

ROB-GY 6323  
reinforcement learning and optimal  
control for robotics

Lecture 4  
Nonlinear Optimal Control

# Course material

All necessary material will be posted on **Brightspace**

Code will be posted on the **Github** site of the class

<https://github.com/righetti/optlearningcontrol>

**Discussions/Forum with Slack**

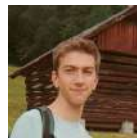
Course Assistant

Armand Jordana

[aj2988@nyu.edu](mailto:aj2988@nyu.edu)

Office hours Monday 1pm to 2pm

Rogers Hall 515



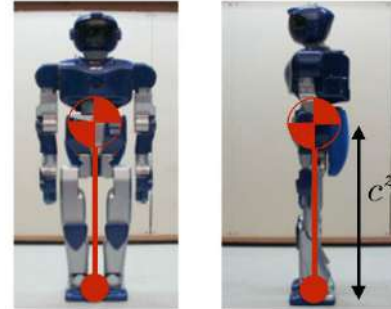
# Reminder

- Homework 1 is due on Sep 28, 2024 11:59 PM
- Homework 2 out soon

# Recap on QP

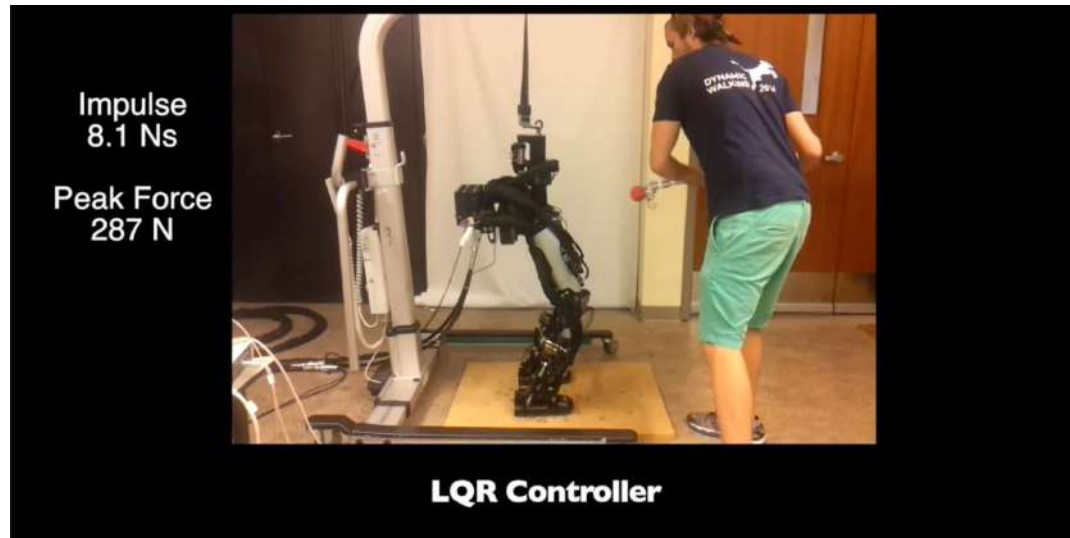
$$\begin{aligned} & \min_x \frac{1}{2} x^T P x + q^T x \\ \text{subject to } & Ax = b \\ & Gx \leq h \end{aligned}$$

$$\ddot{\mathbf{c}}^{x,y} = \frac{g}{c^z} (\mathbf{c}^{x,y} - \underbrace{\mathbf{p}^{x,y}}_{\text{CoP}})$$

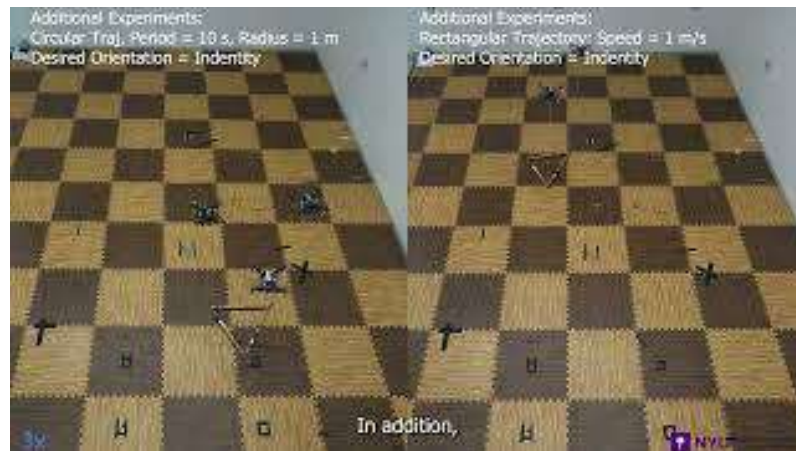


LIPM

# Recap on QP



# We need nonlinear formulations



# Structure of an optimal control problem

$$\min_{x_1, \dots, x_T, u_0, \dots, u_{T-1}} \sum_{t=0}^{T-1} \ell_t(x_t, u_t) + \ell_T(x_T)$$

subject to  $x_{t+1} = f(x_t, u_t)$

$$h_t(x_t, u_t) \leq 0$$

$$h_T(x_T) \leq 0$$

Describe the cost function





# Unconstrained optimization

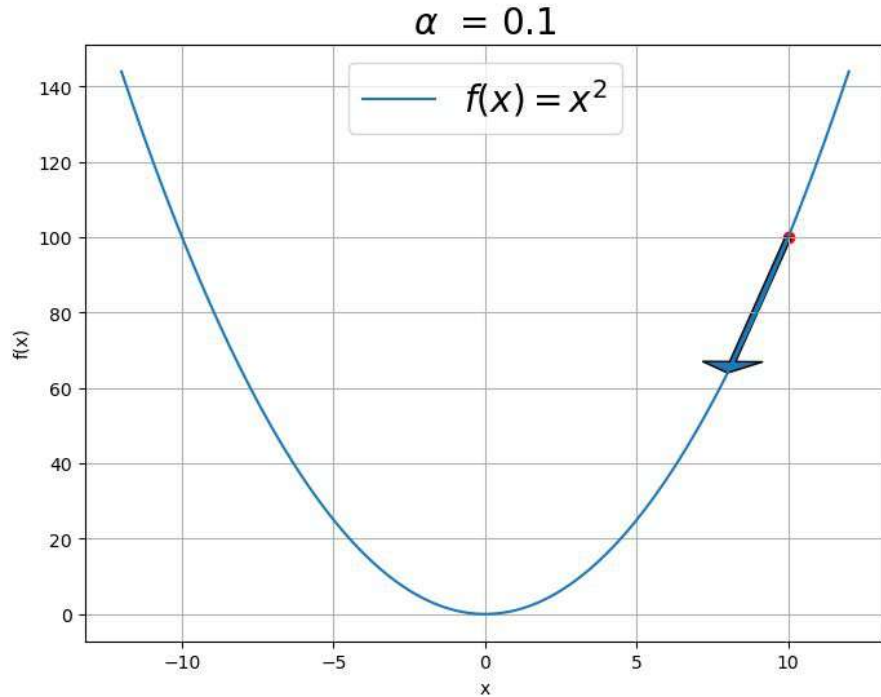
$$\min_x f(x)$$

- Cannot rely on closed form solution.

e.g.  $f(x) = \exp(\cos(x))(\sin(\cos(x)) + 1)^3$

- Need a (local) iterative method.

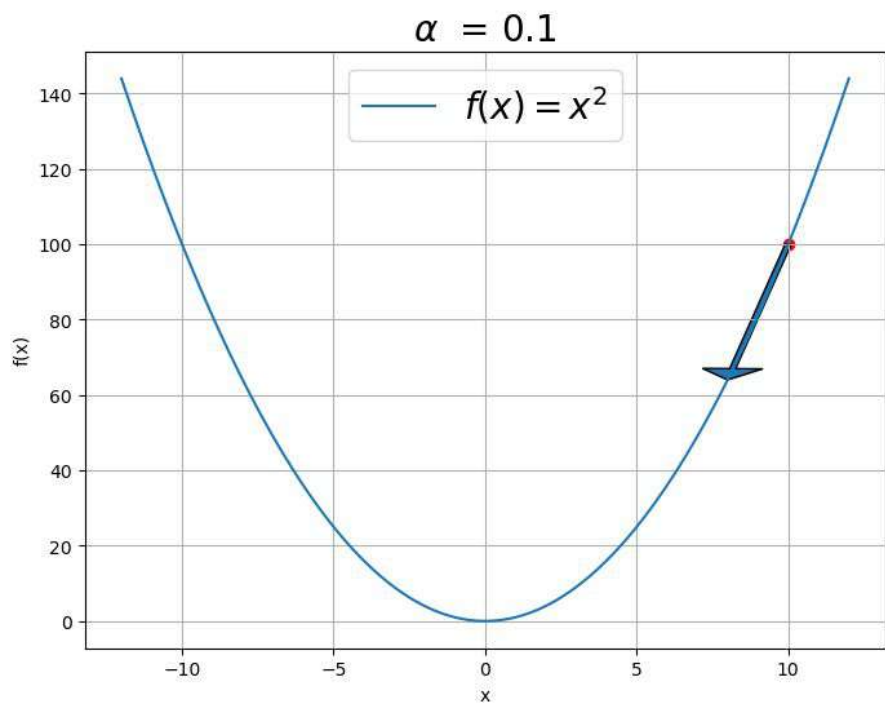
# What is a descent direction?



$$x \leftarrow x + p$$

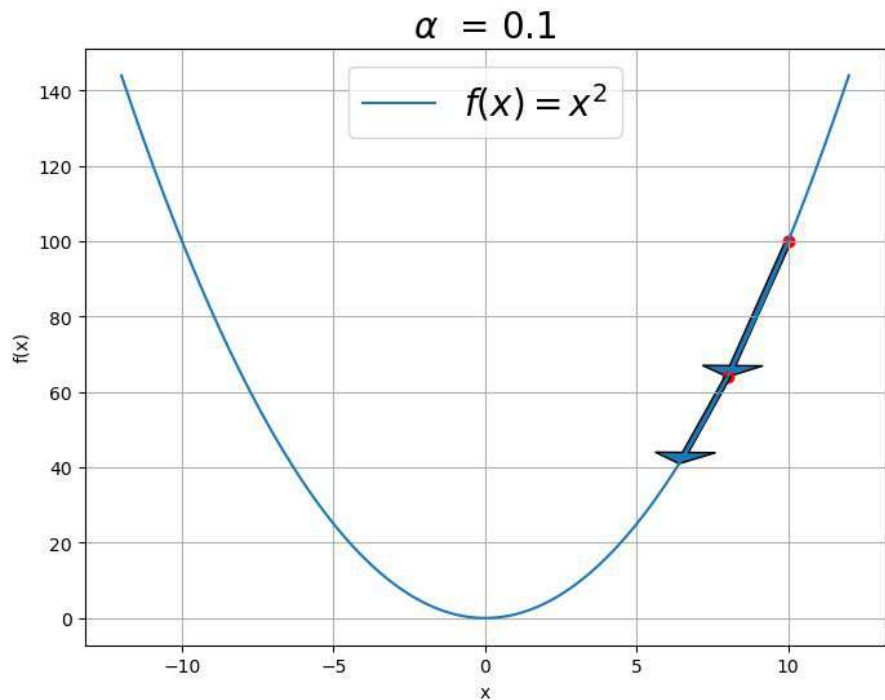
$$p^T \nabla f < 0$$

# Gradient descent



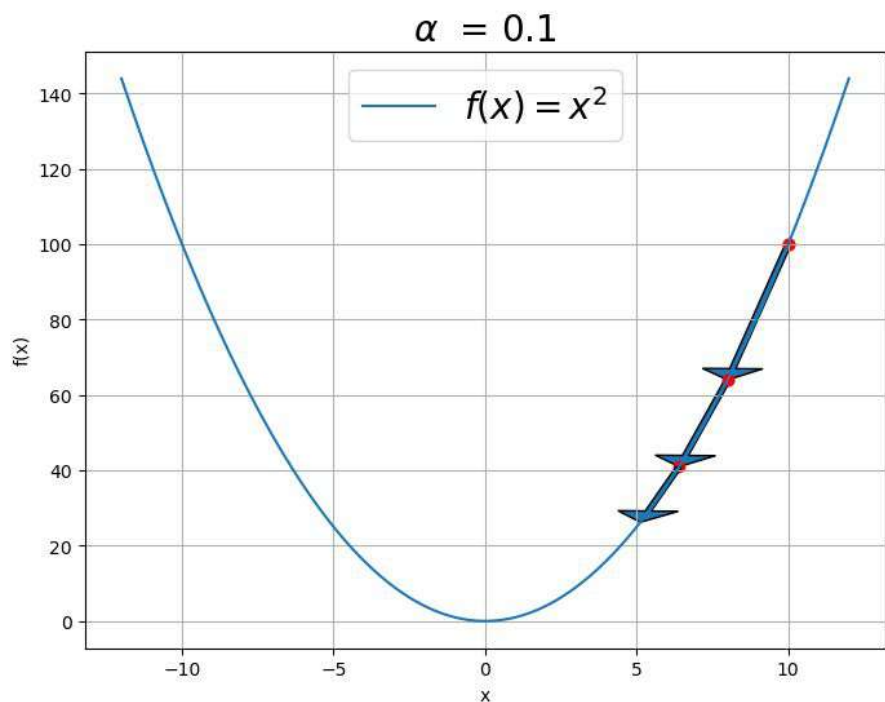
$$x \leftarrow x - \alpha \nabla f$$

# Gradient descent



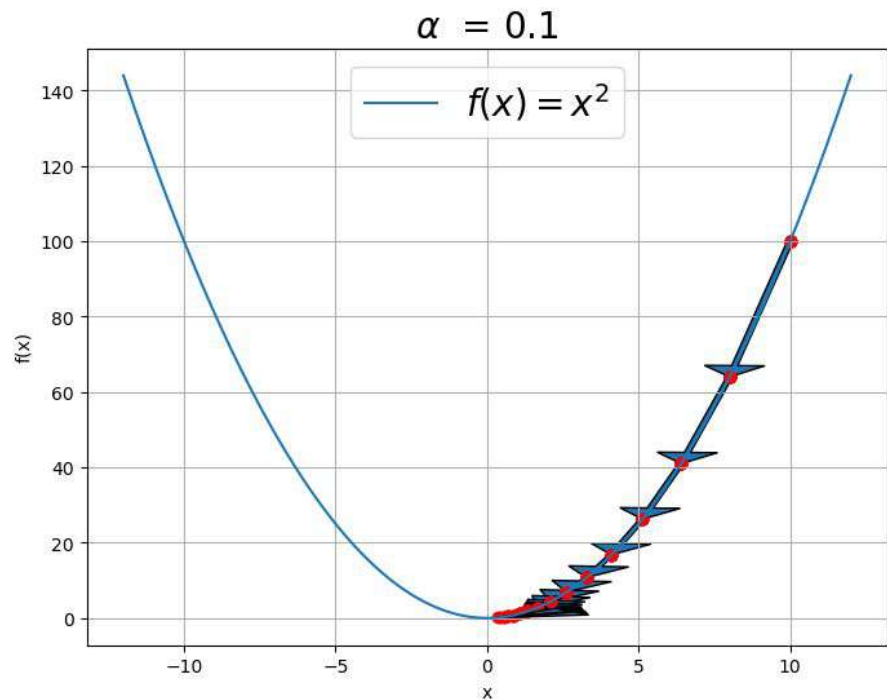
$$x \leftarrow x - \alpha \nabla f$$

# Gradient descent



$$x \leftarrow x - \alpha \nabla f$$

# Gradient descent

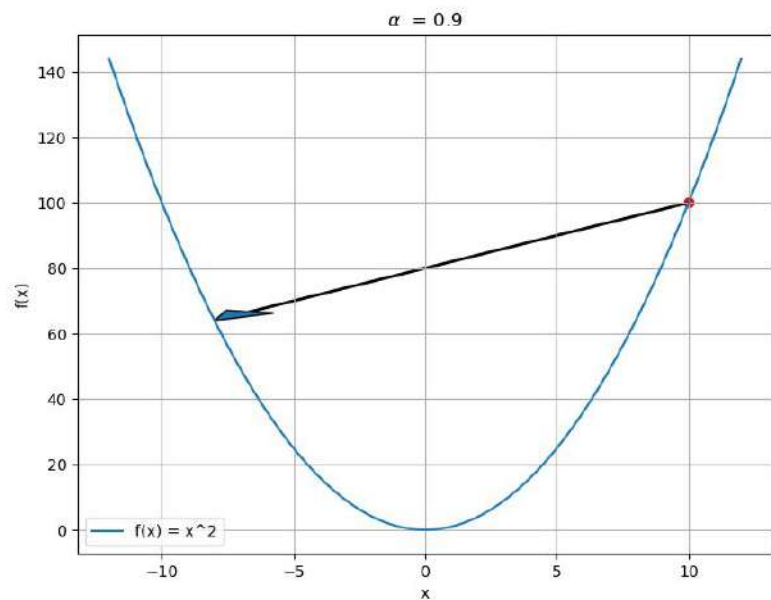


$$x \leftarrow x - \alpha \nabla f$$

What if we increase  $\alpha$ ?

$$x \leftarrow x - \alpha \nabla f$$

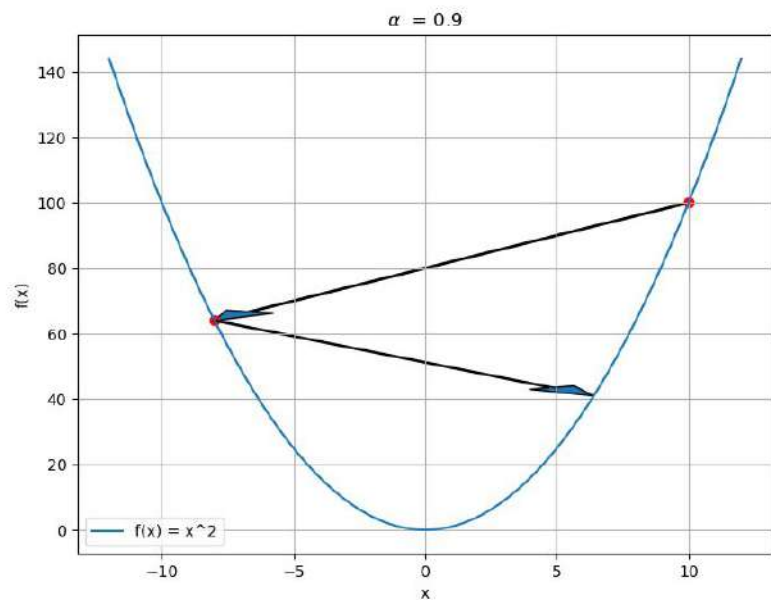
What if we increase  $\alpha$ ?



$$x \leftarrow x - \alpha \nabla f$$

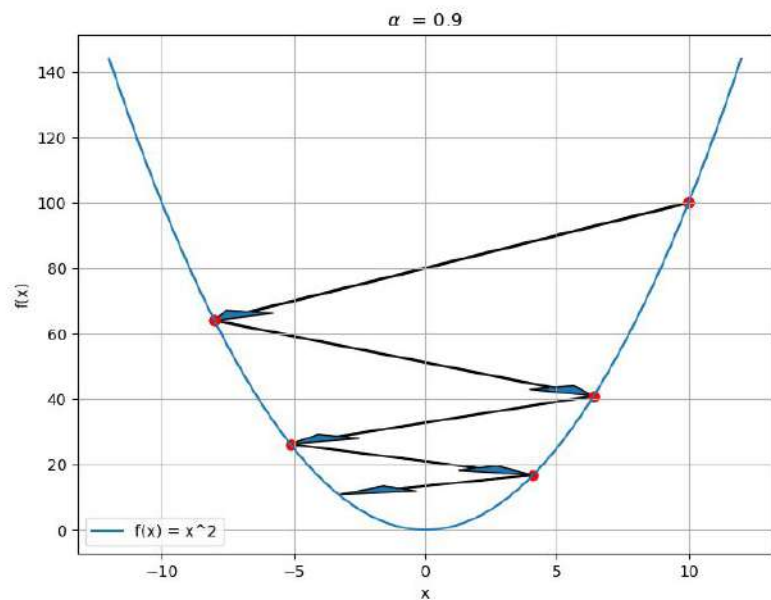


What if we increase  $\alpha$ ?



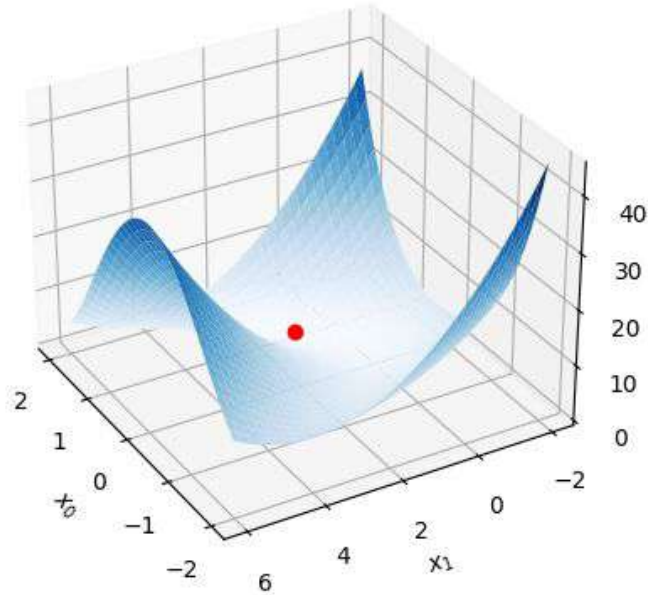
$$x \leftarrow x - \alpha \nabla f$$

What if we increase  $\alpha$ ?



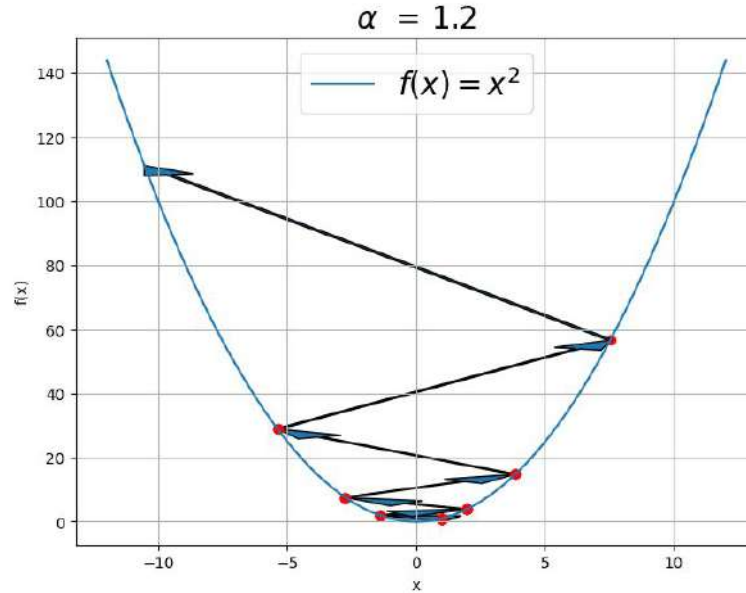
$$x \leftarrow x - \alpha \nabla f$$

# The Rosenbrock function



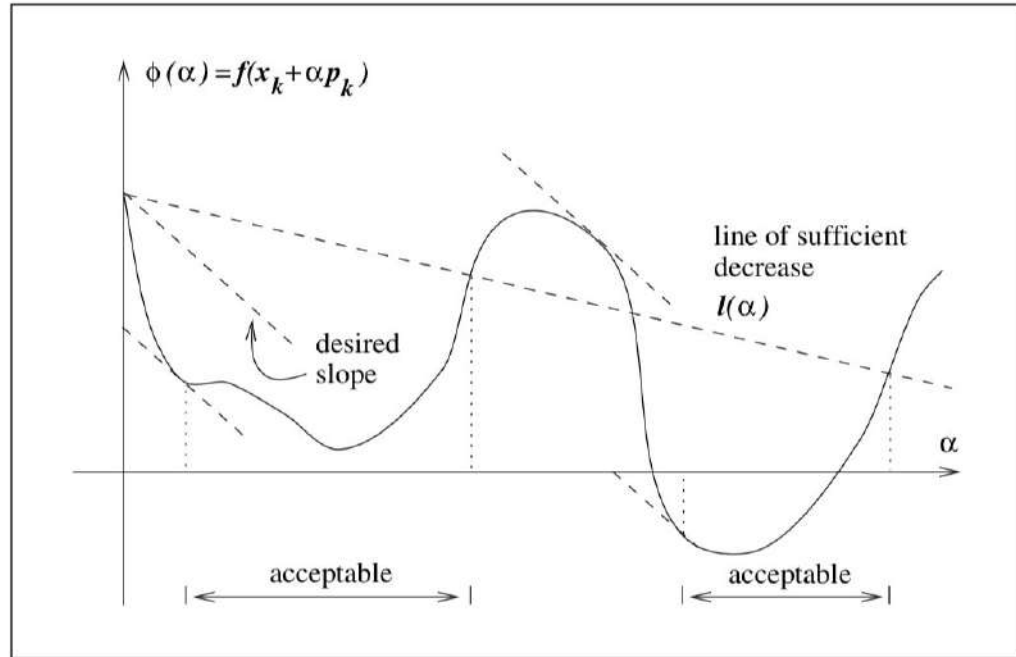
$$f(x, y) = (a - x)^2 + b(y - x^2)^2$$

What if  $\alpha$  is too large?



We need an automated way to choose  $\alpha$

# Line search



# In practice

**Algorithm 3.1** (Backtracking Line Search).

Choose  $\bar{\alpha} > 0$ ,  $\rho \in (0, 1)$ ,  $c \in (0, 1)$ ; Set  $\alpha \leftarrow \bar{\alpha}$ ;

**repeat** until  $f(x_k + \alpha p_k) \leq f(x_k) + c\alpha \nabla f_k^T p_k$

$\alpha \leftarrow \rho\alpha$ ;

**end (repeat)**

Terminate with  $\alpha_k = \alpha$ .

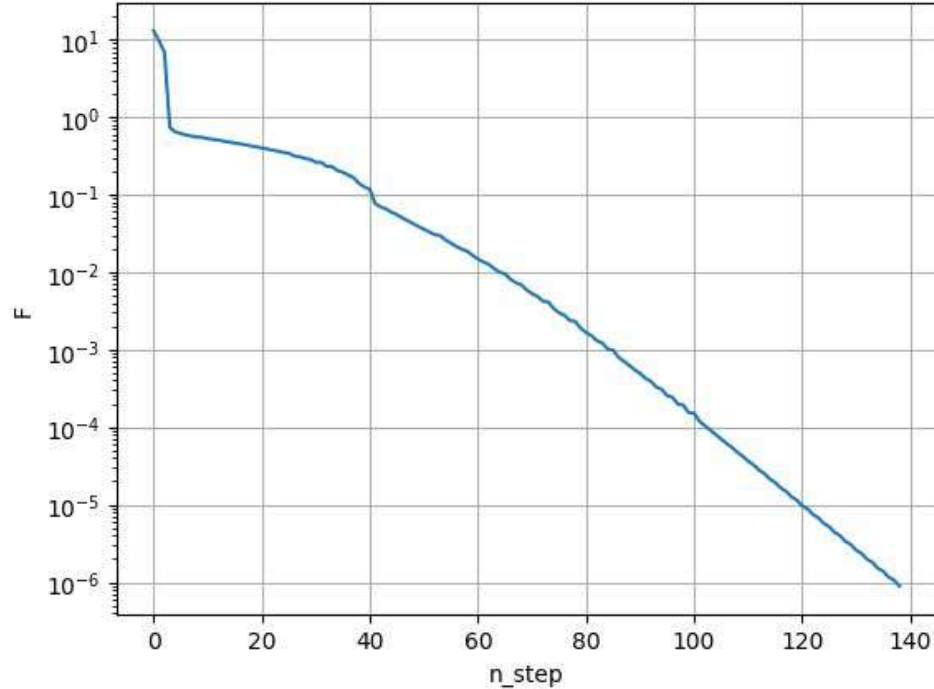
# Theorem

Assuming the gradient is Lipschitz continuous then, the iterates of gradient descent converge to a stationary point:

$$\lim_{k \rightarrow \infty} \|\nabla f_k\| = 0.$$

- Does not guarantee to find a minima
- Will always be local

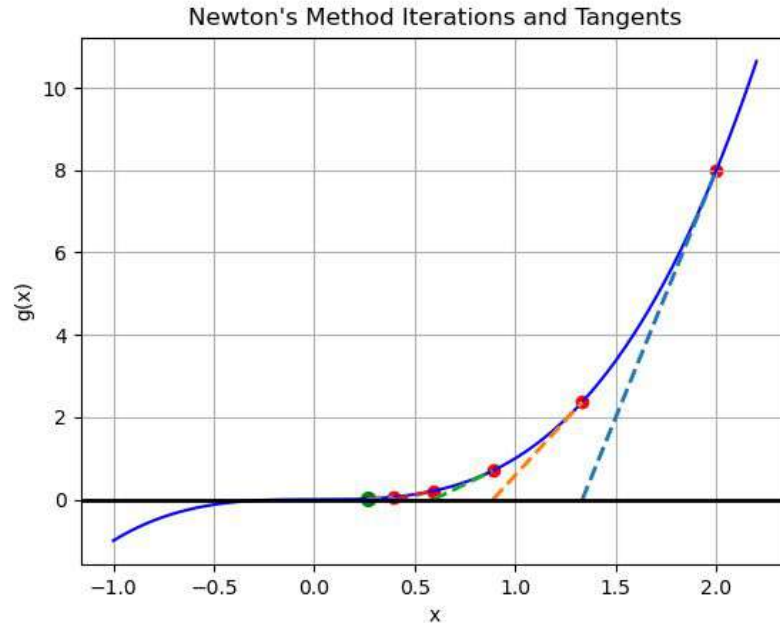
# Convergence rate of gradient descent: Linear





# Newton method

Goal: Find  $x$  such that  $g(x) = 0$



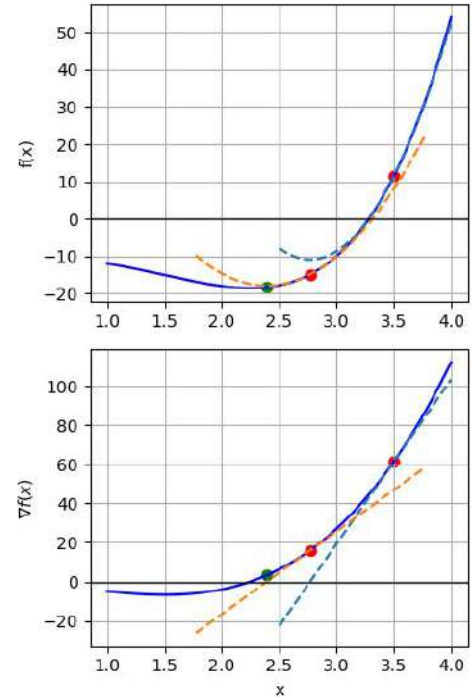
# Application

Derive an algorithm to approximate  $\sqrt{2}$

# Newton method for optimization

$$\min_x f(x)$$

Let's find the zero of  $\nabla f(x)$



## Newton method for optimization

$$x_{k+1} = x_k + \alpha_k p_k$$

$$p_k = -\nabla^2 f(x_k)^{-1} \nabla f(x_k)$$

Under what condition is this a descent direction?

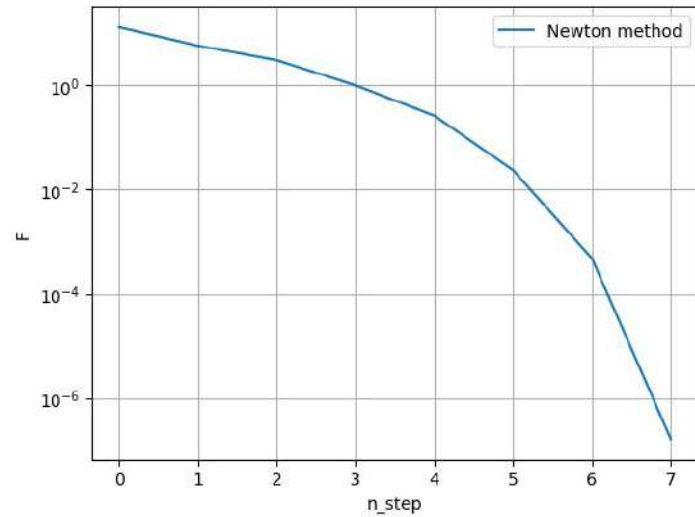
$$\nabla^2 f(x_k) \succ 0$$

Newton method on a quadratic function.

$$f(x) = x^T Q x + q^T x$$

- What is the minimum of  $f$ ?
- Derive a Newton step

# Convergence rate: Quadratic



# Nonlinear constrained optimization

$$\begin{aligned} \min_x f(x) &= 0 \\ \text{subject to } g(x) &= 0 \end{aligned}$$

Recall the KKT condition:

$$\begin{aligned} \nabla f(x) + \lambda \nabla g(x) &= 0 \\ g(x) &= 0 \end{aligned}$$

Sequential Quadratic Programming (SQP): Apply Newton's method on the KKT

# Sequential Quadratic Programming (SQP)

$$x_{k+1} = x_k + p_k$$

where:

$$\begin{aligned} & \min_p p^T \nabla^2 f(x_k) p + p^T \nabla f(x_k) \\ & \text{subject to } \nabla g(x_k)^T p + g(x_k) = 0 \end{aligned}$$



# Example

$$\begin{aligned} \min_x f(x) &= 0 \\ \text{subject to } g(x) &= 0 \end{aligned}$$

$$\begin{aligned} f(x) &= x^T Q x + q^T x \\ g(x) &= A x + b \end{aligned}$$

- How to find the solution?
- Derive an SQP

# Sequential Quadratic Programming (SQP)

$$\begin{aligned} & \min_x f(x) \\ & \text{subject to } g(x) = 0 \end{aligned}$$

Step 1: Find a direction

$$\begin{aligned} & \min_p p^T \nabla_{xx}^2 \mathcal{L}(x_k) p + p^T \nabla f(x_k) \\ & \text{subject to } \nabla g(x_k)^T p + g(x_k) = 0 \end{aligned}$$

Step 2: Find a step length  $\alpha_k$  with a line search

## Merit function

$$\phi(x) = f(x) + \mu ||g(x)||_1$$

Sufficient decrease condition

$$\phi(x_k + \alpha_k p_k) \leq \phi(x_k) + \eta \alpha_k (\nabla f_k^T p_k - \mu ||g(x_k)||_1)$$

# Nonlinear Optimal control

$$\min_{x_1, \dots, x_T, u_0, \dots, u_{T-1}} \sum_{t=0}^{T-1} \ell_t(x_t, u_t) + \ell_T(x_T)$$

subject to  $x_{t+1} = f(x_t, u_t)$

Example: Taichi robot



How can we avoid inverting a matrix of size  $T$ ?

**Proposition 2.** *By applying the Thomas algorithm, we recover the well-known Riccati recursions. Specifically, the **backward pass** can be done by initializing  $V_T = Q_T$  and  $v_T = q_T$ , and then by applying the following equations:*

$$h_k = r_k + B_k^T(v_{k+1} + V_{k+1}\gamma_{k+1}) \quad (9)$$

$$G_k = S_k^T + B_n^T V_{k+1} A_k \quad K_k = -H_k^{-1} G_k$$

$$H_k = R_k + B_k^T V_{k+1} B_k \quad k_k = -H_k^{-1} h_k$$

$$V_k = Q_k + A_k^T V_{k+1} A_k - K_k^T H_k K_k$$

$$v_k = q_k + K_k^T r_k + (A_k + K_k B_k)^T (v_{k+1} + V_{k+1} \gamma_{k+1})$$

*Then, the **forward pass** initializes  $\Delta x_0 = 0$  and unrolls the linearized dynamics:*

$$\Delta x_{k+1} = (A_k + B_k K_k) \Delta x_k + B_k k_k + \gamma_{k+1} \quad (10a)$$

$$\Delta u_k = K_k \Delta x_k + k_k \quad (10b)$$

$$\lambda_k = V_k \Delta x_k + v_k \quad (10c)$$

## Extension to inequality

$$\min_{x_1, \dots, x_T, u_0, \dots, u_{T-1}} \sum_{t=0}^{T-1} \ell_t(x_t, u_t) + \ell_T(x_T)$$

$$\text{subject to } x_{t+1} = f(x_t, u_t)$$

$$h_t(x_t, u_t) \leq 0$$

$$h_T(x_T) \leq 0$$



## Extension to inequality

$$\begin{aligned} & \min_x f(x) \\ & \text{subject to } g(x) = 0 \\ & h(x) \leq 0 \end{aligned}$$

Step 1: Find a direction

$$\begin{aligned} & \min_p p^T \nabla_{xx}^2 \mathcal{L}(x_k) p + p^T \nabla f(x_k) \\ & \text{subject to } \nabla g(x_k)^T p + g(x_k) = 0 \\ & \nabla h(x_k)^T p + h(x_k) \leq 0 \end{aligned}$$

Step 2: Find a step length  $\alpha_k$  with a merit function and line search

# To solve the inequality QP,

You can use any Black box QP solver and implement your own SQP.

Solvers					
Solver	Keyword	Algorithm	API	License	Warm-start
<a href="#">Clarabel</a>	clarabel	Interior point	Sparse	Apache-2.0	×
<a href="#">CVXOPT</a>	cvxopt	Interior point	Dense	GPL-3.0	✓
<a href="#">DAQP</a>	daqp	Active set	Dense	MIT	×
<a href="#">ECOS</a>	ecos	Interior point	Sparse	GPL-3.0	×
<a href="#">Gurobi</a>	gurobi	Interior point	Sparse	Commercial	×
<a href="#">HIGHTS</a>	highs	Active set	Sparse	MIT	×
<a href="#">HPIPM</a>	h1pm	Interior point	Dense	BSD-2-Clause	✓
<a href="#">MOSEK</a>	mosek	Interior point	Sparse	Commercial	✓
<a href="#">NPRO</a>	npro	Active set	Dense	Commercial	✓
<a href="#">OSQP</a>	osqp	Augmented Lagrangian	Sparse	Apache-2.0	✓
<a href="#">PIQP</a>	piqp	Proximal interior point	Dense & Sparse	BSD-2-Clause	×
<a href="#">ProxQP</a>	proxqp	Augmented Lagrangian	Dense & Sparse	BSD-2-Clause	✓
<a href="#">QPALM</a>	qpalm	Augmented Lagrangian	Sparse	LGPL-3.0	✓
<a href="#">qpax</a>	qpax	Interior point	Dense	MIT	×
<a href="#">qpOASES</a>	qpOASES	Active set	Dense	LGPL-2.1	—
<a href="#">qpSWIFT</a>	qpswift	Interior point	Sparse	GPL-3.0	×
<a href="#">quadprog</a>	quadprog	Active set	Dense	GPL-2.0	×
<a href="#">SCS</a>	scs	Augmented Lagrangian	Sparse	MIT	✓

<https://github.com/qpsolvers/qpsolvers>

# Nonlinear optimal control libraries



# Sequential implementation of nonlinear solver

S. Wright, J. Nocedal et al., “Numerical optimization,” Springer Science, vol. 35, no. 67-68, p. 7, 1999.

G. Frison and M. Diehl, “Hpipm: a high-performance quadratic programming framework for model predictive control,” IFAC-PapersOnLine, vol. 53, no. 2, pp. 6563–6569, 2020.

H. J. Ferreau, H. G. Bock, and M. Diehl, “An online active set strategy to overcome the limitations of explicit mpc,” International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal, vol. 18, no. 8, pp. 816–830, 2008

Wang and S. Boyd, “Fast model predictive control using online optimization,” IEEE Transactions on control systems technology, vol. 18, no. 2, pp. 267–278, 2009.

C. Dohrmann and R. Robinett, “Dynamic programming method for constrained discrete-time optimal control,” Journal of Optimization Theory and Applications, vol. 101, pp. 259–283, 1999.

# Contact invariant Optimization



Next week: MPC

# Dynamic Constraint Satisfaction

The Sparse Constrained SQP is able to update its solutions online  
to satisfy dynamically added constraints.