


CS 4758/6758: Robot Learning

Spring 2012.

Ashutosh Saxena

Ashutosh Saxena



Goals



Goals encoded as a Cost Function

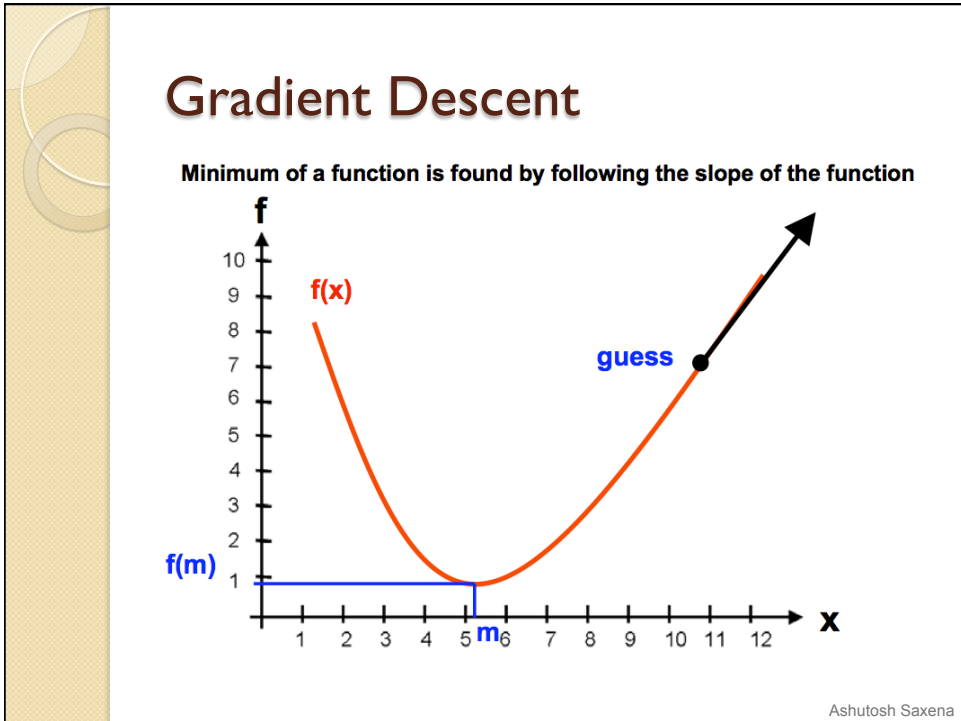
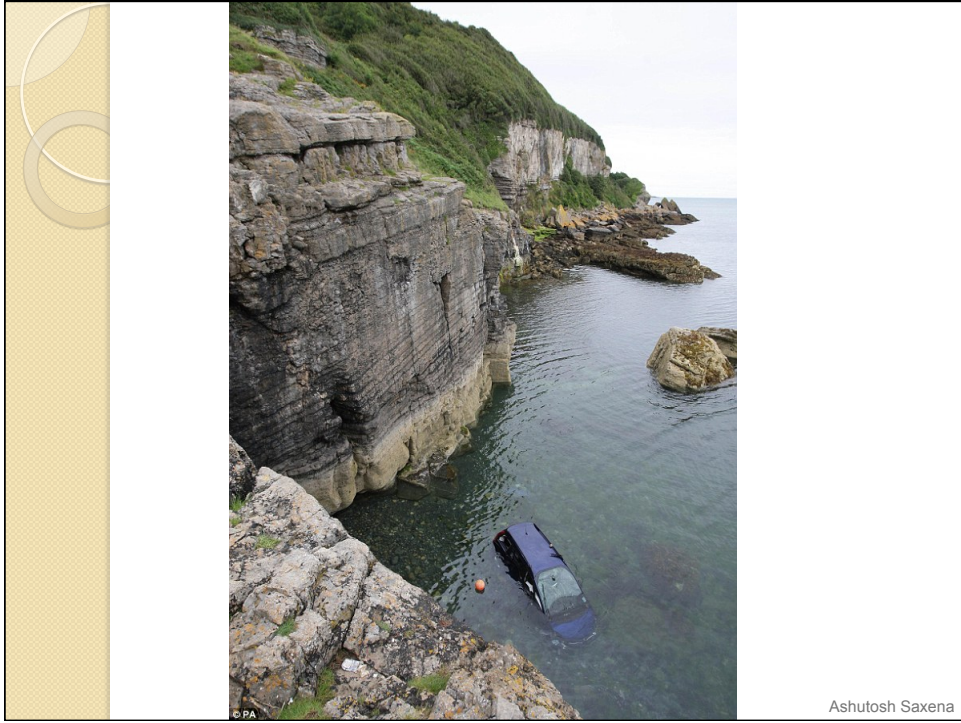
- Which areas on the road are good?

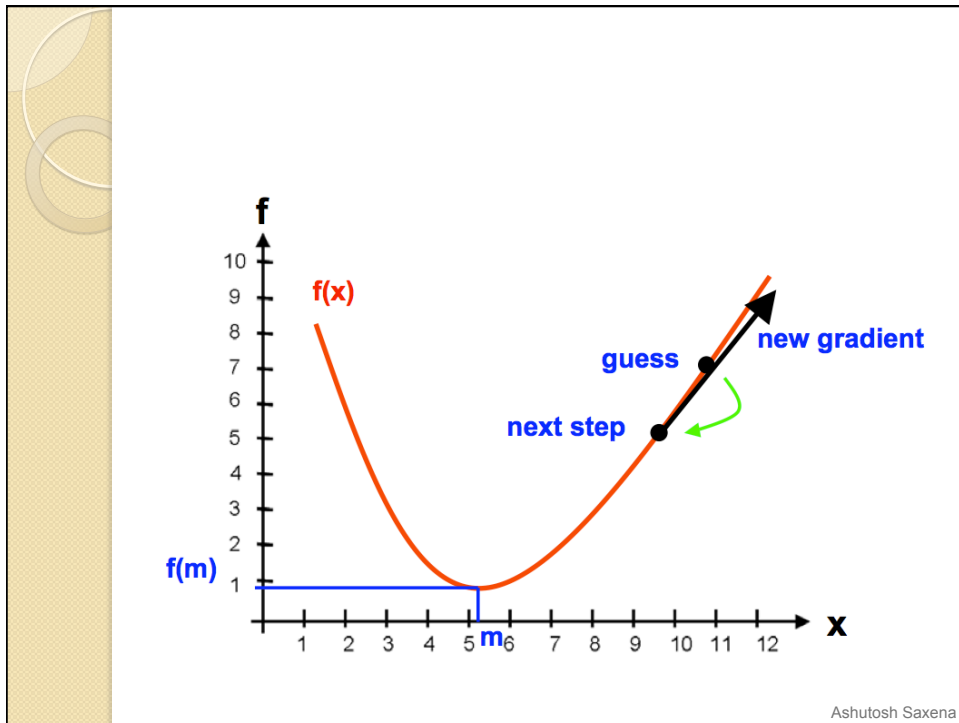
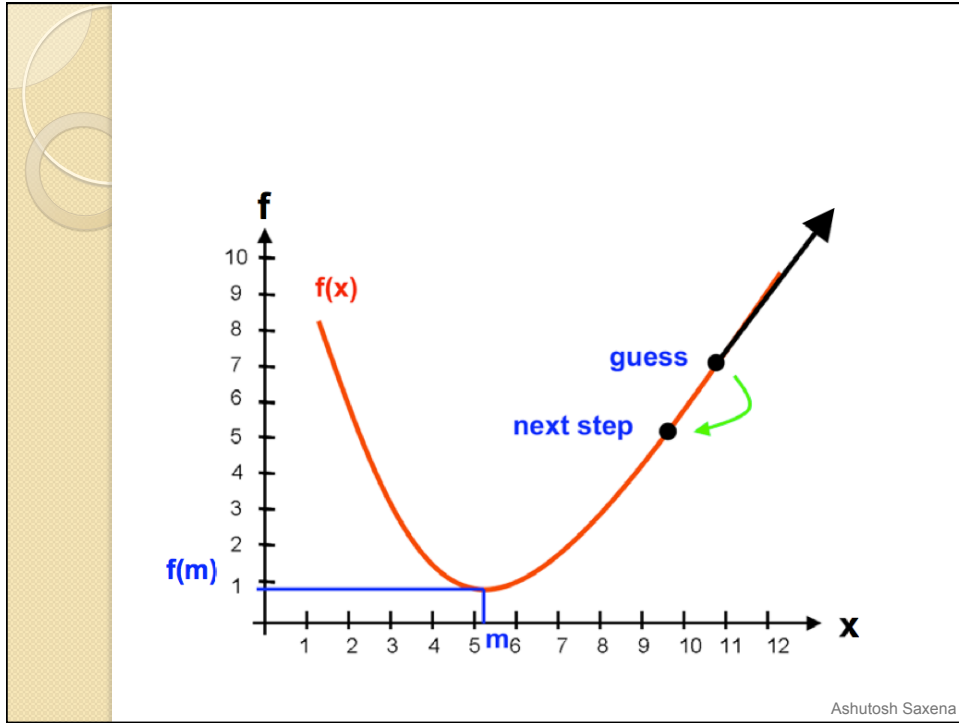
Ashutosh Saxena

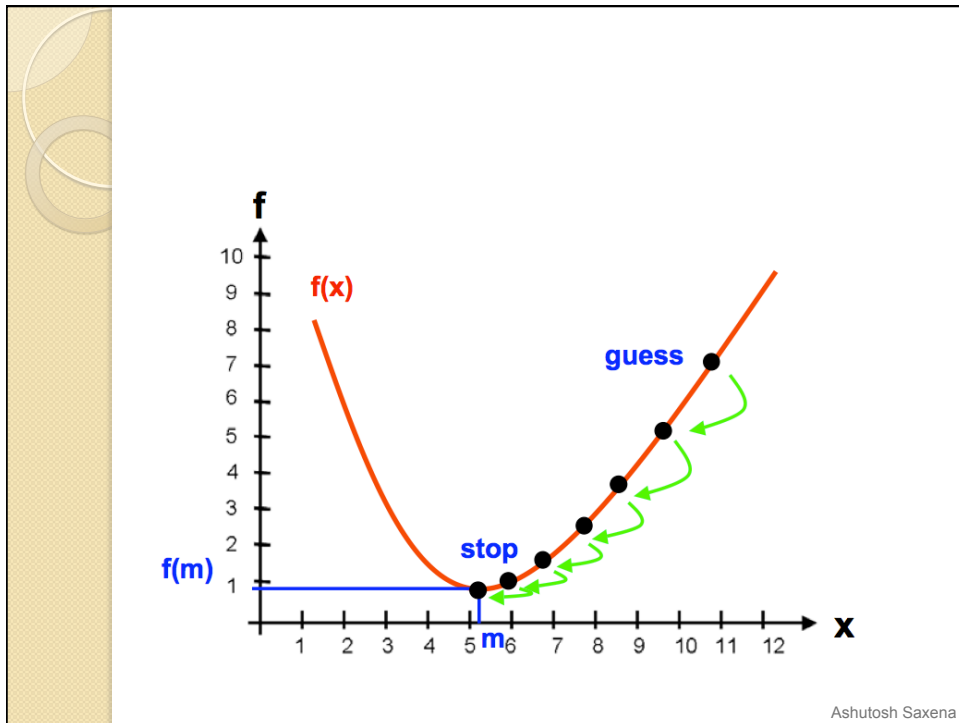
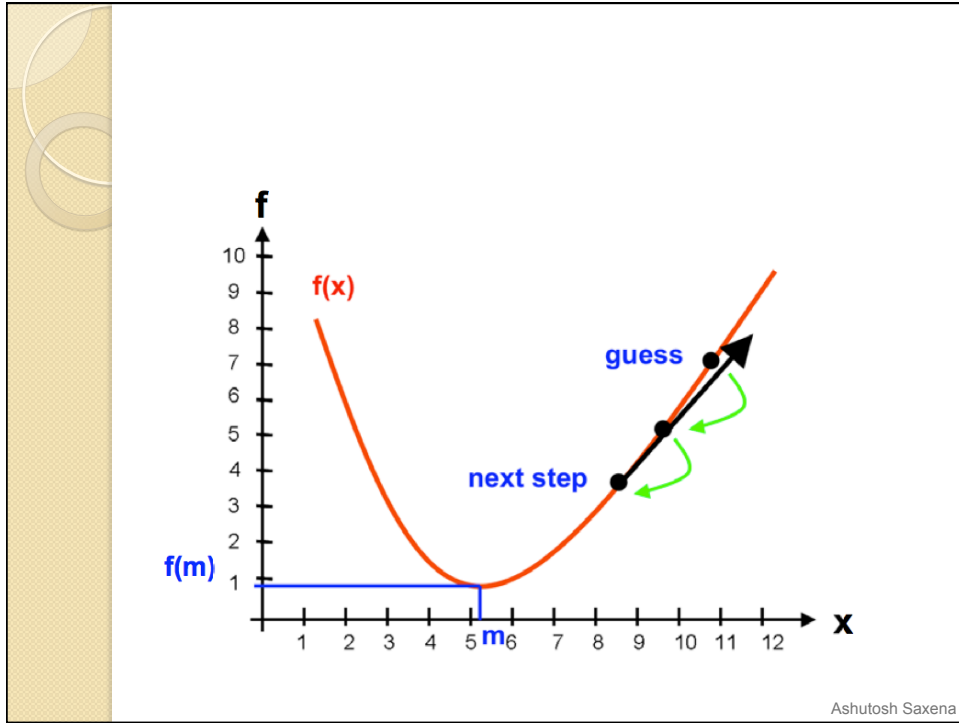
Optimizing cost function: Descent Methods

- General descent algorithm
- Generalization to multiple dimensions
- Problems of descent methods, possible improvements.
- Fixes
- Local Minima

Ashutosh Saxena





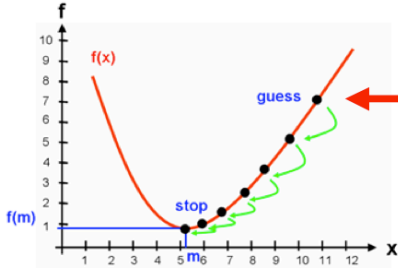


Start with a point (guess)

Repeat

- Determine a descent direction
- Choose a step
- Update

Until stopping criterion is satisfied



The graph shows a coordinate system with x-axis from 1 to 12 and y-axis from 1 to 10. A red curve labeled $f(x)$ starts at (1, 8) and reaches a minimum at (5, 1). A blue horizontal line at $y=1$ is labeled $f(m)$. A point at $x=5$ is labeled m_6 . A point at $x=11$ is labeled 'guess'. A green zigzag line starts at the 'guess' point and moves down towards the minimum, with arrows indicating the direction of descent.

guess

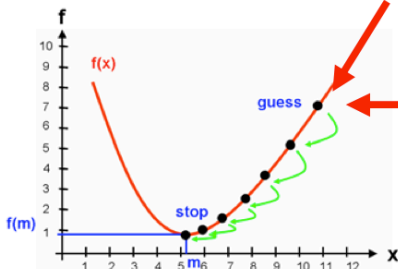
Ashutosh Saxena

Start with a point (guess)

Repeat

- Determine a descent direction
- Choose a step
- Update

Until stopping criterion is satisfied

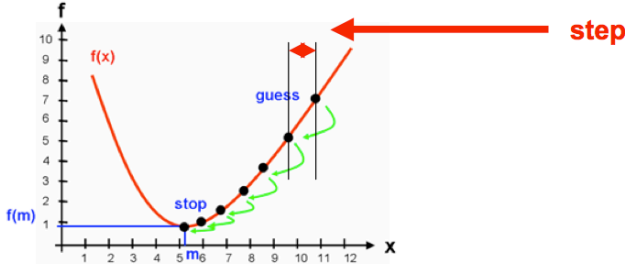


The graph is identical to the one above. A red arrow points from the 'guess' point towards the minimum, labeled 'Direction: downhill'.

Direction: downhill

Ashutosh Saxena

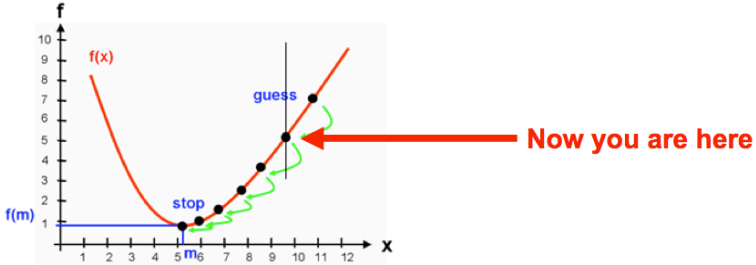
Start with a point (guess)
Repeat
 Determine a descent direction
 Choose a step
 Update
Until stopping criterion is satisfied



The graph shows a function $f(x)$ on a coordinate system with x-axis from 1 to 12 and y-axis from 1 to 10. A red curve represents $f(x)$. A blue horizontal line at $f(m) \approx 1.5$ is labeled 'stop'. A point at $x \approx 10.5$ is labeled 'guess'. A green zig-zag path starts at the 'guess' point and moves towards the 'stop' point. A red arrow labeled 'step' points from the 'guess' point to the next point on the path.

Ashutosh Saxena

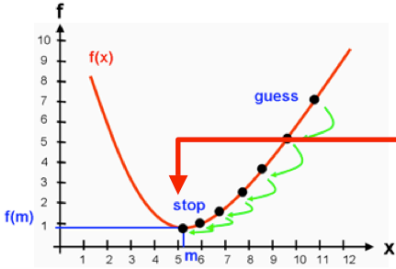
Start with a point (guess)
Repeat
 Determine a descent direction
 Choose a step
 Update
Until stopping criterion is satisfied



The graph is identical to the one above. A red arrow labeled 'Now you are here' points to the 'guess' point at $x \approx 10.5$.

Ashutosh Saxena

Start with a point (guess)
Repeat
Determine a descent direction
Choose a step
Update
Until stopping criterion is satisfied

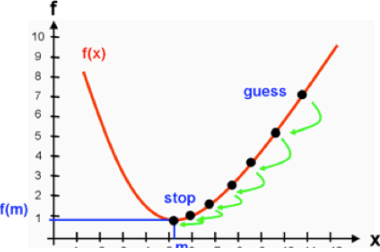


Stop when "close" from minimum

Ashutosh Saxena

Start with a point (guess)
Repeat
Determine a descent direction
Choose a step
Update
Until stopping criterion is satisfied

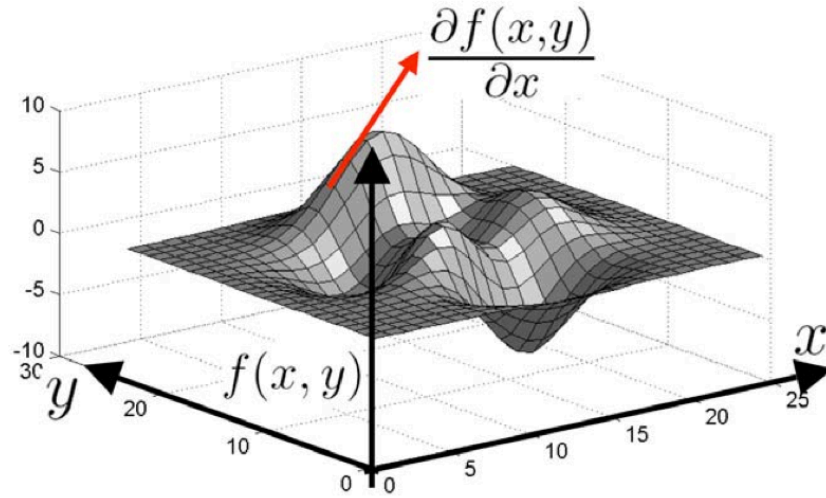
guess = x
direction = $-f'(x)$
step = $h > 0$
 $x := x - hf'(x)$
 $f'(x) \sim 0$



Ashutosh Saxena

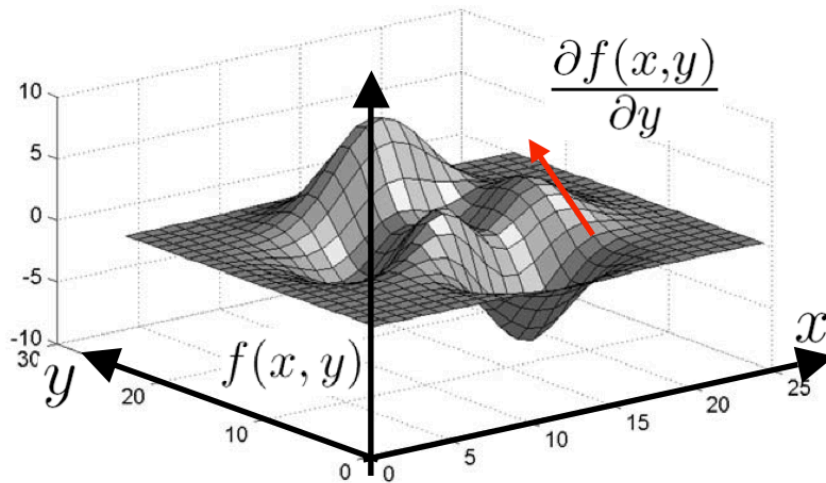
Example of 2D gradient: pic of the MATLAB demo

Illustration of the gradient in 2D



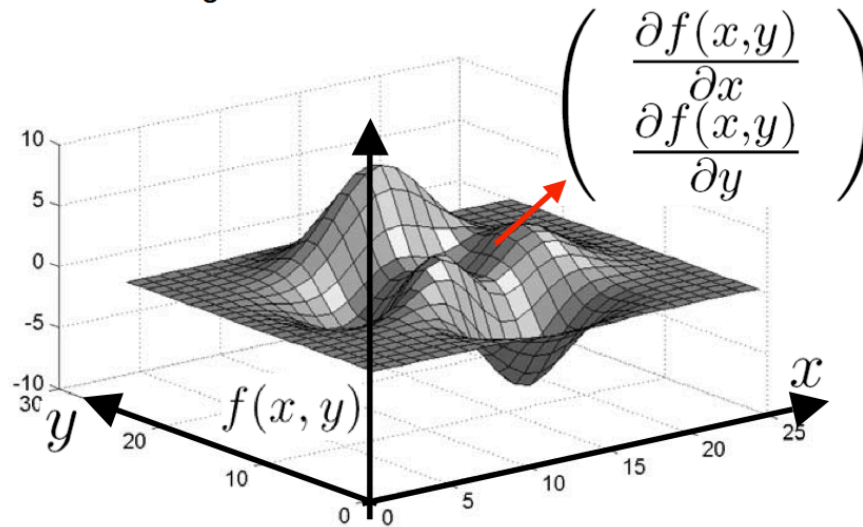
Example of 2D gradient: pic of the MATLAB demo

Illustration of the gradient in 2D



Example of 2D gradient: pic of the MATLAB demo

Illustration of the gradient in 2D



Example of 2D gradient: pic of the MATLAB demo

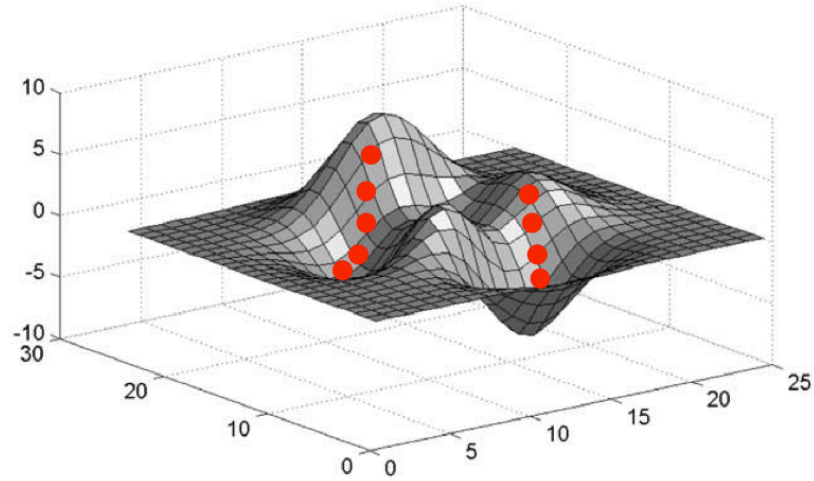
Definition of the gradient in 2D

$$\nabla f(x, y) = \begin{pmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{pmatrix}$$

This is just a generalization of the derivative in two dimensions.
This can be generalized to any dimension.

Example of 2D gradient: pic of the MATLAB demo

Gradient descent works in 2D



Generalization to multiple dimensions

Start with a point (guess)

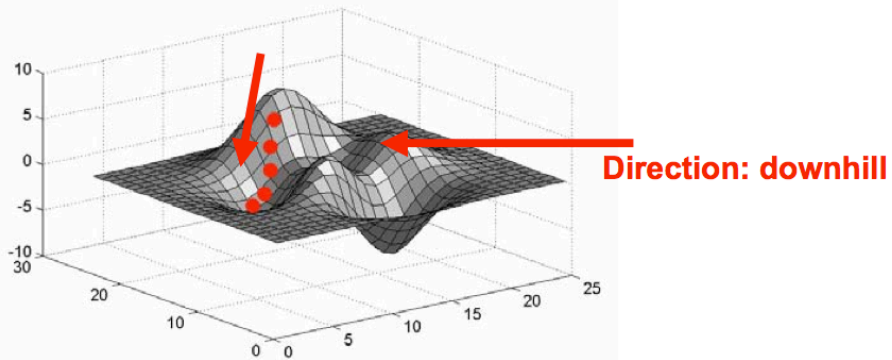
Repeat

Determine a descent direction

Choose a step

Update

Until stopping criterion is satisfied



Generalization to multiple dimensions

Start with a point (guess)

Repeat

Determine a descent direction

Choose a step

Update

Until stopping criterion is satisfied

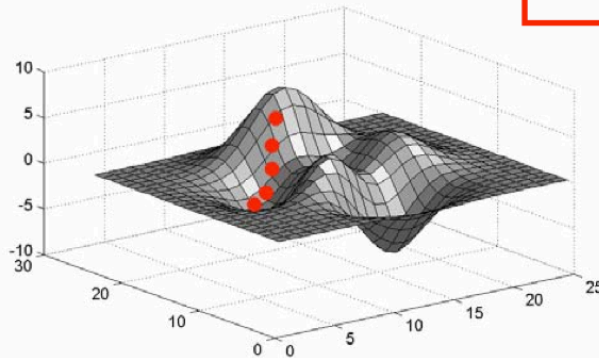
guess = x

direction = $-f'(x)$

step = $h > 0$

$x := x - h \nabla f'(x)$

$\nabla f'(x) \sim 0$



Multiple dimensions

Everything that you have seen with derivatives can be generalized with the gradient.

For the descent method, $f'(x)$ can be replaced by

$$\nabla f(x, y) = \begin{pmatrix} \frac{\partial f(x, y)}{\partial x} \\ \frac{\partial f(x, y)}{\partial y} \end{pmatrix}$$

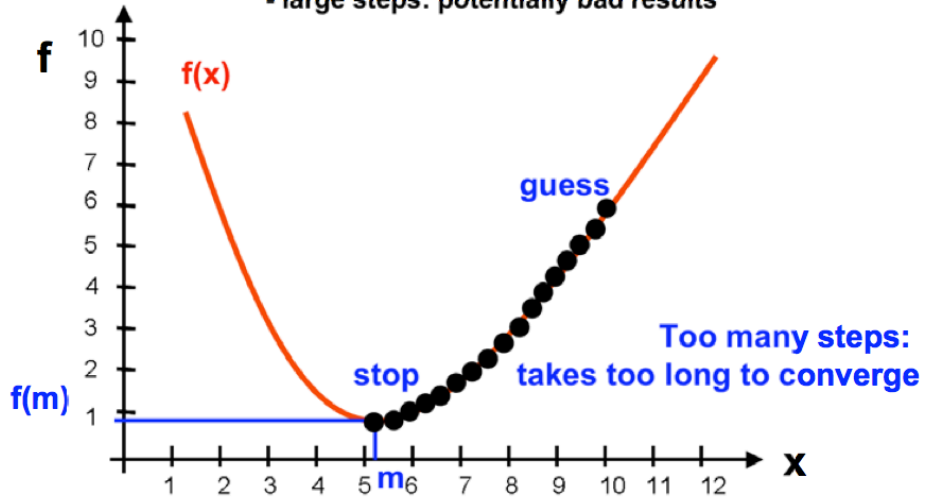
In two dimensions, and by

$$\nabla f(x_1, x_2, \dots, x_i, \dots, x_N) = \begin{pmatrix} \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_i}, \dots, \frac{\partial f}{\partial x_N} \end{pmatrix}$$

in N dimensions.

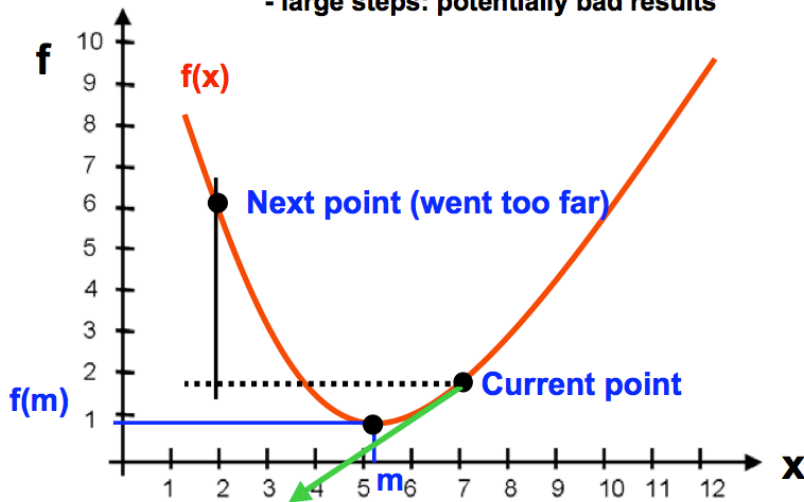
Problem 1: choice of the step

When updating the current computation:
 - small steps: inefficient
 - large steps: potentially bad results



Problem 1: choice of the step

When updating the current computation:
 - small steps: inefficient
 - large steps: potentially bad results



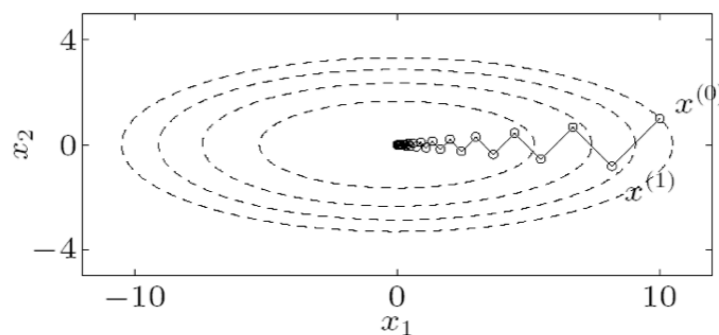
Solution to step size

- Back-tracking line search.
 - Step-size = step-size / 2
 - Until new function value gets smaller.

Ashutosh Saxena

Problem 2: « ping pong effect »

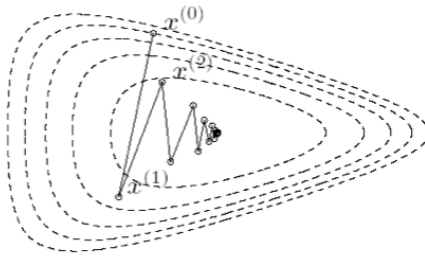
$$f(x) = (1/2)(x_1^2 + \gamma x_2^2) \quad (\gamma > 0)$$



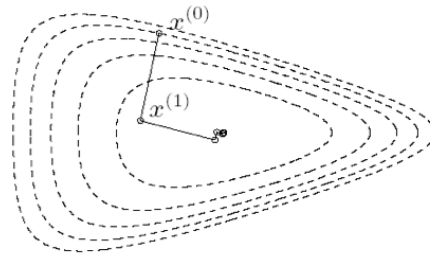
[S. Boyd, L. Vandenberghe, Convex Convex Optimization lect. Notes, Stanford Univ. 2004]

Problem 2: « ping pong effect »

$$f(x_1, x_2) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}$$



backtracking line search



exact line search

[S. Boyd, L. Vandenberghe, *Convex Convex Optimization lect. Notes, Stanford Univ. 2004*]

Fixes

Several methods exist to address this problem

- Line search methods, in particular
 - Backtracking line search
 - Exact line search
- Normalized steepest descent
- Newton steps

Fundamental problem of the method: local minima

