# Efficient and symbolic computation of the $H_2$ norm via the polynomial Diophantine equation

Gareth H. Willetts and Timothy H. Hughes[1]

***Abstract*— The $H_2$ norm is a widely used metric for characterising and optimising system performance for a wide variety of applications. This article presents an alternative method for calculating the $H_2$ norm which is more computationally efficient than existing methods, and delivers greater numerical precision, while also allowing for the $H_2$ norm to be calculated numerically or symbolically. A stability test of the system can be performed at no added computational cost.**

## I. INTRODUCTION

The $H_2$ norm, defined as the square root of the power spectral density of a system's output in response to zero mean white noise input of unit power spectral density, is a widely used metric for characterising system performance. Example applications include optimising renewable energy systems [1], mechanical systems [2] and the Linear Quadratic Gaussian (LQG) controller design. These necessitate an efficient, numerically stable and precise method of calculation of the $H_2$ norm, due to its range of critical applications.

Current methods focus on solving Lyapunov equations to compute the observability or controllability Gramian [3, pp. 365-372], as is the case in MATLAB. However, the number of unknowns in the Lyapunov equation scale with the square of the degree of the system's transfer function, leading to large matrices that are computationally expensive to solve. In contrast, this paper presents an alternative method which calculates the $H_2$ norm using an original algorithm to calculate the solution to a polynomial Diophantine equation. It was recognised in [4] that, for single-input single-output systems, the $H_2$ norm can be obtained by solving a linear equation involving a Hurwitz matrix. Building on this result, we exploit the structure of the Hurwitz matrix to produce a series of recursive polynomial equations to yield an efficient computation of the $H_2$ norm. This method is both faster than the inbuilt MATLAB function, shown through testing in MATLAB r2021b, and more precise. Indeed, it can be computed using exact arithmetic. Finally, it allows for the $H_2$ norm to be calculated symbolically in terms of parameters of the system, illustrated here through an example of optimal design of a passive train suspension system.

We define the following notation, used throughout this article. If $x$ is a real number, then $\lceil x \rceil$ rounds $x$ up to the next integer, and $\lfloor x \rfloor$ rounds $x$ down to the previous integer. If $x$ and $y$ are integers, $x\%y$ denotes the remainder in the division of $x$ by $y$. Finally, let $\mathbb{LC}(q)$ denote the leading coefficient of the polynomial $q(s)$.

[1]Gareth H. Willetts and Timothy H. Hughes are with College of Engineering, Maths and Physical Sciences, University of Exeter, Penryn, Cornwall, ghw205@exeter.ac.uk, t.h.hughes@exeter.ac.uk

## II. METHOD

Our focus is on single-input, single-output systems characterised by their transfer function $G(s)$, which is a real, rational function given as the ratio of two polynomials. The $H_2$ norm of such a system exists if and only if $G(s)$ is proper (i.e., $\lim_{s\to\infty}(G(s) = 0)$), and stable (i.e., all of the poles of $G(s)$ are in the open left half plane), and throughout this article this shall be assumed to be the case. In fact, the method to be presented allows for a stability test to be performed with no added computational effort. The transfer function, $G(s)$, is stated in terms of a numerator polynomial $c(s)$ and denominator polynomial $a(s)$ as follows:

$$G(s) = \frac{c(s)}{a(s)} = \frac{c_{n-1}s^{n-1}+c_{n-2}s^{n-2}+\ldots+c_1 s+c_0}{a_n s^n+a_{n-1}s^{n-1}+\ldots+a_1 s+a_0}. \quad (1)$$

Without loss of generality, let $a_n > 0$ and assume $c(s)$ and $a(s)$ have no common roots in the closed right half plane. All of the poles of $G(s)$ are in the open left half plane; it then follows that we require $a(s)$ to have its roots in the open left half plane.

In this article we will consider the case in which $n$ is odd; indeed, the method works similarly for transfer functions of even degree, with slight changes to the initial polynomial definitions. The full details of these differences, and the proof of the method, will be published in an upcoming journal paper.

Now, let

$$c^e(s) = c_0 + c_2 s + \ldots + c_{2\lfloor\frac{n-1}{2}\rfloor}s^{\lfloor\frac{n-1}{2}\rfloor},$$

$$c^o(s) = c_1 + c_3 s + \ldots + c_{2\lceil\frac{n-3}{2}\rceil+1}s^{\lceil\frac{n-3}{2}\rceil},$$

$$z_0(s) = (c^e(s))^2 - s(c^o(s))^2, \quad (2)$$

$$p_1(s) = a_1 s + \ldots + a_{n-2}s^{\lceil\frac{n}{2}\rceil-1} + a_n s^{\lceil\frac{n}{2}\rceil}, \quad (3)$$

$$\text{and } p_2(s) = a_0 + \ldots + a_{n-3}s^{\frac{\lceil n-1\rceil}{2}-1} + a_{n-1}s^{\frac{\lceil n-1\rceil}{2}}, \quad (4)$$

whereupon $c(s) = c^e(s^2) + sc^o(s^2)$.

To calculate the $H_2$ norm, we recursively compute polynomials $z_1, \ldots, z_{n-1}$ and $p_3, \ldots, p_{n+1}$ as follows. First, we can compute $z_1, p_3$ by

$$z_1(s) = \mathbb{LC}(p_1)z_0(s) - s^{\lceil\frac{n-2}{2}\rceil}z_{0,n-1}p_1(s), \quad (5)$$

$$p_3(s) = \mathbb{LC}(p_2)p_1(s) - s^{n\%2}\mathbb{LC}(p_1)p_2(s) \quad (6)$$

Next, we can compute $z_2, p_4$ by

$$z_2(s) = \mathbb{LC}(p_2)z_1(s) - s^{\lceil\frac{n-3}{2}\rceil}z_{1,n-2}p_2(s), \quad (7)$$

$$p_4(s) = \mathbb{LC}(p_3)p_2(s) - s^{(n-1)\%2}\mathbb{LC}(p_2)p_3(s) \quad (8)$$

The polynomials $p_5(s), p_6(s), \ldots, p_{n+1}(s), z_3(s), z_4(s), \ldots, z_{n-1}(s)$ are further obtained by the following recursive equations:

$$z_i(s) = \frac{\mathbb{LC}(p_i)}{\mathbb{LC}(p_{i-1})} z_{i-1}(s) - s^{\lceil \frac{n-i-1}{2} \rceil} \frac{z_{i-1,n-i}}{\mathbb{LC}(p_{i-1})} p_i(s), \tag{9}$$

$$p_{i+2}(s) = \frac{\mathbb{LC}(p_{i+1})}{\mathbb{LC}(p_{i-1})} p_i(s) - s^{(n-i+1)\%2} \frac{\mathbb{LC}(p_i)}{\mathbb{LC}(p_{i-1})} p_{i+1}(s) \tag{10}$$

for $i = 3, 4, \ldots, n-1$

where $\mathbb{LC}(p_k) > 0$ for all $k = 0, \ldots, n+1$ due to $G(s)$ being stable, and the degrees of the polynomials satisfy $\deg(p_{i+2}(s)) = \deg(p_i(s)) - 1$, and $z_{n-1}(s), p_{n+1}(s)$ are scalar constants.

Having computed this sequence of polynomials, the $H_2$ norm is then obtained as

$$\frac{z_{n-1}}{2a_n p_{n+1}}.$$

The coefficients of $p_{k+1}$ for all $k \in 1, \ldots, n$ will be integers whenever the coefficients of the originating polynomials $p_1, p_2$ and $z_0$ are integers. In this case, all quantities can be calculated exactly and efficiently using integer arithmetic.

## III. RESULTS

A MATLAB r2021b implementation has been tested against the in-built function provided by MathWorks. Tests were performed on an i7-12700k processor and 16GB DDR4 RAM. The method outlined in this article outperforms the inbuilt MATLAB implementation.

| $n$ | MATLAB (inbuilt) | MATLAB (New Algorithm) |
|---|---|---|
| 5 | 0.707 | 0.00909 |
| 7 | 0.742 | 0.0146 |
| 9 | 0.784 | 0.0149 |
| 21 | 1.20 | 0.0323 |

Fig. 1. Execution times in ms for the computation of the $H_2$ norm of the transfer function $G(s)$, averaged over 10000 runs with the first 1000 removed to warm-up the code, where $c(s) = \sum_{i=0}^{n-1} s^i$, $a(s) = \sum_{i=0}^{n-1} \binom{n}{i} s^i$, for varying degrees n. Here, Inbuilt corresponds to the MATLAB function `norm`, and New corresponds to the algorithm described in Section II of this article.

Checking $p_i > 0$ ensures that $G(s)$ is stable and has little effect on computation time; however, MATLAB requires the `isstable` command to do this, adding a further 0.205ms.

## IV. SYMBOLIC COMPUTATION

Wang et al. considered a train suspension detail, repeated on each wheelset [5]. The paper explores suspension designs of the form shown in Fig. 1, where the admittances $Q_1$ and $Q_2$ takes the form $Q_1(s) = \frac{k_s}{s} + K_1(s)$ and $Q_2(s) = \frac{k_b}{s} + K_2(s)$, where $k_s$ and $k_b$ denote stiffness coefficients of the body and bogie, respectively, and $K_1(s)$ and $K_2(s)$ correspond to the suspension admittances (to be designed). The aim was to optimise passenger comfort, corresponding to minimising the $H_2$ norm of the transfer function from the rail track displacement $z_r$ to the velocity of the train body $\frac{dz_s}{dt}$, herein denoted by $J_1$.

The method outlined in this article can be applied symbolically. For example, [5] considered the case in which the suspension admittances $Q_1$ and $Q_2$ took the form $Q_1(s) = \frac{k_s}{s} + c_s$ and $Q_2(s) = \frac{k_b}{s} + c_b$, for some fixed stiffnesses $k_s$ and $k_b$, and where $c_s > 0$ and $c_b > 0$ are chosen to minimise $J_1$. The method provided in Section II of this article can be applied to obtain an analytical expression for $J_1$ in terms of these parameters, where a local minimum is found. This is supported by computing partial derivatives with respect to each parameter, and solving the resulting pair of bivariate polynomial equations to determine the local stationary points. This can be used to prove that the function has only one local minimum. Therefore, the unique global minimum is determined, which agrees with the value for $J_1$ reported in the paper up to the seventh significant figure. The accuracy for this method exceeds the optimisation used in the paper as it finds an exact expression for the global minimum [5].

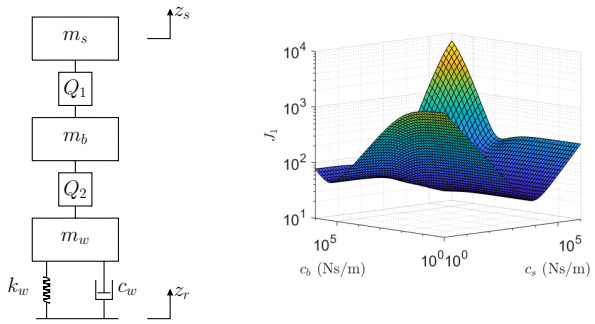


Fig. 2. Left: Schematic of a train suspension system. Here, $m_s$ denotes the spring mass, $m_b$ the mass of the bogie, $m_w$ the mass of the wheel, $k_w$ the tyre stiffness, $c_w$ the tyre damping coefficient, and $Q_1$ and $Q_2$ are suspension admittances to be designed. Right: Surface plot of the $H_2$ norm of the transfer function from $z_r$ to $\frac{dz_s}{dt}$ as a function of the suspension damping coefficients $c_s$ and $c_b$, for the case in which $m_s = 3500\text{kg}, m_b = 250\text{kg}, m_w = 350\text{kg}, k_w = 8 \times 10^9\text{N/m}, c_w = 670 \times 10^3\text{Ns/m}, k_s = 141 \times 10^3\text{N/m}, k_b = 1260 \times 10^3\text{N/m}$.

Attempts to replicate the results of [5, pp. 818] have been difficult; MATLAB r2021b claims the system is unstable when attempting to calculate the $H_2$ norm $J_3$, when $K_1(s) = K_1^{3rd}(s)$ and $K_2(s) = K_2^{3rd}(s)$ (here, $J_3$ characterises the dynamic wheel load).

## REFERENCES


[1] Lescher, Fabien & Zhao, Yun & Martinez, André. (2006). Multiobjective $H_2/H_\infty$ control of a pitch regulated wind turbine for mechanical load reduction.

[2] Zhu Y, Qiu J, Tani J, Urushiyama Y, Hontani Y. Simultaneous Optimization of Structure and Control for Vibration Suppression. Journal of Vibration and Acoustics. 1999;121(2):237-243.

[3] Zhou K, Doyle J, Glover K. Robust and optimal control. Upper Saddle River, NJ: Prentice Hall; 1996.

[4] Betser A, Cohen N, Zeheb E. On solving the Lyapunov and Stein equations for a companion matrix. Systems & Control Letters. 1995;25(3):211-218.

[5] Wang F, Liao M, Liao B, SuW, Chan H. The performance improvements of train suspension systems with mechanical networks employing inerters. Vehicle System Dynamics. 2009;47(7):805-830.


## ACKNOWLEDGMENT


This work was fully supported by the UK Engineering and Physical Sciences Research Council (EPSRC) grant EP/T518049/1 for the University of Exeter.