Certifiably Correct SLAM

David M. Rosen

University of Toronto Robotics Institute
Dec 7, 2020

Motivation

Robotics: The science of *physically embodied* autonomous agents.

Core capabilities:

- Planning
- Navigation
- Manipulation ...

$\Rightarrow$ These require *understanding the world's geometry*.

Example: Autonomous vehicles

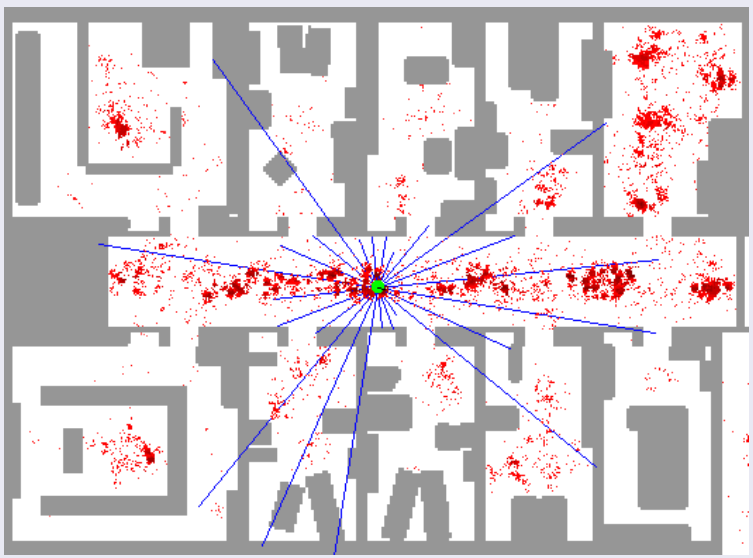
Video credit: Google

Two fundamental problems in robotic spatial perception

Localization ("Where am I?")

Given: Environmental model

Estimate: Robot pose

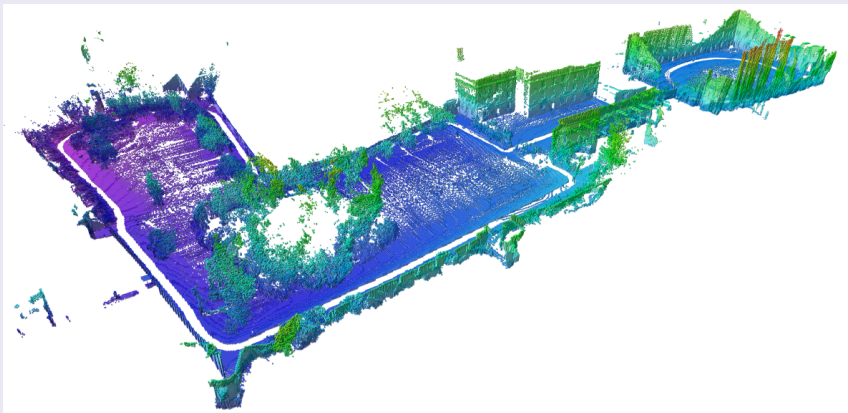


Fox et al. 1999

Mapping ("What's around me?")

Given: Robot pose

Estimate: Environmental model

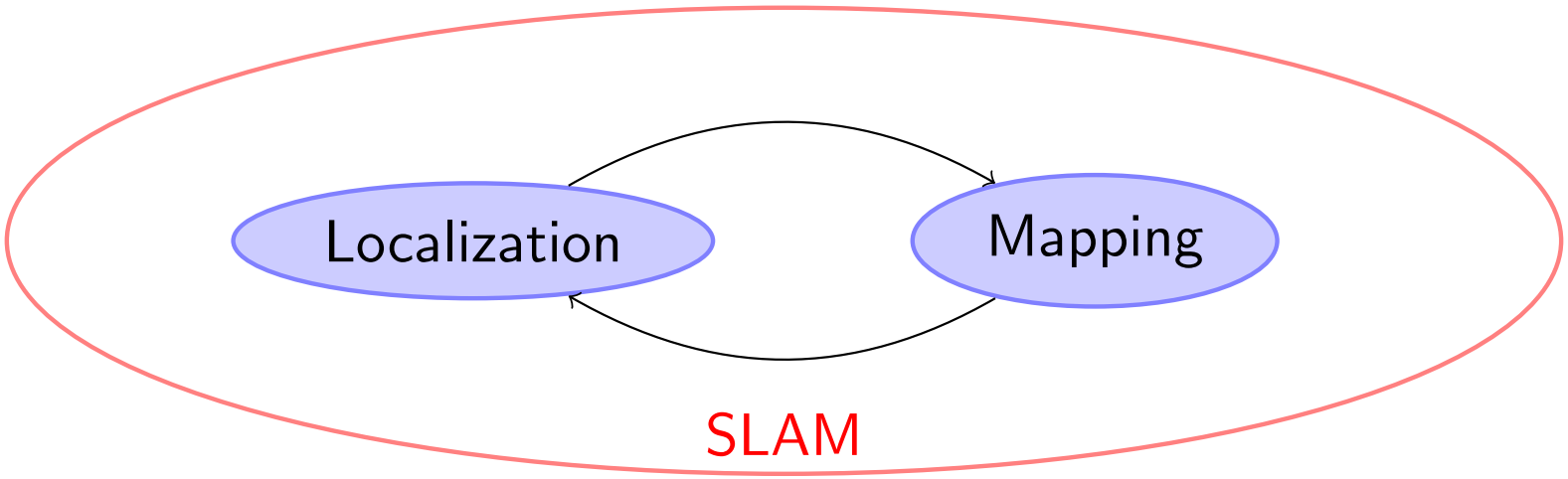


Hornung et al. 2013

$\Rightarrow$ Given *either* pose or map, we can find the other via *recursive Bayesian estimation*.

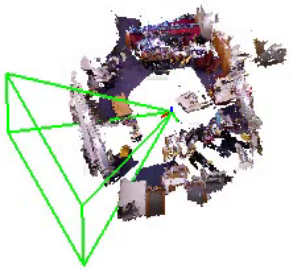
Simultaneous localization and mapping (SLAM)

Given **neither** robot pose nor map, *jointly estimate both*.



- **Much** harder than localization or mapping alone.
- Enables operation in *unknown environments*.
- An *essential enabling technology* for mobile robots.

Simultaneous localization and mapping (SLAM)



H. Johannsson et al. "Temporally Scalable Visual SLAM Using a Reduced Pose Graph". In: *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. Karlsruhe, Germany, May 2013, pp. 54–61

In this talk

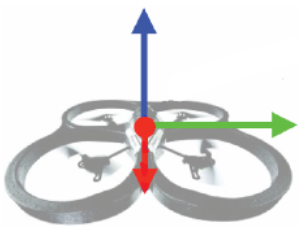
1. Lightning introduction to pose-graph SLAM
   - Problem formulation
   - The problem of *nonconvexity*

2. Our contributions: Certifiably correct SLAM
   - A *convex relaxation* for pose-graph SLAM
   - A fast optimization method to solve this relaxation efficiently

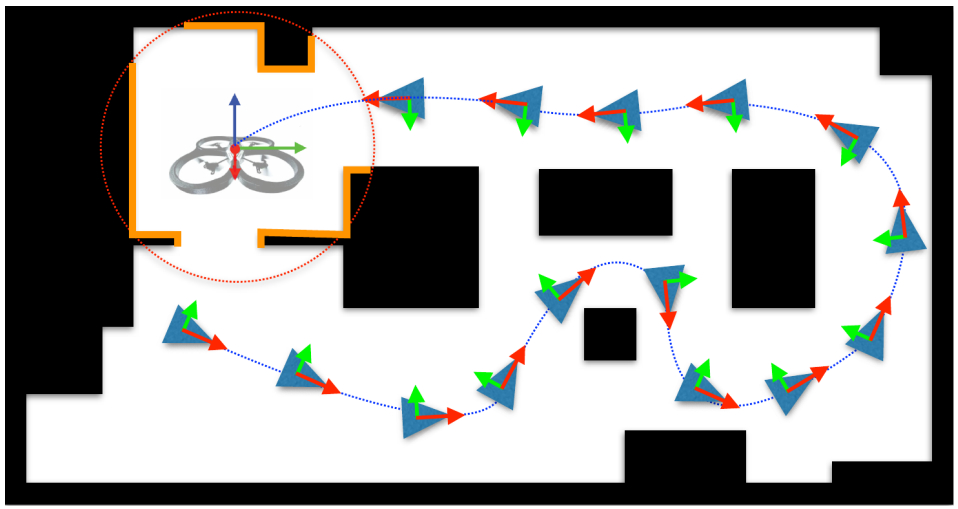
Payoff: SE-Sync, a *certifiably correct* algorithm for SLAM

A concrete example

Consider a robot exploring some initially unknown environment...

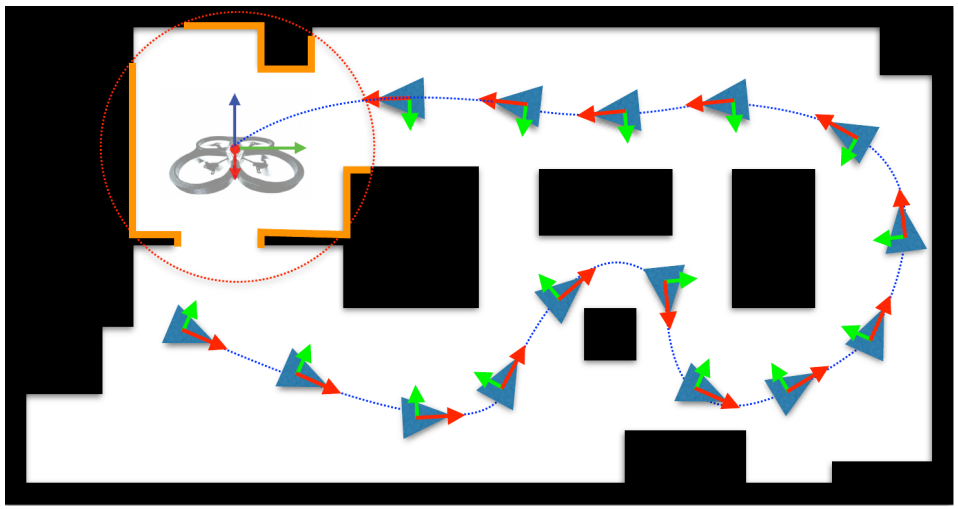


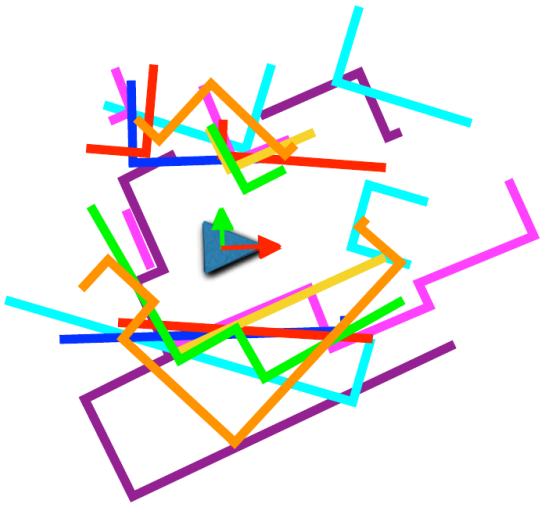
A concrete example



Figures courtesy of Luca Carlone

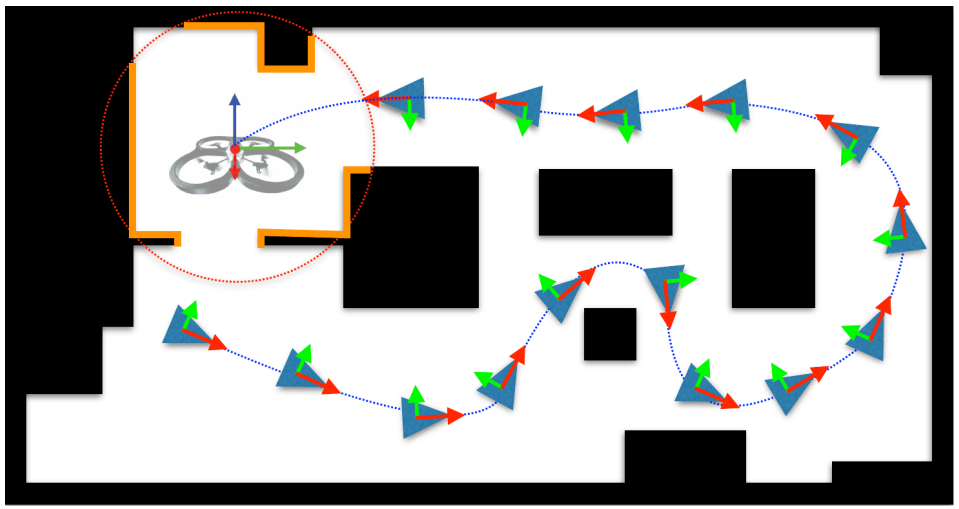
A concrete example

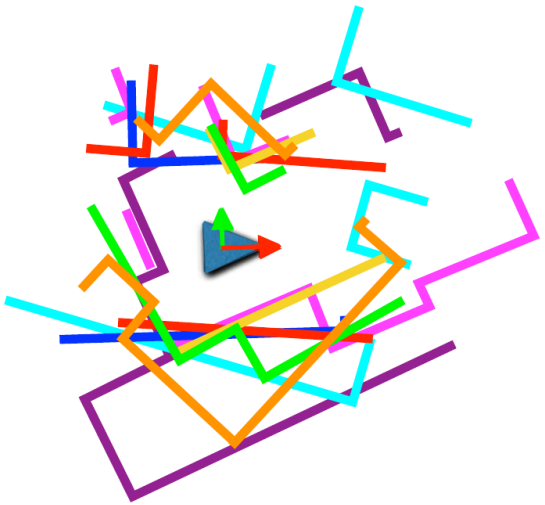


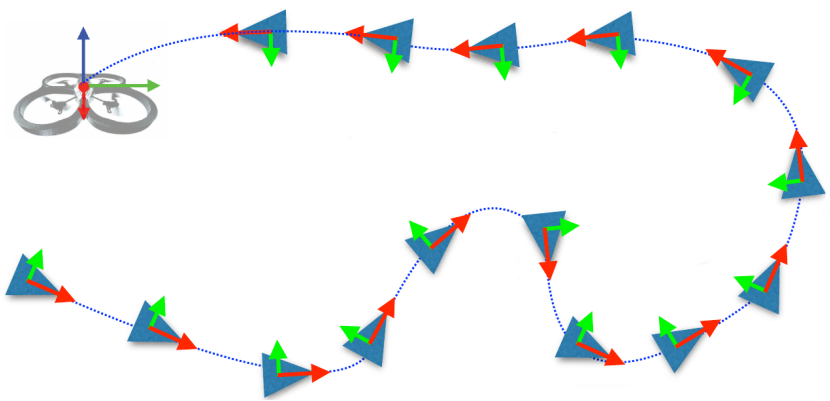


Figures courtesy of Luca Carlone

A concrete example

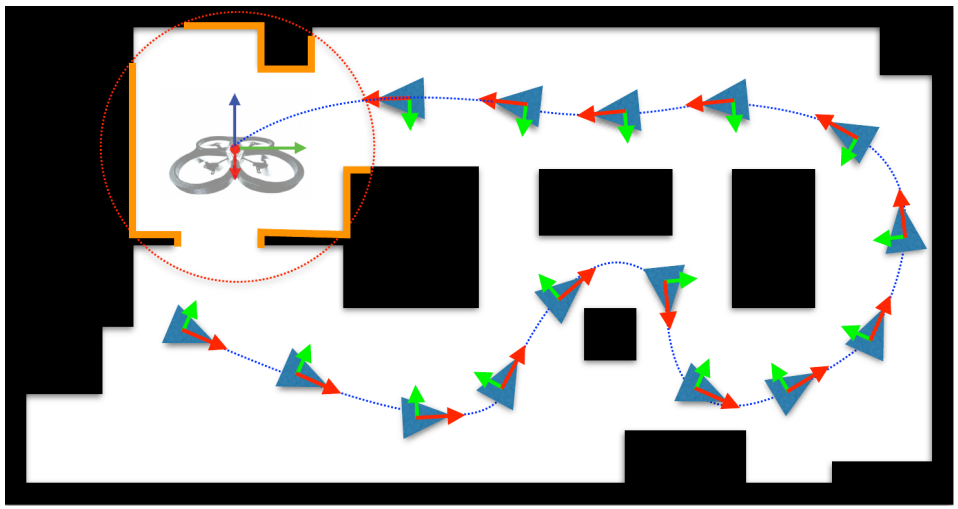


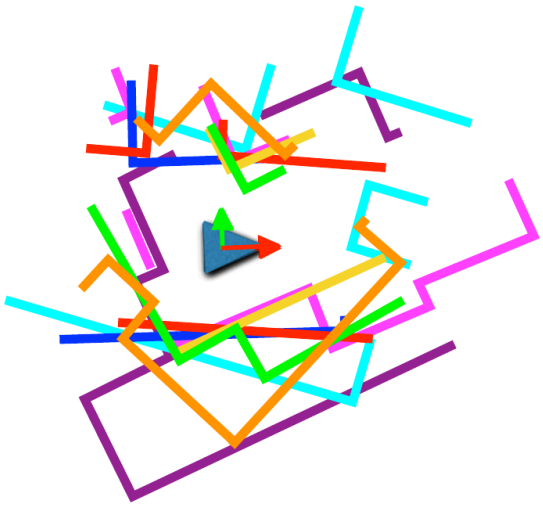


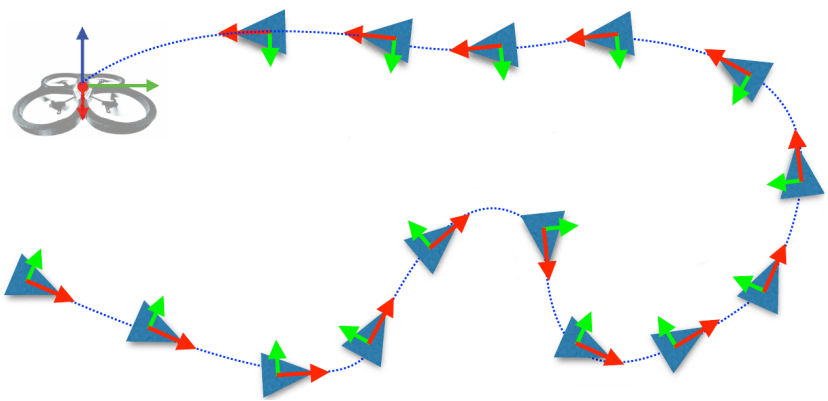


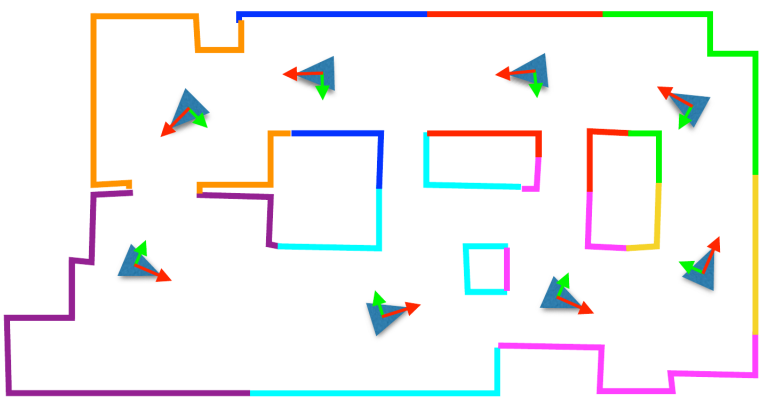
Figures courtesy of Luca Carlone

A concrete example



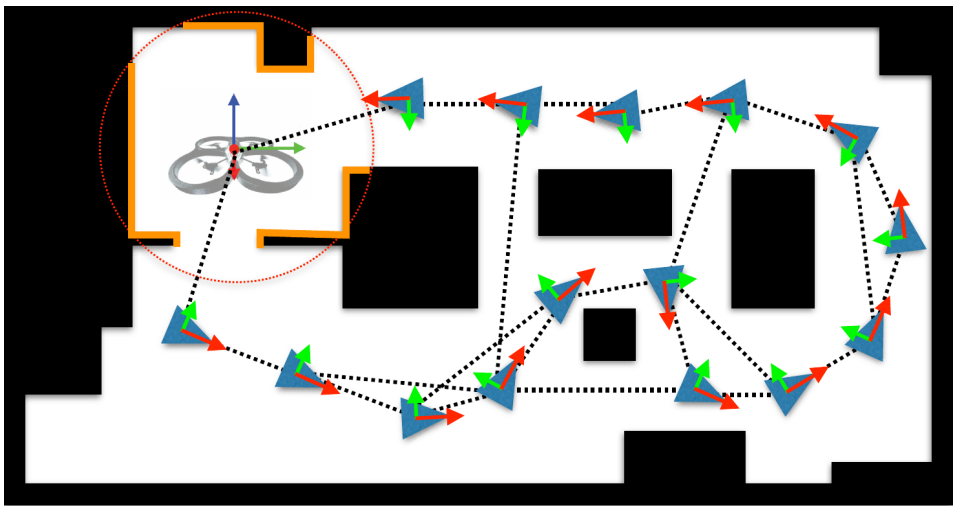


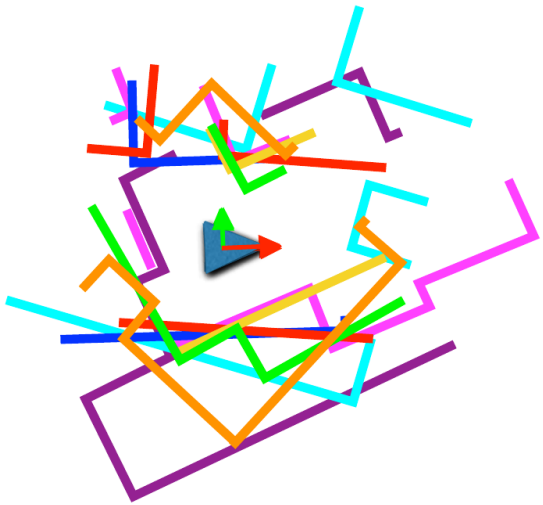


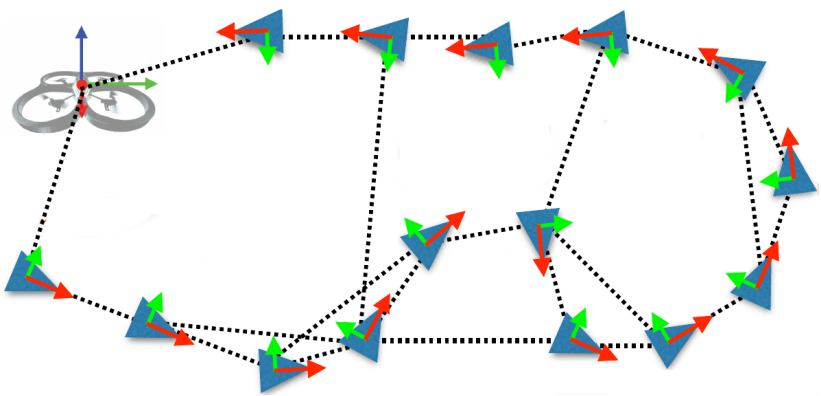


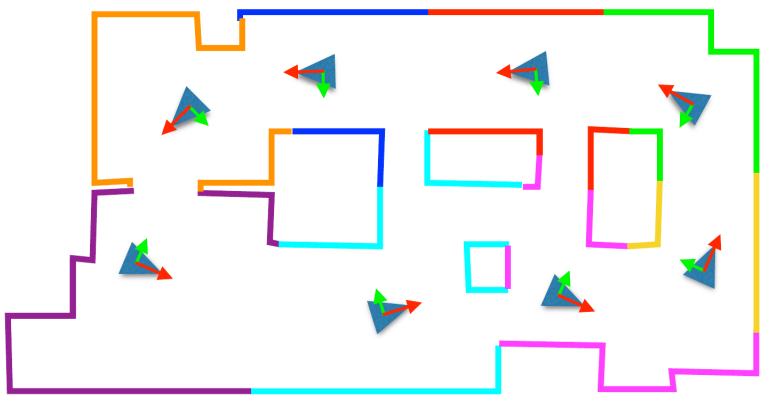
Figures courtesy of Luca Carlone

A concrete example





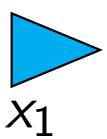




Figures courtesy of Luca Carlone

The network of spatial relations

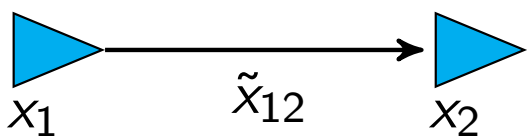
More abstractly, as the robot explores...



...we construct a graph of *noisy observations* of *spatial relations*.

The network of spatial relations

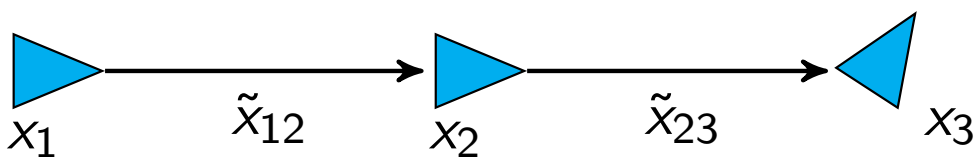
More abstractly, as the robot explores...



...we construct a graph of *noisy observations* of *spatial relations*.

The network of spatial relations

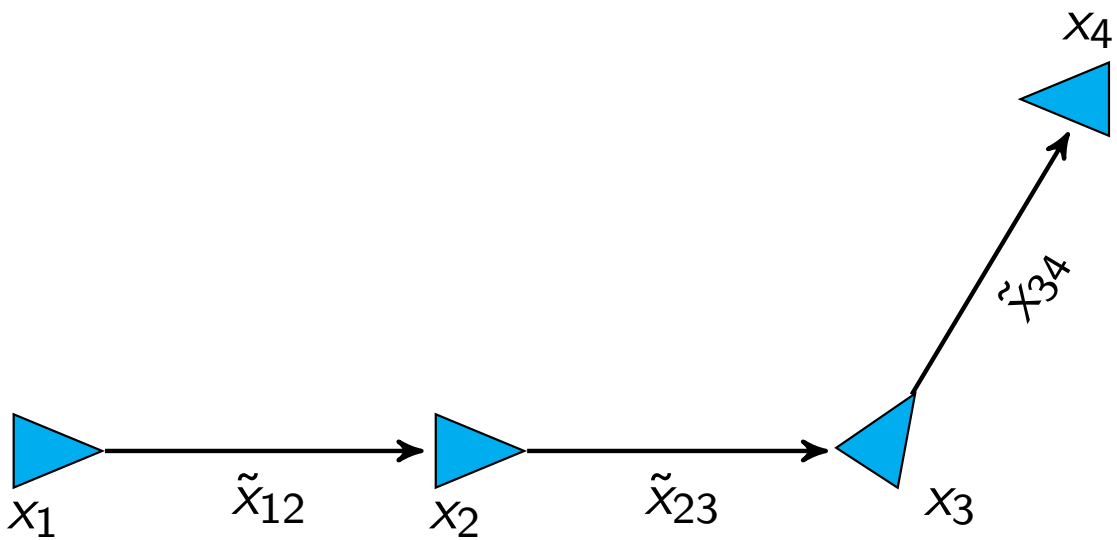
More abstractly, as the robot explores...



...we construct a graph of *noisy observations* of *spatial relations*.

The network of spatial relations

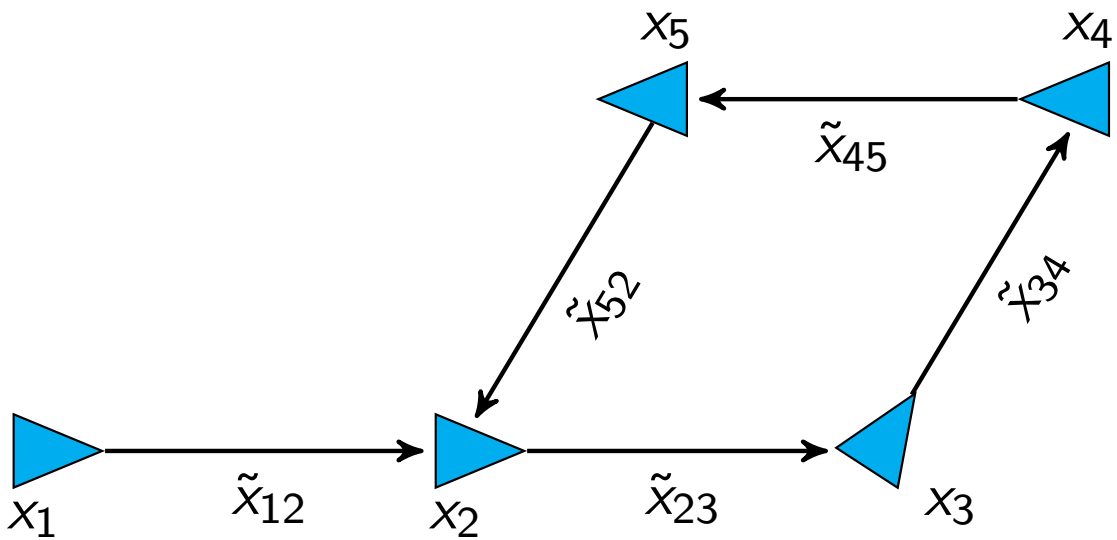
More abstractly, as the robot explores...



...we construct a graph of *noisy observations* of *spatial relations*.

The network of spatial relations

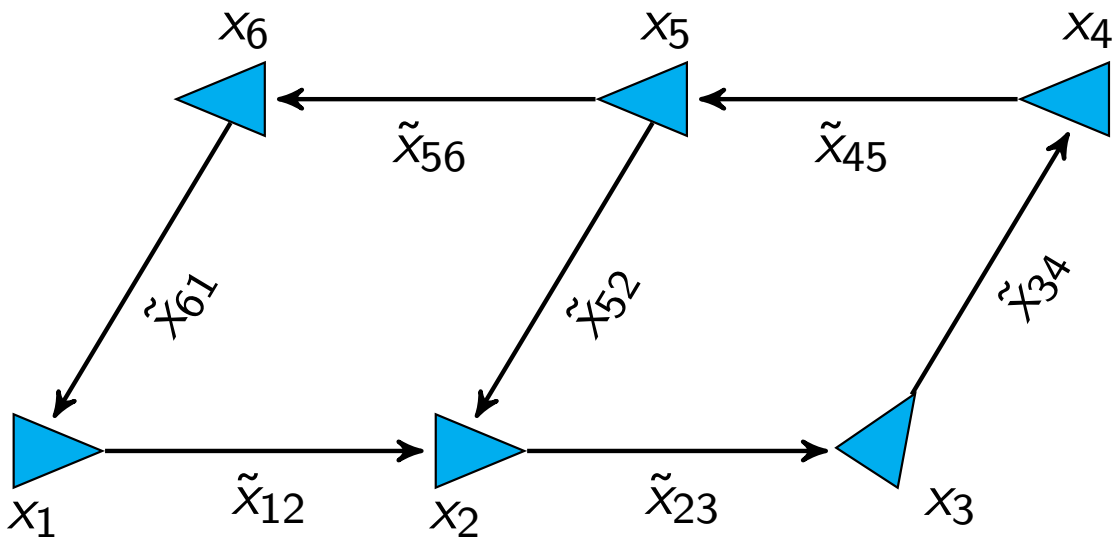
More abstractly, as the robot explores...



...we construct a graph of *noisy observations* of *spatial relations*.

The network of spatial relations

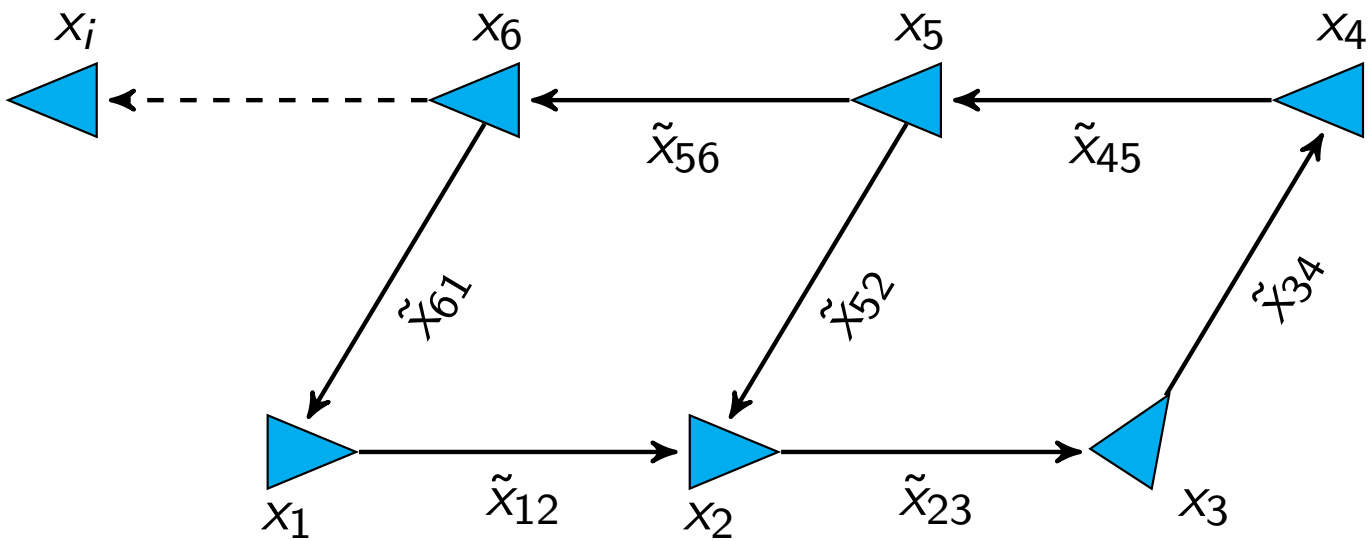
More abstractly, as the robot explores...



...we construct a graph of *noisy observations* of *spatial relations*.

The network of spatial relations

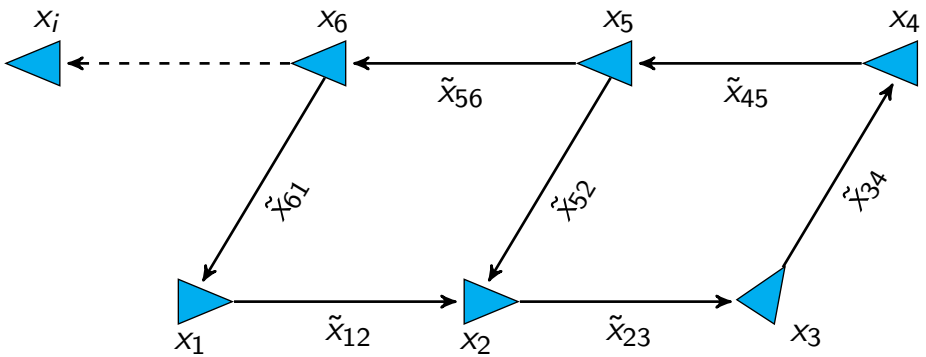
More abstractly, as the robot explores...



...we construct a graph of *noisy observations* of *spatial relations*.

Formalizing the problem: Pose graphs

A *pose-graph* is a directed graph $G = (X, \vec{\mathcal{E}})$ that models the *joint distribution* $p(\tilde{\mathcal{Y}}|X)$ of this network of noisy relative measurements:



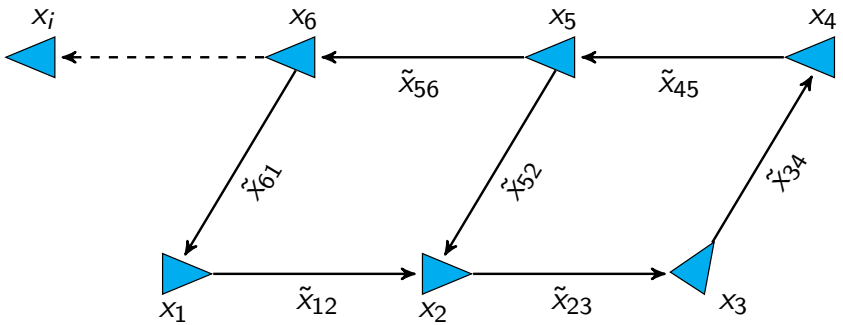
$$p(\tilde{\mathcal{Y}}|X) = \prod_{(i,j)\in\vec{\mathcal{E}}} p_{ij}(\tilde{x}_{ij}|x_i, x_j)$$

Here:

- $X = \{x_1, \ldots, x_n\}$ are the (latent) *robot poses*
- $\tilde{\mathcal{Y}} = \{\tilde{x}_{ij}\}$ are the *measurements* of relative transforms $x_i^{-1} x_j$
- $\tilde{x}_{ij} \sim p_{ij}(\cdot|x_i, x_j)$ is the *measurement model* for $\tilde{x}_{ij}$.

The pose-graph SLAM problem

Given: A pose-graph model $G = (X, \vec{\mathcal{E}})$:

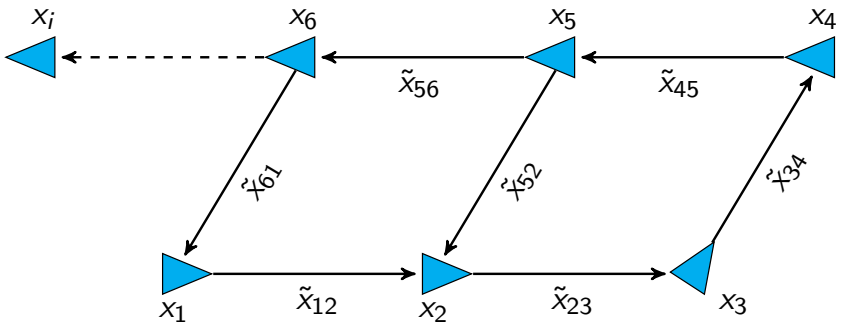


$$p(\tilde{\mathcal{Y}}|X) = \prod_{(i,j)\in\vec{\mathcal{E}}} p_{ij}(\tilde{x}_{ij}|x_i, x_j)$$

Find: A *maximum-likelihood estimate* $\hat{X}_{\mathrm{MLE}} \in \mathrm{SE}(d)^n$ for X.

The pose-graph SLAM problem

Given: A pose-graph model $G = (X, \vec{\mathcal{E}})$:



$$p(\tilde{\mathcal{Y}}|X) = \prod_{(i,j)\in\vec{\mathcal{E}}} p_{ij}(\tilde{x}_{ij}|x_i, x_j)$$

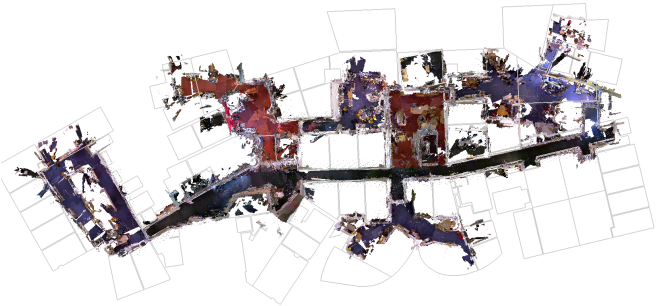
Find: A *maximum-likelihood estimate* $\hat{X}_{\text{MLE}} \in \text{SE}(d)^n$ for X:

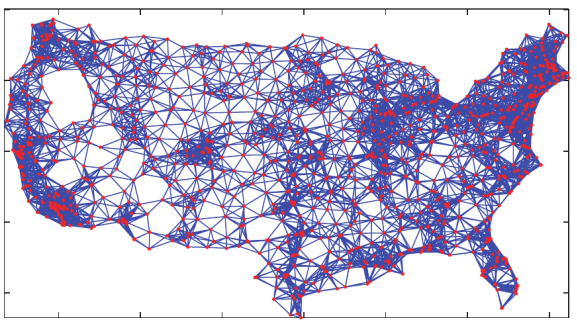
$$\hat{X}_{\text{MLE}} \in \operatorname*{argmin}_{X\in\text{SE}(d)^n} \sum_{(i,j)\in\vec{\mathcal{E}}} -\log p_{ij}(\tilde{x}_{ij}|x_i, x_j).$$

The pose-graph SLAM problem

Many important examples in geometric estimation:

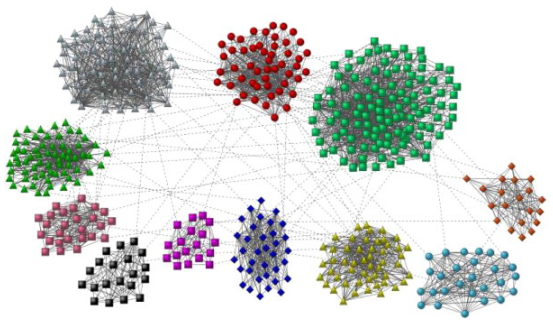
- Robotic mapping
- Camera motion estimation
- Sensor network localization





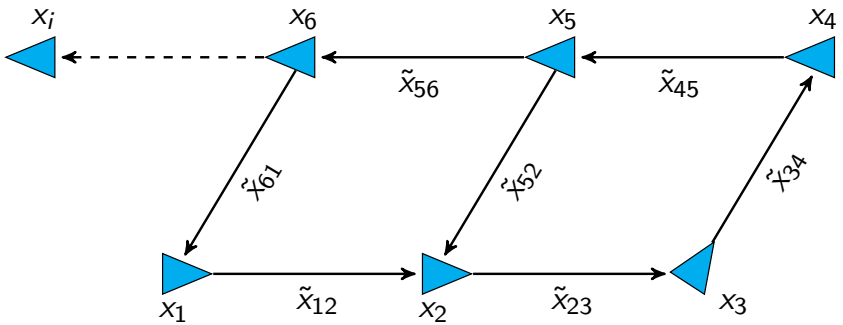
Special cases:

- Rotation averaging
- Community detection



The pose-graph SLAM problem

Given: A pose-graph model $G = (X, \vec{\mathcal{E}})$:



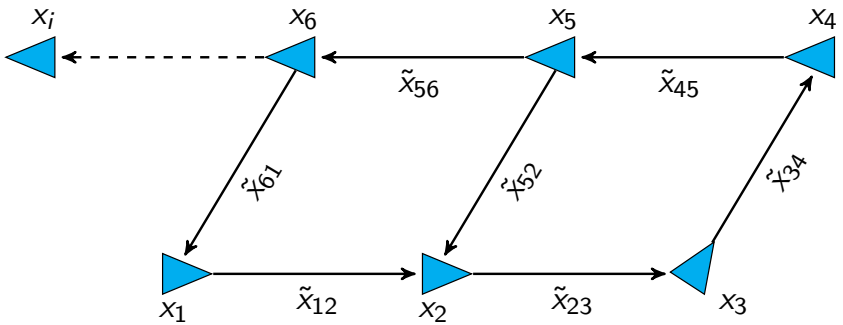
$$p(\tilde{\mathcal{Y}}|X) = \prod_{(i,j)\in\vec{\mathcal{E}}} p_{ij}(\tilde{x}_{ij}|x_i, x_j)$$

Find: A *maximum-likelihood estimate* $\hat{X}_{\text{MLE}} \in \text{SE}(d)^n$ for X:

$$\hat{X}_{\text{MLE}} \in \operatorname*{argmin}_{X\in\text{SE}(d)^n} \sum_{(i,j)\in\vec{\mathcal{E}}} -\log p_{ij}(\tilde{x}_{ij}|x_i, x_j).$$

The pose-graph SLAM problem

Given: A pose-graph model $G = (X, \vec{\mathcal{E}})$:



$$p(\tilde{\mathcal{Y}}|X) = \prod_{(i,j)\in\vec{\mathcal{E}}} p_{ij}(\tilde{x}_{ij}|x_i, x_j)$$

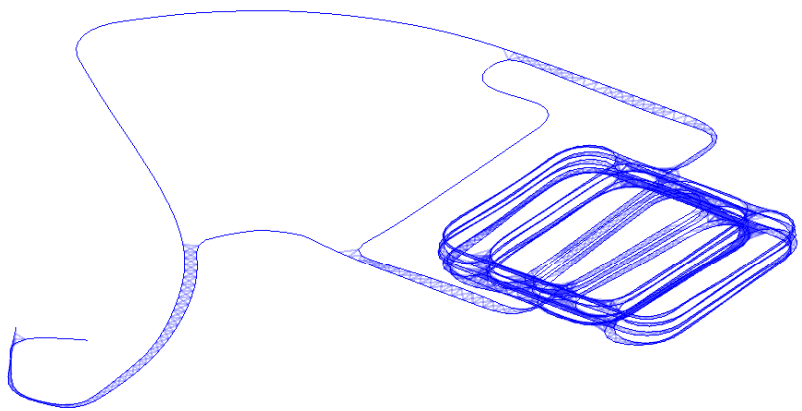
Find: A *maximum-likelihood estimate* $\hat{X}_{\mathrm{MLE}} \in \mathrm{SE}(d)^n$ for X:

$$\hat{X}_{\mathrm{MLE}} \in \operatorname*{argmin}_{X\in\mathrm{SE}(d)^n} \sum_{(i,j)\in\vec{\mathcal{E}}} -\log p_{ij}(\tilde{x}_{ij}|x_i, x_j).$$

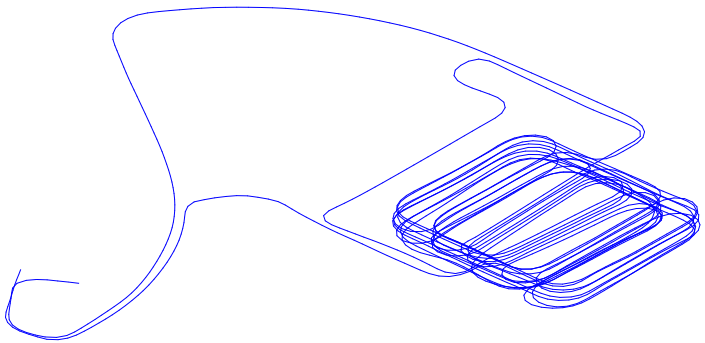
The challenge: This is a *high-dimensional*, *nonconvex* problem
$\Rightarrow$ **LOTS** of (bad) *local minima*

The problem of nonconvexity

The problem of nonconvexity

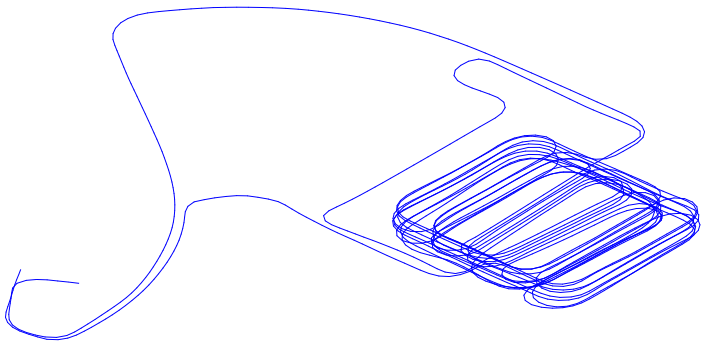


The problem of nonconvexity

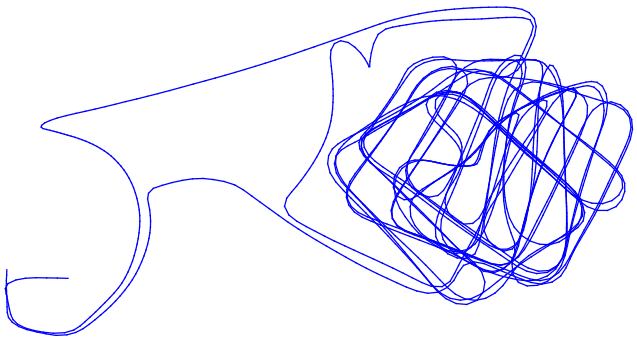


Optimal estimate

The problem of nonconvexity

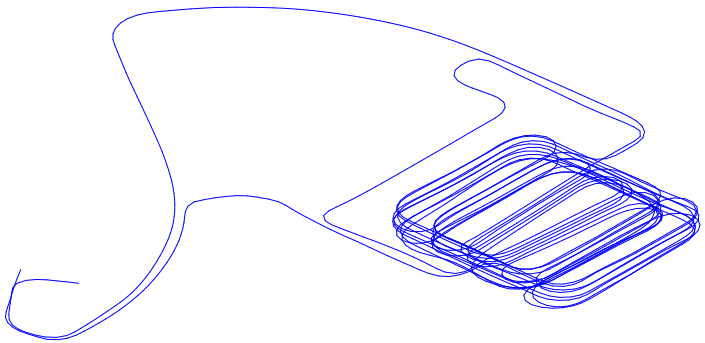


Optimal estimate

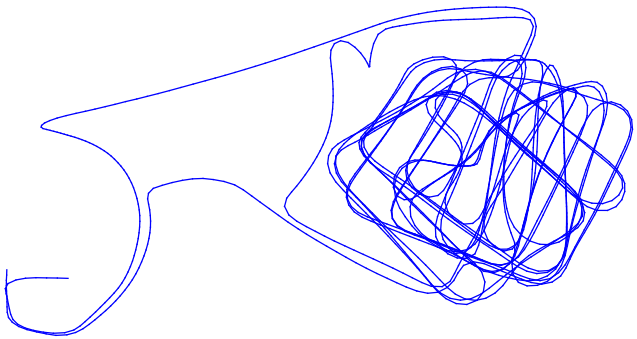


Suboptimal critical point

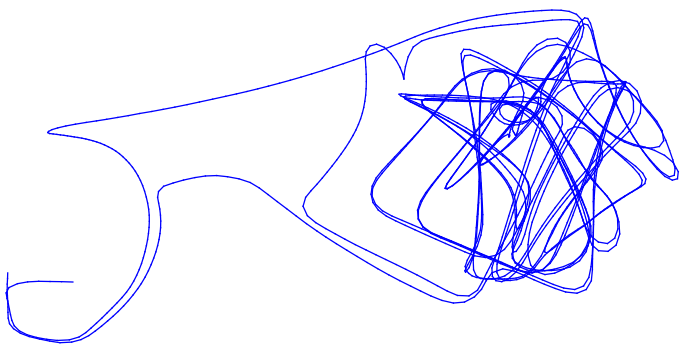
The problem of nonconvexity



Optimal estimate

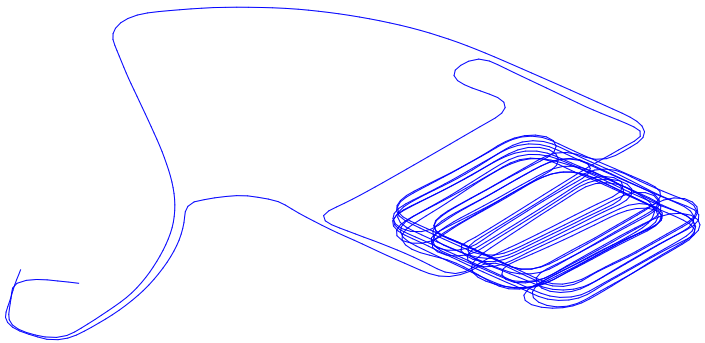


Suboptimal critical point

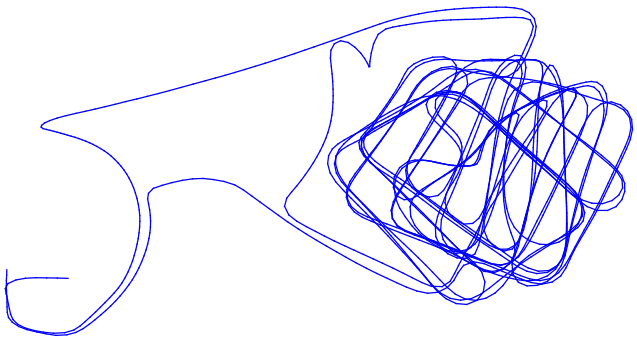


Suboptimal critical point

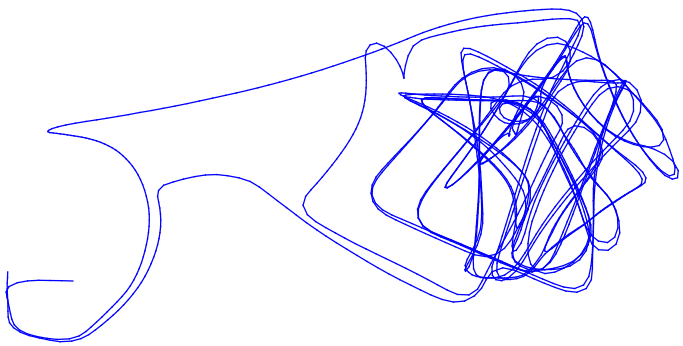
The problem of nonconvexity



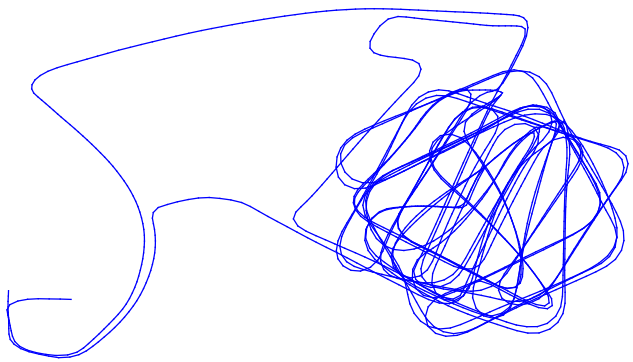
Optimal estimate



Suboptimal critical point



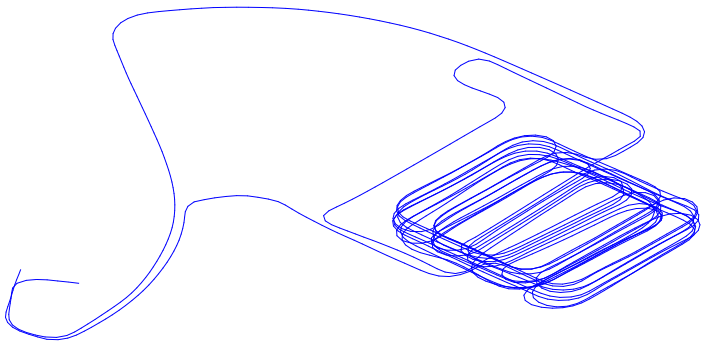
Suboptimal critical point



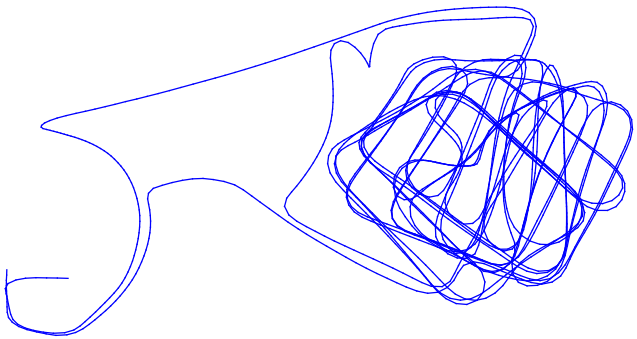
Suboptimal critical point

The problem of nonconvexity

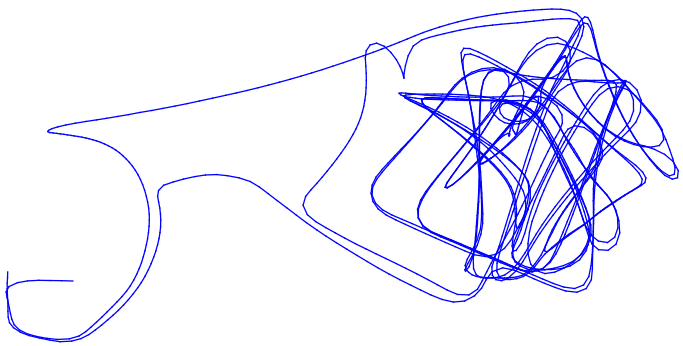
Main question: Can we *ensure* good SLAM performance?



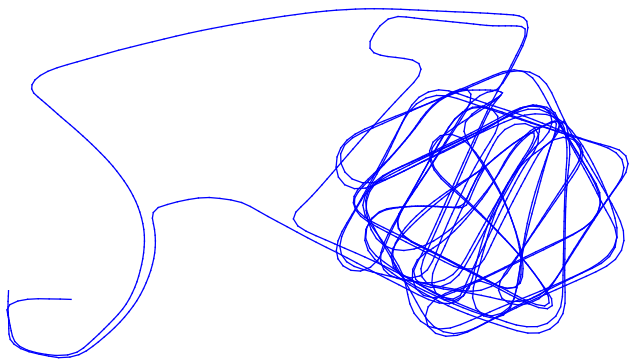
Optimal estimate



Suboptimal critical point



Suboptimal critical point



Suboptimal critical point

In the rest of this talk ...

We develop:

- A *convex relaxation* of pose-graph SLAM whose minimizer provides an *exact MLE* for moderate noise
- A fast optimization method to solve this relaxation efficiently

[1] D.M. Rosen, L. Carlone, et al. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *Intl. J. of Robotics Research* 38.2–3 (Mar. 2019), pp. 95–125

In the rest of this talk ...

We develop:

- A *convex relaxation* of pose-graph SLAM whose minimizer provides an *exact MLE* for moderate noise
- A fast optimization method to solve this relaxation efficiently

⇒**SE-Sync**: a *certifiably correct* algorithm for SLAM[1]

[1] D.M. Rosen, L. Carlone, et al. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *Intl. J. of Robotics Research* 38.2–3 (Mar. 2019), pp. 95–125

The Main Idea

We can't efficiently solve (NP-hard) pose-graph SLAM in general

[2] Bandeira 2016; Chandrasekaran, Recht, et al. 2012; Chandrasekaran and Jordan 2013.

The Main Idea

We can't efficiently solve (NP-hard) pose-graph SLAM *in general*

But: Maybe we can *well-approximate* "*reasonable*" instances?

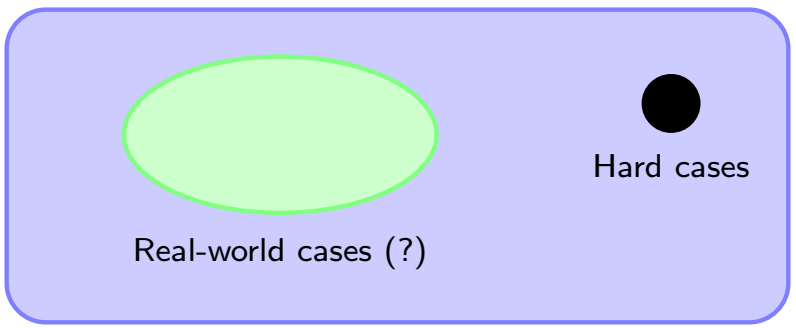
[2] Bandeira 2016; Chandrasekaran, Recht, et al. 2012; Chandrasekaran and Jordan 2013.

The Main Idea

We can't efficiently solve (NP-hard) pose-graph SLAM *in general*

But: Maybe we can *well-approximate* "*reasonable*" instances?

Intuition: Nature is not "out to get you"



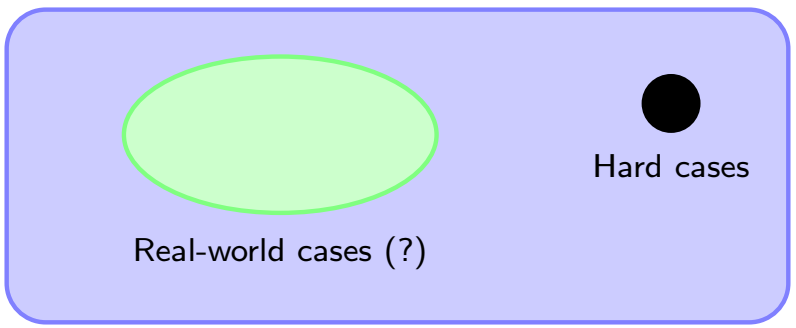
[2] Bandeira 2016; Chandrasekaran, Recht, et al. 2012; Chandrasekaran and Jordan 2013.

The Main Idea

We can't efficiently solve (NP-hard) pose-graph SLAM *in general*

But: Maybe we can *well-approximate* "*reasonable*" instances?

Intuition: Nature is not "out to get you"



Question: How to *generate* these approximations?

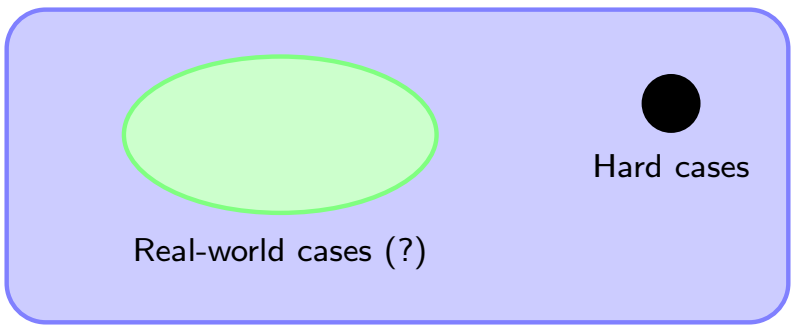
[2] Bandeira 2016; Chandrasekaran, Recht, et al. 2012; Chandrasekaran and Jordan 2013.

The Main Idea

We can't efficiently solve (NP-hard) pose-graph SLAM *in general*

But: Maybe we can *well-approximate* "*reasonable*" instances?

Intuition: Nature is not "out to get you"



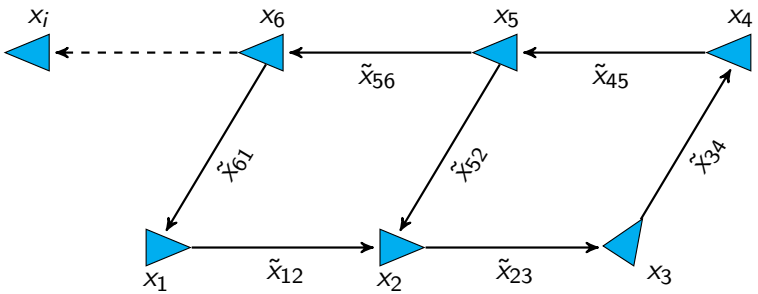
Question: How to *generate* these approximations?

One general approach: *Convex relaxation*

- Standard machinery for constructing these
- Admit (some) formal guarantees on solution quality.[2]

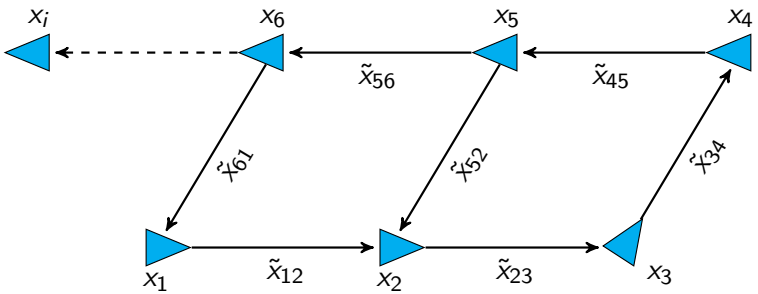
[2] Bandeira 2016; Chandrasekaran, Recht, et al. 2012; Chandrasekaran and Jordan 2013.

The "standard model" of pose-graph SLAM



$$p(\tilde{\mathcal{Y}}|X) = \prod_{(i,j)\in\vec{\mathcal{E}}} p_{ij}(\tilde{x}_{ij}|x_i, x_j)$$

The "standard model" of pose-graph SLAM

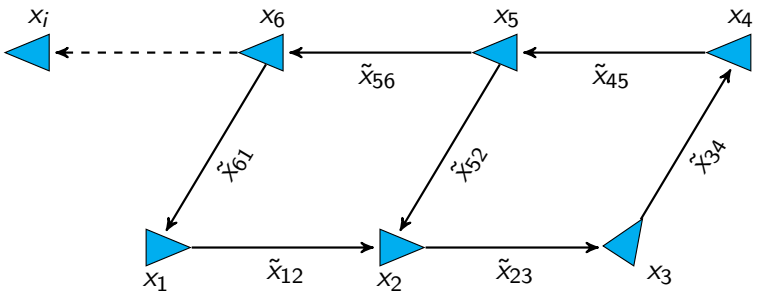


$$p(\tilde{\mathcal{Y}}|X) = \prod_{(i,j)\in\vec{\mathcal{E}}} p_{ij}(\tilde{x}_{ij}|x_i, x_j)$$

Assume the following measurement model for $\tilde{x}_{ij} = (\tilde{t}_{ij}, \tilde{R}_{ij})$:

$$\tilde{t}_{ij} = R_i^T(t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\tau})$$
$$\tilde{R}_{ij} = R_i^T R_j \cdot \exp(\delta R_{ij}), \qquad \delta R_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\rho})$$

The "standard model" of pose-graph SLAM



$$p(\tilde{\mathcal{Y}}|X) = \prod_{(i,j)\in\vec{\mathcal{E}}} p_{ij}(\tilde{x}_{ij}|x_i, x_j)$$

Assume the following measurement model for $\tilde{x}_{ij} = (\tilde{t}_{ij}, \tilde{R}_{ij})$:

$$\tilde{t}_{ij} = R_i^T(t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\tau})$$
$$\tilde{R}_{ij} = R_i^T R_j \cdot \exp(\delta R_{ij}), \qquad \delta R_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\rho})$$

$\Rightarrow$ MLE is a *sparse nonlinear least-squares* problem:

$$\hat{X}_{\text{MLE}} = \underset{\substack{t_i\in\mathbb{R}^d \\ R_i\in \text{SO}(d)}}{\text{argmin}} \sum_{(i,j)\in\vec{\mathcal{E}}} \left\| \log\left(R_j^T R_i \tilde{R}_{ij}\right) \right\|_{\Sigma_{ij}^{\rho}}^2 + \left\| \tilde{t}_{ij} - R_i^T(t_j - t_i) \right\|_{\Sigma_{ij}^{\tau}}^2 .$$

The "standard model" of pose-graph SLAM

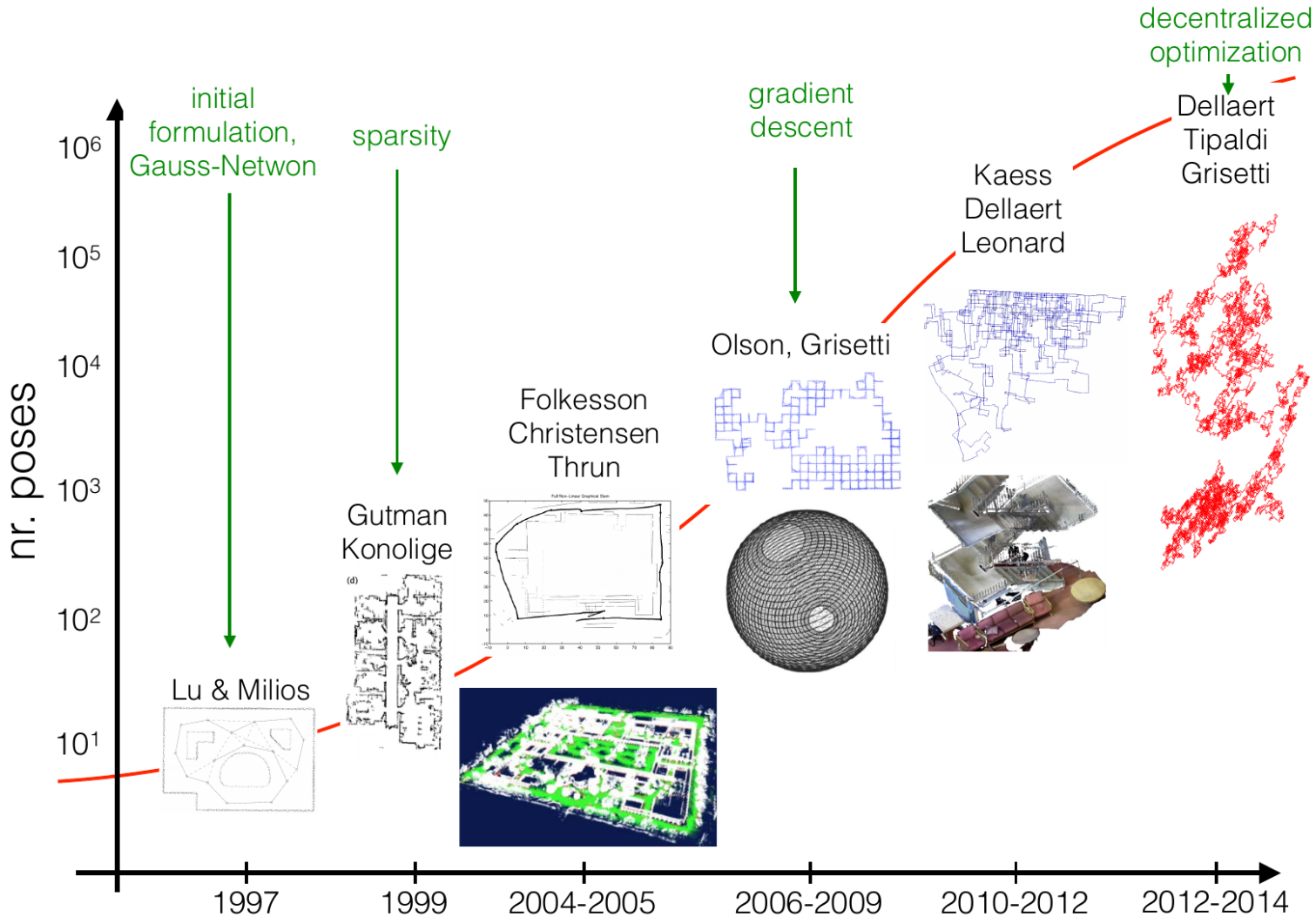
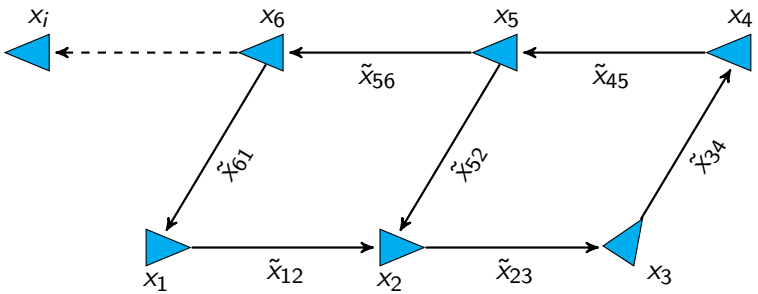


Figure courtesy of Luca Carlone

The "standard model" of pose-graph SLAM



$$p(\tilde{\mathcal{Y}}|X) = \prod_{(i,j)\in\vec{\mathcal{E}}} p_{ij}(\tilde{x}_{ij}|x_i, x_j)$$

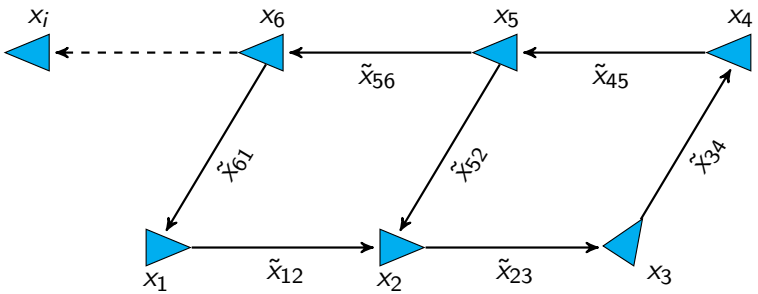
Assume the following measurement model for $\tilde{x}_{ij} = (\tilde{t}_{ij}, \tilde{R}_{ij})$:

$$\tilde{t}_{ij} = R_i^T(t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\tau})$$
$$\tilde{R}_{ij} = R_i^T R_j \cdot \exp(\delta R_{ij}), \qquad \delta R_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\rho})$$

$\Rightarrow$ MLE is a *sparse nonlinear least-squares* problem:

$$\hat{X}_{\text{MLE}} = \underset{\substack{t_i\in\mathbb{R}^d \\ R_i\in \text{SO}(d)}}{\text{argmin}} \sum_{(i,j)\in\vec{\mathcal{E}}} \left\| \log\left(R_j^T R_i \tilde{R}_{ij}\right) \right\|_{\Sigma_{ij}^{\rho}}^2 + \left\| \tilde{t}_{ij} - R_i^T(t_j - t_i) \right\|_{\Sigma_{ij}^{\tau}}^2 .$$

The "standard model" of pose-graph SLAM



$$p(\tilde{\mathcal{Y}}|X) = \prod_{(i,j)\in\vec{\mathcal{E}}} p_{ij}(\tilde{x}_{ij}|x_i, x_j)$$

Assume the following measurement model for $\tilde{x}_{ij} = (\tilde{t}_{ij}, \tilde{R}_{ij})$:

$$\tilde{t}_{ij} = R_i^T(t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\tau})$$
$$\tilde{R}_{ij} = R_i^T R_j \cdot \exp(\delta R_{ij}), \qquad \delta R_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\rho})$$

$\Rightarrow$ MLE is a *sparse nonlinear least-squares* problem:

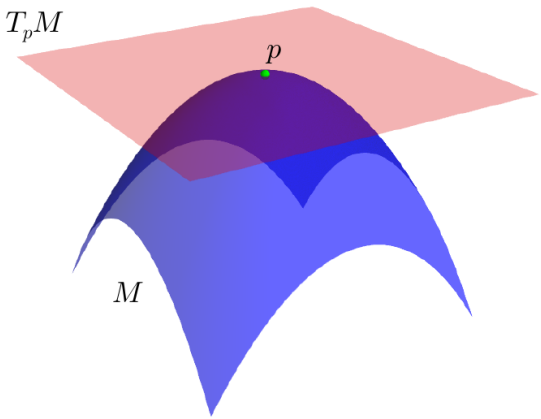
$$\hat{X}_{\mathrm{MLE}} = \underset{\substack{t_i\in\mathbb{R}^d \\ R_i\in \mathrm{SO}(d)}}{\operatorname{argmin}} \sum_{(i,j)\in\vec{\mathcal{E}}} \left\| \log\left(R_j^T R_i \tilde{R}_{ij}\right) \right\|_{\Sigma_{ij}^{\rho}}^2 + \left\| \tilde{t}_{ij} - R_i^T(t_j - t_i) \right\|_{\Sigma_{ij}^{\tau}}^2 .$$

But: How to handle nonconvexity?

Step 1: Reformulating pose-graph SLAM

Standard model assumes *exponentiated Gaussian* rotational noise:

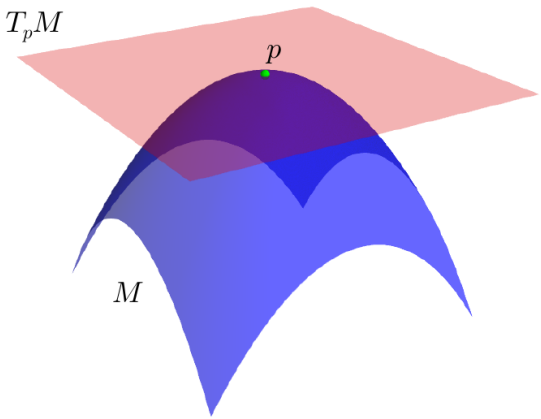
$$\tilde{R}_{ij} = R_i^T R_j \cdot \exp(\delta R_{ij}),$$
$$\delta R_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\rho})$$



Step 1: Reformulating pose-graph SLAM

Standard model assumes *exponentiated Gaussian* rotational noise:

$$\tilde{R}_{ij} = R_i^T R_j \cdot \exp(\delta R_{ij}),$$
$$\delta R_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\rho})$$

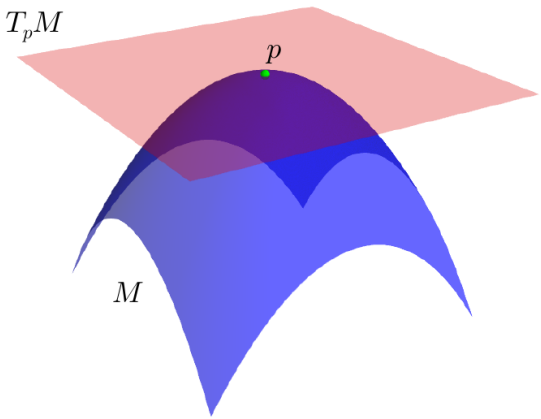


Main insight: This is not a "canonical" choice

Step 1: Reformulating pose-graph SLAM

Standard model assumes *exponentiated Gaussian* rotational noise:

$$\tilde{R}_{ij} = R_i^T R_j \cdot \exp(\delta R_{ij}),$$
$$\delta R_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\rho})$$



Main insight: This is not a "canonical" choice

$\Rightarrow$ Perhaps there is a more convenient formulation ...

Step 1: Reformulating pose-graph SLAM

Our proposal: Use an *exponential family* distribution on $\mathrm{SO}(d)$.

Step 1: Reformulating pose-graph SLAM

Our proposal: Use an *exponential family* distribution on $\mathrm{SO}(d)$.

Isotropic Langevin distribution:

$$p(R; M, \kappa) = \frac{1}{c_d(\kappa)} \exp\left(\kappa \operatorname{tr}(M^{\mathsf{T}} R)\right), \quad R \in \mathrm{SO}(d)$$

w.r.t. Haar measure on $\mathrm{SO}(d)$.

Step 1: Reformulating pose-graph SLAM

Our proposal: Use an *exponential family* distribution on $\mathrm{SO}(d)$.
Isotropic Langevin distribution:

$$p(R; M, \kappa) = \frac{1}{c_d(\kappa)} \exp\left(\kappa \operatorname{tr}(M^{\mathsf{T}} R)\right), \quad R \in \mathrm{SO}(d)$$

w.r.t. Haar measure on $\mathrm{SO}(d)$.

Generative model (3D):

- Sample *axis* $\hat{v} \sim \mathcal{U}(S^2)$

Step 1: Reformulating pose-graph SLAM

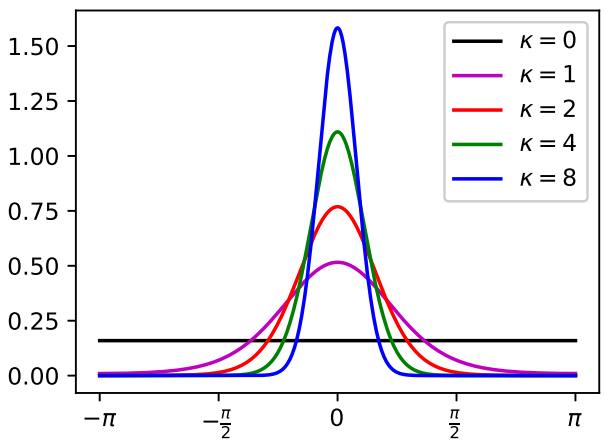
Our proposal: Use an *exponential family* distribution on $\mathrm{SO}(d)$.
Isotropic Langevin distribution:

$$p(R; M, \kappa) = \frac{1}{c_d(\kappa)} \exp\left(\kappa \operatorname{tr}(M^{\mathsf{T}} R)\right), \quad R \in \mathrm{SO}(d)$$

w.r.t. Haar measure on $\mathrm{SO}(d)$.

Generative model (3D):

- Sample *axis* $\hat{v} \sim \mathcal{U}(S^2)$
- Sample *angle* $\theta \sim \mathrm{vonMises}(0, 2\kappa)$



Step 1: Reformulating pose-graph SLAM

Our proposal: Use an *exponential family* distribution on $\mathrm{SO}(d)$.

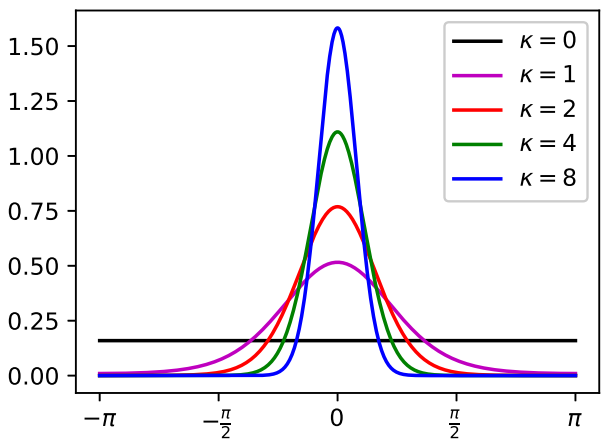
Isotropic Langevin distribution:

$$p(R; M, \kappa) = \frac{1}{c_d(\kappa)} \exp\left(\kappa \operatorname{tr}(M^{\mathsf{T}} R)\right), \quad R \in \mathrm{SO}(d)$$

w.r.t. Haar measure on $\mathrm{SO}(d)$.

Generative model (3D):

- Sample *axis* $\hat{v} \sim \mathcal{U}(S^2)$
- Sample *angle* $\theta \sim \mathrm{vonMises}(0, 2\kappa)$
- Return $R = \exp(\theta[\hat{v}]_\times)$



Step 1: Reformulating pose-graph SLAM

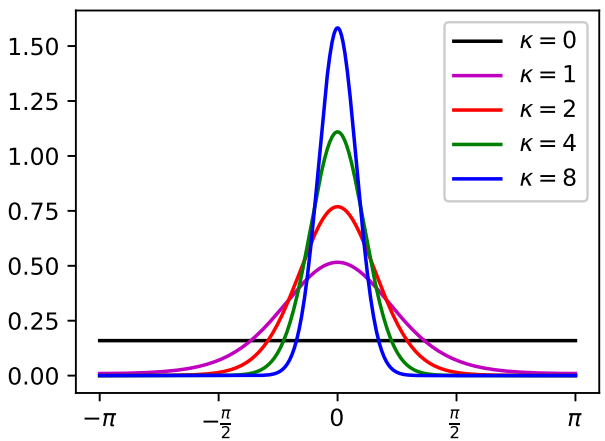
Our proposal: Use an *exponential family* distribution on $\mathrm{SO}(d)$.
Isotropic Langevin distribution:

$$p(R; M, \kappa) = \frac{1}{c_d(\kappa)} \exp\left(\kappa \operatorname{tr}(M^{\mathsf{T}} R)\right), \quad R \in \mathrm{SO}(d)$$

w.r.t. Haar measure on $\mathrm{SO}(d)$.

Generative model (3D):

- Sample *axis* $\hat{v} \sim \mathcal{U}(S^2)$
- Sample *angle* $\theta \sim \mathrm{vonMises}(0, 2\kappa)$
- Return $R = \exp(\theta[\hat{v}]_\times)$



Payoffs:

- $\mathrm{vonMises}(0, 2\kappa) \to \mathcal{N}(0, 1/2\kappa)$ as $\kappa \to \infty$.
  $\Rightarrow$ Isotropic Langevin "looks like" exp. Gaussian for low noise

Step 1: Reformulating pose-graph SLAM

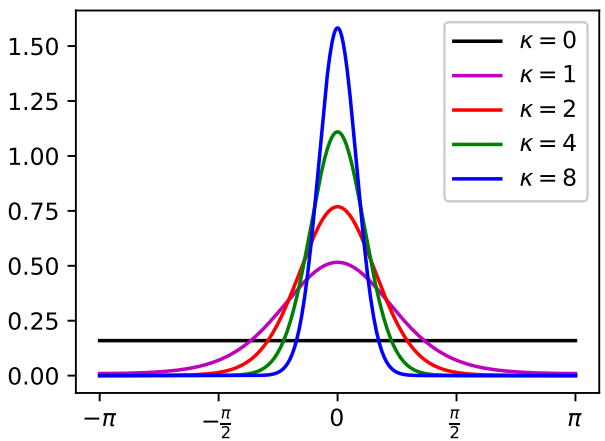
Our proposal: Use an *exponential family* distribution on $\mathrm{SO}(d)$.
Isotropic Langevin distribution:

$$p(R; M, \kappa) = \frac{1}{c_d(\kappa)} \exp\left(\kappa \operatorname{tr}(M^{\mathsf{T}} R)\right), \quad R \in \mathrm{SO}(d)$$

w.r.t. Haar measure on $\mathrm{SO}(d)$.

Generative model (3D):

- Sample *axis* $\hat{v} \sim \mathcal{U}(S^2)$
- Sample *angle* $\theta \sim \mathrm{vonMises}(0, 2\kappa)$
- Return $R = \exp(\theta[\hat{v}]_\times)$



Payoffs:

- $\mathrm{vonMises}(0, 2\kappa) \to \mathcal{N}(0, 1/2\kappa)$ as $\kappa \to \infty$.
  $\Rightarrow$ Isotropic Langevin "looks like" exp. Gaussian for low noise
- Exponential family $\Rightarrow$ MLE has simple algebraic form

Step 1: Reformulating pose-graph SLAM

Standard model:

$$\tilde{t}_{ij} = R_i^T (t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\tau})$$

$$\tilde{R}_{ij} = R_i^T R_j \cdot \exp(\delta R_{ij}), \qquad \delta R_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\rho})$$

$$\hat{X}_{\mathrm{MLE}} = \underset{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}}{\operatorname{argmin}} \sum_{(i,j) \in \vec{\mathcal{E}}} \left\| \log\left(R_j^T R_i \tilde{R}_{ij} \right) \right\|_{\Sigma_{ij}^{\rho}}^2 + \left\| \tilde{t}_{ij} - R_i^T (t_j - t_i) \right\|_{\Sigma_{ij}^{\tau}}^2 .$$

Step 1: Reformulating pose-graph SLAM

Standard model:

$$\tilde{t}_{ij} = R_i^T(t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\tau})$$
$$\tilde{R}_{ij} = R_i^T R_j \cdot \exp(\delta R_{ij}), \qquad \delta R_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\rho})$$

$$\hat{X}_{\mathrm{MLE}} = \underset{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}}{\operatorname{argmin}} \sum_{(i,j)\in\vec{\mathcal{E}}} \left\| \log\left(R_j^T R_i \tilde{R}_{ij}\right) \right\|_{\Sigma_{ij}^{\rho}}^2 + \left\| \tilde{t}_{ij} - R_i^T(t_j - t_i) \right\|_{\Sigma_{ij}^{\tau}}^2.$$

Our proposal:

$$\bar{t}_{ij} = R_i^T(t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \tau_{ij}^{-1} I_d)$$
$$\bar{R}_{ij} = R_i^T R_j \cdot \delta R_{ij}, \qquad \delta R_{ij} \sim \mathrm{Langevin}(I_d, \kappa_{ij})$$

$$\hat{X}_{\mathrm{MLE}} = \underset{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}}{\operatorname{argmin}} \sum_{(i,j)\in\vec{\mathcal{E}}} \left\| R_j - R_i \tilde{R}_{ij} \right\|_F^2 + \left\| \tilde{t}_{ij} - R_i^T(t_j - t_i) \right\|_{\Sigma_{ij}^{\tau}}^2.$$

Step 1: Reformulating pose-graph SLAM

Standard model:

$$\tilde{t}_{ij} = R_i^T(t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\tau})$$
$$\tilde{R}_{ij} = R_i^T R_j \cdot \exp(\delta R_{ij}), \qquad \delta R_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\rho})$$

$$\hat{X}_{\mathrm{MLE}} = \underset{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}}{\operatorname{argmin}} \sum_{(i,j)\in\vec{\mathcal{E}}} \left\| \log\left(R_j^T R_i \tilde{R}_{ij}\right) \right\|_{\Sigma_{ij}^{\rho}}^2 + \left\| \tilde{t}_{ij} - R_i^T(t_j - t_i) \right\|_{\Sigma_{ij}^{\tau}}^2 .$$

Our proposal:

$$\bar{t}_{ij} = R_i^T(t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \tau_{ij}^{-1} I_d)$$
$$\bar{R}_{ij} = R_i^T R_j \cdot \delta R_{ij}, \qquad \delta R_{ij} \sim \mathrm{Langevin}(I_d, \kappa_{ij})$$

$$\hat{X}_{\mathrm{MLE}} = \underset{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}}{\operatorname{argmin}} \sum_{(i,j)\in\vec{\mathcal{E}}} \left\| R_j - R_i \tilde{R}_{ij} \right\|_F^2 + \left\| \tilde{t}_{ij} - R_i^T(t_j - t_i) \right\|_{\Sigma_{ij}^{\tau}}^2 .$$

Step 1: Reformulating pose-graph SLAM

Standard model:

$$\tilde{t}_{ij} = R_i^T(t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\tau})$$
$$\tilde{R}_{ij} = R_i^T R_j \cdot \exp(\delta R_{ij}), \qquad \delta R_{ij} \sim \mathcal{N}(0, \Sigma_{ij}^{\rho})$$

$$\hat{X}_{\mathrm{MLE}} = \underset{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}}{\operatorname{argmin}} \sum_{(i,j)\in\vec{\mathcal{E}}} \left\| \log\left(R_j^T R_i \tilde{R}_{ij}\right) \right\|_{\Sigma_{ij}^{\rho}}^2 + \left\| \tilde{t}_{ij} - R_i^T(t_j - t_i) \right\|_{\Sigma_{ij}^{\tau}}^2.$$

Our proposal:

$$\bar{t}_{ij} = R_i^T(t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \tau_{ij}^{-1} I_d)$$
$$\bar{R}_{ij} = R_i^T R_j \cdot \delta R_{ij}, \qquad \delta R_{ij} \sim \mathrm{Langevin}(I_d, \kappa_{ij})$$

$$\hat{X}_{\mathrm{MLE}} = \underset{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}}{\operatorname{argmin}} \sum_{(i,j)\in\vec{\mathcal{E}}} \left\| R_j - R_i \tilde{R}_{ij} \right\|_F^2 + \left\| \tilde{t}_{ij} - R_i^T(t_j - t_i) \right\|_{\Sigma_{ij}^{\tau}}^2.$$

Payoff: Our formulation has a *convex quadratic* objective.

Step 2: Simplifying the maximum-likelihood estimation

$$p^*_{\mathrm{MLE}} = \min_{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}} \sum_{(i,j)\in\vec{\mathcal{E}}} \kappa_{ij} \left\| R_j - R_i \tilde{R}_{ij} \right\|_F^2 + \tau_{ij} \left\| t_j - t_i - R_i \tilde{t}_{ij} \right\|_2^2$$

Step 2: Simplifying the maximum-likelihood estimation

$$p^*_{\mathrm{MLE}} = \min_{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}} \sum_{(i,j)\in\vec{\mathcal{E}}} \kappa_{ij} \left\| R_j - R_i \tilde{R}_{ij} \right\|_F^2 + \tau_{ij} \left\| t_j - t_i - R_i \tilde{t}_{ij} \right\|_2^2$$

Step 2: Simplifying the maximum-likelihood estimation

$$p^*_{\text{MLE}} = \min_{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}} \sum_{(i,j)\in\vec{\mathcal{E}}} \kappa_{ij} \left\| R_j - R_i \tilde{R}_{ij} \right\|_F^2 + \tau_{ij} \left\| t_j - t_i - R_i \tilde{t}_{ij} \right\|_2^2$$

Solving for $t \triangleq (t_1, \ldots, t_n)$ in terms of $R \triangleq (R_1, \ldots, R_n)$ using a generalized Schur complement...

Step 2: Simplifying the maximum-likelihood estimation

$$p^*_{\text{MLE}} = \min_{\substack{t_i \in \mathbb{R}^d \\ R_i \in \text{SO}(d)}} \sum_{(i,j)\in\vec{\mathcal{E}}} \kappa_{ij} \left\| R_j - R_i \tilde{R}_{ij} \right\|_F^2 + \tau_{ij} \left\| t_j - t_i - R_i \tilde{t}_{ij} \right\|_2^2$$

Solving for $t \triangleq (t_1, \ldots, t_n)$ in terms of $R \triangleq (R_1, \ldots, R_n)$ using a generalized Schur complement...

[Lots of algebra...]

Step 2: Simplifying the maximum-likelihood estimation

$$p^*_{\mathrm{MLE}} = \min_{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}} \sum_{(i,j)\in\vec{\mathcal{E}}} \kappa_{ij} \left\| R_j - R_i \tilde{R}_{ij} \right\|_F^2 + \tau_{ij} \left\| t_j - t_i - R_i \tilde{t}_{ij} \right\|_2^2$$

Solving for $t \triangleq (t_1, \ldots, t_n)$ in terms of $R \triangleq (R_1, \ldots, R_n)$ using a generalized Schur complement:

Simplified ML estimation

$$p^*_{\mathrm{MLE}} = \min_{R \in \mathrm{SO}(d)^n} \mathrm{tr}(\tilde{Q} R^{\mathsf{T}} R)$$

$$\tilde{Q} = L(\tilde{G}^\rho) + \tilde{T}^{\mathsf{T}} \Omega^{\frac{1}{2}} \Pi \Omega^{\frac{1}{2}} \tilde{T}$$

Step 2: Simplifying the maximum-likelihood estimation

$$p^*_{\text{MLE}} = \min_{\substack{t_i \in \mathbb{R}^d \\ R_i \in \text{SO}(d)}} \sum_{(i,j)\in\vec{\mathcal{E}}} \kappa_{ij} \left\| R_j - R_i \tilde{R}_{ij} \right\|_F^2 + \tau_{ij} \left\| t_j - t_i - R_i \tilde{t}_{ij} \right\|_2^2$$

Solving for $t \triangleq (t_1, \ldots, t_n)$ in terms of $R \triangleq (R_1, \ldots, R_n)$ using a generalized Schur complement:

Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\mathsf{T}} R)$$

$$\tilde{Q} = L(\tilde{G}^\rho) + \tilde{T}^{\mathsf{T}} \Omega^{\frac{1}{2}} \Pi \Omega^{\frac{1}{2}} \tilde{T}$$

Payoffs:

- Simplified MLE is over (*compact*) *rotations* only

Step 2: Simplifying the maximum-likelihood estimation

$$p^*_{\text{MLE}} = \min_{\substack{t_i \in \mathbb{R}^d \\ R_i \in \text{SO}(d)}} \sum_{(i,j)\in\vec{\mathcal{E}}} \kappa_{ij} \left\| R_j - R_i \tilde{R}_{ij} \right\|_F^2 + \tau_{ij} \left\| t_j - t_i - R_i \tilde{t}_{ij} \right\|_2^2$$

Solving for $t \triangleq (t_1, \ldots, t_n)$ in terms of $R \triangleq (R_1, \ldots, R_n)$ using a generalized Schur complement:

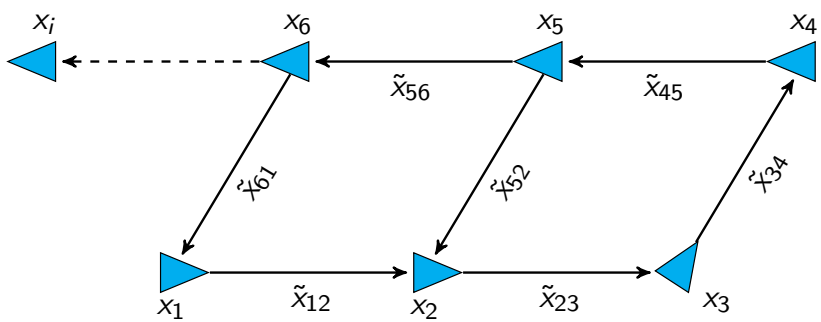
Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\mathsf{T}} R)$$

$$\tilde{Q} = L(\tilde{G}^\rho) + \tilde{T}^{\mathsf{T}} \Omega^{\frac{1}{2}} \Pi \Omega^{\frac{1}{2}} \tilde{T}$$

Payoffs:

- Simplified MLE is over (*compact*) *rotations* only
- Data matrices $L(\tilde{G}^\rho)$, $\tilde{T}$, Ω, Π have *simple interpretations* in G



Step 3: Forming the semidefinite relaxation

Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

Step 3: Forming the semidefinite relaxation

Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\mathsf{T}} R)$$

Step 3: Forming the semidefinite relaxation

Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\mathsf{T}} R)$$

$$R^{\mathsf{T}} R = \begin{pmatrix} I_d & * & \cdots & * \\ * & I_d & & * \\ \vdots & & \ddots & \vdots \\ * & * & \cdots & I_d \end{pmatrix} \succeq 0.$$

Step 3: Forming the semidefinite relaxation

Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\mathsf{T}} R)$$

$$R^{\mathsf{T}} R = \begin{pmatrix} I_d & * & \cdots & * \\ * & I_d & & * \\ \vdots & & \ddots & \vdots \\ * & * & \cdots & I_d \end{pmatrix} \succeq 0.$$

$$\therefore \quad R^{\mathsf{T}} R \in \left\{ Z \in \mathbb{S}^{dn}_+ \mid \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d) \right\} \triangleq \mathcal{C}$$

Step 3: Forming the semidefinite relaxation

Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\mathsf{T}} R)$$

$$R^{\mathsf{T}} R = \begin{pmatrix} I_d & * & \cdots & * \\ * & I_d & & * \\ \vdots & & \ddots & \vdots \\ * & * & \cdots & I_d \end{pmatrix} \succeq 0.$$

$$\therefore \quad R^{\mathsf{T}} R \in \left\{ Z \in \mathbb{S}^{dn}_+ \mid \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d) \right\} \triangleq \mathcal{C}$$

But $\mathcal{C}$ is a *spectrahedron*, and therefore **convex**

Step 3: Forming the semidefinite relaxation

Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\mathsf{T}} R)$$

Semidefinite relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

$$R^{\mathsf{T}} R = \begin{pmatrix} I_d & * & \cdots & * \\ * & I_d & & * \\ \vdots & & \ddots & \vdots \\ * & * & \cdots & I_d \end{pmatrix} \succeq 0.$$

$$\therefore \quad R^{\mathsf{T}} R \in \left\{ Z \in \mathbb{S}^{dn}_+ \mid \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d) \right\} \triangleq \mathcal{C}$$

But $\mathcal{C}$ is a *spectrahedron*, and therefore **convex**

$\Rightarrow$ *Expanding MLE's feasible set to* $\mathcal{C}$ gives a **convex relaxation**

The Main Idea

Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

Semidefinite relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

Payoffs:

- $p^*_{\text{SDP}} \leq p^*_{\text{MLE}}$ (suboptimality lower bound)

The Main Idea

Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

Semidefinite relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

Payoffs:

- $p^*_{\text{SDP}} \leq p^*_{\text{MLE}}$ (suboptimality lower bound)
- $Z^* = R^{\text{T}} R$ with $R \in \text{SO}(d)^n \Rightarrow R$ is MLE

The Main Idea

Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

Semidefinite relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

Payoffs:

- $p^*_{\text{SDP}} \leq p^*_{\text{MLE}}$ (suboptimality lower bound)
- $Z^* = R^{\text{T}} R$ with $R \in \text{SO}(d)^n \Rightarrow R$ is MLE

Proposition (Rosen, Carlone, et al. 2016)

Let $\underline{Q}$ be the matrix constructed using the true relative transforms $\underline{x}_{ij}$. There is a constant $\beta \triangleq \beta(\underline{Q}) > 0$ s.t., if $\|\tilde{Q} - \underline{Q}\|_2 < \beta$, then:

(i) The semidefinite relaxation has a unique solution Z^*, and

(ii) $Z^* = R^{*\text{T}} R^*$, where $R^* \in \text{SO}(d)^n$ is a minimizer of MLE.

The Main Idea

Simplified ML estimation

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

Semidefinite relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

Payoffs:

- $p^*_{\text{SDP}} \leq p^*_{\text{MLE}}$ (suboptimality lower bound)
- $Z^* = R^{\text{T}} R$ with $R \in \text{SO}(d)^n \Rightarrow R$ is MLE

Proposition (Rosen, Carlone, et al. 2016)

Let $\underline{Q}$ be the matrix constructed using the true relative transforms $\underline{x}_{ij}$. There is a constant $\beta \triangleq \beta(\underline{Q}) > 0$ s.t., if $\|\tilde{Q} - \underline{Q}\|_2 < \beta$, then:

(i) The semidefinite relaxation has a unique solution Z^*, and

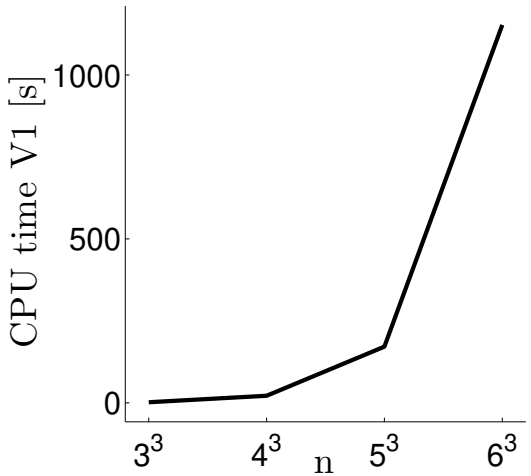
(ii) $Z^* = R^{*\text{T}} R^*$, where $R^* \in \text{SO}(d)^n$ is a minimizer of MLE.

$\Rightarrow$ We can recover an *exact* MLE by (numerically) solving SDP!

The direct approach

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \operatorname{tr}(\tilde{Q}Z)$$

$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

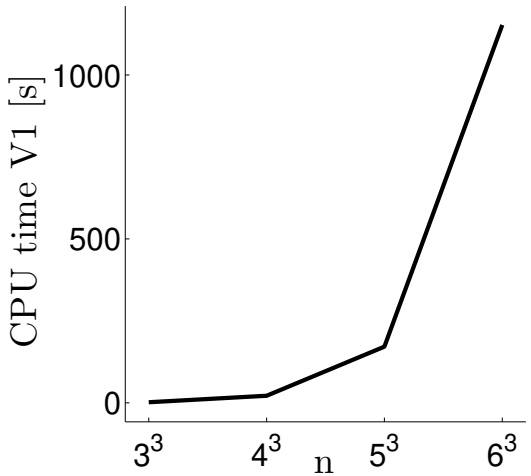


[3] L. Vandenberghe and S. Boyd. "Semidefinite Programming". In: *SIAM Review* 38.1 (Mar. 1996), pp. 49–95.

The direct approach

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \operatorname{tr}(\tilde{Q}Z)$$

$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

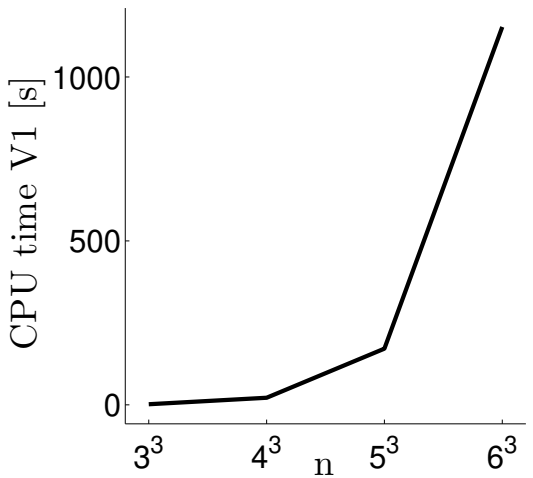


[3] L. Vandenberghe and S. Boyd. "Semidefinite Programming". In: *SIAM Review* 38.1 (Mar. 1996), pp. 49–95.

The direct approach

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \operatorname{tr}(\tilde{Q}Z)$$

$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$



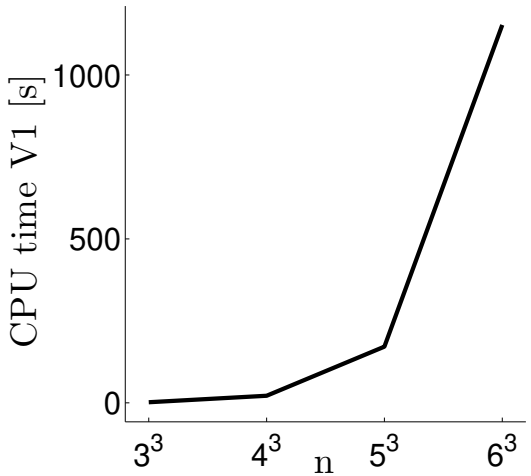
General SDP of order n:

[3] L. Vandenberghe and S. Boyd. "Semidefinite Programming". In: *SIAM Review* 38.1 (Mar. 1996), pp. 49–95.

The direct approach

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \operatorname{tr}(\tilde{Q}Z)$$

$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$



General SDP of order n:

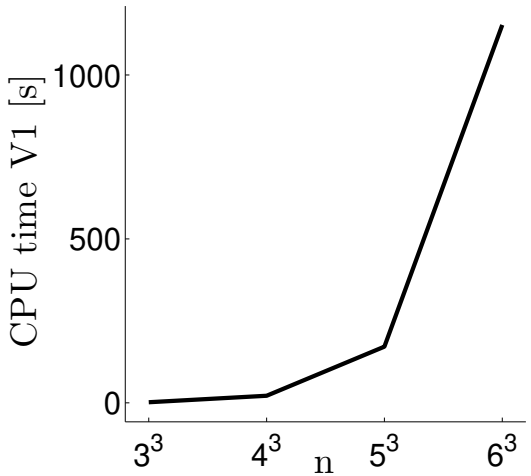
$\Rightarrow$ Matrices have dimension $O(n^2)$

[3] L. Vandenberghe and S. Boyd. "Semidefinite Programming". In: *SIAM Review* 38.1 (Mar. 1996), pp. 49–95.

The direct approach

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \operatorname{tr}(\tilde{Q}Z)$$

$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$



General SDP of order n:

$\Rightarrow$ Matrices have dimension $O(n^2)$

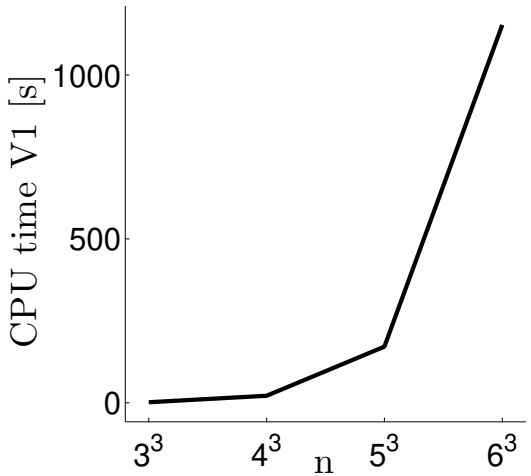
$\Rightarrow$ Newton system has dimension $O(n^4)$[3]

[3] L. Vandenberghe and S. Boyd. "Semidefinite Programming". In: *SIAM Review* 38.1 (Mar. 1996), pp. 49–95.

The direct approach

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \operatorname{tr}(\tilde{Q}Z)$$

$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$



General SDP of order n:

$\Rightarrow$ Matrices have dimension $O(n^2)$

$\Rightarrow$ Newton system has dimension $O(n^4)$[3]

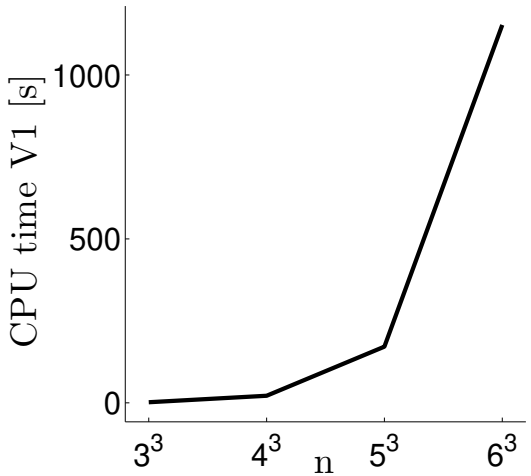
$\Rightarrow$ Solving Newton system (via factorization) has $O(n^6)$ cost

[3] L. Vandenberghe and S. Boyd. "Semidefinite Programming". In: *SIAM Review* 38.1 (Mar. 1996), pp. 49–95.

The direct approach

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \operatorname{tr}(\tilde{Q}Z)$$

$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$



General SDP of order n:

$\Rightarrow$ Matrices have dimension $O(n^2)$

$\Rightarrow$ Newton system has dimension $O(n^4)^3$

$\Rightarrow$ Solving Newton system (via factorization) has $O(n^6)$ cost

But: Maybe we can build a *specialized* solver for PGO?

[3] L. Vandenberghe and S. Boyd. "Semidefinite Programming". In: *SIAM Review* 38.1 (Mar. 1996), pp. 49–95.

Exploiting low-rank structure

We expect a *low-rank* solution $Z^* = {Y^*}^{\mathsf{T}} Y^*$ for:

$$p^*_{\mathrm{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \mathrm{tr}(\tilde{Q} Z) \quad \text{s.t.} \quad \mathrm{Diag}_d(Z) = (I_d, \ldots, I_d).$$

[4] Burer and Monteiro 2003; Burer and Monteiro 2005; Burer and Choi 2006.

Exploiting low-rank structure

We expect a *low-rank* solution $Z^* = Y^{*\mathsf{T}} Y^*$ for:

$$p^*_{\mathrm{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \mathrm{tr}(\tilde{Q} Z) \quad \text{s.t.} \quad \mathrm{Diag}_d(Z) = (I_d, \ldots, I_d).$$

Main idea: Replace Z with its *low-rank factorization* $Y^{\mathsf{T}} Y$:[4]

$$p^*_{\mathrm{SDPLR}} = \min_{Y \in \mathbb{R}^{r \times dn}} \mathrm{tr}(\tilde{Q} Y^{\mathsf{T}} Y) \quad \text{s.t.} \quad \mathrm{Diag}_d(Y^{\mathsf{T}} Y) = (I_d, \ldots, I_d)$$

[4] Burer and Monteiro 2003; Burer and Monteiro 2005; Burer and Choi 2006.

Exploiting low-rank structure

We expect a *low-rank* solution $Z^* = {Y^*}^{\mathsf{T}} Y^*$ for:

$$p^*_{\mathrm{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \mathrm{tr}(\tilde{Q} Z) \quad \text{s.t.} \quad \mathrm{Diag}_d(Z) = (I_d, \ldots, I_d).$$

Main idea: Replace Z with its *low-rank factorization* $Y^{\mathsf{T}} Y$:[4]

$$p^*_{\mathrm{SDPLR}} = \min_{Y \in \mathbb{R}^{r \times dn}} \mathrm{tr}(\tilde{Q} Y^{\mathsf{T}} Y) \quad \text{s.t.} \quad \mathrm{Diag}_d(Y^{\mathsf{T}} Y) = (I_d, \ldots, I_d)$$

Payoffs:

- $Y^{\mathsf{T}} Y \succeq 0$ for *all* $Y \Rightarrow$ PSD constraint is *redundant*
  $\Rightarrow$ Rank-restricted factorization is an *NLP* (vs. SDP)

[4] Burer and Monteiro 2003; Burer and Monteiro 2005; Burer and Choi 2006.

Exploiting low-rank structure

We expect a *low-rank* solution $Z^* = {Y^*}^{\mathsf{T}} Y^*$ for:

$$p^*_{\mathrm{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \mathrm{tr}(\tilde{Q} Z) \quad \text{s.t.} \quad \mathrm{Diag}_d(Z) = (I_d, \ldots, I_d).$$

Main idea: Replace Z with its *low-rank factorization* $Y^{\mathsf{T}} Y$:[4]

$$p^*_{\mathrm{SDPLR}} = \min_{Y \in \mathbb{R}^{r \times dn}} \mathrm{tr}(\tilde{Q} Y^{\mathsf{T}} Y) \quad \text{s.t.} \quad \mathrm{Diag}_d(Y^{\mathsf{T}} Y) = (I_d, \ldots, I_d)$$

Payoffs:

- $Y^{\mathsf{T}} Y \succeq 0$ for *all* $Y \Rightarrow$ PSD constraint is *redundant*
  $\Rightarrow$ Rank-restricted factorization is an *NLP* (vs. SDP)
- Y is *much lower-dimensional* than Z for $r \ll dn$

[4] Burer and Monteiro 2003; Burer and Monteiro 2005; Burer and Choi 2006.

Exploiting low-rank structure

We expect a *low-rank* solution $Z^* = {Y^*}^{\mathsf{T}} Y^*$ for:

$$p^*_{\mathrm{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \mathrm{tr}(\tilde{Q}Z) \quad \text{s.t.} \quad \mathrm{Diag}_d(Z) = (I_d, \ldots, I_d).$$

Main idea: Replace Z with its *low-rank factorization* $Y^{\mathsf{T}} Y$:[4]

$$p^*_{\mathrm{SDPLR}} = \min_{Y \in \mathbb{R}^{r \times dn}} \mathrm{tr}(\tilde{Q}Y^{\mathsf{T}} Y) \quad \text{s.t.} \quad \mathrm{Diag}_d(Y^{\mathsf{T}} Y) = (I_d, \ldots, I_d)$$

Payoffs:

- $Y^{\mathsf{T}} Y \succeq 0$ for *all* $Y \Rightarrow$ PSD constraint is *redundant*
  $\Rightarrow$ Rank-restricted factorization is an *NLP* (vs. SDP)
- Y is *much lower-dimensional* than Z for $r \ll dn$
- $r \geq \mathrm{rank}(Z^*)$ for some optimal $Z^* \Rightarrow (Y^*)^{\mathsf{T}} Y^*$ solves SDP.

[4] Burer and Monteiro 2003; Burer and Monteiro 2005; Burer and Choi 2006.

Exploiting geometric structure[5]

Rank-restricted SDP, NLP form

$$p^*_{\text{SDPLR}} = \min_{Y \in \mathbb{R}^{r \times dn}} \operatorname{tr}(\tilde{Q} Y^{\mathsf{T}} Y)$$

$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Y^{\mathsf{T}} Y) = (I_d, \ldots, I_d)$$

[5] N. Boumal. "A Riemannian Low-Rank Method for Optimization Over Semidefinite Matrices with Block-Diagonal Constraints". arXiv preprint: arXiv:1506.00575v2. 2015

Exploiting geometric structure[5]

Rank-restricted SDP, NLP form

$$p^*_{\text{SDPLR}} = \min_{Y \in \mathbb{R}^{r \times dn}} \operatorname{tr}(\tilde{Q} Y^{\mathsf{T}} Y)$$

$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Y^{\mathsf{T}} Y) = (I_d, \ldots, I_d)$$

The constraints are equivalent to:

$$Y_i^{\mathsf{T}} Y_i = I_d, \quad Y_i \in \mathbb{R}^{r \times d}.$$

[5] N. Boumal. "A Riemannian Low-Rank Method for Optimization Over Semidefinite Matrices with Block-Diagonal Constraints". arXiv preprint: arXiv:1506.00575v2. 2015

Exploiting geometric structure[5]

Rank-restricted SDP, NLP form

$$p^*_{\text{SDPLR}} = \min_{Y \in \mathbb{R}^{r \times dn}} \operatorname{tr}(\tilde{Q} Y^{\mathsf{T}} Y)$$

$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Y^{\mathsf{T}} Y) = (I_d, \ldots, I_d)$$

The constraints are equivalent to:

$$Y_i^{\mathsf{T}} Y_i = I_d, \quad Y_i \in \mathbb{R}^{r \times d}.$$

Notice: This is the definition of the *Stiefel manifold*:

$$\operatorname{St}(k, n) \triangleq \left\{ Y \in \mathbb{R}^{n \times k} \mid Y^{\mathsf{T}} Y = I_k \right\}.$$

[5] N. Boumal. "A Riemannian Low-Rank Method for Optimization Over Semidefinite Matrices with Block-Diagonal Constraints". arXiv preprint: arXiv:1506.00575v2. 2015

Exploiting geometric structure[5]

Rank-restricted SDP, NLP form

$$p^*_{\text{SDPLR}} = \min_{Y \in \mathbb{R}^{r \times dn}} \operatorname{tr}(\tilde{Q} Y^{\mathsf{T}} Y)$$
$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Y^{\mathsf{T}} Y) = (I_d, \ldots, I_d)$$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \operatorname{St}(d,r)^n} \operatorname{tr}(\tilde{Q} Y^{\mathsf{T}} Y)$$

The constraints are equivalent to:

$$Y_i^{\mathsf{T}} Y_i = I_d, \quad Y_i \in \mathbb{R}^{r \times d}.$$

Notice: This is the definition of the *Stiefel manifold*:

$$\operatorname{St}(k,n) \triangleq \left\{ Y \in \mathbb{R}^{n \times k} \mid Y^{\mathsf{T}} Y = I_k \right\}.$$

[5] N. Boumal. "A Riemannian Low-Rank Method for Optimization Over Semidefinite Matrices with Block-Diagonal Constraints". arXiv preprint: arXiv:1506.00575v2. 2015

Exploiting geometric structure[5]

Rank-restricted SDP, NLP form

$$p^*_{\text{SDPLR}} = \min_{Y \in \mathbb{R}^{r \times dn}} \operatorname{tr}(\tilde{Q} Y^{\mathsf{T}} Y)$$
$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Y^{\mathsf{T}} Y) = (I_d, \ldots, I_d)$$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \operatorname{St}(d,r)^n} \operatorname{tr}(\tilde{Q} Y^{\mathsf{T}} Y)$$

The constraints are equivalent to:

$$Y_i^{\mathsf{T}} Y_i = I_d, \quad Y_i \in \mathbb{R}^{r \times d}.$$

Notice: This is the definition of the *Stiefel manifold*:

$$\operatorname{St}(k,n) \triangleq \left\{ Y \in \mathbb{R}^{n \times k} \mid Y^{\mathsf{T}} Y = I_k \right\}.$$

Payoff: This is an *unconstrained* optimization problem.

[5] N. Boumal. "A Riemannian Low-Rank Method for Optimization Over Semidefinite Matrices with Block-Diagonal Constraints". arXiv preprint: arXiv:1506.00575v2. 2015

Ensuring global optimality

Simplified pose-graph MLE

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

$$\Downarrow$$

Semidefinite relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

$$\Downarrow$$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \text{St}(d,r)^n} \text{tr}(\tilde{Q} Y^{\text{T}} Y)$$

Ensuring global optimality

Simplified pose-graph MLE

$$p^*_{\mathrm{MLE}} = \min_{R \in \mathrm{SO}(d)^n} \mathrm{tr}(\tilde{Q} R^{\mathsf{T}} R)$$

$$\Downarrow$$

Semidefinite relaxation

$$p^*_{\mathrm{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \mathrm{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \mathrm{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

$$\Downarrow$$

Riemannian rank-restricted NLP

$$p^*_{\mathrm{SDPLR}} = \min_{Y \in \mathrm{St}(d,r)^n} \mathrm{tr}(\tilde{Q} Y^{\mathsf{T}} Y)$$

Question:

What have we actually gained?

Ensuring global optimality

Simplified pose-graph MLE

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

$\Downarrow$

Semidefinite relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

$\Downarrow$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \text{St}(d,r)^n} \text{tr}(\tilde{Q} Y^{\text{T}} Y)$$

Proposition (Boumal, Voroninski, and Bandeira 2016)

If $Y \in \text{St}(d, r)^n$ is a rank deficient 2nd-order critical point for rank-restricted NLP, then:

- *Y is a global minimizer of Riemannian NLP*
- *$Z^* = Y^{\text{T}} Y$ is a solution of the semidefinite relaxation.*

Ensuring global optimality

Simplified pose-graph MLE

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\mathsf{T}} R)$$

$\Downarrow$

Semidefinite relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

$\Downarrow$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \text{St}(d,r)^n} \text{tr}(\tilde{Q} Y^{\mathsf{T}} Y)$$

Proposition (Boumal, Voroninski, and Bandeira 2016)

If $Y \in \text{St}(d, r)^n$ is a rank deficient 2nd-order critical point for rank-restricted NLP, then:

- *Y is a global minimizer of Riemannian NLP*
- *$Z^* = Y^{\mathsf{T}} Y$ is a solution of the semidefinite relaxation.*

$\Rightarrow$ We can use (fast) *local* search to find *globally optimal* solutions!

The Riemannian Staircase

Semidefinite relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \operatorname{tr}(\tilde{Q}Z)$$
$$\text{s.t.} \quad \text{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \text{St}(d,r)^n} \operatorname{tr}(\tilde{Q}Y^{\mathsf{T}}Y)$$

Input: Initial point $Y \in \text{St}(d, r_0)^n$, $r_0 \geq d$.
Output: Symmetric factor Y^* of SDP solution $Z^* = {Y^*}^{\mathsf{T}} Y^*$.

1: **for** $r = r_0, \ldots, dn+1$ **do**
2: $Y^* \leftarrow \text{SecondOrderLocalSearch}(Y)$ (Riemannian NLP).
3: **if** $\text{rank}(Y^*) < r$ **then**
4: **return** Y^*
5: **else**
6: Set $Y \leftarrow \begin{pmatrix} Y^* \\ 0_{1 \times dn} \end{pmatrix}$.
7: **end if**
8: **end for**

The Riemannian Staircase

Semidefinite relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \operatorname{tr}(\tilde{Q}Z)$$

$$\text{s.t.} \quad \operatorname{BlockDiag}_d(Z) = (I_d, \ldots, I_d)$$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \operatorname{St}(d,r)^n} \operatorname{tr}(\tilde{Q}Y^{\mathsf{T}}Y)$$

Input: Initial point $Y \in \operatorname{St}(d, r_0)^n$, $r_0 \geq d$.
Output: Symmetric factor Y^* of SDP solution $Z^* = {Y^*}^{\mathsf{T}} Y^*$.

1: **for** $r = r_0, \ldots, dn+1$ **do**
2: $Y^* \leftarrow$ SecondOrderLocalSearch(Y) (Riemannian NLP).
3: **if** $\operatorname{rank}(Y^*) < r$ **then**
4: **return** Y^*
5: **else**
6: Set $Y \leftarrow \begin{pmatrix} Y^* \\ 0_{1 \times dn} \end{pmatrix}$.
7: **end if**
8: **end for**

SE-Sync

Simplified pose-graph MLE

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

$\Downarrow$

SDP relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{Diag}_d(Z) = (I_d, \ldots, I_d)$$

$\Downarrow$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \text{St}(d,r)^n} \text{tr}(\tilde{Q} Y^{\text{T}} Y)$$

The SE-Sync algorithm:

1. Find low-rank factor Y^* using fast (2nd-order) NLP method in Riemannian Staircase.

SE-Sync

Simplified pose-graph MLE

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

$\Downarrow$

SDP relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{Diag}_d(Z) = (I_d, \ldots, I_d)$$

$\Downarrow$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \text{St}(d,r)^n} \text{tr}(\tilde{Q} Y^{\text{T}} Y)$$

The SE-Sync algorithm:

1. Find low-rank factor Y^* using fast (2nd-order) NLP method in Riemannian Staircase.
2. Compute SDP lower bound: $p^*_{\text{SDP}} = \text{tr}(\tilde{Q} {Y^*}^{\text{T}} Y^*)$.

SE-Sync

Simplified pose-graph MLE

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

$\Downarrow$

SDP relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{Diag}_d(Z) = (I_d, \ldots, I_d)$$

$\Downarrow$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \text{St}(d,r)^n} \text{tr}(\tilde{Q} Y^{\text{T}} Y)$$

The SE-Sync algorithm:

1. Find low-rank factor Y^* using fast (2nd-order) NLP method in Riemannian Staircase.
2. Compute SDP lower bound: $p^*_{\text{SDP}} = \text{tr}(\tilde{Q} {Y^*}^{\text{T}} Y^*)$.
3. Round $Y^* \to \hat{R} \in \text{SO}(d)^n$ using truncated SVD.

SE-Sync

Simplified pose-graph MLE

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

$\Downarrow$

SDP relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{Diag}_d(Z) = (I_d, \ldots, I_d)$$

$\Downarrow$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \text{St}(d,r)^n} \text{tr}(\tilde{Q} Y^{\text{T}} Y)$$

The SE-Sync algorithm:

1. Find low-rank factor Y^* using fast (2nd-order) NLP method in Riemannian Staircase.
2. Compute SDP lower bound: $p^*_{\text{SDP}} = \text{tr}(\tilde{Q} {Y^*}^{\text{T}} Y^*)$.
3. Round $Y^* \to \hat{R} \in \text{SO}(d)^n$ using truncated SVD.
4. Return $\{\hat{R}, p^*_{\text{SDP}}\}$.

SE-Sync

Simplified pose-graph MLE

$$p^*_{\text{MLE}} = \min_{R \in \text{SO}(d)^n} \text{tr}(\tilde{Q} R^{\text{T}} R)$$

$\Downarrow$

SDP relaxation

$$p^*_{\text{SDP}} = \min_{Z \in \mathbb{S}^{dn}_+} \text{tr}(\tilde{Q} Z)$$

$$\text{s.t.} \quad \text{Diag}_d(Z) = (I_d, \ldots, I_d)$$

$\Downarrow$

Riemannian rank-restricted NLP

$$p^*_{\text{SDPLR}} = \min_{Y \in \text{St}(d,r)^n} \text{tr}(\tilde{Q} Y^{\text{T}} Y)$$

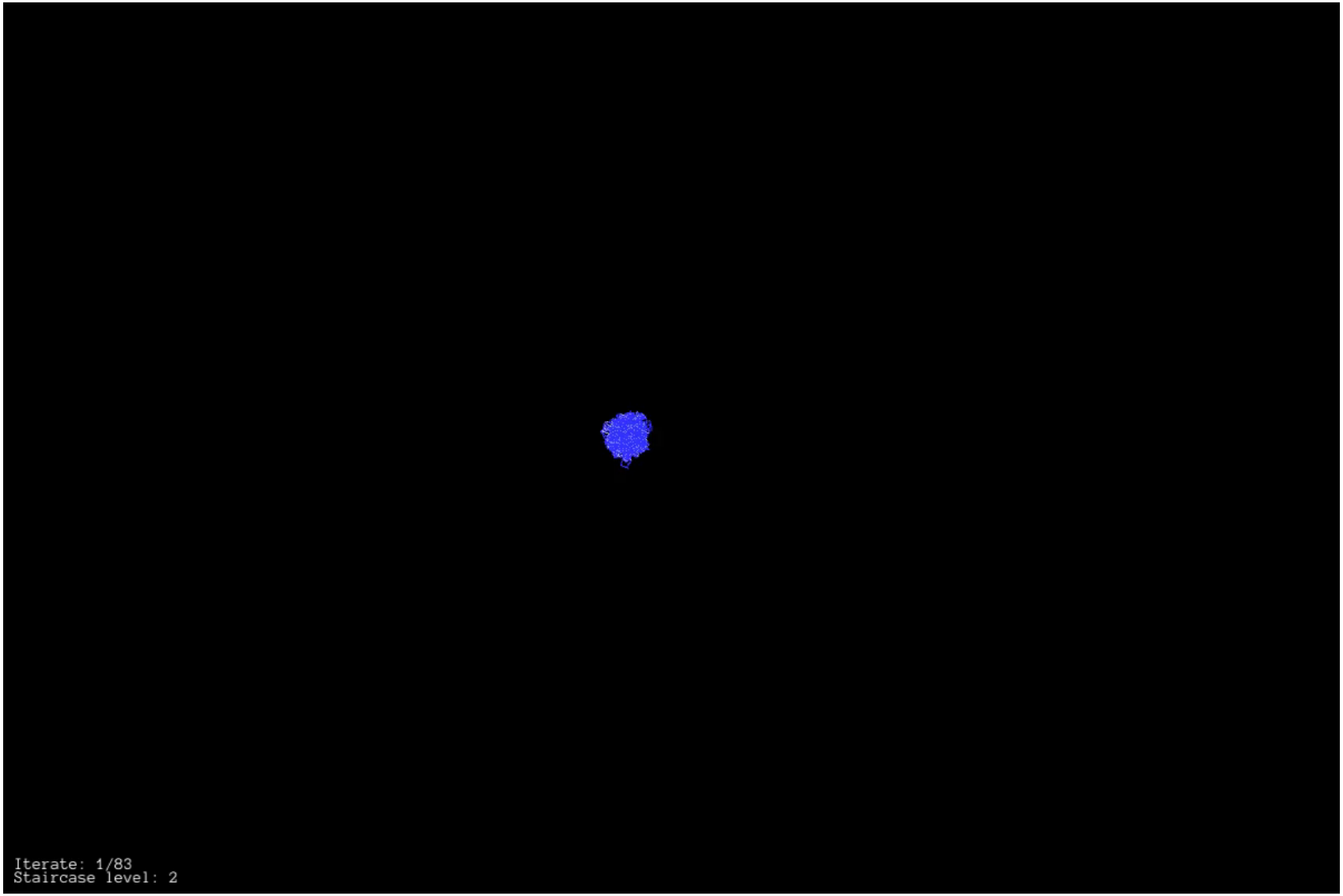
The SE-Sync algorithm:

1. Find low-rank factor Y^* using fast (2nd-order) NLP method in Riemannian Staircase.
2. Compute SDP lower bound: $p^*_{\text{SDP}} = \text{tr}(\tilde{Q} {Y^*}^{\text{T}} Y^*)$.
3. Round $Y^* \to \hat{R} \in \text{SO}(d)^n$ using truncated SVD.
4. Return $\{\hat{R}, p^*_{\text{SDP}}\}$.

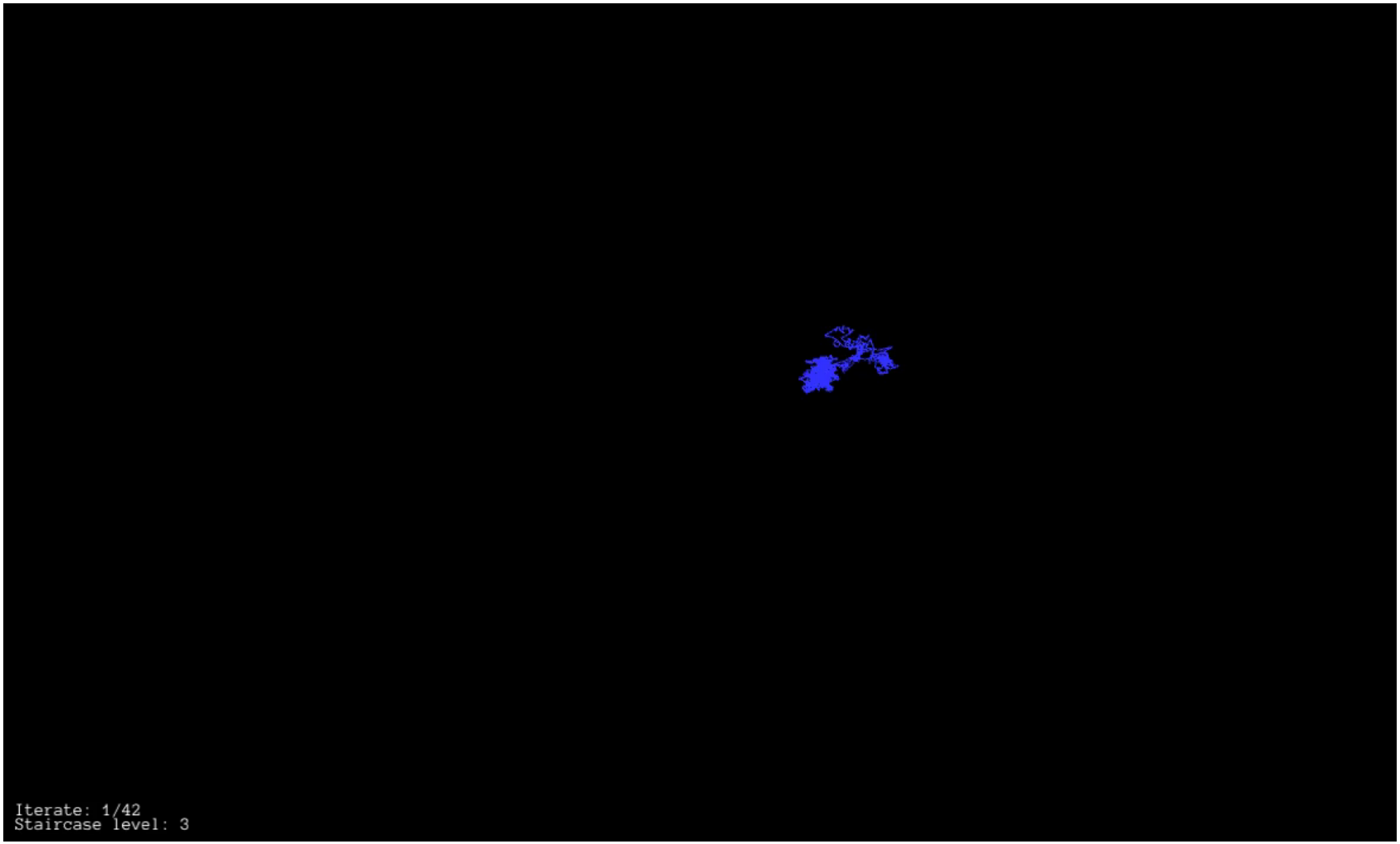
Payoff: If exactness holds:

- $\hat{R}$ is *globally optimal*
- $\text{tr}(\tilde{Q} \hat{R}^{\text{T}} \hat{R}) = p^*_{\text{SDP}}$ *certifies* it

Visualization I: Grid world

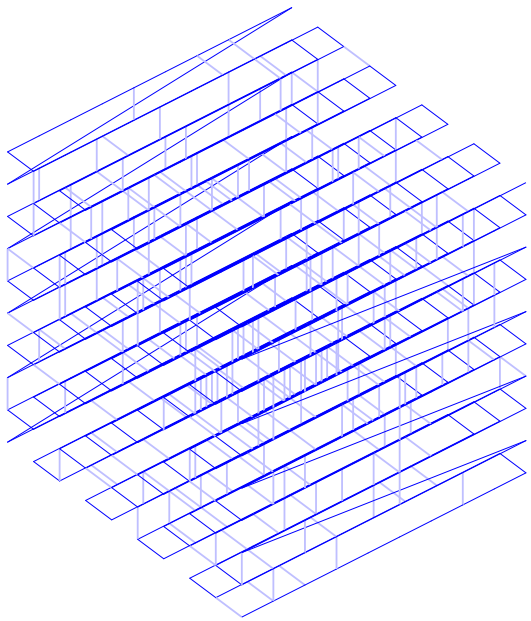


Visualization II: Parking garage



Experimental results I: Grid world simulations

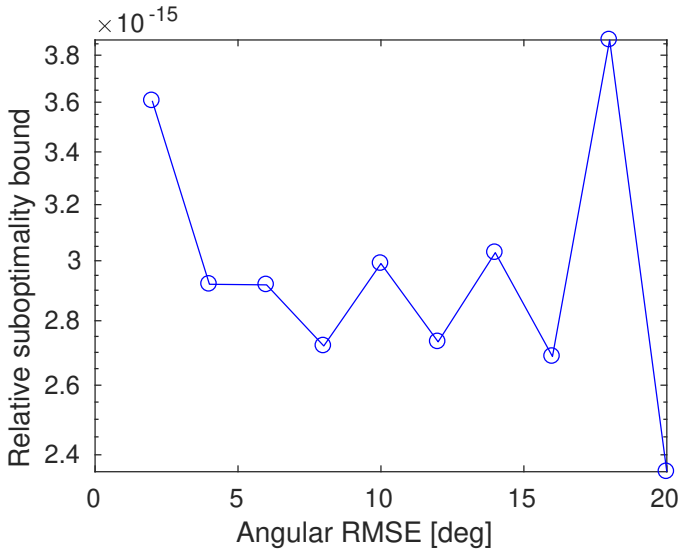
Question: How do noise and problem size affect performance?
Test: Simulate random grid-world, varying κ, τ, and n:

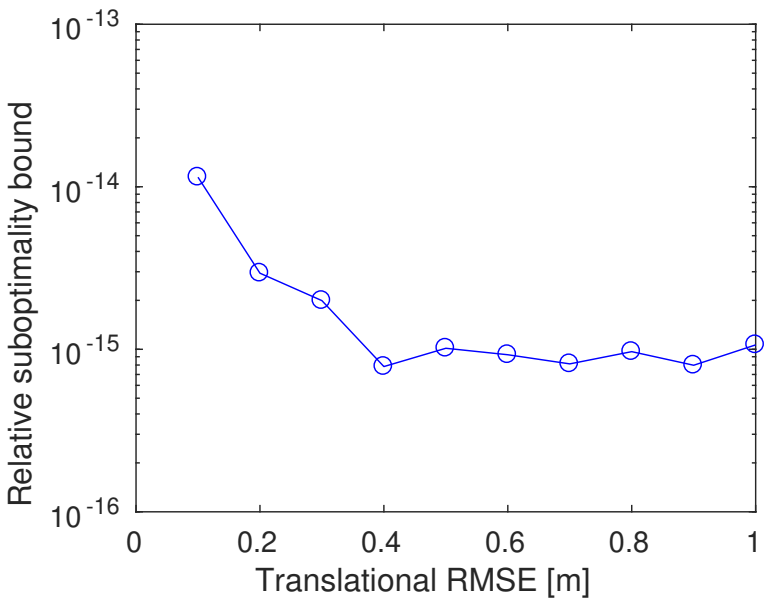


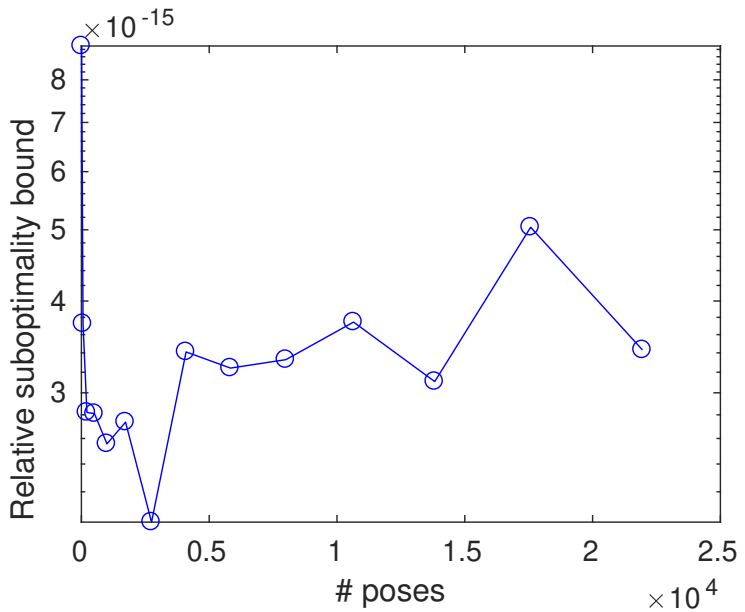
$$\tilde{t}_{ij} = R_i^T(t_j - t_i) + \delta t_{ij}, \qquad \delta t_{ij} \sim \mathcal{N}(0, \tau^{-1} I_d)$$
$$\tilde{R}_{ij} = R_i^T R_j \cdot \delta R_{ij}, \qquad \delta R_{ij} \sim \text{Langevin}(I_d, \kappa)$$

Baseline: Gauss-Newton using *chordal initialization*

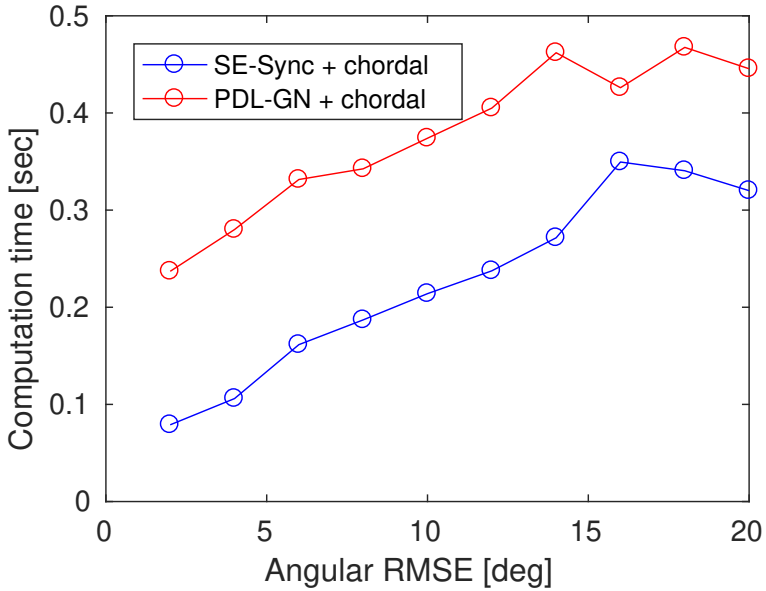
Experimental results I: Grid world simulations

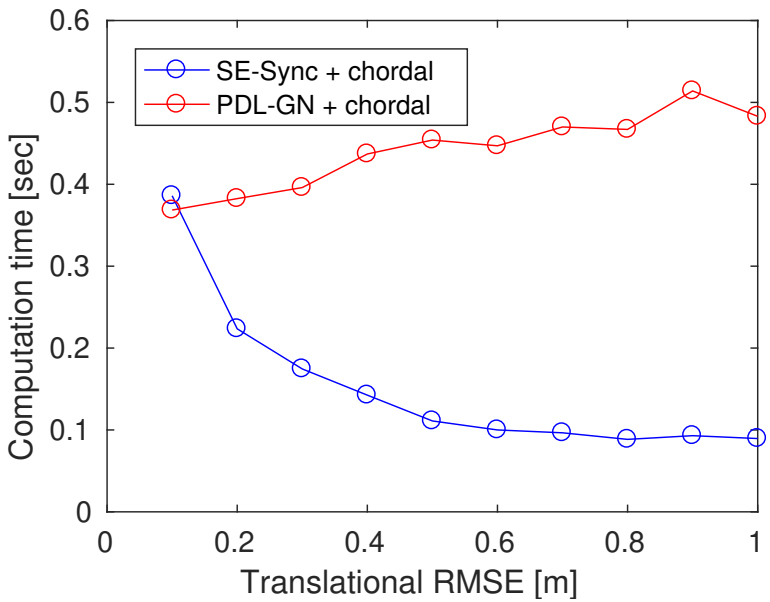


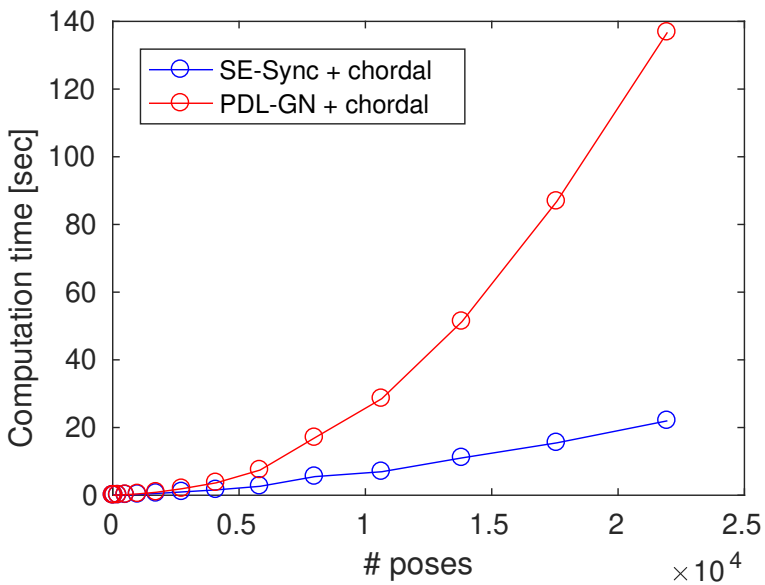




Experimental results I: Grid world simulations

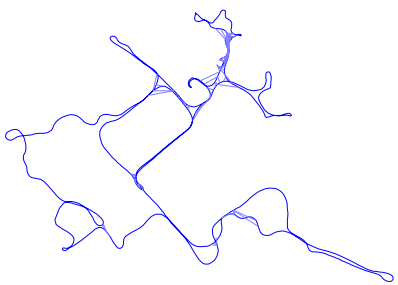




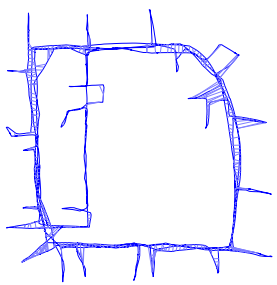


Experimental results II: Large-scale SLAM benchmarks

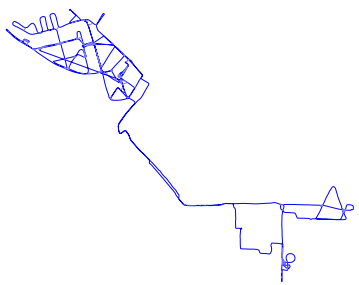
	# Poses	PDL-GN (GTSAM)		SE-Sync		
		Objective value	Time [s]	Objective value	Time [s]	Rel. suboptimality
csail	1045	3.170×10^1	0.029	3.170×10^1	0.010	7.844×10^{-16}
intel	1728	5.235×10^1	0.120	5.235×10^1	0.071	1.357×10^{-16}
ais2klinik	15115	1.885×10^2	12.472	1.885×10^2	1.981	2.412×10^{-15}
garage	1661	1.263×10^0	0.415	1.263×10^0	0.468	1.618×10^{-14}
cubicle	5750	7.171×10^2	2.456	7.171×10^2	0.754	2.061×10^{-15}
rim	10195	5.461×10^3	6.803	5.461×10^3	2.256	5.663×10^{-15}



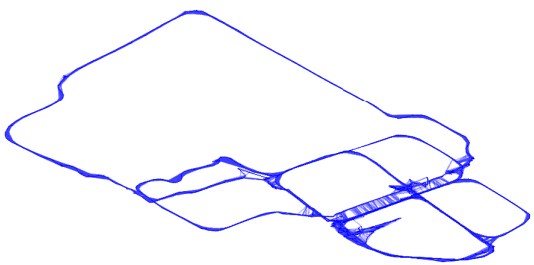
csail



intel



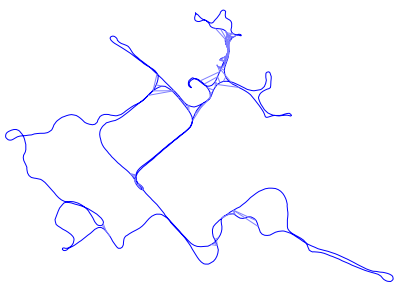
ais2klinik



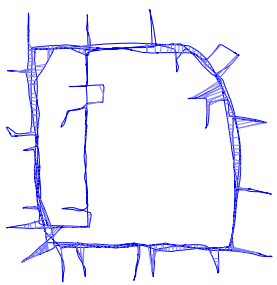
rim

Experimental results II: Large-scale SLAM benchmarks

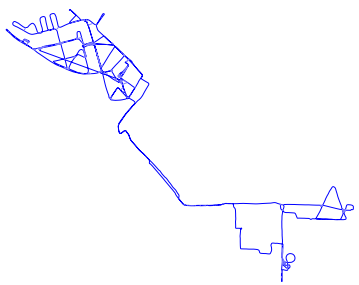
	# Poses	PDL-GN (GTSAM)		SE-Sync		
		Objective value	Time [s]	Objective value	Time [s]	Rel. suboptimality
csail	1045	3.170×10^1	0.029	3.170×10^1	0.010	7.844×10^{-16}
intel	1728	5.235×10^1	0.120	5.235×10^1	0.071	1.357×10^{-16}
ais2klinik	15115	1.885×10^2	12.472	1.885×10^2	1.981	2.412×10^{-15}
garage	1661	1.263×10^0	0.415	1.263×10^0	0.468	1.618×10^{-14}
cubicle	5750	7.171×10^2	2.456	7.171×10^2	0.754	2.061×10^{-15}
rim	10195	5.461×10^3	6.803	5.461×10^3	2.256	5.663×10^{-15}



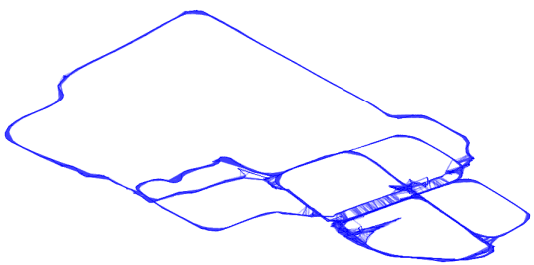
csail



intel



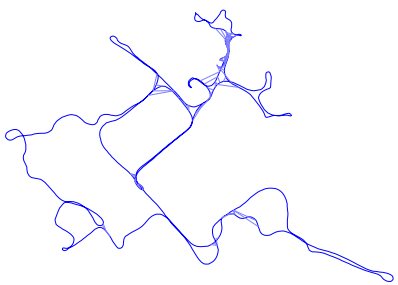
ais2klinik



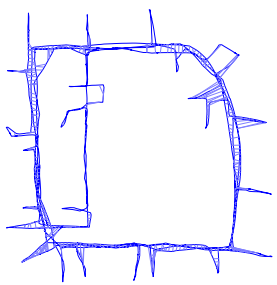
rim

Experimental results II: Large-scale SLAM benchmarks

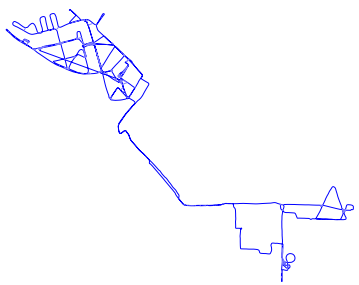
	# Poses	PDL-GN (GTSAM)		SE-Sync		
		Objective value	Time [s]	Objective value	Time [s]	Rel. suboptimality
csail	1045	3.170×10^1	0.029	3.170×10^1	0.010	7.844×10^{-16}
intel	1728	5.235×10^1	0.120	5.235×10^1	0.071	1.357×10^{-16}
ais2klinik	15115	1.885×10^2	12.472	1.885×10^2	1.981	2.412×10^{-15}
garage	1661	1.263×10^0	0.415	1.263×10^0	0.468	1.618×10^{-14}
cubicle	5750	7.171×10^2	2.456	7.171×10^2	0.754	2.061×10^{-15}
rim	10195	5.461×10^3	6.803	5.461×10^3	2.256	5.663×10^{-15}



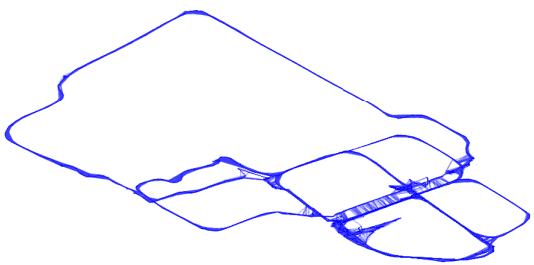
csail



intel



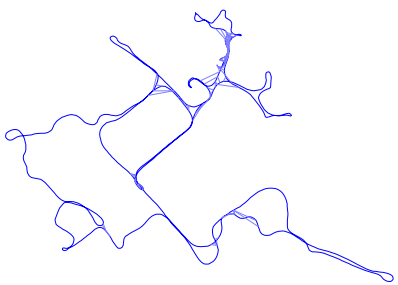
ais2klinik



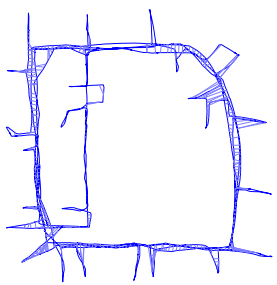
rim

Experimental results II: Large-scale SLAM benchmarks

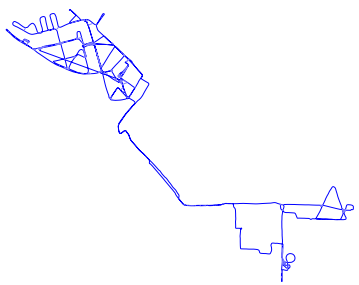
	# Poses	PDL-GN (GTSAM)		SE-Sync		
		Objective value	Time [s]	Objective value	Time [s]	Rel. suboptimality
csail	1045	3.170×10^1	0.029	3.170×10^1	0.010	7.844×10^{-16}
intel	1728	5.235×10^1	0.120	5.235×10^1	0.071	1.357×10^{-16}
ais2klinik	15115	1.885×10^2	12.472	1.885×10^2	1.981	2.412×10^{-15}
garage	1661	1.263×10^0	0.415	1.263×10^0	0.468	1.618×10^{-14}
cubicle	5750	7.171×10^2	2.456	7.171×10^2	0.754	2.061×10^{-15}
rim	10195	5.461×10^3	6.803	5.461×10^3	2.256	5.663×10^{-15}



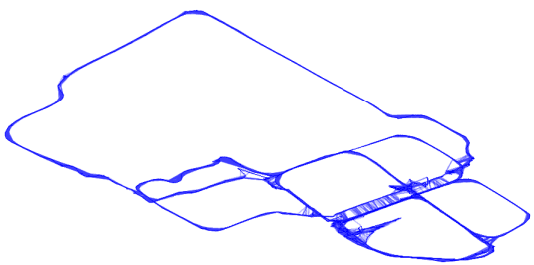
csail



intel



ais2klinik



rim

Contributions

SE-Sync: A *certifiably correct* algorithm for pose-graph SLAM[6]

- Recovers *globally optimal* estimates for moderate noise (up to 10x typical levels)
- Significantly faster than prior state-of-the-art techniques

[6] D.M. Rosen, L. Carlone, et al. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *Intl. J. of Robotics Research* 38.2–3 (Mar. 2019), pp. 95–125

[7] D.M. Rosen. "Towards Provably Robust Machine Perception". Presented at Robotics: Science and Systems (RSS) in the workshop "RSS Pioneers". June 2019

Contributions

SE-Sync: A *certifiably correct* algorithm for pose-graph SLAM[6]

- Recovers *globally optimal* estimates for moderate noise (up to 10x typical levels)
- Significantly faster than prior state-of-the-art techniques

$\Rightarrow$ First practical alg. *provably* able to recover correct solutions

[6] D.M. Rosen, L. Carlone, et al. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *Intl. J. of Robotics Research* 38.2–3 (Mar. 2019), pp. 95–125

[7] D.M. Rosen. "Towards Provably Robust Machine Perception". Presented at Robotics: Science and Systems (RSS) in the workshop "RSS Pioneers". June 2019

Contributions

SE-Sync: A *certifiably correct* algorithm for pose-graph SLAM[6]

- Recovers *globally optimal* estimates for moderate noise (up to 10x typical levels)
- Significantly faster than prior state-of-the-art techniques

$\Rightarrow$ First practical alg. *provably* able to recover correct solutions

Certifiable estimation:

- SE-Sync's SDP relaxation generalizes via *moment relaxation*[7]

[6] D.M. Rosen, L. Carlone, et al. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *Intl. J. of Robotics Research* 38.2–3 (Mar. 2019), pp. 95–125

[7] D.M. Rosen. "Towards Provably Robust Machine Perception". Presented at Robotics: Science and Systems (RSS) in the workshop "RSS Pioneers". June 2019

Contributions

SE-Sync: A *certifiably correct* algorithm for pose-graph SLAM[6]

- Recovers *globally optimal* estimates for moderate noise (up to 10x typical levels)
- Significantly faster than prior state-of-the-art techniques

$\Rightarrow$ First practical alg. *provably* able to recover correct solutions

Certifiable estimation:

- SE-Sync's SDP relaxation generalizes via *moment relaxation*[7]
- Demonstrates feasibility of *large-scale SDP optimization*

[6] D.M. Rosen, L. Carlone, et al. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *Intl. J. of Robotics Research* 38.2–3 (Mar. 2019), pp. 95–125

[7] D.M. Rosen. "Towards Provably Robust Machine Perception". Presented at Robotics: Science and Systems (RSS) in the workshop "RSS Pioneers". June 2019

Contributions

SE-Sync: A *certifiably correct* algorithm for pose-graph SLAM[6]

- Recovers *globally optimal* estimates for moderate noise (up to 10x typical levels)
- Significantly faster than prior state-of-the-art techniques

$\Rightarrow$ First practical alg. *provably* able to recover correct solutions

Certifiable estimation:

- SE-Sync's SDP relaxation generalizes via *moment relaxation*[7]
- Demonstrates feasibility of *large-scale SDP optimization*

$\Rightarrow$ Proof of concept for *general approach* to certifiable estimation

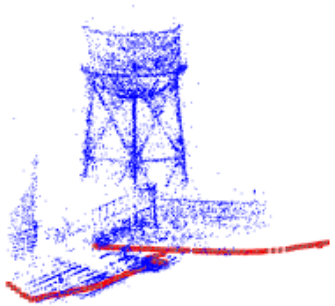
---

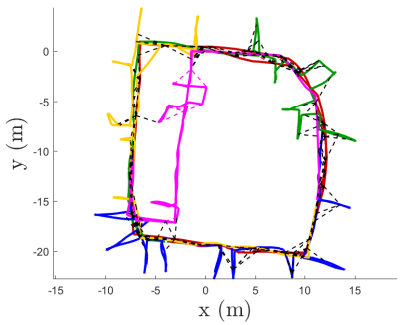
[6] D.M. Rosen, L. Carlone, et al. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *Intl. J. of Robotics Research* 38.2–3 (Mar. 2019), pp. 95–125

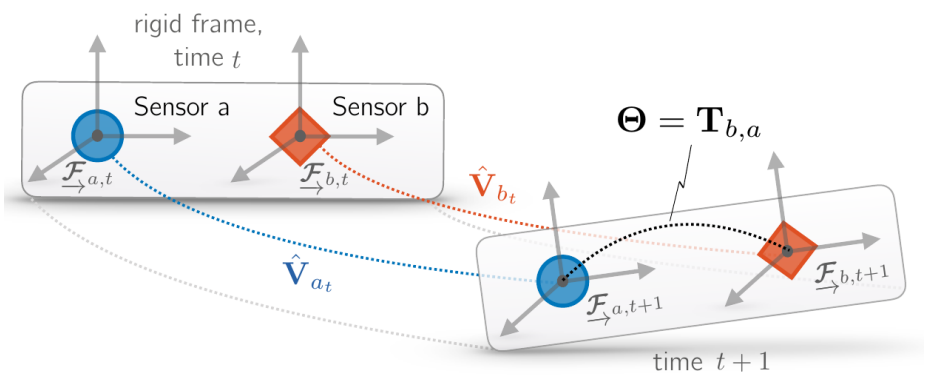
[7] D.M. Rosen. "Towards Provably Robust Machine Perception". Presented at Robotics: Science and Systems (RSS) in the workshop "RSS Pioneers". June 2019

Applications of certifiable estimation

- Rotation averaging[1]
- Two-view registration[2]
- Sensor calibration[3]
- Image segmentation[4]
- Distributed SLAM[5]







[1] Dellaert et al. 2020; Eriksson et al. 2018

[2] Briales, Kneip, and Gonzalez-Jimenez 2018

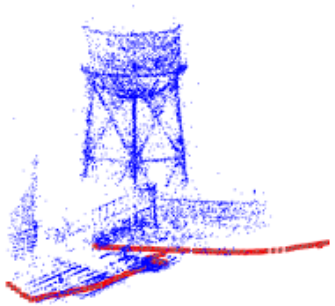
[3] Giamou et al. 2019

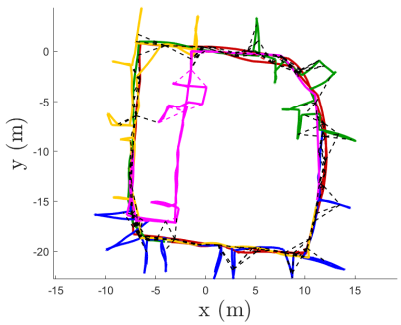
[4] Hu and Carlone 2019

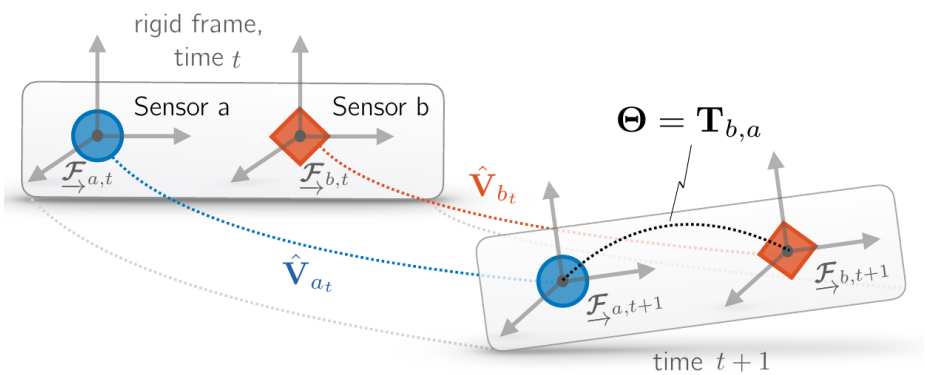
[5] Fan and Murphey 2019; Tian et al. 2020

Applications of certifiable estimation

- Rotation averaging[1]
- Two-view registration[2]
- Sensor calibration[3]
- Image segmentation[4]
- Distributed SLAM[5]







[1] Dellaert et al. 2020; Eriksson et al. 2018

[2] Briales, Kneip, and Gonzalez-Jimenez 2018

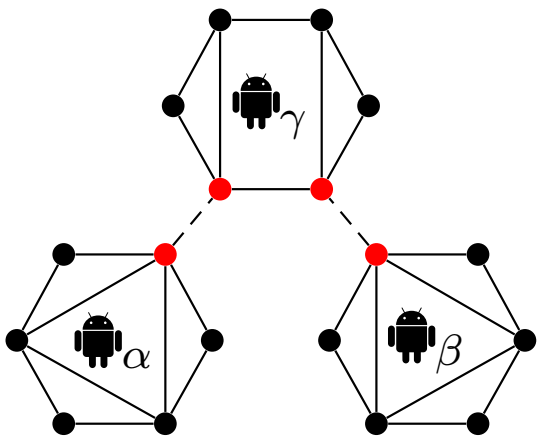
[3] Giamou et al. 2019

[4] Hu and Carlone 2019

[5] Fan and Murphey 2019; Tian et al. 2020

Distributed pose-graph SLAM

DC2-PGO: First *distributed* certifiably correct algorithm for pose-graph SLAM and rotation averaging.[8]

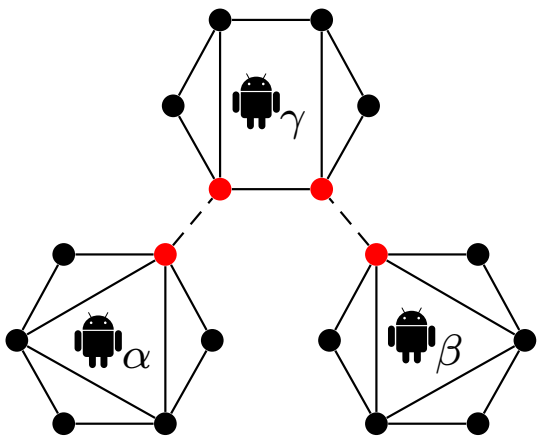


[8] Y. Tian et al. "Distributed Certifiably Correct Pose-Graph Optimization". In: *IEEE Trans. on Robotics* (2020). (under review).

Distributed pose-graph SLAM

DC2-PGO: First *distributed* certifiably correct algorithm for pose-graph SLAM and rotation averaging.[8]

- Solves a *sparse* version of SE-Sync SDP relaxation.

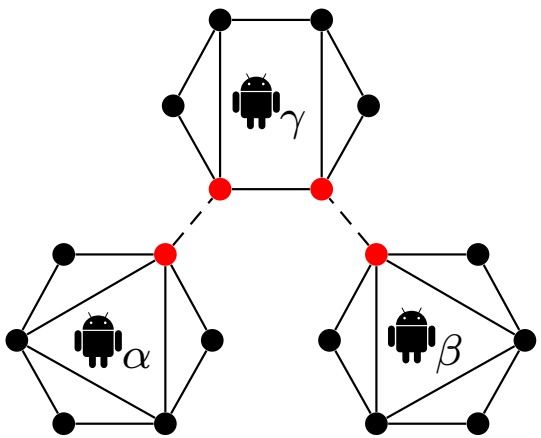


[8] Y. Tian et al. "Distributed Certifiably Correct Pose-Graph Optimization". In: *IEEE Trans. on Robotics* (2020). (under review).

Distributed pose-graph SLAM

DC2-PGO: First *distributed* certifiably correct algorithm for pose-graph SLAM and rotation averaging.[8]

- Solves a *sparse* version of SE-Sync SDP relaxation.
- *Distributed* Riemannian Staircase

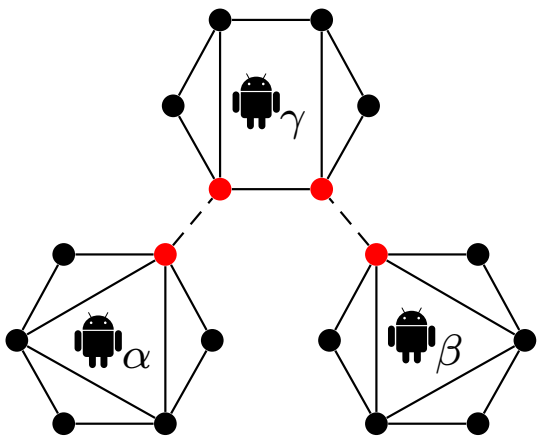


[8] Y. Tian et al. "Distributed Certifiably Correct Pose-Graph Optimization". In: *IEEE Trans. on Robotics* (2020). (under review).

Distributed pose-graph SLAM

DC2-PGO: First *distributed* certifiably correct algorithm for pose-graph SLAM and rotation averaging.[8]

- Solves a *sparse* version of SE-Sync SDP relaxation.
- *Distributed* Riemannian Staircase
  - Distributed (1st-order) local opt: *Riemannian block coordinate descent*

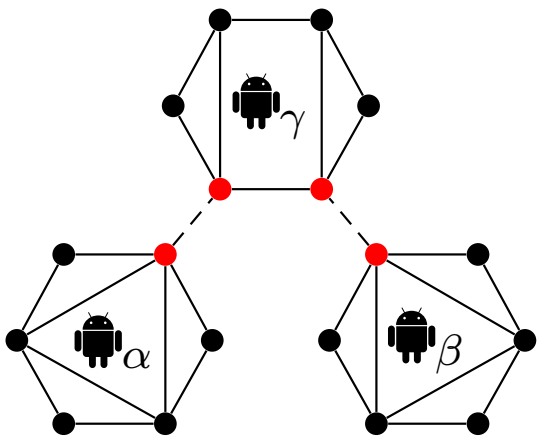


[8] Y. Tian et al. "Distributed Certifiably Correct Pose-Graph Optimization". In: *IEEE Trans. on Robotics* (2020). (under review).

Distributed pose-graph SLAM

DC2-PGO: First *distributed* certifiably correct algorithm for pose-graph SLAM and rotation averaging.[8]

- Solves a *sparse* version of SE-Sync SDP relaxation.
- *Distributed* Riemannian Staircase
  - Distributed (1st-order) local opt: *Riemannian block coordinate descent*
  - Distributed solution certification: *accelerated power method*

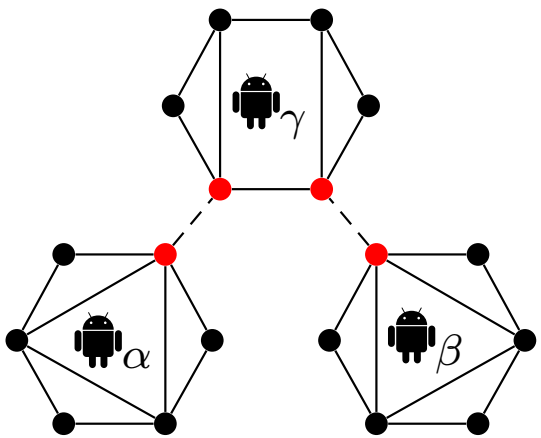


[8] Y. Tian et al. "Distributed Certifiably Correct Pose-Graph Optimization". In: *IEEE Trans. on Robotics* (2020). (under review).

Distributed pose-graph SLAM

DC2-PGO: First *distributed* certifiably correct algorithm for pose-graph SLAM and rotation averaging.[8]

- Solves a *sparse* version of SE-Sync SDP relaxation.
- *Distributed* Riemannian Staircase
  - Distributed (1st-order) local opt: *Riemannian block coordinate descent*
  - Distributed solution certification: *accelerated power method*



$\Rightarrow$ *Fully decentralized* and *privacy preserving*

[8] Y. Tian et al. "Distributed Certifiably Correct Pose-Graph Optimization". In: *IEEE Trans. on Robotics* (2020). (under review).

Distributed pose-graph SLAM

Distributed Certifiably Correct Pose-Graph Optimization

Yulun Tian, Kasra Khosoussi, David M. Rosen, Jonathan P. How

[8] Y. Tian et al. "Distributed Certifiably Correct Pose-Graph Optimization". In: *IEEE Trans. on Robotics* (2020). (under review).

Application: Distributed metric-semantic SLAM[1]

Kimera-Multi: a System for Distributed Multi-Robot Metric-Semantic Simultaneous Localization and Mapping

Yun Chang, Yulun Tian, Jonathan P. How, Luca Carlone

Massachusetts Institute of Technology

[1] Y. Chang et al. "Kimera-Multi: a System for Distributed Multi-Robot Metric-Semantic Simultaneous Localization and Mapping". In: *arXiv preprint arXiv:2011.04087* (2020)

Contributions

SE-Sync: A *certifiably correct* algorithm for pose-graph SLAM[9]

- Recovers *globally optimal* estimates for moderate noise (up to 10x typical levels)
- Significantly faster than prior state-of-the-art techniques

$\Rightarrow$ First practical alg. *provably* able to recover correct solutions

[9] D.M. Rosen, L. Carlone, et al. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *Intl. J. of Robotics Research* 38.2–3 (Mar. 2019), pp. 95–125

[10] D.M. Rosen. "Towards Provably Robust Machine Perception". Presented at Robotics: Science and Systems (RSS) in the workshop "RSS Pioneers". June 2019

Contributions

SE-Sync: A *certifiably correct* algorithm for pose-graph SLAM[9]

- Recovers *globally optimal* estimates for moderate noise (up to 10x typical levels)
- Significantly faster than prior state-of-the-art techniques

$\Rightarrow$ First practical alg. *provably* able to recover correct solutions

Code: `https://github.com/david-m-rosen/SE-Sync`

[9] D.M. Rosen, L. Carlone, et al. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *Intl. J. of Robotics Research* 38.2–3 (Mar. 2019), pp. 95–125

[10] D.M. Rosen. "Towards Provably Robust Machine Perception". Presented at Robotics: Science and Systems (RSS) in the workshop "RSS Pioneers". June 2019

Contributions

SE-Sync: A *certifiably correct* algorithm for pose-graph SLAM[9]

- Recovers *globally optimal* estimates for moderate noise (up to 10x typical levels)
- Significantly faster than prior state-of-the-art techniques

$\Rightarrow$ First practical alg. *provably* able to recover correct solutions

Code: `https://github.com/david-m-rosen/SE-Sync`

Future directions: General certifiable estimation[10]

- General and robust extensions via moment relaxation
- Scalable semidefinite optimizers

---

[9] D.M. Rosen, L. Carlone, et al. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *Intl. J. of Robotics Research* 38.2–3 (Mar. 2019), pp. 95–125

[10] D.M. Rosen. "Towards Provably Robust Machine Perception". Presented at Robotics: Science and Systems (RSS) in the workshop "RSS Pioneers". June 2019