Proceedings of the 8th IFAC Symposium on Robust Control
Design, Bratislava, Slovak Republic, July 8-11, 2015

Fixed Low-Order Controller Design and H_∞ Optimization for Large-Scale Dynamical Systems ⋆

Tim Mitchell * **Michael L. Overton** †

* *Courant Institute of Mathematical Sciences, New York University, New York, NY 10012 (e-mail: tim.mitchell@cims.nyu.edu)*
† *Courant Institute of Mathematical Sciences, New York University, New York, NY 10012 (e-mail: overton@cims.nyu.edu)*

Abstract: Large-scale linear time-invariant dynamical systems with inputs and outputs present major challenges for controller design. Model-order reduction has become popular in recent years, but controllers designed for reduced-order models may result in unstable closed-loop plants when applied to the larger-scale system. We investigate the practicality of fixed low-order controller design applied directly to large-scale continuous-time sparse systems. We assume that it is practical to compute the eigenvalues with largest real part of such systems using MATLAB's `eigs`, which requires only matrix-vector products, but that it is not possible to compute the H_∞ norm using MATLAB's `getPeakGain` or SLICOT's `slinorm`, which use the Boyd-Balakrishnan-Bruinsma-Steinbuch algorithm, requiring both Hamiltonian eigenvalue decompositions and singular value decompositions. Instead, we employ a recently developed efficient algorithm called Hybrid-Expansion-Contraction (HEC), which while not guaranteed to correctly compute the H_∞ norm, finds, under certain assumptions, at least a local maximizer of the associated transfer function. Our controller design code uses nonsmooth optimization techniques first to attempt to stabilize the closed-loop system and then to minimize its H_∞ norm proxy as computed by HEC. It is implemented in a new experimental MATLAB code HIFOOS, based on the public-domain HIFOO toolbox first presented in ROCOND 2006, and will be made available for public use after further investigation and development.

Keywords: Robust stabilization, low-order controller design, H_∞ control, HIFOO

1. INTRODUCTION

Nine years ago, at the ROCOND 2006 conference, a new toolbox, HIFOO, for H-infinity fixed-order optimization (Burke et al., 2006), was presented. The goal was to exploit advances in methods for nonsmooth, nonconvex optimization and make them available in a convenient form for engineers to use to design low-order controllers for linear time-invariant (LTI) dynamical systems with input and output. Open-loop systems could be specified using several different input formats, including both MATLAB's `ss` structure format and examples from a library of systems collected in Leibfritz's *COMPl$_e$ib* (Leibfritz, 2004). HIFOO used two algorithmic phases: a stabilization phase to design a controller to stabilize the resulting closed-loop system, and then an optimization phase to minimize, as far as possible, the H_∞ norm of the closed-loop system. Two optimization techniques were used: the Gradient Sampling method (Burke et al., 2005) and the BFGS method, which turns out to be surprisingly effective for nonsmooth optimization, as later argued by (Lewis and Overton, 2013). The optimization problems addressed by HIFOO, being nonconvex and nonsmooth, are well known to be difficult problems to solve, and may be hard in a theoretical sense such as discussed by (Blondel and Tsitsiklis, 1997, 2000). Consequently, HIFOO offers no guarantees; it simply returns the best controller it can find and it is up to the user to investigate its usefulness.

In later versions developed over the past nine years, HIFOO was extended to have additional functionality of various sorts, including stabilization of multiple plants and adding H_2 performance options: see the HIFOO web page (Hifoo) for more details. Among other things, HIFOO has been used for the design of an aircraft controller for improved gust alleviation and passenger comfort (Wildschek et al., 2009), design of a proton exchange membrane fuel cell system (Wang and Chen, 2009), design of power systems controllers (Dotta et al., 2009), design of winding systems for elastic web materials (Knittel et al., 2007), flight control via static-output-feedback (Yaesh and Shaked, 2012), robust observer-based fault detection and isolation (Wahrburg et al., 2012), influence of tire damping on control of quarter-car suspensions (Akcay and Turkay, 2011), flexible aircraft lateral flight dynamic control (Hanis and Hromcik, 2011), optimal control of aircraft with a blended wing body (Hanis et al., 2011), vibration control of a fluid/plate system (Robu et al., 2010), control design of a nose landing gear steering system (Pouly et al., 2010), and bilateral teleoperation for minimally invasive surgery (Delwiche, 2009). In (Arzelier et al., 2009), the

⋆ The work of T. Mitchell and M.L. Overton was supported by National Science Foundation grant DMS-1317205.

authors wrote in their abstract that "the HIFOO package is considered to be the most effective tool for optimal H_∞ static-output-feedback control". A discrete-time version of HIFOO has also been developed (Popov et al., 2010). HIFOO was and apparently remains the only open-source software for solving such problems, although a routine `hinfstruct`, intended to solve such control design problems and more, has been part of MATLAB's Control Systems Toolbox since 2010. It is based on work of Apkarian and Noll (2006a,b).

However, none of the HIFOO releases up to now have been able to handle large sparse systems, because the codes all rely on the Boyd-Balakrishnan-Bruinsma-Steinbuch (BBBS) algorithm (Boyd and Balakrishnan, 1990; Bruinsma and Steinbuch, 1990) to compute the H_∞ norm, a computation that is much too expensive when the systems to which it is applied are large and sparse. Model-order reduction (Antoulas et al., 2001; Benner et al., 2006) has become popular in recent years, but controllers designed for reduced-order models may result in unstable closed-loop plants when applied to the larger-scale system. In this paper, we investigate the practicality of applying fixed low-order controller design directly to large-scale sparse systems, developing a new experimental code based on the original HIFOO 1.0 toolbox. Instead of the BBBS algorithm, we use a recently developed efficient algorithm called Hybrid-Expansion-Contraction (HEC) (Mitchell and Overton, 2015), which while not guaranteed to correctly compute the H_∞ norm, finds, under certain assumptions, at least a local maximizer of the associated transfer function. In this paper we report the results of our first experiments using HIFOOS. We plan to make the new code available for public use after further investigation and development.

2. PROBLEM STATEMENT

Consider the following open-loop LTI dynamical system (Zhou et al., 1995)

$$\left[\begin{array}{c} \dot{x} \\ \hline z \\ y \end{array}\right] = \left[\begin{array}{c|cc} A_1 & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{array}\right] \left[\begin{array}{c} x \\ \hline w \\ u \end{array}\right] \tag{1}$$

where $x \in \mathbb{R}^{n_x}$ contains the states, $u \in \mathbb{R}^{n_u}$ is the physical (control) input, $w \in \mathbb{R}^{n_w}$ is the performance input, $y \in \mathbb{R}^{n_y}$ is the physical (measured) output, and $z \in \mathbb{R}^{n_z}$ is the performance output. The goal is to design a controller K defining

$$\left[\begin{array}{c} \dot{x}_K \\ u \end{array}\right] = K \left[\begin{array}{c} x_K \\ y \end{array}\right] = \left[\begin{array}{cc} A_K & B_K \\ C_K & D_K \end{array}\right] \left[\begin{array}{c} x_K \\ y \end{array}\right]$$

where $x_K \in \mathbb{R}^{n_K}$ is the controller state and n_K is the order of the controller, resulting in the closed-loop system:

$$\left[\begin{array}{c} \dot{x} \\ \dot{x}_K \\ \hline z \end{array}\right] = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \left[\begin{array}{c} x \\ x_K \\ \hline w \end{array}\right] \tag{2}$$

with, assuming $D_{22} = 0$ purely for simplicity and conciseness of notation:

$$\begin{aligned} A &= \left[\begin{array}{cc} A_1 + B_2 D_K C_2 & B_2 C_K \\ B_K C_2 & A_K \end{array}\right] \\ B &= \left[\begin{array}{c} B_1 + B_2 D_K D_{21} \\ B_K D_{21} \end{array}\right] \\ C &= \left[\begin{array}{cc} C_1 + D_{12} D_K C_2 & D_{12} C_K \end{array}\right] \\ D &= \left[\begin{array}{c} D_{11} + D_{12} D_K D_{21} \end{array}\right]. \end{aligned}$$

The closed-loop transfer matrix function

$$T(s) = C(sI - A)^{-1}B + D \tag{3}$$

maps the performance input w to the performance output z. Key requirements for the controller K are first and foremost that the closed-loop system is stable, that is all the eigenvalues of A are strictly in the left half of the complex plane, and also, in H_∞ control, that the H_∞ norm of $T(s)$ be minimized as much as possible. If $n_K = n_x$, the controller is called a full-order controller and methods to compute it are well known. However, it is often desirable to design a low-order controller with $n_K \ll n_x$, and this is the case HIFOO addresses via optimization techniques.

3. HIFOOS FOR LARGE-SCALE SYSTEMS

In order to investigate the viability of using the HEC H_∞ norm approximation algorithm as an efficient means towards designing fixed, low-order controllers for large-dimensional LTI systems, without resorting to model-order reduction techniques, we forked the v1.0 release of HIFOO to create a new large and sparse-enabled version, which we call HIFOOS for HIFOO–sparse. Several important modifications were made.

First, HIFOO's stabilization phase requires repeatedly computing the spectral abscissa (the maximum of the real parts of the eigenvalues) of the matrix A as the controller K is updated in order to stabilize the closed-loop system defined in (2). For efficient handling of large-scale systems, this necessitates that the dense eigensolver `eig` be replaced with a sparse solver capable of finding a rightmost eigenvalue of a matrix. For this, we used MATLAB's `eigs`, called both on the matrix and its transpose, in order to produce the computed rightmost eigenvalue's corresponding right and left eigenvectors necessary for calculating the gradient of the spectral abscissa of A with respect to the controller variables (Burke et al., 2002).

Second, assuming the stabilization phase is successful, the H_∞ optimization phase begins: the H_∞ norm of (3) is repeatedly calculated as the controller is changed in order to optimize it. In lieu of using MATLAB's `getPeakGain` or SLICOT's `slinorm` (updated versions of the original codes `ss/norm` and `linorm` used by HIFOO 1.0), both of which are implementations of the globally convergent but expensive BBBS algorithm, we instead employed `getStabRadBound`, a MATLAB implementation of the recently developed HEC algorithm (Mitchell and Overton, 2015), for approximating the H_∞ norm. As `getStabRadBound`'s computation can be optionally warm-started, an option was added to allow HIFOOS to always warm-start `getStabRadBound` using the H_∞ norm approximation, and its related singular vectors, found for the previous controller encountered during the BFGS optimization. When warm-starting is disabled, HEC always calculates an initial guess by computing a selected rightmost eigenvalue of A; for more details see (Guglielmi et al., 2013, pages 728-9).

Finally, the method used to compute the gradients of both the spectral abscissa and the H_∞ norm in all HIFOO releases is not efficiently scalable to large-dimensional systems. However, modifying the calculation of the gradient of the spectral abscissa so that it is amenable to large and sparse systems is straightforward. An alternate method for efficiently computing the gradient of the H_∞ norm is also made possible by the result of (Guglielmi et al., 2013, Theorem 2.9), which relates the singular vectors associated with the norm of (3) to the eigenvectors of the perturbed system matrix, all of which can be optionally returned by `getStabRadBound` for no additional cost once it has converged to an approximation. Thus, once `eigs` or `getStabRadBound` has returned a result for the spectral abscissa or H_∞ norm respectively, the remaining computational cost for preparing the gradients has a FLOP count of only

$$\mathcal{O}\left(n_K^2 + (n_x + n_K)(n_u + n_y) + n_u n_y + n_u n_z + n_w n_y\right).$$

By comparison, the standard HIFOO method for computing the gradients has a lower bound FLOP count of $\Omega\left((n_x + n_K)^2\right)$, which is a cost incurred on top of the $\mathcal{O}((n_x + n_K)^3)$ FLOPs required to either compute the spectral abscissa or form the transfer function matrix (a necessary step in HIFOO's H_∞ norm gradient computation), assuming dense matrix methods are used.

There are some potential issues, however, that may adversely affect the practical utility of the current version of HIFOOS and these warrant discussion. As the HEC algorithm is not guaranteed to converge to the true value of the H_∞ norm, there is a danger that the HEC approximations, and the gradients derived from them, may cause the H_∞ optimization phase to fail. On one hand, the approximations returned by `getStabRadBound` may not even correspond to a continuous function, which could lead to BFGS incurring breakdown before reaching a local minimum, as the line search is sensitive to this failure mode. On the other hand, another possibility is that `getStabRadBound` may only return locally maximal values of the norm of the transfer function evaluated along the imaginary axis as its computed approximations to the H_∞ norm. In this scenario, BFGS may be reducing a locally maximal peak of the transfer function as it changes in response to the controller variables being "optimized", but the updated controllers may not actually provide any reduction in the true value of the H_∞ norm of the closed-loop systems. In this case, a plausible outcome is that the true value of the H_∞ norm could in fact be increasing while misleadingly, the reported values computed by `getStabRadBound` as the H_∞ optimization progresses are decreasing.

Moreover, to produce approximations to the H_∞ norm, the HEC algorithm is entirely dependent upon the quality of the sparse eigensolver employed to find rightmost eigenvalues of matrices, which is inherently difficult. Similarly, the initial stabilization phase is also entirely reliant upon the sparse eigensolver returning a rightmost eigenvalue. A bonus for discrete-time systems (not discussed further here) is that it is only necessary to find eigenvalues of largest modulus, which is generally readily done with existing eigensolvers.

While there are potentially effective approaches to increasing the quality of the approximations returned from HEC (e.g. see (Mitchell, 2014, Section 4.4)), in this paper, we first seek to understand how prevalent these issues are in practice and to observe the consequences. It is also worth noting that the common practice of using a reduced-order model as a surrogate while designing and optimizing a controller intended to be used in the large-scale closed-loop system is not without its pitfalls either, as there is no guarantee that the resulting system will even be stable, let alone optimized with respect to the H_∞ norm.

Table 1. Large-scale full-order model (FOM) 2D heat flow problems for which *COMPl$_e$ib* also includes reduced-order models (ROM).

Problem	n_x (FOM)	n_x (ROM)	n_w	n_u	n_z	n_y
`HF2D1`	3796	316	3798	2	3796	3
`HF2D2`	3796	316	3798	2	3796	3
`HF2D5`	4489	289	4491	2	4489	4
`HF2D6`	2025	289	2027	2	2025	4
`HF2D9`	3481	484	3483	2	3481	2
`HF2D_CD1`	3600	256	3602	2	3600	2
`HF2D_CD2`	3600	256	3602	2	3600	2
`HF2D_CD3`	4096	324	4098	2	4096	2
`HF2D_IS1`	4489	361	4491	2	4489	4
`HF2D_IS2`	4489	361	4491	2	4489	4
`HF2D_IS3`	3600	256	3602	2	3600	2
`HF2D_IS4`	3600	256	3602	2	3600	2

4. EXPERIMENTAL SETUP

To evaluate HIFOOS, we used the set of large-scale 2D heat flow problems from the 2006 v1.1 release of *COMPl$_e$ib* (Leibfritz, 2004). Of these, we chose the twelve `HF2Dx` full-order model (FOM) problems for which corresponding medium-scale reduced-order models (ROM) are also available in the library (see Table 1) so that we could compare the quality of the controllers produced using HIFOOS on the large-scale problems with controllers of the same order produced using a slightly modified HIFOO 1.0 on the reduced-order equivalents. The ROM problems have names of the form `NAME_Mxxx` where NAME is the name of the associated FOM problem and xxx indicates the reduced dimension for A_1, shown in the third column of Table 1. As computing the H_∞ norm exactly is already expensive for these medium-scale problems, particularly given how many times it will be evaluated in the course of optimizing a controller, we chose to use only the smaller of the two reduced-order models for each problem. We elected to design order 10 controllers for the entire test set and thus the number of controller variables in K varied from 144 to 168 depending on the problem; the number of controller variables is $(n_K + n_u)(n_K + n_y)$.

Given the large computational costs for running these experiments, for both HIFOO on the reduced-order models and HIFOOS on the original large-dimensional problems, we made the following parameter selections and modifications. We disabled HIFOO's expensive gradient sampling process that is usually done after both the initial BFGS stabilization phase and the BFGS H_∞ optimization phase; we made this modification for both HIFOO and HIFOOS. In a similar vein, we reduced the max BFGS iteration count to 100 for both codes, meaning that their stabilization and H_∞ optimization phases were each allowed up to 100 iterations. Though it is not provided as a user option in HIFOO,

we also enabled setting the `nvec` BFGS option explicitly to 0 (indicating full BFGS) since otherwise the routine defaults to a less effective 10 gradient history limited-memory BFGS scheme when the number of controller variables is greater than 100. A unique seed was randomly generated and stored per problem so that each method, HIFOO, HIFOOS-W (warm-started HEC), and HIFOOS-C (cold-started HEC), could be ensured to start stabilization from the same randomly generated initial controller.

For the two HIFOOS variants, we ran `getStabRadBound` with the default termination conditions but additionally enabled the following options: a 0.01 relative termination tolerance for the expansion phase, vector extrapolation using the five past iterates, and eigenvector recycling enabled, which, in combination, were shown to greatly accelerate the method over the default HEC algorithm in extensive tests (Mitchell and Overton, 2015). In HIFOOS-W, every call to `getStabRadBound` was warm-started via the result computed by the immediately preceding `getStabRadBound` call, except the first which was cold-started. In order to increase the likelihood that `eigs` would return rightmost eigenvalues of matrices, the default options were changed so that 8 eigenvalues were requested and the Krylov subspace dimension was increased to 60 while a maximum iteration limit was set at 200; these settings were applied both to `eigs` calls used by `getStabRadBound` during the H_∞ optimization phase and to those made for spectral abscissa computations during the preceding stabilization phase.

We ran both HIFOOS methods on the twelve large-scale problems and HIFOO on the associated reduced-order models. For each problem, all methods were run three times, using the same three randomly generated initial controllers for that problem. For HIFOO on the reduced-order models, we identified the best of the three controllers per problem by choosing the one with the lowest value of the H_∞ norm computed by `getPeakGain` for the corresponding small-scale closed-loop systems. For the full-order large-scale problems, where it isn't feasible to use `getPeakGain`, we instead calculated an approximation using `getStabRadBound`, without a warm start, for HIFOO's best controllers designed for the reduced-order models. Similarly, for both HIFOOS variants, we again used `getStabRadBound` without a warm start to compute approximations for each method's three controllers and of these, used the lowest approximation reported to select the corresponding best controller. However, as a result, this meant we had two H_∞ norm approximations per initial controller for HIFOOS-W, since enabling warm-starting may cause `getStabRadBound` to converge to a different approximation. Thus, for each of these pairs of approximations, we chose the higher of the two values since it must be a better approximation, and then used these sets of better approximations to choose the best controller of the three.

For all methods, we recorded the total CPU-time incurred for running HIFOO or HIFOOS on three different starting points but did not time the additional cost to compute or approximate the H_∞ norm after the controllers had been returned for the purposes of this evaluation. For comparison purposes, we attempted to run `hinfstruct` on the large-scale FOM problems but the routine threw an "out of memory" exception for even the smallest of these problems (`HF2D6`). All experiments were done using MATLAB R2014a running on a single-user desktop with Ubuntu 14.04 (64-bit) and an Intel i7-3770K CPU with 8 GB of RAM.

Before discussing the results, it is important to keep the following caveats in mind. First, the HEC algorithm is guaranteed only to find a lower bound for the H_∞ norm. On a test set of challenging small problems where the results could be verified, `getStabRadBound` was shown to find the H_∞ norm to four digits of accuracy on 28 out of 33 problems and to eight digits of accuracy on 21 of these same 33 problems (Mitchell and Overton, 2015). However, as the HEC method is new, there is as yet no other data indicating whether this level of performance is usually to be expected. Furthermore, without spending an inordinate amount of computational resources, it is very difficult to empirically assess the performance of HEC on large-dimensional problems. It is thus imperative to be aware that the reported values for H_∞ norm for the large-scale problems evaluated may only be lower bounds.

While we have tried to control the experiments by initializing each method with same problem-specific starting controller, minor differences in rounding errors determined by how exactly the spectral abscissa, the H_∞ norm and their gradients are computed can significantly alter the trajectory of the optimization routines, even if the different methods are essentially returning the same values. To confirm and illustrate this point, we created a dense version of HIFOOS, that is, a version that still uses all the regular dense procedures of HIFOO that are guaranteed to compute the true values of the spectral abscissa and H_∞ norm. However, we retained the new large and sparse-efficient formulation for calculating the gradients that HIFOOS uses, as discussed in §3. The results differed significantly although in the absence of rounding errors they should theoretically be identical.

Finally, it is worth making explicit mention that in our timing report, we do not account for the cost of obtaining reduced-order models for use with HIFOO 1.0, since they were provided by *COMPl$_e$ib*. In practice, however, this cost may be significant and should not be ignored in considering the expense of the standard practice of designing controllers using reduced-order models. Our method that we propose and investigate here has the distinct benefit that it can efficiently begin stabilizing and optimizing large-scale systems without ever having to apply model-order reduction techniques.

5. RESULTS

Table 2 shows our main results. Column 1 gives the FOM problem name. Column 2 reports the H_∞ norm for the closed-loop system using the controller designed by HIFOO for the associated ROM problem, for which the norm was computed exactly. Column 3 shows a lower bound for the H_∞ norm for the closed loop system for the associated FOM, using the same controller as in Column 2. Columns 4 and 5 show lower bounds for the H_∞ norm for the closed-loop system for the FOM, using the controller designed by the HIFOOS codes applied directly to the FOM. For Columns 3-5 the lower bound is calculated by

Table 2. Column 2 gives the H_∞ performance for the closed loop systems using the controllers designed by HIFOO for the ROMs; Columns 4 and 5 give lower bounds on the H_∞ performance for the closed-loop systems designed by the two variants of HIFOOS for the associated FOMs. Column 3 gives lower bounds on the H_∞ performance of the closed-loop system for each FOM using the controller designed for the corresponding ROM.

	H_∞ (ROM)	H_∞^{lb} (FOM)		
Problem	HIFOO	HIFOO	HIFOOS-W	HIFOOS-C
HF2D1	6.73×10^3	3.43×10^3	5.52×10^3	5.08×10^3
HF2D2	5.64×10^3	2.68×10^3	3.10×10^3	2.89×10^3
HF2D5	1.94×10^4	2.83×10^4	2.38×10^6	6.15×10^5
HF2D6	7.86×10^3	1.08×10^4	2.63×10^5	2.37×10^5
HF2D9	7.42×10^1	2.95×10^1	2.95×10^1	2.95×10^1
HF2D_CD1	4.62×10^0	∞	6.23×10^2	2.55×10^2
HF2D_CD2	7.01×10^0	∞	6.18×10^1	1.64×10^1
HF2D_CD3	4.30×10^0	∞	9.84×10^2	3.54×10^2
HF2D_IS1	7.57×10^4	3.32×10^5	4.21×10^6	3.54×10^6
HF2D_IS2	1.17×10^4	4.68×10^3	6.05×10^5	5.97×10^5
HF2D_IS3	8.49×10^0	∞	1.31×10^3	4.12×10^2
HF2D_IS4	6.92×10^0	∞	3.88×10^3	8.36×10^3

getStabRadBound. The value ∞ denotes that stabilization was not achieved, measured using eigs. Each result shown is the best result obtained over the same three randomly generated starting points used by all of HIFOO, HIFOOS-W and HIFOOS-C.

Assuming that getStabRadBound is providing good approximations, we see that for HF2D9 the HIFOOS variants have produced controllers matching the performance of the controller produced by HIFOO using the reduced-order model when plugged into the large-scale system. On HF2D2, HIFOOS-C's controller has nearly matched the performance of HIFOO's reduced-order model derived controller while on HF2D1, the full-order model derived controllers produced by the two HIFOOS methods are also close in performance to the HIFOO controller. On the other hand, on HF2D5, HF2D6, HF2D_IS1, and HF2D_IS2, the controllers returned by HIFOO all provide results that are at least one to two orders of magnitude better than those for the best controllers found by HIFOOS. Perhaps most significantly, for problems HF2D_CD1, HF2D_CD2, HF2D_CD3, HF2D_IS3, and HF2D_IS4, we see that the best controllers optimized using HIFOO on the reduced-order models fail to stabilize any of these five full-order problems. Furthermore, as eigs is generally a very good quality sparse eigensolver, it is reasonable to assume that the corresponding HIFOOS controllers for these problems actually do stabilize the large-scale systems, even if their H_∞ norm values computed by getStabRadBound may be underreporting the true values.

In Table 3, we report the running times of each method for each problem, summing up the time for all three starting points. Interestingly, we see that any potential benefit of warm-starting HEC in HIFOOS-W seems to be problem-specific and that it actually increases the running times on eight of the twelve problems, making it, surprisingly, slightly slower overall across the entire data set compared to HIFOOS-C. Note that the total running time required for the HIFOOS codes to design

Table 3. Total elapsed CPU time in minutes for each method to design a controller using three starting points. HIFOO designed controllers using the reduced-order models (ROM) while the two HIFOOS variants worked directly on the full-order models (FOM).

	Time (ROM)	Time (FOM)	
Problem	HIFOO	HIFOOS-W	HIFOOS-C
HF2D1	419.37	631.70	875.76
HF2D2	714.72	967.42	994.80
HF2D5	314.47	141.75	134.42
HF2D6	316.17	41.23	12.98
HF2D9	68.75	21.23	37.52
HF2D_CD1	170.90	151.47	88.75
HF2D_CD2	175.78	95.80	46.19
HF2D_CD3	418.28	255.25	124.47
HF2D_IS1	625.17	170.09	66.62
HF2D_IS2	597.66	856.67	617.38
HF2D_IS3	164.00	249.72	275.65
HF2D_IS4	193.03	153.63	96.11
TOTAL	4178.30	3735.96	3370.65

controllers for the entire full-order model dataset is just 80 to 90% of the running time needed by HIFOO to compute the controllers for the reduced-order models, despite the fact that HIFOOS is working with problems where the dimensions of the A_1 matrices range from about 7 to 15.5 times larger. However, this is partly explained by our observation that the BFGS routine performing H_∞ optimization in HIFOOS frequently terminated quite early, particularly compared to the behavior displayed by HIFOO. In all cases, this early termination was due to the BFGS line search failing and often occurred after only taking a handful of BFGS iterations. In contrast, HIFOO hit its 100 max iteration count on ten of the twelve problems while it reached normal termination on one example and incurred a line search failure on only one problem. This marked difference in behavior seems to occur because the approximate and sometimes inconsistent values that may be returned by HEC often prevents the line search from succeeding. Consequently, a more stochastic-friendly optimization method might be more effective at handling the approximate nature of getStabRadBound's results.

6. CONCLUSIONS

We have presented a new approach to designing fixed low-order controllers for stabilizing and optimizing the H_∞ norm of large-scale LTI systems. To our knowledge, this is the first attempt to directly and efficiently design such controllers without resorting to model-order reductions techniques, which was made possible by the newly developed efficiently scalable HEC algorithm for approximating the H_∞ norm. We have demonstrated that HIFOOS can be a viable approach, sometimes producing better controllers than are obtained from reducing the dimensionality of the systems and designing controllers for the smaller-scale systems.

ACKNOWLEDGEMENTS

We thank Denis Arzelier and Didier Henrion for many helpful comments.

REFERENCES

Akcay, H. and Turkay, S. (2011). Influence of tire damping on actively controlled quarter-car suspensions. *Journal of Vibration and Acoustics*, 133(054501).

Antoulas, A.C., Sorensen, D.C., and Gugercin, S. (2001). A survey of model reduction methods for large-scale systems. In *Structured matrices in mathematics, computer science, and engineering, I (Boulder, CO, 1999)*, volume 280 of *Contemp. Math.*, 193–219. Amer. Math. Soc., Providence, RI.

Apkarian, P. and Noll, D. (2006a). Controller design via nonsmooth multidirectional search. *SIAM J. Control Optim.*, 44(6), 1923–1949 (electronic).

Apkarian, P. and Noll, D. (2006b). Nonsmooth H_∞ synthesis. *IEEE Trans. Automat. Control*, 51(1), 71–86.

Arzelier, D., Gryazina, E., Peaucelle, D., and Polyak, B. (2009). Mixed LMI/randomized methods for static output feedback control design. Technical Report 09535, LAAS-CNRS, Toulouse.

Benner, P., Freund, R.W., Sorensen, D.C., and Varga, A. (2006). Preface. Special issue on: "Order reduction of large-scale systems". *Linear Algebra Appl.*, 415(2-3), 231–234.

Blondel, V. and Tsitsiklis, J. (1997). NP-hardness of some linear control design problems. *SIAM Journal on Control and Optimization*, 35, 2118–2127.

Blondel, V. and Tsitsiklis, J. (2000). A survey of computational complexity results in systems and control. *Automatica*, 36(9), 1249–1274.

Boyd, S. and Balakrishnan, V. (1990). A regularity result for the singular values of a transfer matrix and a quadratically convergent algorithm for computing its $\mathbf{L}_\infty$-norm. *Systems Control Lett.*, 15(1), 1–7.

Bruinsma, N. and Steinbuch, M. (1990). A fast algorithm to compute the H^∞-norm of a transfer function matrix. *Systems Control Letters*, 14, 287–293.

Burke, J., Henrion, D., Lewis, A., and Overton, M. (2006). HIFOO - a MATLAB package for fixed-order controller design and H_∞ optimization. In *Fifth IFAC Symposium on Robust Control Design, Toulouse*.

Burke, J., Lewis, A., and Overton, M. (2002). Two numerical methods for optimizing matrix stability. *Lin. Alg. Appl.*, 351-352, 117–145.

Burke, J., Lewis, A., and Overton, M. (2005). A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. *SIAM Journal on Optimization*, 15, 751–779.

Delwiche, T. (2009). *Contribution to the Design of Control Laws for Bilateral Teleoperation with a View to Applications in Minimally Invasive Surgery*. Ph.D. thesis, Free University of Brussels.

Dotta, D., e Silva, A.S., and Decker, I.C. (2009). Design of power systems controllers by nonsmooth, nonconvex optimization. In *IEEE Power and Energy Society General Meeting, Calgary*.

Guglielmi, N., Gürbüzbalaban, M., and Overton, M. (2013). Fast approximation of the H_∞ norm via optimization over spectral value sets. *SIAM Journal on Matrix Analysis and Applications*, 34(2), 709–737.

Hanis, T. and Hromcik, M. (2011). Lateral flight dynamic controller for flexible BWB aircraft. In *Proceedings of the 18th International Conference on Process Control, Bratislava, Slovenska technicka univerzita*, 333–337.

Hanis, T., Kucera, V., and Hromcik, M. (2011). Low order H-infinity optimal control for ACFA blended wing body. In *4th European Conference for Aerospace Sciences (EUCASS), St. Petersburg*.

Hifoo (2006). HIFOO (H-infinity fixed-order optimization). `http://www.cs.nyu.edu/overton/software/hifoo/`.

Knittel, D., Henrion, D., Millstone, M., and Vedrines, M. (2007). Fixed-order and structure H-infinity control with model based feedforward for elastic web winding systems. In *Proceedings of the IFAC/IFORS/IMACS/IFIP Symposium on Large Scale Systems, Gdansk, Poland*.

Leibfritz, F. (2004). *COMPl$_e$ib*: *CO*nstrained *M*atrix *O*ptimization *P*roblem *lib*rary. Technical report, Universität Trier. URL `http://www.compleib.de`.

Lewis, A. and Overton, M. (2013). Nonsmooth optimization via quasi-Newton methods. *Math. Program.*, 141(1-2, Ser. A), 135–163.

Mitchell, T. (2014). *Robust and efficient methods for approximation and optimization of stability measures*. Ph.D. thesis, New York University.

Mitchell, T. and Overton, M. (2015). Hybrid expansion-contraction: a robust scaleable method for approximating the H_∞ norm. In revision for *IMA Journal of Numerical Analysis*.

Popov, A., Werner, H., and Millstone, M. (2010). Fixed-structure discrete-time H_∞ controller synthesis with HIFOO. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, 3152 –3155.

Pouly, G., Lauffenburger, J.P., and Basset, M. (2010). Reduced order H-infinity control design of a nose landing gear steering system. In *12th IFAC Symposium on Control in Transportation Systems*.

Robu, B., Budinger, V., Baudouin, L., Prieur, C., and Arzelier, D. (2010). Simultaneous H-infinity vibration control of fluid/plate system via reduced-order controller. In *Proceedings of CDC 2010, Atlanta*, 3146–3151.

Wahrburg, A., Khodaverdian, S., and Adamy, J. (2012). Robust observer-based fault detection and isolation in the standard control problem framework. In *7th IFAC Symposium on Robust Control Design, Aalborg, Denmark*.

Wang, F.C. and Chen, H.T. (2009). Design and implementation of fixed-order robust controllers for a proton exchange membrane fuel cell system. *International Journal of Hydrogen Energy*, 34.

Wildschek, A., Maier, R., Hromcik, M., Hanis, T., Schirrer, A., Kozek, M., Westermayer, C., and Hemedi, M. (2009). Hybrid controller for gust load alleviation and ride comfort improvement using direct lift control flaps. In *Third European Conference for Aerospace Sciences (EUCASS)*.

Yaesh, I. and Shaked, U. (2012). H-infinity optimization with pole constraints of static output-feedback controllers – a non-smooth optimization approach. *IEEE Transactions on Control Systems Technology*, 20, 1066–1072.

Zhou, K., Glover, K., and Doyle, J. (1995). *Robust and Optimal Control*. Prentice Hall.