Machine Learning for Signal Processing

Lecture 4: Optimization

Instructor: Bhiksha Raj
(slides largely by Najim Dehak, JHU)

Course Projects

- Projects will be done by teams of students
  - Ideal team size: 4
  - Find yourself a team
  - If you wish to work alone, that is OK
    - But we will not require less of you for this
  - If you cannot find a team by yourselves, you will be assigned to a team
  - Teams will be listed on the website
  - All currently registered students will be put in a team eventually
- Will require background reading and literature survey
  - Learn about the problem

Projects

- Teams must be formed by 17th Tuesday
- Teams must send us a preliminary project proposal by 30th September 2019
  - Please send us proposals earlier, so that we can vet them
  - The later you start, the less time you will have to work on the project

Quality of projects

- Project must include aspects of signal analysis and machine learning
  - Prediction, classification or compression of signals
  - Using machine learning techniques
- Several projects from previous years have led to publications
  - Conference and journal papers
  - Best paper awards
  - Doctoral and Masters' dissertations

Projects from past years: 2015

- Loop querier – searching the rhythmic pattern
- Vision-based montecarlo localization for autonomous vehicle
- Beatbox to drum conversion
- City localization on flikr videos using only audio
- Facial landmarks based video frontalization and its application in face recognition
- Audioshop: Modifying and editing singing voice
- Predicting and classifying RF signal strength in an environment with obstacles
- Realtime detection of basketball players

Projects from past years: 2014

- IMPROVING SPATIALIZATION ON HEADPHONES FOR STEREO MUSIC
- PREDICTING THE OUTCOME OF ROULETTE
- FACIAL REPLACEMENT IN VIDEOS
- ISOLATED SIGN WORD RECOGNITION SYSTEM
- ACCENTED ENGLISH DIALECT CLASSIFICATION
- BRAIN IMAGE CLASSIFIER
- FACIAL EXPRESSION RECOGNITION
- MOOD BASED CLASSIFICATION OF SONGS TO IDENTIFY ACOUSTIC FEATURES THAT ALLEVIATE DEPRESSION
- PERSON IDENTIFICATION THROUGH FOOTSTEP-INDUCED FLOOR VIBRATION
- DETECT HUMAN HEAD-ORIENTATION BASED ON CONVOLUTIONAL NEURAL NETWORK AND DEPTH CAMERA
- NEURAL NETWORK BASED SLUDGE VOLUME INDEX PREDICTION

Projects from past years: 2014

- 8-BIT MUSIC NOTE IDENTIFICATION - TURNING MARIO INTO METAL
- STREET VIEW HOUSE NUMBER RECOGNITION BASED ON CONVOLUTIONAL NEURAL NETWORKS
- TRAIN-BASED INFRASTRUCTURE MONITORING
- MANIFOLD INTERPOLATION OF X-RAY RADIOGRAPHS
- A SMARTPHONE BASED INDOOR POSITIONING SYSTEM AUGMENTED WITH INFRARED SENSING
- ROCK, PAPER, SCISSORS -- HAND GESTURE RECOGNITION
- LANGUAGE MODELS WITH SEMANTIC CONSTRAINTS
- LEARNING TO PREDICT WHERE A DRIVER LOOKS
- REAL TIME MONITORING OF STUDENT'S LEARNING PERFORMANCE

Projects from past years: 2013

- Automotive vision localization
- Lyric recognition
- Imaging without a camera
- Handwriting recognition with a Kinect
- Gender classification of frontal facial images
- Deep neural networks for speech recognition
- Predicting mortality in the ICU
- Human action tagging
- Art Genre classification
- Soccer tracking
- Image manipulation using patch transforms
- Audio classification
- Foreground detection using adaptive mixture models

Projects from previous years: 2012

- Skin surface input interfaces
  - Chris Harrison
- Visual feedback for needle steering system
- Clothing recognition and search
- Time of flight countertop
  - Chris Harrison
- Non-intrusive load monitoring using an EMF sensor
  - Mario Berges
- Blind sidewalk detection
- Detecting abnormal ECG rhythms
- Shot boundary detection (in video)
- Stacked autoencoders for audio reconstruction
  - Rita Singh
- Change detection using SVD for ultrasonic pipe monitoring
- Detecting Bonobo vocalizations
  - Alan Black
- Kinect gesture recognition for musical control

Projects from previous years: 2011

- Spoken word detection using seam carving on spectrograms
  - Rita Singh
- Bioinformatics pipeline for biomarker discovery from oxidative lipidomics of radiation damage
- Automatic annotation and evaluation of solfege
- Left ventricular segmentation in MR images using a conditional random field
- Non-intrusive load monitoring
  - Mario Berges
- Velocity detection of speeding automobiles from analysis of audio recordings
- Speech and music separation using probabilistic latent component analysis and constant-Q transforms

Project Complexity

- Depends on what you want to do
- Complexity of the project will be considered in grading.
- Projects typically vary from cutting-edge research to reimplementation of existing techniques. Both are fine.
- Only caveat : The term “deep learning” must not relate to your project
  - Absolutely no DL/Nnets

Incomplete Projects

- Be realistic about your goals.

- Incomplete projects can still get a good grade if
  - You can demonstrate that you made progress
  - You can clearly show why the project is infeasible to complete in one semester

- Remember: You will be graded by peers

"Local" Projects..

- Several project ideas routinely proposed by various faculty/industry partners
  – Sarnoff labs, NASA, Mitsubishi, Adobe..
- Local faculty
  – Alan Black is usually good for a project or two
  – LP Morency has fantastic ideas on analysis of multimodal recordings of H-H (and H-C) communication
  – Roger Dannenberg is a world leader in computational music
  – Mario Berges has helped in the past
  – Rita Singh does nice work on speech forensics
  – Others…

Questions?

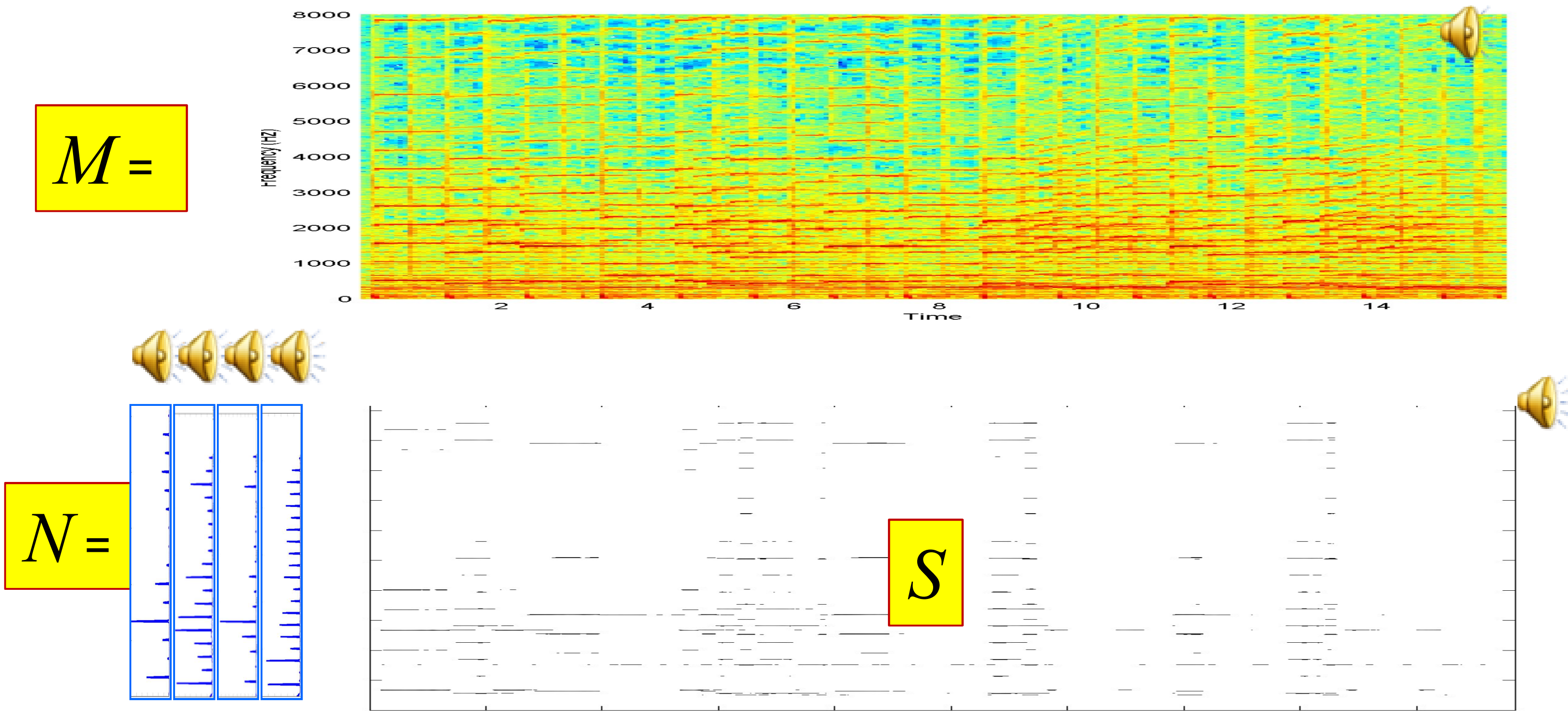
Index

1. The problem of optimization
2. Direct optimization
3. Descent methods
   – Newton's method
   – Gradient methods
4. Online optimization
5. Constrained optimization
   – Lagrange's method
   – Projected gradients
6. Regularization
7. Convex optimization and Lagrangian duals

Index

1. The problem of optimization
2. Direct optimization
3. Descent methods
   – Newton's method
   – Gradient methods
4. Online optimization
5. Constrained optimization
   – Lagrange's method
   – Projected gradients
6. Regularization
7. Convex optimization and Lagrangian duals

A problem we recently saw



- The projection matrix P is the matrix that minimizes the total error between the *projected* matrix S and the *original matrix* M

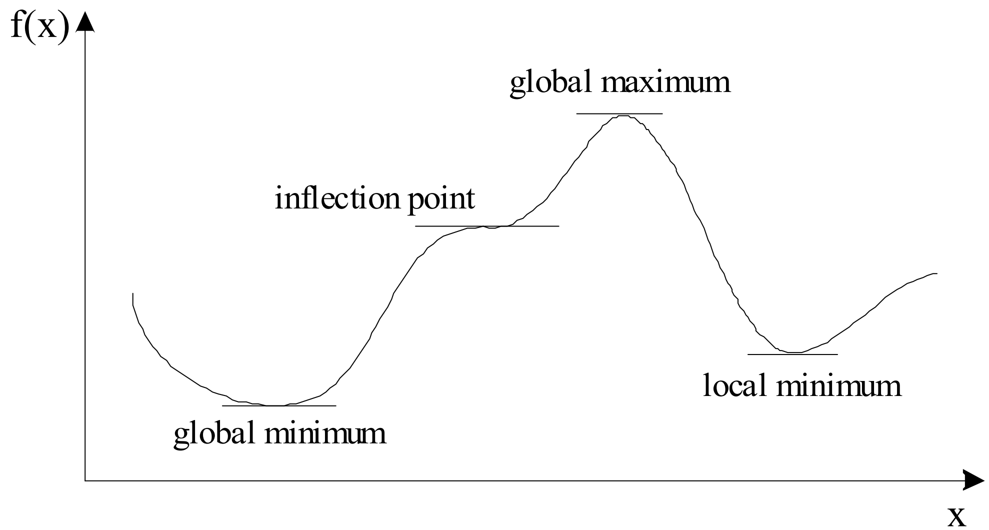
The projection problem

- $S = PM$
- For individual vectors in the spectrogram
  - $S_i = PM_i$
- Total projection error is
  - $E = \sum_i \|M_i - PM_i\|^2$
- The projection matrix projects onto the space of notes in N
  - $P = NC$
- The problem of finding P : Minimize $E = \sum_i \|M_i - PM_i\|^2$ such that $P = NC$
- This is a problem of *constrained optimization*

Optimization

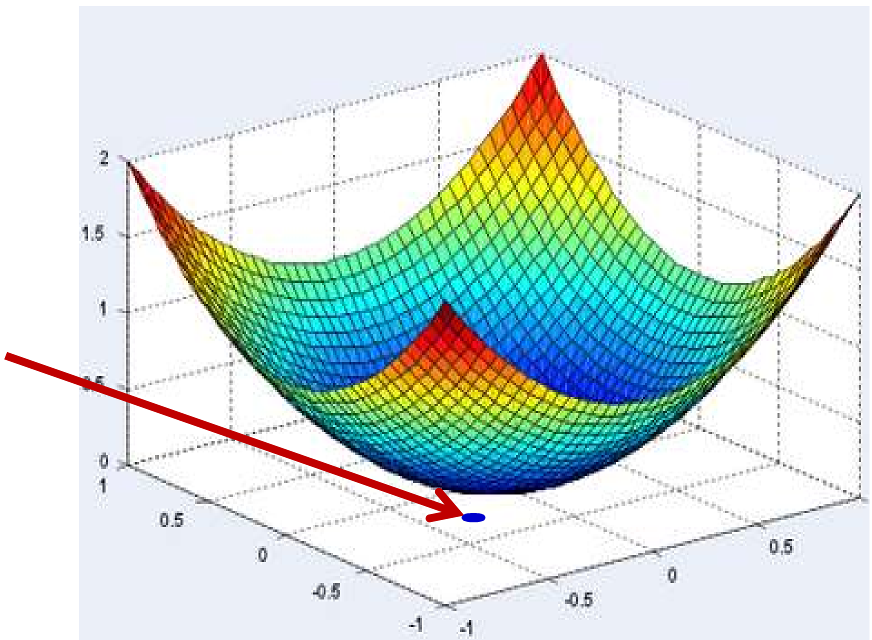
- Optimization is finding the "best" value of a function $f(x)$ (which can be the best minimum)

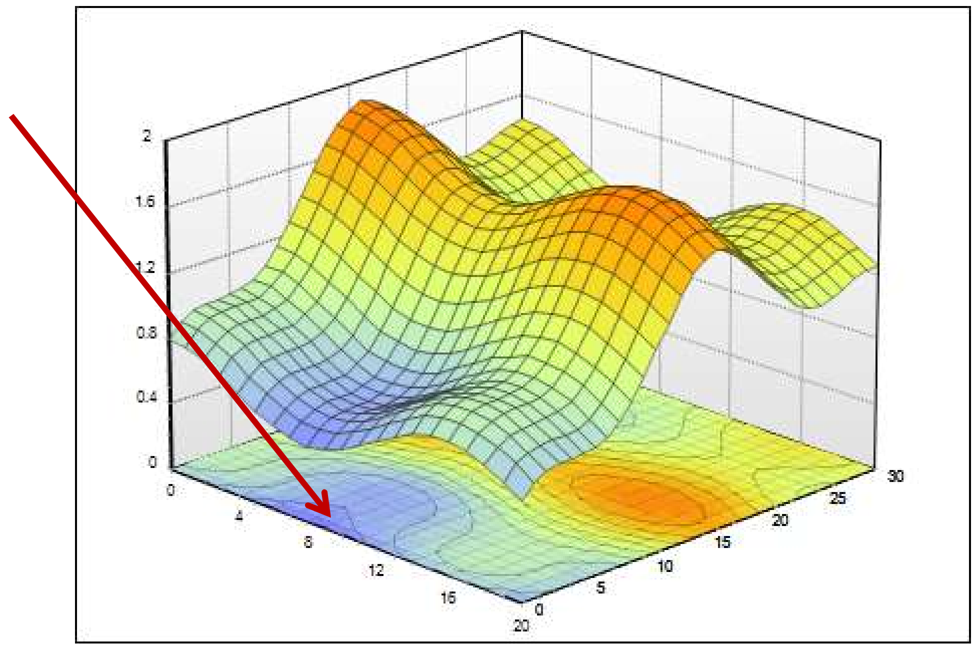
$$\min_{x} f(x)$$



Examples of Optimization : Multivariate functions

- Find the optimal point in these functions

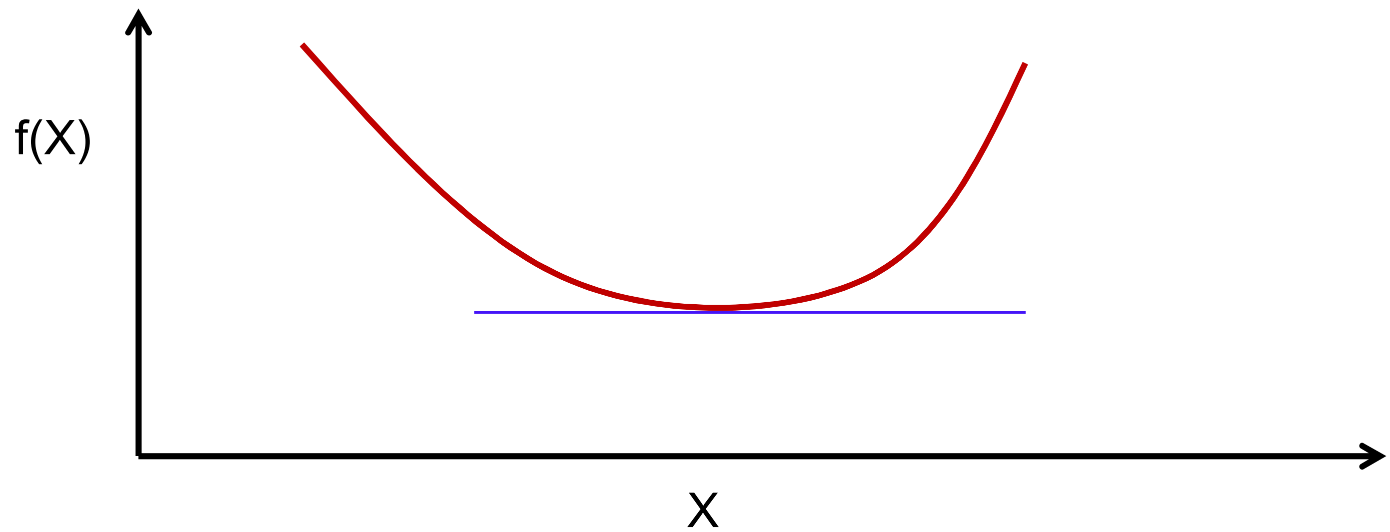




Index

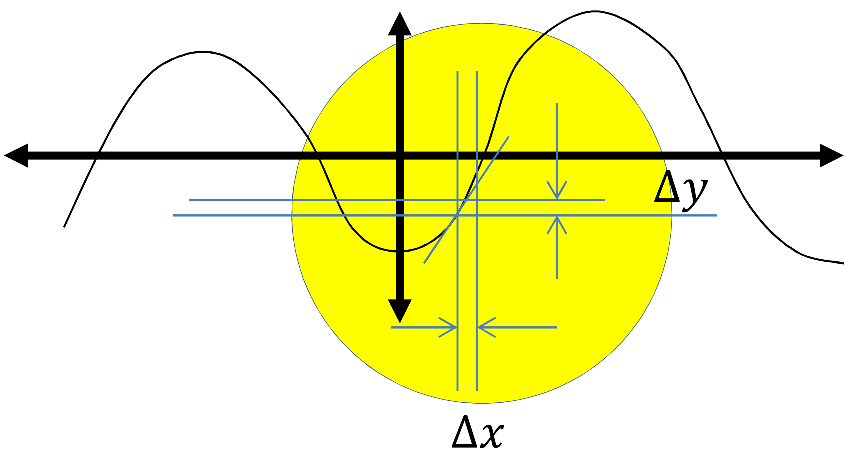
1. The problem of optimization
2. Direct optimization
3. Descent methods
   – Newton's method
   – Gradient methods
4. Online optimization
5. Constrained optimization
   – Lagrange's method
   – Projected gradients
6. Regularization
7. Convex optimization and Lagrangian duals

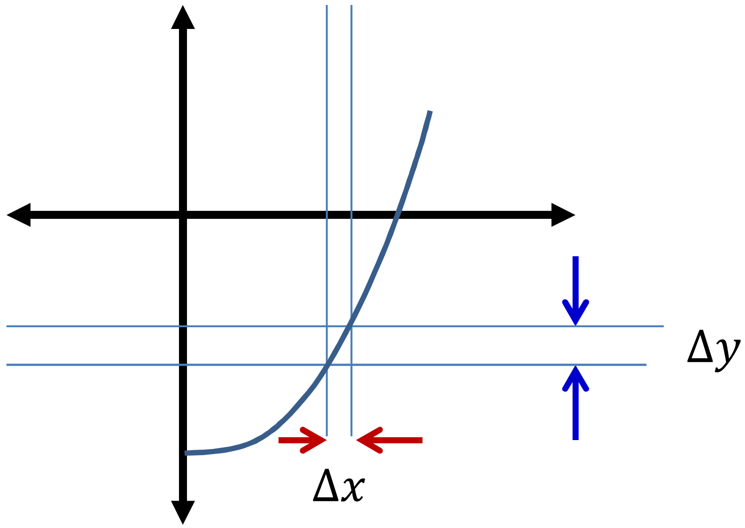
Simple Approach: Turning Point



- The "minimum" of the function is always a "turning point"
  - Points where the function "turns" around
    - In every direction
  - For minima, the function increases on either side
- How to identify these turning points?

The “derivative” of a curve





- The derivative α_x of a curve is a multiplicative factor explaining how much y changes in response to a very small change in x

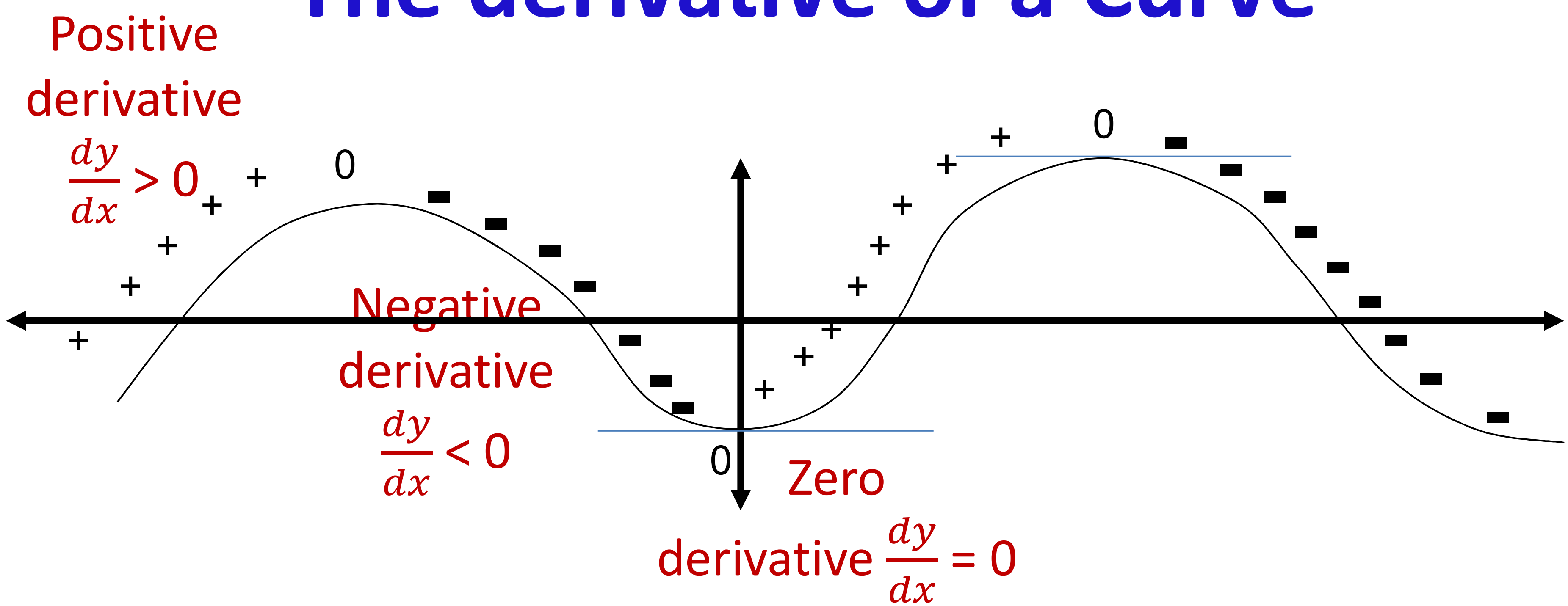
$$\Delta y = \alpha_x \Delta x$$

- For scalar functions of scalar variables, often expressed as $\frac{dy}{dx}$ or as $f'(x)$

$$\Delta y = \frac{dy}{dx}\Delta x \qquad\qquad \Delta y = f'(x)\Delta x$$

- We have all learned how to compute derivatives in basic calculus

The derivative of a Curve



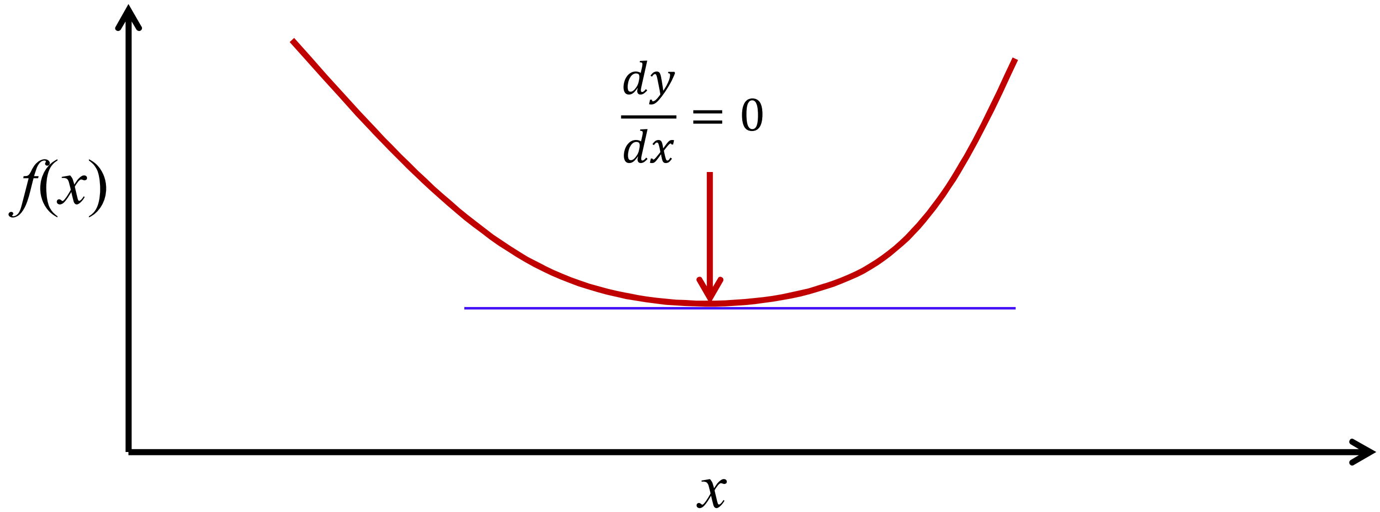
- In upward-rising regions of the curve, the derivative is positive
  - Small increase in X cause Y to increase
- In downward-falling regions, the derivative is negative
- *At turning points, the derivative is 0*
  - *Assumption: the function is differentiable at the turning point*

Geometrical application of Calculus to the derivative of a curve

- Find all values of x for which $f(x) = x^2 - 4x + 4$ is increasing, decreasing and stationary

Increasing	**Decreasing**	**Stationary**
$f(x) = x^2 - 4x + 4$	$f(x) = x^2 - 4x + 4$	$f(x) = x^2 - 4x + 4$
$f'(\text{x}) = 2\text{x} - 4$	$f'(\text{x}) = 2\text{x} - 4$	$f'(\text{x}) = 2\text{x} - 4$
$2\text{x} - 4 > 0$	$2\text{x} - 4 < 0$	$2\text{x} - 4 = 0$
$2\text{x} > 4$	$2\text{x} < 4$	$2\text{x} = 4$
$\text{x} > 2$	$\text{x} < 2$	$\text{x} = 2$

Finding the minimum of a function

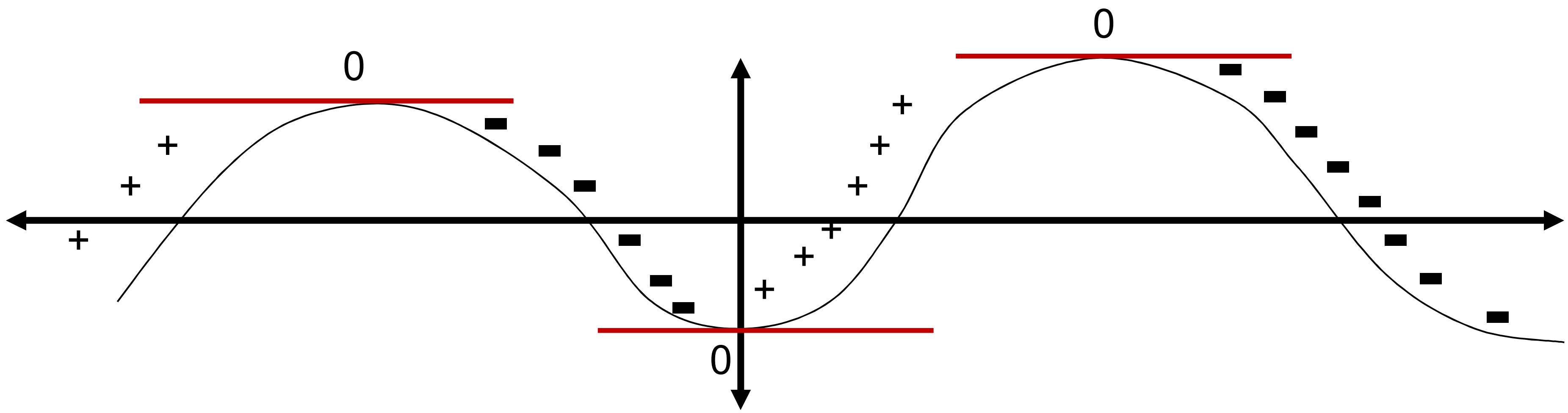


- Find the value x at which $f'(x) = 0$
  - Solve

$$\frac{df(x)}{dx} = 0$$

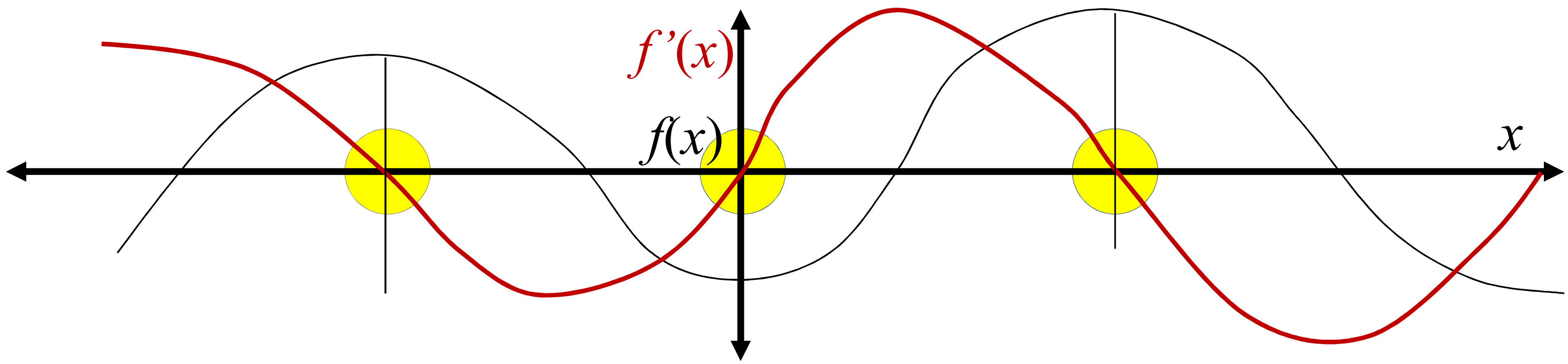
- The solution is a turning point
- But is it a minimum?

Turning Points



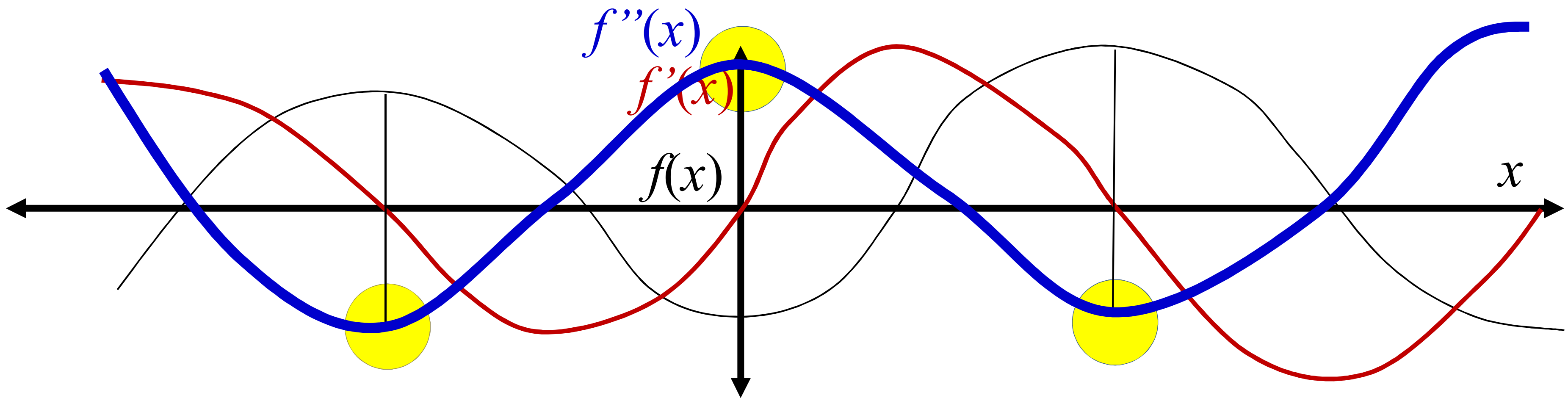
- Both *maxima* and *minima* have zero derivative
  - *Both maxima and minima are turning points*

Derivatives of a curve



- Both *maxima* and *minima* are turning points
- Both *maxima* and *minima* have zero derivative

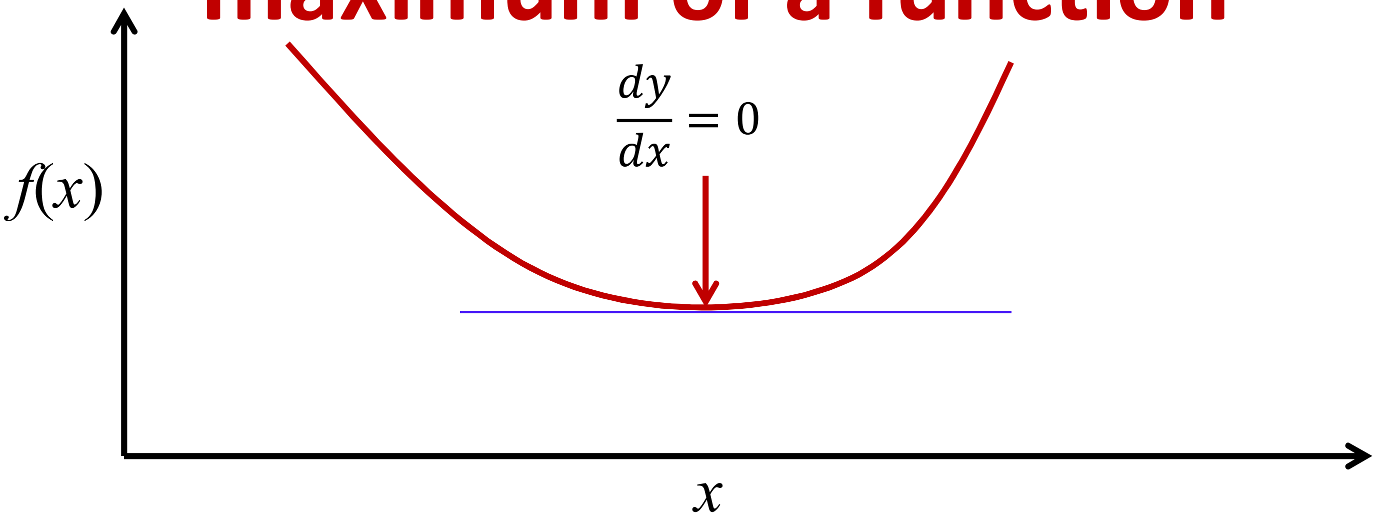
Derivative of the derivative of the curve



- Both *maxima* and *minima* are turning points
- Both *maxima* and *minima* have zero derivative

- The *second derivative* $f''(x)$ is –ve at maxima and +ve at minima!
  - At maxima the derivative goes from +ve to –ve, so the derivative decreases as x increases
  - At minima the derivative goes from –ve to +ve and increases as x increases

Soln: Finding the minimum or maximum of a function



- Find the value x at which $f'(x) = 0$: Solve

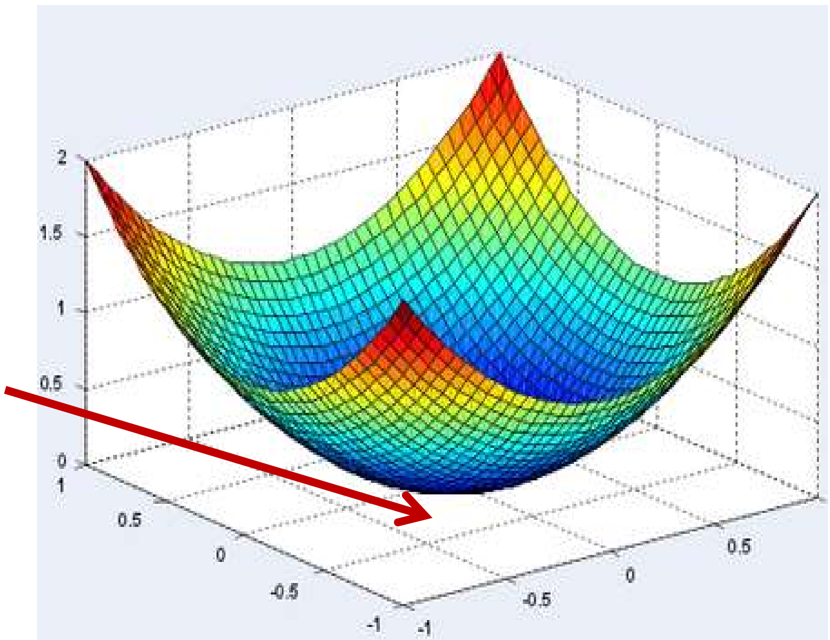
$$\frac{df(x)}{dx} = 0$$

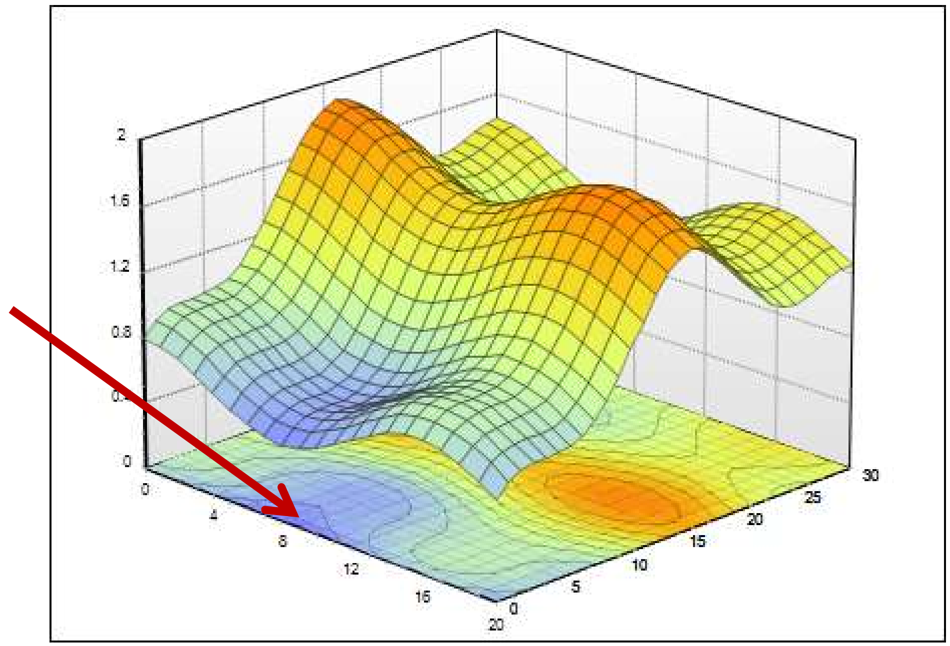
- The solution x_{soln} is a turning point
- Check the double derivative at x_{soln} : compute

$$f''(x_{soln}) = \frac{df'(x_{soln})}{dx}$$

- If $f''(x_{soln})$ is positive x_{soln} is a minimum, otherwise it is a maximum

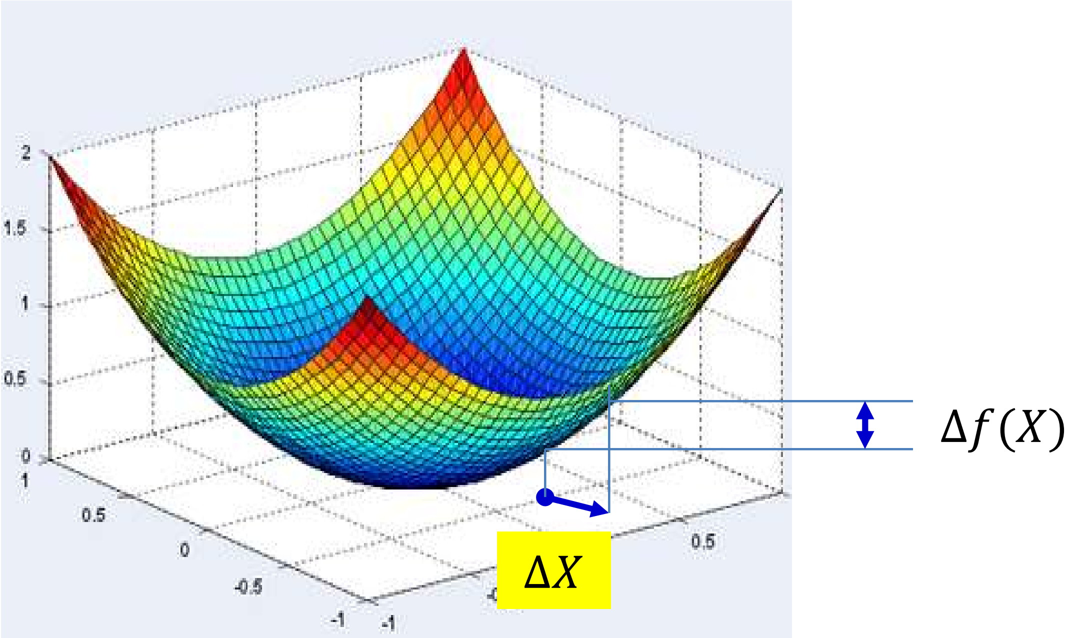
What about functions of multiple variables?





- The optimum point is still "turning" point
  - Shifting in any direction will increase the value
  - For smooth functions, miniscule shifts will not result in any change at all
- We must find a point where shifting in any direction by a microscopic amount will not change the value of the function

The *Gradient* of a scalar function

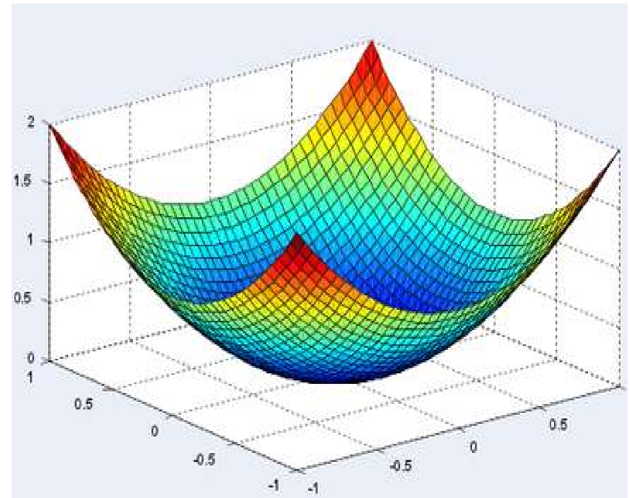


- The *Gradient* $\nabla f(X)$ of a scalar function $f(X)$ of a multi-variate input X is a multiplicative factor that gives us the change in $f(X)$ for tiny variations in X

$$\Delta f(X) = \nabla f(X)^T \Delta X$$

Gradients of scalar functions with multi-variate inputs

- Consider $f(X) = f(x_1, x_2, \ldots, x_n)$



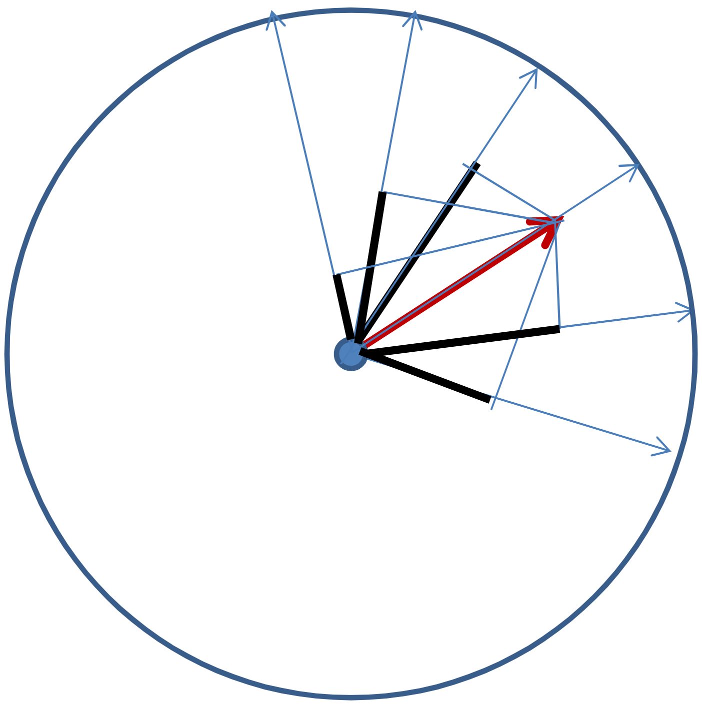
$$\nabla f(X) = \begin{bmatrix} \frac{\partial f(X)}{\partial x_1} \\ \frac{\partial f(X)}{\partial x_2} \\ \vdots \\ \frac{\partial f(X)}{\partial x_n} \end{bmatrix}$$

- Check:

$$\Delta f(X) = \nabla f(X)^T \Delta X$$

$$= \frac{\partial f(X)}{\partial x_1}\Delta x_1 + \frac{\partial f(X)}{\partial x_2}\Delta x_2 + \cdots + \frac{\partial f(X)}{\partial x_n}\Delta x_n$$

A well-known vector property



$$\mathbf{u}.\mathbf{v} = |\mathbf{u}||\mathbf{v}|cos\theta$$

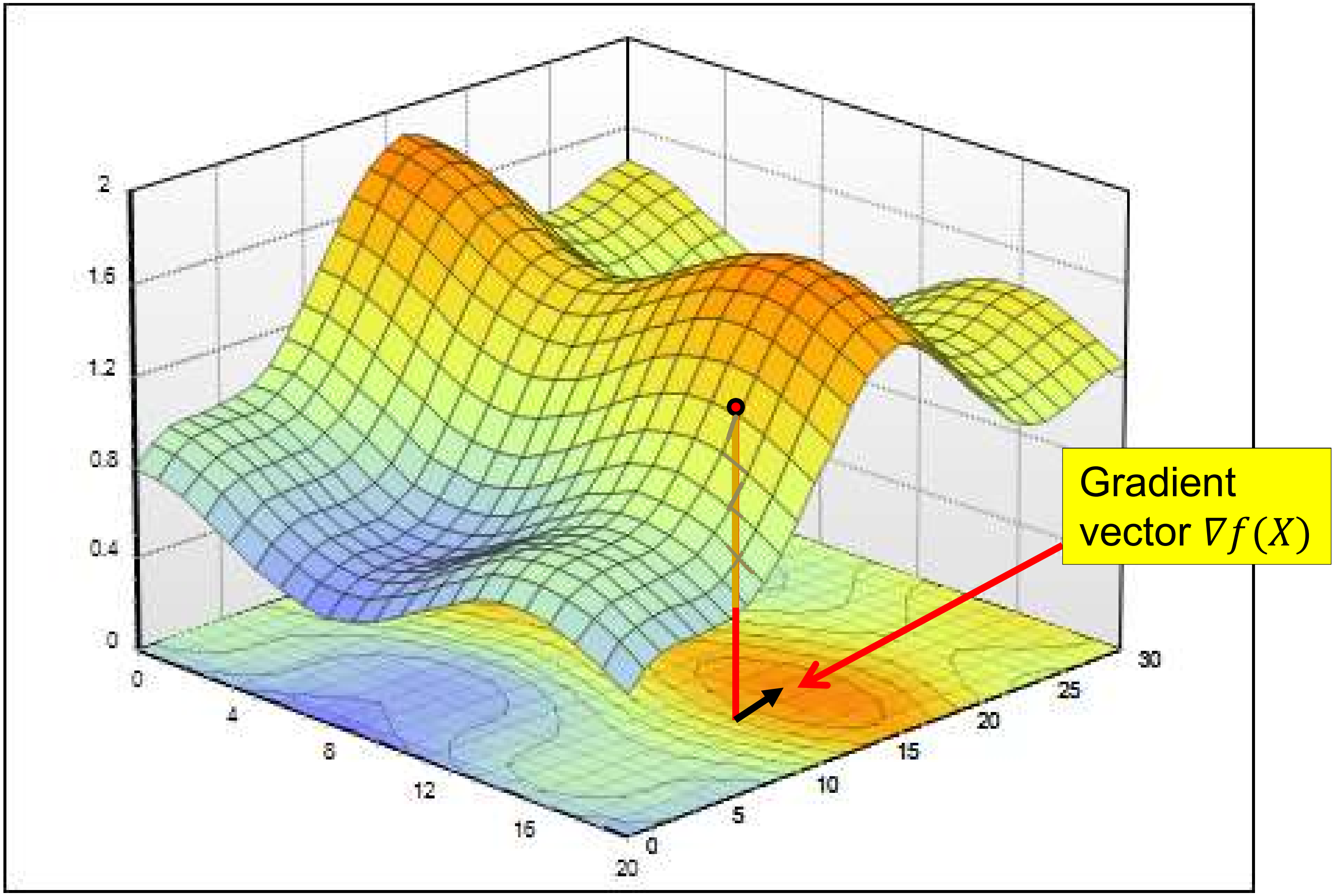
- The inner product between two vectors of fixed lengths is maximum when the two vectors are aligned

Properties of Gradient

- $\Delta f(X) = \nabla f(X)^T \Delta X$
  - The inner product between $\nabla f(X)$ and ΔX

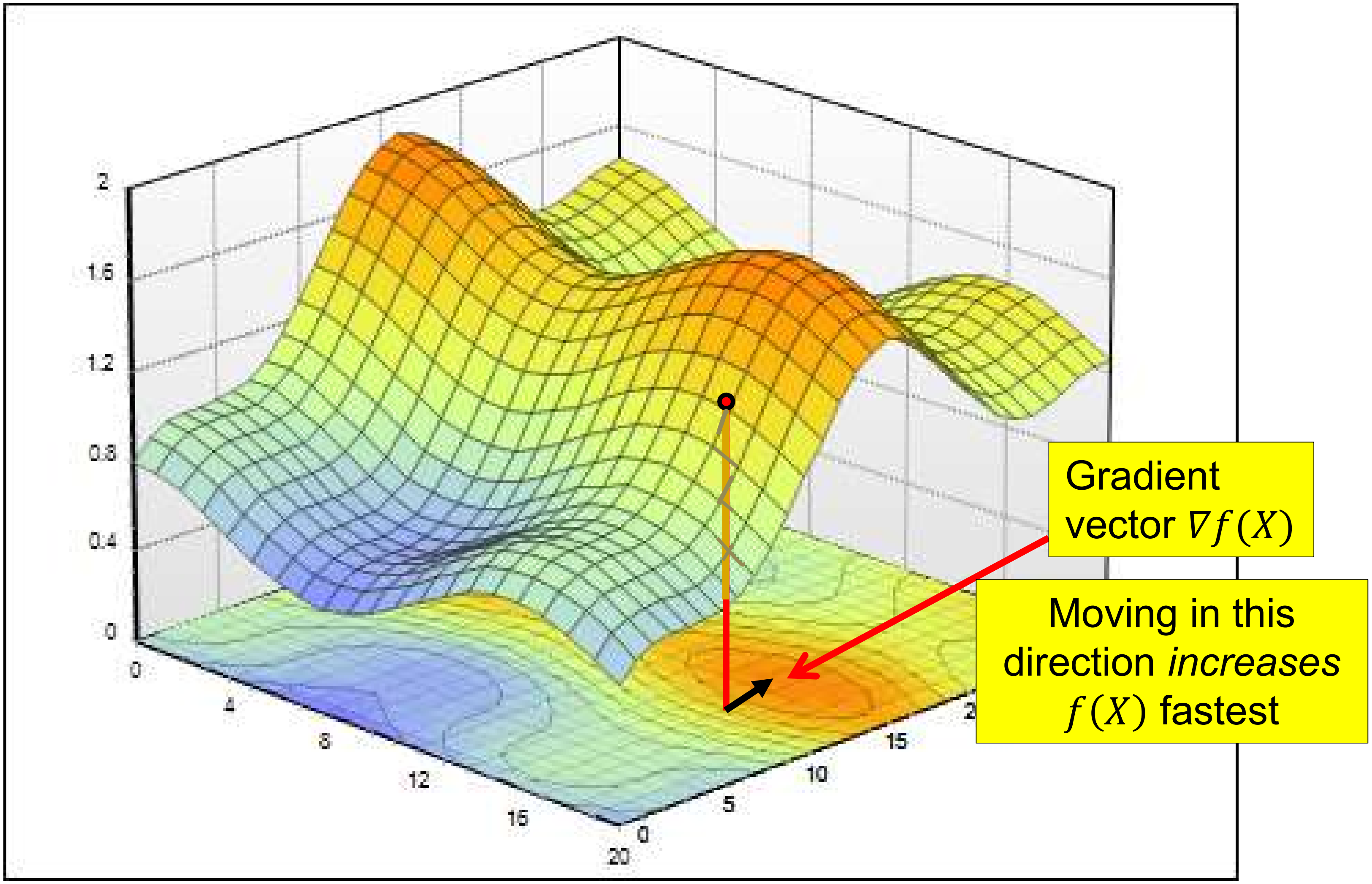
- Fixing the length of ΔX
  - E.g. $|\Delta X| = 1$
- $\Delta f(X)$ is max if $\angle \nabla f(X), \Delta X = 0$
  - The function $f(X)$ increases most rapidly if the input increment ΔX is perfectly aligned to $\nabla f(X)$
- **The gradient is the direction of fastest increase in $f(X)$**

Gradient

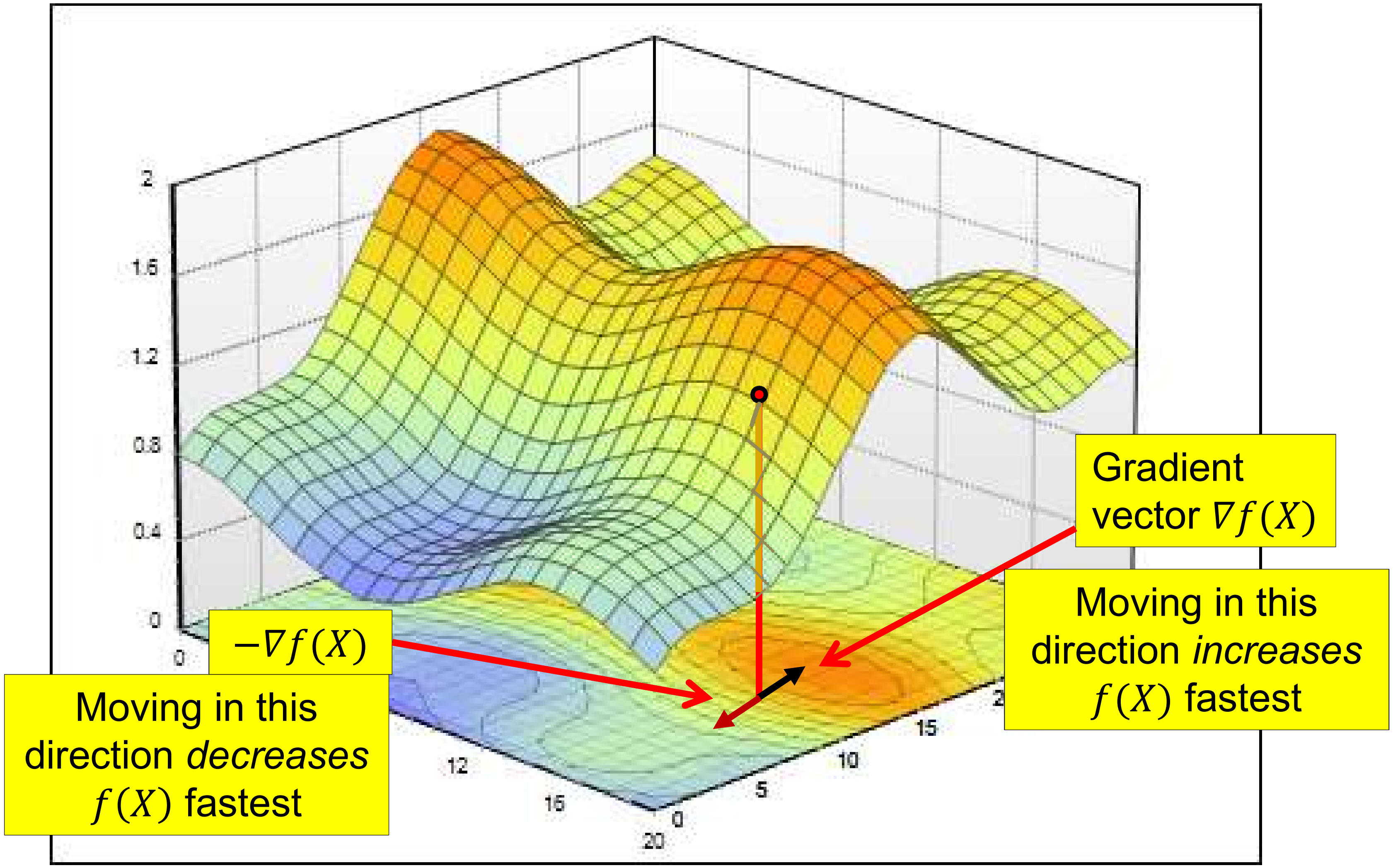


2
1.6
1.2
0.8
0.4
0
0
4
8
12
16
20
0
5
10
15
20
25
30
Gradient vector $\nabla f(X)$

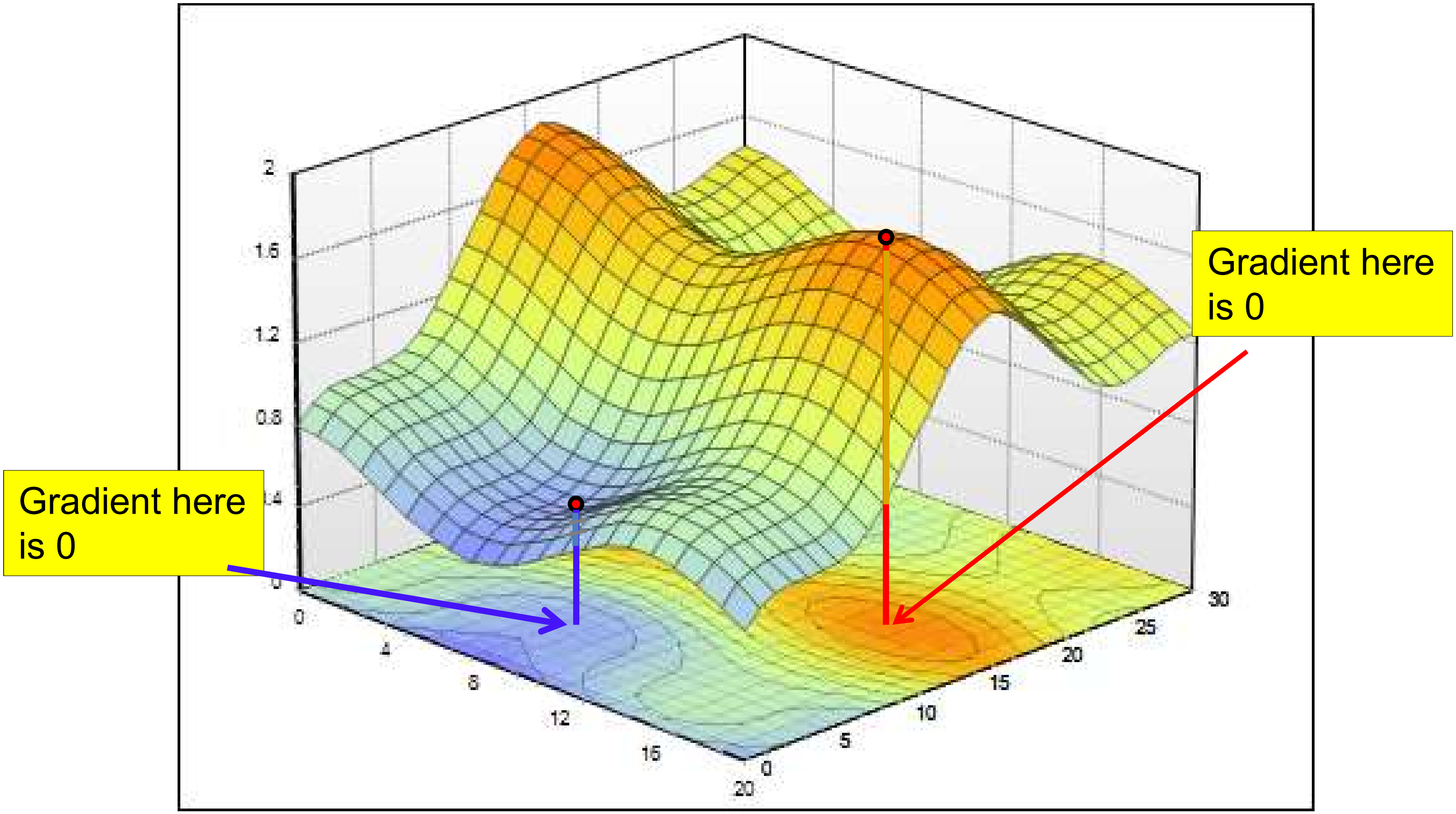
Gradient


2
1.6
1.2
0.8
0.4
0
0
4
8
12
16
20
0
5
10
15
Gradient vector $\nabla f(X)$
Moving in this direction *increases* $f(X)$ fastest

Gradient

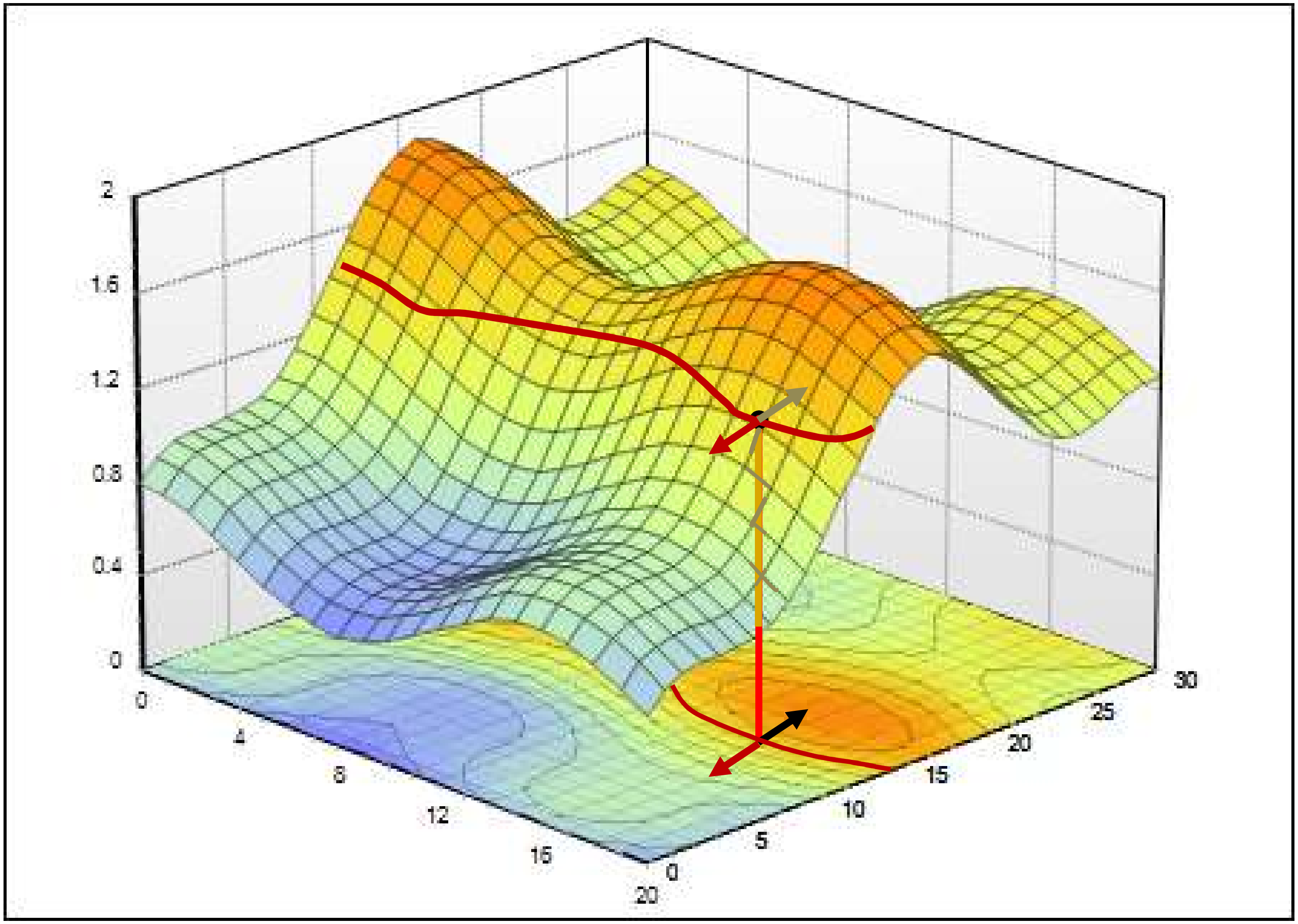

Gradient vector $\nabla f(X)$
Moving in this direction *increases* $f(X)$ fastest
$-\nabla f(X)$
Moving in this direction *decreases* $f(X)$ fastest
2
1.6
1.2
0.8
0.4
0
0
12
16
20
0
5
10
15

Gradient



Gradient here is 0
Gradient here is 0
2
1.6
1.2
0.8
0
4
8
12
16
20
0
5
10
15
20
25
30

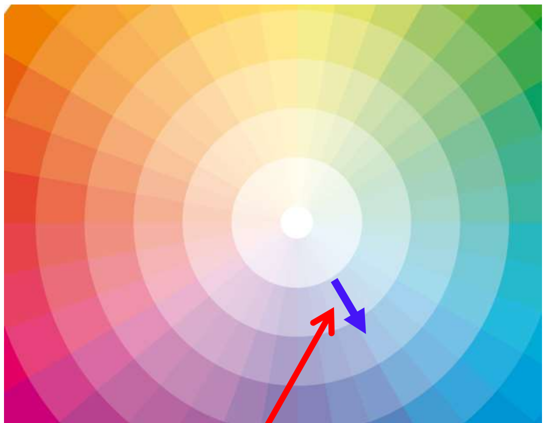
Properties of Gradient: 2



- The gradient vector $\nabla f(X)$ is perpendicular to the level curve

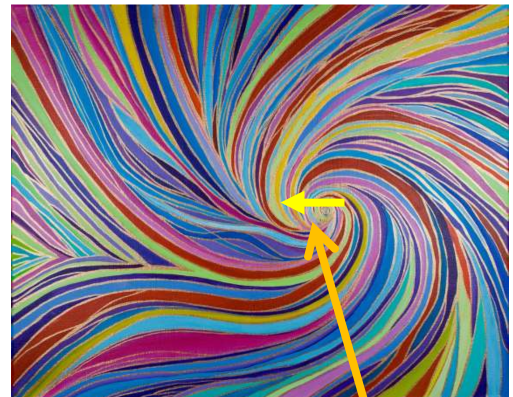
Derivatives of vector function of vector input

$X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$



ΔX

$Y = f(X)$



$Y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$

$\Delta f(X)$

- The *Gradient* $\nabla f(X)$ of a *vector* function $f(X)$ of a multi-variate input X is a multiplicative factor that gives us the change in $f(X)$ for tiny variations in X

$$\Delta f(X) = \nabla f(X)^T \Delta X$$

"Gradient" of vector function of vector input

$$X = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{bmatrix} \qquad f(X) = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_m \end{bmatrix}$$

$$\nabla f(X)^T = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & . & . & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & . & . & \frac{\partial y_2}{\partial x_n} \\ . & . & . & . & . \\ . & . & . & . & . \\ \frac{\partial y_n}{\partial x_1} & \frac{\partial y_n}{\partial x_2} & . & . & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

Properties and interpretations are similar to the case of scalar functions of vector inputs

Chain rule

- The gradient is based on derivatives
- The derivative of composed function $f(g(x))$ or $f \circ g$ can be very complicated to compute
- If $f \circ g$ is the composite of $y = f(u)$ and $u = g(x)$

  Then $(f \circ g)' = f'_{at\ u=g(x)} \bullet g'_{at\ x}$ or $\frac{dy}{dx} = \frac{dy}{du} \cdot \frac{du}{dx}$
- This is known as Chain rule

Example of chain rule

- Differentiate $h(x) = \left(\frac{8x - x^6}{x^3} \right)^{-\frac{4}{5}}$
- Simplification

$$h(x) = \left(\frac{8x - x^6}{x^3} \right)^{-\frac{4}{5}} = \left(\frac{8x}{x^3} - \frac{x^6}{x^3} \right)^{-\frac{4}{5}} = \left(8x^{-2} - x^3 \right)^{-\frac{4}{5}}$$

- Applying Chain rule

$$y = f(u) = (u)^{-\frac{4}{5}} \qquad u = g(x) = 8x^{-2} - x^3$$

Example of chain rule

- Applying Chain rule

$$h(x) = \left(-\frac{4}{5}\right)\left(8x^{-2} - x^{3}\right)^{-\frac{4}{5}-1}(-8x^{-2} - x^{3})'$$

$$h(x) = \left(-\frac{4}{5}\right)\left(8x^{-2} - x^{3}\right)^{-\frac{9}{5}}(-16x^{-3} - 3x^{2})$$

- After simplification

$$h(x) = \frac{4x^{\frac{9}{5}}(16 + 3x^{5})}{5(8 - x^{5})^{\frac{9}{5}}}$$

Vector and Matrix derivatives

- The derivative of vector $x = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{bmatrix}$ by a scalar y is given by

$$\frac{\partial x}{\partial y} = \begin{bmatrix} \frac{\partial x_1}{\partial y} \\ \frac{\partial x_2}{\partial y} \\ . \\ . \\ \frac{\partial x_n}{\partial y} \end{bmatrix}$$

Vector and Matrix derivatives

- The derivative of scalar y by a vector $x = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{bmatrix}$ is given by

$$\frac{\partial y}{\partial x} = \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ . \\ . \\ \frac{\partial y}{\partial x_n} \end{bmatrix}$$

Vector and Matrix derivatives

- The derivative of vector $x = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{bmatrix}$ by a vector $y = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_m \end{bmatrix}$ is given by

$$\frac{\partial x}{\partial y} = \begin{bmatrix} \frac{\partial x_1}{\partial y_1} & \frac{\partial x_2}{\partial y_1} & . & . & \frac{\partial x_n}{\partial y_1} \\ \frac{\partial x_2}{\partial y_1} & \frac{\partial x_2}{\partial y_2} & . & . & \frac{\partial x_2}{\partial y_m} \\ . & . & . & . & . \\ . & . & . & . & . \\ \frac{\partial x_n}{\partial y_1} & \frac{\partial x_n}{\partial y_2} & . & . & \frac{\partial x_n}{\partial y_m} \end{bmatrix}$$

Vector and Matrix derivatives

- The derivative of matrix $X = \begin{bmatrix} x_{1,1} & x_{1,2} & . & . & x_{1,n} \\ x_{2,1} & x_{2,2} & . & . & x_{2,n} \\ . & . & . & . & . \\ . & . & . & . & . \\ x_{m,1} & x_{m,2} & . & . & x_{m,n} \end{bmatrix}$

by a scalar y is given by

$$\frac{\partial X}{\partial y} = \begin{bmatrix} \frac{\partial x_{1,1}}{\partial y} & \frac{\partial x_{1,2}}{\partial y} & . & . & \frac{\partial x_{1,n}}{\partial y} \\ \frac{\partial x_{2,1}}{\partial y} & \frac{\partial x_{2,2}}{\partial y} & . & . & \frac{\partial x_{2,n}}{\partial y} \\ . & . & . & . & . \\ . & . & . & . & . \\ \frac{\partial x_{m,1}}{\partial y} & \frac{\partial x_{m,2}}{\partial y} & . & . & \frac{\partial x_{m,n}}{\partial y} \end{bmatrix}$$

Vector and Matrix derivatives

- The derivative a scalar y by a matrix

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & . & . & x_{1,n} \\ x_{2,1} & x_{2,2} & . & . & x_{2,n} \\ . & . & . & . & . \\ . & . & . & . & . \\ x_{m,1} & x_{m,2} & . & . & x_{m,n} \end{bmatrix}$$

is given by

$$\frac{\partial y}{\partial X} = \begin{bmatrix} \frac{\partial y}{\partial x_{1,1}} & \frac{\partial y}{\partial x_{1,2}} & . & . & \frac{\partial y}{\partial x_{1,n}} \\ \frac{\partial y}{x_{2,1}} & \frac{\partial y}{\partial x_{2,2}} & . & . & \frac{\partial y}{\partial x_{2,n}} \\ . & . & . & . & . \\ . & . & . & . & . \\ \frac{\partial y}{\partial x_{m,1}} & \frac{\partial y}{\partial x_{m,2}} & . & . & \frac{\partial y}{\partial x_{m,n}} \end{bmatrix}$$

Vector and Matrix derivatives

- The derivative of vector x of n elements by a matrix Y of size (p,q) is given by

$$\frac{\partial x}{\partial Y} = \begin{bmatrix} \frac{\partial x}{\partial y_{1,1}} & \frac{\partial x}{\partial y_{1,2}} & . & . & \frac{\partial x}{\partial y_{1,q}} \\ \frac{\partial x}{\partial y_{2,1}} & \frac{\partial x}{\partial y_{2,2}} & . & . & \frac{\partial x}{\partial y_{2,q}} \\ . & . & . & . & . \\ . & . & . & . & . \\ \frac{\partial x}{\partial y_{p,1}} & \frac{\partial x}{\partial y_{p,2}} & . & . & \frac{\partial x}{\partial y_{p,q}} \end{bmatrix}$$

$\frac{\partial x}{\partial y_{i,j}}$ Is the derivative of the vector x by the scalar $y_{i,j}$ which is an element of the matrix Y

Vector and Matrix derivatives

- The derivative of matrix X of size (m,n) by another matrix Y of size (p,q) is given by

$$\frac{\partial X}{\partial Y} = \begin{bmatrix} \frac{\partial X}{\partial y_{1,1}} & \frac{\partial X}{\partial y_{1,2}} & . & . & \frac{\partial X}{\partial y_{1,q}} \\ \frac{\partial X}{\partial y_{2,1}} & \frac{\partial X}{\partial y_{2,2}} & . & . & \frac{\partial X}{\partial y_{2,q}} \\ . & . & . & . & . \\ . & . & . & . & . \\ \frac{\partial X}{\partial y_{p,1}} & \frac{\partial X}{\partial y_{p,2}} & . & . & \frac{\partial X}{\partial y_{p,q}} \end{bmatrix}$$

$\frac{\partial X}{\partial y_{i,j}}$ Is the derivative of the matrix X by the scalar $y_{i,j}$ which is an element of the matrix Y

Gradient Example

- Compute the Gradient of the function

$$f(x_1, x_2, x_3) = 15x_1 + 2(x_2)^2 - 3x_1(x_3)$$

$$\nabla f(x_1, x_2, x_3) := \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \frac{\partial f}{\partial x_3} \end{bmatrix}$$

$$\nabla f(x_1, x_2, x_3) := \begin{bmatrix} 15 - 3(x_3)^2 & 6(x_2)^2 & -6x_1 x_3 \end{bmatrix}$$

The Hessian

- The Hessian of a function $f(x_1, x_2, \ldots, x_n)$ is given by the second derivative

$$\nabla^2 f(x_1, \ldots, x_n) := \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & . & . & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & . & . & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ . & . & . & . & . \\ . & . & . & . & . \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & . & . & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Hessian Example

- Compute the Hessian of the function

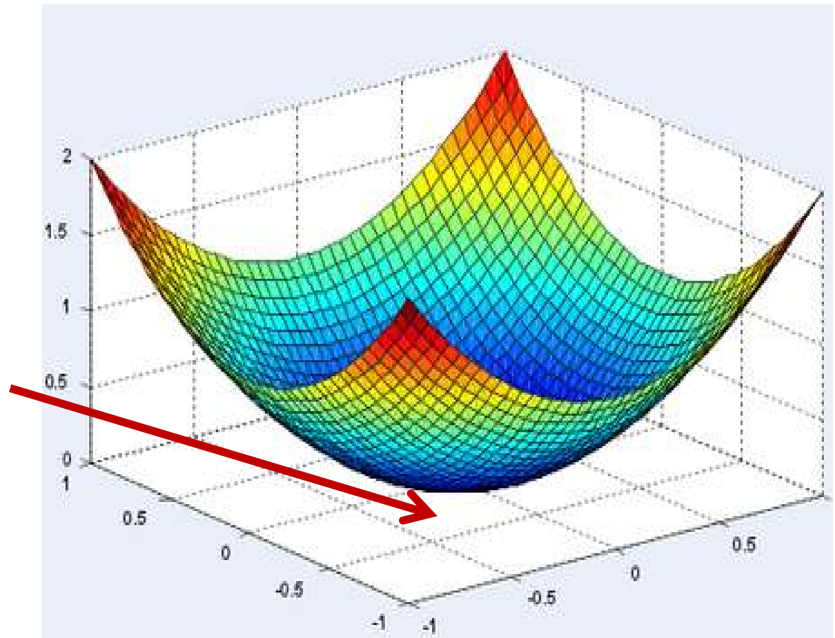
$$f(x_1, x_2, x_3) = 15x_1 + 2(x_2)^2 - 3x_1(x_3)$$

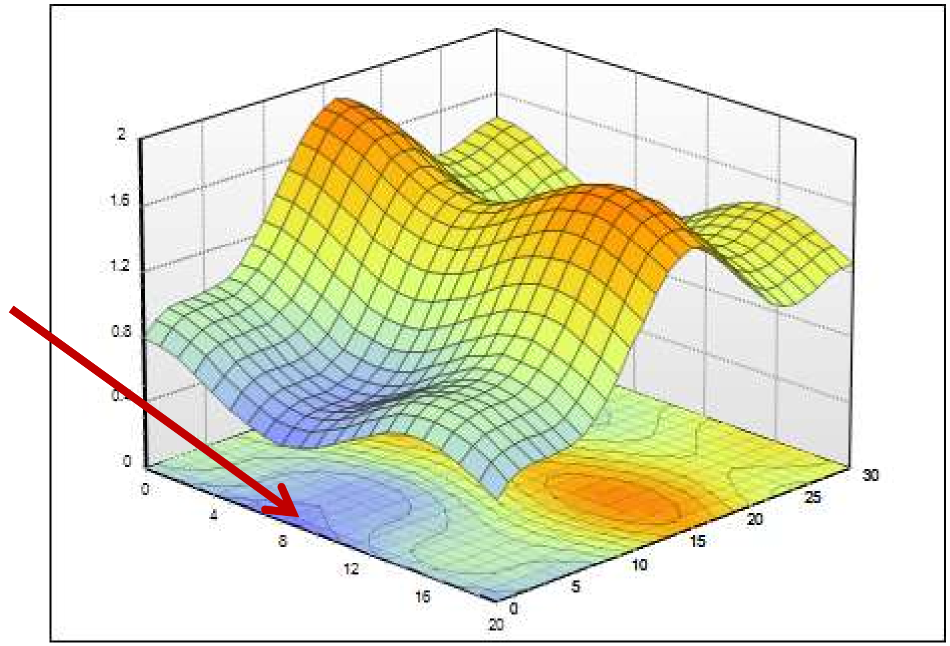
$$\nabla f(x_1, x_2, x_3) := \begin{bmatrix} 15 - 3(x_3)^2 & 6(x_2)^2 & -6x_1x_3 \end{bmatrix}$$

$$\nabla^2 f(x_1, x_2, x_3) := \begin{bmatrix} 0 & 0 & -6x_3 \\ 0 & 12x_2 & 0 \\ -6x_3 & 0 & -6x_1 \end{bmatrix}$$

Returning to direct optimization…

Finding the minimum of a scalar function of a multi-variate input





- The optimum point is a turning point – the gradient will be 0

Unconstrained Minimization of function (Multivariate)

1. Solve for the X where the gradient equation equals to zero

$$\nabla f(X) = 0$$

2. Compute the Hessian Matrix $\nabla^2 f(X)$ at the candidate solution and verify that
   – Hessian is positive definite (eigenvalues positive) -> to identify local minima
   – Hessian is negative definite (eigenvalues negative) -> to identify local maxima

Unconstrained Minimization of function (Example)

- Minimize

$$f(x_1, x_2, x_3) = (x_1)^2 + x_1(1 - x_2) - (x_2)^2 - x_2x_3 + (x_3)^2 + x_3$$

- Gradient

$$\nabla f = \begin{bmatrix} 2x_1 + 1 - x_2 \\ -x_1 + 2x_2 - x_3 \\ -x_2 + 2x_3 + 1 \end{bmatrix}$$

Unconstrained Minimization of function (Example)

- Set the gradient to null

$$\nabla f = 0 \Rightarrow \begin{bmatrix} 2x_1 + 1 - x_2 \\ -x_1 + 2x_2 - x_3 \\ -x_2 + 2x_3 + 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

- Solving the 3 equations system with 3 unknowns

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

Unconstrained Minimization of function (Example)

- Compute the Hessian matrix $\nabla^2 f = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}$
- Evaluate the eigenvalues of the Hessian matrix

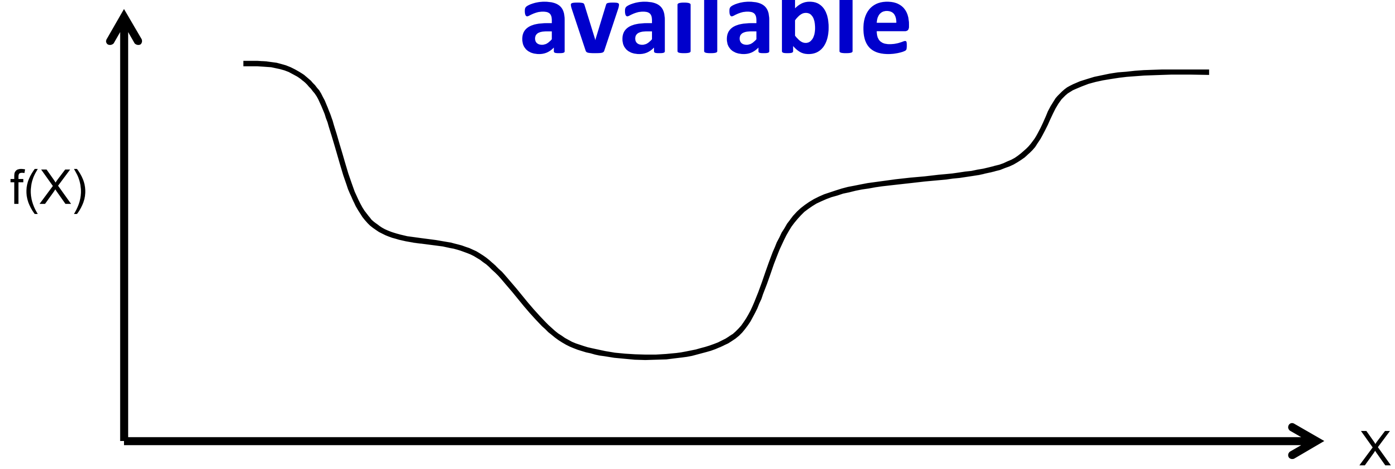
$$\lambda_1 = 3.414, \;\; \lambda_2 = 0.586, \;\; \lambda_3 = 2$$

- All the eigenvalues are positives => the Hessian matrix is positive definite
- The point $x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$ is a minimum

Index

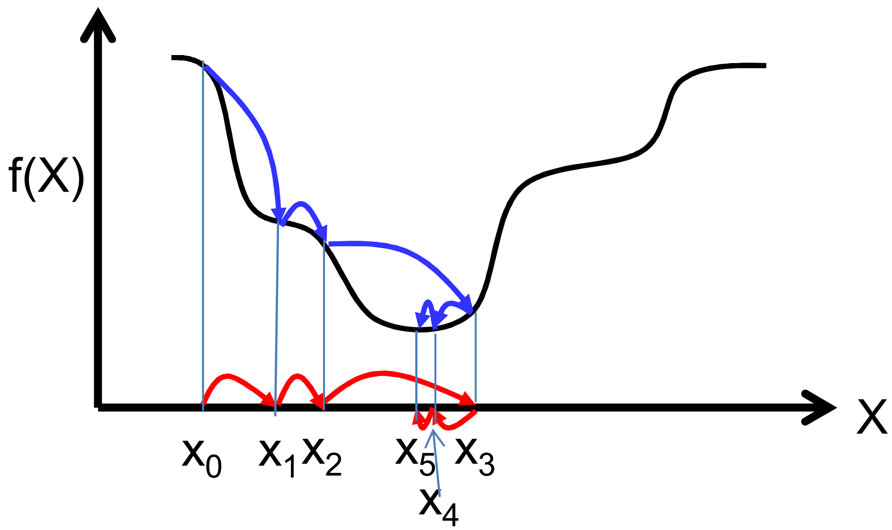
1. The problem of optimization
2. Direct optimization
3. Descent methods
   – Newton's method
   – Gradient methods
4. Online optimization
5. Constrained optimization
   – Lagrange's method
   – Projected gradients
6. Regularization
7. Convex optimization and Lagrangian duals

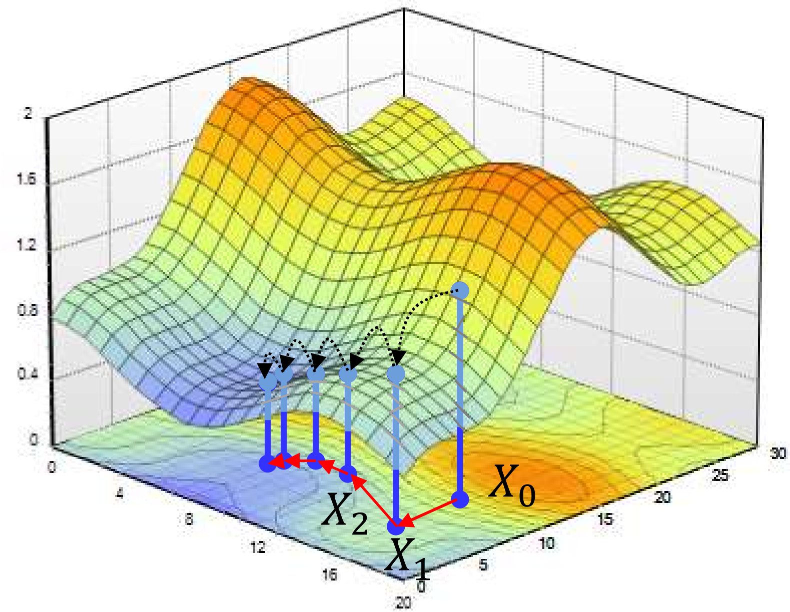
Closed Form Solutions are not always available



- Often it is not possible to simply solve $\nabla f(X) = 0$
  - The function to minimize/maximize may have an intractable form
- In these situations, iterative solutions are used
  - Begin with a "guess" for the optimal X and refine it iteratively until the correct value is obtained

Iterative solutions





- Iterative solutions
  - Start from an initial guess X_0 for the optimal X
  - Update the guess towards a (hopefully) "better" value of $\mathrm{f}(X)$
  - Stop when $\mathrm{f}(X)$ no longer decreases
- Problems:
  - Which direction to step in
  - How big must the steps be

Descent methods

- Iterative solutions that attempt to "descend" the function in steps to arrive at the minimum

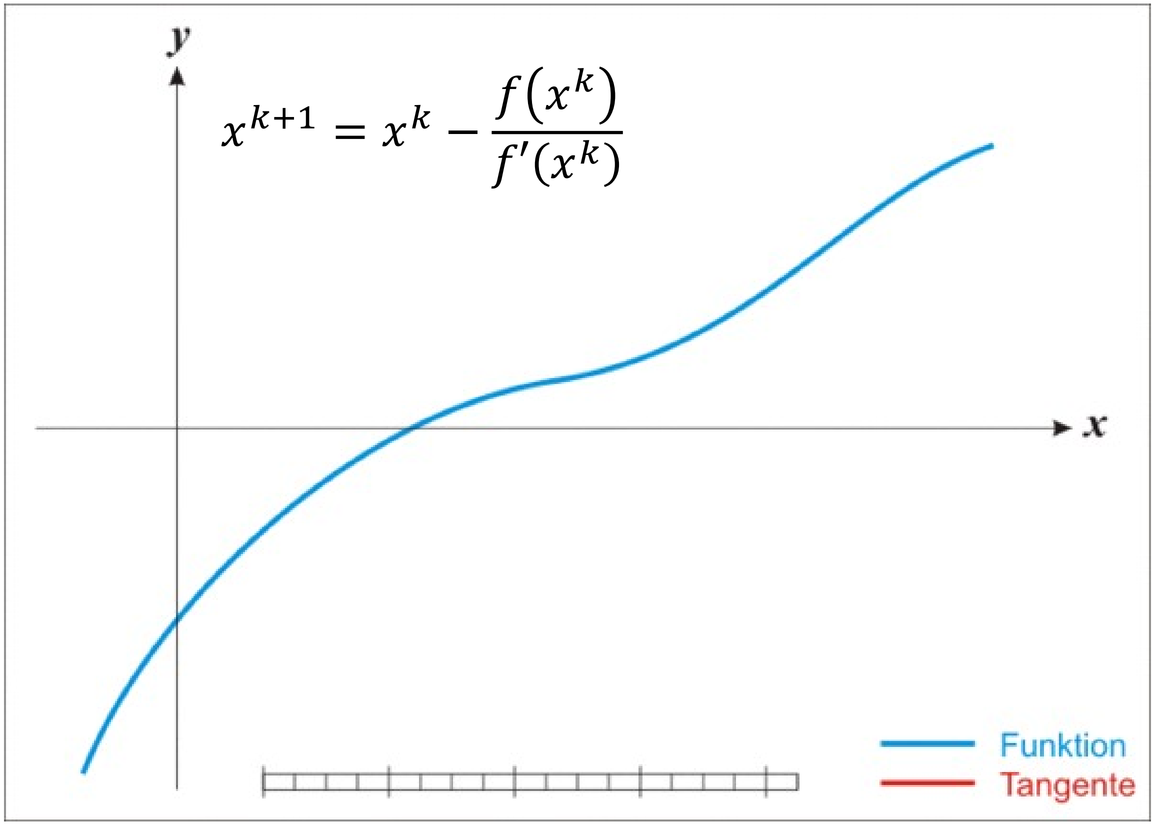
- Based on the first order derivatives (gradient) and in some cases the second order derivatives (Hessian).
  - **Newton's method** is based on both first and second derivatives
  - **Gradient descent** is based only on the first derivative

Descent methods

- Iterative solutions that attempt to "descend" the function in steps to arrive at the minimum

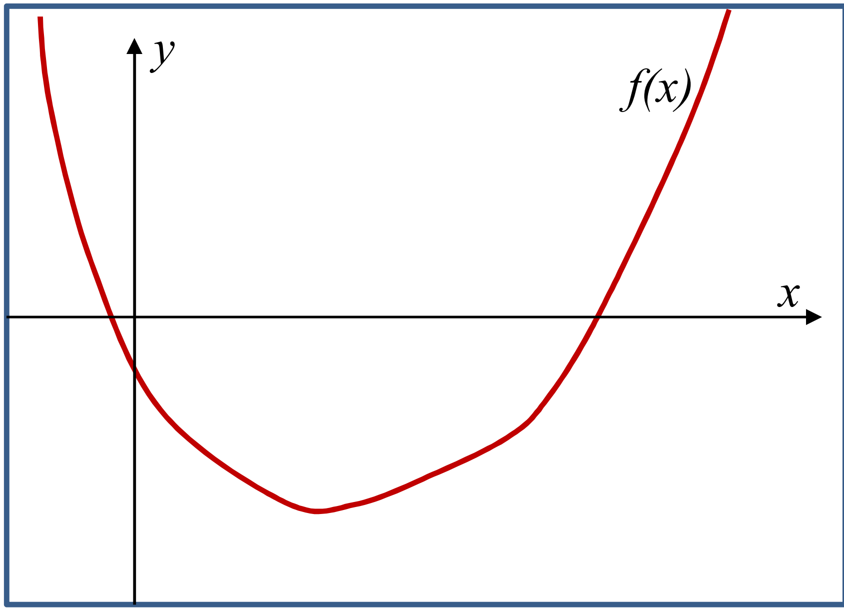
- Based on the first order derivatives (gradient) and in some cases the second order derivatives (Hessian).
  - **Newton's method** is based on both first and second derivatives
  - **Gradient descent** is based only on the first derivative

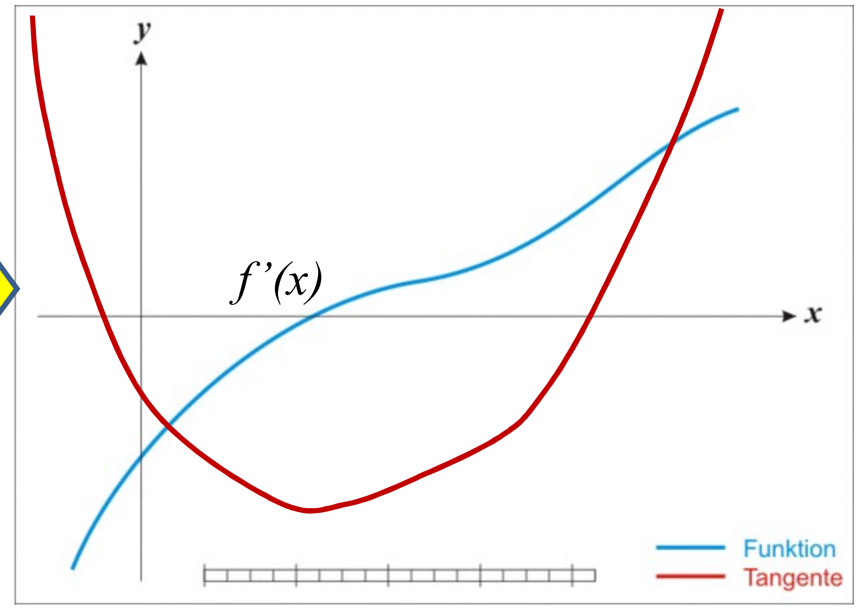
Newton's iterative method to find the zero of a function



- Newton's method to find the "zero" of a function
  - Initialize estimate
  - Approximate function by the tangent at initial value
  - Update estimate to location where tangent becomes 0
  - Iterate

Newton's Method to optimize a function





- Apply Newton's method to the ***derivative*** of the function!
  - The derivative goes to 0 at the optimum
- Algorithm:
  - Initialize x_0
  - K[th] iteration: Approximate $f'(x)$ by the tangent at x_k
  - Find the location $x_{\text{intersect}}$ where the tangent goes to 0. Set $x_{k+1} = x_{\text{intersect}}$
  - Iterate

Newton's method to minimize univariate functions

- Apply Newton's algorithm to find the zero of the derivative $f'(x)$

$$x^{k+1} = x^k - \frac{f'(x^k)}{f''(x^k)}$$

- k is the current iteration
- The iterations continue until we achieve the stopping criterion $|x^{k+1} - x^k| < \epsilon$

Newton's method for multivariate functions

1. Select an initial starting point X^0
2. Evaluate the gradient $\nabla f\left(X^k\right)$ and Hessian $\nabla^2 f\left(X^k\right)$ at X^k
3. Calculate the new X^{k+1} using the following

$$X^{k+1} = X^k - \left[\nabla^2 f(X^k)\right]^{-1} . \nabla f(X^k)$$

4. Repeat Steps 2 and 3 until convergence

Newton's Method example

- This is the same optimization problem we saw previously
- Minimize

$$f(x_1,x_2,x_3)=(x_1)^2+x_1(1-x_2)-(x_2)^2-x_2x_3+(x_3)^2+x_3$$

- Gradient

$$\nabla f=\begin{bmatrix} 2x_1+1-x_2 \\ -x_1+2x_2-x_3 \\ -x_2+2x_3+1 \end{bmatrix}$$

Newton's Method example

- Initial Value of $X^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

- The gradient for the vector X^0

$$\nabla f(0,0,0) = \begin{bmatrix} 0-0+1 \\ -0+0-0 \\ -0-0+1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

Unconstrained Minimization of function (Example)

- The Hessian matrix is

$$\nabla^2 f = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}$$

- The inverse of the Hessian is needed as well

$$\left[\nabla^2 f\right]^{-1} = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}^{-1} = \begin{bmatrix} \frac{3}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{3}{4} \end{bmatrix}$$

Newton's Method example

- The new vector x after iteration 1 is as follow

$$X^1 = X^0 - \left[\nabla^2 f(X^0)\right]^{-1} . \nabla f(X^0)$$

$$X^1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} \frac{3}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{3}{4} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$X^1 = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

Newton's Method example

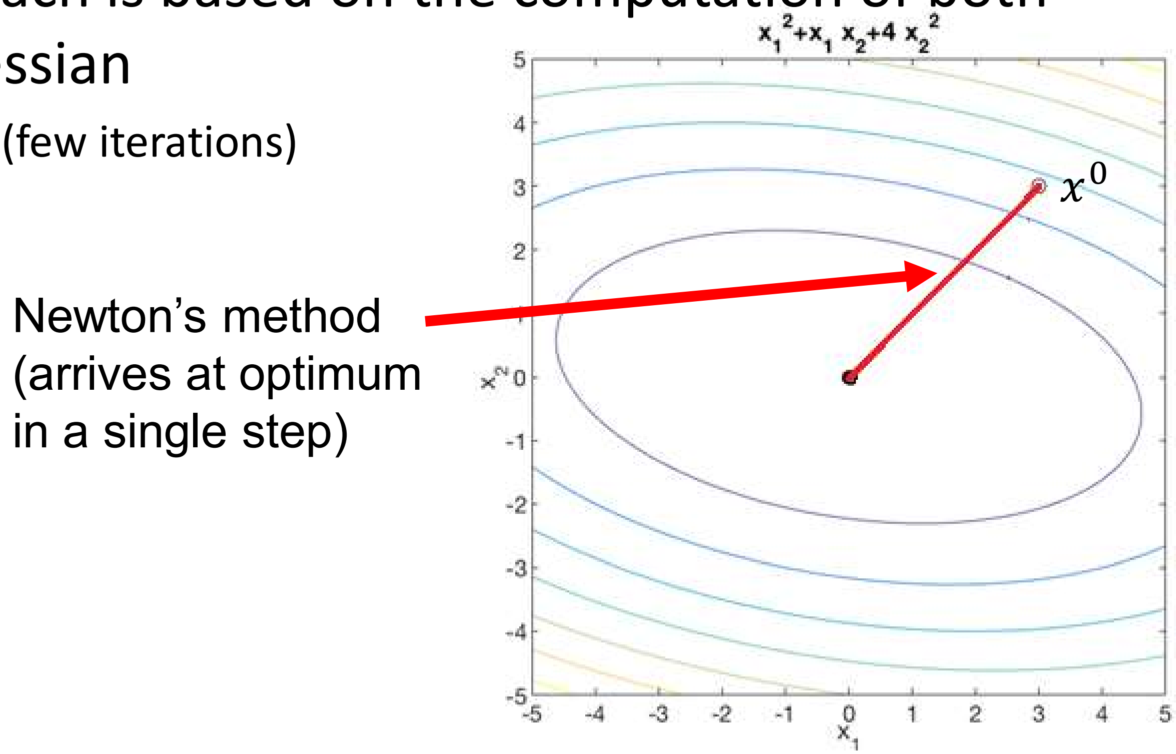
- The updated value of the gradient for $x^1 = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$

$$\nabla f(-1,-1,-1) = \begin{bmatrix} 2+1+1 \\ -1+2-1 \\ -1-2+1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

- The Gradient is zero => The Newton method has converged

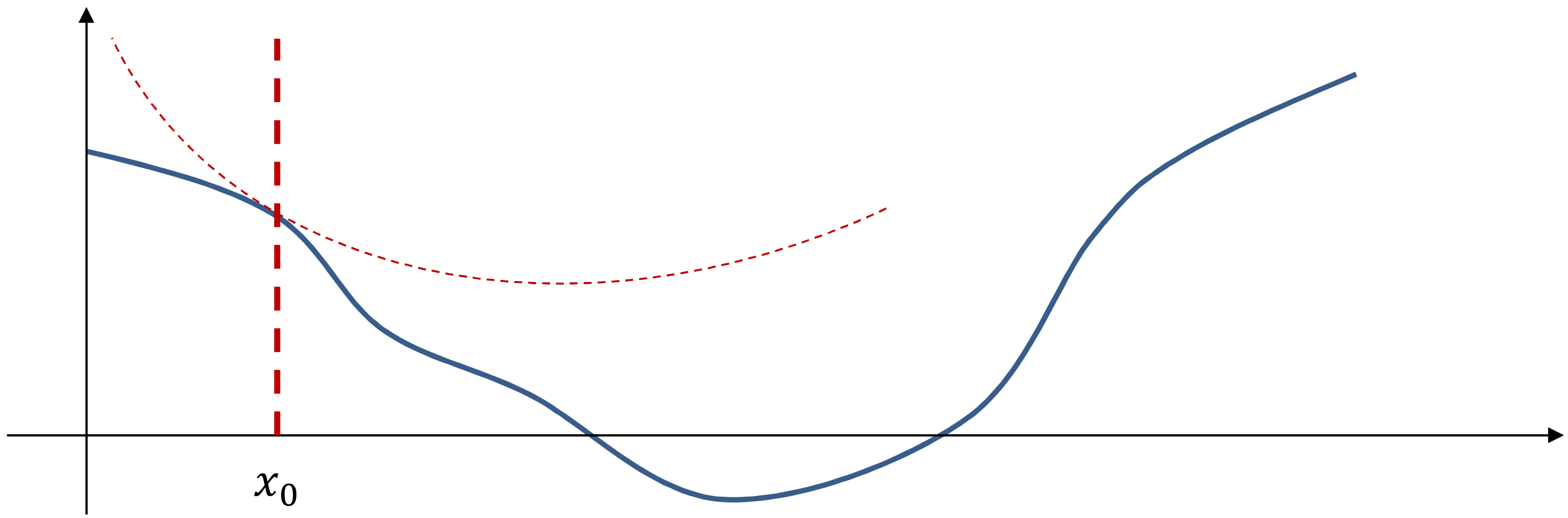
Newton’s Method

- Newton’s approach is based on the computation of both gradient and Hessian
  - Fast to converge (few iterations)
  - Slow to compute



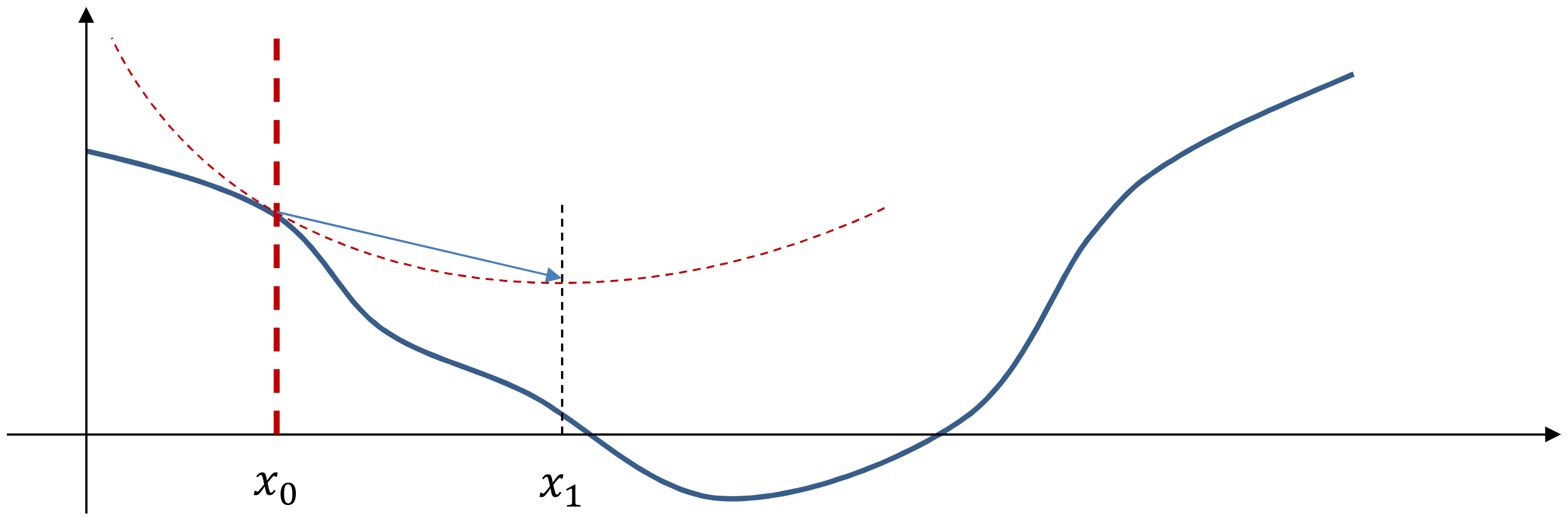
- Can arrive at the optimal solution in a *single* step for a quadratic function

Newton's method: generic case



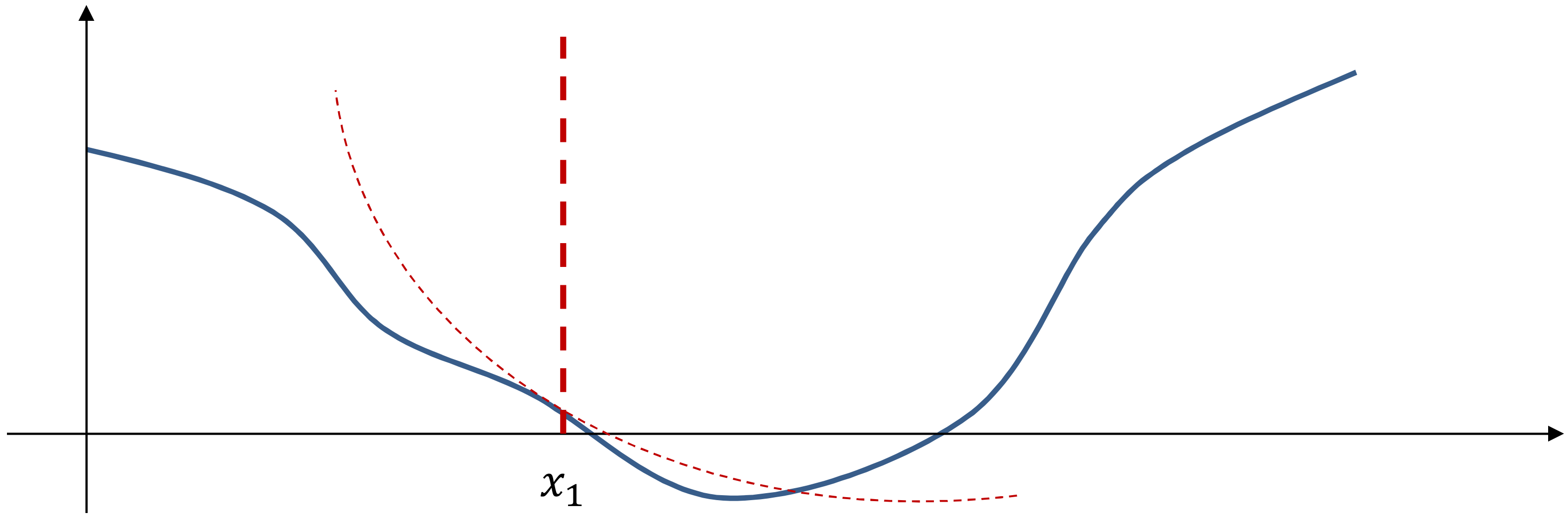
- Approximates function by a quadratic Taylor series at the current estimate
- Solves for the optimum of the quadratic approximation
  - Single step
- Repeat

Newton's method: generic case



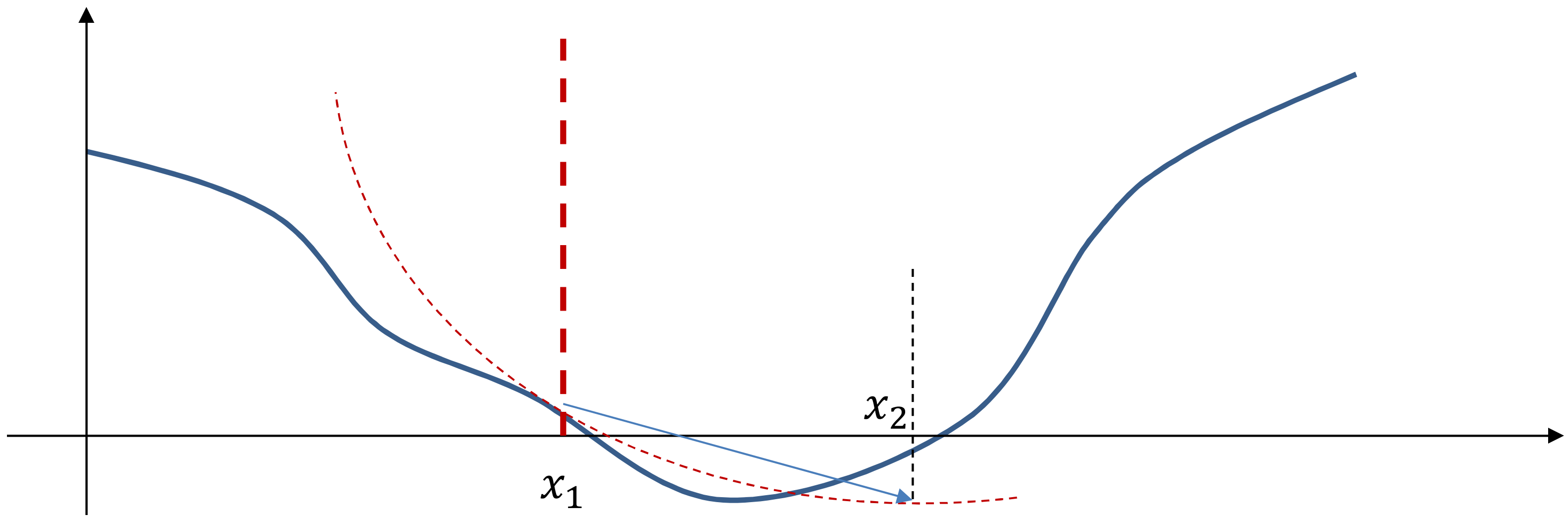
- Approximates function by a quadratic Taylor series at the current estimate
- Solves for the optimum of the quadratic approximation
  - Single step
- Repeat

Newton's method: generic case



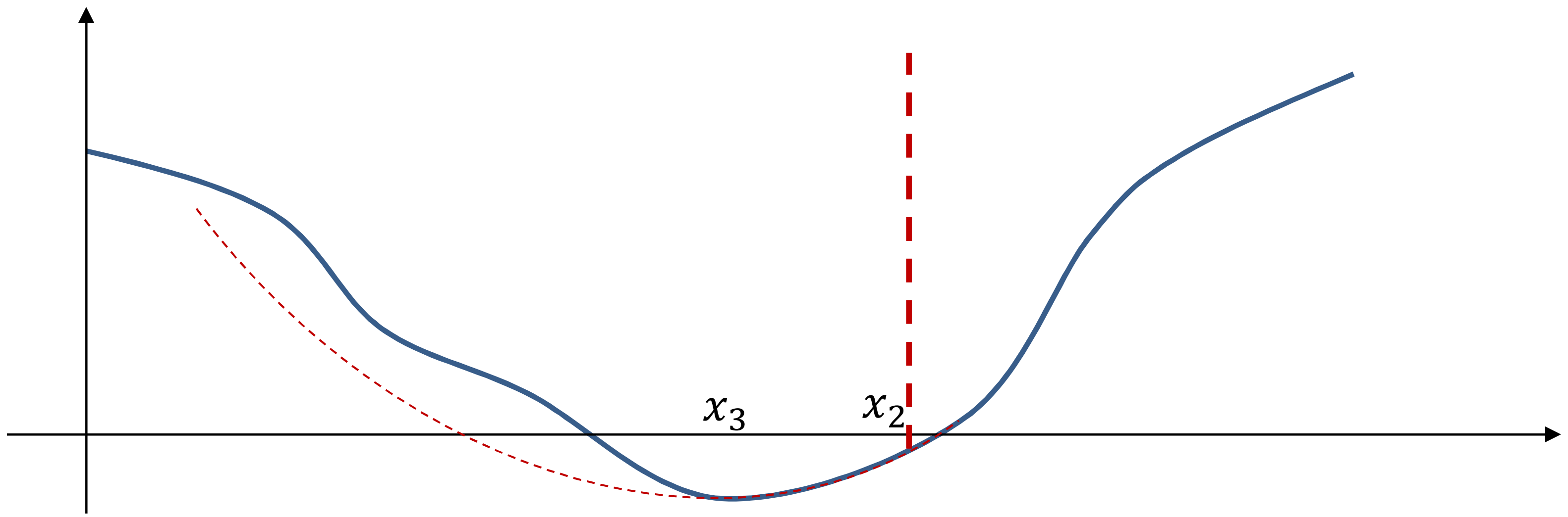
- Approximates function by a quadratic Taylor series at the current estimate
- Solves for the optimum of the quadratic approximation
  - Single step
- Repeat

Newton's method: generic case



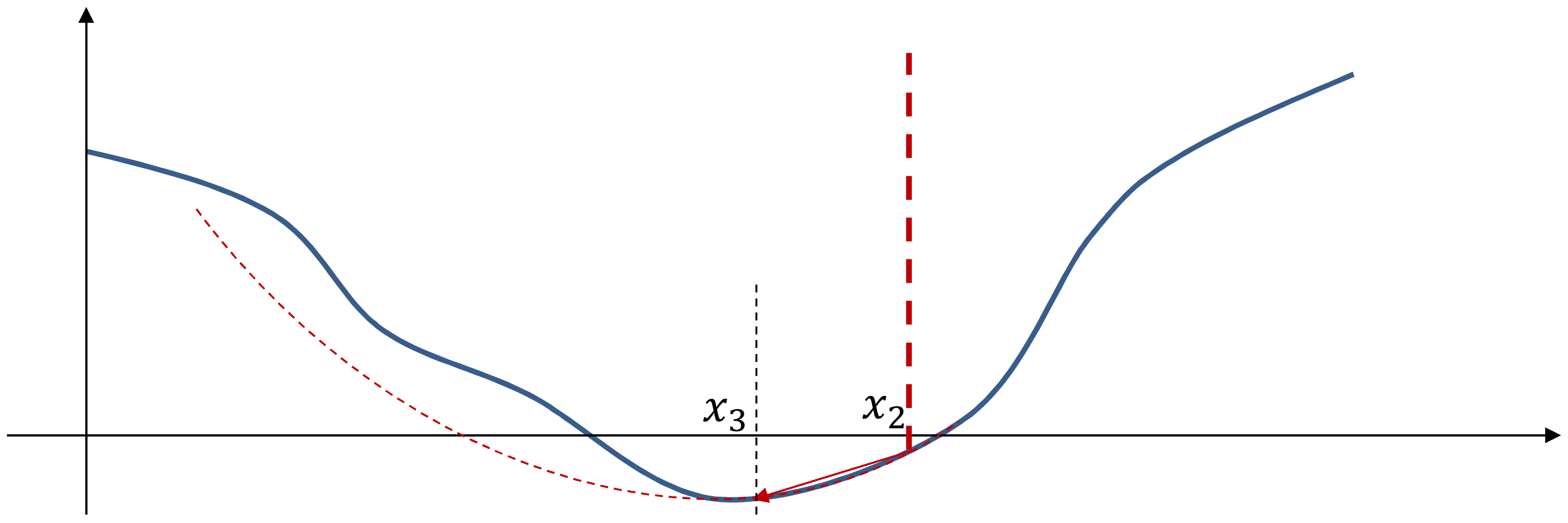
- Approximates function by a quadratic Taylor series at the current estimate
- Solves for the optimum of the quadratic approximation
  - Single step
- Repeat

Newton's method: generic case



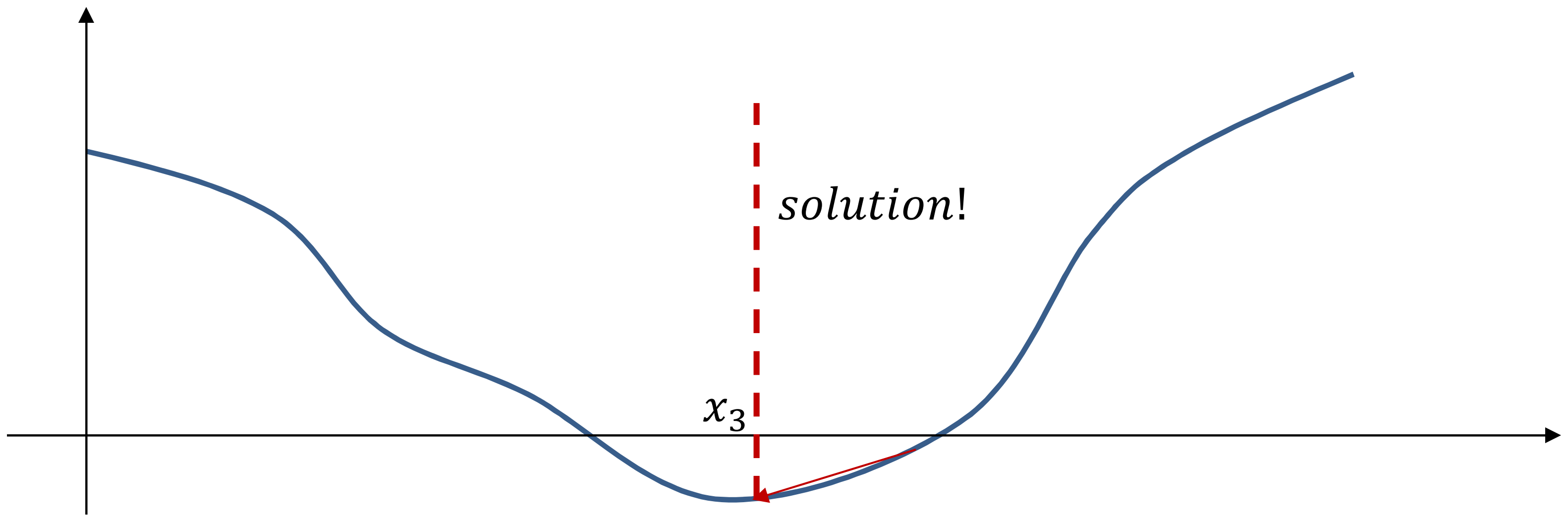
- Approximates function by a quadratic Taylor series at the current estimate
- Solves for the optimum of the quadratic approximation
  - Single step
- Repeat

Newton's method: generic case



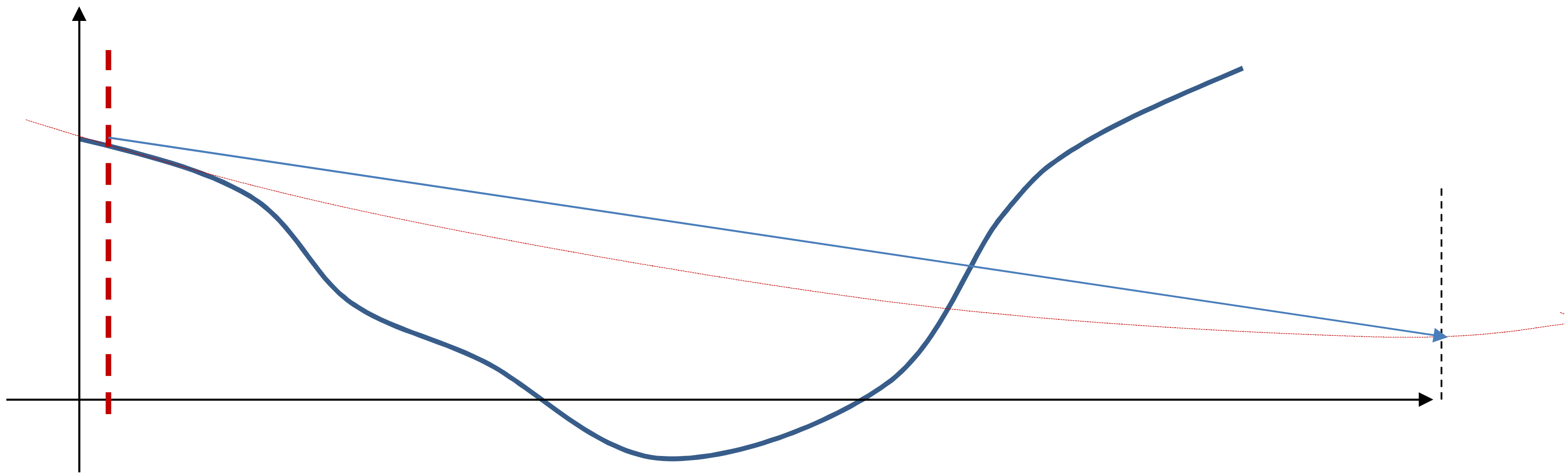
- Approximates function by a quadratic Taylor series at the current estimate
- Solves for the optimum of the quadratic approximation
  - Single step
- Repeat

Newton's method: generic case



- Approximates function by a quadratic Taylor series at the current estimate
- Solves for the optimum of the quadratic approximation
  - Single step
- Repeat

Newton's method: generic case

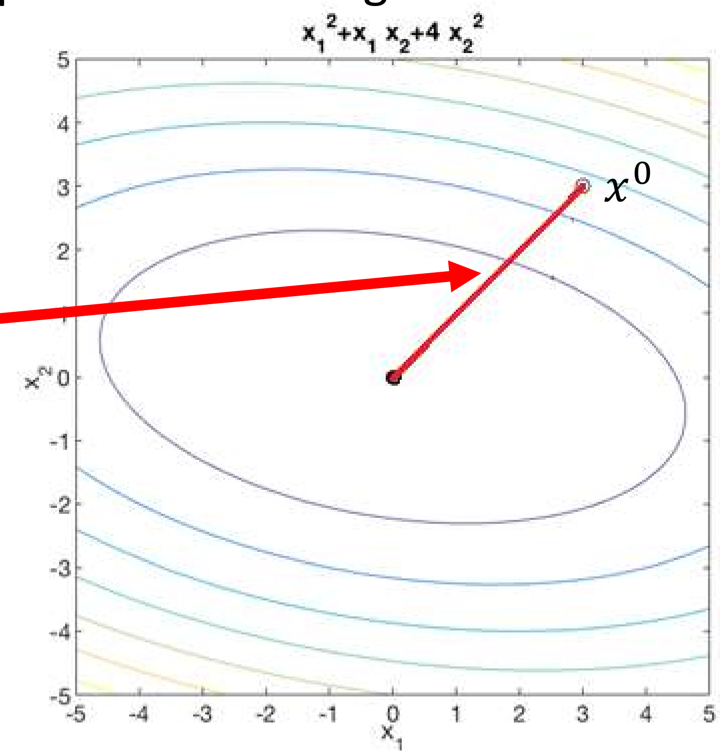


- Approximates function by a quadratic Taylor series at the current estimate
- Solves for the optimum of the quadratic approximation
  - Single step
- Repeat
  - Can easily get lost if the initial point is poor

Newton's Method

- Newton's approach is based on the computation of both gradient and Hessian
  - Fast to converge (few iterations)
  - Slow to compute

Newton's method (arrives at optimum in a single step)



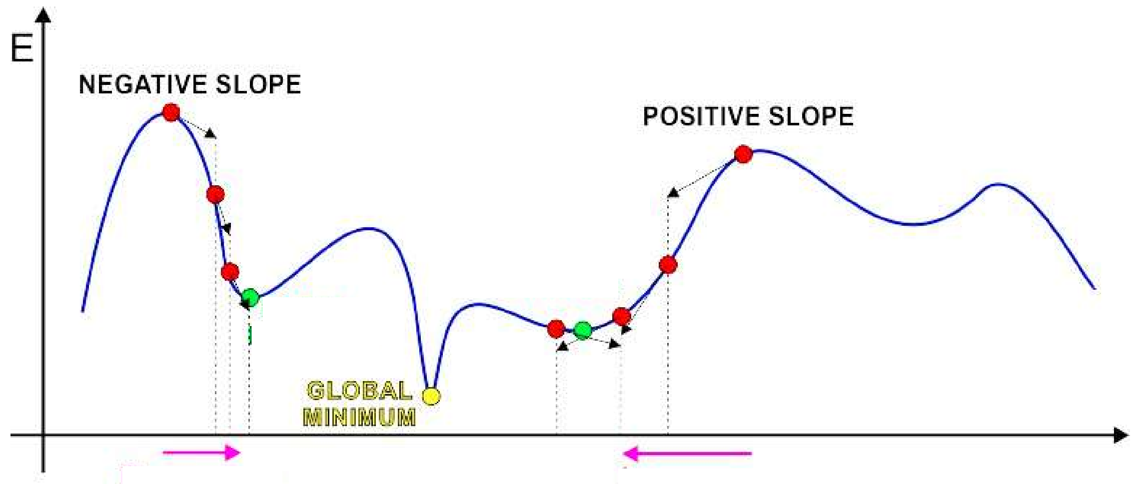
- Can be very efficient
- This method is very sensitive to the initial point
  - If the initial point is very far from the optimal point, the optimization process may not converge

Descent methods

- Iterative solutions that attempt to “descend” the function in steps to arrive at the minimum

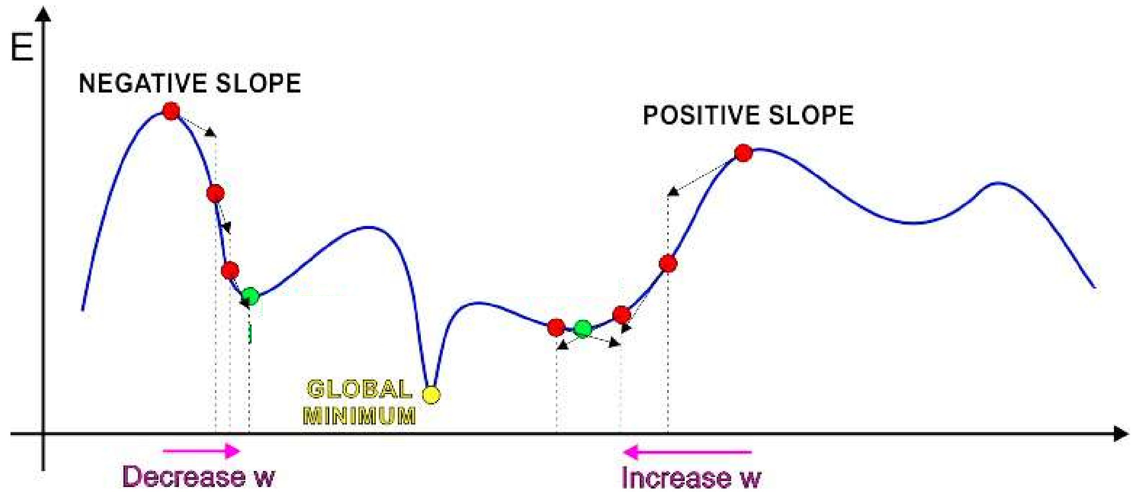
- Based on the first order derivatives (gradient) and in some cases the second order derivatives (Hessian).
  - **Newton’s method** is based on both first and second derivatives
  - **Gradient descent** is based only on the first derivative

The Approach of Gradient Descent



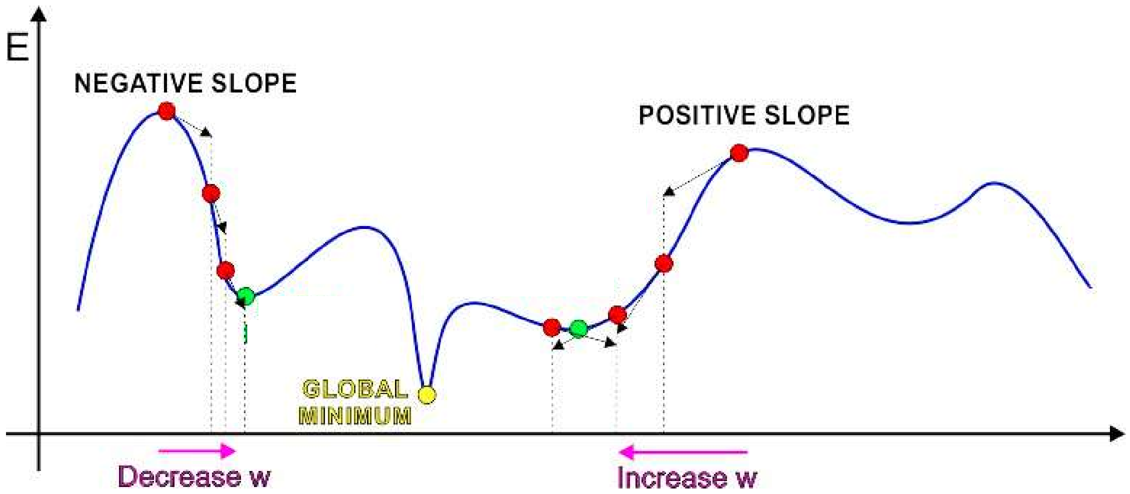
- Iterative solution:
  - Start at some point
  - Find direction in which to shift this point to decrease error
    - This can be found from the derivative of the function
      - A positive derivative → moving left decreases error
      - A negative derivative → moving right decreases error
  - Shift point in this direction

The Approach of Gradient Descent



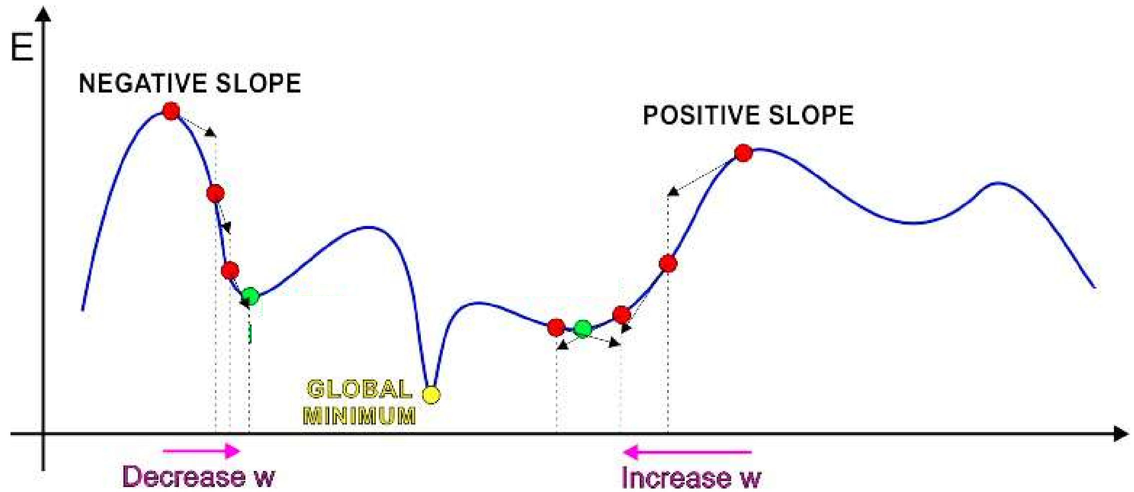
- Iterative solution: Trivial algorithm
  - Initialize x^0
  - While $f'\left(x^k\right) \neq 0$
    - If $sign\left(f'\left(x^k\right)\right)$ is positive:
      - $x^{k+1} = x^k - step$
    - Else
      - $x^{k+1} = x^k + step$
  - But what must step be to ensure we actually get to the optimum?

The Approach of Gradient Descent



- Iterative solution: Trivial algorithm
  - Initialize x^0
  - While $f'\left(x^k\right) \neq 0$
    - $x^{k+1} = x^k - sign\left(f'\left(x^k\right)\right).step$
  - Identical to previous algorithm

The Approach of Gradient Descent



- Iterative solution: Trivial algorithm
  - Initialize x_0
  - While $f'(x^k) \neq 0$
    - $x^{k+1} = x^k - \eta^k f'(x^k)$
  - η^k is the "step size"
    - What must the step size be?

Gradient descent/ascent (multivariate)

- The gradient descent/ascent method to find the minimum or maximum of a function f iteratively
  - To find a *maximum* move *in the direction of the gradient*

$$x^{k+1} = x^k + \eta^k \nabla f\left(x^k\right)$$

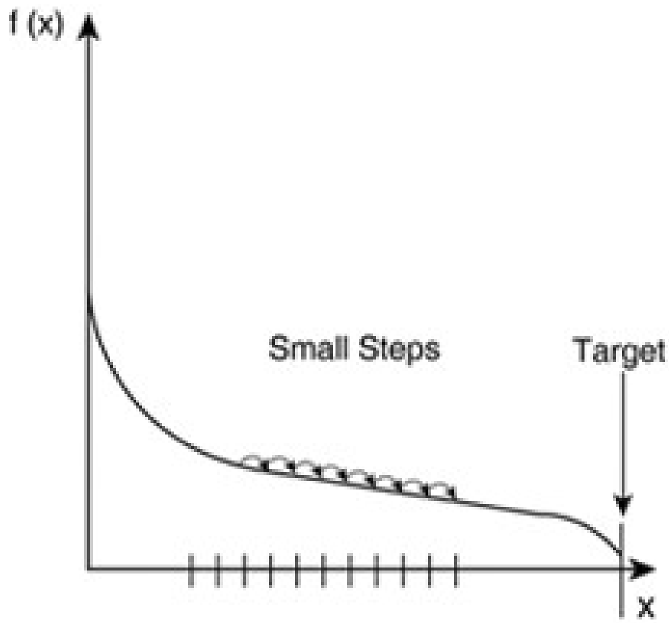
  - To find a *minimum* move *exactly opposite the direction of the gradient*

$$x^{k+1} = x^k - \eta^k \nabla f\left(x^k\right)$$

- What is the step size η^k

1. Fixed step size

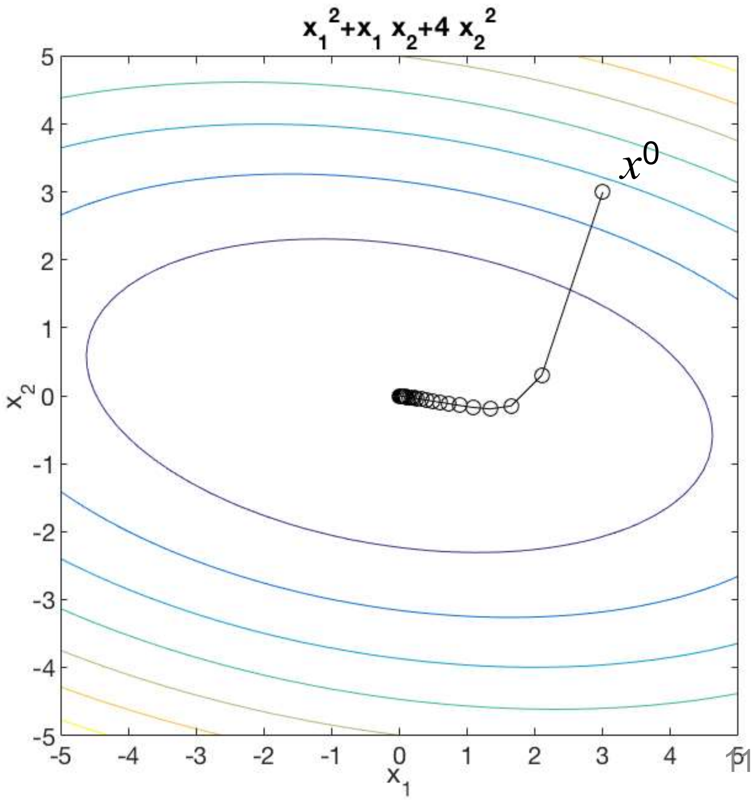
- Fixed step size
  - Use fixed value for η^k



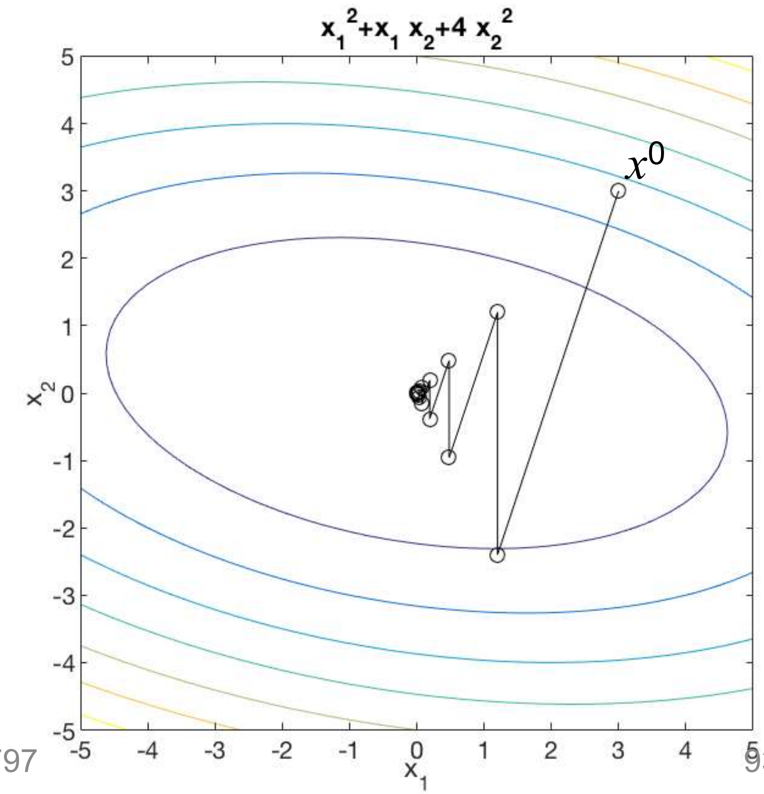
Influence of step size example (constant step size)

$$f(x_1, x_2) = (x_1)^2 + x_1 x_2 + 4(x_2)^2 \qquad x^{initial} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$\eta = 0.1$



$\eta = 0.2$



Variable step size

- Shrink step size by a constant factor each iteration:
$$\eta^k = \alpha\eta^{k-1}$$
  - Where $\alpha < 1$
- Gradient descent algorithm:
  - Initialize x^0, η^0
  - While $f'(x^k) \neq 0$
    - $x^{k+1} = x^k - \eta^k f'(x^k)$
    - $\eta^{k+1} = \alpha\eta^k$
    - $k = k + 1$

Optimal step size

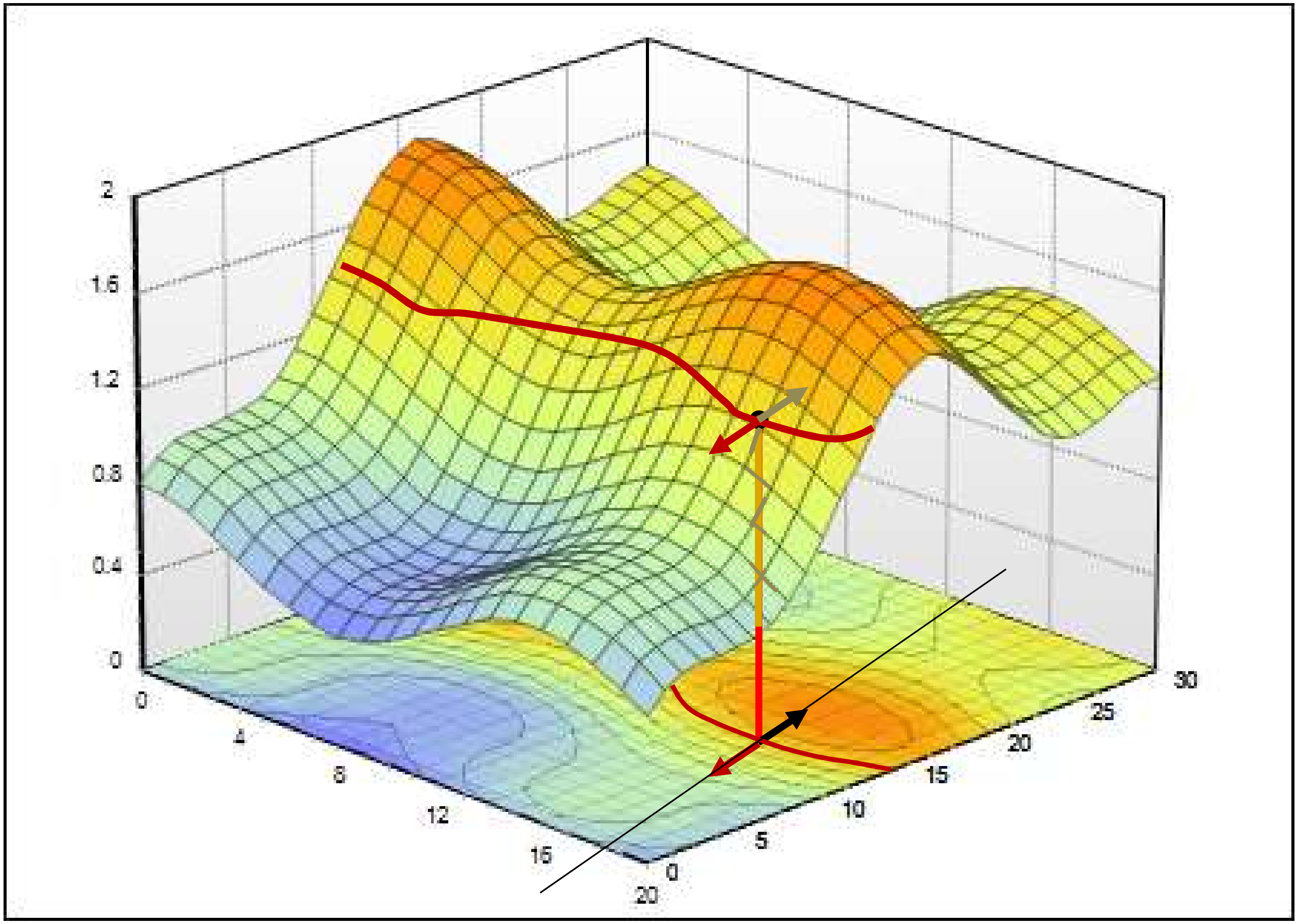
- Finding the optimal step size is a challenge
- Ideally, step size changes with iteration
- Several algorithms to find optimal step size
  - On slides
    - Please read the slides, this will appear in the quiz

2. Backtracking line search for step size

- Two parameters α (typically 0.5) and β (typically 0.8)
- At each iteration, estimate step size as follows:
  - Set η^k = 1
  - Update η^k = $\beta\eta^k$ until
    $f\left(x^k - \eta^k \nabla f\left(x^k\right)\right) \leq f\left(x^k\right) - \alpha\eta^k \left\|\nabla f\left(x^k\right)\right\|^2$
  - Update $x^{k+1} = x^k - \eta^k \nabla f\left(x^k\right)$

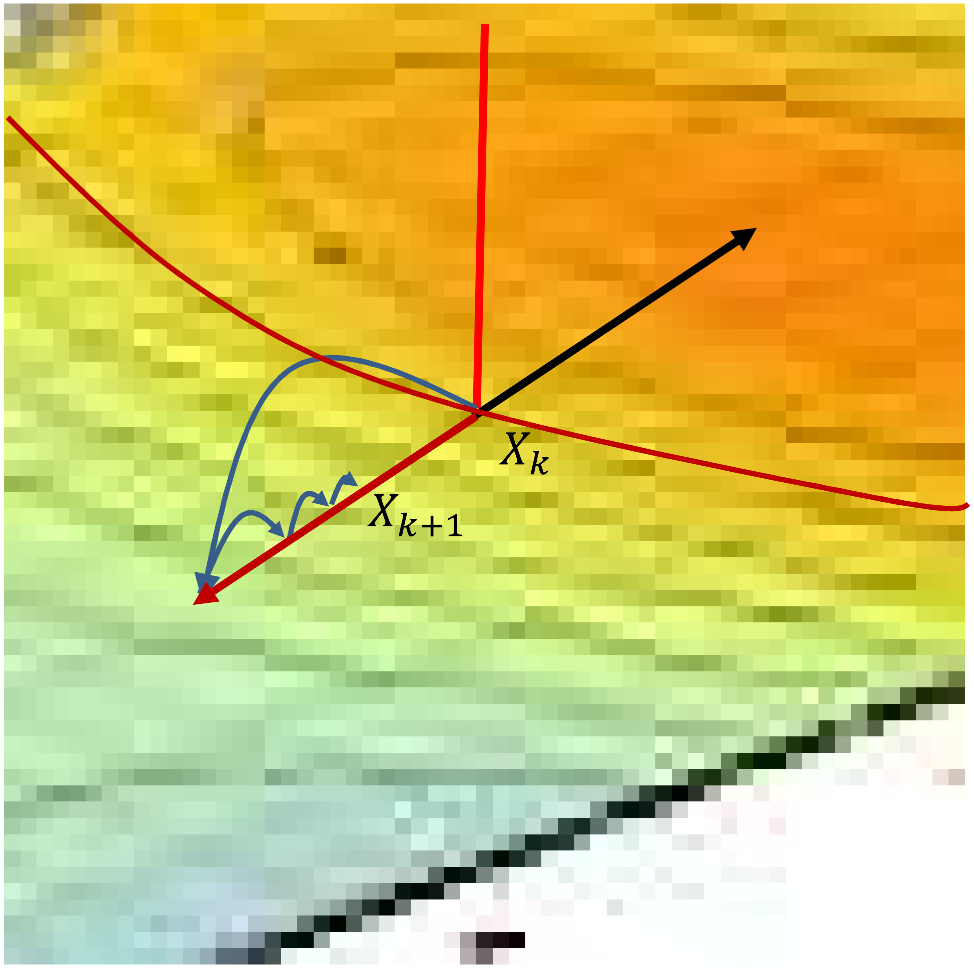
- Intuitively: At each iteration
  - Take a unit step size and keep shrinking it until we arrive at a place where the function $f\left(x^k - \eta^k \nabla f\left(x^k\right)\right)$ actually decreases sufficiently w.r.t $f\left(x^k\right)$

2. Backtracking line search for step size



- Keep shrinking step size till we find a good one

2. Backtracking line search for step size



- Keep shrinking step size till we find a good one
- Update estimate to the position at the converged step size

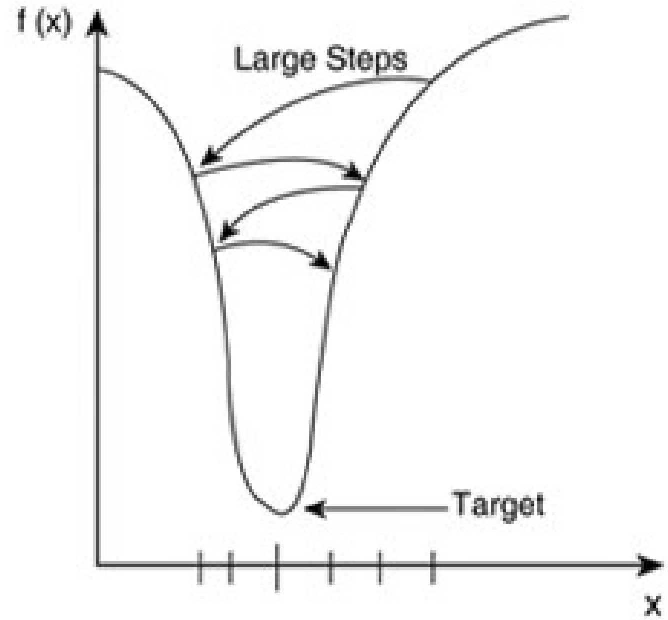
2. Backtracking line search for step size

- At each iteration, estimate step size as follows:
  - Set $\eta^k = 1$
  - Update $\eta^k = \beta\eta^k$ until

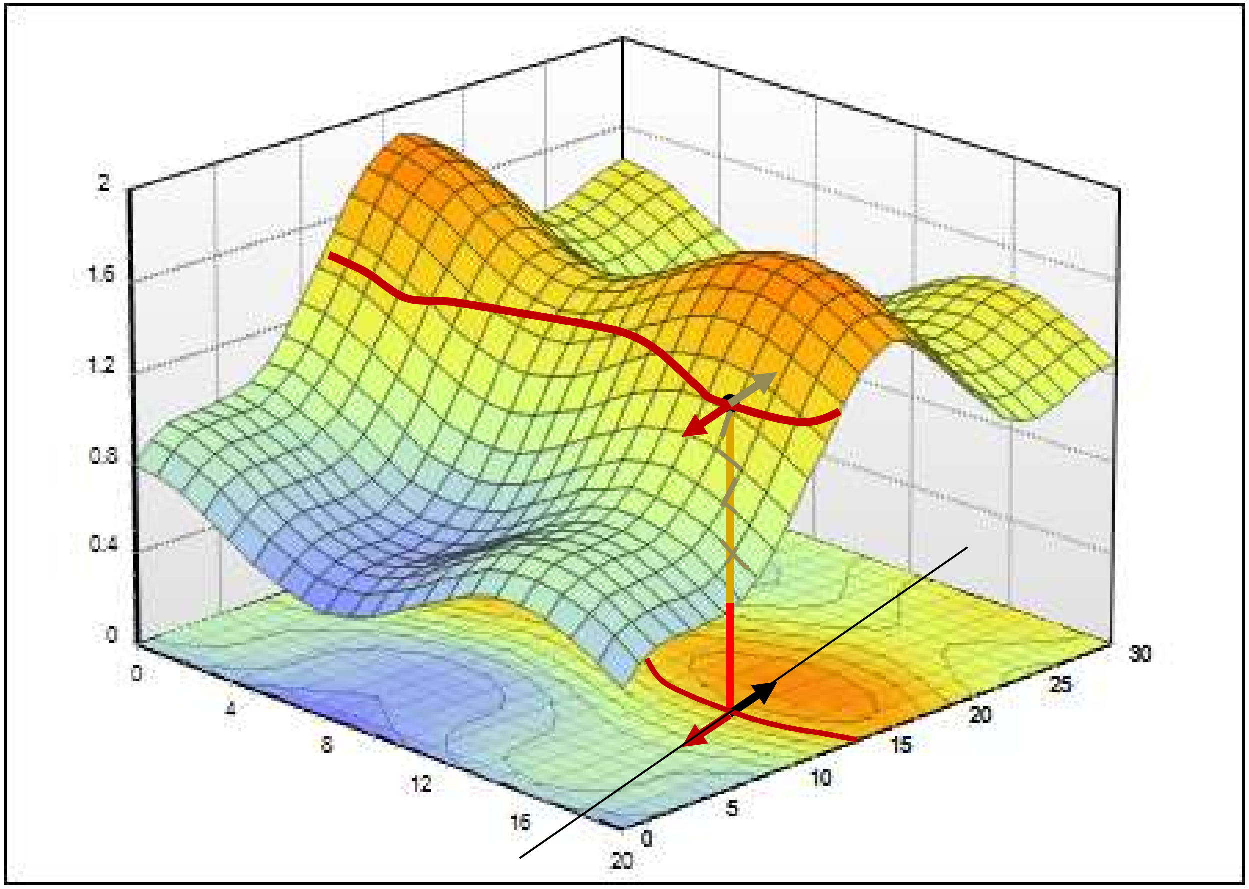
$$f\left(x^k - \eta^k \nabla f(x^k)\right) \leq f(x^k) - \alpha\eta^k \|\nabla f(x^k)\|^2$$

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

- Figure shows actual evolution of x^k

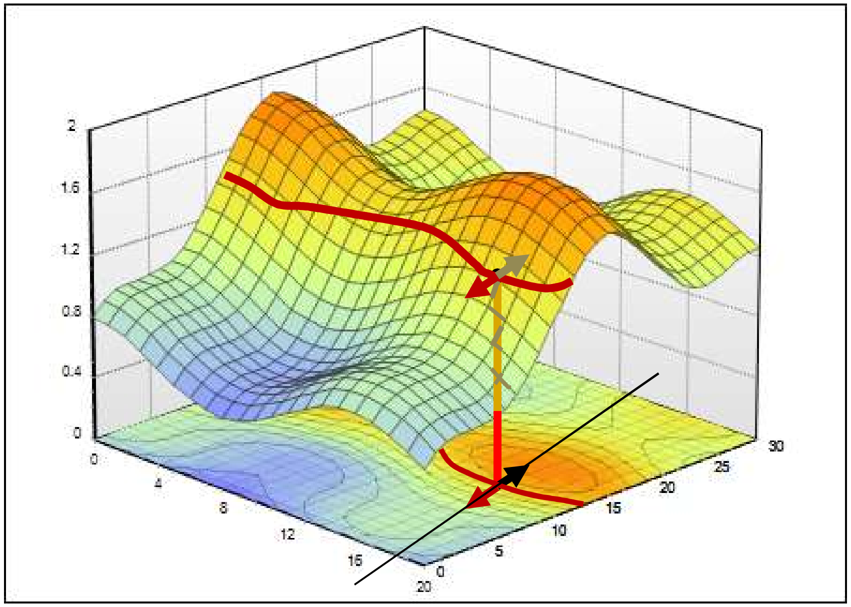


3. Full line search for step size



- At each iteration scan for η_k that minimizes $f\left(x^k - \eta^k \nabla f\left(x^k\right)\right)$
- Update $x^k = x^k - \eta^k \nabla f\left(x^k\right)$

3. Full line search for step size



- At each iteration scan for η_k that minimizes $f\left(x^k - \eta^k \nabla f\left(x^k\right)\right)$
- Can be computed by solving

$$\frac{df\left(x^k - \eta^k \nabla f\left(x^k\right)\right)}{d\eta^k} = 0$$

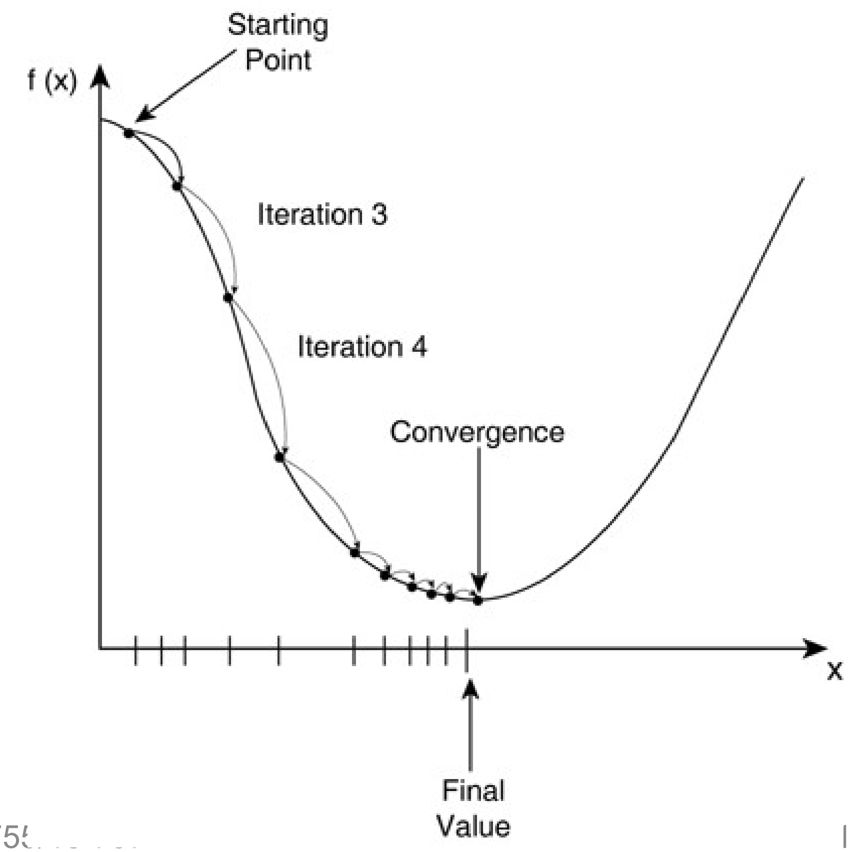
- Update $x^k = x^k - \eta^k \nabla f\left(x^k\right)$

Gradient descent convergence criteria

- The gradient descent algorithm converges when one of the following criteria is satisfied

$$\left| f(x^{k+1}) - f(x^k) \right| < \varepsilon_1$$

- Or $\left\| \nabla f(x^k) \right\| < \varepsilon_2$



Gradient descent example

- This is the same optimization problem as previously
- Minimize

$$f(x_1, x_2, x_3) = (x_1)^2 + x_1(1 - x_2) - (x_2)^2 - x_2x_3 + (x_3)^2 + x_3$$

- Gradient initial vector

$$\nabla f = \begin{bmatrix} 2x_1 + 1 - x_2 \\ -x_1 + 2x_2 - x_3 \\ -x_2 + 2x_3 + 1 \end{bmatrix} \qquad x^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Gradient descent example

$$\nabla f(x^0) = \begin{bmatrix} 2\cdot 0 + 1 - 0 \\ -0 + 2\cdot 0 - 0 \\ -0 + 2\cdot 0 + 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$x^1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} - \alpha^0 \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -\alpha^0 \\ 0 \\ -\alpha^0 \end{bmatrix}$$

- Find the best step value α^0

Gradient descent example

$$f(x^1) = (-\alpha^0)^2 - \alpha^0 + (-\alpha^0)^2 - \alpha^0$$
$$= 2(\alpha^0)^2 - 2(\alpha^0)$$

$$\frac{\partial f(x^1)}{\partial \alpha^0} = 4(\alpha^0) - 2$$

- Set the derivative equal to zero

$$\frac{\partial f(x^1)}{\partial \alpha^0} = 4(\alpha^0) - 2 = 0 \Rightarrow \alpha^0 = \frac{1}{2} \qquad x^1 = \begin{bmatrix} -\alpha^0 \\ 0 \\ -\alpha^0 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} \\ 0 \\ -\frac{1}{2} \end{bmatrix}$$

Gradient descent example

- Iteration 2

$$\nabla f(-\frac{1}{2}, 0, -\frac{1}{2}) = \begin{bmatrix} -1+1+0 \\ \frac{1}{2}+0+\frac{1}{2} \\ 0-1+1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$x^2 = \begin{bmatrix} -\frac{1}{2} \\ 0 \\ -\frac{1}{2} \end{bmatrix} - \alpha^1 \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} \\ -\alpha^1 \\ -\frac{1}{2} \end{bmatrix}$$

Gradient descent example

$$f(x^2) = \frac{1}{4} - \frac{1}{2}(1+\alpha^1) + (\alpha^1)^2 - \frac{1}{2}\alpha^1 + \frac{1}{4} - \frac{1}{2}$$

$$= (\alpha^1)^2 - \alpha^1 - \frac{1}{2}$$

$$\frac{\partial f(x^2)}{\partial \alpha^1} = 2(\alpha^1) - 1$$

- Set the derivative equal to zero

$$\frac{\partial f(x^2)}{\partial \alpha^1} = 2(\alpha^1) - 1 = 0 \Rightarrow \alpha^1 = \frac{1}{2}$$

$$x^2 = \begin{bmatrix} -\frac{1}{2} \\ -\alpha^1 \\ -\frac{1}{2} \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} \\ -\frac{1}{2} \\ -\frac{1}{2} \end{bmatrix}$$

Gradient descent example

- Iteration 3

$$\nabla f\left(-\frac{1}{2},-\frac{1}{2},-\frac{1}{2}\right)=\begin{bmatrix} -1+1+\frac{1}{2} \\ \frac{1}{2}-1+\frac{1}{2} \\ \frac{1}{2}-1+1 \end{bmatrix}=\begin{bmatrix} \frac{1}{2} \\ 0 \\ \frac{1}{2} \end{bmatrix}$$

$$x^3=\begin{bmatrix} -\frac{1}{2} \\ -\frac{1}{2} \\ -\frac{1}{2} \end{bmatrix}-\alpha^2\begin{bmatrix} \frac{1}{2} \\ 0 \\ \frac{1}{2} \end{bmatrix}=\begin{bmatrix} -\frac{1}{2}(\alpha^2+1) \\ -\frac{1}{2} \\ -\frac{1}{2}(\alpha^2+1) \end{bmatrix}$$

Gradient descent example

$$f(x^3) = \frac{1}{2}(\alpha^2 + 1)^2 - \frac{3}{2}(\alpha^2 + 1) + \frac{1}{4}$$

$$\frac{\partial f(x^3)}{\partial \alpha^2} = (\alpha^2 + 1) - \frac{3}{2}$$

- Set the derivative equal to zero

$$\frac{\partial f(x^3)}{\partial \alpha^2} = (\alpha^2 + 1) - \frac{3}{2} = 0 \Rightarrow \alpha^2 = \frac{1}{2} \qquad x^3 = \begin{bmatrix} -\frac{1}{2}(\alpha^2 + 1) \\ -\frac{1}{2} \\ -\frac{1}{2}(\alpha^2 + 1) \end{bmatrix} = \begin{bmatrix} -\frac{3}{4} \\ -\frac{1}{2} \\ -\frac{3}{4} \end{bmatrix}$$

Gradient descent example

- Iteration 4

$$\nabla f\left(-\frac{3}{4}, -\frac{1}{2}, -\frac{3}{4}\right) = \begin{bmatrix} 0 \\ \frac{1}{2} \\ 0 \end{bmatrix}$$

$$x^4 = \begin{bmatrix} -\frac{3}{4} \\ -\frac{1}{2} \\ -\frac{3}{4} \end{bmatrix} - \alpha^3 \begin{bmatrix} 0 \\ \frac{1}{2} \\ 0 \end{bmatrix} = \begin{bmatrix} -\frac{3}{4} \\ -\frac{1}{2}(\alpha^3 + 1) \\ -\frac{3}{4} \end{bmatrix}$$

Gradient descent example

$$f(x^4) = \frac{1}{4}(\alpha^3 + 1)^2 - \frac{3}{2}(\alpha^3) - \frac{3}{2}$$

$$\frac{\partial f(x^4)}{\partial \alpha^3} = \frac{1}{2}(\alpha^3 + 1) - \frac{9}{8}$$

- Set the derivative equal to zero

$$\frac{\partial f(x^4)}{\partial \alpha^3} = \frac{1}{2}(\alpha^3 + 1) - \frac{9}{8} = 0 \Rightarrow \alpha^3 = \frac{5}{4}$$

$$x^4 = \begin{bmatrix} -\frac{3}{4} \\ -\frac{1}{2}(\alpha^3 + 1) \\ -\frac{3}{4} \end{bmatrix} = \begin{bmatrix} -\frac{3}{4} \\ -\frac{9}{8} \\ -\frac{3}{4} \end{bmatrix}$$

Gradient descent example

- Iteration 5

$$\nabla f\left(-\frac{3}{4},-\frac{9}{8},-\frac{3}{4}\right)=\begin{bmatrix}\frac{5}{8}\\-\frac{3}{4}\\\frac{5}{8}\end{bmatrix}$$

$$x^4=\begin{bmatrix}-\frac{3}{4}\\-\frac{9}{8}\\-\frac{3}{4}\end{bmatrix}-\alpha^4\begin{bmatrix}\frac{5}{8}\\-\frac{3}{4}\\\frac{5}{8}\end{bmatrix}=\begin{bmatrix}-\frac{1}{4}(3+\frac{5}{2}\alpha^4)\\-\frac{3}{4}(\frac{3}{2}-\alpha^4)\\-\frac{1}{4}(3+\frac{5}{3}\alpha^4)\end{bmatrix}$$

Gradient descent example

$$f(x^5) = \frac{73}{32}(\alpha^4)^2 - \frac{43}{32}(\alpha^4) - \frac{51}{64}$$

$$\frac{\partial f(x^5)}{\partial \alpha^4} = \frac{73}{16}\alpha^4 - \frac{43}{32}$$

- Set the derivative equal to zero

$$\frac{\partial f(x^5)}{\partial \alpha^4} = \frac{73}{16}\alpha^4 - \frac{43}{32} = 0 \Rightarrow \alpha^4 = \frac{43}{146}$$

$$x^5 = \begin{bmatrix} -\frac{1091}{1168} \\ -\frac{66}{73} \\ -\frac{1091}{1168} \end{bmatrix}$$

Gradient descent example

- Verifying the stopping criteria $\left\|\nabla f(x^5)\right\|$

$$\nabla f(x^5) = \begin{bmatrix} \frac{21}{584} \\ \frac{35}{584} \\ \frac{21}{584} \end{bmatrix}$$

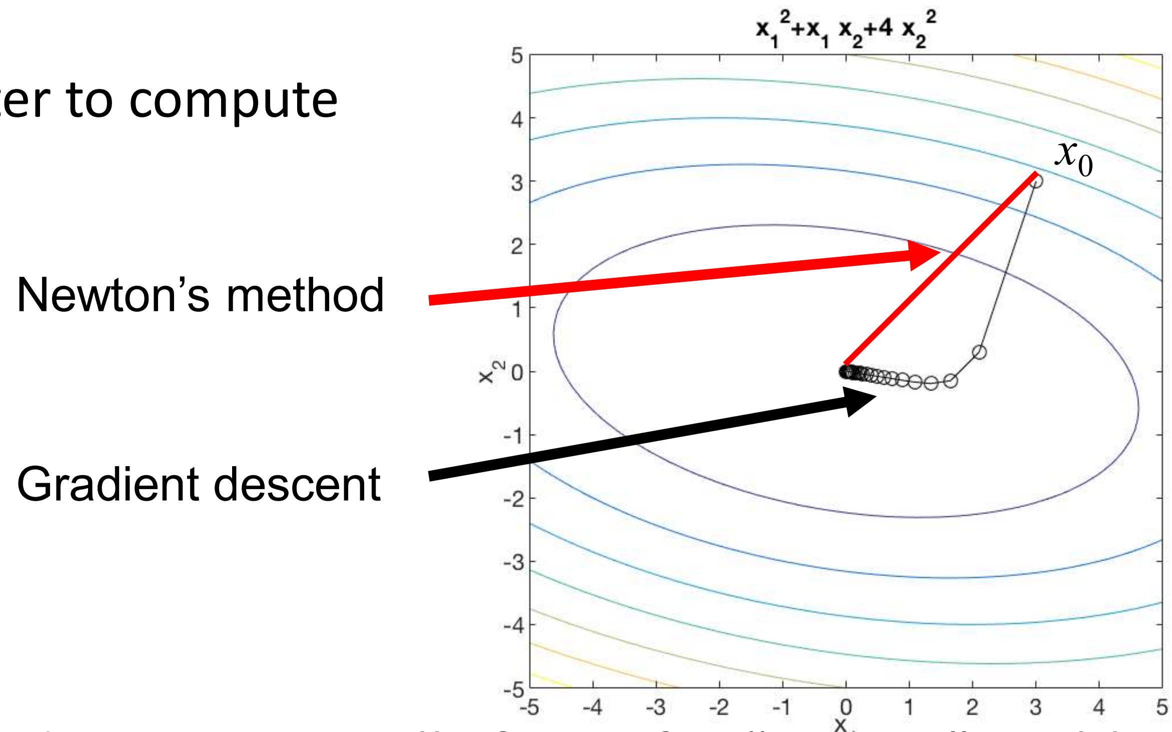
$$\left\|\nabla f(x^5)\right\| = \sqrt{\left(\frac{21}{584}\right)^2 + \left(\frac{35}{584}\right)^2 + \left(\frac{21}{584}\right)^2} = 0.0786$$

Gradient descent example

- $\left\|\nabla f(x^5)\right\| = 0.0786$ is very small. The stopping criteria is satisfied.
- The vector $x^5 = \begin{bmatrix} -\frac{1091}{1168} \\ -\frac{66}{73} \\ -\frac{1091}{1168} \end{bmatrix}$ can be taken as the minimum
- The vector x^5 is very close to the optimal minimum $x^{optimal} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$

Gradient descent vs. Newton's

- Gradient descent is typically much slower to converge than Newton's
  - But *much* faster to compute



- Newton's method is exponentially faster for "convex" problems
  - Although derivatives and Hessians may be hard to derive
  - May not converge for non-convex problems

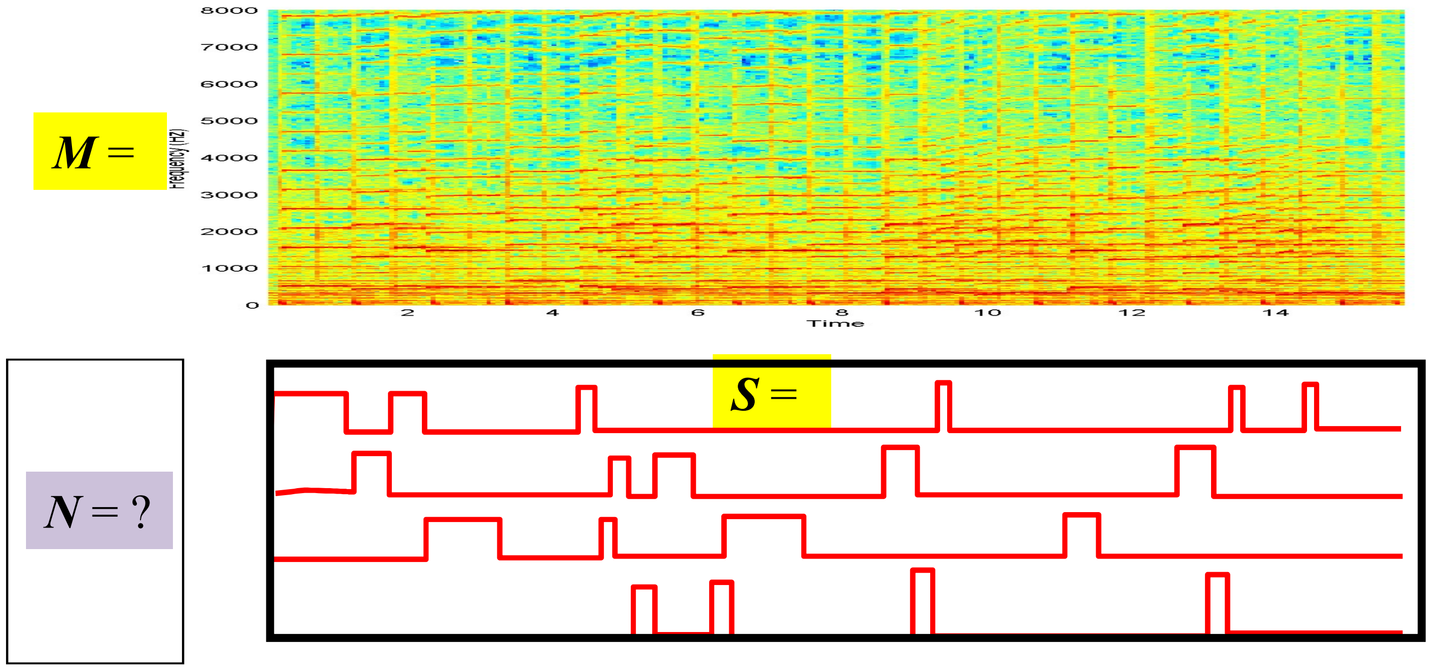
Index

1. The problem of optimization
2. Direct optimization
3. Descent methods
   – Newton's method
   – Gradient methods
4. Online optimization
5. Constrained optimization
   – Lagrange's method
   – Projected gradients
6. Regularization
7. Convex optimization and Lagrangian duals

Online Optimization

- Often our objective function is an *error*
- The error is the *cumulative* error from many signals
  - E.g. $E(W) = \sum_x \|y - f(x, W)\|^2$
- Optimization will find the W that minimizes total error across all x
- What if wanted to update our parameters after *each* input x instead of waiting for all of them to arrive?

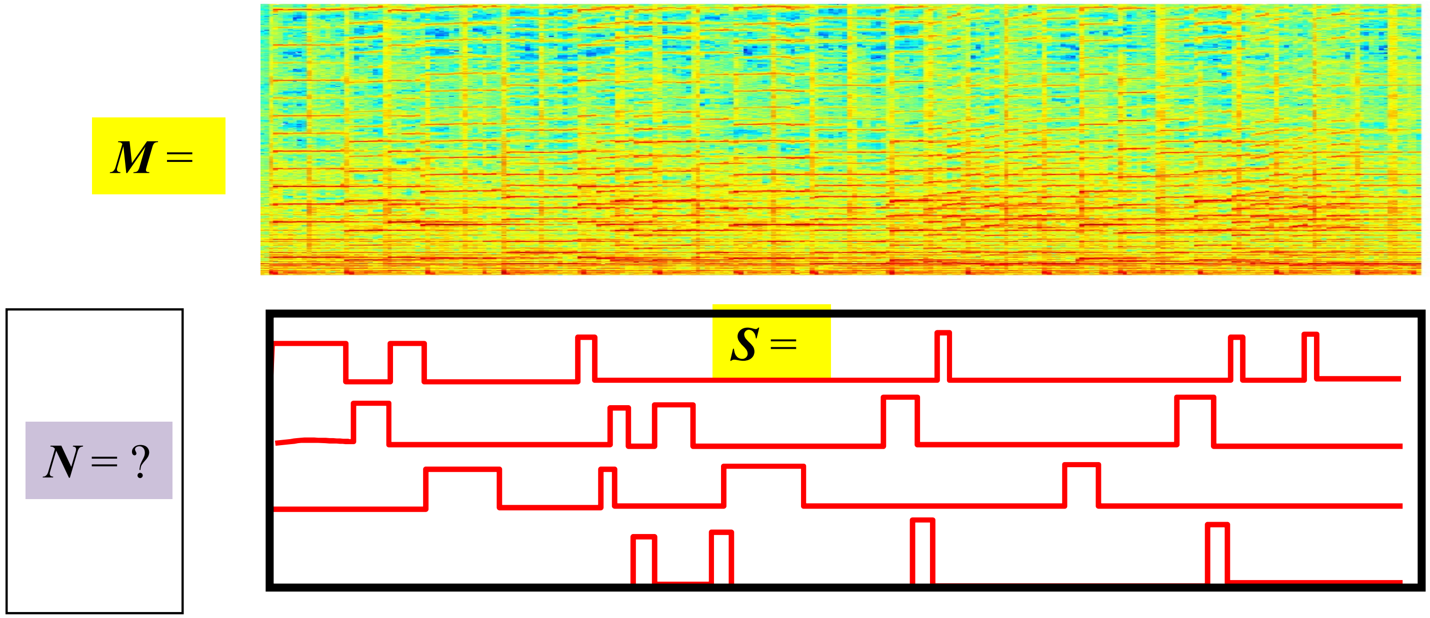
A problem we saw



- Given the *music* M and the *score* S of only four of the notes, but not the notes themselves, find the notes

$$M = NS \quad \Longrightarrow \quad N = M\,Pinv(S)$$

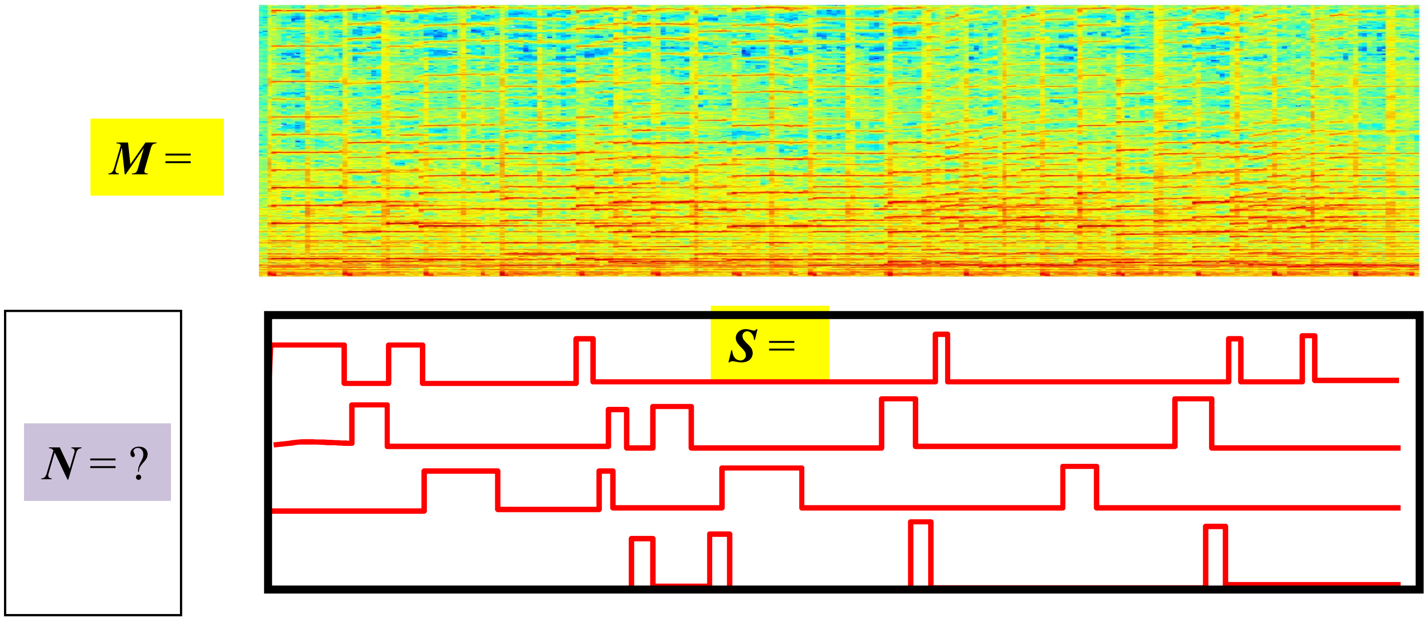
The Actual Problem



- Given the *music* $\boldsymbol{M}$ and the *score* $\boldsymbol{S}$ find a matrix $\boldsymbol{N}$ such the error of reconstruction
  - $E = \sum_i \|M_i - \mathbf{N}S_i\|^2$

  is minimized
- This is a standard optimization problem
- The solution gives us $\boldsymbol{N} = \boldsymbol{M}Pinv(\boldsymbol{S})$

The Actual Problem

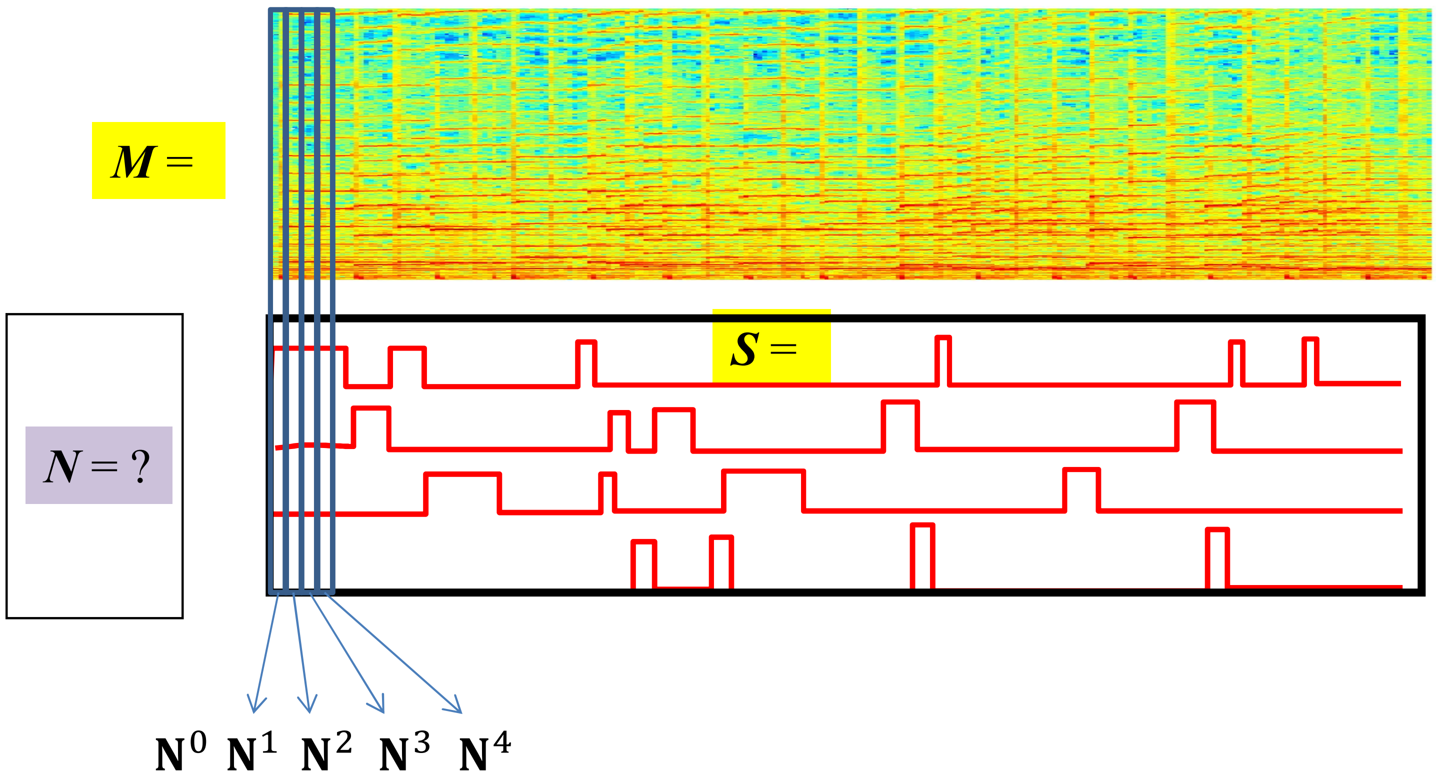


- Given the *music* $\boldsymbol{M}$ and the *score* $\boldsymbol{S}$ find a matrix $\boldsymbol{N}$ such the error of reconstruction
  - $E = \sum_i ||M_i - \mathbf{N}S_i||^2$

  is minimized
- This is a standard optimization problem
- The solution gives us $\boldsymbol{N} = \boldsymbol{M}Pinv(\boldsymbol{S})$

This requires "seeing" *all* of $\boldsymbol{M}$ and $\boldsymbol{S}$ to estimate $\boldsymbol{N}$

Online Updates



- What if we want to update our estimate of the notes after *every input*
  - After observing each vector of music and its score
  - A situation that arises in many similar problems

Incremental Updates

- Easy solution: To obtain the kth estimate $\mathbf{N}^k$, minimize the error on the kth input
  - The error on the kth input is:
$$E_k = M_K - \mathbf{N}S_K$$
  - The *squared error* is:
$$L_k = E_K^2 = \|M_K - \mathbf{N}S_K\|^2$$
  - Differentiating it gives us
$$\nabla\mathbf{N} = 2(M_K - \mathbf{N}S_K)S_K^T = 2E_K S_K^T$$
- Update the parameter to move in the direction of this update
$$\mathbf{N}^{k+1} = \mathbf{N}^k + \eta E_K S_K^T$$

- η must typically be very small to prevent the updates from being influenced entirely by the latest observation

Online update: Non-quadratic functions

- The earlier problem has a *linear* predictor as the underlying model

$$\hat{M}_k = \mathbf{N}S_k$$

- We often have *non-linear* predictors

$$\hat{Y}_k = \mathrm{g}(\mathbf{W}X_k)$$

$$E_k = Y_k - \mathrm{g}(\mathbf{W}X_k)$$

- The derivative of the squared error E_K^2 w.r.t $\mathbf{W}$ is often ugly or intractable
- For such problems we will still use the following generalization of the online update rule for linear predictors

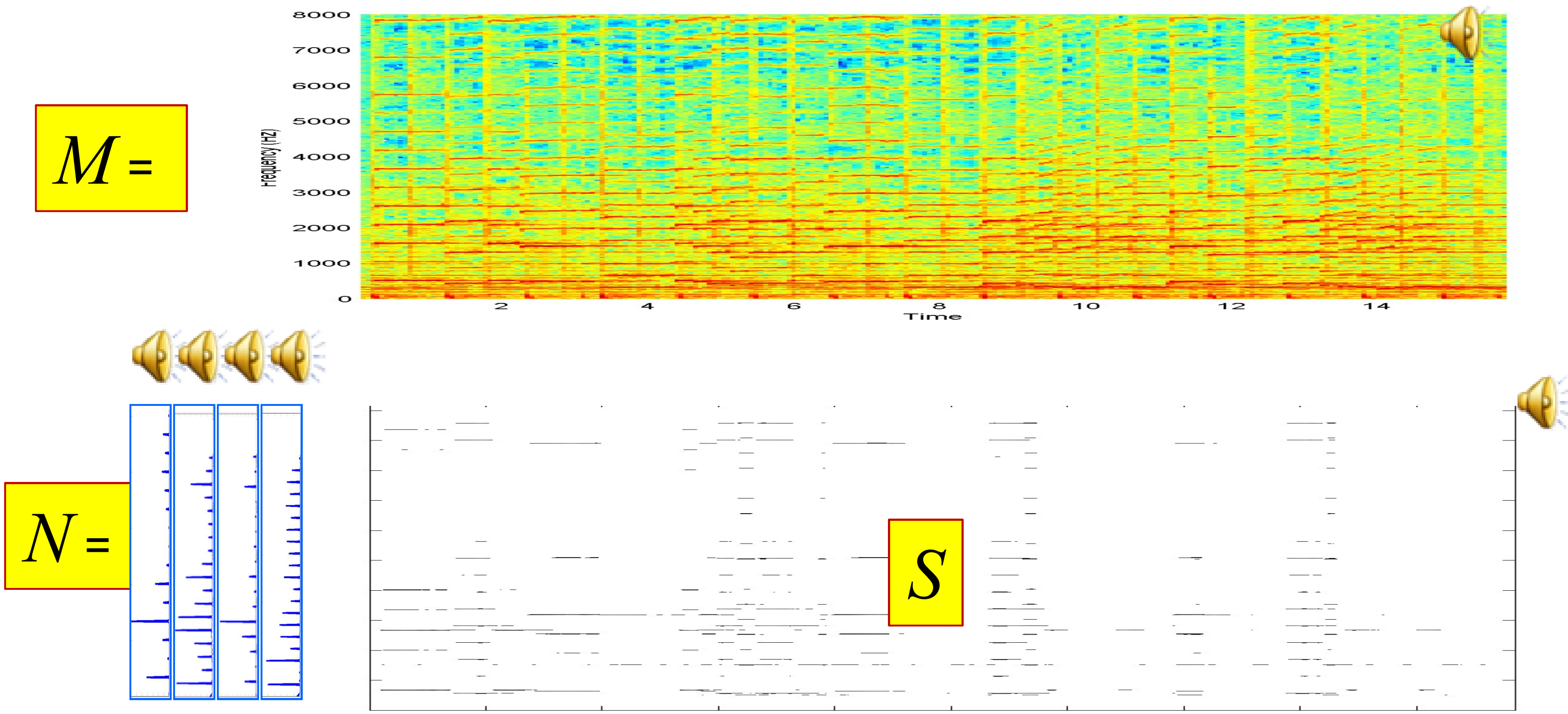
$$\boxed{\mathbf{W}^{k+1} = \mathbf{W}^k + \eta E_k X_k^T}$$

- This is the ***Widrow-Hoff*** rule
  - Based on quadratic Taylor series approximation of $\mathrm{g}(.)$

Index

1. The problem of optimization
2. Direct optimization
3. Descent methods
   - Newton's method
   - Gradient methods
4. Online optimization
5. Constrained optimization
   - Lagrange's method
   - Projected gradients
6. Regularization
7. Convex optimization and Lagrangian duals

A problem we recently saw



- The projection matrix P is the matrix that minimizes the total error between the *projected* matrix S and the *original matrix* M

CONSTRAINED optimization

- Recall the projection problem:
- Find P such that we minimize

$$E = \sum_i ||M_i - PM_i||^2$$

- AND such that the projection is composed of the notes in N

$$P = NC$$

- This is a problem of *constrained optimization*

Optimization problem with constraints

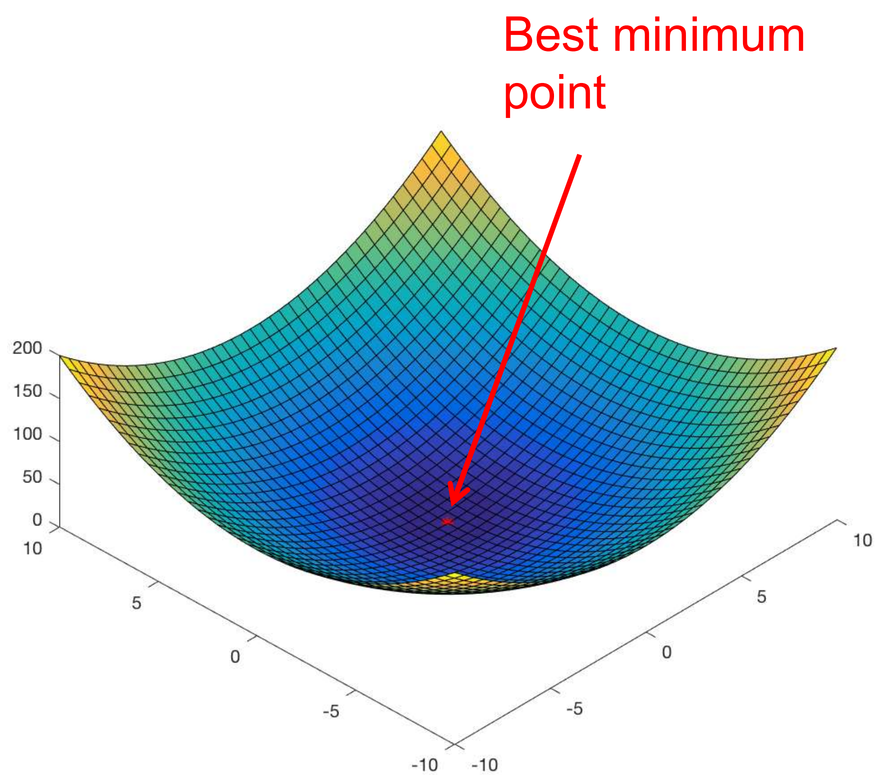
- Finding the minimum of a function $f: \Re^N \longrightarrow \Re$ subject to constraints

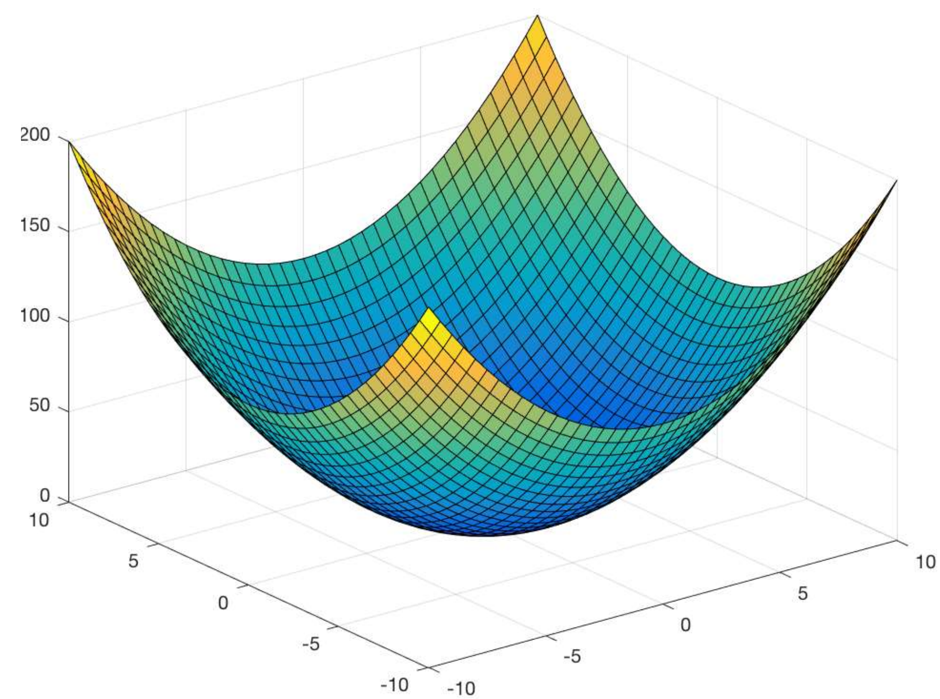
$$\begin{aligned} &\min_x \quad f(x) \\ &s.t.\ \ g_i(x) \le 0 \quad i = \{1, \ldots, k\} \\ &\qquad h_j(x) = 0 \quad j = \{1, \ldots, l\} \end{aligned}$$

- Constraints define a feasible region, which is nonempty

Optimization without constraints

- No Constraints $\min_{x} \quad f(x,y,z) = x^2 + y^2$



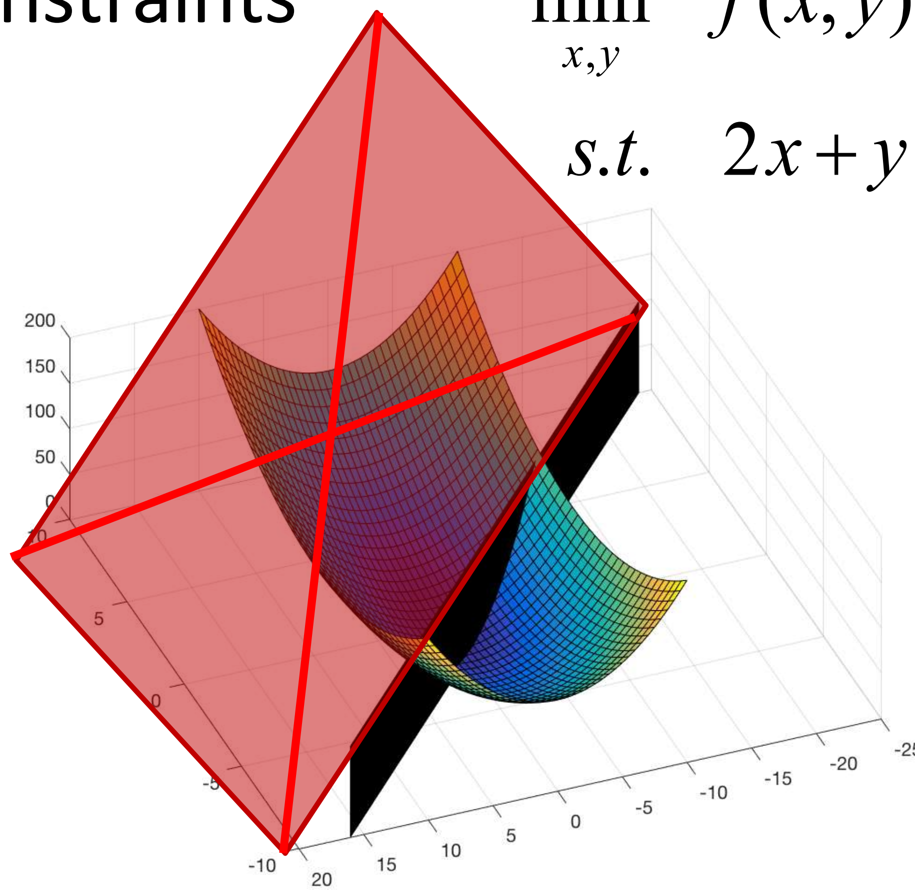


Optimization with constraints

- With Constraints

$$\min_{x,y} \quad f(x,y) = x^2 + y^2$$

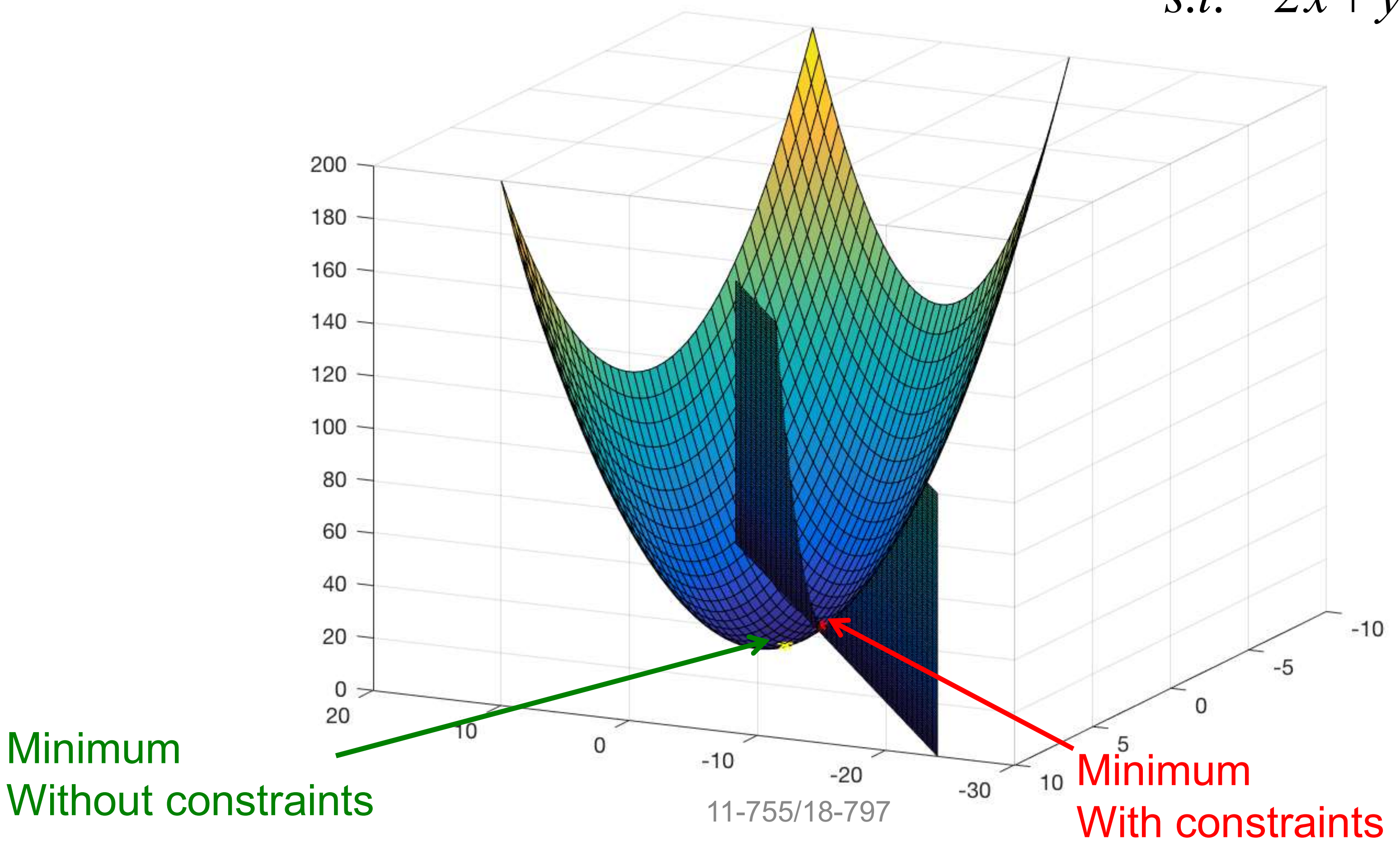
$$s.t. \quad 2x + y \le -4$$



Optimization with constraints

- Minima w/ and w/o constraints

$$\min_{x,y} \quad f(x,y) = x^2 + y^2$$
$$s.t. \quad 2x + y \leq -4$$



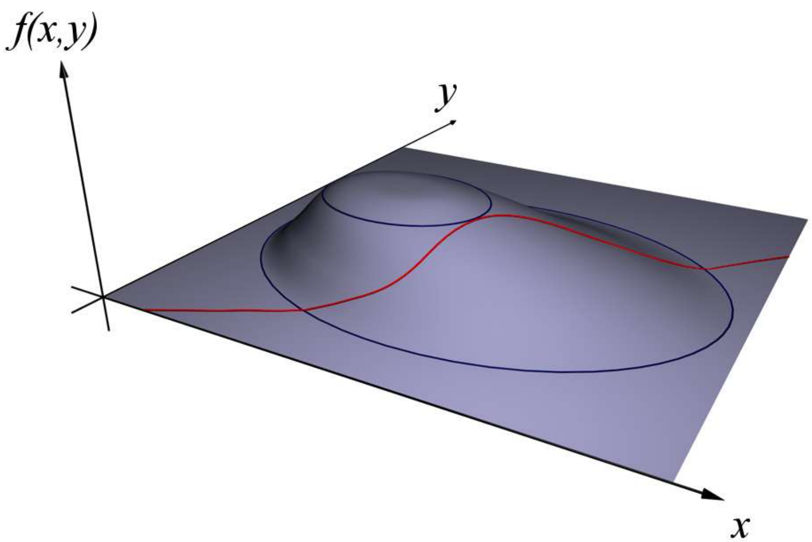
Solving for constrained optimization: the method of Lagrangians

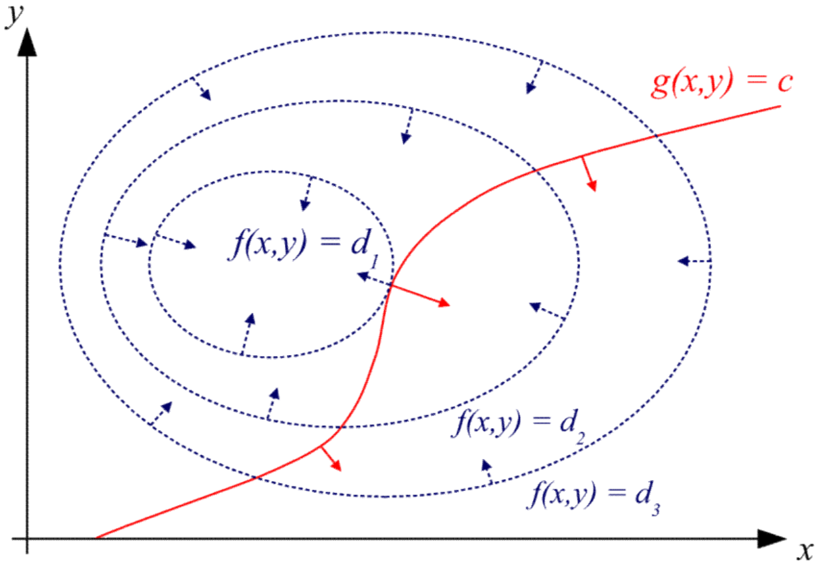
- Consider a function $f(x, y)$ that must be maximized w.r.t (x, y) subject to

$$g(x, y) = c$$

  - Note, we're using a *maximization* example to go with the figures that have been obtained from Wikipedia

The Lagrange Method

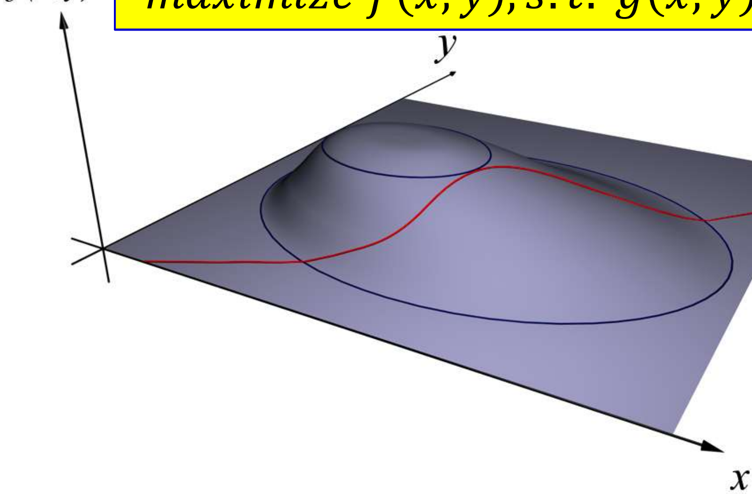


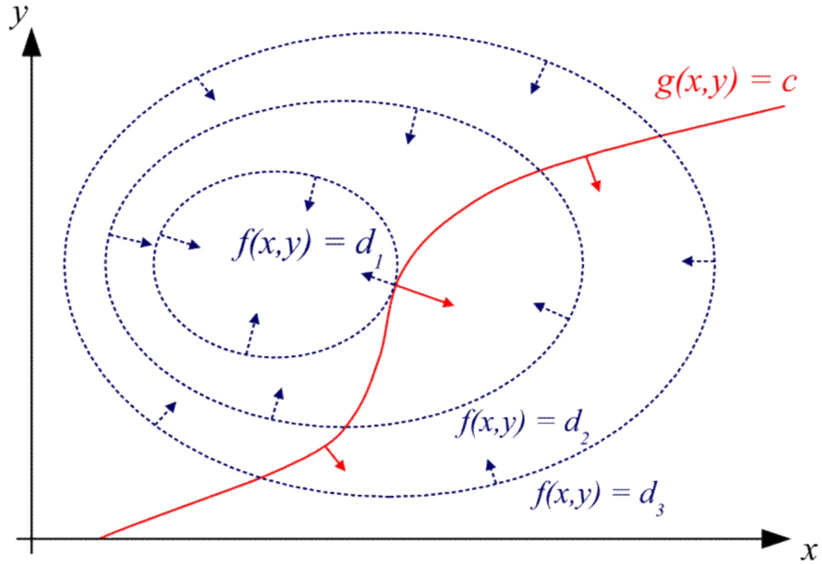


- Purple surface is $f(x, y)$
  - Must be maximized
- Red curve is constraint $g(x, y) = c$
  - All solutions *must* line on this curve
- Problem: Find the position of the largest $f(x, y)$ on the red curve!

The Lagrange Method

$$\boxed{maximize\ f(x,y), s.t.\ g(x,y) = c}$$

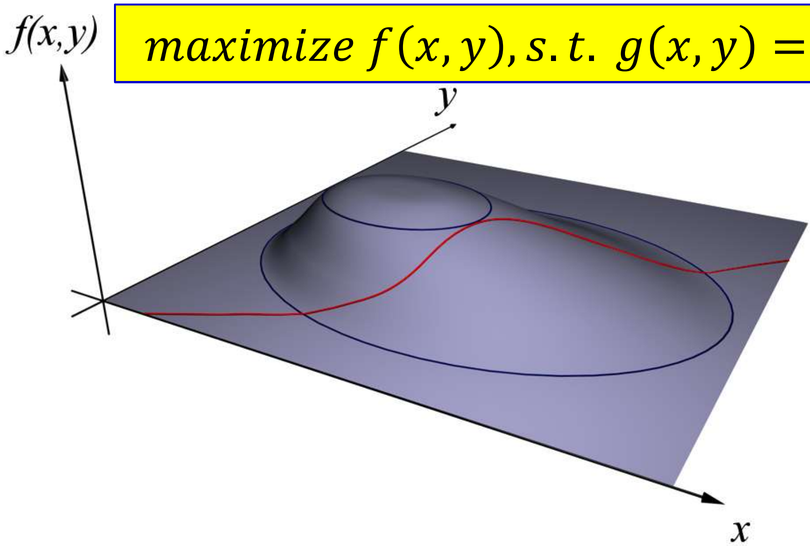


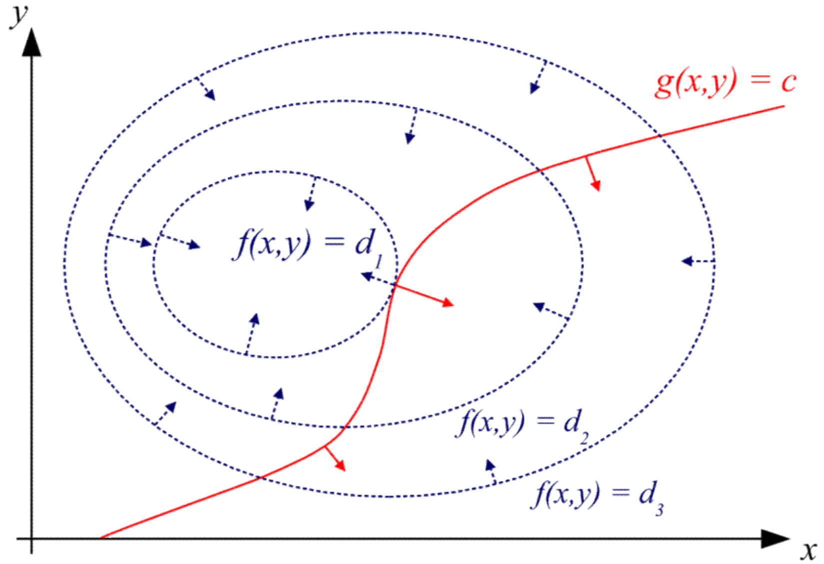


- Dotted lines are constant-value contours $f(x, y) = C$
  - $f(x, y)$ has the same value C at all points on a contour
- The constrained optimum will be at the point where the highest constant-value contour touches the red curve
  - **It will be *tangential* to the red curve**

The Lagrange Method

$maximize\ f(x,y), s.t.\ g(x,y) = c$



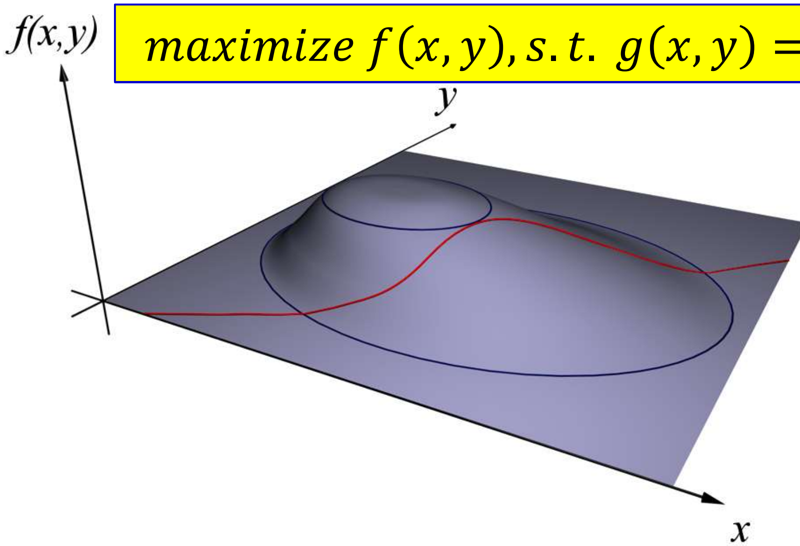


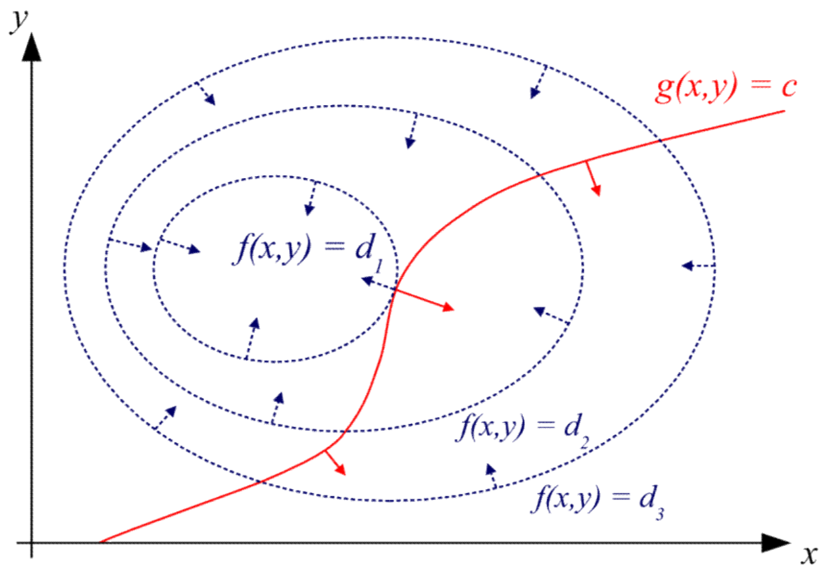
- The constrained optimum is where the highest constant-value contour is tangential to the red curve

- The *gradient* of $f(x,y) \ = \ C$ will be parallel to the gradient of $g(x,y) \ = \ c$

The Lagrange Method

$maximize\ f(x,y), s.t.\ g(x,y) = c$





- At the optimum

$$\nabla f(x,y) = \lambda \nabla g(x,y)$$
$$g(x,y) = c$$

- Find (x,y) that satisfies both above conditions

The Lagrange Method

$$\nabla f(x, y) = \lambda \nabla g(x, y)$$
$$g(x, y) = c$$

- Find (x, y) that satisfies both above conditions
- Combine the above two into one equation

$$\boxed{L(x, y, \lambda) = f(x, y) - \lambda(g(x, y) - c)}$$

- Optimize it for (x, y, λ)

- Solving for (x, y),

$$\nabla_{x,y} L(x, y, \lambda) = 0 \quad \Longrightarrow \quad \nabla f(x, y) = \lambda \nabla g(x, y)$$

- Solving for λ

$$\frac{\partial L(x, y, \lambda)}{\partial \lambda} = 0 \quad \Longrightarrow \quad g(x, y) = c$$

The Lagrange Method

$$\nabla f(x, y) = \lambda \nabla g(x, y)$$
$$g(x, y) = c$$

- Find (x, y) that satisfies both above conditions
- Combine the above two into one equation

$$\boxed{L(x, y, \lambda) = f(x, y) - \lambda(g(x, y) - c)}$$

- Optimize it for (x, y, λ)
- Sol
- Sol

Formally:

to maximize $f(x, y)$: $\max_{x,y}\left(\min_{\lambda} L(x, y, \lambda)\right)$

to minimize $f(x, y)$: $\min_{x,y}\left(\max_{\lambda} L(x, y, \lambda)\right)$

Generalizes to inequality constraints

- Optimization problem with constraints

$$\min_x f(x)$$
$$s.t. g_i(x) \leq 0 \; i = \{1,\ldots,k\}$$
$$h_j(x) = 0 \; j = \{1,\ldots,l\}$$

- Lagrange multipliers $\lambda_i \geq 0, \nu \in \Re$

$$L(x,\lambda,\nu) = f(x) + \sum_{i=1}^{k} \lambda_i g_i(x) + \sum_{j=1}^{l} \nu_j h_j(x)$$

- The necessary condition

$$\nabla L(x,\lambda,\nu) = 0 \Leftrightarrow \frac{\partial L}{\partial x} = 0, \; \frac{\partial L}{\partial \lambda} = 0, \frac{\partial L}{\partial \nu} = 0$$

Generalizes to inequality constraints

- Optimization problem with c

$$\min_x f(x)$$
$$s.t.\, g_i(x) \le 0 \; i = \{1,..$$
$$h_j(x) = 0 \; j = \{1,\ldots$$

Maximize w.r.t λ

If constraint is not satisfied
this term can be made to
go to inf with high choice of λ

Minimizing the loss while maximizing
λ forces constraint to be satisfied
and λ to go to 0

- Lagrange multipliers $\lambda_i \ge 0, \nu \in \Re$

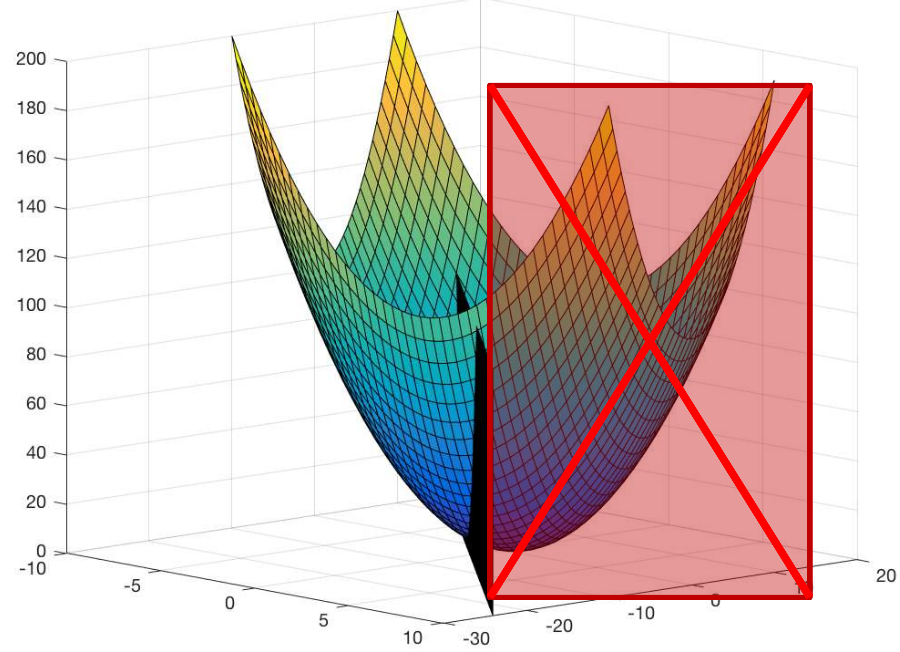
$$L(x,\lambda,\nu) = f(x) + \sum_{i=1}^{k} \lambda_i g_i(x) + \sum_{j=1}^{l} \nu_j h_j(x)$$

- The necessary condition

$$\nabla L(x,\lambda,\nu) = 0 \Leftrightarrow \frac{\partial L}{\partial x} = 0, \; \frac{\partial L}{\partial \lambda} = 0, \frac{\partial L}{\partial \nu} = 0$$

Lagrange multiplier example

$$\min_{x,y} f(x,y) = x^2 + y^2$$
$$s.t.\, 2x + y \leq -4$$



- Lagrange multiplier

$$L = x^2 + y^2 + \lambda(2x + y + 4)$$

- Evaluate

$$\nabla L(x, \lambda, \nu) = 0 \Leftrightarrow \frac{\partial L}{\partial x} = 0, \frac{\partial L}{\partial \lambda} = 0, \frac{\partial L}{\partial \nu} = 0$$

Lagrange multiplier example

- Critical point

$$\frac{\partial L}{\partial x} = 2x + 2\lambda = 0$$

$$\frac{\partial L}{\partial y} = 2y + \lambda = 0$$

$$\frac{\partial L}{\partial \lambda} = 2x + y + 4 = 0$$

$$x = -\lambda$$

$$y = -\frac{\lambda}{2}$$

$$2x + y + 4 = 0$$

$$-2\lambda + \left(-\frac{\lambda}{2}\right) + 4 = 0$$

$$-\frac{5}{2}\lambda = -4$$

$$\lambda = \frac{8}{5}$$

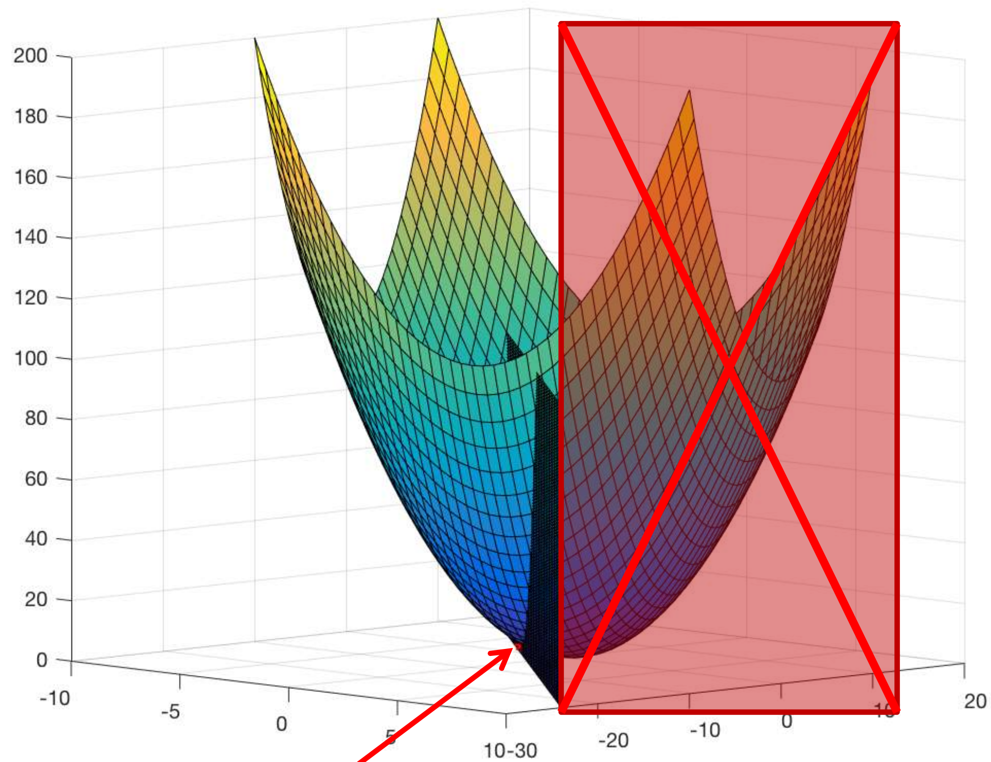
$$x = -\frac{8}{5}$$

$$y = -\frac{4}{2}$$

Optimization with constraints

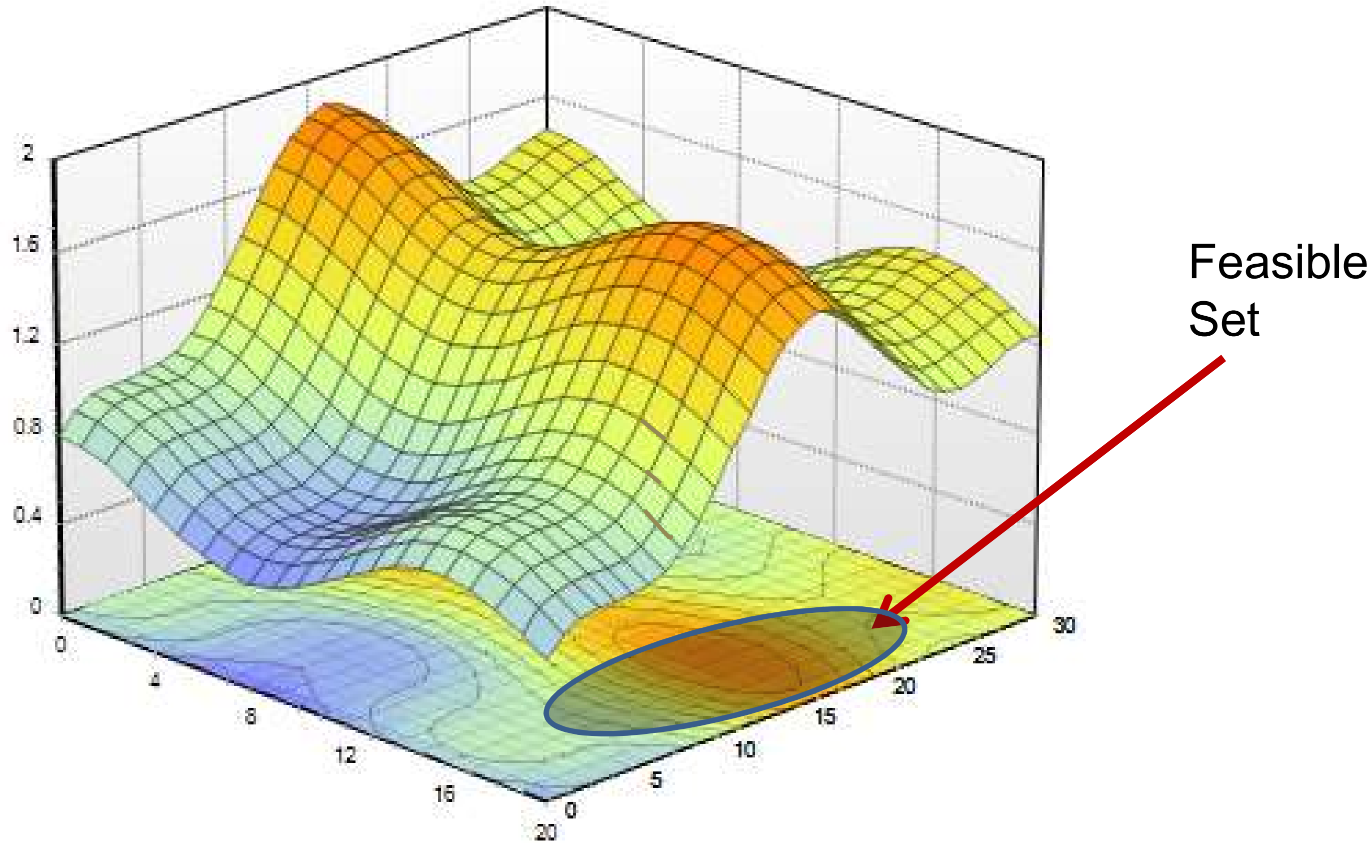
- Lagrange Multiplier results

$$\min_{x,y} \quad f(x,y) = x^2 + y^2$$
$$s.t. \quad 2x + y \le -4$$



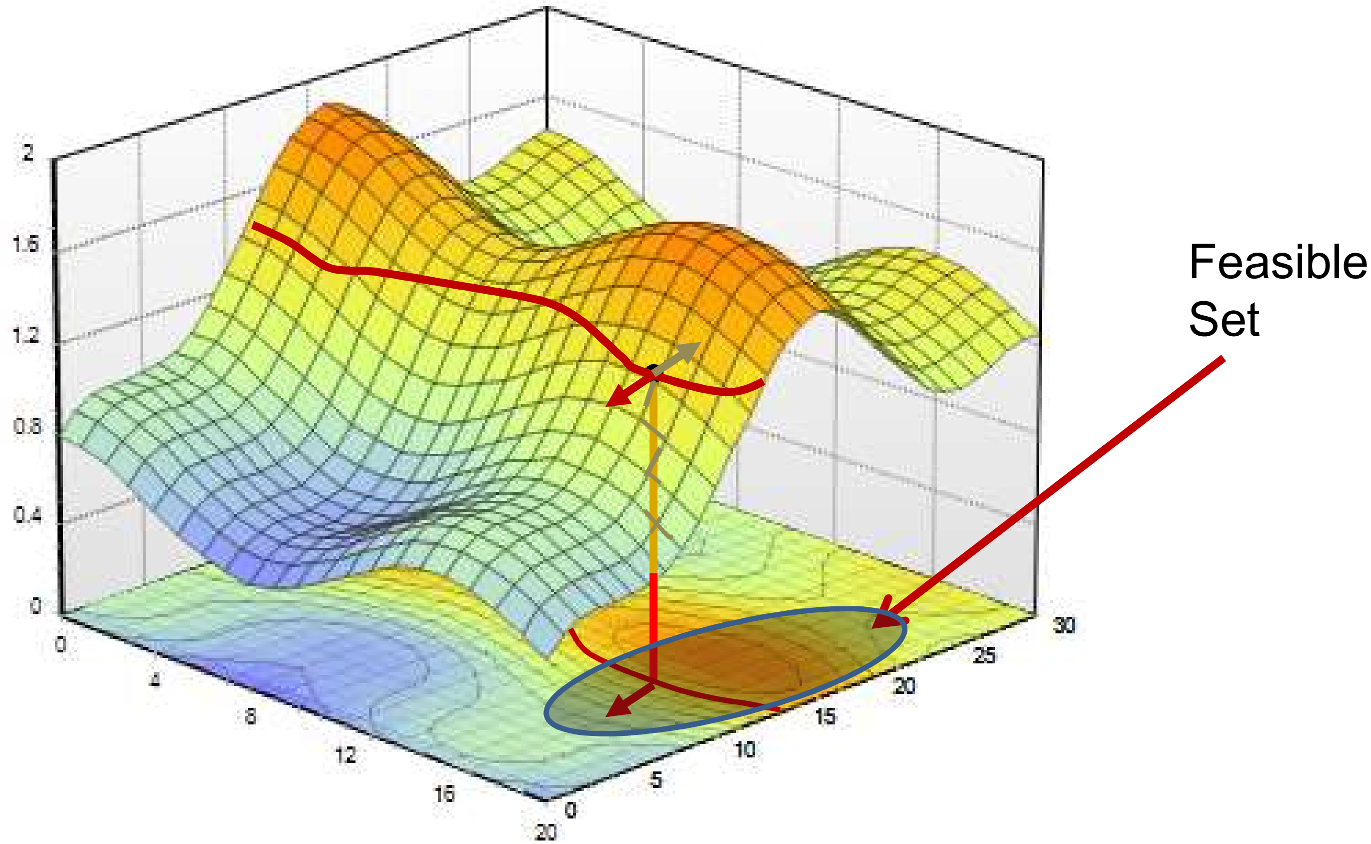
Minimum With constraints
(-8/5,-4/5,16/5)

An Alternate Approach: Projected Gradients



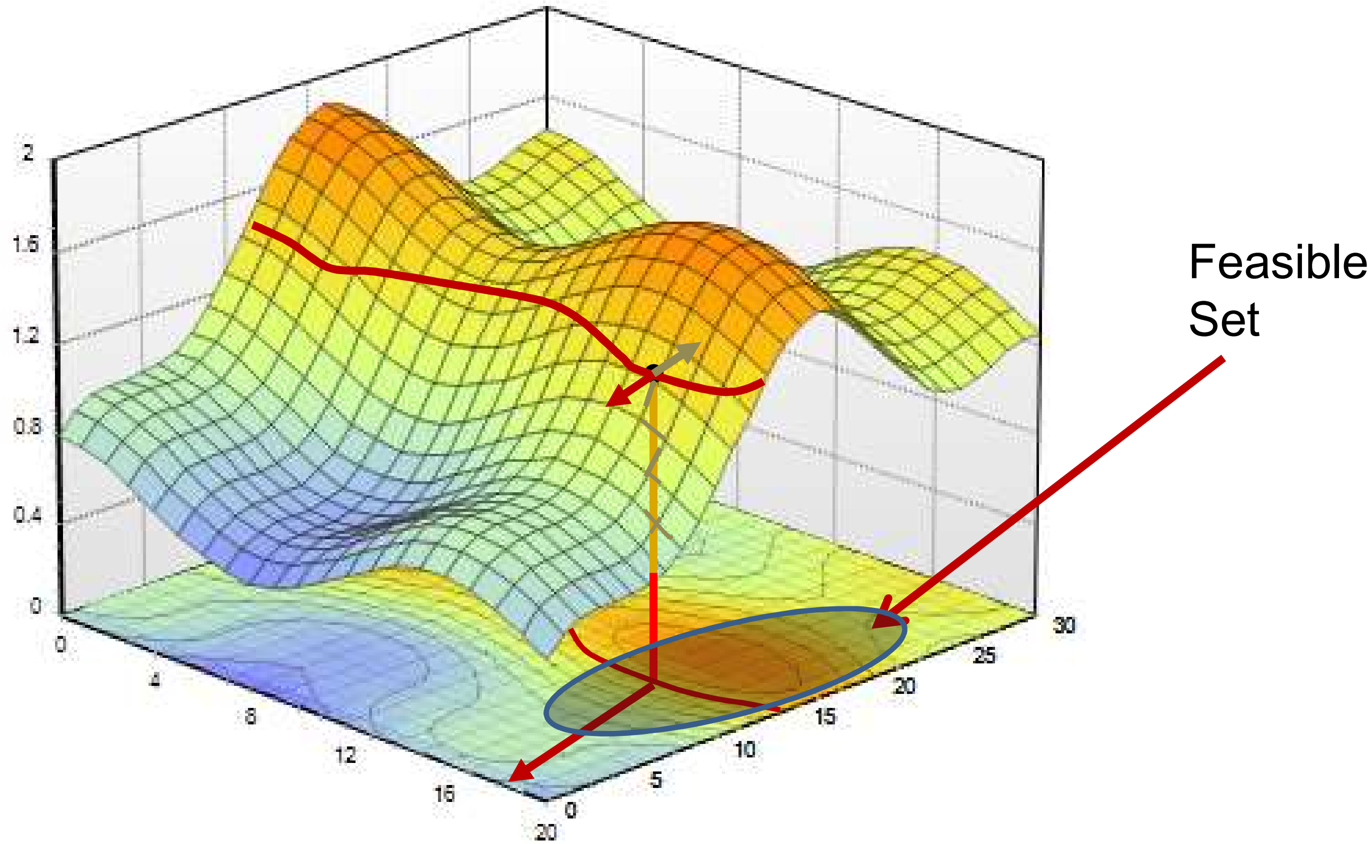
- The constraints specify a "feasible set"
  - The region of the space where the solution can lie

An Alternate Approach: Projected Gradients



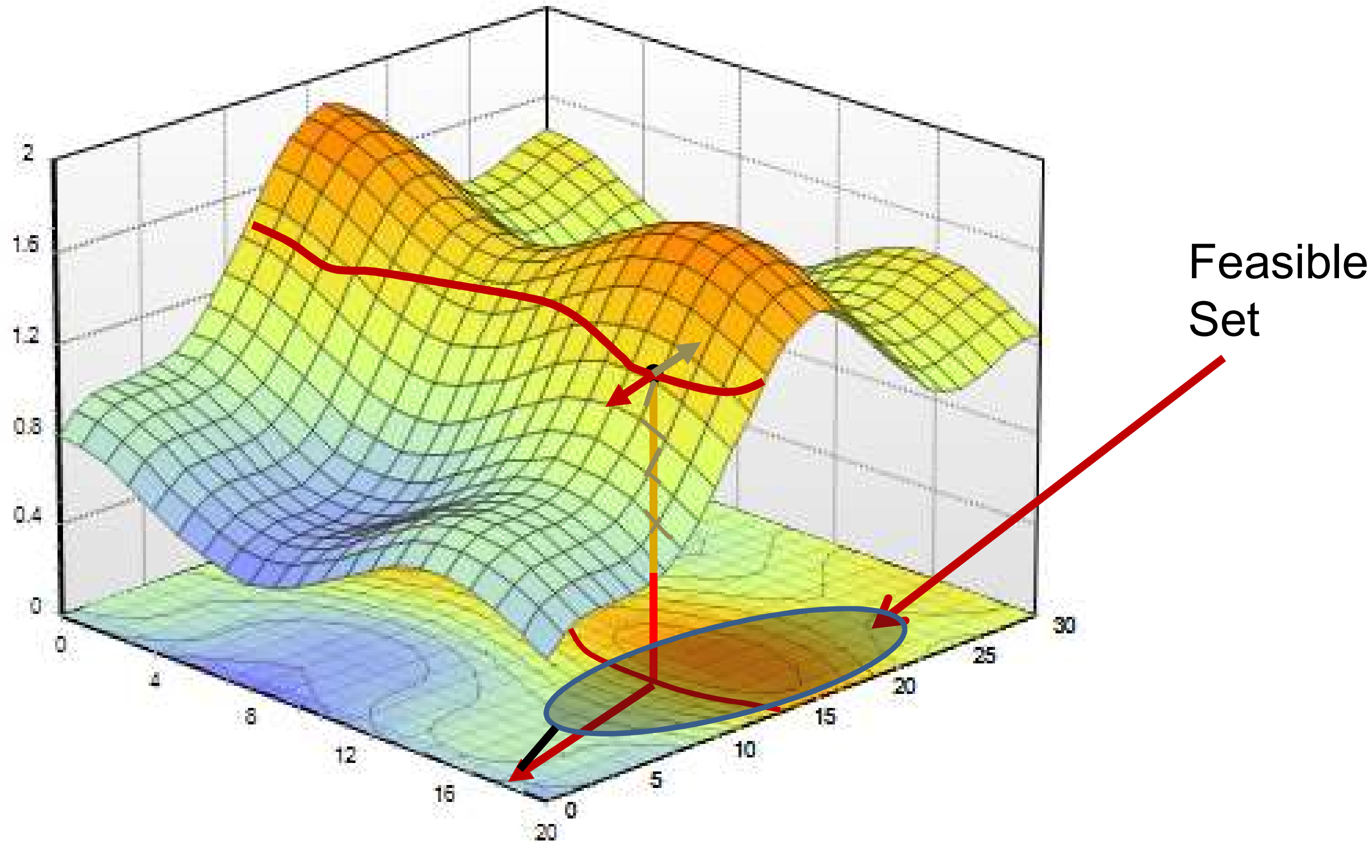
- From the current estimate, take a step using the conventional gradient descent approach
  - If the update is inside the feasible set, no further action is required

An Alternate Approach: Projected Gradients



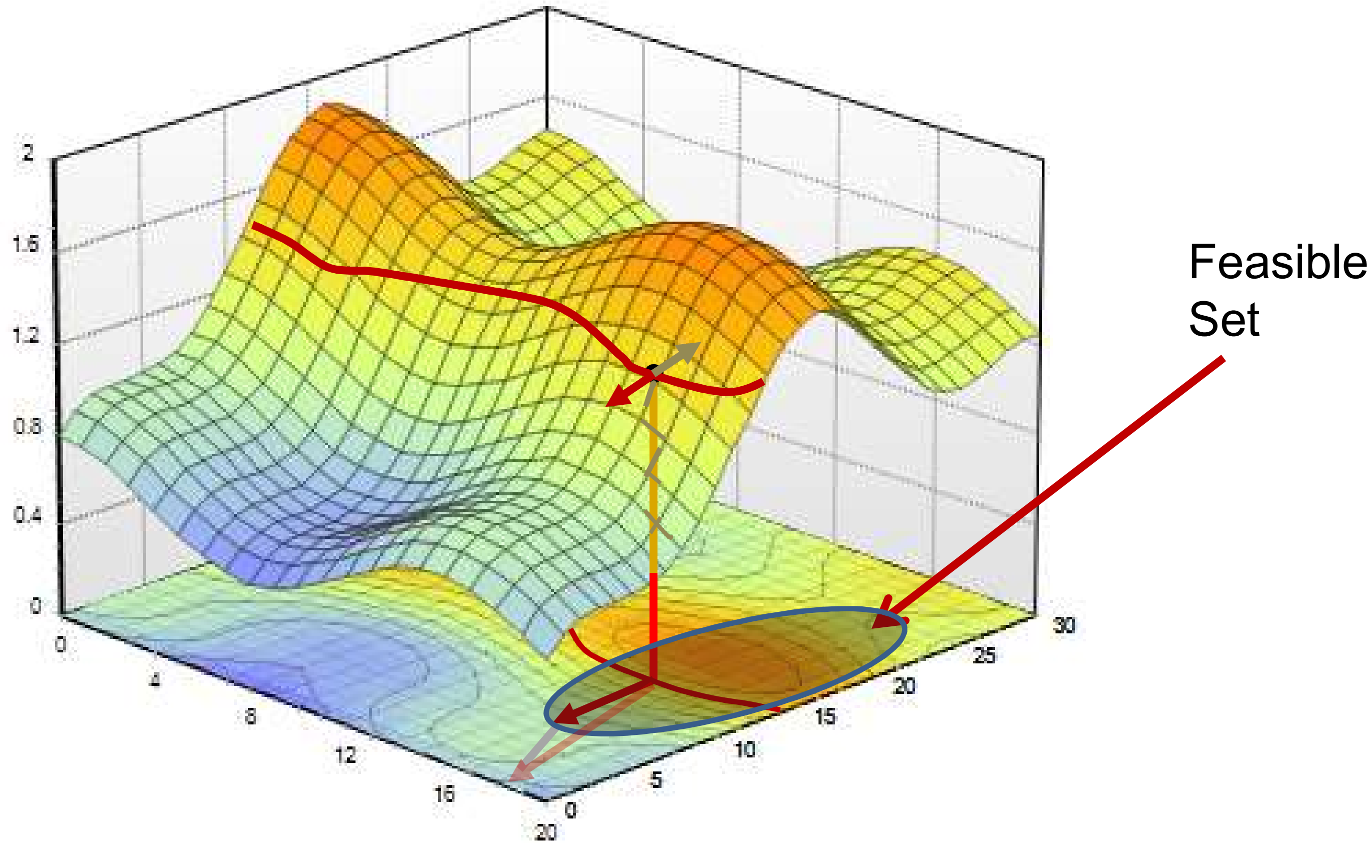
- If the update falls *outside* the feasible set,

An Alternate Approach: Projected Gradients



- If the update falls *outside* the feasible set,
  - find the closest point to the update on the boundary of the feasible set

An Alternate Approach: Projected Gradients



- If the update falls *outside* the feasible set,
  - find the closest point to the update on the boundary of the feasible set
  - And *move* the updated estimate to this new point

The method of projected gradients

$$\min_x f(x)$$
$$s.t. g_i(x) \leq 0 \; i = \{1,\ldots,k\}$$

- The constraints specify a "feasible set"
  - The region of the space where the solution can lie

- Update current estimate using the conventional gradient descent approach
  - If the update is inside the feasible set, no further action is required
  - If the update falls *outside* the feasible set,
    - find the closest point to the update on the boundary of the feasible set
    - And *move* the updated estimate to this new point
- The closest point "projects" the update onto the feasible set
- **For many problems, however, finding this "projection" can be difficult or intractable**

Index

1. The problem of optimization
2. Direct optimization
3. Descent methods
   – Newton's method
   – Gradient methods
4. Online optimization
5. Constrained optimization
   – Lagrange's method
   – Projected gradients
6. Regularization
7. Convex optimization and Lagrangian duals

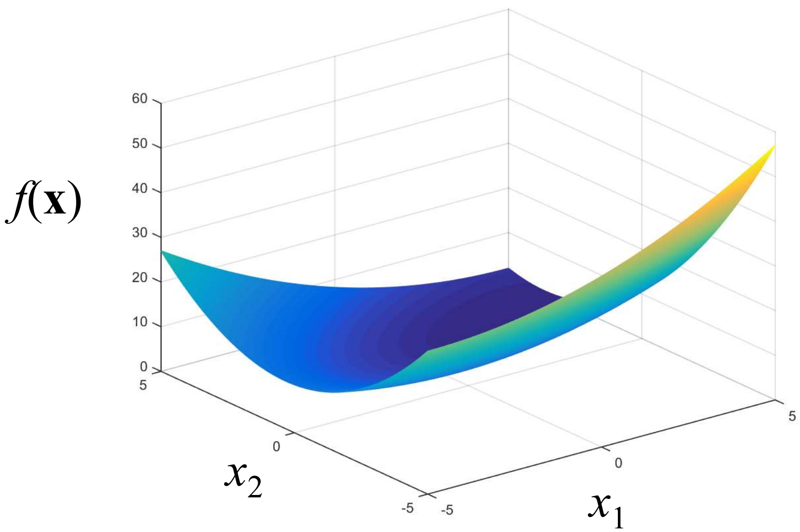
Regularization

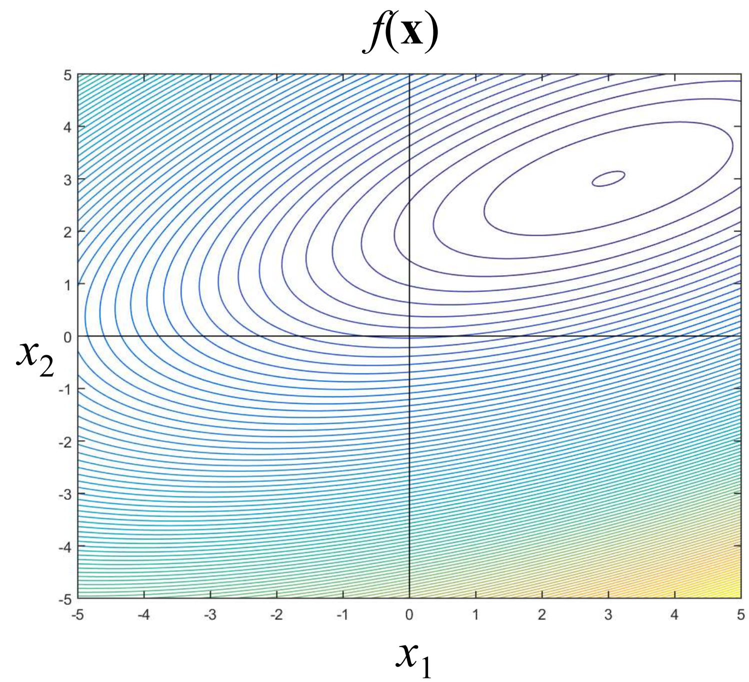
- Sometimes we have additional "regularization" on the parameters
  - Note these are not hard constraints

- E.g.
  - Minimize $f(X)$ while requiring that the length $\|X\|^2$ is also minimum
  - Minimize $f(X)$ while requiring that $|X|_1$ is also minimal
  - Minimize $f(X)$ such that $g(X)$ is maximum

- We will encounter problems where such requirements are logical

Contour Plot of a Quadratic Objective





- Left: Actual 3D plot
  - $\mathbf{x} = [x_1, x_2]$
- Right: constant-value contours
  - Innermost contour has lowest value
- Unconstrained/unregularized solution: The center of the innermost contour

Examples of regularization

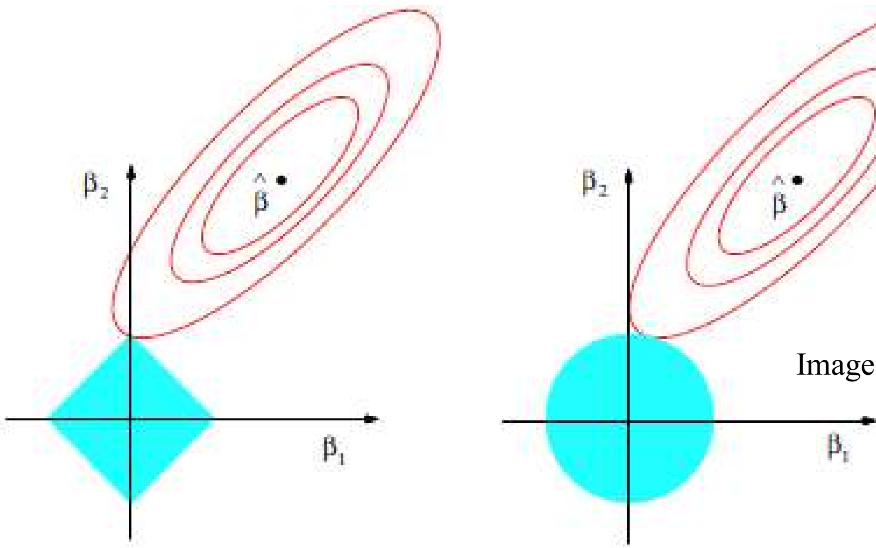


Image Credit: Tibshirani

- Left: "L_1" regularization, find $\mathbf{x}$ that minimizes $f(\mathbf{x})$
  - Also minimize $|\mathbf{x}|_1$
  - $|\mathbf{x}|_1$ = const is a diamond
  - Find $\mathbf{x}$ that also minimizes "diameter" of diamond

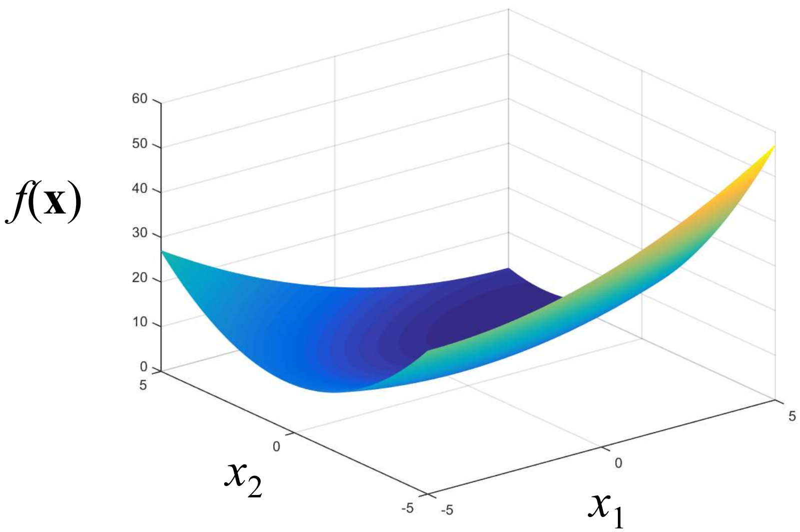
- Right: "L_2" or Tikhonov regularization
  - Also minimize $||\mathbf{x}||^2$
  - $||\mathbf{x}||^2$ = const is a circle (sphere)
  - Find $\mathbf{x}$ that also minimizes "diameter" of circle

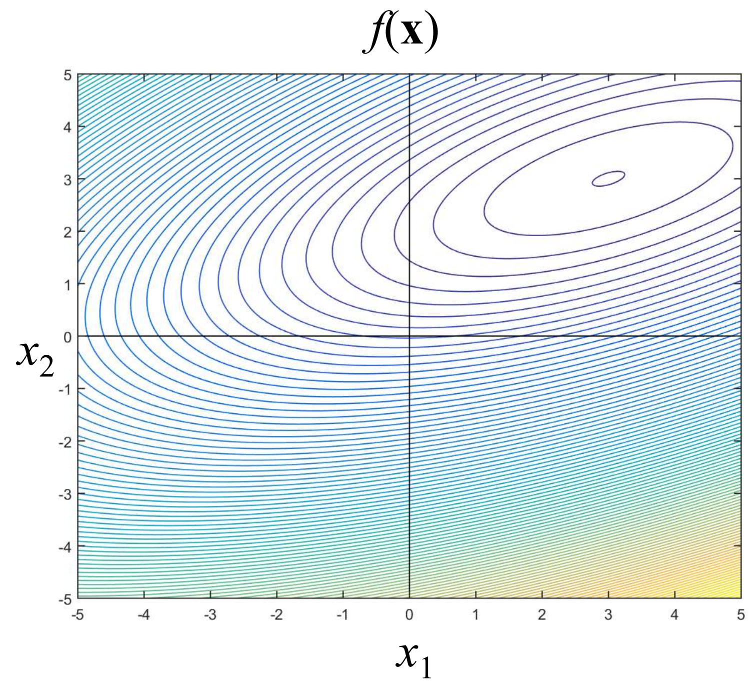
Regularization

- The problem: multiple simultaneous objectives
  - Minimize $f(X)$
  - Also minimize $g_1(X), g_2(X), \ldots$
    - These are "regularizers"

- Solution: Define
  - $L(X) = f(X) + \lambda_1 g_1(X) + \lambda_2 g_2(X) + \cdots$
  - λ_1, λ_2 etc are regularization parameters. These are set and not estimated
    - Unlike Lagrange multipliers
  - Minimize $L(X)$

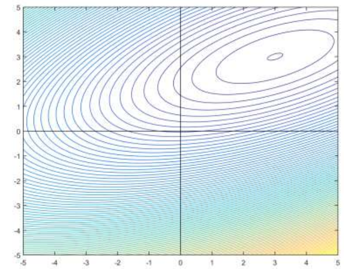
Contour Plot of a Quadratic Objective

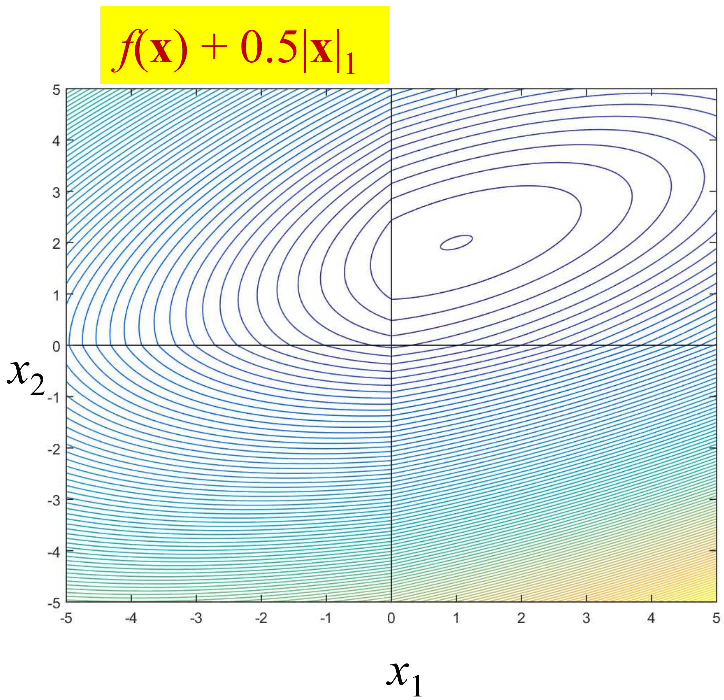


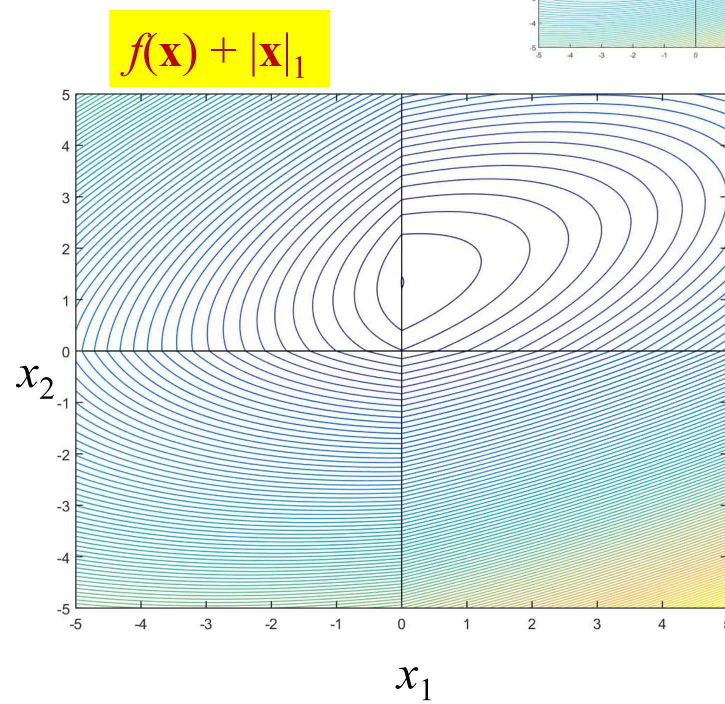


- Left: Actual 3D plot
  - $\mathbf{x} = [x_1, x_2]$
- Right: equal-value contours of $f(\mathbf{x})$
  - Innermost contour has lowest value

With L_1 regularization

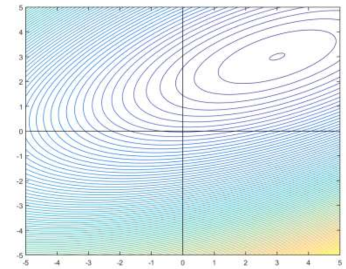




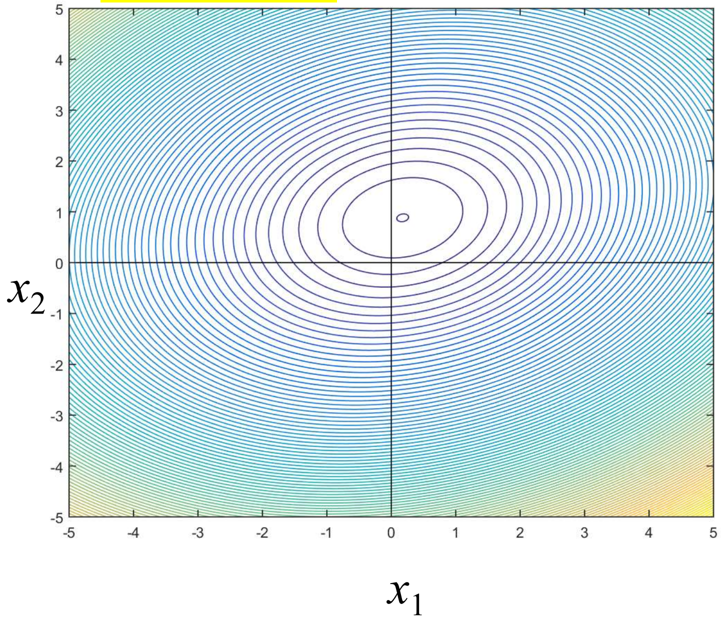


- L_1 regularized objective $f(\mathbf{x}) + \lambda|\mathbf{x}|_1$, for different values of regularization parameter λ
  - Note: Minimum value occurs on x_1 axis for $\lambda = 1$
    - "Sparse" solution

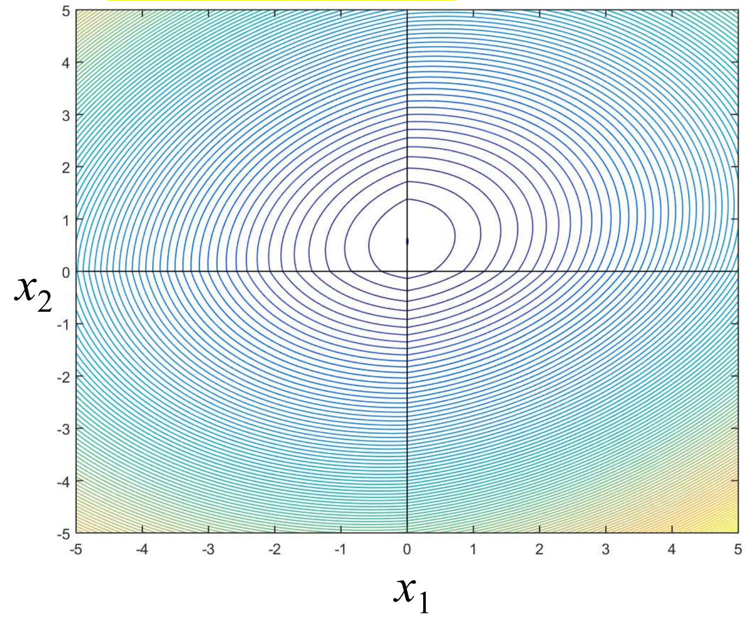
L_2 and L_1-L_2regularization



$f(\mathbf{x}) + |\mathbf{x}|_2^2$



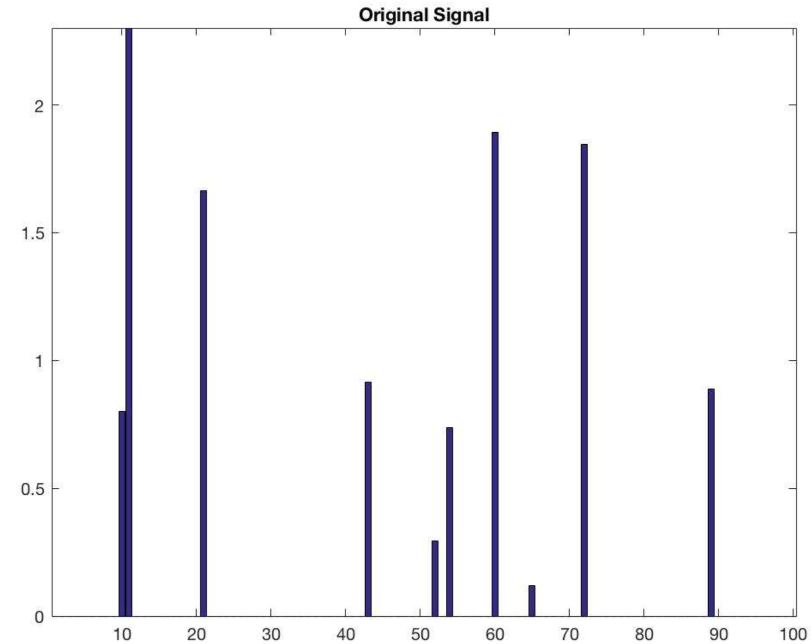
$f(\mathbf{x}) + |\mathbf{x}|_1 + |\mathbf{x}|_2^2$



- L_2 regularized objective $f(\mathbf{x}) + \lambda\|\mathbf{x}\|^2$ results in "shorter" optimum
- L_1-L_2 regularized objective results in sparse, short optimum
  - λ = 1 for both regularizers in example

Regularization

- Sparse signal reconstruction
  - Minimum Square Error
- Signal $\hat{x}$ of length 100
- 10 non-zero components

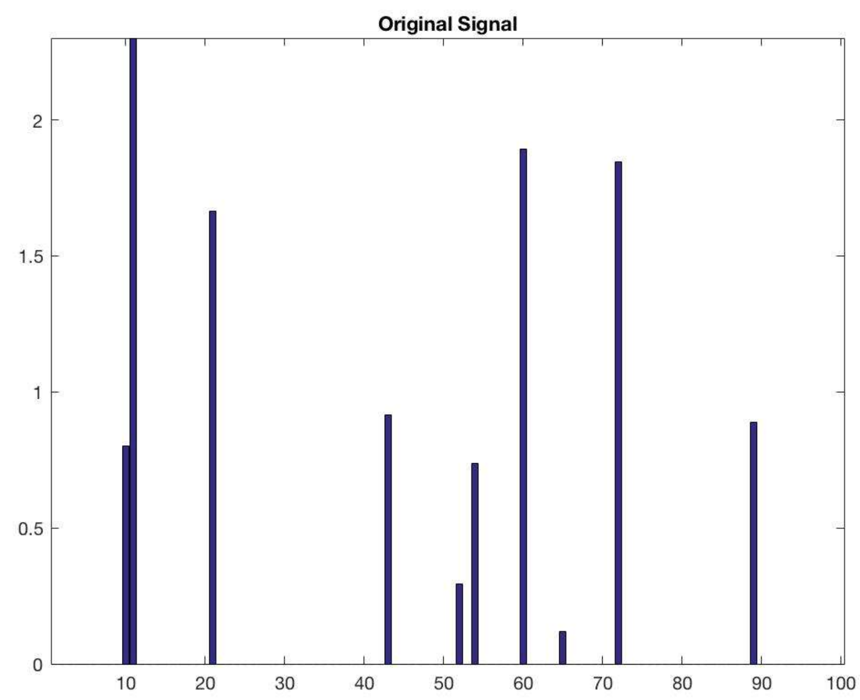


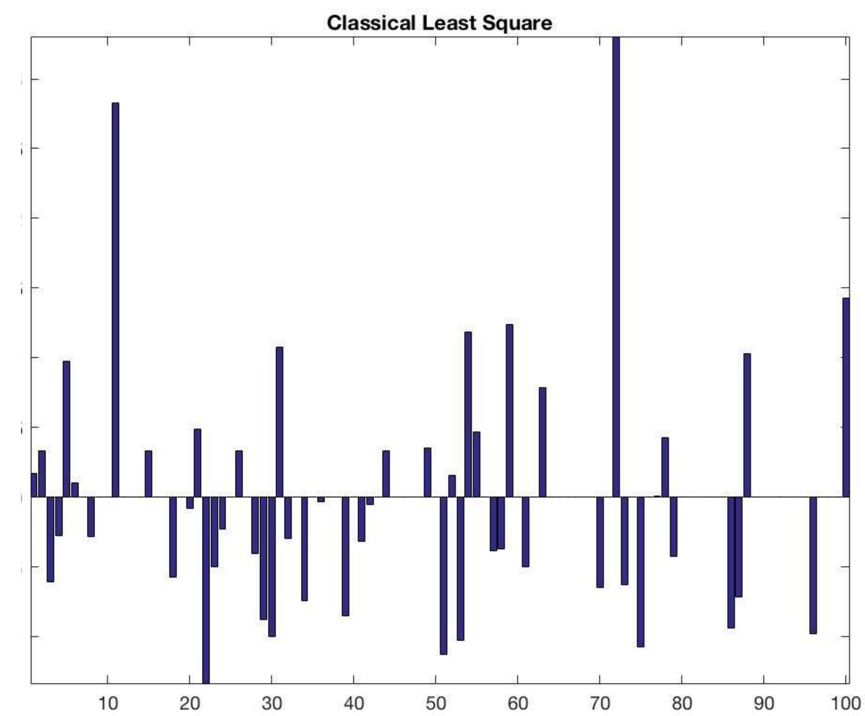
- Reconstructing the original signal from noisy 50 measurements

$$b = A\hat{x} + \varepsilon$$

Signal reconstruction
Minimum Square Error

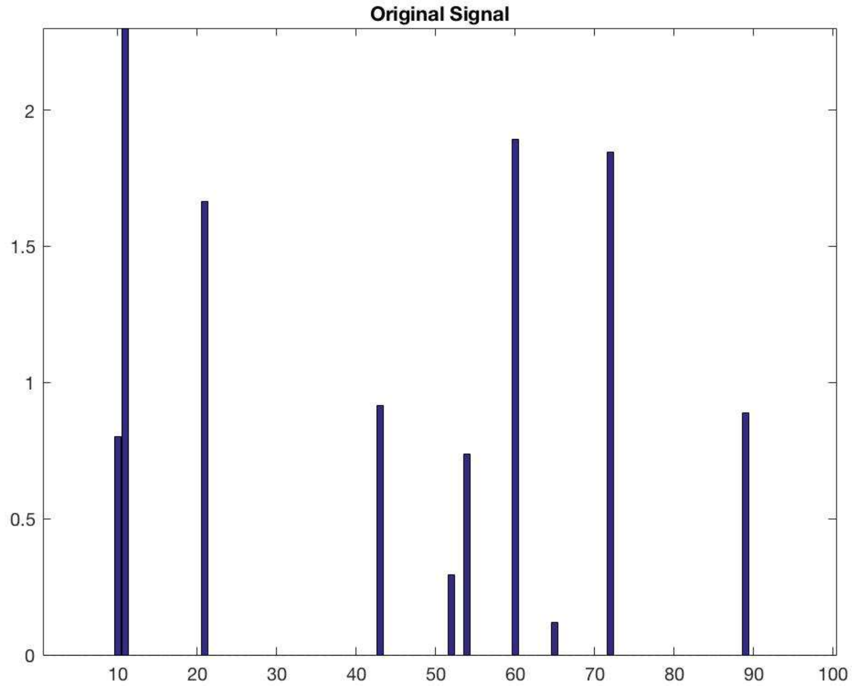
- Signal reconstruction
- Least square problem $\min \|Ax - b\|_2^2$

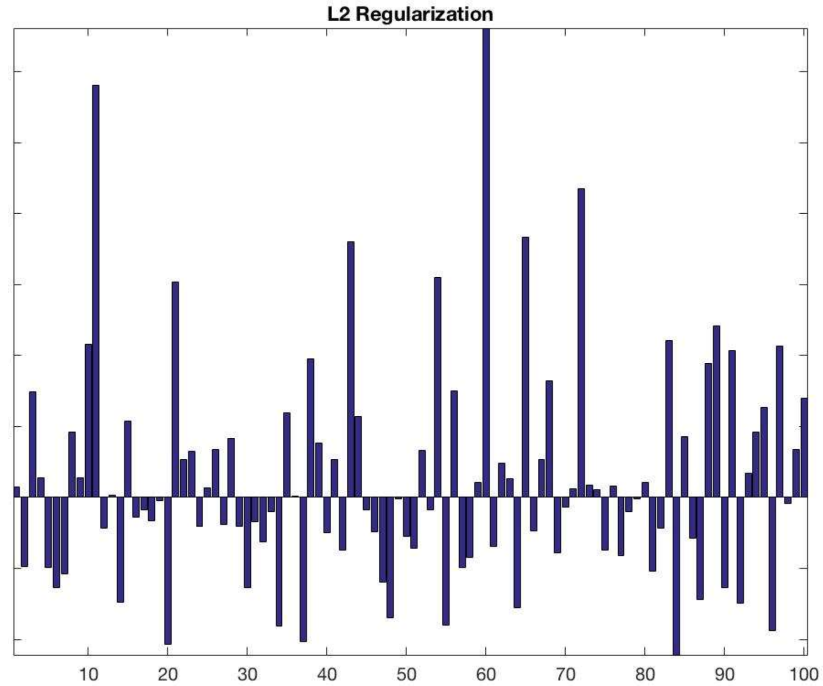




L2-Regularization

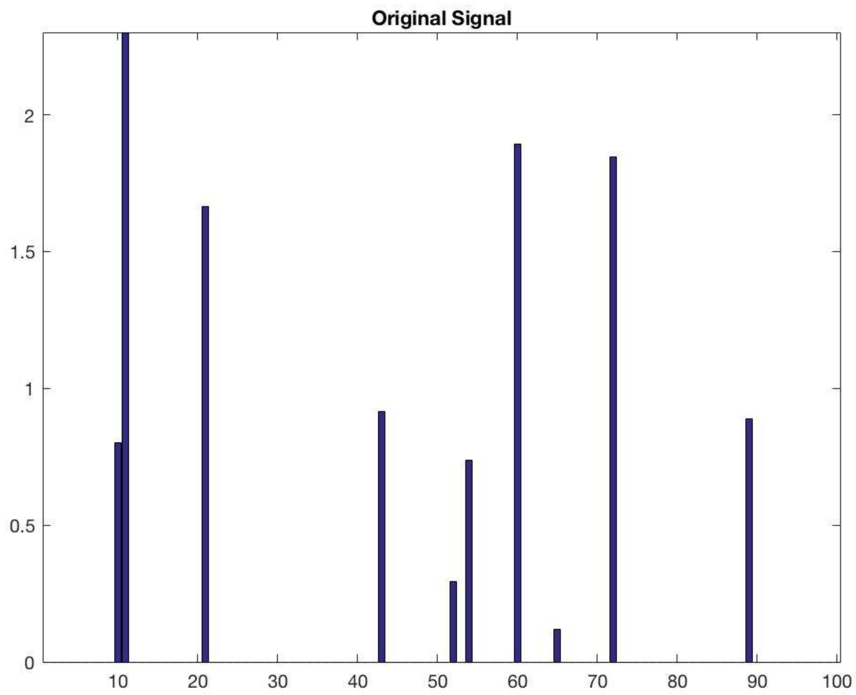
- Signal reconstruction
- Least squares problem $\min \|Ax-b\|_2^2 + \gamma \|x\|_2^2$

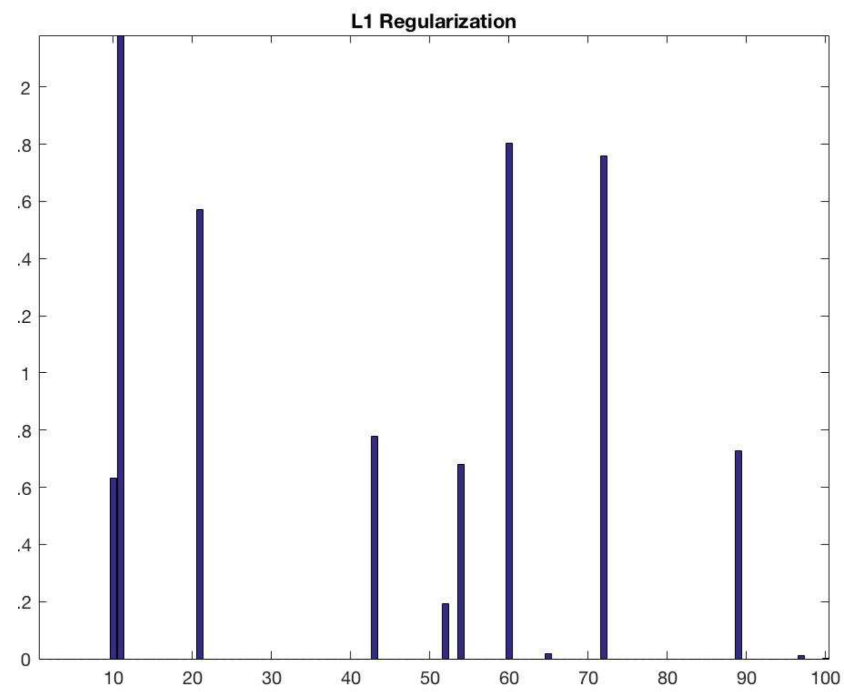




L1-Regularization

- Signal reconstruction
- Least square problem $\min \|Ax - b\|_2^2 + \gamma \|x\|_1$





Index

1. The problem of optimization
2. Direct optimization
3. Descent methods
   - Newton’s method
   - Gradient methods
4. Online optimization
5. Constrained optimization
   - Lagrange’s method
   - Projected gradients
6. Regularization
7. Convex optimization and Lagrangian duals

Convex optimization Problems

- An convex optimization problem is defined by
  - convex objective function
  - Convex inequality constraints f_i
  - Affine equality constraints h_j

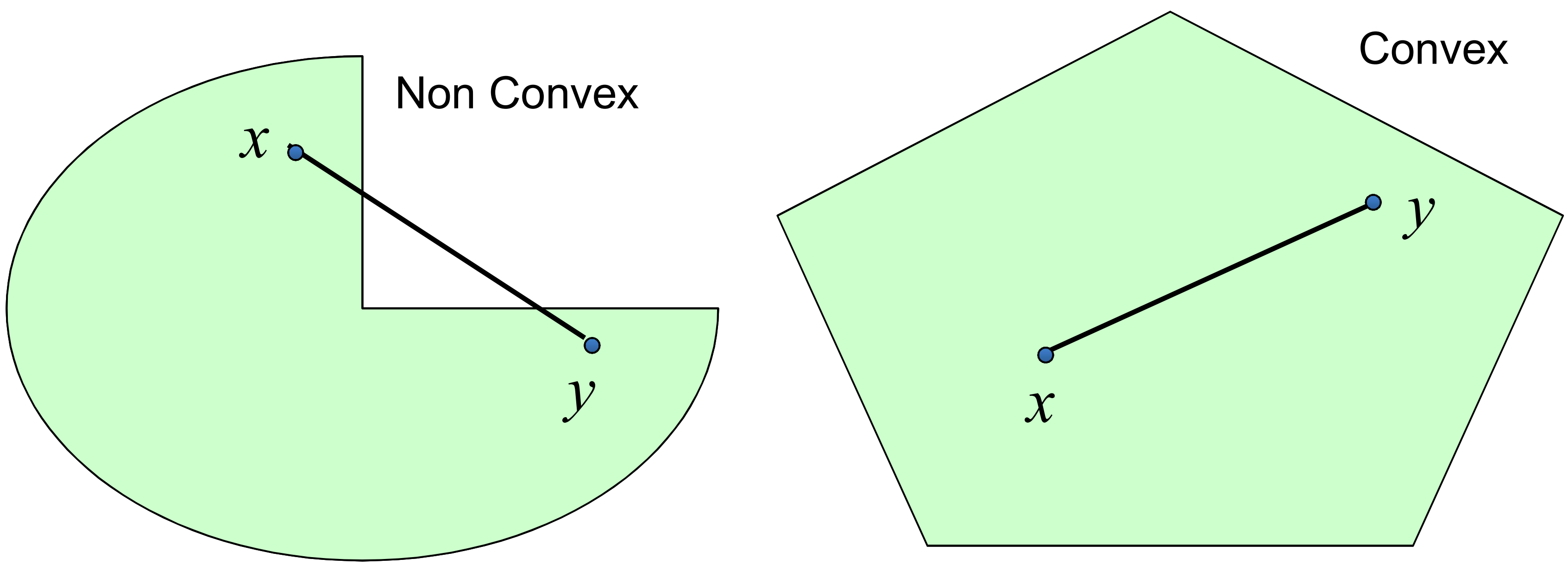
$$\min_x \quad f_0(x) \qquad (convex\ function)$$

$$s.t.\ \ f_i(x) \leq 0\ (convex\ sets)$$

$$h_j(x) = 0\ (Affine)$$

Convex Sets

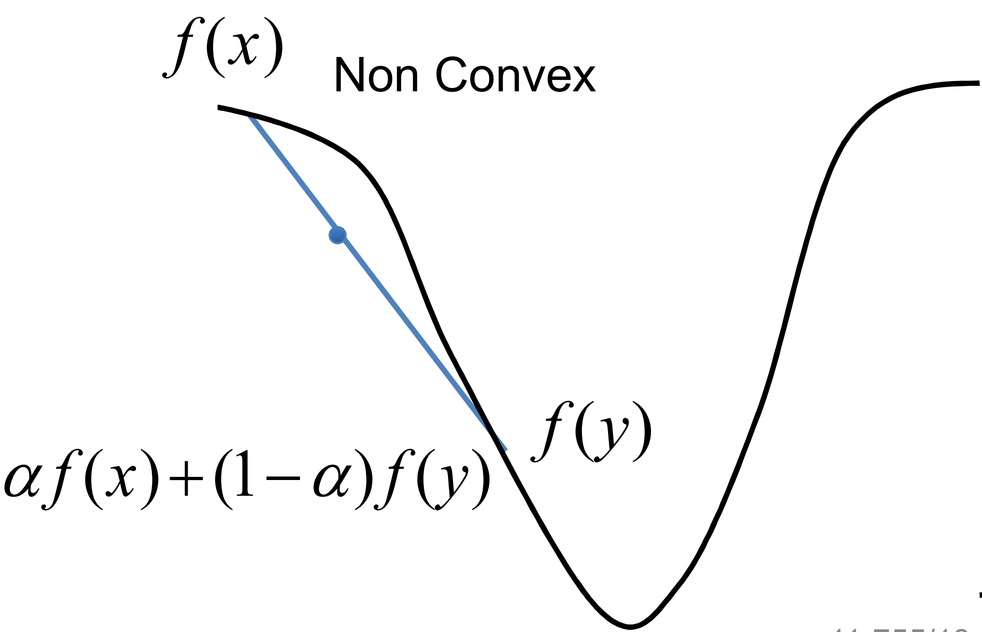
- a set $C \in \Re^n$ is convex, if for each $x, y \in C$ and $\alpha \in [0,1]$ then $\alpha x + (1-\alpha)y \in C$

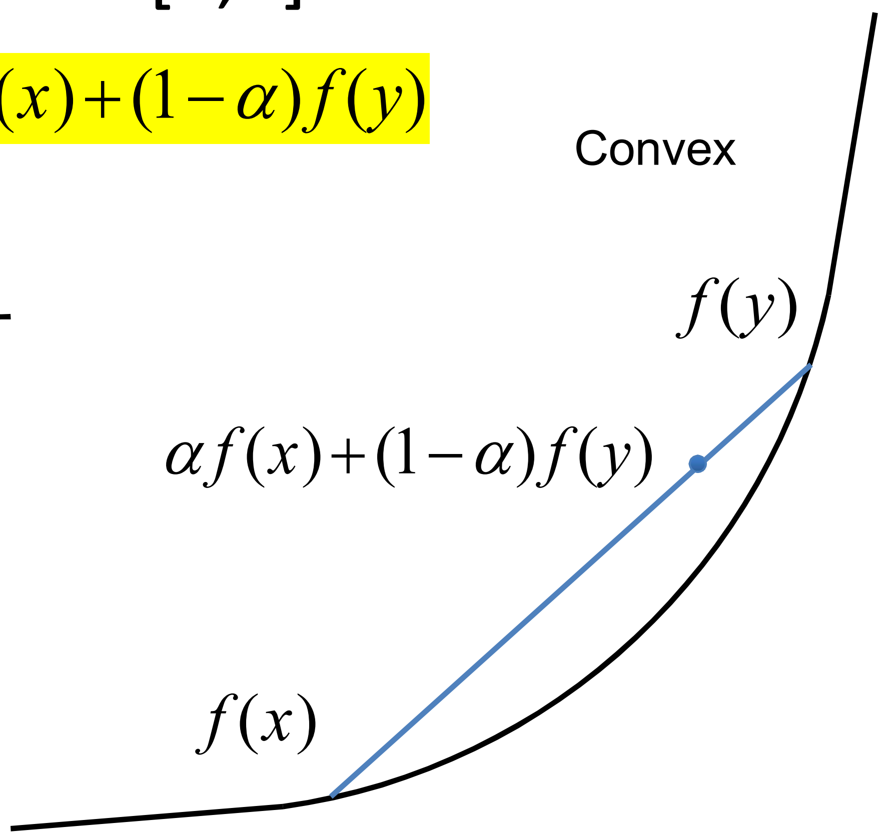


Convex functions

- A function $f: \mathcal{R}^N \longrightarrow \mathcal{R}$ is convex if for each $x, y \in domain(f)$ and $\alpha \in [0,1]$

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



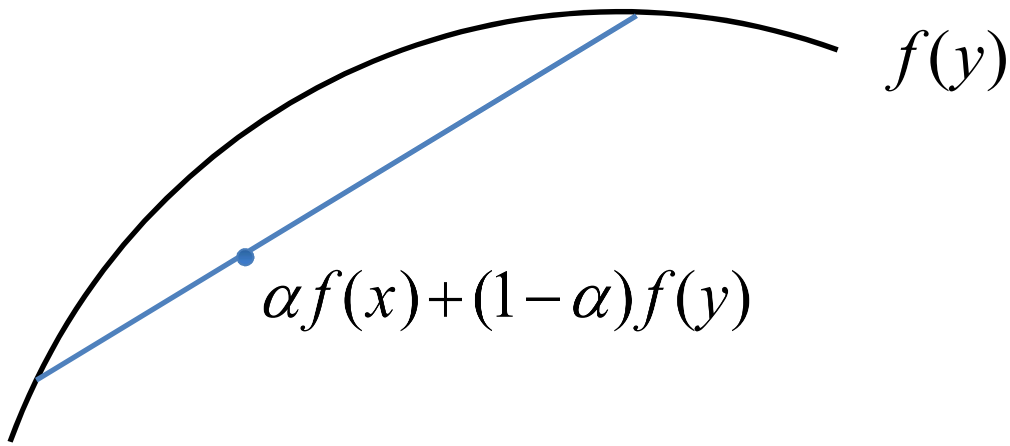


Concave functions

- A function $f: \mathcal{R}^N \longrightarrow \mathcal{R}$ is convex if for each $x, y \in domain(f)$ and $\alpha \in [0,1]$

$$f(\alpha x + (1-\alpha)y) \geq \alpha f(x) + (1-\alpha) f(y)$$

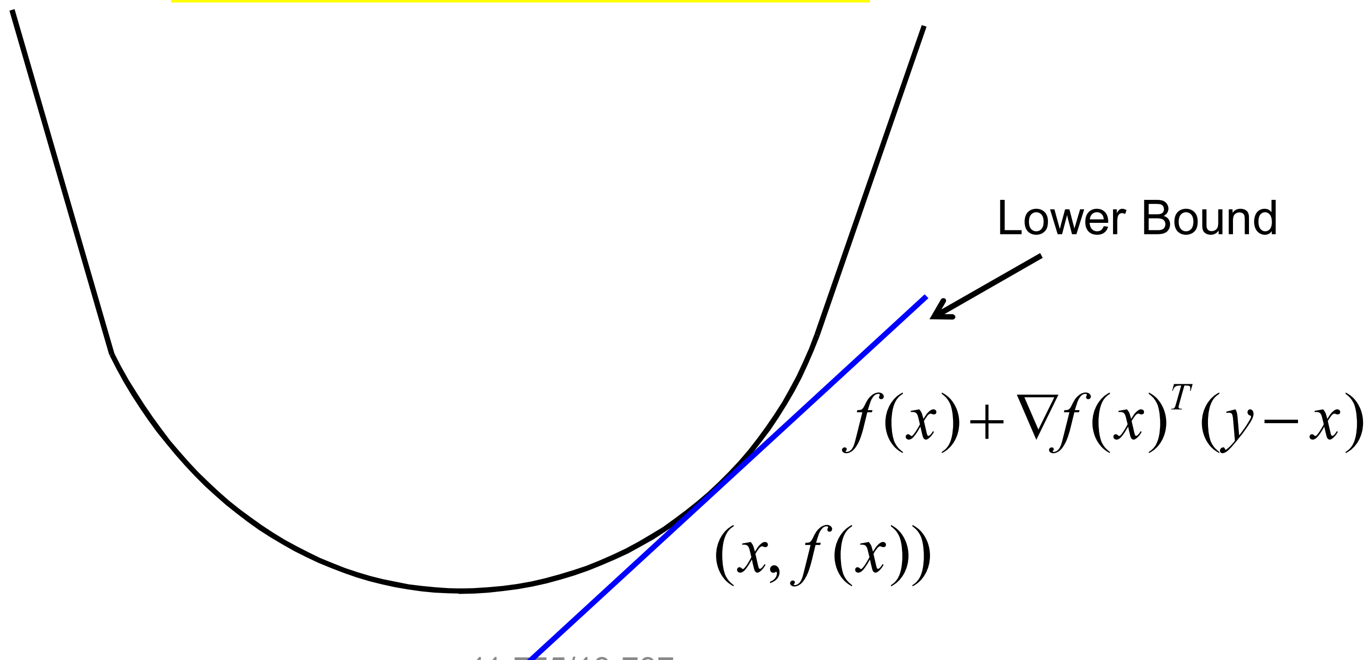
Concave



First order convexity conditions

- A differentiable function $f: \mathcal{R}^N \longrightarrow \mathcal{R}$ is convex if and only if for $x, y \in domain(f)$ the following condition is satisfied

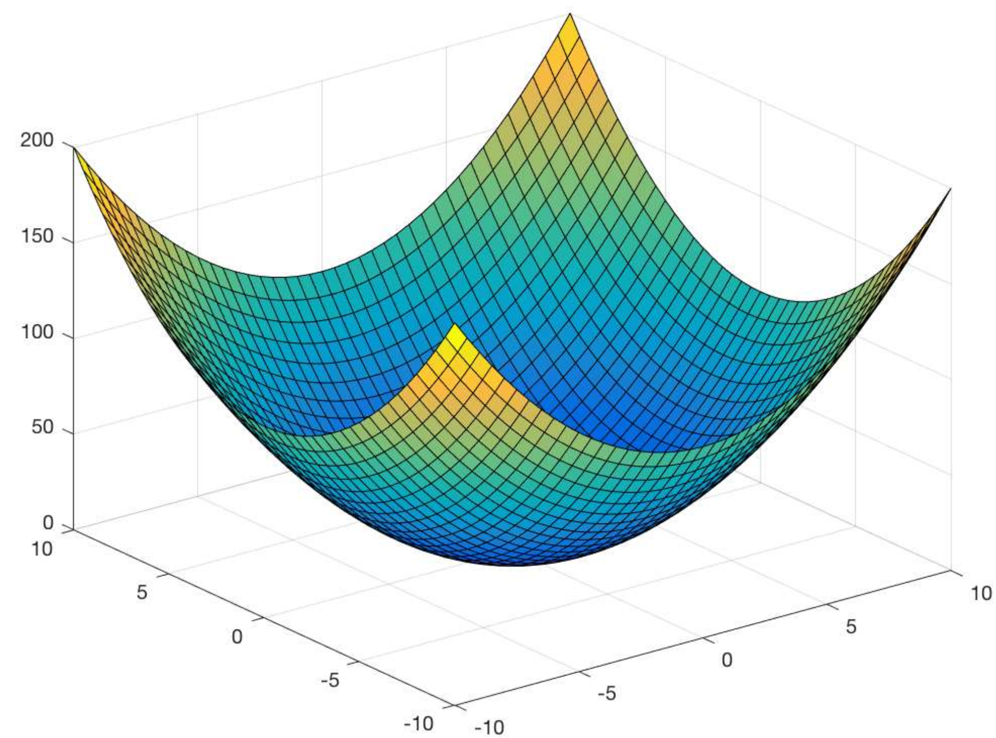
$$f(y) \geq f(x) + \nabla f(x)^t (y - x)$$



Second order convexity conditions

- A twice-differentiable function $f: \mathcal{R}^N \longrightarrow \mathcal{R}$ is convex if and only if for all $x, y \in domain(f)$ the Hessian is superior or equal to zero

$$\nabla^2 f(x) \geq 0$$



Properties of Convex Optimization

- For convex objectives over convex feasible sets, the optimum value is unique
  - There are no local minima/maxima that are not also the global minima/maxima
- Any gradient-based solution will find this optimum eventually
  - Primary problem: speed of convergence to this optimum

Lagrange multiplier duality

- Optimization problem with constraints

$$\min_x \quad f(x)$$
$$s.t.\ \ g_i(x) \le 0 \quad i = \{1, \ldots, k\}$$
$$h_j(x) = 0 \quad j = \{1, \ldots, l\}$$

- Lagrange multipliers $\lambda_i \geq 0, \nu \in \Re$

$$L(x, \lambda, \nu) = f(x) + \sum_{i=1}^{k} \lambda_i g_i(x) + \sum_{j=1}^{l} \nu_j h_j(x)$$

- The Dual function

$$\inf_x L(x, \lambda, \nu) = \inf_x \left\{ f(x) + \sum_{i=1}^{k} \lambda_i g_i(x) + \sum_{j=1}^{l} \nu_j h_j(x) \right\}$$

Lagrange multiplier duality

- The Original optimization problem

$$\min_{x}\left\{\sup_{\lambda\geq 0,\nu} L(x,\lambda,\nu)\right\}$$

- The Dual optimization

$$\max_{\lambda\geq 0,\nu}\left\{\inf_{x} L(x,\lambda,\nu)\right\}$$

- Property of the Dual for convex function

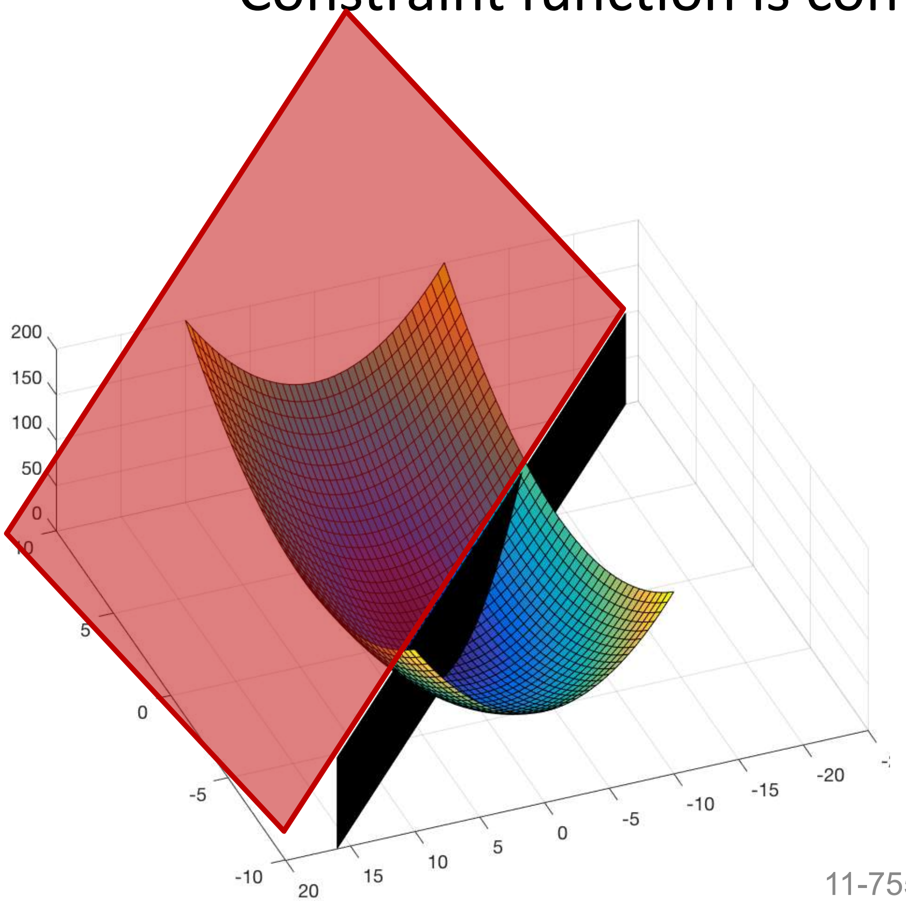
$$\sup_{\lambda\geq 0,\nu}\left\{\inf_{x} L(x,\lambda,\nu)\right\} = f(x^{*})$$

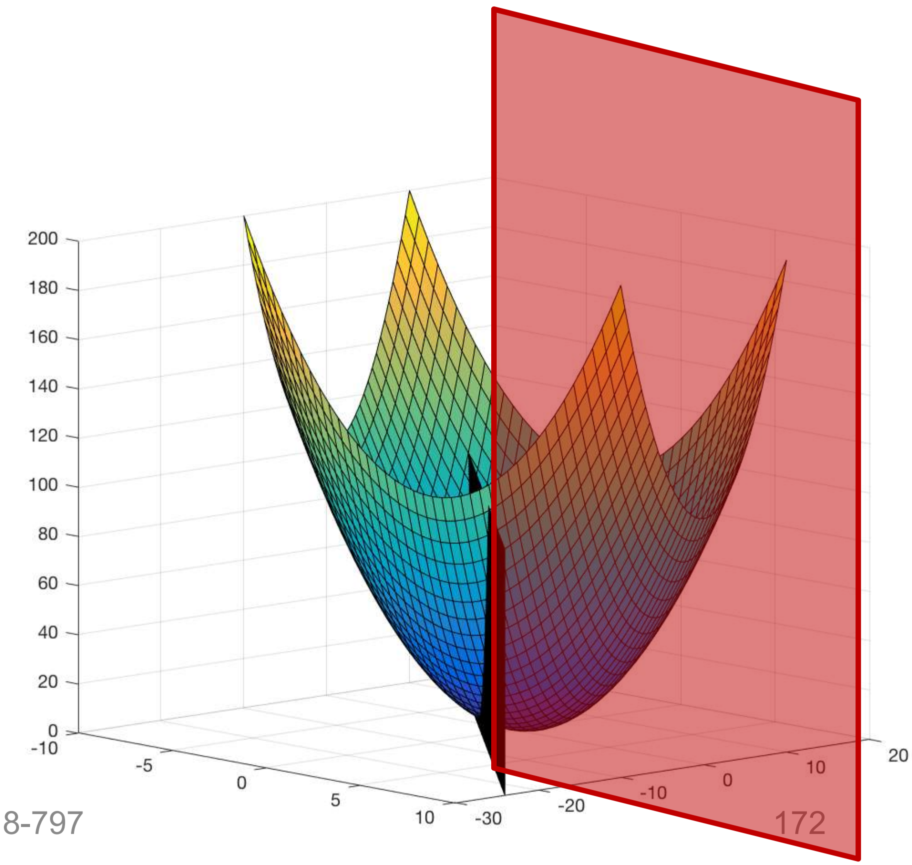
Lagrange multiplier duality

- Previous Example
  - $f(x,y)$ is convex
  - Constraint function is convex

$$\min_{x,y} \quad f(x,y) = x^2 + y^2$$

$$s.t. \quad 2x + y \le -4$$





Lagrange multiplier duality

- Primal system

$$\min_{x,y} \quad f(x,y) = x^2 + y^2$$

$$s.t. \quad 2x + y \le -4$$

- Lagrange Multiplier

$$L = x^2 + y^2 + \lambda(2x + y - 4)$$

$$\frac{\partial L}{\partial x} = 2x + 2\lambda = 0 \Rightarrow x = -\lambda$$

$$\frac{\partial L}{\partial y} = 2y + \lambda = 0 \Rightarrow y = -\frac{\lambda}{2}$$

- Dual system

$$\max_{\lambda} w(\lambda) = \frac{5}{4}\lambda^2 + 4\lambda$$

$$s.t. \quad \lambda \ge 0$$

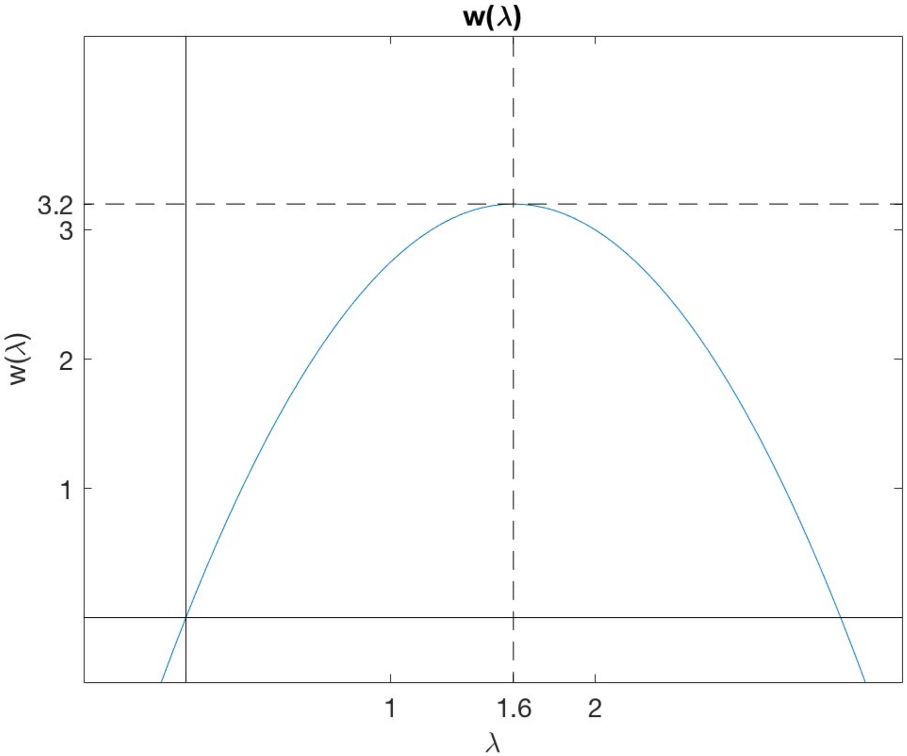
- Property

$$w(\lambda^*) = f(x^*, y^*)$$

Lagrange multiplier duality

- Dual system

$$\max_{\lambda} w(\lambda) = \frac{5}{4}\lambda^2 + 4\lambda$$
$$s.t.\ \ \lambda \geq 0$$



- Concave function
  - Convex function become concave function in dual problem

$$\frac{\partial w}{\partial x} = -\frac{5}{2}\lambda + 4 = 0 \Rightarrow \lambda^* = \frac{8}{5}$$

Lagrange multiplier duality

- Primal system

$$\min_{x,y} \quad f(x,y) = x^2 + y^2$$
$$s.t. \quad 2x + y \leq -4$$

- Dual system

$$\max_{\lambda} w(\lambda) = \frac{5}{4}\lambda^2 + 4\lambda$$
$$s.t. \ \lambda \geq 0$$

- Evaluating $w(\lambda^*) = f(x^*, y^*)$

$$x^* = -\frac{8}{5}, y^* = -\frac{4}{5}$$
$$f(x^*, y^*) = \left(-\frac{8}{5}\right)^2 + \left(-\frac{4}{5}\right)^2$$
$$f(x^*, y^*) = \frac{16}{5}$$

$$\lambda^* = \frac{8}{5}$$
$$w(\lambda^*) = -\frac{5}{4}\left(\frac{8}{5}\right)^2 + \frac{32}{5}$$
$$w(\lambda^*) = \frac{16}{5}$$