# CMSC 510 – L03
# Regularization Methods for Machine Learning

Instructor:

Dr. Tom Arodz

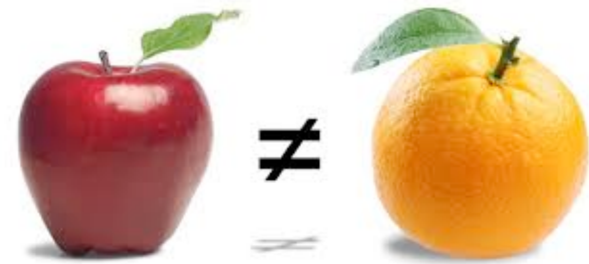
# Recap: Linear Models

- A very simple but often powerful family of machine learning models (classifiers or regressors)

- "predicted y" $=\langle w,x\rangle+b$
  $= w^T x + b$
  $= w_1 x_1 + w_2 x_2 + b$

- How to find w, b that leads to lowest MSE?
  - We need to talk about optimization methods
    - Plug in known values of features (x) and targets (y)
    - Treat MSE as a function MSE(w,b) - that is, not a function of x anymore!
    - Find w,b that leads to minimum of the function MSE(w,b)

# Recap: iterative learning

- Apple vs Orange

- Let:
  - **x** denote object's **color** (wavelength in nm)
  - **y** denote object's **class** (-1 = orange, 1 = apple)
- Prediction will be made by evaluating a function:
  - **f(x)=sign(x – b)** - it returns either -1 or 1 (or 0: tough to predict)
- Example of an algorithm:
  - **Set initial values of b (0, or random, or a guess)**
  - **Loop:**
    - **Present a sample $x_i$ and predict $f(x_i)=\text{sign}(x_i - b)$**
    - **Compare true class $y_i$ with predicted class $f(x_i)$**
    - **If prediction is right, go to next sample (i=i+1)**
    - **If prediction is wrong, update b**
      - **In a way that would make it more likely to get correct prediction for the current sample**
      - **if $y_i = 1$, $f(x_i)=-1$, what should we do? Decrease b to increase f**
      - **if $y_i = -1$, $f(x_i)=1$, what should we do? Increase b to decrease f**

# Recap: General Scheme

Any method based on an oracle operates by

performing a series of queries to oracle O,

acting based on information I obtained from the queries

Any oracle-based optimization method M is an iterative scheme:

0. **Create initial solution $x_0$, set k=0, $I_{-1}$=empty**
1. **Call oracle O at $x_k$**
2. **Update accumulated information $I_k = I_{k-1} + (x_k, O(x_k))$**
3. **Apply internal rules of method M to $I_k$, to generate $x_{k+1}$**
4. **Calculate error ε, check stopping criterion based on ε**

   **If criterion met, exit with the solution,**

   **if no, k=k+1, go back to step 1.**

Two measures of computational complexity of M on problem P:

Analytical complexity A(M,ε):

number of calls to oracle O required to get error at most ε

Arithmetic complexity:

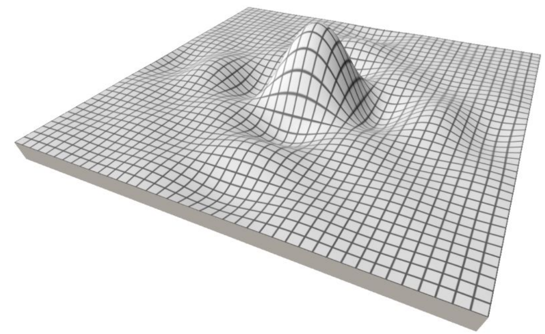
number of all operations (typically proportional to analytical complexity)

# Recap: simple grid search

Simple method $M_{grid}$:

Generate $(p+1)^n$ points on a n-dimensional grid in [0,1]

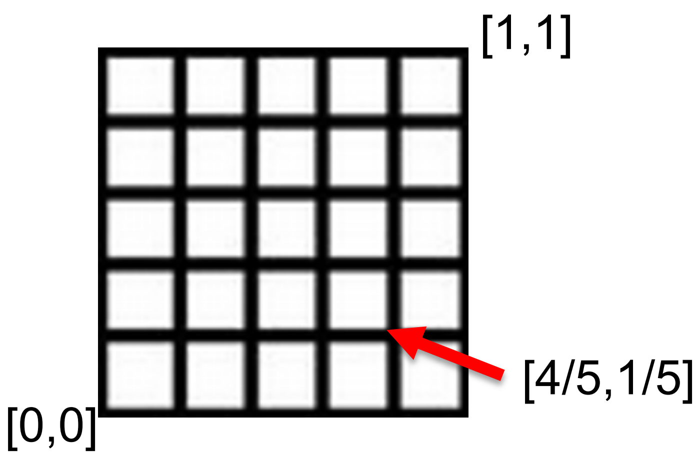
$x(i_1,i_2,\ldots,i_n)=(i_1/p,\ i_2/p,\ \ldots,\ i_n/p),\quad i_k\in\{0,1,\ldots,p\}$



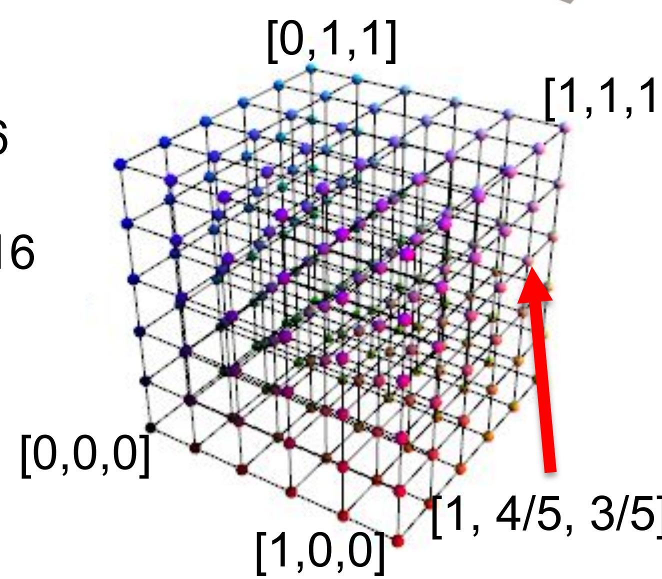
Call oracle $(p+1)^n$ times,

get values of f(x) for each grid point x

Return the grid point x' with lowest f




- p=5,
- If n=2, we need $6^2=36$ points in the grid
- If n=3, we need $6^3=216$ points in the grid
- each cell has side length of 1/5



# Optimization

Problem: Find minimum $f(x)$ s.t. $x \in B_n$, assuming f is Lipschitz contin.

$| f(x) - f(y) | <= L \| x-y \|_\infty$ for all $x,y \in B_n,$ where: $\| x \|_\infty = \max_{1<=i<=n} |x^{(i)}|$

What is the highest error we can get with method $M_{grid}$?

Let $f^*=f(x^*)$ be the value of f in the real global minimum $x^*$

$e(x')= | f^* - f(x') |$

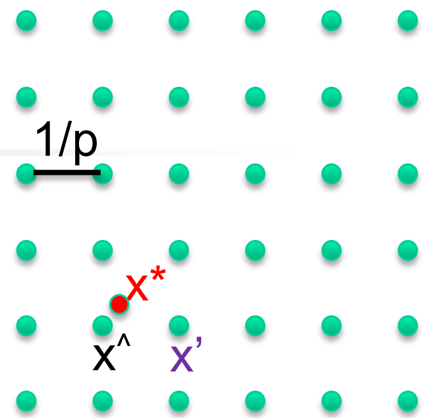
We can prove that **$e(x')<=L/2p$** (next slide)

For given $\varepsilon$, if we want certainty we get $x'$ with error below $\varepsilon$, $e(x') <= \varepsilon$, we need p that gives $\varepsilon=L/2p$,
i.e., $p>=\text{floor}(L/2\varepsilon)+1$

$A(M,\varepsilon)$: number of calls to O required to get error at most $\varepsilon$

$$A(M_{grid},\varepsilon) = (p+1)^n >= ( \text{floor}(L/2\varepsilon) + 2 )^n = O( (L/\varepsilon)^n )$$

# Optimization




Simple method $M_{grid}$:

Generate $(p+1)^n$ point on a n-dimensional grid in [0,1], return lowest grid point $x'$

What is the worst error we can get with this method?

Let $f^*=f(x^*)$ where $x^*$ is global minimum: $e(x') = | f^* - f(x') |$

**We can prove that $e(x')<=L/2p$**

$x^*$ must be in one of the grid cells, cell side length=1/p

there is at least one corner $x^\wedge$ of that cell with

$$|| x^* - x^\wedge ||_\infty <= ½*1/p$$

Thus, from Lipschitz, we get

$$| f(x^*) - f(x^\wedge) | <= L \; || x^* - x^\wedge ||_\infty \;\; <= L*½*1/p = L/2p$$

We analyzed all grid points, including $x^\wedge$, and found $x'$ as minimum, so $\quad f(x^*) <= f(x') <= f(x^\wedge)$

Thus: $e(x') = f(x^*) - f(x') \;\; <= f(x^*) - f(x^\wedge) <= L/2p$

# Optimization

Problem: Find minimum f(x) s.t. $x \in B_n$, assuming f is Lipschitz cont.

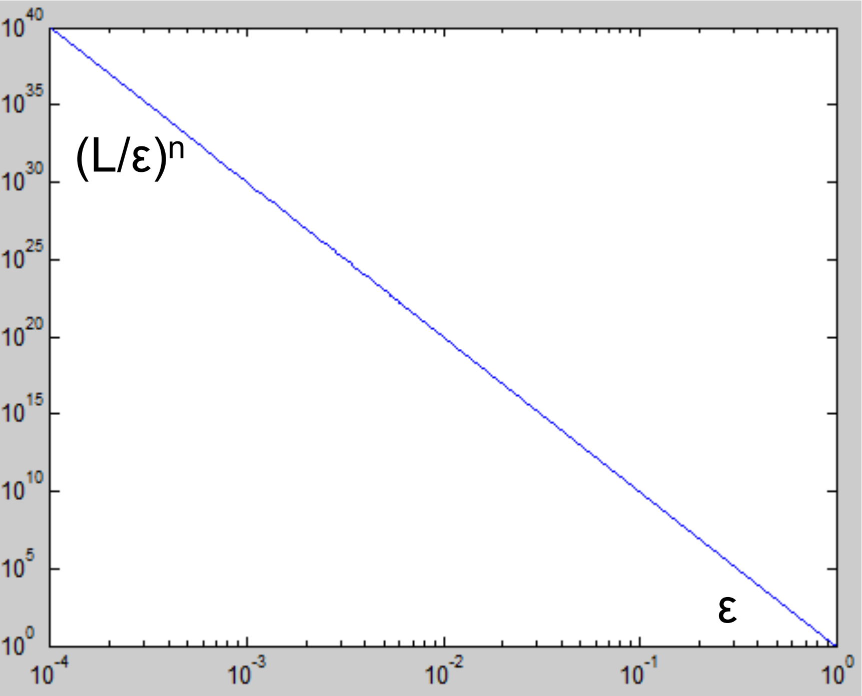
$| f(x) - f(y) | <= L \, || x-y ||_\infty$ for all $x,y \in B_n$, where: $|| x ||_\infty = \max_{1<=i<=n} |x^{(i)}|$

Simple method $M_{grid}$: Test $(p+1)^n$ points on a n-dimensional grid

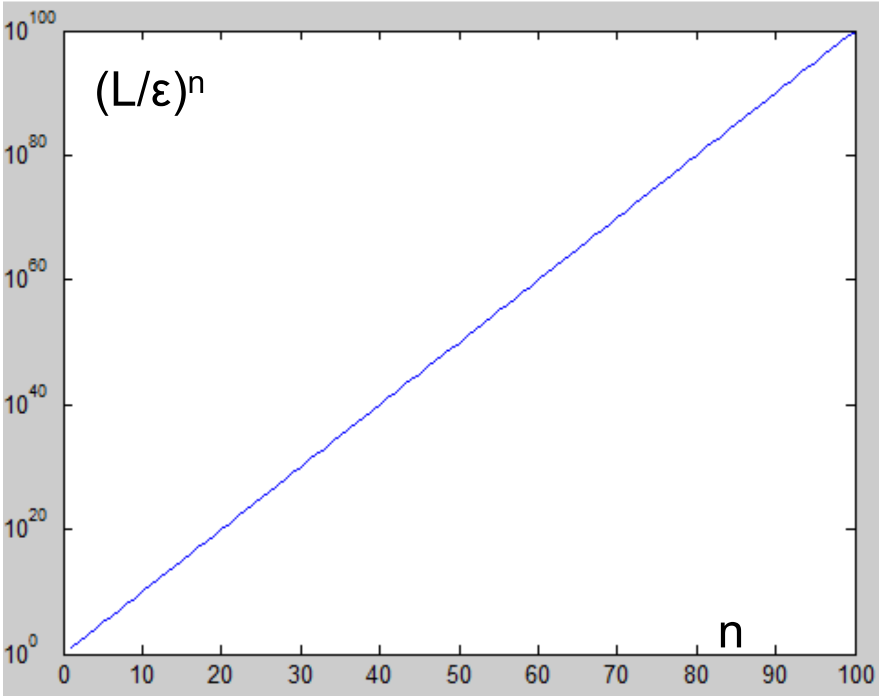
$$A(M_{grid},\varepsilon) = ( \text{floor}(L/2\varepsilon) + 2 )^n >= (L/\varepsilon)^n$$

$A(M_{grid},\varepsilon)$: number of calls to oracle O required to get error at most $\varepsilon$

Let L=1, n = 10, what is $(L/\varepsilon)^n$ for different $\varepsilon$?




Let L=1, $\varepsilon$ = 0.1, what is $(L/\varepsilon)^n$ for different n?



# Optimization

Problem: Find minimum $f(x)$ s.t. $x \in B_n$, assuming f is Lipschitz cont.

$| f(x) - f(y) | <= L \| x-y \|_\infty$ for all $x,y \in B_{n,}$ where: $\| x \|_\infty = \max_{1<=i<=n} |x^{(i)}|$

$M_{grid}$ is a very simple method, maybe we can to better?

Better than this: $A(M_{grid},\varepsilon) = ( \text{floor}(L/2\varepsilon) + 2 )^n >= O((L/\varepsilon)^n)$

$M_{grid}$ shows an upper bound on the analytical complexity of solving arbitrary Lipschitz-continuous minimization problems

What is the lower bound?

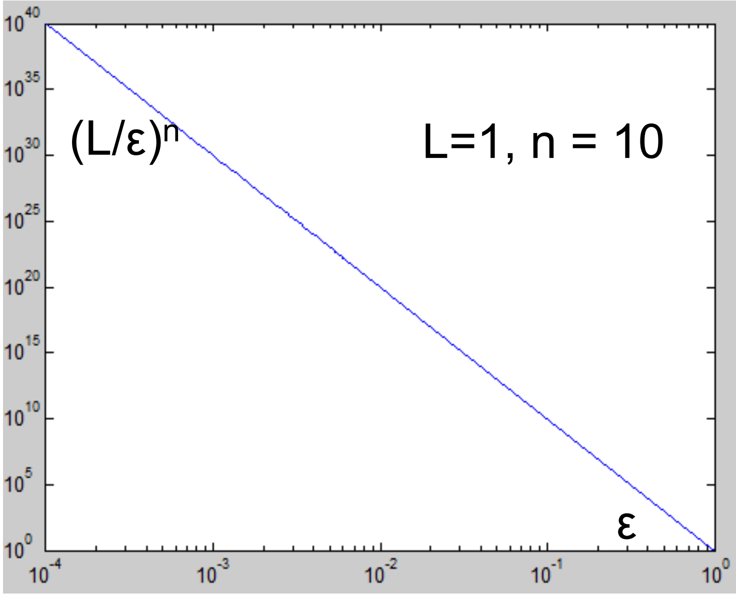
That is, the lowest complexity $A(M_{best},\varepsilon)$ we can get for any method working with a black-box oracle?

# Optimization

Problem: Find minimum $f(x)$ s.t. $x \in B_n$, assuming f is Lipschitz cont. with constant L

$| f(x) - f(y) | <= L \, || x-y ||_\infty$ for all $x,y \in B_n$, where: $|| x ||_\infty = \max_{1<=i<=n} |x^{(i)}|$

We will prove that for any ε small enough ( ε < L/2 ) the analytical complexity is $A(M_{any},\varepsilon) >= (\text{floor}(L/2\varepsilon))^n$ for all methods that use oracle that returns f(x) for query x




This is practically the same as $A(M_{grid},\varepsilon) = ( \text{floor}(L/2\varepsilon) + 2 )^n$ we had for the simple grid method

Proof: involves a *resisting* oracle, an oracle that chooses the worst problem on the fly, based on the queries from method M

# Optimization

Problem: Find minimum $f(x)$ s.t. $x \in B_n$, assuming f is Lipschitz cont. with constant L

$| f(x) - f(y) | \le L \, \| x-y \|_\infty$ for all $x,y \in B_n$, where: $\| x \|_\infty = \max_{1 \le i \le n} |x^{(i)}|$

## A *resisting* oracle:

- Initially, we don't know anything about the behavior of function f
- So, before the first query, the oracle can choose any function f for us
- After the first answer $f_1=f(x_1)$, the possibilities narrow, only functions with $f(x_1)=f_1$ are possible, but there's still many of them
- Let's say we ask for $f(x_2)$ for $x_2$ that is close to $x_1$ e.g. $\| x_1 - x_2 \|_\infty = d_{12}$
- Oracle cannot return an arbitrary number, it has to be consistent with:

  $| f(x_1) - f(x_2) | \le d_{12} L$ $\qquad\qquad$ $f(x_1)=f_1$
- But the oracle can choose any function that meets the constraints above

## As long as the answers:

- are consistent with previous answers, and
- don't result in an empty set of functions for future answers

the oracle can pick any function f to make search for min. slow

# Optimization

Problem: Find minimum f(x) s.t. $x \in B_n$, assuming f is Lipschitz continuous with constant L

$| f(x) - f(y) | <= L \| x-y \|_\infty$ for all $x,y \in B_{n,}$ where: $\| x \|_\infty = \max_{1<=i<=n} |x^{(i)}|$

We will prove that for any ε small enough ( ε < L/2 ) the analytical complexity is $A(M_{any},\varepsilon) >= (\mathrm{floor}(L/2\varepsilon))^n$

**Let p=floor(L/2ε)>=1.**

**Assume there exists $M_{best}$ that needs $N<p^n$ calls to oracle to get error <=ε no matter what (L-Lipschitz continuous) f we have**

Recipe for the resisting Oracle:

return f(x)=0 for any point for the first $p^n$ queries

In $N<p^n$ steps, the method can find only $x^\sim$ with $f(x^\sim)=0$

# Optimization

Problem: Find minimum $f(x)$ s.t. $x \in B_n$, assuming $f$ is Lipschitz continuous with constant $L$

$| f(x) - f(y) | <= L \, || x-y ||_\infty$ for all $x,y \in B_n$, where: $|| x ||_\infty = \max_{1<=i<=n} |x^{(i)}|$

**Prove:** For any $\varepsilon < L/2$, $A(M_{any},\varepsilon) >= (\text{floor}(L/2\varepsilon))^n$

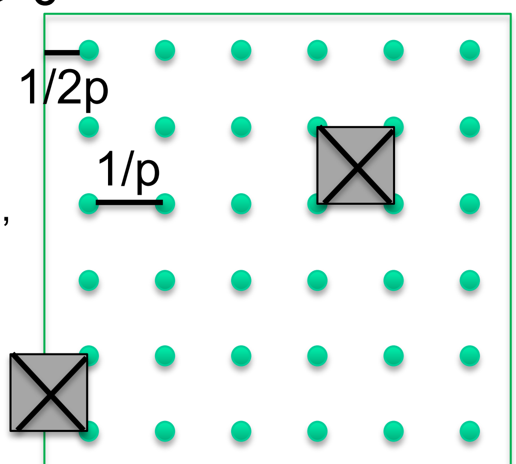
Let $p=\text{floor}(L/2\varepsilon)>=1$ Assume $M_{best}$ needs $N<p^n$ oracle calls to get error $<=\varepsilon$

Resisting Oracle: return $f(x^\sim)=0$ for any point $x^\sim$ in the first $p^n$ queries

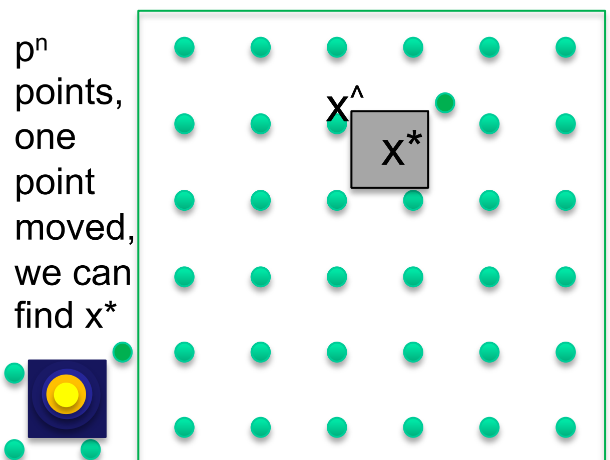
Inside $B_n$, there is n-dimensional box with side $1/p$ and center $x^*$ that doesn't contain any of the $N<p^n$ query points

the closest query $x^\wedge$ is outside , so $> 1/2p$ away: $|| x^*-x^\wedge ||_\infty = 1/2p+\delta$, $\delta>0$

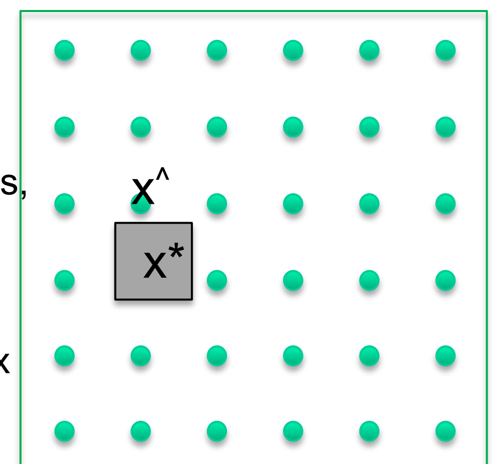
$p^n$ points in grid pattern, no $x^*$ can be found




$p^n$ points, one point moved, we can find $x^*$




less than $p^n$ points, we can find a box with $x^*$




We can show that $f(x^*) = -(\varepsilon+\delta L)$ is not inconsistent with previous oracle answers, the error of method $M_{best}$ is $\varepsilon+\delta L > \varepsilon$

# Optimization

**Assume there exists $M_{best}$**
**that needs $N<p^n$ calls to oracle to get error $e(x') <= \varepsilon$**
**no matter what (L-Lipschitz continuous) f we have**

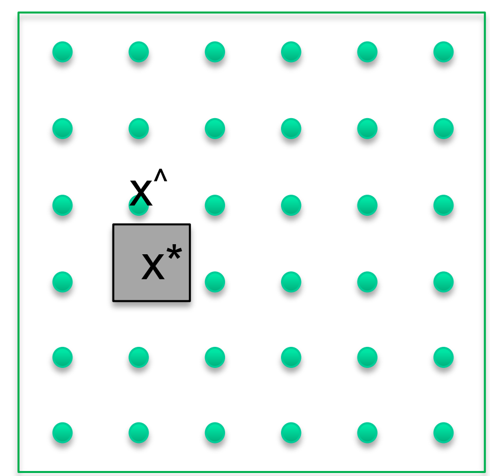
$f(x^{\sim})=0$ for all queried N points

But $f(x^*) = -(\varepsilon+L\delta)$

The box is wide enough for $f(x^*)$ to be more negative than $-\varepsilon$ even though all queries outside of the box have $f(x)=0$

Thus, $e(x') = \varepsilon+L\delta$

and error is not $<= \varepsilon$




Our assumption that **there exists $M_{best}$**
**that needs $N<p^n$ calls to oracle**
**is false**

# Optimization

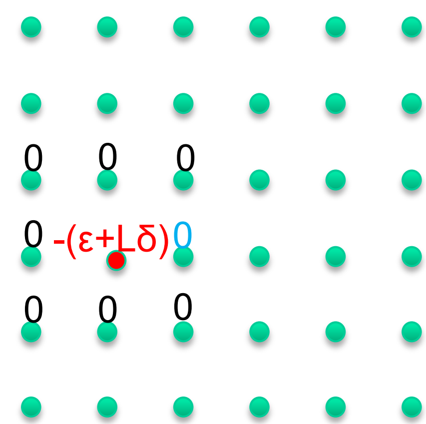
**Assume there exists $M_{best}$ that needs $N<p^n$ calls to oracle to get error $e(x') <= \varepsilon$ no matter what (L-Lipschitz continuous) f we have**

$f(x^{\sim})=0$ for all queried N points

But, inside $B_n$, there is n-dimensional box with side 1/p and center $x^*$ that doesn't contain any of the N queries

the closest query $x^{\wedge}$ is > 1/2p away: $\| x^*-x^{\wedge} \|_\infty = 1/2p+\delta$, $\delta>0$

Let $f(x^*) = -(\varepsilon+L\delta)$. Is this choice ok?




We need to show: $| f(x^*) - f(x^{\wedge}) | <= L \| x^*-x^{\wedge} \|_\infty$

$| f(x^*) - f(x^{\wedge}) | = | f(x^*) - 0| = | f(x^*) | = \varepsilon+L\delta$

$L \| x^*-x^{\wedge} \|_\infty = L(1/2p+\delta) = L/2p+L\delta$

$\varepsilon+L\delta <= L/2p+L\delta$

$\varepsilon <= L/2p$ is it true?

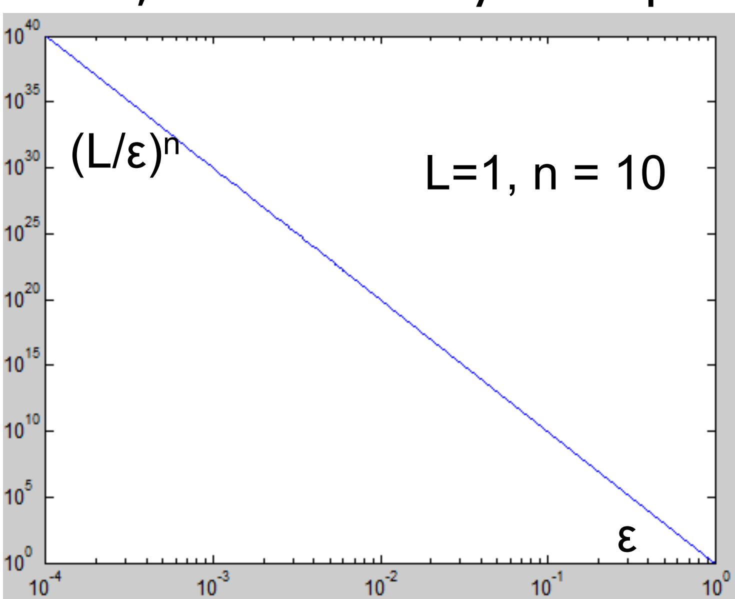
Yes, we defined $p=\text{floor}(L/2\varepsilon)$, that is $p<=L/2\varepsilon$ and $\varepsilon<=L/2p$

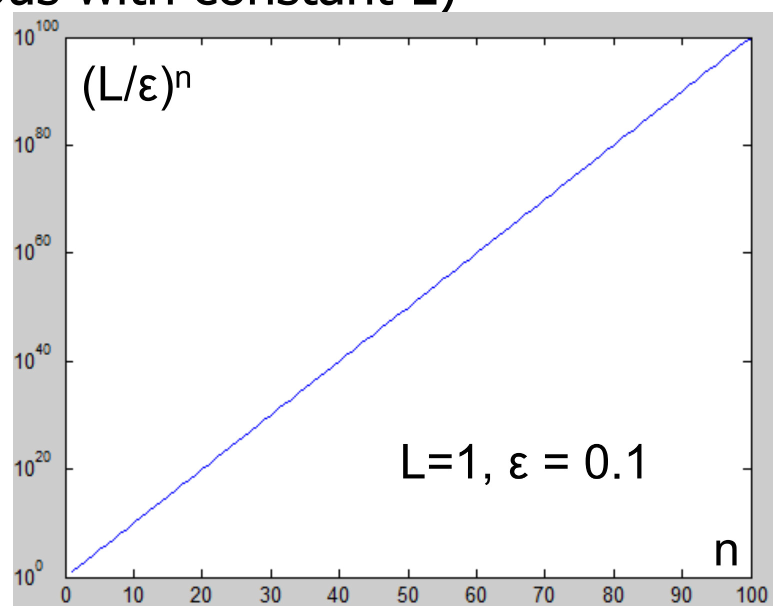
# Optimization

Problem: find minimum f(x) s.t. $x \in B_n$,
assuming f is Lipschitz cont. with constant L

For any ε small enough ( ε < L/2 ) the analytical complexity is
$A(M_{any},\varepsilon) >= (\mathrm{floor}(L/2\varepsilon))^n$
for all methods that use oracle that returns f(x) for query x

Very slow: **minimization of general functions is hard**
(even if we have some bounds on their local fluctuations,
i.e., we know they are Lipschitz continuous with constant L)



# Optimization => machine learning

**Minimization of general functions is hard**

**If we build arbitrary machine learning model, and want to train it to find the best set of parameters, it may take forever**

**Solutions:**

- **Focus on special classes of functions (e.g. linear models: y=ax+b)**

  **or**

- **Aim for "good enough" not best (e.g. deep neural networks)**

# REVIEW: Optimization

Categorization of Oracles:

Zero-order: for x, return f(x)

First-order:

for x, return f(x) and **derivative/gradient** f'(x)

Second-order:

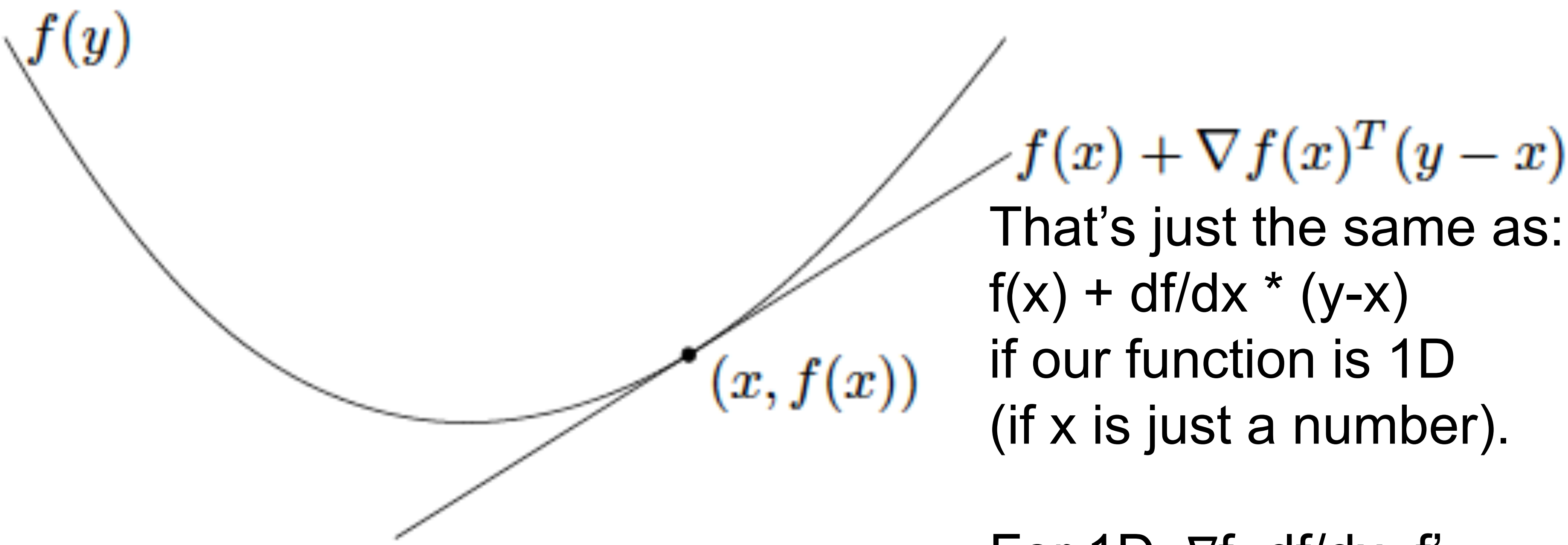
for x, return f(x), first derivative/gradient f'(x), and second derivative/Hessian f''(x)

We will use basic concepts from calculus

# Optimization

First derivative or gradient: denoted $\nabla f$ or $f'$

- gradient of a n-dimensional function $f:R^n \rightarrow R$ is a generalization of derivative of a single-dimensional function $f:R \rightarrow R$




That's just the same as:
f(x) + df/dx * (y-x)
if our function is 1D
(if x is just a number).

For 1D, $\nabla f = df/dx = f'$

# Optimization

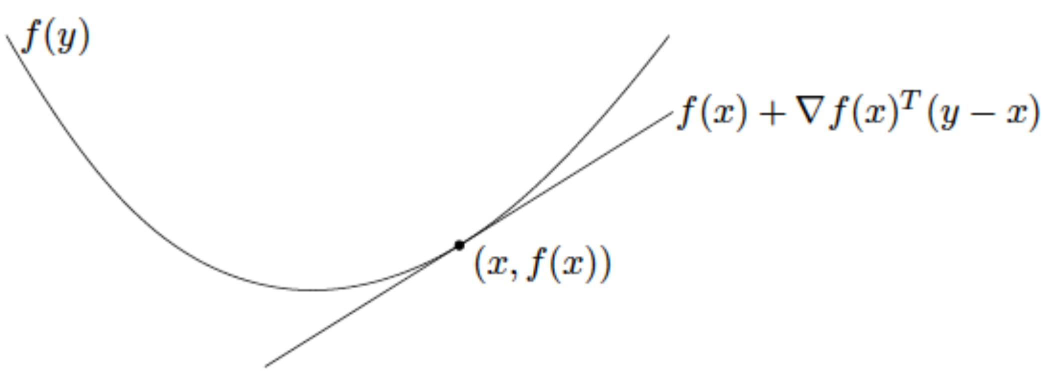
gradient of a n-dimensional function $f: R^n \to R$ is:

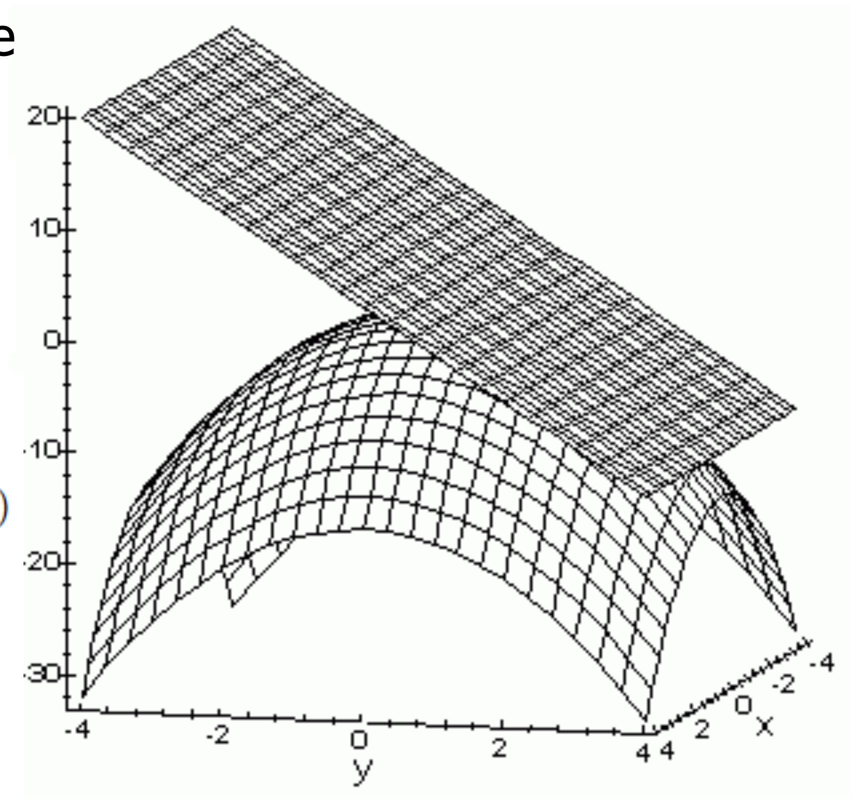
an n-dimensional **vector** of partial derivatives,
pointing (for a given x) in direction of steepest slope of f at that x

- Pointing always in the direction in which the function grows (locally)
- positive value in a vector = f grows (vector points) towards +∞; negative value = f grows towards –∞, vector points towards –∞

Defines a line or plane or hyperplane tangent to f

$$\nabla f(\mathbf{z}) = \left( \frac{\partial}{\partial x_1} f(\mathbf{x})|_{\mathbf{x}=\mathbf{z}}, \ldots, \frac{\partial}{\partial x_n} f(\mathbf{x})|_{\mathbf{x}=\mathbf{z}} \right)$$



# Optimization

$$\nabla f(\mathbf{z}) = \left( \frac{\partial}{\partial x_1} f(\mathbf{x})|_{\mathbf{x}=\mathbf{z}}, \ldots, \frac{\partial}{\partial x_n} f(\mathbf{x})|_{\mathbf{x}=\mathbf{z}} \right)$$

Gradient $\nabla f(x)$ gives a linear approximation of function f at x:

$f(y)=f(x) + \langle\nabla f(x),y-x\rangle + o(||y-x||)$

$o(r)$ for $r>0$: $\lim_{r\to 0} o(r)/r=0$

Just like derivative df/dx gives a linear approximation of a 1D function in the neighborhood of x

$f(y)=f(x) + df/dx\,(y-x) + o(||y-x||)$

**<x,y> represents inner product (dot product) of two vectors x,y**

In 1D, it reduces to simple multiplication

$$\left\langle \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \right\rangle := x^{\mathrm{T}} y = \sum_{i=1}^{n} x_i y_i = x_1 y_1 + \cdots + x_n y_n$$

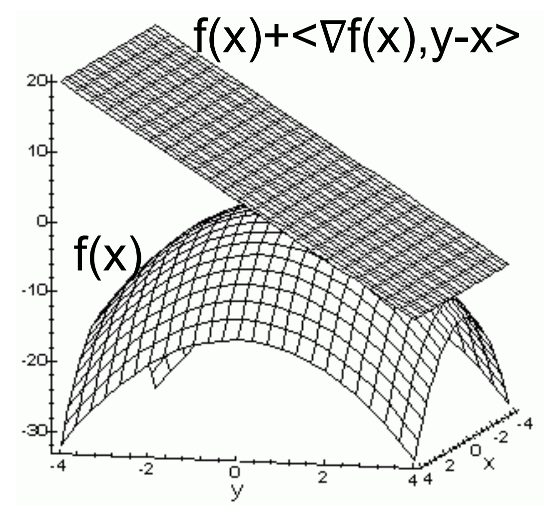
**Dot product: coordinate-wise multiplication**

$\langle x,z+y\rangle=\langle x,z\rangle+\langle x,y\rangle$

$\langle ax,by\rangle=ab\langle x,y\rangle$ for real a,b; in particular $\langle x,0\rangle=0$

$\langle x,x\rangle=||x||^2$ $\quad$ $\langle x,y\rangle=\langle y,x\rangle$

vectors with $\langle x,y\rangle=0$ are called orthogonal

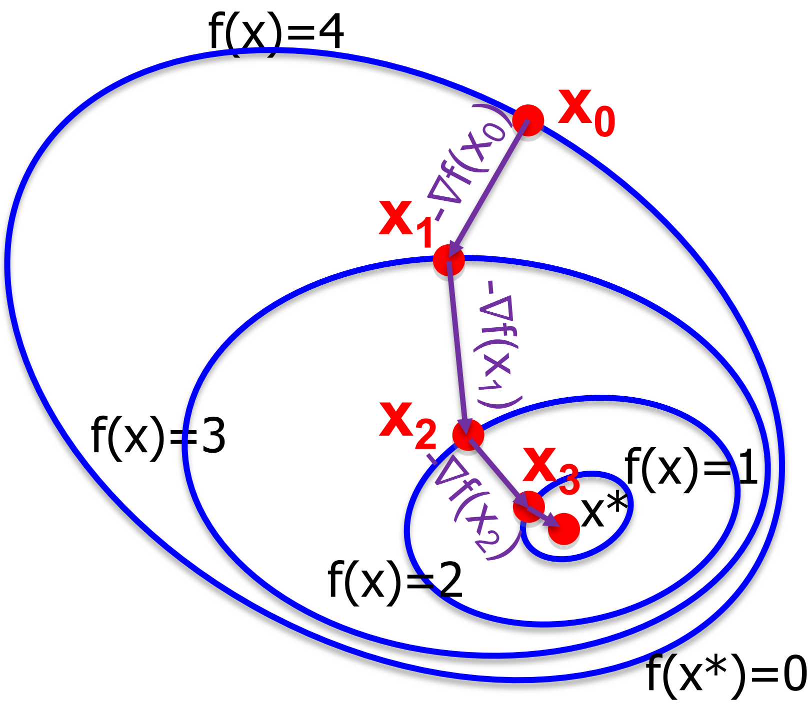


# Gradient descent

Gradient descent:

- We start from $x_0$
- We calculate $x_1 = x_0 - \nabla f(x_0)/L$
- We calculate $x_2 = x_1 - \nabla f(x_1)/L$
- $\mathbf{x_{n+1} = x_n - \nabla f(x_n)/L}$

If we choose L large enough
- g.d. goes down in each step, converging towards
- global minimum (e.g. for convex f)
- or
- local minimum



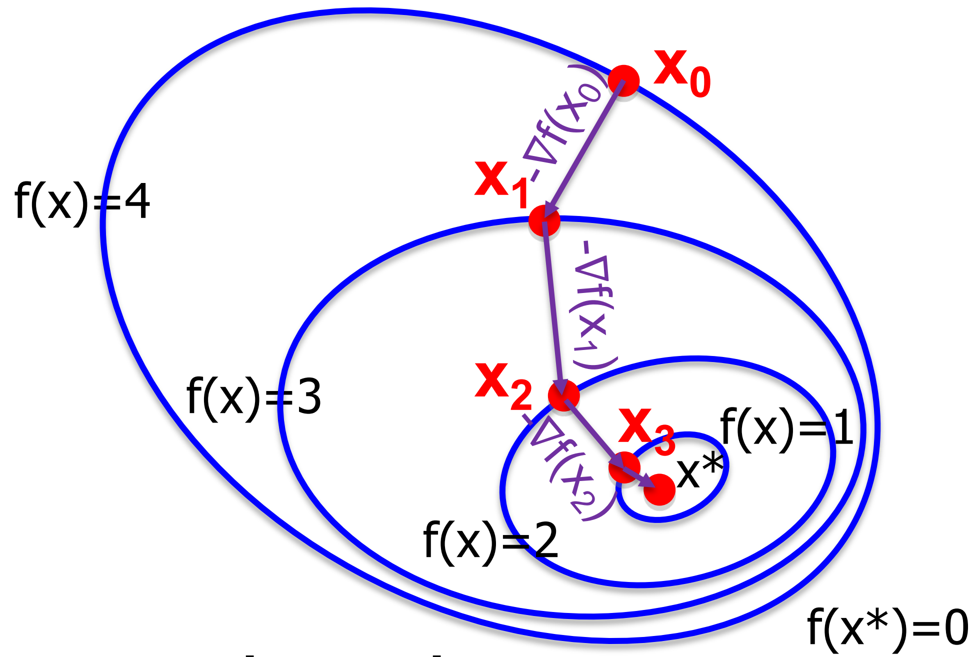
# Convex optimization

We can do infinite number of steps,

each smaller & smaller

each closer to the global optimum

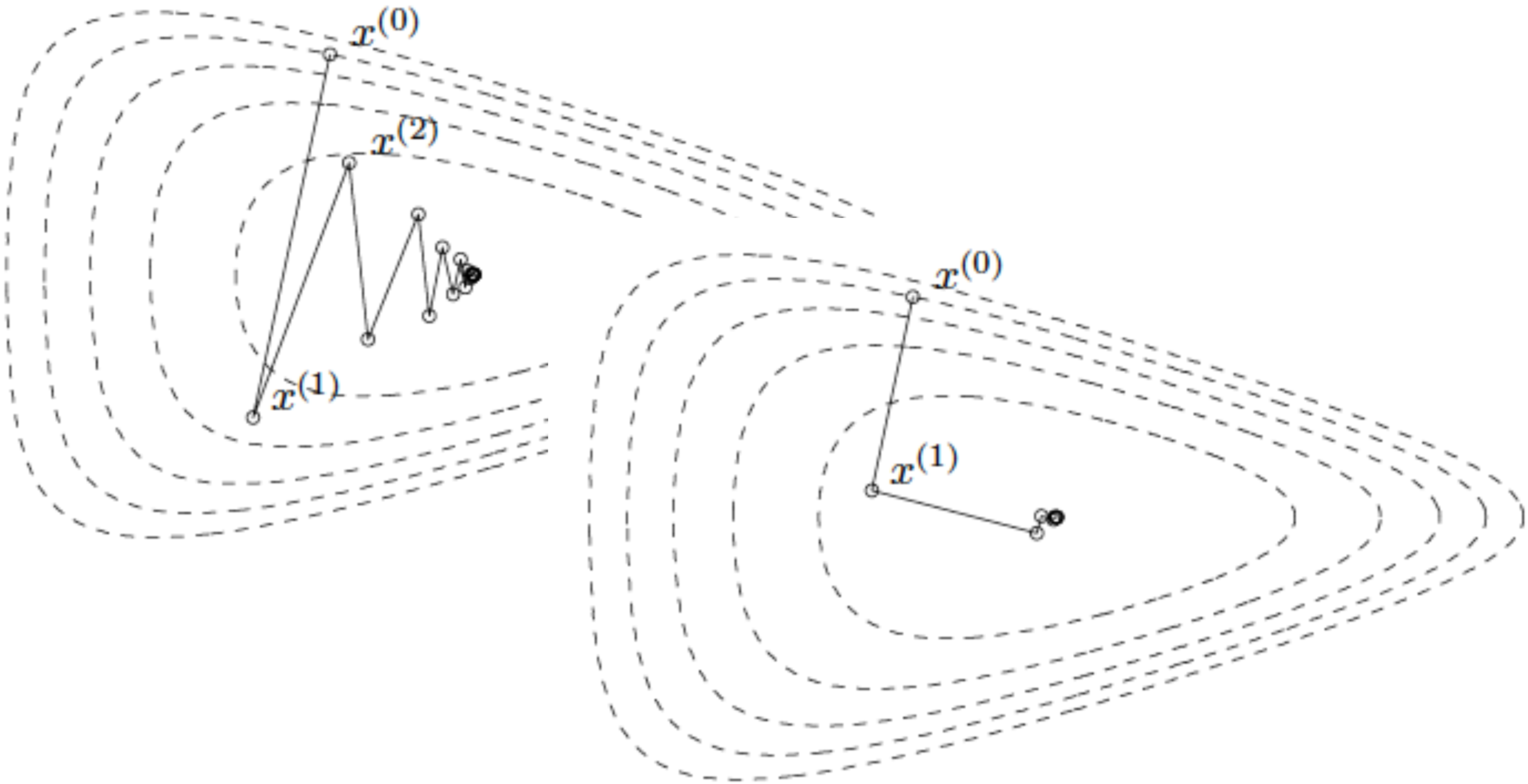
when to stop?




Simple criterion – when gradient becomes close to zero – slope is almost flat

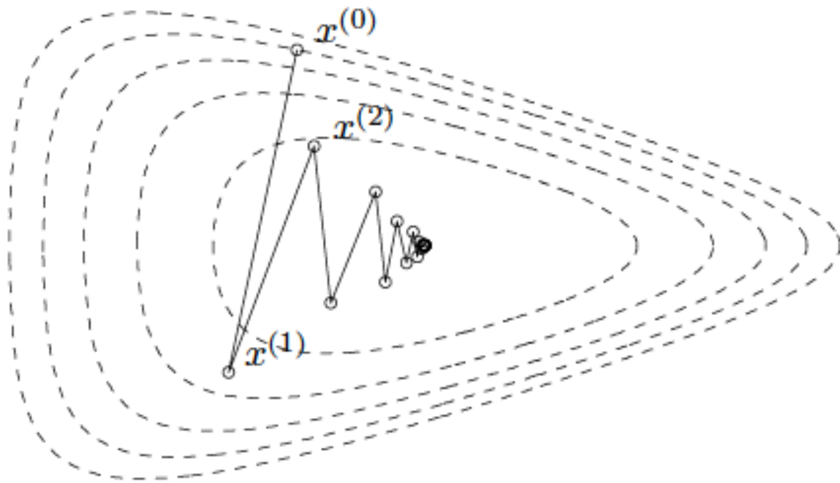
# Convex optimization

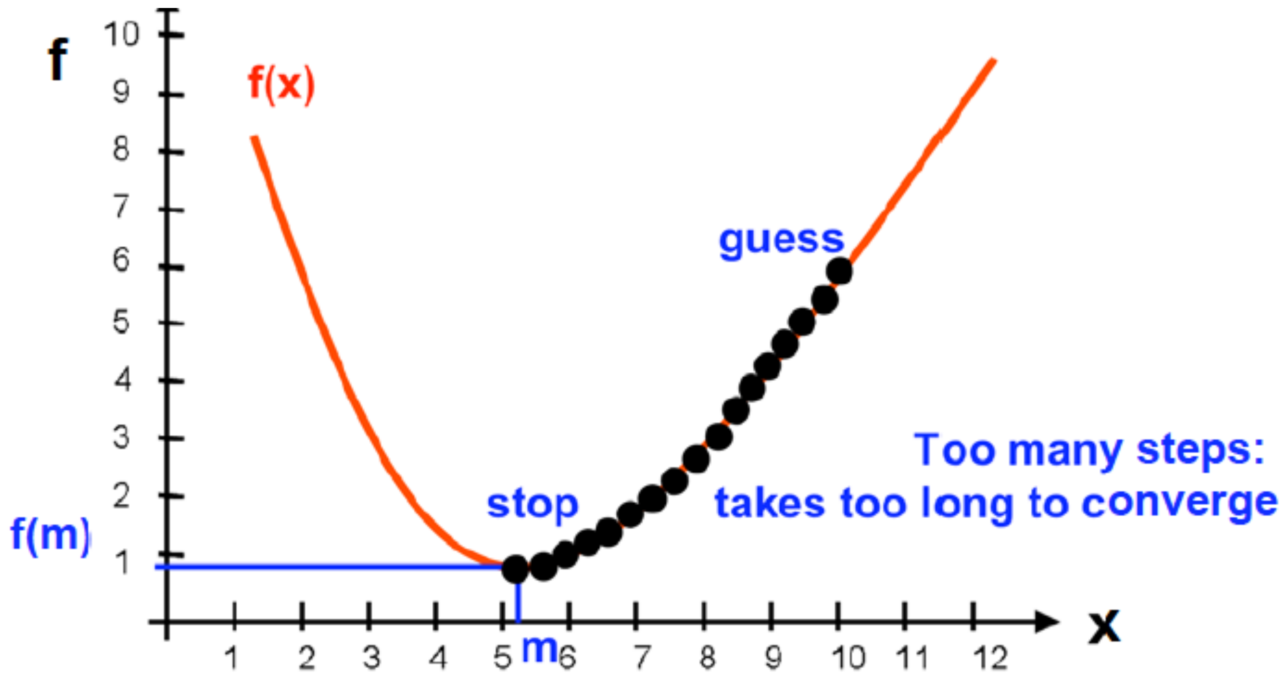
step size and direction influences what the new step will be



# Convex optimization

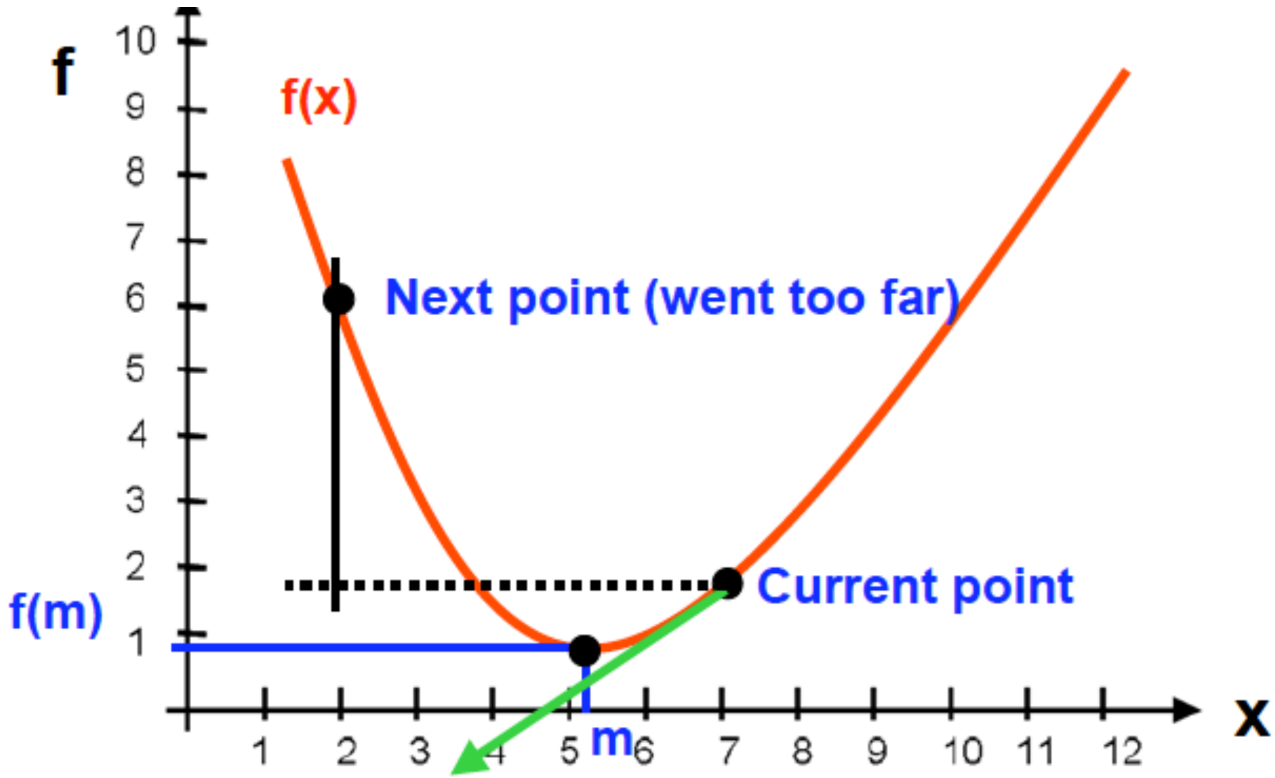
Gradient descent:
- can be slow
  - Zig-zags
  - Tiny steps



# Convex optimization

A step size that is too large can overshoot the minimum:



# Convex optimization

$x_{n+1} = x_n - \nabla f(x_n)/L$

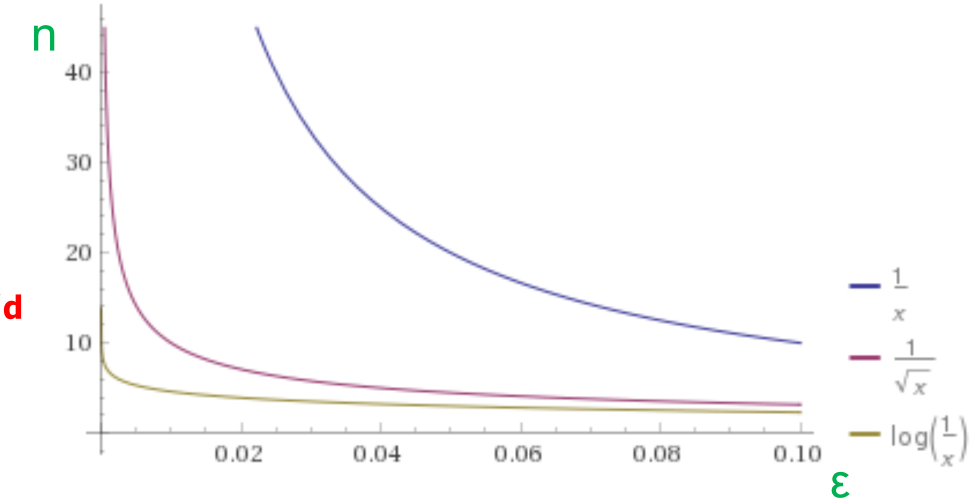
n: number of iterations required to obtain error $|f(x_n)-f^*| \leq \varepsilon$

**Convergence rates:**

$C_L^{0,0}$ – arbitrary function $f:R^d \to R$,
Lipschitz continuous

Grid search:

**$n \sim (1/\varepsilon)^d$ or $\varepsilon \sim 1/n^{1/d}$**




$F_L^{1,1}$ – convex $f:R^d \to R$, with Lipschitz continuous gradient

Gradient descent:

**$n \sim 1/\varepsilon$ or $\varepsilon \sim 1/n$**

Optimal gradient methods: (e.g. Nesterov's accelerated gradient):

**$n \sim 1/\sqrt{\varepsilon}$ or $\varepsilon \sim 1/n^2$**

$S_{\mu,L}^{1,1}$ – strongly convex $f:R^d \to R$, with Lipschitz continuous gradient

Gradient descent (accelerated gradient = better constant):

**$n \sim \ln 1/\varepsilon$ or $\varepsilon \sim 1/\rho^n$**

# Gradient of a linear model / MSE

- Apply chain rule of differentiation to:

- $\partial(y-(ax+b))^2 / \partial a$

$$\frac{\partial}{\partial a}\left((y-(a\,x+b))^2\right) = -2\,x\,(-a\,x-b+y)$$

- $\partial(y-(ax+b))^2 / \partial b$

$$\frac{\partial}{\partial b}\left((y-(a\,x+b))^2\right) = -2\,(-a\,x-b+y)$$

- Might be helpful to define:
  - h(x,a,b)=ax+b
  - E(y,x,a,b)=y-h(x,a,b)