Scenario Optimization for Robust Design

foundations and recent developments

Giuseppe Carlo Calafiore

Dipartimento di Elettronica e Telecomunicazioni
Politecnico di Torino – ITALY

Learning for Control Workshop
2018 IEEE Conference on Decision and Control
December 2018

Summary of contents

1. **Random Convex Programs (RCP)**
   - Preliminaries
   - Probabilistic properties of scenario solutions
   - Applications in control

2. **Repetitive Scenario Design (RSD)**
   - Iterating scenario design and feasibility checks
   - Example: robust finite-horizon input design

RCP theory
Introduction

- Random convex programs (RCPs) are *convex optimization problems subject to a finite number of constraints (scenarios) that are extracted according to some probability distribution.*
- The optimal objective value of an RCP and its associated optimal solution (when it exists), are random variables.
- RCP theory is mainly concerned with providing probabilistic assessments on the objective and on the probability of constraint violation for RCPs.
- We give a synthetic overview of RCP theory.
- Discuss impact and some applicative examples, with focus on control applications.

RCP theory
Preliminaries

- A finite-dimensional convex optimization problem

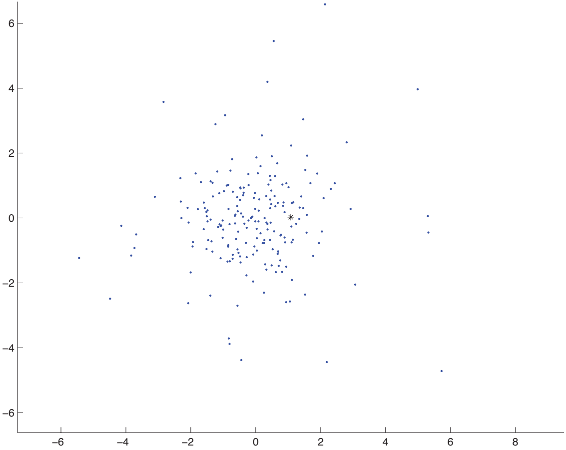
$$P[K]: \quad \min_{x \in \mathcal{X}} \; c^\top x \quad \text{subject to:} \qquad (1)$$
$$f_j(x) \leq 0, \qquad \forall j \in K,$$

  $x \in \mathcal{X}$ is the optimization variable, $\mathcal{X} \subset \mathbb{R}^d$ is a compact and convex domain, $c \neq 0$ is the objective direction, K is a *finite* set of indices, and $f_j(x) : \mathbb{R}^d \to \mathbb{R}$ are convex in x for each $j \in K$.

- Each constraint thus defines a convex set $\{x : f_j(x) \leq 0\}$.

A model paradigm

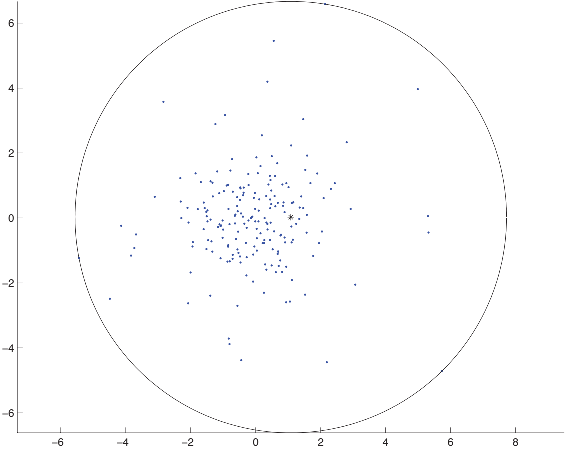
See N points



A model paradigm

See N points

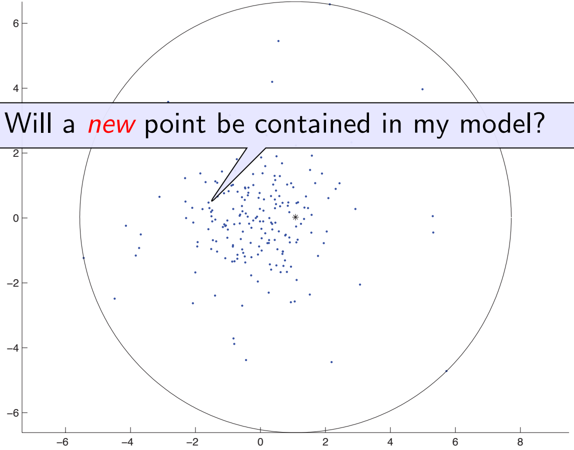
Fit model...



A model paradigm

See N points

Fit model...

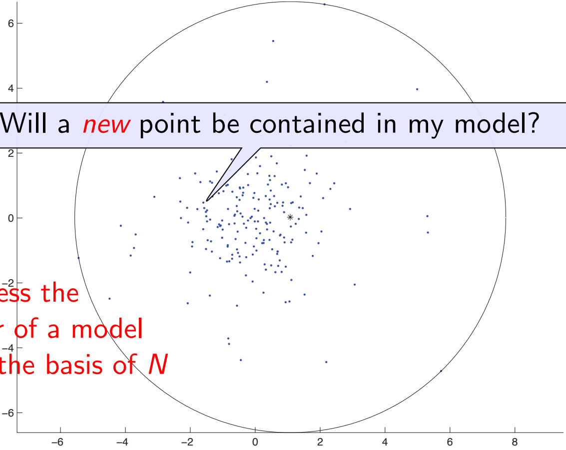


Will a *new* point be contained in my model?

A model paradigm

See N points

Fit model...



Will a *new* point be contained in my model?

We want to assess the predictive power of a model constructed on the basis of N *examples*...

Example model

- The variable is $x = (c, r)$, where $c \in \mathbb{R}^2$ is the center and $r \in \mathbb{R}$ is the radius of the circle (i.e., our "model").
- The (convex) problem we solve is:

$$\begin{aligned} \min_{(c,r)} \quad & r \\ \text{s.t.:} \quad & \|c - \delta^{(i)}\|_2 \leq r, \quad i = 1, \ldots, N, \end{aligned}$$

  where $\delta^{(1)}, \ldots, \delta^{(N)} \in \mathbb{R}^2$ are the N random points, coming from an unknown distribution.
- Let c^* and r^* be the optimal solutions obtained in an instance of the above problem...
- What is the probability that a new, unseen, random point, say δ, is "explained" by our model. That is, can we say something a-priori about

$$\mathbb{P}\{\|c^* - \delta\|_2 \leq r^*\}?$$

RCP theory
Formalization

- Let $\delta \in \Delta$ denote a vector of random parameters, with $\Delta \subseteq \mathbb{R}^\ell$, and let $\mathbb{P}$ be a probability measure on Δ.
- Let $x \in \mathbb{R}^d$ be a design variable, and consider a family of functions $f(x, \delta) : (\mathbb{R}^d \times \Delta) \to \mathbb{R}$ defining the design constraints and parameterized by δ.

  Specifically, for a given design vector x and realization δ of the uncertainty, the design constraint are satisfied if $f(x, \delta) \leq 0$.

Assumption (convexity)

The function $f(x, \delta) : (\mathbb{R}^d \times \Delta) \to \mathbb{R}$ is convex *in x, for each fixed $\delta \in \Delta$.*

RCP theory
Formalization

- Define
$$\omega \doteq (\delta^{(1)}, \ldots, \delta^{(N)}) \in \Delta^N,$$
where $\delta^{(i)} \in \Delta$, $i = 1, \ldots, N$, are independent random variables, identically distributed (iid) according to $\mathbb{P}$, and where $\Delta^N = \Delta \times \Delta \cdots \Delta$ (N times).
- Let $\mathbb{P}^N$ denote the product probability measure on Δ^N.
- To each $\delta^{(j)}$ we associate a constraint function
$$f_j(x) \doteq f(x, \delta^{(j)}), \quad j = 1, \ldots, N.$$
Therefore, to each randomly extracted ω there correspond N random constraints $f_j(x)$, $j = 1, \ldots, N$.

RCP theory
Formalization

- Given $\omega = (\delta^{(1)}, \dots, \delta^{(N)}) \in \Delta^N$ we define the following convex optimization problem:

$$P[\omega]: \quad \min_{x \in \mathcal{X}} \; c^\top x \quad \text{subject to:} \qquad (2)$$
$$f_j(x) \le 0, \qquad j = 1, \dots, N,$$

  where $f_j(x) = f(x, \delta^{(j)})$.

- For each random extraction of ω, problem (2) has the structure of a generic convex optimization problem $P[\omega]$, as defined in (1).
- We denote with $J^* = J^*(\omega)$ the optimal objective value of $P[\omega]$, and with $x^* = x^*(\omega)$ the optimal solution of problem (2), when it exists.
- Problem (2) is named a *random convex program* (RCP), and the corresponding optimal solution x^* is named a *scenario solution*.

RCP theory

Remarks on the generality of the model

Model (2) encloses a quite general family of uncertain convex programs.

- Problems with multiple uncertain (convex) constraints of the form

$$\min_{x \in \mathcal{X}} \; c^\top x \quad \text{subject to:}$$
$$f^{(1)}(x, \delta^{(j)}) \leq 0, \ldots, f^{(m)}(x, \delta^{(j)}) \leq 0;$$
$$j = 1, \ldots, N,$$

  can be readily cast in the form of (2) by condensing the multiple constraints in a single one:

$$f(x, \delta) \doteq \max_{i=1,\ldots,m} f^{(i)}(x, \delta).$$

- The case when the problem has an uncertain and nonlinear (but convex) objective function $g(x, \delta)$ can also be fit in the model by adding one slack decision variable t and reformulating the problem with linear objective t and an additional constraint $g(x, \delta) - t \leq 0$.

Violation probability

Definition (Violation probability)

The violation probability of problem $P[\omega]$ is defined as

$$V^*(\omega) \doteq \mathbb{P}\{\delta \in \Delta : J^*(\omega, \delta) > J^*(\omega)\}.$$

- To each random extraction of $\omega \in \Delta^N$ it corresponds a value of V^*, which is therefore itself a random variable with values in $[0, 1]$.
- For given $\epsilon \in (0, 1)$, let us define the "bad" event of having a violation larger than ϵ:
$$\mathcal{B} \doteq \{\omega \in \Delta^N : V^* > \epsilon\}$$
- We prove that it holds that $\mathbb{P}^N\{\mathcal{B}\} \leq \beta(\epsilon)$, for some explicitly given function $\beta(\epsilon)$ that goes to zero as N grows.
- In other words, if N is large enough, the scenario objective is a-priori guaranteed with probability at least $1 - \beta(\epsilon)$ to have violation probability smaller than ϵ.

RCP theory
Technical hypotheses

- When problem $P[\omega]$ admits an optimal solution, this solution is *unique.*

- Problem $P[\omega]$ is "nondegenerate" with probability one. This essentially requires that the constraints are in "general position."

...both these technical conditions can be lifted.

RCP theory
Main result

Theorem

- *Consider problem (2), with $N \geq d+1$. Let the above Hp. hold, and*

$$V^*(\omega) \doteq \mathbb{P}\{\delta \in \Delta : J^*(\omega, \delta) > J^*(\omega)\}.$$

- *Then,*

$$\mathbb{P}^N\{\omega \in \Delta^N : V^*(\omega) > \epsilon\} \leq \Phi(\epsilon; d, N)$$

where

$$\Phi(\epsilon; d, N) \doteq \sum_{j=0}^{d} \binom{N}{j} \epsilon^j (1-\epsilon)^{N-j}$$

RCP theory
Main result

Theorem

- *Consider problem (2), with $N \geq d+1$. Let the above Hp. hold, and*

$$V^*(\omega) \doteq \mathbb{P}\{\delta \in \Delta : J^*(\omega,\delta) > J^*(\omega)\}.$$

- *Then,*

$$\mathbb{P}^N\{\omega \in \Delta^N : V^*(\omega) > \epsilon\} \leq \Phi(\epsilon; d, N)$$

where

$$\Phi(\epsilon; d, N) \doteq \sum_{j=0}^{d} \binom{N}{j} \epsilon^j (1-\epsilon)^{N-j}$$

The proof of this result is far from obvious...

Remark

Beta distribution

- Considering the complementary event $V^*(\omega) \leq \epsilon$, we have a upper bound on the cumulative distribution function of $V^*(\omega)$:

$$\mathbb{P}^N\{V^*(\omega) \leq \epsilon\} \geq 1 - \Phi(\epsilon; d, N)$$

- $\Phi(\epsilon; d, N)$ is the cumulative distribution of a beta random variable:

$$\Phi(\epsilon; d, N) = \int_0^\epsilon \mathrm{beta}(x; d+1, N-d)\mathrm{d}x,$$

where

$$\mathrm{beta}(x; d+1, N-d) = \frac{1}{B(d+1, N-d)} x^d (1-x)^{N-d-1}.$$

Remark

$$\mathbb{P}^N\{\omega \in \Delta^N : \; V^*(\omega) > \epsilon\} \leq \Phi(\epsilon; d, N)$$

$$\Phi(\epsilon; d, N) \doteq \sum_{j=0}^{d} \binom{N}{j} \epsilon^j (1-\epsilon)^{N-j}$$

This bound is UNIVERSAL:

- Does not depend on problem type (LP, QP, SDP, generic convex);
- Does not depend on the *distribution* law $\mathbb{P}$ of the uncertain parameters;
- Depends on the problem structure only via the *dimension*, d;
- Provides an explicit assessment on the violation probability tail, for *finite* N.

Learning-theoretic flavor: "training" on a finite batch of samples N provides a solution which is still optimal, with high probability, on a new, unseen, scenario...

Remark

$$\mathbb{P}^N\{\omega \in \Delta^N : V^*(\omega) > \epsilon\} \leq \Phi(\epsilon; d, N)$$

$$\Phi(\epsilon; d, N) \doteq \sum_{j=0}^{d} \binom{N}{j} \epsilon^j (1-\epsilon)^{N-j}$$

This bound is UNIVERSAL:

- Does not depend on problem type (LP, QP, SDP, generic convex);
- Does not depend on the *distribution* law $\mathbb{P}$ of the uncertain parameters;
- Depends on the problem structure only via the *dimension*, d;
- Provides an explicit assessment on the violation probability tail, for *finite* N.

Learning-theoretic flavor: "training" on a finite batch of samples N provides a solution which is still optimal, with high probability, on a new, unseen, scenario...

Reversing the bound

Corollary

Given $\epsilon \in (0,1)$, $\beta \in (0,1)$. *If* N *is an integer such that*

$$N \geq \frac{2}{\epsilon}\left(\ln \beta^{-1} + d\right).$$

then it holds that

$$\mathbb{P}^N\{V^* > \epsilon\} \leq \beta.$$

Observe that β^{-1} is under a log: achieving small β is "cheap" in terms of the required number of samples N.

Reversing the bound

Corollary

Given $\epsilon \in (0,1)$, $\beta \in (0,1)$. *If* N *is an integer such that*

$$N \geq \frac{2}{\epsilon}\left(\ln \beta^{-1} + d\right).$$

then it holds that

$$\mathbb{P}^N\{V^* > \epsilon\} \leq \beta.$$

Observe that β^{-1} is under a log: achieving small β is "cheap" in terms of the required number of samples N.

A Practitioner's viewpoint

- Set β to a very small level, say $\beta = 10^{-10}$
- Bound becomes
$$N \geq \frac{2}{\epsilon}(21 + d).$$
- The event $\{V^* > \epsilon\}$ has vanishing probability $\leq 10^{-10}$, that is, the complementary event $\{V^* \leq \epsilon\}$ holds with *practical certainty*.
- Scenario optimization guarantees, with practical certainty, that
$$V^* \leq \epsilon.$$
- These statements are more easily understandable by engineers. The neglected event is so remote that before worrying about it the designer should better check many other simplifying assumptions and uncertainties on her model...

A Practitioner's viewpoint

- Set β to a very small level, say $\beta = 10^{-10}$
- Bound becomes

$$N \geq \frac{2}{\epsilon}(21 + d).$$

- The event $\{V^* > \epsilon\}$ has vanishing probability $\leq 10^{-10}$, that is, the complementary event $\{V^* \leq \epsilon\}$ holds with *practical certainty*.
- Scenario optimization guarantees, with practical certainty, that

$$V^* \leq \epsilon.$$

- These statements are more easily understandable by engineers. The neglected event is so remote that before worrying about it the designer should better check many other simplifying assumptions and uncertainties on her model...

Ok... so why all this may be interesting in control applications?

A Practitioner's viewpoint

- Set β to a very small level, say $\beta = 10^{-10}$
- Bound becomes
$$N \geq \frac{2}{\epsilon}\left(21 + d\right).$$
- The event $\{V^* > \epsilon\}$ has vanishing probability $\leq 10^{-10}$, that is, the complementary event $\{V^* \leq \epsilon\}$ holds with *practical certainty*.
- Scenario optimization guarantees, with practical certainty, that
$$V^* \leq \epsilon.$$
- These statements are more easily understandable by engineers. The neglected event is so remote that before worrying about it the designer should better check many other simplifying assumptions and uncertainties on her model...

Ok... so why all this may be interesting in control applications?

Let's see an example...

Example

Robust Model Predictive Control

A discrete-time LTI system

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \quad x(0) = x_0 \\ y(k) &= Cx(k) \end{aligned}$$

Determine a sequence of control actions $u(0), u(1), \ldots, u(T-1)$, such that a suitable performance index is minimized over a finite horizon T, while satisfying a given set of constraints on the input and output signals:

$$\begin{array}{lll} \min & \gamma & \\ \text{s.t.:} & J(u(0), u(1), \ldots, u(T-1)) \leq \gamma & \\ & y_{\min} \leq y(k) \leq y_{\max}, & k = 1, \ldots, T \\ & u_{\min} \leq u(k) \leq u_{\max}, & k = 0, \ldots, T-1, \end{array}$$

where

$$J(u(0), u(1), \ldots, u(T-1)) = \sum_{k=0}^{T-1} \left(x^\top(k) Q x(k) + u^\top(k) R u(k) \right) + x^\top(T) P x(T),$$

with Q, R, P given positive definite matrices.

Example

Robust Model Predictive Control

- We consider an important variation on the problem, where the system matrices $A(\delta), B(\delta), C(\delta)$ are nonlinear functions of an uncertainty vector of random parameters $\delta \in \Delta$. The constraints in the problem need then be enforced in some "robust" sense.
- In a probabilistic approach, we ask that the command and output constraints are met with high probability, that is for most (if not all) possible realization of δ.
- Let $\theta \doteq [u^\top(0)\, u^\top(1) \cdots u^\top(T-1)]^\top$, we rewrite the constraints as

$$f(\theta, \delta) \doteq \max\{J - \gamma, \max_{k=1,\dots,T}\{y(k) - y_{\max}, y_{\min} - y(k)\}, \\ \max_{k=0,\dots,T-1}\{u(k) - y_{\max}, u_{\min} - u(k)\}\}.$$

Example

Robust Model Predictive Control

- We consider an important variation on the problem, where the system matrices $A(\delta), B(\delta), C(\delta)$ are nonlinear functions of an uncertainty vector of random parameters $\delta \in \Delta$. The constraints in the problem need then be enforced in some "robust" sense.
- In a probabilistic approach, we ask that the command and output constraints are met with high probability, that is for most (if not all) possible realization of δ.
- Let $\theta \doteq [u^\top(0)\, u^\top(1) \cdots u^\top(T-1)]^\top$, we rewrite the constraints as

$$
\begin{aligned}
f(\theta, \delta) \doteq \quad & \max \{ J - \gamma, \max_{k=1,\ldots,T} \{ y(k) - y_{\max}, y_{\min} - y(k) \}, \\
& \max_{k=0,\ldots,T-1} \{ u(k) - y_{\max}, u_{\min} - u(k) \} \} \,.
\end{aligned}
$$

Example
Robust Model Predictive Control

- We consider an important variation on the problem, where the system matrices $A(\delta), B(\delta), C(\delta)$ are nonlinear functions of an uncertainty vector of random parameters $\delta \in \Delta$. The constraints in the problem need then be enforced in some "robust" sense.
- In a probabilistic approach, we ask that the command and output constraints are met with high probability, that is for most (if not all) possible realization of δ.
- Let $\theta \doteq [u^\top(0)\, u^\top(1) \cdots u^\top(T-1)]^\top$, we rewrite the constraints as

$$\begin{aligned} f(\theta,\delta) \doteq \quad & \max\{J-\gamma, \max_{k=1,\ldots,T}\{y(k)-y_{\max}, y_{\min}-y(k)\}, \\ & \max_{k=0,\ldots,T-1}\{u(k)-y_{\max}, u_{\min}-u(k)\}\}\,. \end{aligned}$$

Example
Robust Model Predictive Control

- Define the probability of violation for the constraints at θ as

$$V(\theta) = \mathbb{P}\{\delta \in \Delta : f(\theta, \delta) > 0\}.$$

- Then, fixing a probability level $\epsilon \in (0, 1)$, we say that the control sequence θ is a probabilistically feasible control to level ϵ, if it satisfies $V(\theta) \leq \epsilon$.

- The RCP technology can then be used effectively to determine such a probabilistically robust control law.

Example
Robust Model Predictive Control

- Define the probability of violation for the constraints at θ as

$$V(\theta) = \mathbb{P}\{\delta \in \Delta : f(\theta, \delta) > 0\}.$$

- Then, fixing a probability level $\epsilon \in (0, 1)$, we say that the control sequence θ is a probabilistically feasible control to level ϵ, if it satisfies $V(\theta) \leq \epsilon$.

- The RCP technology can then be used effectively to determine such a probabilistically robust control law.

Example
Robust Model Predictive Control

- Define the probability of violation for the constraints at θ as

$$V(\theta) = \mathbb{P}\{\delta \in \Delta : f(\theta, \delta) > 0\}.$$

- Then, fixing a probability level $\epsilon \in (0, 1)$, we say that the control sequence θ is a probabilistically feasible control to level ϵ, if it satisfies $V(\theta) \leq \epsilon$.

- The RCP technology can then be used effectively to determine such a probabilistically robust control law.

Repetitive scenario design (RSD)
Introduction

Repetitive scenario design (RSD)

Repetitive scenario design (RSD)

- Repetitive Scenario Design (RSD) is a randomized approach to robust design based on iterating two phases:
  - a standard scenario design phase that uses N scenarios (design samples), followed by
  - a randomized feasibility test phase that uses N_o test samples on the scenario solution.
- The above two steps are repeated until the desired level of probabilistic feasibility is eventually obtained.
- In the following, we assume that the scenario problem is feasible w.p. one and it attains a unique optimal solution θ^*.

Repetitive scenario design (RSD)

- This novel approach broadens the applicability of the scenario technology, since the user is now presented with a clear tradeoff between the number N of design samples and the ensuing expected number of repetitions required by the RSD algorithm.

- The plain (one-shot) scenario design becomes just one of the possibilities, sitting at one extreme of the tradeoff curve, in which one insists in finding a solution in a single repetition: this comes at the cost of possibly high N.
- Other possibilities along the tradeoff curve use lower N values, but possibly require more than one repetition.

Repetitive scenario design (RSD)
Idea

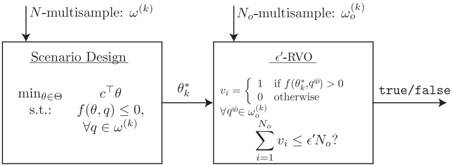
- Each time we solve a scenario problem with N scenarios, we "toss a coin." The toss is successful if $V(\theta^*) \leq \epsilon$, while it fails if $V(\theta^*) > \epsilon$.
- The a-priori probability of success in a coin toss is $\geq 1 - \beta_\epsilon(N)$, where

$$\beta_\epsilon(N) = \Phi(\epsilon; d, N) \doteq \sum_{j=0}^{d} \binom{N}{j} \epsilon^j (1-\epsilon)^{N-j}$$

- Plain scenario technology works by selecting N such that $\beta_\epsilon(N)$ is very small (say, $\leq 10^{-9}$).
- This means biasing the coin so to be successful with practical certainty (i.e., w.p. $\geq 1 - 10^{-9}$) in one single coin toss!
- Success in one toss comes at the price of possibly high N...

Repetitive scenario design (RSD)
Idea

- What if we use a lower N (i.e., we bias the coin with higher $\beta_\epsilon(N)$) and then check the resulting solution?
- Idea: while the probability of being successful in one shot is low, if we toss the coin repeatedly, the probability of being successful at some toss becomes arbitrarily high...
- We thus set up an iterative approach in two stages: a scenario optimization stage, and a feasibility check phase.



Repetitive scenario design (RSD)
Ideal oracle

- We first assume we have an ideal feasibility oracle, called a Deterministic Violation Oracle (DVO), that returns `true` if $V(\theta^*) \leq \epsilon$ and `false` otherwise.
- We apply the following algorithm:

Algorithm (RSD with ϵ-DVO)
Input data: integer $N \geq n$, level $\epsilon \in [0, 1]$.
Output data: solution θ^*. Initialization: set iteration counter to $k = 1$.

1. (Scenario step) Generate N i.i.d. samples $\omega^{(k)} \doteq \{q_k^{(1)}, \ldots, q_k^{(N)}\}$ according to $\mathbb{P}$, and solve scenario problem. Let θ_k^* be the resulting optimal solution.
2. (ϵ-DVO step) If $V(\theta_k^*) \leq \epsilon$, then set flag to `true`, else set it to `false`.
3. (Exit condition) If flag is `true`, then exit and return current solution $\theta^* \leftarrow \theta_k^*$; else set $k \leftarrow k + 1$ and goto 1.

Repetitive scenario design (RSD)

Ideal oracle

Theorem

Given $\epsilon \in [0, 1]$ and $N \geq n$, define the running time K of the algorithm with DVO as the value of the iteration counter k when the algorithm exits. Then:

1. *The solution θ^* returned by the algorithm is an ϵ-probabilistic robust design, i.e., $V(\theta^*) \leq \epsilon$.*
2. *The expected running time of the algorithm is $\leq (1 - \beta_\epsilon(N))^{-1}$.*
3. *The running time of the algorithm is $\leq k$ with probability $\geq 1 - \beta_\epsilon(N)^k$.*

Repetitive scenario design (RSD)
Randomized oracle

- Since the ideal oracle is hardly realizable in practice, we next introduce a Randomized Violation Oracle (RVO):

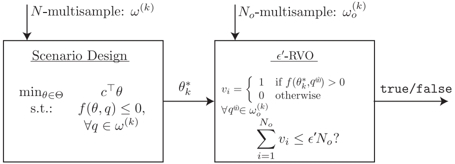
ϵ'-**RVO** (*Randomized* ϵ'*-violation oracle*)
Input data: integer N_o, level $\epsilon' \in [0,1]$, and $\theta \in \mathbb{R}^n$. Output data: a logic flag, `true` or `false`.

1. Generate N_o i.i.d. samples $\omega_o \doteq \{q^{(1)}, \ldots, q^{(N_o)}\}$, according to $\mathbb{P}$.
2. For $i = 1, \ldots, N_o$, let $v_i = 1$ if $f(\theta, q^{(i)}) > 0$ and $v_i = 0$ otherwise.
3. If $\sum_i v_i \leq \epsilon' N_o$, return `true`, else return `false`.

Repetitive scenario design (RSD)
Randomized oracle

Algorithm (RSD with ϵ'-RVO)
Input data: integers N, N_o, level $\epsilon' \in [0,1]$. Output data: solution θ^*.
Initialization: set iteration counter to $k = 1$.

1. (Scenario step) Generate N i.i.d. samples $\omega^{(k)} \doteq \{q_k^{(1)}, \ldots, q_k^{(N)}\}$ according to $\mathbb{P}$, and solve scenario problem. Let θ_k^* be the resulting optimal solution.
2. (ϵ'-RVO step) Call the ϵ'-RVO with current θ_k^* as input, and set flag to `true` or `false` according to the output of the ϵ'-RVO.
3. (Exit cond.) If flag is `true`, then exit and return current solution $\theta^* \leftarrow \theta_k^*$; else set $k \leftarrow k + 1$ and goto 1.



Repetitive scenario design (RSD)
Randomized oracle

Theorem (RSD with ϵ'-RVO)
Let $\epsilon, \epsilon' \in [0,1]$, $\epsilon' \leq \epsilon$, and $N \geq n$ be given. Define the event $\mathrm{BadExit}$ in which the algorithm exits returning a "bad" solution θ^*:

$$\mathrm{BadExit} \doteq \{\text{algorithm returns } \theta^*\colon\ V(\theta^*) > \epsilon\}.$$

The following statements hold.

1. $$\mathbb{P}\{\mathrm{BadExit}\} \leq \frac{\mathrm{Fbeta}((1-\epsilon')N_o, \epsilon' N_o + 1; 1-\epsilon)}{1 - H_{1,\epsilon'}(N, N_o)} \beta_\epsilon(N).$$
2. The expected running time of the algorithm is $\leq (1 - H_{1,\epsilon'}(N, N_o))^{-1}$.
3. The running time of the algorithm is $\leq k$ with probability $\geq 1 - H_{1,\epsilon'}(N, N_o)^k$.

Here, Fbeta denotes the cumulative distribution of a beta density, and $H_{1,\epsilon'}(N, N_o)$ has an explicit expression in terms of beta-Binomial distributions.

Repetitive scenario design (RSD)
Randomized oracle

- A key quantity related to the expected running time of the algorithm is $H_{1,\epsilon'}(N, N_o)$, which is the upper tail of a beta-Binomial distribution.
- It is useful to have a more manageable approximate expression:

Corollary

For $N_o \to \infty$ it holds that $H_{1,\epsilon'}(N, N_o) \to \beta_{\epsilon'}(N)$.

- For large N_o, and $\epsilon' \leq \epsilon$, we have $H_{1,\epsilon'}(N, N_o) \simeq \beta_{\epsilon'}(N) \geq \beta_{\epsilon}(N)$, whence

$$\hat{K} \doteq \frac{1}{1 - H_{1,\epsilon'}(N, N_o)} \simeq \frac{1}{1 - \beta_{\epsilon'}(N)} \geq \frac{1}{1 - \beta_{\epsilon}(N)}.$$

This last equation gives us an approximate, asymptotic, expression for the upper bound $\hat{K}$ on the expected running time of the algorithm.

Example
Robust finite-horizon input design

- We consider a system of the form

$$x(t+1) = A(q)x(t) + Bu(t), \quad t = 0, 1, \ldots; \; x(0) = 0,$$

where $u(t)$ is a scalar input signal, and $A(q) \in \mathbb{R}^{n_a, n_a}$ is an interval uncertain matrix of the form

$$A(q) = A_0 + \sum_{i,j=1}^{n_a} q_{ij} e_i e_j^\top, \quad |q_{ij}| \leq \rho, \; \rho > 0,$$

where e_i is a vector of all zeros, except for a one in the i-th entry.

- Given a final time $T \geq 1$ and a target state $\bar{x}$, the problem is to determine an input sequence $\{u(0), \ldots, u(T-1)\}$ such that *(i)* the state $x(T)$ is robustly contained in a small ball around the target state $\bar{x}$, and *(ii)* the input energy $\sum_k u(k)^2$ is not too large.

Example

Robust finite-horizon input design

- We write $x(T) = x(T;q) = \mathcal{R}(q)u$, where $\mathcal{R}(q)$ is the T-reachability matrix of the system (for a given q), and $u \doteq (u(0), \ldots, u(T-1))$.
- We formally express our design goals in the form of minimization of a level γ such that

$$\|x(T;q) - \bar{x}\|_2^2 + \lambda \sum_{t=0}^{T-1} u(t)^2 \leq \gamma,$$

where $\lambda \geq 0$ is a tradeoff parameter. Letting $\theta = (u, \gamma)$, the problem is formally stated in our framework by setting

$$f(\theta, q) \leq 0, \quad \text{where } f(\theta, q) \doteq \|\mathcal{R}(q)u - \bar{x}\|_2^2 + \lambda\|u\|_2^2 - \gamma.$$

Example

Robust finite-horizon input design

- Assuming that the uncertain parameter q is random and uniformly distributed in the hypercube $\mathbb{Q} = [-\rho, \rho]^{n_a \times n_a}$, our scenario design problem takes the following form:

$$\begin{aligned} \min_{\theta=(u,\gamma)} \quad & \gamma \\ \text{s.t.:} \quad & f(\theta, q^{(i)}) \leq 0, \quad i = 1, \ldots, N. \end{aligned}$$

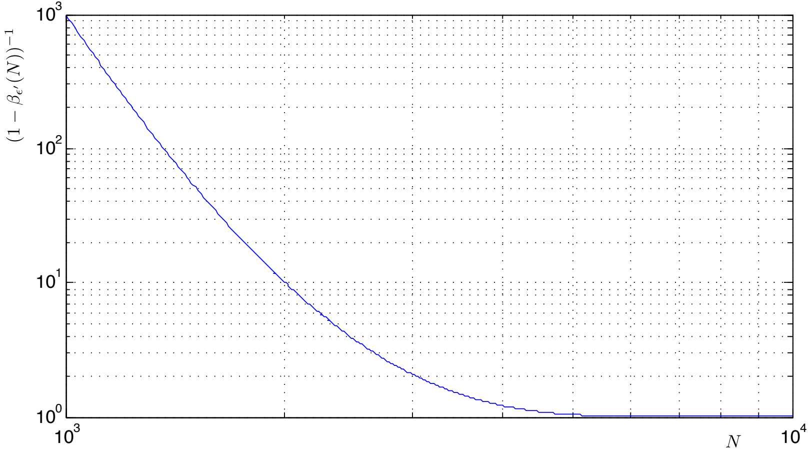
- We set $T = 10$, thus the size of the decision variable $\theta = (u, \gamma)$ of the scenario problem is $n = 11$.
- We set the desired level of probabilistic robustness to $1 - \epsilon = 0.995$, i.e., $\epsilon = 0.005$, and require a level of failure of the randomized method below $\beta = 10^{-12}$, that is, we require the randomized method to return a good solution with "practical certainty."

Example

Robust finite-horizon input design

- Using a plain (one-shot) scenario approach, imposing $\beta_\epsilon(N) \leq \beta$ would require $N \geq 10440$ scenarios.
- Let us now see how we can reduce this N figure by resorting to a repetitive scenario design approach.
- Let us fix $\epsilon' = 0.7\epsilon = 0.0035$, thus $\delta = \epsilon - \epsilon' = 0.0015$.
- A plot of the (asymptotic) bound on expected number of iterations, $(1 - \beta_{\epsilon'}(N))^{-1}$, as a function of N is shown in the next figure. We see from this plot, for instance, that the choice $N = 2000$ corresponds to a value of about 10 for the upper bound on the expected number of iterations.
- Let us choose this value of $N = 2000$ for the scenario block.

Example
Robust finite-horizon input design



Log-log plot of $(1-\beta_{\epsilon'}(N))^{-1}$ vs. N.

Example

Robust finite-horizon input design

- For $N = 2000$ a reliability level $\beta = 10^{-12}$ is achieved for $N_o \geq 62403$. Let us then choose $N_o = 63000$ samples to be used in the oracle.
- With the above choices we have $H_{1,\epsilon'}(N, N_o) = 0.8963$, thus the algorithm's upper bound on average running time is

$$\hat{K} = (1 - H_{1,\epsilon'}(N, N_o))^{-1} = 9.64.$$

- Notice that this upper bound is conservative (worst case) in general. Thus, we may expect a performance which is in practice better than the one predicted by the bound.

Example

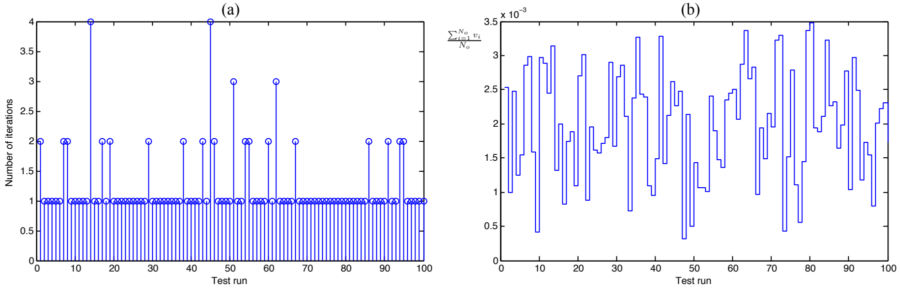
Robust finite-horizon input design

- We considered the following data:

$$A_0 = \begin{bmatrix} -0.7214 & -0.0578 & 0.2757 & 0.7255 & 0.2171 & 0.3901 \\ 0.5704 & 0.1762 & 0.3684 & -0.0971 & 0.6822 & -0.5604 \\ -1.3983 & -0.1795 & 0.1511 & 1.0531 & -0.1601 & 0.9031 \\ -0.6308 & -0.0058 & 0.4422 & 0.8169 & 0.512 & 0.2105 \\ 0.7539 & 0.1423 & 0.2039 & -0.3757 & 0.5088 & -0.6081 \\ -1.3571 & -0.1769 & 0.1076 & 1.0032 & -0.1781 & 0.9151 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

- We set target state $\bar{x} = [1, -1/2, 2, 1, -1, 2]^\top$, $\rho = 0.001$, and $\lambda = 0.005$.
- We run the RSD algorithm for 100 times, and on each test run we recorded the number of iterations and the solution returned upon exit. The algorithm exited most of the times in a single repetition, with a maximum of 4 repetitions.

Example
Robust finite-horizon input design



(a) Repetitions of RSD algorithm in the 100 test runs.
(b) Levels of empirical violation probability evaluated by the oracle upon exit, in the 100 test runs.

Example
Robust finite-horizon input design

Computational improvements

- Substantial reduction of the number of design samples (from the 10440 to just 2000), at the price of a very moderate number of repetitions (the average number of repetitions in the 100 test runs was 1.27).
- On average over the 100 test experiments, the RSD method (with $N = 2000$, $N_o = 63000$) required 224 s to return a solution.
- A plain, one-shot, scenario optimization with $N = 10440$ scenarios required 2790 s. Using the RSD approach instead of a plain one-shot scenario design thus yielded a reduction in computing time of about one order of magnitude.
- The reason for this improvement is due to the fact that the scenario optimization problem in the RSD approach (which uses $N = 2000$ scenarios) took about 173 s to be solved on a typical run, and the subsequent randomized oracle test (with $N_o = 63000$) is computationally cheap, taking only about 3.16 s.

Conclusions

- Scenario design is a flexible technology that permits attacking a class of robust design problems that are hard to deal with via conventional deterministic methods.
- Widely used in control design. Recently became particularly popular in Model Predictive Control.
- Interesting data-driven approaches in many other domains (e.g., computational finance).
- The repetitive approach further broadens the applicability of scenario design to problems in which dealing with "large N" may be a problem in practice (e.g., robust SDP problems).

THANK YOU!

Beta and Beta-Binomial distributions

- We denote by $\mathrm{beta}(\alpha, \beta)$ the beta density function with parameters $\alpha > 0$, $\beta > 0$:

$$\mathrm{beta}(\alpha, \beta; t) \doteq \frac{1}{B(\alpha, \beta)} t^{\alpha - 1}(1 - t)^{\beta - 1}, \quad t \in [0, 1],$$

where $B(\alpha, \beta) \doteq \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$, and Γ is the Gamma function (for α, β integers, it holds that $B(\alpha, \beta)^{-1} = \alpha\binom{\alpha+\beta-1}{\beta-1}$).

- We denote by $\mathrm{Fbeta}(\alpha, \beta)$ the cumulative distribution function of the $\mathrm{beta}(\alpha, \beta)$ density:

$$\mathrm{Fbeta}(\alpha, \beta; t) \quad \doteq \quad \int_0^t \mathrm{beta}(\alpha, \beta; \vartheta)\mathrm{d}\vartheta, \quad t \in [0, 1].$$

$\mathrm{Fbeta}(\alpha, \beta; t)$ is the regularized incomplete beta function, and a standard result establishes that, for α, β integers, it holds that

$$\mathrm{Fbeta}(\alpha, \beta; t) = \sum_{i=\alpha}^{\alpha+\beta-1} \binom{\alpha + \beta - 1}{i} t^i (1 - t)^{\alpha+\beta-1-i}.$$

- The number x of successes in d independent Bernoulli trials each having success probability t is a random variable with Binomial distribution

$$\mathbb{P}\{x \leq z\} \quad = \quad \sum_{i=0}^{\lfloor z \rfloor} \binom{d}{i} t^i (1 - t)^{d-i} \leq \mathrm{Fbeta}(d - z, z + 1; 1 - t) = 1 - \mathrm{Fbeta}(z + 1, d - z; t).$$

- The number x of successes in d binary trials, where each trial has success probability t, and t is itself a random variable with $\mathrm{beta}(\alpha, \beta)$ distribution, is a random variable with so-called beta-Binomial density: for $i = 0, 1, \ldots, d$,

$$f_{\mathrm{bb}}(d, \alpha, \beta; i) \doteq \binom{d}{i} \frac{B(i + \alpha, d - i + \beta)}{B(\alpha, \beta)}.$$