OPTIMIZATION TECHNIQUES

Robust optimization

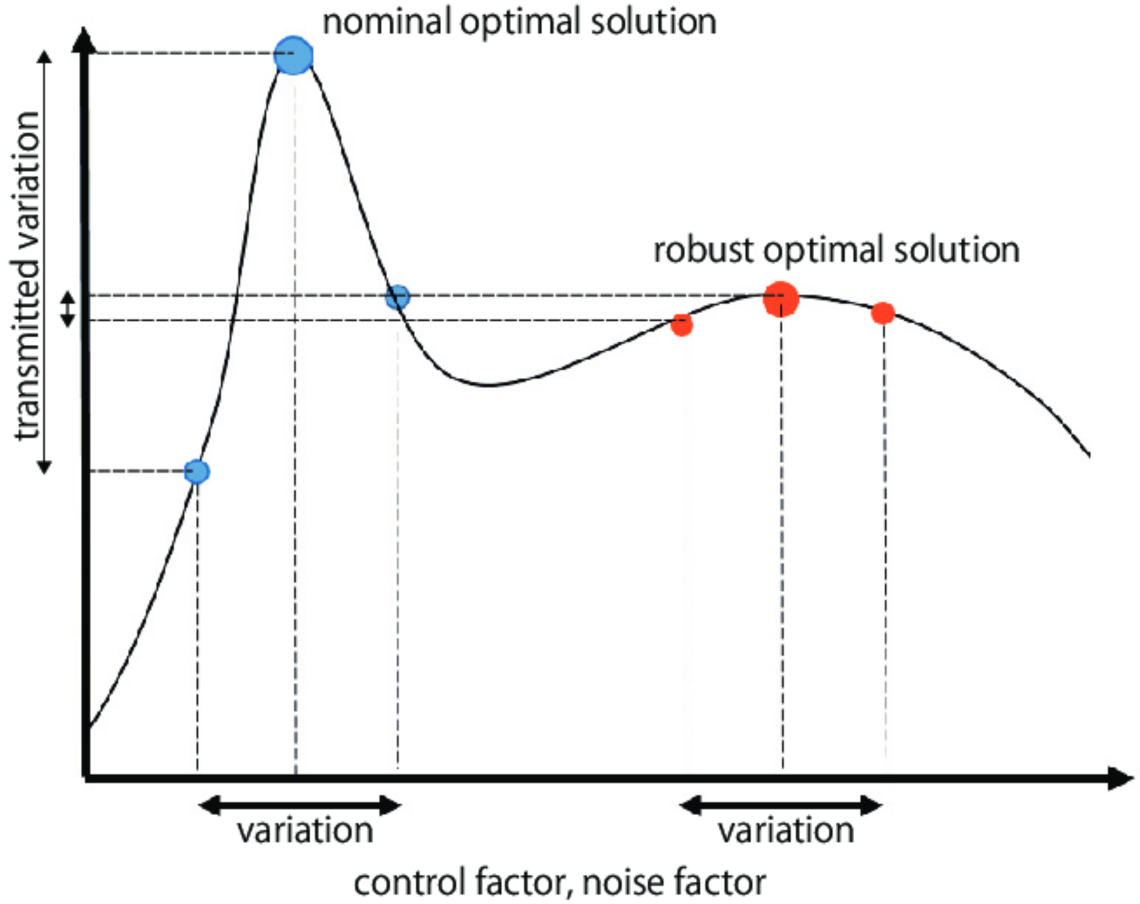
Prof. Giovanni Iacca

giovanni.iacca@unitn.it

UNIVERSITY OF TRENTO - Italy

Information Engineering
and Computer Science Department

Robust optimization



ROBUST OPTIMIZATION

INTRODUCTION

- **Robust Optimization** (RO) is a subset of optimization theory that deals with a certain measure of **robustness** vs **uncertainty**. This balance of robustness and uncertainty is represented as variability in the parameters of the problem at hand and/or in its solution.

- In RO, the user aims to find decisions that are optimal for the *worst-case realization* of the uncertainties, within a given (finite or infinite) set.

- RO dates back to the beginning of modern decision theory in the 1950's. It became a discipline of its own in the 1970's, with paralleled development in other technological fields.

- Sources of **uncertainty** could be due to at least three different conditions:
    1. Lack of knowledge - such as not knowing exactly how much oil is in a reserve
    2. Noise - such as measurement errors, or incomplete data
    3. Events that have not yet occurred - such as future product demand

- Robustness refers to the ability of a system to *cope with errors* during an execution. It can also be defined as the ability of an algorithm to continue operating despite abnormalities in calculations. Most algorithms try to find a balance between robustness and efficiency/execution time.

ROBUST OPTIMIZATION

APPLICATIONS

- RO has traditionally been applied in statistics, but is now is applied in several other fields, including (but not limited to):
  - Operations research
  - Control theory
  - Finance (e.g., portfolio management, macroeconomics, etc.)
  - Logistics (e.g., supply chain/inventory management under stochastic demands)
  - TLC engineering (e.g., design of networks under traffic uncertainty)
  - Manufacturing engineering
  - Chemical engineering
  - Medicine

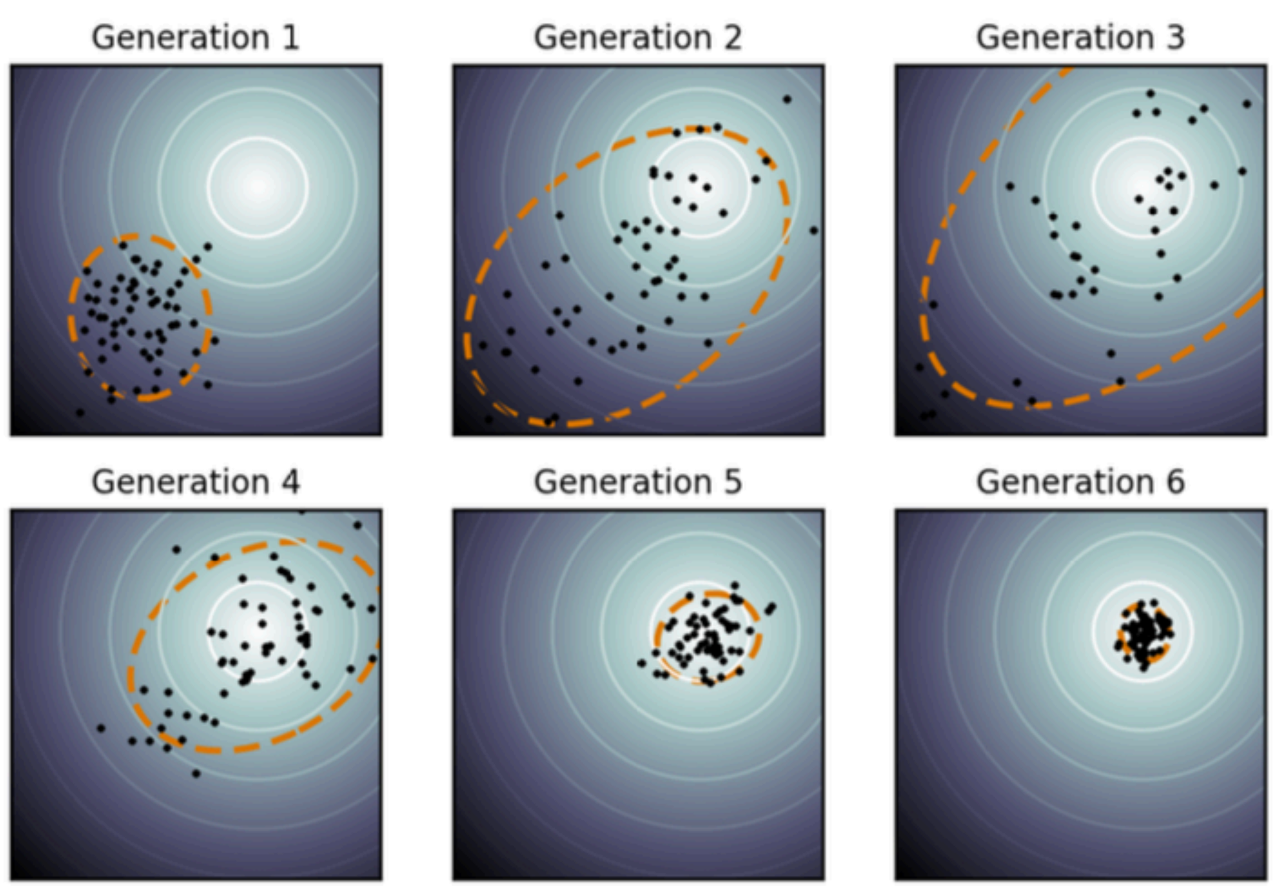
- In engineering problems, robust optimization formulations often take the name of "Robust Design Optimization'' (RDO) or "Reliability Based Design Optimization'' (RBDO)

ROBUST OPTIMIZATION

PREVIOUSLY SEEN METHODS

- Bayesian Optimization
  As we have seen, BO is especially meant for computationally expensive problems *with uncertainty*.

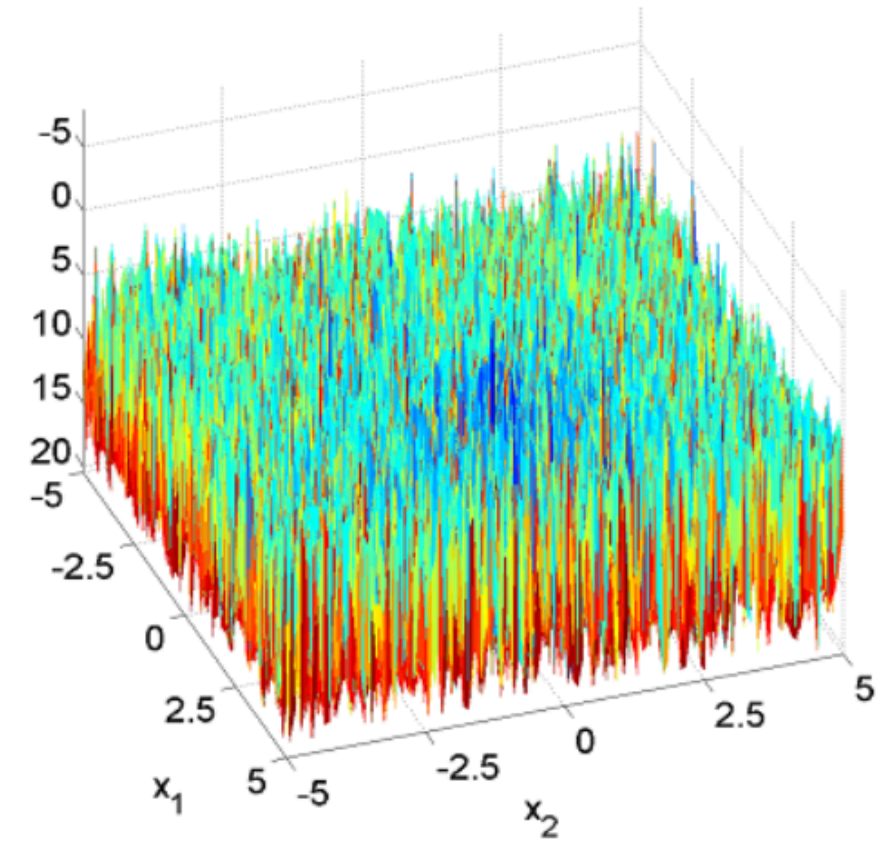
- Evolutionary Algorithms
  - It is known that population-based stochastic optimizers (e.g. EAs) can locate the region of the optimum of noisy functions. For instance, CMA-ES.
  - Drawback: typically, many function evaluations are needed! (recent CMA-ES variants need less)



ROBUST OPTIMIZATION

CONTINUOUS DOMAINS - NOISY CASES

$$\min_x f(x, U)$$
$$\text{such that } g(x, U) \leq 0$$



- $x \in \mathcal{X}$, vector of deterministic, controlled variables over which the optimization is carried out.
- $U : \Omega \rightarrow \mathcal{A}$, vector of random variables of pdf $p_U(.)$.
  $\Rightarrow$ $f(x, U)$, $g(x, U)$ are dependent random variables.
- This double parameterization, underlying Taguchi's methods in the 80's, is general. It is also called "augmented space" or "hybrid space". Cf. [Beyer and Sendhoff, 2007], [Pujol et al., 2009].

Beyer, H.-G. and Sendhoff, B. (2007). Robust optimization–a comprehensive survey. Computer methods in applied mechanics and engineering, 196(33-34):3190–3218.
Pujol, G., Le Riche, R., Roustant, O., and Bay, X. (2009). Optimisation multidisciplinaire en mécanique: Réduction de modèles, robustesse, fiabilité, réalisations logicielles, chapter L'incertitude en conception: formalisation, estimation. Hermes Science Publications, Paris.

ROBUST OPTIMIZATION

CONTINUOUS DOMAINS - NOISY CASES

Remove the uncertainty (mathematically) by formulating the problem with statistical risk measures, either as a constrained optimization problem,

$$\min_x \rho^f(x) \quad , \quad \rho^f(x) = \rho^f\left(f(x,U)\right)$$
$$\text{such that } \rho^g(x) \leq 0 \quad , \quad \rho^g(x) = \rho^g\left(g(x,U)\right)$$

or through an aggregation of statistical measures (remember $f(.,U)$ and $g(.,U)$ are dependent),

$$\min_x \rho(x) \quad , \quad \rho(x) = \rho\left(f(x,U), g(x,U)\right)$$

or a combination thereof, $\min_x \rho(x)$ s.t. $\rho^g(x) \leq 0$.

CONTINUOUS DOMAINS - RISK MEASURES FOR THE OBJECTIVE FUNCTION 1/2

The “robust” formulations, single criteria:

ρ^f $\equiv$ expectation ($\mathbb{E}$) as average performance.

$\equiv$ variance ($\mathbb{V}$) as performance dispersion.

$\equiv$ quantile ($\mathbb{Q}_\alpha$) as guaranteed performance since α% of the realizations will be better. $\mathbb{Q}_{50}$, the median, as representative performance. $\mathbb{Q}_{100}$ is the worst-case formulation.

$\equiv$ a quantile difference ($\mathbb{Q}_\alpha - \mathbb{Q}_{1-\alpha}$) as a performance dispersion.

$\equiv$ super-quantile as guaranteed performance with an account for extreme values.

CONTINUOUS DOMAINS - RISK MEASURES FOR THE OBJECTIVE FUNCTION 2/2

The "robust" formulations, multiple criteria:

$\mathbb{p}^f$ $\equiv$ a multi-objective formulation accounting for an average performance and a deviation measure [Park et al., 2006], with various subsequent resolution approaches (goal programming, ordering, full Pareto, aggregations) and in particular

$\equiv$ a linear combination of average and dispersion measures, typically $\mathbb{E}f(x, U) + \alpha\sqrt{\mathbb{V}f(x, U)}$: minimize average cost penalized by dispersion.
For $f(x, U) \sim \mathcal{N}$, it is equivalent to the quantile (e.g. $\alpha = 1.645$ for $\mathbb{Q}_{90}$).

Park, G.-J., Hwang, K.-H., Lee, T., and Hee Lee, K. (2006). Robust design: An overview. AIAA Journal, 44:181–191.

CONTINUOUS DOMAINS - RISK MEASURES FOR THE CONSTRAINTS

The “reliable” formulations:

$\mathbb{p}^g$ $\equiv$ $\alpha - \mathbb{P}(g(x, U) \leq 0) \leq 0$, as at least $\alpha\%$ chances of satisfying the constraints.

$$\mathbb{P}(g(x, U) \leq 0) = \int_{g(x,u) \leq 0} p_U(u) du = \int_{\mathcal{A}} \mathbb{1}_{g(x,u) \leq 0} p_U(u) du$$
$$= \mathbb{E}(\mathbb{1}_{g(x,u) \leq 0})$$

$\equiv$ Equivalently, $\mathbb{Q}_\alpha(g(x, U)) \leq 0$.

CONTINUOUS DOMAINS - RISK MEASURES FOR BOTH OBJ. FUNCTION & CONSTRAINTS

The robust and reliable formulations:

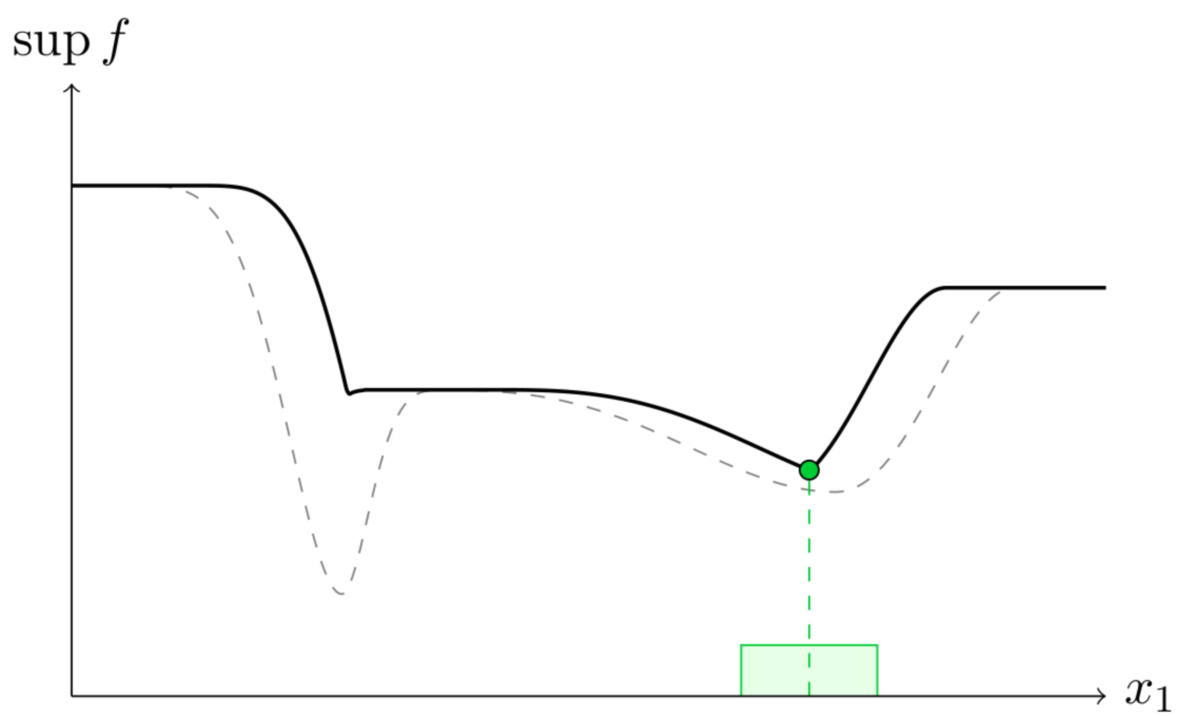
$\mathbb{p}$
- They are based on the feasible trajectories, $[f(x,U) \mid g(x,U) \leq 0\,]$, used within the $\mathbb{p}^f$ measures, e.g., $\mathbb{E}\,(f(x,U) \mid g(x,U))$,
- associated to a constraint risk measure $\mathbb{p}^g$.

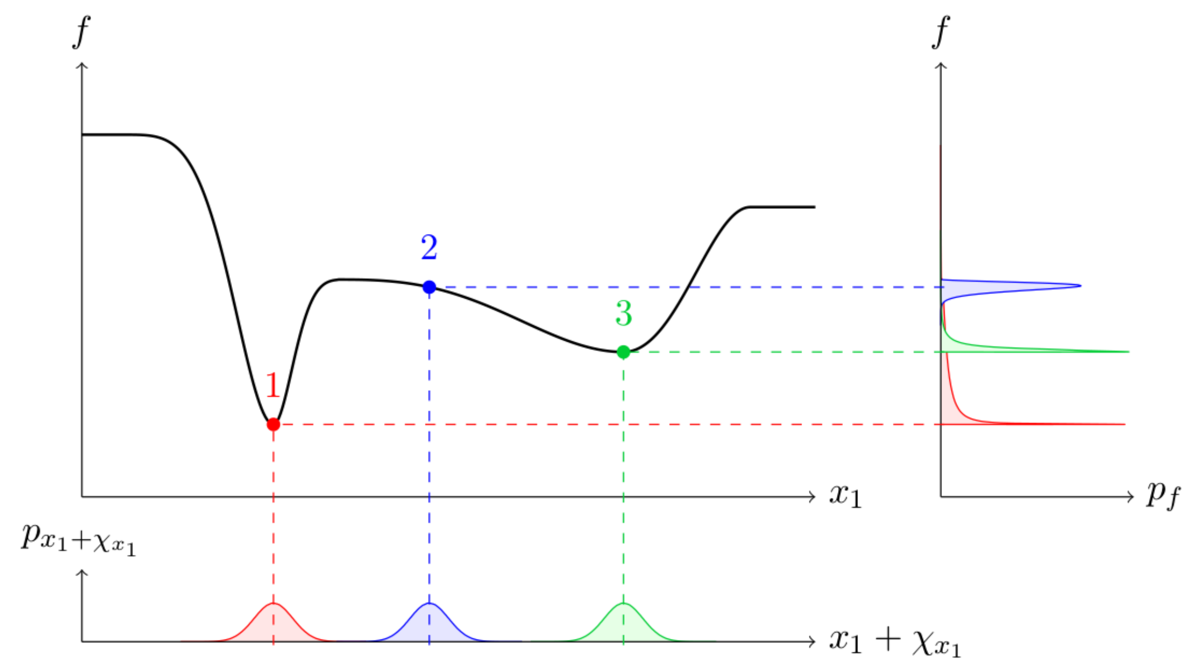
An ideal series formulation:

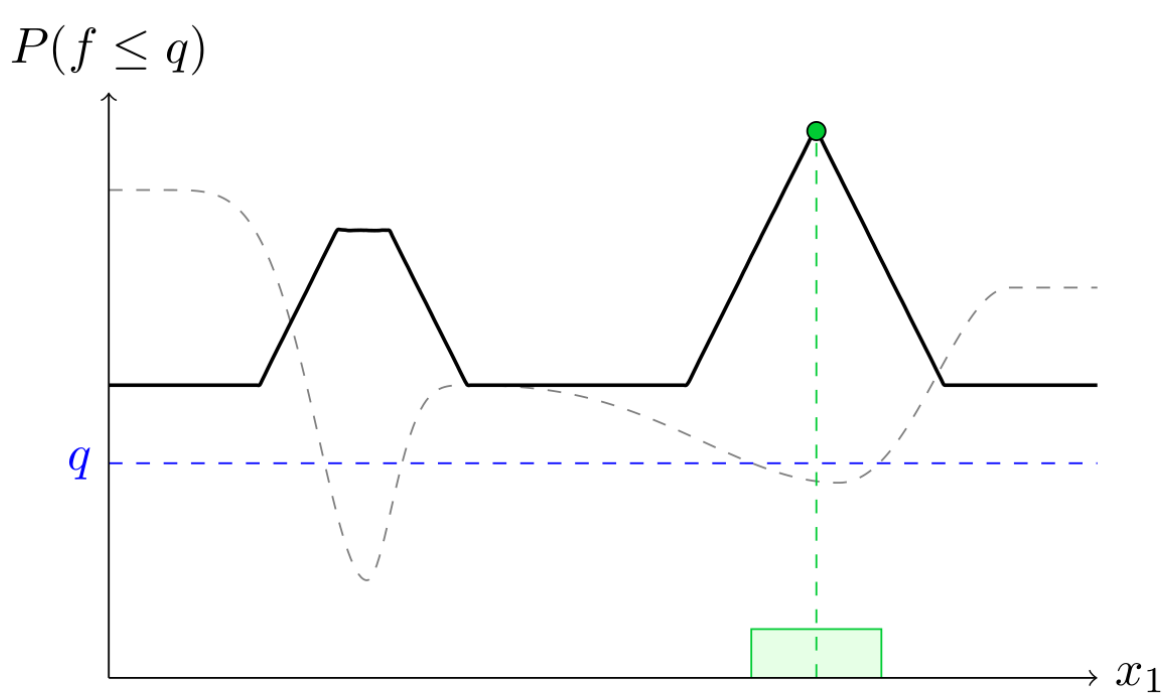
- $G(x)$ the random event "all constraints are satisfied at x", $G(x) \equiv \bigcap_i [g_i(x,U) \leq 0] \leftarrow$ now several constraints $g_i(,)$,
- $$\begin{array}{l} \min_{x\in\mathcal{X}} \mathbb{Q}_\alpha(f(x,U) \mid G(x)) \\ \text{such that } \alpha' - \mathbb{P}(G(x)) \ \leq \ 0 \end{array}$$

ROBUST OPTIMIZATION

CONTINUOUS DOMAINS - RISK MEASURES







The choice of the risk measure changes the optimum and the mathematical properties of the problem!

CONTINUOUS DOMAINS - RISK MEASURES

Many risk measures can be written as sums over u samples:

$$\frac{1}{N}\sum_{i=1}^{N} f(x,u^i) \overset{N\nearrow}{\to} \mathbb{E}f(x,U)$$

$$\frac{1}{N^2}\sum_{i<j}^{N} \left(f(x,u^i) - f(x,u^j)\right)^2 \overset{N\nearrow}{\to} \mathbb{V}f(x,U)$$

$$\frac{1}{N}\sum_{i=1}^{N} \mathbb{1}_{g(x,u^i)\leq 0} \overset{N\nearrow}{\to} \mathbb{P}(g(x,U)\leq 0)$$

which makes them appropriate for stochastic gradients. Parallel with machine learning methods where the u_i's are data samples.

ROBUST OPTIMIZATION

CONTINUOUS DOMAINS - THE "DOUBLE LOOP" ISSUE

The cumulation of the search on x with the estimation of the risk measures creates a double loop that takes too long to calculate in the context of expensive functions.

(this, in addition with the importance of taking uncertainties into account, explains the number of articles dealing with both uncertainties and optimization.)

1 Optimization loop: propose new x based on past $[x, \rho(x)]$, n times

x

2 Risk estimation loop: at a given x loop on u's to estimate $\rho(x)$.

u For example, crude Monte Carlo and mean, $\mathbb{E}f(x, U) \approx \frac{1}{m}\sum_{i=1}^{m} f(s(x, u^i))$

ROBUST OPTIMIZATION

CONTINUOUS DOMAINS - APPROXIMATION METHODS: 2, 1, 0 LOOPS

2 LOOPS

1) Optimization algorithm: proposes x^{t+1}
2) Calculate $\beta(x^{t+1})$ (reliability analysis, sub-optimization with iteration on u through $T_{vu}(v)$) and $f(x^{t+1})$

   Stop or go back to 1

$$\Rightarrow \quad \begin{array}{l} \min_{x \in \mathcal{X}} f(x) \\ \text{such that } \beta(x) \geq \beta^{\text{tol}} \end{array}$$

where $\beta^{\text{tol}} = \Phi^{-1}(\alpha)$ and the reliability index is

$$\beta(x) \;=\; \arg\min_{v} \|v\| \text{ such that } g(x, T_{vu}(v)) > 0$$

and $T_{vu}()$ is a transformation from a vector of random variables V that are standard normal to U, typically the inverse of $v = \Phi^{-1}(\text{CDF}(u))$

1 LOOP

1) Optimization algorithm: proposes x^{t+1} and approximate reliability analyses, e.g., $\widehat{\beta}(x^{t+1}) = \beta(x^t) + \nabla_x \beta(x^t)(x^{t+1} - x^t)$

   Stop or sometimes update approximation $\widehat{\beta}(.)$ and go back to 1

0 LOOP (DECOUPLED)

$$\begin{array}{ccc} \text{(find worst uncertainty)} & v^t & \text{(optim with fixed uncertainty)} \\ \max_v g(x^t, T_{vu}(v)) & \rightleftharpoons & \min_x f(x) \\ \text{such that } \|v\| = \beta^{\text{tol}} & x^t & \text{such that } g(x, T_{vu}(v^t)) \end{array}$$

WHAT ABOUT LP, QP, ETC.?

There are different concepts of robust optimization (Cornuejols and Tütüncü, 2007). Here we only consider robust optimization in the worst-case sense (Ben-Tal et al., 2009).

The idea is to deal with uncertain parameters that are not associated with a stochastic characterization of uncertainty.

Typical uncertainty sets are:

- Finite set of scenarios: $\mathcal{U} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_k\}$
- Interval (box) uncertainty: $\mathcal{U} = \{\mathbf{l} \leq \mathbf{p} \leq \mathbf{u}\}$
- Polyhedral uncertainty, possibly obtained as the convex hull of a finite set (polytopic uncertainty): $\mathcal{U} = \mathrm{conv}(\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_k)$
- Ellipsoidal uncertainty: $\mathcal{U} = \{\mathbf{p} \mid \mathbf{p} = \mathbf{p}_0 + \mathbf{M}\mathbf{u}, \|\mathbf{u}\|_2 \leq 1\}$

G. Cornuejols, R. Tütüncü. Optimization Methods in Finance. Cambridge University Press, 2007.
A. Ben-Tal, L. El Ghaoui, A. Nemirovski. Robust Optimization. Princeton University Press, 2009.

ROBUST COUNTERPARTS

The emphasis is on constraint robustness in an uncertain problem like

$$\begin{aligned} \min \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{x}, \mathbf{p}) \in K \end{aligned}$$

where $\mathbf{p} \in \mathcal{U}$ and K defines the feasible set, typically in terms of inequalities (e.g., when $K = \mathbb{R}^m_+$). The uncertain problem can be considered as a *collection* of problems.

Note that we do not consider uncertainty in the objective function. If f depends on the uncertain parameters $\mathbf{p}$, we can use the transformation:

$$\begin{aligned} \min \quad & t \\ \text{s.t.} \quad & t - f(\mathbf{x}, \mathbf{p}) \geq 0 \\ & \mathbf{g}(\mathbf{x}, \mathbf{p}) \in K \end{aligned}$$

The *robust counterpart* is the semi-infinite (unless $\mathcal{U}$ is finite) problem

$$\begin{aligned} \min \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{x}, \mathbf{p}) \in K, \quad \forall \mathbf{p} \in \mathcal{U} \end{aligned}$$

ROBUST OPTIMIZATION

ROBUST COUNTERPARTS

Notice that with no loss of generality we are assuming that there is no uncertainty in the objective function. This is because of the following equations

$$
\begin{aligned}
&\min_{x}.\ \max_{c \in U_c}.\ c^T x \\
&\text{s.t. } a_i^T x \le b_i,\ \forall a_i \in U_{a_i},\ \forall b_i \in U_{b_i},\ i = 1, \dots, m. \\
&\qquad \Updownarrow \\
&\min_{x,\alpha}.\ \alpha \\
&c^T x \le \alpha,\ \forall c \in U_c \\
&a_i^T x \le b_i,\ \forall a_i \in U_{a_i},\ \forall b_i \in U_{b_i},\ i = 1, \dots, m.
\end{aligned}
$$

TRACTABLE ROBUST COUNTERPARTS

Semi-infinite problems feature an infinite set of constraints, but a finite set of variables. They are intractable in general, even though one can resort to sampling.

However, when the uncertain problem is relatively easy (e.g., convex problems like LPs or QPs) and the uncertainty set is not too difficult (e.g., convex sets like polyhedra or ellipsoids), we may be able to reformulate the robust counterpart as a convex problem like

- LP
- QP
- QCQP (quadratically constrained QP)
- SOCP (second order cone programming)
- SDP (semidefinite programming)

In what follows we just consider some examples, rather than a general theory.

ROBUST OPTIMIZATION

UNCERTAIN LP

A general uncertain LP is a collection of LP models

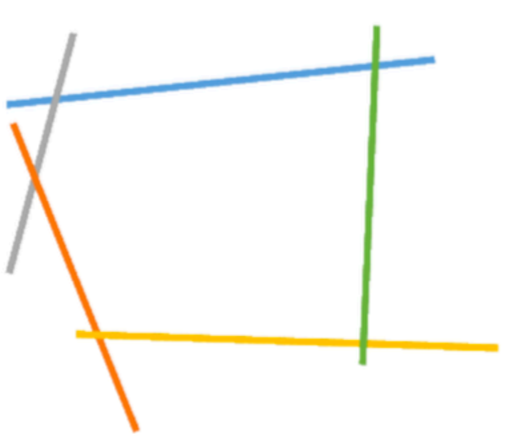
$$\begin{aligned} \min \quad & \mathbf{c}^T \mathbf{x} + d \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \end{aligned}$$

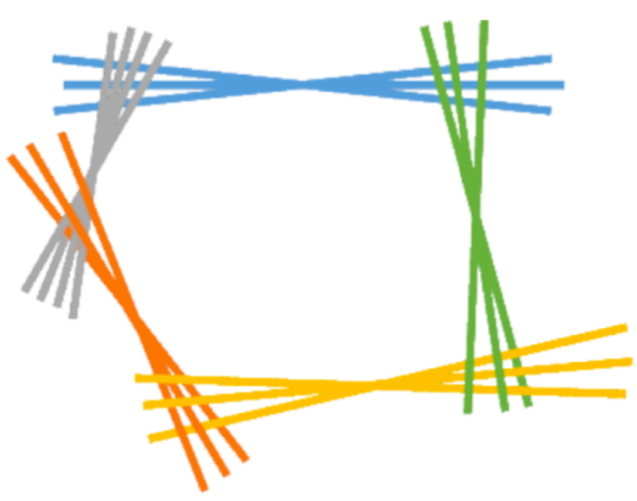
where $(\mathbf{c}, d, \mathbf{A}, \mathbf{b}) \in \mathcal{U}$.

We have already seen that we may assume a certain objective function.

It also turns out that we may evaluate feasibility constraint-wise (unless we deal with more complicated modeling frameworks, like adjustable robust models).

So, we may consider an uncertain constraint like $\{\mathbf{a}^T \mathbf{x} \leq b\}_{[\mathbf{a};b] \in \mathcal{U}}$.





We won't see the details, but it can be shown that an uncertain LP with box or polyhedral uncertainty can be reformulated as an LP.

ROBUST OPTIMIZATION

UNCERTAIN QP

Let us consider an uncertain convex quadratic program:

$$\begin{aligned} \min \quad & \frac{1}{2}\mathbf{x}^T\mathbf{P}\mathbf{x} + \mathbf{q}^T\mathbf{x} + r \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \end{aligned}$$

where we have only uncertainty on the Hessian matrix $\mathbf{P} \in S^n_+$ (Note: for a change, here we have optimality issues, rather than feasibility issues).

Then, we may formulate the robust counterpart

$$\begin{aligned} \min \quad & \sup_{\mathbf{P}\in\mathcal{U}} \left\{ \frac{1}{2}\mathbf{x}^T\mathbf{P}\mathbf{x} + \mathbf{q}^T\mathbf{x} + r \right\} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \end{aligned}$$

The type of optimization problem we end up with depends on the nature of the uncertainty set $\mathcal{U}$.

UNCERTAIN QP - FINITE UNCERTAINTY SET

Let us consider the uncertainty set $\mathcal{U} = \{\mathbf{P}_1, \ldots, \mathbf{P}_K\}$, with $\mathbf{P}_i \in S_+^n$, $i = 1, \ldots, K$.

The robust counterpart is

$$\begin{aligned} \min \quad & t \\ \text{s.t.} \quad & \frac{1}{2}\mathbf{x}^T\mathbf{P}_i\mathbf{x} + \mathbf{q}^T\mathbf{x} + r \leq t, \qquad i = 1, \ldots, K \\ & \mathbf{Ax} \leq \mathbf{b} \end{aligned}$$

which is a QCQP with variables t and $\mathbf{x}$, a convex optimization problem.

Again, we won't see the details, but it can be shown that the robust counterpart of uncertain QP is just another QP.

ROBUST OPTIMIZATION IN PRACTICE

- Essentially, RO seeks a solution that will have an "acceptable" performance under most (if possible, *all*) realizations of the uncertain inputs.
  - Usually, no distribution assumption is made on the uncertain parameters (if such information is available, it can be used beneficially).
  - Usually, it is a conservative (*worst-case oriented*) methodology: in RO, we do not tolerate a violation of the constraints *for any values of the uncertain parameters in the uncertainty set*. Among the possible solutions, we pick the one that minimizes the objective function value.

- RO is useful when:
  - Some parameters come from an estimation process and may be contaminated with errors.
  - There are "hard" constraints that must be satisfied no matter what.
  - The objective function value/optimal solutions are highly sensitive to perturbations.
  - The user cannot afford low probability high-magnitude risks (e.g., designing a bridge).

- Applying the Robust Counterpart (RC) approach is an "art". It is important to formulate the problem (i.e., *find an appropriate uncertainty set*) in a way suitable for the application of RC-based techniques. Otherwise, the RC may be useless (if not intractable, as mentioned earlier).

ROBUST OPTIMIZATION

ALTERNATIVES TO ROBUST OPTIMIZATION

- **Sensitivity Analysis**. Solve the problem with fixed values of the parameters (perhaps the most likely values) and see how the optimal solution is affected by *small perturbations* (a posteriori).

- **Stochastic Programming**. Under an *assumed probability distribution* of the uncertain parameters, the objective function becomes a collection of random variables. Choose the "best" in this collection according to some criterion, e.g., the expected value or some other suitable utility function. But, this approach results in huge optimization problems with heavy data requirements.

- **Adjustable Robustness Optimization (ARO)**. Consider a multi-period optimization problem with uncertain parameters where *uncertainty is revealed progressively* through periods. A subset of the decision variables can be chosen after observing realizations of some uncertain parameters. This allows to *correct* the earlier decision made under a smaller information set (in stochastic programming, this correction is called "recourse"). Example:

$$\min_{x^1,x^2} \{c^T x^1 : A^1 x^1 + A^2 x^2 \leq b\}$$

  where x^1 is the first stage variables, and x^2 are the second-stage variables, A^1, A^2, b are uncertain parameters, and A^1 is revealed to the user *only after choosing* x^1. So, at the moment of choosing x^2 the user knows the value of A^1.

ALTERNATIVES TO ROBUST OPTIMIZATION

- **Adjustable Robustness Optimization (ARO)** (cont'd)

$$\min_{x^1}\{c^T x^1 : \forall (A^1, A^2, b) \in \mathcal{U} \exists x^2 = x^2(A^1, A^2, b) : A^1 x^1 + A^2 x^2 \leq b\}$$

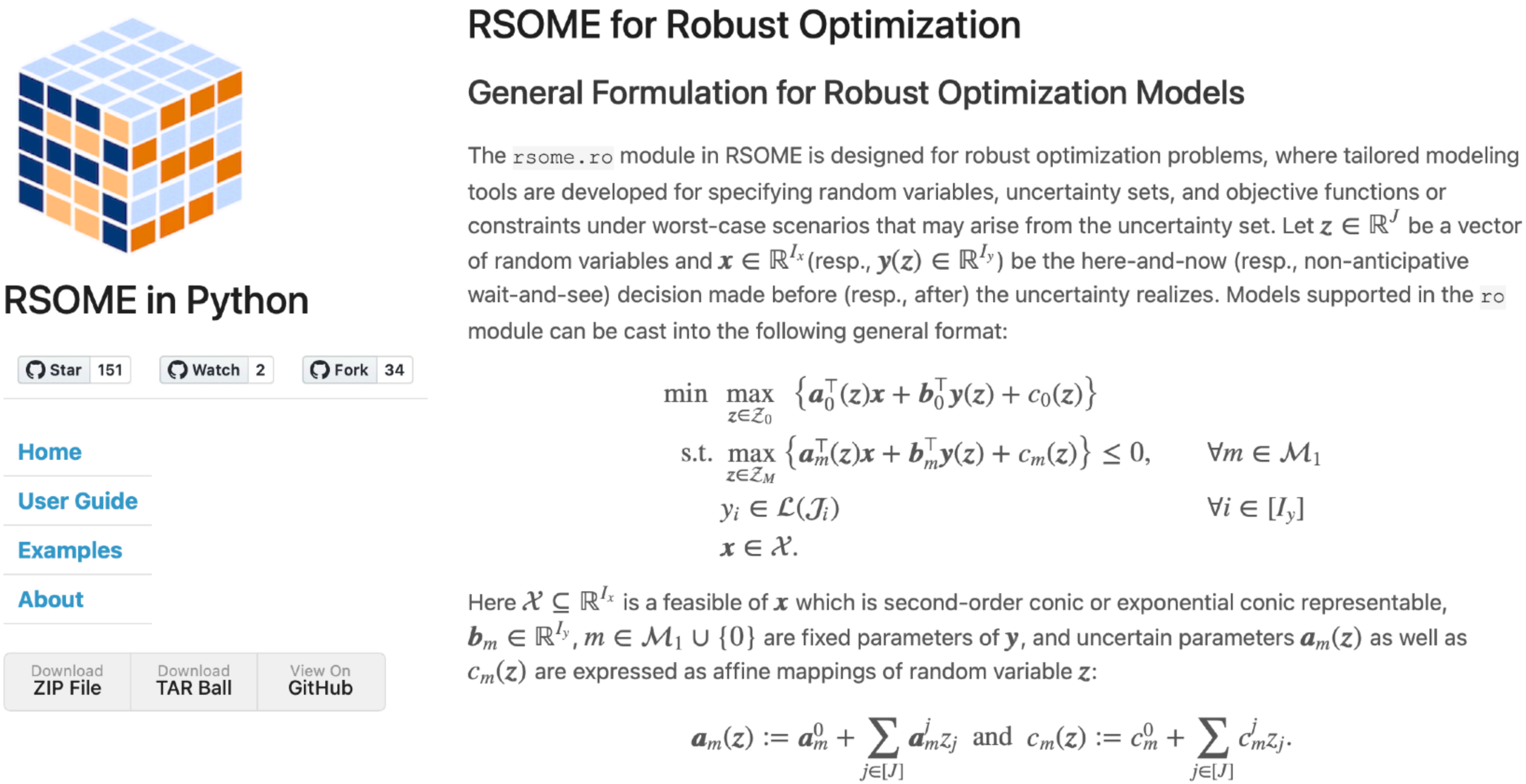
  - The feasible set of the second problem (ARO) is larger than the feasible set of the RC.
  - ARO is therefore less conservative compared to RC… However, ARO is harder to formulate explicitly as an easy optimization problem (e.g., LP), because we do not know the functional form of the dependency.
  - Only very simple uncertainty sets allow "nice" ARO formulations. Otherwise, we have to assume a simple functional form for dependencies.

ROBUST OPTIMIZATION

ROBUST OPTIMIZATION IN PYTHON

Various packages available, e.g. **RSOME** (https://xiongpengnus.github.io/rsome/ro_rsome) and **ROmodel** (https://github.com/cog-imperial/romodel), the latter based on Pyomo (https://github.com/Pyomo/pyomo).

Both provide interfaces to define different kinds of uncertainty sets (polyhedral, ellipsoidal, etc.)



RSOME in Python

Star 151 | Watch 2 | Fork 34

- Home
- User Guide
- Examples
- About

Download ZIP File | Download TAR Ball | View On GitHub

RSOME for Robust Optimization

General Formulation for Robust Optimization Models

The `rsome.ro` module in RSOME is designed for robust optimization problems, where tailored modeling tools are developed for specifying random variables, uncertainty sets, and objective functions or constraints under worst-case scenarios that may arise from the uncertainty set. Let $\boldsymbol{z} \in \mathbb{R}^J$ be a vector of random variables and $\boldsymbol{x} \in \mathbb{R}^{I_x}$ (resp., $\boldsymbol{y}(\boldsymbol{z}) \in \mathbb{R}^{I_y}$) be the here-and-now (resp., non-anticipative wait-and-see) decision made before (resp., after) the uncertainty realizes. Models supported in the `ro` module can be cast into the following general format:

$$\begin{aligned}
\min\ \max_{\boldsymbol{z}\in\mathcal{Z}_0}\ & \left\{\boldsymbol{a}_0^\top(\boldsymbol{z})\boldsymbol{x} + \boldsymbol{b}_0^\top\boldsymbol{y}(\boldsymbol{z}) + c_0(\boldsymbol{z})\right\} && \\
\text{s.t.}\ \max_{\boldsymbol{z}\in\mathcal{Z}_M}\ & \left\{\boldsymbol{a}_m^\top(\boldsymbol{z})\boldsymbol{x} + \boldsymbol{b}_m^\top\boldsymbol{y}(\boldsymbol{z}) + c_m(\boldsymbol{z})\right\} \leq 0, && \forall m \in \mathcal{M}_1 \\
& y_i \in \mathcal{L}(\mathcal{J}_i) && \forall i \in [I_y] \\
& \boldsymbol{x} \in \mathcal{X}.
\end{aligned}$$

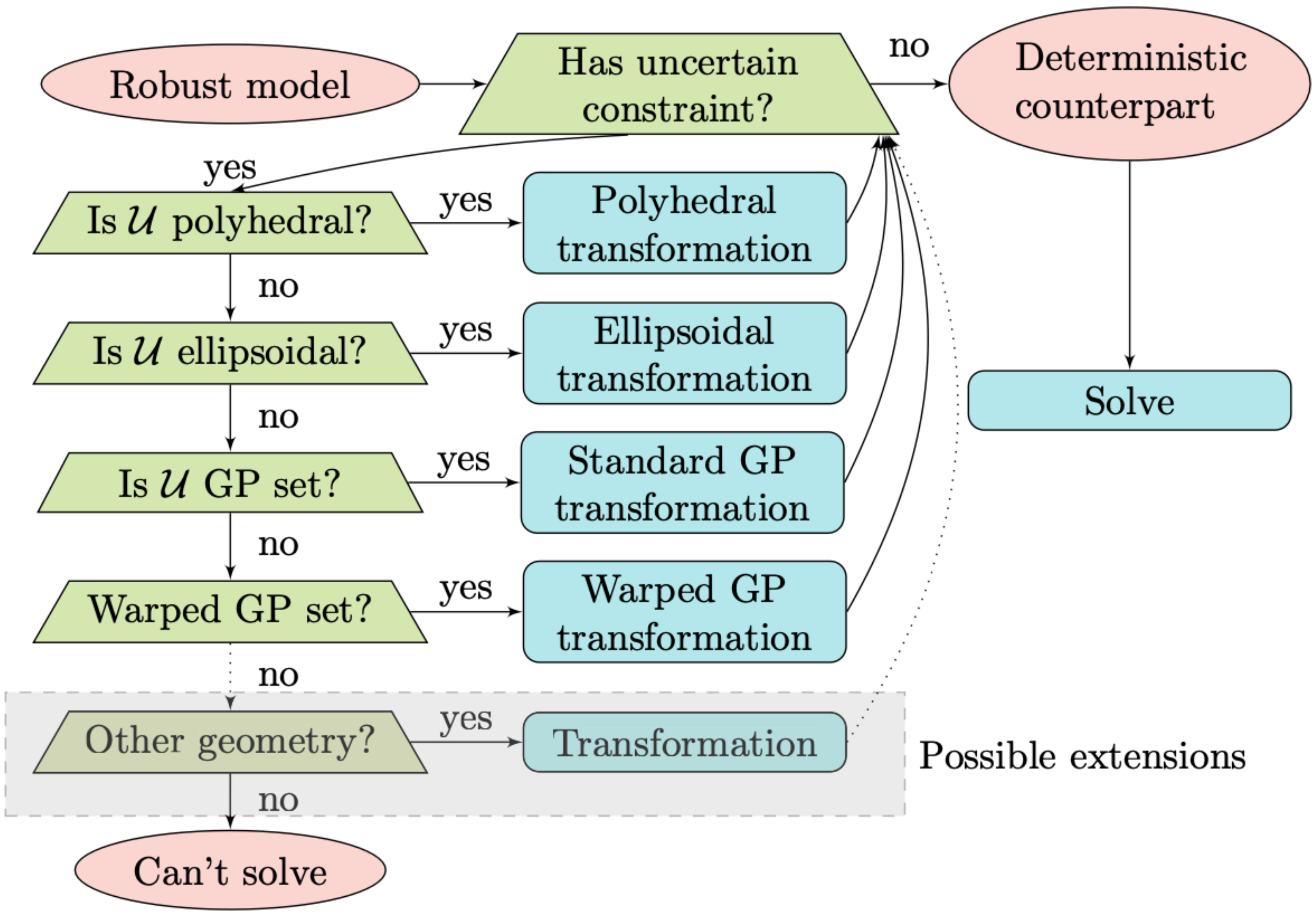
Here $\mathcal{X} \subseteq \mathbb{R}^{I_x}$ is a feasible of $\boldsymbol{x}$ which is second-order conic or exponential conic representable, $\boldsymbol{b}_m \in \mathbb{R}^{I_y}$, $m \in \mathcal{M}_1 \cup \{0\}$ are fixed parameters of $\boldsymbol{y}$, and uncertain parameters $\boldsymbol{a}_m(\boldsymbol{z})$ as well as $c_m(\boldsymbol{z})$ are expressed as affine mappings of random variable $\boldsymbol{z}$:

$$\boldsymbol{a}_m(\boldsymbol{z}) := \boldsymbol{a}_m^0 + \sum_{j\in[J]} \boldsymbol{a}_m^j z_j \ \text{ and }\ c_m(\boldsymbol{z}) := c_m^0 + \sum_{j\in[J]} c_m^j z_j.$$

Z. Chen and X. Peng. "RSOME in Python: An open-source package for robust stochastic optimization made easy." *Optimization Online*.

ROBUST OPTIMIZATION

ROBUST OPTIMIZATION IN PYTHON



RC Reformulation scheme carried out in ROmodel

J. Wiebe and R. Misener. "ROmodel: modeling robust optimization problems in Pyomo." *Optimization and Engineering* 23.4 (2022): 1873-1894

ROBUST OPTIMIZATION

FURTHER READING

- Duchi 2018, "Optimization with uncertain data".
- Powell 2016, "A Unified Framework for Optimization under Uncertainty".
- Bertsimas et al. 2011, "Theory and Applications of Robust Optimization".
- Bertsimas and Thiele 2006, "Robust and Data-Driven Optimization: Modern Decision-Making Under Uncertainty".
- Sahinidis 2003, "Optimization under uncertainty: state-of-the-art and opportunities".
- Rockafellar 2001, "Optimization under uncertainty" (lecture notes).

Questions?