Adaptive Stochastic Optimization with Constraints

Mladen Kolar (mkolar@usc.edu)

Collaborators

Sen Na

Yuchen Fang

Ilgee Hong

Mihai Anitescu

Michael Mahoney

Stochastic optimization problems with constraints

Stochastic nonlinear optimization problem with deterministic constraints:

$$\begin{aligned} \min_{\boldsymbol{x} \in \mathbb{R}^d} \quad & f(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x}; \xi)] \\ \text{s.t.} \quad & c_{\mathcal{E}}(\boldsymbol{x}) = \boldsymbol{0} \\ & c_{\mathcal{I}}(\boldsymbol{x}) \leqslant \boldsymbol{0} \end{aligned}$$

- $f : \mathbb{R}^d \to \mathbb{R}$ is the stochastic objective
- $c_{\mathcal{E}} : \mathbb{R}^d \to \mathbb{R}^m$ are the deterministic equality constraints
- $c_{\mathcal{I}} : \mathbb{R}^d \to \mathbb{R}^r$ are the deterministic inequality constraints

We do not have access to f and its derivatives

Have access to i.i.d. samples $\{\xi_i\}_i$ from $\mathcal{P}$ and the realizations $\{f(\cdot\,; \xi_i)\}_i$ that we use to estimate f and its derivatives

Applications in statistical machine learning

Finite sum objective

$$f(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}; \xi_i) = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}; y_i, \boldsymbol{z}_i)$$

- distribution $\mathcal{P}$ is uniform over feature-outcome pairs $\{\xi_i = (y_i, \boldsymbol{z}_i)\}_{i=1}^n$

Applications in statistical machine learning

Finite sum objective

$$f(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}; \xi_i) = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}; y_i, \boldsymbol{z}_i)$$

- distribution $\mathcal{P}$ is uniform over feature-outcome pairs $\{\xi_i = (y_i, \boldsymbol{z}_i)\}_{i=1}^{n}$

Constrained maximum likelihood optimization

Nagaraj and Fuller [1991], Dupacova and Wets [1988], Shapiro [2000]

- constraints encode some prior knowledge on the model parameters

Applications in statistical machine learning

Finite sum objective

$$f(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}; \xi_i) = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}; y_i, \boldsymbol{z}_i)$$

- distribution $\mathcal{P}$ is uniform over feature-outcome pairs $\{\xi_i = (y_i, \boldsymbol{z}_i)\}_{i=1}^{n}$

Constrained lasso Gaines et al. [2018]

$$\min \frac{1}{n}\sum_{i=1}^{n}(y_i - \boldsymbol{z}_i^\top \boldsymbol{x})^2 + \lambda \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad A\boldsymbol{x} \leqslant \mathbf{b}$$

- see also James et al. [2020]

$$\min f(\boldsymbol{x}) + \lambda \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad A\boldsymbol{x} \leqslant \mathbf{b}$$

- portfolio estimation
- monotone curve estimation
- generalized lasso

Applications in statistical machine learning

Finite sum objective

$$f(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}; \xi_i) = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}; y_i, \boldsymbol{z}_i)$$

- distribution $\mathcal{P}$ is uniform over feature-outcome pairs $\{\xi_i = (y_i, \boldsymbol{z}_i)\}_{i=1}^{n}$

Constrained deep neural networks

Nandwani et al. [2019], Ravi et al. [2019], Prach and Lampert [2022]

- constraints improve generalization performance
- constraints encode expert's knowledge
- constructing Lipschitz networks

Applications in statistical machine learning

Finite sum objective

$$f(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x};\xi_i) = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}; y_i, \boldsymbol{z}_i)$$

- distribution $\mathcal{P}$ is uniform over feature-outcome pairs $\{\xi_i = (y_i, \boldsymbol{z}_i)\}_{i=1}^{n}$

Constrained deep neural networks

Nandwani et al. [2019], Ravi et al. [2019], Prach and Lampert [2022]

- constraints improve generalization performance
- constraints encode expert's knowledge
- constructing Lipschitz networks

Machine learning with physics constraints

Willard et al. [2020], Karniadakis et al. [2021]

Other applications and related problems

Fairness constraints

Chen et al. [2022]

Optimal control

Kupfer and Sachs [1992], Betts [2010]

Nonlinear equality-constrained dynamic program

Na et al. [2021a]

PDE-constrained optimization

Rees et al. [2010]

Network flow

Bertsekas [1998]

Safe reinforcement learning

Shalev-Shwartz et al. [2016], Yu et al. [2019]

Unconstrained optimization

Gradient descent

$$\min_{x \in \mathbb{R}^n} f(x)$$

- $\nabla f : \mathbb{R}^n \mapsto \mathbb{R}^n$ is Lipschitz continuous with constant L

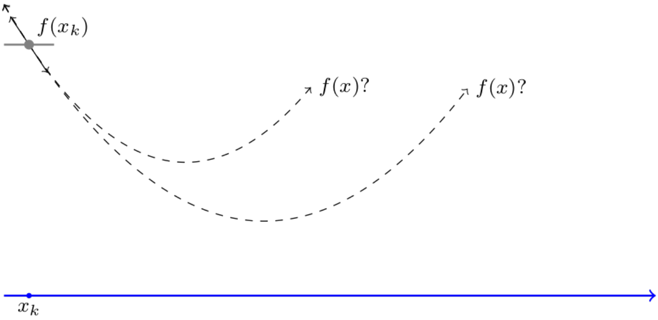
Gradient descent: choose an initial point $x_0 \in \mathbb{R}^n$, repeat:

$$x_{k+1} = x_k - \alpha \nabla f(x_k), \qquad k \geqslant 1$$

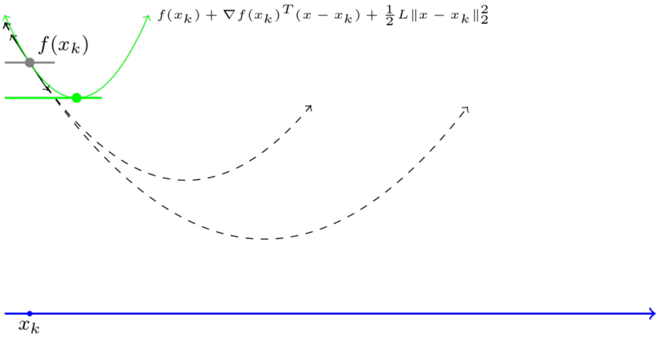
Stop at some point

Gradient descent

Gradient descent



Gradient descent

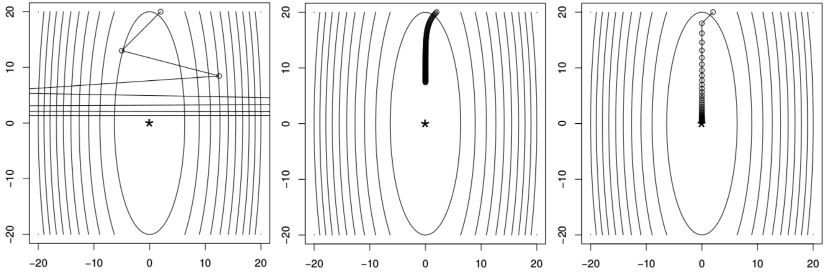


Gradient descent

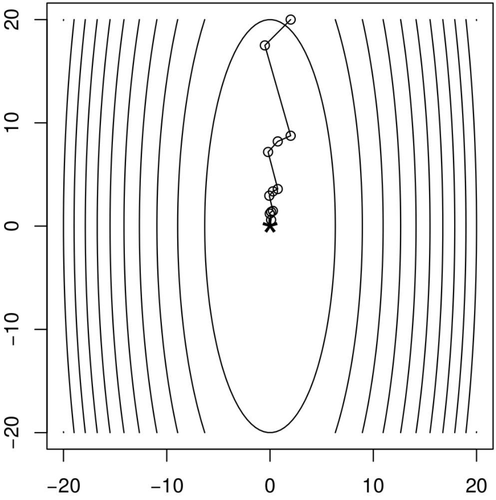
Theorem:
If $\alpha \in (0, 2/L)$, then $\sum_{k=0}^{\infty} \|\nabla f(x_k)\|_2^2 \leqslant \infty$, which implies $\{\nabla f(x_k)\} \to 0$.

$$\begin{aligned} f(x_{k+1}) - f(x_k) &\leqslant \nabla f(x_k)^T (x_{k+1} - x_k) + \frac{L}{2}\|x_{k+1} - x_k\|_2^2 \\ &= -\alpha\|\nabla f(x_k)\|_2^2 + \frac{L}{2}\alpha^2\|\nabla f(x_k)\|_2^2 \\ &\leqslant -\frac{1}{2}\alpha\|\nabla f(x_k)\|_2^2 \end{aligned}$$

How to choose an appropriate step?



Backtracking line search



Stochastic gradient descent

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} \; f(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x};\xi)]$$

- $\nabla f : \mathbb{R}^n \mapsto \mathbb{R}^n$ is Lipschitz continuous with constant L

Stochastic gradient descent: choose an initial point $x_0 \in \mathbb{R}^n$ and stepsizes $\{\alpha_k\}$, repeat:

$$x_{k+1} = x_k - \alpha_k g_k, \qquad k \geqslant 1$$

- where $\mathbb{E}_k[g_k] = \nabla f(x_k)$

Stop at some point

Stochastic gradient descent

Not a descent method

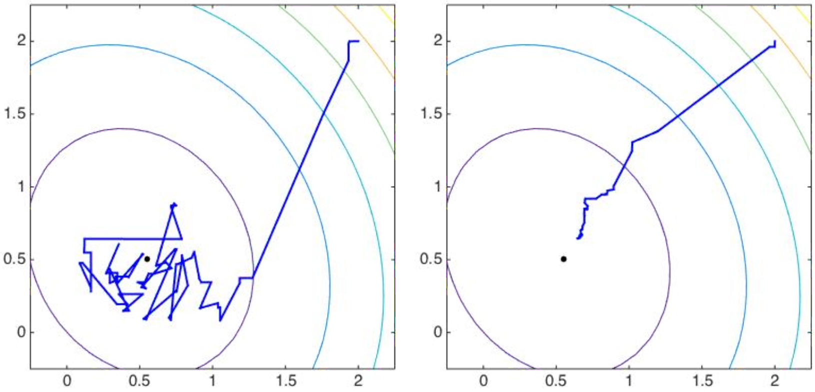
$$
\begin{aligned}
f(x_{k+1}) - f(x_k) &\leqslant \nabla f(x_k)^T (x_{k+1} - x_k) + \frac{L}{2}\|x_{k+1} - x_k\|_2^2 \\
&= -\alpha_k \nabla f(x_k)^T g_k + \frac{L}{2}\alpha_k^2 \|g_k\|_2^2 \\
\implies \mathbb{E}_k[f(x_{k+1})] - f(x_k) &\leqslant -\alpha_k \|\nabla f(x_k)\|_2^2 + \frac{L}{2}\alpha_k^2 \mathbb{E}_k[\|g_k\|_2^2]
\end{aligned}
$$

- eventual descent in expectation

Theorem:
If $\mathbb{E}_k[\|g_k - \nabla f(x_k)\|_2^2] \leqslant M$, then

$$
\begin{aligned}
\alpha_k &= \frac{1}{L} & \implies \mathbb{E}\left[\frac{1}{k}\sum_{j=1}^{k} \|\nabla f(x_j)\|_2^2\right] \leqslant \mathcal{O}(M) \\
\alpha_k &= \mathcal{O}\left(\frac{1}{k}\right) & \implies \mathbb{E}\left[\frac{1}{\sum_{j=1}^{k} \alpha_j}\sum_{j=1}^{k} \alpha_j \|\nabla f(x_j)\|_2^2\right] \to 0.
\end{aligned}
$$

Stochastic gradient descent



- stochastic line search Paquette and Scheinberg [2020]

Goal

Consider **equality constrained** stochastic optimization problem:

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} \; f(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x};\xi)]$$
$$\text{s.t.} \; c(\boldsymbol{x}) = \boldsymbol{0}$$

Develop an **adaptive** stochastic procedure based on *sequential quadratic optimization*.

Related approaches

Consider equality constrained stochastic optimization problem:

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^d} \quad & f(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x};\xi)] \\ \text{s.t.} \quad & c(\boldsymbol{x}) = \mathbf{0} \end{aligned}$$

Penalty based methods

Ravi et al. [2019], Nandwani et al. [2019]

Projected stochastic first- and second-order methods

Nemirovski et al. [2009], Bertsekas [1982]

- projection to the null space may not be easily computed

Related approaches

Stochastic optimization **with** constraints

- ℓ_1-StoSQP – random projection used to select stepsize Berahas et al. [2021b]
    - fully stochastic setup
    - rank-deficient Jacobians Berahas et al. [2021a], inexactly solved Newton systems Curtis et al. [2021], SVRG acceleration Berahas et al.
- line-search StoSQP Na et al. [2022] , inequalities Na et al. [2021b]
    - random model setup

Stochastic optimization **without** constraints

- TRish, a fully stochastic trust-region method for unconstrained problems Curtis et al. [2019], Curtis and Shi [2020]
    - fully stochastic setup
- random model setup
    - stochastic line search Paquette and Scheinberg [2020]
    - trust region methods Bandeira et al. [2014], Chen et al. [2017], Blanchet et al. [2019]

Outline of the talk

SQP in a deterministic setting

Adaptive Trust Region Stochastic SQP

Extensions

Conclusion

Sequential quadratic programming (SQP)

Consider: $\min_{\boldsymbol{x} \in \mathbb{R}^d} \quad f(\boldsymbol{x}) \quad \text{subject to} \quad c(\boldsymbol{x}) = \mathbf{0}$

- the Lagrangian function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T c(\boldsymbol{x})$

Sequential quadratic programming (SQP)

Consider: $\min_{\boldsymbol{x}\in\mathbb{R}^d} \quad f(\boldsymbol{x}) \quad \text{subject to} \quad c(\boldsymbol{x}) = \mathbf{0}$

- the Lagrangian function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T c(\boldsymbol{x})$

SQP aims at finding a KKT point $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ that satisfies $(G \equiv \nabla^T c)$

$$\begin{pmatrix} \nabla_x \mathcal{L}(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) \\ \nabla_\lambda \mathcal{L}(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) \end{pmatrix} = \begin{pmatrix} \nabla f(\boldsymbol{x}^\star) + G^T(\boldsymbol{x}^\star)\boldsymbol{\lambda}^\star \\ c(\boldsymbol{x}^\star) \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix}$$

Sequential quadratic programming (SQP)

Consider: $\min_{\boldsymbol{x}\in\mathbb{R}^d} \; f(\boldsymbol{x}) \quad \text{subject to} \quad c(\boldsymbol{x}) = \boldsymbol{0}$

- the Lagrangian function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T c(\boldsymbol{x})$

SQP aims at finding a KKT point $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ that satisfies $(G \equiv \nabla^T c)$

$$\begin{pmatrix} \nabla_x \mathcal{L}(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) \end{pmatrix} = \begin{pmatrix} \nabla f(\boldsymbol{x}^\star) + G^T(\boldsymbol{x}^\star)\boldsymbol{\lambda}^\star \\ c(\boldsymbol{x}^\star) \end{pmatrix} = \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{pmatrix}$$

Alternative view point

$$\begin{aligned} \min_{\Delta \boldsymbol{x}\in\mathbb{R}^d} \quad & f(\boldsymbol{x}) + \nabla^T f(\boldsymbol{x})\Delta\boldsymbol{x} + \frac{1}{2}(\Delta\boldsymbol{x})^T \nabla_x^2 \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda})\Delta\boldsymbol{x} \\ \text{s.t.} \quad & c(\boldsymbol{x}) + G(\boldsymbol{x})\Delta\boldsymbol{x} = \boldsymbol{0} \end{aligned}$$

Sequential quadratic programming (SQP)

Consider: $\min_{\boldsymbol{x}\in\mathbb{R}^d} \; f(\boldsymbol{x}) \quad \text{subject to} \; c(\boldsymbol{x}) = \boldsymbol{0}$

- the Lagrangian function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T c(\boldsymbol{x})$

SQP aims at finding a KKT point $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ that satisfies $(G \equiv \nabla^T c)$

$$\begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) \end{pmatrix} = \begin{pmatrix} \nabla f(\boldsymbol{x}^\star) + G^T(\boldsymbol{x}^\star)\boldsymbol{\lambda}^\star \\ c(\boldsymbol{x}^\star) \end{pmatrix} = \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{pmatrix}$$

Alternative view point

$$\begin{aligned} \min_{\Delta\boldsymbol{x}\in\mathbb{R}^d} \quad & f(\boldsymbol{x}) + \nabla^T f(\boldsymbol{x})\Delta\boldsymbol{x} + \frac{1}{2}(\Delta\boldsymbol{x})^T \nabla_x^2 \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda})\Delta\boldsymbol{x} \\ \text{s.t.} \quad & c(\boldsymbol{x}) + G(\boldsymbol{x})\Delta\boldsymbol{x} = \boldsymbol{0} \end{aligned}$$

The resulting Newton system

$$\begin{pmatrix} B_k & G_k^T \\ G_k & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{\lambda}_k \end{pmatrix} = - \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}_k \\ c_k \end{pmatrix}$$

- B_k is an approximation of the Lagrangian Hessian $\nabla_{\boldsymbol{x}}^2 \mathcal{L}_k$
- $\Delta\boldsymbol{x}_k$ is the search direction

Trust-region sequential quadratic programming (TR-SQP)

$$\min_{\Delta \boldsymbol{x}_k \in \mathbb{R}^d} \quad \nabla^T f_k \Delta \boldsymbol{x}_k + \frac{1}{2}(\Delta \boldsymbol{x}_k)^T B_k \Delta \boldsymbol{x}_k$$
$$\text{s.t.} \quad c_k + G_k \Delta \boldsymbol{x}_k = \mathbf{0}, \quad \|\Delta \boldsymbol{x}_k\| \leqslant \Delta_k$$

- $\Delta \boldsymbol{x}_k$ is the trial step at $\boldsymbol{x}_k$; $\Delta_k > 0$ is the trust-region radius

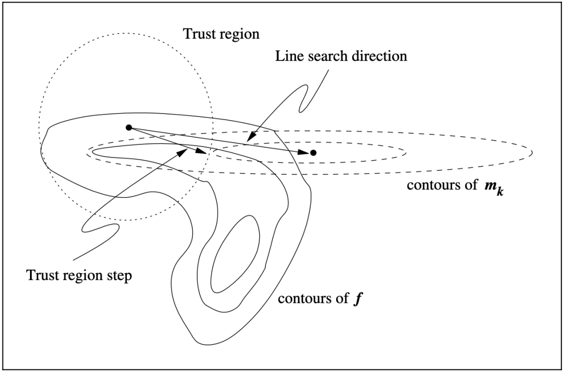


Figure from Nocedal and Wright [2006]

Why trust-region method?

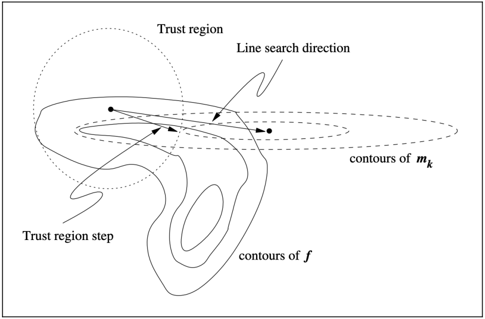


Figure from Nocedal and Wright [2006]

Computes the search direction and stepsize jointly

- can yield a more significant reduction in f than line search methods

Stronger ability to explore negative curvatures of the Hessian matrix

- Hessian modifications can be avoided

The trust-region constraint helps normalize steps

- more robust to ill-conditioning

Outline of the talk

SQP in a deterministic setting

Trust-Region Stochastic SQP

Extensions

Conclusion

Trust-region stochastic SQP (TR-StoSQP)

TR-StoSQP subproblem at $\boldsymbol{x}_k$

$$\begin{aligned}\min_{\Delta \boldsymbol{x}_k \in \mathbb{R}^d} \quad & \bar{g}_k \Delta \boldsymbol{x}_k + \frac{1}{2}(\Delta \boldsymbol{x}_k)^T B_k \Delta \boldsymbol{x}_k \\ \text{s.t.} \quad & c_k + G_k \Delta \boldsymbol{x}_k = \mathbf{0}, \quad \|\Delta \boldsymbol{x}_k\| \leqslant \Delta_k\end{aligned}$$

In the stochastic setting, $f(\boldsymbol{x})$, $\nabla f(\boldsymbol{x})$, $\nabla^2 f(\boldsymbol{x})$ need to be estimated

- $\bar{g}(\boldsymbol{x})$ is the estimate of $\nabla f(\boldsymbol{x})$
  - based on one observation in this talk (fully stochastic setting)
- overlined quantities represent estimates

B_k is an approximation of the Lagrangian Hessian $\nabla^2_{\boldsymbol{x}} \mathcal{L}_k$

- B_k is deterministic conditional on $\boldsymbol{x}_k$

Fully Stochastic Trust-Region Sequential Quadratic Programming for Equality-Constrained Optimization Problems
Yuchen Fang, Sen Na, Michael Mahoney, Mladen Kolar
SIAM Journal on Optimization
https://arxiv.org/abs/2211.15943

Trust-region methods for problems with constraints

Infeasibility of the subproblem

$$\{\Delta \boldsymbol{x}_k \in \mathbb{R}^d : c_k + G_k \Delta \boldsymbol{x}_k = \mathbf{0}\} \cap \{\Delta \boldsymbol{x}_k \in \mathbb{R}^d : \|\Delta \boldsymbol{x}_k\| \leqslant \Delta_k\} = \varnothing$$

- subproblem is unsolvable

Reason: trust-region radius is too short

- increasing the trust-region radius would violate the spirit of the method

Solution: relax the linearized constraints

How to relax linearized constriants?

Replace linearized constraints by inequalities

Celis et al. [1985], Powell and Yuan [1990]

- $\|c_k + G_k \Delta \boldsymbol{x}_k\| \leqslant \theta_k$ for some $\theta_k > 0$
- TR-SQP subproblems are hard to solve due to inequalities

Maintain equality constraints but conduct a step decomposition

Vardi [1985], Byrd et al. [1987], Omojokun [1989]

- no clear guidance for choosing involved user-specified parameters

We propose an **adaptive relaxation technique**

- extends the method in Byrd et al. [1987]
- adaptive *without* the need for user to specify parameters

Adaptive relaxation technique

The trial step $\Delta \boldsymbol{x}_k$ is decomposed as $\Delta \boldsymbol{x}_k = \boldsymbol{w}_k + \boldsymbol{t}_k$

- the normal step $\boldsymbol{w}_k \in \mathrm{im}(G_k^T)$
- the tangential step $\boldsymbol{t}_k \in \ker(G_k)$

Adaptive relaxation technique

The trial step $\Delta \boldsymbol{x}_k$ is decomposed as $\Delta \boldsymbol{x}_k = \boldsymbol{w}_k + \boldsymbol{t}_k$

- the normal step $\boldsymbol{w}_k \in \mathrm{im}(G_k^T)$
- the tangential step $\boldsymbol{t}_k \in \ker(G_k)$

Normal step

$\boldsymbol{v}_k = -G_k^T [G_k G_k^T]^{-1} c_k$

- solves $c_k + G_k \boldsymbol{v}_k = \mathbf{0}$ without the trust-region constraint
- the trust-region may prevent us from setting $\boldsymbol{w}_k = \boldsymbol{v}_k$

Adaptive relaxation technique

The trial step $\Delta \boldsymbol{x}_k$ is decomposed as $\Delta \boldsymbol{x}_k = \boldsymbol{w}_k + \boldsymbol{t}_k$

- the normal step $\boldsymbol{w}_k \in \mathrm{im}(G_k^T)$
- the tangential step $\boldsymbol{t}_k \in \ker(G_k)$

Normal step

$\boldsymbol{v}_k = -G_k^T [G_k G_k^T]^{-1} c_k$

- solves $c_k + G_k \boldsymbol{v}_k = \mathbf{0}$ without the trust-region constraint
- the trust-region may prevent us from setting $\boldsymbol{w}_k = \boldsymbol{v}_k$
- $\boldsymbol{w}_k = \gamma_k \boldsymbol{v}_k$ for a scalar $\gamma_k \in (0, 1]$

Adaptive relaxation technique

The trial step $\Delta \boldsymbol{x}_k$ is decomposed as $\Delta \boldsymbol{x}_k = \boldsymbol{w}_k + \boldsymbol{t}_k$

- the normal step $\boldsymbol{w}_k \in \text{im}(G_k^T)$
- the tangential step $\boldsymbol{t}_k \in \ker(G_k)$

Normal step

$\boldsymbol{v}_k = -G_k^T [G_k G_k^T]^{-1} c_k$

- solves $c_k + G_k \boldsymbol{v}_k = \mathbf{0}$ without the trust-region constraint
- the trust-region may prevent us from setting $\boldsymbol{w}_k = \boldsymbol{v}_k$
- $\boldsymbol{w}_k = \gamma_k \boldsymbol{v}_k$ for a scalar $\gamma_k \in (0, 1]$

Tangential step

$\boldsymbol{t}_k = P_k \boldsymbol{u}_k$ for some vector $\boldsymbol{u}_k \in \mathbb{R}^d$

- $P_k = \boldsymbol{I} - G_k^T [G_k G_k^T]^{-1} G_k$ is the projection matrix to the null space of G_k

Adaptive relaxation technique

The trial step $\Delta \boldsymbol{x}_k = \gamma_k \boldsymbol{v}_k + P_k \boldsymbol{u}_k$

How to chose γ_k and $\boldsymbol{u}_k$ so that $\|\Delta \boldsymbol{x}_k\| \leqslant \Delta_k$?

Adaptive relaxation technique

The trial step $\Delta \boldsymbol{x}_k = \gamma_k \boldsymbol{v}_k + P_k \boldsymbol{u}_k$

How to chose γ_k and $\boldsymbol{u}_k$ so that $\|\Delta \boldsymbol{x}_k\| \leqslant \Delta_k$?

Adaptively decompose the trust-region radius into two segments

$$\breve{\Delta}_k = \frac{\|c_k\|}{\|\bar{\nabla}\mathcal{L}_k\|}\Delta_k \qquad \text{and} \qquad \widetilde{\Delta}_k = \frac{\|\bar{\nabla}_{\boldsymbol{x}}\mathcal{L}_k\|}{\|\bar{\nabla}\mathcal{L}_k\|}\Delta_k$$

Adaptive relaxation technique

The trial step $\Delta \boldsymbol{x}_k = \gamma_k \boldsymbol{v}_k + P_k \boldsymbol{u}_k$

How to chose γ_k and $\boldsymbol{u}_k$ so that $\|\Delta \boldsymbol{x}_k\| \leqslant \Delta_k$?

Adaptively decompose the trust-region radius into two segments

$$\breve{\Delta}_k = \frac{\|c_k\|}{\|\bar{\nabla}\mathcal{L}_k\|}\Delta_k \qquad \text{and} \qquad \widetilde{\Delta}_k = \frac{\|\bar{\nabla}_{\boldsymbol{x}}\mathcal{L}_k\|}{\|\bar{\nabla}\mathcal{L}_k\|}\Delta_k$$

Choose γ_k: $\gamma_k = \min\left\{\breve{\Delta}_k/\|\boldsymbol{v}_k\|, 1\right\}$

Adaptive relaxation technique

The trial step $\Delta \boldsymbol{x}_k = \gamma_k \boldsymbol{v}_k + P_k \boldsymbol{u}_k$

How to chose γ_k and $\boldsymbol{u}_k$ so that $\|\Delta \boldsymbol{x}_k\| \leqslant \Delta_k$?

Adaptively decompose the trust-region radius into two segments

$$\breve{\Delta}_k = \frac{\|c_k\|}{\|\bar{\nabla}\mathcal{L}_k\|}\Delta_k \qquad \text{and} \qquad \widetilde{\Delta}_k = \frac{\|\bar{\nabla}_{\boldsymbol{x}}\mathcal{L}_k\|}{\|\bar{\nabla}\mathcal{L}_k\|}\Delta_k$$

Choose γ_k: $\gamma_k = \min\left\{\breve{\Delta}_k / \|\boldsymbol{v}_k\|, 1\right\}$

Compute $\boldsymbol{u}_k$: Approximately solve

$$\min_{\boldsymbol{u}_k \in \mathbb{R}^d} \quad m(\boldsymbol{u}_k) = \bar{g}_k^T P_k \boldsymbol{u}_k + \frac{1}{2}\boldsymbol{u}_k^T P_k B_k P_k \boldsymbol{u}_k \quad \text{s.t.} \quad \|\boldsymbol{u}_k\| \leqslant \widetilde{\Delta}_k$$

- needs to satisfy the Cauchy reduction

Fully stochastic trust-region SQP (TR-StoSQP)

Input:

- sequence $\{\beta_k\}_k \subseteq (0, 1]$ — related to trust-region radius
- $\zeta > 0$ — controls the control parameters
- μ_{-1} — the initial merit parameter
- $\rho > 1$ — controls the merit parameter update

Algorithm: Until convergence, repeat:

1. Generate control parameters
2. Estimate the gradient and generate the trust-region radius
3. Compute the trial step and update the merit parameter

Fully stochastic trust-region SQP (TR-StoSQP)

Step 1: Generate control parameters

- Generate a deterministic matrix B_k (conditional on $\boldsymbol{x}_k$)
  - approximation to the Hessian of the Lagrangian $\nabla^2_{\boldsymbol{x}}\mathcal{L}_k$

Fully stochastic trust-region SQP (TR-StoSQP)

Step 1: Generate control parameters

- Generate a deterministic matrix B_k (conditional on $\boldsymbol{x}_k$)
  - approximation to the Hessian of the Lagrangian $\nabla^2_{\boldsymbol{x}}\mathcal{L}_k$
- Control parameters:
  - $\eta_{1,k} = \zeta \min\left\{\frac{1}{\|B_k\|}, \frac{6}{\|G_k\|}\right\}$
  - $\tau_k = L_{\nabla f,k} + L_{G,k}\bar{\mu}_{k-1} + \|B_k\|$
  - $\alpha_k = \frac{\beta_k}{4\eta_{1,k}\tau_k + 6\zeta}$
  - $\eta_{2,k} = \eta_{1,k} - \frac{1}{2}\zeta\eta_{1,k}\alpha_k$

Fully stochastic trust-region SQP (TR-StoSQP)

Step 1: Generate control parameters

- Generate a deterministic matrix B_k (conditional on $\boldsymbol{x}_k$)
  - approximation to the Hessian of the Lagrangian $\nabla_{\boldsymbol{x}}^2 \mathcal{L}_k$
- Control parameters:
  - $\eta_{1,k} = \zeta \min\left\{\frac{1}{\|B_k\|}, \frac{6}{\|G_k\|}\right\}$
  - $\tau_k = L_{\nabla f,k} + L_{G,k}\bar{\mu}_{k-1} + \|B_k\|$
  - $\alpha_k = \frac{\beta_k}{4\eta_{1,k}\tau_k + 6\zeta}$
  - $\eta_{2,k} = \eta_{1,k} - \frac{1}{2}\zeta\eta_{1,k}\alpha_k$

Note: $L_{\nabla f,k}, L_{G,k}$ are (estimated) Lipschitz constants of $\nabla f(\boldsymbol{x}), G(\boldsymbol{x})$ at $\boldsymbol{x}_k$

- can be estimated Curtis and Robinson
- can be replaced by universal quantities $L_{\nabla f}, L_G$ such that $L_{\nabla f,k} \leqslant L_{\nabla f}, L_{G,k} \leqslant L_G$

Fully stochastic trust-region SQP (TR-StoSQP)

Step 2: Estimate the gradient and generate the trust-region radius

- Estimate gradient $\bar{g}_k = \nabla F(\boldsymbol{x}_k; \xi)$ and compute $\bar{\nabla}\mathcal{L}_k$

Fully stochastic trust-region SQP (TR-StoSQP)

Step 2: Estimate the gradient and generate the trust-region radius

- Estimate gradient $\bar{g}_k = \nabla F(\boldsymbol{x}_k; \xi)$ and compute $\bar{\nabla}\mathcal{L}_k$
- Compute the Lagrangian multiplier $\bar{\boldsymbol{\lambda}}_k = -[G_k G_k^T]^{-1} G_k \bar{g}_k$
- Compute the trust-region radius is generated

$$\Delta_k = \begin{cases} \eta_{1,k}\alpha_k \|\bar{\nabla}\mathcal{L}_k\| & \text{if } \|\bar{\nabla}\mathcal{L}_k\| \in (0, 1/\eta_{1,k}) \\ \alpha_k & \text{if } \|\bar{\nabla}\mathcal{L}_k\| \in [1/\eta_{1,k}, 1/\eta_{2,k}] \\ \eta_{2,k}\alpha_k \|\bar{\nabla}\mathcal{L}_k\| & \text{if } \|\bar{\nabla}\mathcal{L}_k\| \in (1/\eta_{2,k}, \infty). \end{cases}$$

Fully stochastic trust-region SQP (TR-StoSQP)

Step 2: Estimate the gradient and generate the trust-region radius

- Estimate gradient $\bar{g}_k = \nabla F(\boldsymbol{x}_k; \xi)$ and compute $\bar{\nabla}\mathcal{L}_k$
- Compute the Lagrangian multiplier $\bar{\boldsymbol{\lambda}}_k = -[G_k G_k^T]^{-1} G_k \bar{g}_k$
- Compute the trust-region radius is generated

$$\Delta_k = \begin{cases} \eta_{1,k}\alpha_k \|\bar{\nabla}\mathcal{L}_k\| & \text{if } \|\bar{\nabla}\mathcal{L}_k\| \in (0, 1/\eta_{1,k}) \\ \alpha_k & \text{if } \|\bar{\nabla}\mathcal{L}_k\| \in [1/\eta_{1,k}, 1/\eta_{2,k}] \\ \eta_{2,k}\alpha_k \|\bar{\nabla}\mathcal{L}_k\| & \text{if } \|\bar{\nabla}\mathcal{L}_k\| \in (1/\eta_{2,k}, \infty). \end{cases}$$

Note:

- When $\|\bar{\nabla}\mathcal{L}_k\|$ is small, $\Delta_k < \alpha_k$
  - close to a first-order stationary point; require careful steps
  - **different** from deterministic setting – trust-region constraint is inactive when iterates are close to a stationary point to maintain a fast convergence rate
  - ensure that the stationary point is not skipped due to errors in estimation

Fully stochastic trust-region SQP (TR-StoSQP)

Step 2: Estimate the gradient and generate the trust-region radius

- Estimate gradient $\bar{g}_k = \nabla F(\boldsymbol{x}_k; \xi)$ and compute $\bar{\nabla}\mathcal{L}_k$
- Compute the Lagrangian multiplier $\bar{\boldsymbol{\lambda}}_k = -[G_k G_k^T]^{-1} G_k \bar{g}_k$
- Compute the trust-region radius is generated

$$\Delta_k = \begin{cases} \eta_{1,k}\alpha_k \|\bar{\nabla}\mathcal{L}_k\| & \text{if } \|\bar{\nabla}\mathcal{L}_k\| \in (0, 1/\eta_{1,k}) \\ \alpha_k & \text{if } \|\bar{\nabla}\mathcal{L}_k\| \in [1/\eta_{1,k}, 1/\eta_{2,k}] \\ \eta_{2,k}\alpha_k \|\bar{\nabla}\mathcal{L}_k\| & \text{if } \|\bar{\nabla}\mathcal{L}_k\| \in (1/\eta_{2,k}, \infty). \end{cases}$$

Note:

- When $\|\bar{\nabla}\mathcal{L}_k\|$ is small, $\Delta_k < \alpha_k$
  - close to a first-order stationary point; require careful steps
  - **different** from deterministic setting – trust-region constraint is inactive when iterates are close to a stationary point to maintain a fast convergence rate
  - ensure that the stationary point is not skipped due to errors in estimation
- When $\|\bar{\nabla}\mathcal{L}_k\|$ is large, $\Delta_k > \alpha_k$
  - far from a stationary point – allow for larger improvement

Fully stochastic trust-region SQP (TR-StoSQP)

Step 3: Compute the trial step and update the merit parameter

- Compute the trial step using the adaptive relaxation technique

$$\Delta \boldsymbol{x}_k = \gamma_k \boldsymbol{v}_k + P_k \boldsymbol{u}_k$$

Fully stochastic trust-region SQP (TR-StoSQP)

Step 3: Compute the trial step and update the merit parameter

- Compute the trial step using the adaptive relaxation technique

$$\Delta \boldsymbol{x}_k = \gamma_k \boldsymbol{v}_k + P_k \boldsymbol{u}_k$$

- Update the iterate $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta \boldsymbol{x}_k$

Fully stochastic trust-region SQP (TR-StoSQP)

Step 3: Compute the trial step and update the merit parameter

- Compute the trial step using the adaptive relaxation technique

$$\Delta \boldsymbol{x}_k = \gamma_k \boldsymbol{v}_k + P_k \boldsymbol{u}_k$$

- Update the iterate $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta \boldsymbol{x}_k$

- Set $\bar{\mu}_k = \bar{\mu}_{k-1}$. Compute the estimated predicted reduction

$$\text{Pred}_k = \bar{g}_k^T \Delta \boldsymbol{x}_k + \frac{1}{2} \Delta \boldsymbol{x}_k^T B_k \Delta \boldsymbol{x}_k + \bar{\mu}_k (\|c_k + G_k \Delta \boldsymbol{x}_k\| - \|c_k\|)$$

  Update $\bar{\mu}_k \longleftarrow \rho \bar{\mu}_k$ until

$$\text{Pred}_k \leqslant -\|\bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k\| \widetilde{\Delta}_k - \frac{1}{2}\|c_k\| \breve{\Delta}_k + \frac{1}{2}\|B_k\| \widetilde{\Delta}_k^2 + \|B_k\| \breve{\Delta}_k \widetilde{\Delta}_k$$

Fully stochastic trust-region SQP (TR-StoSQP)

Step 3: Compute the trial step and update the merit parameter

- Compute the trial step using the adaptive relaxation technique

$$\Delta \boldsymbol{x}_k = \gamma_k \boldsymbol{v}_k + P_k \boldsymbol{u}_k$$

- Update the iterate $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta \boldsymbol{x}_k$

- Set $\bar{\mu}_k = \bar{\mu}_{k-1}$. Compute the estimated predicted reduction

$$\text{Pred}_k = \bar{g}_k^T \Delta \boldsymbol{x}_k + \frac{1}{2} \Delta \boldsymbol{x}_k^T B_k \Delta \boldsymbol{x}_k + \bar{\mu}_k (\|c_k + G_k \Delta \boldsymbol{x}_k\| - \|c_k\|)$$

  Update $\bar{\mu}_k \longleftarrow \rho \bar{\mu}_k$ until

$$\text{Pred}_k \leqslant -\|\bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k\| \widetilde{\Delta}_k - \frac{1}{2}\|c_k\| \breve{\Delta}_k + \frac{1}{2}\|B_k\| \widetilde{\Delta}_k^2 + \|B_k\| \breve{\Delta}_k \widetilde{\Delta}_k$$

Note:

- iterates are always updated
- the merit parameter is used for generating the trust-region radius

Discussion of TR-StoSQP

Trust-region radius is constructed based on

- input $\{\beta_k\} \in (0, 1]$
- control parameters $\{\eta_{1,k}\}, \{\eta_{2,k}\}, \{\tau_k\}$
- current KKT residual $\|\bar{\nabla}\mathcal{L}_k\|$

Control parameters are **automatically** computed in each iteration of TR-StoSQP

Discussion of TR-StoSQP

Trust-region radius is constructed based on

- input $\{\beta_k\} \in (0, 1]$
- control parameters $\{\eta_{1,k}\}, \{\eta_{2,k}\}, \{\tau_k\}$
- current KKT residual $\|\bar{\nabla}\mathcal{L}_k\|$

Control parameters are **automatically** computed in each iteration of TR-StoSQP

Compared with Curtis and Shi [2020]

- upper bound on β_k is simplified to be 1
- TR-StoSQP does not require $\{\eta_{1,k}\}, \{\eta_{2,k}\}$ as input
- growth condition on the gradient estimate
- for a decaying $\{\beta_k\}$, we do not require the gradient error to decay

Discussion of TR-StoSQP

Trust-region radius is constructed based on

- input $\{\beta_k\} \in (0, 1]$
- control parameters $\{\eta_{1,k}\}, \{\eta_{2,k}\}, \{\tau_k\}$
- current KKT residual $\|\bar{\nabla}\mathcal{L}_k\|$

Control parameters are **automatically** computed in each iteration of TR-StoSQP

Compared with Curtis and Shi [2020]

- upper bound on β_k is simplified to be 1
- TR-StoSQP does not require $\{\eta_{1,k}\}, \{\eta_{2,k}\}$ as input
- growth condition on the gradient estimate
- for a decaying $\{\beta_k\}$, we do not require the gradient error to decay

An ℓ_2 merit function is used to balance objective value and constraint violation

$$\mathcal{L}_{\bar{\mu}}(\boldsymbol{x}) = f(\boldsymbol{x}) - f_{\text{inf}} + \bar{\mu}\|c(\boldsymbol{x})\|,$$

- the merit function is not explicitly used in the algorithm
- the stochastic merit parameter is adaptively chosen in each iteration

Convergence theory

Assumption:

- the iterates $\boldsymbol{x}_k$ lie in some open convex set Ω
- f and c are continuously differentiable; f is bounded below by f_{inf}
- ∇f and the Jacobian $G(\boldsymbol{x})$ are Lipschitz continuous
- $\|B_k\| \leqslant \kappa_B$, $\|c_k\| \leqslant \kappa_c$, $\|\nabla f_k\| \leqslant \kappa_{\nabla f}$, $\kappa_{1,G} \cdot I \preceq G_k G_k^T \preceq \kappa_{2,G} \cdot I$

Convergence theory

Assumption:

- the iterates $\boldsymbol{x}_k$ lie in some open convex set Ω
- f and c are continuously differentiable; f is bounded below by f_{inf}
- ∇f and the Jacobian $G(\boldsymbol{x})$ are Lipschitz continuous
- $\|B_k\| \leqslant \kappa_B$, $\|c_k\| \leqslant \kappa_c$, $\|\nabla f_k\| \leqslant \kappa_{\nabla f}$, $\kappa_{1,G} \cdot I \preceq G_k G_k^T \preceq \kappa_{2,G} \cdot I$

Assumption:

There exists a stochastic $\bar{K} < \infty$ and a deterministic constant $\widehat{\mu}$, such that for all $k > \bar{K}$, $\bar{\mu}_k = \bar{\mu}_{\bar{K}} \leqslant \widehat{\mu}$.

- this assumption is satisfied if $\bar{g}_k$ is bounded

Convergence theory

Assumption:

- the iterates $\boldsymbol{x}_k$ lie in some open convex set Ω
- f and c are continuously differentiable; f is bounded below by $f_{\inf}$
- ∇f and the Jacobian $G(\boldsymbol{x})$ are Lipschitz continuous
- $\|B_k\| \leqslant \kappa_B$, $\|c_k\| \leqslant \kappa_c$, $\|\nabla f_k\| \leqslant \kappa_{\nabla f}$, $\kappa_{1,G} \cdot I \preceq G_k G_k^T \preceq \kappa_{2,G} \cdot I$

Assumption:

There exists a stochastic $\bar{K} < \infty$ and a deterministic constant $\widehat{\mu}$, such that for all $k > \bar{K}$, $\bar{\mu}_k = \bar{\mu}_{\bar{K}} \leqslant \widehat{\mu}$.

- this assumption is satisfied if $\bar{g}_k$ is bounded

Assumption:

The estimate $\bar{g}_k$ is an unbiased estimator of ∇f_k, $\mathbb{E}[\bar{g}_k \mid \boldsymbol{x}_k] = \nabla f_k$.
There exist constants $M_g \geqslant 1, M_{g,1} \geqslant 0$ such that

$$\mathbb{E}[\|\nabla f_k - \bar{g}_k\|^2 \mid \boldsymbol{x}_k] \leqslant M_g + M_{g,1}(f_k - f_{\inf}).$$

Global convergence

Theorem: Suppose that $\beta_k = \beta$. If $M_{g,1} = 0$, then

$$\frac{1}{K}\sum_{k=\bar{K}+1}^{\bar{K}+K}\mathbb{E}[\|\nabla\mathcal{L}_k\|^2] \leqslant \frac{4}{\eta_{\min}\alpha_l\beta K}\mathcal{L}_{\bar{\mu}_{\bar{K}}}(\boldsymbol{x}_{\bar{K}+1}) + \frac{4\Upsilon_1 M_g}{\eta_{\min}\alpha_l}\beta \xrightarrow{K\to\infty} \frac{4\Upsilon_1 M_g}{\eta_{\min}\alpha_l}\beta$$

Global convergence

Theorem: Suppose that $\beta_k = \beta$. If $M_{g,1} = 0$, then

$$\frac{1}{K}\sum_{k=\bar{K}+1}^{\bar{K}+K} \mathbb{E}[\|\nabla \mathcal{L}_k\|^2] \leqslant \frac{4}{\eta_{\min}\alpha_l\beta K}\mathcal{L}_{\bar{\mu}_{\bar{K}}}(\boldsymbol{x}_{\bar{K}+1}) + \frac{4\Upsilon_1 M_g}{\eta_{\min}\alpha_l}\beta \xrightarrow{K\to\infty} \frac{4\Upsilon_1 M_g}{\eta_{\min}\alpha_l}\beta$$

Theorem: Suppose that $\sum_{k=0}^{\infty}\beta_k = \infty$ and $\sum_{k=0}^{\infty}\beta_k^2 < \infty$. Then

$$\mathbb{E}\left[\sum_{k=\bar{K}+1}^{\infty}\beta_k\|\nabla \mathcal{L}_k\|^2\right] < \infty \quad \text{and} \quad \lim_{K\to\infty}\frac{1}{\sum_{k=\bar{K}+1}^{\bar{K}+K}\beta_k}\sum_{k=\bar{K}+1}^{\bar{K}+K}\beta_k\mathbb{E}\left[\|\nabla \mathcal{L}_k\|^2\right] = 0.$$

In addition,

$$\lim_{k\to\infty}\|\nabla \mathcal{L}_k\| = 0 \quad \text{almost surely.}$$

Empirical setup

ℓ_1-StoSQP (Berahas et al. 2021)

- $\bar{\tau}_{-1} = 1, \epsilon = 10^{-6}, \sigma = 0.5, \bar{\xi}_{-1} = 1, \theta = 10$

TR-StoSQP

- $\zeta = 10, \bar{\mu}_{-1} = 1, \rho = 1.5$

Hessian approximation

- Identity matrix (Id)
- Symmetric rank-one (SR1) update
- Estimated Hessian (EstH)
- Averaged Hessian (AveH)

Choices for β_k

- Two constant $\beta_k \in \{0.5, 1\}$
- Two decaying $\beta_k \in \{k^{-0.6}, k^{-0.8}\}$

CUTEst test set

Constrained nonlinear optimization problems

- `BT4, BT5, BT8, BT9, MARATOS, HS39, HS40, HS42, HS78, HS79`
- Singularity of $G_k G_k^T$ is not reported for all iterations
- The initialization of primal-dual variables is given by CUTEst package

Stochastic oracle

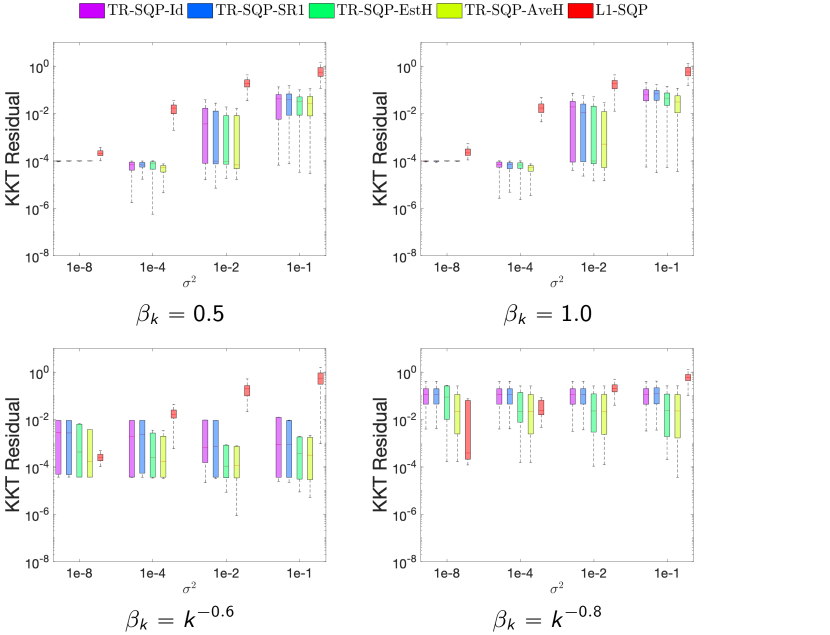
- the estimator of ∇f_k is drawn from $\mathcal{N}(\nabla f_k, \sigma^2(\boldsymbol{I} + \mathbf{1}\mathbf{1}^T))$
- the estimator of $(\nabla^2 f_k)_{i,j}$ is drawn from $\mathcal{N}((\nabla^2 f_k)_{i,j}, \sigma^2)$
- $\sigma^2 \in \{10^{-8}, 10^{-4}, 10^{-2}, 10^{-1}\}$.

The stopping criterion

$$\|\nabla \mathcal{L}_k\| \leqslant 10^{-4} \quad \text{OR} \quad k \geqslant 10^5$$

We perform 5 independent runs

KKT residuals



$\beta_k = 0.5$ $\beta_k = 1.0$

$\beta_k = k^{-0.6}$ $\beta_k = k^{-0.8}$

Constrained logistic regression

$$\min_{x \in \mathbb{R}^d} f(\boldsymbol{x}) = \frac{1}{N} \sum_{i=1}^{N} \log \left(1 + e^{-y_i (\boldsymbol{z}_i^T \boldsymbol{x})} \right) \quad \text{s.t.} \quad A\boldsymbol{x} = \boldsymbol{b}$$

LIBSVM collection

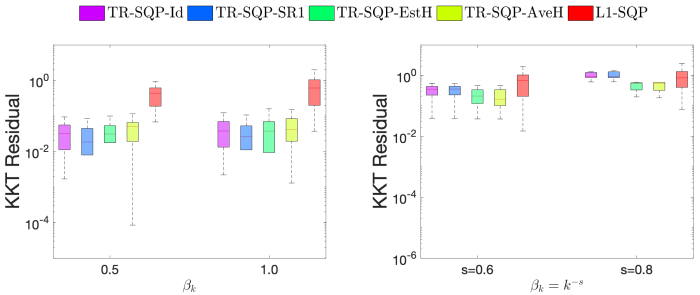
- `austrilian`, `breast-cancer`, `diabetes`, `heart`, `ionosphere`, `sonar`, `splice`,`svmguide3`
- The initial iterate is set as all one vector
- One sample is selected from the given N samples in each iteration
- $A \in \mathbb{R}^{5 \times d}$ and $b \in \mathbb{R}^5$
  - each entry follows a standard normal distribution
  - A has full row rank in all problems

Stopping criterion

$$\|\nabla \mathcal{L}_k\| \leqslant 10^{-4} \quad \text{OR} \quad \text{20 epochs}$$

We perform 5 independent runs

KKT residuals



constant β_k decaying β_k

Outline of the talk

SQP in a deterministic setting

Adaptive Trust Region Stochastic SQP

Extensions

Conclusion

Inequality constraints

Consider stochastic nonlinear optimization problem:

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} \quad f(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x};\xi)]$$
$$\text{s.t.} \quad c_{\mathcal{E}}(\boldsymbol{x}) = \mathbf{0}$$
$$c_{\mathcal{I}}(\boldsymbol{x}) \leqslant \mathbf{0}$$

Additional challenges:

- inequality constrained (nonconvex) quadratic programs
- SQP generates a descent direction of augmented Lagrangian only in a neighborhood of a KKT point

Proposed solutions:

- active-set SQP framework
  - subproblem is an equality constrained QP
- the scheme uses a backup search direction
  - use SQP direction if it provides a descent direction
  - use a regularized Newton step or a steepest descent step of augmented Lagrangian, otherwise

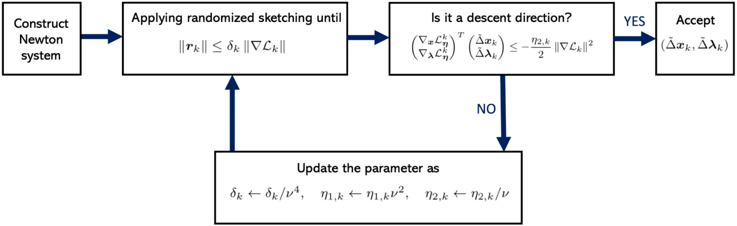
Inequality Constrained Stochastic Nonlinear Optimization via Active-Set Sequential Quadratic Programming
Mathematical Programming
https://arxiv.org/abs/2109.11502

Randomized solvers for the Newton system

Solving the Newton system impractical for large scale problems

Solve the Newton system *inexactly* via *randomized sketching*

$$\underbrace{\begin{pmatrix} S_1 \\ S_2 \end{pmatrix}^T}_{d \times (n+m)} \underbrace{\begin{pmatrix} B_k & G_k^T \\ G_k & \mathbf{0} \end{pmatrix}}_{(n+m)\times(n+m)} \begin{pmatrix} \widetilde{\Delta} \boldsymbol{x}_k \\ \widetilde{\Delta} \boldsymbol{\lambda}_k \end{pmatrix} = \underbrace{-\begin{pmatrix} S_1 \\ S_2 \end{pmatrix}^T \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}_k \\ c_k \end{pmatrix}}_{d \times 1}$$



Main results

- Almost sure global convergence guarantee.
- Almost sure local linear convergence guarantee.

Constrained Optimization via Exact Augmented Lagrangian and Randomized Iterative Sketching
Ilgee Hong, Sen Na, Michael Mahoney, Mladen Kolar
ICML 2023

Outline of the talk

Conclusion

Consider **equality constrained** stochastic optimization problem:

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} \; f(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x};\xi)]$$
$$\text{s.t.} \;\; c(\boldsymbol{x}) = \boldsymbol{0}$$

- *Adaptive* stochastic SQP method
    - trust-region for fully stochastic setting
- Almost sure global convergence
- Exciting numerical results

Future work

- local convergence analysis
- sample complexity analysis
- finding second order stationary points

- distributed optimization (federated learning) with constraints
- safe RL

- statistical inference

Thank you!

An Adaptive Stochastic Sequential Quadratic Programming with Differentiable Exact Augmented Lagrangians
https://arxiv.org/abs/2102.05320

Inequality Constrained Stochastic Nonlinear Optimization via Active-Set Sequential Quadratic Programming
https://arxiv.org/abs/2109.11502

Fully Stochastic Trust-Region Sequential Quadratic Programming for Equality-Constrained Optimization Problems
https://arxiv.org/abs/2211.15943

Constrained Optimization via Exact Augmented Lagrangian and Randomized Iterative Sketching
https://arxiv.org/abs/2305.18379

ℓ_1 penalized AdapSQP

The ℓ_1 penalized merit function

$$\mathcal{L}_\mu(\boldsymbol{x}) = f(\boldsymbol{x}) + \mu\|c(\boldsymbol{x})\|_1.$$

The condition in the first step

$$\mathcal{A}_k = \left\{\|\bar{g}_k - \nabla f_k\| \leqslant \kappa_{grad} \cdot \bar{\alpha}_k \|\bar{\nabla}\mathcal{L}_k\|\right\}$$

The search direction $(\bar{\Delta}\boldsymbol{x}_k, \bar{\Delta}\boldsymbol{\lambda}_k)$ is obtained by solving

$$\begin{pmatrix} B_k & G_k^T \\ G_k & \mathbf{0} \end{pmatrix} \begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ \bar{\Delta}\boldsymbol{\lambda}_k \end{pmatrix} = -\begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}}\mathcal{L}_k \\ c_k \end{pmatrix}$$

The penalty parameter is updated as $\bar{\mu}_k = \bar{g}_k^T \bar{\Delta}\boldsymbol{x}_k / \{(\rho - 1)\|c_k\|_1\}$

The condition in the third step

$$\mathcal{B}_k = \left\{|\bar{\mathcal{L}}_{\bar{\mu}_k}^k - \mathcal{L}_{\bar{\mu}_k}^k| \vee |\bar{\mathcal{L}}_{\bar{\mu}_k}^{s_k} - \mathcal{L}_{\bar{\mu}_k}^{s_k}| \leqslant -\kappa_f \alpha_k^2 \left(\bar{g}_k^T \bar{\Delta}\boldsymbol{x}_k - \bar{\mu}_k\|c_k\|_1\right)\right\}$$

Implementation details (I)

ℓ_1 **SQP** in Berahas et al. [2021b]

- $\bar{\tau}_{-1} = 1$, $\epsilon = 10^{-6}$, $\sigma = 0.5$, $\bar{\xi}_{-1} = 1$, $\theta = 10$
- the Lipschitz constant is estimated around the initialization
- the stepsize related sequence $\{\beta_k\}_k$
  - constant case: $\beta_k = \{0.01, 0.1, 0.5, 1\}$
  - decaying case: $\beta_k = \{1/k^{0.6}, 1/k^{0.9}\}$

non-adaptive stochastic SQP

- setup as above
- the stepsize sequence
  - $\alpha_k = \{0.01, 0.1, 0.5, 1\}$
  - $\alpha_k = \{1/k^{0.6}, 1/k^{0.9}\}$

Implementation details (II)

adaptive stochastic SQP

- $\nu = 0.001$ — make $\mathcal{L}_{\mu,\nu}$ similar to standard augmented Lagrangian
- $\bar{\alpha}_0 = \alpha_{max} = 1.5$ — the selected stepsize may be greater than 1
- $\bar{\mu}_0 = \bar{\epsilon}_0 = 1$
- $\kappa_{grad} = 1$, $p_{grad} = p_f = 0.1$, $\kappa_f = \beta/(4\alpha_{max}) = 0.05$
- $C_{grad} = C_f = \{1, 5, 10, 50\}$
- $\rho = 1.2$
- $\beta = 0.3$ — a (nearly) middle value of interval $(0, 0.5)$
    - a fast local rate in deterministic case Lucidi [1990]

adaptive stochastic SQP

- same as above, but without ν

Comment on Adaptive SQP

Note that the dual search direction is $\Delta\boldsymbol{\lambda}_k$.

- if $B_k \approx \nabla^2_{\boldsymbol{x}}\mathcal{L}_k$ and $(\boldsymbol{x}_k, \boldsymbol{\lambda}_k)$ is close to a KKT point, then $\Delta\boldsymbol{\lambda}_k \approx \widehat{\Delta}\boldsymbol{\lambda}_k$
- if an iterate is far from a KKT point or $B_k \not\approx \nabla^2_{\boldsymbol{x}}\mathcal{L}_k$, $\Delta\boldsymbol{\lambda}_k$ and $\widehat{\Delta}\boldsymbol{\lambda}_k$ are significantly different

We want a dual direction and $\Delta\boldsymbol{x}_k$ to be a descent direction of $\mathcal{L}^k_{\mu,\nu}$, $\nu > 0$, $\mu > 0$ sufficiently large

- sufficient condition: $\lim_{\alpha\to 0} G_k G_k^T \Delta\boldsymbol{\lambda}_k(\alpha) = -(G_k \nabla_{\boldsymbol{x}}\mathcal{L}_k + M_k^T \Delta\boldsymbol{x}_k)$

Non-adaptive stochastic SQP

Non-adaptive stochastic SQP

Notation:

$$\bar{g}_k = \nabla f(\boldsymbol{x}_k; \xi_g^k), \quad \bar{\nabla}_{\boldsymbol{x}}\mathcal{L}_k = \bar{g}_k + G_k^T \boldsymbol{\lambda}_k,$$

$$\bar{H}_k = \nabla^2 f(\boldsymbol{x}_k; \xi_H^k), \quad \bar{\nabla}_{\boldsymbol{x}}^2\mathcal{L}_k = \bar{H}_k + \sum_{j=1}^{m} (\boldsymbol{\lambda}_k)_j \nabla^2 c_j(\boldsymbol{x}_k),$$

$$\bar{T}_k = \left(\nabla^2 c_1(\boldsymbol{x}_k)\bar{\nabla}_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x}_k, \boldsymbol{\lambda}_k; \xi_H^k), \cdots, \nabla^2 c_m(\boldsymbol{x}_k)\bar{\nabla}_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x}_k, \boldsymbol{\lambda}_k; \xi_H^k)\right),$$

$$\bar{M}_k = \bar{\nabla}_{\boldsymbol{x}}^2\mathcal{L}_k G_k^T + \bar{T}_k.$$

The stochastic search direction $(\bar{\Delta}\boldsymbol{x}_k, \bar{\Delta}\boldsymbol{\lambda}_k)$:

$$\begin{pmatrix} B_k & G_k^T \\ G_k & \mathbf{0} \end{pmatrix} \begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ \tilde{\Delta}\boldsymbol{\lambda}_k \end{pmatrix} = -\begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}}\mathcal{L}_k \\ c_k \end{pmatrix}$$

$$G_k G_k^T \bar{\Delta}\boldsymbol{\lambda}_k = -(G_k \bar{\nabla}_{\boldsymbol{x}}\mathcal{L}_k + \bar{M}_k^T \bar{\Delta}\boldsymbol{x}_k)$$

- B_k, G_k, c_k are deterministic given $(\boldsymbol{x}_k, \boldsymbol{\lambda}_k)$

Next iterate:

$$\begin{pmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{\lambda}_{k+1} \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_k \\ \boldsymbol{\lambda}_k \end{pmatrix} + \alpha_k \begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ \bar{\Delta}\boldsymbol{\lambda}_k \end{pmatrix}$$

- $\{\alpha_k\}$: a prespecified stepsize sequence

Convergence theory

Assumption:

- the iterates $(\boldsymbol{x}_k, \boldsymbol{\lambda}_k)$ lie in a convex compact set $\mathcal{X} \times \Lambda$
- f and c are thrice continuously differentiable over $\mathcal{X}$
- the Jacobian $G(\boldsymbol{x}) = \nabla^T c(\boldsymbol{x})$ has full row rank over $\mathcal{X}$
- $\boldsymbol{x}^T B_k \boldsymbol{x} \geqslant \gamma_{RH} \|\boldsymbol{x}\|^2$ for all $\boldsymbol{x} \in \{\boldsymbol{x} : G_k \boldsymbol{x} = \boldsymbol{0}, \boldsymbol{x} \neq \boldsymbol{0}\}$, $\|B_k\| \leqslant \kappa_B$

Lemma: There exists a constant $\Upsilon_0 > 0$ such that

$$\mathbb{E}_{\xi_g^k, \xi_H^k}\left[\begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ \bar{\Delta}\boldsymbol{\lambda}_k \end{pmatrix}\right] = \begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{\lambda}_k \end{pmatrix},$$

$$\mathbb{E}_{\xi_g^k, \xi_H^k}\left[\left\|\begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ \bar{\Delta}\boldsymbol{\lambda}_k \end{pmatrix}\right\|^2\right] \leqslant \Upsilon_0 \left(\left\|\begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{\lambda}_k \end{pmatrix}\right\|^2 + \psi\right).$$

Convergence theory

Applying Taylor's expansion and Lemma:

$$\mathbb{E}_{\xi_g^k,\xi_H^k}[\mathcal{L}_{\mu,\nu}^{k+1}] \leqslant \mathcal{L}_{\mu,\nu}^k + \alpha_k \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}_{\mu,\nu}^k \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}_{\mu,\nu}^k \end{pmatrix}^T \begin{pmatrix} \Delta \boldsymbol{x}_k \\ \Delta \boldsymbol{\lambda}_k \end{pmatrix} + \frac{\Upsilon_0 \kappa_{\mathcal{L}_{\mu,\nu}} \alpha_k^2}{2} \left\{ \left\| \begin{pmatrix} \Delta \boldsymbol{x}_k \\ \Delta \boldsymbol{\lambda}_k \end{pmatrix} \right\|^2 + \psi \right\}$$

Convergence theory

Applying Taylor's expansion and Lemma:

$$\mathbb{E}_{\xi_g^k,\xi_H^k}[\mathcal{L}_{\mu,\nu}^{k+1}] \leqslant \mathcal{L}_{\mu,\nu}^{k} + \alpha_k \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}_{\mu,\nu}^{k} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}_{\mu,\nu}^{k} \end{pmatrix}^T \begin{pmatrix} \Delta \boldsymbol{x}_k \\ \Delta \boldsymbol{\lambda}_k \end{pmatrix} + \frac{\Upsilon_0 \kappa_{\mathcal{L}_{\mu,\nu}} \alpha_k^2}{2} \left\{ \left\| \begin{pmatrix} \Delta \boldsymbol{x}_k \\ \Delta \boldsymbol{\lambda}_k \end{pmatrix} \right\|^2 + \psi \right\}$$

If $\mu \geqslant \widetilde{\mu}$, then

$$\begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}_{\mu,\nu}^{k} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}_{\mu,\nu}^{k} \end{pmatrix}^T \begin{pmatrix} \Delta \boldsymbol{x}_k \\ \Delta \boldsymbol{\lambda}_k \end{pmatrix} \leqslant -\widetilde{\delta} \left\| \begin{pmatrix} \Delta \boldsymbol{x}_k \\ G_k \nabla_{\boldsymbol{x}} \mathcal{L}_k \end{pmatrix} \right\|^2$$

Convergence theory

Applying Taylor's expansion and Lemma:

$$\mathbb{E}_{\xi_g^k,\xi_H^k}[\mathcal{L}_{\mu,\nu}^{k+1}] \leqslant \mathcal{L}_{\mu,\nu}^k + \alpha_k \begin{pmatrix} \nabla_{\boldsymbol{x}}\mathcal{L}_{\mu,\nu}^k \\ \nabla_{\boldsymbol{\lambda}}\mathcal{L}_{\mu,\nu}^k \end{pmatrix}^T \begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{\lambda}_k \end{pmatrix} + \frac{\Upsilon_0 \kappa_{\mathcal{L}_{\mu,\nu}} \alpha_k^2}{2} \left\{ \left\| \begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{\lambda}_k \end{pmatrix} \right\|^2 + \psi \right\}$$

If $\mu \geqslant \widetilde{\mu}$, then

$$\begin{pmatrix} \nabla_{\boldsymbol{x}}\mathcal{L}_{\mu,\nu}^k \\ \nabla_{\boldsymbol{\lambda}}\mathcal{L}_{\mu,\nu}^k \end{pmatrix}^T \begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{\lambda}_k \end{pmatrix} \leqslant -\widetilde{\delta} \left\| \begin{pmatrix} \Delta\boldsymbol{x}_k \\ G_k \nabla_{\boldsymbol{x}}\mathcal{L}_k \end{pmatrix} \right\|^2$$

From the system that gives the search direction:

$$\left\| \begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{\lambda}_k \end{pmatrix} \right\|^2 \leqslant \frac{3\kappa_M^2}{\kappa_{1,G}^2} \left\| \begin{pmatrix} \Delta\boldsymbol{x}_k \\ G_k \nabla_{\boldsymbol{x}}\mathcal{L}_k \end{pmatrix} \right\|^2$$

Convergence theory

Applying Taylor's expansion and Lemma:

$$\mathbb{E}_{\xi_g^k,\xi_H^k}[\mathcal{L}_{\mu,\nu}^{k+1}] \leqslant \mathcal{L}_{\mu,\nu}^k + \alpha_k \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}_{\mu,\nu}^k \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}_{\mu,\nu}^k \end{pmatrix}^T \begin{pmatrix} \Delta \boldsymbol{x}_k \\ \Delta \boldsymbol{\lambda}_k \end{pmatrix} + \frac{\Upsilon_0 \kappa_{\mathcal{L}_{\mu,\nu}} \alpha_k^2}{2} \left\{ \left\| \begin{pmatrix} \Delta \boldsymbol{x}_k \\ \Delta \boldsymbol{\lambda}_k \end{pmatrix} \right\|^2 + \psi \right\}$$

If $\mu \geqslant \widetilde{\mu}$, then

$$\begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}_{\mu,\nu}^k \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}_{\mu,\nu}^k \end{pmatrix}^T \begin{pmatrix} \Delta \boldsymbol{x}_k \\ \Delta \boldsymbol{\lambda}_k \end{pmatrix} \leqslant -\widetilde{\delta} \left\| \begin{pmatrix} \Delta \boldsymbol{x}_k \\ G_k \nabla_{\boldsymbol{x}} \mathcal{L}_k \end{pmatrix} \right\|^2$$

From the system that gives the search direction:

$$\left\| \begin{pmatrix} \Delta \boldsymbol{x}_k \\ \Delta \boldsymbol{\lambda}_k \end{pmatrix} \right\|^2 \leqslant \frac{3\kappa_M^2}{\kappa_{1,G}^2} \left\| \begin{pmatrix} \Delta \boldsymbol{x}_k \\ G_k \nabla_{\boldsymbol{x}} \mathcal{L}_k \end{pmatrix} \right\|^2$$

Then

$$\mathbb{E}_{\xi_g^k,\xi_H^k}[\mathcal{L}_{\mu,\nu}^{k+1}] \leqslant \mathcal{L}_{\mu,\nu}^k - \alpha_k \left\{ \widetilde{\delta} - \frac{3\Upsilon_0 \kappa_{\mathcal{L}_{\mu,\nu}} \kappa_M^2}{2\kappa_{1,G}^2} \alpha_k \right\} \left\| \begin{pmatrix} \Delta \boldsymbol{x}_k \\ G_k \nabla_{\boldsymbol{x}} \mathcal{L}_k \end{pmatrix} \right\|^2 + \frac{\Upsilon_0 \kappa_{\mathcal{L}_{\mu,\nu}} \psi}{2} \alpha_k^2$$

Convergence theory

Applying Taylor's expansion and Lemma:

$$\mathbb{E}_{\xi_g^k,\xi_H^k}[\mathcal{L}_{\mu,\nu}^{k+1}] \leqslant \mathcal{L}_{\mu,\nu}^k + \alpha_k \begin{pmatrix} \nabla_{\boldsymbol{x}}\mathcal{L}_{\mu,\nu}^k \\ \nabla_{\boldsymbol{\lambda}}\mathcal{L}_{\mu,\nu}^k \end{pmatrix}^T \begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{\lambda}_k \end{pmatrix} + \frac{\Upsilon_0\kappa_{\mathcal{L}_{\mu,\nu}}\alpha_k^2}{2}\left\{\left\|\begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{\lambda}_k \end{pmatrix}\right\|^2 + \psi\right\}$$

If $\mu \geqslant \widetilde{\mu}$, then

$$\begin{pmatrix} \nabla_{\boldsymbol{x}}\mathcal{L}_{\mu,\nu}^k \\ \nabla_{\boldsymbol{\lambda}}\mathcal{L}_{\mu,\nu}^k \end{pmatrix}^T \begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{\lambda}_k \end{pmatrix} \leqslant -\widetilde{\delta}\left\|\begin{pmatrix} \Delta\boldsymbol{x}_k \\ G_k\nabla_{\boldsymbol{x}}\mathcal{L}_k \end{pmatrix}\right\|^2$$

From the system that gives the search direction:

$$\left\|\begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{\lambda}_k \end{pmatrix}\right\|^2 \leqslant \frac{3\kappa_M^2}{\kappa_{1,G}^2}\left\|\begin{pmatrix} \Delta\boldsymbol{x}_k \\ G_k\nabla_{\boldsymbol{x}}\mathcal{L}_k \end{pmatrix}\right\|^2$$

Then

$$\mathbb{E}_{\xi_g^k,\xi_H^k}[\mathcal{L}_{\mu,\nu}^{k+1}] \leqslant \mathcal{L}_{\mu,\nu}^k - \alpha_k\left\{\widetilde{\delta} - \frac{3\Upsilon_0\kappa_{\mathcal{L}_{\mu,\nu}}\kappa_M^2}{2\kappa_{1,G}^2}\alpha_k\right\}\left\|\begin{pmatrix} \Delta\boldsymbol{x}_k \\ G_k\nabla_{\boldsymbol{x}}\mathcal{L}_k \end{pmatrix}\right\|^2 + \frac{\Upsilon_0\kappa_{\mathcal{L}_{\mu,\nu}}\psi}{2}\alpha_k^2$$

If $\alpha_k \leqslant \frac{\widetilde{\delta}\kappa_{1,G}^2}{3\Upsilon_0\kappa_{\mathcal{L}_{\mu,\nu}}\kappa_M^2}$, then

$$\mathbb{E}_{\xi_g^k,\xi_H^k}[\mathcal{L}_{\mu,\nu}^{k+1}] \leqslant \mathcal{L}_{\mu,\nu}^k - \frac{\alpha_k\widetilde{\delta}}{2}\left\|\begin{pmatrix} \Delta\boldsymbol{x}_k \\ G_k\nabla_{\boldsymbol{x}}\mathcal{L}_k \end{pmatrix}\right\|^2 + \frac{\Upsilon_0\kappa_{\mathcal{L}_{\mu,\nu}}\psi}{2}\alpha_k^2$$

Adaptive stochastic SQP

Sample size in Step 1

The sample size $|\xi_g^k|$ is monotonically increasing and chosen so that the event

$$\mathcal{A}_k = \left\{ \left\| \begin{pmatrix} \bar{g}_k - \nabla f_k + \nu \left(\bar{M}_k G_k \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k - M_k G_k \nabla_{\boldsymbol{x}} \mathcal{L}_k \right) \\ \nu G_k G_k^T G_k \left(\bar{g}_k - \nabla f_k \right) \end{pmatrix} \right\| \right.$$
$$\left. \leqslant \kappa_{grad} \cdot \bar{\alpha}_k \left\| \begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k + \nu \bar{M}_k G_k \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k + G_k^T c_k \\ \nu G_k G_k^T G_k \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k \end{pmatrix} \right\| \right\}$$

satisfies $P(\mathcal{A}_k^c \mid \boldsymbol{x}_k, \boldsymbol{\lambda}_k) \leqslant p_{grad}$

- $\kappa_{grad} > 0$, $p_{grad} \in (0, 1)$ are inputs to the algorithm
- samples can be generated before selecting μ

Sample size in Step 1

The sample size $|\xi_g^k|$ is monotonically increasing and chosen so that the event

$$\mathcal{A}_k = \left\{ \left\| \begin{pmatrix} \bar{g}_k - \nabla f_k + \nu \left(\bar{M}_k G_k \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k - M_k G_k \nabla_{\boldsymbol{x}} \mathcal{L}_k \right) \\ \nu G_k G_k^T G_k \left(\bar{g}_k - \nabla f_k \right) \end{pmatrix} \right\| \right.$$
$$\left. \leqslant \kappa_{grad} \cdot \bar{\alpha}_k \left\| \begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k + \nu \bar{M}_k G_k \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k + G_k^T c_k \\ \nu G_k G_k^T G_k \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k \end{pmatrix} \right\| \right\}$$

satisfies $P(\mathcal{A}_k^c \mid \boldsymbol{x}_k, \boldsymbol{\lambda}_k) \leqslant p_{grad}$

- $\kappa_{grad} > 0$, $p_{grad} \in (0, 1)$ are inputs to the algorithm
- samples can be generated before selecting μ

Algorithm (*Sample size selection*):

While true **do**

1: Generate $|\xi_g^k|$ samples ξ_g^k

2: If

$$|\xi_g^k| < \frac{C_{grad} \log \left(\frac{4d}{p_{grad}} \right)}{\kappa_{grad}^2 \cdot \bar{\alpha}_k^2 \left\| \begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k + \nu \bar{M}_k G_k \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k + G_k^T c_k \\ \nu G_k G_k^T G_k \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k \end{pmatrix} \right\|^2 \wedge 1}$$

then $|\xi_g^k| = \rho |\xi_g^k|$

Stochastic SQP is well-posed

Lemma: Algorithm that selects the sample size $|\xi_g^k|$ terminates in finite time (with probability 1) and $P(\mathcal{A}_k^c \mid \boldsymbol{x}_k, \boldsymbol{\lambda}_k) \leqslant p_{grad}$ a large enough constant C_{grad}.

Stochastic SQP is well-posed

Lemma: Algorithm that selects the sample size $|\xi_g^k|$ terminates in finite time (with probability 1) and $P(\mathcal{A}_k^c \mid \boldsymbol{x}_k, \boldsymbol{\lambda}_k) \leqslant p_{grad}$ a large enough constant C_{grad}.

- the effect of the tuning parameter C_{grad} is negligible

Stochastic SQP is well-posed

Lemma: Algorithm that selects the sample size $|\xi_g^k|$ terminates in finite time (with probability 1) and $P(\mathcal{A}_k^c \mid \boldsymbol{x}_k, \boldsymbol{\lambda}_k) \leqslant p_{grad}$ a large enough constant C_{grad}.

- the effect of the tuning parameter C_{grad} is negligible

Lemma: If

$$|\xi_f^k| \geqslant \frac{C_f \log\left(\frac{8d}{p_f}\right)}{\left\{\kappa_f \bar{\alpha}_k^2 \begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_{\bar{\mu}_k,\nu}^k \\ \bar{\nabla}_{\boldsymbol{\lambda}} \mathcal{L}_{\bar{\mu}_k,\nu}^k \end{pmatrix}^T \begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ \bar{\Delta}\boldsymbol{\lambda}_k \end{pmatrix}\right\}^2 \wedge \bar{\epsilon}_k^2 \wedge 1}$$

for a large enough constant C_f, then

$$P(\mathcal{B}_k^c \mid \boldsymbol{x}_k, \boldsymbol{\lambda}_k, \bar{\Delta}\boldsymbol{x}_k, \bar{\Delta}\boldsymbol{\lambda}_k) \leqslant p_f \quad \text{and} \quad \mathbb{E}_{\xi_f^k}[|\bar{\mathcal{L}}_{\bar{\mu}_k,\nu}^k - \mathcal{L}_{\bar{\mu}_k,\nu}^k|^2] \vee \mathbb{E}_{\xi_f^k}[|\bar{\mathcal{L}}_{\bar{\mu}_k,\nu}^{s_k} - \mathcal{L}_{\bar{\mu}_k,\nu}^{s_k}|^2] \leqslant \bar{\epsilon}_k^2,$$

where

$$\mathcal{B}_k = \left\{ \left|\bar{\mathcal{L}}_{\bar{\mu}_k,\nu}^k - \mathcal{L}_{\bar{\mu}_k,\nu}^k\right| \vee \left|\bar{\mathcal{L}}_{\bar{\mu}_k,\nu}^{s_k} - \mathcal{L}_{\bar{\mu}_k,\nu}^{s_k}\right| \leqslant -\kappa_f \bar{\alpha}_k^2 \begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_{\bar{\mu}_k,\nu}^k \\ \bar{\nabla}_{\boldsymbol{\lambda}} \mathcal{L}_{\bar{\mu}_k,\nu}^k \end{pmatrix}^T \begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ \bar{\Delta}\boldsymbol{\lambda}_k \end{pmatrix} \right\}.$$

Stochastic SQP is well-posed

Lemma: Algorithm that selects the sample size $|\xi_g^k|$ terminates in finite time (with probability 1) and $P(\mathcal{A}_k^c \mid \boldsymbol{x}_k, \boldsymbol{\lambda}_k) \leqslant p_{grad}$ a large enough constant C_{grad}.

- the effect of the tuning parameter C_{grad} is negligible

Lemma: If

$$|\xi_f^k| \geqslant \frac{C_f \log\left(\frac{8d}{p_f}\right)}{\left\{\kappa_f \bar{\alpha}_k^2 \begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_{\bar{\mu}_k,\nu}^k \\ \bar{\nabla}_{\boldsymbol{\lambda}} \mathcal{L}_{\bar{\mu}_k,\nu}^k \end{pmatrix}^T \begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ \bar{\Delta}\boldsymbol{\lambda}_k \end{pmatrix}\right\}^2 \wedge \bar{\epsilon}_k^2 \wedge 1}$$

for a large enough constant C_f, then

$$P(\mathcal{B}_k^c \mid \boldsymbol{x}_k, \boldsymbol{\lambda}_k, \bar{\Delta}\boldsymbol{x}_k, \bar{\Delta}\boldsymbol{\lambda}_k) \leqslant p_f \quad \text{and} \quad \mathbb{E}_{\xi_f^k}[|\bar{\mathcal{L}}_{\bar{\mu}_k,\nu}^k - \mathcal{L}_{\bar{\mu}_k,\nu}^k|^2] \vee \mathbb{E}_{\xi_f^k}[|\bar{\mathcal{L}}_{\bar{\mu}_k,\nu}^{s_k} - \mathcal{L}_{\bar{\mu}_k,\nu}^{s_k}|^2] \leqslant \bar{\epsilon}_k^2,$$

where

$$\mathcal{B}_k = \left\{ \left|\bar{\mathcal{L}}_{\bar{\mu}_k,\nu}^k - \mathcal{L}_{\bar{\mu}_k,\nu}^k\right| \vee \left|\bar{\mathcal{L}}_{\bar{\mu}_k,\nu}^{s_k} - \mathcal{L}_{\bar{\mu}_k,\nu}^{s_k}\right| \leqslant -\kappa_f \bar{\alpha}_k^2 \begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_{\bar{\mu}_k,\nu}^k \\ \bar{\nabla}_{\boldsymbol{\lambda}} \mathcal{L}_{\bar{\mu}_k,\nu}^k \end{pmatrix}^T \begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ \bar{\Delta}\boldsymbol{\lambda}_k \end{pmatrix} \right\}.$$

- the effect of the tuning parameter C_f is negligible
- we do not need a While loop to select

Stochastic SQP is well-posed

Lemma: The condition

$$\begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_k,\nu} \\ \bar{\nabla}_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_k,\nu} \end{pmatrix}^T \begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ \bar{\Delta}\boldsymbol{\lambda}_k \end{pmatrix} \leqslant -\frac{\gamma_{RH} \wedge \nu}{2} \left\| \begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ G_k \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k \end{pmatrix} \right\|^2 \quad \text{and} \quad \|c_k\| \leqslant \|\bar{\nabla} \mathcal{L}^k_{\bar{\mu}_k,\nu}\|$$

can be satisfied by the While loop in the stochastic SQP algorithm.

Stochastic SQP is well-posed

Lemma: The condition

$$\begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_k,\nu} \\ \bar{\nabla}_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_k,\nu} \end{pmatrix}^T \begin{pmatrix} \bar{\Delta} \boldsymbol{x}_k \\ \bar{\Delta} \boldsymbol{\lambda}_k \end{pmatrix} \leqslant -\frac{\gamma_{RH} \wedge \nu}{2} \left\| \begin{pmatrix} \bar{\Delta} \boldsymbol{x}_k \\ G_k \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k \end{pmatrix} \right\|^2 \quad \text{and} \quad \|c_k\| \leqslant \|\bar{\nabla} \mathcal{L}^k_{\bar{\mu}_k,\nu}\|$$

can be satisfied by the While loop in the stochastic SQP algorithm.

Furthermore, there exists a deterministic constant $\widetilde{\mu} > 0$ such that $\bar{\mu}_k = \bar{\mu}_{\bar{K}} \leqslant \widetilde{\mu}$, $\forall k \geqslant \bar{K}$ for some $\bar{K} < \infty$.

Stochastic SQP is well-posed

Lemma: The condition

$$\begin{pmatrix} \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_k,\nu} \\ \bar{\nabla}_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_k,\nu} \end{pmatrix}^T \begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ \bar{\Delta}\boldsymbol{\lambda}_k \end{pmatrix} \leqslant -\frac{\gamma_{RH} \wedge \nu}{2} \left\| \begin{pmatrix} \bar{\Delta}\boldsymbol{x}_k \\ G_k \bar{\nabla}_{\boldsymbol{x}} \mathcal{L}_k \end{pmatrix} \right\|^2 \quad \text{and} \quad \|c_k\| \leqslant \|\bar{\nabla} \mathcal{L}^k_{\bar{\mu}_k,\nu}\|$$

can be satisfied by the While loop in the stochastic SQP algorithm.

Furthermore, there exists a deterministic constant $\widetilde{\mu} > 0$ such that $\bar{\mu}_k = \bar{\mu}_{\bar{K}} \leqslant \widetilde{\mu}$, $\forall k \geqslant \bar{K}$ for some $\bar{K} < \infty$.

- for each run of the algorithm, the merit function is invariant after certain number of iterations
- the threshold $\bar{K}$ is random and might be different for each run
- we study iterations after $\bar{K}$ to establish global convergence
- $\bar{\mu}_{\bar{K}}$ has a deterministic upper bound $\widetilde{\mu}$

Comment on convergence analysis

We set ω to satisfy

$$\frac{1-\omega}{\omega} \leqslant \frac{\beta(\gamma_{RH} \wedge \nu)}{32\rho \left\{\kappa_{\mathcal{L}_{\tilde{\mu},\nu}} \alpha_{max} \Upsilon_3 \vee \left(\kappa_{grad} \alpha_{max} \Upsilon_1 + \Upsilon_4\right)\right\}^2} \wedge \frac{1}{4(\rho-1)}$$

One-step error recursion

Lemma:

- On the event $\mathcal{A}_k \cap \mathcal{B}_k$,

$$\Phi^{k+1}_{\bar{\mu}_{\bar{K}},\nu,\omega} - \Phi^{k}_{\bar{\mu}_{\bar{K}},\nu,\omega} = -\frac{1}{2}(1-\omega)\left(1-\frac{1}{\rho}\right)\left(\bar{\epsilon}_k + \bar{\alpha}_k \left\| \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \end{pmatrix} \right\|^2 \right).$$

One-step error recursion

Lemma:

- On the event $\mathcal{A}_k \cap \mathcal{B}_k$,

$$\Phi^{k+1}_{\bar{\mu}_{\bar{K}},\nu,\omega} - \Phi^{k}_{\bar{\mu}_{\bar{K}},\nu,\omega} = -\frac{1}{2}(1-\omega)\left(1-\frac{1}{\rho}\right)\left(\bar{\epsilon}_k + \bar{\alpha}_k \left\| \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \end{pmatrix} \right\|^2 \right).$$

- On the event $\mathcal{A}_k^c \cap \mathcal{B}_k$,

$$\Phi^{k+1}_{\bar{\mu}_{\bar{K}},\nu,\omega} - \Phi^{k}_{\bar{\mu}_{\bar{K}},\nu,\omega} \leqslant \rho(1-\omega)\bar{\alpha}_k \left\| \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \end{pmatrix} \right\|^2.$$

One-step error recursion

Lemma:

- On the event $\mathcal{A}_k \cap \mathcal{B}_k$,

$$\Phi^{k+1}_{\bar{\mu}_{\bar{K}},\nu,\omega} - \Phi^{k}_{\bar{\mu}_{\bar{K}},\nu,\omega} = -\frac{1}{2}(1-\omega)\left(1-\frac{1}{\rho}\right)\left(\bar{\epsilon}_k + \bar{\alpha}_k \left\| \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \end{pmatrix} \right\|^2 \right).$$

- On the event $\mathcal{A}_k^c \cap \mathcal{B}_k$,

$$\Phi^{k+1}_{\bar{\mu}_{\bar{K}},\nu,\omega} - \Phi^{k}_{\bar{\mu}_{\bar{K}},\nu,\omega} \leqslant \rho(1-\omega)\bar{\alpha}_k \left\| \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \end{pmatrix} \right\|^2.$$

- On the event $\mathcal{B}_k^c$,

$$\begin{aligned}\Phi^{k+1}_{\bar{\mu}_{\bar{K}},\nu,\omega} - \Phi^{k}_{\bar{\mu}_{\bar{K}},\nu,\omega} \leqslant \rho(1-\omega)\bar{\alpha}_k & \left\| \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \end{pmatrix} \right\|^2 \\ & + \omega\left(|\bar{\mathcal{L}}^{s_k}_{\bar{\mu}_{\bar{K}},\nu} - \mathcal{L}^{s_k}_{\bar{\mu}_{\bar{K}},\nu}| + |\bar{\mathcal{L}}^{k}_{\bar{\mu}_{\bar{K}},\nu} - \mathcal{L}^{k}_{\bar{\mu}_{\bar{K}},\nu}|\right).\end{aligned}$$

One-step error recursion

Lemma:

- On the event $\mathcal{A}_k \cap \mathcal{B}_k$,

$$\Phi^{k+1}_{\bar{\mu}_{\bar{K}},\nu,\omega} - \Phi^{k}_{\bar{\mu}_{\bar{K}},\nu,\omega} = -\frac{1}{2}(1-\omega)\left(1-\frac{1}{\rho}\right)\left(\bar{\epsilon}_k + \bar{\alpha}_k \left\| \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \end{pmatrix} \right\|^2 \right).$$

- On the event $\mathcal{A}_k^c \cap \mathcal{B}_k$,

$$\Phi^{k+1}_{\bar{\mu}_{\bar{K}},\nu,\omega} - \Phi^{k}_{\bar{\mu}_{\bar{K}},\nu,\omega} \leqslant \rho(1-\omega)\bar{\alpha}_k \left\| \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \end{pmatrix} \right\|^2 .$$

- On the event $\mathcal{B}_k^c$,

$$\Phi^{k+1}_{\bar{\mu}_{\bar{K}},\nu,\omega} - \Phi^{k}_{\bar{\mu}_{\bar{K}},\nu,\omega} \leqslant \rho(1-\omega)\bar{\alpha}_k \left\| \begin{pmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}^k_{\bar{\mu}_{\bar{K}},\nu} \end{pmatrix} \right\|^2 + \omega(|\bar{\mathcal{L}}^{s_k}_{\bar{\mu}_{\bar{K}},\nu} - \mathcal{L}^{s_k}_{\bar{\mu}_{\bar{K}},\nu}| + |\bar{\mathcal{L}}^{k}_{\bar{\mu}_{\bar{K}},\nu} - \mathcal{L}^{k}_{\bar{\mu}_{\bar{K}},\nu}|).$$

- if either function or gradient are imprecisely estimated, then there is no guarantee that $\Phi^{k}_{\bar{\mu}_{\bar{K}},\nu,\omega}$ will decrease
- the increase of $\Phi^{k}_{\bar{\mu}_{\bar{K}},\nu,\omega}$ can be controlled, when p_f, p_{grad} are small enough

References I

A. S. Bandeira, K. Scheinberg, and L. N. Vicente. Convergence of trust-region methods based on probabilistic models. *SIAM Journal on Optimization*, 24(3): 1238–1264, January 2014. ISSN 1052-6234. doi: 10.1137/130915984. URL https://doi.org/10.1137/130915984.

Albert S. Berahas, Jiahao Shi, Zihong Yi, and Baoyu Zhou. Accelerating stochastic sequential quadratic programming for equality constrained optimization using predictive variance reduction.

Albert S Berahas, Frank E Curtis, Michael J O'Neill, and Daniel P Robinson. A stochastic sequential quadratic optimization algorithm for nonlinear equality constrained optimization with rank-deficient jacobians. *arXiv preprint arXiv:2106.13015*, 2021a. URL https://arxiv.org/abs/2106.13015.

Albert S. Berahas, Frank E. Curtis, Daniel Robinson, and Baoyu Zhou. Sequential quadratic optimization for nonlinear equality constrained stochastic optimization. *SIAM Journal on Optimization*, 31(2):1352–1379, January 2021b. doi: 10.1137/20m1354556. URL https://doi.org/10.1137/20M1354556.

Dimitri Bertsekas. *Network optimization: continuous and discrete methods*. Athena Scientific, Belmont, Mass, 1998. ISBN 1886529027.

Dimitri P. Bertsekas. Projected newton methods for optimization problems with simple constraints. *SIAM Journal on Control and Optimization*, 20(2): 221–246, March 1982. ISSN 0363-0129. doi: 10.1137/0320018. URL https://doi.org/10.1137/0320018.

John T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, volume 19. Society for Industrial and Applied Mathematics, jan 2010. doi: 10.1137/1.9780898718577.

Jose Blanchet, Coralia Cartis, Matt Menickelly, and Katya Scheinberg. Convergence rate analysis of a stochastic trust-region method via supermartingales. *INFORMS Journal on Optimization*, 1(2):92–119, April 2019. doi: 10.1287/ijoo.2019.0016. URL https://doi.org/10.1287/ijoo.2019.0016.

Richard H. Byrd, Robert B. Schnabel, and Gerald A. Shultz. A trust region algorithm for nonlinearly constrained optimization. 24(5):1152–1170, 1987. doi: 10.1137/0724076. URL https://doi.org/10.1137/0724076.

MR Celis, JE Dennis, and RA Tapia. Nonlinear equality constrained optimization. In *Numerical Optimization 1984: Proceedings of the SIAM Conference on Numerical Optimization, Boulder, Colorado, June 12-14, 1984*, number 20, page 71. Society for Industrial & Applied, 1985.

R. Chen, M. Menickelly, and K. Scheinberg. Stochastic optimization using a trust-region method and random models. *Mathematical Programming*, 169(2): 447–487, April 2017. ISSN 0025-5610. doi: 10.1007/s10107-017-1141-8. URL https://doi.org/10.1007/s10107-017-1141-8.

You-Lin Chen, Zhaoran Wang, and Mladen Kolar. Provably training overparameterized neural network classifiers with non-convex constraints. *Electronic Journal of Statistics*, 16(2):5812 – 5851, 2022. doi: 10.1214/22-EJS2036. URL https://doi.org/10.1214/22-EJS2036.

Frank E. Curtis and Daniel P. Robinson. Exploiting negative curvature in deterministic and stochastic optimization. 176(1-2):69–94. doi: 10.1007/s10107-018-1335-8.

Frank E. Curtis and Rui Shi. A fully stochastic second-order trust region method. *Optimization Methods and Software*, pages 1–34, November 2020. doi: 10.1080/10556788.2020.1852403. URL https://doi.org/10.1080/10556788.2020.1852403.

Frank E. Curtis, Katya Scheinberg, and Rui Shi. A stochastic trust region algorithm based on careful step normalization. *INFORMS Journal on Optimization*, 1(3):200–220, July 2019. doi: 10.1287/ijoo.2018.0010. URL https://doi.org/10.1287/ijoo.2018.0010.

Frank E Curtis, Daniel P Robinson, and Baoyu Zhou. Inexact sequential quadratic optimization for minimizing a stochastic objective function subject to deterministic nonlinear equality constraints. *arXiv preprint arXiv:2107.03512*, 2021. URL https://arxiv.org/abs/2107.03512.

Jitka Dupacova and Roger Wets. Asymptotic behavior of statistical estimators and of optimal solutions of stochastic optimization problems. *The Annals of Statistics*, 16(4):1517–1549, December 1988. ISSN 0090-5364. doi: 10.1214/aos/1176351052. URL https://doi.org/10.1214/aos/1176351052.

Brian R. Gaines, Juhyun Kim, and Hua Zhou. Algorithms for fitting the constrained lasso. *J. Comput. Graph. Statist.*, 27(4):861–871, 2018. ISSN 1061-8600. doi: 10.1080/10618600.2018.1473777. URL https://doi.org/10.1080/10618600.2018.1473777.

Gareth M. James, Courtney Paulson, and Paat Rusmevichientong. Penalized and constrained optimization: an application to high-dimensional website advertising. *J. Amer. Statist. Assoc.*, 115(529):107–122, 2020. ISSN 0162-1459. doi: 10.1080/01621459.2019.1609970. URL https://doi.org/10.1080/01621459.2019.1609970.

George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021. ISSN 2522-5820. doi: 10.1038/s42254-021-00314-5. URL https://doi.org/10.1038/s42254-021-00314-5.

F.-S. Kupfer and E. W. Sachs. Numerical solution of a nonlinear parabolic control problem by a reduced SQP method. *Comput. Optim. Appl.*, 1(1): 113–135, 1992. ISSN 0926-6003. doi: 10.1007/BF00247656. URL https://doi.org/10.1007/BF00247656.

S. Lucidi. Recursive quadratic programming algorithm that uses an exact augmented lagrangian function. *Journal of Optimization Theory and Applications*, 67(2):227–245, November 1990. ISSN 0022-3239. doi: 10.1007/bf00940474. URL https://doi.org/10.1007/BF00940474.

Sen Na, Mihai Anitescu, and Mladen Kolar. A fast temporal decomposition procedure for long-horizon nonlinear dynamic programming. *Technical report, arXiv:2107.11560*, July 2021a.

Sen Na, Mihai Anitescu, and Mladen Kolar. Inequality constrained stochastic nonlinear optimization via active-set sequential quadratic programming. *Technical report*, September 2021b.

Sen Na, Mihai Anitescu, and Mladen Kolar. An adaptive stochastic sequential quadratic programming with differentiable exact augmented lagrangians. 2022. doi: 10.1007/s10107-022-01846-z.

Neerchal K. Nagaraj and Wayne A. Fuller. Estimation of the parameters of linear time series models subject to nonlinear restrictions. *The Annals of Statistics*, 19(3):1143–1154, September 1991. ISSN 0090-5364. doi: 10.1214/aos/1176348242. URL https://doi.org/10.1214/aos/1176348242.

Yatin Nandwani, Abhishek Pathak, Mausam, and Parag Singla. A primal dual formulation for deep learning with constraints. In *Advances in Neural Information Processing Systems 32*, pages 12157–12168. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/9385-a-primal-dual-formulation-for-deep-learning-with-constraints.pdf.

A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, January 2009. ISSN 1052-6234. doi: 10.1137/070704277. URL https://doi.org/10.1137/070704277.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer New York, 2nd edition, 2006. ISBN 978-0387-30303-1; 0-387-30303-0. doi: 10.1007/978-0-387-40065-5. URL https://doi.org/10.1007/978-0-387-40065-5.

Emmanuel Omotayo Omojokun. *Trust region algorithms for optimization with nonlinear equality and inequality constraints*. PhD thesis, 1989.

Courtney Paquette and Katya Scheinberg. A stochastic line search method with expected complexity analysis. *SIAM Journal on Optimization*, 30(1):349–376, January 2020. ISSN 1052-6234. doi: 10.1137/18m1216250. URL https://doi.org/10.1137/18M1216250.

M. J. D. Powell and Y. Yuan. A trust region algorithm for equality constrained optimization. 49(1-3):189–211, 1990. doi: 10.1007/bf01588787. URL https://doi.org/10.1007/BF01588787.

Bernd Prach and Christoph H. Lampert. Almost-orthogonal layers for efficient general-purpose lipschitz networks. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXI*, volume 13681 of *Lecture Notes in Computer Science*, pages 350–365. Springer, 2022. doi: 10.1007/978-3-031-19803-8_21. URL https://doi.org/10.1007/978-3-031-19803-8_21.

Sathya N. Ravi, Tuan Dinh, Vishnu Suresh Lokhande, and Vikas Singh. Explicitly imposing constraints in deep networks via conditional gradients gives improved generalization and faster convergence. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 4772–4779. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33014772. URL https://doi.org/10.1609/aaai.v33i01.33014772.

Tyrone Rees, H. Sue Dollar, and Andrew J. Wathen. Optimal solvers for PDE-constrained optimization. *SIAM Journal on Scientific Computing*, 32(1): 271–298, jan 2010. doi: 10.1137/080727154.

Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *ArXiv e-prints 1610.03295*, 2016.

Alexander Shapiro. On the asymptotics of constrained local m-estimators. *The Annals of Statistics*, 28(3):948–960, May 2000. ISSN 0090-5364. doi: 10.1214/aos/1015952006. URL https://doi.org/10.1214/aos/1015952006.

Avi Vardi. A trust region algorithm for equality constrained minimization: Convergence properties and implementation. 22(3):575–591, 1985. doi: 10.1137/0722035. URL https://doi.org/10.1137/0722035.

Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar. Integrating scientific knowledge with machine learning for engineering and environmental systems. March 2020.

Ming Yu, Zhuoran Yang, Mladen Kolar, and Zhaoran Wang. Convergent policy optimization for safe reinforcement learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 3121–3133. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/8576-convergent-policy-optimization-for-safe-reinforcement-learning.pdf.