Model-Free Safe Exploration and Optimization

Master's Thesis Defense

Yang Wang
Supervisors: Prof. Dr.-Ing. Daniel Görges[1],
Prof. Dr.-Ing. Giancarlo Ferrari Trecate[2], Doctoral Assistant Baiwei Guo[2]

[1]Electrical Engineering and Information Technology
Technical University of Kaiserslautern

[2]Automatic Control Laboratory
Swiss Federal Institute of Technology in Lausanne

April 29, 2022

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN EPFL

Background

Conventionally,

- Solving optimization problems requires models of objective and constraint functions;

Background

Conventionally,

- Solving optimization problems requires models of objective and constraint functions;

For increasingly complex problems:

- The objective and/or constraints may not be explicitly described in mathematical expressions;

Background

Conventionally,

- Solving optimization problems requires models of objective and constraint functions;

For increasingly complex problems:

- The objective and/or constraints may not be explicitly described in mathematical expressions;
- Only the function values can be accessed.

Background

Conventionally,

- Solving optimization problems requires models of objective and constraint functions;

For increasingly complex problems:

- The objective and/or constraints may not be explicitly described in mathematical expressions;
- Only the function values can be accessed.

How can we solve the optimization problems in this setting?

Model-free optimization methods

Model-free optimization methods:

- Do not require the expressions of the objective and/or constraint functions;

Model-free optimization methods

Model-free optimization methods:

- Do not require the expressions of the objective and/or constraint functions;
- Treat the unknown system as a black-box;

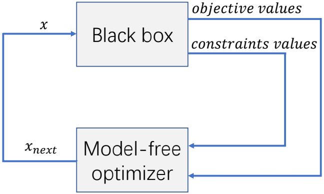


Figure 1: Model-free optimization

Model-free optimization methods

Model-free optimization methods:

- Do not require the expressions of the objective and/or constraint functions;
- Treat the unknown system as a black-box;
- Minimize the objective function using the outputs of the system.

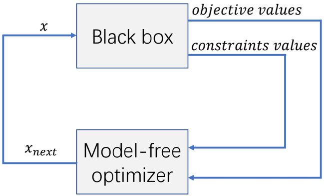


Figure 1: Model-free optimization

Safety constraints

Safety: At every function evaluations, all constraints are satisfied.

Safety constraints

Safety: At every function evaluations, all constraints are satisfied.

Violating safety constraints:

- Operations of systems may not be able to perform;

Safety constraints

Safety: At every function evaluations, all constraints are satisfied.

Violating safety constraints:

- Operations of systems may not be able to perform;
- May lead to severe consequences, e.g. device damages, human injuries.

1. Introduction
   - Model-Free Optimization Problem
   - Safety Constraints
   - Problem Formulation
2. Literature Review
3. Methods
4. Results
5. Conclusion

Model-free safe optimization problem

Problem: Model-Free Safe Optimization Problem

$$\underset{\boldsymbol{x}\in\mathbb{R}^d}{\text{minimize}} \quad f^0(\boldsymbol{x}) \tag{1}$$

$$\text{subject to} \quad f^i(\boldsymbol{x}) \leq 0,\ i = 1,\ldots,m \tag{2}$$

where $\boldsymbol{x} \in \mathbb{R}^d$ *is the decision variable,* $f^0(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ *and* $f^i(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ *are the objective and the constraint functions that are not explicitly known.* d *is the number of dimensions and* m *is the number of constraints.* $\mathcal{D} := \left\{\boldsymbol{x} \in \mathbb{R}^d : f^i(\boldsymbol{x}) \leq 0,\ \forall i = 1,\ldots,m\right\}$ *is the feasible region.*

1 Introduction

2 Literature Review
- Classification of Approaches
- Model-Based Methods
- Comparison of Different Algorithms

3 Methods

4 Results

5 Conclusion

Classification of approaches

The majority of methods can be classified into two types:

- Deterministic methods;

Classification of approaches

The majority of methods can be classified into two types:

- Deterministic methods;
- Randomized methods.

Classification of approaches

The majority of methods can be classified into two types:

- Deterministic methods;
  - Direct-search methods;
- Randomized methods.

Classification of approaches

The majority of methods can be classified into two types:

- Deterministic methods;
  - Direct-search methods;
  - Model-based methods;
- Randomized methods.

Classification of approaches

The majority of methods can be classified into two types:

- Deterministic methods;
  - Direct-search methods;
  - Model-based methods;
- Randomized methods.
  - Pure random search methods;

Classification of approaches

The majority of methods can be classified into two types:

- Deterministic methods;
  - Direct-search methods;
  - Model-based methods;
- Randomized methods.
  - Pure random search methods;
  - Randomized direct-search methods.

Classification of approaches

The majority of methods can be classified into two types:

- Deterministic methods;
  - Direct-search methods;
  - Model-based methods;
- Randomized methods.
  - Pure random search methods;
  - Randomized direct-search methods.

Model-based method

Model-based method:
Use estimated models to determine the next decision point.

Model-based method

Model-based method:
Use estimated models to determine the next decision point.

Depending on the type of estimated models, the next decision points are computed differently.

Model-based method

Model-based method:
Use estimated models to determine the next decision point.

Depending on the type of estimated models, the next decision points are computed differently.

The common models are:

- Polynomial models;

Model-based method

Model-based method:
Use estimated models to determine the next decision point.

Depending on the type of estimated models, the next decision points are computed differently.

The common models are:

- Polynomial models;
- Logarithmic models;

Model-based method

Model-based method:
Use estimated models to determine the next decision point.

Depending on the type of estimated models, the next decision points are computed differently.

The common models are:

- Polynomial models;
- Logarithmic models;
- Probabilistic models.

Model-based method

Model-based method:
Use estimated models to determine the next decision point.

Depending on the type of estimated models, the next decision points are computed differently.

The common models are:

- Polynomial models;
  - Derivative-Free Optimization with Trust-Region (DFO-TR) [8];
- Logarithmic models;
- Probabilistic models.

Model-based method

Model-based method:
Use estimated models to determine the next decision point.

Depending on the type of estimated models, the next decision points are computed differently.

The common models are:

- Polynomial models;
  - Derivative-Free Optimization with Trust-Region (DFO-TR) [8];
- Logarithmic models;
  - Safe Logarithmic Barrier Method (s0-LBM) [7];
- Probabilistic models.

Model-based method

Model-based method:
Use estimated models to determine the next decision point.

Depending on the type of estimated models, the next decision points are computed differently.

The common models are:

- Polynomial models;
  - Derivative-Free Optimization with Trust-Region (DFO-TR) [8];
- Logarithmic models;
  - Safe Logarithmic Barrier Method (s0-LBM) [7];
- Probabilistic models.
  - Safe Bayesian Optimization (SafeOpt) [6].

1 Introduction

2 Literature Review
- Classification of Approaches
- Model-Based Methods
- Comparison of Different Algorithms

3 Methods

4 Results

5 Conclusion

Advantages and disadvantages

DFO-TR

- Superlinear convergence rate;
- Higher computation cost;

Advantages and disadvantages

DFO-TR

- Superlinear convergence rate;
- Higher computation cost;

s0-LBM

- Low computation cost;
- Slow convergence near boundaries;

Advantages and disadvantages

DFO-TR

- Superlinear convergence rate;
- Higher computation cost;

s0-LBM

- Low computation cost;
- Slow convergence near boundaries;

SafeOpt

- Global convergence;
- Limited scalability.

Advantages and disadvantages

DFO-TR

- Superlinear convergence rate;
- Higher computation cost;

s0-LBM

- Low computation cost;
- Slow convergence near boundaries;

SafeOpt

- Global convergence;
- Limited scalability.

We proposed a scalable, with low computation cost optimization algorithm.

Main Idea of The Algorithm

Safe line-search optimization algorithm

- Model-free optimization;
- Line-search method;
- Consider safety constraints;
- Using noisy function measurements.

Main Idea of The Algorithm

Safe line-search optimization algorithm

- Model-free optimization;
- Line-search method;
- Consider safety constraints;
- Using noisy function measurements.

Main Idea of The Algorithm

Safe line-search optimization algorithm

- Model-free optimization;
- Line-search method;
- Consider safety constraints;
- Using noisy function measurements.

Main Idea of The Algorithm

Safe line-search optimization algorithm

- Model-free optimization;
- Line-search method;
- Consider safety constraints;
- Using noisy function measurements.

Main Idea of The Algorithm

Safe line-search optimization algorithm

- Model-free optimization;
- Line-search method;
- Consider safety constraints;
- Using noisy function measurements.

Algorithms in two situations:

- Safe Line-Search Optimization for Exact Measurements (e-SLS);

Main Idea of The Algorithm

Safe line-search optimization algorithm

- Model-free optimization;
- Line-search method;
- Consider safety constraints;
- Using noisy function measurements.

Algorithms in two situations:

- Safe Line-Search Optimization for Exact Measurements (e-SLS);
- Safe Line-Search Optimization for Noisy Measurements (n-SLS).

Main Idea of The Algorithm

Steps of the algorithm

The algorithm iteratively executes the following steps:

1. Estimate gradients of the unknown functions;

Main Idea of The Algorithm

Steps of the algorithm

The algorithm iteratively executes the following steps:

1. Estimate gradients of the unknown functions;
2. Determine a descent search direction using gradient estimators;

Main Idea of The Algorithm

Steps of the algorithm

The algorithm iteratively executes the following steps:

1. Estimate gradients of the unknown functions;
2. Determine a descent search direction using gradient estimators;
3. Adjust the search direction if near boundaries;

Main Idea of The Algorithm

Steps of the algorithm

The algorithm iteratively executes the following steps:

1. Estimate gradients of the unknown functions;
2. Determine a descent search direction using gradient estimators;
3. Adjust the search direction if near boundaries;
4. Compute a local safe set for the next iteration point;

Main Idea of The Algorithm

Steps of the algorithm

The algorithm iteratively executes the following steps:

1. Estimate gradients of the unknown functions;
2. Determine a descent search direction using gradient estimators;
3. Adjust the search direction if near boundaries;
4. Compute a local safe set for the next iteration point;
5. Compute a suitable step length along the search direction within the local safe set.

Step 1: Compute gradient estimators

A gradient estimator is computed by the finite-difference method:

$$G^i(\boldsymbol{x}_k, \nu_k) = \sum_{j=1}^{d} \frac{f^i(\boldsymbol{x}_k + \nu_k \boldsymbol{e}_j) - f^i(\boldsymbol{x}_k)}{\nu_k} \boldsymbol{e}_j \tag{3}$$

where $\boldsymbol{e}_j$ is the j-th coordinate vector.

Step 1: Compute gradient estimators

A gradient estimator is computed by the finite-difference method:

$$G^i(\boldsymbol{x}_k, \nu_k) = \sum_{j=1}^{d} \frac{f^i(\boldsymbol{x}_k + \nu_k \boldsymbol{e}_j) - f^i(\boldsymbol{x}_k)}{\nu_k} \boldsymbol{e}_j \tag{3}$$

where $\boldsymbol{e}_j$ is the j-th coordinate vector.

The estimation deviation is upper bounded by:

$$\Delta_k^i = G^i(\boldsymbol{x}_k, \nu_k) - \nabla f^i(\boldsymbol{x}_k) \tag{4}$$

$$\|\Delta_k^i\|_2 \leq \frac{\sqrt{d} M \nu_k}{2} \tag{5}$$

Step 1: Compute gradient estimators

A gradient estimator is computed by the finite-difference method:

$$G^i(\boldsymbol{x}_k, \nu_k) = \sum_{j=1}^{d} \frac{f^i(\boldsymbol{x}_k + \nu_k \boldsymbol{e}_j) - f^i(\boldsymbol{x}_k)}{\nu_k} \boldsymbol{e}_j \tag{3}$$

where $\boldsymbol{e}_j$ is the j-th coordinate vector.

The estimation deviation is upper bounded by:

$$\Delta_k^i = G^i(\boldsymbol{x}_k, \nu_k) - \nabla f^i(\boldsymbol{x}_k) \tag{4}$$

$$\|\Delta_k^i\|_2 \leq \frac{\sqrt{d} M \nu_k}{2} \tag{5}$$

Difference step length

The use of the difference step length ν_k shall satisfy some conditions:

Difference step length

The use of the difference step length ν_k shall satisfy some conditions:

- To guarantee safety:

$$\nu_k \leq \frac{\min_i -f^i(\boldsymbol{x}_k)}{2L} \tag{6}$$

where L is the *Lipschitz* constant of $f^i(\boldsymbol{x}_k)$.

Difference step length

The use of the difference step length ν_k shall satisfy some conditions:

- To guarantee safety:

$$\nu_k \leq \frac{\min_i -f^i(\boldsymbol{x}_k)}{2L} \tag{6}$$

where L is the *Lipschitz* constant of $f^i(\boldsymbol{x}_k)$.

- To guarantee accuracy:

$$\nu_k = \frac{2\mu}{\sqrt{dM}} \tag{7}$$

where M is the smoothness constant of $f^i(\boldsymbol{x}_k)$ and $\|\Delta_k^i\|_2 \leq \mu$.

Difference step length

The use of the difference step length ν_k shall satisfy some conditions:

- To guarantee safety:

$$\nu_k \leq \frac{\min_i -f^i(\boldsymbol{x}_k)}{2L} \tag{6}$$

where L is the *Lipschitz* constant of $f^i(\boldsymbol{x}_k)$.

- To guarantee accuracy:

$$\nu_k = \frac{2\mu}{\sqrt{d}M} \tag{7}$$

where M is the smoothness constant of $f^i(\boldsymbol{x}_k)$ and $\|\Delta_k^i\|_2 \leq \mu$.

- In total:

$$\nu_k = \min\left\{\frac{2\mu}{\sqrt{d}M}, \frac{\min_i -f^i(\boldsymbol{x}_k)}{2L}\right\} \tag{8}$$

Step 2: Determine a search direction

The search direction $\boldsymbol{p}_k$ can be chosen from the following options:

- The steepest descent direction:

$$\boldsymbol{p}_k = -G^0(\boldsymbol{x}_k, \nu_k) \tag{9}$$

Step 2: Determine a search direction

The search direction $\boldsymbol{p}_k$ can be chosen from the following options:

- The steepest descent direction:

$$\boldsymbol{p}_k = -G^0(\boldsymbol{x}_k, \nu_k) \tag{9}$$

- The *quasi-Newton* direction:

$$\boldsymbol{p}_k^N = -\boldsymbol{H}_k G^0(\boldsymbol{x}_k, \nu_k) \tag{10}$$

where $\boldsymbol{H}_k$ is an approximate inverse Hessian matrix that can be updated after each iteration.

Step 3: Search direction adjustment

Drawbacks of abovementioned search directions:

- Consider only about the objective function;

Step 3: Search direction adjustment

Drawbacks of abovementioned search directions:

- Consider only about the objective function;
- Neglect the limitations of safety boundaries.

Step 3: Search direction adjustment

Drawbacks of abovementioned search directions:

- Consider only about the objective function;
- Neglect the limitations of safety boundaries.

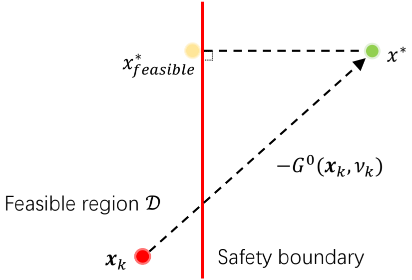


Figure 2: Search direction before adjustment

Step 3: Search direction adjustment

Drawbacks of abovementioned search directions:

- Consider only about the objective function;
- Neglect the limitations of safety boundaries.

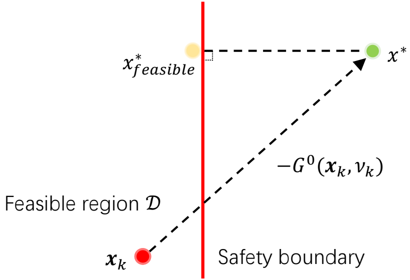


Figure 2: Search direction before adjustment

How can we achieve the optimal feasible solution?

Step 3 Search direction adjustment

Solution: Project the search direction on the the normal vector of the gradient of active constraint.

Step 3 Search direction adjustment

Solution: Project the search direction on the the normal vector of the gradient of active constraint.

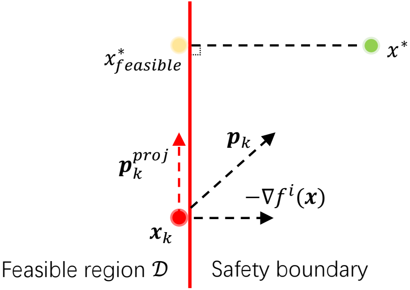


Figure 3: Search direction projection

Step 3 Search direction adjustment

Solution: Project the search direction on the the normal vector of the gradient of active constraint.

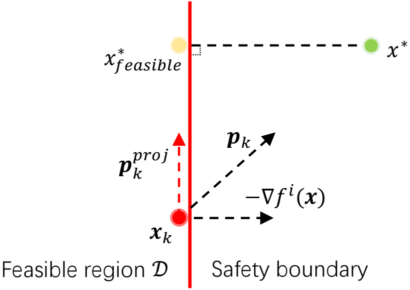


Figure 3: Search direction projection

How to find the $\boldsymbol{p}_k^{proj}$ universally?

Search direction projection

When near safety boundaries by the following condition:

$$0 \leq \min\left\{-f^i(\boldsymbol{x}_k)\right\} \leq h \tag{11}$$

where $h > 0$ is a safety threshold,

Search direction projection

When near safety boundaries by the following condition:

$$0 \leq \min\left\{-f^i(\boldsymbol{x}_k)\right\} \leq h \tag{11}$$

where $h > 0$ is a safety threshold, we do:

- Project $\boldsymbol{p}_k$ onto $\boldsymbol{p}_k^{proj}$;

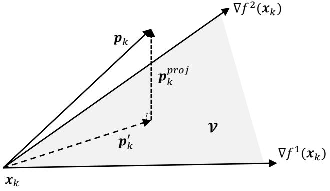


Figure 4: Search direction projection in $\mathbb{R}^3$

Search direction projection

When near safety boundaries by the following condition:

$$0 \leq \min\left\{-f^i(\boldsymbol{x}_k)\right\} \leq h \tag{11}$$

where $h > 0$ is a safety threshold, we do:

- Project $\boldsymbol{p}_k$ onto $\boldsymbol{p}_k^{proj}$;
- $\boldsymbol{p}_k^{proj} \perp \boldsymbol{p}'_k$;

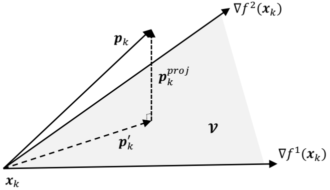


Figure 4: Search direction projection in $\mathbb{R}^3$

Search direction projection

When near safety boundaries by the following condition:

$$0 \leq \min\left\{-f^i(\boldsymbol{x}_k)\right\} \leq h \tag{11}$$

where $h > 0$ is a safety threshold, we do:

- Project $\boldsymbol{p}_k$ onto $\boldsymbol{p}_k^{proj}$;
- $\boldsymbol{p}_k^{proj} \perp \boldsymbol{p}'_k$;
- $\boldsymbol{p}'_k = \arg\min\{\|\boldsymbol{p}'_k - \boldsymbol{p}_k\|_2,\ \boldsymbol{p}'_k \in \mathcal{V}\}$.

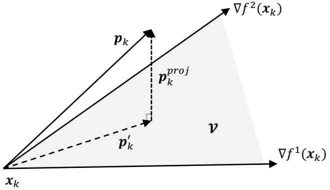


Figure 4: Search direction projection in $\mathbb{R}^3$

Search direction projection

Obtain $\boldsymbol{p}_k^{proj}$ and $\boldsymbol{p}_k'$: Solve a non-negative least-squares problem.

Search direction projection

Obtain $\boldsymbol{p}_k^{proj}$ and $\boldsymbol{p}_k'$: Solve a non-negative least-squares problem.

Problem: Non-Negative Least-Squares Problem

$$\underset{\boldsymbol{\lambda}\in\mathbb{R}^{|\mathcal{A}|}}{\text{minimize}} \quad \|\boldsymbol{C}_k\boldsymbol{\lambda} - \boldsymbol{p}_k\|_2^2 \tag{12}$$

$$\text{subject to} \quad \lambda^i \geq 0, \; i \in \mathcal{A} \tag{13}$$

where $\mathcal{A}$ is the set of indices of active constraints, and

$$\boldsymbol{C}_k = \left[\hat{G}^i(\boldsymbol{x}_k, \nu_k), \ldots, \hat{G}^j(\boldsymbol{x}_k, \nu_k)\right], \; i, j \in \mathcal{A} \tag{14}$$

Search direction projection

Obtain $\boldsymbol{p}_k^{proj}$ and $\boldsymbol{p}'_k$: Solve a non-negative least-squares problem.

Problem: Non-Negative Least-Squares Problem

$$\underset{\boldsymbol{\lambda}\in\mathbb{R}^{|\mathcal{A}|}}{\text{minimize}} \quad \|\boldsymbol{C}_k\boldsymbol{\lambda} - \boldsymbol{p}_k\|_2^2 \tag{12}$$

$$\text{subject to} \quad \lambda^i \geq 0, \; i \in \mathcal{A} \tag{13}$$

where $\mathcal{A}$ is the set of indices of active constraints, and

$$\boldsymbol{C}_k = \left[\hat{G}^i(\boldsymbol{x}_k, \nu_k), \ldots, \hat{G}^j(\boldsymbol{x}_k, \nu_k)\right], \; i, j \in \mathcal{A} \tag{14}$$

Then, the two vectors are:

$$\boldsymbol{p}'_k = \boldsymbol{C}_k \lambda_k^{sol} \qquad \boldsymbol{p}_k^{proj} = \boldsymbol{p}_k - \boldsymbol{C}_k \boldsymbol{\lambda}_k^{sol} \tag{15}$$

Projection deviation

Except $\boldsymbol{p}_k^{proj}$, one can also use $\boldsymbol{p}_k^{proj,\theta}$:

Projection deviation

Except $\boldsymbol{p}_k^{proj}$, one can also use $\boldsymbol{p}_k^{proj,\theta}$:

- Direction deviating from $\boldsymbol{p}_k^{proj}$ by a specified angle θ;

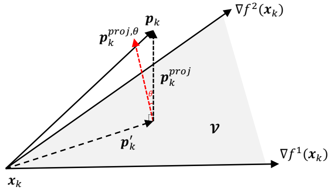


Figure 5: Projection deviation in $\mathbb{R}^3$

Projection deviation

Except $\boldsymbol{p}_k^{proj}$, one can also use $\boldsymbol{p}_k^{proj,\theta}$:

- Direction deviating from $\boldsymbol{p}_k^{proj}$ by a specified angle θ;
- Increase the convergence speed when near boundaries;

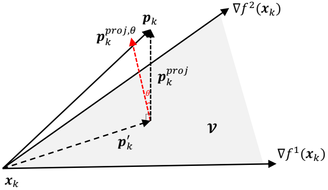


Figure 5: Projection deviation in $\mathbb{R}^3$

Projection deviation

Except $\boldsymbol{p}_k^{proj}$, one can also use $\boldsymbol{p}_k^{proj,\theta}$:

- Direction deviating from $\boldsymbol{p}_k^{proj}$ by a specified angle θ;
- Increase the convergence speed when near boundaries;

$$\boldsymbol{p}_k^{proj,\theta} = a\boldsymbol{p}'_k + \boldsymbol{p}_k^{proj} \tag{16}$$

where $\theta \in [0, \angle(\boldsymbol{p}_k, \boldsymbol{p}'_k)]$ and

$$a = \text{solve}\left\{\cos^2\theta \|\boldsymbol{p}'_k\|_2^2 a^2 + 2\cos^2\theta {\boldsymbol{p}'_k}^{\mathrm{T}} \boldsymbol{p}_{\boldsymbol{k}}^{proj} a + \left(\cos^2\theta - 1\right)\|\boldsymbol{p}_k^{proj}\|_2^2 = 0\right\} \leq 0 \tag{17}$$

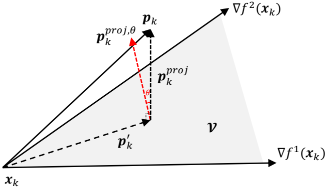


Figure 5: Projection deviation in $\mathbb{R}^3$

Step 4: Safe set formulation

Local safe set $\mathcal{S}_k \in \mathcal{D}$:

- A set within which all points are safety feasible;

Step 4: Safe set formulation

Local safe set $\mathcal{S}_k \in \mathcal{D}$:

- A set within which all points are safety feasible;
- The intersection of individual constraint-wise safe set $\mathcal{S}_k^i$.

$$\mathcal{S}_k := \cap_{i=1}^{m} \mathcal{S}_k^i \tag{18}$$

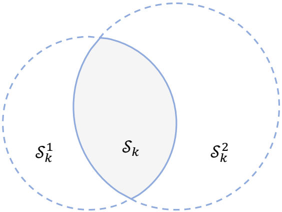


Figure 6: Illustration of a local safe set

Step 4: Safe set formulation

Local safe set $\mathcal{S}_k \in \mathcal{D}$:

- A set within which all points are safety feasible;
- The intersection of individual constraint-wise safe set $\mathcal{S}_k^i$.

$$\mathcal{S}_k := \cap_{i=1}^{m} \mathcal{S}_k^i \tag{18}$$

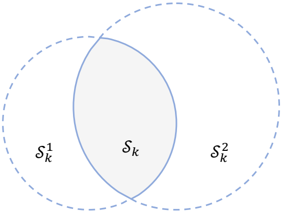


Figure 6: Illustration of a local safe set

How to formulate the constraint-wise safe set?

Constraint-wise safe set

Constraint-wise safe set:

- Solution set of an approximate safety constraint using Taylor's Theorem:

Constraint-wise safe set

Constraint-wise safe set:

- Solution set of an approximate safety constraint using Taylor's Theorem:

$$
\begin{aligned}
f^i(\boldsymbol{x}_{k+1}) \leq & f^i(\boldsymbol{x}_k) + \nabla f^i(\boldsymbol{x}_k)^{\mathrm{T}} (\boldsymbol{x}_{k+1} - \boldsymbol{x}_k) + \frac{1}{2} M \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2^2 \\
\leq & \, 0
\end{aligned} \tag{19}
$$

Constraint-wise safe set

Constraint-wise safe set:

- Solution set of an approximate safety constraint using Taylor's Theorem:

$$
\begin{aligned}
f^i(\boldsymbol{x}_{k+1}) &\leq f^i(\boldsymbol{x}_k) + \nabla f^i(\boldsymbol{x}_k)^{\mathrm{T}} \left(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\right) + \frac{1}{2} M \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2^2 \\
&\leq 0 \qquad (19)
\end{aligned}
$$

- Consider the estimation deviation $\|\Delta_k^i\|_2$;

Constraint-wise safe set

Constraint-wise safe set:

- Solution set of an approximate safety constraint using Taylor's Theorem:

$$\begin{aligned} f^i(\boldsymbol{x}_{k+1}) &\leq f^i(\boldsymbol{x}_k) + \nabla f^i(\boldsymbol{x}_k)^{\mathrm{T}} \left(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\right) + \frac{1}{2} M \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2^2 \\ &\leq 0 \end{aligned} \tag{19}$$

- Consider the estimation deviation $\|\Delta_k^i\|_2$;
- Replace $\nabla f^i(\boldsymbol{x}_k)$ by a modified gradient estimator $\hat{G}^i(\boldsymbol{x}_k, \nu_k)$.

Constraint-wise safe set

Constraint-wise safe set:

- Solution set of an approximate safety constraint using Taylor's Theorem:

$$f^i(\boldsymbol{x}_{k+1}) \leq f^i(\boldsymbol{x}_k) + \nabla f^i(\boldsymbol{x}_k)^{\mathrm{T}} (\boldsymbol{x}_{k+1} - \boldsymbol{x}_k) + \frac{1}{2} M \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2^2 \leq 0 \qquad (19)$$

- Consider the estimation deviation $\|\Delta_k^i\|_2$;
- Replace $\nabla f^i(\boldsymbol{x}_k)$ by a modified gradient estimator $\hat{G}^i(\boldsymbol{x}_k, \nu_k)$.

How to compute the modified gradient estimator?

Gradient estimator modification

What is $\hat{G}^i(\boldsymbol{x}_k, \nu_k)$?

- Compensate the estimation deviation $\|\Delta_k^i\|_2$ along the search direction $\boldsymbol{p}_k$.

Gradient estimator modification

What is $\hat{G}^i(\boldsymbol{x}_k, \nu_k)$?

- Compensate the estimation deviation $\|\Delta_k^i\|_2$ along the search direction $\boldsymbol{p}_k$.

The modified gradient estimator is computed by:

$$\hat{G}^i(\boldsymbol{x}_k, \nu_k) = G^i(\boldsymbol{x}_k, \nu_k) + \frac{\sqrt{d}M\nu_k\|\boldsymbol{p}_k\|_2}{2\boldsymbol{e}^{\mathrm{T}}\boldsymbol{p}_k}\boldsymbol{e} \tag{20}$$

where $\boldsymbol{p}_k$ is the search direction and $\boldsymbol{e}$ is an unit coordinate vector such that:

$$\boldsymbol{e}^{\mathrm{T}}\boldsymbol{p}_k \neq 0 \tag{21}$$

Gradient estimator modification

What is $\hat{G}^i(\boldsymbol{x}_k, \nu_k)$?

- Compensate the estimation deviation $\|\Delta_k^i\|_2$ along the search direction $\boldsymbol{p}_k$.

The modified gradient estimator is computed by:

$$\hat{G}^i(\boldsymbol{x}_k, \nu_k) = G^i(\boldsymbol{x}_k, \nu_k) + \frac{\sqrt{d}M\nu_k\|\boldsymbol{p}_k\|_2}{2\boldsymbol{e}^{\mathrm{T}}\boldsymbol{p}_k}\boldsymbol{e} \tag{20}$$

where $\boldsymbol{p}_k$ is the search direction and $\boldsymbol{e}$ is an unit coordinate vector such that:

$$\boldsymbol{e}^{\mathrm{T}}\boldsymbol{p}_k \neq 0 \tag{21}$$

Remark: Equation (20) is the solution of the following inequality:

$$\hat{G}^i(\boldsymbol{x}_k, \nu_k)^{\mathrm{T}}\boldsymbol{p}_k - G^i\left(\boldsymbol{x}_k, \nu_k\right)^{\mathrm{T}}\boldsymbol{p}_k \geq \frac{\sqrt{d}M\nu_k}{2} \geq \|\Delta_k^i\|_2$$

Safe set formulation

Substituted with $\hat{G}^i(\boldsymbol{x}_k, \nu_k)$ (20), Inequality (19)[1] defines a hyper-ball with the center and radius as following:

$$\hat{\mathcal{O}}_k^i = \boldsymbol{x}_k - \frac{1}{M}\hat{G}^i(\boldsymbol{x}_k, \nu_k) \tag{22}$$

$$\hat{\mathcal{R}}_k^i = \sqrt{\frac{1}{M}\left(\frac{\|\hat{G}^i(\boldsymbol{x}_k, \nu_k)\|_2^2}{M} - 2f^i(\boldsymbol{x}_k)\right)} \tag{23}$$

[1] $f^i(\boldsymbol{x}_k) + \nabla f^i(\boldsymbol{x}_k)^{\mathrm{T}}\left(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\right) + \frac{1}{2}M\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2^2 \leq 0$

Safe set formulation

Substituted with $\hat{G}^i(\boldsymbol{x}_k, \nu_k)$ (20), Inequality (19)[1] defines a hyper-ball with the center and radius as following:

$$\hat{\mathcal{O}}_k^i = \boldsymbol{x}_k - \frac{1}{M}\hat{G}^i(\boldsymbol{x}_k, \nu_k) \tag{22}$$

$$\hat{\mathcal{R}}_k^i = \sqrt{\frac{1}{M}\left(\frac{\|\hat{G}^i(\boldsymbol{x}_k, \nu_k)\|_2^2}{M} - 2f^i(\boldsymbol{x}_k)\right)} \tag{23}$$

Therefore, the local safe set $\mathcal{S}_k$ is defined as:

$$\mathcal{S}_k := \left\{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x} - \hat{\mathcal{O}}_k^i\|_2 \leq \hat{\mathcal{R}}_k^i, \ \forall i = 1, \ldots, m\right\} \tag{24}$$

[1] $f^i(\boldsymbol{x}_k) + \nabla f^i(\boldsymbol{x}_k)^{\mathrm{T}}(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k) + \frac{1}{2}M\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2^2 \leq 0$

Step 5: Step length selection

Conditions for step length selection:

1. It is safety-guaranteed;

Step 5: Step length selection

Conditions for step length selection:

1. It is safety-guaranteed;

2. It provides a sufficient decrease in the objective function.

Step 5: Step length selection

Conditions for step length selection:

1. It is safety-guaranteed;
   It is fulfilled when the following holds:

$$\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k \in \mathcal{S}_k \tag{25}$$

2. It provides a sufficient decrease in the objective function.

Step 5: Step length selection

Conditions for step length selection:

1. It is safety-guaranteed;
   It is fulfilled when the following holds:

$$\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k \in \mathcal{S}_k \tag{25}$$

2. It provides a sufficient decrease in the objective function.
   It can be checked using the *Armijo condition*:

$$f^0(\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k) < f^0(\boldsymbol{x}_k) + c\alpha_k G^0(\boldsymbol{x}_k, \nu_k)^{\mathrm{T}} \boldsymbol{p}_k \tag{26}$$

where $c \in (0, 1)$ is a tuning parameter that affects the convergence speed.

Armijo condition

Let

$$l(\alpha) = f^0(\boldsymbol{x}_k) + c\alpha_k G^0(\boldsymbol{x}_k, \nu_k)^{\mathrm{T}} \boldsymbol{p}_k \tag{27}$$

an illustration of the *Armijo condition* is shown in Figure 7.

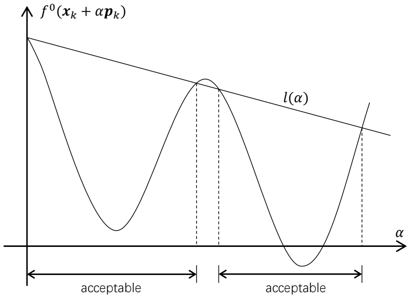


Figure 7: Illustration of the *Armijo condition*

Illustration of the algorithm

The main idea of the algorithm is shown in Figure 8.

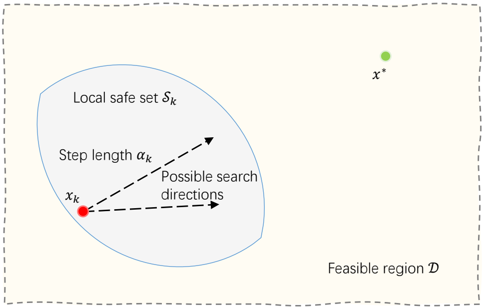


Figure 8: Illustration of the algorithm

Algorithm 1 Safe line-search optimization for exact measurements (e-SLS) (Pseudocode)

while not convergent **do**
  Compute gradient estimators $G^i(\boldsymbol{x}, \nu_k), i = 0, \ldots, m$ with selected difference step length ν_k;
  Determine the search direction $\boldsymbol{p}_k$;
  Compute $\hat{G}^i(\boldsymbol{x}, \nu_k),\ i = 1, \ldots, m$;
  if $\min\{-f^i(\boldsymbol{x}_k)\} \leq h$ **then**
    Determine the projected search direction $\boldsymbol{p}_k = \boldsymbol{p}_k^{proj}$;
    Recompute $\hat{G}^i(\boldsymbol{x}, \nu_k),\ i = 1, \ldots, m$;
  end if
  Compute centers $\mathcal{O}_k^i$ and radii $\mathcal{R}_k^i$ of the safe set $\mathcal{S}_k$;
  Select a step length α_k along $\boldsymbol{p}_k$;
  Evaluate $f^i(\boldsymbol{x})$ at the next iterate $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k$.
end while

Influences of measurement noise

The main influences of the measurement noise:

- Increase the uncertainty of function measurements of safety constraints;

Influences of measurement noise

The main influences of the measurement noise:

- Increase the uncertainty of function measurements of safety constraints;
- Increase the estimation deviation of gradient estimators.

Influences of measurement noise

The main influences of the measurement noise:

- Increase the uncertainty of function measurements of safety constraints;
- Increase the estimation deviation of gradient estimators.

How to deal with the measurement noise?

Handling measurement noises

Measurement noise ξ: Independently and identically distributed (i.i.d) *zero-mean-σ-sub-Gaussian* random variable.

Handling measurement noises

Measurement noise ξ: Independently and identically distributed (i.i.d) *zero-mean-σ-sub-Gaussian* random variable.

Algorithm modifications:

1. Replace the single function measurement by the average of multiple measurements;

Handling measurement noises

Measurement noise ξ: Independently and identically distributed (i.i.d) *zero-mean-σ-sub-Gaussian* random variable.

Algorithm modifications:

1. Replace the single function measurement by the average of multiple measurements;
2. Add a safety filter to the step length selection conditions.

Average of multiple measurements

Estimation deviation upper bound with high probability:

$$\mathbb{P}\left\{\|\Delta^i_{k,\xi^i}\|_2 \leq \sqrt{\frac{dM^2\nu_k^2}{4} - \frac{4d\sigma^2\ln\delta}{\nu_k^2 n_k}}\right\} \geq 1-\delta \tag{28}$$

Average of multiple measurements

Estimation deviation upper bound with high probability:

$$\mathbb{P}\left\{\|\Delta^i_{k,\xi^i}\|_2 \leq \sqrt{\frac{dM^2\nu_k^2}{4} - \frac{4d\sigma^2 \ln \delta}{\nu_k^2 n_k}}\right\} \geq 1-\delta \qquad (28)$$

With the number of measurements $n_k = -\frac{16\sigma^2 \ln \delta}{3\nu_k^4 M^2}$, the upper bound of $\|\Delta^i_{k,\xi^i}\|_2$ becomes:

$$\mathbb{P}\left\{\|\Delta^i_{k,\xi^i}\|_2 \leq \sqrt{d}M\nu_k\right\} \geq 1-\delta \qquad (29)$$

Average of multiple measurements

Estimation deviation upper bound with high probability:

$$\mathbb{P}\left\{\|\Delta^i_{k,\xi^i}\|_2 \leq \sqrt{\frac{dM^2\nu_k^2}{4} - \frac{4d\sigma^2 \ln\delta}{\nu_k^2 n_k}}\right\} \geq 1-\delta \tag{28}$$

With the number of measurements $n_k = -\frac{16\sigma^2 \ln\delta}{3\nu_k^4 M^2}$, the upper bound of $\|\Delta^i_{k,\xi^i}\|_2$ becomes:

$$\mathbb{P}\left\{\|\Delta^i_{k,\xi^i}\|_2 \leq \sqrt{d}M\nu_k\right\} \geq 1-\delta \tag{29}$$

Modified gradient estimator with $\|\Delta^i_{k,\xi^i}\|_2$ for noisy measurement:

$$\hat{G}^i(\boldsymbol{x}_k, \nu_k; \xi^i) = \bar{G}^i(\boldsymbol{x}_k, \nu_k; \xi^i) + \sqrt{\frac{dM^2\nu_k^2}{4} - \frac{4d\sigma^2 \ln\delta}{\nu_k^2 n_k}}\frac{\|\boldsymbol{p}_k\|_2 \boldsymbol{e}}{\boldsymbol{e}^{\mathrm{T}}\boldsymbol{p}_k} \tag{30}$$

Safety filter in the step length selection

Additional step length selection condition:

1. It is safety-guaranteed with high probability;
2. It provides a sufficient decrease in the objective function;
3. It passes a safety filter.

Safety filter in the step length selection

Additional step length selection condition:

1. It is safety-guaranteed with high probability;
2. It provides a sufficient decrease in the objective function;
3. It passes a safety filter.
   The safety filter is defined as:

$$\min \left\{ -\bar{f}^i(\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k; \xi^i) \right\} \geq h \qquad (31)$$

   where $h > 0$ is a safety threshold.

Safety filter in the step length selection

Additional step length selection condition:

1. It is safety-guaranteed with high probability;
2. It provides a sufficient decrease in the objective function;
3. It passes a safety filter.
   The safety filter is defined as:

$$\min\left\{-\bar{f}^i(\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k; \xi^i)\right\} \geq h \tag{31}$$

where $h > 0$ is a safety threshold.

This ends the modification of the algorithm for the case of noisy measurements.

1. Introduction
2. Literature Review
3. Methods
4. Results
   - Simulation Analysis Outline
   - Numerical Examples
   - Optimal Power Flow Problem
5. Conclusion

1. Introduction
2. Literature Review
3. Methods
4. Results
   - Simulation Analysis Outline
   - Numerical Examples
   - Optimal Power Flow Problem
5. Conclusion

Simulation analysis outline

Algorithm test problems;

- Three numerical examples;
- Optimal power flow problem.

Simulation analysis outline

Algorithm test problems;

- Three numerical examples;
- Optimal power flow problem.

Algorithm only accesses function values.

1. Introduction
2. Literature Review
3. Methods
4. Results
   - Simulation Analysis Outline
   - Numerical Examples
   - Optimal Power Flow Problem
5. Conclusion

Numerical example 1

We test the algorithm on the first example:

$$\underset{x\in\mathbb{R}^2}{\text{minimize}} \quad (x_1-2.7)^2+0.5(x_2-0.5)^2-5 \tag{32}$$

$$\text{subject to} \quad x_1 \leq 2.7 \tag{33}$$

$$x_2 \geq -5 \tag{34}$$

with linear constraints.

Results of example 1

Optimization trajectory of example 1 and an illustration of the safe set formulation in Figure 9.

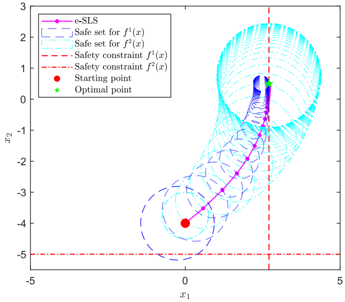
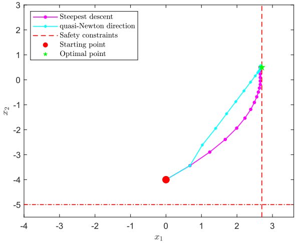


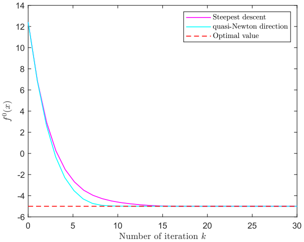
Figure 9: Optimization trajectory

Results of example 1

Comparison between the steepest descent and the *quasi-Newton* direction in Figure 10.



(a) Optimization trajectory

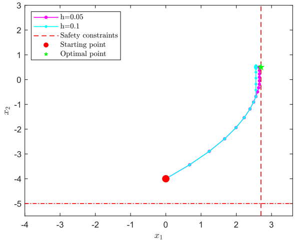


(b) Objective function

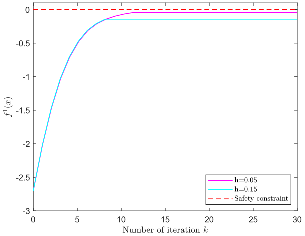
Figure 10: Comparison of different search directions

Results of example 1

Comparison between different safety thresholds h in Figure 11.



(a) Optimization trajectory

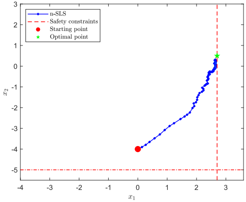


(b) Constraint function

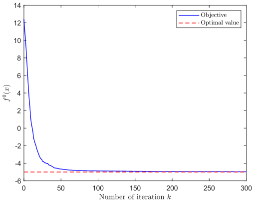
Figure 11: Comparison of different safety threshold parameters

Results of example 1

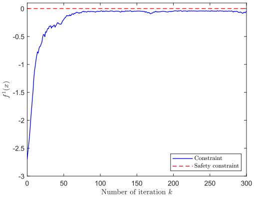
Optimization result with noisy measurements, $\sigma = 0.01$, $n_k = 10$.



(a) Optim. trajectory



(b) Objective function



(c) Constraint functions

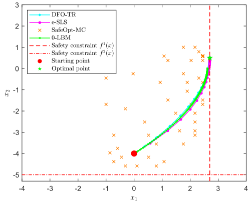
Figure 12: Optimization results with noisy measurements of example 1

Algorithm comparisons

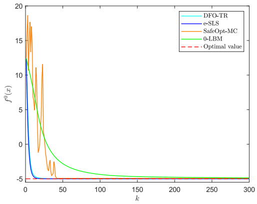
Comparison with aforementioned model-free optimization algorithms:

- Safe Logarithmic Barrier method (0-LBM);
- Derivative-Free Optimization with Trust-Region (DFO-TR);
- Safe Bayesian Optimization with Multiple Constraints (SafeOpt-MC).

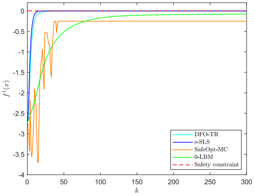
Results of example 1



(a) Optim. trajectory



(b) Objective function



(c) Constraint functions

Figure 13: Optimization results of different algorithms for example 1

Results of example 1

Table 1: Optimization performance of different algorithms

Algorithms	Number of Iterations to Convergence	Computation Time to Convergence	Optimality Gap
DFO-TR	22	28.071s	0.240%
e-SLS	**19**	0.060s	**0.083%**
SafeOpt-MC	42	35.872	1.342%
0-LBM	300	**0.031s**	2.601%

Numerical example 2

We test the algorithm on the second example:

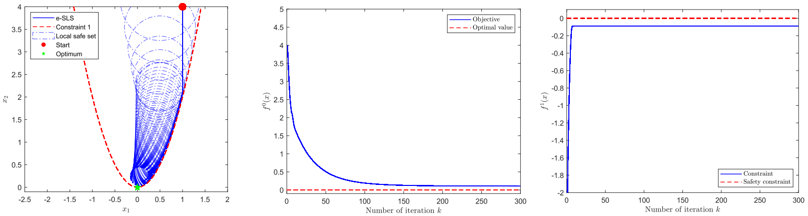
$$\underset{x\in\mathbb{R}^2}{\text{minimize}} \quad x_2 \tag{35}$$

$$\text{subject to} \quad 2x_1^2 - x_2 \leq 0 \tag{36}$$

with a nonlinear convex constraint.

Results of example 2

Optimization result with search direction projection $\boldsymbol{p}_k^{proj}$.

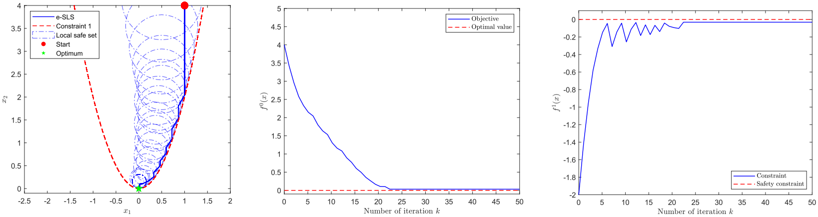


(a) Optim. trajectory (b) Objective function (c) Constraint functions

Figure 14: Optimization result of example 2

Results of example 2

Optimization result with search direction projection deviation $\boldsymbol{p}_k^{proj,\theta}$, $\theta = 15°$.



(a) Optim. trajectory (b) Objective function (c) Constraint functions

Figure 15: Optimization results with projection deviation of example 2

Numerical example 3

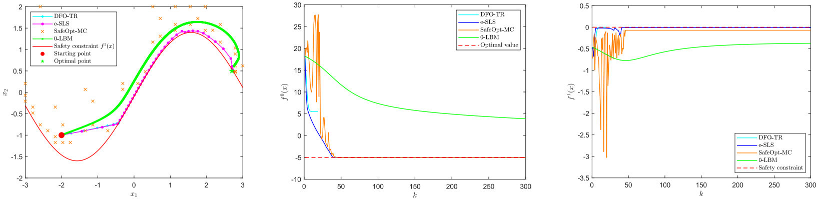
We test the algorithm on the third example:

$$\underset{x \in \mathbb{R}^2}{\text{minimize}} \quad (x_1 - 2.7)^2 + 0.5(x_2 - 0.5)^2 - 5 \tag{37}$$

$$\text{subject to} \quad 1.5 \sin(x_1) - x_2 \leq 0 \tag{38}$$

with a nonlinear nonconvex constraint.

Results of example 3



(a) Optim. trajectory (b) Objective function (c) Constraint functions

Figure 16: Optimization results of different algorithms for example 3

Problem description

The AC-Optimal Power Flow (AC-OPF) problem is generally a large-scale nonlinear nonconvex optimization problem, which concerns in its basic form about optimizing the total generation cost considering various physical, operational constraints.

Problem formulation

The AC-OPF problem is formulated as:

$$\text{minimize} \quad \sum_{k \in G} c_{1k} {P_k^g}^2 + c_{1k} P_k^g + c_{0k}$$

subject to

$$S_i^g - S_i^d = |V_i| \sum_{(l,i,j) \in E} |Y_{ij}||V_j| \angle(\delta_i - \theta_{ij} - \delta_j), \ \forall i \in N \quad (39)$$

$$\left(P_k^g\right)^{lb} \leq P_k^g \leq \left(P_k^g\right)^{ub}, \ \forall k \in G \quad (40)$$

$$\left(Q_k^g\right)^{lb} \leq Q_k^g \leq \left(Q_k^g\right)^{ub}, \ \forall k \in G \quad (41)$$

$$|V_i|^{lb} \leq |V_i| \leq |V_i|^{ub}, \ \forall i \in N \quad (42)$$

$$|S_{ij}| \leq |S_{ij}|^{ub}, \ (l,i,j) \in E \quad (43)$$

$$|S_{ji}| \leq |S_{ji}|^{ub}, \ (l,i,j) \in E \quad (44)$$

IEEE-30 bus system

The algorithm is applied to an IEEE-30 bus test system as shown in Figure 17:

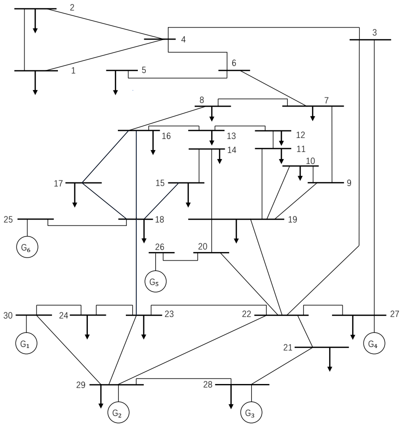
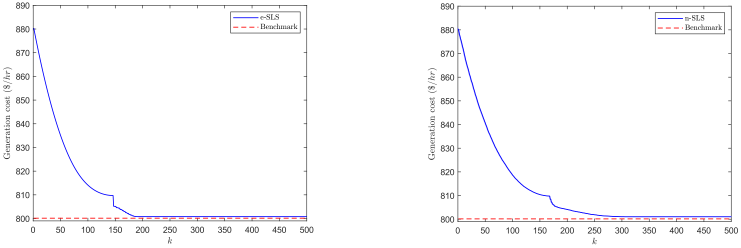


Figure 17: Single line diagram of IEEE-30 bus system

- 30 buses
- 6 generators
- 11 decision variables
- 158 constraints

Results of the AC-OPF problem for IEEE-30 bus system

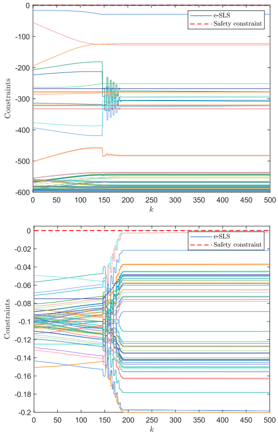


(a) Objective with exact measurements

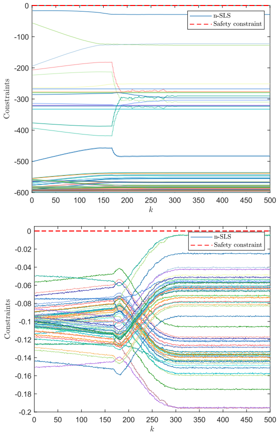
(b) Objective with noisy measurements

Figure 18: Objective function of AC-OPF problem for IEEE-30 bus system

Results of AC-OPF problem on IEEE-30 bus system



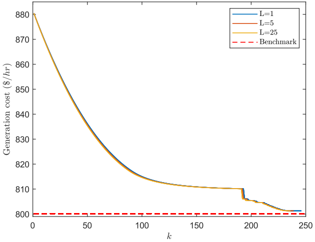
(a) Constraints with exact measurements



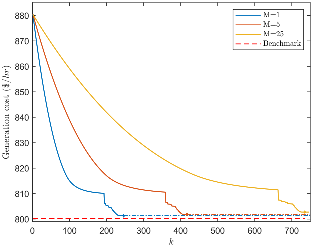
(b) Constraints with noisy measurements

Figure 19: Constraints of AC-OPF problem for IEEE-30 bus system

Influences of L and M constants



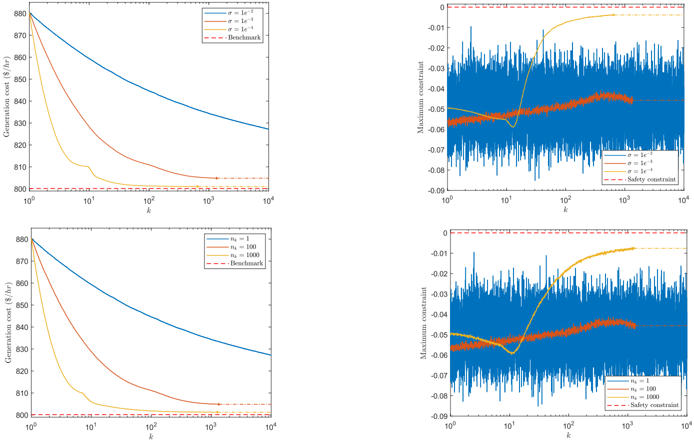
(a) Objective with different Lipschitz constants



(b) Objective with different smothness constants

Figure 20: Influences of L and M constants

Influences of σ and n_k



(a) Objective with different σ and n_k

(b) Constraints with different σ and n_k

Figure 21: Influences of σ and n_k

Conclusion

In this Master's thesis, a model-free safe optimization algorithm was developed. This algorithm utilizes a line-search optimization scheme, exploiting the smoothness properties of unknown functions to provide safety-guarantee.

References

[1] Charles Audet and Warren Hare.
Derivative-free and blackbox optimization, volume 2.
Springer, 2017.

[2] Felix Berkenkamp, Andreas Krause, and Angela P Schoellig.
Bayesian optimization with safety constraints: safe and automatic parameter tuning in robotics.
Machine Learning, pages 1–35, 2021.

[3] Peter I Frazier.
A tutorial on bayesian optimization.
arXiv preprint arXiv:1807.02811, 2018.

[4] Jeffrey Larson, Matt Menickelly, and Stefan M Wild.
Derivative-free optimization methods.
Acta Numerica, 28:287–404, 2019.

[5] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas.
Taking the human out of the loop: A review of bayesian optimization.
Proceedings of the IEEE, 104(1):148–175, 2015.

[6] Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause.
Safe exploration for optimization with gaussian processes.
In *International conference on machine learning*, pages 997–1005. PMLR, 2015.

[7] Ilnura Usmanova, Andreas Krause, and Maryam Kamgarpour.
Log barriers for safe non-convex black-box optimization.
arXiv preprint arXiv:1912.09478, 2019.

[8] Stephen Wright, Jorge Nocedal, et al.
Numerical optimization.
Springer Science, 35(67-68):7, 1999.

[9] MingHao Xu.
Ritsumeikan beamer theme.
In *How to write beautiful LaTeX*, 2022.

Thank You