

Real-time Neural Network Based Semiactive Model Predictive Control of Structural Vibrations

Tianhao Yu^a, Zeyu Mu^{a,1}, Erik A. Johnson^{a,*}

^a*Sonny Astani Department of Civil and Environmental Engineering, University of Southern California, Los Angeles, CA 90089, USA*

Abstract

Semiactive model predictive control (sMPC) can be very effective, but its computational cost due to the inherent mixed-integer quadratic programming (MIQP) optimization precludes its use in real-time vibration control. This study proposes training neural networks (NNs) to predict in real-time only the MIQP's integer variables' values, called a strategy, for a given structure state. Because the number of strategies is exponential in the number of sMPC horizon steps, the resulting NN can be massive. This study proposes to reduce the NN dimension by exploiting the homogeneity-of-order-one nature of this control problem and, using state vector statistics, to efficiently choose training samples. The single large NN is proposed to be split into several much smaller NNs, each predicting a strategy grouping, that together uniquely and efficiently predict the strategy. Given the strategy's integer

*Corresponding author at: Sonny Astani Department of Civil and Environmental Engineering, University of Southern California, Los Angeles, California 90089, USA.

Email address: JohnsonE@usc.edu (Erik A. Johnson)

¹Present address: the Link Lab and Department of Engineering Systems and Environment, University of Virginia, Charlottesville, VA 22904 USA.

values, the MIQP optimization reduces to a quadratic programming (QP) problem, solved using a fast QP solver with proposed adaptations: exploiting optimization efficiencies and bounding sub-optimality; using several NN predictions; and reverting to a simpler (suboptimal) semiactive control algorithm upon occasional incorrect NN predictions or QP solver nonconvergence. Shear building examples demonstrate significant online computational cost reductions with control performance comparable to the conventional MIQP-based control.

Keywords: Model predictive control, neural network, real-time control, semiactive control

1. Introduction

1.1. Challenges in semiactive model predictive control implementation

Semiactive controllable damping strategies can perform better than passive control algorithms, consume less energy than active control algorithms, and are inherently stable because the control forces are naturally dissipative (the control forces always act to oppose motion across the devices), providing real benefits in the vibration control of civil structures. However, existing semiactive control algorithms are either sub-optimal or computationally prohibitive for real-time implementation.

Sub-optimality occurs when the control forces are first determined while ignoring the dissipativity constraint, and then clipped (*e.g.*, set to zero, or as close to zero as is possible) with a secondary control device — *e.g.*, the clipped linear quadratic regulator (clipped LQR or CLQR) (Figure 1a) — which

takes a toll on the original (pre-clipping) control performance, especially when frequent clipping is required.

A control algorithm that avoids the clipping sub-optimality is the semi-active model predictive control (semiaactive MPC or sMPC), a form of hybrid MPC (hMPC) that introduces integer variables to explicitly enforce the dissipative constraint (Figure 1b). However, finding the hMPC solution involves solving mixed-integer quadratic programming (MIQP) problems, making it too slow for real-time implementation in structural vibration control.



Figure 1: Comparison of control algorithms

One way of accelerating the hMPC while avoiding the excessive memory requirements of large look-up tables (Elhaddad and Johnson, 2013) is to apply the algorithm in Bertsimas and Stellato (2019, 2021): create a list of candidate combinations of integer variables’ values and inequality constraints that are active at optimality (active constraints) — each unique combination of integer variables’ values and active constraints is termed a “strategy” and denoted herein by S_{II}^s , $s = 0, 1, \dots, M - 1$ — and train offline a feedforward neural network (NN), as in Figure 2, to predict online the probability $\mathcal{P}(S_{\text{II}}^s)$

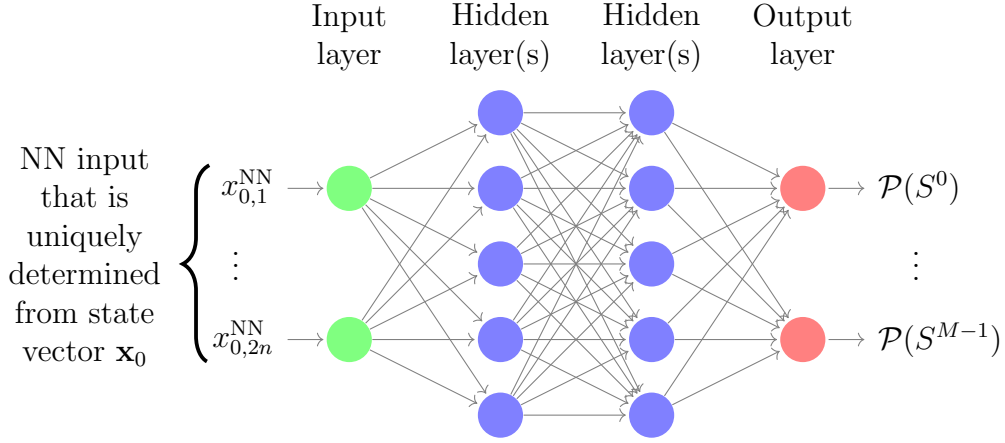


Figure 2: An example of neural network classifiers

that strategy S_{II}^s leads to the optimal solution when given the current state \mathbf{x}_0 . Adopting the strategy $S_{\text{II}}^{s^*}$ that is predicted most likely to be correct, *i.e.*, $s^* = \arg \max_s \mathcal{P}(S_{\text{II}}^s)$, the online computation of controllable damping forces simplifies to solving a set of linear equations. However, a direct implementation of this approach is difficult due to two challenges. First, because the NN's input vector (the current state vector) is generally high-dimensional and the number of possible strategies is huge, many samples are required to effectively train the NN, which is computationally expensive; *e.g.*, it took 19 CPU-years to generate enough samples for a direct solution to the ten-state numerical example in Section 5.1. Second, because there are too many strategies — *e.g.*, for the same example, the training samples constituted millions of unique S_{II} strategies — any NN classifier with a fair performance must be of gigantic scale, with many hidden layers and many nodes, again leading to both long training time and prohibitive prediction time.

The major contribution of this study is to address these challenges: first, by using multiple smaller NNs in parallel via the “codeword” approach (Zhang et al., 2018); and second, by predicting only the integer variables’ values, not the active constraints, which reduces the online optimization problem from MIQP to quadratic programming (QP), which is more complex than just a set of linear equations but can be efficiently solved by the fast MPC (fMPC) algorithm (Wang and Boyd, 2010) with essential adaptations proposed herein. The idea of predicting the integer variables’ values was mentioned previously in the field of robotic control (Cauligi et al., 2020), but was also independently developed and presented by the first author (Yu, 2020) a few months before Cauligi et al. (2020) was made public. Further, the problems addressed in Cauligi et al. (2020) are of far smaller scale compared with the MIQP problem in structural vibration control, leaving challenges, unique to this application, that are addressed as follows:

1. Straightforward NN training would use a vector of the system states as the NN input. However, by exploiting the control algorithm’s homogeneity-of-order-one nature (*i.e.*, the control forces scale with the magnitude of the state vector and the integer variables remain unchanged), all input state vectors for NN training are normalized, leading to a moderate reduction of the sampling space/effort.
2. To train the NNs on the resulting unit hypersphere in a manner that approximately uniformly represents the relative state magnitudes, the

state covariance matrix is approximated from the hMPC-controlled structure’s (offline) simulated responses to a Gaussian excitation.

3. The approach proposed in Cauligi et al. (2020) to reduce the scale of the NN does not apply herein; thus, the NN remains very large and computationally prohibitive. To cut both the training and the prediction time, a “codewords” (Zhang et al., 2018) paradigm is introduced to replace a single large NN with multiple smaller NNs, each predicting the likelihoods of each strategy within a unique set of strategies, chosen so that the NNs together uniquely predict individual strategies: strategy number s is represented as a set of *digits* of some base, and each smaller NN predicts one such digit to uniquely determine s .
4. A small-scale problem can be solved efficiently by a commercial optimization solver after the integer variables are available; however, commercial solvers are too slow for the real-time implementation required herein. To guarantee sufficient computing speed, a C-language implementation of the fMPC in Wang and Boyd (2010) is tailored to find the control forces given the NN(s)-predicted integer variables.

In addition to these considerations, three techniques are proposed to promote the accuracy and/or the computational efficiency of the proposed algorithm.

5. Instead of a fixed penalty coefficient in the QP solution (*i.e.*, when implementing the fMPC), this study proposes a per-time-step adaptive penalty coefficient that promotes the QP solution accuracy.

6. An easily computed, guaranteed dissipative, but (likely) suboptimal semiactive control strategy (run in parallel with the proposed algorithm) is taken as the initial solution for the QP solver for the fMPC algorithm, speeding up the computation in the same fashion as what some may call a “warm-start” technique (Wright, 2000; Yildirim and Wright, 2002) in solving optimization problems.
7. Previous studies never considered the effects of inaccurate NN predictions, which are inevitable in real practice; further, the fMPC optimization can, occasionally, result in a non-converged sub-optimal solution. To address these, this study proposes using several strategies predicted most likely by the NNs, not just the one most probable, comparing those strategies’ predicted costs with each other and against that from an alternate easily computed semiactive control strategy, and choosing at each time step the one that leads to the smaller predicted cost. This ensures that the proposed algorithm will always perform at least comparably to, and usually better than, the suboptimal-but-easy strategy.

These proposed solutions to the unique challenges as well as these proposed techniques enable an effective real-time sMPC for structural vibration control. Simulations herein using single-degree-of-freedom (SDOF), 3DOF and 5DOF structure models show that the proposed algorithm can achieve control performance nearly identical to that of the hMPC but at a much faster speed that is sufficient for real-time implementation.

1.2. Related Work

For applications of MPC in structural vibration control and in a range of other industries, see Mei et al. (2001, 2002, 2004) and Qin and Badgwell (2003), respectively. Modeling systems with “on-off” logic (*e.g.*, enforcing the dissipativity constraints) through MIQP was covered in Sontag (1981), Bemporad and Morari (1999), Heemels et al. (2001) and Bemporad (2002). Transforming a semiactive structural vibration control algorithm into a MIQP problem was detailed in Elhaddad and Johnson (2013).

The idea of training NNs offline to accelerate solution of online optimization problems was first proposed in Bertsimas and Stellato (2019, 2021). A similar study (Cauligi et al., 2020) was conducted independently and simultaneously with this study, but is incapable of handling large-scale problems arising from structural vibration control. Separating one huge NN into several smaller NNs was studied in Zhang et al. (2018), which is a generalization of the error-correcting output codes (ECOC) method (Dietterich and Bakiri, 1994; Allwein et al., 2000; Passerini et al., 2004).

A fast solver for the QP problem was introduced in Wang and Boyd (2010), which exploited the block-diagonal structure of the matrices (Wright, 1997; Boyd and Vandenberghe, 2004), applied a warm-start technique (Wright, 2000; Yildirim and Wright, 2002) (though it was found to be less effective than an alternate warm-start proposed herein), and made approximations to accelerate the computation.

2. Problem formulation

For an n DOF building model subjected to an external excitation and controlled by m smart damping devices, the equation of motion is

$$\mathbf{M}\ddot{\tilde{\mathbf{q}}}(t) + \mathbf{C}\dot{\tilde{\mathbf{q}}}(t) + \mathbf{K}\tilde{\mathbf{q}}(t) = \hat{\mathbf{B}}_{\mathbf{u}}\tilde{\mathbf{u}}(t) + \hat{\mathbf{B}}_{\mathbf{w}}\tilde{\mathbf{w}}(t) \quad (1)$$

where \mathbf{M} , \mathbf{C} and \mathbf{K} are the model's $n \times n$ symmetric mass, damping and stiffness matrices; $\tilde{\mathbf{q}}$ is an $n \times 1$ vector of displacements; $\tilde{\mathbf{u}}$ is an $m \times 1$ vector of control forces with $n \times m$ influence matrix $\hat{\mathbf{B}}_{\mathbf{u}}$ that depends on the control device locations; and $\tilde{\mathbf{w}}$ is an $n_w \times 1$ vector of external zero-mean excitations with influence matrix $\bar{\mathbf{B}}_{\mathbf{w}}$. With a zero-order-hold time discretization with sampling time duration Δt , equation (1) can then be written in discrete-time state-space form:

$$\tilde{\mathbf{x}}_{l+1} = \mathbf{A}\tilde{\mathbf{x}}_l + \mathbf{B}_{\mathbf{u}}\tilde{\mathbf{u}}_l + \mathbf{B}_{\mathbf{w}}\tilde{\mathbf{w}}_l \quad (2)$$

where $\tilde{\mathbf{x}}_l = [\tilde{\mathbf{q}}_l^T \quad \dot{\tilde{\mathbf{q}}}_l^T]^T$, $\tilde{\mathbf{u}}_l$ and $\tilde{\mathbf{w}}_l$ are the state, control force and excitation vectors, respectively, at time $l\Delta t$; $\mathbf{A} = e^{\bar{\mathbf{A}}\Delta t}$, $\mathbf{B}_{\mathbf{u}} = \bar{\mathbf{A}}^{-1}(\mathbf{A} - \mathbf{I}_{2n})\bar{\mathbf{B}}_{\mathbf{u}}$, and $\mathbf{B}_{\mathbf{w}} = \bar{\mathbf{A}}^{-1}(\mathbf{A} - \mathbf{I}_{2n})\bar{\mathbf{B}}_{\mathbf{w}}$ and $\bar{\mathbf{A}} = [[\mathbf{0}_n \quad \mathbf{KM}^{-1}]^T \quad [\mathbf{I}_n \quad \mathbf{CM}^{-1}]^T]$, $\bar{\mathbf{B}}_{\mathbf{u}} = [\mathbf{0}_{m \times n} \quad \hat{\mathbf{B}}_{\mathbf{u}}^T \mathbf{M}^{-1}]^T$ and $\bar{\mathbf{B}}_{\mathbf{w}} = [\mathbf{0}_{n_w \times n} \quad \hat{\mathbf{B}}_{\mathbf{w}}^T \mathbf{M}^{-1}]^T$ are the discrete- and continuous-time state-space matrices, respectively; \mathbf{I}_{2n} is a $2n \times 2n$ identity matrix; $\mathbf{0}_n$ is an $n \times n$ zero matrix; and $\mathbf{0}_{(\cdot) \times (\cdot)}$ is an all-zero matrix of the given size.

The control objective is to minimize a quadratic cost

$$\tilde{J} = \Delta t \sum_{l=0}^{n_t} [\tilde{\mathbf{x}}_l^T \mathbf{Q} \tilde{\mathbf{x}}_l + 2\tilde{\mathbf{x}}_l^T \mathbf{N} \tilde{\mathbf{u}}_l + \tilde{\mathbf{u}}_l^T \mathbf{R} \tilde{\mathbf{u}}_l] \quad (3)$$

where \mathbf{R} is a symmetric positive-definite matrix, $\begin{bmatrix} \mathbf{Q} & \mathbf{N} \\ \mathbf{N}^T & \mathbf{R} \end{bmatrix}$ is symmetric and positive semidefinite, and n_t is the number of time steps ($n_t \Delta t$ is the final time). At time step l , the goal is to determine the optimal choice of control force $\tilde{\mathbf{u}}_l$ when the state is $\tilde{\mathbf{x}}_l$. In the absence of constraints, the linear quadratic regulator (LQR) minimizes the expected cost $\mathbb{E}[\tilde{J}]$; for an infinite horizon (*i.e.*, large n_t , tending to infinity), the discrete-time LQR is $\tilde{\mathbf{u}}_l = -\mathbf{K}_{\text{LQR}} \tilde{\mathbf{x}}_l$, with the control gain $\mathbf{K}_{\text{LQR}} = (\mathbf{B}_u^T \mathbf{Q}_p \mathbf{B}_u + \mathbf{R})^{-1} (\mathbf{B}_u^T \mathbf{Q}_p \mathbf{A} + \mathbf{N}^T)$, where \mathbf{Q}_p is the solution to the algebraic Riccati equation $\mathbf{A}^T \mathbf{Q}_p \mathbf{A} - \mathbf{Q}_p - (\mathbf{A}^T \mathbf{Q}_p \mathbf{B}_u + \mathbf{N})(\mathbf{B}_u^T \mathbf{Q}_p \mathbf{B}_u + \mathbf{R})^{-1} (\mathbf{B}_u^T \mathbf{Q}_p \mathbf{A} + \mathbf{N}^T) + \mathbf{Q} = \mathbf{0}$.

Instead, MPC minimizes, subject to the constraints (dissipativity) and state equation, the predicted cost over the next p time steps. Let \mathbf{x}_k be the model-predicted value of the future state $\tilde{\mathbf{x}}_{k+l}$ when the future control force sequence is $\{\mathbf{u}_0, \dots, \mathbf{u}_{p-1}\}$, where \mathbf{u}_k is a possible value of the future control force $\tilde{\mathbf{u}}_{k+l}$. Herein, the cost is further augmented with a term $\mathbf{x}_p^T \mathbf{Q}_p \mathbf{x}_p$ that approximates the residual cost beyond p time steps by emulating a subsequent infinite-horizon discrete LQR cost:

$$J(\mathbf{X}, \mathbf{U}; \mathbf{x}_0) = \Delta t \mathbf{x}_p^T \mathbf{Q}_p \mathbf{x}_p + \Delta t \sum_{k=0}^{p-1} \begin{bmatrix} \mathbf{x}_k \\ \mathbf{u}_k \end{bmatrix}^T \begin{bmatrix} \mathbf{Q} & \mathbf{N} \\ \mathbf{N}^T & \mathbf{R} \end{bmatrix} \begin{bmatrix} \mathbf{x}_k \\ \mathbf{u}_k \end{bmatrix} \quad (4)$$

where $\mathbf{X} = [\mathbf{x}_1^T \ \mathbf{x}_2^T \ \dots \ \mathbf{x}_p^T]^T$ and $\mathbf{U} = [\mathbf{u}_0^T \ \mathbf{u}_1^T \ \dots \ \mathbf{u}_{p-1}^T]^T$ collect the future state predictions and the candidate control forces, respectively. The dissipativity constraints for an ideal controllable damper can be expressed as nonlinear equalities $\text{sgn } v_{k,j}^{\text{rel}} = \text{sgn } u_{k,j}$ or nonlinear inequalities $v_{k,j}^{\text{rel}} u_{k,j} \geq 0$

at time step k for each damper $k = 1, 2, \dots, m$, where $\mathbf{v}_k^{\text{rel}} = -\hat{\mathbf{B}}_{\mathbf{u}}^{\text{T}} \dot{\mathbf{q}}_k = [\mathbf{0}_{m \times n} \quad -\hat{\mathbf{B}}_{\mathbf{u}}^{\text{T}}] \mathbf{x}_k = \mathbf{V} \mathbf{x}_k$ is the vector of velocities across the damping devices.

The constrained minimization of the expected cost $\mathbb{E}[J]$ can be recast as an hMPC (MIQP) problem. Define $\boldsymbol{\Delta} \equiv [\boldsymbol{\delta}_1^{\text{T}} \quad \boldsymbol{\delta}_2^{\text{T}} \quad \dots \quad \boldsymbol{\delta}_{p-1}^{\text{T}}]^{\text{T}}$, where $\boldsymbol{\delta}_k$ is an $m \times 1$ vector of binary numbers $\delta_{k,j}$ (the ‘‘integer’’ parts of the MIQP) that will, upon solution, be $H(v_{k,j}^{\text{rel}})$, where $H(\cdot)$ is the Heaviside unit step function (*i.e.*, $\delta_{k,j} = 1$ implies $v_{k,j}^{\text{rel}} > 0$); $\boldsymbol{\delta}_0$ is known *a priori* because \mathbf{x}_0 is known. The MIQP problem in search space $\boldsymbol{\xi} \equiv [\mathbf{X}^{\text{T}} \quad \mathbf{U}^{\text{T}} \quad \boldsymbol{\Delta}^{\text{T}}]^{\text{T}}$ is, then,

$$\boldsymbol{\xi}^* = \arg \min_{\boldsymbol{\xi}} J(\boldsymbol{\xi}; \mathbf{x}_0) \quad (5a)$$

$$\text{s.t.:} \quad \mathbf{x}_{k+1} = \mathbf{A} \mathbf{x}_k + \mathbf{B}_{\mathbf{u}} \mathbf{u}_k \quad (5b)$$

$$-(\mathbf{v}^{\min} - \epsilon) \circ \boldsymbol{\delta}_k \leq \mathbf{v}_k^{\text{rel}} - \mathbf{v}^{\min}, \quad k > 0 \quad (5c)$$

$$-(\mathbf{v}^{\max} + \epsilon) \circ \boldsymbol{\delta}_k \leq -\mathbf{v}_k^{\text{rel}} - \epsilon, \quad k > 0 \quad (5d)$$

$$-(\mathbf{u}^{\min} - \epsilon) \circ \boldsymbol{\delta}_k \leq \mathbf{u}_k - \mathbf{u}^{\min} \quad (5e)$$

$$-(\mathbf{u}^{\max} + \epsilon) \circ \boldsymbol{\delta}_k \leq -\mathbf{u}_k - \epsilon \quad (5f)$$

$$\mathbf{x}^{\min} \leq \mathbf{x}_{k+1} \leq \mathbf{x}^{\max} \quad (5g)$$

where: $k = 0, \dots, p-1$ throughout the remainder of the paper unless otherwise noted; ϵ is a tiny positive scalar (required for solvers unable to handle strict inequality constraints); ‘‘ \circ ’’ is the Hadamard (element-by-element) product operator of two matrices/vectors; and the lower $\mathbf{x}^{\min} = -\mathbf{x}^{\max} < 0$ and upper $\mathbf{x}^{\max} > 0$ bounds on states \mathbf{x}_k , lower $\mathbf{v}^{\min} = -\mathbf{v}^{\max} < 0$ and upper $\mathbf{v}^{\max} > 0$ bounds on cross-damper velocities $\mathbf{v}_k^{\text{rel}}$, and lower $\mathbf{u}^{\min} = -\mathbf{u}^{\max} < 0$ and upper $\mathbf{u}^{\max} > 0$ bounds on control forces \mathbf{u}_k are introduced as actual

bounds — if needed — or as artificial bounds (of sufficiently large magnitude that the corresponding inequalities are never active) that improve the numerical conditioning of the solver. (For the numerical examples herein, the lower and upper bounds for each quantity are of equal magnitude but opposite sign, and chosen large enough that they are inactive.) Constraints (5c)–(5d) ensure that $\mathbf{v}^{\min} \leq \mathbf{v}_k^{\text{rel}} \leq \mathbf{v}^{\max}$ (element-wise) and that $\delta_{k,j} = H(v_{k,j}^{\text{rel}})$; constraints (5e)–(5f) do likewise for \mathbf{u}_k . Each constraint in (5c)–(5g) can be cast in the form $\mathbf{f}_i^{\text{T}} \boldsymbol{\xi} \leq h_i$; these can be combined to form a vector inequality $\mathbf{F} \boldsymbol{\xi} \leq \mathbf{h}$, where $\mathbf{F} = [\mathbf{f}_1 \ \mathbf{f}_2 \ \dots \ \mathbf{f}_{4(m+n)p-2m}]^{\text{T}}$ and $\mathbf{h} = [h_1 \ h_2 \ \dots \ h_{4(m+n)p-2m}]^{\text{T}}$. It should be noted that the external excitation \mathbf{w}_k , if zero-mean, vanishes from (5b) as proven in Elhaddad and Johnson (2013).

3. Neural networks and samples for offline training

This section focuses on training NNs to predict the binary variable vector $\boldsymbol{\Delta}^* = [\boldsymbol{\delta}_1^{\text{T}} \ \dots \ \boldsymbol{\delta}_{p-1}^{\text{T}}]^{\text{T}}$ for a given \mathbf{x}_0 . Two classes of strategies can be defined. This study focuses on a strategy class, denoted S_{I} herein, that is defined solely by a particular value of the binary vector $\boldsymbol{\Delta}$. The second strategy class, denoted S_{II} and used in previous studies, is additionally defined by a vector $\boldsymbol{\theta} = [\boldsymbol{\theta}_{\mathbf{v}}^{\text{T}} \ \boldsymbol{\theta}_{\mathbf{u}}^{\text{T}}]^{\text{T}}$ of integer values indicating which constraints (5c)–(5g) are active; the $(mk + j)^{\text{th}}$ element ($1 \leq j \leq m$) of $mp \times 1$ vector $\boldsymbol{\theta}_{\mathbf{v}}$ is -1 if the corresponding element of the lower bound (5c) is active, $+1$ if the upper bound (5d) is active, and 0 if both are inactive; $\boldsymbol{\theta}_{\mathbf{u}}$ is defined similarly but for constraints (5e) and (5f).

In this study, only fully-connected feedforward NNs are used, in which the numbers of nodes in the input and output layers match the dimensions of the input and the output, respectively, but the numbers of hidden layers and nodes in each hidden layer are chosen case-by-case so that the NNs can be trained to have prediction accuracy (fraction of NN most-likely-strategy predictions that are correct) of around 95% for both the training data (randomly selected from the depositories of all samples) and the testing data (the remaining data not used for NN training). The activation functions involved are the `Tanh` function for the input and hidden layers and the `Softmax` function for the output layer. It should be noted that the NNs are fixed after training — they do not change with regard to either the excitations or the control forces.

3.1. Shrinking input parameter space — homogeneity

The input to the optimization (5) is a vector of system states $\mathbf{x}_0 \in \mathbb{R}^{2n}$; the effort for NN training grows exponentially with $2n$. Thus, it is ideal to shrink this space so that fewer samples are needed to train the NN(s) and so that the NN(s) is(are) less complex, *i.e.*, of smaller scale.

One way to reduce the optimization input space dimension by one is to exploit the homogeneity-of-order-one nature of the problem. For this homogeneity to hold, two assumptions are made: (a) the building can be safely controlled so that its states remain bounded and inequality (5g) will never be active; (b) the control device is capable of producing the required control forces so that the force constraints $\mathbf{u}^{\min} \leq \mathbf{u}_k \leq \mathbf{u}^{\max}$ will never be

active (otherwise, a force saturation can be introduced, but is never required in the numerical examples herein). Because homogeneity-of-order-one holds with these assumptions, the sampling space for \mathbf{x}_0 may be contracted from \mathbb{R}^{2n} to only those on the unit hypersphere because the control force for any other state vector can be scaled based on the state vector’s distance from the origin. Further, without loss of generality, the entire unit hypersphere need not be sampled, but only half of it (as the reflection across the state space origin will give control forces of identical magnitudes but opposite signs); thus, only $v_{0,1}^{\text{rel}} = \bar{\mathbf{v}}^T \mathbf{x}_0 \geq 0$ is sampled, with matrix subdivision $\mathbf{V} \equiv [\bar{\mathbf{v}} \quad \bar{\mathbf{V}}^T]^T$ to extract the first row of \mathbf{V} . As proven in Appendix E of Yu (2022), the optimal strategy for the opposite sign is $\Delta|_{\mathbf{v}_0^{\text{rel}}} = \mathbf{1}_{m(p-1) \times 1} - \Delta|_{-\mathbf{v}_0^{\text{rel}}}$ and $\theta|_{\mathbf{v}_0^{\text{rel}}} = -\theta|_{-\mathbf{v}_0^{\text{rel}}}$, where $\mathbf{1}_{(\cdot) \times (\cdot)}$ is a matrix of ones of the given size. Thus, the number of possible S_I strategies is 2^{mp-1} , where m is the dimension of the force vector and p is the MPC prediction horizon, though many of these strategies are unlikely to occur (the “ -1 ” in the power is because $\delta_{0,1}$ is always 1 because the first damper’s relative velocity $v_{0,1}^{\text{rel}}$ is chosen positive for training). Note that 2^{mp-1} is far smaller than the corresponding number of S_{II} strategies, which is $2^{mp-1} 3^{2mp}$.

3.2. Sampling in the input parameter space

In the previous section, the sampling space for \mathbf{x}_0 is limited to $\{\mathbf{x}_0 \mid \mathbf{x}_0 \in \mathbb{R}^{2n}, \|\mathbf{x}_0\|_2 = 1, \bar{\mathbf{v}}^T \mathbf{x}_0 \geq 0\}$. One can quite easily sample (uniformly) on this half unit hypersphere: choose $\boldsymbol{\psi}$ to be a $2n \times 1$ vector of independent standard unit normal random variables, and then use sample

$\mathbf{x}_0 = \boldsymbol{\psi} \|\boldsymbol{\psi}\|_2^{-1} \text{sgn } \psi_{n+1}$ (discarding, of course, an all zero vector if it were to occur). However, the state vector during simulations will not contain independent and identically-distributed states, but rather having relative magnitudes roughly similar to the closed-loop state vector covariance matrix (the response is, of course, not exactly Gaussian, but the second order statistics are sufficient to improve the distribution of samples). Thus, because the proposed algorithm seeks to mimic the hMPC, except for faster online execution, the offline NN training can use samples scaled according to an approximation of the covariance matrix of the hMPC-controlled state response to a Gaussian white noise excitation. That said, the sampling need not follow this covariance exactly (indeed, one could even use a response controlled by the CLQR or another suitable strategy), so one may use an approximate state covariance matrix from the response to some earthquake ground motion, and the time duration of the hMPC simulation need not be infinitely long (but also should not be so short that the covariance is poorly approximated).

Thus, the NN training samples are generated through the following steps:

1. Run the hMPC algorithm on the to-be-controlled structure while it is subjected to a realization of Gaussian white noise excitation for 200 seconds (the numerical examples herein use a zero-order-hold with the sampling time 0.02 s; thus, the hMPC runs for 10,000 time steps, which was found to be sufficient for the numerical examples herein), and estimate the covariance matrix \mathbb{X} of the state vector response.

2. For each of the N samples for NN training, generate system state sample $\mathbf{x}_0 = \hat{\boldsymbol{\psi}} \|\hat{\boldsymbol{\psi}}\|_2^{-1} \text{sgn } \hat{\psi}_{n+1}$, where $\hat{\boldsymbol{\psi}} = \mathbf{L}_{\mathbb{X}} \boldsymbol{\psi}$ is a Gaussian random vector with covariance matrix $\mathbb{X} = \mathbf{L}_{\mathbb{X}} \mathbf{L}_{\mathbb{X}}^T$, and run the hMPC algorithm with that initial state to find the corresponding strategy S (either S_I or S_{II}), providing (\mathbf{x}_0, S) as an input/output pair for NN training.

3.3. Breaking a large NN into smaller NNs

In this study, the number of strategies M can be huge, growing exponentially with the number of MPC horizon steps p . For the 5DOF shear building numerical example in Section 5.1, $M \simeq 13,400$ different strategies are found among seven million samples generated for NN training. Because the number of neurons in the output layer must match M , the number of strategies, and the number of neurons in the last hidden layer, just before the output layer, must be at least M for good NN prediction performance (Zhang et al., 2018); large M means that the scale of the NN is huge, rendering training the NN generally infeasible.

The scale of a NN classifier can be reduced by using the “codeword” approach (Zhang et al., 2018), using the intersection of the predictions from P smaller-scale NNs — which can be trained offline in parallel and run online to make predictions in parallel — to uniquely predict the correct strategy. Instead of labeling each unique strategy with an integer $s \in [0, M - 1]$ and one-hot encode (Harris and Harris, 2012) it, strategy S^s can be labeled with a set of smaller integer values, using one of two approaches. If no ordering of these integers is considered, choose a set $\{\lambda_1, \lambda_2, \dots, \lambda_P\}$ of relatively

prime integers $\lambda_j > 1$ (*i.e.*, for $i \neq j$, λ_i and λ_j have no common positive factors aside from 1) such that $\prod_{j=1}^P \lambda_j \geq M$; because the λ_j values are relatively prime, strategy S^s is uniquely defined by the set $\{s_1, s_2, \dots, s_P\}$ where $s_j = s \bmod \lambda_j$. An alternate approach (not used herein) represents s as the number $s_P \dots s_2 s_1$, with digits $s_j = \lfloor s / \prod_{i=0}^{j-1} \lambda_i \rfloor \bmod \lambda_j$ of base $\lambda_j > 1$ (not necessarily relatively prime), where $\lfloor \cdot \rfloor$ is the “floor” operator (largest integer that is no more than its argument; *i.e.*, $\lfloor a/b \rfloor$ represents integer division), and $\lambda_0 \equiv 1$; if all $\lambda_i = \lambda$, $i > 0$, then this representation is a normal base λ numbering system. With either approach, the j^{th} smaller-scale NN classifier is trained to predict the probabilities of each of the λ_j possibly values $\{0, 1, \dots, \lambda_j - 1\}$ of s_j so that the P individual predictions of most likely s_j together uniquely correspond to the optimal strategy S^s . Because $\lambda_j \ll M$, the scale of each NN classifier is much smaller than the original single large NN classifier; further, both the offline training and the online prediction using these smaller NNs can be performed in parallel.

If the relatively prime numbers λ_j are chosen so that $\prod_{j=1}^{P-1} \lambda_j < M \leq \prod_{j=1}^P \lambda_j$, then a correct strategy prediction requires that all P NNs simultaneously predict correctly, which may be challenging because, in practice, no trained NN is perfect. One way to relax this concern is to use $\gamma_j \geq 1$ most likely predictions from the j^{th} NN, instead of using only the single most likely prediction, leading to $\Gamma = \prod_{j=1}^P \gamma_j$ possible strategy predictions, and computing the corresponding cost for each; then the strategy leading to the smallest cost will be taken as the correct/best strategy. A

second approach would be using additional relatively prime numbers so that $\prod_{j=1}^{P-L-1} \lambda_j < M \leq \prod_{j=1}^{P-L} \lambda_j$ for some positive integer L , leading to as many as $\Gamma' = {}^P C_{P-L} = P! / [(P-L)!L!]$ possible strategy predictions, all of which would need to be evaluated as in the previous approach. Although both approaches could be used together to promote the accuracy of the NN strategy prediction, the second is not explored in this study.

4. Real-time implementation

In Wang and Boyd (2010), a few techniques were adopted to computationally accelerate the QP solution inherent in the fMPC, including a primal barrier method, an infeasible start Newton method, fast computation of the Newton step, a warm-start technique, using fixed $\kappa > 0$ — the coefficient of the log barrier terms associated with the inequalities in (7b), also called the barrier parameter in Wang and Boyd (2010) — and using a fixed iteration limit for the primal barrier method. Most of these techniques or their variants are applicable herein except for using fixed κ ; note that although Wang and Boyd (2010) chose to fix κ to accelerate the algorithm, they also note that κ typically varies in similar algorithms.

The applications in Wang and Boyd (2010) used a fixed κ because the state and the control force ranges were small and because their choice of a warm-start technique requires it — neither of which is the case herein. The warm-start herein (Section 4.3) does not require a constant κ . Further, if κ is fixed and too large, the control performance is good when the cost in (5a)

is large, but becomes overly sub-optimal when the cost is small; in contrast, algorithm convergence within a fixed iteration limit is problematic when the cost is large and κ is fixed and too small, leading to poor control performance. Thus, avoiding a fixed κ is reasonable and promotes greater accuracy.

Another feature of the proposed algorithm, unlike that in Wang and Boyd (2010), is to discard the occasional non-converged fMPC solution, which cannot be trusted to satisfy the equality constraints and/or optimality conditions, and to instead fall back to an easily computed alternative control strategy (the CLQR herein) for that time step. To implement this alternative, the C-language fMPC implementation is augmented with an additional output indicating whether a solution converged.

The remainder of this section focuses on (a) adapting κ at each time step, (b) applying a variant of the warm-start technique, (c) bounding the sub-optimality of the final solution, (d) formulating the fMPC algorithm, and (e) implementing the proposed algorithm in real-time.

4.1. Adapting κ at each time step

According to Wang and Boyd (2010) and Cauligi et al. (2020), the cost of the optimal solution derived from the primal barrier method is suboptimal by up to κ times the number of effective inequality constraints, which herein is $(4mp - 2m)/2 = m(2p - 1)$: ignoring the state bounds (5g) because they will not be active in practice given that they are chosen to be of large magnitudes; minus $2m$ because cross-damper velocity bounds (5c)–(5d) are automatically satisfied at the current time $k = 0$; and divided by two because, given a

value of δ_k and assuming the cross-damper velocity bounds are chosen of sufficiently large magnitude, either the lower bound (5c) or the upper (5d) will never be active — and similar for inequality constraints (5e)–(5f) on \mathbf{u}_k . In other words, if the NN predicts binary vector $\hat{\Delta}_1$ leads to corresponding theoretical minimal cost $J_1^* = \min_{\mathbf{x}, \mathbf{u}} J([\mathbf{X}^T \ \mathbf{U}^T \ \hat{\Delta}_1^T]^T; \mathbf{x}_0)$, but the fMPC solver instead finds the suboptimal cost $\hat{J}_1 = \min_{\mathbf{x}, \mathbf{u}} J([\mathbf{X}^T \ \mathbf{U}^T \ \hat{\Delta}_1^T]^T; \mathbf{x}_0) \geq J_1^*$, then the level of sub-optimality $\Delta J_1 = \hat{J}_1 - J_1^* \leq m(2p - 1)\kappa$. Thus, to maintain a constant level of relative sub-optimality — *i.e.*, $\Delta J_1/J_1^* \approx$ constant or, nearly equivalently assuming ΔJ_1 is small, $\Delta J_1/\hat{J}_1 \approx$ constant — it is reasonable to let the barrier weight κ be proportional to J_1^* or \hat{J}_1 . However, this means that the fMPC’s input κ value depends on the fMPC result, which would require a costly algorithm that iteratively solves fMPC solutions. One approach is to choose an initial value of κ , find the optimal cost \hat{J}_1 through the fMPC solution, and then run the fMPC just one more time; this would be faster than a full iterative solution but still requires two fMPC solutions. Instead, this study proposes approximating J_1^* using the corresponding cost $J^{\text{CLQR}} = J(\boldsymbol{\xi}^{\text{CLQR}}; \mathbf{x}_0)$ that is computed assuming a CLQR algorithm is used to compute the control forces over the next p time steps.

The CLQR algorithm uses the previously-defined LQR control gain \mathbf{K}^{LQR} but clips the control force to zero if nondissipative; *i.e.*, the p -step ahead prediction of the CLQR-controlled system is

$$\left. \begin{aligned} \mathbf{u}_k^d &= -\mathbf{K}^{\text{LQR}} \mathbf{x}_k^{\text{CLQR}} \\ u_{k,j}^{\text{CLQR}} &= u_{k,j}^d H(u_{k,j}^d v_{k,j}^{\text{CLQR}}), \quad j = 1, 2, \dots, m \\ \mathbf{x}_{k+1}^{\text{CLQR}} &= \mathbf{A} \mathbf{x}_k^{\text{CLQR}} + \mathbf{B}_u \mathbf{u}_k^{\text{CLQR}} \end{aligned} \right\}, \quad k = 0, 1, \dots, p-1 \quad (6)$$

where $\boldsymbol{\xi}^{\text{CLQR}} = [\mathbf{X}^{\text{CLQR}\text{T}} \quad \mathbf{U}^{\text{CLQR}\text{T}} \quad \boldsymbol{\Delta}^{\text{CLQR}\text{T}}]^\text{T}$, $\mathbf{X}^{\text{CLQR}} = [\mathbf{x}_1^{\text{CLQR}\text{T}} \quad \mathbf{x}_2^{\text{CLQR}\text{T}} \quad \dots \quad \mathbf{x}_p^{\text{CLQR}\text{T}}]^\text{T}$, $\mathbf{U}^{\text{CLQR}} = [\mathbf{u}_0^{\text{CLQR}\text{T}} \quad \mathbf{u}_1^{\text{CLQR}\text{T}} \quad \dots \quad \mathbf{u}_{p-1}^{\text{CLQR}\text{T}}]^\text{T}$, $\mathbf{u}_k^d = [u_{k,1}^d \quad u_{k,2}^d \quad \dots \quad u_{k,m}^d]^\text{T}$ is the ‘‘desired’’ force vector, and $\boldsymbol{\Delta}^{\text{CLQR}}$ can be ignored as it is not needed for computing the resulting cost (4).

Preliminary numerical simulations demonstrated that the hMPC algorithm can result in a cost reduction of up to 50% relative to that using the CLQR algorithm; thus, J^{CLQR} is on the same order of magnitude as J_1^* and \hat{J}_1 . Given that κ can take on a wide range of values of the correct magnitude and still lead to sufficiently accurate solutions (Wang and Boyd, 2010), the choice of κ need not be perfect and using a κ proportional to J^{CLQR} should provide solutions that are sufficiently accurate.

4.2. Bounding the sub-optimality

By adjusting the barrier coefficient κ , the sub-optimality is properly bounded when $\hat{\boldsymbol{\Delta}}_1 \simeq \boldsymbol{\Delta}^*$. However, even well-trained NNs will occasionally and inevitably provide incorrect predictions (prediction accuracy can be close, but not strictly equal, to 100%). To maintain consistently good control performance, the sub-optimality resulting from these incorrect predictions must be bounded.

To reduce the effect of incorrect NN predictions, this study proposes using not just the NNs’ prediction of the single most likely strategy $\hat{\boldsymbol{\Delta}}_1$,

but rather to evaluate the costs of the $\Gamma > 1$ most likely strategies — each of which requires a fMPC solution $\hat{J}_r = \min_{\mathbf{x}, \mathbf{u}} J([\mathbf{X}^T \quad \mathbf{U}^T \quad \hat{\Delta}_r^T]^T; \mathbf{x}_0)$, $r = 1, 2, \dots, \Gamma$ — as well as CLQR cost J^{CLQR} , and choose the control force at each time step corresponding to the smallest cost. This approach ensures that the proposed algorithm performs at least as well as the CLQR algorithm regardless of the performance of the NNs. When the NN predictions are extremely accurate, the proposed real-time control algorithm will outperform the CLQR approximately just as much as the hMPC; in the worst case, when the NN predictions are undesirable, the proposed algorithm will perform very closely to the CLQR algorithm. Thus, the overall sub-optimality of the proposed algorithm is bounded by the CLQR algorithm. (While the CLQR algorithm is used herein, any other easily computed control algorithm, preferably with an analytical solution, could be used, *e.g.*, the one in Scruggs et al. (2007).)

For the numerical examples demonstrated in Section 5, given the excitations in Table 2, the percentage of time the NN does not output the optimal strategy ranges from less than 1% to 17%, and the percentage of time the CLQR control force vector is adopted (instead of the fMPC control force) ranges from less than 1% to 19% — note that, with the optimal strategy, the (approximate) fMPC performs better than the CLQR most of the time; when it does not, the fallback CLQR control force generally does not deviate much from the optimal semiactive control force, so the proposed algorithm performs almost identical to the hMPC (see numerical examples in Section 5).

4.3. Warm-start technique

For MPC, a “warm start” refers to initiating the optimization problem at the current time step using most of the optimal solution from the previous time step. Let $\underline{\mathbf{X}}^* = [\underline{\mathbf{x}}_1^T \ \underline{\mathbf{x}}_2^T \ \dots \ \underline{\mathbf{x}}_p^T]^T$ and $\underline{\mathbf{U}}^* = [\underline{\mathbf{u}}_0^T \ \underline{\mathbf{u}}_1^T \ \dots \ \underline{\mathbf{u}}_{p-1}^T]^T$ be the solutions from the previous time step; one warm-start is to use $[\underline{\mathbf{x}}_2^T \ \dots \ \underline{\mathbf{x}}_p^T \ \mathbf{0}_{1 \times 2n}]^T$ and $[\underline{\mathbf{u}}_1^T \ \dots \ \underline{\mathbf{u}}_{p-1}^T \ \mathbf{0}_{1 \times m}]^T$ as the initial guesses for the current time step’s \mathbf{X} and \mathbf{U} , respectively (Wright, 2000; Yildirim and Wright, 2002). Three drawbacks arise herein for this warm-start: first, $\underline{\mathbf{u}}_1$ might fail to satisfy the dissipativity constraints; second, letting the last n and m entries of the initial guesses of \mathbf{X} and \mathbf{U} take on $\mathbf{0}_{n \times 1}$ and $\mathbf{0}_{m \times 1}$, respectively, is purely heuristic with no mathematical justification (arguably one could instead use $\underline{\mathbf{x}}_p$ again or $\mathbf{A}\underline{\mathbf{x}}_p + \mathbf{B}_u \underline{\mathbf{u}}_{p-1}$, and $\underline{\mathbf{u}}_{p-1}$ again, but that last term has little effect); third, the sub-optimality of this initial guess cannot be estimated, so this warm-start technique cannot be used to estimate the initial κ . Thus, the typically used warm-start technique is not applicable herein. However, as noted in Wang and Boyd (2010), a warm-start technique can take other forms. In this study, at each time step, $\mathbf{x}_{k+1}^{\text{CLQR}}$ and $\mathbf{u}_k^{\text{CLQR}}$, $k = 0, 1, \dots, p-1$, used to generate the cost J^{CLQR} are taken as the initial guess for minimizing $J([\mathbf{X}^T \ \mathbf{U}^T \ \hat{\Delta}_r^T]^T; \mathbf{x}_0)$. Note that although $\mathbf{x}_{k+1}^{\text{CLQR}}$ and $\mathbf{u}_k^{\text{CLQR}}$, $k = 0, 1, \dots, p-1$, satisfy (5b)–(5g) for some Δ , they may not satisfy these inequalities for the NN predicted $\hat{\Delta}_1$, in which case the elements of $\mathbf{v}_k^{\text{relCLQR}} = \mathbf{V}\mathbf{x}_k^{\text{CLQR}}$ and $\mathbf{u}_k^{\text{CLQR}}$ causing the (5b)–(5g) violation(s) will be set to zero(s); note: when making these adjustments, state equation (5b) is not re-

quired to initially hold because the infeasible start Newton method adopted herein to solve the fMPC problem does not require the initial solution to satisfy equality constraints.

To evaluate the improvement provided to the proposed algorithm (NN+fMPC) by using this warm-start technique instead of the “cold start” initial guesses of $\mathbf{X} = \mathbf{0}$ and $\mathbf{U} = \mathbf{0}$, both are used on the first three numerical examples introduced in Section 5 when the structure is subjected to a realization of Gaussian white noise excitation. The fMPC parameters (*e.g.*, maximum iteration limit) are tuned so that the costs computed with the proposed algorithm differ by less than 5% from those with the hMPC algorithm. Table 1 summarizes the average computation time for running the hMPC as well as the proposed algorithm with both starting approaches, and the speedups achieved by the proposed algorithm. The warm-start technique proposed herein provides no significant improvement for the hMPC computation time (because of the branch-and-bound logic inherent in the baseline hMPC algorithm solver); in contrast, this warm-start speeds up the proposed NN+fMPC algorithm by 1.3–2.2 times compared with the zero cold start. Further, from Table 1, it is reasonable to expect a speedup of around one order of magnitude to be achieved by using the proposed algorithm (compared against the hMPC algorithm). For the SDOF system, the speedup provided by the NN+fMPC seems to be much greater than for the 3DOF and 5DOF examples; however, this is an artifact of the SDOF exploring only the first $\Gamma = 2$ most likely strategies, thus requiring two fMPC solutions per time

Table 1: Computation time (in milliseconds) for running one step of the hMPC and the NN+fMPC algorithms (one-controllable-damper examples)

	Cold start ($\mathbf{X} = \mathbf{0}, \mathbf{U} = \mathbf{0}$)			Warm-start using CLQR solution		
	Comp. Time (ms)		Time Ratio (Speedup)	Comp. Time (ms)		Time Ratio (Speedup)
	hMPC	NN+fMPC		hMPC	NN+fMPC	
SDOF	37.7	1.9	19.8	34.3	1.4	24.5
3DOF	64.6	15.3	4.2	56.5	6.9	8.2
5DOF	102.0	14.5	7.0	98.0	10.7	9.2

step, whereas the 3DOF and 5DOF examples solve $\Gamma = 4$ fMPC solutions; if this difference were eliminated, the SDOF NN+fMPC speedup would be of the same order of magnitude as the other two examples, particularly with the proposed warm start. Additionally, because multiple fMPC solutions are evaluated, there is additional room for a 2–4-fold increase in speedup if the fMPC solutions are pursued in parallel.

4.4. Formulating the fMPC problem at each time step

The C implementation of the fMPC introduced in Wang and Boyd (2010) handles a separable purely quadratic cost function with box constraints; this C code is generalized in two ways for this study. First, as mentioned previously, a convergence flag is output. Second, to accommodate the off-diagonal \mathbf{N} matrix in the cost function, a change of variable is used: let $\mathbf{u}'_k = \mathbf{u}_k + \mathbf{R}^{-1}\mathbf{N}^T\mathbf{x}_k$ and $\mathbf{U}' = [\mathbf{u}'_0^T \ \mathbf{u}'_1^T \ \dots \ \mathbf{u}'_{p-1}^T]^T$, $\mathbf{A}' = \mathbf{A} - \mathbf{R}^{-1}\mathbf{N}^T$ and $\mathbf{Q}' = \mathbf{Q} - \mathbf{NR}^{-1}\mathbf{N}^T$. Then, the optimization problem can be written

$$\min_{\mathbf{x}, \mathbf{U}'} J(\boldsymbol{\xi}'; \mathbf{x}_0) = \mathbf{x}_0^T \mathbf{Q}' \mathbf{x}_0 + \min_{\boldsymbol{\xi}'} \boldsymbol{\xi}'^T \mathbf{H} \boldsymbol{\xi}' \quad (7a)$$

$$\text{subject to } \boldsymbol{\xi}' = [\mathbf{X}^T \ \mathbf{U}'^T \ \hat{\Delta}_r^T]^T \text{ and } \mathbf{G} \boldsymbol{\xi}' = \mathbf{b}, \quad \bar{\mathbf{F}} \boldsymbol{\xi}' \leq \bar{\mathbf{h}} \quad (7b)$$

where the final two conditions impose the variable-changed state equation (5b) and the constraints (5c)–(5g). The matrices in (7) are detailed in Appendix F of Yu (2022).

4.5. Steps for real-time implementation of the proposed algorithm

Given the current state vector \mathbf{x}_0 , the following steps should be taken to find the control forces \mathbf{u}_0 for the current step using the proposed real-time semiactive control algorithm.

1. Based on \mathbf{x}_0 , run the CLQR in the background for p time steps and store J^{CLQR} , \mathbf{X}^{CLQR} and \mathbf{U}^{CLQR} .
2. For the $n \times 1$ input vector $\mathbf{x}_0 \|\mathbf{x}_0\|_2^{-1} \text{sgn } x_{0,n+1}$, let the P trained NNs make predictions (while not implemented for the numerical examples herein, these NNs can predict in parallel).
3. If the j^{th} ($1 \leq j \leq P$) NN provides γ_j possible predictions, then $\Gamma = \prod_{j=1}^P \gamma_j$ codewords are available; for each $r = 1, 2, \dots, \Gamma$, check a predefined look-up table for the corresponding strategy $\hat{\Delta}_r$.
4. If $v_{0,1}^{\text{rel}} = \bar{\mathbf{v}}^T \mathbf{x}_0 < 0$, let the value of each $\hat{\Delta}_r$ be replaced by $\mathbf{1}_{m(p-1) \times 1} - \hat{\Delta}_r$ (*i.e.*, its complement), $r = 1, 2, \dots, \Gamma$.
5. For each strategy $\hat{\Delta}_r$, formulate the fMPC optimization problem as in Section 4.4.
6. Determine κ based on J^{CLQR} .

7. For each strategy $\hat{\Delta}_r$, run the fMPC (while not implemented for the numerical examples herein, multiple fMPC algorithms can run in parallel) with initial solution guess provided by the warm-start technique: use ξ^{CLQR} directly if it satisfies (7b) or adjust it to so that it is in the space defined by (5b)–(5g). For the fMPC runs that converge, make a record of both $\hat{J}_r \equiv J([\mathbf{X}_r^T \ \mathbf{U}_r^T \ \hat{\Delta}_r^T]^T; \mathbf{x}_0)$ and the optimal zeroth step control force \mathbf{u}_0 .
8. Find the smallest value among $\{\hat{J}_1, \hat{J}_2, \dots, \hat{J}_\Gamma, J^{\text{CLQR}}\}$, selecting the corresponding control force vector to apply to the structure.

Note that, if the NNs are used to predict the active constraints as well as the binary variables $\hat{\Delta}_r$ — *i.e.*, strategies of the S_{II} type — then, instead of running fMPC in step 7, an optimization problem with only equality constraints would be solved through simple matrix manipulations at very high speed (Bertsimas and Stellato, 2019) (though the cost to generate samples and to train the NN is excessive, so S_{II} are not implemented for the examples herein).

5. Numerical examples

Four shear-building numerical examples are used to demonstrate the NN training and the tuning of the fMPC parameters, and to evaluate the performance of the proposed algorithm. In the first three examples, one controllable damper is installed between the ground and the first building level,

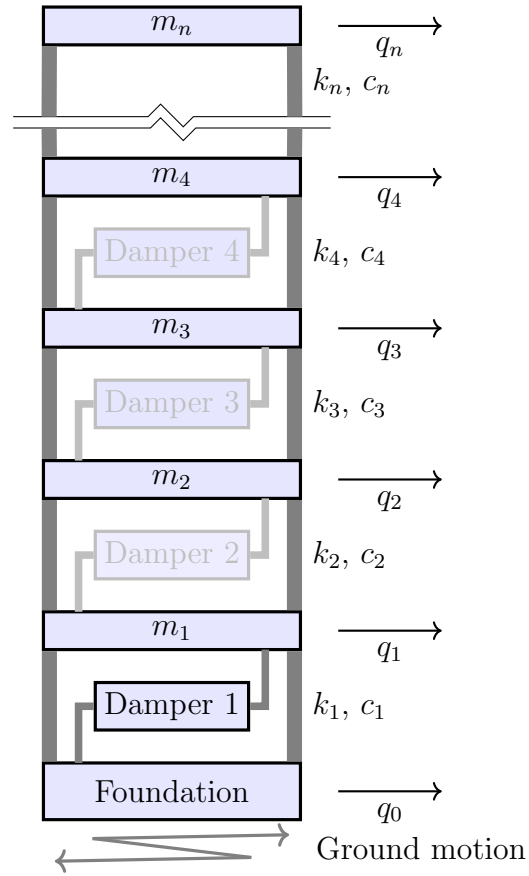


Figure 3: Multi-DOF shear building with multiple controllable dampers

as shown in Figure 3, for buildings with one-, three-, and five levels, respectively. In the last example, four controllable dampers are installed in stories 1–4, one in each story, of a five-story building (Figure 3). For brevity, only the five-story examples are presented in detail herein because the conclusions drawn from that structure are similar to those from the one- and three-story examples, for which the interested reader is directed to Yu (2022) for further details. In these examples: the displacements are ground-relative;

the excitation $\mathbf{w}_l = -\ddot{q}_l^g \mathbf{M} \mathbf{1}_{n \times 1}$ where \ddot{q}_l^g is the earthquake ground acceleration at time l ; the j^{th} column of $\hat{\mathbf{B}}_{\mathbf{u}}$ is $[-1 \quad \mathbf{0}_{1 \times (n-1)}]^T$ if $i = 1$ and $[\mathbf{0}_{1 \times (i-2)} \quad 1 \quad -1 \quad \mathbf{0}_{1 \times (n-i)}]^T$ for $i > 1$; the state and control force bounds are chosen sufficiently large that they are inactive; and the cross-damper velocity bounds are chosen to be $v_i^{\max} = -v_i^{\min} = x_{n+i}^{\max} - x_{n+i-1}^{\min}$ where $x_0^{\min} \equiv 0$. These examples use a serviceability goal, minimizing the absolute accelerations $\ddot{\mathbf{q}}_l^{\text{abs}} = \ddot{\mathbf{q}}_l + \ddot{q}_l^g \mathbf{1}_{n \times 1}$ (plus an optimal ρ -weighted penalty on control force) — *i.e.*, the cost function is $\tilde{J} = \Delta t \sum_{l=0}^{n_t} \|\ddot{\mathbf{q}}_l^{\text{abs}}\|_2^2 + \Delta t \sum_{l=0}^{n_t} \rho \|\mathbf{u}_l\|_2^2$ — resulting in weighting matrices

$$\begin{aligned} \mathbf{Q} &= [\mathbf{M}^{-1} \mathbf{K} \quad \mathbf{M}^{-1} \mathbf{C}]^T [\mathbf{M}^{-1} \mathbf{K} \quad \mathbf{M}^{-1} \mathbf{C}] \\ \mathbf{N} &= [\mathbf{M}^{-1} \mathbf{K} \quad \mathbf{M}^{-1} \mathbf{C}]^T \bar{\mathbf{B}}_{\mathbf{u}} \\ \mathbf{R} &= \bar{\mathbf{B}}_{\mathbf{u}}^T \bar{\mathbf{B}}_{\mathbf{u}} + \rho \mathbf{I}_m. \end{aligned} \tag{8}$$

In the first three examples, the control forces have reasonable magnitudes and need not be explicitly penalized, so ρ is set to zero.

The hMPC MIQP problems are solved using the Gurobi[®] solver (Gurobi, 2020) — with multi-core parallelism allowed by default) and programmed with its MATLAB[®] application programming interface (API). The fMPC QP problems are solved with the C implementation introduced in Wang and Boyd (2010) but adapted with the generalizations explained in Section 4. The computing platform used herein is a self-assembled workstation, with an Intel[®] i7-10700 eight-core processor clocked at 4.8 GHz (when boosted) and 96 GB of RAM (DDR4 2666 MHz) running MATLAB[®] R2020b (MathWorks, 2020) on a 64-bit Windows[®] 10 operating system.

The earthquake ground motion records used in the simulations, selected from the NGA-West 2 ground motion database (Ancheta et al., 2013) provided by the Pacific Earthquake Engineering Research (PEER) center, are summarized in Table 2; earthquakes Nos. 1–10 are far-field, and earthquakes Nos. 11–15/Nos. 16–20 are near-field without/with impulses.

Time-history analyses are performed using MATLAB’s `ode45` function with these excitations. The control forces are computed at each time step by the hMPC, the NN+fMPC, and the CLQR algorithms, respectively. A relative tolerance of 10^{-7} and an absolute tolerance of 10^{-9} are used in `ode45` to ensure the convergence of the peak absolute acceleration metric.

At each time step, the barrier coefficient κ is determined so that $\Delta J/J^{\text{CLQR}} \leq 5\%$, *i.e.*, $\kappa = (5\%/(2mp - m))J^{\text{CLQR}}$, which is $J^{\text{CLQR}}/780$ for the

Table 2: Information for Earthquake Records

No.	Name	Year	Station	Mag.	PGA [g]
1	Landers	1992	Yermo Fire Station	7.28	0.152
2	Hector Mine	1999	Hector	7.13	0.264
3	Kocaeli Turkey	1999	Duzce	7.51	0.310
4	Chi-Chi Taiwan	1999	CHY101	7.62	0.313
5	Imperial Valley-02	1940	El Centro Array #9	6.95	0.349
6	Imperial Valley-06	1979	El Centro Array #11	6.53	0.367
7	Kobe Japan	1995	Nishi-Akashi	6.90	0.483
8	Manjil Iran	1990	Abbar	7.37	0.515
9	Northridge-01	1994	Beverly Hills-14145 Mulhol	6.69	0.601
10	Duzce Turkey	1999	Bolu	7.14	0.739
11	Chi-Chi Taiwan	1999	TCU054	7.62	0.140
12	Christchurch New Zealand	2011	Christchurch Botanical Gardens	6.20	0.141
13	Gazli USSR	1976	Karakyr	6.80	0.599
14	Nahanni Canada	1985	Site 1	6.76	0.945
15	Cape Mendocino	1992	Cape Mendocino	7.01	1.494
16	Irpinia Italy-01	1980	Bagnoli Irpinio	6.90	0.190
17	Superstition Hills-02	1987	Parachute Test Site	6.54	0.368
18	Loma Prieta	1989	Saratoga-Aloha Ave	6.93	0.512
19	Niigata Japan	2004	NIGH11	6.63	0.598
20	Chi-Chi Taiwan	1999	TCU065	7.62	0.761

Note: “Mag.” = Moment Magnitude; $g = 9.81 \text{ m/s}^2$ is the gravitational acceleration.

first three examples and $J^{\text{CLQR}}/1520$ for the fourth example (in which four controllable dampers are used). The iteration limit for the infeasible-start Newton method, which is used to solve the optimization problem formulated by the primal-barrier method (Wang and Boyd, 2010), is set to be 13 for the four examples considered herein, so that proposed algorithm (NN+fMPC) performs almost the same as the hMPC algorithm (further increasing this iteration limit does not lead to noticeable control performance improvements).

The simulation results include the cost \tilde{J} , defined in (3), and the peak and root-mean-square (RMS) metrics of inter-story drifts, absolute accelerations, and (normalized by the building weight $W = \sum_{i=1}^n m_i g$) controllable damper forces. The i^{th} inter-story drift at time l is defined as $d_{l,i} = q_{l,i} - q_{l,i-1}$, $i = 1, 2, \dots, n$, where $q_{l,0} \equiv 0$. The RMS metrics of an $n \times 1$ vector response $\mathbf{y}_l = [y_{l,1} \ y_{l,2} \ \dots \ y_{l,n}]^T$ over times $l = 0, 1, \dots, n_t$ are $y_i^{\text{rms}} = [n_t^{-1} \sum_{l=0}^{n_t} y_{l,i}^2]^{1/2}$ and $y^{\text{rms}} = [n^{-1} n_t^{-1} \sum_{i=1}^n \sum_{l=0}^{n_t} y_{l,i}^2]^{1/2}$. The corresponding peak metrics are $y_i^{\text{peak}} = \max_l |y_{l,i}|$ and $y^{\text{peak}} = \max_{i,l} |y_{l,i}|$. The RMS control force is computed slightly differently, only counting those time steps with nonzero control forces: $u^{\text{rms}} = \left[\sum_{j=1}^m \sum_{l=0}^{n_t} u_{l,j}^2 I_{>0}(|u_{l,j}|) \right]^{1/2} \left[\sum_{j=1}^m \sum_{l=0}^{n_t} I_{>0}(|u_{l,j}|) \right]^{-1/2}$ for $j = 1, 2, \dots, m$, where $I_{>0}(\cdot)$ is an indicator function that is unity when the argument is nonzero (in practice here, a threshold of 10^{-16} N is used) and zero otherwise; thus, u^{rms} is the RMS of the control forces when activated (nonzero).

Table 3: Average, largest and smallest performance improvements of the NN+fMPC relative to the CLQR in terms of the cost and the peak and RMS metrics of inter-story drifts, absolute accelerations, and controllable damper forces when the structures are subjected to 20 historical earthquake records

	Average / Largest / Smallest Improvement [%]				
	SDOF (one device)	3DOF (one device)	5DOF (one device)	5DOF (four devices)	5DOF (four devices) (force saturated)
d^{rms}	-30 / -39 / -25	-25 / -36 / -16	-25 / -36 / -14	-4 / -7 / -1	-4 / -7 / -2
\ddot{q}^{abs}	-12 / -16 / -1	-15 / -25 / -5	-18 / -30 / -4	-18 / -23 / -8	-18 / -23 / -9
u^{rms}	-42 / -49 / -32	-25 / -38 / -5	-22 / -32 / -10	-8 / -11 / -4	-8 / -11 / -4
cost	-22 / -30 / -2	-27 / -44 / -10	-31 / -50 / -8	-25 / -34 / -12	-25 / -34 / -12
d^{peak}	-25 / -33 / -9	-19 / -34 / -2	-18 / -33 / 0	0 / -6 / 7	0 / -6 / 7
\ddot{q}^{abs}	-24 / -32 / -7	-19 / -46 / 18	-22 / -54 / 8	-40 / -68 / -2	-38 / -65 / 3
u^{peak}	-24 / -33 / 13	-15 / -42 / 12	-15 / -39 / 10	0 / -11 / 29	-3 / -11 / 0

Note: This table shows the percent changes of the values computed from the NN+fMPC relative to those computed from the CLQR; negative numbers denote improvements. “w/o” and “w/” are abbreviations for “without” and “with”, respectively.

5.1. Five degree of freedom system with one controllable damper

This model represents a base-isolated building with a four-story superstructure, with base mass $m_1 = 150$ Mg, superstructure masses $m_2 = m_3 = m_4 = m_5 = 100$ Mg, isolation-layer stiffness $k_1 = 6.546$ MN/m, and superstructure stiffnesses $k_2 = k_3 = k_4 = 32.730$ MN/m and $k_5 = 16.365$ MN/m, resulting in natural frequencies 0.50 Hz, 1.67 Hz, 2.79 Hz, 4.15 Hz and 5.32 Hz. Rayleigh damping is applied, for damping ratios 1.58%, 2.00%, 3.00%, 4.31% and 5.45%, respectively. $\rho = 10^{-16}$ is used in the control weight matrix \mathbf{R} in (4) to avoid numerical issues due to finite precision computation. The sampling frequency is set to be 50 Hz, to which all the earthquake records used in the numerical simulations are decimated. The hMPC horizon is set to $p = 20$ time steps, which provides much better performance than the CLQR while not running unnecessarily slow.

To train the NNs offline to predict $S_I = \mathbf{\Delta}^*$, around 7.1 million system state vectors (\mathbf{x}_0) are generated (as per Section 3.2) so that, by Theorem 9 in McAllester and Schapire (2000), the probability of finding a new strategy with additional samples is less than 0.5%. Solving the MIQP for each leads to 13,403 unique strategies. The NNs are trained using the Adam optimization algorithm (Kingma and Ba, 2014) in the PyTorch software package (Paszke et al., 2019) with initial learning rate 0.0005. The number of samples necessary for a small possibility of missing a strategy is more than is needed for high NN accuracy; thus, only 10% of the samples corresponding to each strategy are used for NN training and another 20% used as testing data, resulting in a sufficient NN prediction accuracy of about 95% (when evaluated with the training data and the testing data).

Each of the two trained NNs has one input layer with ten neurons, five hidden layers each with 200 neurons, and one output layer with 113 and 127 neurons, respectively. When making predictions, the best two predictions of each NN will be taken; thus, the fMPC algorithm must be run $\Gamma = 2 \cdot 2 = 4$ times at each time step (or, occasionally, fewer if some predictions do not correspond to valid strategies).

In terms of RMS responses (including the cost), as summarized in Table 3 and detailed in Figures 4a–4d, the NN+fMPC outperforms the CLQR without exception. In terms of peak responses, the NN+fMPC leads to decreases in peak drifts (shown in Figure 4e) without exception. The NN+fMPC generally leads to the decreases of peak accelerations $\ddot{q}_{\text{peak}}^{\text{abs}}$ (shown in Fig-

ure 4f), except for the small increase when the structure is subjected to earthquake No. 7 (< 1% increase), No. 9 (4% increase), No. 11 (1.5% increase), No. 12 (8.3% increase) or No. 15 (4.7% increase) — all because the hMPC either performs similarly to [earthquake No. 11 (10.5% decrease) or No. 15 (8.3% decrease)] or slightly worse than [earthquake No. 7 (2.7% increase), No. 9 (< 1% increase) or No. 12 (12.0% increase)] the CLQR in these cases. Again, because the objective is to minimize the RMS responses, some increases in the peak responses are possible. Further, the increases of $\ddot{q}_{\text{peak}}^{\text{abs}}$ when the structure is controlled by either the hMPC or the NN+fMPC usually occur on the level above the isolation layer because of the swift changes in control force level when going from one time step to another. In reality, when the control force is commanded to change gradually within the $\Delta t = 0.02$ second time step, these acceleration increases will be alleviated.

As shown in Figure 4g, small increases in peak force u^{peak} occur when the hMPC uses peak forces slightly larger than [earthquake No. 1(9.1% increase) or No. 7 (9.3% increase)] or similar to [earthquake No. 11 (< 1% decrease)] the CLQR. When the larger control forces commanded by the NN+fMPC are saturated to the same level as the CLQR's u^{peak} , the other metrics' performance improvements are, again, not visibly compromised. Further, the peak control forces are generally less than 20% of the total weight of the building except when the structure is subjected to earthquake No. 20, in which case a large-amplitude impulse leads to a huge control demand — very challenging for all control algorithms to handle.

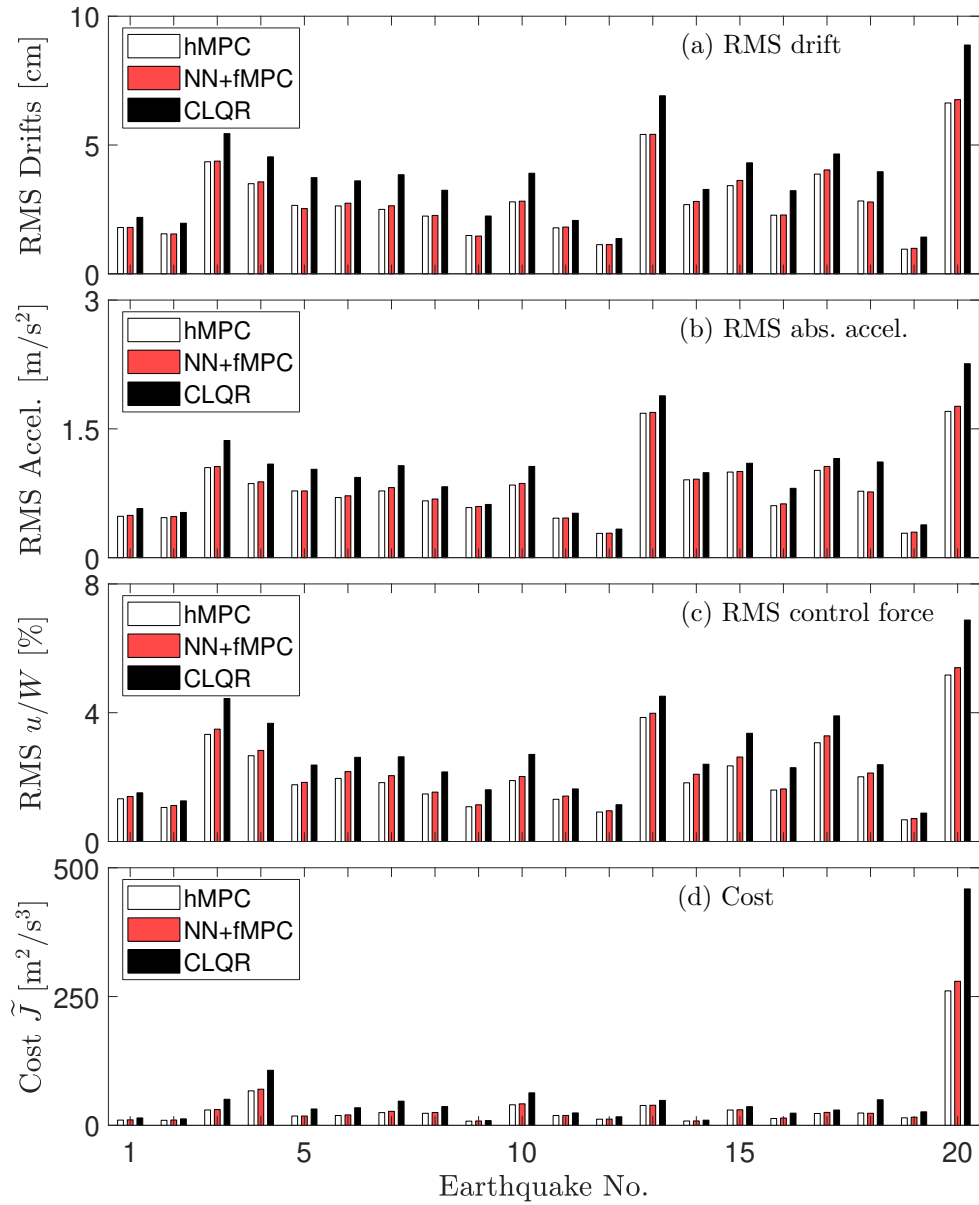


Figure 4: Comparisons of the costs and the RMS and peak response and control force metrics calculated by the hMPC, the NN+fMPC and the CLQR algorithms using 20 historical earthquake records (5DOF system with one controllable damper)

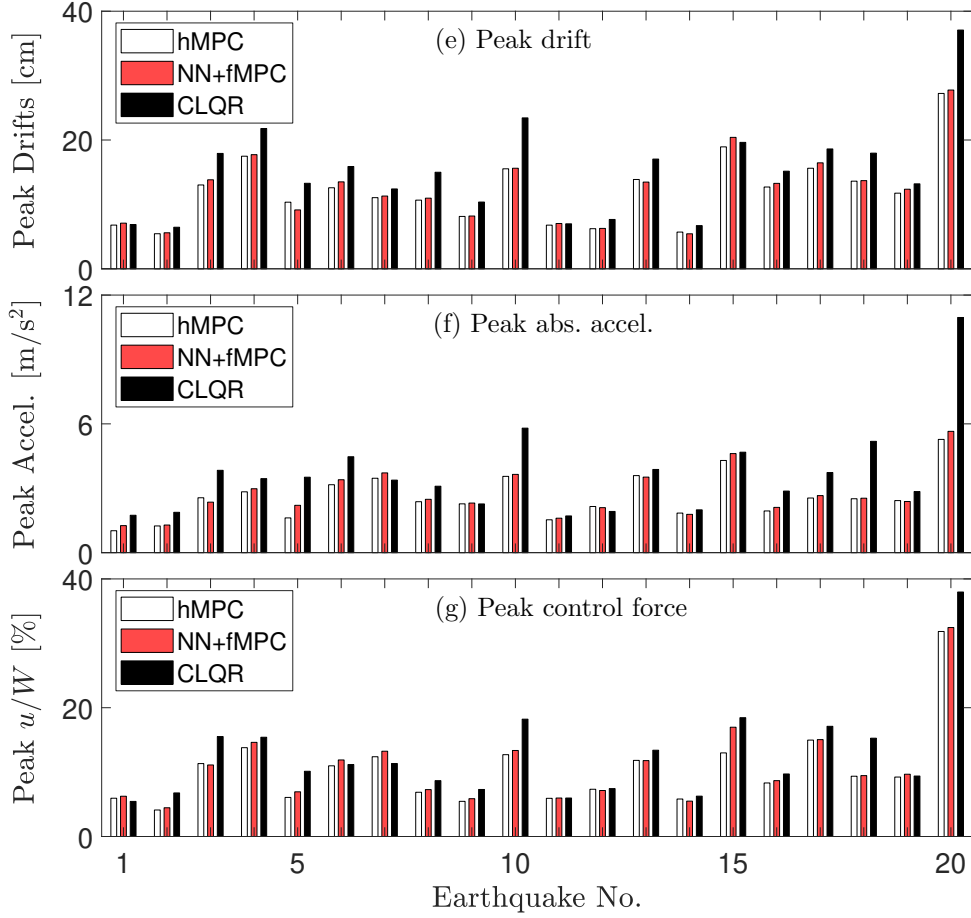


Figure 4: Comparisons of the costs and the RMS and peak response and control force metrics calculated by the hMPC, the NN+fMPC and the CLQR algorithms using 20 historical earthquake records (5DOF system with one controllable damper) (cont.)

To demonstrate sample time histories, the isolation-layer drifts, the roof absolute accelerations and the control forces (normalized by the building weight), when the structure is subjected to the 1940 El Centro earthquake ground motion record (earthquake No. 5), are shown in Figure 5, in which it can be easily observed that the NN+fMPC resembles the hMPC and well outperforms the CLQR.

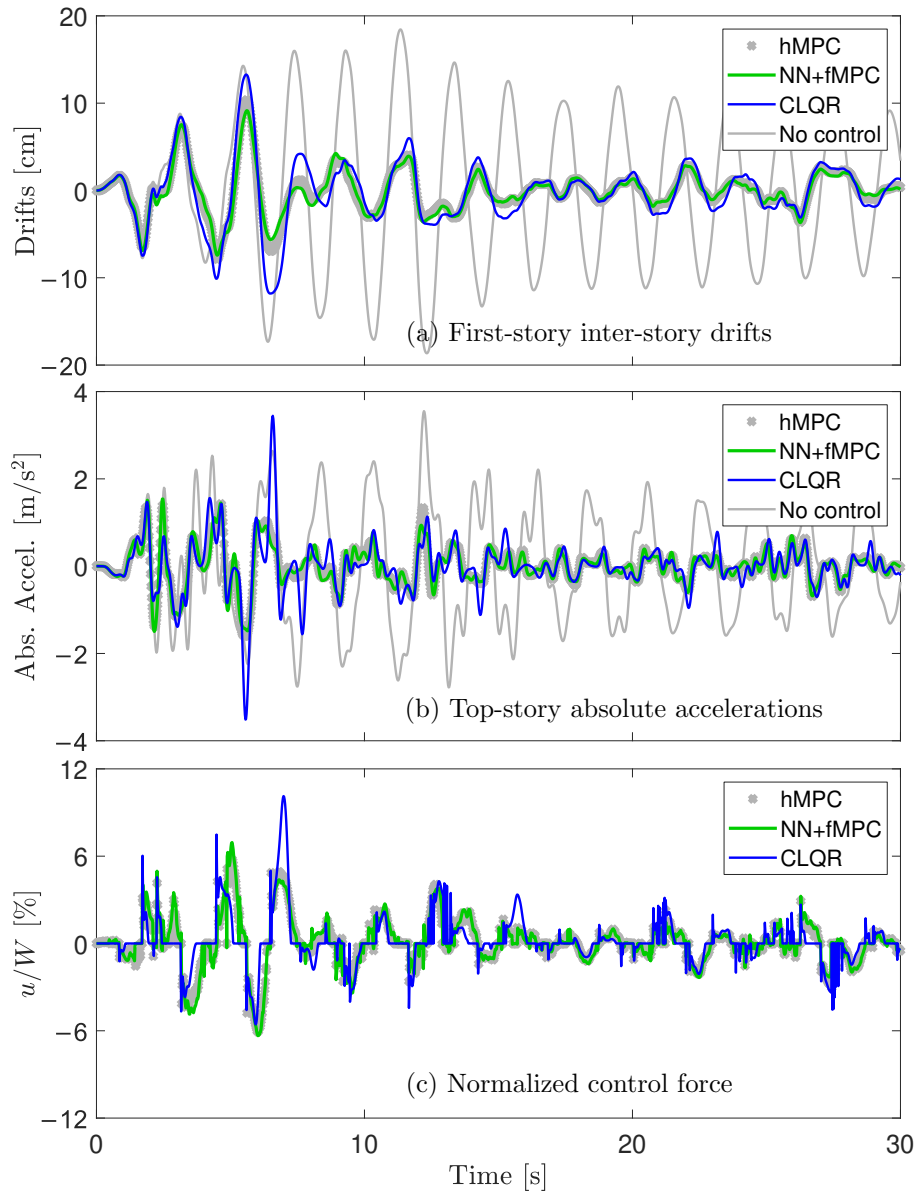


Figure 5: Comparisons of the first story inter-story drifts, the top story absolute accelerations and the control forces computed by the hMPC, the NN+fMPC and the CLQR (5DOF building subjected to earthquake No. 5)

Aside from the earthquake ground motions listed in Table 2, the 5DOF building model was also subjected to a Gaussian white noise excitation as well as Kanai-Tajimi stationary models of earthquake-induced ground accelerations (Soong and Grigoriu, 1993; Johnson et al., 1995, 1997; Ramallo et al., 2002) with the fundamental frequency of the ground model ranging from 1 Hz to 5 Hz. In all of these cases, the NN+fMPC performance improvements relative to the CLQR are consistent with the historical earthquakes, so the performance metrics of these stochastic responses are omitted for brevity.

Finally, a study (Yu, 2022) of the robustness of control performance for the CLQR, the NN+fMPC and the hMPC algorithms demonstrated that the NN+fMPC algorithm is as robust as the hMPC algorithm, and that both of them consistently outperform the CLQR algorithm.

5.2. Five degree of freedom system with four controllable dampers

Some structures with some control objectives require multiple controllable dampers. In this example, the building model is a fixed-base 5DOF structure, with one controllable damper in each of the first four stories, as shown in Figure 3, with all floor masses 100 Mg and all story stiffnesses 48.727 MN/m, respectively, resulting in natural frequencies 1.00 Hz, 2.92 Hz, 4.60 Hz, 5.91 Hz and 6.74 Hz. Rayleigh damping is applied, resulting in damping ratios 2.00%, 2.52%, 3.56%, 4.43% and 5.00%. $\rho = 10^{-11}$ so that the maximum CLQR control force when the building is subjected to 1940 El Centro earthquake (earthquake No. 5) is 11.84% of the building weight. The number of hMPC horizon steps is chosen to be ten, *i.e.*, $p = 10$, leading to much better control

performance than that of the CLQR. Note that while $p = 20$ was used in the single-damper cases, preliminary studies showed that using $p > 10$ does not lead to significant hMPC control performance improvement for this example.

Each of the two trained NNs has one input layer with ten neurons, five hidden layers each with 500 neurons, and one output layer with 347 and 349 neurons, respectively. When making predictions, the best two predictions of each NN will be taken; thus, the fMPC algorithm must be run at most $\Gamma = 2 \cdot 2 = 4$ times during each time step. To train these NNs offline, to predict $S_I = \Delta^*$, using enough samples to make the probability of finding a new strategy less than 0.5%, as in the previous examples, would require 40.6 million system state vectors resulting in 248,755 unique strategies, posing a challenge for computer memory during training if only two large NNs are used and (because more NNs may require a larger number Γ of predictions to ensure a low level of incorrect-NN-prediction-induced suboptimality) longer online computation time for the multiple (Γ) serial fMPC runs (a future parallel implementation would permit a larger number of smaller NNs). Instead, the likelihood of missing a strategy is raised to 1%, requiring around 10.1 million system state vectors (\mathbf{x}_0) that result in 117,295 unique strategies.

The NN+fMPC performance improvements (both as normal and saturating damper forces to CLQR’s level with the same excitation) relative to the CLQR are summarized in Table 3. Without force saturation, all structure performance metrics are improved; the peak control force u^{peak} on average is comparable to that of the CLQR but is modestly larger for some earthquakes;

the significant improvements in $\ddot{q}_{\text{rms}}^{\text{abs}}$ and $\ddot{q}_{\text{peak}}^{\text{abs}}$ indicate that the NN+fMPC is much better at attenuating the accelerations (the main goal of the control design) than the CLQR; the inter-story drifts (see the rows for d^{rms} and d^{peak}) are additionally modestly reduced. These improvements are not compromised after imposing force saturation, indicating that the NN+fMPC — as well as the hMPC — achieve better control performance by using the controllable dampers efficiently, not by using higher control force levels, which is echoed by the u^{rms} improvements without and with force saturation.

Finally, the computational cost of the NN+fMPC is around 7 ms per time step, 13 times faster than that of the hMPC (taking 95 ms per time step to run), and also faster than what was achieved in the one-controllable-damper 5DOF example, because there are fewer parameters for the fMPC to optimize — $(n + m)p = (10 + 4)10 = 140$ parameters herein against $(n + m)p = (10 + 1)20 = 220$ parameters in the previous one-controllable-damper 5DOF example — due to half the number of prediction steps.

6. Conclusions and future research

The proposed semiactive structural vibration control algorithm is proven to perform almost as well as the hMPC and runs sufficiently fast for real-time implementation for typical civil engineering structures, by reducing the computational complexity of the MIQP problem using NN predicted integer variables' values. The success of this algorithm relies on a skillful selection of the sampling space for the system state vector \mathbf{x}_0 , the use of the “codeword”

approach to reduce NN complexities, a novel technique to tune the fMPC penalty coefficient κ at each time step, a novel application of the warm-start technique, the proposed way of bounding the sub-optimality (either from inaccurate NN predictions or the QP solver) and advanced programming (efficient manipulations of matrices and exploitations of possible parallelisms).

Even without exploiting the possible parallelisms, the proposed algorithm is applicable to the real-time vibration control of 5DOF systems; the computation time reduces from about 0.1 s — which is too slow relative to the dynamics of typical civil structures — to about 0.01 s. When parallelisms are fully exploited (making this algorithm roughly 3–4 times faster) in future theoretical and experimental studies, it will be able to accommodate even more complex structural models (with around ten DOFs). Further, for even larger-scale structures, a decentralized strategy can be exploited in the future studies to further the application of the proposed algorithm. Finally, future research that can further reduce the sample generation computational effort (currently, the sample generation process, *i.e.*, solving the MIQP problems given the randomly generated initial state vectors, preferably requires high performance computing facilities) and that of the fMPC are welcomed.

Acknowledgments

This work was partially supported by the National Science Foundation, under awards CMMI 14-36018 and 16-63667. The first author also gratefully acknowledge the support of a Provost’s Fellowship from the University

of Southern California. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or the USC. The USC CARC and the UC Berkeley PEER center are also gratefully acknowledged for the computational resources and the earthquake records used herein, respectively.

References

- Allwein, E.L., Schapire, R.E., Singer, Y., 2000. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research* 1, 113–141.
- Ancheta, T.D., Darragh, R.B., Stewart, J.P., Seyhan, E., Silva, W.J., Chiou, B.S., Wooddell, K.E., Graves, R.W., Kottke, A.R., Boore, D.M., Kishida, T., Donahue, J.L., 2013. PEER NGA-West2 Database. Technical Report. Pacific Earthquake Engineering Research Center.
- Bemporad, A., 2002. An efficient technique for translating mixed logical dynamical systems into piecewise affine systems, in: *Proceedings of the IEEE Conference on Decision and Control*, Las Vegas, NV, USA.
- Bemporad, A., Morari, M., 1999. Control of systems integrating logic, dynamics, and constraints. *Automatica* 35, 407–427.
- Bertsimas, D., Stellato, B., 2019. Online mixed-integer optimization in milliseconds. arXiv preprint arXiv:1907.02206 .

- Bertsimas, D., Stellato, B., 2021. The voice of optimization. *Machine Learning* 110, 249–277.
- Boyd, S., Vandenberghe, L., 2004. *Convex Optimization*. Cambridge University Press, Cambridge, U.K.
- Cauligi, A., Culbertson, P., Stellato, B., Bertsimas, D., Schwager, M., Pavane, M., 2020. Learning mixed-integer convex optimization strategies for robot planning and control, in: *Proceedings of the IEEE Conference on Decision and Control*, Jeju Island, Republic of Korea.
- Dietterich, T.G., Bakiri, G., 1994. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research* 2, 263–286.
- Elhaddad, W.M., Johnson, E.A., 2013. Hybrid MPC: An application to semiactive control of structures, in: *Conference Proceedings of the Society for Experimental Mechanics Series*, New York, NY, USA.
- Gurobi, 2020. *Gurobi optimizer reference manual (Version 9.0)*. Gurobi Optimization, LLC, Beaverton, OR, USA.
- Harris, D.M., Harris, S.L., 2012. *Digital design and computer architecture*. Second edition ed., Morgan Kaufmann, Waltham, MA, USA.
- Heemels, W.P.M.H., De Schutter, B., Bemporad, A., 2001. On the equivalence of classes of hybrid dynamical models, in: *Proceedings of the IEEE Conference on Decision and Control*, Orlando, FL, USA.

- Johnson, E.A., Wojtkiewicz, S.F., Bergman, L.A., 1995. Some experiments with massively parallel computation for Monte Carlo simulation of stochastic dynamical systems, in: Spanos, P.D. (Ed.), *Computational Stochastic Mechanics*, Balkema, Rotterdam, Netherlands. pp. 325–336.
- Johnson, E.A., Wojtkiewicz, S.F., Bergman, L.A., Spencer, Jr., B.F., 1997. Observations with regard to massively parallel computation for Monte Carlo simulation of stochastic dynamical systems. *International Journal of Non-Linear Mechanics* 32, 721–734. Third International Stochastic Structural Dynamics Conference.
- Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 .
- MathWorks, 2020. *MATLAB User’s Guide (R2020a)*. The MathWorks Inc., Natick, Massachusetts, USA.
- McAllester, D.A., Schapire, R.E., 2000. On the convergence rate of Good-Turing estimators, in: *Proceedings of the 13th Annual Conference on Computational Learning Theory*, Palo Alto, CA, USA.
- Mei, G., Kareem, A., Kantor, J.C., 2001. Real-time model predictive control of structures under earthquakes. *Earthquake Engineering & Structural Dynamics* 30, 995–1019.
- Mei, G., Kareem, A., Kantor, J.C., 2002. Model predictive control of struc-

- tures under earthquakes using acceleration feedback. *Journal of Engineering Mechanics* 128, 574–585.
- Mei, G., Kareem, A., Kantor, J.C., 2004. Model predictive control of wind-excited building: Benchmark study. *Journal of Engineering Mechanics* 130, 459–465.
- Passerini, A., Pontil, M., Frasconi, P., 2004. New results on error correcting output codes of kernel machines. *IEEE Transactions on Neural Networks* 15, 45–54.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S., 2019. Pytorch: An imperative style, high-performance deep learning library, in: Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché Buc, F., Fox, E., Garnett, R. (Eds.), *Advances in Neural Information Processing Systems* 32. Curran Associates, Inc., pp. 8024–8035.
- Qin, S.J., Badgwell, T.A., 2003. A survey of industrial model predictive control technology. *Control Engineering Practice* 11, 733–764.
- Ramallo, J.C., Johnson, E.A., Spencer, Jr., B.F., 2002. “Smart” base isolation systems. *Journal of Engineering Mechanics* 128, 1088–1099.
- Scruggs, J.T., Taflanidis, A.A., Iwan, W.D., 2007. Non-linear stochastic controllers for semiactive and regenerative systems with guaranteed quadratic

- performance bounds-Part 2: Output feedback control. *Structural Control and Health Monitoring* 14, 1121–1137.
- Sontag, E., 1981. Nonlinear regulation: The piecewise linear approach. *IEEE Transactions on Automatic Control* 26, 346–358.
- Soong, T.T., Grigoriu, M., 1993. Random vibrations of mechanical and structural systems. Prentice Hall, Englewood Cliffs, New Jersey, USA.
- Wang, Y., Boyd, S., 2010. Fast model predictive control using online optimization. *IEEE Transactions on Control Systems Technology* 18, 267–278.
- Wright, S.J., 1997. Applying new optimization algorithms to model predictive control. *Chemical Process Control-V* 93, 147–155.
- Wright, S.J., 2000. Interior-point methods. *Journal of Computational and Applied Mathematics* 124, 281–302.
- Yildirim, E.A., Wright, S.J., 2002. Warm-start strategies in interior-point methods for linear programming. *SIAM Journal on Optimization* 12, 782–810.
- Yu, T., 2020. Novel model updating approaches and machine learning based semiactive model predictive control algorithms. Technical Report. Ph.D Dissertation Proposal 28 August 2020, University of Southern California. Ph.D Dissertation Proposal, University of Southern California.

Yu, T., 2022. Novel multi-stage and CTLS-based model updating methods and real-time neural network-based semiactive model predictive control algorithms. Ph.D Dissertation, University of Southern California.

Zhang, Q., Lee, K.C., Bao, H., You, Y., Li, W., Guo, D., 2018. Large scale classification in deep neural network with label mapping, in: Proceedings of the IEEE International Conference on Data Mining Workshops, Singapore, Singapore.