$\mathcal{H}_\infty$ Reduced Order Control for Nanopositioning: Numerical Implementability

Michael R. P. Ragazzon * **Arnfinn A. Eielsen** **
J. Tommy Gravdahl ***

* *Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim, Norway, (e-mail: ragazzon@stud.ntnu.no)*
** *Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim, Norway, (e-mail: eielsen@itk.ntnu.no)*
*** *Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim, Norway, (e-mail: Tommy.Gravdahl@itk.ntnu.no)*

Abstract: A robust $\mathcal{H}_\infty$ multiple-input multiple-output (MIMO) controller is designed and implemented for the lateral stage of an Atomic Force Microscope (AFM). Such a model-based controller can quickly become complex and may be difficult to run in real-time on hardware with limited computational power. Handling this difficulty is the main topic of this paper. The resulting controller can be considered to be stiff, which is characterized by a large spread of eigenvalues. Continuous-time systems running in real-time are usually solved using explicit Runge-Kutta (ERK) methods, which easily becomes unstable for stiff systems. We show how small the time-step for a given controller needs to be for a selection of ERK methods. We also consider model reduction on the controller and how this affects the required step-size and how much it reduces the computational complexity. We have shown that the original 18th order $\mathcal{H}_\infty$ controller could be reduced to a 10th order controller without any significant reduction in performance or stability, which resulted in a 46.7% reduction in execution time partly because the order reduction enabled us to use a simpler solver type.

1. INTRODUCTION

Atomic Force Microscopy (AFM) is a tool capable of studying matter down to the atomic scale. This has made it one of the fundamental tools within the field of nanotechnology. Control of the lateral stage of an AFM has been shown to be challenging because of several reasons, including non-linearities such as (1) hysteresis and (2) creep, (3) lightly-damped vibration dynamics, and (4) large uncertainties. For high performance under such conditions, model-based controllers has been widely employed in the literature such as $\mathcal{H}_\infty$ controllers (Schitter et al., 2001; Salapaka et al., 2002; Salapaka and Sebastian, 2003; Schitter and Stemmer, 2004; Ladjal et al., 2009; Yong et al., 2010). Such controllers tends to become complex in terms of computational complexity. Because of the high uncertainties and the non-linearities it is important to consider robustness in nanopositioning applications. This topic has been studied in Salapaka et al. (2002); Sebastian and Salapaka (2005); Ladjal et al. (2009).

The majority of the literature on nanopositioning seems to perform control design in the continuous time domain as opposed to the discrete-time z-domain. Such controllers can be described using continuous-time state-space models which we will base our discussion around. For a real-time implementation however, the model is solved at discrete time-steps using a fixed step-size. Many popular solver types are based on the family of explicit Runge-Kutta (ERK) methods. These solvers become unstable if the step-size is too large. At the same time, the complexity of the controller running on hardware with limited computational power puts a lower limit on the step-size the hardware needs to perform the necessary calculations. Thus we have both a lower and an upper limit on the step-size determined by various factors. For a controller to be implementable we need the limits to intersect. We will discuss some of these factors in this paper and what we can do about them.

To control a system with a mechanically high bandwidth, we need to have high bandwidth on the control loop as well. Thus, the step-size needs to be sufficiently small. On hardware with limited computational power, we often need to simplify the model such that it becomes feasible. The most widely used method to reduce the computational complexity of a controller is to perform model reduction on an already existing controller. Model reduction aims to keep the input-output behavior as close as possible to the original model while removing states from a state-space representation of the controller. Model reduction has been used extensively, some approaches performs reduction on the plant model (Dong et al., 2007; Lee and Salapaka, 2009). Using a model-based approach, this results in a

controller with less complexity. Another approach is to perform reduction after synthesis using a high-order model plant (Schitter and Stemmer, 2004; Kuiper and Schitter, 2012). The discussion of whether to reduce the plant model and then perform model reduction, or perform reduction on the controller is treated in Anderson and Liu (1989); Anderson (1992), where it is generally concluded that reduction should be performed as a last step in the control design process. Even if the system is already implementable, there is an advantage of reducing the complexity, because we can then run the system on a smaller step-size which reduces the overall noise floor of the system.

In this paper, we will base our discussion around a $\mathcal{H}_\infty$ controller which is designed for the lateral positioning of a commercial AFM. The controller is designed to be robustly stable for a given description of plant uncertainty. We will present equations for how to determine the solver stability of a controller and the required maximum step-size for a variety of ERK methods. Additionally, we will show the effect of model reduction on the complex controller and how this affects the solver stability and computational complexity. This paper is based to a large extent on Ragazzon (2013).

The paper is organized as follows. In Section 2, the experimental set-up is explained and identified model of the plant is found. In section 3, we present the control law design. In Section 4 we present solver stability, specifically for some ERK methods. Section 5 describes the model reduction of the controller. Section 6 gives experimental results of closed-loop characteristics and execution time. The results are discussed in Section 7. Finally, some conclusions are drawn in Section 8.

2. SYSTEM IDENTIFICATION

2.1 Device Description

All experiments are done using a commercial AFM of the type Park Systems XE-70. In this device, the sample is placed on a parallel kinematic 2d flexure scanner for motion in the horizontal xy-plane. Motion along the vertical z-axis is completely decoupled and not regarded for our purposes. The signals from the AFM are routed to an electronic processing and controller box that comes with the microscope. As well as having its own controller circuits, it provides access to analogue measurements from the sensors. It can also receive external signals for manual control of the AFM's actuators which we will use to control the piezoelectric elements.

See a schematic overview of the setup in Fig. 1. The controllers are implemented in a Simulink model, which is compiled and transferred to a dedicated computer, an xPC, which runs a real-time operating system. The xPC performs the number crunching, and is externally connected to a DAC and ADC. These input-output signals are run through anti-aliasing and reconstruction filters, which are constructed as low-pass filters with a bandwidth of less than half the sample frequency.

For our purpose, we have overridden control of the piezo-actuators in the x- and y-axes. This is connected to a "PiezoDrive PDL200", a linear voltage amplifier. The input into this amplifier is considered as the input to the system. The voltage output from the distance sensors in the x- and y-axes located on the AFM is used as the output of the system.

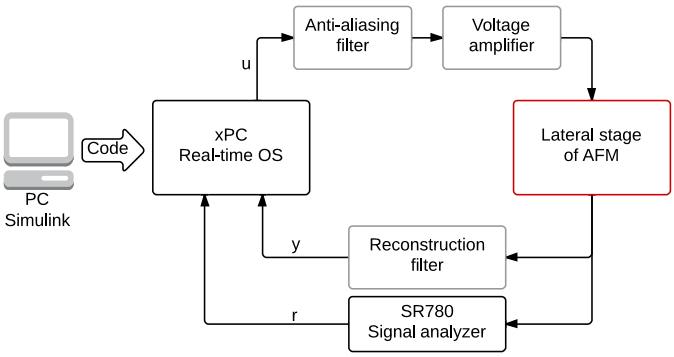


Fig. 1. Block diagram of the experimental setup for the closed-loop system. For the plant frequency response the SR780 device is connected directly to u.

2.2 Frequency Response and Model Fit

The lateral positioning stage of the AFM is considered to be dominantly linear, therefore the system can be described by its frequency response. The system has two inputs u_1, u_2 and two outputs y_1, y_2, along the x- and y-axis respectively. The frequency response of the plant $G(s)$ was gathered using a Stanford SR780 frequency analyser using a white noise source signal. One of several gathered frequency responses is plotted in Fig. 2 together with the fitted models. The transfer functions were fitted using the Matlab function `tfest` on the experimental data. The diagonal elements of $G(s)$ were approximated by a third-degree transfer function, while the off-diagonal elements were approximated by a second-degree function. The nominal plant model was found as shown in (1) at the top of the next page.

The exponential term represents the time delay between input and output. A time delay will present itself as a linear reduction of the phase as a function of frequency. Thus, we may find the time delay of the system between input and output by taking a look at the phase plot of the elements of $\hat{G}$. By assuming that the change in phase at lower frequencies is dominated by the time delay, and other sources of phase change is close to zero, we can deduce that the time delay is proportional to the slope at the start of the phase plot. This is how we identified the time delay $T_d = 4.58 \times 10^{-4}\,\mathrm{s}$.

We can see that the phase starts at 180° which means that the system has an inverse response, i.e. positive inputs give negative outputs and vice versa. This is just the sign convention of our raw data, and we decided not to change it for simplicity.

We can observe that the off-diagonal elements of $G(s)$ are relatively small compared to the diagonal elements, this indicates that the two axes are physically well decoupled. This indicates that the system is well suited for independent control of the axes where the cross-coupling is not considered. However, we will treat the system as a

$$G(s) = e^{-4.58e\text{-}04s} \begin{bmatrix} \dfrac{-5924s^2 - 1.709e07s - 9.878e10}{s^3 + 4703s^2 + 1.82e07s + 7.806e10} & \dfrac{-0.04567s^2 + 69.72s - 6.043e04}{s^2 + 104.5s + 2.29e07} \\ \dfrac{-0.04705s^2 + 89.17s - 1.253e05}{s^2 + 134.1s + 2.288e07} & \dfrac{-8708s^2 + 2.618e07s - 9.214e11}{s^3 + 3.865e04s^2 + 4.379e07s + 9.44e11} \end{bmatrix} \quad (1)$$

single multiple-input multiple-output (MIMO) plant and design a single more complex controller rather than two independent slightly simpler controllers.

2.3 Robust Stability and Uncertainty Weighting

Since the system has large uncertainties and inaccuracies in the model fits, we need to make sure it is robustly stable for a specified set of perturbations of the plant. We chose to model the uncertainties as multiplicative output uncertainty. The perturbed plant is described as

$$G_p = (I + \Delta W)\, G \quad (2)$$

where Δ is the uncertainty variable with $\|\Delta\|_\infty \leq 1$ and W is a specified weighting transfer function. The block-diagram of the feedback system is plotted in Fig. 3.

For a given controller K the robust stability (RS) condition for the described set of perturbations is (Skogestad and Postlethwaite, 2007)

$$\text{RS} \Leftrightarrow \|WT\|_\infty < 1 \quad (3)$$

where $T \triangleq (I + GK)^{-1}GK$ is the complementary sensitivity function. Similarly we have the sensitivity function $S \triangleq (I + GK)^{-1}$.

To find a suitable W that fits the uncertainties in our system, we can record a set of plant frequency responses $\hat{G} \in \Pi$. Then we can guarantee robust stability for at least all of these responses by finding a W such that $|W(j\omega)| > \hat{W}(\omega)$ where (Skogestad and Postlethwaite, 2007)

$$\hat{W}(\omega) \triangleq \max_{\hat{G}\in\Pi} \bar{\sigma}\left(\left(\hat{G}(\omega) - G(j\omega)\right) G^{-1}(j\omega)\right) \quad (4)$$

and $\bar{\sigma}(\cdot)$ is the maximum singular value. Other equations exist for different perturbation descriptions such as input multiplicative uncertainty or additive uncertainty.

We recorded three different frequency responses at different set-points and input amplitudes, and fitted the calculated $\hat{W}$ by the transfer function

$$W(s) = 3.8254\frac{(s + 210)(s + 1850)}{(s + 2400)(s + 3200)} \quad (5)$$

which is plotted together with $\hat{W}$ in Fig. 4.

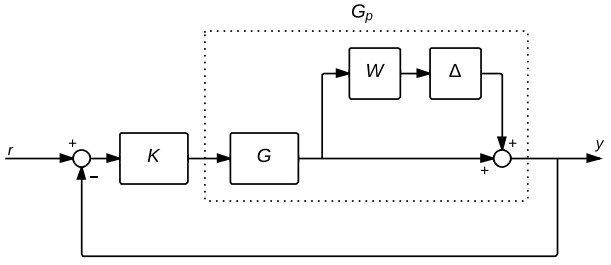


Fig. 3. Feedback system with a multiplicative output uncertainty

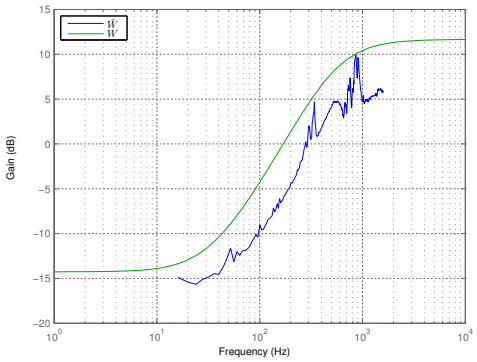


Fig. 4. Robustness fit, $W(s)$ and $\hat{W}(\omega)$

3. CONTROL LAW DESIGN

This section will present the design of the $\mathcal{H}_\infty$ controller. We will explain the choice of weightings for the mixed-sensitivity problem used to synthesize the controller based on the identified model G.

The $\mathcal{H}_\infty$ mixed sensitivity problem can be formulated as

$$\min_K N(K) = \left\| \begin{matrix} W_S S \\ W_T T \\ W_{KS} KS \end{matrix} \right\|_\infty \quad (6)$$

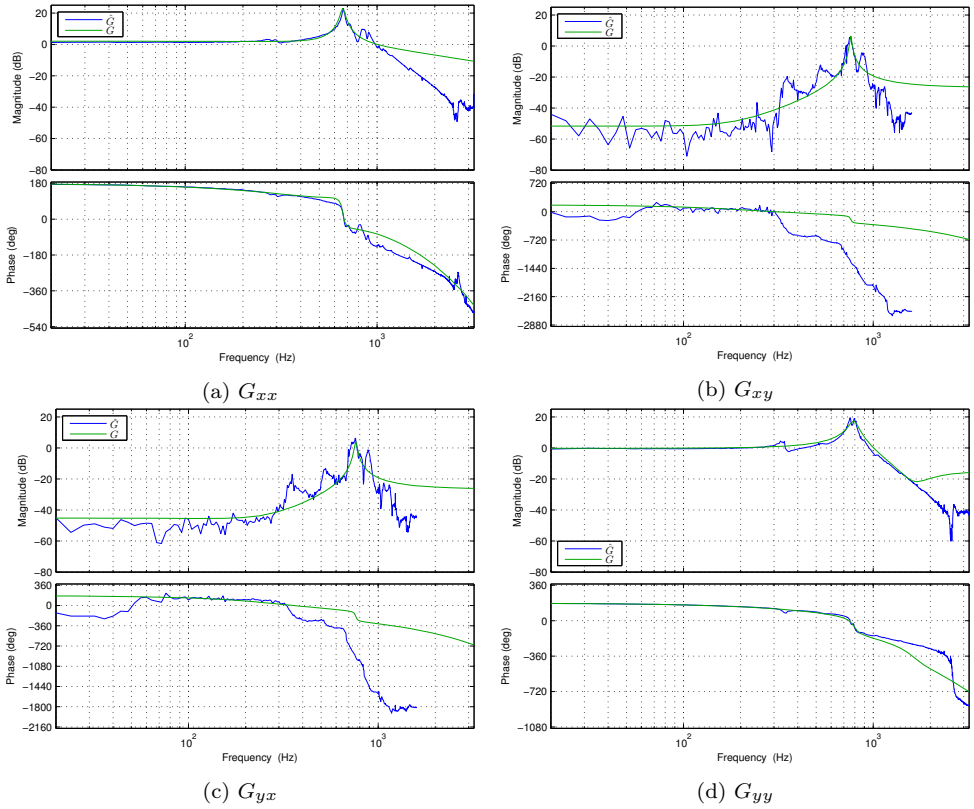
where W_S, W_T, and W_{KS} are user-defined weightings.

The sensitivity function S is the closed-loop transfer function from r to $e \triangleq y - r$. Therefore we want this to be as small as possible, especially in the bandwidth we would like to achieve effective control. Thus, within the desired bandwidth we want W_S to be large, so it was chosen as a first-order filter with large gains at low frequencies and low gains at high frequencies.

The complementary sensitivity function T is the closed-loop transfer function from r to y. Thus, we would like this to be close to one within the desired bandwidth for good tracking behavior. For higher frequencies we would like it to be as small as possible to attenuate measurement noise. Thus W_T is chosen to be small at low frequencies and large at high frequencies. Since WT is a measurement of the robust stability we have chosen $W_T = W$ to form the system to become more easily robustly stable.

Finally, we have the weighting W_{KS}. In fact KS is the closed-loop transfer function from r to u, so it describes the control effort for a given reference signal. We want to punish high frequencies of u since this means a large energy usage by the controller, and we know that the system won't respond to very high frequencies. Thus W_{KS} is modeled as a high-pass filter. The resulting weighting functions are summarized in Table 1.

The problem was solved using the Matlab function `mixsyn`. This resulted in a controller with 18 states. For the sake of comparison later on, we also designed a similar controller using independent axis design, i.e. one $\mathcal{H}_\infty$ controller for each axis, as well as a simple PID controller. The



(a) G_{xx} (b) G_{xy}

(c) G_{yx} (d) G_{yy}

Fig. 2. Experimental frequency response for both axes including cross-terms, and the corresponding fitted models, i.e. the elements of $G(s)$ and $\hat{G}(j\omega)$.

bandwidth of all three controllers as well as the robustness properties and model order is given in Table 2. The table shows us that the closed-loop system is robustly stable because $\|WT\|_\infty < 1$ for both of the $\mathcal{H}_\infty$ controllers. This is not the case for the PID-controller, so in fact we can not guarantee that this controller is stable for all perturbations of the system.

Table 1. Summary of weighting transfer functions

$W_S(s)$	$\dfrac{0.8333s + 439.8}{s + 0.04398}$
$W_T(s)$	$3.8254\dfrac{(s+210)(s+1850)}{(s+2400)(s+3200)}$
$W_{KS}(s)$	$0.8333\dfrac{s}{s+439.8}$

Table 2. Bandwidth and robustness comparison between the three controllers. TF = transfer function, SS = state space.

	$\omega_{b,S}$ [Hz]	$\omega_{b,T}$ [Hz]	$\|WT\|_\infty$	Implementation
PID	58.0	93.1	1.073	4th order TF
$\mathcal{H}_\infty$ SISO	75.6	96.4	0.9938	14th order SS
$\mathcal{H}_\infty$ MIMO	69.8	98.6	0.6717	18th order SS

4. SOLVER STABILITY

We will consider the case where a controller is represented by a continuous-time state-space model. A real-time implementation of such a model will use a solver to perform the necessary integration steps at fixed discrete time intervals, denoted by the step-size h. A complex controller will require a larger step-size because of limited computational power in the hardware, while at the same time an increased step-size can make the solver unstable. In this section we will see that the solver stability depends on the eigenvalues of the controller, the step-size, as well as the solver type.

Let us consider the scalar test system

$$\dot{y} = \lambda y \tag{7}$$

which is applied to a solver taking the discrete state y_n to the next time step y_{n+1} with step-size h,

$$y_{n+1} = \Phi(h\lambda) y_n \tag{8}$$
$$= [\Phi(h\lambda)]^n y_0 \tag{9}$$

where $\Phi(h\lambda)$ is called the stability function. It is evident that (9) is stable, i.e. $|y_n| \leq c < \infty \ \forall\, n \geq 0$, if and only if

$$|\Phi(h\lambda)| \leq 1 \tag{10}$$

All solvers we will consider have such a stability function, and the region of stability, i.e. the region of the complex plane where (10) is satisfied, varies between each solver type.

Example 1: Euler's Method applied to (7) gives

$$y_{n+1} = y_n + h(\lambda y_n)$$
$$= (1 + h\lambda)\, y_n \tag{11}$$

thus $\Phi(h\lambda) = 1 + h\lambda$, which is stable in the region $\{z \in \mathbb{C} \,|\, |1 + z| < 1\}$, in other words the unit circle with center -1. △

4.1 Runge-Kutta Methods

The family of explicit Runge-Kutta (ERK) methods can be written as

$$y_{n+1} = y_n + \sum_{i=1}^{s} b_i k_i \tag{12}$$

where s describes the number of stages of the Runge-Kutta method and

$$\begin{aligned} k_1 &= hf(t_n, y_n) \\ k_2 &= hf(t_n + c_2 h, y_n + a_{21} k_1) \\ k_3 &= hf(t_n + c_3 h, y_n + a_{31} k_1 + a_{32} k_2) \\ &\vdots \\ k_s &= hf(t_n + c_s h, y_n + a_{s1} k_1 + a_{s2} k_2 + \cdots + a_{s,s-1} k_{s-1}) \end{aligned}$$

where the coefficients a_{ij}, b_i, and c_i are elements of A, b, and c respectively, which are specified by a given ERK solver. Note that Euler's method is a first order ERK method.

4.2 Stability of Explicit Runge-Kutta Methods

The stability function of ERK methods are given by (Egeland and Gravdahl, 2002)

$$\Phi(z) = \det(I - zA + z\mathbf{1}b^T) \tag{13}$$

where $\mathbf{1}$ is a column vector of one-elements. It can be shown that this can be simplified for an ERK method of order $p = s$ to (Hairer and Wanner, 1996)

$$\Phi(z) = 1 + z + \cdots + \frac{z^p}{p!}$$

which is only possible for methods of order up to 4. Higher order ERK methods need more stages s than the order p. We have plotted the stability region of a selection of ERK methods in Fig. 5. Specifically the region of erk1-erk4 with $p = s$, Dormand-Prince 5 (erk5) and Dormand-Prince 8 (erk8). The last two with coefficients taken from Dormand and Prince (1980); Prince and Dormand (1981). The stability functions of these methods are reckoned to be the same as for the fixed-step solver methods available in Simulink.

4.3 Linear System

We have given the stability methods for several ERK methods for the scalar test system. In this section, we will show that the stability of the solvers applied to a linear system of ordinary differential equations (ODE) is given by its eigenvalues.

Theorem 1. Consider the system

$$\dot{y} = Ay \tag{14}$$

where A is an $n \times n$ diagonalizable matrix having eigenvalues $\lambda_1, \ldots, \lambda_n$. Let us apply an explicit Runge-Kutta method to this system. Then the ERK method has a stable point at the origin if and only if the same method has a stable point for

$$\dot{z} = \lambda_i z \quad \forall\, i \in [1, \ldots, n]$$

△

This is a standard result in numerical methods theory, see e.g. Ascher and Petzold (1998). We have used a wording similar to Frank (2008) which is also used for the next corollary.

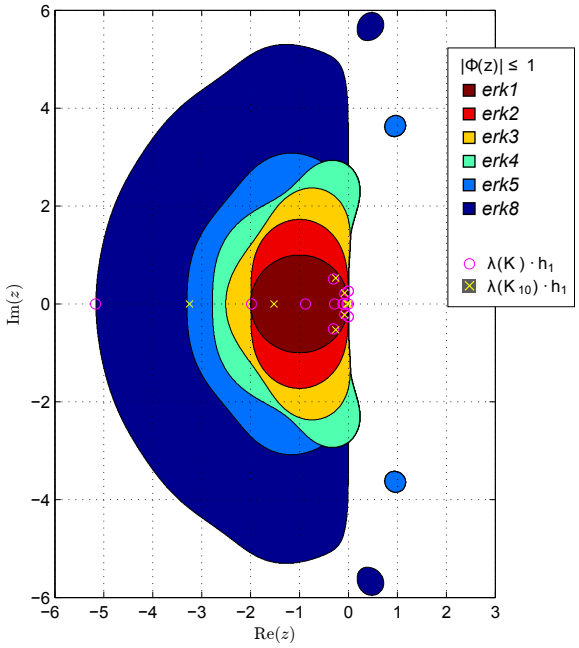


Fig. 5. Stability region of a variety of ERK methods of order 1-5 and 8. *Magenta circles*: Eigenvalues of the full order controller K scaled by the maximum step-size achieving stability for erk8. *Yellow X's*: Eigenvalues of the reduced tenth order controller K_{10} scaled by the same step-size. Note that K_{10} is stable for erk5.

Corollary 2. Consider a Runge-Kutta method with stability function $\Phi(z)$ applied to the system $\dot{y} = Ay$. Then the origin is stable for the numerical method with step-size h if and only if

$$|\Phi(h\lambda_i)| \leq 1, \quad \forall\, i \in [1, \ldots, n]$$

where λ_i are the eigenvalues of A. △

In other words, if all the eigenvalues of A are within the region of stability for a given solver at a specific step-size h, then the solver applied to (14) is stable. This gives us a tool to check for the required maximum step-size for a given controller and solver.

5. CONTROL ORDER REDUCTION

5.1 Model Reduction Theory

There exist several methods to perform model reduction Obinata and Anderson (2001), most widely used is possibly balanced residualization. Here the controller is first transformed to a balanced realization. A realization is said to be balanced if the controllability and observability *Gramians* are equal, thus a balanced model can be said to be as observable as it is controllable. The model states are ordered by decreasing Hankel singular values to form a state-space model (A, B, C, D). The last states are removed and the system is transformed such that

$$\left[\begin{array}{cc|c} A_{11} & A_{12} & B_1 \\ A_{21} & A_{22} & B_2 \\ \hline C_1 & C_2 & D \end{array}\right] \Rightarrow \left[\begin{array}{c|c} A_{11} - A_{12}A_{22}^{-1}A_{21} & B_1 - A_{21}A_{22}^{-1}B_2 \\ \hline C_1 - C_2 A_{22}^{-1} A_{21} & D - C_2 A_{22}^{-1} B_2 \end{array}\right]$$

The DC-gain of the system is maintained using this method, at some cost to the accuracy in the faster modes.

If we instead of a balanced realization used a canonical modal realization where the A-matrix is diagonal with elements equal to the eigenvalues, and performed the same reduction technique we can essentially remove the system eigenvalues of choice. This may be useful if some of the eigenvalues are larger than wanted, but can also result in large errors and possibly instability. Other methods include the truncation method which is more accurate at high frequencies at the cost of low frequencies, the optimal Hankel norm method, and using Linear Matrix Inequalities.

5.2 Results of Control Reduction

The designed controller K was first transformed to a balanced realization using the Matlab command `balreal`. The Hankel singular values for K are given in Table 3 which gives an idea of the error to be expected from removing each state.

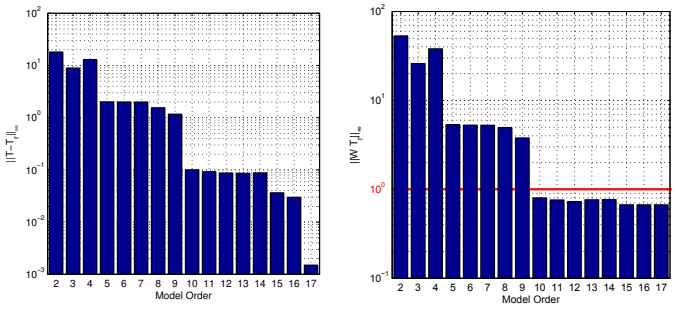
We performed model reduction on K by model residualization to several new controllers K_r of order 2 to order 17. This was done in Matlab with the command `modred`. The closed-loop $\mathcal{H}_\infty$ error norm on $T - T_r$, where T_r is the complementary sensitivity of the reduced controller, for each reduced controller is shown in Fig. 6a. We can see that there are significant drops specifically between order 4-5, 9-10, and 16-17. The error changes relatively little in-between these drops. Since we would like a controller with as low order as possible while maintaining the performance characteristics, we are inclined to select one of the orders after such a drop, i.e. 5, 10, or 17. The robustness norm is shown in Fig. 6b where we can see that only controller order 10 and higher are robustly stable with $\|WT_r\|_\infty < 1$. The previous discussion clearly favors choosing the 10th order controller as it provides robust stability with little error. This choice is further reinforced by considering the simulated step responses as shown in Fig. 7. The 10th order controller gives nearly indistinguishable results to the original controller, while the 8th and 9th order controllers shows some oscillatory behavior. The 7th order model is unstable, so we clearly want to avoid it.

Table 3. Hankel singular values of the balanced realization of the controller, σ_i

1)	5.436e+03	*7)*	6.799e-02	*13)*	1.324e-02
2)	4.227e+03	*8)*	6.049e-02	*14)*	4.091e-03
3)	3.307e-01	*9)*	3.885e-02	*15)*	4.073e-03
4)	2.775e-01	*10)*	1.977e-02	*16)*	1.400e-03
5)	1.246e-01	*11)*	1.514e-02	*17)*	1.044e-03
6)	7.628e-02	*12)*	1.430e-02	*18)*	5.722e-05

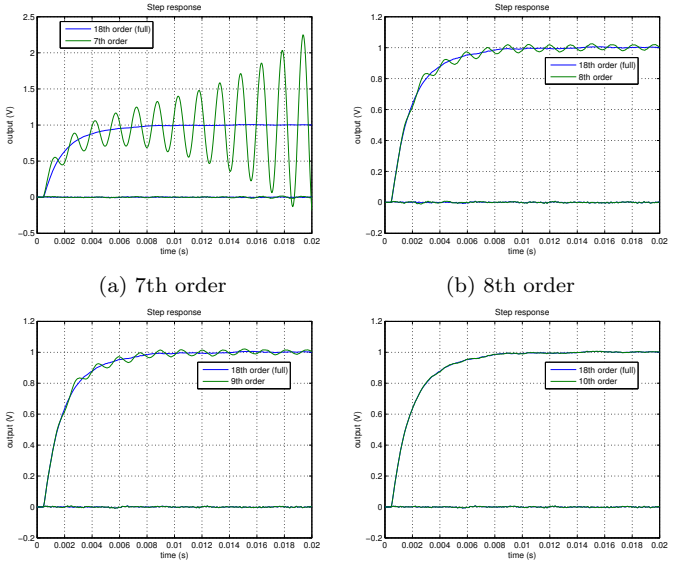
5.3 Eigenvalues and Maximum Step-Size

We have previously shown that the stability of an explicit Runge-Kutta method applied to a state-space model depends on the eigenvalues of the A-matrix and the step-size. Thus, for a given controller we can find the maximum step-size needed for stability. The maximum step-size for controller K was found to be 54.62 µs using the erk8 solver. The eigenvalues scaled by step-size can be seen in Fig. 5 (magenta circles) which are seen to lie within the stability region of erk8. It is not stable for erk5 as the eigenvalues



(a) (b)

Fig. 6. Reduced controller order properties. **(a)** Closed-loop error $\|T - T_r\|_\infty$. **(b)** Robustness $\|WT_r\|_\infty$, must be < 1 for robust stability (marked red).



(a) 7th order (b) 8th order

(c) 9th order (d) 10th order

Fig. 7. Simulated step-response in reference signal on the x -axis, reduced vs original controller. Shows output from both x-axis and y-axis.

are outside the stability region for this solver. The eigenvalues of the reduced controller K_{10} has also been plotted (yellow x's) for the same step-size and we can see how the controller reduction has affected the eigenvalues. We can see that they have become smaller which has resulted in the controller becoming stable even for the erk5 solver. So not only does model reduction reduce the computational complexity of the controller, but it can also enable us to use a simpler solver type or alternatively a larger step-size.

The maximum step-size for a variety of controllers and solver types are shown in Table 4. Note that the maximum step-size does not strictly increase with lower model-orders, since the model residualization method used does not necessarily reduce the eigenvalues. Other reduction methods could be considered if this is critical, such as methods directly removing states with large eigenvalues.

6. EXPERIMENTAL RESULTS

Experiments were performed for two reasons. The first was to see how well the reduced order controllers performed

Table 4. Maximum step-size for a given explicit Runge-Kutta (ERK) method for the various reduced controllers as well as some simpler controllers and the nominal plant model. Larger values are generally better because they are stable at higher step-sizes.

Order	h_{max} [µs]					
	erk1	erk2	erk3	erk4	erk5	erk8
7 (unst.)	32.18	71.33	95.74	100.8	127.2	203.6
8	58.37	58.37	73.34	81.29	96.51	150.8
9	37.54	37.54	47.17	52.28	62.07	96.99
10	33.61	33.61	42.22	46.8	55.56	86.82
11	39.70	39.7	49.87	55.28	65.63	102.5
12	24.96	32.66	41.03	45.48	54.00	84.37
13	2.821	21.55	27.07	30.01	35.63	55.67
14	2.818	19.11	24.01	26.62	31.60	49.38
15	2.843	21.73	27.30	30.26	35.92	56.13
16	2.948	22.64	28.44	31.53	37.43	58.48
17	2.918	21.18	26.61	29.49	35.01	54.71
18 (full)	2.910	21.14	26.56	29.45	34.96	54.62
PID	80.00	80.00	100.5	111.4	132.3	206.7
$\mathcal{H}_{\infty SISO}$	21.45	21.45	26.95	29.87	35.46	55.41
$G(s)$	4.564	52.42	65.85	73.00	86.66	135.4

compared to our simulations. The second was to record the *average task execution time* (TET), the time it takes the hardware to perform calculations from one time-step to the next. This can be considered a measurement of the computational complexity of the controller, and is a lower limit on the step-size. Any lower than this and the hardware will not be able to meet its deadline and exit with a "CPU overload" error.

The experiments were performed in the setup shown in Fig. 1. The closed loop frequency response of the original controller K compared to the reduced controllers K_{10} and K_8 is given in Fig. 8. The average TET for the various controllers and solver types are given in Table 5. The system was run with step-size $h = 40\,\mu s$, and only the modes that are stable at this step-size as can be seen from Table 4 was tested, i.e. $h_{max} \geq 40\,\mu s$.

7. DISCUSSION

7.1 Model Reduction and Performance

The model reduction showed us that the original 18th order controller could be reduced to a 10th order model with no noticeable difference in the simulated step-response or the experimental closed-loop frequency response. Additionally, it was shown to maintain robust stability, thus it is a very viable controller choice. In terms of the impact on computational complexity, we can see that we have reduction of 25.5%, from 20.11 to 14.99 µs if we use the erk8 solver for both controllers.

From Table 4 we can see that the 10th order controller can run using the erk3 solver at a step-size of $h = 40\,\mu s$, while the full $\mathcal{H}_\infty$ MIMO controller needs erk8 for stability at this step-size. By choosing erk3 for the 10th order controller we can see from Table 5 that this reduces the execution time to 10.71 µs, or a 46.7% reduction from the original controller.

Table 5. Average Task Execution Time (TET) with different controller model order and solver types which gives an indication on the computational complexity. Step-size $h = 40\,\mu s$. Dash (-) unstable, not tested. (x) CPU overload.

Order	Average TET [µs]					
	erk1	erk2	erk3	erk5	erk8	ode14x
≤7	-	-	-	-	-	-
8	-	10.25	10.43	11.10	13.64	26.72
9	-	-	10.55	11.43	14.26	x
10	-	-	10.71	11.64	14.99	x
11	-	-	10.89	11.91	15.79	x
12	-	-	11.07	12.36	16.79	x
13	-	-	-	-	17.68	x
14	-	-	-	-	18.82	x
15	-	-	-	-	19.61	x
16	-	-	-	-	20.91	x
17	-	-	-	-	22.61	x
18	-	-	-	-	20.11	x
PID	-	9.95	9.98	10.13	11.12	14.91
$\mathcal{H}_{\infty SISO}$	-	-	-	-	14.58	x

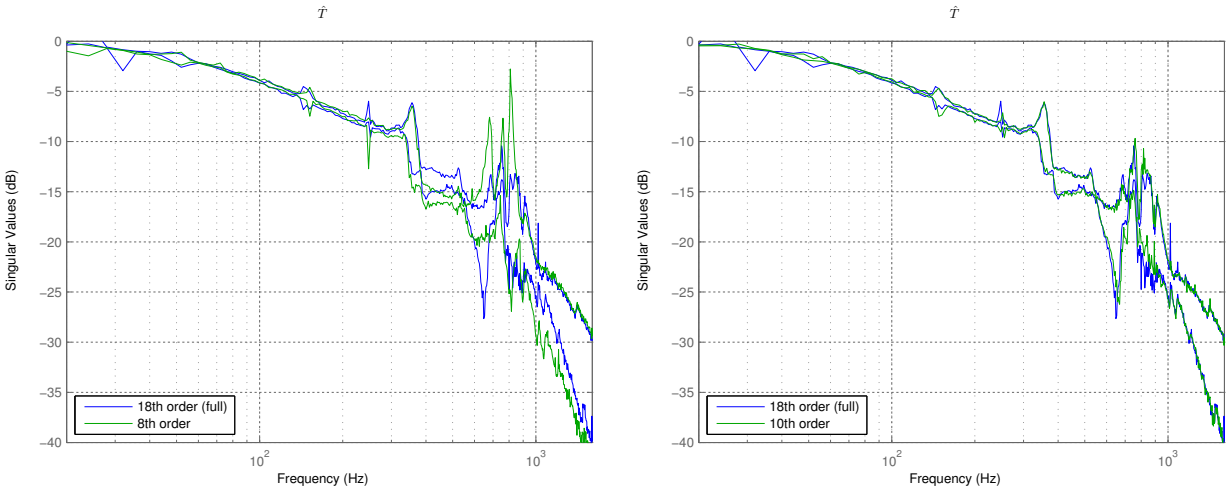
We tried to run the controller with the implicit solver *ode14x (Extrapolation)* supplied with Simulink, but it was found to be such computationally demanding that we were only able to run it with the 8th order controller in addition to the PID controller. Additionally, the solutions was found to explode at times in our simulations so this solver was not further considered.

It is also interesting to note that the execution time does not strictly decrease with increased controller order, e.g. the 17th to 18th order controller. By inspecting the state-space model of each of these controllers, we notice that the 18th order controller has a lot more zero-valued elements. We speculate that the compiler simplifies the arithmetic on these elements.

7.2 Eigenvalues and Stability

As we have seen, the eigenvalues of the controller are one of the decisive factors for stability of an applied ERK method. Hence, it is important to consider how controller reduction changes the eigenvalues, especially for a fast and stiff system such as our lateral positioning platform of an AFM. The eigenvalues of our controller tended to become smaller with reduced orders, but this need not be the case. One should be careful when performing model reduction and possibly verify that the eigenvalues are within the stable region of the solver considered. If the solver stability becomes a problem, one should consider a different model reduction method, such as removing the eigenvalues directly from a canonical modal realization of the controller.

We have also seen how increased solver order increases the stability region, but at the same time it increases the execution time. This will ultimately be a trade-off between moving the lower limit (due to computation time) and the upper limit (for stability) of the step-size, as illustrated in Fig. 9. Halving the step-size usually doubles the computational complexity (per unit of time), while



(a) Reduced 8th Order

(b) Reduced 10th Order

Fig. 8. Singular values of the closed-loop $\sigma(\hat{T})$ for the 8th and 10th order reduced controller versus the original controller.

increasing the solver complexity is harder to predict, but will generally depend on the controller order.

Note that we have not considered the accuracy of the solver methods. This is because we have assumed that the system is stiff, and stiff systems are characterized such that instability comes before any accuracy problems.

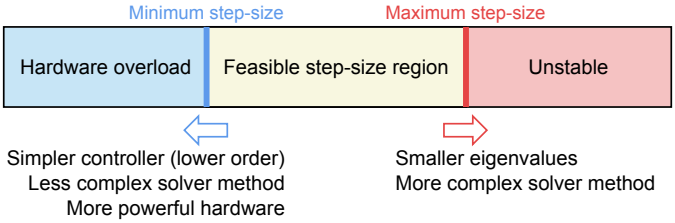


Fig. 9. Illustration of the trade-off between step-size, controller complexity, solver complexity, eigenvalues of the controller, and hardware performance for a real-time controller implementation.

8. CONCLUSION

This paper concerns itself with some practical issues for a controller running in real-time. Since the capability of hardware is limited in terms of computational performance, a complex controller can become difficult to implement with a step-size small enough for stability. This paper tries to achieve two goals, (1) to show how to determine the maximum step-size a controller requires for numerical stability, and (2) to show what can be done to reduce the computational complexity of an already existing controller with a focus on model reduction.

To show how this can be done, a robustly stable $\mathcal{H}_\infty$ controller was designed for a nanopositioning device. We showed that the numerical stability of an explicit Runge-Kutta method is determined by the eigenvalues of the controller. Model reduction was performed on the controller and the reduced controllers were compared in terms of performance and stability both in simulations and experiments which showed that the 18th order controller could be reduced to a 10th order model without any significant reduction in performance or stability. After the reduction process, the largest eigenvalues were reduced in size so the system also became numerically stable for even simpler solver types which allow further reduction of the computational requirement.

An experiment was run to determine the change in computational complexity by recording the execution time of the various controllers. The reduction to a 10th order model, as well as the possibility of using a simpler solver type resulted in a 46.7% reduction in task execution time.

REFERENCES

Anderson, B.D.O. (1992). Controller Design : Moving from Theory to Practice. *IEEE Control Systems Magazine*, 16–25.

Anderson, B.D.O. and Liu, Y. (1989). Controller reduction: concepts and approaches. *Automatic Control, IEEE Transactions on*, 34(8).

Ascher, U. and Petzold, L. (1998). *Computer methods for ordinary differential equations and differential-algebraic equations*.

Dong, J., Salapaka, S.M., and Ferreira, P.M. (2007). Robust MIMO control of a parallel kinematics nanopositioner for high resolution high bandwidth tracking and repetitive tasks. *Decision and Control, 2007. 46th IEEE Conference on*, 4495–4500.

Dormand, J. and Prince, P. (1980). A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1), 19–26.

Egeland, O. and Gravdahl, J.T. (2002). *Modeling and Simulation for Automatic Control*.

Frank, J. (2008). Stability of Runge-Kutta Methods (Lecture notes for course "Numerical Modelling of Dynamical Systems").

Hairer, E. and Wanner, G. (1996). *Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems*.

Kuiper, S. and Schitter, G. (2012). Model-based feedback controller design for dual actuated atomic force microscopy. *Mechatronics*, 22(3), 327–337.

Ladjal, H., Hanus, J.L., and Ferreira, A. (2009). H_inf robustification control of existing piezoelectric-stack actuated nanomanipulators. *2009 IEEE International Conference on Robotics and Automation*, 3353–3358.

Lee, C. and Salapaka, S.M. (2009). Fast Robust Nanopositioning: A Linear-Matrix-Inequalities-Based Optimal Control Approach. *Mechatronics, IEEE/ASME Transactions on*, 14(4), 414–422.

Obinata, G. and Anderson, B.D.O. (2001). *Model reduction for control system design*. Springer-Verlag New York, Inc.

Prince, P. and Dormand, J. (1981). High order embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*.

Ragazzon, M.R.P. (2013). *Nanopositioning in Atomic Force Microscopes: Robust Control Design, Order Reduction and Numerical Implementability*. Master's thesis, Norwegian University of Science and Technology.

Salapaka, S., Sebastian, a., Cleveland, J.P., and Salapaka, M.V. (2002). High bandwidth nano-positioner: A robust control approach. *Review of Scientific Instruments*, 73(9), 3232.

Salapaka, S. and Sebastian, A. (2003). Control of the nanopositioning devices. *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, 3(December), 3644–3649.

Schitter, G., Menold, P., Knapp, H.F., Allgöwer, F., and Stemmer, A. (2001). High performance feedback for fast scanning atomic force microscopes. *Review of Scientific Instruments*, 72(8), 3320.

Schitter, G. and Stemmer, a. (2004). Identification and Open-Loop Tracking Control of a Piezoelectric Tube Scanner for High-Speed Scanning-Probe Microscopy. *IEEE Transactions on Control Systems Technology*, 12(3), 449–454.

Sebastian, A. and Salapaka, S.M. (2005). Design methodologies for robust nano-positioning. *IEEE Transactions on Control Systems Technology*, 13(6), 868–876.

Skogestad, S. and Postlethwaite, I. (2007). *Multivariable feedback control: analysis and design*. Wiley-Interscience, 2 edition edition.

Yong, Y., Liu, K., and Moheimani, S. (2010). Reducing cross-coupling in a compliant XY nanopositioner for fast and accurate raster scanning. *Control Systems Technology, IEEE Transactions on*, 18(5), 1172–1179.