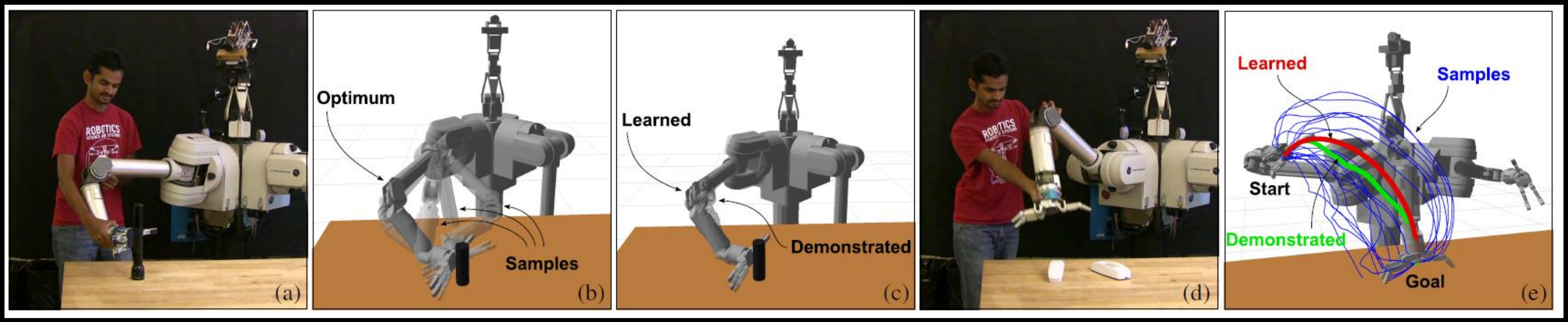
Convex Optimization I

Using material from Stephen Boyd

Why do we need optimization in robotics?

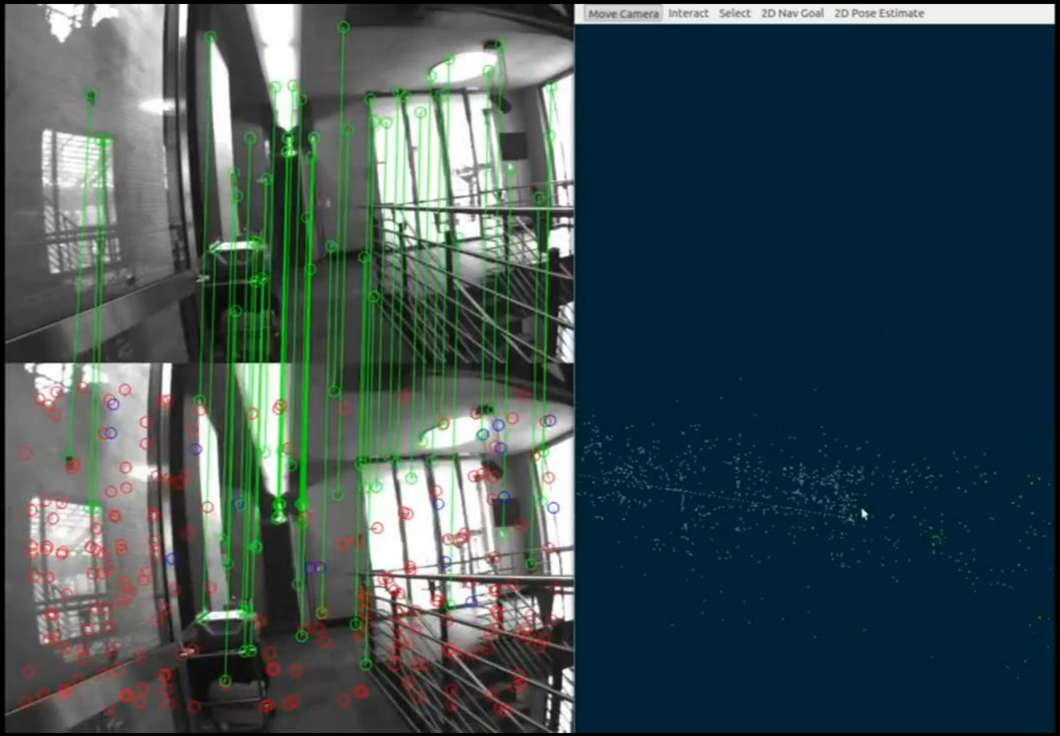
- Gives us a way to frame robotics problems mathematically
- VERY widely used
- Example: Inverse Optimal Control:



Learning Objective Functions for Manipulation
[Kalakrishnan et al., ICRA 2013]

Why do we need optimization in robotics?

- Example: Simultaneous Localization and Mapping (SLAM)



Keyframe-Based Visual-Inertial SLAM Using Nonlinear Optimization
[Leutenegger et al., RSS 2013]

Convex Optimization

- Convex optimization is a mature field with deep mathematical foundations
- It is so powerful that it's often worth it to
  - Work hard to reformulate your problem as convex
  - Approximate non-convex objective functions as convex
  - Use solution to approximation to start search for solution to the real problem
- It scales well with dimensionality
  - Convex optimization routinely solves problems with 1000s of variables
- Convex optimizers are fast (usually)

Outline

- Calculus Review
- Convex Sets
- Convex Functions
- Unconstrained Optimization

Set Notation

$$X = \{x | a^T x \leq b, x \in C, \quad a \in \boldsymbol{R}^n\}$$

X is the set of *xs* such that $a^T x \leq b$ is true for x in the set C where *a* is a vector in a Euclidian space of dimension *n*

Review: Functions

- Functions are defined as:

$$f : A \rightarrow B$$

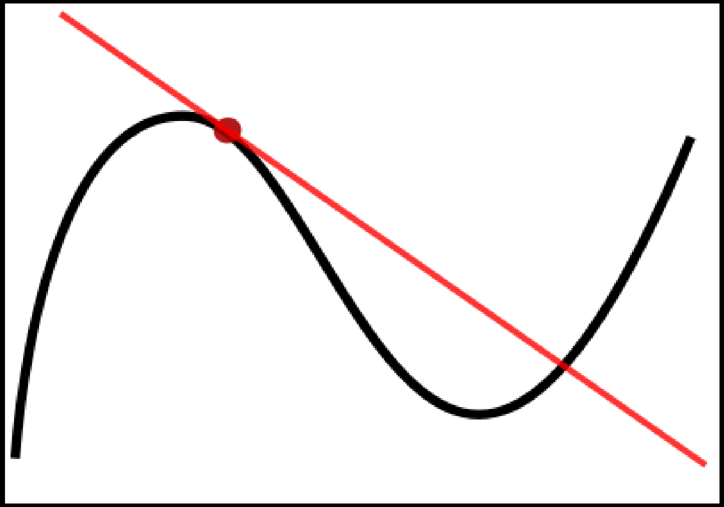
- "f maps elements in the set A to elements in the set B"
- The set A is the domain of f
- The set B is the range of f
- Example:

$$f : \mathbf{R}^n \rightarrow \mathbf{R}^m$$

"Function f maps n-dimensional vectors to some m-dimensional vectors"

Review: Derivatives

- Derivatives can get complicated!
- Keep this in mind: A derivative is a linear approximation of how a function changes a certain point



- The derivative of f(x) is the ratio between an infinitesimal change in an input variable x and the resulting change in the output f(x)

Review: Derivatives

- Recall the definition for a derivative $f: \mathbf{R} \to \mathbf{R}$

$$Df(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

- We can write a similar definition for $f: \mathbf{R}^n \to \mathbf{R}^m$

Review: Derivatives

- Suppose $f : \mathbf{R}^n \to \mathbf{R}^m$
- The function *f* is differentiable at *x* if there exists a matrix $\boxed{Df(x)} \in \mathbf{R}^{m \times n}$ that satisfies

This is a matrix

$$\lim_{z \in \mathbf{dom}\, f,\ z \neq x,\ z \to x} \frac{\|f(z) - f(x) - Df(x)(z - x)\|_2}{\|z - x\|_2} = 0$$

- *Df(x)* is called the derivative (or Jacobian) of the function
- *Df(x)* can be computed by computing partial derivatives

$$Df(x)_{ij} = \frac{\partial f_i(x)}{\partial x_j}, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, n$$

Review: Gradient

- When *f* is real-valued $(i.e.,\ f : \mathbf{R}^n \to \mathbf{R})$ the derivative *Df(x)* is a row vector (a 1 x n matrix)

Range must be 1-dimensional!

- The transpose of the derivative is the gradient:

$$\nabla f(x) = Df(x)^T$$

- Again, you can compute the gradient by taking partial derivatives:

$$\nabla f(x)_i = \frac{\partial f(x)}{\partial x_i}, \quad i = 1, \ldots, n.$$

Review: Second Derivative

- When f is real-valued $(i.e.,\ f : \mathbf{R}^n \to \mathbf{R})$ the **second** derivative is called the Hessian Matrix: $\nabla^2 f(x)$

$$\nabla^2 f(x)_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}, \qquad i = 1, \dots n, \quad j = 1, \dots, n,$$

- Recall that the second derivative is the derivative of the first derivative:

$$D\nabla f(x) = \nabla^2 f(x)$$

Questions

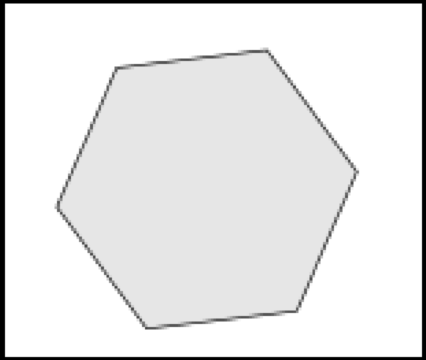
- Suppose we have a real-valued function $f : \mathrm{R}^n \rightarrow \mathrm{R}$

1. What are the dimensions of the gradient vector $\nabla f(x)$?
2. What are the dimensions of the Hessian matrix $\nabla^2 f(x)$?

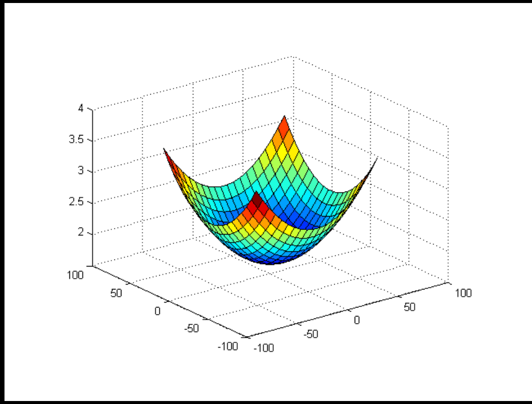
Convex Sets

Convex sets and functions

- Convexity is a restriction on shapes and functions
  - Convex optimization only works when everything is convex!
- We will cover definitions of convexity for shapes and functions
- You can use these to build convex sets/functions for the problems you care about



A convex set



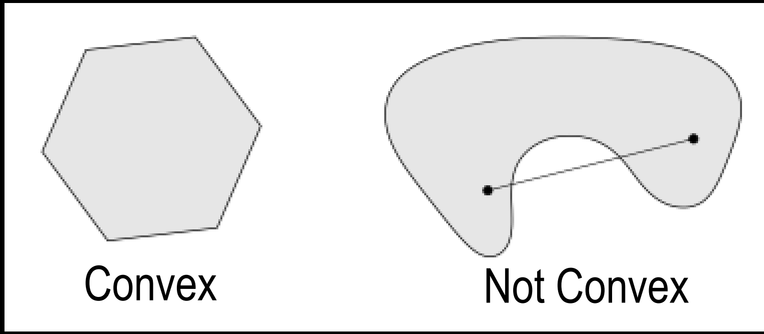
A convex function

Convex Sets

- Convex set: contains line segment between any two points in the set. *C* is a *convex set* if:

$$x_1, x_2 \in C, \quad 0 \leq \theta \leq 1 \quad \implies \quad \theta x_1 + (1-\theta) x_2 \in C$$

- Examples:



Important Types of Convex Sets: Hyperplane

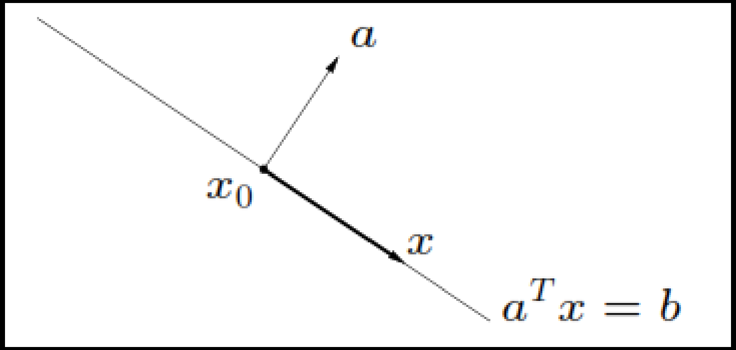
- Hyperplane: A set of points that have a constant inner product with vector a

$$\{x \mid a^T x = b\} \ (a \neq 0)$$

same as $a \cdot x$

- Another way to define it:

$$\{x \mid a^T(x - x_0) = 0\}$$



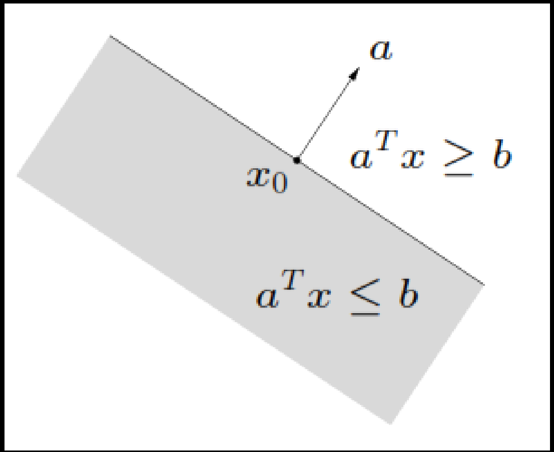
Important Types of Convex Sets: Halfspace

- Halfspace: A hyperplane with an inequality

$$\{x \mid a^T x \leq b\} \;\; (a \neq 0)$$

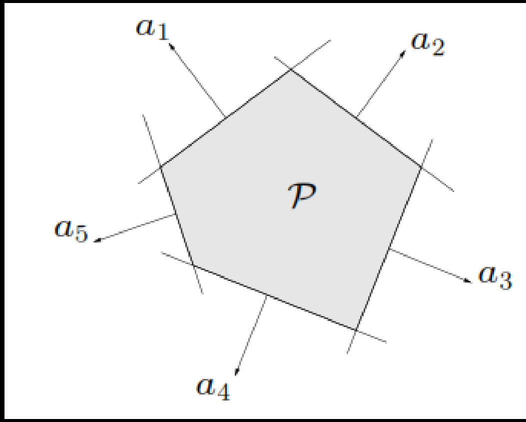
- Another way to define it :

$$\{x \mid a^T(x - x_0) \leq 0\}$$



Important Types of Convex Sets: Polyhedron

- Polyhedron: The intersection of a finite number of halfspaces and hyperplanes



- Another way to define it: The set of solutions to a set of linear inequalities and equalities:

$$Ax \leq b$$

$$Cx = d$$

Important Convexity-Preserving Operations on Sets

- Intersection preserves convexity

  If S_1 and S_2 are convex, then $S_1 \cap S_2$ is convex

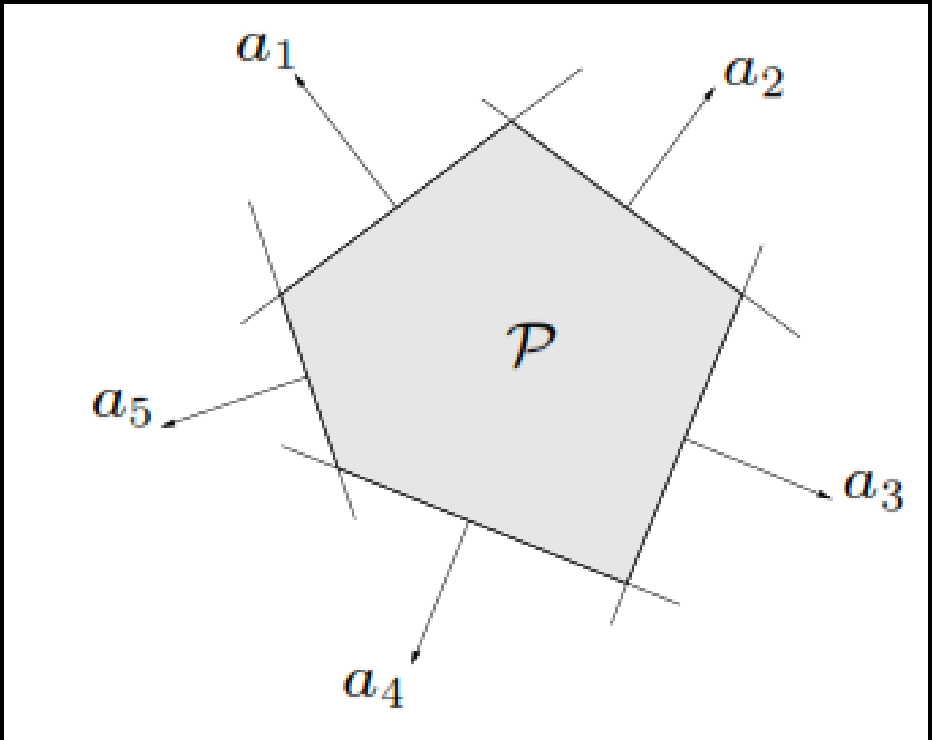
- It follows that the intersection of any number of convex sets is convex

- Affine functions preserve convexity

$$f : \mathbf{R}^n \to \mathbf{R}^m \qquad f(x) = Ax + b \text{ with } A \in \mathbf{R}^{m \times n},\ b \in \mathbf{R}^m$$

- Examples of affine functions
  - Scaling
  - Translation
  - Projection

How do we know a polyhedron is always convex?



Convex Functions

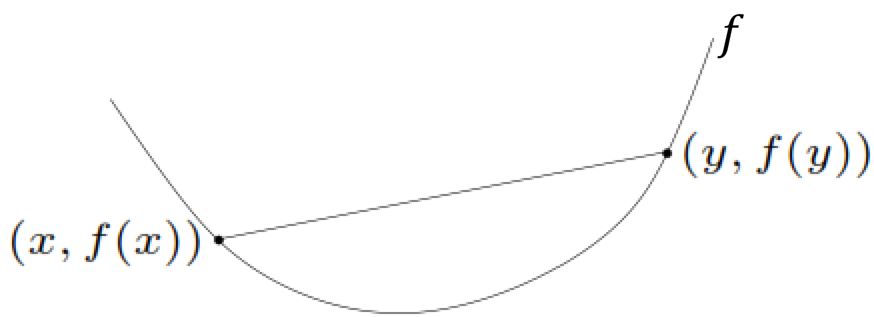
Convex Functions

The domain of the function

$f : \mathbf{R}^n \to \mathbf{R}$ is convex if $\mathbf{dom}\, f$ is a convex set and

$$f(\theta x + (1-\theta)y) \le \theta f(x) + (1-\theta) f(y)$$

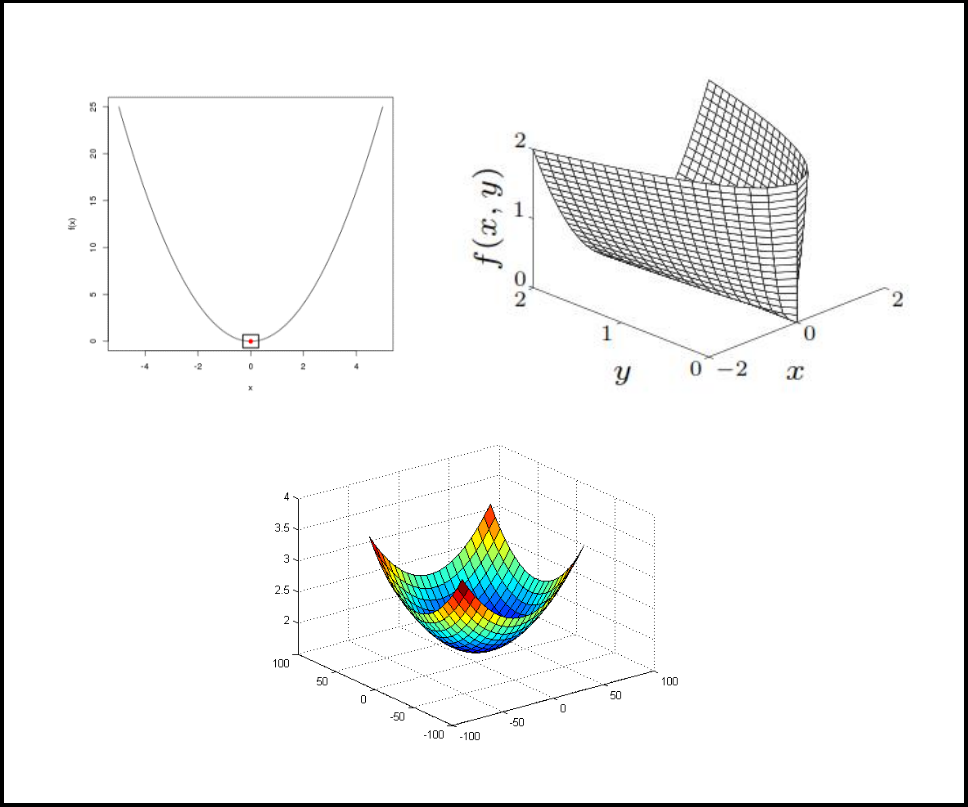
for all $x, y \in \mathbf{dom}\, f$, $0 \le \theta \le 1$



- I.e. the line segment between $(x, f(x))$ and $(y, f(y))$ lies above the graph of f

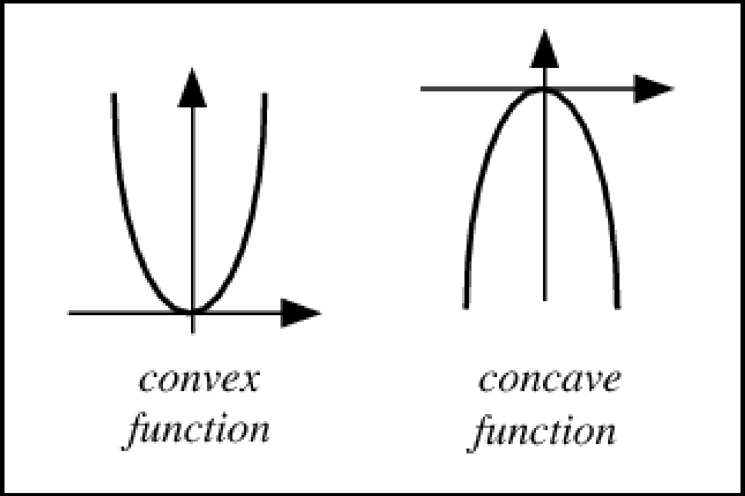
Advantage of convex functions

- Convex functions have only one local minimum!
  - That means local methods can find the global optimum!



Concave Functions

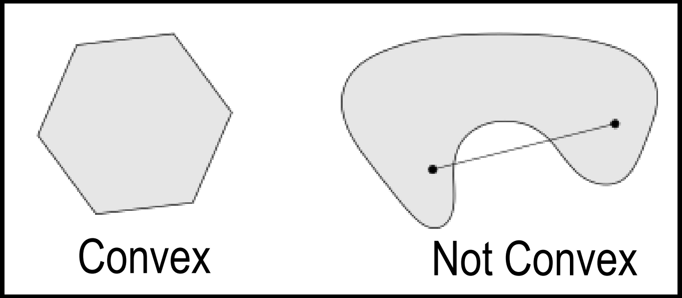
- Concave functions are convex functions that are “upside down”



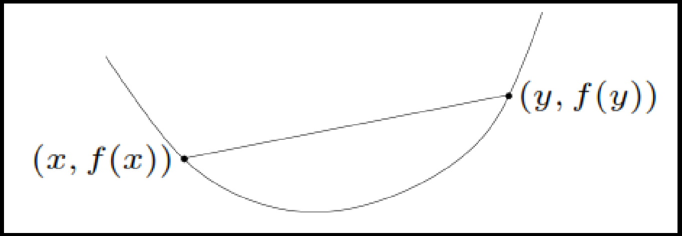
- If $f(x)$ is convex, $-f(x)$ is concave.
- Some $f(x)$ are **both** concave and convex
  - Example?

Summary

- Convex sets are sets where a line segment between any two points is part of the set



- Convex functions are functions where the line segment between any two points is above the graph of the function

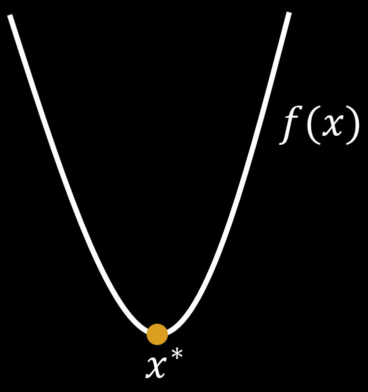


Break

Unconstrained Optimization

Unconstrained Minimization Problem

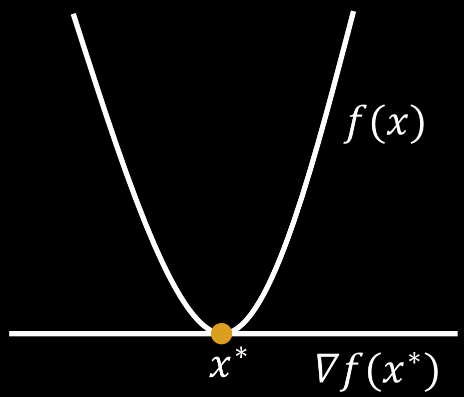
$$\underset{x}{\text{minimize}}\, f(x)$$



- Assumptions
  - *f* is convex
  - No constraints on x
- There are MANY methods for this kind of problem
  - Some are general, some exploit a specific structure of *f*
- Usually decide what to use based on
  - Differentiability of *f*
  - How you compute $\nabla f(x)$
- We will cover several important methods common in robotics

Review: Minimizing a simple function

- For a simple function, e.g. $f(x) = x^2 - 4x$, we can use calculus to find the minimum
- For an optimal point x^*, $\nabla f(x^*) = 0$
- So,
  1. Take the derivative of $f(x)$
  2. Set it equal to 0
  3. Solve for x



- Example for $f(x) = x^2 - 4x$
  1. $\nabla f(x) = 2x - 4$
  2. $2x - 4 = 0$
  3. $x = 2$ is the minimum

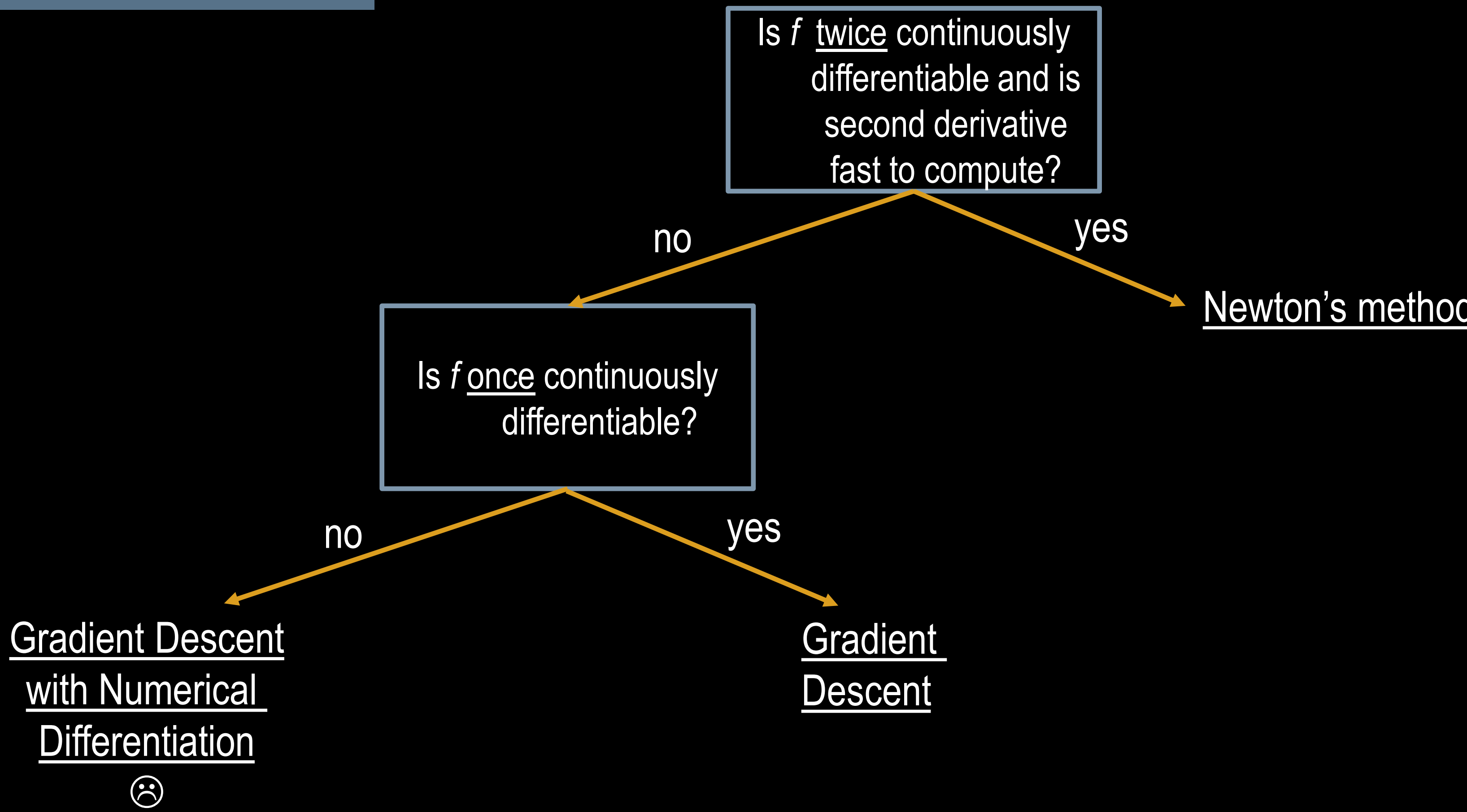
What about a more complicated function?

- $f(x) = e^{0.5x+0.9} + e^{-0.5x^2-0.4} + 4x$
  1. $\nabla f(x) = 0.5e^{0.5x+0.9} - xe^{-0.5x^2-0.4} + 4$
  2. $0.5e^{0.5x+0.9} - xe^{-0.5x^2-0.4} + 4 = 0$
  3. $x = ???$

Problem: No way to solve arbitrary equations using algebra!

$$\underset{x}{\text{minimize}}\, f(x)$$

- assume f is convex
- assume x is unconstrained



*Many more methods not covered!

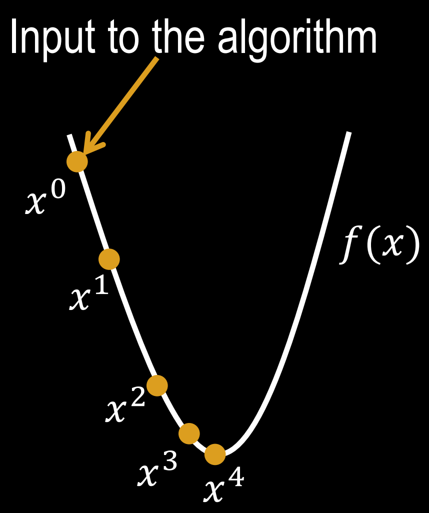
Descent Methods

Unconstrained Minimization Methods

- Let p^* be the optimal value of $f(x)$
- Let x^* be a value of x that produces p^*
  - $p^* = f(x^*)$
- These methods produce a sequence of points:

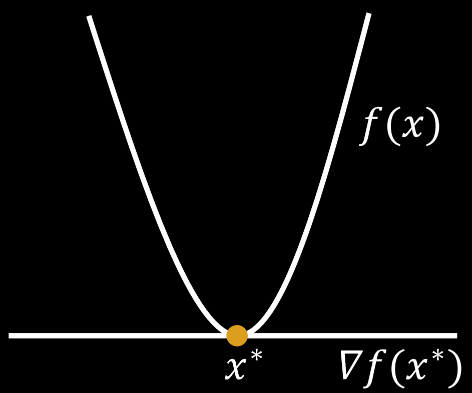
$$x^{(k)} \in \mathbf{dom}\, f,\ k = 0, 1, \ldots$$

$$f(x^{(k)}) \to p^\star$$



- Can interpret as iteratively finding an x* that solves optimality condition:

$$\nabla f(x^\star) = 0$$



Descent methods

- We will cover two types of descent methods:
  - Gradient descent
  - Newton's method
    - Advantage: affine invariant
- Descent methods generate points with this property:

$$x^{(k+1)} = x^{(k)} + t^{(k)} \Delta x^{(k)} \quad \text{with } f(x^{(k+1)}) < f(x^{(k)})$$

Other notation:

$$x := x + t\Delta x$$

- Δx is the step, or search direction
- t is the step size, or step length

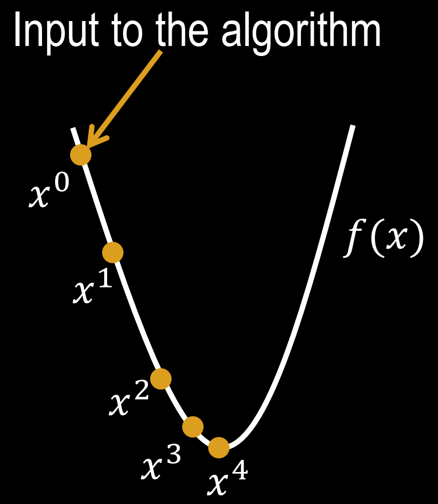
General descent algorithm

Many ways to do these

given a starting point $x \in \mathbf{dom}\, f$.
repeat

1. Determine a descent direction Δx.
2. *Line search.* Choose a step size $t > 0$.
3. *Update.* $x := x + t\Delta x$.

until stopping criterion is satisfied.



Gradient Descent

- Most common optimization algorithm
- Easy to implement, but may be slow to converge
- Descent direction:

$$\Delta x = -\nabla f(x)$$

- Termination condition:

$$\|\nabla f(x)\|_2 \leq \epsilon$$

e.g. $\epsilon = 0.001$

Gradient Descent: Step size

- Ideally, we would use *exact line search* to determine step size *t*:

$$t = \mathrm{argmin}_{t>0}\, f(x + t\Delta x)$$

- But this is slow to compute in general, so often use backtracking line search:

given a descent direction Δx for f at $x \in \textbf{dom}\, f$, $\alpha \in (0, 0.5)$, $\beta \in (0, 1)$.
$t := 1$.
while $f(x + t\Delta x) > f(x) + \alpha t \nabla f(x)^T \Delta x, \quad t := \beta t.$

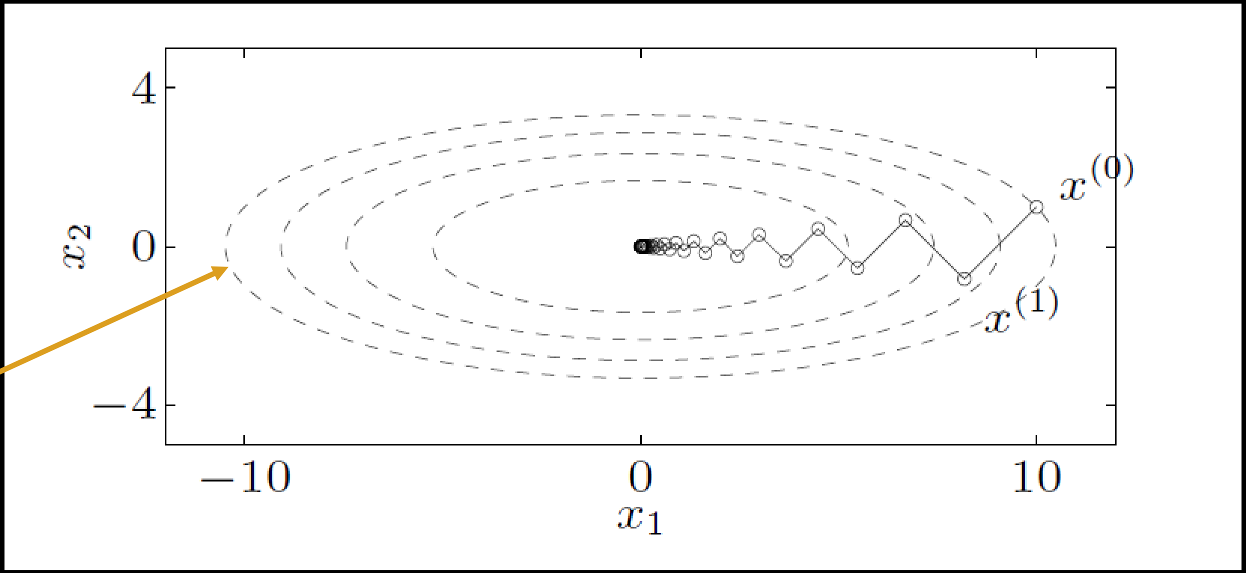
- I.e. decrease magnitude of step until you meet the stopping condition

Gradient Descent Example

- Find the minimum of this function:

$$f(x_1, x_2) = 0.5(x_1^2 + 10x_2^2)$$

- Starting at $x^{(0)} = (10,1)$, using exact line search

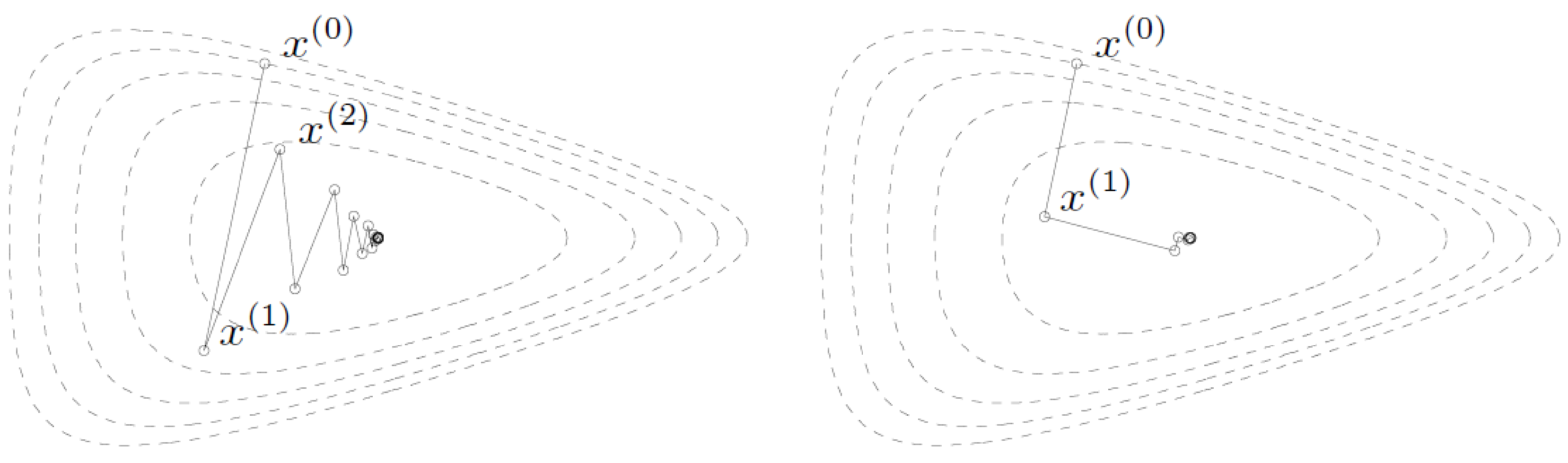


isocontours:
levels of constant
f(x) value

Gradient Descent Example

- Find the minimum of this function:

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



backtracking line search

exact line search

Problems with Gradient Descent

- Sensitive to the condition number of the Hessian
  - High condition number means very slow convergence
- Sensitive to the coordinates you use (not affine invariant)
  - Apply a linear transform to x and you may get different results!

- Newton's method overcomes these problems by using the Hessian of the function
  - For a price (the Hessian can be expensive to compute)

Newton's method: Descent direction

- Determine a descent direction:

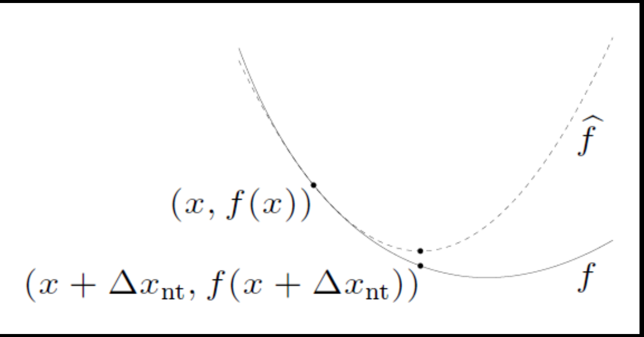
$$\Delta x_{\mathrm{nt}} = -\nabla^2 f(x)^{-1} \nabla f(x)$$

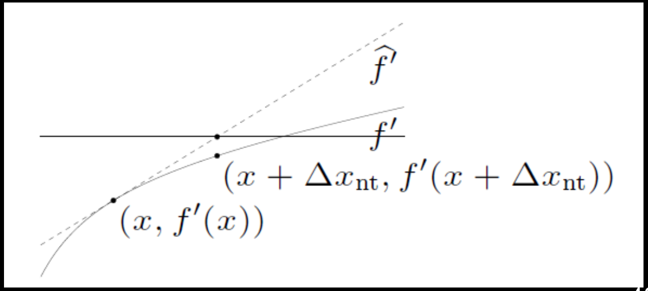
- Why?
  - Let's approximate $f(x)$ with a quadratic function (remember Taylor series):

$$\widehat{f}(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v$$

  - $x + \Delta x_{nt}$ solves the linearized optimality condition:

$$\nabla f(x+v) \approx \nabla \widehat{f}(x+v) = \nabla f(x) + \nabla^2 f(x) v = 0$$





Newton's method: Stopping criterion

- The Newton decrement $\lambda(x)$ leads to the stopping criterion:

$$\lambda(x) = \left(\nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x)\right)^{1/2}$$

- $\lambda(x)$ is an estimate of the distance between f(x) and p*
- $\lambda(x)^2$ is the directional derivative in the direction of the Newton step:

$$\nabla f(x)^T \Delta x_{\text{nt}} = -\lambda(x)^2$$

- If the directional derivative is very close to 0, f(x) is not changing much in this direction
  - I.e. you're very close to the optimum
  - So, when $\frac{\lambda(x)^2}{2}$ is below some small tolerance ϵ, stop

Newton's method

given a starting point $x \in \mathbf{dom}\, f$, tolerance $\epsilon > 0$.

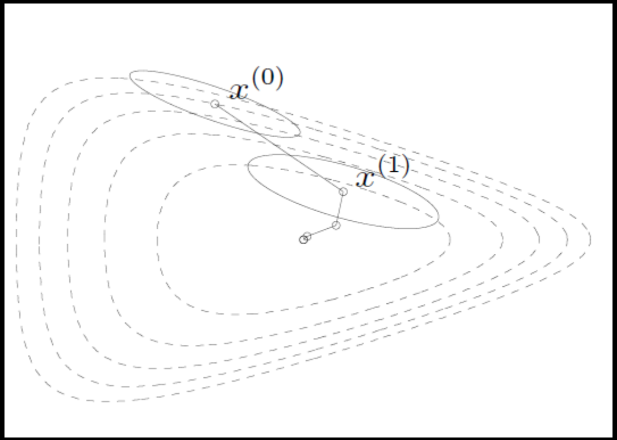
repeat

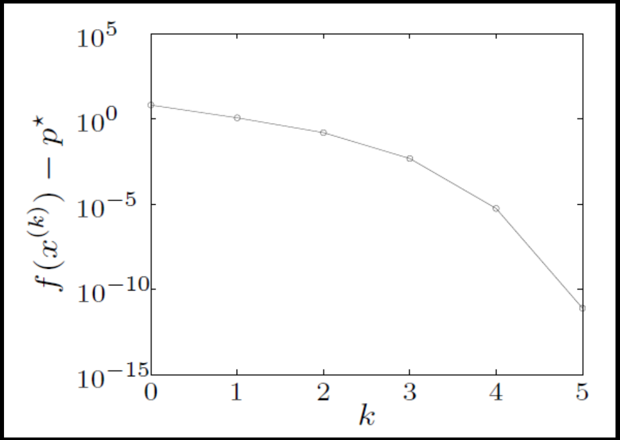
1. *Compute the Newton step and decrement.*
   $$\Delta x_{\mathrm{nt}} := -\nabla^2 f(x)^{-1} \nabla f(x); \quad \lambda^2 := \nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x).$$
2. *Stopping criterion.* **quit** if $\lambda^2/2 \leq \epsilon$.
3. *Line search.* Choose step size t by backtracking line search.
4. *Update.* $x := x + t\Delta x_{\mathrm{nt}}$.

Newton's method example

- Find the optimum of this function:

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



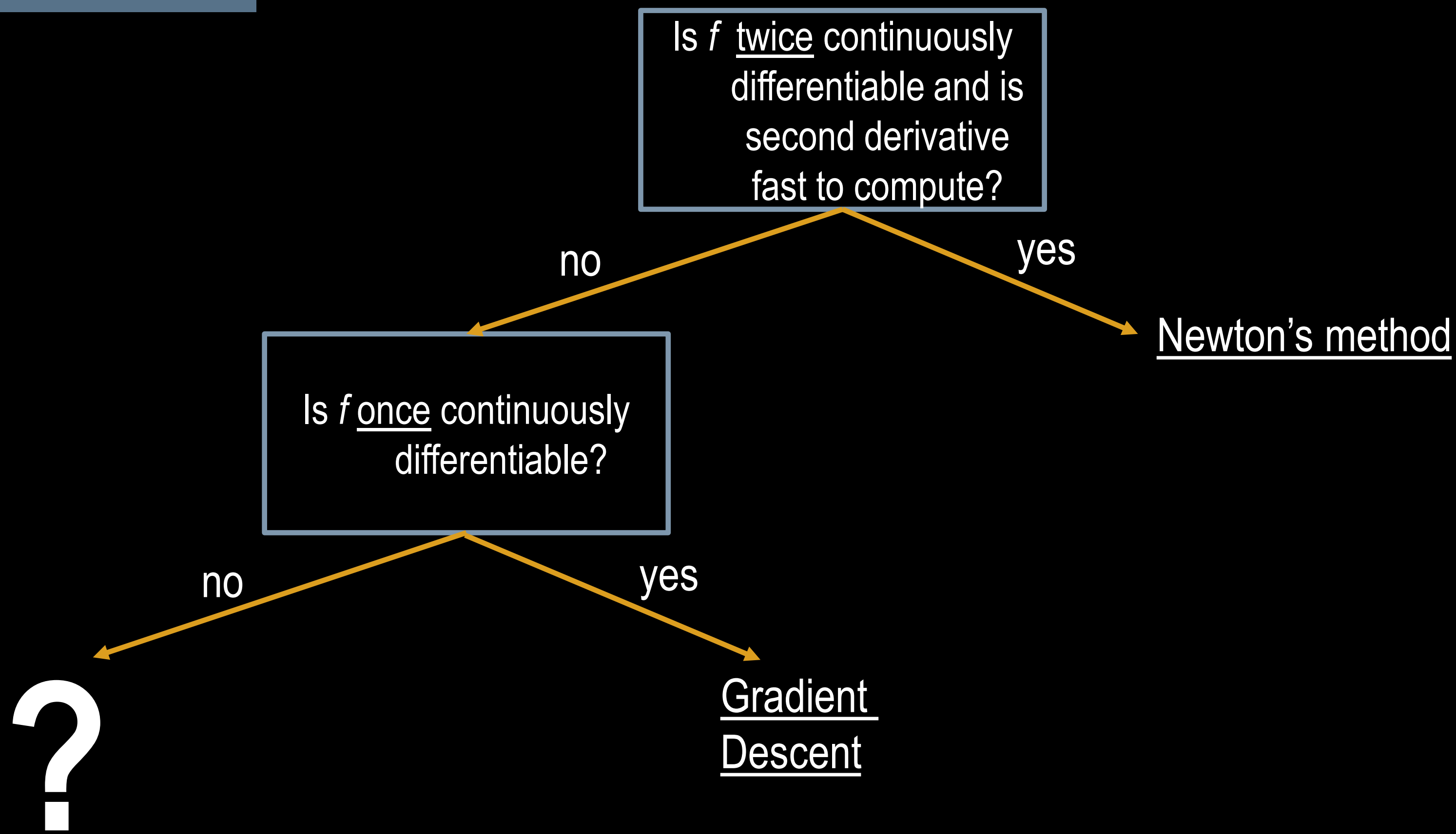


- Backtracking line search parameters:

$$\alpha = 0.1, \ \beta = 0.7$$

$$\underset{x}{\text{minimize}}\, f(x)$$

- assume f is convex
- assume x is unconstrained



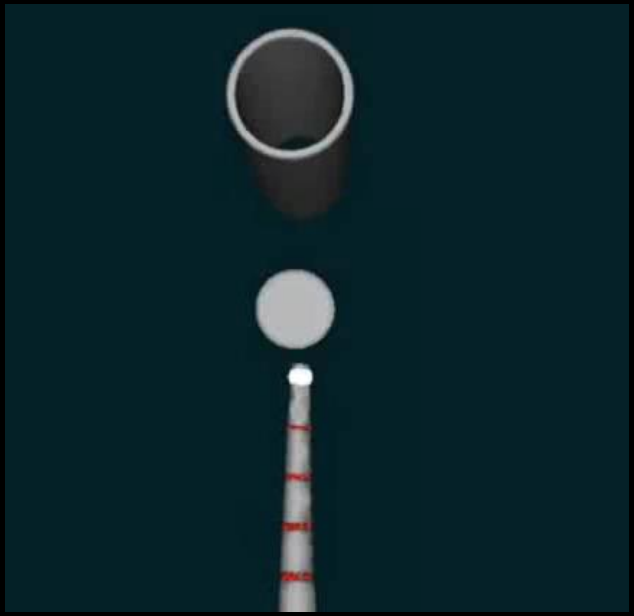
*Many more methods not covered!

Numerical Differentiation

What about functions that you don't know analytically?

- So far *f(x)* is always represented analytically
- What if *f(x)* is this:

x is actuator forces/torques



$f(x)$ outputs distance of tip to goal

"Optimization-based inverse model of Soft Robots with Contact Handling"
Eulalie Coevoet, Adrien Escande, Christian Duriez

Numerical Differentiation

- Need a way to differentiate when the function is not represented analytically
- Assume we can evaluate the function at any x
  - E.g. by running some code like a simulation
- Recall standard derivative definition for $f: \mathbf{R} \to \mathbf{R}$

$$Df(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

- Key idea: Evaluate function at two points per dimension and estimate the derivative

Numerical differentiation for univariate functions

1. Pick a small scalar h
2. Use a Finite Difference method. Two common ones:
   a) Newton's Difference Quotient

$$Df(x) \approx \frac{f(x+h) - f(x)}{h}$$

   b) Symmetric Difference Quotient

$$Df(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

- There are other numerical methods which can give better estimates but use more function evaluations

Numerical differentiation for multidimensional functions

- For $f: \mathbf{R}^n \to \mathbf{R}^m$ we do the same thing to compute the Jacobian
- Recall:

$$Df(x)_{ij} = \frac{\partial f_i(x)}{\partial x_j}, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, n$$

index *j*

- Let $\delta(j, h) = [0, \ldots, h, \ldots 0]^T$

$$Df(x)_{ij} \approx \frac{f(x + \delta(j,h))_i - f(x)_i}{h}$$

- Similar process for Symmetric Difference Quotient
- Thus we can use numerical differentiation to compute the gradient for <u>gradient descent</u>

Limitations

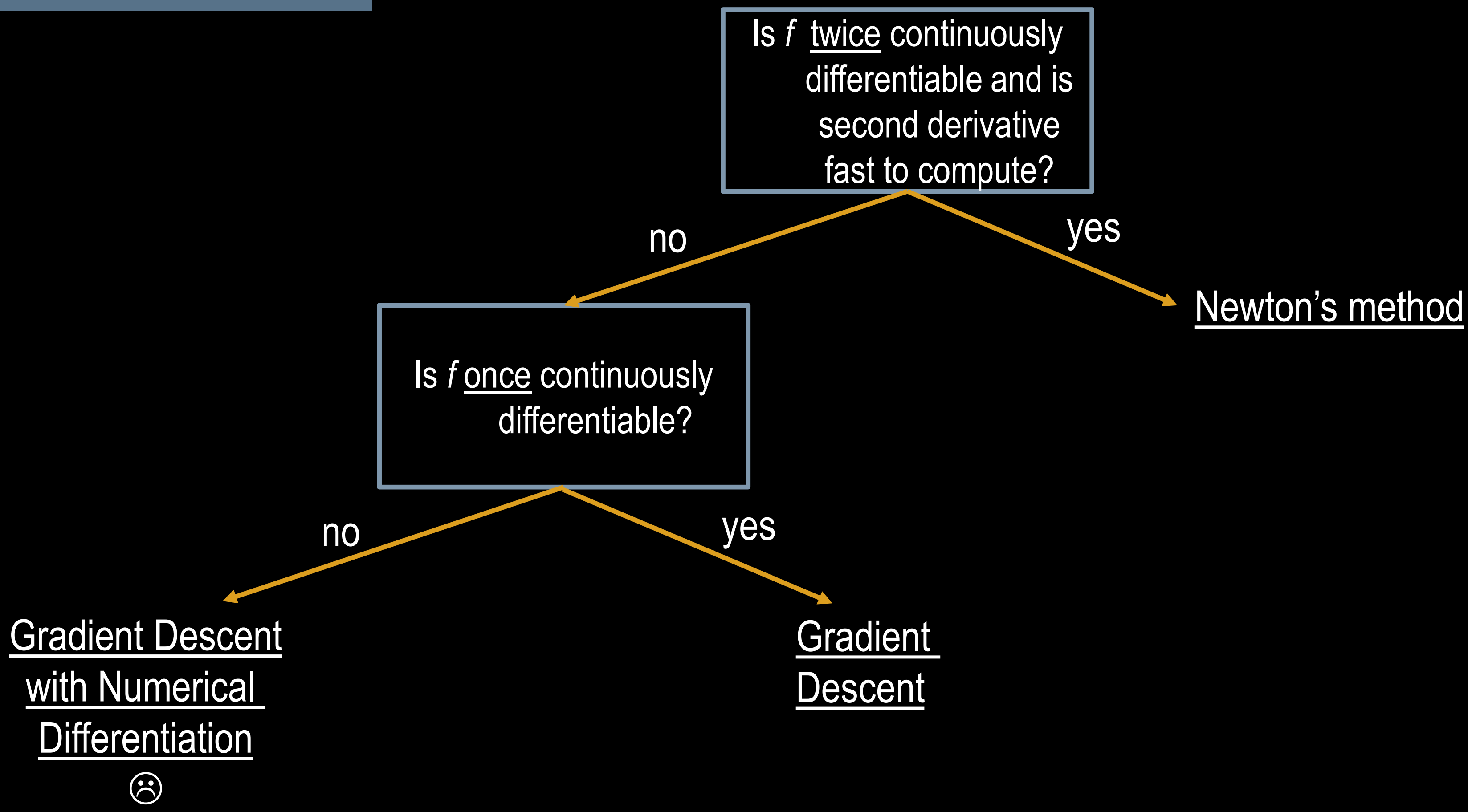
- Choosing h well is difficult in general (it is function-dependent)
  - Many use a fixed h for simplicity

- Numerical methods can be very sensitive to the choice of h

- There can be errors due to machine precision and floating point arithmetic

$$\underset{x}{\text{minimize}}\ f(x)$$

- assume f is convex
- assume x is unconstrained



*Many more methods not covered!

Homework

- Reading from Optimization Book
  - Ch. 4.1-4.1.2, 4.3-4.3.1 (skip examples), 4.4-4.4.1 (only read first example in 4.4.1)
- Homework 4 due tonight
- Homework 5 posted tonight