A reduced-space method for chance constrained optimization problems with inner-outer smoothing approximations (a tutorial)

Dr. rer. nat. habil. Abebe Geletu
E-mail: abebe.geletu@tu-ilmenau.de

German Research Chair
(Applied Mathematics and Artificial Intelligence)
AIMS Rwanda

AIMS
African Institute for Mathematical Sciences
RWANDA

Alexander von Humboldt Stiftung/Foundation

DAAD
Deutscher Akademischer Austausch Dienst
German Academic Exchange Service

Topics

1. Introduction
2. The reduced-space method
3. Smoothing inner-outer approximation - overview
4. The reduced-space method for inner-outer approximation problems
5. Some comments on parallel implementation
6. An exercise
7. References

1. Introduction

General form

The general form of chance constrained Optimization.

$$(CCOPT) \qquad \min_{u}\{E[f(x,u,\xi)]\} \tag{1}$$

subject to:

$$G(x,u,\xi) = 0, \tag{2}$$

$$Pr\{g(u,x,\xi) \leq 0\} \geq \alpha, \tag{3}$$

$$u \in \mathcal{U} \subset \mathbb{R}^m, \tag{4}$$

- ξ $(\in \Omega \subset \mathbb{R}^p)$ - **random vector** with **continuous pdf** $\phi(\xi)$
- ξ is associated with a **complete probability space** (Ω, Σ, Pr)
- u - **deterministic decision variable**
- x - **state variable** (dummy variable)

1. Introduction ... problem parts description

Assumption: The functions $f, g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ are at least **one-time differentiable**.

Equality constraint: $G(x, u, \xi) = 0$

- arises as a model equation of physical process
- $G(x, u, \xi) = 0$ implies that both decision u and random variable ξ have influence on the state variable x; i.e., $x(u, \xi)$.

Objective function: $E[f(x, u, \xi)]$

- by definition

$$E[f(x, u, \xi)] = \int_{\Omega} f(x, u, \xi)\phi(\xi)d\xi$$

  is a **multidimensional integral** if $\xi \in \mathbb{R}^p$.
- $E[f(x, u, \xi)] = f(x, u)$ if ξ does not appear in the objective

1. Introduction ... problem parts description...

Chance (probability) constraint: $Pr\{g(u,x,\xi) \leq 0\} \geq \alpha$

- The expression $Pr\{g(u,x,\xi) \leq 0\}$ is a compact form for

$$Pr\{\xi \in \Omega \mid g(u,x,\xi) \leq 0\}$$

- Integral representation

$$Pr\{g(u,x,\xi) \leq 0\} = \int_{\{\xi \in \Omega \mid g(u,x,\xi) \leq 0\}} \phi(\xi) d\xi$$

- Probability function

$$p(u) := Pr\{g(u,x,\xi) \leq 0\} \quad \text{(defintion)},$$

Note that:

  - $p(u) = Pr\{g(u,x(u,x),\xi) \leq 0\}$
  - $p : \mathcal{U} \subset \mathbb{R}^m \to [0,1]$.

- α reliability level, α commonly a pre-given value near 1.

1. Introduction ...

Modern applications areas of CCOPT

- Risk-metrics and portfolio optimization
- (Planning or managerial) decision making under uncertainties
- Reliability-based engineering design optimization (design with fault-tolerance)
- Predictive control (decision making) on a moving horizon in the presence of uncertainties
- Optimum decision making for maximum probability of gains (profits) (or with minimum probability of losses (defaults)
- Designing reliable machine learning models
- etc.

2. Reduced space method

Consider a nonlinear optimization problem

$$(NLP) \qquad \min_u f(x,u) \tag{5}$$

subject to:

$$G(x,u) = 0 \tag{6}$$

$$g(x,u) \leq 0 \tag{7}$$

$$u \in \mathcal{U} = \{u \in \mathbb{R}^m \mid u_{min} \leq u \leq u_{max}\}, \tag{8}$$

where $u_{min}, u_{max} \in \mathbb{R}^m$ are given (fixed) lower and upper bound vectors; i.e., $\mathcal{U}$ is a box constraint on u.

Problems of type (NLP) arise from CCOPT, discretized optimal control problems, or from optimization with PDE constraints, etc. **Commonly, a few decisions (controls) u than state variables x; i.e., $m << n$.**

2. Reduced space method ... Idea

Generally, if x is not a decision variable and there are a lot of x's, do not involve the x's directly in the optimization procedure. Thus, for a given $u \in \mathcal{U}$, we solve for x from $G(x, u) = 0$ to obtain x in terms of u; i.e., $x(u)$.
Hence,

- we avoid x and drop the model equation (equality constraints) $G(x, u) = 0$ from direct consideration in the optimization procedure.

Reduced problem

$$(NLP)_{Red} \qquad \min_{u} f(x(u), u) \tag{9}$$

subject to:

$$g(x(u), u) \leq 0 \tag{10}$$

$$u \in \mathcal{U} = \{u \in \mathbb{R}^m \mid u_{min} \leq u \leq u_{max}\}, \tag{11}$$

2. Reduced space method ... Idea

- Any gradient-based algorithm to solve $(NLP)_{Red}$ uses the iteration scheme

$$u^{(k+1)} = u^{(k)} + \alpha_k d_k, k = 0, 1, 2, \ldots$$

- The search direction d_k and the step-length α_k are computed (at least) using
  (i) the values $f(x(u^k), u^{(k)})$ and $g(x(u^k), u^{(k)})$; and
  (ii) gradients $\nabla_u \left[f(x(u^{(k)}), u^{(k)})\right]$ and $\nabla_u \left[g(x(u^{(k)}), u^{(k)})\right]$.
  The data at (i) and (ii) need to be made available to the optimization solver.

Note that, for $F(u) = f(x(u), u)$, we have

$$\frac{\partial F(u)}{\partial u_j} = \nabla_x f(x(u), u) \circ \frac{\partial x(u)}{\partial u_j} + \frac{\partial f(x(u), u)}{\partial u_j}, j = 1, \ldots, m.$$

Here, $\circ$ represents dot (or scalar) product.

2. Reduced space method ... Idea

Since x is a vector, we have

$$\frac{\partial x(u)}{\partial u_j} = \begin{pmatrix} \frac{\partial x_1(u)}{\partial u_j} \\ \frac{\partial x_2(u)}{\partial u_j} \\ \vdots \\ \frac{\partial x_n(u)}{\partial u_j} \end{pmatrix} \in \mathbb{R}^n, j = 1, \ldots, m.$$

The derivative of vector $x(u)$ w.r.t. the vector u is represented by the matrix

$$D_u x(u) = \left[\frac{\partial x(u)}{\partial u_1} | \frac{\partial x(u)}{\partial u_2} | \ldots | \frac{\partial x(u)}{\partial u_m} \right] \in \mathbb{R}^{n \times m}.$$

Therefore, we can write

$$\nabla F(u) = [D_u x(u)]^\top \nabla_x f(x(u), u) + \nabla_u f(x(u), u). \tag{12}$$

Similarly, setting $g(u) := g(x(u), u)$,

$$\nabla g(u) = [D_u x(u)]^\top \nabla_x g(x(u), u) + \nabla_u g(x(u), u). \tag{13}$$

2. Reduced space method ... Idea ...

A representation for $[D_u x(u)]$ is obtained through the total differentiation of the system of equations $G(x(u), u) = 0$ w.r.t. u.

$$[D_u x(u)]^\top \nabla_x G_\ell(x(u), u) + \nabla_u G_\ell(x(u), u) = 0, \ell = 1, \ldots, q. \tag{14}$$

Assuming the system $G(x(u), u) = 0$ consists of q equations.

Remark:

Observe that, once u is given and $x(u)$ is found by solving $G(x(u), u) = 0$, the vectors $\nabla_u G_\ell(x(u), u)$ and $\nabla_x G_\ell(x(u), u)$, $\ell = 1, \ldots, q$, in equation (14) are easy to determine. Subsequently, $D_u x(u)$ is found by solving the linear system of equations (14).

2. Reduced space method ... A general algorithm

Algorithm 1: A general algorithm of the reduced space method

1: Choose a termination tolerance $\varepsilon > 0$ (e.g., $\varepsilon = 10E-7$, etc.);
2: Choose an initial iterate $u^{(0)}$;
3: Set $k \leftarrow 0$;
4: **while** (termination criteria not satisfied) **do**
5: Solve the system of equations $G(x, u^{(k)}) = 0$ for x to obtain $x^{(k)} = x(u^{(k)})$;
6: Determine $D_u x(u^{(k)})$ by using $x(u^{(k)})$ and solving the system

$$[D_u x(u)]^\top \nabla_x G_\ell(x(u^{(k)}), u^{(k)}) + \nabla_u G_\ell(x(u^{(k)}), u^{(k)}) = 0, \ell = 1, \ldots, q.$$

7: Use the results from Steps 6 and 7 to obtain the values $f(x(u^k), u^{(k)})$ and $g(x(u^k), u^{(k)})$; and gradients $\nabla_u \left[f(x(u^{(k)}), u^{(k)})\right]$ (from eqn. (12)), $\nabla_u \left[g(x(u^{(k)}), u^{(k)})\right]$ (from eqn. (13)).
8: Determine a search direction d_k and step-length α_k ;
9: Update the iterate

$$u^{(k+1)} = u^{(k)} + \alpha_k d_k$$

10: Set $k \leftarrow k + 1$;
11: **end while**

2. Reduced space method ... A general algorithm...

Remark

Each iteration of the reduced space Algorithm 1 needs the solution of

- the nonlinear system $G(x, u^{(k)}) = 0$, and
- the solution of the linear system

$$[D_u x(u)]^\top \nabla_x G_\ell(x(u^{(k)}), u^{(k)}) + \nabla_u G_\ell(x(u^{(k)}), u^{(k)}) = 0, \quad \ell = 1, \ldots, q.$$

Suggestion: To solve $(NLP)_{Red}$ use the optimization solver
- **IpOpt**: https://coin-or.github.io/Ipopt/
- also try **pyIpOpt**: https://github.com/xuy/pyipopt
in conjunction with solvers of nonlinear and linear systems of equations.
- **GAMS** is also a very good choice (licensed version includes IpOpt)

2. Reduced space method ... some large-scale equation solvers...

Solvers for (large-sacle) nonlinear equations:

- KINSOL: https://computing.llnl.gov/projects/sundials/kinsol
- NITSOL: https://users.wpi.edu/walker/NITSOL/
- dfsane: https://www.rdocumentation.org/packages/BB/versions/2019.10-1/topics/dfsane
- GSL: https://www.gnu.org/software/gsl/doc/html/multiroots.html

Solvers for (large-sacle) linear equations:

- PARDISO: https://www.pardiso-project.org/
- Harwell Software Library: https://www.hsl.rl.ac.uk/
- Armadillo: https://arma.sourceforge.net/ https://sourceforge.net/projects/arma/
- pyarmadillo:https://gitlab.com/jason-rumengan/pyarma
- Eigen: https://eigen.tuxfamily.org/dox/index.html

2. Reduced space method ... some large-scale equation solvers...

... large-sacle linear equations solvers...

- csparse:
  https://people.sc.fsu.edu/ jburkardt/c_src/csparse/csparse.html
- UMFPACK:
  https://people.sc.fsu.edu/ jburkardt/f77_src/umfpack/umfpack.html
- SuiteSparse:
  https://github.com/DrTimothyAldenDavis/SuiteSparse

The last three libraries are related to the work of Tim Davis.

For more read:

- Tim Davis *etal*. A survey of direct methods for sparse linear systems, Acta Numerica (2016), pp. 383566. (available online)
- Timothy Davis (Book): Direct Methods for Sparse Linear Systems. SIAM 2006.

Wherever possible use shared- or distributed-memory parallel implementation.

3. Smoothing inner-outer approximation - overview

Recall the chance constrained optimization problem

$$
\begin{aligned}
(CCOPT) \qquad & \min_{u}\{E[f(x,u,\xi)]\} && (15)\\
& \text{subject to:} \\
& \qquad G(x,u,\xi) = 0, && (16)\\
& \qquad p(u) = Pr\{g(u,x,\xi) \leq 0\} \geq \alpha, && (17)\\
& \qquad u \in \mathcal{U} \subset \mathbb{R}^m, && (18)
\end{aligned}
$$

Assumptions:

- For each fixed u and realization of the random variable ξ, we can solve $G(x,u,\xi) = 0$ to obtain $x(u,\xi)$.
- (**MZP**) The set $\Gamma_0(u) := \{\xi \in \Omega \mid g(x(u,\xi),u,\xi) = 0\}$ is of measure zero.

3.Smoothing inner-outer approximation ... Properties

Reduced form

$$(CCOPT) \qquad \min_{u} E\left[f(u,\xi)\right] \tag{19}$$

$$s.t.$$

$$p(u) = Pr\{g(u,\xi) \leq 0\} \geq \alpha \tag{20}$$

$$u \in U, \tag{21}$$

with $f(u,\xi) := f(x(u,\xi),u,\xi)$ and $g(u,\xi) := g(x(u,\xi),u,\xi)$.

Major difficulties with CCOPT:

- the probability function $p(u)$ is difficult to directly evaluate
- commonly $p(u)$ is non-differentiable and non-convex
- the knowledge of differentiable or convexity do not provide simpler evaluation schemes for $p(u)$ and $\nabla p(u)$
- generally, CCOPT belongs to the class of hard optimization problems

3. Smoothing inner-outer approximation - overview

Equivalent representations

$$p(u) = Pr\{g(u,\xi) \leq 0\} \geq \alpha$$

$$\Leftrightarrow \qquad 1 - p(u) = Pr\{g(u,\xi) \geq 0\} \leq 1 - \alpha$$

$$\Leftrightarrow \qquad \mathbf{E}\,[\mathbf{h}(\mathbf{g}(\mathbf{u},\xi))] \leq \mathbf{1} - \alpha \quad \text{(see tutorial slides)}$$

$$h(s) := \begin{cases} 1, & \text{if } s \geq 0, \\ 0, & \text{if } s < 0, \end{cases} \quad \textbf{(Heaviside step function)}.$$

3. Smoothing inner-outer approximation

Geletu-Hoffmann (GH) function

The parametric family (Geletu *et al.* 2015, 2017, 2020)

$$\Theta(\tau, s) = \frac{1 + m_1\tau}{1 + m_2\tau \exp\left(-\frac{s}{\tau}\right)}, \quad \textit{for} \quad \tau \in (0,1),\ s \in \mathbb{R}, \tag{22}$$

where m_1, m_2 are constants with $0 < m_2 \leq m_2/(1+m_1)$. Define also, $\Pi(\tau, s) := \Theta(\tau, -s)$. Thus,

$$1 - \Theta(\tau, s) < h(-s) < \Theta(\tau, -s) = \Pi(\tau, s) \tag{23}$$

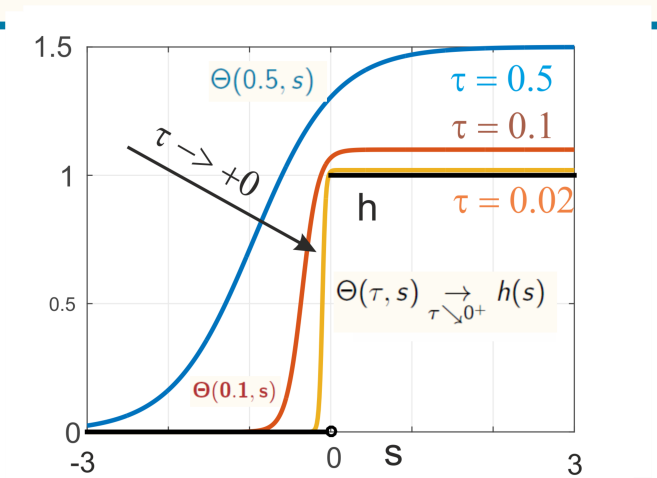

Figure: Convergence of Θ to h

3. Smoothing inner-outer approximation ...

The Geletu-Hoffmann function $\Theta(\tau, s)$ is a smoothing and monotonically convergent approximation of the heaviside (step) function $h(s)$.

Smoothing Approximation functions for $1-p(u)$ and $p(u)$

Define the functions (Geletu *et al.* 2015, 2017)

$$\psi(\tau, u) := E\left[\Theta(\tau, g(u, \xi))\right], \tag{24}$$
$$\varphi(\tau, u) := E\left[\Pi(\tau, g(u, \xi))\right], \tag{25}$$

where $\tau \in (0, 1)$.

3. Smoothing inner-outer approximation ... Problems

Inner-outer approximation problems

Inner Approximation

$$(IA_\tau) \quad \min_u \ F(u) \quad \text{s.t.} \quad \psi(\tau, u) \leq 1 - \alpha, \quad u \in U, \tau \in (0,1).$$

Outer Approximation

$$(OA_\tau) \quad \min_u \ F(u) \quad \text{s.t.} \quad \varphi(\tau, u) \geq \alpha, \quad u \in U, \tau \in (0,1).$$

Respective feasible sets of IA_τ and OA_τ

$$\mathcal{M}(\tau) := \{u \in U \mid \psi(\tau, u) \leq 1 - \alpha\}, \tau \in (0,1),$$

$$\mathcal{S}(\tau) := \{u \in U \mid \varphi(\tau, u) \geq \alpha\}, \tau \in (0,1),$$

where $F(u) := E[f(x(u,\xi), u, \xi)]$.

3. Smoothing inner-outer approximation ... Properties

Geletu et al. 2015

Suppose $0 < \tau_2 \leq \tau_1 < 1$ and $g(\cdot, \xi)$ is continuous w.r.t. u. Then,

(1) **smoothness**: $\psi(\tau, \cdot)$ and $\varphi(\tau, \cdot)$ are smooth if $g(\cdot, \xi)$ is smooth, for each fixed $\tau \in (0, 1)$.

(2) **monotonicity**: For $u \in U$,

$$\varphi(\tau_1, u) \geq \varphi(\tau_2, u) \geq p(u) \geq 1 - \psi(\tau_2, u) \geq 1 - \psi(\tau_1, u).$$

(3) **tight approximation**: For each $u \in \mathcal{U}$,

$$p(u) = \inf_{\tau \in (0,1)} \varphi(\tau, u) \quad \text{and} \quad \sup_{\tau \in (0,1)} (\psi(\tau, u)) = 1 - p(u),$$

(4) **Convergence of the feasible sets**: $M(\tau) \subset \mathcal{P} \subset S(\tau)$, for all $\tau \in (0, 1)$, and

$$\lim_{\tau \searrow 0^+} \mathrm{M}(\tau) = \mathcal{P}, \quad \lim_{\tau \searrow 0^+} S(\tau) = \mathcal{P},$$

monotonically.

3. Smoothing inner-outer approximation ... Properties...

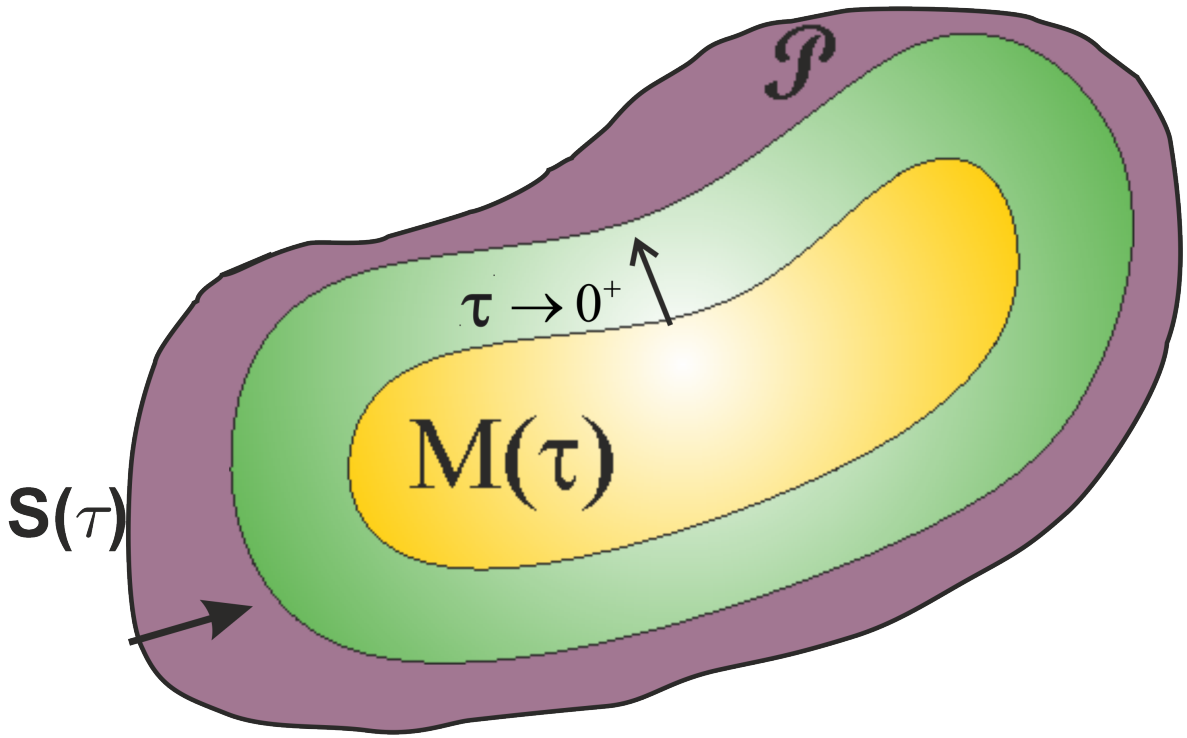

Figure: Inner-Outer approximation for the feasible set of CCOPT

3. Inner-outer approximation ...

Convergence of Gradients ...Geletu *et al.* 2015, 2017

(4) If $g(\cdot,\xi)$ is differentiable function, the under standard assumptions the functions $\psi(\tau,\cdot)$ and $\varphi(\tau,\cdot)$ are differentiable w.r.t. u and

$$\begin{aligned}\nabla(1-\psi(\tau,u)) &= -\int_\Omega \left.\frac{\partial\Theta(\tau,s)}{\partial s}\right|_{s=g(u,\xi)} \nabla_u g(u,\xi)\phi(\xi)d\xi,\\ \nabla\varphi(\tau,u) &= -\int_\Omega \left.\frac{\partial\Theta(\tau,s)}{\partial s}\right|_{s=-g(u,\xi)} \nabla_u g(u,\xi)\phi(\xi)d\xi.\end{aligned}$$

Note that:

- $\psi(\tau,u)$ and $\varphi(\tau,u)$ are differentiable irrespective of the differentiability of $p(\cdot)$
  $\Longrightarrow$ IA_τ **and** OA_τ **are smoothing approximations of CCOPT**.
- If p is differentiable, then $\nabla\psi(\tau,u)$ converges to $-\nabla p(u)$ and $\nabla\varphi(\tau,u)$ converges to $\nabla p(u)$.

(**For further details, see tutorial slides**)

4. The reduced-space method OA-IA problems

(a) Generate sufficient samples $\xi^{(1)}, \xi^{(2)}, \ldots, \xi^{(N)}$ for the random variable ξ. (Use any one of quasi Monte-Carlo, Latin Hyper Cube, or Sparse-grid samples, etc.)

(b) For a given u and the samples $\xi^{(1)}, \xi^{(2)}, \ldots, \xi^{(N)}$, solve the system of equations

$$G(x, u, \xi^{(i)}) = 0, i = 1, \ldots, N,$$

to obtain $x(u, \xi^{(i)}), i = 1, \ldots, N$.

(c) Efficiently evaluate multidimensional integrals associated with $E[f(x(u,\xi), u, \xi)]$, $\psi(\tau, u) = E[\Theta(\tau, g(x(u,\xi), u, \xi))]$ $\varphi(\tau, u) = E[\Theta(\tau, g(x(u,\xi), u, \xi))]$

(d) Efficiently evaluate the gradients $\nabla_u E[f(x(u,\xi), u, \xi)]$, $\nabla_u \psi(\tau, u) = \nabla_u E[\Theta(\tau, g(x(u,\xi), u, \xi))]$ $\nabla_u \varphi(\tau, u) = \nabla_u E[\Theta(\tau, g(x(u,\xi), u, \xi))]$

(Conditions guaranteeing to exchange operators $\nabla E((\cdot)) = \nabla E(\nabla(\cdot))$ are to be found in Geletu etal. 2015 and references therein.)

4. The reduced-space method OA-IA problems

Consider only the problem $(\mathrm{IA})_\tau$.
For a given u, let the values $x(u, \xi^{(i)}), i = 1, \ldots, N$ are available. Then we write for

- objective function

$$F(u) = E[f(x(u,\xi), u, \xi)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x(u,\xi^{(i)}), u, \xi^{(i)}) \tag{26}$$

- for a given $\tau \in (0,1)$, the constraint

$$\psi(\tau, u) \approx \frac{1}{N}\sum_{i=1}^{N} \Theta(\tau, g(x(u,\xi^{(i)}), u, \xi^{(i)})) - (1-\alpha) \tag{27}$$

- the gradient of objective function

$$\nabla F(u) = \nabla_u \left(E[f(x(u,\xi), u, \xi)]\right) \approx \frac{1}{N}\sum_{i=1}^{N} \left\{ \left[D_u x(u,\xi^{(i)})\right]^\top \nabla_x f(x(u,\xi^{(i)}), u, \xi^{(i)}) \right.$$
$$\left. + \nabla_u f(x(u,\xi^{(i)}), u, \xi^{(i)}) \right\} \tag{28}$$

4. The reduced-space method OA-IA problems

- gradient of the constraint function

$$\nabla_u \psi(\tau, u) \approx \frac{1}{N} \sum_{i=1}^{N} \Theta'(\tau, g(x(u, \xi^{(i)}), u, \xi^{(i)})) \times \tag{29}$$
$$\left\{ \left[D_u x(u, \xi^{(i)}) \right]^{\top} \nabla_x g(x(u, \xi^{(i)}), u, \xi^{(i)}) + \nabla_u g(x(u, \xi^{(i)}), u, \xi^{(i)}) \right\}$$

The derivative w.r.t. s of the scalar function Θ' (is easy) and

$$\Theta'(\tau, s) = \frac{m_2(1 + m_1\tau)exp(-\frac{s}{\tau})}{\tau[1 + m_2\tau exp(-\frac{s}{\tau})]^2}$$

4. A reduced space algorithm for the outer approximation

Algorithm 2: A reduced space method for $(\mathrm{IA})_{\tau_k}$ (τ_k fixed)

1: Choose a termination tolerance $\varepsilon > 0$ (e.g., $\varepsilon = 10E-7$, etc.) and small $\tau_k \in (0,1)$;
2: Generate sufficient samples $\xi^{(i)}$, $i = 1, \ldots, N$, from Ω according to $\phi(\xi)$;
3: Choose an initial iterate $u^{(0)}$;
4: Set $k \leftarrow 0$;
5: **while** (termination criteria not satisfied) **do**
6: Solve the system of equations

$$G_\ell(x, u^{(k)}, \xi^{(i)}) = 0,\ \ell = 1, \ldots, q;\ i = 1, \ldots, N, \tag{30}$$

for x to obtain $x^{(k)} = x(u^{(k)}, \xi^{(i)})$, $i = 1, \ldots, N$;
7: Determine $D_u x(u^{(k)}, \xi^{(i)})$, $i = 1, \ldots, N$ by using $x(u^{(k)}, \xi^{(i)})$, $i = 1, \ldots, N$, and solving the system

$$\left[D_u x(u^{(k)}, \xi^{(i)})\right]^\top \nabla_x G_\ell(x(u^{(k)}, \xi^{(i)}), u^{(k)}, \xi^{(i)}) + \nabla_u G_\ell(x(u^{(k)}, \xi^{(i)}), u^{(k)}, \xi^{(i)}) = 0,\ \ell = 1:q;\ i = 1:N. \tag{31}$$

8: Use the results from Steps 6 and 7 to obtain the function values $F(u^k)$ (from eqn. (26)) and $\psi(\tau_k, u^{(k)})$ (from eqn. (27)); the gradients $\nabla F(u^{(k)})$ (from eqn. (28)) and $\nabla \psi(\tau_k, u^{(k)})$ (from eqn. (29)).
9: Determine a search direction d_k and step-length α_k ;
10: Update the iterate

$$u^{(k+1)} = u^{(k)} + \alpha_k d_k$$

11: Set $k \leftarrow k + 1$;
12: **end while**

4. A reduced space algorithm ...

Remark:

- In Algorithm 2 is designed for a fixed (small) parameter $\tau_k \in (0,1)$.
- It is easy to adapt Algorithm 2 to the outer approximation problem $(\mathrm{OA})_{\tau_k}$ for a given $\tau_k \in (0,1)$.
- To guarantee the convergence of solutions of $(\mathrm{IA})_{\tau_k}$ and $(\mathrm{OA})_{\tau_k}$ to an approximate solution of CCOPT, we need to solve these problems for a sequence $\{\tau_k\}_{k\in\mathbb{N}}$, where $\tau_k \searrow 0^+$.

  $\Rightarrow$ we need an outer-loop w.r.t. τ_k over Algortihm 2 for both $(\mathrm{IA})_{\tau_k}$ and $(\mathrm{OA})_{\tau_k}$.

- we solve $(\mathrm{IA})_{\tau_k}$ and $(\mathrm{OA})_{\tau_k}$ repeatedly for decreasing values of τ_k, and we stop the outer-loop when the objective functions values of $(\mathrm{IA})_{\tau_k}$ and $(\mathrm{OA})_{\tau_k}$ are sufficiently close.

4. Algorithm for approximate solution of CCOPT

Algorithm 3: Inner-outer approximation

1: Choose an initial parameter $\tau_0 \in (0,1)$;
2: Solve the optimization problems $(\mathrm{IA})_{\tau_0}$ and $(\mathrm{OA})_{\tau_0}$;
3: Select the termination tolerance *tol*;
4: Set $k \leftarrow 0$
5: **while** $\left(\left|F_{IA}(u^*_{\tau_k}) - F_{OA}(\widehat{u}^*_{\tau_k})\right| > tol\right)$ **do**
6: Reduce the parameter τ_k (e.g., $\tau_{k+1} = \rho\tau_k$, for $\rho \in (0,1)$);
7: Set $k \leftarrow k+1$;
8: Solve the optimization problems $(\mathrm{IA})_{\tau_k}$ and $(\mathrm{OA})_{\tau_k}$ using Algorithm 2;
9: **end while**

5. Some comments on parallel implementation

What can implemented in parallel?

- In Algorithm 3, $(\mathrm{IA})_{\tau_k}$ and $(\mathrm{OA})_{\tau_k}$ can be solved in parallel.
- Observe that the solution of equations (30) and (31) can be parallelized for the samples $\xi^{(i)}, i = 1, \ldots, N$. Moreover, large-scale equations posses sparse and block-structured matrices which can be exploited for parallelization.
- The values $F(u^k)$, $\psi(\tau_k, u^{(k)})$,$\nabla F(u^{(k)})$, $\nabla\psi(\tau_k, u^{(k)})$ in Step 8 of Algorithm 2 can be obtained through parallel computation.
- Generally, use a combination of distributed memory (e.g., using MPI) and shared-memory parallel implementations.

The best way to learn is to do it yourself!

6. An exercise (adapted from the tutorial slides)

$$(P) \qquad \min_{u \in \mathbb{R}^2} \{ f(u) = \sqrt{u_1} + \sqrt{u_2} \} \tag{32}$$

subject to:

$$x_1 + \xi_1 x_1 u_1 = 1, \tag{33}$$
$$x_2 - x_1 + \xi_2 x_2 u_2 = 0, \tag{34}$$
$$x_3 + x_1 + \xi_3 x_3 u_1 = 1, \tag{35}$$
$$x_4 - x_3 + x_2 - x_1 + \xi_4 x_4 u_2 = 0, \tag{36}$$
$$Pr\{x_4 \geq x_{\min}\} \geq \alpha, \tag{37}$$
$$0 \leq u_1 \leq 16, \ 0 \leq u_2 \leq 16, \tag{38}$$

	Expected value	Standard deviation	Correlation matrix
ξ_1	0.715	0.0215	$\begin{bmatrix} 1 & 0.5 & 0.3 & 0.2 \\ 0.5 & 1 & 0.5 & 0.1 \\ 0.3 & 0.5 & 1 & 0.3 \\ 0.2 & 0.1 & 0.3 & 1 \end{bmatrix}$
ξ_2	0.182	0.0055	
ξ_3	6665.948	200	
ξ_4	7965.248	240	

Table: Mean, standard deviation and correlation matrix of the random variables (i.e., normal distribution)

6. An exercise ...

Exercises:

1. Write the reduced form of (P)
2. Set-up inner and outer approximation problems for (P)
3. Generate **sufficient and efficient samples** for $\xi_1, \ldots, \xi_4$ from $\mathbb{R}^4$ according to the normal distribution.

   Note:
   - Some sample generators may need de-correlation of the random variables.
   (see Geletu etal.(2011). Monotony analysis and sparse-grid integration for nonlinear chance constrained process optimization. Engineering Optimization, 43(10), 1019-1041.)
   - Mostly quasi Monte-Carlo samples (e.g., Sobol sequences).
   - Samples can be generated once and used repeatedly.
   - However, adaptively increasing samples may provide better results.
4. Solve the problems $(\mathrm{IA})_{\tau_k}$ and $(\mathrm{OA})_{\tau_k}$ (in parallel) for decreasing values of $\tau_k \in (0, 1)$. (e.g., use IpOpt).
5. Demonstrate graphically, how the optimal objective function values of $(\mathrm{IA})_{\tau_k}$ and $(\mathrm{OA})_{\tau_k}$ get closer with decreasing values of τ_k.

7. References

- Geletu, A., Hoffmann, A., Klöppel, M., Li, P., 2017. An inner-outer approximation approach to chance constrained optimization. SIAM Journal on Optimization, 27(3), 1834 - 1857.
- Geletu, A., Klöppel, M., Hoffmann, A., Li, P., 2015. A tractable approximation of nonconvex chance constrained optimization with non-Gaussian uncertainties. *Journal of Engineering Optimization*, 47(4), pp. 495 - 520.
- Geletu, A., Klöppel, M., Zhang, H., Li, P., 2012. Advances and applications of chance-constrained approaches to systems optimization under uncertainty. *International Journal of Systems Science*, 44(7): 1209–1232.
- Geletu, A., Li, P., 2014. Recent developments in computational approaches to optimization under uncertainty and application in process systems engineering, *ChemBioEng Reviews*, 1(4), 170–190.
- Geletu, A., Hoffmann, A., Klöppel, M., Li, P. 2011. Monotony analysis and sparse-grid integration for nonlinear chance-constrained chemical process optimizationa problems. *Engineering Optimization*, 43(10): 1019–1041.
- Klöppel, M., Geletu, A., Hoffmann, A., Li, P., (2011). Using sparse-grid methods to improve computation efficiency in solving dynamic nonlinear chance-constrained optimization problems. Industrial Engineering Chemical Research, 50, 5693-5704.
- Klöppel, M.: Efficient numerical solution of chance constrained optimization problems with engineering applications. Ph.D. Dissertation, Faculty of Mathematics and Natural Sciences, TU Ilmenau, Germany, 2014.
- Lazutkin, E.: Efficient solution approach to nonlinear optimal control problems and applications to autonomous driving. Ph.D. Dissertation, Faculty of Computer Science and Automation, TU Ilmenau, Germany, 2018.