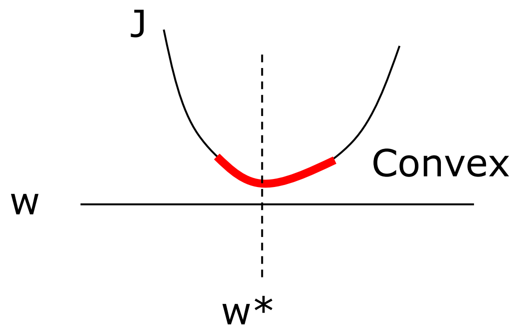
Optimization Method
Lecture 4

Jeong-Yean Yang

2020/10/22

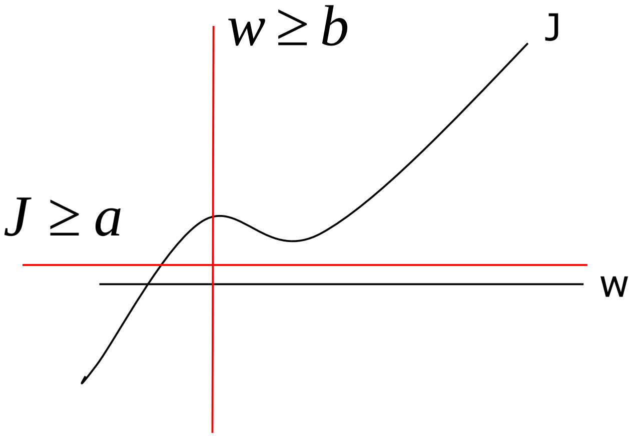
Optimization

- Optimization Definition
  - Find parameters,w for minimizing scalar function, J.
- Cost(Scalar ,Objective) function
  - Define J and try to minimize it $J = J(\omega), \quad \omega : Parameter$
  - J= scalar function
- J Must be Convex Hull
  - Convex(볼록) Vs. Concave(오목)
  - At a Convex hull, Differentiation is zero $\frac{\partial J}{\partial w} = 0$



If My Cost Function has No Convex Hull,..

- 1. Constraint is needed



- Constraints
  - Equality constraint

    $$J = c, \quad \min J = ?$$

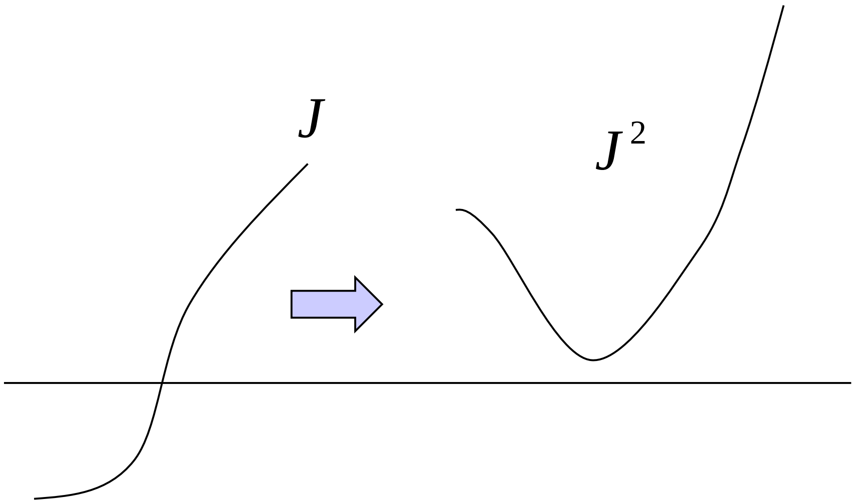
  - Inequality constraint

    $$J \geq a, \quad \min J = ?$$

- 2. Use Square

$$J = J(\omega)$$

$$J_{new} = J^2$$



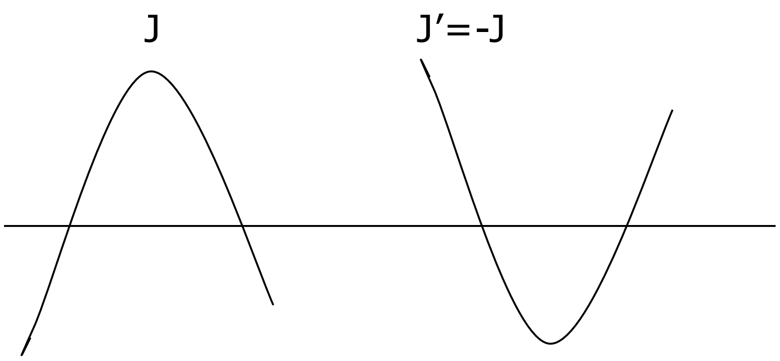
  - Ex) Minimum Squared Error

If My J has Maximum Convex Hull,…

- Use Minus

$$J = J(\omega)$$

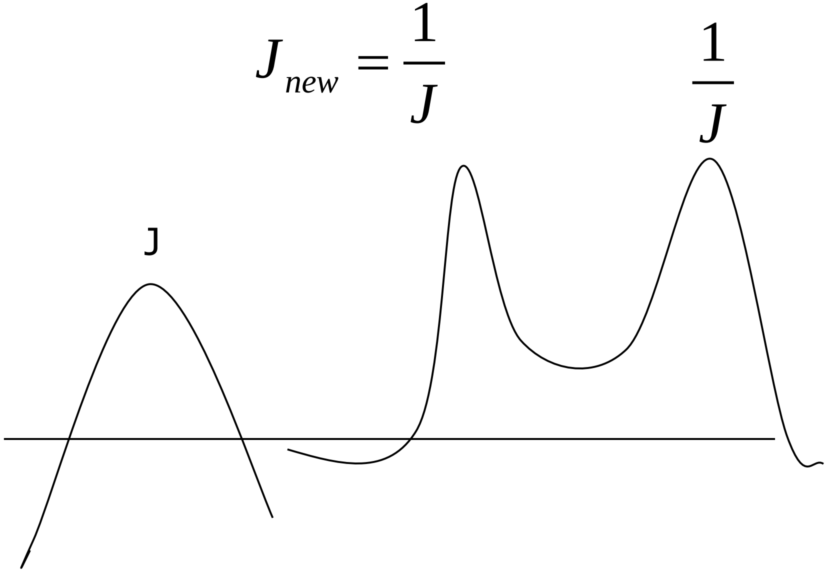
$$J_{new} = -J$$



- 2. Use Denomination

$$J = J(\omega)$$

$$J_{new} = \frac{1}{J}$$



What We learn in Last Week

- 1.Differentiation=0

$$J = J(\omega)$$
$$\frac{\partial J}{\partial w} = 0$$

⇨

$$w = (a, b)$$
$$J = J(\omega) = \mathrm{J(a, b)} \Rightarrow \frac{\partial J}{\partial a} = 0, \frac{\partial J}{\partial b} = 0$$

  – Case 1) Linear equation

$$\frac{\partial J}{\partial w_i} = 0 \quad ⇨ \quad Aw = b$$

  – Case 2) Non-Linear equation

$$\frac{\partial J}{\partial w_i} = 0 \quad ⇨ \quad \begin{aligned} f(w) &= 0 \\ g(w) &= 0 \end{aligned}$$

- Non-Linear Newton-Raphson Method

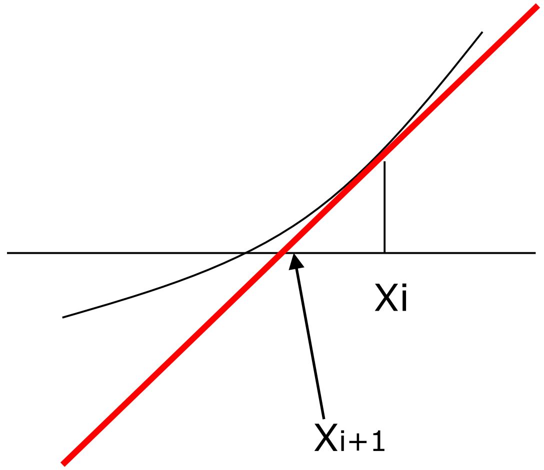
Simple Optimization

- How to do optimization?

$$J = J(\omega) \qquad \frac{\partial J}{\partial w} = 0$$

- We know Minimum exists where $\frac{\partial J}{\partial w} = 0$

- Think Newton Method again, f(x)=0 → $\frac{\partial J}{\partial w} = f(x) = 0$



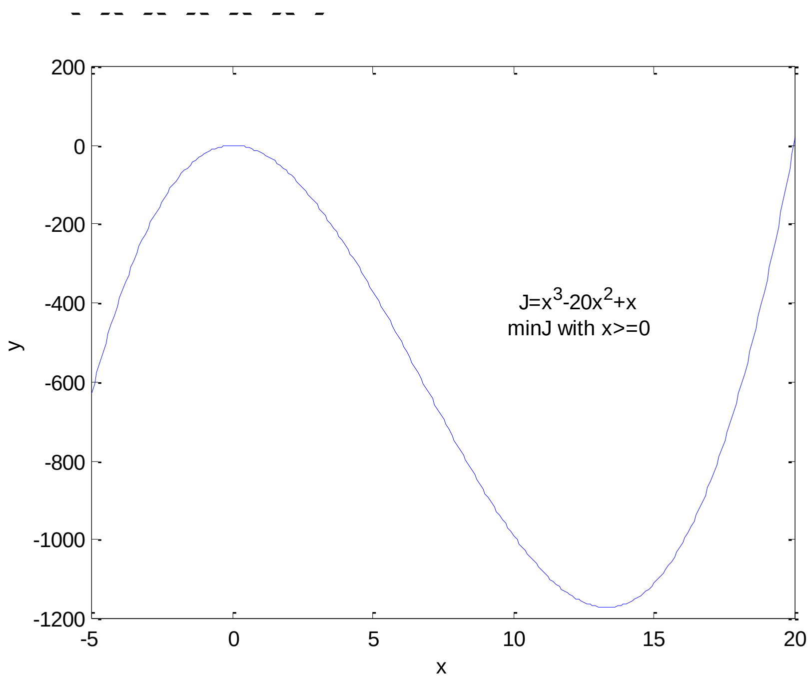
$$y = \frac{\partial f}{\partial x}(x - x_i) + f(x_i) = 0$$

$$x_{i+1} = x = -\frac{f(x_i)}{f'(x_i)} + x_i$$

Ex) Optimization with Newton Method

Test1.m

```
%J = x^3-20x^2+x
%F = 3x^2-40x+1

X0=20;
X=X0;
Xold=X0;

Xs=[];
Js=[];

for i=1:100
    % Newton Method for F= dJ/dX=0
    dF= 6*X-40;
    F = 3*X^2-40*X+1;
    Xnew = X-F/dF;

    if (abs(Xold-Xnew)<1e-5)
       break;
    end

    % display
    J = X^3 -20*X^2 +X;
    Js =[Js;J];
    Xs =[Xs;X];
    [X J]

    % update new X
    Xold = X;
    X = Xnew;
end
```

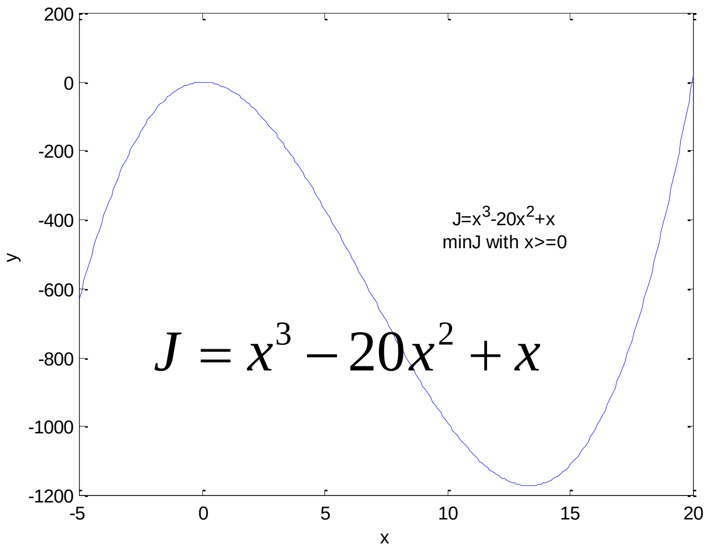


$$J = x^3 - 20x^2 + x$$

$$F = \frac{\partial J}{\partial x} = 3x^2 - 40x + 1 = 0$$

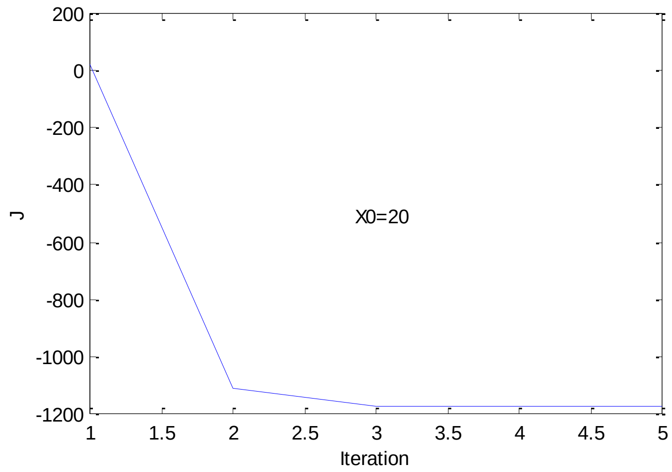
$$F' = 6x - 40$$

Effect of First Guess Value

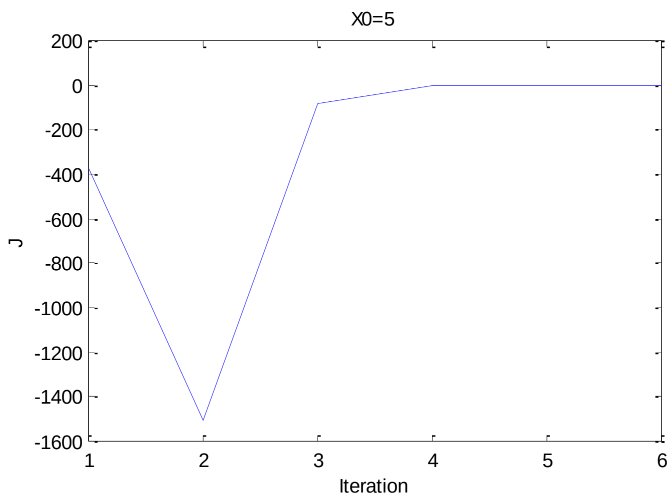


$$J = x^3 - 20x^2 + x$$

- Start with X0=20→ Xmin=13.3



- Start with X0=5 → Xmin=0.025



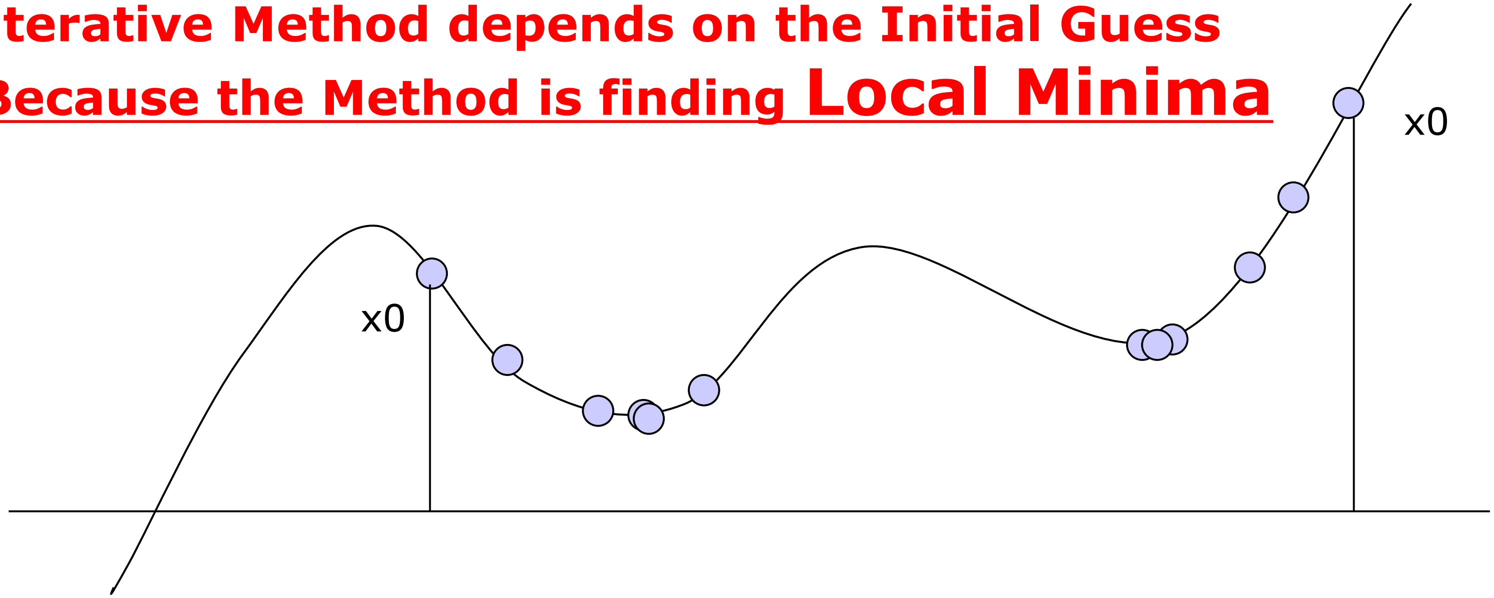

```
Xs =

    5.0000
   -7.4000
   -1.9346
   -0.1982
    0.0214
    0.0250
```

Guess with Multiple Minimum(=Minima)

- Where You Start?
- Solution Highly Depends on Initial Guess.
  - Keep it "If there are More than One Minimum"

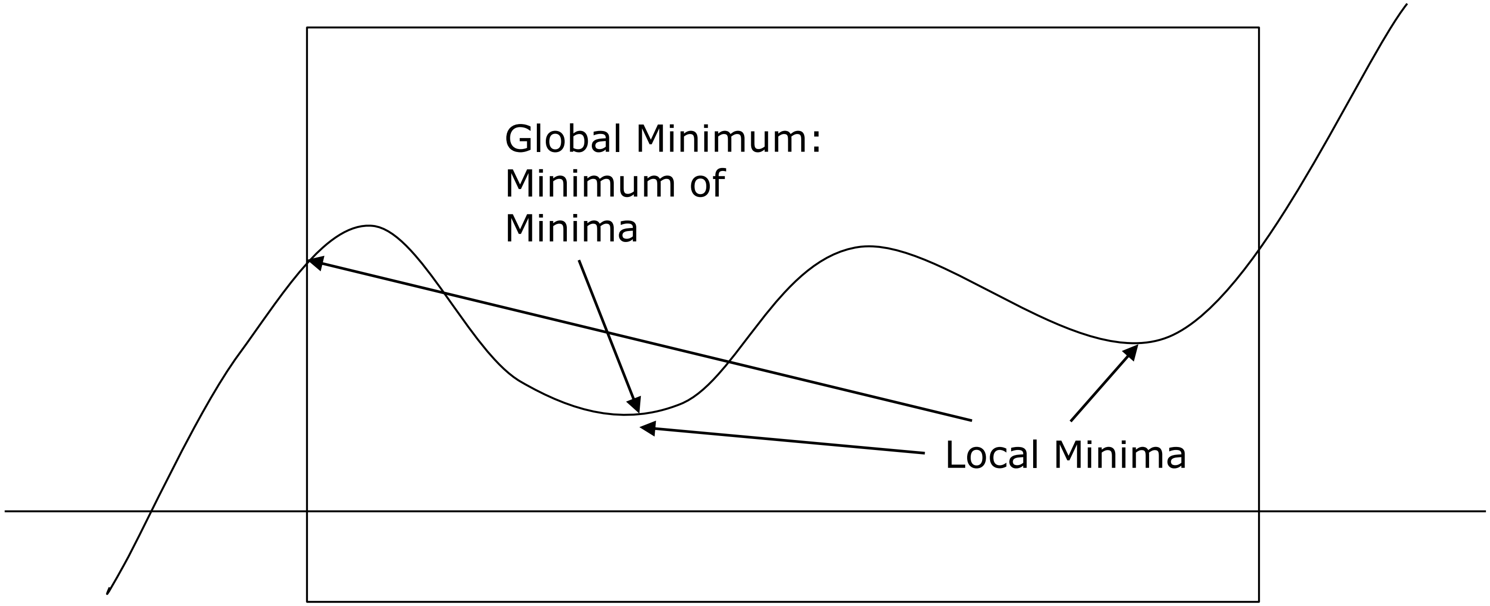
Iterative Method depends on the Initial Guess

Because the Method is finding Local Minima



Guess with Multiple Minimum(=Minima)

- Where You Start?
- Solution Highly Depends on Initial Guess.
  - Keep it "there are More than one Minimum"



1. Constraint =?

Optimization with Differentiation in a Multi Dimensional Vector Space

- Differentiation is the KEY for Any Equation in Any Space
- You learned Differentiation in Function Space
  - Function Space : a set of functions from a set X to a set Y
  - Function : X → F(X)=Y

$$y = f(x)$$

$$dy = df(x) = \frac{\partial f(x)}{\partial x} dx$$

$$\therefore \frac{dy}{dx} = \frac{\partial f(x)}{\partial x} = f'(x)$$

- Well, Differentiation in a Multi dimensional Vector Space?
  - You have used Differentiation with X and Y in the Vector Space!!

Gradient Definition:
Differentiation in a Multidimensional Vector Space

- Y=F(x) is a function. In other words,

$$y = f(x) \quad \Rightarrow \quad g(x,y) = y - f(x) = 0$$

Function Equation

- How we do Differentiation in a x,y vector space?
- Define Gradient

$$Grad = \nabla = \frac{\partial}{\partial x}\hat{i} + \frac{\partial}{\partial y}\hat{j} + \frac{\partial}{\partial z}\hat{k}$$ x,y,z space

$$Grad = \nabla = \frac{\partial}{\partial x}\hat{i} + \frac{\partial}{\partial y}\hat{j}$$ x,y space

$$Grad = \nabla = \frac{\partial}{\partial w_1}\hat{n}_1 + \frac{\partial}{\partial w_2}\hat{n}_2 + \ldots \frac{\partial}{\partial w_N}\hat{n}_N$$ N space → **Neural Network**

Function Vs. Equation

$$y = f(x) \quad \Rightarrow \quad g(x, y) = y - f(x) = 0$$

Function — Equation

- Use Gradient,

$$\nabla g(x, y) = \frac{\partial}{\partial x}(y - f(x))\hat{i} + \frac{\partial}{\partial y}(y - f(x))\hat{j}$$

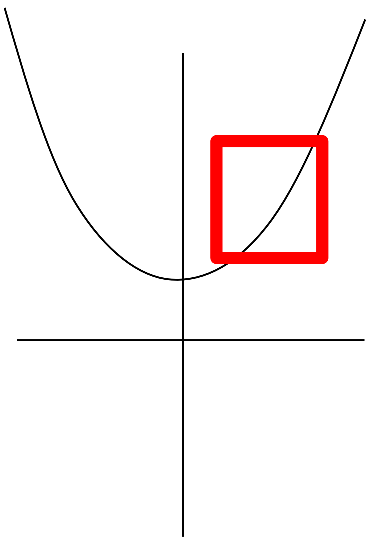
$$= -f'(x)\hat{i} + \hat{j}$$

- Use Total derivative

$$T.D.\ \text{of}\quad \text{g} = dg(x, y) \equiv \frac{\partial g}{\partial x}dx + \frac{\partial g}{\partial y}dy$$

$$dg = \left(\frac{\partial g}{\partial x}\hat{i} + \frac{\partial g}{\partial y}\hat{j}\right) \bullet \left(dx\hat{i} + dy\hat{j}\right) = \nabla g \bullet d\hat{X}$$

Gradient: Normal Vector



$$y = x^2 + 1$$

$$\frac{dy}{dx} = 2x$$

$$g(x, y) = y - x^2 - 1 = 0$$

$$\nabla g = \frac{\partial g}{\partial x}\hat{i} + \frac{\partial g}{\partial y}\hat{j} = -2x\hat{i} + 1\hat{j}$$

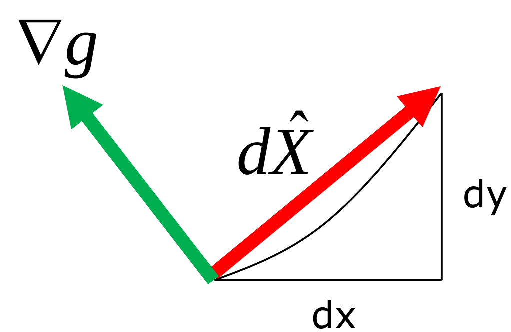
$$dg = \frac{\partial g}{\partial x}dx + \frac{\partial g}{\partial y}dy = \nabla g \bullet d\hat{X} = -2xdx + dy$$

$$= 0$$

- Focus on that

$$dg = \nabla g \bullet d\hat{X} = 0$$

$$\therefore \nabla g \perp d\hat{X}$$

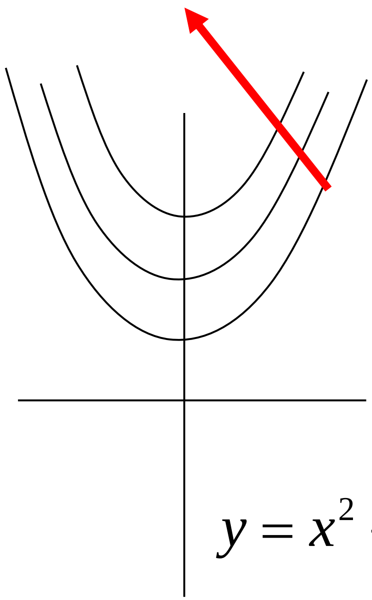


- Gradient is a Normal Vector to a $d\hat{X}$ vector

Gradient Direction

- Gradient Direction
  - The Greatest Rate of Function Increment

∇g Function Increment

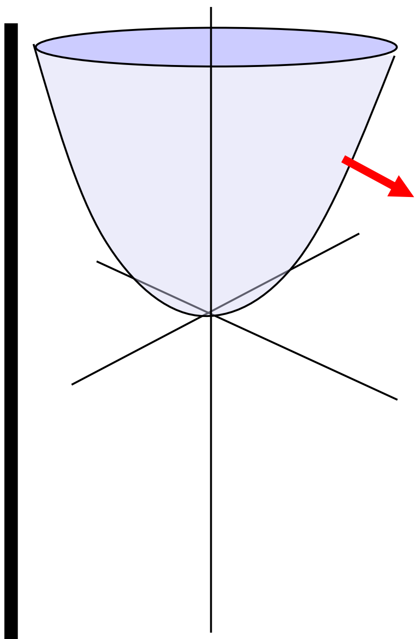


$$g(x,y) = y - x^2 - 1 = 0$$
$$g(x,y) = y - x^2 - 1 = 1$$
$$g(x,y) = y - x^2 - 1 = 2$$
$$\cdots$$

$y = x^2 + 1$

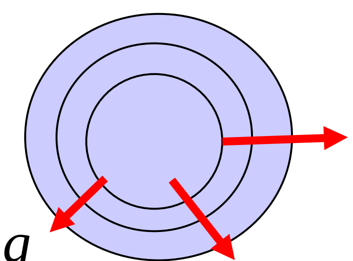
$g = y - x^2 - 1$

Function Increment



$$g(x,y) = x^2 + y^2 = 1$$
$$g(x,y) = x^2 + y^2 = 2$$
$$g(x,y) = x^2 + y^2 = 3$$
$$\cdots$$

Function Increment

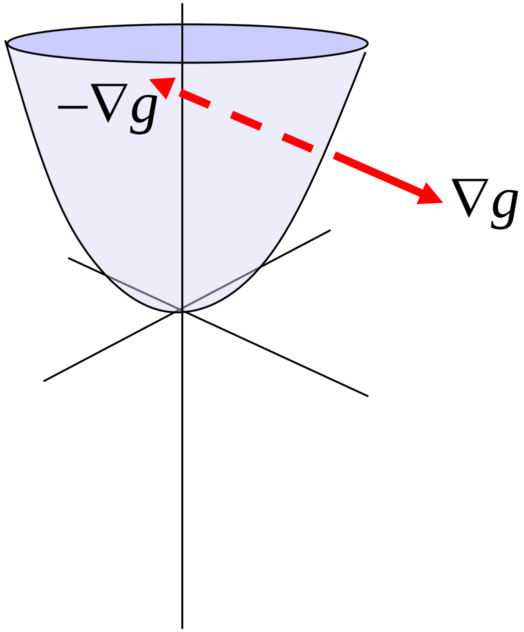


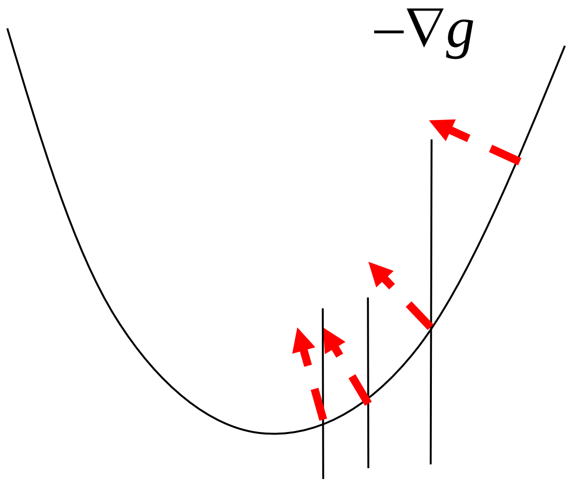
∇g

Optimization with Gradient

- Gradient Direction = Function Increment
- Optimization = Find Function Minimum

→ Optimization = Reverse Direction of Gradient ?

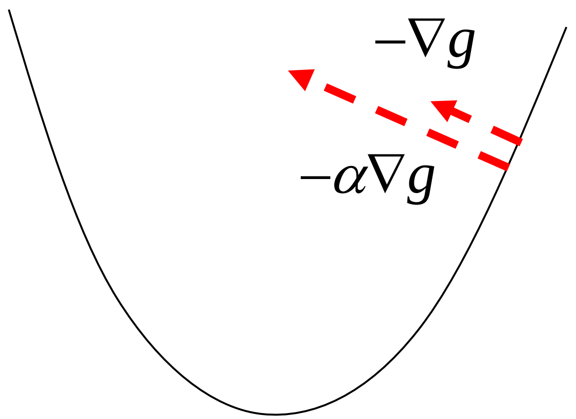


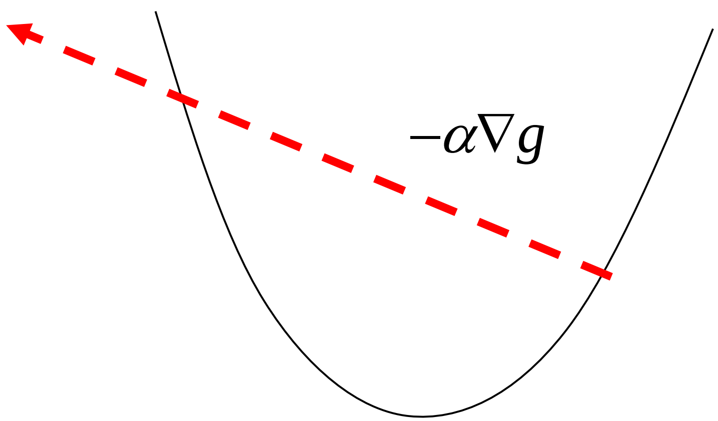


Focus on that Minus Gradient Direction is changing.

At the minima, Minus Gradient direction sees top

Gradient Descent Method I





Big alpha passes minima

- Gradient Descent

  $-\nabla g$

- How far we go?

  $-\alpha\nabla g$

  - With Alpha scalar..
- Small alpha
  - It could be a Long Journey
  - Too many iterations
- Big alpha
  - It passes minima.
  - Solution becomes unstable.

Gradient Descent Method II

- 1. Guess X = X0
- 2. Calculate Next Point

$$X_{n+1} = X_n - \alpha \nabla g$$

- 3. Check Stop Condition

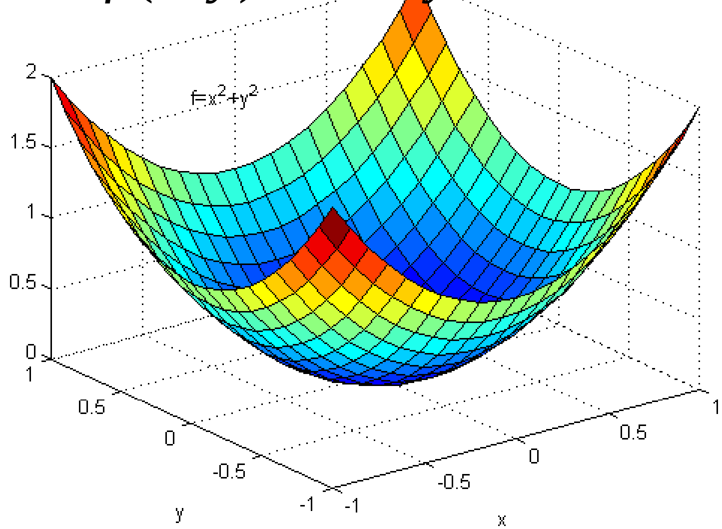
$$if \quad \frac{| X_{n+1} - X_n |}{| X_n |} < \varepsilon, \quad \text{stop}$$

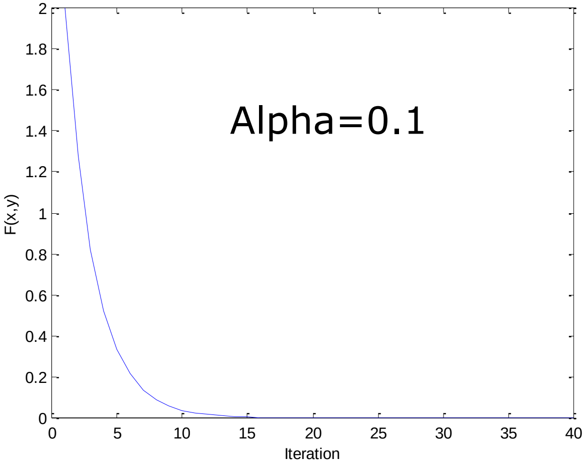
- 4. Go to 2

GDM Example (test2.m)

```
for i=1:1000
    x = X(1);
    y = X(2);
    F = x^2+y^2;

    %calcuate gradient
    gx = 2*x;
    gy = 2*y;
    G = [gx,gy];

    %next point
    Xnew = X-alpha*G;

    Fs=[Fs;F]
    Xs=[Xs;X];

    % stop condition
    d = (X-Xnew)*(X-Xnew)';
    if (d/norm(X)<1e-5)
        break;
    end

    X = Xnew;
end
```

$$f(x,y) = x^2 + y^2$$





Effect of Alpha(Learning Rate)