Machine Dynamics Research
201x, Vol. xx, No XX, XX-XX

Adaptive Stabilization of Partially Damaged Vibrating Structures

Maciej Wasilewski*, Dominik Pisarski†, and Czesław I. Bajer‡
Institute of Fundamental Technological Research Polish Academy of Sciences

Abstract
In this paper an online adaptive continuous-time control algorithm will be studied in the vibration control problem. The examined algorithm is a Reinforcement Learning based scheme able to adapt to the changing system's dynamics and providing control converging to the optimal control. Firstly, a brief description of the algorithm is provided. Then, the algorithm is studied by the numeric simulation. The controlled model is a simple conjugate oscillator with sudden change of its rigidity. The effectiveness of the adaptation of the algorithm is compared to the simulation results of controlling the same object by the traditional Linear Quadratic Regulator. Because of the lack of constraints for a system size or its linearity, this algorithm is suitable for optimal stabilization of more complex vibrating structures.

Keywords: vibration control, adaptive control, optimal control, policy iterations, Hamilton-Jacobi-Bellman equation.

1 Introduction

The problem of steering objects subjected to vibrations is present in many branches of the modern engineering. Bridges vibrate under moving vehicles and side winds, this type of motion can result in the damage of a construction. The need of achieving a high spatial accuracy of robotic manipulators is often unreachable because of robotic arms vibration induced by moving relatively large masses.

There exists a rich literature concerning the vibration control, which can be divided into the development of a specific hardware (i.e., actuators and sensors) and control algorithms. The ways of steering vibrating systems consist of active, passive and semi-active types of control. Brief description of this three types of control systems was provided in (Symans and Constantinou, 1999).

Passive control systems are described as systems which do not need a power supply to operate. The control in passive systems is developed by a utilization of a

*mwasil@ippt.pan.pl
†dpisar@ippt.pan.pl
‡cbajer@ippt.pan.pl

structure's motion. Overview of passive control systems was provided in (Soong and Constantinou, 1994). Active systems require a large power supply to operate force actuators (e.g., electrohydraulic or electromechanical), which attenuate system's vibrations. Review of active systems was presented in (Soong and Constantinou, 1994) and (Fujino et al., 1996). Semi-active control systems may be defined as systems requiring a small external power source and which control is developed based on feedback from sensor and/or response of the structure. Overview of a semi-active control was provided in (Spencer Jr, 1996).

A semi-active attenuation of vibrations in structures can be performed by using piezoelectric sensors and actuators. Piezoelectric elements are an attractive choice for vibration control because of their low mass, high bandwidth and low cost (Peng et al., 2005). Because of the reversibility of a piezoelectric effect, piezoelectric elements work as an actuators and sensors as well. The possibility of the efficient control of the vibrating plate by the thin layer of piezoelectric sensors and actuators was proposed in (Tzou and Tseng, 1990) and (Hu and Ng, 2005). In (Youn et al., 2000) the authors used the piezoelectric actuator to control vibrations of composite beams. The actuator placement optimization for a vibrating plate control was presented in (Peng et al., 2005).

Magnetorheological (MR) dampers are used in the field of a vibration suppression as well. In (Dyke et al., 1996) the authors developed the model of the MR damper and studied its effectiveness in a control of a three-story building. Structural vibration control of a building utilizing MR damper was also presented in (Sakai et al., 2003). In (Pisarski, 2011) and (Pisarski and Bajer, 2010) vibration control of 1D continuum under a travelling load using MR dampers was presented.

Apart from hardware development, algorithms for vibration control are also in great research attention. The Input Shaping scheme is utilized in (Hillsley and Yurkovich, 1991), (Tzes and Yurkovich, 1993), (Mohamed et al., 2006) and (Singhose, 2009). The optimal controllers for vibrational systems were used in (Li et al., 1994), (Kucuk et al., 2013), (Pisarski and Bajer, 2010) and (Pisarski, 2011). As well as open-loop algorithms, the field of vibration suppression utilizes feedback controllers. Close-loop robust controllers based on H_∞ control are presented in (Kar et al., 2000b) and (Kar et al., 2000a).

An additional impediment which may occur in a control of systems exposed to a vibration is a change of its dynamics. Robots may move loads of varying unknown masses, a structure of bridges changes its shape depending on a temperature. There also may occur sudden damages in systems, e.g., shot aircraft acts differently under the control and has different air resistance, a bridge after breaking up of one of suspension wires has different dynamics. An effective controller designed to work in vibrational systems has to have an adaptive property.

One type of adaptive optimal controller is a Model Predictive Controller. This controller solves an optimal control problem on each iteration by predicting the system response on finite horizon. The efficiency of the algorithm is achieved by the prediction of the future states of the controlled system. The knowledge of system dynamics is crucial for this controller. The adaptive property is achieved for Model Predictive Controller by linking with the system identification algorithm.

The aim of this paper is to study other type of controller suitable for problems of vibrations in mechanics. The algorithm, known in literature as Generalized Policy Iteration (GPI) was firstly presented in (Vrabie and Lewis, 2009). This algorithm originates from GPI algorithms based on the Reinforcement Learning (RL), the branch of machine learning science. Roots of Reinforcement Learning are based on a biological observation of animals in their natural environment. Reinforcement Learning was firstly introduced in (Sutton et al., 1992). This technique was basically used for finding optimal control for Markovian discrete systems. The continuous version of this scheme was given in (Baird, 1994). The basic idea of RL is that successful control should be remembered and more likely used (reinforced) a second time.

The algorithm provides optimal control and learns system dynamics in indirect way, both actions are executed in parallel. The GPI learns optimal control policy by interacting with the system. This type of acting is characteristic for dual control methods.

The GPI algorithm shows its main advantage in the presence of the change of the dynamics, e. g. mentioned above. It detects this change and after fulfilling few conditions it provides control converging to optimal control. In contrast to LQR and MPC regulators, the GPI has no need to know the system dynamics. General form of the algorithm works for nonlinear problem, what also distinguish GPI form LQR and MPC.

In the Section 2. the problem definition in mathematical manner is formulated. The derivation of the GPI algorithm is presented in the Section 3. The next section presents the way of neural network adaptation of this algorithm is made. The Section 5. presents simulation results of control generated by the GPI algorithm in comparison to the LQR case.

2 Problem formulation

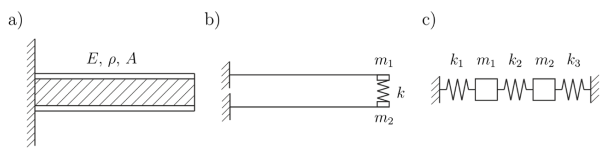

Fig 1. Examples of considered structures: a) sandwich beam, b) simplified model, c) model reduced to control analysis.

Layered structures are commonly applied as elements in complex structures. Thin beams or plates with a filling material (Fig. 1 a) exhibiting properties that can be controlled have greater strength to dynamic load than structures with constant parameter filling material. Unfortunately, the elaboration of the control strategy is complex and we have to reduce the continuous structure to more simple system with a single element being controlled (Fig. 1 b). Moreover, we can reduce our considerations to the first mode that in practice dominates in vibrations. In such a case the spring-mass system can sufficiently reproduce the continuous structure. This reduced scheme of the structure with a suddenly varying system parameter, for example sudden damage of structure element, will be assumed to our analysis (Fig. 1 c). The taken action can be performed in various ways: by the damping control, the force control or the stiffness control. In our work we assume the force action.

Throughout this work we will consider a controlled vibrating system defined by the linear dynamic equation and initial value:

$$\dot{x} = Ax + Bu, \qquad x(0) = x_0 \tag{1}$$

with $x \in \mathbb{R}^n$ is the state of the system, $A \in \mathbb{R}^{n \times n}$ is the system matrix, $B \in \mathbb{R}^{n \times m}$ is the control input matrix, x_0 is the initial state and $u \in U \subset \mathbb{R}^m$ is the control input. The set of admissible control U conforms to physical constraints of the control device, e.g., extreme forces possible to execute by actuators. We assume that the system is controllable on $\Omega \subseteq \mathbb{R}^n$, i.e., the system can be steered from any initial state to any other final state in finite time interval by the control $u(t) \in U$. Recall from (Kalman et al., 1960) that the linear system (1) given by the pair (A, B) is controllable iff $rank\left(\begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix}\right) = n$.

Classical approaches use the control that minimize integral objective of the form:

$$V(x(0)) = \int_0^\infty \left(x^T(\tau)Qx(\tau) + u^T(\tau)Ru(\tau)\right)d\tau \tag{2}$$

where $Q \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{m \times m}$ are positive semi-definite and positive definite matrices, respectively. Such form of the objective means that both system energy (related to quadratic form of the state) and control (often referred to the energy supplied to the system) are the aim of the minimization. The R matrix is usually selected by a trial and error to provide that the optimal control belongs to the admissible set.

The control policy minimizing quadratic objective (2) for linear dynamics is referred to as linear-quadratic regulator (LQR). The optimal control is given in the state feedback form

$$u = -Kx, \qquad K \in \mathbb{R}^{m \times n} \tag{3}$$

and for the assumed infinite time horizon problem (2), the K matrix is time invariant and given by

$$K = R^{-1}B^T P \tag{4}$$

Here P is the solution of the algebraic Riccati equation

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{5}$$

The described system controlled by the LQR is governed by the closed loop dynamics of the form

$$\dot{x} = Ax + Bu = (A - BK)x \tag{6}$$

Naturally from (5) the feedback matrix K can be computed only if the system matrices, A and B are fully determined.

Now let us assume that the system controlled by the LQ regulator changes its dynamics at time $t_1 > 0$, e.g., because of mass added to system, fatigue, loss of rigidity, etc. The change of the system can be represented by the unknown change of the system matrix from A to . Assuming that the state of the system at time t_1 is x_1, evolution of the modified system is now governed by:

$$\dot{x} = (A + \Delta A - BK)x,\ x(t_1) = x_1 \tag{7}$$

It can be shown that the feedback matrix K calculated by (4) for the system (1) does not provide the optimal control for the system (7). It may also occur that the control given by (3) destabilizes the system (7). The sufficient condition for the system to be unstable is that at least one eigenvalue of the matrix has a real part greater than 0. Such situation will be studied in the Section 5.

The aim of this work is to study the control algorithm minimizing (2) without any knowledge of the system matrix disturbance .

The algorithm was firstly presented (Vrabie and Lewis, 2009). In the next section, we will recall the crucial results.

3 The GPI algorithm

In this section we give the short derivation of the algorithm of adaptive optimal control for linear mechanical systems. The algorithm was firstly formulated in (Al-Tamimi and Lewis, 2007) for discrete-time systems, where the convergence proof is also presented. The continuous-time version is presented in (Vrabie and Lewis, 2009).

3.1 Preliminaries

Let us concern a dynamic system defined by (1) and an objective to minimize by (2). The cost-to-go associated with the control input u at time t is defined by:

$$V^u(x(t)) = \int_t^\infty \left(x^T(\tau)Qx(\tau) + u^T(\tau)Ru(\tau)\right)d\tau = \int_t^\infty F(x(\tau), u(\tau))\,d\tau \tag{8}$$

The value of (8) is associated with the value of the objective which will be obtained for every future moments starting from the t.

Definition 1. (Beard et al., 1997) Stabilizing policy μ is such policy that control $\mu(x)$ is stabilizing with respect to (8) on Ω, denoted by $\mu \in \Psi(\Omega)$, if $\mu(x)$ is continuous on Ω, $\mu(0) = 0$, $\mu(x)$ stabilizes (1) on Ω and $V^\mu(x_0)$ is finite $\forall x_0 \in \Omega$.

The objective function V associated with any admissible policy $\mu \in \Psi$ is

$$V^{\mu}(x(t)) = \int_{t}^{\infty} \left(x^T(\tau) Q x(\tau) + \mu(x(\tau))^T(\tau) R \mu(x(\tau)) \right) d\tau \tag{9}$$

By differentiating (9) by time we have

$$0 = x^T(t) Q x(t) + \mu(x(t))^T(\tau) R \mu(x(t)) + (\nabla \mathrm{V}_{\mathrm{x}}^{\mu})^T (A + B\mu(x)),\ V^{\mu}(0) = 0 \tag{10}$$

It is important to see that

$$V^{\mu}(x(t)) = \int_{t}^{t+T} \left(x^T(\tau) Q x(\tau) + \mu(x(\tau))^T(\tau) R \mu(x(\tau)) \right) d\tau + V^{\mu}(x(t+T)) \tag{11}$$

The optimal control problem is then formulated (Vrabie and Lewis, 2009): Given the continuous-time system (1), the set $u \in \Psi(\Omega)$ of admissible control policies, and the infinite horizon objective functional (2), find an admissible control policy such that the objective index (2) associated with the system (1) is minimized.

By the definition of the Hamiltonian:

$$H(x,\ u,\ \nabla V_x) = x^T(t) Q x(t) + \mu(x(t))^T(\tau) R \mu(x(t)) + (\nabla V_x)^T (A + B\mu(x)) \tag{12}$$

the optimal objective function satisfies the Hamilton-Jacobi-Bellman (HJB) equation:

$$0 = \min_{u \in \Psi(\Omega)} H(x,\ u, \nabla V_x^*) \tag{13}$$

Assuming that the minimum of (13) exist and is unique then the optimal control policy is given by:

$$u = \mu(x) = -\frac{1}{2} R^{-1} B^T \nabla V_x^* \tag{14}$$

After inserting (14) to (9)

$$0 = x^T(t) Q x(t) + \nabla V_x^{*^T} A - \frac{1}{4} \nabla V_x^{*^T} B R^{-1} B^T \nabla V_x^*,\ V^*(0) = 0 \tag{15}$$

one can see that (15) is equivalent to the Riccati equation (5) and can be solved to obtain the optimal control. Analogically to the LQR case, solving (15) also requires complete information about the system dynamics.

3.2 Iterative solution of HJB equation

In this section the exact form of the iterative algorithm is presented. The convergence proof is provided both in (Al-Tamimi and Lewis, 2007) and (Vrabie and Lewis, 2009). If $\mu^{(0)}(x(t)) \in \Psi(\Omega)$ and $T > 0$ such that $x(t),\ x(t+T) \in \Omega$, then the iteration between: 1. the value function evaluation:

$$V^{\mu^{(i)}} = \int_{t}^{t+T} \left(x^T(\tau) Q x(\tau) + \mu^{(i)}(x(\tau))^T(\tau) R \mu^{(i)}(x(\tau)) \right) d\tau + V^{\mu(i)}(x(t+T)) \tag{16}$$

and

2. the policy improvement:

$$\mu^{(i+1)} = \arg \min_{u\in\Psi(\Omega)} \left[H\left(x,\, u,\, \nabla V^{\mu^{(i)}}\right)\right] = -\frac{1}{2}R^{-1}B^T\nabla V_x^{\mu^{(i)}} \tag{17}$$

converges to the optimal control policy $\mu^* \in \Psi(\Omega)$ with the corresponding objective $V^*(x_0) = \min\limits_{\mu}\left(\int\limits_0^\infty \left(x^T(\tau)\,Qx(\tau) + \mu(x(\tau))^T(\tau)\,R\mu(x(\tau))\right)d\tau\right)$.

As one can see, the need of knowing the system dynamics for this algorithm reduces to the knowledge of the matrix B. This allows to use this algorithm in situation where the state matrix A is unknown and changing in time.

It should be emphasized that this algorithm can be derived for more general case with nonlinear dynamics:

$$\dot{x} = f(x) + g(x)u,\; x(0) = x_0,\; f(0) = 0 \tag{18}$$

but in this article only the linear case is concerned.

Application of the algorithm needs to use any approximation structure for (16). The most common choice is a neural network structure because of its simplicity and effectiveness.

4 The neural network adaptation

In order to solve (16) one needs to use any approximating structure for the value function $V^{\mu^{(i)}}(x)$. In this algorithm the simple one-layered neural network will be used, but it should be emphasized that any approximating structure that allows to calculate the gradient of the function can be used.

4.1 The neural network topology

In general, a neural network can have multiple layers with complex topology and activation functions can be non-linear. Below, we will show that for the special case of linear quadratic problems the neural network can be much simpler.

Let us consider that the value function $V^{\mu^{(i)}}$can be approximated for $x \in \Omega$ by one-layered network:

$$V^{\mu^{(i)}} = \sum_{j=1}^{L} w_j^{\mu^{(i)}}\phi_j(x) = \left(W^{\mu^{(i)}}\right)^T\varphi(x) \tag{19}$$

and

$$\nabla V_x^{\mu^{(i)}} = \left(\left(W^{\mu^{(i)}}\right)^T\nabla\varphi_x(x)\right)^T \tag{20}$$

where $W^{\mu^{(i)}} \in \mathbb{R}^L$ is a vector of constant (constant in each iteration) weights, $\varphi(x) \in \mathbb{R}^L$ is a vector of activation functions and L denotes number of neurons in layer.

For linear-quadratic problems with infinite horizon it is widely known that $V^*(x) = x^T P x$, where P is the solution of (5) (Kalman et al., 1960). If $V^*(x)$ is a quadratic form of vector $x = \left[\begin{array}{cccc} x_1 & x_2 & \cdots & x_n \end{array}\right]^T$, then it is the linear combination of functions of the form:

$$\phi_s(x) = x_i^d x_j^{2-d},\ d \in \mathbb{N},\ d \leqslant 2 \tag{21}$$

so it is sufficient that the activation functions of the neural network approximation will have form (19), e.g., x_1^2, $x_1 x_2$, $x_2 x_4$, etc. It is easy to show that number of neurons is equal to $\mathrm{L} = \overline{\mathrm{C}_\mathrm{n}}^2 = \left(\begin{array}{c} 1+\mathrm{n} \\ 2 \end{array}\right)$, where $\overline{C_n}^k$ denotes the number of k-combinations with repetition of the set of n objects.

4.2 The teaching algorithm

It is easy to conclude from (16) that the residual error of the neural approximation of value function has the form:

$$\begin{gathered} \varepsilon^{\mu^{(i)}} = V^{\mu^{(i)}}(x(t+T)) - V^{\mu^{(i)}}(x(t)) + \\ \int_t^{t+T} \left(x^T(\tau) Q x(\tau) + \mu^{(i)}(x(\tau))^T(\tau) R \mu^{(i)}(x(\tau))\right) \\ d\tau = \left(W^{\mu^{(i)}}\right)^T (\phi(x(t+T)) - \phi(x(t))) + \\ \int_t^{t+T} \left(x^T(\tau) Q x(\tau) + \mu^{(i)}(x(\tau))^T(\tau) \mu^{(i)}(x(\tau))\right) d\tau \end{gathered} \tag{22}$$

In general way, multi-layered networks need to evaluate weights using iterative gradient-based algorithms (back-propagation algorithms) but the special simple form of neural network described in this paper allows us to use quick, non-iterative algorithm, introduced in (Vrabie and Lewis, 2009). The algorithm calculates best weights in the least-square meaning by:

$$\begin{gathered} W^{\mu^{(i)}} = -\Phi^{-1}(\varphi(x(t+T)) - \varphi(x(t))), \\ \int\limits_t^{t+T} \left(x^T(\tau) Q x(\tau) + \mu^{(i)}(x(\tau))^T(\tau) R \mu^{(i)}(x(\tau))\right) d\tau_\Omega \end{gathered} \tag{23}$$

where $\Phi = (\varphi(x(t+T)) - \varphi(x(t))), (\varphi(x(t+T)) - \varphi(x(t)))_\Omega$ and $f(x), g(x)_\Omega$ denotes inner product for Lebesgue integral on Ω. More elaborate derivation and the proof of convergence is located in (Vrabie and Lewis, 2009).

4.3 The online algorithm

In this section the ultimate algorithm structure is derived with the concern on the practical use.

Let us assume that the object to be controlled is an autonomic (i.e., not time-dependent) linear dynamic system with known at least B matrix and that the controller can measure state $x(t)$ of the system in any time.

T denotes time-interval after which the control policy μ and value function approximation V will be updated. Let us also assume that the controller is able to calculate or approximate the cost function
$J = \int_{t_1}^{t_2} \left(x^T(\tau) Q x(\tau) + \mu^{(i)}(x(\tau))^T(\tau) R \mu^{(i)}(x(\tau)) \right) d\tau$ with the sufficient accuracy.
One iteration of the algorithm described in 4.2. on the interval $(0,\ T)$ takes the form:
Initialization: Initialize values of the weights $W^{\mu^{(0)}}$. The control policy used on the first interval $(0,\ T)$ is denoted by $\mu^{(0)} = -\frac{1}{2}R^{-1}B^T \nabla V_x^{\mu^{(0)}}$.
1. At the time intervals $(0,\ t_1)$, $(t_1,\ 2t_1)$, $\ldots$, $((i-1)t_i,\ it_1)$,..., $((n-1)t_1,\ T)$ ($T = nt_1,\ n \in \mathbb{Z}$) the local objective is measured:

$$J_i = \int_{(i-1)t_1}^{it_1} \left(x^T(\tau) Q x(\tau) + \mu^{(0)}(x(\tau))^T(\tau) R \mu^{(0)}(x(\tau)) \right) d\tau \tag{24}$$

2. At time $T,\ 2T,\ \ldots$ matrices $\hat{\Phi}$ and $\hat{\Psi}$ are built:

$$\hat{\Phi} = -\begin{bmatrix} \varphi(x(t_1))^T - \varphi(x(0))^T \varphi(x(t_2))^T - \varphi(x(t_1))^T \\ \cdots \varphi(x(T))^T - \varphi(x((n-1)t_1))^T \end{bmatrix}^T,\ \hat{\Phi} \in \mathbb{R}^{n \times L}, \tag{25}$$

$$\hat{\Psi} = \begin{bmatrix} J_1 & J_2 & \cdots & J_n \end{bmatrix},\ \hat{\Psi} \in \mathbb{R}^n \tag{26}$$

3. The next set of weights are calculated:

$$W^{\mu^{(1)}} = \hat{\Phi}^+ \hat{\Psi} \tag{27}$$

where $\hat{\Phi}^+$denotes a pseudoinverse of rectangular matrix $\hat{\Phi}$.

On the following intervals $(T,\ 2T)$, $(2T,\ 3T)$, etc. the algorithm works the same way.

Implementation of this algorithm differ from the one used in (Vrabie and Lewis, 2009) by stop and start conditions:

- The algorithm should stop if error of approximation of V is small enough, i.e. $\left\| \varepsilon^{\mu^{(i)}} \right\| < \delta_1$ where $\varepsilon^{\mu^{(i)}}$ is denoted by (22).
- The algorithm should restart if the dynamics has changed, i.e., the present approximation of V has big enough error: $\left\| \varepsilon^{\mu^{(i)}} \right\|) > \delta_2,\ \delta_2 \geq \delta_1$.
- It is essential that when the change of dynamics occurs and the procedure starts again, the initial weights are such that the initial policy μ is stabilizable. This condition appears because the change of the dynamics can be such significant that old, then-optimal policy could destabilize the new system. In the case of

control of mechanical systems, to ensure that initial weights provide stabilizing policy control, one can assign 0 to all weights but that associated with velocities, which should be set to positive values. In this case initial control policy μ resembles damping forces, because it depends linearly on velocities. Damping is of course a stabilizable type of control, because it dissipates energy in the system. More elaborate description of initial set of weights is provided in the Section 5.

The algorithm procedure

Initialization. Choose the quadratic objective $V = \int_0^\infty F\left(x(\tau),\, u(\tau)\right) d\tau$ intended to be minimized. Choose the set $\varphi(x)$ of activating function. Set $i = 0$. Set weights $W^{\mu^{(i=0)}}$to ensure that policy control $\mu^{(0)}$ is stabilizable. Choose duration time T of each iteration and the number $n \in \mathbb{N}^+$ of measurements of the objective function in each iteration. Choose the value δ_1 of the stop condition and the value δ_2 of the restart condition.

Step 1. Steer the object utilizing the policy control $u = -\frac{1}{2}R^{-1}B^T\ \nabla\varphi_x(x)^T W^{\mu^{(i)}}$. Measure the objective function $J_l = \int_{t_l}^{t_{l+1}} F\left(x(\tau),\, u(\tau)\right) d\tau$ and the state $\mathrm{x_l = x\,(t_l)}$ at the time interval $\mathrm{(iT,\ [i+1]\,T)}$, where $l = 0,\ 1,\ \ldots,\ n$ and $t_l = (l/n)T$.

Step 2. Build matrices *hat*Φ and $\hat{\Psi}$ using according to (25) and (26).

Step 3. Calculate error of approximation $\varepsilon^{\mu^{(i)}} = \left[\begin{array}{cccc} \varepsilon_1 & \varepsilon_2 & \cdots & \varepsilon_n \end{array}\right]^T$, where $\varepsilon_l = V^{\mu^{(i)}}\left(x(t_l)\right) - V^{\mu^{(i)}}\left(x(t_{l+1})\right) + J_l$.

Step 4. If $\left\|\varepsilon^{\mu^{(i)}}\right\| < \delta_1$ then don't change weights, i.e., $W^{\mu^{(i+1)}} = W^{\mu^{(i)}}$ set $i \to i+1$ and jump to the Step 1. If $\left\|\varepsilon^{\mu^{(i)}}\right\| \geq \delta_1$ then proceed to the Step 5.

Step 5. If $\left\|\varepsilon^{\mu^{(i)}}\right\| > \delta_2$ then restart algorithm, i.e., set next weights as the initial ones $W^{\mu^{(i+1)}} = W^{\mu^{(0)}}$, set $i \to i+1$ and return to Step 2. If $\left\|\varepsilon^{\mu^{(i)}}\right\| \leq \delta_2$ then set next weights according to (27), i.e., $W^{\mu^{(i+1)}} = \hat{\Phi}^{+}\hat{\Psi}$, $i \to i+1$ and return to the Step 1.

In the real-life implementations, where any measurement has disturbance, choosing δ_1 and δ_2 will be a trade-off between sensitivity of the change of the dynamics and robustness to disturbances. It is good to point out that the integral J_i (24) has a property of a downpass filter and that small changes of dynamics should not change significantly the effectiveness of the controller.

5 Illustrative example - conjugate oscillators under sudden loss of stiffnes

In this section the real-life case of control problem is presented: vibration control of the system with jump damage (breaking off the spring). Numerical simulation of the vibrating system is conducted with control provided by the GPI regulator. The example is provided with comparison of the LQR regulator.

The controlled system is composed of two masses m_1 and m_2, linked to the rigid bases by springs k_1 and k_3 and joined by spring k_2. The object is controlled by the input force u applied to the mass m_2. The scheme of the object is presented in Fig. 2.

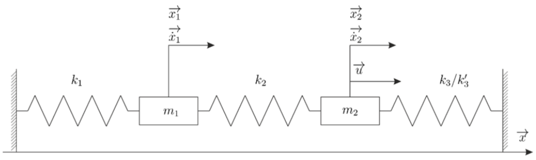

Fig 2. Scheme of the example object. Two masses linked by three springs.

The local coordinates are chosen in such way that the system has equilibrium at the point $(x_1, \dot{x}_1, x_2, \dot{x}_2) = (0, 0, 0, 0)$. The system dynamics is represented by

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \ddot{x}_1 \\ \dot{x}_2 \\ \ddot{x}_2 \end{bmatrix} = Ax + Bu = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_1+k_2}{m_1} & 0 & \frac{k_2}{m_1} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k_2}{m_2} & 0 & -\frac{k_2+k_3}{m_2} & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u \tag{28}$$

with initial point $x(0) = x_0$, where x_0 reflects the displacement of the first mass by $10^{-1}\ m$ in the direction of the x axis:

$$x_0 = \begin{bmatrix} 10^{-1}\ m & 0 & 0 & 0 \end{bmatrix}^T \tag{29}$$

We assume the following set of model parameters:

$$k_1 = k_2 = 1 \text{ [N/m]},\ \ m_1 = m_2 = 1 \text{ [kg]},\ \ k_3 = 30 \text{ [N/m]} \tag{30}$$

For the objective function defined by (2) we assume:

$$Q = \mathbb{I},\ \ R = 1 \tag{31}$$

where $\mathbb{I}$ stands for the identity matrix. The choice of the parameters (31) means that all states and the control will be minimized with the same weight.

The simulation is divided into three parts. At the time interval $t \in [0,\ 30[\text{s}])$ both regulators the LQR regulator and the GPI regulator are tested for control of dynamic system denoted by (28) with the initial point (29). The feedback matrix K for the LQR regulator is calculated by (4) with the assumption of full knowledge of the system matrices and the GPI regulator starts with initial weights. At the time $t = 30$[s] the structural damage is simulated by the instantaneous drop of k_3 to $k_3^{'} = 0.3$[N/m] and the system state is set again to x_0. The change of the spring stiffness is represented by the change of the system matrix by ΔA:

$$\dot{x} = (A + \Delta A)\,x + Bu \tag{32}$$

At the time interval (30, 60] the simulation of the damaged system is conducted. Both the feedback matrix K for the LQR algorithm and weights for the GPI regulator are not explicitly changed.

5.1 GPI regulator setup

Because $x \in \mathbb{R}^4$, sufficient number of activation functions for the GPI algorithm is equal to $\overline{C_4}^2 = 10$:

$$\varphi(x) = \begin{bmatrix} x_1^2 & x_1\dot{x}_1 & x_1x_2 & x_1\dot{x}_2 & \dot{x}_1^2 & \dot{x}_1x_2 & \dot{x}_1\dot{x}_2 & x_2^2 & x_2\dot{x}_2 & \dot{x}_2^2 \end{bmatrix}^T \tag{33}$$

Gradient of the activation functions has then below form:

$$\nabla\varphi_y = \begin{bmatrix} 2x_1 & \dot{x}_1 & x_2 & \dot{x}_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & x_1 & 0 & 0 & 2\dot{x}_1 & x_2 & \dot{x}_2 & 0 & 0 & 0 \\ 0 & 0 & x_1 & 0 & 0 & \dot{x}_1 & 0 & 2x_2 & \dot{x}_2 & 0 \\ 0 & 0 & 0 & x_1 & 0 & 0 & \dot{x}_1 & 0 & x_2 & 2\dot{x}_2 \end{bmatrix}^T \tag{34}$$

We assume initial weights $W^{\mu^{(0)}} \in \mathbb{R}^{10}$ as follows:

$$W^{\mu^{(0)}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 \end{bmatrix}^T \tag{35}$$

Then initial policy equals to:

$$\begin{gathered} u^0(t) = \mu^{(0)}(x(t)) = -\tfrac{1}{2}R^{-1}B^T\nabla V_x^{\mu^{(0)}} = \\ -\tfrac{1}{2}B^T\left(W^{\mu^{(0)}T}\nabla\phi_y\right)^T = -\tfrac{1}{2}B^T\begin{bmatrix} 0 \\ 0 \\ 0 \\ 8\dot{x}_2 \end{bmatrix} = -4\dot{x}_2(t) \end{gathered} \tag{36}$$

The control $u^0(t)$ is the force proportional to the velocity of the second mass with the opposite direction to it, so $u^0(t)$ has damping nature which ensures that $u^0(t)$ is a stabilizable control.

The choice of numerical coefficients δ_1 for the halt and δ_2 for the fire of the GPI algorithm was run by the trial and error and these coefficients ultimately equals 10^{-10} and 1, respectively.

The choice of the duration T and the number n of measurements in each iteration is a tradeoff between the speed of convergence of the algorithm and the level of the numerical error in calculation of the weights. For the accuracy of the computation it is important to ensure that measurements are applied to as wide as possible range of the state values, so for the systems with relatively small time constant, T can be set to a smaller value. The adaptation of T to changing system dynamics, i.e., extending interval between measurements if the state change is not big enough is workable, but this development of the algorithm is not in the scope of our work. It is natural to expect that the number n of the measurements should be greater or equal to the size of the weights vector which in our case equals 10. The values T and n for the simulation conducted in this work are selected by the trial and error and equal 6.6 [s] and 30, respectively.

5.2 Numerical results

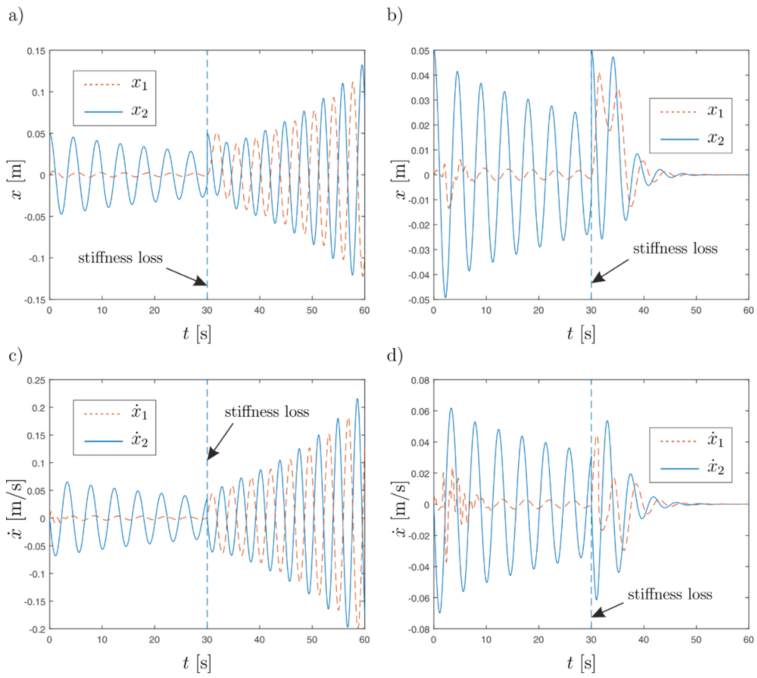

Fig 3. The simulation of the control of the system; a) - displacements of the masses under the LQR control; b) - displacements of the masses under the GPI control; c), d) - velocities of the masses under the LQR and GPI control. Blue dotted line indicates the time when dynamics of the model has changed.

Below, the results of the numerical experiment are provided. In Fig. 3. the states of the system is presented. In Fig. 4. shapes of the controls produced by both algorithms is provided. Fig. 5. Shows objective functionals $J(t) = \int_0^t x(\tau)^T Qx(\tau) + u(\tau)^T Ru(\tau)\ d\tau$ achieved by both algorithms. One can see in the first part of simulation (when $k_3 = 30$ [N/m]) that the best results is provided by the LQR regulator. It happened of course because the feedback matrix K was calculated before simulation, with exact knowledge of the system dynamics. In the other hand, the GPI algorithm presented in this paper starts with no knowledge of the system and begin with the safe but not optimal "damping-like" control. From $t = 1.8$[s] to $t \approx 12$[s] one can distinguish transition phase of the GPI algorithm where weights are not in the steady state. It is important to point out that after this transition phase, the algorithm achieve the same effectiveness as the LQR algorithm. One can deduce then the main advantage of using the GPI algorithm - its effectiveness converges to the effectiveness of the LQR algorithm, but it does not need the full knowledge of the system dynamics.

After 30 seconds, when dynamics is changed and simulation starts from x_0 the LQR regulation causes instability of the system. All system variables gain values, energy is added to the system, the envelope of the control is increasing.

Response of the GPI algorithm is dramatically different. As presented in Fig. 4. b) the change in dynamics is detected almost immediately. The weights are set to initial, stabilizing values. The system is taken to the equilibrium state at about 15 seconds of the control.

These observations are validated by the chart presented in the Fig. 5. At the transition time the objective functional for the GPI control is quickly increasing but after the control policy converges, the difference between both objectives became steady. At the part of the simulation responding to the change of the system's dynamics the GPI control quickly converges and steers the system to its origin. In the contrast, objective of the LQR simulation became superlinear and evidently means that the system is unstable. Considering only the second part of the simulation, the final objective functionals achieved by both algorithms are 0.1258 for the LQR algorithm and 0.0554 for the GPI algorithm, which is the 55% better result than for the LQR. Although, it is hard to compare these values because only the GPI control simulation achieves steady state within the duration of the simulation. The value of J_{GPI} do not change after increasing the final time, but the J_{LQR} do, so this comparison is not definitive.

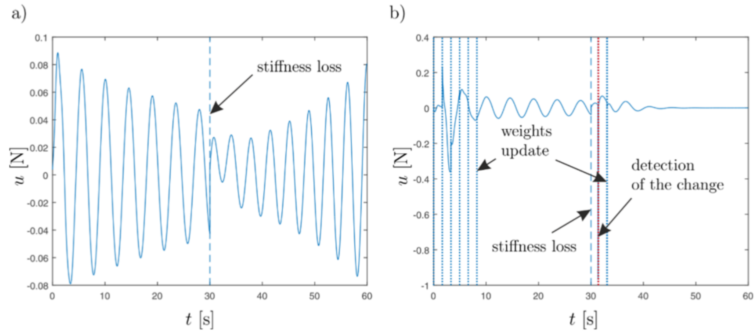

Fig 4. The simulation of the control of the system; a), b) - control generated by the LQR and GPI algorithms. Blue dotted lines on the GPI control chart point to times when the weights are updated, the red ones point to the times when change of the system is detected and the weights are set to its initial values.

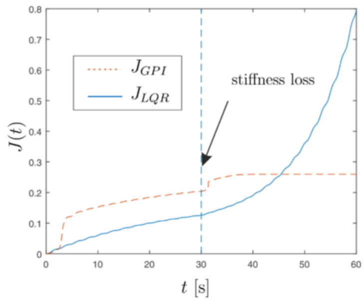

Fig 5. The objective functional for both the GPI and the LQR control case. In the first part of the simulation, minimization of the objective by the GPI control is worse than by the LQR control because of the time required to achieve convergence of the control policy. After that time the disparity between both objectives is steady. In the second part of the simulation, the objective of the GPI algorithm quickly became constant, i.e., the system is quickly steered to its origin. The second objective is superlinear, i.e., the LQR controller made the system unstable.

6 Conclusions

In this paper optimal adaptive controller of vibrational mechanical systems was studied. The algorithm works efficiently without exact knowledge of system dynamics. It evinces rapidity of achieving optimal control. Simulation results shows that this algorithm successfully controls systems for which classic LQR approach results in instability. The studied algorithm requires that first control policy to be admissible. For mechanical problems this requirement is particularly easy to fulfill, because predominantly they have velocities of their parts explicitly located in the state vector, and the control proportional to velocities of the elements will eventually dissipate energy of the system.

References

Al-Tamimi, A. and Lewis, F. (2007). Discrete-time nonlinear hjb solution using approximate dynamic programming: Convergence proof. In *2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*, pages 38–43. IEEE.

Baird, L. C. (1994). Reinforcement learning in continuous time: Advantage updating. In *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on*, volume 4, pages 2448–2453. IEEE.

Beard, R. W., Saridis, G. N., and Wen, J. T. (1997). Galerkin approximations of the generalized hamilton-jacobi-bellman equation. *Automatica*, 33(12):2159–2177.

Dyke, S., Sain, M., Carlson, J., et al. (1996). Modeling and control of magnetorheological dampers for seismic response reduction. *Smart Materials and Structures*, 5(5):565–575.

Fujino, Y., Soong, T., and Spencer, B. (1996). Structural control: Basic concepts and applications. In *Building an International Community of Structural Engineers*, pages 1277–1287. ASCE.

Hillsley, K. and Yurkovich, S. (1991). Vibration control of a two-link flexible robot arm. In *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pages 212–216. IEEE.

Hu, Y.-R. and Ng, A. (2005). Active robust vibration control of flexible structures. *Journal of sound and vibration*, 288(1):43–56.

Kalman, R. E. et al. (1960). Contributions to the theory of optimal control.

Kar, I., Miyakura, T., and Seto, K. (2000a). Bending and torsional vibration control of a flexible plate structure using $H\infty$ - based robust control law. *IEEE Transactions on Control Systems Technology*, 8(3):545–553.

Kar, I. N., Seto, K., and Doi, F. (2000b). Multimode vibration control of a flexible structure using $H\infty$-based robust control. *IEEE/ASME transactions on Mechatronics*, 5(1):23–31.

Kucuk, I., Yildirim, K., Sadek, I., and Adali, S. (2013). Active control of forced vibrations in a beam via maximum principle. In *Modeling, Simulation and Applied Optimization (ICMSAO), 2013 5th International Conference on*, pages 1–4. IEEE.

Li, X., Agarwal, R. K., and Shue, S.-P. (1994). Optimal control and H∞ ufilter for control of Timoshenko beam vibrations using piezoelectric material. In *Decision and Control, 1998. Proceedings of the 37th IEEE Conference on*, volume 2, pages 1566–1571. IEEE.

Mohamed, Z., Chee, A., Hashim, A. M., Tokhi, M. O., Amin, S. H., and Mamat, R. (2006). Techniques for vibration control of a flexible robot manipulator. *Robotica*, 24(04):499–511.

Peng, F., Ng, A., and Hu, Y.-R. (2005). Actuator placement optimization and adaptive vibration control of plate smart structures. *Journal of Intelligent Material Systems and Structures*, 16(3):263–271.

Pisarski, D. (2011). Semi-active control system for trajectory optimization of a moving load on an elastic continuum.

Pisarski, D. and Bajer, C. I. (2010). Semi-active control of 1d continuum vibrations under a travelling load. *Journal of sound and vibration*, 329(2):140–149.

Sakai, C., Ohmori, H., and Sano, A. (2003). Modeling of mr damper with hysteresis for adaptive vibration control. In *42nd IEEE Conference on Decision and Control*.

Singhose, W. (2009). Command shaping for flexible systems: A review of the first 50 years. *International Journal of Precision Engineering and Manufacturing*, 10(4):153–168.

Soong, T. and Constantinou, M. (1994). Passive and active structural vibration control in civil engineering.

Spencer Jr, B. (1996). Recent trends in vibration control in the usa. In *Proc., 3rd Int. Conf. on Motion and Vibr. Control*, pages K1–K6.

Sutton, R. S., Barto, A. G., and Williams, R. J. (1992). Reinforcement learning is direct adaptive optimal control. *IEEE Control Systems*, 12(2):19–22.

Symans, M. D. and Constantinou, M. C. (1999). Semi-active control systems for seismic protection of structures: a state-of-the-art review. *Engineering structures*, 21(6):469–487.

Tzes, A. and Yurkovich, S. (1993). An adaptive input shaping control scheme for vibration suppression in slewing flexible structures. *IEEE Transactions on Control Systems Technology*, 1(2):114–121.

Tzou, H. and Tseng, C. (1990). Distributed piezoelectric sensor/actuator design for dynamic measurement/control of distributed parameter systems: a piezoelectric finite element approach. *Journal of sound and vibration*, 138(1):17–34.

Vrabie, D. and Lewis, F. (2009). Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural Networks*, 22(3):237–246.

Youn, S.-H., Han, J.-H., and Lee, I. (2000). Neuro-adaptive vibration control of composite beams subject to sudden delamination. *Journal of sound and vibration*, 238(2):215–231.