Quantum Algorithm for Robust Optimizaiton via Stochastic-Gradient Online Learning

Debbie Lim, João F. Doriguello, Patrick Rebentrost

Centre for Quantum Technologies

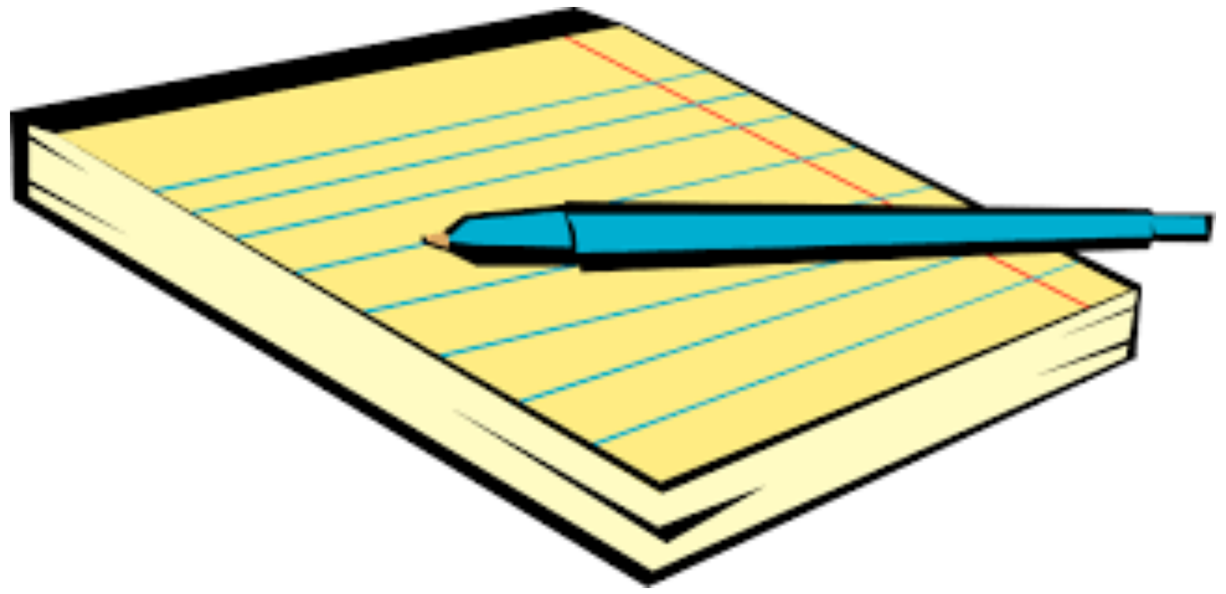
NUS National University of Singapore

Latvian Quantum Initiative

Funded by the European Union NextGenerationEU

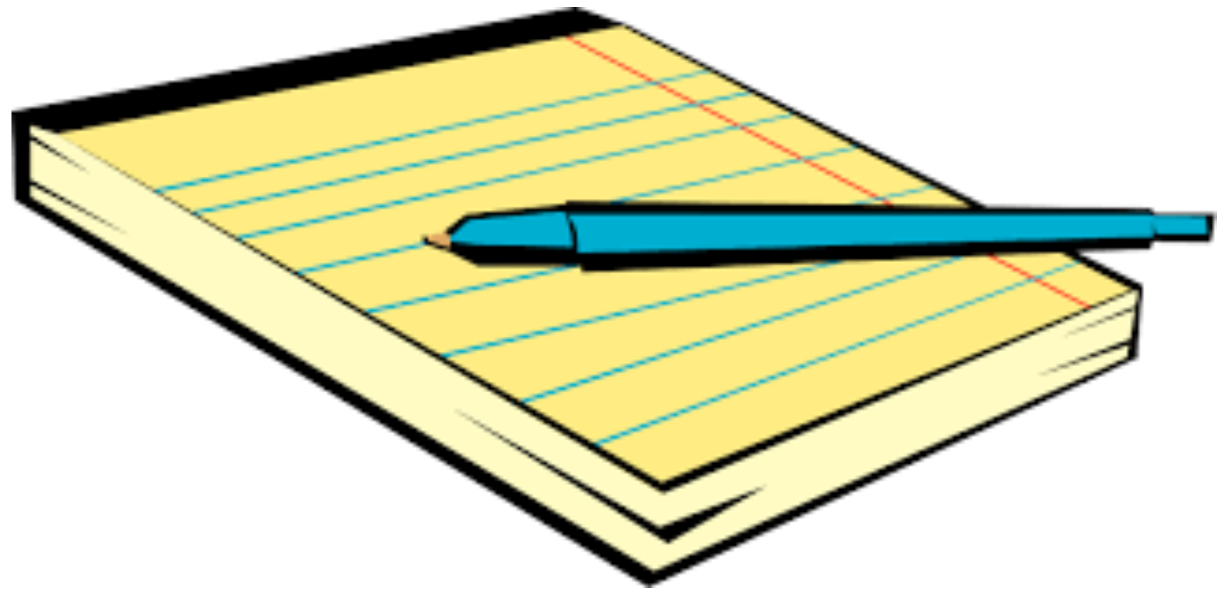
2027 National Development Plan

Outline



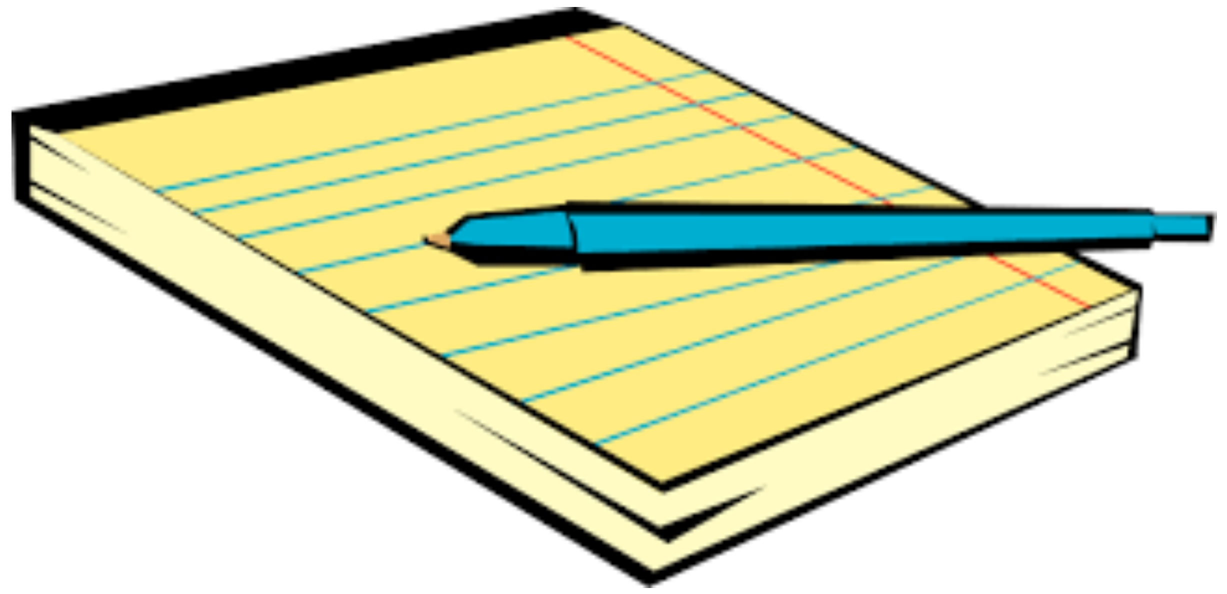
Outline

- Robust convex optimization



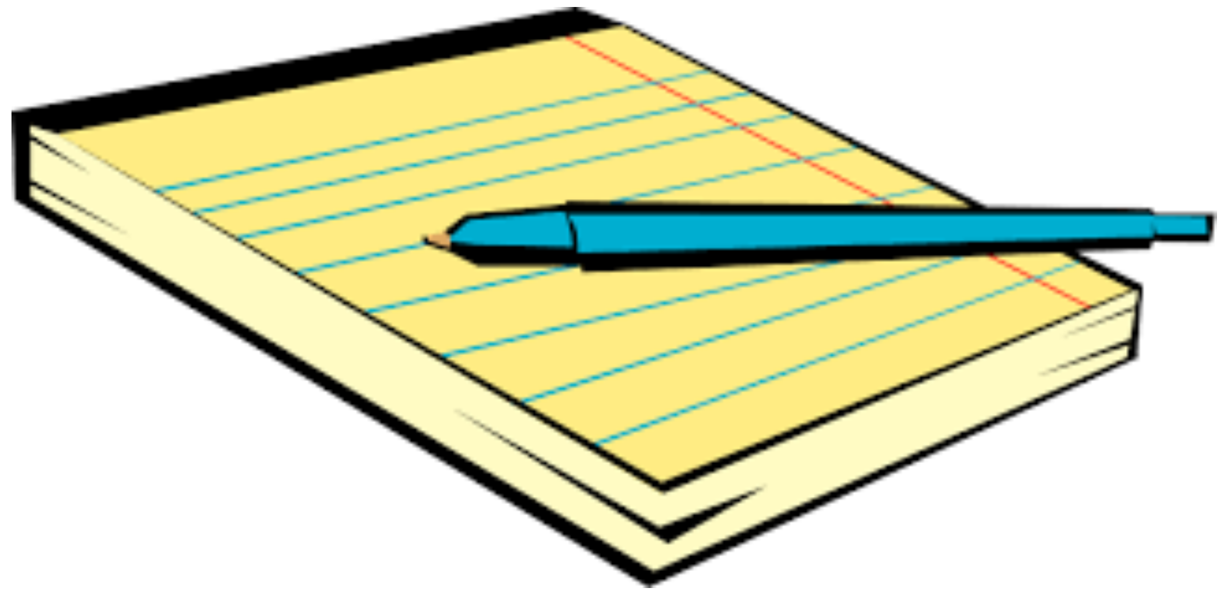
Outline

- Robust convex optimization
- Motivation



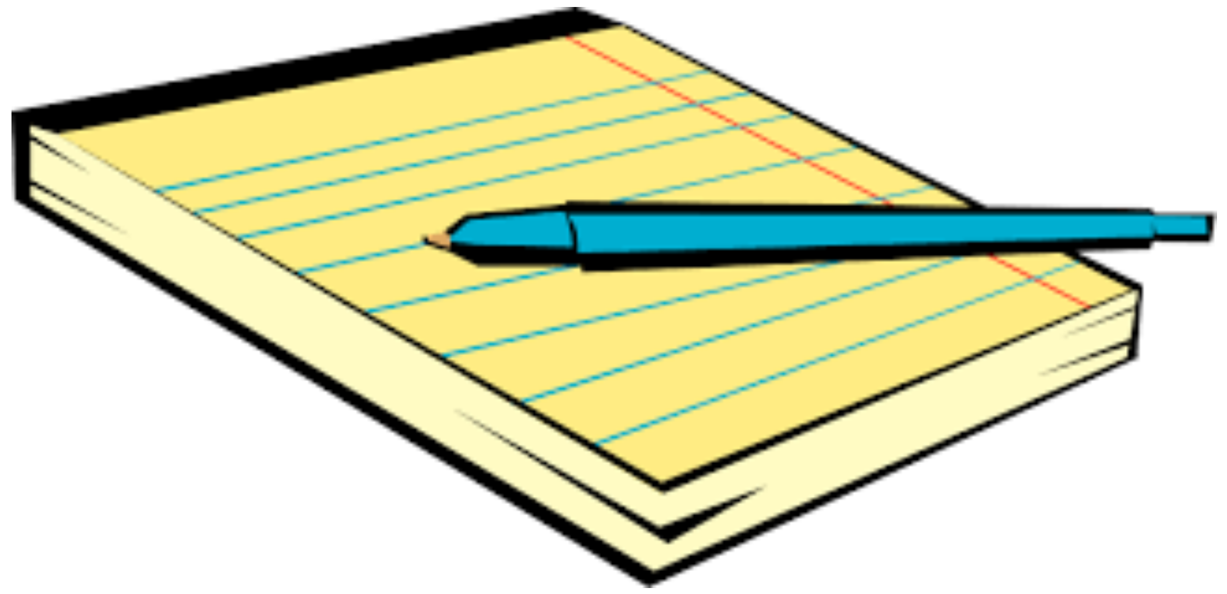
Outline

- Robust convex optimization
- Motivation
- Ben-Tal *et.al*'s dual-subgradient robust optimizaiton algorithm



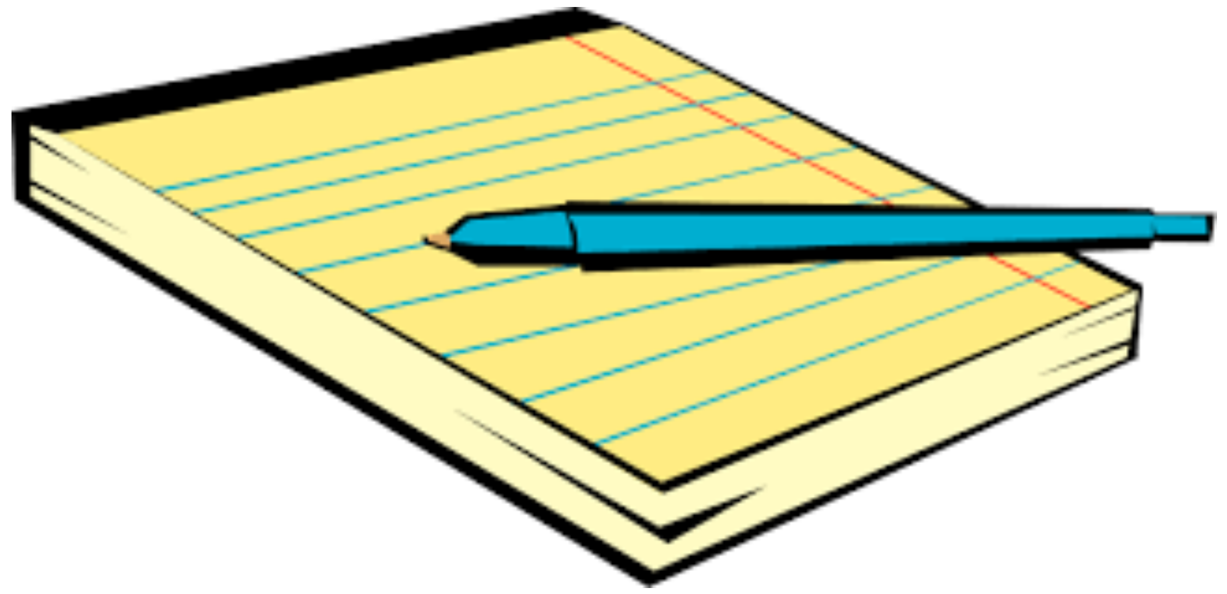
Outline

- Robust convex optimization
- Motivation
- Ben-Tal *et.al*'s dual-subgradient robust optimizaiton algorithm
- Achieving a speedup



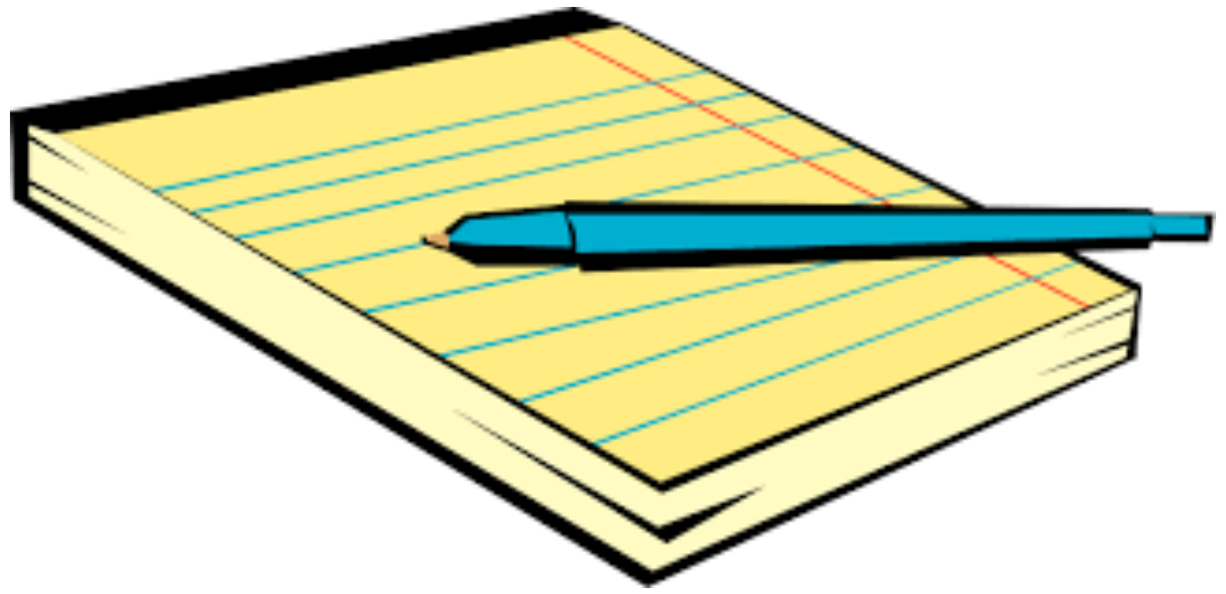
Outline

- Robust convex optimization
- Motivation
- Ben-Tal *et.al*’s dual-subgradient robust optimizaiton algorithm
- Achieving a speedup
- Applications



Outline

- Robust convex optimization
- Motivation
- Ben-Tal *et.al*'s dual-subgradient robust optimizaiton algorithm
- Achieving a speedup
- Applications
- Conclusion



Robust convex optimization

Convex optimization

$$
\begin{aligned}
\textbf{minimize} \quad & f_0(x) & \\
\textbf{subject to} \quad & f_i(x, u_i) \le 0, & \forall i \in [m] \\
& x \in \mathcal{D} &
\end{aligned}
$$

- $u_1, \cdots, u_m \in \mathbb{R}^d$ are fixed parameters.
- $f_0, \cdots, f_m$ are convex in x.
- Domain $\mathcal{D} \subseteq \mathbb{R}^n$ is convex.

Robust convex optimization

$$
\begin{array}{rrll}
\textbf{minimize} & f_0(x) & & \\
\textbf{subject to} & f_i(x, u_i) \le 0, & \forall u_i \in \mathcal{U}, & \forall i \in [m] \\
 & x \in \mathcal{D} & &
\end{array}
$$

- $f_0, \cdots, f_m$ are convex in x.
- $f_1, \cdots, f_m$ are concave in $u_1, \cdots, u_m$.
- Domain $\mathcal{D} \subseteq \mathbb{R}^n$ and uncertainty set $\mathcal{U} \subseteq \mathbb{R}^d$ are convex.

Optimization problem $\rightarrow$ feasibility problem

Optimization problem $\to$ feasibility problem

Use binary search over its optimal values

Optimization problem → feasibility problem

Use binary search over its optimal values

$$\exists? \qquad x \in \mathcal{D} : f_i(x, u_i) \leq 0 \qquad \forall u_i \in \mathcal{U}, \qquad \forall i \in [m]$$

Motivation

Why robust optimization

Why robust optimization

- Addresses the issue of data inaccuracy

Why robust optimization

- Addresses the issue of data inaccuracy
- First introduced by Ben-Tal and Nemirovski in 1998

Why robust optimization

- Addresses the issue of data inaccuracy
- First introduced by Ben-Tal and Nemirovski in 1998
- Computational cost for large scale problems can be highly prohibitive

Why robust optimization

- Addresses the issue of data inaccuracy
- First introduced by Ben-Tal and Nemirovski in 1998
- Computational cost for large scale problems can be highly prohibitive
- A meta-algorithm to approximately solve the robust counterpart of a convex optimisation problem, using only an algorithm for the original optimization formulation

Dual-subgradient robust optimization algorithm

Oracle-Based Robust Optimization via Online Learning

Aharon Ben-Tal
Technion
abental@ie.technion.ac.il

Elad Hazan
Technion
ehazan@ie.technion.ac.il

Tomer Koren
Technion
tomerk@technion.ac.il

Shie Mannor
Technion
shie@ee.technion.ac.il

February 27, 2014

Oracles

Oracles

- Noise memory: $|i\rangle|j\rangle|\bar{0}\rangle \to |i\rangle|j\rangle|(u_i)_j\rangle$

Oracles

- Noise memory: $|i\rangle|j\rangle|\bar{0}\rangle \to |i\rangle|j\rangle|(u_i)_j\rangle$
- Subgradient oracle $\mathcal{O}_{\nabla}: |i\rangle|j\rangle|\bar{0}\rangle \to |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$

Oracles

- Noise memory: $|i\rangle|j\rangle|\bar{0}\rangle \to |i\rangle|j\rangle|(u_i)_j\rangle$
- Subgradient oracle $\mathcal{O}_\nabla : |i\rangle|j\rangle|\bar{0}\rangle \to |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathcal{O}_\mathcal{P} : u \to$ argmin$_{q\in\mathcal{U}}\|q-u\|_2$

Oracles

- Noise memory: $|i\rangle|j\rangle|\bar{0}\rangle \rightarrow |i\rangle|j\rangle|(u_i)_j\rangle$
- Subgradient oracle $\mathcal{O}_\nabla : |i\rangle|j\rangle|\bar{0}\rangle \rightarrow |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathcal{O}_\mathcal{P} : u \rightarrow$ argmin$_{q\in\mathcal{U}}\|q-u\|_2$
- Optimization oracle $\mathcal{O}_\epsilon$: Takes $u_1, \cdots, u_m \in \mathcal{U}$ as input. Outputs $x \in \mathcal{D}$ such that

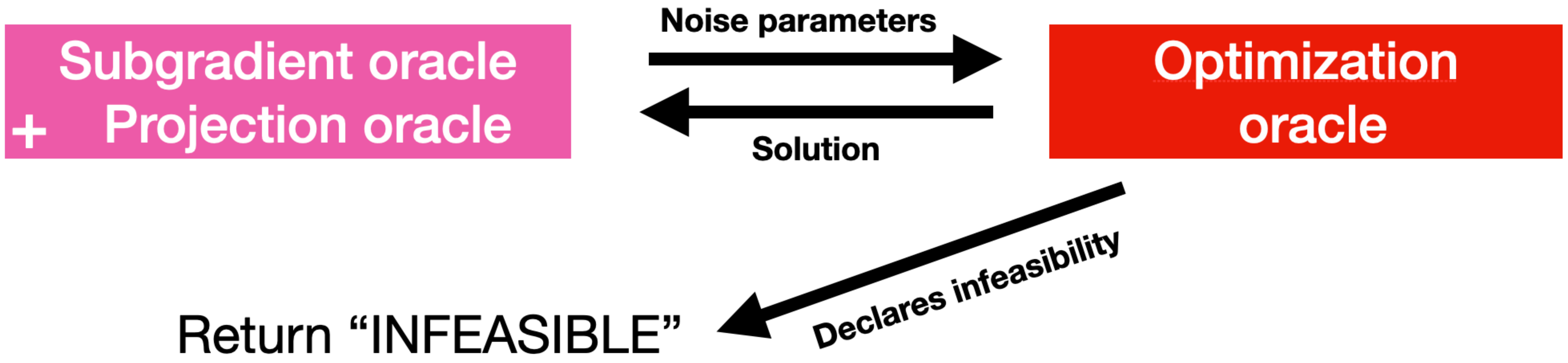
$$f_i\left(x, u_i\right) \leq \epsilon, \forall i \in [m]$$

or returns "INFEASIBLE" if $\nexists x \in \mathcal{D}$ such that

$$f_i\left(x, u_i\right) \leq 0, \forall i \in [m]$$

Ben-Tal *et al.*'s algorithm

- Initialize noise parameters $\left(u_1^{(0)}, \ldots, u_m^{(0)}\right) \in \mathscr{U}^m$ and $x^{(0)} \in \mathscr{D}$ arbitrarily;
- **For $t = 0$ to $T - 1$, do**



- **Output:** $\bar{x} = \dfrac{1}{T} \sum_{t=1}^{T} x^{(t)}$

Ben-Tal *et al.*'s algorithm

- Subgradient oracle $\mathcal{O}_{\nabla} : |i\rangle|j\rangle|\bar{0}\rangle \rightarrow |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathcal{O}_{\mathscr{P}} : u \rightarrow \text{argmin}_{q\in\mathscr{U}}\|q - u\|_2$
- Optimization oracle $\mathcal{O}_{\epsilon}$: Takes $u_1, \cdots, u_m \in \mathscr{U}$ as input. Outputs $x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq \epsilon, \forall i \in [m]$$

or returns "INFEASIBLE" if $\nexists x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq 0, \forall i \in [m]$$

Ben-Tal *et al.*'s algorithm

- Initialize noise parameters $\left(u_1^{(0)}, \ldots, u_m^{(0)}\right) \in \mathcal{U}^m$ and $x^{(0)} \in \mathcal{D}$ arbitrarily;

- Subgradient oracle $\mathcal{O}_\nabla : |i\rangle|j\rangle|\bar{0}\rangle \to |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathcal{O}_\mathcal{P} : u \to \text{argmin}_{q \in \mathcal{U}} \|q - u\|_2$
- Optimization oracle $\mathcal{O}_\epsilon$: Takes $u_1, \cdots, u_m \in \mathcal{U}$ as input. Outputs $x \in \mathcal{D}$ such that

$$f_i\left(x, u_i\right) \leq \epsilon, \forall i \in [m]$$

or returns "INFEASIBLE" if $\nexists x \in \mathcal{D}$ such that

$$f_i\left(x, u_i\right) \leq 0, \forall i \in [m]$$

Ben-Tal *et al.*'s algorithm

- Initialize noise parameters $\left(u_1^{(0)}, \ldots, u_m^{(0)}\right) \in \mathscr{U}^m$ and $x^{(0)} \in \mathscr{D}$ arbitrarily;
- **For** $t = 0$ **to** $T - 1$**, do**

- Subgradient oracle $\mathcal{O}_\nabla : |i\rangle|j\rangle|\bar{0}\rangle \to |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathcal{O}_\mathscr{P} : u \to \text{argmin}_{q\in\mathscr{U}}\|q - u\|_2$
- Optimization oracle $\mathcal{O}_\epsilon$: Takes $u_1, \cdots, u_m \in \mathscr{U}$ as input. Outputs $x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq \epsilon, \forall i \in [m]$$

or returns "INFEASIBLE" if $\nexists x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq 0, \forall i \in [m]$$

Ben-Tal *et al.*'s algorithm

- Initialize noise parameters $\left(u_1^{(0)}, \ldots, u_m^{(0)}\right) \in \mathscr{U}^m$ and $x^{(0)} \in \mathscr{D}$ arbitrarily;
- **For** $t = 0$ **to** $T - 1$**, do**
    - $\nabla_1, \cdots, \nabla_m \leftarrow$ Query $\mathcal{O}_\nabla$ with $x^{(t)}, u_1^{(t)}, \cdots, u_m^{(t)}$

- Subgradient oracle $\mathcal{O}_\nabla : |i\rangle|j\rangle|\bar{0}\rangle \rightarrow |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathcal{O}_\mathscr{P} : u \rightarrow \text{argmin}_{q \in \mathscr{U}} \|q - u\|_2$
- Optimization oracle $\mathcal{O}_\epsilon$: Takes $u_1, \cdots, u_m \in \mathscr{U}$ as input. Outputs $x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq \epsilon, \forall i \in [m]$$

or returns "INFEASIBLE" if $\nexists x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq 0, \forall i \in [m]$$

Ben-Tal *et al.*'s algorithm

- Initialize noise parameters $\left(u_1^{(0)}, \ldots, u_m^{(0)}\right) \in \mathscr{U}^m$ and $x^{(0)} \in \mathscr{D}$ arbitrarily;
- **For $t = 0$ to $T - 1$, do**
  - $\nabla_1, \cdots, \nabla_m \leftarrow$ Query $\mathcal{O}_\nabla$ with $x^{(t)}, u_1^{(t)}, \cdots, u_m^{(t)}$
  - **For $i = 1, \cdots, m$, d0**

- Subgradient oracle $\mathcal{O}_\nabla : |i\rangle|j\rangle|\bar{0}\rangle \to |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathcal{O}_\mathscr{P} : u \to \text{argmin}_{q \in \mathscr{U}} \|q - u\|_2$
- Optimization oracle $\mathcal{O}_\epsilon$: Takes $u_1, \cdots, u_m \in \mathscr{U}$ as input. Outputs $x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq \epsilon, \forall i \in [m]$$

or returns "INFEASIBLE" if $\nexists x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq 0, \forall i \in [m]$$

Ben-Tal *et al.*'s algorithm

- Initialize noise parameters $\left(u_1^{(0)}, \ldots, u_m^{(0)}\right) \in \mathscr{U}^m$ and $x^{(0)} \in \mathscr{D}$ arbitrarily;
- **For** $t = 0$ **to** $T-1$**, do**
  - $\nabla_1, \cdots, \nabla_m \leftarrow$ Query $\mathscr{O}_\nabla$ with $x^{(t)}, u_1^{(t)}, \cdots, u_m^{(t)}$
  - **For** $i = 1, \cdots, m$**, d0**
    - $u_i^{(t+1)} = \mathscr{O}_{\mathscr{P}}\left(u_i^{(t)} + \eta \nabla_i^{(t)}\right)$
  - **EndFor**

- Subgradient oracle $\mathscr{O}_\nabla : |i\rangle|j\rangle|\bar{0}\rangle \rightarrow |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathscr{O}_{\mathscr{P}} : u \rightarrow \text{argmin}_{q\in\mathscr{U}}\|q-u\|_2$
- Optimization oracle $\mathscr{O}_\epsilon$: Takes $u_1, \cdots, u_m \in \mathscr{U}$ as input. Outputs $x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq \epsilon, \forall i \in [m]$$

or returns "INFEASIBLE" if $\nexists x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq 0, \forall i \in [m]$$

Ben-Tal *et al.*'s algorithm

- Initialize noise parameters $\left(u_1^{(0)}, \ldots, u_m^{(0)}\right) \in \mathscr{U}^m$ and $x^{(0)} \in \mathscr{D}$ arbitrarily;
- **For** $t = 0$ **to** $T - 1$**, do**
    - $\nabla_1, \cdots, \nabla_m \leftarrow$ Query $\mathcal{O}_\nabla$ with $x^{(t)}, u_1^{(t)}, \cdots, u_m^{(t)}$
    - **For** $i = 1, \cdots, m$**, d0**
        - $u_i^{(t+1)} = \mathcal{O}_{\mathscr{P}}\left(u_i^{(t)} + \eta \nabla_i^{(t)}\right)$
    - **EndFor**
    - $x^{(t+1)} = \mathcal{O}_\epsilon\left(u_1^{(t+1)}, \cdots, u_m^{(t+1)}\right)$

- Subgradient oracle $\mathcal{O}_\nabla : |i\rangle |j\rangle |\bar{0}\rangle \rightarrow |i\rangle |j\rangle |(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathcal{O}_{\mathscr{P}} : u \rightarrow \text{argmin}_{q \in \mathscr{U}} \|q - u\|_2$
- Optimization oracle $\mathcal{O}_\epsilon$: Takes $u_1, \cdots, u_m \in \mathscr{U}$ as input. Outputs $x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq \epsilon, \forall i \in [m]$$

or returns "INFEASIBLE" if $\nexists x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq 0, \forall i \in [m]$$

Ben-Tal *et al.*'s algorithm

- Initialize noise parameters $\left(u_1^{(0)}, \ldots, u_m^{(0)}\right) \in \mathscr{U}^m$ and $x^{(0)} \in \mathscr{D}$ arbitrarily;
- **For** $t = 0$ **to** $T - 1$**, do**
  - $\nabla_1, \cdots, \nabla_m \leftarrow$ Query $\mathscr{O}_\nabla$ with $x^{(t)}, u_1^{(t)}, \cdots, u_m^{(t)}$
  - **For** $i = 1, \cdots, m$**, d0**
    - $u_i^{(t+1)} = \mathscr{O}_{\mathscr{P}}\left(u_i^{(t)} + \eta \nabla_i^{(t)}\right)$
  - **EndFor**
  - $x^{(t+1)} = \mathscr{O}_\epsilon\left(u_1^{(t+1)}, \cdots, u_m^{(t+1)}\right)$
  - **If** oracle declares infeasibility

- Subgradient oracle $\mathscr{O}_\nabla : |i\rangle|j\rangle|\bar{0}\rangle \rightarrow |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathscr{O}_{\mathscr{P}} : u \rightarrow \text{argmin}_{q \in \mathscr{U}} \|q - u\|_2$
- Optimization oracle $\mathscr{O}_\epsilon$: Takes $u_1, \cdots, u_m \in \mathscr{U}$ as input. Outputs $x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq \epsilon, \forall i \in [m]$$

or returns "INFEASIBLE" if $\nexists x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq 0, \forall i \in [m]$$

Ben-Tal *et al.*'s algorithm

- Initialize noise parameters $\left(u_1^{(0)}, \ldots, u_m^{(0)}\right) \in \mathscr{U}^m$ and $x^{(0)} \in \mathscr{D}$ arbitrarily;
- **For** $t = 0$ **to** $T - 1$**, do**
  - $\nabla_1, \cdots, \nabla_m \leftarrow$ Query $\mathcal{O}_\nabla$ with $x^{(t)}, u_1^{(t)}, \cdots, u_m^{(t)}$
  - **For** $i = 1, \cdots, m$**, d0**
    - $u_i^{(t+1)} = \mathcal{O}_{\mathscr{P}}\left(u_i^{(t)} + \eta \nabla_i^{(t)}\right)$
  - **EndFor**
  - $x^{(t+1)} = \mathcal{O}_\epsilon\left(u_1^{(t+1)}, \cdots, u_m^{(t+1)}\right)$
  - **If** oracle declares infeasibility
    - Return "INFEASIBLE"
  - **EndIf**
- **EndFor**

- Subgradient oracle $\mathcal{O}_\nabla : |i\rangle|j\rangle|\bar{0}\rangle \rightarrow |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathcal{O}_{\mathscr{P}} : u \rightarrow \text{argmin}_{q \in \mathscr{U}} \|q - u\|_2$
- Optimization oracle $\mathcal{O}_\epsilon$: Takes $u_1, \cdots, u_m \in \mathscr{U}$ as input. Outputs $x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq \epsilon, \forall i \in [m]$$

or returns "INFEASIBLE" if $\nexists x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq 0, \forall i \in [m]$$

Ben-Tal *et al.*'s algorithm

- Initialize noise parameters $\left(u_1^{(0)}, \ldots, u_m^{(0)}\right) \in \mathscr{U}^m$ and $x^{(0)} \in \mathscr{D}$ arbitrarily;
- **For** $t = 0$ **to** $T-1$**, do**
  - $\nabla_1, \cdots, \nabla_m \leftarrow$ Query $\mathcal{O}_\nabla$ with $x^{(t)}, u_1^{(t)}, \cdots, u_m^{(t)}$
  - **For** $i = 1, \cdots, m$**, d0**
    - $u_i^{(t+1)} = \mathcal{O}_{\mathscr{P}}\left(u_i^{(t)} + \eta \nabla_i^{(t)}\right)$
  - **EndFor**
  - $x^{(t+1)} = \mathcal{O}_\epsilon\left(u_1^{(t+1)}, \cdots, u_m^{(t+1)}\right)$
  - **If** oracle declares infeasibility
    - Return "INFEASIBLE"
  - **EndIf**
- **EndFor**
- **Output:** $\bar{x} = \frac{1}{T}\sum_{t=1}^{T} x^{(t)}$

- Subgradient oracle $\mathcal{O}_\nabla : |i\rangle|j\rangle|\bar{0}\rangle \rightarrow |i\rangle|j\rangle|(\nabla_u f_i(x, u_i))_j\rangle$
- Projection oracle $\mathcal{O}_{\mathscr{P}} : u \rightarrow$ argmin$_{q\in\mathscr{U}}\|q-u\|_2$
- Optimization oracle $\mathcal{O}_\epsilon$: Takes $u_1, \cdots, u_m \in \mathscr{U}$ as input. Outputs $x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq \epsilon, \forall i \in [m]$$

or returns "INFEASIBLE" if $\nexists x \in \mathscr{D}$ such that

$$f_i\left(x, u_i\right) \leq 0, \forall i \in [m]$$

Quantum Robust Optimization

Quantum computational model

Quantum computational model

- Quantum circuit model

Quantum computational model

- Quantum circuit model
    - An application of a quantum gate is equivalent to performing an elementary operation

Quantum computational model

- Quantum circuit model
    - An application of a quantum gate is equivalent to performing an elementary operation
- Quantum arithmetic model

Quantum computational model

- Quantum circuit model
  - An application of a quantum gate is equivalent to performing an elementary operation
- Quantum arithmetic model
  - Arithmetic operations take constant time

Quantum computational model

- Quantum circuit model
  - An application of a quantum gate is equivalent to performing an elementary operation
- Quantum arithmetic model
  - Arithmetic operations take constant time
  - Ignores issues from fixed-point representation of real numbers

Quantum computational model

- Quantum circuit model
  - An application of a quantum gate is equivalent to performing an elementary operation
- Quantum arithmetic model
  - Arithmetic operations take constant time
  - Ignores issues from fixed-point representation of real numbers
- Query complexity: maximum number of queries the algorithm makes on any input

Quantum computational model

- Quantum circuit model
  - An application of a quantum gate is equivalent to performing an elementary operation
- Quantum arithmetic model
  - Arithmetic operations take constant time
  - Ignores issues from fixed-point representation of real numbers
- Query complexity: maximum number of queries the algorithm makes on any input
- Given states $|x_1\rangle, \cdots, |x_n\rangle$, where $x_i \in [0,1]$ for $i \in [n]$, we can do

Quantum computational model

- Quantum circuit model
    - An application of a quantum gate is equivalent to performing an elementary operation
- Quantum arithmetic model
    - Arithmetic operations take constant time
    - Ignores issues from fixed-point representation of real numbers
- Query complexity: maximum number of queries the algorithm makes on any input
- Given states $|x_1\rangle, \cdots, |x_n\rangle$, where $x_i \in [0,1]$ for $i \in [n]$, we can do

$$|x_i\rangle|0\rangle \to |x_i\rangle\left(\sqrt{x_i}|0\rangle + \sqrt{1-x_i}|1\rangle\right)$$

Quantum computational model

- Quantum circuit model
    - An application of a quantum gate is equivalent to performing an elementary operation
- Quantum arithmetic model
    - Arithmetic operations take constant time
    - Ignores issues from fixed-point representation of real numbers
- Query complexity: maximum number of queries the algorithm makes on any input
- Given states $|x_1\rangle, \cdots, |x_n\rangle$, where $x_i \in [0,1]$ for $i \in [n]$, we can do

$$|x_i\rangle|0\rangle \to |x_i\rangle\left(\sqrt{x_i}|0\rangle + \sqrt{1-x_i}|1\rangle\right)$$

on a superposition of $|x_i\rangle$ in $O(1)$ time

Achieving a speedup

Achieving a speedup

Instead of	
Querying the subgradient oracle on all the entries of every subgradient	$\nabla_1, \cdots, \nabla_m \leftarrow$ Query $\mathcal{O}_\nabla$ with $x^{(t)}, u_1^{(t)}, \cdots, u_m^{(t)}$

Achieving a speedup

Instead of

Querying the subgradient oracle on all the entries of every subgradient

We do

- Perform ℓ_1**-multi-sampling** on the subgradients.
- Query the subgradient oracle on the **sampled** indices to create **stochastic** subgradients.

Achieving a speedup

Instead of	We do
Querying the subgradient oracle on all the entries of every subgradient	• Perform ℓ_1**-multi-sampling** on the subgradients. • Query the subgradient oracle on the **sampled** indices to create **stochastic** subgradients.
Updating the noise parameters using subgradients	

Achieving a speedup

Instead of	We do
Querying the subgradient oracle on all the entries of every subgradient	• Perform ℓ_1**-multi-sampling** on the subgradients. • Query the subgradient oracle on the **sampled** indices to create **stochastic** subgradients.
Updating the noise parameters using subgradients	Update the noise parameters using **stochastic** subgradients.

Algorithm 3 Quantum online sampling-based dual subgradient robust optimization algorithm

Input: Target accuracy $\epsilon > 0$, failure probability $\delta \in (0,1)$, parameters D, G_1, G_2, G_∞, F;

Set $T = \left\lceil \frac{1}{\epsilon^2} \max\left\{4F\log(\frac{m}{\delta}), \frac{225D^2}{16}\left(G_2^2 + \frac{G_1G_\infty - G_2^2}{s}\right)\right\}\right\rceil$ and $\eta^{(t)} = \frac{4D}{3\sqrt{t+1}}\left(G_2^2 + \frac{G_1G_\infty - G_2^2}{s}\right)^{-1/2}$;

Initialize $(u_1^{(0)}, \ldots, u_m^{(0)}) \in \mathcal{U}^m$ and $x^{(0)} \in \mathcal{D}$ arbitrarily;

for $t = 0$ to $T-1$ **do**

  Sample s pairs $S^{(t)} = ((i_1, j_1), \ldots, (i_s, j_s)) \in ([m]\times[d])^s$ with probability at least $1-\delta/T$ by measuring s copies of the quantum state $\sum_{i=1}^m \sum_{j=1}^d \sqrt{p^{(t)}(i,j)}|i\rangle|j\rangle$ (Fact 1), where

$$p^{(t)}(i,j) = \frac{\left|\left(\nabla_u f_i(x^{(t)}, u_i^{(t)})\right)_j\right|}{\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1};$$

  Compute an estimate $\Gamma^{(t)} \in \mathbb{R}$ of $\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1$ with relative error $1/4$ (Fact 2);

  Query the oracle $\mathcal{O}_\nabla$ with inputs $(i,j) \in S^{(t)}$, $x^{(t)}$, and $u_i^{(t)}$, to prepare $g_i^{(t)} \in \mathbb{R}^d$ as

$$(g_i^{(t)})_j = \frac{|\{(k,\ell) \in S^{(t)} : k = i, \ell = j\}|}{s} \frac{\text{sign}\left[(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right]}{(\Gamma^{(t)})^{-1}};$$

  for i in $S^{(t)}$ **do**

    $u_i'^{(t+1)} \leftarrow u_i^{(t)} + \eta^{(t)} g_i^{(t)}$;

    $u_i^{(t+1)} \leftarrow \mathcal{P}_\mathcal{U}(u_i'^{(t+1)})$; ▷ Update noise memory

  end for

  $x^{(t+1)} \leftarrow \mathcal{O}_\epsilon(u_1^{(t+1)}, \ldots, u_m^{(t+1)})$;

  if oracle declares infeasibility **then return** `INFEASIBLE`;

  end if

end for

Output: $\bar{x} = \frac{1}{T}\sum_{t=1}^T x^{(t)}$;

Algorithm 3 Quantum online sampling-based dual subgradient robust optimization algorithm

Input: Target accuracy $\epsilon > 0$, failure probability $\delta \in (0,1)$, parameters D, G_1, G_2, G_∞, F;

Set $T = \left\lceil \frac{1}{\epsilon^2} \max\left\{4F \log(\frac{m}{\delta}), \frac{225D^2}{16}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right)\right\}\right\rceil$ and $\eta^{(t)} = \frac{4D}{3\sqrt{t+1}}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right)^{-1/2}$;

Initialize $(u_1^{(0)}, \ldots, u_m^{(0)}) \in \mathcal{U}^m$ and $x^{(0)} \in \mathcal{D}$ arbitrarily;

for $t = 0$ to $T-1$ **do**

Sample s pairs $S^{(t)} = ((i_1, j_1), \ldots, (i_s, j_s)) \in ([m] \times [d])^s$ with probability at least $1 - \delta/T$ by measuring s copies of the quantum state $\sum_{i=1}^m \sum_{j=1}^d \sqrt{p^{(t)}(i,j)}|i\rangle|j\rangle$ (Fact 1), where

$$p^{(t)}(i,j) = \frac{\left|\left(\nabla_u f_i(x^{(t)}, u_i^{(t)})\right)_j\right|}{\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1};$$

Compute an estimate $\Gamma^{(t)} \in \mathbb{R}$ of $\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1$ with relative error $1/4$ (Fact 2);

Query the oracle $\mathcal{O}_\nabla$ with inputs $(i,j) \in S^{(t)}$, $x^{(t)}$, and $u_i^{(t)}$, to prepare $g_i^{(t)} \in \mathbb{R}^d$ as

$$(g_i^{(t)})_j = \frac{|\{(k,\ell) \in S^{(t)} : k = i, \ell = j\}|}{s} \frac{\text{sign}\left[(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right]}{(\Gamma^{(t)})^{-1}};$$

for i in $S^{(t)}$ **do**

$u_i'^{(t+1)} \leftarrow u_i^{(t)} + \eta^{(t)} g_i^{(t)}$;

$u_i^{(t+1)} \leftarrow \mathcal{P}_\mathcal{U}(u_i'^{(t+1)})$; ▷ Update noise memory

end for

$x^{(t+1)} \leftarrow \mathcal{O}_\epsilon(u_1^{(t+1)}, \ldots, u_m^{(t+1)})$;

if oracle declares infeasibility **then return** `INFEASIBLE`;

end if

end for

Output: $\bar{x} = \frac{1}{T}\sum_{t=1}^T x^{(t)}$;

Algorithm 3 Quantum online sampling-based dual subgradient robust optimization algorithm

Input: Target accuracy $\epsilon > 0$, failure probability $\delta \in (0,1)$, parameters D, G_1, G_2, G_∞, F;

Set $T = \left\lceil \frac{1}{\epsilon^2} \max\left\{4F\log(\frac{m}{\delta}), \frac{225D^2}{16}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right)\right\}\right\rceil$ and $\eta^{(t)} = \frac{4D}{3\sqrt{t+1}}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right)^{-1/2}$;

Initialize $(u_1^{(0)}, \ldots, u_m^{(0)}) \in \mathcal{U}^m$ and $x^{(0)} \in \mathcal{D}$ arbitrarily;

for $t = 0$ to $T-1$ **do**

  Sample s pairs $S^{(t)} = ((i_1, j_1), \ldots, (i_s, j_s)) \in ([m] \times [d])^s$ with probability at least $1 - \delta/T$ by measuring s copies of the quantum state $\sum_{i=1}^m \sum_{j=1}^d \sqrt{p^{(t)}(i,j)}|i\rangle|j\rangle$ (Fact 1), where

$$p^{(t)}(i,j) = \frac{\left|(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right|}{\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1};$$

Quantum multi-sampling: samples s numbers from $[n]$ in $O\left(\sqrt{sm}\log\left(\frac{1}{\delta}\right)\right)$ time.

  Compute an estimate $\Gamma^{(t)} \in \mathbb{R}$ of $\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1$ with relative error $1/4$ (Fact 2);

  Query the oracle $\mathcal{O}_\nabla$ with inputs $(i,j) \in S^{(t)}$, $x^{(t)}$, and $u_i^{(t)}$, to prepare $g_i^{(t)} \in \mathbb{R}^d$ as

$$(g_i^{(t)})_j = \frac{|\{(k,\ell) \in S^{(t)} : k = i, \ell = j\}|}{s} \frac{\mathrm{sign}\left[(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right]}{(\Gamma^{(t)})^{-1}};$$

  for i in $S^{(t)}$ **do**

    $u_i'^{(t+1)} \leftarrow u_i^{(t)} + \eta^{(t)} g_i^{(t)}$;

    $u_i^{(t+1)} \leftarrow \mathcal{P}_{\mathcal{U}}(u_i'^{(t+1)})$; ▷ Update noise memory

  end for

  $x^{(t+1)} \leftarrow \mathcal{O}_\epsilon(u_1^{(t+1)}, \ldots, u_m^{(t+1)})$;

  if oracle declares infeasibility **then return** `INFEASIBLE`;

  end if

end for

Output: $\bar{x} = \frac{1}{T}\sum_{t=1}^T x^{(t)}$;

Algorithm 3 Quantum online sampling-based dual subgradient robust optimization algorithm

Input: Target accuracy $\epsilon > 0$, failure probability $\delta \in (0,1)$, parameters D, G_1, G_2, G_∞, F;

Set $T = \left\lceil \frac{1}{\epsilon^2} \max\left\{4F\log(\frac{m}{\delta}), \frac{225D^2}{16}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right)\right\}\right\rceil$ and $\eta^{(t)} = \frac{4D}{3\sqrt{t+1}}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right)^{-1/2}$;

Initialize $(u_1^{(0)}, \dots, u_m^{(0)}) \in \mathcal{U}^m$ and $x^{(0)} \in \mathcal{D}$ arbitrarily;

for $t = 0$ to $T-1$ **do**

  Sample s pairs $S^{(t)} = ((i_1, j_1), \dots, (i_s, j_s)) \in ([m] \times [d])^s$ with probability at least $1 - \delta/T$ by measuring s copies of the quantum state $\sum_{i=1}^m \sum_{j=1}^d \sqrt{p^{(t)}(i,j)}|i\rangle|j\rangle$ (Fact 1), where

$$p^{(t)}(i,j) = \frac{\left|\left(\nabla_u f_i(x^{(t)}, u_i^{(t)})\right)_j\right|}{\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1};$$

  Compute an estimate $\Gamma^{(t)} \in \mathbb{R}$ of $\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1$ with relative error $1/4$ (Fact 2);

  Query the oracle $\mathcal{O}_\nabla$ with inputs $(i,j) \in S^{(t)}$, $x^{(t)}$, and $u_i^{(t)}$, to prepare $g_i^{(t)} \in \mathbb{R}^d$ as

$$(g_i^{(t)})_j = \frac{|\{(k,\ell) \in S^{(t)} : k = i, \ell = j\}|}{s} \frac{\operatorname{sign}\left[(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right]}{(\Gamma^{(t)})^{-1}};$$

  for i in $S^{(t)}$ **do**

    $u_i'^{(t+1)} \leftarrow u_i^{(t)} + \eta^{(t)} g_i^{(t)}$;

    $u_i^{(t+1)} \leftarrow \mathcal{P}_\mathcal{U}(u_i'^{(t+1)})$; ▻ Update noise memory

  end for

  $x^{(t+1)} \leftarrow \mathcal{O}_\epsilon(u_1^{(t+1)}, \dots, u_m^{(t+1)})$;

  if oracle declares infeasibility **then return** `INFEASIBLE`;

  end if

end for

Output: $\bar{x} = \frac{1}{T}\sum_{t=1}^T x^{(t)}$;

Quantum multi-sampling: samples s numbers from $[n]$ in $O\left(\sqrt{sm}\log\left(\frac{1}{\delta}\right)\right)$ time.

Algorithm 3 Quantum online sampling-based dual subgradient robust optimization algorithm

Input: Target accuracy $\epsilon > 0$, failure probability $\delta \in (0,1)$, parameters D, G_1, G_2, G_∞, F;

Set $T = \left\lceil \frac{1}{\epsilon^2} \max\left\{4F\log(\frac{m}{\delta}), \frac{225D^2}{16}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right)\right\}\right\rceil$ and $\eta^{(t)} = \frac{4D}{3\sqrt{t+1}}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right)^{-1/2}$;

Initialize $(u_1^{(0)}, \ldots, u_m^{(0)}) \in \mathcal{U}^m$ and $x^{(0)} \in \mathcal{D}$ arbitrarily;

for $t = 0$ to $T-1$ **do**

  Sample s pairs $S^{(t)} = ((i_1, j_1), \ldots, (i_s, j_s)) \in ([m] \times [d])^s$ with probability at least $1 - \delta/T$ by measuring s copies of the quantum state $\sum_{i=1}^m \sum_{j=1}^d \sqrt{p^{(t)}(i,j)}|i\rangle|j\rangle$ (Fact 1), where

$$p^{(t)}(i,j) = \frac{\left|\left(\nabla_u f_i(x^{(t)}, u_i^{(t)})\right)_j\right|}{\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1};$$

  Compute an estimate $\Gamma^{(t)} \in \mathbb{R}$ of $\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1$ with relative error $1/4$ (Fact 2);

  Query the oracle $\mathcal{O}_\nabla$ with inputs $(i,j) \in S^{(t)}$, $x^{(t)}$, and $u_i^{(t)}$, to prepare $g_i^{(t)} \in \mathbb{R}^d$ as

$$(g_i^{(t)})_j = \frac{|\{(k,\ell) \in S^{(t)} : k = i, \ell = j\}|}{s} \frac{\operatorname{sign}\left[(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right]}{(\Gamma^{(t)})^{-1}};$$

  for i in $S^{(t)}$ **do**

    $u_i'^{(t+1)} \leftarrow u_i^{(t)} + \eta^{(t)} g_i^{(t)}$;

    $u_i^{(t+1)} \leftarrow \mathcal{P}_\mathcal{U}(u_i'^{(t+1)})$; ▷ Update noise memory

  end for

  $x^{(t+1)} \leftarrow \mathcal{O}_\epsilon(u_1^{(t+1)}, \ldots, u_m^{(t+1)})$;

  if oracle declares infeasibility **then return** `INFEASIBLE`;

  end if

end for

Output: $\bar{x} = \frac{1}{T}\sum_{t=1}^T x^{(t)}$;

Quantum multi-sampling: samples s numbers from $[n]$ in $O\left(\sqrt{sm}\log\left(\frac{1}{\delta}\right)\right)$ time.

Quantum norm estimation

Algorithm 3 Quantum online sampling-based dual subgradient robust optimization algorithm

Input: Target accuracy $\epsilon > 0$, failure probability $\delta \in (0,1)$, parameters D, G_1, G_2, G_∞, F;

Set $T = \left\lceil \frac{1}{\epsilon^2} \max\left\{4F\log(\frac{m}{\delta}), \frac{225D^2}{16}\left(G_2^2 + \frac{G_1G_\infty - G_2^2}{s}\right)\right\}\right\rceil$ and $\eta^{(t)} = \frac{4D}{3\sqrt{t+1}}\left(G_2^2 + \frac{G_1G_\infty - G_2^2}{s}\right)^{-1/2}$;

Initialize $(u_1^{(0)}, \ldots, u_m^{(0)}) \in \mathcal{U}^m$ and $x^{(0)} \in \mathcal{D}$ arbitrarily;

for $t = 0$ to $T-1$ **do**

Sample s pairs $S^{(t)} = ((i_1, j_1), \ldots, (i_s, j_s)) \in ([m] \times [d])^s$ with probability at least $1 - \delta/T$ by measuring s copies of the quantum state $\sum_{i=1}^m \sum_{j=1}^d \sqrt{p^{(t)}(i,j)}|i\rangle|j\rangle$ (Fact 1), where

$$p^{(t)}(i,j) = \frac{\left|\left(\nabla_u f_i(x^{(t)}, u_i^{(t)})\right)_j\right|}{\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1};$$

Compute an estimate $\Gamma^{(t)} \in \mathbb{R}$ of $\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1$ with relative error $1/4$ (Fact 2);

Query the oracle $\mathcal{O}_\nabla$ with inputs $(i,j) \in S^{(t)}$, $x^{(t)}$, and $u_i^{(t)}$, to prepare $g_i^{(t)} \in \mathbb{R}^d$ as

$$(g_i^{(t)})_j = \frac{|\{(k,\ell) \in S^{(t)} : k = i, \ell = j\}|}{s} \frac{\operatorname{sign}\left[(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right]}{(\Gamma^{(t)})^{-1}};$$

for i in $S^{(t)}$ **do**

$u_i'^{(t+1)} \leftarrow u_i^{(t)} + \eta^{(t)} g_i^{(t)}$;

$u_i^{(t+1)} \leftarrow \mathcal{P}_\mathcal{U}(u_i'^{(t+1)})$; ▻ Update noise memory

end for

$x^{(t+1)} \leftarrow \mathcal{O}_\epsilon(u_1^{(t+1)}, \ldots, u_m^{(t+1)})$;

if oracle declares infeasibility **then return** `INFEASIBLE`;

end if

end for

Output: $\bar{x} = \frac{1}{T}\sum_{t=1}^T x^{(t)}$;

Quantum multi-sampling: samples s numbers from $[n]$ in $O\left(\sqrt{sm}\log\left(\frac{1}{\delta}\right)\right)$ time.

Quantum norm estimation

Algorithm 3 Quantum online sampling-based dual subgradient robust optimization algorithm

Input: Target accuracy $\epsilon > 0$, failure probability $\delta \in (0,1)$, parameters D, G_1, G_2, G_∞, F;

Set $T = \left\lceil \frac{1}{\epsilon^2} \max\left\{4F\log(\frac{m}{\delta}), \frac{225D^2}{16}\left(G_2^2 + \frac{G_1G_\infty - G_2^2}{s}\right)\right\}\right\rceil$ and $\eta^{(t)} = \frac{4D}{3\sqrt{t+1}}\left(G_2^2 + \frac{G_1G_\infty - G_2^2}{s}\right)^{-1/2}$;

Initialize $(u_1^{(0)}, \ldots, u_m^{(0)}) \in \mathcal{U}^m$ and $x^{(0)} \in \mathcal{D}$ arbitrarily;

for $t = 0$ to $T-1$ **do**

Sample s pairs $S^{(t)} = ((i_1, j_1), \ldots, (i_s, j_s)) \in ([m] \times [d])^s$ with probability at least $1 - \delta/T$ by measuring s copies of the quantum state $\sum_{i=1}^m \sum_{j=1}^d \sqrt{p^{(t)}(i,j)}|i\rangle|j\rangle$ (Fact 1), where

$$p^{(t)}(i,j) = \frac{\left|(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right|}{\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1};$$

Compute an estimate $\Gamma^{(t)} \in \mathbb{R}$ of $\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1$ with relative error 1/4 (Fact 2);

Query the oracle $\mathcal{O}_\nabla$ with inputs $(i,j) \in S^{(t)}$, $x^{(t)}$, and $u_i^{(t)}$, to prepare $g_i^{(t)} \in \mathbb{R}^d$ as

$$(g_i^{(t)})_j = \frac{|\{(k,\ell) \in S^{(t)} : k = i, \ell = j\}|}{s} \frac{\text{sign}\left[(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right]}{(\Gamma^{(t)})^{-1}};$$

for i in $S^{(t)}$ **do**

$u_i^{\prime(t+1)} \leftarrow u_i^{(t)} + \eta^{(t)} g_i^{(t)}$;

$u_i^{(t+1)} \leftarrow \mathcal{P}_{\mathcal{U}}(u_i^{\prime(t+1)})$; ▷ Update noise memory

end for

$x^{(t+1)} \leftarrow \mathcal{O}_\epsilon(u_1^{(t+1)}, \ldots, u_m^{(t+1)})$;

if oracle declares infeasibility **then return** `INFEASIBLE`;

end if

end for

Output: $\bar{x} = \frac{1}{T}\sum_{t=1}^T x^{(t)}$;

Quantum multi-sampling: samples s numbers from $[n]$ in $O\left(\sqrt{sm}\log\left(\frac{1}{\delta}\right)\right)$ time.

Quantum norm estimation

Stochastic gradient satisfies

$$\mathbb{E}\left[g_i^{(t)}\right] = \lambda \nabla_u f_i\left(x^{(t)}, u_i^{(t)}\right)$$

$$\mathbb{E}\left[\left\|g_i^{(t)}\right\|_2^2\right] \leq \tilde{G}_2^2$$

Algorithm 3 Quantum online sampling-based dual subgradient robust optimization algorithm

Input: Target accuracy $\epsilon > 0$, failure probability $\delta \in (0,1)$, parameters D, G_1, G_2, G_∞, F;

Set $T = \left\lceil \frac{1}{\epsilon^2} \max\left\{4F\log(\frac{m}{\delta}), \frac{225D^2}{16}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right)\right\}\right\rceil$ and $\eta^{(t)} = \frac{4D}{3\sqrt{t+1}}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right)^{-1/2}$;

Initialize $(u_1^{(0)}, \dots, u_m^{(0)}) \in \mathcal{U}^m$ and $x^{(0)} \in \mathcal{D}$ arbitrarily;

for $t = 0$ to $T-1$ **do**

Sample s pairs $S^{(t)} = ((i_1, j_1), \dots, (i_s, j_s)) \in ([m] \times [d])^s$ with probability at least $1 - \delta/T$ by measuring s copies of the quantum state $\sum_{i=1}^m \sum_{j=1}^d \sqrt{p^{(t)}(i,j)}|i\rangle|j\rangle$ (Fact 1), where

$$p^{(t)}(i,j) = \frac{\left|\left(\nabla_u f_i(x^{(t)}, u_i^{(t)})\right)_j\right|}{\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1};$$

Compute an estimate $\Gamma^{(t)} \in \mathbb{R}$ of $\sum_{k=1}^m \|\nabla_u f_k(x^{(t)}, u_k^{(t)})\|_1$ with relative error $1/4$ (Fact 2);

Query the oracle $\mathcal{O}_\nabla$ with inputs $(i,j) \in S^{(t)}$, $x^{(t)}$, and $u_i^{(t)}$, to prepare $g_i^{(t)} \in \mathbb{R}^d$ as

$$(g_i^{(t)})_j = \frac{|\{(k,\ell) \in S^{(t)} : k = i, \ell = j\}|}{s} \frac{\operatorname{sign}\left[(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right]}{(\Gamma^{(t)})^{-1}};$$

for i in $S^{(t)}$ **do**

$u_i'^{(t+1)} \leftarrow u_i^{(t)} + \eta^{(t)} g_i^{(t)}$;

$u_i^{(t+1)} \leftarrow \mathcal{P}_\mathcal{U}(u_i'^{(t+1)})$; ▷ Update noise memory

end for

$x^{(t+1)} \leftarrow \mathcal{O}_\epsilon(u_1^{(t+1)}, \dots, u_m^{(t+1)})$;

if oracle declares infeasibility **then return** `INFEASIBLE`;

end if

end for

Output: $\bar{x} = \frac{1}{T}\sum_{t=1}^T x^{(t)}$;

Quantum multi-sampling: samples s numbers from $[n]$ in $O\left(\sqrt{sm}\log\left(\frac{1}{\delta}\right)\right)$ time.

Quantum norm estimation

Stochastic gradient satisfies

$$\mathbb{E}\left[g_i^{(t)}\right] = \lambda \nabla_u f_i\left(x^{(t)}, u_i^{(t)}\right)$$

$$\mathbb{E}\left[\left\|g_i^{(t)}\right\|_2^2\right] \le \tilde{G}_2^2$$

Algorithm 3 Quantum online sampling-based dual subgradient robust optimization algorithm

Input: Target accuracy $\epsilon > 0$, failure probability $\delta \in (0,1)$, parameters D, G_1, G_2, G_∞, F;

Set $T = \left\lceil \frac{1}{\epsilon^2} \max\left\{4F\log(\frac{m}{\delta}), \frac{225D^2}{16}(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s})\right\}\right\rceil$ and $\eta^{(t)} = \frac{4D}{3\sqrt{t+1}}(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s})^{-1/2}$;

Initialize $(u_1^{(0)}, \ldots, u_m^{(0)}) \in \mathcal{U}^m$ and $x^{(0)} \in \mathcal{D}$ arbitrarily;

for $t = 0$ to $T-1$ **do**

Sample s pairs $S^{(t)} = ((i_1, j_1), \ldots, (i_s, j_s)) \in ([m] \times [d])^s$ with probability at least $1 - \delta/T$ by measuring s copies of the quantum state $\sum_{i=1}^m \sum_{j=1}^d \sqrt{p^{(t)}(i,j)}|i\rangle|j\rangle$ (Fact 1), where

$$p^{(t)}(i,j) = \frac{\left|(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right|}{\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1};$$

Compute an estimate $\Gamma^{(t)} \in \mathbb{R}$ of $\sum_{k=1}^m \left\|\nabla_u f_k(x^{(t)}, u_k^{(t)})\right\|_1$ with relative error $1/4$ (Fact 2);

Query the oracle $\mathcal{O}_\nabla$ with inputs $(i,j) \in S^{(t)}$, $x^{(t)}$, and $u_i^{(t)}$, to prepare $g_i^{(t)} \in \mathbb{R}^d$ as

$$(g_i^{(t)})_j = \frac{|\{(k,\ell) \in S^{(t)} : k = i, \ell = j\}|}{s} \frac{\text{sign}\left[(\nabla_u f_i(x^{(t)}, u_i^{(t)}))_j\right]}{(\Gamma^{(t)})^{-1}};$$

for i in $S^{(t)}$ **do**

$u_i'^{(t+1)} \leftarrow u_i^{(t)} + \eta^{(t)} g_i^{(t)}$;

$u_i^{(t+1)} \leftarrow \mathcal{P}_\mathcal{U}(u_i'^{(t+1)})$; **On sampled indices** ▷ Update noise memory

end for

$x^{(t+1)} \leftarrow \mathcal{O}_\epsilon(u_1^{(t+1)}, \ldots, u_m^{(t+1)})$;

if oracle declares infeasibility **then return** `INFEASIBLE`;

end if

end for

Output: $\bar{x} = \frac{1}{T}\sum_{t=1}^T x^{(t)}$;

Quantum multi-sampling: samples s numbers from $[n]$ in $O\left(\sqrt{sm}\log\left(\frac{1}{\delta}\right)\right)$ time.

Quantum norm estimation

Stochastic gradient satisfies

$$\mathbb{E}\left[g_i^{(t)}\right] = \lambda \nabla_u f_i\left(x^{(t)}, u_i^{(t)}\right)$$

$$\mathbb{E}\left[\left\|g_i^{(t)}\right\|_2^2\right] \le \tilde{G}_2^2$$

Summary of results

- When $T = \left\lceil \frac{1}{\epsilon^2} \max \left\{ 4F\left(\frac{m}{\delta}\right), \frac{9}{4}(1+4\nu)^2 D^2 \widetilde{G}_2^2 \right\} \right\rceil$ and $\eta^{(t)} = \frac{D}{(1-\nu)\widetilde{G}_2\sqrt{t+1}}$

Algorithm	Calls to $\mathcal{O}_\nabla$	Calls to $\mathcal{O}_\mathcal{P}$	Calls to $\mathcal{O}_\epsilon$
Ben-Tal *et. al*	$\frac{G_2^2 m d D^2}{\epsilon^2}$	$\frac{G_2^2 m D^2}{\epsilon^2}$	$\frac{G_2^2 D^2}{\epsilon^2}$

- $D \geq \max_{u,v \in \mathcal{U}} \|u - v\|_2$
- $F \geq \max_{x \in \mathcal{D}, u \in \mathcal{U}} |f_i(x,u)|$
- $G_2 \geq \max_{x \in \mathcal{D}, u \in \mathcal{U}} \|\nabla_u f_i(x,u)\|_2$
- $G_1 \geq \sum_{k=1}^{m} \|\nabla_u f_k(x, u_k)\|_1$
- $G_\infty \geq \max_{k \in [m]} \|\nabla_u f_k(x, u_k)\|_1$,

$\forall x \in \mathcal{D}, \forall u_1, \ldots, u_m \in \mathcal{U}$

- When $T = \left\lceil \frac{1}{\epsilon^2} \max \left\{ 4F \log\left(\frac{m}{\delta}\right), \frac{225 D^2}{16}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right) \right\} \right\rceil$ and $\eta^{(t)} = \frac{4D}{3\sqrt{t+1}}\left(G_2^2 + \frac{G_1 G_\infty - G_2^2}{s}\right)^{-1/2}$

Algorithm	Calls to $\mathcal{O}_\nabla$	Calls to $\mathcal{O}_\mathcal{P}$	Calls to $\mathcal{O}_\epsilon$
Our work	$\frac{\sqrt{G_1 G_\infty} G_2 \sqrt{md} D^2}{\epsilon^2} \log\left(\frac{D G_2}{\epsilon\delta}\right)$	$\min\{G_1 G_\infty, G_2^2 m\}\frac{D^2}{\epsilon^2}$	$\frac{G_2^2 D^2}{\epsilon^2}$

Summary of results

- When $T = \left\lceil \frac{1}{\epsilon^2} \max \left\{ 4F \left(\frac{m}{\delta} \right), \frac{9}{4}(1 \ldots \right.\right.$

Algorithm	
Ben-Tal et …	

- When $T = \left\lceil \frac{1}{\epsilon^2} \max \ldots - G_2^2 \ldots s \right)^{-1/2}$

Algorithm	$\mathcal{O}_\nabla$		… calls to $\mathcal{O}_\epsilon$
Our work	$\frac{\sqrt{G_1 G_\infty} G_2 \sqrt{md} D^2}{\epsilon^2} \log \left(\frac{D G_2}{\epsilon \delta} \right)$	$\min\{G_1 G_\infty, G_2^2 m\} \frac{D^2}{\epsilon^2}$	$\frac{G_2^2 D^2}{\epsilon^2}$

- $G_1 G_\infty = O\left(m G_2^2\right) \Rightarrow$ quadratic speedup in the dimension of the noise parameters d.
- $G_1 G_\infty$ is comparable to $G_2^2 \Rightarrow$ quadratic speedup in $m = \left| \mathcal{U} \right|, d$.

- $D \geq \max_{u,v \in \mathcal{U}} \|u - v\|_2$
- $F \geq \max_{x \in \mathcal{D}, u \in \mathcal{U}} |f_i(x, u)|$
- $G_2 \geq \max_{x \in \mathcal{D}, u \in \mathcal{U}} \|\nabla_u f_i(x, u)\|_2$
- $G_1 \geq \sum_{k=1}^{m} \|\nabla_u f_k(x, u_k)\|_1$
- $G_\infty \geq \max_{k \in [m]} \|\nabla_u f_k(x, u_k)\|_1,$

$\forall x \in \mathcal{D}, \forall u_1, \ldots, u_m \in \mathcal{U}$

Applications

Robust linear programs

$$\exists? \qquad x \in \mathscr{D}$$

$$\text{s.t.} \qquad \left(a_i + P_i u_i\right)^\top x - b_i \leq 0, \qquad \forall u_i \in \mathscr{U}, \quad i \in [m],$$

$$\mathscr{U} = \{u \in \mathbb{R}^d : \ \| u \|_2 \leq 1\}$$
$$\mathscr{D} \subseteq \{x \in \mathbb{R}^n : \ \| x \|_1 \leq 1\}$$

Robust linear programs

$$\exists? \quad x \in \mathcal{D}$$

$$\text{s.t.} \quad \left(a_i + P_i u_i\right)^\top x - b_i \le 0, \qquad \forall u_i \in \mathcal{U}, \quad i \in [m],$$

$$\mathcal{U} = \{u \in \mathbb{R}^d : \| u \|_2 \le 1\}$$
$$\mathcal{D} \subseteq \{x \in \mathbb{R}^n : \| x \|_1 \le 1\}$$

Global maximum return portfolio (GMVP)

Global maximum return portfolio (GMVP)

- n assets, m markets,

Global maximum return portfolio (GMVP)

- n assets, m markets,
- Portfolio vector: $x \in \mathbb{R}^n_{\geq 0}$ such that $\sum_{i=1}^{n} x_i = 1$.

Global maximum return portfolio (GMVP)

- n assets, m markets,
- Portfolio vector: $x \in \mathbb{R}^n_{\geq 0}$ such that $\sum_{i=1}^{n} x_i = 1$.
- Expected return of assets in market i: $r_i \in \mathbb{R}^n$, $\forall i \in [m]$.

Global maximum return portfolio (GMVP)

- n assets, m markets,
- Portfolio vector: $x \in \mathbb{R}^n_{\geq 0}$ such that $\sum_{i=1}^{n} x_i = 1$.
- Expected return of assets in market i: $r_i \in \mathbb{R}^n$, $\forall i \in [m]$.

Find a portfolio that maximizes the return (without considering the variance)

Robust semidefinite programs

$$\exists? \qquad X \in \mathcal{D}$$

s.t. $$\left(A_i + \sum_{j=1}^{d} u_{ij} P_j\right) \bullet X - b_i \leq 0, \quad \forall u_i \in \mathcal{U},\ i \in [m],$$

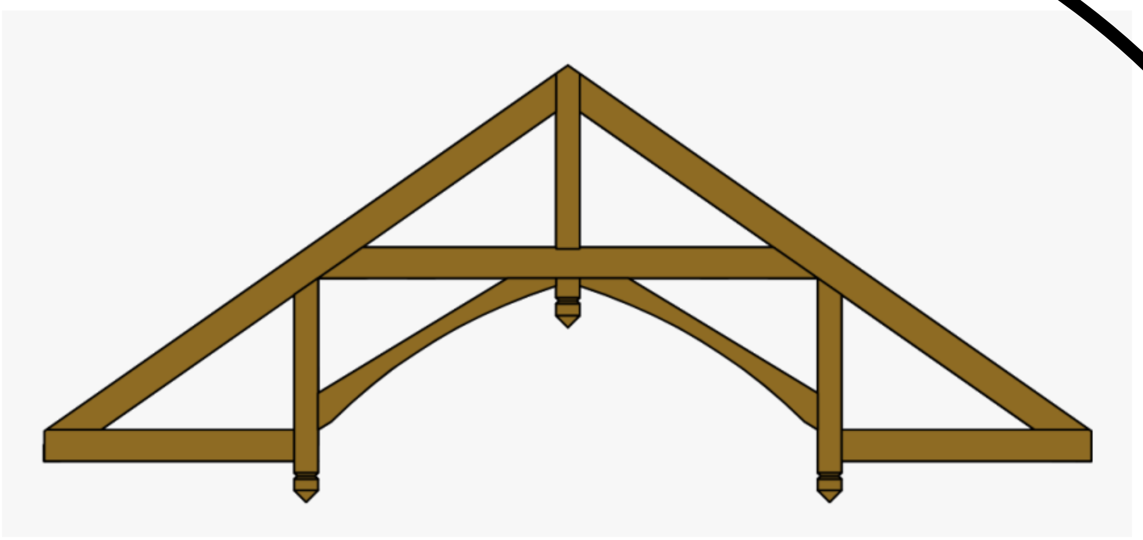
$$\mathcal{U} = \{u \in \mathbb{R}^d : \ \| u \|_2 \leq 1\}$$
$$\mathcal{D} \subseteq \{X \in \mathbb{S}_n^+ : \ \| X \|_F \leq 1\}$$

Robust semidefinite programs

$$\exists? \qquad X \in \mathscr{D}$$

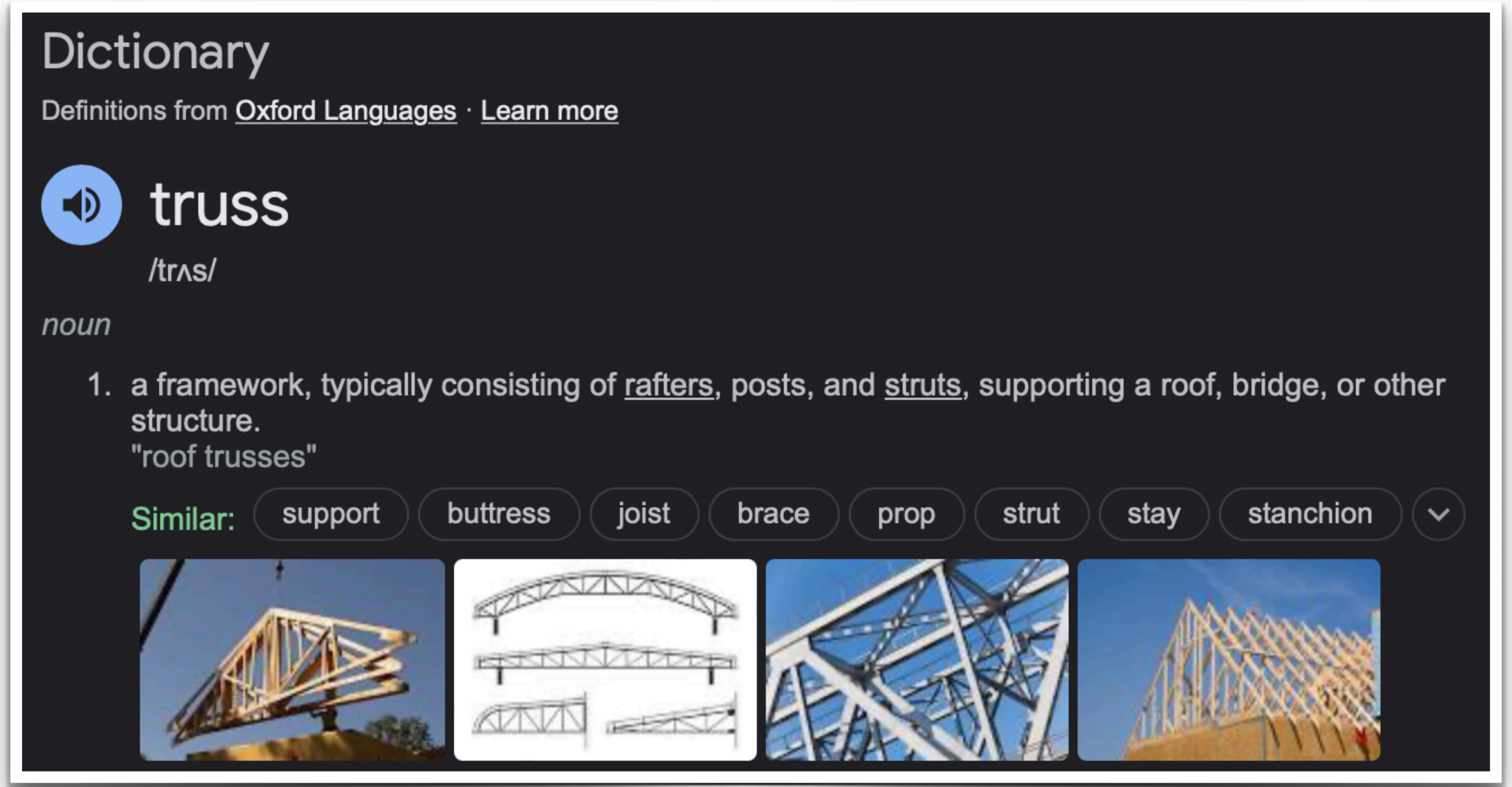
$$\textbf{s.t.} \qquad \left(A_i + \sum_{j=1}^{d} u_{ij} P_j \right) \bullet X - b_i \leq 0, \quad \forall u_i \in \mathscr{U},\ i \in [m],$$

$$\mathscr{U} = \{u \in \mathbb{R}^d : \ \| u \|_2 \leq 1\}$$
$$\mathscr{D} \subseteq \{X \in \mathbb{S}_n^+ : \ \| X \|_F \leq 1\}$$

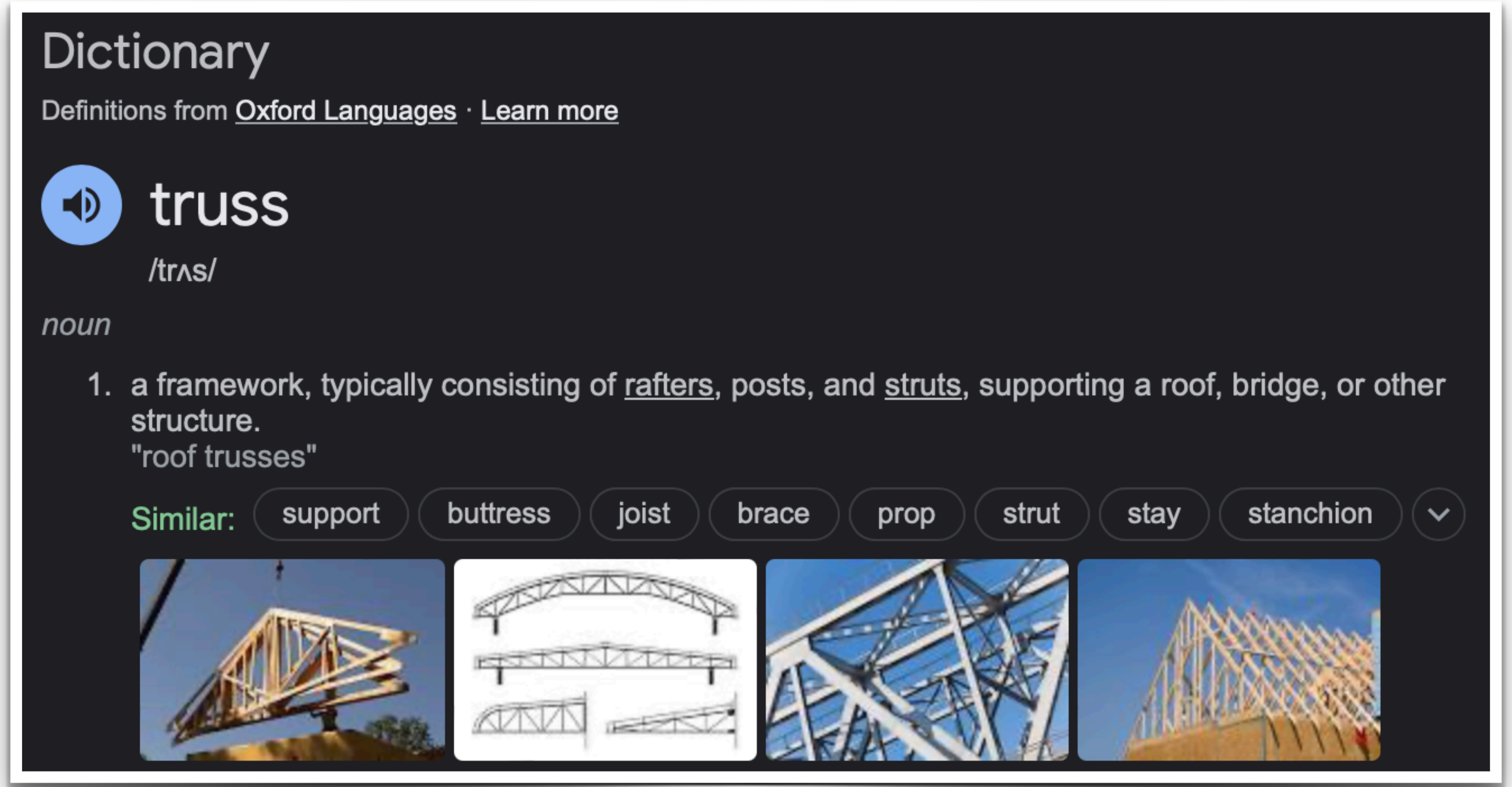


Truss Topological Design

Truss topological design (TTD)

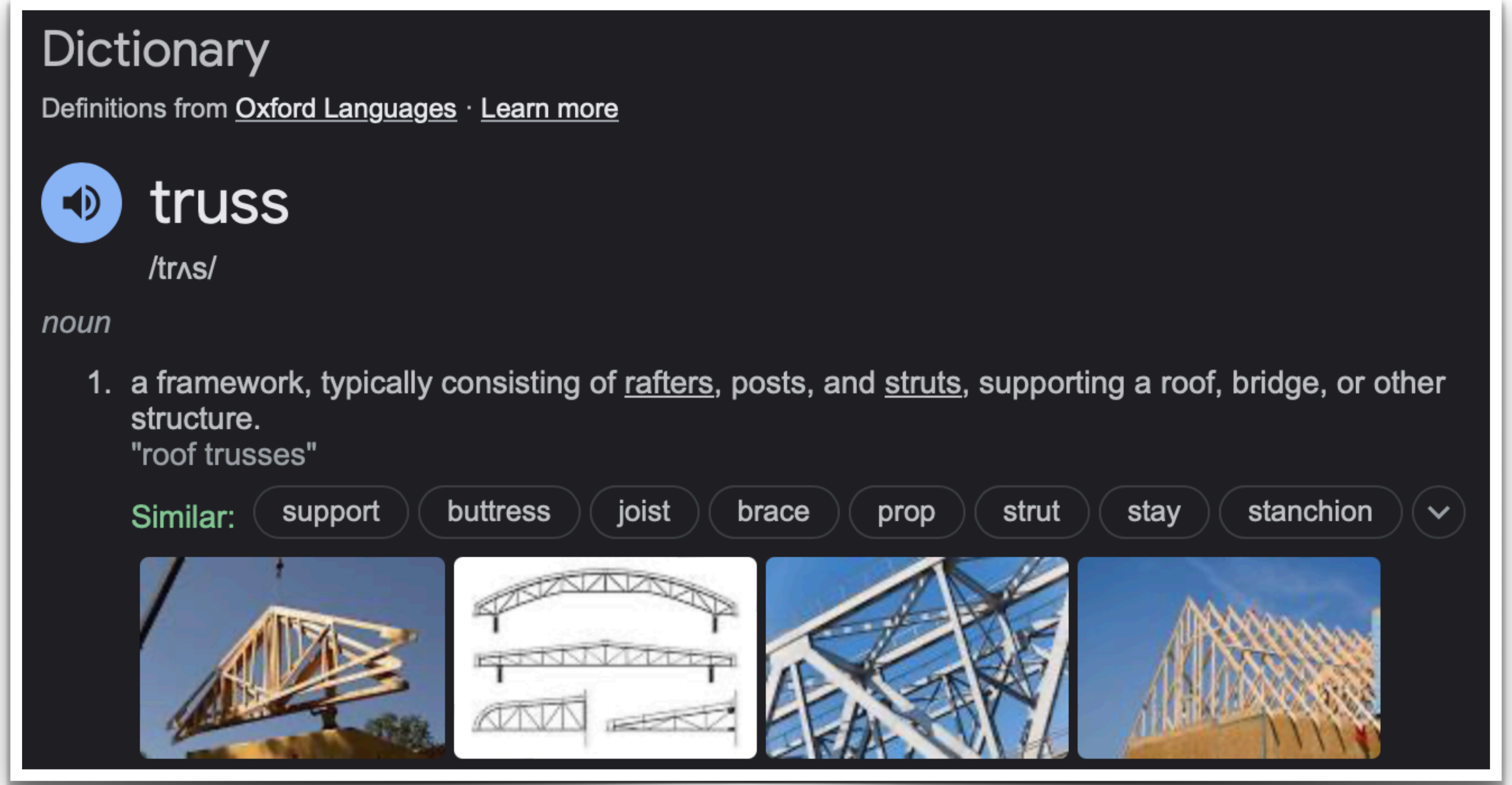


Truss topological design (TTD)



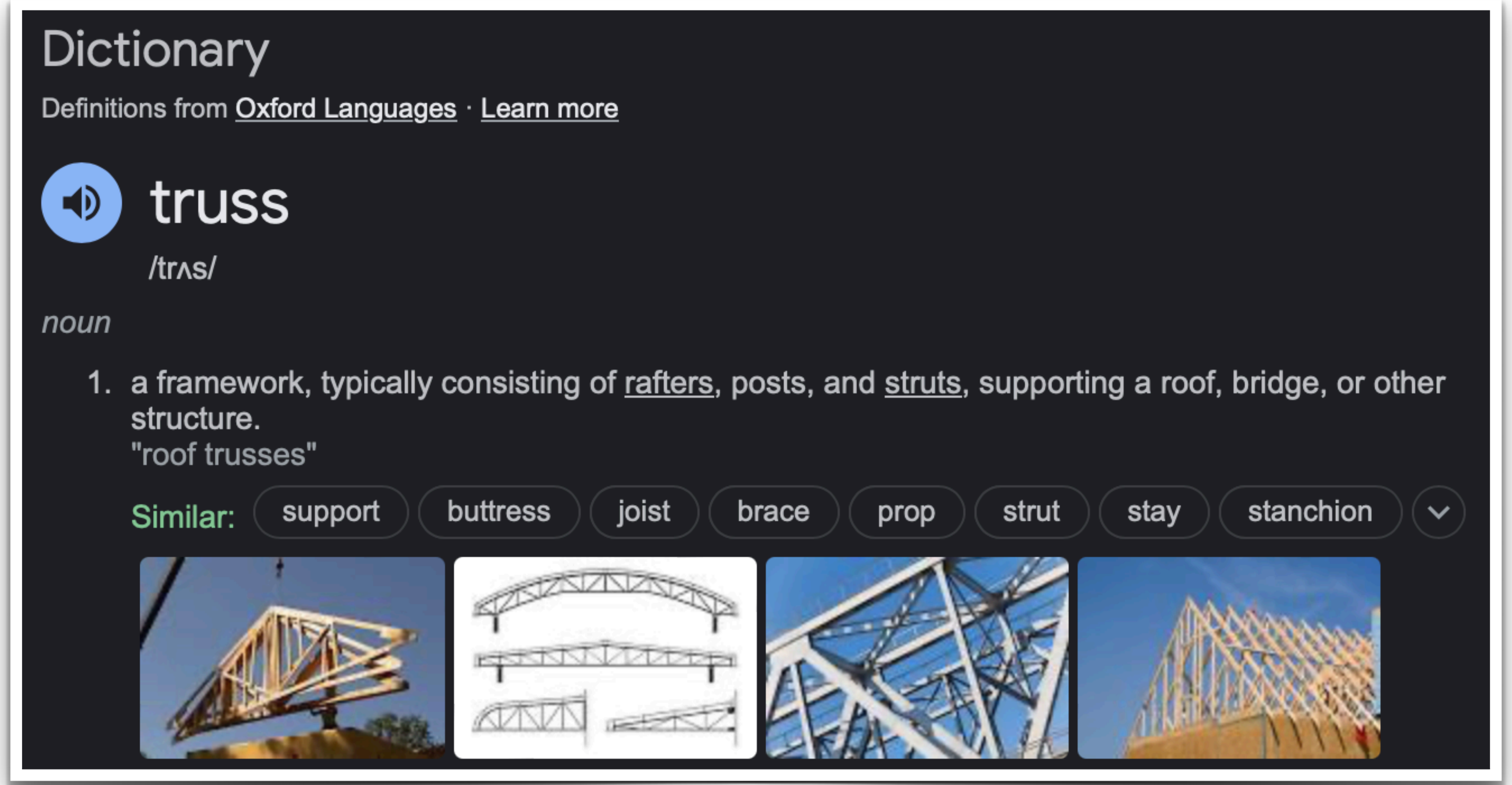
- Find the right geometry, topology and size of a truss structure to withstand external loads.

Truss topological design (TTD)



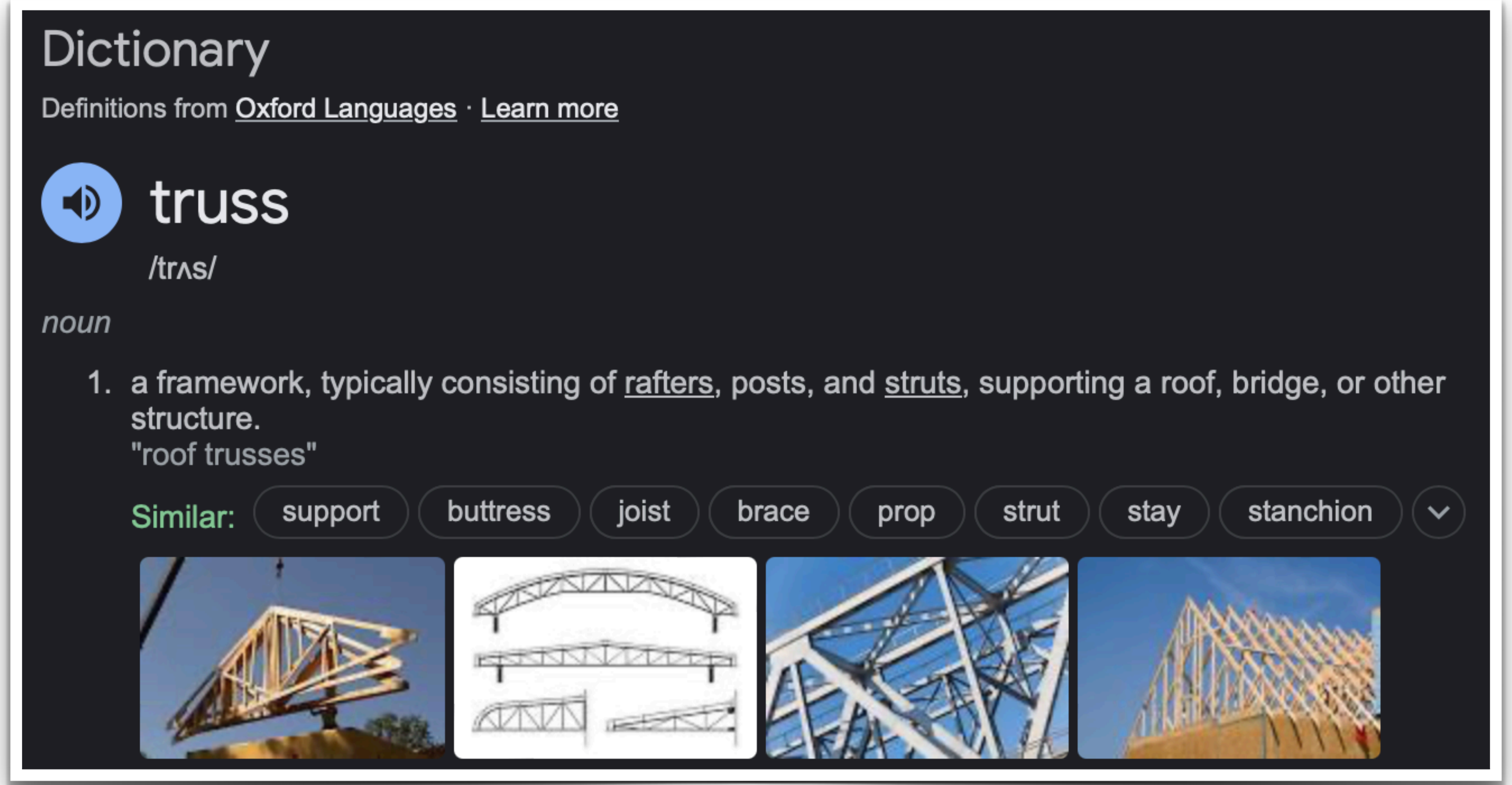
- Find the right geometry, topology and size of a truss structure to withstand external loads.
- Information on external load is not perfectly known, assumed to belong to an uncertainty set.

Truss topological design (TTD)



- Find the right geometry, topology and size of a truss structure to withstand external loads.
- Information on external load is not perfectly known, assumed to belong to an uncertainty set.
- Can be cast as a robust SDP.

Truss topological design (TTD)



- Find the right geometry, topology and size of a truss structure to withstand external loads.
- Information on external load is not perfectly known, assumed to belong to an uncertainty set.
- Can be cast as a robust SDP.

Runtime dependent on the Frobenius, ℓ_∞-norm of the matrices that control the shape of the ellipsoid.

Conclusion

Conclusion

Conclusion

- Quantum meta-algorithm for robust optimization

Conclusion

- Quantum meta-algorithm for robust optimization
- At most quadratic speedup in terms of the dimension of the noise parameters

Conclusion

- Quantum meta-algorithm for robust optimization
- At most quadratic speedup in terms of the dimension of the noise parameters
- Applications

Conclusion

- Quantum meta-algorithm for robust optimization
- At most quadratic speedup in terms of the dimension of the noise parameters
- Applications
  - Robust linear programs

Conclusion

- Quantum meta-algorithm for robust optimization
- At most quadratic speedup in terms of the dimension of the noise parameters
- Applications
  - Robust linear programs
  - Robust semidefinite programs


Thank
you