A Hybrid Symbolic-Numerical Simulation Method for Some Typical Boundary Control Problems

Jinsong Liang
YangQuan Chen
Center for Self-Organizing and Intelligent Systems (CSOIS)
Department of Electrical and Computer Engineering
4160 Old Main Hill
Utah State University, Logan, UT 84322-4160
yqchen@ece.usu.edu

Bao-Zhu Guo
Institute of Systems Sciences
Academy of Mathematics and System Sciences
Academia Sinica
Beijing 100080
People's Republic of China

Although boundary control of linear partial differential equations has become an important research area, there is still no readily available simulation tool to help researchers analyze and design. In this article, a simulation method for some typical boundary control problems, combining symbolic math and a numerical method, is presented with application examples. In the intermediate steps of the simulation, an important by-product, transfer function of the controlled system, can be obtained, which makes the design of more advanced boundary controllers possible and much easier.

Keywords: Simulation, symbolic, numeric, boundary control, beam equation, numerical inverse Laplace transform

1. Introduction

Boundary control of linear partial differential equations (PDEs) has become an important research area in recent years [1-6] due to the increasing demand on the high precision control of many mechanical systems, such as spacecraft with flexible attachments or robots with flexible links, which are governed by PDEs rather than ordinary differential equations (ODEs). Contrary to the progress made in theoretical analysis, simulation examples in publications are very few, even though simulation plays such an important role in verifying theoretical analysis and design, identifying potential problems, reducing investment, and selecting the optimal solution. The reason for this, we would like to suggest, is that the difficulty of simulating boundary control problems is far beyond the capability of most commonly available mathematical tools such as Matlab, Maple, and even FEMLAB (see www.femlab.com). For example, the Matlab PDE Toolbox is only able to solve second-order PDEs with Dirichlet and/or generalized Neumann boundary conditions [7], while the PDEs of most boundary control problems are either of higher order or/and the boundary conditions are much more complicated than what the Matlab PDE Toolbox could handle.

In this article, we present an easy to implement, yet powerful, boundary control simulation method, which combines the analytical method, the numerical method, and modern symbolic algebra. The simulation examples show that this method applies to a wide range of boundary control problems. The method is also much easier to implement than finite element method (FEM) or the finite difference method (FDM) [8]. No extra software is needed, except Matlab and the Matlab Symbolic Math Toolbox.

This article is organized as follows. Section 2 introduces the principle and the implementation procedure via an example. Section 3 shows some interesting applications of this simulation method. Finally, section 4 concludes the article.

SIMULATION, Vol. 80, Issue 11, November 2004 635-643

DOI: 10.1177/0037549704050183

2. Problem Formulation, Principle, and Implementation

In this section, the following example will be used to show how a typical boundary control problem is formulated. The same example will be used to demonstrate the basic principle and implementation details of the simulation method presented in this article.

Consider a string whose behavior is governed by the wave equation. Denote the displacement of the string by $u(x, t)$ at $x \in [0, 1]$ and $t \geq 0$. The string is fixed at one end and stabilized by a boundary controller at the other end. The system is represented by

$$u_{tt}(x, t) - u_{xx}(x, t) = 0, \tag{1}$$

$$u(0, t) = 0, \tag{2}$$

$$u_x(1, t) = f(t), \tag{3}$$

where the subscript (e.g., the t as in u_t) denotes a partial differential with respect to the corresponding variable. $f(t)$ is the combination of boundary control force and the disturbance $n(t)$ applied at the free end of the string. The control objective is to stabilize $u(x, t)$, given the initial conditions.

Suppose we have designed the following dynamic boundary controller:

$$\hat{f}(s) = \left(d + \frac{ks}{s^2 + \omega^2}\right)\hat{u}_t(1, s) + \hat{n}(s), \tag{4}$$

where $\hat{f}(s)$ is the Laplace transform of the combination of boundary control force and disturbance force, $\hat{n}(s)$ is the Laplace transform of the disturbance force $n(t)$, $\hat{u}_t(1, s)$ is the Laplace transform of the velocity of the free end, d and k are the control gains, and ω is the frequency of the noise.

We want to show that the dynamic boundary controller ($k > 0$) is better than the static boundary controller ($k = 0$) to reject the effect of the noise $n(t)$. This problem was raised in Morgül [4], one of the very few studies with simulation examples. FDM was used in Morgül to simulate the system.

It is well known that such linear PDEs can be solved by means of the Laplace transform [9]. The following is a summary of this method. We assume that the solution of a PDE is a function $u(x, t)$ of the two independent variables x and t.

1. Transform $u(x, t)$ with respect to t by means of the Laplace transform, so we obtain an ODE for the transformed variable $U(x, s)$:

$$f\left(U(x, s), \frac{dU(x, s)}{dx}, \ldots, \frac{d^n U(x, s)}{dx^n}, x, s\right) = 0. \tag{5}$$

2. Solve the ODE (5) for $U(x, s)$ as a function of x, with the transform variable s still appearing as a parameter in the solution, and use the boundary conditions of the original problem to determine the precise form of $U(x, s)$.

3. Take the inverse Laplace transform of $U(x, s)$ with respect to s to find the solution $u(x, t)$.

Several problems make the above method hard to use in practice to solve a PDE boundary control problem. First, if (5) is of high order, the general solution is too complicated to obtain. Second, due to the high order of the ODE and the complicated boundary conditions, the arbitrary constants in the general solution of the ODE are hard to determine. Third, even if we can determine the undefined constants, usually the inverse Laplace transform cannot be performed by looking up a table of transform pairs.

We solve the above problems using the Matlab Symbolic Math Toolbox [10] and the numerical inverse Laplace transform [11, 12]. The detailed implementation procedures are demonstrated in the following example.

The initial conditions are chosen as

$$u(x, 0) = -0.5 \sin(0.5\pi x), \tag{6}$$

$$u_t(x, 0) = 0. \tag{7}$$

The disturbance $n(t)$ is chosen as

$$n(t) = \cos(10t). \tag{8}$$

We will simulate the following two cases to show that the dynamic controller ($k > 0$) is better than the static controller ($k = 0$) to reject the noise:

- *Case 1*: $d = 1,\ k = 10,\ \omega = 10.$
- *Case 2*: $d = 1,\ k = 0,\ \omega = 10.$

The simulation of case 1 will be taken as an example to show the simulation steps. First, take the Laplace transform of (1), (2), and (3) with respect to t, which gives

$$\frac{\mathrm{d}^2 U(x, s)}{\mathrm{d}x^2} - (s^2 U(x, s) - su(x, 0) - u_t(x, 0)) = 0, \tag{9}$$

$$U(0, s) = 0, \tag{10}$$

$$\frac{\mathrm{d}U(1, s)}{\mathrm{d}x} = \left(d + \frac{ks}{s^2 + \omega^2}\right)(sU(1, s) - u(1, 0)) + \frac{s}{s^2 + \omega^2}, \tag{11}$$

where $U(x, s)$ is the Laplace transform of $u(x, t)$.

Substituting the initial conditions (6) and (7) into (9) and (11), we have

$$\frac{\mathrm{d}^2U(x,s)}{\mathrm{d}x^2} - s^2U(x,s) + s(-0.5\sin(0.5\pi x)) = 0, \tag{12}$$

$$\frac{\mathrm{d}U(1,s)}{\mathrm{d}x} = \left(d + \frac{ks}{s^2+\omega^2}\right)(sU(1,s)+0.5) + \frac{s}{s^2+\omega^2}. \tag{13}$$

Solving equations (12), (10), and (13) manually is daunting. However, we can take advantage of the already highly developed computer symbolic math, the Matlab Symbolic Math Toolbox, in our case. First, we will use `dsolve()`, which symbolically solves the ODE(s) and the boundary and/or initial condition(s). Although `dsolve()` is able to determine the arbitrary constants in the solution using the boundary and/or initial condition(s), we find that its ability is rather weak. So, we supply only (12) to `dsolve()` rather than supply (12), (10), and (13) together and get the following solution with two arbitrary constants, $C1$ and $C2$, in it. For some formulas too long to be printed within a line, the original Matlab format will be used.

$$U(x,s) = C2e^{-sx} + C1e^{sx} - 2\,\frac{s\sin(1/2\,\pi\,x)}{4\,s^2+\pi^2}. \tag{14}$$

Next, we differentiate $U(x,s)$ with respect to x to get the first-order derivative of $U(x,s)$ using the Matlab Symbolic Math Toolbox function `diff()`, which gives

$$\frac{dU(x,s)}{dx} = -\frac{C2\,s}{e^{sx}} + C1se^{sx} - \frac{s\cos(1/2\,\pi\,x)\,\pi}{4\,s^2+\pi^2}. \tag{15}$$

Substituting $U(x,s)$ (14) and its first-order derivative (15) into the boundary conditions (10) and (11), we have two boundary conditions, (16) and (17), with two undetermined constants, $C1$ and $C2$.

$$C1 + C2 = 0, \tag{16}$$

$$C1\,se^s - \frac{C2\,s}{e^s} + \left(1 + \frac{10s}{s^2+100}\right)\left(s\left(e^{-s}C2 + e^sC1 - \frac{2s}{4\,s^2+\pi^2}\right) + \frac{1}{2}\right) + \frac{s}{s^2+100} = 0. \tag{17}$$

Next, we solve the two algebraic equations (16) and (17) symbolically using the Matlab Symbolic Math Toolbox function `solve()`, which gives

$$C1 = -1/4 \tag{18}$$

$$\frac{e^s\left(12\,s\pi^2 + 100\,\pi^2 + 8\,s^3 + s^2\pi^2\right)}{s\,(4\,s^2+\pi^2)\left(-5\,s + s^2\,(e^s)^2 + 100\,\,(e^s)^2 + 5\,s\,(e^s)^2\right)},$$

$$C2 = 1/4 \tag{19}$$

$$\frac{e^s\left(12\,s\pi^2 + 100\,\pi^2 + 8\,s^3 + s^2\pi^2\right)}{s\,(4\,s^2+\pi^2)\left(-5\,s + s^2\,(e^s)^2 + 100\,\,(e^s)^2 + 5\,s\,(e^s)^2\right)}.$$

Now, we have actually obtained the explicit expression of $U(x,s)$, shown in (20).

```
U(x,s) = 1/4*exp(-s*x)*exp(s)*(12*s*pi^2+100*
pi^2+8*s^3+s^2*pi^2)/s/(4*s^2+pi^2)/(-5*s+s^2
*exp(s)^2+100*exp(s)^2+5*s*exp(s)^2)-1/4*exp
(s*x)*exp(s)*(12*s*pi^2+100*pi^2+8*s^3+s^2*
pi^2)/s/(4*s^2+pi^2)/(-5*s+s^2*exp(s)^2+100
*exp(s)^2+5*s*exp(s)^2)-2*s*sin(1/2*pi*x)/
(4*s^2+2778046668940015/281474976710656).
```
(20)

Although obtaining the explicit expression of $U(x,s)$ is just an intermediate step of this simulation method, it has been shown in another paper [13] that this is critical to designing more advanced boundary controllers because if we use a symbolic boundary controller $F(s)$ instead of equation (4) and divide $U(x,s)$ by $F(s)$ afterward, we obtain $U(x,s)/F(s)$, the transfer function of this control system. The role that transfer function plays in control system design cannot be overestimated. Using the complete numerical methods, such as FEM or FDM, it is impossible to obtain the transfer function. This is another advantage of this simulation method.

To obtain $u(x,t)$, we need to take the inverse Laplace transform of $U(x,s)$. We should *not* use the Matlab Symbolic Math Toolbox function `ilaplace()`, which takes the inverse Laplace transform symbolically, since for such a complicated expression of $U(x,s)$, the explicit expression of $u(x,t)$ is usually unavailable. However, we can make use of the numeric inverse Laplace transform. There are many numerical techniques available for the inverse Laplace transform [11]. Among the existing numeric inverse Laplace transform methods, we choose the method introduced in Brančik [12] for its accuracy and fastness. The method uses fast Fourier transform (FFT) first and then speeds up the convergence of infinite complex Fourier series by the ϵ-algorithm. Interested readers can refer to Brančik [14, 15] for detailed theory and Brančik [12] for implementation and readily available Matlab code.

At this point, we have actually finished the time domain simulation. In what follows, we present some simulation results for both case 1 and case 2.

The plots of tip displacement in cases 1 and 2 are shown in Figures 1 and 2, respectively. It shows clearly that the dynamic controller is better than the static controller in

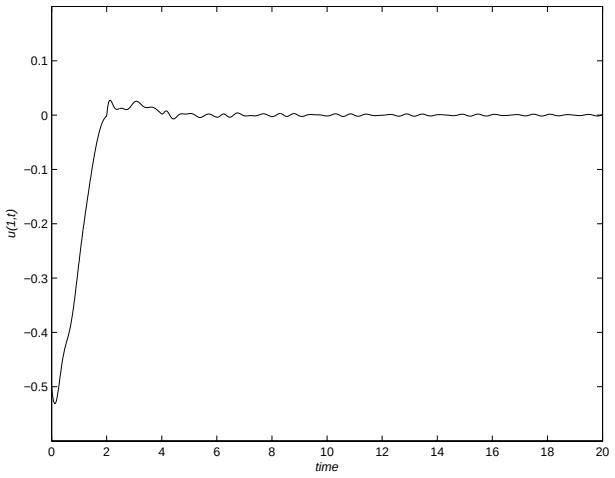

Figure 1. Tip displacement for case 1

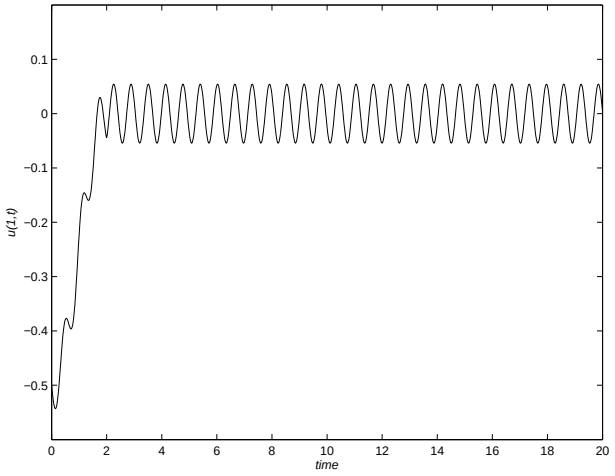

Figure 2. Tip displacement for case 2

rejecting the noise. The two plots are identical to the simulation results reported in Morgül [4].

It is also very easy to show the displacement of the whole string, which is shown in Figures 3 and 4 for cases 1 and 2, respectively.

3. Two Application Examples

In this section, we use two application example to show that this simulation method is able to simulate difficult problems.

3.1 Boundary Control of a Fractional Wave Equation Via a fractional Order Boundary Controller

Fractional diffusion and wave equations are obtained from the classical diffusion and wave equations by replacing the

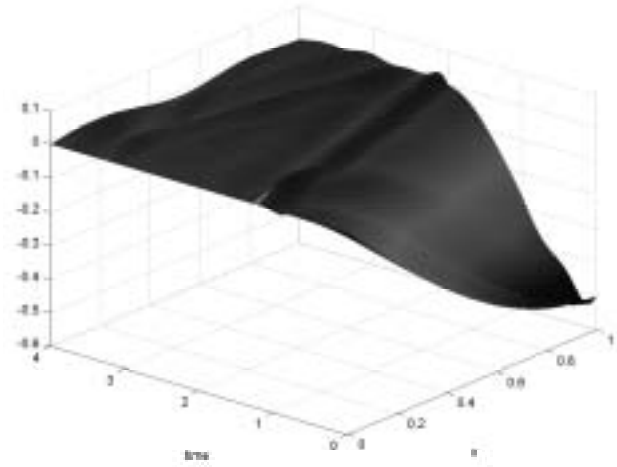

Figure 3. Displacement of the whole string for case 1

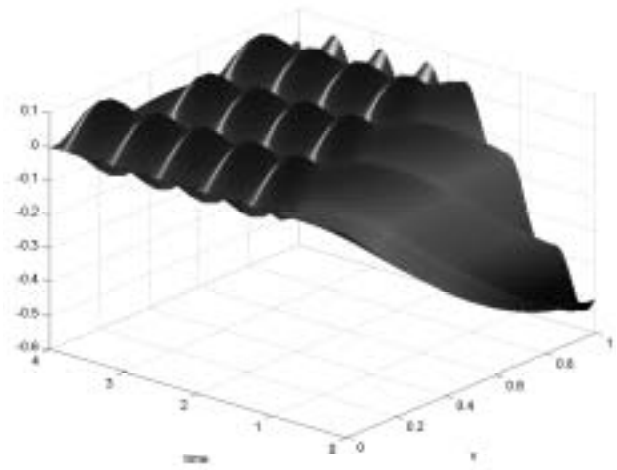

Figure 4. Displacement of the whole string for case 2

first- and second-order time derivative term by a fractional derivative of an order satisfying $0 < \alpha \leq 1$ and $1 < \alpha \leq 2$, respectively. Since many of the universal phenomena can be modeled accurately using the fractional diffusion and wave equations [16], there has been a growing interest in investigating the solutions and properties of these evolution equations [17-20]. In this section, we simulate the stabilization of a cable governed by the fractional wave equation using a fractional order boundary controller. Interested readers can refer to Liang et al. [21] for more information.

Consider a cable made with special smart materials governed by the fractional wave equation, fixed at one end and stabilized by a boundary controller at the other end. The

system can be represented by

$$\frac{\partial^\alpha u}{\partial t^\alpha} = \frac{\partial^2 u}{\partial x^2}, \quad 1 < \alpha \le 2, \quad x \in [0,1], \quad t \ge 0 \tag{21}$$

$$u(0,t) = 0, \tag{22}$$

$$u_x(1,t) = f(t), \tag{23}$$

$$u(x,0) = u_0(x), \tag{24}$$

$$u_t(x,0) = v_0(x), \tag{25}$$

where $u(x,t)$ is the displacement of the cable at x and t, $f(t)$ is the boundary control force at the free end of the cable, and $u_0(x)$ and $v_0(x)$ are the initial conditions of displacement and velocity, respectively.

We adopt the Caputo definition for the fractional derivative of order α of function $f(t)$ [22, 23]:

$$\frac{d^\alpha f(t)}{dt^\alpha} = \frac{1}{\Gamma(\alpha - n)} \int_0^t \frac{f^{(n)}(\tau) d\tau}{(t-\tau)^{\alpha+1-n}},$$

$$(n-1 < \alpha \le n). \tag{26}$$

Based on the definition of (26), the Laplace transform of the fractional derivative is

$$\mathcal{L}\left\{\frac{\mathrm{d}^\alpha f(t)}{\mathrm{d}t^\alpha}\right\} = s^\alpha F(s) - \sum_{k=0}^{n-1} f^k(0^+) s^{\alpha-1-k}. \tag{27}$$

The initial conditions are assumed to be

$$u_0(x) = -\sin(0.5\pi x), \quad v_0(x) = 0. \tag{28}$$

We will simulate the system response with a boundary controller in the following format:

$$f(t) = -k \frac{\mathrm{d}^\mu u(1,t)}{\mathrm{d}t^\mu}, \quad 0 < \mu \le 1 \tag{29}$$

where k is the controller gain, and μ is the order of the fractional derivative of the displacement at the free end of the cable.

When $\mu = 1$, the controller (29) is called an integer order controller. When $0 < \mu < 1$, the controller (29) is called a fractional order controller. Boundary control of fractional wave equations via a fractional order boundary controller is a new research area. Results of both analytical and simulation studies are still very limited.

Following the same procedures described in section 2, the explicit expression of $U(x,s)$ is obtained with the symbolic expression listed in the appendix.

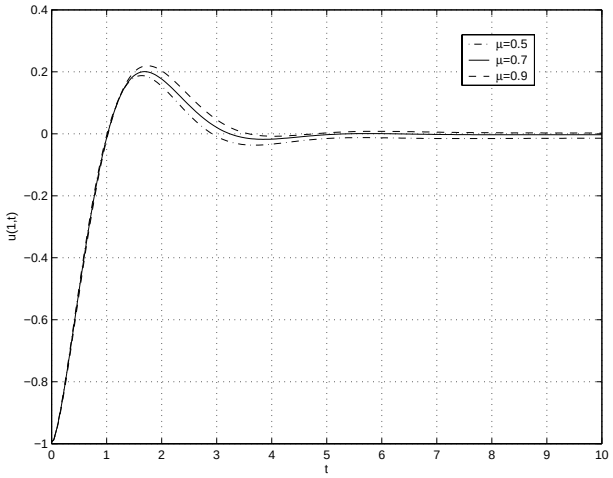

Figure 5. Displacement of the free end, α = 1.5

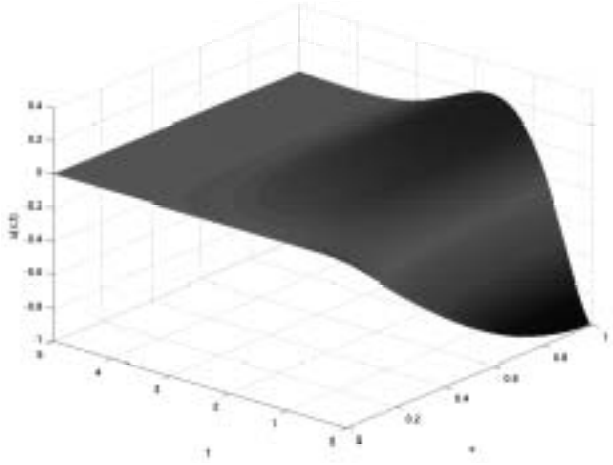

Figure 6. Displacement of the whole cable, μ = 0.7

For $\alpha = 1.5$, $k = 0.1$, and $\mu = 0.5, 0.7, 0.9$, the displacement of the free end is shown in Figure 5. The displacement of the whole cable for $\alpha = 1.5$, $k = 0.1$, and $\mu = 0.7$ is shown in Figure 6.

We can see that all simulated controllers stabilize the system.

3.2 Boundary Control of the Beam Equation with Time Delay Using the Smith Predictor

Consider a flexible beam clamped at one end and free at the other end. We denote the displacement of the beam by $u(x,t)$ at $x \in [0,1]$ and $t \ge 0$. The beam is controlled by a boundary control force at the free end. The equations are given as follows:

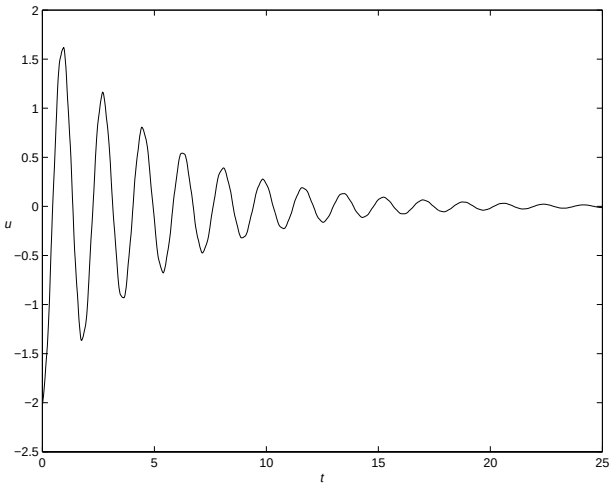

Figure 7. Displacement of the free end

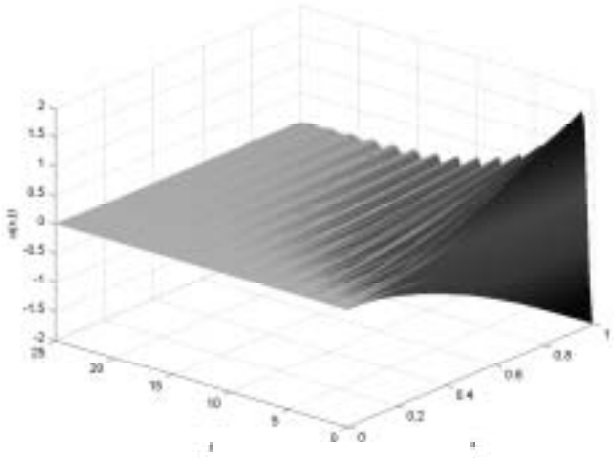

Figure 8. Displacement of the whole beam

$$u_{tt} + u_{xxxx} = 0, \tag{30}$$

$$u(0,t) = 0, \tag{31}$$

$$u_x(0,t) = 0, \tag{32}$$

$$u_{xx}(1,t) = 0, \tag{33}$$

$$u_{xxx}(1,t) = f(t), \tag{34}$$

where $f(t)$ is the boundary control force applied at the free end of the beam.

The initial conditions are chosen as

$$u(x,0) = x^3 - 3x^2, \tag{35}$$

$$u_t(x,0) = 0, \tag{36}$$

where (35) is a typical displacement profile when the beam is subject to a static force $f = -1$ at the free end [24].

It is well known that the following controller stabilizes the displacement of the beam [25]:

$$f(t) = ku_t(1,t), \tag{37}$$

where $k > 0$ is the constant gain.

The simulation results of the controller (37) are shown in Figures 7 and 8 for displacement of the free end and the whole beam, respectively.

However, in Datko, Lagnese, and Polis [26] and Datko [27], it was shown that boundary control of the wave equation and the beam equation becomes unstable when an arbitrary small time delay is introduced into the feedback loop, that is,

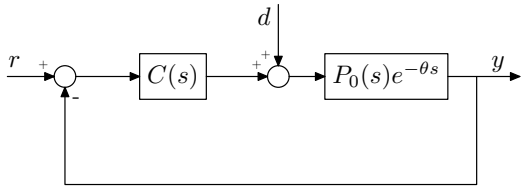

Figure 9. A feedback control system with a time delay

$$f(t) = ku_t(1, t-\theta), \tag{38}$$

where θ is the time delay. The diagram of this feedback control system is shown in Figure 9, where d denotes the disturbance, $P_0(s)$ is the transfer function of the system to be controlled, and $C(s)$ is the transfer function of the controller.

We will simulate this phenomenon to see how it happens. To understand the reason for the instability, it helps to plot both the tip velocity and the tip displacement. It is easy to calculate the velocity profile at any point. After the expression of $U(x,s)$ is obtained, $sU(x,s)$ is the Laplace transform of the velocity at any point x.

The simulation results are shown in Figures 10 and 11. We can see that the controller works at the beginning, driving the tip end to the zero position. However, the frequency of the vibration is increasing over time. When the frequency is high enough, the time delay causes the control force to be in phase rather than out of phase with the tip velocity, thus making the system unstable. Although this phenomenon was discovered more than 10 years ago, no simulation had

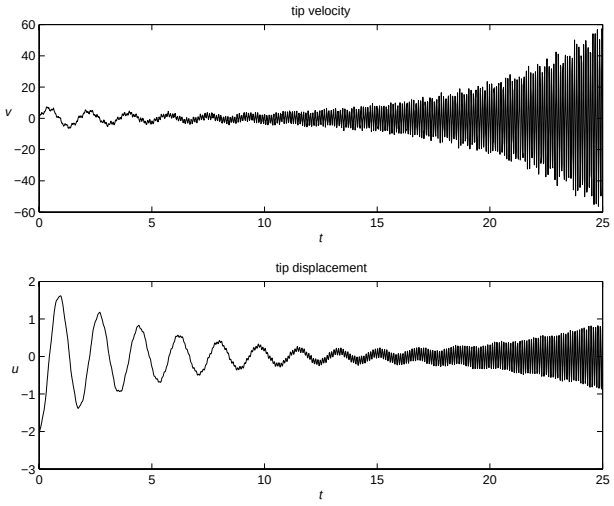

Figure 10. Tip velocity and displacement, $\theta = 0.05$

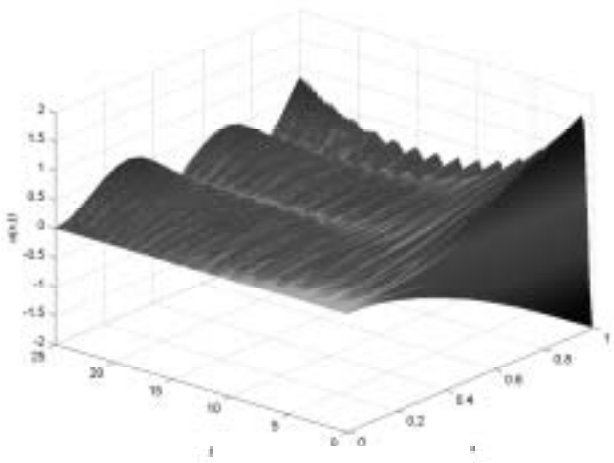

Figure 11. Displacement of the whole beam, $\theta = 0.05$

ever been carried out before, and no solution has been proposed so far. We will design a boundary controller based on the transfer function obtained in the intermediate steps of the simulation and simulate the response of this controller. Interested readers can refer to Liang, Chen, and Guo [13] for more details.

The Smith predictor is probably the most famous method for the control of systems with time delays [28]. The classical configuration of a system containing a Smith predictor is depicted in Figure 12, where $P(s) = P_0(s)e^{-\theta s}$. $\hat{P}_0(s)$ and $\hat{P}(s)$ are nominal models of $P_0(s)$ and $P(s)$, respectively. The block $C(s)$, combined with the block $\hat{P}_0(s) - \hat{P}(s)$, is called the Smith predictor.

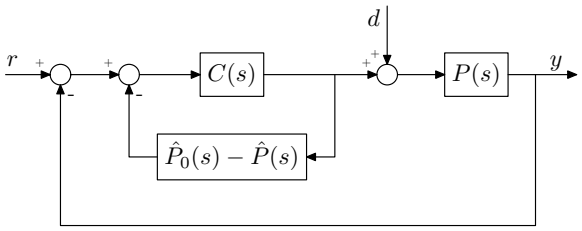

Figure 12. The Smith predictor

One of the key factors in applying a Smith predictor is the knowledge of $\hat{P}_0(s)$, the transfer function of the plant to be controlled. Although this is a difficult task for a fourth-order PDE, using our simulation method, $P_0(s)$ can be obtained very easily as follows:

$$P_0(s) = \tag{39}$$

$$\frac{(1-i)\sqrt[4]{-s^2}\left(1+ie^{2i\sqrt[4]{-s^2}}-ie^{-2\sqrt[4]{-s^2}}-e^{(-2+2i)\sqrt[4]{-s^2}}\right)}{s\left(e^{2i\sqrt[4]{-s^2}}+e^{(-2+2i)\sqrt[4]{-s^2}}+1+e^{-2\sqrt[4]{-s^2}}+4e^{(-1+i)\sqrt[4]{-s^2}}\right)}.$$

With (39) embedded in the transfer function of the controller, the transfer function of the whole system becomes prohibitively complicated for manual derivation and can only be obtained with the help of computers. The final $U(x, s)$ is a 30, 680-character long formula expressed in Matlab notation. The simulation results are shown in Figures 13 and 14. We can see that the system becomes stable now, although the beam vibrates with nondecreasing magnitude, which can be removed almost completely by using modified Smith predictors [13].

4. Concluding Remarks

A hybrid symbolic and numerical method based on the Matlab Symbolic Math Toolbox has been developed in this article to simulate typical boundary control problems. For illustration and validation, three different boundary control problems are simulated using the proposed method. The proposed simulation method can be applied to a wide range of boundary control problems. Furthermore, the transfer function can be calculated in the intermediate steps of the simulation, which can be used to design some more advanced boundary controllers. We hope that this method is helpful for researchers to study the boundary control problems with easy simulation explorations in the future.

5. Appendix

Since the expression of $U(x, s)$ in section 3.1 is too long to be printed within a line, it is copied here in its original Matlab format. The purpose of this appendix is to make

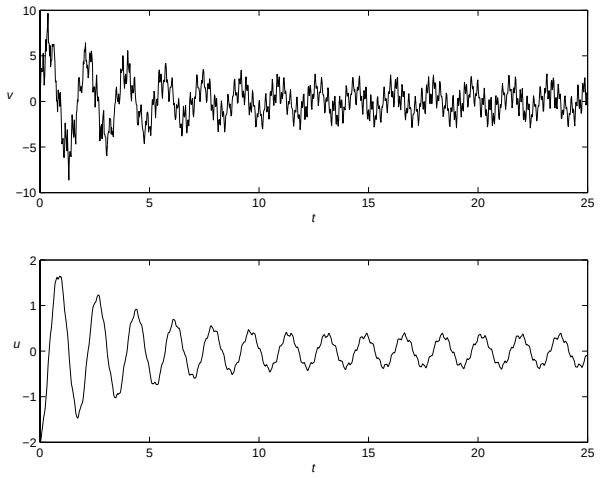

Figure 13. Tip velocity and displacement with the Smith predictor added

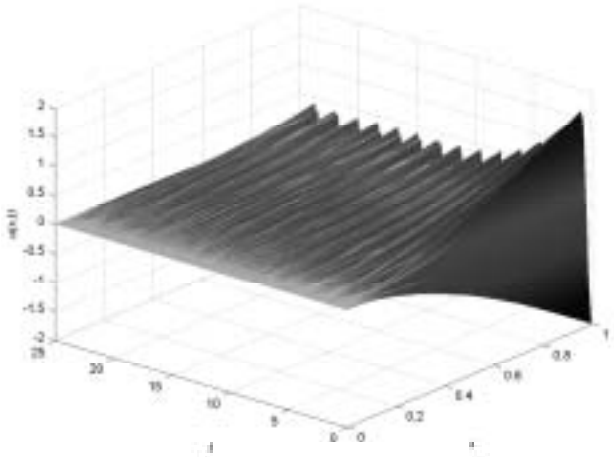

Figure 14. Displacement of the whole beam with the Smith predictor added

the results reported in this article more easily reproducible by others. The $U(x, s)$ in section 3.2 is a 30, 680-character long formula, which cannot even be printed within a page, so it is omitted.

```
U(x,s) = -(1125899906842624*sˆ(3/2*alfa)*sin
(1/2*pi*x)*exp(2*sˆ(1/2*alfa))+1125899906842624
*sˆ(3/2*alfa)*sin(1/2*pi*x)+1 125899906842624*
sˆ(mu+alfa)*sin(1/2*pi*x)*k*exp(2*sˆ(1/2*alf a))
-1125899906842624*sˆ(mu+alfa)*sin(1/2*pi*x)*k+
27780466689 40015*exp(sˆ(1/2*alfa)*x+sˆ(1/2*
alfa))*k*sˆmu-27780466689400 15*exp(-sˆ(1/2*
alfa)*x+sˆ(1/2*alfa))*k*sˆmu)/s/(-k*sˆmu+k*sˆmu
*exp(2*sˆ(1/2*alfa))+sˆ(1/2*alfa)+sˆ(1/2*alfa)*
exp(2*sˆ(1/2*alfa)))/(1125899906842624*sˆalfa+
2778046668940015).
```

6. References

[1] Morgül, Ö. 2002. An exponential stability result for the wave equation. *Automatica* 38:731-5.

[2] Morgül, Ö. 2001. Stabilization and disturbance rejection for the beam equation. *IEEE Transactions on Automatic Control* 46 (12): 1913-8.

[3] Conrad, F., and Ö. Morgül. 1998. On the stability of a flexible beam with a tip mass. *SIAM Journal on Control and Optimization* 36 (6): 1962-86.

[4] Morgül, Ö. 1998. Stabilization and disturbance rejection for the wave equation. *IEEE Transactions on Automatic Control* 43 (1): 89-95.

[5] Guo, B.-Z. 2001. Riesz basis approach to the stabilization of a flexible beam with a tip mass. *SIAM Journal on Control and Optimization* 39 (6): 1736-47.

[6] Guo, B.-Z. 2002. Riesz basis property and exponential stability of controlled Euler-Bernoulli beam equations with variable coefficients. *SIAM Journal on Control and Optimization* 40 (6): 1905-23.

[7] *Partial differential equation (PDE) tool box user's guide*. 2002. Natick, MA: The Mathworks, Inc.

[8] Mitchell, A. R., and D. Griffiths. 1980. *The finite difference method in partial differential equations*. New York: John Wiley.

[9] Jeffrey, A. 2002. *Advanced engineering mathematics*. New York: Harcourt/Academic Press.

[10] *Symbolic math toolbox user's guide*. 2002. Natick, MA: The Mathworks, Inc.

[11] Duffy, D. G. 1993. On the numerical inversion of Laplace transforms: Comparison of three new methods on characteristic problems from applications. *ACM Transactions on Mathematical Software* 19:333-59.

[12] Brančík, L. 1999. Programs for fast numerical inversion of Laplace transforms in Matlab language enviroment. In *Konference MATLAB '99 ZCU Plzen*, pp. 27-39.

[13] Liang, J., Y. Chen, and B.-Z. Guo. 2003. A new boundary control method for beam equation with delayed boundary measurement using modified Smith predictors. In *Proceedings of the 41st IEEE Conference on Decision and Control*, December, Maui, HI, pp. 809-14.

[14] Brančík, L. 1998. The fast computing method of numerical inversion of Laplace transforms using FFT algorithm. In *Proceedings of the 5th EDS 98 International Conference*, June, Brno, Czech Republic, pp. 97-100.

[15] Brančík, L. 1999. An improvement of FFT-based numerical ILT procedure by application of ϵ-algorithm. In *Sborník přednášek moderní směry výuky elektrotechniky a elektroniky STO-7*, Brno, Září, pp. 196-9.

[16] Nigmatullin, R. R. 1986. Realization of the generalized transfer equation in a medium with fractal geometry. *Physical Status Solidi B* 133:425-30.

[17] Wyss, W. 1986. The fractional diffusion equation. *Journal of Mathematical Physics* 27 (11): 2782-5.

[18] Schneider, W. R., and W. Wyss. 1989. Fractional diffusion and wave equation. *Journal of Mathematical Physics* 30 (1): 134-44.

[19] Mainardi, F., and P. Paradisi. 1997. A model of diffusive waves in viscoelasticity based on fractional calculus. In *Proceedings of the 36th IEEE Conference on Decision and Control*, San Diego, CA.

[20] Agrawal, O. P. 2002. Solution for a fractional diffusion-wave equation defined in a bounded domain. *Nonlinear Dynamics* 29:145-55.
[21] Liang, J., Y. Chen, B. M. Vinagre, and I. Podlubny. 2004. Boundary stabilization of a fractional wave equation via a fractional order boundary controller. In *The First IFAC Symposium on Fractional Derivatives and Applications (FDA'04)*, July, Bordeaux, France.
[22] Caputo, M. 1967. Linear models of dissipation whose Q is almost frequency independent—II. *Geophysics Journal of the Royal Astronomical Society* 13:529-39.
[23] Podlubny, I. 1999. *Fractional differential equations*. New York: Academic Press.
[24] Kassimali, A. 1993. *Structural analysis (Pws-Kent series in Engineering)*. Boston, MA: PWS Publishing.
[25] Chen, G., M. C. Delfour, A. M. Krall, and G. Payre. 1987. Modelling, stabilization and control of serially connected beams. *SIAM Journal on Control and Optimization* 25:526-46.
[26] Datko, R., J. Lagnese, and M. P. Polis. 1986. An example on the effect of time delays in boundary feedback stabilization of wave equations. *SIAM Journal on Control and Optimization* 24:152-6.
[27] Datko, R. 1993. Two examples of ill-posedness with respect to small time delays in stabilized elastic systems. *IEEE Transactions on Automatic Control* 38 (1): 163-6.
[28] Levine, W., ed. 1996. *The control handbook*. Boca Raton, FL: CRC Press.

Jinsong Liang *is with the Center for Self-Organizing and Intelligent Systems (CSOIS) in the Department of Electrical and Computer Engineering at Utah State University, Logan.*

YangQuan Chen *is with the Center for Self-Organizing and Intelligent Systems (CSOIS) in the Department of Electrical and Computer Engineering at Utah State University, Logan.*

Bao-Zhu Guo *is with the Institute of Systems Sciences at the Academy of Mathematics and System Sciences, Academia Sinica, in Beijing, the People's Republic of China.*