Proceedings of the
American Control Conference
Baltimore, Maryland • June 1994

A Framework for Robust Control Based Model Invalidation

Mitchell M. Livstone[1], Munther A. Dahleh[2], and Jay A. Farrell[3]

Abstract

In this paper, the authors establish a connection between robust control design and model validation. In particular, we present a new framework for invalidating a set of models which has been assumed to *"adequately describe"* the unknown underlying process. The technique uses robust control synthesis and observations of a new set of variables, rather than the usual inputs and outputs of the process, to invalidate models. In contrast to existing techniques for model validation where the input/output experiments are specified *a priori*, an important contribution of this scheme is that it automatically generates plant inputs which are necessary to test the model with respect to the goal of trying to achieve some desired performance for the unknown process. Another important byproduct is that if the set is not invalidated (meaning it *may* adequately describe the process) we have a controller that, so far, achieves the desired closed loop performance *for the unknown process.*

1. Introduction

Recently, several researchers have considered the problem of model validation [4, 5, 6, 7]. The motivation comes from the fact that a robust control design is not worth much unless one has some confidence that the plant can be "adequately described" by the uncertain set of models used in the robust control design. Model validation is the process of taking input/output experiments, subsequent to the modeling and/or system identification phase, and deciding whether or not the assumed model set is actually a valid description of the underlying process.

In previous work, the following problem was considered. We are given a model set represented by the pair $(\mathcal{M}_0, \mathcal{D})$, where $\mathcal{M}_0$ is a set of systems and $\mathcal{D}$ is a set of disturbances which enters the system in some assumed way. It is assumed that this model set adequately describes the unknown plant. Additionally, we are given a set of inputs u which were applied to the plant and the resulting outputs y which were observed. The question is *does there exist a process model* $(h, d) \in (\mathcal{M}_0, \mathcal{D})$ *which could have produced (or is consistent with) the experiment pair* (u, y). If the answer is yes, we gain some confidence in our assumed model set. On the other hand, if the answer is no, we conclude that the process can not be adequately described by the assumed model set and thus invalidate the pair $(\mathcal{M}_0, \mathcal{D})$.

In this way, model validation can be thought of as a decision map from model set assumptions $(\mathcal{M}_0, \mathcal{D})$ and experiments (u, y) to the set {yes,no}, and can in general be computationally very difficult to evaluate (see the results in [4, 7]). In the work of Smith et.al., the assumed model set consists of the set $\mathcal{M}_0$ which is a LTI continuous–time nominal system with an H_∞ norm bounded perturbation entering in a linear fractional way, along with $\mathcal{D}$, a set of L_2 norm bounded disturbance entering at the input. Poolla, et.al., consider $\mathcal{M}_0$ to be a discrete–time nominal system with an additive H_∞ norm bounded perturbation and $\mathcal{D}$ as the set of l_∞ norm bounded disturbances added to the output.

In this paper we take the view that model validation should be accomplished in a way that depends on how the model will be used. In the following sections we propose a different way of thinking about model validation which allows us

[1]Graduate Student, EECS Dept., Massachusetts Institute of Technology mml@lids.mit.edu
[2]Associate Professor, EECS Dept., Massachusetts Institute of Technology dahleh@lids.mit.edu
[3]Associate Professor, Dept. of Engineering, University of California, Riverside j.a.farrell@ieee.org

to tackle the above problem when the inputs, u, are not given *a priori*. We then show that given some performance objective with respect to which one defines the term "*adequately describes*", an input, u^*, for model validation can be chosen by solving a corresponding robust control synthesis problem. The model validation step can then be performed on the resulting pair (u^*, y^*). We further show that once the controller is obtained, this step can be performed much more easily by observing the performance variables (w, z) rather than (u^*, y^*) and simply checking if a certain performance was met. In this way we establish a connection between model validation and robust control design by showing that invalidating performance accomplishes model invalidation. This is different from the work of [5] and [7] where the connection is that the structure of $\mathcal{M}_0$ has to be consistent with a robust control design methodology. The development in this paper essentially justifies and formalizes the following procedure which was introduced in [2] as one step of an iterative identification/control scheme.

1. Design a controller, K, that achieves a certain performance level for the model set $(\mathcal{M}_0, \mathcal{D})$.
2. Use K to close the loop around the process P.
3. Monitor the performance variables (w, z) and decide if the performance level was met (i.e., use a test such as $\|z\| \le \gamma\|w\|$).
4. If the answer is no, we conclude that $(\mathcal{M}_0, \mathcal{D})$ is not consistent with (u^*, y^*), or equivalently, does not adequately describe the process *in view of the objectives*. However, if the answer is yes, we have a controller that achieves the performance objective for the given signals w.

2. Preliminaries

We first establish some notation which will be used throughout the paper. The system/noise pair $(\mathcal{M}_0, \mathcal{D})$ represents some model set which is assumed to describe the unknown process P. The input/output relations for P and some model $(h, d) \in (\mathcal{M}_0, \mathcal{D})$ are written as $y = Pu$ and $y = (h, d)u$, respectively. The reason for using this notation is that the development remains general, and we are not forced to write $y = h*u+d$ or $y = h*(u+d)$, for example. Now, assume that some controller K is used to close the loop around the plant. Depending on the performance objectives, one defines some exogenous input w and some output z which is to be kept small relative to w. We assume that w and z lie in some spaces of sequences of p and q dimensional vectors. The relation for w and z is now some LFT of the plant and controller and will be represented as $\mathcal{F}_l(P, K)$. With a slight abuse of notation we will similarly represent the closed loop relation with (h, d) in place of P as $\mathcal{F}_l((h, d), K)$ and the set of relations with $(\mathcal{M}_0, \mathcal{D})$ as $\mathcal{F}_l((\mathcal{M}_0, \mathcal{D}), K)$. Finally, given an exogenous signal w_0, assuming uniqueness of solutions in the closed loop system, we can define a map G which takes (w_0, P, K) into u_0, the input to the plant, which we write as $u_0 = G(w_0, P, K)$. We next establish the way in which we view performance.

When an engineer designs a control system which must satisfy performance defined as an induced norm bound on some transfer function, there are no guarantees that performance will really be met since there is no guarantee that $P \in (\mathcal{M}_0, \mathcal{D})$. In order to test such performance, it is necessary to generate *every* signal in some ball in the input space, which would take infinite time. The engineer can increase his/her confidence by observing the closed loop system for some finite set of exogenous inputs thought to represent the *typical* signals which the system will have to face in the future. Following this philosophy, we view performance in relationship to some *finite collection of finite duration* exogenous signals and the metrics used on the input and output spaces.

One can think of the standard model validation problem in the following way. Given $(\mathcal{M}_0, \mathcal{D}), u$ and $y = Pu$, if $(\mathcal{M}_0, \mathcal{D})$ is inconsistent with (u, y), then we say that $(\mathcal{M}_0, \mathcal{D})$ does not adequately describe P *with respect to the input u*. We next propose a new method of invalidating $(\mathcal{M}_0, \mathcal{D})$ and then show that it is equivalent to invalidating $(\mathcal{M}_0, \mathcal{D})$ with respect to some special input, u^*, which is well defined.

3. Main Results

We assume that we are given a set $\mathcal{W} = \{w_i \in R^{N \times p} \mid i \in [1, M_w]\}$ of exogenous inputs and the goal is to achieve $\|\mathcal{F}_l(P, K)w_i\| \le \gamma\|w_i\| \quad \forall i \in [1, M_w]$. The choice of norms here dictates which control design methodology one will have to utilize [1, 3]. We propose the following alternative way of thinking about model validation, which is similar to the idea in [2].

Definition 3.1 *Given a model set* $(\mathcal{M}_0, \mathcal{D})$ *and a* K *which achieves*

$\|\mathcal{F}_l((\mathcal{M}_0, \mathcal{D}), K)w_i\| \leq \gamma\|w_i\| \ \ \forall i \in [1, M_w]$,

if $\exists j \in [1, M_w]$ *s.t.* $\|\mathcal{F}_l(P, K)w_j\| > \gamma\|w_j\|$,

then *we say that* $(\mathcal{M}_0, \mathcal{D})$ *does not adequately describe* P *in view of the objectives.*

In other words, if we can find a controller K which achieves the desired performance for $(\mathcal{M}_0, \mathcal{D})$, yet does not achieve it for the process P, we invalidate $(\mathcal{M}_0, \mathcal{D})$. The following theorem establishes a connection between this viewpoint and the standard model validation problem.

Theorem 3.2 *Given* $(\mathcal{M}_0, \mathcal{D})$ *and* $\mathcal{W}$, *let* K *be a controller which achieves*

$$\|\mathcal{F}_l((\mathcal{M}_0, \mathcal{D}), K)w_i\| \leq \gamma\|w_i\| \ \ \forall i \in [1, M_w]$$

Furthermore, for each $i \in [1, M_w]$, *let* $u_i^* = G(w_i, P, K)$ *and* $y_i^* = Pu_i^*$.

If $\exists \ \{(h_i, d_i)\}_1^{M_w} \subset (\mathcal{M}_0, \mathcal{D})$ *s.t.* $y_i^* = (h_i, d_i)u_i^* \ \forall i \in [1, M_w]$

then $\|\mathcal{F}_l(P, K)w_i\| \leq \gamma\|w_i\| \ \forall i \in [1, M_w]$.

See Appendix for the proof.

Remark 3.2: This theorem says that if we cannot invalidate the set $(\mathcal{M}_0, \mathcal{D})$ w.r.t. (u_i^*, y_i^*) for any $i \in [1, M_w]$, then we will not be able to invalidate it based on observing the performance w.r.t. $\mathcal{W}$. Notice that the reverse implication is not necessarily true because K is not designed to achieve performance **only** for $(\mathcal{M}_0, \mathcal{D})$. Even if $(\mathcal{M}_0, \mathcal{D})$ is not consistent with (u_i^*, y_i^*), K may inadvertantly achieve the desired performance for some $(h, d) \notin (\mathcal{M}_0, \mathcal{D})$ which is consistent with (u_i^*, y_i^*). Also note that the above theorem can be easily extended to performance objectives such as $\|W_p\mathcal{F}_l(P, K)w_i\| \leq \gamma\|w_i\|$, where W_p is some weighting function. In the procedure below, we actually use the converse of this theorem which we now state as a corollary.

Corollary 3.3 *Given the assumptions of theorem 3.2,* **if** $\exists j \in [1, M_w]$ *s.t.* $\|\mathcal{F}_l(P, K)w_j\| > \gamma\|w_j\|$, **then** *there does not exist any* $(h, d) \in (\mathcal{M}_0, \mathcal{D})$ *s.t.* $y_j^* = (h, d)u_j^*$.

Remark 3.3: This says that if we invalidate the set $(\mathcal{M}_0, \mathcal{D})$ based on performance, then we would have also invalidated it by using the resulting (u^*, y^*). This suggests the following procedure for model validation.

1. Synthesize a controller, K, which achieves $\|\mathcal{F}_l((\mathcal{M}_0, \mathcal{D}), K)w_i\| \leq \gamma\|w_i\| \quad \forall i \in [1, M_w]$.
2. Use K to close the loop around the process P.
3. Monitor the performance variables (w, z) and compare $\|\mathcal{F}_l(P, K)w_i\|$ to $\|w_i\|$.
4. If, for some j, $\|\mathcal{F}_l(P, K)w_j\| > \gamma\|w_j\|$, we say that $(\mathcal{M}_0, \mathcal{D})$ does not adequately describe P w.r.t. the objectives. Furthermore, Corollary 3.3 tells us that $(\mathcal{M}_0, \mathcal{D})$ is also inconsistent with (u_j^*, y_j^*).
5. If we cannot invalidate $(\mathcal{M}_0, \mathcal{D})$, then we have a controller that achieves the performance objective w.r.t. $\mathcal{W}$.

Examining the steps in this procedure shows that the computational burden is embedded in the robust control design, while the decision step (performance check) is quite simple. This suggests that the computational burden of the consistency check in previous work is transferred to robust control design in our framework, and Corollary 3.3 shows that there is some relation between the two. It is important to reiterate the significance of the unidirectional implication in Theorem 3.2 and Corollary 3.3. This means that even though the observed performance is satisfied, (u^*, y^*) still may not be consistent with $(\mathcal{M}_0, \mathcal{D})$. In this way, the performance test is only sufficient for invalidation of $(\mathcal{M}_0, \mathcal{D})$ w.r.t. (u^*, y^*). This is not a deficiency in the procedure, but merely reflects the fact that model invalidation in the sense of Definition 3.1 is only sufficient for model invalidation in the traditional sense. In other words, as long as the controller we have designed for $(\mathcal{M}_0, \mathcal{D})$ is delivering the desired performance for the process P, we have no reason to invalidate the set $(\mathcal{M}_0, \mathcal{D})$.

Finally, we point out that performing model invalidation in this way is further motivated by parts of the iterative scheme for identification/control proposed in [2]. Here, we are given a larger model set $(\mathcal{M}, \mathcal{D})$ which we strongly believe describes the process P. We next partition $(\mathcal{M}, \mathcal{D})$ into a collection of model sets $\{(\mathcal{M}_i, \mathcal{D}_i)\}$ which cover $(\mathcal{M}, \mathcal{D})$. We then perform this model invalidation procedure on each $\{(\mathcal{M}_i, \mathcal{D}_i)\}$ until we either invalidate the entire model set $(\mathcal{M}, \mathcal{D})$ and conclude that we must change our biases, or we find a controller which achieves the desired performance for P.

4. Conclusion

In this paper, we have presented a way of thinking about model validation that is intimately tied to robust control theory. The main idea is that perhaps models should be (in)validated depending on what they will be used for. Assuming that the model set will be used for control design, we have proposed a method of (in)validating sets of models using robust control theory. The main contribution was in making this robust control based invalidation idea rigorous and showing how it relates to model validation in previous work. In the process, we have established an important connection between model validation and robust control.

The new method has several desirable features. Given the performance objectives, the control design produces inputs for testing the consistency of the given model set. Furthermore, if the model set is not invalidated, the procedure produces a controller which achieves the desired level of performance for the unknown process (w.r.t. a certain set of exogenous inputs).

References

[1] M.A. Dahleh, M. Khammash, "Controller Design for Plants with Structured Uncertainty", *Automatica*, Vol. 21, Jan., 1993.

[2] M.A. Dahleh, J. Doyle, "From Data to Control", *Proc. Workshop on Modeling of Uncertainty in Control Systems*, Springer–Verlag, 1992.

[3] J. Maciejowski. *Multivariable Feedback Design*. Addison–Wesley 1989.

[4] K. Poolla, P. Khargonekar, A. Tikku, J. Krause, K. Nagpal, "A Time–Domain Approach to Model Validation", *Proc. Amer. Contr. Conf.*, pp. 313–317, 1992.

[5] R. Smith, J. Doyle, "Model Invalidation–A Connection Between Robust Control and Identification", *Proc. Amer. Contr. Conf.*, pp. 1435–1440, 1989.

[6] R. Smith, "Model Validation and Parameter Identification for Systems in H_∞ and l_1 ", *Proc. Amer. Contr. Conf.*, pp. 2852–2856, 1992.

[7] R. Smith, J. Doyle, "Model Validation: A connection Between Robust Control and Identification", *IEEE Trans. Auto. Contr.*, vol. AC-37, pp. 942–952, July 1992.

5. Appendix

Proof of Theorem 3.2

Given $\mathcal{W}$, take any $i \in [1, M_w]$. During the experiment, the closed loop system generates unique u_i^*, y_i^* and $z_i^* = \mathcal{F}_l(P, K)w_i$. By assumption, $y_i^* = (h_i, d_i)u_i^*$ (i.e., (h_i, d_i) can interpolate u_i^* to y_i^*). Because of uniqueness of solutions, if we consider (h_i, d_i) in the loop instead of P, the resulting z must be the same as z_i^* from the experiment. This means exactly that

$$\mathcal{F}_l(P, K)w_i = \mathcal{F}_l((h_i, d_i), K)w_i$$

and since K was designed to achieve

$$\|\mathcal{F}_l((\mathcal{M}_0, \mathcal{D}), K)w_i\| \leq \gamma \|w_i\|$$

it certainly achieves

$$\|\mathcal{F}_l((h_i, d_i), K)w_i\| \leq \gamma \|w_i\| .$$

In view of equation 5, it is clear that this implies $\|\mathcal{F}_l(P, K)w_i\| \leq \gamma \|w_i\|$. This completes the proof.