# Friday Seminar
# 2021-06-11

**Manu Upadhyaya**
manu.upadhyaya@control.lth.se

**Lund University**

# Overview

# About me - Education

- BSc in mathematics (2015)
- Study abroad at University of California, Berkeley (2016-2017)
- MSc in engineering physics, specialization in financial modelling (2020)
- MSc in finance (2020)

# About me - Education

- BSc in mathematics (2015)
- Study abroad at University of California, Berkeley (2016-2017)
- MSc in engineering physics, specialization in financial modelling (2020)
- MSc in finance (2020)

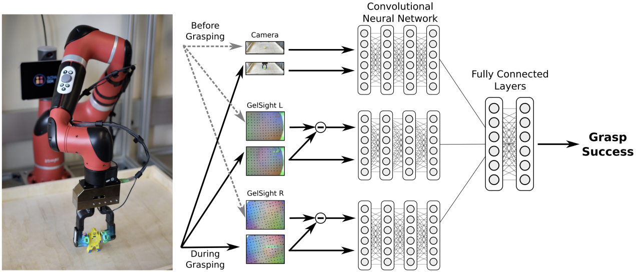
# About me - Education

- BSc in mathematics (2015)
- Study abroad at University of California, Berkeley (2016-2017)
- MSc in engineering physics, specialization in financial modelling (2020)
- MSc in finance (2020)

# About me - Education

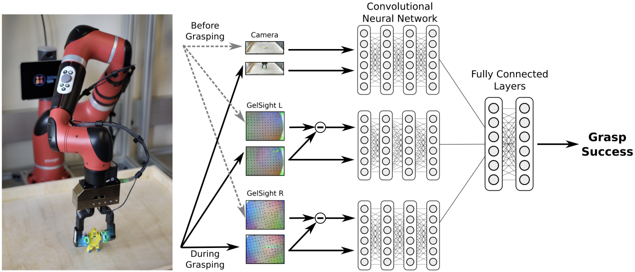
- BSc in mathematics (2015)
- Study abroad at University of California, Berkeley (2016-2017)
- MSc in engineering physics, specialization in financial modelling (2020)
- MSc in finance (2020)

# About me - Before joining the department
# Summer 2017




- Worked on vision and tactile sensing for robotic manipulation using deep neural network predictive models
- @ Sergey Levine's research group at UC Berkeley

# About me - Before joining the department
# Summer 2017




- Worked on vision and tactile sensing for robotic manipulation using deep neural network predictive models
- @ Sergey Levine's research group at UC Berkeley

# About me - Before joining the department
# Summer 2018

$$
\begin{array}{ccccccccc}
\emptyset & \xrightarrow{k_s} & \boldsymbol{S}, & \emptyset & \xrightarrow{k_i} & \boldsymbol{I}, & \boldsymbol{S} & \xrightarrow{\gamma_s} & \emptyset \\
\boldsymbol{I} & \xrightarrow{\gamma_i} & \emptyset, & \boldsymbol{R} & \xrightarrow{\gamma_r} & \emptyset, & \boldsymbol{S}+\boldsymbol{I} & \xrightarrow{k_{si}} & 2\boldsymbol{I} \\
\boldsymbol{I} & \xrightarrow{k_{ir}} & \boldsymbol{R}, & \boldsymbol{R} & \xrightarrow{k_{rs}} & \boldsymbol{S}. & & &
\end{array}
$$

- Looked at stability issues of stochastic biochemical reaction networks (populations of a finite number of species that evolve through predefined interactions)
- *Automated construction of Foster-Lyapunov functions to prove ergodicity of continuous-time Markov processes via convex optimization*
- @ Mustafa Khammash's research group at ETH Zürich, D-BSSE

# About me - Before joining the department
# Summer 2018

$$
\begin{array}{llllll}
\emptyset \xrightarrow{k_s} \boldsymbol{S}, & \emptyset \xrightarrow{k_i} \boldsymbol{I}, & \boldsymbol{S} \xrightarrow{\gamma_s} \emptyset \\
\boldsymbol{I} \xrightarrow{\gamma_i} \emptyset, & \boldsymbol{R} \xrightarrow{\gamma_r} \emptyset, & \boldsymbol{S}+\boldsymbol{I} \xrightarrow{k_{si}} 2\boldsymbol{I} \\
\boldsymbol{I} \xrightarrow{k_{ir}} \boldsymbol{R}, & \boldsymbol{R} \xrightarrow{k_{rs}} \boldsymbol{S}. &
\end{array}
$$

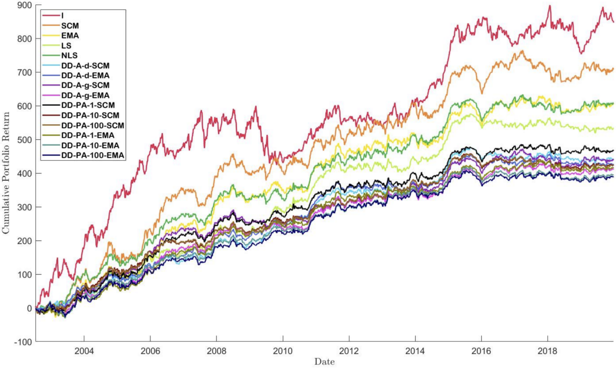
- Looked at stability issues of stochastic biochemical reaction networks (populations of a finite number of species that evolve through predefined interactions)
- *Automated construction of Foster-Lyapunov functions to prove ergodicity of continuous-time Markov processes via convex optimization*
- @ Mustafa Khammash's research group at ETH Zürich, D-BSSE

# About me - Before joining the department Summer 2018

$$
\begin{array}{lllllllll}
\emptyset & \xrightarrow{k_s} & \boldsymbol{S}, & \emptyset & \xrightarrow{k_i} & \boldsymbol{I}, & \boldsymbol{S} & \xrightarrow{\gamma_s} & \emptyset \\
\boldsymbol{I} & \xrightarrow{\gamma_i} & \emptyset, & \boldsymbol{R} & \xrightarrow{\gamma_r} & \emptyset, & \boldsymbol{S}+\boldsymbol{I} & \xrightarrow{k_{si}} & 2\boldsymbol{I} \\
\boldsymbol{I} & \xrightarrow{k_{ir}} & \boldsymbol{R}, & \boldsymbol{R} & \xrightarrow{k_{rs}} & \boldsymbol{S}. & & &
\end{array}
$$

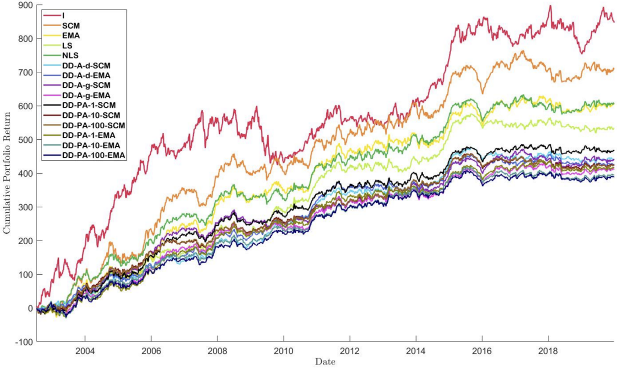
- Looked at stability issues of stochastic biochemical reaction networks (populations of a finite number of species that evolve through predefined interactions)
- *Automated construction of Foster-Lyapunov functions to prove ergodicity of continuous-time Markov processes via convex optimization*
- @ Mustafa Khammash's research group at ETH Zürich, D-BSSE

# About me - Before joining the department
# Master thesis



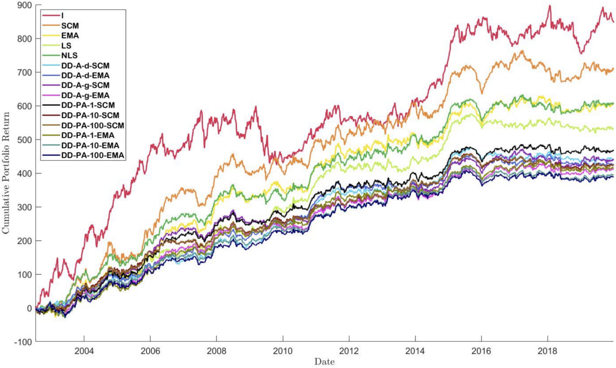

- Data-driven and non-parametric methods for covariance matrix regularization for portfolio selection
- @ Lynx Asset Management in Stockholm
- Supervisors: Tobias Rydén, Magnus Wiktorsson, **Pontus Giselsson**, Frederik Lundtofte

# About me - Before joining the department
# Master thesis




- Data-driven and non-parametric methods for covariance matrix regularization for portfolio selection
- @ Lynx Asset Management in Stockholm
- Supervisors: Tobias Rydén, Magnus Wiktorsson, **Pontus Giselsson**, Frederik Lundtofte

# About me - Before joining the department
# Master thesis




- Data-driven and non-parametric methods for covariance matrix regularization for portfolio selection
- @ Lynx Asset Management in Stockholm
- Supervisors: Tobias Rydén, Magnus Wiktorsson, **Pontus Giselsson**, Frederik Lundtofte

# Performance estimation problems - The work this presentation is based on

**Performance of first-order methods for smooth convex minimization**
**(Drori and Teboulle, 2014)**

# Performance estimation problems - Motivation

- Class of functions $\mathcal{F}$:
  - Collection of functions $f : \mathbf{R}^d \to \mathbf{R}$ with some properties
  - Assume $\exists x_* \in X_*(f)$, where $X_*(f)$ is the set of minimizers of $f$
- Want to minimize functions in $\mathcal{F}$ via some algorithm
- First-order black-box optimization method on $\mathcal{F}$ is an algorithm $\mathcal{A}$:
  - $x_0 \in \mathbf{R}^d$ initial point
  - $f \in \mathcal{F}$ fixed
  - $x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right)$ for each $i = 0, \dots, N-1$
- Worst-case analysis: Given $\mathcal{A}$, what is

$$\max_{f \in \mathcal{F}} (f(x_N) - f(x_*))?$$

- Worst-case design: Given some class of algorithms $\mathbb{A}$, what is

$$\mathcal{A}^* = \arg\min_{\mathcal{A} \in \mathbb{A}} \left( \max_{f \in \mathcal{F}} (f(x_N) - f(x_*)) \right)?$$

*(We will not cover worst-case design today)*

# Performance estimation problems - Motivation

- Class of functions $\mathcal{F}$:
  - Collection of functions $f : \mathbf{R}^d \to \mathbf{R}$ with some properties
  - Assume $\exists x_* \in X_*(f)$, where $X_*(f)$ is the set of minimizers of $f$
- Want to minimize functions in $\mathcal{F}$ via some algorithm
- First-order black-box optimization method on $\mathcal{F}$ is an algorithm $\mathcal{A}$:
  - $x_0 \in \mathbf{R}^d$ initial point
  - $f \in \mathcal{F}$ fixed
  - $x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^i, \{f(x_j)\}_{j=0}^i, \{\nabla f(x_j)\}_{j=0}^i\right)$ for each $i = 0, \ldots, N-1$
- Worst-case analysis: Given $\mathcal{A}$, what is

$$\max_{f \in \mathcal{F}} (f(x_N) - f(x_*))?$$

- Worst-case design: Given some class of algorithms $\mathbb{A}$, what is

$$\mathcal{A}^* = \operatorname*{arg\,min}_{\mathcal{A} \in \mathbb{A}} \left( \max_{f \in \mathcal{F}} (f(x_N) - f(x_*)) \right)?$$

*(We will not cover worst-case design today)*

# Performance estimation problems - Motivation

- Class of functions $\mathcal{F}$:
  - Collection of functions $f : \mathbf{R}^d \to \mathbf{R}$ with some properties
  - Assume $\exists x_* \in X_*(f)$, where $X_*(f)$ is the set of minimizers of $f$
- Want to minimize functions in $\mathcal{F}$ via some algorithm
- First-order black-box optimization method on $\mathcal{F}$ is an algorithm $\mathcal{A}$:
  - $x_0 \in \mathbf{R}^d$ initial point
  - $f \in \mathcal{F}$ fixed
  - $x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right)$ for each $i = 0, \ldots, N-1$
- Worst-case analysis: Given $\mathcal{A}$, what is

$$\max_{f \in \mathcal{F}} (f(x_N) - f(x_*))?$$

- Worst-case design: Given some class of algorithms $\mathbb{A}$, what is

$$\mathcal{A}^* = \arg\min_{\mathcal{A} \in \mathbb{A}} \left( \max_{f \in \mathcal{F}} (f(x_N) - f(x_*)) \right)?$$

*(We will not cover worst-case design today)*

# Performance estimation problems - Motivation

- Class of functions $\mathcal{F}$:
  - Collection of functions $f : \mathbf{R}^d \to \mathbf{R}$ with some properties
  - Assume $\exists x_* \in X_*(f)$, where $X_*(f)$ is the set of minimizers of $f$
- Want to minimize functions in $\mathcal{F}$ via some algorithm
- First-order black-box optimization method on $\mathcal{F}$ is an algorithm $\mathcal{A}$:
  - $x_0 \in \mathbf{R}^d$ initial point
  - $f \in \mathcal{F}$ fixed
  - $x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right)$ for each $i = 0, \ldots, N-1$
- Worst-case analysis: Given $\mathcal{A}$, what is

$$\max_{f \in \mathcal{F}} (f(x_N) - f(x_*))?$$

- Worst-case design: Given some class of algorithms $\mathbb{A}$, what is

$$\mathcal{A}^* = \arg\min_{\mathcal{A} \in \mathbb{A}} \left( \max_{f \in \mathcal{F}} (f(x_N) - f(x_*)) \right)?$$

*(We will not cover worst-case design today)*

# Performance estimation problems - Motivation

- Class of functions $\mathcal{F}$:
  - Collection of functions $f : \mathbf{R}^d \to \mathbf{R}$ with some properties
  - Assume $\exists x_* \in X_*(f)$, where $X_*(f)$ is the set of minimizers of $f$
- Want to minimize functions in $\mathcal{F}$ via some algorithm
- First-order black-box optimization method on $\mathcal{F}$ is an algorithm $\mathcal{A}$:
  - $x_0 \in \mathbf{R}^d$ initial point
  - $f \in \mathcal{F}$ fixed
  - $x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right)$ for each $i = 0, \ldots, N-1$
- Worst-case analysis: Given $\mathcal{A}$, what is

$$\max_{f \in \mathcal{F}} (f(x_N) - f(x_*))?$$

- Worst-case design: Given some class of algorithms $\mathbb{A}$, what is

$$\mathcal{A}^* = \arg\min_{\mathcal{A} \in \mathbb{A}} \left( \max_{f \in \mathcal{F}} (f(x_N) - f(x_*)) \right)?$$

*(We will not cover worst-case design today)*

# Performance estimation problems - Motivation

- Class of functions $\mathcal{F}$:
  - Collection of functions $f : \mathbf{R}^d \to \mathbf{R}$ with some properties
  - Assume $\exists x_* \in X_*(f)$, where $X_*(f)$ is the set of minimizers of $f$
- Want to minimize functions in $\mathcal{F}$ via some algorithm
- First-order black-box optimization method on $\mathcal{F}$ is an algorithm $\mathcal{A}$:
  - $x_0 \in \mathbf{R}^d$ initial point
  - $f \in \mathcal{F}$ fixed
  - $x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right)$ for each $i = 0, \ldots, N-1$
- Worst-case analysis: Given $\mathcal{A}$, what is

$$\max_{f \in \mathcal{F}} (f(x_N) - f(x_*))?$$

- Worst-case design: Given some class of algorithms $\mathbb{A}$, what is

$$\mathcal{A}^* = \arg\min_{\mathcal{A} \in \mathbb{A}} \left( \max_{f \in \mathcal{F}} (f(x_N) - f(x_*)) \right)?$$

*(We will not cover worst-case design today)*

# Performance estimation problems - Motivation

- Class of functions $\mathcal{F}$:
  - Collection of functions $f : \mathbf{R}^d \to \mathbf{R}$ with some properties
  - Assume $\exists x_* \in X_*(f)$, where $X_*(f)$ is the set of minimizers of $f$
- Want to minimize functions in $\mathcal{F}$ via some algorithm
- First-order black-box optimization method on $\mathcal{F}$ is an algorithm $\mathcal{A}$:
  - $x_0 \in \mathbf{R}^d$ initial point
  - $f \in \mathcal{F}$ fixed
  - $x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right)$ for each $i = 0, \ldots, N-1$
- Worst-case analysis: Given $\mathcal{A}$, what is

$$\max_{f \in \mathcal{F}} (f(x_N) - f(x_*))?$$

- Worst-case design: Given some class of algorithms $\mathbb{A}$, what is

$$\mathcal{A}^* = \operatorname*{arg\,min}_{\mathcal{A} \in \mathbb{A}} \left( \max_{f \in \mathcal{F}} (f(x_N) - f(x_*)) \right)?$$

*(We will not cover worst-case design today)*

# Performance estimation problems - Motivation

- Class of functions $\mathcal{F}$:
  - Collection of functions $f : \mathbf{R}^d \to \mathbf{R}$ with some properties
  - Assume $\exists x_* \in X_*(f)$, where $X_*(f)$ is the set of minimizers of $f$
- Want to minimize functions in $\mathcal{F}$ via some algorithm
- First-order black-box optimization method on $\mathcal{F}$ is an algorithm $\mathcal{A}$:
  - $x_0 \in \mathbf{R}^d$ initial point
  - $f \in \mathcal{F}$ fixed
  - $x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right)$ for each $i = 0, \ldots, N-1$
- Worst-case analysis: Given $\mathcal{A}$, what is

$$\max_{f \in \mathcal{F}} (f(x_N) - f(x_*))?$$

- Worst-case design: Given some class of algorithms $\mathbb{A}$, what is

$$\mathcal{A}^* = \arg\min_{\mathcal{A} \in \mathbb{A}} \left( \max_{f \in \mathcal{F}} (f(x_N) - f(x_*)) \right)?$$

*(We will not cover worst-case design today)*

# Performance estimation problems - Motivation

- Class of functions $\mathcal{F}$:
  - Collection of functions $f : \mathbf{R}^d \to \mathbf{R}$ with some properties
  - Assume $\exists x_* \in X_*(f)$, where $X_*(f)$ is the set of minimizers of $f$
- Want to minimize functions in $\mathcal{F}$ via some algorithm
- First-order black-box optimization method on $\mathcal{F}$ is an algorithm $\mathcal{A}$:
  - $x_0 \in \mathbf{R}^d$ initial point
  - $f \in \mathcal{F}$ fixed
  - $x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right)$ for each $i = 0, \ldots, N-1$
- Worst-case analysis: Given $\mathcal{A}$, what is

$$\max_{f \in \mathcal{F}} \left(f(x_N) - f(x_*)\right)?$$

- Worst-case design: Given some class of algorithms $\mathbb{A}$, what is

$$\mathcal{A}^* = \underset{\mathcal{A} \in \mathbb{A}}{\arg\min} \left(\max_{f \in \mathcal{F}} \left(f(x_N) - f(x_*)\right)\right)?$$

*(We will not cover worst-case design today)*

# Performance estimation problems - Motivation

- Class of functions $\mathcal{F}$:
  - Collection of functions $f : \mathbf{R}^d \to \mathbf{R}$ with some properties
  - Assume $\exists x_* \in X_*(f)$, where $X_*(f)$ is the set of minimizers of $f$
- Want to minimize functions in $\mathcal{F}$ via some algorithm
- First-order black-box optimization method on $\mathcal{F}$ is an algorithm $\mathcal{A}$:
  - $x_0 \in \mathbf{R}^d$ initial point
  - $f \in \mathcal{F}$ fixed
  - $x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^i, \{f(x_j)\}_{j=0}^i, \{\nabla f(x_j)\}_{j=0}^i\right)$ for each $i = 0, \ldots, N-1$
- Worst-case analysis: Given $\mathcal{A}$, what is

$$\max_{f \in \mathcal{F}} \left(f(x_N) - f(x_*)\right)?$$

- Worst-case design: Given some class of algorithms $\mathbb{A}$, what is

$$\mathcal{A}^* = \arg\min_{\mathcal{A} \in \mathbb{A}} \left(\max_{f \in \mathcal{F}} \left(f(x_N) - f(x_*)\right)\right)?$$

*(We will not cover worst-case design today)*

# Performance estimation problems - Assumptions

- Let $L > 0$. $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$ if and only if $f : \mathbf{R}^d \to \mathbf{R}$ is continuously differentiable, convex and the gradient $\nabla f$ is $L$-Lipschitz continuous
- Let $\mathcal{A}$ be a first-order black-box optimization method on $\mathcal{F}_L^{1,1}(\mathbf{R}^d)$
- Consider only $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$ such that $X_*(f) := \arg\min_{x \in \mathbf{R}^d} f(x)$ is non-empty
- Let $R > 0$. For each $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$, consider only initial points $x_0 \in \mathbf{R}^d$ such that there exists an $x_* \in X_*(f)$ such that $\|x_* - x_0\|_2 \leq R$
- $\mathcal{A}$ generates a finite sequence of length $N + 1$ (including the initial point)

# Performance estimation problems - Assumptions

- Let $L > 0$. $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$ if and only if $f : \mathbf{R}^d \to \mathbf{R}$ is continuously differentiable, convex and the gradient $\nabla f$ is $L$-Lipschitz continuous
- Let $\mathcal{A}$ be a first-order black-box optimization method on $\mathcal{F}_L^{1,1}(\mathbf{R}^d)$
- Consider only $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$ such that $X_*(f) := \arg\min_{x \in \mathbf{R}^d} f(x)$ is non-empty
- Let $R > 0$. For each $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$, consider only initial points $x_0 \in \mathbf{R}^d$ such that there exists an $x_* \in X_*(f)$ such that $\|x_* - x_0\|_2 \leq R$
- $\mathcal{A}$ generates a finite sequence of length $N + 1$ (including the initial point)

# Performance estimation problems - Assumptions

- Let $L > 0$. $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$ if and only if $f : \mathbf{R}^d \to \mathbf{R}$ is continuously differentiable, convex and the gradient $\nabla f$ is $L$-Lipschitz continuous
- Let $\mathcal{A}$ be a first-order black-box optimization method on $\mathcal{F}_L^{1,1}(\mathbf{R}^d)$
- Consider only $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$ such that $X_*(f) := \arg\min_{x \in \mathbf{R}^d} f(x)$ is non-empty
- Let $R > 0$. For each $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$, consider only initial points $x_0 \in \mathbf{R}^d$ such that there exists an $x_* \in X_*(f)$ such that $\|x_* - x_0\|_2 \leq R$
- $\mathcal{A}$ generates a finite sequence of length $N + 1$ (including the initial point)

# Performance estimation problems - Assumptions

- Let $L > 0$. $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$ if and only if $f : \mathbf{R}^d \to \mathbf{R}$ is continuously differentiable, convex and the gradient $\nabla f$ is $L$-Lipschitz continuous
- Let $\mathcal{A}$ be a first-order black-box optimization method on $\mathcal{F}_L^{1,1}(\mathbf{R}^d)$
- Consider only $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$ such that $X_*(f) := \arg\min_{x \in \mathbf{R}^d} f(x)$ is non-empty
- Let $R > 0$. For each $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$, consider only initial points $x_0 \in \mathbf{R}^d$ such that there exists an $x_* \in X_*(f)$ such that $\|x_* - x_0\|_2 \leq R$
- $\mathcal{A}$ generates a finite sequence of length $N + 1$ (including the initial point)

# Performance estimation problems - Assumptions

- Let $L > 0$. $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$ if and only if $f : \mathbf{R}^d \to \mathbf{R}$ is continuously differentiable, convex and the gradient $\nabla f$ is $L$-Lipschitz continuous
- Let $\mathcal{A}$ be a first-order black-box optimization method on $\mathcal{F}_L^{1,1}(\mathbf{R}^d)$
- Consider only $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$ such that $X_*(f) := \arg\min_{x \in \mathbf{R}^d} f(x)$ is non-empty
- Let $R > 0$. For each $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$, consider only initial points $x_0 \in \mathbf{R}^d$ such that there exists an $x_* \in X_*(f)$ such that $\|x_* - x_0\|_2 \leq R$
- $\mathcal{A}$ generates a finite sequence of length $N + 1$ (including the initial point)

# Performance estimation problems - The problem

$$
\begin{array}{ll}
\text{maximize} & f(x_N) - f(x_*) \\
\text{subject to} & f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d), \\
& x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right),\ i = 0, \ldots, N-1, \\
& x_* \in X_*(f), \\
& \|x_* - x_0\|_2 \leq R, \\
& x_0, \ldots, x_N, x_* \in \mathbf{R}^d
\end{array}
\tag{P}
$$

- Variables: $x_0, \ldots, x_N, x_*, f$
- Problem data: $\mathcal{F}_L^{1,1}(\mathbf{R}^d), \mathcal{A}, R, N$

**Difficulty**: Optimization problem (P) is abstract, hard and infinite dimensional

**Approach**: Relax constraints in (P), reduce and reformulate as tractable finite dimensional optimization problem

**Note**: Relaxing constraints in (P) may increase the maximum value. Sometimes relaxing constraints does not increase the maximum value and gives tight bounds on the performance of $\mathcal{A}$

# Performance estimation problems - The problem

$$
\begin{array}{ll}
\text{maximize} & f(x_N) - f(x_*) \\
\text{subject to} & f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d), \\
& x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right),\ i = 0, \ldots, N-1, \\
& x_* \in X_*(f), \\
& \|x_* - x_0\|_2 \leq R, \\
& x_0, \ldots, x_N, x_* \in \mathbf{R}^d
\end{array}
\tag{P}
$$

- Variables: $x_0, \ldots, x_N, x_*, f$
- Problem data: $\mathcal{F}_L^{1,1}(\mathbf{R}^d), \mathcal{A}, R, N$

**Difficulty**: Optimization problem (P) is abstract, hard and infinite dimensional

**Approach**: Relax constraints in (P), reduce and reformulate as tractable finite dimensional optimization problem

**Note**: Relaxing constraints in (P) may increase the maximum value. Sometimes relaxing constraints does not increase the maximum value and gives tight bounds on the performance of $\mathcal{A}$

# Performance estimation problems - The problem

$$
\begin{array}{ll}
\text{maximize} & f(x_N) - f(x_*) \\
\text{subject to} & f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d), \\
& x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right),\ i = 0, \ldots, N-1, \\
& x_* \in X_*(f), \\
& \|x_* - x_0\|_2 \leq R, \\
& x_0, \ldots, x_N, x_* \in \mathbf{R}^d
\end{array} \tag{P}
$$

- Variables: $x_0, \ldots, x_N, x_*, f$
- Problem data: $\mathcal{F}_L^{1,1}(\mathbf{R}^d), \mathcal{A}, R, N$

**Difficulty**: Optimization problem (P) is abstract, hard and infinite dimensional

**Approach**: Relax constraints in (P), reduce and reformulate as tractable finite dimensional optimization problem

**Note**: Relaxing constraints in (P) may increase the maximum value. Sometimes relaxing constraints does not increase the maximum value and gives tight bounds on the performance of $\mathcal{A}$

# Performance estimation problems - The problem

$$
\begin{array}{ll}
\text{maximize} & f(x_N) - f(x_*) \\
\text{subject to} & f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d), \\
& x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right),\ i = 0, \ldots, N-1, \\
& x_* \in X_*(f), \\
& \|x_* - x_0\|_2 \le R, \\
& x_0, \ldots, x_N, x_* \in \mathbf{R}^d
\end{array} \tag{P}
$$

- Variables: $x_0, \ldots, x_N, x_*, f$
- Problem data: $\mathcal{F}_L^{1,1}(\mathbf{R}^d), \mathcal{A}, R, N$

**Difficulty**: Optimization problem (P) is abstract, hard and infinite dimensional

**Approach**: Relax constraints in (P), reduce and reformulate as tractable finite dimensional optimization problem

**Note**: Relaxing constraints in (P) may increase the maximum value. Sometimes relaxing constraints does not increase the maximum value and gives tight bounds on the performance of $\mathcal{A}$

# Performance estimation problems - The problem

$$
\begin{array}{ll}
\text{maximize} & f(x_N) - f(x_*) \\
\text{subject to} & f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d), \\
& x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right),\ i = 0, \ldots, N-1, \\
& x_* \in X_*(f), \\
& \|x_* - x_0\|_2 \leq R, \\
& x_0, \ldots, x_N, x_* \in \mathbf{R}^d
\end{array}
\qquad \text{(P)}
$$

- Variables: $x_0, \ldots, x_N, x_*, f$
- Problem data: $\mathcal{F}_L^{1,1}(\mathbf{R}^d), \mathcal{A}, R, N$

**Difficulty**: Optimization problem (P) is abstract, hard and infinite dimensional

**Approach**: Relax constraints in (P), reduce and reformulate as tractable finite dimensional optimization problem

**Note**: Relaxing constraints in (P) may increase the maximum value. Sometimes relaxing constraints does not increase the maximum value and gives tight bounds on the performance of $\mathcal{A}$

# Performance estimation problems - The problem

$$
\begin{array}{ll}
\text{maximize} & f(x_N) - f(x_*) \\
\text{subject to} & f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d), \\
& x_{i+1} = \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right),\ i = 0, \ldots, N-1, \\
& x_* \in X_*(f), \\
& \|x_* - x_0\|_2 \le R, \\
& x_0, \ldots, x_N, x_* \in \mathbf{R}^d
\end{array}
\qquad \text{(P)}
$$

- Variables: $x_0, \ldots, x_N, x_*, f$
- Problem data: $\mathcal{F}_L^{1,1}(\mathbf{R}^d), \mathcal{A}, R, N$

**Difficulty**: Optimization problem (P) is abstract, hard and infinite dimensional

**Approach**: Relax constraints in (P), reduce and reformulate as tractable finite dimensional optimization problem

**Note**: Relaxing constraints in (P) may increase the maximum value. Sometimes relaxing constraints does not increase the maximum value and gives tight bounds on the performance of $\mathcal{A}$

# Performance estimation problems - The gradient method

For simplicity, we illustrate the methodology on gradient decent:

**Gradient decent (GD) with constant step-size**

- Pick $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$, $N \in \mathbf{N}$, $x_0 \in \mathbf{R}^d$ and $h > 0$
- For $i = 0, \ldots, N-1$, let

$$\begin{aligned} x_{i+1} &= \mathcal{A}\left(\{x_j\}_{j=0}^{i}, \{f(x_j)\}_{j=0}^{i}, \{\nabla f(x_j)\}_{j=0}^{i}\right) \\ &= x_i - \frac{h}{L}\nabla f(x_i) \end{aligned}$$

# Performance estimation problems - The gradient method

For GD, (P) becomes

$$
\begin{array}{ll}
\text{maximize} & f(x_N) - f(x_*) \\
\text{subject to} & f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d), \\
& x_{i+1} = x_i - \frac{h}{L} \nabla f(x_i),\ i = 0, \ldots, N-1, \\
& x_* \in X_*(f), \\
& \|x_* - x_0\|_2 \le R, \\
& x_0, \ldots, x_N, x_* \in \mathbf{R}^d
\end{array}
\tag{P-GD}
$$

# Performance estimation problems - The gradient method
## A property

Property for functions in $\mathcal{F}_L^{1,1}(\mathbf{R}^d)$, e.g. see Nesterov (2018, Theorem 2.1.5)

**Proposition 1**

Suppose that $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$. Then

$$\frac{1}{2L}\left\|\nabla f(x) - \nabla f(y)\right\|_2^2 \leq f(x) - f(y) - \left\langle \nabla f(y), x - y \right\rangle,$$

for all $x, y \in \mathbf{R}^d$.

- Hence, know that

$$\frac{1}{2L}\left\|\nabla f(x_i) - \nabla f(x_j)\right\|_2^2 \leq f(x_i) - f(x_j) - \left\langle \nabla f(x_j), x_i - x_j \right\rangle, \; i, j = 0, \ldots, N, * \qquad (1)$$

- **Idea:** In (P-GD), drop the constraint that $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$, but keep (1). Moreover, replace function and gradient evaluations with variables, i.e.

$$\begin{aligned} f_i &:= f(x_i), \quad i = 0, \ldots, N, *, \\ g_i &:= \nabla f(x_i), \quad i = 0, \ldots, N, *. \end{aligned}$$

  Also, drop $x_* \in X_*(f)$, but keep $g_* = 0$. This gives a relaxation of (P-GD) (and may increase the maximum value). See the next slide

# Performance estimation problems - The gradient method
# A property

Property for functions in $\mathcal{F}_L^{1,1}(\mathbf{R}^d)$, e.g. see Nesterov (2018, Theorem 2.1.5)

**Proposition 1**

Suppose that $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$. Then

$$\frac{1}{2L} \left\| \nabla f(x) - \nabla f(y) \right\|_2^2 \leq f(x) - f(y) - \langle \nabla f(y), x - y \rangle ,$$

for all $x, y \in \mathbf{R}^d$.

- Hence, know that

$$\frac{1}{2L} \left\| \nabla f(x_i) - \nabla f(x_j) \right\|_2^2 \leq f(x_i) - f(x_j) - \langle \nabla f(x_j), x_i - x_j \rangle , \; i, j = 0, \ldots, N, * \qquad (1)$$

- **Idea:** In (P-GD), drop the constraint that $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$, but keep (1). Moreover, replace function and gradient evaluations with variables, i.e.

$$f_i := f(x_i), \quad i = 0, \ldots, N, *,$$
$$g_i := \nabla f(x_i), \quad i = 0, \ldots, N, *.$$

Also, drop $x_* \in X_*(f)$, but keep $g_* = 0$. This gives a relaxation of (P-GD) (and may increase the maximum value). See the next slide

# Performance estimation problems - The gradient method
# A property

Property for functions in $\mathcal{F}_L^{1,1}(\mathbf{R}^d)$, e.g. see Nesterov (2018, Theorem 2.1.5)

## Proposition 1

Suppose that $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$. Then

$$\frac{1}{2L} \left\|\nabla f(x) - \nabla f(y)\right\|_2^2 \leq f(x) - f(y) - \left\langle \nabla f(y), x - y \right\rangle,$$

for all $x, y \in \mathbf{R}^d$.

- Hence, know that

$$\frac{1}{2L} \left\|\nabla f(x_i) - \nabla f(x_j)\right\|_2^2 \leq f(x_i) - f(x_j) - \left\langle \nabla f(x_j), x_i - x_j \right\rangle, \ i, j = 0, \ldots, N, * \qquad (1)$$

- **Idea:** In (P-GD), drop the constraint that $f \in \mathcal{F}_L^{1,1}(\mathbf{R}^d)$, but keep (1). Moreover, replace function and gradient evaluations with variables, i.e.

$$\begin{aligned} f_i &:= f(x_i), \quad i = 0, \ldots, N, *, \\ g_i &:= \nabla f(x_i), \quad i = 0, \ldots, N, *. \end{aligned}$$

  Also, drop $x_* \in X_*(f)$, but keep $g_* = 0$. This gives a relaxation of (P-GD) (and may increase the maximum value). See the next slide

# Performance estimation problems - The gradient method
# Relaxed PEP

$$
\begin{aligned}
\text{maximize} \quad & f_N - f_* \\
\text{subject to} \quad & \frac{1}{2L}\left\|g_i - g_j\right\|_2^2 \leq f_i - f_j - \langle g_j, x_i - x_j \rangle\,,\ i,j = 0,\ldots,N,*, \\
& x_{i+1} = x_i - \frac{h}{L} g_i,\ i = 0,\ldots,N-1, \\
& \left\|x_* - x_0\right\|_2 \leq R, \\
& g_* = 0, \\
& x_0,\ldots,x_N, x_* \in \mathbf{R}^d, \\
& f_0,\ldots,f_N, f_* \in \mathbf{R}, \\
& g_0,\ldots,g_N, g_* \in \mathbf{R}^d
\end{aligned}
$$

# Performance estimation problems - The gradient method Rewriting the relaxed PEP

Using standard tricks in the optimization literature, the relaxed PEP can be written as:

$$
\begin{array}{ll}
\text{maximize} & LR^2\delta_N \\
\text{subject to} & \operatorname{tr}\left(G^T A_{i,j} G\right) \le \delta_i - \delta_j,\ 0 \le i < j \le N, \\
& \operatorname{tr}\left(G^T B_{i,j} G\right) \le \delta_i - \delta_j,\ 0 \le j < i \le N, \\
& \operatorname{tr}\left(G^T C_i G\right) \le \delta_i,\ i = 0, \ldots, N, \\
& \operatorname{tr}\left(G^T D_i G + v u_i^T G\right) \le -\delta_i,\ i = 0, \ldots, N, \\
& \delta \in \mathbf{R}^{N+1}, \\
& G \in \mathbf{R}^{(N+1)\times d}
\end{array} \tag{G}
$$

for some matrices $A_{i,j}, B_{i,j}, C_i, D_i \in \mathbf{S}^{N+1}$ and any unit vector $v \in \mathbf{R}^d$

- (G) is a so-called non-homogeneous *quadratic matrix program* (Beck, 2007)
- Proceed by relaxing (G) by dropping some of the constraints. See the next slide

# Performance estimation problems - The gradient method
## Rewriting the relaxed PEP

Using standard tricks in the optimization literature, the relaxed PEP can be written as:

$$
\begin{aligned}
\text{maximize} \quad & LR^2\delta_N \\
\text{subject to} \quad & \operatorname{tr}\left(G^T A_{i,j} G\right) \le \delta_i - \delta_j,\ 0 \le i < j \le N, \\
& \operatorname{tr}\left(G^T B_{i,j} G\right) \le \delta_i - \delta_j,\ 0 \le j < i \le N, \\
& \operatorname{tr}\left(G^T C_i G\right) \le \delta_i,\ i = 0, \ldots, N, \\
& \operatorname{tr}\left(G^T D_i G + v u_i^T G\right) \le -\delta_i,\ i = 0, \ldots, N, \\
& \delta \in \mathbf{R}^{N+1}, \\
& G \in \mathbf{R}^{(N+1)\times d}
\end{aligned} \tag{G}
$$

for some matrices $A_{i,j}, B_{i,j}, C_i, D_i \in \mathbf{S}^{N+1}$ and any unit vector $v \in \mathbf{R}^d$

- (G) is a so-called non-homogeneous *quadratic matrix program* (Beck, 2007)
- Proceed by relaxing (G) by dropping some of the constraints. See the next slide

# Performance estimation problems - The gradient method
# Rewriting the relaxed PEP

Using standard tricks in the optimization literature, the relaxed PEP can be written as:

$$
\begin{array}{ll}
\text{maximize} & LR^2\delta_N \\
\text{subject to} & \operatorname{tr}\left(G^T A_{i,j} G\right) \le \delta_i - \delta_j,\ 0 \le i < j \le N, \\
& \operatorname{tr}\left(G^T B_{i,j} G\right) \le \delta_i - \delta_j,\ 0 \le j < i \le N, \\
& \operatorname{tr}\left(G^T C_i G\right) \le \delta_i,\ i = 0, \ldots, N, \\
& \operatorname{tr}\left(G^T D_i G + v u_i^T G\right) \le -\delta_i,\ i = 0, \ldots, N, \\
& \delta \in \mathbf{R}^{N+1}, \\
& G \in \mathbf{R}^{(N+1)\times d}
\end{array} \tag{G}
$$

for some matrices $A_{i,j}, B_{i,j}, C_i, D_i \in \mathbf{S}^{N+1}$ and any unit vector $v \in \mathbf{R}^d$

- (G) is a so-called non-homogeneous *quadratic matrix program* (Beck, 2007)
- Proceed by relaxing (G) by dropping some of the constraints. See the next slide

# Performance estimation problems - The gradient method
# Twice relaxed PEP

$$
\begin{aligned}
\text{maximize} \quad & LR^2\delta_N \\
\text{subject to} \quad & \operatorname{tr}\left(G^T A_{i-1,i} G\right) \leq \delta_{i-1} - \delta_i,\ i = 1, \ldots, N, \\
& \operatorname{tr}\left(G^T D_i G + v u_i^T G\right) \leq -\delta_i,\ i = 0, \ldots, N, \\
& \delta \in \mathbf{R}^{N+1}, \\
& G \in \mathbf{R}^{(N+1)\times d}
\end{aligned}
\tag{G'}
$$

- Recall that val(P-GD) $\leq$ val(G) $\leq$ val(G'). I.e. (G') is an upper bound on the worst-case performance of GD
- Next, construct a Lagrangian dual problem to (G')

# Performance estimation problems - The gradient method
# Twice relaxed PEP

$$
\begin{aligned}
\text{maximize} \quad & LR^2\delta_N \\
\text{subject to} \quad & \operatorname{tr}\left(G^T A_{i-1,i} G\right) \le \delta_{i-1} - \delta_i,\ i = 1, \ldots, N, \\
& \operatorname{tr}\left(G^T D_i G + v u_i^T G\right) \le -\delta_i,\ i = 0, \ldots, N, \\
& \delta \in \mathbf{R}^{N+1}, \\
& G \in \mathbf{R}^{(N+1)\times d}
\end{aligned} \qquad \text{(G')}
$$

- Recall that val(P-GD) $\le$ val(G) $\le$ val(G'). I.e. (G') is an upper bound on the worst-case performance of GD
- Next, construct a Lagrangian dual problem to (G')

# Performance estimation problems - The gradient method Twice relaxed PEP

$$
\begin{array}{ll}
\text{maximize} & LR^2\delta_N \\
\text{subject to} & \operatorname{tr}\left(G^T A_{i-1,i} G\right) \leq \delta_{i-1} - \delta_i,\ i = 1, \ldots, N, \\
& \operatorname{tr}\left(G^T D_i G + v u_i^T G\right) \leq -\delta_i,\ i = 0, \ldots, N, \\
& \delta \in \mathbf{R}^{N+1}, \\
& G \in \mathbf{R}^{(N+1)\times d}
\end{array}
\tag{G'}
$$

- Recall that val(P-GD) $\leq$ val(G) $\leq$ val(G'). I.e. (G') is an upper bound on the worst-case performance of GD
- Next, construct a Lagrangian dual problem to (G')

# Performance estimation problems - The gradient method
# A dual to (G')

## Lemma 1

Consider (G') for any fixed $h \in \mathbf{R}$ and $L, R > 0$. A Lagrangian dual of (G') is given by the following convex program:

$$
\begin{array}{ll}
\text{minimize} & \frac{1}{2} L R^2 t \\
\text{subject to} & S(\lambda, t) \succeq 0, \\
& \lambda \in \Lambda \subseteq \mathbf{R}^N, \\
& t \in \mathbf{R}
\end{array}
\tag{DG'}
$$

where $\Lambda = \left\{ \lambda \in \mathbf{R}^N \;\middle|\; \lambda_{i+1} \geq \lambda_i, i = 1 \ldots, N-1,\ 1 \geq \lambda_N,\ \lambda_i \geq 0, i = 1, \ldots, N \right\}$,

$$
S(\lambda, t) = \begin{bmatrix} (1-h) S_0 + h S_1 & q \\ q^T & t \end{bmatrix} \in \mathbf{S}^{N+2}, \quad q = (\lambda_1, \lambda_2 - \lambda_1, \ldots, \lambda_N - \lambda_{N-1}, 1 - \lambda_N) \in \mathbf{R}^{N+1},
$$

$$
S_0 = \begin{bmatrix}
2\lambda_1 & -\lambda_1 & & & & \\
-\lambda_1 & 2\lambda_2 & -\lambda_2 & & & \\
& -\lambda_2 & 2\lambda_3 & -\lambda_2 & & \\
& & \ddots & \ddots & \ddots & \\
& & & -\lambda_{N-1} & 2\lambda_N & -\lambda_N \\
& & & & -\lambda_N & 1
\end{bmatrix} \in \mathbf{S}^{N+1},
$$

$$
S_1 = \begin{bmatrix}
2\lambda_1 & \lambda_2 - \lambda_1 & \cdots & \lambda_N - \lambda_{N-1} & 1 - \lambda_N \\
\lambda_2 - \lambda_1 & 2\lambda_2 & & \lambda_N - \lambda_{N-1} & 1 - \lambda_N \\
\vdots & & \ddots & & \vdots \\
\lambda_N - \lambda_{N-1} & \lambda_N - \lambda_{N-1} & & 2\lambda_N & 1 - \lambda_N \\
1 - \lambda_N & 1 - \lambda_N & \cdots & 1 - \lambda_N & 1
\end{bmatrix} \in \mathbf{S}^{N+1}
$$

## Performance estimation problems - The gradient method
## Tight worst-case estimate

- Note that val(P-GD) $\leq$ val(G) $\leq$ val(G') $\leq$ val(DG'). In particular, any feasible point to (DG') will yield an upper bound to (P-GD)

**Theorem 1**

Suppose that $f \in \mathcal{F}_L^{1,1}(R^d)$, $x_* \in X_*(f)$, $R > 0$ and let $\{x_i\}_{i=0}^N$ be generated by GD with $0 < h \leq 1$ such that $\|x_* - x_0\|_2 \leq R$. Then

$$f(x_N) - f(x_*) \leq \frac{LR^2}{4Nh + 2} \tag{2}$$

- *Remark:* The proof follows by finding a feasible point to (DG')

**Theorem 2**

Let $L, R > 0$, $N \in \mathbf{N}$ and $d \in \mathbf{N}$. Then for every $h > 0$, there exists $\phi \in \mathcal{F}_L^{1,1}(R^d)$ and $x_0 \in \mathbf{R}^d$ such that

$$\phi(x_N) - \phi(x_*) = \frac{LR^2}{4Nh + 2}$$

where $x_N$ is the point after $N$ iterations of GD

- *Remark:* In particular, this shows that the bound in (2) is *tight*

# Performance estimation problems - The gradient method
# Tight worst-case estimate

- Note that val(P-GD) $\leq$ val(G) $\leq$ val(G') $\leq$ val(DG'). In particular, any feasible point to (DG') will yield an upper bound to (P-GD)

**Theorem 1**

Suppose that $f \in \mathcal{F}_L^{1,1}(R^d)$, $x_* \in X_*(f)$, $R > 0$ and let $\{x_i\}_{i=0}^N$ be generated by GD with $0 < h \leq 1$ such that $\|x_* - x_0\|_2 \leq R$. Then

$$f(x_N) - f(x_*) \leq \frac{LR^2}{4Nh + 2} \tag{2}$$

- *Remark:* The proof follows by finding a feasible point to (DG')

**Theorem 2**

Let $L, R > 0$, $N \in \mathbf{N}$ and $d \in \mathbf{N}$. Then for every $h > 0$, there exists $\phi \in \mathcal{F}_L^{1,1}(R^d)$ and $x_0 \in \mathbf{R}^d$ such that

$$\phi(x_N) - \phi(x_*) = \frac{LR^2}{4Nh + 2}$$

where $x_N$ is the point after $N$ iterations of GD

- *Remark:* In particular, this shows that the bound in (2) is *tight*

# Performance estimation problems - The gradient method Tight worst-case estimate

- Note that val(P-GD) $\leq$ val(G) $\leq$ val(G') $\leq$ val(DG'). In particular, any feasible point to (DG') will yield an upper bound to (P-GD)

## Theorem 1

Suppose that $f \in \mathcal{F}_L^{1,1}(R^d)$, $x_* \in X_*(f)$, $R > 0$ and let $\{x_i\}_{i=0}^N$ be generated by GD with $0 < h \leq 1$ such that $\|x_* - x_0\|_2 \leq R$. Then

$$f(x_N) - f(x_*) \leq \frac{LR^2}{4Nh + 2} \tag{2}$$

- *Remark:* The proof follows by finding a feasible point to (DG')

## Theorem 2

Let $L, R > 0$, $N \in \mathbf{N}$ and $d \in \mathbf{N}$. Then for every $h > 0$, there exists $\phi \in \mathcal{F}_L^{1,1}(R^d)$ and $x_0 \in \mathbf{R}^d$ such that

$$\phi(x_N) - \phi(x_*) = \frac{LR^2}{4Nh + 2}$$

where $x_N$ is the point after $N$ iterations of GD

- *Remark:* In particular, this shows that the bound in (2) is *tight*

# Performance estimation problems - Extensions in the literature

- Other measures of inaccuracy than $f(x_N) - f(x_*)$:
  - $\|\nabla f(x_N)\|_2^2$
  - $\|x_N - x_*\|_2^2$
  - $\min_{i=0,\ldots,N} f(x_i) - f(x_*)$
  - $\min_{i=0,\ldots,N} \|\nabla f(x_i)\|_2^2$
  - $\min_{i=0,\ldots,N} \|x_i - x_*\|_2^2$
  - Add expectation $\mathbb{E}[\cdot]$ everywhere for stochastic algorithms
- Introduce so-called interpolation/extension conditions for a priori provably tight worst-case bounds. See e.g. Taylor et al. (2017)
- Other function classes $\mathcal{F}$ or even operator classes
- Other classes of algorithms $\mathcal{A}$:
  - Subgradient, Nesterov's method, heavy ball method
  - Proximal point algorithm
  - Projected and proximal gradient, with accelerated/momentum versions
  - Douglas-Rachford/operator splitting ($\leftarrow$ due to Carolina Bergeling and Pontus Giselsson)
  - Conditional gradient (Frank-Wolfe) method
  - Inexact gradient
  - Krasnoselskii-Mann and Halpern fixed-point iterations
  - Mirror descent
  - Stochastic methods: SAG, SAGA, SGD, etc.

# Performance estimation problems - Extensions in the literature

- Other measures of inaccuracy than $f(x_N) - f(x_*)$:
  - $\|\nabla f(x_N)\|_2^2$
  - $\|x_N - x_*\|_2^2$
  - $\min_{i=0,\ldots,N} f(x_i) - f(x_*)$
  - $\min_{i=0,\ldots,N} \|\nabla f(x_i)\|_2^2$
  - $\min_{i=0,\ldots,N} \|x_i - x_*\|_2^2$
  - Add expectation $\mathbb{E}[\cdot]$ everywhere for stochastic algorithms
- Introduce so-called interpolation/extension conditions for a priori provably tight worst-case bounds. See e.g. Taylor et al. (2017)
- Other function classes $\mathcal{F}$ or even operator classes
- Other classes of algorithms $\mathcal{A}$:
  - Subgradient, Nesterov's method, heavy ball method
  - Proximal point algorithm
  - Projected and proximal gradient, with accelerated/momentum versions
  - Douglas-Rachford/operator splitting ($\leftarrow$ due to Carolina Bergeling and Pontus Giselsson)
  - Conditional gradient (Frank-Wolfe) method
  - Inexact gradient
  - Krasnoselskii-Mann and Halpern fixed-point iterations
  - Mirror descent
  - Stochastic methods: SAG, SAGA, SGD, etc.

# Performance estimation problems - Extensions in the literature

- Other measures of inaccuracy than $f(x_N) - f(x_*)$:
  - $\|\nabla f(x_N)\|_2^2$
  - $\|x_N - x_*\|_2^2$
  - $\min_{i=0,\dots,N} f(x_i) - f(x_*)$
  - $\min_{i=0,\dots,N} \|\nabla f(x_i)\|_2^2$
  - $\min_{i=0,\dots,N} \|x_i - x_*\|_2^2$
  - Add expectation $\mathbb{E}[\cdot]$ everywhere for stochastic algorithms
- Introduce so-called interpolation/extension conditions for a priori provably tight worst-case bounds. See e.g. Taylor et al. (2017)
- Other function classes $\mathcal{F}$ or even operator classes
- Other classes of algorithms $\mathcal{A}$:
  - Subgradient, Nesterov's method, heavy ball method
  - Proximal point algorithm
  - Projected and proximal gradient, with accelerated/momentum versions
  - Douglas-Rachford/operator splitting ($\leftarrow$ due to Carolina Bergeling and Pontus Giselsson)
  - Conditional gradient (Frank-Wolfe) method
  - Inexact gradient
  - Krasnoselskii-Mann and Halpern fixed-point iterations
  - Mirror descent
  - Stochastic methods: SAG, SAGA, SGD, etc.

# Performance estimation problems - Extensions in the literature

- Other measures of inaccuracy than $f(x_N) - f(x_*)$:
  - $\|\nabla f(x_N)\|_2^2$
  - $\|x_N - x_*\|_2^2$
  - $\min_{i=0,\ldots,N} f(x_i) - f(x_*)$
  - $\min_{i=0,\ldots,N} \|\nabla f(x_i)\|_2^2$
  - $\min_{i=0,\ldots,N} \|x_i - x_*\|_2^2$
  - Add expectation $\mathbb{E}[\cdot]$ everywhere for stochastic algorithms
- Introduce so-called interpolation/extension conditions for a priori provably tight worst-case bounds. See e.g. Taylor et al. (2017)
- Other function classes $\mathcal{F}$ or even operator classes
- Other classes of algorithms $\mathcal{A}$:
  - Subgradient, Nesterov's method, heavy ball method
  - Proximal point algorithm
  - Projected and proximal gradient, with accelerated/momentum versions
  - Douglas-Rachford/operator splitting ($\leftarrow$ due to Carolina Bergeling and Pontus Giselsson)
  - Conditional gradient (Frank-Wolfe) method
  - Inexact gradient
  - Krasnoselskii-Mann and Halpern fixed-point iterations
  - Mirror descent
  - Stochastic methods: SAG, SAGA, SGD, etc.

# Performance estimation problems - Related line of work
# IQCs

- A technique in the robust control literature is to use *integral quadratic constraint*s (IQCs) to capture features of the behavior of partially known components
- Can be used to study optimization algorithms described by a linear system interconnected in feedback to an (possibly uncertain) nonlinear system representing the gradient
- Lessard et al. (2016) used this to study the rate of convergence of optimization algorithms
- Several papers in this direction followed (e.g one by Anders Rantzer)
- Benefit:
  - Fast/scales well: Bisection search over a small LMI
- Limitation:
  - Considers only asymptotic rates
  - The rates are not necessarily tight, i.e. provides only sufficiency

# Performance estimation problems - Related line of work IQCs

- A technique in the robust control literature is to use *integral quadratic constraint*s (IQCs) to capture features of the behavior of partially known components
- Can be used to study optimization algorithms described by a linear system interconnected in feedback to an (possibly uncertain) nonlinear system representing the gradient
- Lessard et al. (2016) used this to study the rate of convergence of optimization algorithms
- Several papers in this direction followed (e.g one by Anders Rantzer)
- Benefit:
  - Fast/scales well: Bisection search over a small LMI
- Limitation:
  - Considers only asymptotic rates
  - The rates are not necessarily tight, i.e. provides only sufficiency

# Performance estimation problems - Related line of work IQCs

- A technique in the robust control literature is to use *integral quadratic constraint*s (IQCs) to capture features of the behavior of partially known components
- Can be used to study optimization algorithms described by a linear system interconnected in feedback to an (possibly uncertain) nonlinear system representing the gradient
- Lessard et al. (2016) used this to study the rate of convergence of optimization algorithms
- Several papers in this direction followed (e.g one by Anders Rantzer)
- Benefit:
  - Fast/scales well: Bisection search over a small LMI
- Limitation:
  - Considers only asymptotic rates
  - The rates are not necessarily tight, i.e. provides only sufficiency

# Performance estimation problems - Related line of work IQCs

- A technique in the robust control literature is to use *integral quadratic constraint*s (IQCs) to capture features of the behavior of partially known components
- Can be used to study optimization algorithms described by a linear system interconnected in feedback to an (possibly uncertain) nonlinear system representing the gradient
- Lessard et al. (2016) used this to study the rate of convergence of optimization algorithms
- Several papers in this direction followed (e.g one by Anders Rantzer)
- Benefit:
  - Fast/scales well: Bisection search over a small LMI
- Limitation:
  - Considers only asymptotic rates
  - The rates are not necessarily tight, i.e. provides only sufficiency

# Performance estimation problems - Related line of work IQCs

- A technique in the robust control literature is to use *integral quadratic constraint*s (IQCs) to capture features of the behavior of partially known components
- Can be used to study optimization algorithms described by a linear system interconnected in feedback to an (possibly uncertain) nonlinear system representing the gradient
- Lessard et al. (2016) used this to study the rate of convergence of optimization algorithms
- Several papers in this direction followed (e.g one by Anders Rantzer)
- Benefit:
  - Fast/scales well: Bisection search over a small LMI
- Limitation:
  - Considers only asymptotic rates
  - The rates are not necessarily tight, i.e. provides only sufficiency

# Performance estimation problems - Related line of work
# IQCs

- A technique in the robust control literature is to use *integral quadratic constraint*s (IQCs) to capture features of the behavior of partially known components
- Can be used to study optimization algorithms described by a linear system interconnected in feedback to an (possibly uncertain) nonlinear system representing the gradient
- Lessard et al. (2016) used this to study the rate of convergence of optimization algorithms
- Several papers in this direction followed (e.g one by Anders Rantzer)
- Benefit:
  - Fast/scales well: Bisection search over a small LMI
- Limitation:
  - Considers only asymptotic rates
  - The rates are not necessarily tight, i.e. provides only sufficiency

# Performance estimation problems - Related line of work
## IQCs

- A technique in the robust control literature is to use *integral quadratic constraint*s (IQCs) to capture features of the behavior of partially known components
- Can be used to study optimization algorithms described by a linear system interconnected in feedback to an (possibly uncertain) nonlinear system representing the gradient
- Lessard et al. (2016) used this to study the rate of convergence of optimization algorithms
- Several papers in this direction followed (e.g one by Anders Rantzer)
- Benefit:
  - Fast/scales well: Bisection search over a small LMI
- Limitation:
  - Considers only asymptotic rates
  - The rates are not necessarily tight, i.e. provides only sufficiency

# Performance estimation problems - Related line of work
# IQCs

- A technique in the robust control literature is to use *integral quadratic constraint*s (IQCs) to capture features of the behavior of partially known components
- Can be used to study optimization algorithms described by a linear system interconnected in feedback to an (possibly uncertain) nonlinear system representing the gradient
- Lessard et al. (2016) used this to study the rate of convergence of optimization algorithms
- Several papers in this direction followed (e.g one by Anders Rantzer)
- Benefit:
  - Fast/scales well: Bisection search over a small LMI
- Limitation:
  - Considers only asymptotic rates
  - The rates are not necessarily tight, i.e. provides only sufficiency

# Performance estimation problems - Related line of work
# IQCs

- A technique in the robust control literature is to use *integral quadratic constraint*s (IQCs) to capture features of the behavior of partially known components
- Can be used to study optimization algorithms described by a linear system interconnected in feedback to an (possibly uncertain) nonlinear system representing the gradient
- Lessard et al. (2016) used this to study the rate of convergence of optimization algorithms
- Several papers in this direction followed (e.g one by Anders Rantzer)
- Benefit:
  - Fast/scales well: Bisection search over a small LMI
- Limitation:
  - Considers only asymptotic rates
  - The rates are not necessarily tight, i.e. provides only sufficiency

# Performance estimation problems - What I'm looking at

Main idea:

- Use interpolation conditions from PEP framework
- Use algorithm formulation and Lyapunov functions as in IQC framework
- Goal is to provide conditions for tight worst-case performance in the combined framework. At the very least conditions for good estimates of the worst-case performance
- Secondary goal would be design optimization algorithms that are optimal w.r.t. these conditions

Approach:

- Algorithm $\mathcal{A}$: Linear system with a nonlinear feedback given by some operator
- Operator class: Has interpolation condition that only involves quadratic inequalities
- Lyapunov functions: Quadratic anzats

# Performance estimation problems - What I'm looking at

Main idea:

- Use interpolation conditions from PEP framework
- Use algorithm formulation and Lyapunov functions as in IQC framework
- Goal is to provide conditions for tight worst-case performance in the combined framework. At the very least conditions for good estimates of the worst-case performance
- Secondary goal would be design optimization algorithms that are optimal w.r.t. these conditions

Approach:

- Algorithm $\mathcal{A}$: Linear system with a nonlinear feedback given by some operator
- Operator class: Has interpolation condition that only involves quadratic inequalities
- Lyapunov functions: Quadratic anzats

# Performance estimation problems - What I'm looking at

Main idea:

- Use interpolation conditions from PEP framework
- Use algorithm formulation and Lyapunov functions as in IQC framework
- Goal is to provide conditions for tight worst-case performance in the combined framework. At the very least conditions for good estimates of the worst-case performance
- Secondary goal would be design optimization algorithms that are optimal w.r.t. these conditions

Approach:

- Algorithm $\mathcal{A}$: Linear system with a nonlinear feedback given by some operator
- Operator class: Has interpolation condition that only involves quadratic inequalities
- Lyapunov functions: Quadratic anzats

# Performance estimation problems - What I'm looking at

Main idea:

- Use interpolation conditions from PEP framework
- Use algorithm formulation and Lyapunov functions as in IQC framework
- Goal is to provide conditions for tight worst-case performance in the combined framework. At the very least conditions for good estimates of the worst-case performance
- Secondary goal would be design optimization algorithms that are optimal w.r.t. these conditions

Approach:

- Algorithm $\mathcal{A}$: Linear system with a nonlinear feedback given by some operator
- Operator class: Has interpolation condition that only involves quadratic inequalities
- Lyapunov functions: Quadratic anzats

# Performance estimation problems - What I'm looking at

Main idea:

- Use interpolation conditions from PEP framework
- Use algorithm formulation and Lyapunov functions as in IQC framework
- Goal is to provide conditions for tight worst-case performance in the combined framework. At the very least conditions for good estimates of the worst-case performance
- Secondary goal would be design optimization algorithms that are optimal w.r.t. these conditions

Approach:

- Algorithm $\mathcal{A}$: Linear system with a nonlinear feedback given by some operator
- Operator class: Has interpolation condition that only involves quadratic inequalities
- Lyapunov functions: Quadratic anzats

# Performance estimation problems - What I'm looking at

Main idea:

- Use interpolation conditions from PEP framework
- Use algorithm formulation and Lyapunov functions as in IQC framework
- Goal is to provide conditions for tight worst-case performance in the combined framework. At the very least conditions for good estimates of the worst-case performance
- Secondary goal would be design optimization algorithms that are optimal w.r.t. these conditions

Approach:

- Algorithm $\mathcal{A}$: Linear system with a nonlinear feedback given by some operator
- Operator class: Has interpolation condition that only involves quadratic inequalities
- Lyapunov functions: Quadratic anzats

# Performance estimation problems - What I'm looking at

Main idea:

- Use interpolation conditions from PEP framework
- Use algorithm formulation and Lyapunov functions as in IQC framework
- Goal is to provide conditions for tight worst-case performance in the combined framework. At the very least conditions for good estimates of the worst-case performance
- Secondary goal would be design optimization algorithms that are optimal w.r.t. these conditions

Approach:

- Algorithm $\mathcal{A}$: Linear system with a nonlinear feedback given by some operator
- Operator class: Has interpolation condition that only involves quadratic inequalities
- Lyapunov functions: Quadratic anzats

# Performance estimation problems - What I'm looking at

Main idea:

- Use interpolation conditions from PEP framework
- Use algorithm formulation and Lyapunov functions as in IQC framework
- Goal is to provide conditions for tight worst-case performance in the combined framework. At the very least conditions for good estimates of the worst-case performance
- Secondary goal would be design optimization algorithms that are optimal w.r.t. these conditions

Approach:

- Algorithm $\mathcal{A}$: Linear system with a nonlinear feedback given by some operator
- Operator class: Has interpolation condition that only involves quadratic inequalities
- Lyapunov functions: Quadratic anzats

# Performance estimation problems - What I'm looking at

Main idea:

- Use interpolation conditions from PEP framework
- Use algorithm formulation and Lyapunov functions as in IQC framework
- Goal is to provide conditions for tight worst-case performance in the combined framework. At the very least conditions for good estimates of the worst-case performance
- Secondary goal would be design optimization algorithms that are optimal w.r.t. these conditions

Approach:

- Algorithm $\mathcal{A}$: Linear system with a nonlinear feedback given by some operator
- Operator class: Has interpolation condition that only involves quadratic inequalities
- Lyapunov functions: Quadratic anzats

# References I


Beck, A. (2007), 'Quadratic matrix programming', *SIAM Journal on Optimization* **17**(4), 1224–1238.

Drori, Y. and Teboulle, M. (2014), 'Performance of first-order methods for smooth convex minimization: a novel approach.', *Mathematical Programming* **145**(1/2), 451 – 482.

Lessard, L., Recht, B. and Packard, A. (2016), 'Analysis and design of optimization algorithms via integral quadratic constraints', *SIAM Journal on Optimization* **26**(1), 57–95.
**URL:** *https://doi.org/10.1137/15M1009597*

Nesterov, Y. (2018), *Lectures on Convex Optimization.*, Springer International Publishing.

Taylor, A., Hendrickx, J. and Glineur, F. (2017), 'Smooth strongly convex interpolation and exact worst-case performance of first-order methods.', *Mathematical Programming* **161**(1/2), 307 – 345.