Variable Robustness Control: Principles and Algorithms

Marco C. Campi Simone Garatti

***Abstract*— Robust control is grounded on the idea that a design should be guaranteed against all possible occurrences of the uncertain elements in the problem. When this philosophy is applied to securing a desired performance, it often leads to conservative, low performing, designs because emphasis is all placed on the worst-case situation. On the other hand, in many applications a 100%-guarantee is not necessary, and it may be convenient to opt for a small compromise in the guarantee level, say 99%, in favor of a (possibly significant) improvement in the performance. While the above reasoning sets a sensible principle, to date the real stumbling-block to its practical use is the lack of computationally-tractable algorithmic methods to trade guarantees for performance. This paper aims to open new directions to address this problem, and we show that this result can be achieved through randomization.**

***Index Terms*— robust control, modulation of robustness, randomized algorithms, probabilistic uncertainty, scenario approach.**

I. Introduction

Many problems in systems and control such as *controller synthesis*, *noise compensation*, or *prediction* are often cast in mathematics as *optimization programs* where the cost function captures in a quantitative manner the objective to be pursued, e.g. trajectory tracking, attenuation of noise or prediction capabilities. Along this optimization process, however, an additional element often comes into play, and this is *uncertainty*. The real world is beset with uncertainty due to partial knowledge of the environment and unpredictable and/or variable conditions in which the plant is called to operate. Uncertainty demands to exercise caution and solving optimization programs incorporating uncertainty represents a central challenge in nowadays systems theory.

Different approaches can be adopted to deal with uncertainty. In this paper, uncertainty is given a probabilistic description and we aim to design methods to achieve *a performance that is guaranteed with a desired level of probability*. Level of probability and performance are linked one to the other: like pulling down one end of a rope wrapped around a pulley lifts the other end, similarly decreasing the probability improves the performance. One main thrust of the present paper is to study this trade-off and offer the user quantitative tools to guide his choice towards a suitable compromise between the two.

This work is supported by MIUR (Ministero dell'Istruzione, dell'Università e della Ricerca) under the project *"Nuovi algoritmi ed applicazioni di identificazione e controllo adattativo"*.

M.C. Campi is with the Dipartimento di Ingegneria dell'Informazione, Università di Brescia, via Branze 38, 25123 Brescia, Italy. E-mail: marco.campi@ing.unibs.it

S. Garatti is with the Dipartimento di Elettronica ed Informazione, Politecnico di Milano, piazza L. da Vinci 32, 20133 Milano, Italy. E-mail: sgaratti@elet.polimi.it

After introducing the general principles in PART I of the paper, PART II presents algorithmic solutions. Algorithms are developed to deal with cost functions exhibiting a convex dependence on the optimization variables and an arbitrary dependence with respect to uncertainty, a set-up of theoretical importance and that covers many situations of practical interest. The output of the algorithm is given in terms of a performance-violation plot like the one displayed in Figure 1 and that anticipates results referring to a numerical example developed later in Section IV. The k variable in the horizontal axis runs over the choices among which the user can select. To each k, there is associated a different solution design, e.g. different parameter values for the controller or for the predictor being designed; the algorithm also explicitly provides these design parameter values. The solid curve is the performance, while the dashed curve is the probability with which the performance is not guaranteed, that is the probability that bad circumstances occur during the actual operation of the system so that the incurred performance is worse than that given by the first curve. These two curves are constructed on a solid theoretical ground and are the quantitative support to guide the designer in making a decision. Through these results this paper opens new avenues to address the algorithmic challenge to make designs that suitably compromise performance against guarantees.

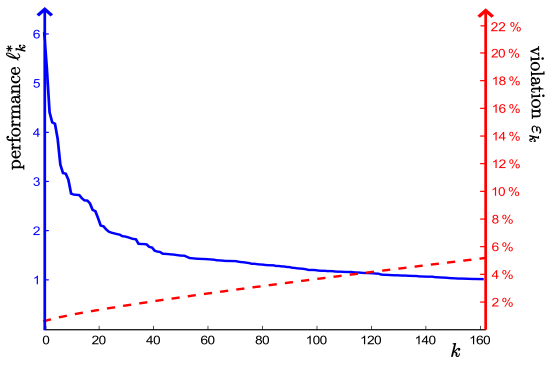

Fig. 1. Performance-violation plot.

II. PART I – Variable Robustness Control: Principles

Uncertainty may generate in different ways, and it is common practice in the systems and control community to distinguish between *structural uncertainty* (parametric or non-parametric) and uncertainty introduced by a-priori unknown *exogenous signals* (disturbances) steering the plant away from its nominal behavior, see Figure 2. In this paper,

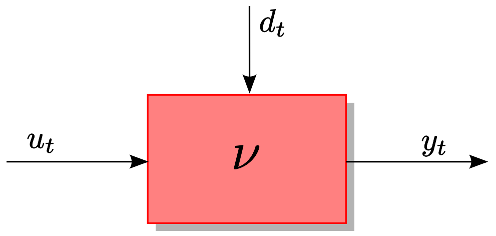

Fig. 2. A system can have uncertain parameters (ν) and be affected by disturbances (d_t).

we take a somehow more abstract point of view which encompasses as special cases both types of uncertainty with the idea of setting up a theory that can be applied across different kinds of uncertainty. Our aim is to provide the designer with general tools for use in various application endeavors.

Throughout, an uncertain element will be indicated, independently of its nature, with the symbol δ, while Δ will be the range set for δ. If for example the pole p of a stable continuous-time system is uncertain (structural uncertainty), then $\delta = p$ and Δ is the left half complex plane; if instead uncertainty stays with a disturbance $d(t) = A \cdot \text{step}(t - t_0)$ whose amplitude and step time is unknown, then $\delta = (A, t_0)$ and $\Delta = \mathbb{R}^2$.

Letting $\theta \in \mathbb{R}^d$ be the vector of design variables – it can e.g. contain the parameters of a controller, of a compensator, or those of a predictor – the cost function to be minimized is written as $\ell(\theta, \delta)$, where the simultaneous presence of θ and δ reflects that we have only partial grasp of the final optimization result through θ because it also depends on uncertainty δ. What we are facing is a so-called *Uncertain Optimization Problem*, namely:

$$\text{U-OP}: \min_{\theta \in \mathbb{R}^d} \ell(\theta, \delta), \quad \delta \in \Delta. \tag{1}$$

An U-OP cannot be considered as a complete formalization of the optimization problem though since it misses to describe how the uncertain element δ should be accounted for. Addressing the issue of providing a complete formalization requires to broaden our discussion about uncertainty.

A. Alternative ways to describe uncertainty

The worst-case approach

Uncertainty is inescapably linked to the concept of set, as there cannot be uncertainty without a set of possible uncertain outcomes. Without any further structure given to uncertainty apart that it ranges in a set Δ, a natural way to pose the problem is along a *worst-case* approach:

$$\text{WC-OP}: \min_{\theta \in \mathbb{R}^d} \left[\max_{\delta \in \Delta} \ell(\theta, \delta) \right]. \tag{2}$$

The worst-case philosophy has been adopted in the control literature out of concerns for stability. Spurred by the work of Doyle, [20], and others that showed that the classical LQG regulator had no stability margins, control theorists started in the 1980s to seek for alternative approaches able to provide robust guarantees of stability. Much influential was the work of Zames, [53], which *de facto* launched the era of H_∞ control as the robust alternative to LQG control, see [26], [54], [18]. Ever since, the worst-case approach has been ubiquitous in the robust control literature, and it has been applied to a variety of control problems.

The average approach

At times, it may be convenient to adopt a more structured, probabilistic, point of view in the description of uncertainty.

In formal terms, suppose that Δ is endowed with a probability Pr. Depending on the problem at hand, Pr can have different interpretations. Sometimes it is a measure of the likelihood with which situations occur, other times it simply describes the relative importance we attribute to different uncertainty instances. Using Pr allows one to weigh situations so that one can form an overall cost by *averaging*, and then solve an average optimization problem:

$$\text{A-OP}: \min_{\theta \in \mathbb{R}^d} E_\Delta [\ell(\theta, \delta)] = \min_{\theta \in \mathbb{R}^d} \int_\Delta \ell(\theta, \delta) \text{dPr}.$$

This framework has been widely adopted when uncertainty is associated with disturbance signals, [5], [31]. A typical example is quadratic stochastic control where the average cost is in discrete time given by

$$E_\Delta \left[\sum_{t=0}^{T} \left\{ x(t)^T Q x(t) + u(t)^T R u(t) \right\} \right], \tag{3}$$

where $x(t)$ indicates the system state, $u(t)$ the system input, and expectation is taken with respect to the realizations of the disturbance affecting the system. Thus, here we identify one δ with a noise realization and Δ is the set of all realizations with a probability Pr dictated by the type of noise we are considering. If e.g. the noise is white and Gaussian, Pr is the product probability of T Gaussians.

The average approach is normally less conservative than the worst-case approach, but its conservatism can be enhanced by the introduction of an exponential cost. E.g. in stochastic control, with the notation $J := x(t)^T Q x(t) + u(t)^T R u(t)$, instead of considering a quadratic cost $E_\Delta[J]$ as in (3), one can use $E_\Delta[\exp(J)]$. Because of the exponential function, the penalty in the occurrence of values of J larger than $E_\Delta[J]$ outweighs the alleviation in penalty caused by the occurrence of some values less than $E_\Delta[J]$, corresponding to a pessimistic viewpoint. A connection between this average-exponential approach and the worst-case approach was established by Jacobson and Whittle for linear systems, [27], [52], and then highlighted in a nonlinear set-up in [28], [16].

VRC: Variable robustness control

Within a probabilistic set-up, minimizing an average cost function is not the only route one can follow. Alternatively, probability can be used to quantify the chance that a certain performance specification is not achieved. The origin of this approach traces back to [43] which used it for analysis purposes in the context of flight control, then followed by many other contributions among which [44], [39], [6], [33]. One way to evaluate the probability that a certain performance is not achieved is through randomization, and [30], [45], [46] provide explicit bounds on the number of samples needed to this end. Moreover, sequential methods have been proposed to solve synthesis problems where one wants to determine a controller that attains a specified performance level with high probability, [12], [36], [21], [29], [22], [35]. This is a *feasibility* problem where a controller is feasible if it achieves the indicated performance; on the other hand, sequential methods are not a natural tool to address *optimization* problems.

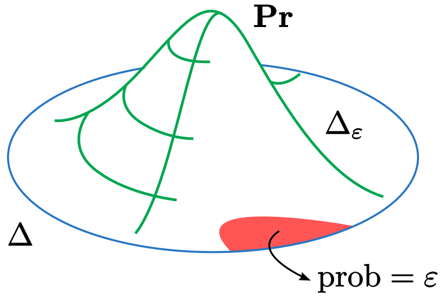

Fig. 3. The reduced Δ_ε set has probability $1-\varepsilon$.

Within this set-up of using probability to quantify the chance of negative events, we here consider optimization problems along a minimax approach where the max requirement is relaxed in a probabilistic sense. Precisely, referring to Figure 3, one is content with minimizing the max cost with max taken over a reduced set $\Delta_\varepsilon \subset \Delta$ having large enough probability, $\Pr\{\Delta_\varepsilon\} = 1-\epsilon$, and write:

$$\text{WC}_\varepsilon\text{-OP}: \ \min_{\theta\in\mathbb{R}^d}\left[\max_{\delta\in\Delta_\varepsilon}\ell(\theta,\delta)\right]. \tag{4}$$

Indicating by θ^*_ε the optimal solution of problem (4) and by ℓ^*_ε its optimal value, we have $\ell^*_\varepsilon = \max_{\delta\in\Delta_\varepsilon}\ell(\theta^*_\varepsilon,\delta)$, i.e. ℓ^*_ε is guaranteed against all uncertainty outcomes in Δ_ε, that is it is guaranteed with probability $1-\varepsilon$.

One question arises naturally along this approach and it is: how should Δ_ε be chosen in (4)? Clearly, once Δ_ε has been selected, WC$_\varepsilon$-OP is simply a WC-OP over the reduced set Δ_ε. On the other hand, the original design problem is specified in terms of $\ell(\theta,\delta)$, Δ, and Pr only, that is in terms of the ingredients of U-OP in (1) and probability Pr. Therefore, Δ_ε is not an initial assignment in the design problem and its determination has to be thought of as an integral part of the solution of the problem.

So, how should Δ_ε be selected? One obvious criterion is that leaving out an ε-probability set should improve the cost value. And, the more the better. Pushing this process all the way down to an optimal selection corresponds to solving program

$$\min_{\theta\in\mathbb{R}^d,\Delta_\varepsilon}\left[\max_{\delta\in\Delta_\varepsilon}\ell(\theta,\delta)\right]. \tag{5}$$

So formulated, the problem is well known in the optimization literature under the name of "chance-constrained" optimization, [37], [38], [19], the name meaning that a chance that bound ℓ^*_ε is violated exists, but this chance is constrained by ε. However, it has to be said immediately that finding an exact solution to the chance-constrained problem (5) is generally beyond reach due to the associated overwhelming computational difficulties. If optimality cannot be reached, one can aim for good, even though suboptimal, selections of Δ_ε. This is the route this paper follows and the focus of PART II is towards designing algorithms to obtain this result.

One aspect that deserves a mention at this point is that parameter ε should not be seen as a fixed value, rather it is a "knob" the user can tune. The bigger ε, the better the performance, but the higher the risk of performance violation. Thus, the level of robustness is adjustable, corresponding to a "variable robustness control" (VRC) approach. The choice of a suitable ε stays with the user, who will select it depending on his attitude to the risk. Our principal goal in PART II is to provide the user with tools that allow him to make the selection on solid quantitative grounds (refer back to Figure 1).

The paradigm here described of optimization under probabilistic constraints has received to date almost no attention by the systems and control community, with perhaps only the exception of [13], [1], [2]. In our appreciation, this state of things is due to two different reasons.

(i) One is *tradition*. As said above, robustness in control originated in the 1980s out of concerns for stability. When dealing with stability, a performance-risk compromise can hardly be accepted, and thus worst-case was the adopted approach in this context. Since then, robustness and worst-case have traveled hand in hand becoming almost synonyms in the control community. Meanwhile, stochastic control theorists have been working with the average cost approach.

(ii) One second crucial fact is that the chance-constrained approach *has so far lacked suitable algorithmic methods to practically find solutions*, a circumstance that has by and large hindered the applicability of this synthesis paradigm.

Our thrust with this paper is to introduce the chance-constrained/VRC paradigm to the large audience of scientists working in systems and control. Due to a new scheme rooting in randomization, we shall try to open algorithmic routes to address robustness along this paradigm in an attempt to crack the so far fundamental obstacle posed by practical computability.

B. A look at the different robustness paradigms in the optimization domain

Worst-case, average and chance-constrained approaches can be best visualized in the domain where the cost ℓ is displayed against the design variable θ.

For a given uncertainty instance δ, $\ell(\theta, \delta)$ is a deterministic function of the design variable θ. One such function is profiled in Figure 4. As δ is let vary, functions $\ell(\theta, \delta)$ form a cloud, as shown again in Figure 4. The top border of this cloud represents the worst-case performance, while cutting the cloud vertically corresponding to a given θ and taking average over the δ's returns $E_\Delta[\ell(\theta, \delta)]$. Worst-case and average designs are then obtained as indicated in Figure 5.

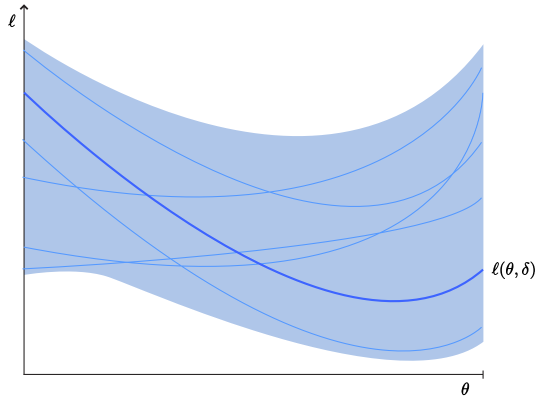

Fig. 4. One $\ell(\theta, \delta)$ and the cloud of $\ell(\theta, \delta)$ as δ is let vary.

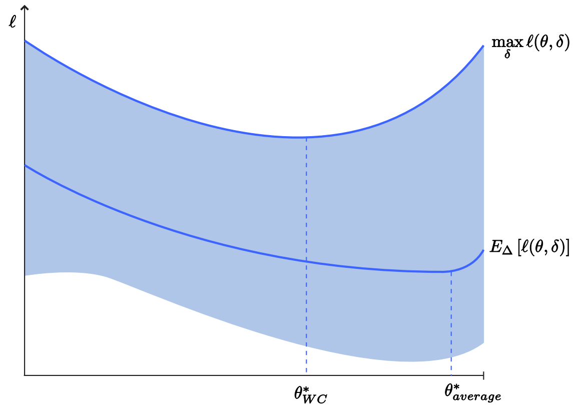

Fig. 5. θ^*_{WC} and $\theta^*_{average}$.

Moving to the chance-constrained approach, for a given θ, $\max_{\delta \in \Delta_\varepsilon} \ell(\theta, \delta)$ where Δ_ε is optimally chosen so as to improve the cost corresponds to the ℓ value that leaves an ε portion of the cloud above it.[1] Picking e.g. $\varepsilon = 1\%$ and letting θ vary, $\max_{\delta \in \Delta_{1\%}} \ell(\theta, \delta)$ describes the curve with index 1% in Figure 6. For a given θ, a 1%-fraction of the $\ell(\theta, \delta)$ functions parameterized in δ fare above ℓ and these functions represent our risk not to meet performance ℓ. The chance-constrained solutions at $\varepsilon = 1\%$ is the minimizer of this curve. As ε assumes different values, one obtains a full range of risk-performance possibilities, as visualized in Figure 6.

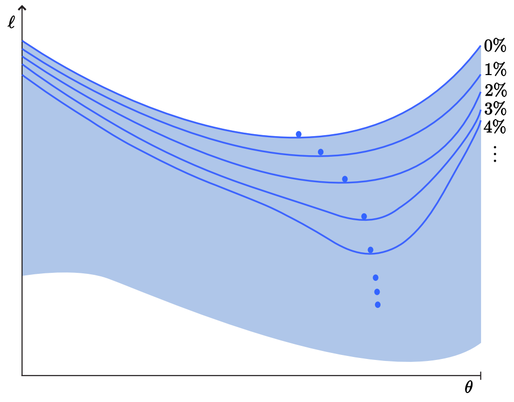

Fig. 6. Chance-constrained cost functions for different values of ε.

C. Discussion about VRC

Before moving to PART II dealing with algorithms, we feel it is advisable to pose a moment to better put VRC into perspective with other methods.

Robustness along the worst-case approach has been a successful story in control. When the uncertainty level is moderate enough so that a controller exists securing an adequate performance for all uncertainty instances, the worst-case paradigm indicates the route. Yet, at times uncertainty goes beyond this point. This can be true for structural uncertainty, and it is almost invariantly true when uncertainty stays with external disturbance signals that normally range over a vast support. Developing proper designs requires in these cases to follow a route alternative to the worst-case method. The average approach is one valid alternative that has been used for long mainly to deal with disturbance signals, but which has also been imported into structural uncertainty by the works [50], [51]. VRC follows a different route of finding solutions carrying a guaranteed performance-risk tradeoff.

To pursue a performance-risk tradeoff, a simpler route than VRC can in principle be conceived. Indicating by $\rho \cdot \Delta$, $\rho \in [0, 1]$, a re-sizing of the uncertainty set (if e.g. Δ is the unitary H_∞ ball, $\rho \cdot \Delta$ is the ball of radius ρ, and if Δ is a unitary box hosting the parameters, $\rho \cdot \Delta$ is the box of side ρ) one can explore the worst-case solution for various values of ρ to find a compromise. This is in general a loosing approach however. What one typically finds is that the ρ-performance plot is something as shown in Figure 7. To improve performance, ρ has to be decreased to, say,

[1] Said more formally, $\max_{\delta \in \Delta_\varepsilon} \ell(\theta, \delta)$ is the $(1-\varepsilon)$-level of the quantile function of the random variable $\ell(\theta, \cdot)$.

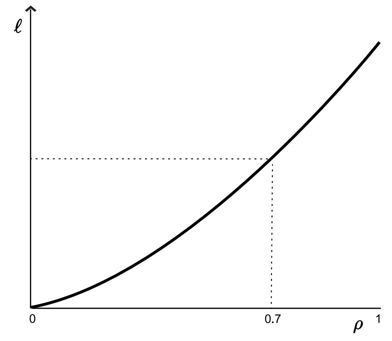

Fig. 7. ρ-performance plot.

0.7 or less, which however can correspond to a dramatic drop in terms of probability or volume. E.g. in a box in 10 dimensions, $\rho = 0.7$ shrinks the volume down from 1 to $0.7^{10} = 0.028 < 3\%$, that is a 97% portion is left out.
On the other hand, concentrating on the whole set Δ and using VRC can well show that the same performance as for $\rho = 0.7$ is attainable by leaving out a small portion of Δ only. This portion is hardly the outer shell of Δ (this is what the $\rho \cdot \Delta$ re-sizing approach leaves out) and *the reduced set Δ_ε has to be determined based on an inspection of the optimization problem*. Figure 8 illustrates the situation.

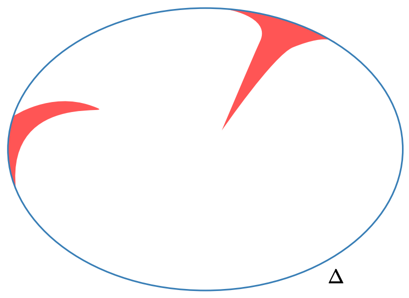

Fig. 8. Shaded region is the portion to be left out.

The region to be left out is represented as having an elongated shape to visualize that it typically has a small volume despite it can have a significant linear extension.[2] This state of things has been previously observed in [32] where the fortunate terminology "icicle geometry" has been introduced to indicate this phenomenon. In general, finding a suitable region to leave out is a formidable task and the VRC Algorithm of PART II explores using randomization to this purpose.

III. PART II – Variable Robustness Control: Algorithms

In this second part we develop an algorithm for the implementation of the VRC logic. The idea is to replace the infinite set Δ with a finite approximant obtained by randomization and to further remove elements of this approximant so as to improve the performance value.

[2]For example, in the box $[0,1]^{10}$, the region $[0,\frac{1}{2}]^{10}$ stretches from one corner to the center of the box but has only volume $\frac{1}{2^{10}} < 0.1\%$.

The algorithm, and its theoretical properties, are derived under the following assumption of convexity.

Assumption 1 (convexity): For every δ, function $\ell(\theta,\delta)$ is convex with respect to θ. $*$

Instead, the dependence of $\ell(\theta,\delta)$ on δ can be arbitrary. Albeit convexity in x represents a limitation to the use of the algorithm, this condition applies naturally to a number of problems, and, moreover, many problems are susceptible of convex reformulation via e.g. LMIs (Linear Matrix Inequalities), [7], [23], [3], [4], [40], [41].

For simplicity, we will also make the assumption to follow which guarantees that every optimization problem we will encounter in the sequel admits a unique solution.

Assumption 2: The solution of every minimax problem where $\max$ is taken over a *finite* uncertainty set $F = \{\delta_1, \delta_2, \ldots, \delta_p\} \subseteq \Delta$, i.e. the solution of every problem of the type

$$\min_{\theta \in \mathbb{R}^d} \left[\max_{\delta \in F} \ell(\theta,\delta) \right],$$

exists and is unique. $*$

Though this assumption could be relaxed, we prefer to maintain it to avoid technical complications that have little conceptual interest.

The approach of the present paper builds on the so-called "scenario approach" introduced in [8], and further developed in [9], [14], [15]. Moving a fundamental step ahead with respect to these contributions, we in this paper consider constraints removal to trade guarantees for performance. An alternative, complementary, approach is presented in the most interesting paper [2] which allows for non-convex cost functions, provided they have finite Vapnik-Chervonenkis dimension, [49], [48].

A. The VRC Algorithm

Let $\delta^{(1)}, \delta^{(2)}, \ldots, \delta^{(N)}$ be N uncertainty instances, hereafter called "scenarios", extracted independently of each other from Δ according to probability Pr.[3] This set of scenarios is used as a "surrogate", or a descriptor, of Δ and it comes handy in the development of practical algorithms because it only contains finitely many elements. What is crucial is that a procedure like VRC which is only based on $\delta^{(1)}, \delta^{(2)}, \ldots, \delta^{(N)}$ comes with precise guarantees related to the whole Δ set even though the procedure totally neglects the vast majority of the $\delta \in \Delta$ (see Theorem 1 below). This is the "magic" of randomization.

The rest of this section III-A is structured as follows.
We first introduce a "Procedure for computing θ^*_k and ℓ^*_k",

[3]See [11], [10], [47] for algorithms to perform random extractions.

where θ_k^* and ℓ_k^* are the design parameters and performance levels obtained after the removal of an increasing number k of scenarios. This procedure is then embedded in the complete "VRC Algorithm"; the VRC Algorithm also delivers complementary probabilistic guarantees that the performance levels be achieved.

procedure for computing θ_k^* and ℓ_k^*
Replacing Δ with $\delta^{(1)}, \delta^{(2)}, \ldots, \delta^{(N)}$ in the worst-case approach of (2) yields

$$\min_{\theta \in \mathbb{R}^d} \left[\max_{i=1,\ldots,N} \ell(\theta, \delta^{(i)}) \right].$$

The solution θ_0^* of this problem is the starting point for the procedure.

Moving from θ_0^*, the procedure executes a progressive elimination of scenarios according to a greedy logic. To be specific, let G_k be the set of "survived" scenarios after the elimination of k scenarios, and let θ_k^* (also called the *design at level* k) be the solution of the problem

$$\min_{\theta \in \mathbb{R}^d} \left[\max_{i \in G_k} \ell(\theta, \delta^{(i)}) \right],$$

and $\ell_k^* := \max_{i \in G_k} \ell(\theta_k^*, \delta^{(i)})$ be the corresponding performance level. To update G_k, one has to select which scenario has to be removed next. To this aim, the procedure scans one by one the scenarios in G_k and selects the one that, if removed, gives the largest improvement in performance. Eliminating this scenario from G_k gives G_{k+1}.

The following pseudo-code implements the above scheme.

Procedure for computing θ_k^* and ℓ_k^* for $k = 0, 1, \ldots, \overline{k}$

```
1: set $G_0 = \{1, 2, \ldots, N\}$;
   solve program
        $\min_{\theta \in \mathbb{R}^d} \left[ \max_{i \in G_0} \ell(\theta, \delta^{(i)}) \right]$;[4]
   let $\theta_0^*$ be the optimal solution and $\ell_0^* = \max_{i \in G_0} \ell(\theta_0^*, \delta^{(i)})$ the optimal value;
   set $Z = G_0$, $\widehat{\theta} = \theta_0^*$, and $\widehat{\ell} = \ell_0^*$;
2: FOR $k := 1$ TO $\overline{k}$
  2.1: set $A = \left\{ j \in Z : \ell(\widehat{\theta}, \delta^{(j)}) = \widehat{\ell} \right\}$;[5]
  2.2: FOR $j \in A$
         solve program
              $\min_{\theta \in \mathbb{R}^d} \left[ \max_{i \in Z-j} \ell(\theta, \delta^{(i)}) \right]$;
         let $\widehat{\theta}^j$ be the optimal solution and $\widehat{\ell}^j = \max_{i \in Z-j} \ell(\widehat{\theta}^j, \delta^{(i)})$ the optimal value;
         IF $\widehat{\ell}^j < \widehat{\ell}$ THEN set $\widehat{\theta} = \widehat{\theta}^j$ and $\widehat{\ell} = \widehat{\ell}^j$;
       END FOR
  2.3: set $Z = \left\{ i \in \{1, 2, \ldots, N\} : \ell(\widehat{\theta}, \delta^{(i)}) \leq \widehat{\ell} \right\}$;
  2.4: IF $|Z| > N - k$ THEN GOTO 2.1
                                  % ($|Z|$ = cardinality of set $Z$)
       ELSE  set $G_k = Z$, $\theta_k^* = \widehat{\theta}$, and $\ell_k^* = \widehat{\ell}$;
   END FOR
```

[4]This convex problem and that in `2.2` can be efficiently solved via standard solvers, such as the openly distributed `CVX`, [25], [24], or `YALMIP`, [34].
[5]A contains the "active scenarios".

The `FOR` cycle in `2.2` removes one by one the scenarios in search of the one whose elimination gives the largest improvement in the cost value; the search is restricted to the $\delta^{(j)}$ such that $\ell(\widehat{\theta}, \delta^{(j)}) = \widehat{\ell}$ (the "active scenarios") since the elimination of nonactive scenarios cannot possibly improve the cost value. Upon exiting the `FOR` cycle in `2.2`, the solution is stored in $\widehat{\theta}$ and the corresponding optimal value is $\widehat{\ell}$. The procedure, however, does not immediately update G_k at this stage, this is due to a detail that has been omitted in the discussion made before the procedure. Indeed, $\widehat{\theta}$ is only a potential solution at level k since removing one scenario $\delta^{(j)}$ can generate a pair $(\widehat{\theta}, \widehat{\ell})$ such that $\ell(\widehat{\theta}, \delta^{(i)}) \leq \widehat{\ell}$ for some previously removed scenario $\delta^{(i)}$. If so, we can reinstate $\delta^{(i)}$ and proceed further to eliminate another scenario before outputting the design θ_k^* at level k, and point `2.4` executes the test to decide if this is the case. The outer `FOR` in point `2` cycles over the k values, and point `1` is an initialization.
The procedure comes to termination whenever in the `FOR` cycle in `2.2` a scenario is found whose elimination improves the cost value. For this not to be true, the $\ell(\theta, \delta^{(i)})$ functions must cluster in an anomalous way, and termination of the algorithm with probability one with respect to the $\delta^{(1)}, \delta^{(2)}, \ldots, \delta^{(N)}$ extractions is assumed throughout the following.

The procedure for computing θ_k^* and ℓ_k^* implements the idea of discarding scenarios so as to improve performance. Scenarios discarding is according to a greedy logic, and this makes the procedure computationally feasible.

The procedure operates over a finite set of scenarios $\delta^{(1)}, \delta^{(2)}, \ldots, \delta^{(N)}$ and the sole interpretation of ℓ_k^* that is available at this point is that ℓ_k^* bounds the cost $\ell(\theta_k^*, \delta)$ for $\delta \in G_k$. Scenarios $\delta^{(1)}, \delta^{(2)}, \ldots, \delta^{(N)}$ are the "visible" uncertainty instances, those the procedure uses, and this is only a tiny portion of the total uncertainty set Δ. From a higher perspective, a question arises now quite naturally as how guaranteed the performance level ℓ_k^* is for the vast multitude of the other unseen $\delta \in \Delta$. This is the question of inferring the *invisible* from the *visible*, and this issue is addressed in the VRC Algorithm that is next developed.

VRC Algorithm
We start with the following definition.

Definition 1 (performance violation probability): Given a design variable $\bar{\theta}$ and a performance level $\bar{\ell}$, the *performance violation probability* of $(\bar{\theta}, \bar{\ell})$ is defined as

$$V(\bar{\theta}, \bar{\ell}) := \Pr\{\delta \in \Delta : \ell(\bar{\theta}, \delta) > \bar{\ell}\},$$

i.e. $V(\bar{\theta}, \bar{\ell})$ is the probability with which the cost obtained with $\bar{\theta}$ is bigger than level $\bar{\ell}$. $*$

Hence, being able to quantify $V(\bar{\theta}, \bar{\ell})$ corresponds to the ability of ascertaining the level of robustness that performance $\bar{\ell}$ has relative to the whole uncertain set Δ when the design variable is $\bar{\theta}$. The VRC Algorithm outputs θ_k^* and ℓ_k^* as the procedure before and it additionally returns ε_k, a tight upper bound for $V(\theta_k^*, \ell_k^*)$. Thus, it succeeds in providing a design and further complementing it with solid performance guarantees, the ingredients to build the plot of Figure 1. In a real application, the user inspects the numerical values ℓ_k^* against the probabilities ε_k to meet a suitable tradeoff before "buying" the appropriate solution design θ_k^*.

The algorithm requires three inputs, namely $\bar{\varepsilon}$, $\alpha\%$, and β, which have the following interpretation:
1. $\bar{\varepsilon}$ sets an upper bound for ε_0, the performance violation probability for $k = 0$;
2. $\alpha\%$ is the proportion of the total number N of scenarios discarded upon exiting the algorithm;
3. β is a so-called *confidence parameter* whose understanding requires a bit of additional explanation provided here. Due to randomization, θ_k^* and ℓ_k^* are random quantities depending on $\delta^{(1)}, \delta^{(2)}, \ldots, \delta^{(N)}$ and so is the result that $V(\theta_k^*, \ell_k^*) \leq \varepsilon_k$ so that one cannot totally exclude that $V(\theta_k^*, \ell_k^*)$ be bigger than ε_k for some unfortunate extractions $\delta^{(1)}, \delta^{(2)}, \ldots, \delta^{(N)}$. The input β allows the user to exercise his option to reduce the chance for this to happen below a desired level β. Although necessary from a theoretical point of view, β has a very minor practical relevance because selecting a fairly small value for β (e.g. $\beta = 10^{-7}$, or even smaller) only very modestly affect the computational burden of the algorithm as discussed later in Section III-B.

VRC Algorithm
INPUT: $\bar{\varepsilon}$, $\alpha\%$, β
OUTPUT: θ_k^*, ℓ_k^*, ε_k

```
1: compute the smallest integer $N \geq d$ (recall that $d$ is the size of $\theta$) such that
   $$\sum_{i=0}^{d} \binom{N}{i} \bar{\varepsilon}^i (1-\bar{\varepsilon})^{N-i} \leq \frac{\beta}{\alpha\% \cdot N + 1};^6 \qquad (6)$$
   let $\bar{k} := \lfloor \alpha\% \cdot N \rfloor$, the integer part of $\alpha\% \cdot N$;
2: sample $N$ independent scenarios $\delta^{(1)}, \delta^{(2)}, \ldots, \delta^{(N)}$ from $\Delta$ according to probability Pr;
3: run the "Procedure for computing $\theta_k^*$ and $\ell_k^*$ for $k = 0, 1, \ldots, \bar{k}$";
   RETURN $\theta_k^*$ and $\ell_k^*$ for $k = 0, 1, \ldots, \bar{k}$;
4: for $k = 0, 1, \ldots, \bar{k}$ solve for $\varepsilon_k$ equation
   $$\binom{d+k}{k} \sum_{i=0}^{d+k} \binom{N}{i} \varepsilon_k^i (1-\varepsilon_k)^{N-i} = \frac{\beta}{\bar{k}+1};^7 \qquad (7)$$
   RETURN $\varepsilon_k$ for $k = 0, 1, \ldots, \bar{k}$.
```

The role of ε_k, and the fact that it indeed bounds $V(\theta_k^*, \ell_k^*)$, is analyzed in the next theorem.

Theorem 1: Relation $V(\theta_k^*, \ell_k^*) \leq \varepsilon_k$ holds true simultaneously for all $k = 0, 1, \ldots, \bar{k}$ with probability at least $1 - \beta$. $*$

The proof of the theorem rests on establishing the fundamental result that the left-hand-side of equation (7) bounds the probability that $V(\theta_k^*, \ell_k^*) > \varepsilon_k$. This quantity is set equal to $\frac{\beta}{\bar{k}+1}$ so that $V(\theta_k^*, \ell_k^*) \leq \varepsilon_k$ holds true simultaneously for all $k = 0, 1, \ldots, \bar{k}$ with probability $1 - \sum_{k=0}^{\bar{k}} \frac{\beta}{\bar{k}+1} = 1 - \beta$. The technical proof is given in Appendix A.

B. Further comments on N and ε_k

sample complexity N
The size N of the scenario set is the most significant factor in determining the computational complexity of the VRC Algorithm. It can be proved (see Appendix B) that the N given in point 1 of the VRC Algorithm is bounded by

$$N \leq \left\lfloor \frac{2}{\bar{\varepsilon}} \left(d + \ln \frac{1}{\beta} + 1 \right) + \frac{4}{\bar{\varepsilon}} \ln \left(\frac{2}{\bar{\varepsilon}} \left(d + \ln \frac{1}{\beta} + 1 \right) \right) \right\rfloor + 1, \qquad (8)$$

where this formula is valid for any value of parameter $\alpha\%$. (8) exhibits an approximately linear dependence on $\frac{1}{\bar{\varepsilon}}$ and a logarithmic dependence on $\frac{1}{\beta}$. Thus, β can be made very small (e.g. $\beta = 10^{-7}$ or even smaller) with no significant increase of the computational complexity.
One remarkable fact is that the design problem to which VRC is applied only enters (8) through d, the size of the design variables; the dependence on d is linear. Instead, N does not depend on the uncertainty set Δ. This is in contrast with approximation schemes based on a gridding of Δ, an approach which suffers from the so-called curse of dimensionality. This opens up the possibility of a wide usage of the VRC Algorithm irrespective of the form of the uncertainty entering the problem.

performance violation parameter ε_k
ε_k given by point 4 of the VRC algorithm is bounded by

$$\varepsilon_k \leq \frac{k}{N} + \frac{d + h + \sqrt{h^2 + 2(d+k)h}}{N}, \qquad (9)$$

where

$$h = \ln(\bar{k}+1) + \ln \frac{1}{\beta} + d \cdot \left[1 + \ln \frac{d+k}{d} \right]$$

[6] N can be computed via the `betainc` function of MATLAB, a complete code is provided in Appendix D.

[7] Appendix E provides a MATLAB code for this computation.

(see Appendix C for a proof).

(9) reveals important features of ε_k. The first term in the bound is $\frac{k}{N}$, the empirical violation of the solution at level k. Due to stochastic fluctuation, one cannot expect that the real violation be below $\frac{k}{N}$ with high confidence $1-\beta$, and the second term accounts for this.

As N increases, the second term goes to zero approximately as $O(\frac{1}{\sqrt{N}})$, so that, for $k = \gamma \cdot N$, ε_k approaches γ as N grows. This behavior is depicted in Figure 9, where ε_k

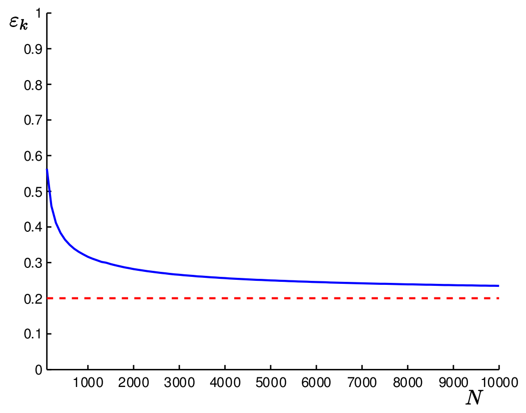

Fig. 9. ε_k as a function of N for $k = 0.2 \cdot N$ ($d = 2$, $\beta = 10^{-7}$, and $\overline{k} = 0.2 \cdot N$).

given by (7) is profiled against N when $k = 0.2 \cdot N$ (other parameters were $d = 2$, $\beta = 10^{-7}$, and $\overline{k} = 0.2 \cdot N$).

C. An easy to implement VRC Algorithm

In this section, we present a simplified version of the VRC Algorithm that requires less intervention by the user and makes VRC more practical to implement.

A first simplification is obtained by providing explicit values for N and ε_k. The idea is as follows. An inspection of the original VRC Algorithm reveals that the result in Theorem 1 continues to hold if one chooses

i. any $N \geq d$ that satisfies the inequality in point 1 of the VRC Algorithm, not necessarily the smallest one;
ii. ε_k values bigger than those given by the solution of the equation (7) in point 4 of the VRC Algorithm.

Plainly, some price is paid by choosing N and ε_k as indicated in i. and ii., both in terms of an increased computational complexity (N is larger than required), as well as because the probabilistic guarantees ε_k become looser. The choices of N and ε_k that we suggest to use are given by the right-hand-side of equations (8) and (9).

To further streamline the usage of the algorithm, we drop β from the set of inputs the user has to assign. Indeed, this parameter has a minor impact on the computational complexity of the algorithm and can therefore be set to a fixed low value. Specifically, we picked $\beta = 10^{-7}$, a number small enough to be negligible for practical purposes.[8]

[8] 10^{-7} is below the proportion of airplane crashes over the number of flights. So, traveling by plane entails neglecting probabilities of this order of magnitude.

The following "VRC – easy to implement Algorithm" implements the above simplifications. Inputs $\bar{\varepsilon}$ and $\alpha\%$ and all outputs have the same meaning as in the original VRC Algorithm.

VRC – easy to implement Algorithm
```
INPUT: $\bar{\varepsilon}$, $\alpha\%$
OUTPUT: $\theta_k^*$, $\ell_k^*$, $\varepsilon_k$

1: set
     $N = \left\lfloor \frac{2}{\bar{\varepsilon}}(d+17.2) + \frac{4}{\bar{\varepsilon}} \ln\left(\frac{2}{\bar{\varepsilon}}(d+17.2)\right) \right\rfloor + 1;$
                                        % ($\lfloor\cdot\rfloor$ = integer part)
   let $\overline{k} := \lfloor \alpha\% \cdot N \rfloor$;
2: sample $N$ independent scenarios $\delta^{(1)}, \delta^{(2)}, \ldots, \delta^{(N)}$
   from $\Delta$ according to probability Pr;
3: run the "Procedure for computing $\theta_k^*$ and $\ell_k^*$ for $k = 0, 1, \ldots, \overline{k}$";
   RETURN $\theta_k^*$ and $\ell_k^*$ for $k = 0, 1, \ldots, \overline{k}$;
4: for $k = 0, 1, \ldots, \overline{k}$ set
     $\varepsilon_k = \frac{k}{N} + \frac{d+h+\sqrt{h^2+2(d+k)h}}{N}$,
   where
     $h = \ln(\overline{k}+1) + 16.2 + d \cdot \left[1 + \ln\frac{d+k}{d}\right]$;
   RETURN $\varepsilon_k$ for $k = 0, 1, \ldots, \overline{k}$.
```

IV. A Simulation Example

An example is presented to illustrate the methodology introduced in this paper. The example is simple enough so that different aspects can be easily explained and visualized.

Consider the following ARMA (Auto-Regressive Moving-Average) system

$$y_{t+1} = ay_t + bu_t + c_1 w_t + c_2 w_{t-1}, \tag{10}$$

where u_t and y_t are input and output, and w_t is a $WN(0,1)$ (white noise with zero mean and unitary

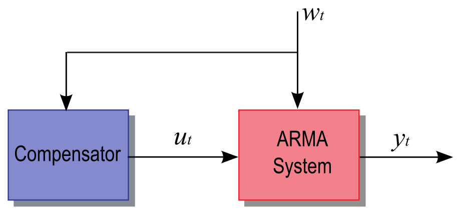

Fig. 10. The feed-forward compensation scheme.

variance) disturbance; a, b, c_1, and c_2 are real parameters, with $|a| < 1$ (stability condition) and $b \neq 0$ (controllability condition), whose values are not precisely known.

We assume that w_t is measured, and the objective is to design a feed-forward compensator with structure

$$u_t = \theta_1 w_t + \theta_2 w_{t-1}$$

that minimizes the asymptotic variance of y_t, see Figure 10.

If the system parameters a, b, c_1, and c_2 *were known*, an optimal compensator would be easily found. Indeed, substituting $u_t = \theta_1 w_t + \theta_2 w_{t-1}$ in (10) gives

$$y_{t+1} = ay_t + (c_1 + b\theta_1)w_t + (c_2 + b\theta_2)w_{t-1},$$

from which the expression for the asymptotic variance of y_t is computed as

$$E[y_t^2] = \frac{(c_1 + b\theta_1)^2 + (c_2 + b\theta_2)^2 + 2a(c_1 + b\theta_1)(c_2 + b\theta_2)}{1 - a^2}.$$

Hence, the values of θ_1 and θ_2 minimizing $E[y_t^2]$ are seen to be

$$\theta_1 = -\frac{c_1}{b} \quad \text{and} \quad \theta_2 = -\frac{c_2}{b}, \qquad (11)$$

resulting in $E[y_t^2] = 0$.

On the other hand, the system parameter values are not always available in practical situations. More realistically, the parameters are only partially known, and they take value in a given uncertainty set Δ, so that the choice of the compensator parameters θ_1 and θ_2 have to be made taking into account the different dynamical behaviors that the system can possibly have.
As an example, suppose that δ has two components σ_1 and σ_2 both ranging in $[-1, 1]$, that is $\delta = (\sigma_1, \sigma_2)$ and $\Delta = [-1, 1]^2$, and that the system parameters are expressed as:

$$\begin{aligned}
a &= \frac{3.5\sigma_1^2 - 0.2}{3\sigma_1^2 + 0.3} \cdot (0.32\sigma_1 + 0.6),\\
b &= 1 + \frac{\sigma_1\sigma_2^2}{10},\\
c_1 &= \frac{-0.01 + (\sigma_1 + \sigma_2^2)^2}{0.02 + (\sigma_1 + \sigma_2^2)^2} \cdot \left(1 - \frac{(\sigma_1 - 1)(\sigma_2 - 1)}{2}\right),\\
c_2 &= \frac{0.05}{0.025 + (\sigma_1 + \sigma_2 - 2)^2}.
\end{aligned}$$

The nominal values for σ_1 and σ_2 are $\sigma_1^{nom} = 0$ and $\sigma_2^{nom} = 0$ corresponding to $a^{nom} = -0.4$, $b^{nom} = 1$, $c_1^{nom} = -0.25$, and $c_2^{nom} = 0.0124$. Based on (11), the ensuing nominal compensator has parameters $\theta_1^{nom} = 0.25$ and $\theta_2^{nom} = -0.0124$. Figure 11 shows the output obtained

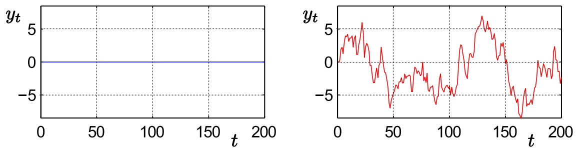

Fig. 11. Outputs obtained with the nominal compensator (left: nominal system; right: perturbed system).

when this compensator is connected to the *nominal system* and to another system (*perturbed system*) picked at random in the uncertainty domain. The dramatic deterioration in performance for the latter does not come as a surprise since the nominal compensator is conceived with no concern for uncertainty.

Moving to VRC we set $\bar{\varepsilon} = 0.5\%$, $\beta = 10^{-7}$, $\alpha\% = 3\%$, and run the VRC Algorithm in Section III-A where Pr was uniform over $[-1, 1]^2$. N was $N = 5427$. The obtained performance-violation plot is displayed in Figure 1. k by k the plot offers the user different trade-off choices: the solid curve represents the value of $E[y_t^2]$ which is guaranteed for all systems but an ε_k proprotion, as displayed by the dashed curve.

Based on an inspection of the curves, we selected $k = 60$, a choice which is largely subjective and others could have opted for a different choice. With this choice, $\varepsilon_{60} = 2.5\%$ and $\ell^*_{60} = 1.42$, with an improvement of 76% over the initial performance value of 6.04 obtained for $k = 0$. The compensator parameters were $\theta^*_{1,60} = -0.24$ and $\theta^*_{2,60} = -0.59$. According to Theorem 1, with probability $1 - \beta = 1 - 10^{-7}$ (in practice with probability 1) the compensator $u_t = -0.24w_t - 0.59w_{t-1}$ guarantees that $E[y_t^2] \leq 1.42$ for all plants in the uncertainty set Δ but a small proportion of size no more than $\varepsilon_{60} = 2.5\%$. Figure

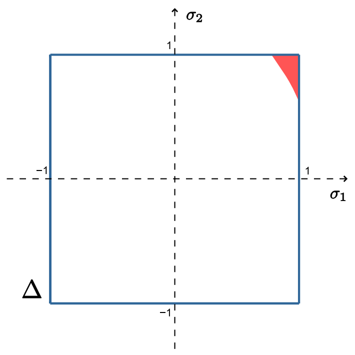

Fig. 12. Region where the cost value is not guaranteed.

12 depicts the region in the Δ domain where the cost value 1.42 is not met, the volume of the region is 1.2% of the total volume of the uncertainty domain $[-1, 1]$, below threshold 2.5%. What is fundamental is that the VRC Algorithm has been able to determine a region of small volume whose elimination guarantees a large improvement in the cost value. This has been achieved by letting the problem speak, while an a priori choice of Δ_ε obtained e.g. by a re-sizing of the Δ domain as discussed in Section II-C would have instead produced little benefit.[9]

Figures 13–15 depict the value of $E[y_t^2]$ achieved for

[9]For completeness, we also considered a re-sizing of $\Delta = [-1, 1]^2$ to $[-0.9874, 0.9874]^2$ (which leaves out a 2.5% of the total volume), and found that the robust compensator for $[-0.9874, 0.9874]^2$ achieved performance 5.46.

the various systems in Δ by the nominal compensator (Figure 13), and the compensators obtained by VRC for $k = 0$ (Figure 14) and for $k = 60$ (Figure 15). In figure

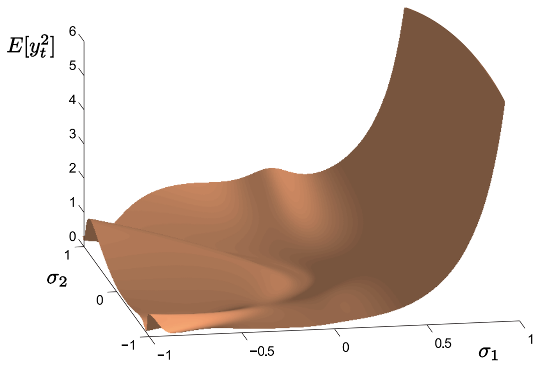

Fig. 13. $E[y_t^2]$ for nominal compensator.

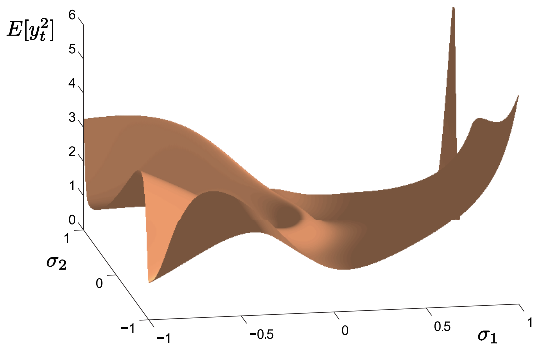

Fig. 14. $E[y_t^2]$ for compensator obtained for $k = 0$.

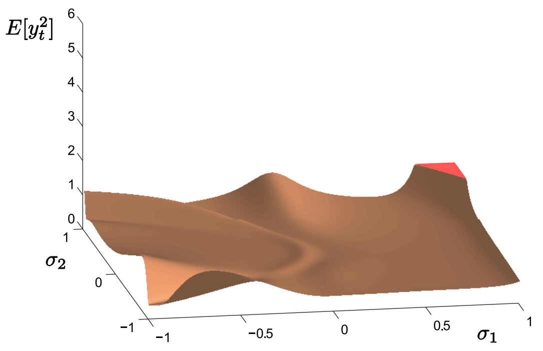

Fig. 15. $E[y_t^2]$ for compensator obtained for $k = 60$.

15, the flat zone close to the corner $(1, 1)$ corresponds to the region where the performance is not guaranteed (so that $E[y_t^2]$ is in reality above the cutting value 1.42 represented in the figure).

Injecting a disturbance in the nominal and in the perturbed systems as done in Figure 11 but this time using the compensator obtained for $k = 60$ we measured the outputs shown in Figure 16. Not surprisingly, the performance with

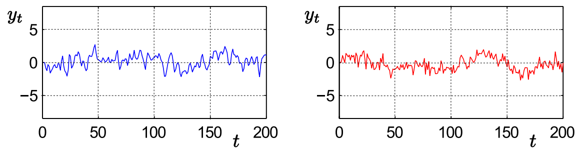

Fig. 16. Outputs obtained with the compensator obtained for $k = 60$ (left: nominal system; right: perturbed system).

the nominal system becomes worse, while the performance with the perturbed system improves, in line with the provided guarantees.

Appendix

A. Proof of Theorem 1

To shorten the notation let $\boldsymbol{\delta} := (\delta^{(1)}, \dots, \delta^{(N)})$. Note that θ_k^*, ℓ_k^* are stochastic elements depending on $\boldsymbol{\delta}$, although such a dependence is not explicitly indicated to ease the reading. Define

$$B = \Big\{ \boldsymbol{\delta} \in \Delta^N : \quad \exists k \in \{0, 1, \dots, \overline{k}\} \text{ such that } V(\theta_k^*, \ell_k^*) > \varepsilon_k \Big\},$$

i.e. B is the set of "bad" multi-extractions $\boldsymbol{\delta}$ from Δ^N leading for some k to a violation bigger than ε_k. In these notations, the theorem statement writes $\mathrm{Pr}^N\{B\} \leq \beta$.
Letting

$$B_k = \Big\{ \boldsymbol{\delta} \in \Delta^N : \ V(\theta_k^*, \ell_k^*) > \varepsilon_k \Big\}$$

be the event where the performance violation probability for a given k is bigger than ε_k, we have that $B = \bigcup_{k=0}^{\overline{k}} B_k$, leading to the bound

$$\mathrm{Pr}^N\{B\} \leq \sum_{k=0}^{\overline{k}} \mathrm{Pr}^N\{B_k\}. \tag{12}$$

The theorem will be proved by computing $\mathrm{Pr}^N\{B_k\}$ for $k = 0, 1, \dots, \overline{k}$, and then by summing over k.
Anticipating the result, we will show that

$$\mathrm{Pr}^N\{B_k\} \leq \binom{d+k}{k} \sum_{i=0}^{d+k} \binom{N}{i} \varepsilon_k^i (1-\varepsilon_k)^{N-i}, \tag{13}$$

so that the thesis follows by substitution in (12):

$$\begin{aligned} \mathrm{Pr}^N\{B\} &\leq \sum_{k=0}^{\overline{k}} \left[\binom{d+k}{k} \sum_{i=0}^{d+k} \binom{N}{i} \varepsilon_k^i (1-\varepsilon_k)^{N-i} \right] \\ &= [\text{thanks to (7)}] \\ &= \sum_{k=0}^{\overline{k}} \frac{\beta}{\overline{k}+1} = \beta. \end{aligned}$$

Thus, to complete the proof, we have to establish the fundamental relation (13).

Fix a value for k. Given a subset $I = \{i_1, \dots, i_k\}$ of k indexes from $\{1, \dots, N\}$ ($I = \emptyset$, the empty set, if $k = 0$), let θ_I^* be the solution to the minimax problem where the scenarios with index in I have been removed, i.e.

$$\theta_I^* := \arg\min_{\theta \in \mathbb{R}^d} \max_{i \in \{1,\dots,N\}-I} \ell(\theta, \delta^{(i)}), \tag{14}$$

and let ℓ_I^* be the corresponding cost value, i.e.

$$\ell_I^* := \max_{i \in \{1,\dots,N\}-I} \ell(\theta_I^*, \delta^{(i)}).$$

Moreover, let

$$\Delta_I^N = \{\boldsymbol{\delta} \in \Delta^N : \ \ell(\theta_I^*, \delta^{(i)}) > \ell_I^*, \quad \forall i \in I\}.$$

Thus, a $\boldsymbol{\delta}$ is in Δ_I^N if the performance value ℓ_I^* is violated in correspondence of all the scenarios in I that have been removed in the construction (14) of θ_I^*.
Since the pair θ_k^*, ℓ_k^* generated by the VRC Algorithm are such that the performance value ℓ_k^* is violated in correspondence of exactly k scenarios, it is clear that $(\theta_k^*, \ell_k^*) = (\theta_I^*, \ell_I^*)$ for some I such that $\boldsymbol{\delta} \in \Delta_I^N$. Thus,

$$\begin{aligned} B_k &= \{\boldsymbol{\delta} \in \Delta^N : \ V(\theta_k^*, \ell_k^*) > \varepsilon_k\} \\ &\subseteq \bigcup_{I \in \mathcal{I}} \{\boldsymbol{\delta} \in \Delta_I^N : \ V(\theta_I^*, \ell_I^*) > \varepsilon_k\} \end{aligned} \tag{15}$$

up to a zero probability set, where $\mathcal{I}$ is the collection of all possible choices of k indexes from $\{1, \dots, N\}$.
The sought bound for $\mathrm{Pr}^N\{B_k\}$ is now obtained by first bounding $\mathrm{Pr}^N\{\boldsymbol{\delta} \in \Delta_I^N : \ V(\theta_I^*, \ell_I^*) > \varepsilon_k\}$, and then summing over $I \in \mathcal{I}$.

Fix an $I = \{i_1, \dots, i_k\}$, and write

$$\begin{aligned} &\mathrm{Pr}^N\{\boldsymbol{\delta} \in \Delta_I^N : \ V(\theta_I^*, \ell_I^*) > \varepsilon_k\} \\ &\quad = \int_{(\varepsilon_k, 1]} \mathrm{Pr}^N \left\{ \Delta_I^N \ \middle| \ V(\theta_I^*, \ell_I^*) = v \right\} \, \mathrm{d}F_V(v) \\ &\quad = \int_{(\varepsilon_k, 1]} \mathrm{Pr}^N \Big\{ \ell(\theta_I^*, \delta^{(i)}) > \ell_I^*, \ \forall i \in I \ \Big| \\ &\qquad\qquad V(\theta_I^*, \ell_I^*) = v \Big\} \, \mathrm{d}F_V(v), \end{aligned} \tag{16}$$

where F_V is the cumulative distribution function of the random variable $V(\theta_I^*, \ell_I^*)$, and $\mathrm{Pr}^N\{\Delta_I^N \mid V(\theta_I^*, \ell_I^*) = v\}$ is the conditional probability of the event Δ_I^N under the condition that $V(\theta_I^*, \ell_I^*) = v$ (see eq.(17), § 7, Chapter II of [42]).
To evaluate the integrand in (16), remind that $V(\theta_I^*, \ell_I^*) = v$ means that $\mathrm{Pr}\{\delta : \ \ell(\theta_I^*, \delta) > \ell_I^*\} = v$; then, owing to the independence of the scenarios, the integrand equals v^k. Substituting in (16) yields

$$\mathrm{Pr}^N\{\boldsymbol{\delta} \in \Delta_I^N : \ V(\theta_I^*, \ell_I^*) > \varepsilon_k\} = \int_{(\varepsilon_k, 1]} v^k \, \mathrm{d}F_V(v). \tag{17}$$

To proceed, we have now to appeal to a result on F_V from [14]:

$$F_V(v) \ge \bar{F}_V(v) := 1 - \sum_{i=0}^{d} \binom{N-k}{i} v^i (1-v)^{N-k-i}.\ ^{10}$$

This inequality is tight, i.e. it holds with equality for a whole class of problems, that called "fully-supported" in [14], Definition 3.
Now, the integrand v^k in (17) is an increasing function of v, so that $F_V(v) \ge \bar{F}_V(v)$ implies that

$$\int_{(\varepsilon_k, 1]} v^k \, \mathrm{d}F_V(v) \le \int_{(\varepsilon_k, 1]} v^k \, \mathrm{d}\bar{F}_V(v). \tag{18}$$

This can be verified by the calculation:

$$\begin{aligned} &\int_{(\varepsilon_k, 1]} v^k \, \mathrm{d}F_V(v) \\ &\quad = \ \text{[Theorem 11, §6, Chapter II of [42]]} \\ &\quad = \ 1 - \varepsilon_k^k F_V(\varepsilon_k) - \int_{(\varepsilon_k, 1]} F_V(v) k v^{k-1} \, \mathrm{d}v \\ &\quad \le \ 1 - \varepsilon_k^k \bar{F}_V(\varepsilon_k) - \int_{(\varepsilon_k, 1]} \bar{F}_V(v) k v^{k-1} \, \mathrm{d}v \\ &\quad = \ \int_{(\varepsilon_k, 1]} v^k \, \mathrm{d}\bar{F}_V(v). \end{aligned}$$

Hence, $\mathrm{Pr}^N\{\boldsymbol{\delta} \in \Delta_I^N : \ V(\theta_I^*, \ell_I^*) > \varepsilon_k\}$ can finally be bounded as follows:

$$\begin{aligned} &\mathrm{Pr}^N\{\boldsymbol{\delta} \in \Delta_I^N : \ V(\theta_I^*, \ell_I^*) > \varepsilon_k\} \\ &\quad \le \ \text{[use (17) and (18)]} \\ &\quad \le \ \int_{(\varepsilon_k, 1]} v^k \, \mathrm{d}\bar{F}_V(v) \\ &\quad = \ \text{[the density of } \bar{F}_V \text{ is} \\ &\qquad\ (N-k-d) \binom{N-k}{d} v^d (1-v)^{N-k-d-1}] \\ &\quad = \ \int_{(\varepsilon_k, 1]} (N-k-d) \binom{N-k}{d} v^{k+d} (1-v)^{N-k-d-1} \, \mathrm{d}v \\ &\quad = \ \text{[integration by parts]} \\ &\quad = \ \frac{\binom{N-k}{d}}{\binom{N}{d+k}} \sum_{i=0}^{d+k} \binom{N}{i} \varepsilon_k^i (1-\varepsilon_k)^{N-i}. \end{aligned} \tag{19}$$

[10]To be precise, this result follows from Theorem 1 in [14] by noting that the minimax problem $\min_{\theta \in \mathbb{R}^d} \max_{i \in \{1,\dots,N\}-I} \ell(\theta, \delta^{(i)})$ can be rewritten as

$$\begin{aligned} &\min_{\theta \in \mathbb{R}^d, h \in \mathbb{R}} \ h \\ &\text{subject to: } \ell(\theta, \delta^{(i)}) \le h, \ i \in \{1, \dots, N\} - I, \end{aligned}$$

i.e. a program with $d+1$ optimization variables and $N-k$ constraints.

To conclude the proof, go back to (15) and write:

$$\begin{aligned}
\Pr{}^N\{B_k\} &\leq \sum_{I\in\mathcal{I}} \Pr{}^N\{\boldsymbol{\delta}\in\Delta_I^N : V(\theta_I^*,\ell_I^*)>\varepsilon_k\} \\
&= [\mathcal{I} \text{ contains } \binom{N}{k} \text{ choices}] \\
&= \binom{N}{k}\cdot \Pr{}^N\{\boldsymbol{\delta}\in\Delta_I^N : V(\theta_I^*,\ell_I^*)>\varepsilon_k\} \\
&\leq [\text{use (19)}] \\
&\leq \binom{N}{k}\frac{\binom{N-k}{d}}{\binom{N}{d+k}}\sum_{i=0}^{d+k}\binom{N}{i}\varepsilon_k^i(1-\varepsilon_k)^{N-i} \\
&= \binom{d+k}{k}\sum_{i=0}^{d+k}\binom{N}{i}\varepsilon_k^i(1-\varepsilon_k)^{N-i},
\end{aligned}$$

which is (13). $\ast$

B. *Proof of* (8)

Letting

$$M := \left\lfloor \frac{2}{\bar\varepsilon}\left(d+\ln\frac{1}{\beta}+1\right)+\frac{4}{\bar\varepsilon}\ln\left(\frac{2}{\bar\varepsilon}\left(d+\ln\frac{1}{\beta}+1\right)\right)\right\rfloor+1,$$

we have

$$\begin{aligned}
M &\geq \frac{2}{\bar\varepsilon}\left(d+\ln\frac{1}{\beta}+1\right)+\frac{4}{\bar\varepsilon}\ln\left(\frac{2}{\bar\varepsilon}\left(d+\ln\frac{1}{\beta}+1\right)\right) \\
&= [\text{put } \mu = d+\ln\frac{1}{\beta}+1] \\
&= \frac{2\mu}{\bar\varepsilon}+\frac{2}{\bar\varepsilon}\cdot 2\cdot\ln\left(\frac{2\mu}{\bar\varepsilon}\right) \\
&\geq [\text{since } 2\geq\frac{\mu}{\mu-1}] \\
&\geq \frac{2\mu}{\bar\varepsilon}+\frac{2}{\bar\varepsilon}\cdot\frac{\mu}{\mu-1}\ln\left(\frac{2\mu}{\bar\varepsilon}\right) \\
&\geq \frac{2}{\bar\varepsilon}\cdot\frac{\mu}{\mu-1}\left[\mu-1+\ln\left(\frac{2\mu}{\bar\varepsilon}\right)\right] \\
&= \frac{1}{\frac{\bar\varepsilon}{2}-\frac{1}{\frac{2\mu}{\bar\varepsilon}}}\left[\mu-1+\ln\left(\frac{2\mu}{\bar\varepsilon}\right)\right],
\end{aligned}$$

which implies

$$\begin{aligned}
M\frac{\bar\varepsilon}{2} &\geq \mu-1+\ln\left(\frac{2\mu}{\bar\varepsilon}\right)+\frac{1}{\frac{2\mu}{\bar\varepsilon}}M \\
&\geq [\text{since } 1-\frac{2\mu}{\bar\varepsilon}\leq 0] \\
&\geq \mu-1+\ln\left(\frac{2\mu}{\bar\varepsilon}\right)+\frac{1}{\frac{2\mu}{\bar\varepsilon}}(M+1-\frac{2\mu}{\bar\varepsilon}) \\
&\geq [\text{since } \ln(x)+\frac{1}{x}(y-x)\geq\ln(y)] \\
&\geq \mu-1+\ln(M+1) \\
&= d+\ln\frac{1}{\beta}+\ln(M+1).
\end{aligned}$$

Hence,

$$\begin{aligned}
M\frac{\bar\varepsilon}{2}-d &\geq \ln\frac{M+1}{\beta} \\
&\Downarrow \quad [\text{since } \frac{(M\bar\varepsilon-d)^2}{2M\bar\varepsilon}\geq M\frac{\bar\varepsilon}{2}-d] \\
\frac{(M\bar\varepsilon-d)^2}{2M\bar\varepsilon} &\geq \ln\frac{M+1}{\beta} \\
&\Downarrow \\
\mathrm{e}^{-\frac{(M\bar\varepsilon-d)^2}{2M\bar\varepsilon}} &\leq \frac{\beta}{M+1},
\end{aligned}$$

which, by applying the Chernoff's bound for the Binomial tail, see [17] or [49], gives

$$\sum_{i=0}^{d}\binom{M}{i}\bar\varepsilon^i(1-\bar\varepsilon)^{M-i}\leq\frac{\beta}{M+1}\leq\frac{\beta}{\alpha\%M+1}.$$

Thus, M satisfies (6); since the N selected in point 1 of the VRC Algorithm is the smallest integer satisfying (6), we have $N\leq M$. $\ast$

C. *Proof of* (9)

If $\varepsilon_k<\frac{k+d}{N}$, then (9) is trivially true.
If instead $\varepsilon_k\geq\frac{k+d}{N}$, then the well-known Chernoff bound for the Binomial tail (see [17] or [49]) applies, yielding

$$\sum_{i=0}^{d+k}\binom{N}{i}\varepsilon_k^i(1-\varepsilon_k)^{N-i}\leq e^{-\frac{(N\varepsilon_k-d-k)^2}{2N\varepsilon_k}}.$$

Moreover, it holds that

$$\binom{d+k}{k}\leq\frac{(d+k)^d\mathrm{e}^d}{d^d},$$

and so the left-hand-side of (7) is bounded by $\frac{(d+k)^d\mathrm{e}^d}{d^d}\mathrm{e}^{-\frac{(N\varepsilon_k-d-k)^2}{2N\varepsilon_k}}$. Hence,

$$\frac{(d+k)^d\mathrm{e}^d}{d^d}\cdot\mathrm{e}^{-\frac{(\varepsilon_kN-d-k)^2}{2N\varepsilon_k}}\geq\frac{\beta}{\overline{k}+1}.$$

This inequality can be rewritten as

$$(N\varepsilon_k-d-k)^2\leq 2N\varepsilon_k\cdot\ln\left[\frac{\overline{k}+1}{\beta}\cdot\frac{(d+k)^d\mathrm{e}^d}{d^d}\right],$$

which, made explicit for ε_k, gives

$$\varepsilon_k\leq\frac{k}{N}+\frac{d+h+\sqrt{h^2+2(d+k)h}}{N},$$

where we have posed

$$\begin{aligned}
h &= \ln\left[\frac{\overline{k}+1}{\beta}\cdot\frac{(d+k)^d\mathrm{e}^d}{d^d}\right] \\
&= \ln(\overline{k}+1)+\ln\frac{1}{\beta}+d\cdot\left[1+\ln\frac{d+k}{d}\right].
\end{aligned}$$

$\ast$

D. MATLAB code to compute N

Function inputs: `eps` $= \bar{\varepsilon}$; `alph` $= \alpha\%$; `bet` $= \beta$; `d` $=$ no. of design variables d.
Notes: in the function, N is computed by bisection; `N1` is the initial lower-bound, while `N2` is the initial upper-bound and corresponds to formula (8).

function findN

```
function N = findN(eps,alph,bet,d)

N1 = d;
N2 = floor( 2/eps*(d+log(1/bet)+1) +
4/eps*log(2/eps*(d+log(1/bet)+1)) ) + 1;

while N2-N1>1
   N = floor((N1+N2)/2);
   if betainc(1-eps,N-d,d+1)
   > bet/(alph*N+1)
   N1=N;
   else
   N2=N;
   end
end

N = N2;
```

E. MATLAB code to compute ε_k

Function inputs: `k` $=$ no. of removed scenarios k; `N` $=$ no. of scenarios N; `kmax` $= \bar{k}$; `bet` $= \beta$; `d` $=$ no. of design variables d.

function findepsk

```
function epsk = findepsk(k,N,kmax,bet,d)

eps1 = 0;
eps2 = 1;
coeff = 1/(d*beta(k+1,d));

while eps2-eps1 > 1e-10
   epsk = (eps1+eps2)/2;
   if coeff*betainc(1-epsk,N-k-d,d+k+1)
   > bet/(kmax+1)
   eps1 = epsk;
   else
   eps2 = epsk;
   end
end

epsk = eps2;
```

References

[1] T. Alamo, R. Tempo, and Camacho. Revisiting statistical learning theory for uncertain feasibility and optimization problems. In *Proceedings of the 46^{th} IEEE conference on decision and control*, New Orleans, LA, USA, 2007.

[2] T. Alamo, R. Tempo, and E.F. Camacho. Randomized strategies for probabilistic solutions of uncertain feasibility and optimization problems. *IEEE Transactions on Automatic Control*, 54:2545–2559, 2009.

[3] P. Apkarian and H.D. Tuan. Parameterized LMIs in control theory. *SIAM Journal on Control and Optimization*, 38(4):1241–1264, 2000.

[4] P. Apkarian, H.D. Tuan, and J. Bernussou. Continuous-time analysis, eigenstructure assignment, and h_2 synthesis with enhanced linear matrix inequalities (LMI) characterizations. *IEEE Transactions on Automatic Control*, 46(12):1941–1946, 2001.

[5] K.J. Åström. *Introduction to stochastic control theory*. Academic Press, Inc., London, UK, 1970.

[6] B.R. Barmish and C.M. Lagoa. The uniform distribution: A rigorous justification for its use in robustness analysis. *Mathematics of Control, Signals, and Systems*, 10:203–222, 1997.

[7] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, volume 15 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, 1994.

[8] G. Calafiore and M.C. Campi. Uncertain convex programs: randomized solutions and confidence levels. *Mathematical Programming*, 102(1):25–46, 2005.

[9] G. Calafiore and M.C. Campi. The scenario approach to robust control design. *IEEE Transactions on Automatic Control*, 51(5):742–753, 2006.

[10] G. Calafiore and F. Dabbene. A probabilistic framework for problems with real structured uncertainty in systems and control. *Automatica*, 38(8):1265–1276, 2002.

[11] G. Calafiore, F. Dabbene, and R. Tempo. Randomized algorithms for probabilistic robustness with real and complex structured uncertainty. *IEEE Transactions on Automatic Control*, 45(12):2218–2235, 2000.

[12] G. Calafiore and B.T. Polyak. Stochastic algorithms for exact and approximate feasibility of robust LMIs. *IEEE Transactions of Automatic Control*, 46:1755–1759, 2001.

[13] M.C. Campi and S. Garatti. Modulating robustness in robust control: making it easy through randomization. In *Proceedings of the 46^{th} IEEE conference on decision and control*, New Orleans, LA, USA, 2007.

[14] M.C. Campi and S. Garatti. The exact feasibility of randomized solutions of uncertain convex programs. *SIAM Journal on Optimization*, 19(3):1211–1230, 2008.

[15] M.C. Campi, S. Garatti, and M. Prandini. The scenario approach for systems and control design. *Annual Reviews in Control*, 33:149–157, 2000.

[16] M.C. Campi and M. James. Nonlinear discrete-time risk-sensitive optimal control. *International Jornal of Robust and Nonlinear Control*, 6:1–19, 1996.

[17] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–507, 1952.

[18] P. Colaneri, J.C. Geromel, and A. Locatelli. *Control theory and design: an RH_2 and RH_∞ viewpoint*. Academic Press, San Diego, CA, USA, 1997.

[19] D. Dentcheva. Optimization models with probabilistic constraints. In G. Calafiore and F. Dabbene, editors, *Probabilistic and Randomized Methods for Design under Uncertainty*, London, 2006. Springer-Verlag.

[20] J.C. Doyle. Guaranteed margins for LQG regulators. *IEEE Transaction on Automatic Control*, 23:756–757, 1978.

[21] Y. Fujisaki, F. Dabbene, and R. Tempo. Probabilistic robust design of LPV control systems. *Automatica*, 39:1323–1337, 2003.

[22] Y. Fujisaki and Y. Oishi. Guaranteed cost regulator design: a probabilistic solution and a randomized algorithm. *Automatica*, 43:317–324, 2007.

[23] P. Gahinet. Explicit controller formulas for LMI-based H_∞ synthesis. *Automatica*, 32(7):1007–1014, 1996.

[24] M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008.

[25] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. http://cvxr.com/cvx, April 2010.

[26] M. Green and D.J.N. Limebeer. *Linear robust control*. Prentice-Hall, Upper Saddle River, NJ, USA, 1995.

[27] D.H. Jacobson. Optimal stochastic linear systems with exponential performance criteria and their relation to deterministic differential games. *IEEE Transactions on Automatic Control*, 18:124–131, 1973.

[28] M.R. James. Asymptotic analysis of nonlinear stochastic risk sensitive control and differential games. *Mathematics of Control, Signals, and Systems*, 5:401–417, 1992.

[29] S. Kanev, B. De Schutter, and M. Verhaegen. An ellipsoid algorithm for probabilistic robust controller design. *Systems & Control Letters*, 49:365–375, 2003.

[30] P.P. Khargonekar and A. Tikku. Randomized algorithms for robust control analysis have polynomial time complexity. In *Proceedings of the 35^{th} IEEE Conference on Decision and Control*, Kobe, Japan, 1996.

[31] P.R. Kumar and P. Varaiya. *Stochastic systems: estimation, identification and adaptive control*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1986.

[32] C.M. Lagoa and B.R. Barmish. Distributionally robust Monte Carlo simulation: A tutorial survey. In *Proceedings of the 15th IFAC World Congress*, Barcelona, Spain, 2002.

[33] C.M. Lagoa, P.S. Shcherbakov, and B.R. Barmish. Probabilistic enhancement of classical robustness margins: The unirectangularity concept. *Systems and Control Letters*, 35:31–43, 1998.

[34] J. Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[35] Y. Oishi. Polynomial-time algorithms for probabilistic solutions of parameter-dependent linear matrix inequalities. *Automatica*, 43:538–545, 2007.

[36] B.T. Polyak and R. Tempo. Probabilistic robust design with linear quadratic regulators. *Systems & Control Letters*, 43:343–353, 2001.

[37] A. Prèkopa. *Stochastic programming*. Kluwer, Boston, MT, USA, 1995.

[38] A. Prèkopa. Probabilistic programming. In A. Ruszczyǹski and A. Shapiro, editors, *Stochastic Programming*, volume 10 of *handbooks in operations research and management science*, London, UK, 2003. Elsevier.

[39] L.R. Ray and R.F. Stengel. A Monte Carlo approach to the analysis of control system robustness. *Automatica*, 29:229–236, 1993.

[40] C.W. Scherer. Relaxations for robust linear matrix inequality problems with verifications for exactness. *SIAM Journal on Matrix Analysis and Applications*, 27(2):365–395, 2005.

[41] C.W. Scherer. LMI relaxations in robust control. *European Journal of Control*, 12:3–29, 2006.

[42] A.N. Shiryaev. *Probability*. Springer, New York, NY, USA, 1996.

[43] R.F. Stengel. Some effects of parameter variations on the lateral-directional stability of aircraft. *AIAA Journal of Guidance and Control*, 3:124–131, 1980.

[44] R.F. Stengel and L.R. Ray. Stochastic robustness of linear time-invariant control systems. *IEEE Transactions on Automatic Control*, 36:82–87, 1991.

[45] R. Tempo, E.W. Bai, and F. Dabbene. Probabilistic robustness analysis: explicit bounds for the minimum number of samples. In *Proceedings of 35th IEEE Conference on Decision and Control*, Kobe, Japan, 1996.

[46] R. Tempo, E.W. Bai, and F. Dabbene. Probabilistic robustness analysis: explicit bounds for the minimum number of samples. *Systems and control letters*, 30:237–242, 1997.

[47] R. Tempo, G. Calafiore, and F. Dabbene. *Randomized algorithms for analysis and control of uncertain systems*. Springer-Verlag, London, UK, 2005.

[48] V. Vapnik. *Statistical learning theory*. John Wiley and sons, New York, NY, USA, 1998.

[49] M. Vidyasagar. *A Theory of Learning and Generalization*. Springer-Verlag, London, UK, 1997.

[50] M. Vidyasagar. Statistical learning theory and randomized algorithms for control. *IEEE Control Systems Magazine*, 18(6):69–85, 1998.

[51] M. Vidyasagar. Randomized algorithms for robust controller synthesis using statistical learning theory. *Automatica*, 37(10):1515–1528, 2001.

[52] P. Whittle. Risk-sensitive linear/quadratic/gaussian control. *Advances in Applied Probability*, 13:764–777, 1981.

[53] G. Zames. Feedback and optimizal sensitivity: Model reference transformations, multiplicative seminorms, and approximate inverses. *IEEE Transaction on Automatic Control*, 26:301–320, 1981.

[54] K. Zhou, J.C. Doyle, and K. Glover. *Robust and optimal control*. Prentice Hall, Upper Saddle River, NJ, USA, 1996.