a vector $g \in \mathbb{R}^d$ is a **sub-gradient** at x if it satisfies

$$f(y) \geq f(x) + g^T(y - x) \text{ for all } y \in \mathbb{R}^d$$

Smooth Convex Function



Non-smooth Convex Function



- for smooth convex functions,
  - gradient is the unique sub-gradient, and
  - the global minimum is achieved at points where gradient is zero

- for non-smooth convex functions,
  - the minimum is achieved at points where sub-gradient set includes the zero vector

Sub-Gradient Descent for non-smooth functions

$\sum_t \eta_t \to \infty \qquad \sum_t \eta_t^2 < \infty$

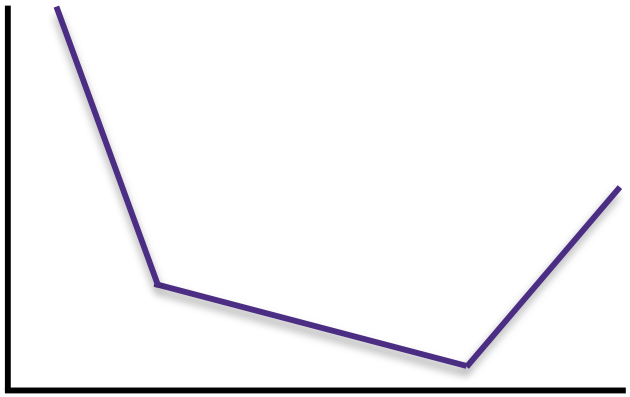
Initialize: $w_0 = 0$

for $t = 1, 2, \ldots$

Find any g_t such that $f(y) \geq f(w_t) + g_t^\top (y - w_t)$

$w_{t+1} \leftarrow w_t - \eta_t g_t$

Convex Function



Non-convex Function

- Strength: finds global minima for **non-smooth convex functions**
- Weakness: it is slower than gradient descent on convex smooth functions, because the gradient do not get smaller near the global minima
  - Instead of last iterate w_t, we use the best one we saw in all iterates
  - The stepsize needs to decrease with t

Optimization

- You can always run gradient descent whether f is convex or not. But you only have guarantees if f is convex
- Many bells and whistles can be added onto gradient descent such as momentum and dimension-specific step-sizes (Nesterov, Adagrad, ADAM, etc.)

Questions?

Stochastic Gradient Descent

Machine Learning Problems

$$\ell_i(w) = (y_i - w^T x_i)^2$$

- **Given data:**

$$\{(x_i, y_i)\}_{i=1}^n \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$

- **Learning a model's parameters:** $\frac{1}{n}\sum_{i=1}^{n} \ell_i(w)$

Gradient Descent:

$$w_{t+1} = w_t - \eta \nabla_w \left(\frac{1}{n} \sum_{i=1}^{n} \ell_i(w) \right) \Big|_{w=w_t}$$

Machine Learning Problems

- **Given data:**

$$\{(x_i, y_i)\}_{i=1}^n \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$

- **Learning a model's parameters:** $\sum_{i=1}^{n} \ell_i(w)$

Gradient Descent:

$$w_{t+1} = w_t - \eta \nabla_w \left(\frac{1}{n} \sum_{i=1}^{n} \ell_i(w) \right) \Big|_{w=w_t}$$

Stochastic Gradient Descent:

$$w_{t+1} = w_t - \eta \nabla_w \ell_{I_t}(w) \Big|_{w=w_t}$$

I_t drawn uniform at random from $\{1, \ldots, n\}$

$$\mathbb{E}[\nabla \ell_{I_t}(w)] = \sum_{i=1}^{n} P(I_t = i) \nabla \ell_i(w) = \frac{1}{n} \sum_{i=1}^{n} \nabla \ell_i(w) = \nabla \left(\frac{1}{n} \sum_{i=1}^{n} \ell_i(w) \right)$$

Machine Learning Problems

- **Learning a model's parameters:** $\sum_{i=1}^{n} \ell_i(w)$

Stochastic Gradient Descent:

$$w_{t+1} = w_t - \eta \nabla_w \ell_{I_t}(w)\Big|_{w=w_t}$$

I_t drawn uniform at random from $\{1, \ldots, n\}$

$$\mathbb{E}[\nabla \ell_{I_t}(w)] = \nabla \ell(w)$$

Stochastic Gradient Descent

$$w_* = \operatorname*{argmin}_w \frac{1}{n}\sum_{i=1}^{n} \ell_i(w)$$

Theorem

Let $$w_{t+1} = w_t - \eta \nabla_w \ell_{I_t}(w)\Big|_{w=w_t}$$ I_t drawn uniform at random from $\{1,\dots,n\}$ so that

$$\mathbb{E}\big[\nabla \ell_{I_t}(w)\big] = \frac{1}{n}\sum_{i=1}^{n} \nabla \ell_i(w) =: \nabla \ell(w)$$

If $\|w_* - w_0\|_2^2 \le R$ and $\sup_w \max_i \|\nabla \ell_i(w)\|_2 \le G$ then

$$\mathbb{E}[\ell(\bar{w}) - \ell(w_*)] \le \frac{R}{2T\eta} + \frac{\eta G}{2} \le \sqrt{\frac{RG}{T}} \qquad \eta = \sqrt{\frac{R}{GT}}$$

$$\bar{w} = \frac{1}{T}\sum_{t=1}^{T} w_t$$

(In practice use last iterate)

Stochastic Gradient Descent

Proof

$$\mathbb{E}[||w_{t+1} - w_*||_2^2] = \mathbb{E}[||w_t - \eta \nabla \ell_{I_t}(w_t) - w_*||_2^2]$$

Stochastic Gradient Descent

Proof

$$\mathbb{E}[||w_{t+1} - w_*||_2^2] = \mathbb{E}[||w_t - \eta \nabla \ell_{I_t}(w_t) - w_*||_2^2]$$

$$= \mathbb{E}[||w_t - w_*||_2^2] - 2\eta \mathbb{E}[\nabla \ell_{I_t}(w_t)^T (w_t - w_*)] + \eta^2 \mathbb{E}[||\nabla \ell_{I_t}(w_t)||_2^2]$$

$$\leq \mathbb{E}[||w_t - w_*||_2^2] - 2\eta \mathbb{E}[\ell(w_t) - \ell(w_*)] + \eta^2 G$$

$$\mathbb{E}[\nabla \ell_{I_t}(w_t)^T (w_t - w_*)] = \mathbb{E}\Big[\mathbb{E}[\nabla \ell_{I_t}(w_t)^T (w_t - w_*) | I_1, w_1, \ldots, I_{t-1}, w_{t-1}]\Big]$$

$$= \mathbb{E}\Big[\nabla \ell(w_t)^T (w_t - w_*)\Big]$$

$$\geq \mathbb{E}\Big[\ell(w_t) - \ell(w_*)\Big]$$

$$\sum_{t=1}^{T} \mathbb{E}[\ell(w_t) - \ell(w_*)] \leq \frac{1}{2\eta} \left(\mathbb{E}[||w_1 - w_*||_2^2] - \mathbb{E}[||w_{T+1} - w_*||_2^2] + T\eta^2 G\right)$$

$$\leq \frac{R}{2\eta} + \frac{T\eta G}{2}$$

Stochastic Gradient Descent

Proof

Jensen's inequality:
For any random $Z \in \mathbb{R}^d$ and convex function $\phi : \mathbb{R}^d \to \mathbb{R}$, $\phi(\mathbb{E}[Z]) \leq \mathbb{E}[\phi(Z)]$

$$\mathbb{E}[\ell(\bar{w}) - \ell(w_*)] \leq \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[\ell(w_t) - \ell(w_*)] \qquad \bar{w} = \frac{1}{T} \sum_{t=1}^{T} w_t$$

Mini-batch SGD

- Instead of one iterate, average B stochastic gradient together

- Advantages:
  - Smaller variance: the variance of the stochastic gradient is smaller by a factor of $1/\sqrt{B}$
  - Parallelization: each gradient in the mini-batch can be computed in parallel

- If you have regularizer, $\frac{1}{n}\sum_{i=1}^{n} \ell_i(w) + r(w)$, then update with the stochastic gradient of the loss and gradient of the regularizer

Questions?

Memory vs compute

```python
# generate some nonsense data for an example
X = np.random.randn(n,d)
y = np.random.randn(n)
```

```python
# generate the random features
G = np.random.randn(p, d)*np.sqrt(.1)
b = np.random.rand(p)*2*np.pi
```

```python
# construct HTH
HTH = np.zeros((p,p))
HTy = np.zeros(p)
for i in range(n):
    hi = np.dot(X[i,:], G.T)+b
    HTH += np.outer(hi, hi)
    HTy += y[i]*hi
    if i % 1000==0: print(i)
```

```python
H = np.dot(X, G.T) + b.T
HTH = np.dot(H.T, H)
HTy = np.dot(H.T, y)
```

```python
# construct HTH
HTH = np.zeros((p,p))
HTy = np.zeros(p)
block = p
for i in range(int(np.ceil(n/block))+1):
    Hi = np.dot(X[i*block:min(n,(i+1)*block),:], G.T)+b
    HTH += np.dot(Hi.T, Hi)
    HTy += np.dot(Hi.T, y[i*block:min(n,(i+1)*block)])
```

```python
w = np.linalg.solve(HTH + lam*np.eye(p), HTy)
```

1 float in NumPy $= 8$ bytes
$10^6 \approx 2^{20}$ bytes $= 1$ MB
$10^9 \approx 2^{30}$ bytes $= 1$ GB

For each block compute the memory required in terms of n, p, d.

If d << p << n, what is the most memory efficient program (blue, green, red)?

If you have unlimited memory, what do you think is the fastest program?