aerospace

MDPI

Article

Equilibrium-Consistent Reservoir Prediction and Risk-Adaptive Model Predictive Control for Nonlinear Wing-Flutter Suppression

Hang Zhang [1], Jiani Xu [1], Yang Li [1,*] and Xianbin Liu [2]

[1] School of Automation, Nanjing University of Science and Technology, Nanjing 210094, China; 124110023421@njust.edu.cn (H.Z.); xujiani@njust.edu.cn (J.X.)
[2] State Key Laboratory of Mechanics and Control of Mechanical Structures, College of Aerospace Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China; xbliu@nuaa.edu.cn
* Correspondence: liyangbx5433@163.com

Abstract

We investigate equilibrium-consistent reservoir-computing (RC) prediction and adaptively weighted model predictive control (MPC) for a nonlinear pitch–plunge flutter benchmark at $Q_f = 8$. A conditional residual-mean network compensates for RC prediction errors, and a post-training correction enforces the known zero equilibrium. The optimizer uses deterministic corrected-mean prediction; K-nearest-neighbor (KNN) residual-error covariance adjusts state and terminal penalties, without stochastic scenario propagation. Against separately tuned fixed weights, KNN-based weighting reduced normalized integral squared error (ISE) by 1.66% in a reference Gaussian-disturbance setting with five paired realizations. Four systematic noise conditions included independent and synthetic AR(1) disturbances. An additional exploratory comparison on two new paired Gaussian noise realizations used a fully KNN-independent state-feedback weight scheduler. This simpler scheduler lowered mean normalized ISE (0.151199 versus 0.155306 for KNN) but required more input effort (1.579680 versus 1.231068) and showed slightly larger late-stage normalized root mean square error (0.008832 versus 0.008410). Thus, the comparison exposes a regulation–effort trade-off but does not establish an incremental advantage specifically attributable to KNN covariance information. The KNN trace remains an empirical weighting signal rather than calibrated uncertainty. These results are limited to offline numerical simulations, without real-time or stochastic stability guarantees.

Keywords: wing flutter; reservoir computing; equilibrium consistency; model predictive control; KNN risk estimation; adaptive weighting; correlated process noise

Check for updates

Academic Editor: Bosko Rasuo

Received: 14 September 2026
Revised: 23 October 2026
Accepted: 30 October 2026
Published: 5 November 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Wing flutter is a typical aeroelastic instability phenomenon caused by the coupling among structural elasticity, aerodynamic forces, and inertial forces [1]. When an aircraft operates near the critical dynamic pressure, or when it is affected by external disturbances, parameter variations, and modeling errors, the system response may exhibit sustained oscillations or even divergence. Therefore, wing-flutter suppression is an important problem in aircraft structural control and aeroelastic stability research.

Traditional flutter control methods usually rely on relatively accurate dynamic or reduced-order aeroelastic models [2]. For real wings or wing section systems, however, an accurate model is often difficult to obtain. On the one hand, structural, aerodynamic, and damping parameters may be uncertain. On the other hand, the real system may be affected

by process noise, unmodeled dynamics, and external disturbances. If a controller relies completely on a fixed nominal model, then large model mismatch can directly degrade prediction accuracy and control performance and may even reduce closed-loop stability. Aircraft structural aeroelastic analysis and active control have also been extensively studied, including nonlinear aerodynamic effects, structural nonlinearities, control loop coupling, active flutter suppression, and gust load alleviation [3,4].

Data-driven reduced-order modeling has also been introduced into aeroelastic analysis and control, where low-order models are learned from simulation or experimental data to support real-time prediction and control design [5,6]. In particular, data-driven models have been developed for unsteady aerodynamic prediction, aeroelastic reduced-order modeling, and efficient flow–structure stability analysis [7–9]. In recent years, data-driven modeling has provided new ideas for nonlinear-system control. Recent studies on learning-based model predictive control (MPC) and data-driven MPC have shown that learned prediction models can be combined with constrained predictive control to improve performance under model uncertainty [10,11]. Among these methods, reservoir computing (RC) models, especially echo state networks, have a simple structure, fast training speed, and strong capability for representing nonlinear dynamics [12,13], which makes them suitable for state prediction in complex dynamical systems. This structure follows the reservoir computing principle, where the recurrent reservoir is fixed, and only the readout layer is trained. Compared with deep recurrent neural networks, RC models have lower training costs and are more suitable for embedding in a receding-horizon optimization framework.

A single RC model, however, usually provides deterministic point predictions whose residual errors vary across the state–control domain. Conditional residual regression can compensate for the mean of these errors, whereas local covariance estimates can help identify regions where the predictor is less reliable. Diffusion probabilistic models and conditional time-series forecasting provide additional research directions for representing non-Gaussian residual distributions [14–17]; however, the closed-loop results reported here use the conditional residual mean,not diffusion-generated stochastic prediction scenarios. This distinction is important because independently sampled scenarios would require a different objective and different computational evidence.

Particle-based MPC formulations such as in Dyro et al. [18] address prediction under uncertainty through particle propagation. By contrast, the implementation validated here propagates one deterministic conditional-mean trajectory and does not approximate uncertainty by scenario particles. The assessed contribution is therefore limited to equilibrium-anchored prediction and empirical Q/P weight scheduling, including a direct exploratory comparison with a fully KNN-independent adaptive-weight baseline; no advantage uniquely attributable to KNN covariance or new particle-sampling scheme is claimed.

A second difficulty arises near a known equilibrium. An unconstrained learned state-transition map need not preserve the physical equilibrium $F(0,0) = 0$. Even a small model offset can generate a nonzero regulated position in model predictive control. Here, the conditional-mean predictor is anchored at the physical zero equilibrium without retraining the original networks. The KNN covariance of residual-mean prediction errors is then used as a local reliability proxy to adjust the state and terminal MPC penalties. This risk measure does not directly estimate exogenous turbulence intensity or imply a chance-constraint guarantee. Moreover, improvements over fixed Q/P penalties cannot alone establish the necessity of KNN-based covariance information; we therefore report an exploratory KNN-independent state-feedback weight-scheduling ablation.

The primary contributions of this study are as follows:

1. An RC plus conditional residual-mean predictor is constructed for a nonlinear pitch–plunge benchmark, and an equilibrium-anchoring correction enforces the known

zero-state equilibrium of the one-step prediction map. Its effect is compared against the uncorrected predictor in a separate ablation, and the three prediction variants are evaluated on independent one-step and freely rolled multi-step test trajectories.

2. A KNN estimate of local residual-mean prediction-error covariance is used to modulate state and terminal weighting matrices in a deterministic mean-prediction MPC problem. The control-increment weight remains fixed; independent diffusion samples, risk-adaptive scenario counts, and risk-dependent constraint tightening are not active in the reported optimizer.
3. The controller is compared to a separately tuned fixed-Q/P comparator under paired disturbances across two independent-Gaussian and two synthetic AR(1) noise conditions (five held-out seeds per condition). A separate exploratory white-noise ablation on two new paired seeds additionally compares KNN weighting with an independently defined, KNN-free state-feedback adaptive-Q/P scheduler, and both regulation errors and input effort are reported. The limited ablation does not establish statistical or KNN-specific superiority.
4. The analysis retains the original nominal open-loop Hopf/LCO characterization and analytically resolves the additional symmetry-breaking equilibrium branches at $Q_f = 50/7$. An independent Jacobian eigenvalue analysis shows that all three nominal equilibria are unstable at $Q_f = 8.0$, without inferring their stability from the periodic orbit's Floquet multipliers.
5. The parameter-perturbed plant is independently examined through 120-model-time-unit unforced simulations and an illustrative 24-model-time-unit controlled realization, complementing the systematic eight-model-time-unit paired comparisons.

The remainder of this paper is organized as follows: Section 2 describes the nonlinear pitch–plunge plant and its equilibria. Section 3 presents the RC residual-mean predictor, equilibrium correction, KNN error-covariance risk, and weight-adaptive MPC. Section 4 presents independent one-step/multi-step predictor validation, paired numerical tests under white and correlated process noise, and an exploratory KNN-independent adaptive-weighting ablation, including strict sustained-regulation outcomes and computational limitations. Section 5 summarizes the conclusions and limitations.

2. System Description and Problem Formulation

2.1. Two-Degree-of-Freedom Wing-Flutter System

A two-degree-of-freedom pitch–plunge section is a standard low-order benchmark for bending–torsion coupling and nonlinear flutter control [2,19]. Pitch describes rotation about the elastic axis, and plunge describes vertical translation. The model is intended for numerical validation rather than full-aircraft aeroelastic prediction.

2.2. Nonlinear Pitch–Plunge Benchmark Plant

The nominal vector field used here follows the nondimensional two-degree-of-freedom pitch–plunge benchmark reported by Mao and Liu [20], particularly their Equations (2.4) and (3.12). The deterministic drift coefficients and cubic pitch nonlinearity in Equation (2) match that published benchmark. We omit the source paper's stochastic Gaussian/Lévy excitation terms when defining the nominal deterministic plant and add two idealized generalized inputs for the present control simulations; we do not reuse the source paper's identification method or sliding-mode controller.

2.2.1. Origin and Nondimensional Scaling of the Governing Equations

The cited source starts from a coupled two-degree-of-freedom airfoil model with mass, damping, stiffness, and aerodynamic loading) and introduces the normalized quantities

$$\tau = \omega_\alpha t_{\text{phys}}, \qquad H = \frac{h_{\text{phys}}}{b}, \qquad Q_f = Q = \left(\frac{V}{b\omega_\alpha}\right)^2, \tag{1}$$

where b is the airfoil semichord, ω_α is the source reference pitch-frequency parameter, and V is the dimensional airflow speed. Thus Q_f denotes a dimensionless squared airflow-speed parameter, not a speed measured in m/s. In the state vector below, we retain the symbol h used in our numerical code for the normalized plunge coordinate H of the source paper; overdots in the nominal vector field refer to differentiation with respect to the model-time coordinate τ. The published source uses $b = 1$ m in its numerical example, so its normalized plunge and physical plunge can coincide numerically, but they have different dimensional meanings. No independent dimensional rescaling of the present numerical trajectories or aircraft-specific calibration of input magnitudes was documented. Consequently, the intervals and periods below are validated in simulation time coordinates; their conversion to SI seconds requires an explicit and consistent choice of the reference time scale. This distinction does not affect comparisons expressed in the same simulation-time coordinate or their ratios. Throughout the revised numerical reporting, τ and H are dimensionless model variables; $\mathrm{d}\alpha/\mathrm{d}\tau$ and $\mathrm{d}H/\mathrm{d}\tau$ are derivatives in the same coordinate. Quantities labeled 8, 24, or 120 refer to model-time units, not measured SI seconds. Only measured optimizer wall-clock timings retain seconds.

The state vector is defined as

$$x = [x_1, x_2, x_3, x_4]^T = [\alpha, \dot{\alpha}, h, \dot{h}]^T,$$

where α is the pitch angle, and h is the *nondimensional* plunge displacement (denoted H in the source reference). Its two velocity components are rates with respect to normalized model time. The control input is

$$u = [u_1, u_2]^T,$$

where u_1 denotes the control moment applied to the pitch degree of freedom, and u_2 denotes the control force applied to the plunge degree of freedom. These are idealized generalized inputs to the reduced-order benchmark, not calibrated physical actuators. The simulations do not model actuator bandwidth, rate limits, structural coupling of actuator channels, or specific hardware realizations; consequently the numerical input bounds cannot be interpreted as verified aircraft actuation limits.

$$\begin{aligned}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= \frac{-2.0 + 0.38Q_f}{1.75}x_1 - \frac{0.4}{1.75}x_2 + \frac{0.2}{1.75}x_3 + \frac{0.1}{1.75}x_4 - \frac{80.0}{1.75}x_1^3 + \frac{1}{1.75}u_1, \\
\dot{x}_3 &= x_4, \\
\dot{x}_4 &= \frac{0.5 - 0.27Q_f}{1.75}x_1 + \frac{0.1}{1.75}x_2 - \frac{0.4}{1.75}x_3 - \frac{0.2}{1.75}x_4 + \frac{20.0}{1.75}x_1^3 + \frac{1}{1.75}u_2.
\end{aligned} \tag{2}$$

Here, Q_f is the dimensionless squared airflow-speed parameter defined in Equation (1). The subscript f distinguishes it from the MPC state-weighting matrix $\mathbf{Q}$. The explicit structural nonlinearity in this benchmark is the cubic pitch stiffness. In the first-order pitch–plunge state-space form, this torsional nonlinearity appears as the x_1^3 terms in both $\dot{x}_2$ and $\dot{x}_4$ because of the pitch–plunge coupling.

At sampling instant k, the state and input are denoted $x_k = [\alpha_k, \dot{\alpha}_k, h_k, \dot{h}_k]^T$ and $u_k = [u_{1,k}, u_{2,k}]^T$, respectively.

In discrete time, the state-transition relationship can be abstractly written as

$$x_{k+1} = F(x_k, u_k) + w_k, \tag{3}$$

where $F(\cdot)$ denotes the state-evolution mapping over one sampling period and w_k denotes process disturbance. The additive increment w_k represents the prescribed process disturbance; modeling errors and numerical integration errors are treated separately.

This work focuses on the influence of process disturbance and model-prediction uncertainty on control performance. Measurement noise is not considered at this stage; that is, the system state is assumed to be directly available:

$$y_k = x_k.$$

Under this setting, the controller can directly use the current state as the initial state of the model predictive control (MPC) optimization problem at every sampling instant. This removes the effects of measurement noise and state-estimation error, so that equilibrium-consistent mean prediction and risk-adaptive weight selection can be examined separately from state-estimation errors.

Nonzero equilibrium branches and symmetry-breaking bifurcation.

The equilibrium structure of the nominal, unforced, noise-free cubic plant in Equation (2) can be obtained algebraically, without relying on a continuation package. Setting $u = 0$ and $\dot{x} = 0$ gives $\dot{\alpha} = \dot{h} = 0$ and the two independent steady-state equations

$$\begin{aligned} 0 &= (-2 + 0.38Q_f)\alpha + 0.2h - 80\alpha^3, \\ 0 &= (0.5 - 0.27Q_f)\alpha - 0.4h + 20\alpha^3. \end{aligned} \tag{4}$$

Eliminating h gives

$$\alpha\left[(-1.75 + 0.245Q_f) - 70\alpha^2\right] = 0, \qquad h = (1.25 - 0.675Q_f)\alpha + 50\alpha^3. \tag{5}$$

Hence the origin $E_0 = (0, 0, 0, 0)^T$ exists for every Q_f, while a pair of symmetric nonzero equilibrium states exists if and only if $Q_f > Q_{f,\mathrm{PF}} = 50/7$:

$$\begin{aligned} Q_{f,\mathrm{PF}} &= \frac{50}{7} = 7.142857\ldots, \\ E_{\pm}(Q_f) &= (\alpha_{\pm}, 0, h_{\pm}, 0)^T, \\ \alpha_{\pm} &= \pm\sqrt{\frac{0.245Q_f - 1.75}{70}}, \\ h_{\pm} &= (1.25 - 0.675Q_f)\alpha_{\pm} + 50\alpha_{\pm}^3. \end{aligned} \tag{6}$$

The vector field is odd under $x \mapsto -x$ for $u = 0$. Accordingly, the symmetric branches merge with the origin at $Q_{f,\mathrm{PF}}$ in a pitchfork-type equilibrium bifurcation; this is distinct from the earlier Hopf bifurcation. The pitchfork designation describes the branch geometry, not the stabilization of the emerging equilibria. These formulae concern the nominal unforced plant; the coordinates need not remain unchanged for a separately parameter-perturbed plant.

Linearization and equilibrium stability

For an equilibrium $E = (\alpha^*, 0, h^*, 0)^T$, the Jacobian of the nominal continuous-time vector field is

$$A(E; Q_f) = \frac{1}{1.75}\begin{bmatrix} 0 & 1.75 & 0 & 0 \\ -2 + 0.38Q_f - 240(\alpha^*)^2 & -0.4 & 0.2 & 0.1 \\ 0 & 0 & 0 & 1.75 \\ 0.5 - 0.27Q_f + 60(\alpha^*)^2 & 0.1 & -0.4 & -0.2 \end{bmatrix}. \tag{7}$$

We classify an equilibrium as locally asymptotically stable only when $\max_j \operatorname{Re}\lambda_j(A) < 0$ and as unstable if at least one eigenvalue has positive real part. At the origin, the characteristic equation simplifies to

$$\lambda^4 + \frac{12}{35}\lambda^3 + \frac{1220 - 190Q_f}{875}\lambda^2 + \frac{140 - 14Q_f}{875}\lambda + \frac{200 - 28Q_f}{875} = 0. \tag{8}$$

The first Hopf crossing occurs at $Q_{f,\mathrm{H}} = 3.079669\ldots$, where $\lambda \approx \pm 0.568286\mathrm{i}$ and the other pair has a negative real part. At $Q_f = 50/7$, the origin has a simple zero eigenvalue, in agreement with Equation (5); the remaining eigenvalues are -0.675238 and $0.166190 \pm 0.200205\mathrm{i}$. Therefore the origin is already unstable at the symmetry-breaking bifurcation. In particular, neither the origin nor the newly emerging pair can be assumed stable merely from its existence.

At the controller's operating parameter $Q_f = 8$, the complete nominal equilibrium set and its Jacobian stability classification are reported in Table 1. The origin has three eigenvalues with positive real part (counting the complex-conjugate pair separately), whereas each nonzero equilibrium has two. Both nonzero equilibria are therefore linearly unstable at $Q_f = 8$; their appearance does not provide a stable alternative static attractor. The largest-real-part eigenvalue is about 0.48016 at E_0 and 0.24192 at each $E_\pm$.

Table 1. Nominal unforced equilibria at $Q_f = 8$: analytic coordinates and numerical Jacobian stability. The unstable-eigenvalue count includes multiplicity; zero input and zero process noise are assumed.

Branch	α^* (rad)	H^* (—)	$\max \operatorname{Re}(\lambda)$	Unstable Eig.	Classification
E_0	0	0	0.480161	3	Unstable
E_+	+0.054772	−0.219089	0.241920	2	Unstable
E_-	−0.054772	+0.219089	0.241920	2	Unstable

The full spectra at $Q_f = 8$ are approximately $\{0.48016, -0.84151, 0.00924 \pm 0.26038\mathrm{i}\}$ for E_0 and $\{0.24192 \pm 0.41264\mathrm{i}, -0.41335 \pm 0.26249\mathrm{i}\}$ for both $E_\pm$. Direct numerical root-finding recovers the analytic coordinates with equilibrium residuals below 2×10^{-18}; finite-difference and analytic Jacobians agree to within 5×10^{-11} in maximum absolute entry. These values are for the stated benchmark coefficients and should not be interpreted as measured aircraft stability characteristics.

Equilibria of parameter-perturbed plant used for control

The controller's actual evaluation plant is not the nominal vector field in Equation (2). The following coefficient perturbations were introduced for this study to test a specified model mismatch; they are not additional parameter values identified in Mao and Liu [20]. The perturbed plant applies $A^r_{21} = 0.92A_{21}$, $A^r_{22} = 1.15A_{22}$, $A^r_{41} = 1.08A_{41}$, and $A^r_{44} = 0.88A_{44}$, leaving all other coefficients unchanged. In its uncontrolled deterministic steady state, the perturbed plant satisfies

$$\alpha^*\left(-1.57 + 0.2038Q_f - 70(\alpha^*)^2\right) = 0, \qquad h^* = 2.7(0.5 - 0.27Q_f)\alpha^* + 50(\alpha^*)^3. \tag{9}$$

Consequently, its pair of nonzero equilibria emerges at $Q_{f,\mathrm{PF}}^{r} = 1.57/0.2038 \approx 7.703631$, distinct from the nominal value $50/7$. At $Q_f = 8$, the two additional points are $(\alpha^*, \dot{\alpha}^*, h^*, \dot{h}^*) = (\pm 0.029374, 0, \mp 0.130389, 0)$. Direct eigenvalue evaluation of the Jacobian of the perturbed vector field yields maximum real parts 0.292632 for the origin and 0.249309 for either nonzero equilibrium. All three perturbed equilibria are therefore linearly unstable at the test parameter, but this local calculation alone does not establish the existence or stability of a perturbed-plant limit cycle.

2.3. Open-Loop Bifurcation Characterization

To characterize the nonlinear open-loop flutter behavior, numerical continuation was performed on the deterministic nominal plant with $u_1 = u_2 = 0$ and without process noise. As specified in Equation (1), continuation periods are measured in the adopted model-time coordinate; a dimensional time conversion must not be inferred solely from older simulation-time axis conventions. The flow-speed parameter Q_f was selected as the continuation parameter. The origin equilibrium and its Hopf point were independently characterized by linearization. The recovered MATCONT record identifies a Hopf-initialized limit-cycle continuation), with periodic-orbit stability assessed using stored Floquet multipliers.

The nominal origin loses stability at the Hopf point $Q_{f,\mathrm{H}} = 3.079669\ldots$ (Figure 1). Figure 2 separately depicts the symmetric equilibrium branches at $Q_f = 50/7$; their instability does not contradict an attracting periodic orbit, because equilibrium eigenvalues and orbit Floquet multipliers assess different invariant solutions.

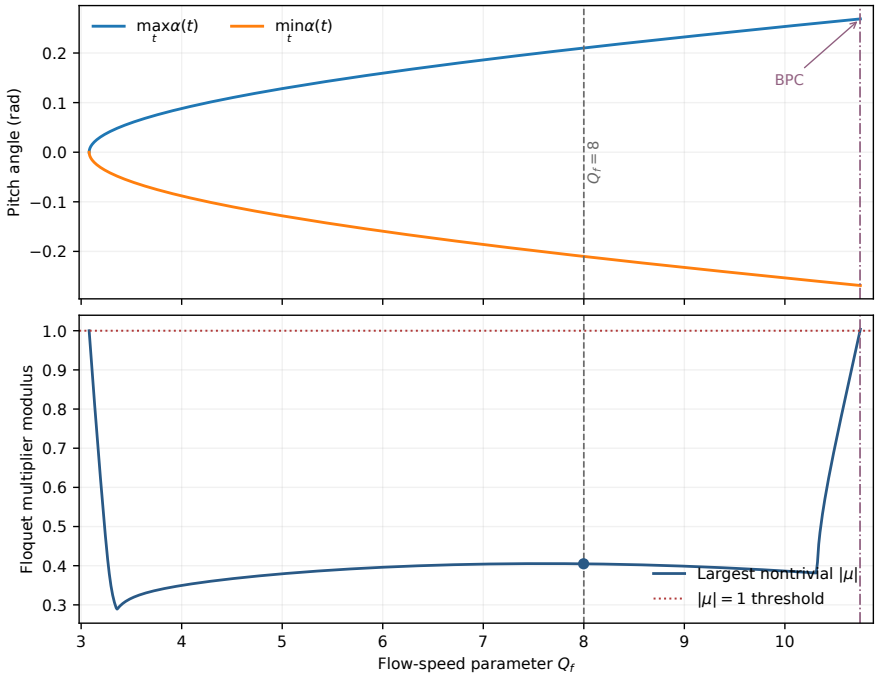


Figure 1. Nominal unforced limit-cycle branch extracted from the recovered 611-point MATCONT record. Top: maximum and minimum pitch angles along each archived periodic orbit; bottom: largest modulus among the three nontrivial Floquet multipliers, with the unit-modulus stability threshold marked by the dotted line. The dashed vertical line marks the test condition $Q_f = 8$; the dash-dotted line identifies the saved BPC near $Q_f = 10.7475$, outside the control operating condition. A nontrivial modulus below unity supports local orbital stability; near bifurcation points, unit multipliers and finite numerical precision require caution. The plot visualizes stored results, not newly rerun continuation. Equilibrium stability is shown separately in Figure 2.

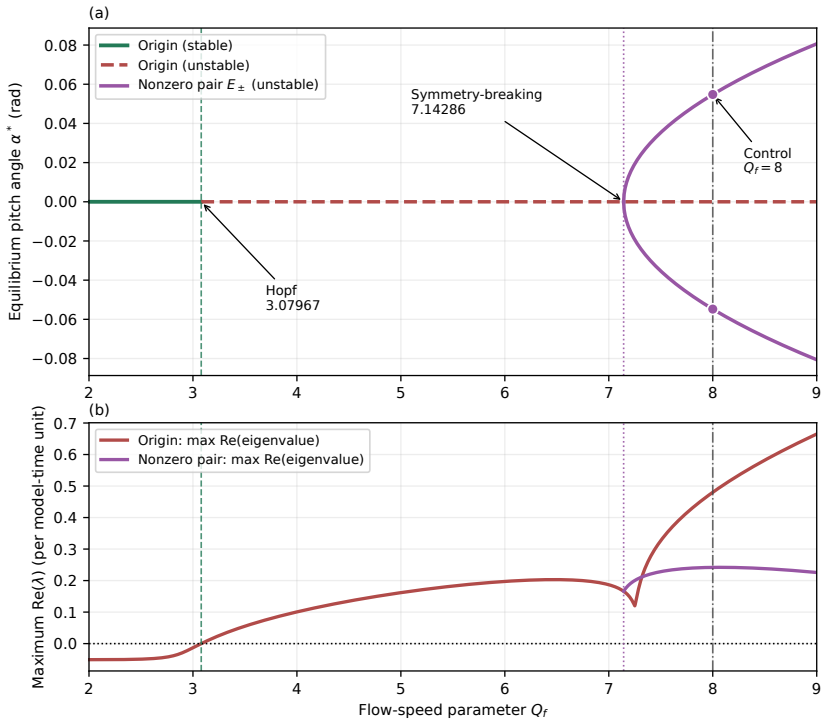


Figure 2. Analytically computed nominal equilibrium branches and Jacobian stability for $2 \leq Q_f \leq 9$. (**a**) Stable (solid green) and unstable (dashed red) origin segments and unstable symmetric nonzero equilibrium branches (purple). (**b**) Maximum real part of the equilibrium Jacobian eigenvalues, expressed per dimensionless model-time unit: positive values indicate local instability. Vertical markers indicate the Hopf point (3.079669), the symmetry-breaking equilibrium branch point (50/7), and the closed-loop test parameter (8). This diagram contains equilibria only, not limit cycles; periodic-orbit stability must instead be assessed using Floquet multipliers as in Figure 1.

The stability of the nominal LCO branch near the selected operating condition is supported by the saved Floquet multipliers. The closest archived point to $Q_f = 8$ is $Q_f = 7.997810$, with period $T = 9.522054$ model-time units. Its three nontrivial Floquet moduli are 0.404876, 0.404876, and 0.233084. At the additional nearby continuation point $Q_f = 8.1098$, the period is $T = 9.5099$, and the Floquet multiplier moduli are

$$|\mu| = \{0.99999996,\ 0.4045,\ 0.4045,\ 0.2345\}.$$

The unit multiplier is the trivial multiplier associated with phase invariance of the periodic orbit. The remaining nontrivial multipliers are all inside the unit circle at this nearby reported continuation point, supporting periodic-orbit stability in that neighborhood; this is distinct from, and does not establish stability of, the equilibria $E_0, E_\pm$ at $Q_f = 8$.

The saved continuation data are visualized with their Floquet stability threshold in Figure 1. The original limit-cycle results were verified against a recovered MATCONT archive containing 611 saved continuation points ($3.079699 \leq Q_f \leq 10.750698$), collocated with 40 mesh intervals and 4 collocation points per interval. The archived MATLAB function handles referred to `MATLAB R2024b` and the `MatCont7p6` installation directory. The saved settings specified an initial continuation step of 0.01, minimum and maximum steps of 10^{-5} and 0.02, function and variable tolerances of 10^{-6}, test tolerance 10^{-5}, 10 maximum correction iterations, 3 maximum Newton iterations, and enabled Floquet-multiplier computation. These were parameters recovered from the original saved result, not retrospectively chosen settings. A stored MATCONT directory name was not independent version certification, and the periodic-orbit continuation was not rerun here. The source

ODE file `jiyihopf.m`, referenced in the archive, was not recovered with this record; rerunning the original continuation from source therefore remains unverified.

For reproducibility of the new equilibrium analysis, branch coordinates were evaluated analytically from Equation (6), while Jacobian eigenvalues were calculated in double precision over a Q_f grid with spacing of 0.005. A positive-real-part tolerance of 10^{-8} was used for the stability classification away from bifurcation points. The Hopf location was independently recovered from the Routh–Hurwitz condition in Equation (8) using a bracketed scalar root solver with absolute tolerance 10^{-14}. Equilibrium coordinates and Jacobians were cross-checked using numerical root-finding and centered finite differences. The recovered continuation array, associated Floquet multipliers, an independent Python extraction script, and the separate equilibrium-analysis script are provided as numerical reproducibility materials; the Python extraction reproduced *recorded results* and was not a new MATCONT continuation run.

No limit point of cycles, period-doubling bifurcation, or Neimark–Sacker/torus bifurcation was detected before $Q_f = 8.0$. A branch point of cycles was detected only at a higher flow-speed parameter, $Q_f = 10.7475$, which is outside the closed-loop operating range considered in this paper. Therefore, the closed-loop simulations focused on suppressing a stable post-flutter LCO rather than controlling near a fold, torus, or period-doubling bifurcation.

Long-time, uncontrolled response of the actual perturbed plant.

To verify the uncontrolled behavior of the *same* parameter-perturbed plant used in the subsequent controller experiments, the nominal and perturbed vector fields were simulated side by side at $Q_f = 8.0$, with $u(t) \equiv (0, 0)^T$, initial state $x_0 = (0.15, 0, 0.10, 0)^T$, and no process noise so that the effects of parameter mismatch could be isolated. Both simulations used the same fourth-order Runge–Kutta (RK4) integration as the plant generator, with internal step $\Delta t_{\mathrm{RK4}} = 0.05$ model-time units, and were extended to 120 model-time units. This duration spans more than twelve observed oscillation periods, in contrast with the eight-model-time-unit controlled evaluation interval.

Figure 3 shows that both uncontrolled plants continue to oscillate well beyond the original eight-model-time-unit window. Over the late model-time 60–120 interval, the peak-to-peak half-ranges and periods inferred from successive positive pitch-angle maxima are summarized in Table 2. The perturbed plant exhibits a roughly 19.7% larger plunge half-range than the nominal plant; the pitch-angle half-ranges are similar. The reported nearby nominal continuation period 9.5099 model-time units was computed at $Q_f = 8.1098$, whereas the direct nominal and perturbed periods in the present table refer to $Q_f = 8.0$. These are not identical operating points or vector fields.

Table 2. Direct 120-model-time-unit uncontrolled RK4 simulation at $Q_f = 8$ and the common initial state $x_0 = (0.15, 0, 0.10, 0)^T$; $u = 0$, with no process noise. The half-range of each state and the mean separation between successive positive pitch-angle peaks are evaluated after 60 model-time units. The reported period standard deviation is the variation across five interpeak intervals, not confidence in an exact Floquet cycle period.

Unforced Plant	Pitch Half-Range (rad)	Plunge Half-Range (Normalized)	Mean Period (Model Time)	Period SD (Model Time)
Nominal	0.21025	0.49894	9.530	0.027
Parameter-perturbed	0.21099	0.59732	9.660	0.022

The late-time trajectories indicate sustained approximately periodic behavior. This direct simulation does not replace Floquet analysis of the perturbed plant's periodic orbit.

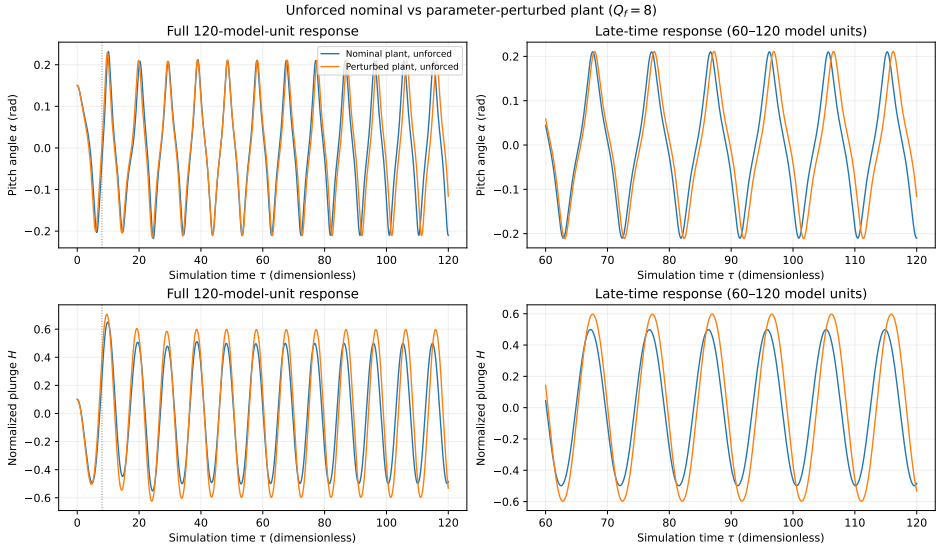


Figure 3. Direct uncontrolled response of the nominal and actual parameter-perturbed plants at $Q_f = 8$. Both start from $[0.15, 0, 0.10, 0]^T$ with $u(t) = 0$ and no process noise. Left panels show the full 120-model-time-unit trajectory, with a vertical dashed marker at the 8 model-time units duration used in the reported closed-loop comparisons; right panels enlarge the late model-time 60–120 interval. Persistent near-periodic oscillations, rather than natural decay to zero, are observed in both numerical trajectories.

For an integration-step check, the same trajectories were reintegrated with RK4 step 0.01 model-time units and compared at shared sample times. The largest absolute discrepancy for the perturbed trajectory was 2.98×10^{-6} rad in pitch and 8.61×10^{-6} normalized-plunge units in plunge over the full 120 model-time units; the maximum normalized four-state discrepancy was 3.47×10^{-5}. Five additional uncontrolled trajectories with independent Gaussian velocity-state increments $w_k \sim \mathcal{N}(0, \mathrm{diag}(0, 0.03^2\Delta t, 0, 0.03^2\Delta t))$ were generated using seeds 1010–1014 to document noise-induced variability; their individual histories and all reproduction settings are provided with the accompanying analysis script and data.

The 120-model-time-unit integration is an uncontrolled reference. The systematic paired closed-loop experiments lasted 8 model-time units, while a separate representative 24-model-time-unit controlled trajectory is evaluated in Section 4.8.

2.4. Data-Driven State-Transition Modeling Problem

The offline training data comprise state–input–next-state transitions,

$$\mathcal{D} = \{(x_i, u_i, x_{i+1})\}_{i=1}^{N_d}. \tag{10}$$

An RC readout learns state increments $\Delta x_i = x_{i+1} - x_i$, and a separate conditional network estimates the mean remaining RC prediction error. The models are fitted to simulated transitions of the parameter-perturbed plant; the explicit nominal equations are used for benchmark definition and dynamical analysis, not for online MPC prediction. The training and predictor equations are given in Section 3.

2.5. Control Objective and Constraints

The control objective is to suppress pitch and plunge deviations and their velocities toward a small neighborhood of the origin. Under persistent process noise, the empirical criterion is practical sustained regulation, defined in Section 4, rather than an assertion of pointwise convergence $x_k \to 0$.

The control input is subject to amplitude constraints:

$$u_{\min} \leq u_k \leq u_{\max}. \tag{11}$$

Actuator amplitude bounds are enforced in the implemented MPC. The following state limits are used for normalization and empirical violation monitoring, but no risk-dependent soft state-constraint penalty is active in the reported optimization:

$$|\alpha_k| \leq \alpha_{\max}, \quad |\dot{\alpha}_k| \leq \dot{\alpha}_{\max}, \quad |h_k| \leq h_{\max}, \quad |\dot{h}_k| \leq \dot{h}_{\max}. \tag{12}$$

In the fixed-weight comparison, the Q/P matrices remain constant throughout a trajectory. In the risk-adaptive controller, a local KNN prediction-error risk changes the state and terminal cost weights. Other control weights and the actuator bounds remain unchanged.

This study evaluated whether equilibrium-consistent learned prediction and local-error-weighted MPC improve regulation relative to a tuned constant-weight comparator under prescribed stochastic disturbances.

2.6. Difference from Conventional Model Predictive Control

Conventional MPC predicts trajectories with a specified state-transition model [21,22]. Here the optimizer instead uses an offline-fitted, equilibrium-corrected RC/residual-mean map; local residual-error covariance modifies only the state and terminal penalties. Thus the principal distinctions are the learned prediction map and its risk-dependent weighting, not stochastic scenario propagation or a chance-constrained formulation [23].

Scope of controller comparison

A nominal-equation-based MPC would be a relevant, distinct reference method, but it was *not* implemented or evaluated in the present experiments. Our controlled comparison instead held the learned predictor, prediction horizon, input bounds, control and increment penalties, plant, and paired disturbances fixed; only the Q/P state and terminal weighting policy differed between the independently tuned fixed-weight and risk-adaptive controllers. This design examined the incremental effect of weight adaptation for the tested learned-prediction architecture; a separate exploratory KNN-independent state-feedback scheduler is compared in Section 4.11. It neither isolated the advantage of the learned predictor over the analytical equations nor established superiority over a separately tuned physics-based MPC. A meaningful physics-based comparison would require a specified nominal or mismatch-aware physics predictor, independent weight tuning, the same constraints and disturbance realizations, and reported computational costs. We therefore regard such a benchmark as an unperformed extension, not a theoretical performance result.

3. Equilibrium-Consistent RC Residual-Mean Prediction and Risk-Adaptive MPC

3.1. Implementation Scope

The implemented controller combined an RC state-transition predictor, a conditional residual-mean network, KNN covariance of residual-mean errors, and equilibrium-corrected, risk-weighted deterministic MPC. A diffusion denoiser was trained separately offline, but no denoiser-generated residual entered the optimized rollouts; stochastic scenario allocation and chance constraints were therefore not evaluated. The actual perturbed plant was advanced using RK4 with a 0.05 model-time-unit sampling interval and an additive process increment in the velocity channels. Figure 4 summarizes the complete offline-training and online-control workflow used in the reported experiments.

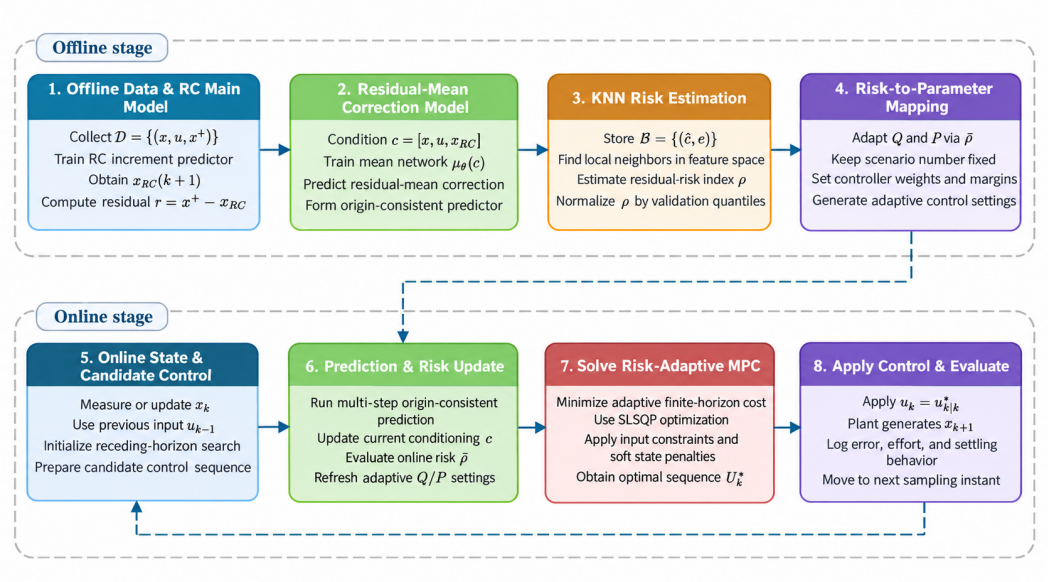


Figure 4. Workflow of the equilibrium-consistent RC residual-mean and KNN risk-adaptive MPC framework. In the offline stage, simulated transitions of the parameter-perturbed plant are used to train the RC predictor and the conditional residual-mean compensator, then to build a fixed KNN error library and calibrate the normalized risk score. In the online stage, the current state is propagated by the learned equilibrium-corrected predictor, the KNN-based local risk updates only the state and terminal weighting matrices Q_k and P_k, and SLSQP solves a deterministic receding-horizon optimization whose first control input is applied to the plant. The fixed-weight comparator uses the same learned predictor, horizon, constraints, and plant, but keeps Q_k and P_k constant.

3.2. RC Main Dynamics Prediction Model

The implemented reservoir-computing (RC) predictor uses the six-dimensional input $z_i = [x_i^T, u_i^T]^T$, standardized componentwise with a Standard Scaler fitted on the training inputs. For reservoir state $s_i \in \mathbb{R}^{n_r}$, the update is

$$s_{i+1} = (1 - \lambda)s_i + \lambda \tanh(W_{\text{in}}\tilde{z}_i + W_r s_i). \tag{13}$$

The fixed input matrix has independent entries sampled uniformly from $[-0.6, 0.6]$. The recurrent matrix is initially sampled from a standard normal distribution, retained entry-wise with probability 0.08, and rescaled to spectral radius 0.9. The reservoir dimension is $n_r = 250$, the leakage rate is $\lambda = 0.25$, and the ridge parameter is $\beta = 10^{-4}$. Network initialization uses the NumPy random-number stream seeded with 20261010 for the reproducible control experiments.

Training samples and meaning of washout

The reported implementation generates 6000 independent one-step state–control–successor triples from the parameter-perturbed plant, using RK4 with $\Delta t = 0.05$ model-time units. Initial states are sampled independently from $[-0.5, 0.5] \times [-1, 1] \times [-0.3, 0.3] \times [-1, 1]$; control inputs follow the four-mode excitation policy in the supplied code. For the training data, independent process increments have velocity-component standard deviations $0.02\sqrt{\Delta t}$ and zero position-component standard deviations; measurement noise is disabled. A shuffled 80%/20% split (`random_state=42`) gives 4800 fitting and 1200 validation transitions. These are *not* chronological sections of a single physical trajectory.

The reservoir is initialized to $s_0 = 0$ for training. The first 80 shuffled training inputs are processed as an initialization pass; subsequently the complete shuffled fitting array is processed again and all 4800 resulting reservoir states are retained for the readout fit.

Thus the variable named `washout=80` does not discard the first 80 regression rows, and the resulting recurrent states do not represent chronological aeroelastic histories. The fitting procedure has no sequence-boundary reset within this shuffled pass. This distinction is relevant when interpreting reservoir memory and the separate history-warmup sensitivity experiment in Section 4.3.

Readout dimensions and ridge solution

The output targets are the four-dimensional increments $\Delta x_i = x_{i+1} - x_i$. A second training-only standardizer is fitted to these increments. Define $\widetilde{Y}_\Delta \in \mathbb{R}^{N\times 4}$ as their standardized rows and $\Phi = \begin{bmatrix} s_1^T & 1 \\ \vdots & \vdots \\ s_N^T & 1 \end{bmatrix} \in \mathbb{R}^{N\times(n_r+1)}$ as the stored reservoir features. The code fits the coefficient matrix $B \in \mathbb{R}^{(n_r+1)\times 4}$ by

$$B = (\Phi^T\Phi + \beta I)^{\dagger}\Phi^T\widetilde{Y}_\Delta, \qquad W_{\text{out}} = B^T \in \mathbb{R}^{4\times(n_r+1)}, \tag{14}$$

where † denotes the pseudoinverse actually used by the program. Thus the mathematical column-vector convention is $\widetilde{\Delta x}_i = W_{\text{out}}[s_{i+1}^T, 1]^T$, whereas the NumPy array named `Wout` stores B and is multiplied from the right by a row feature vector. The standardized prediction is inverse-transformed, and its four increment components are clipped to $[-0.20, 0.20]$ before adding x_i to obtain $\hat{x}_{i+1}^{\text{RC}}$. The ridge penalty in the code also acts on the appended bias coefficient.

Reservoir state in reported controller

At the beginning of a closed-loop simulation, the controller resets the shared RC state to zero. A candidate MPC rollout passes a local copy of that state through Equation (13); this copy is propagated along the predicted horizon and does not modify the shared RC state. Between successive actual receding-horizon control updates, the reported perfect-measurement simulation does not explicitly advance this shared state using the newly observed transition. Consequently, the implemented optimization is not a continuously synchronized, trajectory-trained stateful RC controller. The held-out zero-state and history-warmup results quantify an associated limitation. A chronologically trained and explicitly synchronized variant is left for future work rather than implicitly assumed in the reported results.

3.3. Conditional Residual-Mean Estimator

Let the residual of the RC one-step prediction be $r_k = x_{k+1} - \hat{x}_{k+1}^{\text{RC}}$ and define the condition vector

$$c_k = \left[x_k^T,\ u_k^T,\ (\hat{x}_{k+1}^{\text{RC}})^T\right]^T. \tag{15}$$

A conditional regression network μ_θ is fitted by minimizing the empirical residual mean-squared error,

$$\min_\theta \frac{1}{N}\sum_{i=1}^{N} \|r_i - \mu_\theta(c_i)\|_2^2. \tag{16}$$

Code-verified residual-model settings

The active conditional-mean regressor is a scikit-learn `MLPRegressor` with two hidden layers of 64 units each, ReLU activation, Adam optimization, initial learning rate 10^{-3}, and `max_iter=600`. Its ten-dimensional condition and four-dimensional residual targets are separately standardized on the fitting partition. The exploratory denoiser, which is not queried by the implemented MPC, uses hidden layers of 96 and 96 units with the same

activation, optimizer, initial learning rate, and iteration cap. In the reported training entry point, 15 diffusion steps, a linear beta schedule from 10^{-4} to 0.02, two augmentation draws per fitting example, and DDIM noise parameter $\eta = 0$ are configured. These parameters describe the separately fitted network, not a sampled uncertainty model used in control, and no distributional calibration is claimed. The uncorrected composite mean predictor is

$$\widehat{F}_{\text{mean}}(x, u; s) = f_{\text{RC}}(x, u; s) + \mu_\theta(c(x, u; s)), \tag{17}$$

where $c(x, u; s) = [x^T, u^T, f_{\text{RC}}(x, u; s)^T]^T$, and s denotes the RC reservoir state used for that evaluation. Although a separate conditional diffusion denoiser was trained in the exploratory workflow, its sampling path was not invoked by the MPC optimization in the experiments here. In particular, residual-mean compensation is not evidence of a diffusion scenario effect.

3.4. Equilibrium-Consistent Mean Prediction

The physical benchmark has a zero-state equilibrium under zero input. In contrast, the learned mean predictor may return a nonzero next state at the origin. We therefore anchor the learned map at the known zero equilibrium:

$$\widehat{F}_{\text{eq}}(x, u; s) = \widehat{F}_{\text{mean}}(x, u; s) - \widehat{F}_{\text{mean}}(0, 0; s). \tag{18}$$

Both terms must be evaluated at the *same* reservoir state s, giving $\widehat{F}_{\text{eq}}(0, 0; s) = 0$ by construction. This post-training correction does not imply that the resulting closed-loop system is asymptotically stable; it removes a structural equilibrium bias in the predictor.

For a candidate horizon trajectory, the implemented mean prediction is

$$\begin{aligned} \widehat{x}_{k|k} &= x_k, \\ \widehat{x}_{k+\ell+1|k} &= \widehat{F}_{\text{eq}}(\widehat{x}_{k+\ell|k}, u_{k+\ell|k}; s_{k+\ell|k}), \qquad \ell = 0, \ldots, N_p - 1. \end{aligned} \tag{19}$$

The reservoir-state handling follows the frozen implementation used in the numerical experiments; the present results should not be interpreted as independent validation of a stateful trajectory-trained RC update scheme. Section 4.3 reports held-out, dynamically consistent one-step and multi-step tests; these do not remove the training sequence limitation.

3.5. Risk Estimation Based on KNN Local Residual Covariance

The risk estimator is a fixed, offline K-nearest-neighbor (KNN) summary of residual-mean prediction errors. It modulates the MPC state and terminal weights but does not estimate the external process noise covariance or provide a calibrated probability of constraint violation.

Risk library and dataset partition

For each of the 4800 fitting transitions, the original RC prediction $\hat{x}_{i+1}^{\text{RC}}$ is used to construct $c_i = [x_i^T, u_i^T, (\hat{x}_{i+1}^{\text{RC}})^T]^T \in \mathbb{R}^{10}$. The separately trained conditional mean network gives $\mu_\theta(c_i)$. The risk-library error is

$$e_i = \left(x_{i+1} - \hat{x}_{i+1}^{\text{RC}}\right) - \mu_\theta(c_i). \tag{20}$$

The input features c_i are standardized using the condition scaler fitted on these training samples; the library stores $\mathcal{B} = \{(\tilde{c}_i, e_i)\}_{i=1}^{4800}$. These e_i are in-sample errors of the fitted residual-mean network, not out-of-fold errors. The separate 1200-sample validation split was used only to set the risk normalization quantiles, not to extend the neighbor library. Held-out prediction trajectories were not used for this calibration.

KNN covariance and units.

Given an online condition c, find the $K = 60$ nearest neighbors under Euclidean distance between the standardized ten-dimensional conditions. Let $\widehat{\Sigma}_{\mathrm{loc}}(c)$ be the ordinary sample covariance of their four-dimensional e_i vectors in their original state units. The actual implementation computes

$$\begin{aligned} \Sigma_g &= \mathrm{Cov}\{e_i\}_{i=1}^{4800} + 10^{-6} I_4, \\ \Sigma_{\mathrm{KNN}}(c) &= (1 - 0.05)\left[\widehat{\Sigma}_{\mathrm{loc}}(c) + 10^{-6} I_4\right] + 0.05\Sigma_g, \\ \rho(c) &= \mathrm{tr}\,\Sigma_{\mathrm{KNN}}(c). \end{aligned} \tag{21}$$

The covariance multiplier is 1.0; no additional residual-coordinate standardization or state-scale weighting is applied before the trace. Accordingly, ρ is a mixed-unit numerical dispersion score, not a dimensionless physical uncertainty measure. Its magnitude depends on the chosen state units, and a small trace cannot rule out prediction bias or out-of-distribution queries. The isotropic floor is likewise a numerical regularizer in the implemented coordinates, not a physical noise floor. Replacing it with a normalized covariance would alter the controller and would require rerunning the closed-loop comparisons; this change is not made retrospectively here.

Validation-quantile normalization.

The code randomly selects at most 800 of the 1200 validation conditions using seed 42 and evaluates ρ for each. Let $q_{.05}$ and $q_{.99}$ be their empirical fifth and ninety-ninth percentiles. The risk input to the controller is

$$\bar{\rho}_k = \mathrm{clip}\left(\frac{\rho(c_k) - q_{.05}}{q_{.99} - q_{.05} + 10^{-8}}, 0, 1\right), \qquad \rho_{\mathrm{eff},k} = \mathrm{clip}(1.5\bar{\rho}_k, 0, 1). \tag{22}$$

If $q_{.99} \le q_{.05} + 10^{-10}$, the program replaces the upper bound by $q_{.05} + 10^{-6}$ before applying this normalization. These quantile bounds are held fixed for the online runs; the KNN library is not updated using observed closed-loop prediction errors.

Risk query timing and interpretation

At time k, the pre-optimization condition uses the current measured state x_k and a reference control u_k^{ref} equal to zero at the first update and to the first input of the preceding optimized sequence thereafter. The uncorrected RC predictor computes $\hat{x}_{k+1}^{\mathrm{RC}}(x_k, u_k^{\mathrm{ref}})$ and constructs $c_k = [x_k^T, (u_k^{\mathrm{ref}})^T, (\hat{x}_{k+1}^{\mathrm{RC}})^T]^T$. The resulting risk and Q_k, P_k are fixed during the subsequent SLSQP minimization. This avoids conditioning the risk on an unknown future optimal input; risk is not recomputed for every candidate input sequence. The origin correction is applied to the prediction rollouts, not to the RC component used to build the online KNN query.

As observed in the white noise and AR(1) comparisons, the average ρ changes little when externally imposed noise intensity is doubled. This is consistent with a fixed offline prediction-error library: the risk can change as its query condition changes, but it has no explicit online mechanism for learning a changed disturbance variance. The independent held-out risk–error ranking test, synthetic bias counterexample, and out-of-domain feature-space diagnostic are reported in Section 4.9. They assess empirical error discrimination but do not establish calibrated covariance coverage or probabilistic reliability.

3.6. Risk-Adaptive Deterministic Mean-Prediction MPC

At the beginning of a control update, the KNN risk computed using the previously applied control determines $\rho_{\mathrm{eff},k}$ according to Equation (22); the cost weights remain constant

throughout that optimization. With $W = \mathrm{diag}(1.5, 0.2, 1.5, 0.2)$, the implemented adaptive cost matrices are

$$\begin{aligned} Q_k &= Q_0 + 120\rho_{\mathrm{eff},k} W, \\ P_k &= P_0 + 280\rho_{\mathrm{eff},k} W, \\ R_k &= R_0, \qquad S_k = S_0. \end{aligned} \tag{23}$$

The baseline matrices for the numerical tests are

$$\begin{aligned} Q_0 &= \mathrm{diag}(50, 30, 50, 30), & P_0 &= \mathrm{diag}(200, 100, 200, 100), \\ R_0 &= \mathrm{diag}(0.10, 0.10), & S_0 &= \mathrm{diag}(0.02, 0.02). \end{aligned} \tag{24}$$

In particular, S_k is fixed ($\alpha_S = 0$); only the state and terminal penalties change with the local risk. The numerical implementation starts from the core-class default $R_{\mathrm{code}} = \mathrm{diag}(0.05, 0.05)$ and applies the common multiplier 2 in the reported controllers, so the actual penalty is $R_k = R_0 = 2R_{\mathrm{code}} = \mathrm{diag}(0.10, 0.10)$ for both the systematic eight-unit experiments and the extended 24-unit realization. Here R_0 always denotes the post-multiplier matrix; it must not be multiplied by two a second time.

The tuned time-invariant comparator uses the *same* corrected mean predictor, risk-estimation database, R_0, S_0, and plant. Its state and terminal matrices are obtained by fixing $\rho_{\mathrm{eff}} = 0.24$ in Equation (23):

$$\begin{aligned} Q_{\mathrm{frozen}} &= \mathrm{diag}(93.20, 35.76, 93.20, 35.76), \\ P_{\mathrm{frozen}} &= \mathrm{diag}(300.80, 113.44, 300.80, 113.44). \end{aligned} \tag{25}$$

This value was selected on separate development noise realizations (seeds 1005 and 1006) from a prespecified second-stage candidate set $\{0.20, 0.24\}$ and was locked before the independent tests using seeds 1010–1014. Comparisons were made under similar realized input effort rather than asserting identical control effort from equal numerical weight matrices.

Let $U_k = \{u_{k|k}, \ldots, u_{k+N_p-1|k}\}$, $N_p = 20$, and $\Delta u_{k+\ell|k} = u_{k+\ell|k} - u_{k+\ell-1|k}$ with the preceding applied control used at $\ell = 0$. The optimized cost is

$$\begin{aligned} J_k(U_k) = \sum_{\ell=0}^{N_p-1} \big[& \widehat{x}_{k+\ell|k}^T Q_k \widehat{x}_{k+\ell|k} + u_{k+\ell|k}^T R_0 u_{k+\ell|k} \\ & + \Delta u_{k+\ell|k}^T S_0 \Delta u_{k+\ell|k} \big] + \widehat{x}_{k+N_p|k}^T P_k \widehat{x}_{k+N_p|k}. \end{aligned} \tag{26}$$

The applied receding-horizon control is

$$U_k^\star = \arg\min_{U_k} J_k(U_k), \quad u_{\min} \le u_{k+\ell|k} \le u_{\max}, \quad u_k = u_{k|k}^\star. \tag{27}$$

The predicted state satisfies Equation (19); sequential least squares programming (SLSQP) solves the finite-horizon problem. The experimental configuration retains a scenario-count placeholder equal to four, but all optimized predictions use the same conditional residual mean. Therefore this placeholder has no independent scenario-distribution effect. The soft state-constraint penalty and risk-tightening coefficients are both zero, so neither chance constraints nor risk-dependent state-bound tightening are claimed. Empirical state-limit violations were monitored after simulation.

SLSQP implementation and failure handling

The constrained decision vector has $2N_p = 40$ components, with box bounds $-8 \le u_{j,k+\ell|k} \le 8$ for both control channels. The optimization calls `scipy.optimize.minimize` with `method='SLSQP'`, `maxiter=30`, and `ftol=1e-4`; no analytical objective Jacobian or non-

linear state-constraint objects are supplied. SciPy therefore uses its numerical differentiation defaults for the objective gradient. The first optimization is initialized with a zero control sequence; subsequently the previous length-N_p solution is shifted by one control move, and its final two components are repeated as the terminal guess. Only the first optimized control is applied, with the first element of the preceding optimization sequence used as the previous input in the control-increment cost.

For transparency, the unchanged core solver code takes `res.x` as the control sequence even when `res.success=False`, provided the call returns normally; it does not retry, reject a non-success status, or switch to a certified feasible fallback on that basis. Only an exception triggers a zero-input return. The diagnostic wrappers used in the reported experiments recorded `success`, `status`, `message`, iteration count, function evaluations, objective value, and elapsed wall-clock time without changing the returned solution. All 6400 calls in the four eight-model-time-unit disturbance conditions, as well as all 480 calls in the single 24-model-time-unit run, reported success status. This empirical result does not remove the deficiency in the fallback policy for untested operating conditions, and a successful solver flag is not a proof of closed-loop stability. The reported average solver time is about 9.6–11.0 SI seconds of wall-clock time, whereas the numerical plant advances by 0.05 dimensionless model-time units per control step. Without specifying the model reference-frequency scaling, these two durations cannot be compared directly in SI units; optimizer execution delays were not inserted into the plant clock. The results describe offline numerical MPC, not deployable real-time actuation.

3.6.1. Recorded Computational Environment and Timing Provenance

A separate environment audit collected on 14 October 2026 reported a 64-bit Python 3.11 installation on Microsoft Windows (platform identifier `Windows-10-10.0.26300-SP0`), with an AMD CPU identified as Family 25, Model 97, Stepping 2 (32 logical processors), and 31.69 GiB of installed memory. Recorded packages included NumPy 2.1.2, SciPy 1.15.3, scikit-learn 1.8.0, and PyTorch 2.7.0+cu126. The audit established these details for the machine when the report was collected; the archived optimization-time logs do not independently record platform metadata. Thus the correspondence between this later audit and every earlier timed run was not independently verified, and the measured solver times are not presented as a hardware-normalized performance benchmark. The full environment report is retained with the revision records.

3.6.2. Closed-Loop Procedure

The receding-horizon control procedure is summarized in Algorithm 1. It combines the local KNN risk query, adaptive state and terminal weights, and equilibrium-corrected mean-prediction MPC described above.

Algorithm 1 Equilibrium-consistent KNN risk-adaptive mean-prediction MPC.

1: **Input:** current measured state x_k, previous input u_{k-1}, and fitted RC, conditional mean, and KNN models.
2: Set $u_k^{\mathrm{ref}} = 0$ initially; otherwise, take the first input of the previously optimized sequence; form c_k using x_k, u_k^{ref}, and the uncorrected RC output.
3: Evaluate the fixed-library KNN residual-error covariance and normalized risk $\bar{\rho}_k$.
4: Compute $\rho_{\mathrm{eff},k}$ and set Q_k, P_k by Equation (23), or use the locked constant matrices for the comparator.
5: For each SLSQP candidate input sequence, recursively propagate the equilibrium-corrected mean map, Equation (19).
6: Run SLSQP with box bounds. Apply the first returned input; log the success flag. If the optimizer raises an exception, the original code returns a zero input.
7: Advance the mismatched plant and add its designated process disturbance; repeat at the next sampling instant.

4. Simulation Experiments and Result Analysis

4.1. Numerical Setup and Comparison Protocol

The closed-loop plant was the parameter-perturbed model defined in Section 2 at $Q_f = 8$, with full-state measurement and initial state $x_0 = [0.15, 0, 0.10, 0]^T$. Each systematic paired run lasted 8 model-time units (160 control moves at $\Delta t = 0.05$ model-time units). Offline models were fitted using the 6000 independent one-step examples (of which 4800 were fitting and 1200 validation samples) with the initialization, input scaling, and reservoir-state procedure described in Section 3.2. The KNN settings and quantile calibration are specified in Section 3.5; models and risk-library parameters remained fixed during testing. SLSQP optimized an equilibrium-corrected mean predictor over $N_p = 20$ steps, without independently sampled diffusion residuals or active soft constraint tightening. A separate 24-model-time-unit realization is reported in Section 4.8.

The frozen Q/P baseline in Equation (25) was selected using development seeds 1005, 1006. It was subsequently compared with adaptive weighting using five new noise seeds 1010–1014. For any given seed and noise condition, both controllers received the same sampled disturbance increments. Both used the same physical plant, initial condition, state normalization, R_0, S_0, horizon, and equilibrium correction. The controller weights were never retuned based on validation or robustness test results.

Two synthetic disturbance families were examined at scale parameters $\sigma = 0.03$ and 0.06. The white-noise increments in the velocity states had standard deviation $\sigma\sqrt{\Delta t}$; the position-state increments were zero. The temporally correlated test used the stationary AR(1) recursion

$$z_k = \phi z_{k-1} + \sqrt{1-\phi^2}\,\xi_k, \quad \phi = 0.85, \quad \xi_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I), \quad w_k = \sigma\sqrt{\Delta t}\,[0, z_{k,1}, 0, z_{k,2}]^T. \tag{28}$$

An independently sampled z_0 from the standard normal distribution was used to represent a stationary initial condition. This synthetic correlated process is *not* claimed to be a physically calibrated gust or turbulence model.

4.2. Performance Indicators and Sustained-Regulation Definition

The normalization used $D = \mathrm{diag}(0.35, 1.80, 0.25, 1.80)$ and $e_k = \|D^{-1}x_k\|_2$. The normalized integral squared error, raw integral squared error, and input effort are

$$J_N = \Delta t \sum_{k=0}^{N_T-1} e_k^2, \qquad J_{\text{raw}} = \Delta t \sum_{k=0}^{N_T-1} \|x_k\|_2^2, \qquad E_u = \Delta t \sum_{k=0}^{N_T-1} \|u_k\|_2^2. \tag{29}$$

E_u is an input-effort surrogate and is not a calibrated actuator energy in joules. The late-window normalized root mean square error is

$$E_{\text{RMS}} = \sqrt{\frac{1}{|\mathcal{W}|}\sum_{k\in\mathcal{W}} e_k^2}, \qquad \mathcal{W} = \{k : 6 \le \tau_k \le 8\}. \tag{30}$$

Strict 5% sustained regulation is declared only if $e_k \le 0.05 e_0 = 0.029312$ was maintained for at least 0.5 model-time units and remained satisfied through the end of the simulation. Failures are reported explicitly; average settlement times over different subsets of successful runs are not treated as unconditional paired comparisons. Empirical state-limit violations and SLSQP success flags are reported independently of this regulation indicator.

4.3. Independent One-Step and Multi-Step Prediction Validation

The prediction models were evaluated independently of the closed-loop experiments to distinguish genuine out-of-sample forecasting ability from controller performance. The three predictors were the RC model alone, the RC model with conditional residual-mean compensation in Equation (17), and the same composite predictor with equilibrium cor-

rection in Equation (18). The original training procedure used 6000 randomly sampled, independently generated one-step state–control examples; its RC reservoir states were formed over shuffled samples rather than chronologically continuous physical trajectories. No new RC parameters were fitted using the prediction-test trajectories.

Twenty independently generated, dynamically continuous trajectories of the nonlinear plant at $Q_f = 8$ were used for testing, with 70 integration steps per trajectory. Ten trajectories were generated under a prescribed excitation/feedback input and ten under zero input. Five prediction starts (steps 10, 20, 30, 40, and 50) were evaluated on each trajectory, giving 100 forecast windows for each predictor and horizon. Test trajectories were generated from the deterministic physical map without injected process noise, even though process perturbations were included in the offline training samples. Thus, the following errors measure deterministic state-transition forecasting on independent simulated trajectories; they do not quantify noisy-process forecast calibration or closed-loop stochastic stability. The evaluation used the same $\Delta t = 0.05$ model-time units as the MPC and horizons $H \in \{1, 5, 10, 20\}$, corresponding to 0.05, 0.25, 0.50, and 1.00 model-time units.

For trajectory i and forecast start k_j, the H-step prediction $\widehat{x}^{(m)}_{i,k_j+H|k_j}$ of method m was recursively generated using the prescribed future inputs but without resetting the predicted states to subsequent measured states. The endpoint error was normalized using the state-scale matrix D specified above. The reported metric is

$$\mathrm{NRMSE}_H^{(m)} = \left[\frac{1}{N_{\mathrm{tr}} N_{\mathrm{start}}} \sum_{i=1}^{N_{\mathrm{tr}}} \sum_{j=1}^{N_{\mathrm{start}}} \left\| D^{-1} \left(\widehat{x}^{(m)}_{i,k_j+H|k_j} - x_{i,k_j+H} \right) \right\|_2^2 \right]^{1/2}, \qquad (31)$$
$$N_{\mathrm{tr}} = 20, \qquad N_{\mathrm{start}} = 5.$$

This is the root mean squared four-state normalized Euclidean endpoint error; it does not divide the squared norm by the number of state components. The equal-sized five-window groups are first averaged within each independent trajectory and then across trajectories. Exploratory 95% intervals in Figure 5 were obtained by bootstrap resampling complete trajectories (not the overlapping forecast windows) as the independent units.

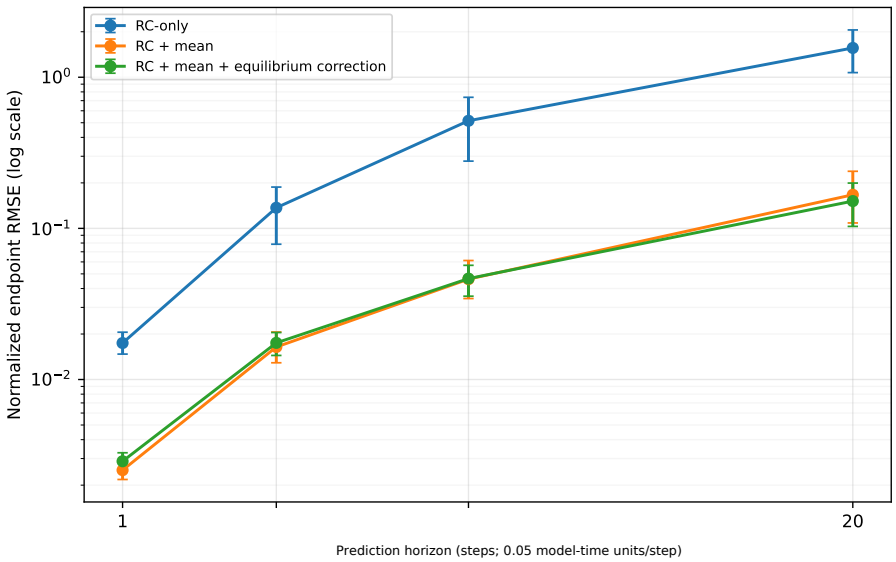


Figure 5. Independent one-step and free-running multi-step prediction error from Table 3. The ordinate uses a logarithmic scale to resolve the two residual-compensated predictors. Error bars are exploratory 95% trajectory-bootstrap intervals, not confidence intervals for pairwise differences.

Table 3. Independent continuous-trajectory forecast validation: normalized four-state endpoint RMSE versus open-loop prediction horizon. Reservoir state is initialized at zero at each forecast start, matching the implemented MPC initialization. Each number summarizes 20 independent trajectories and five forecast windows per trajectory. Lower values indicate smaller error.

Horizon H (s)	RC-Only	RC + Residual Mean	RC + Mean + Equilibrium Correction
1 (0.05)	0.01743	0.00251	0.00288
5 (0.25)	0.13693	0.01641	0.01748
10 (0.50)	0.51515	0.04615	0.04653
20 (1.00)	1.56246	0.16687	0.15151

As shown in Table 3, residual-mean compensation reduces the normalized endpoint RMSE of RC-only prediction from 0.01743 to 0.00251 at one step and from 1.56246 to 0.16687 at the 20-step MPC horizon. Its relative improvement over RC-only is substantial for all four evaluated horizons; nevertheless, all three predictors exhibit increasing free-running error with increasing horizon. The equilibrium-corrected predictor obtains 0.00288, 0.01748, and 0.04653 at one, five, and ten steps, respectively, slightly above the uncorrected mean model's 0.00251, 0.01641, and 0.04615. At 20 steps, the corrected model gives 0.15151 versus 0.16687 (a 9.20% lower sample RMSE). This 20-step difference is descriptive: a paired trajectory-level bootstrap interval for the difference includes zero and does not establish a statistically reliable correction-specific advantage. Likewise, the correction is not uniformly superior across individual physical-state RMSEs. Its principal justification is preservation of the known zero equilibrium, independently evaluated by the regulation experiments.

A separate set of 500 held-out, randomly generated one-step samples gave normalized RMSE values of 0.03470, 0.00271, and 0.00397 for RC-only, RC + residual mean, and RC + mean + correction, respectively. This provides a second test of mean-compensation accuracy but is not a substitute for the continuous-trajectory multi-step results. Figure 6 illustrates one representative excited-input 20-step rollout; its four-state traces are an example rather than ensemble means.

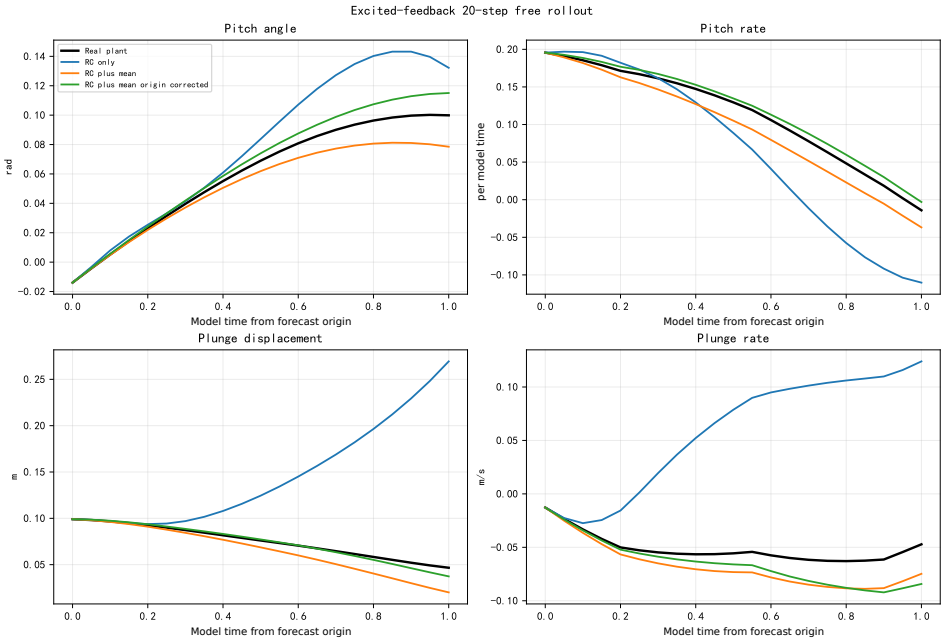


Figure 6. Representative independent held-out excited-input trajectory: physical plant versus 20-step free rollouts of RC-only, RC + residual mean, and equilibrium-corrected RC + mean. Future prescribed control inputs are supplied to each predictor, but no future physical states are used to correct the rollout. This individual example is not the basis for the aggregate RMSE values in Table 3.

Reservoir-state handling is a material limitation. The primary evaluation resets the reservoir to zero at each forecast start, reproducing the starting context of the implemented MPC prediction path. An additional diagnostic instead preheated the reservoir with observed past trajectory states before the forecast began; subsequent forecast steps still received no future observed states. For the equilibrium-corrected predictor, the normalized endpoint RMSE increased from 0.00288 to 0.00534 (one step) and from 0.15151 to 0.21429 (20 steps) under this history-warmed initialization. Because the training sequence consisted of shuffled independent one-step samples, such reservoir-state sensitivity was not unexpected; however, the test does not establish a unique causal mechanism. These findings limit claims about learned temporal memory and do not demonstrate reliable online history propagation. In this configuration, the fixed nonlinear reservoir acts primarily as a feature expansion for one-step state–input regression, rather than an experimentally validated benefit from recurrent temporal memory. A non-recurrent MLP provides a relevant alternative for this fully observed four-state transition map. The supplementary matched-data baseline below directly addresses this comparison. The reported prediction differences do not establish a benefit from reservoir memory, and temporally consistent RC training would require independent retraining and closed-loop validation.

4.4. Supplementary Matched-Data MLP Predictor Baseline

To assess whether the recurrent reservoir is necessary for accurate state-transition prediction, an independent feedforward MLP-only predictor was trained on exactly the same 4800 fitting transitions as the original RC, with the same 1200 held-out validation transitions. The data generator, perturbed plant, RK4 sampling interval ($\Delta t = 0.05$ model-time units), training process-noise model (velocity-coordinate intensity 0.020), random data seed, and shuffled 80%/20% split were preserved. The RC implementation and its shuffled independent-transition reservoir-state handling were not modified. The feedforward comparator mapped $[x_k^T, u_k^T]^T$ to the four-state increment $x_{k+1} - x_k$ using two hidden layers of 64 ReLU units, Adam optimization (learning rate 10^{-3}, maximum 600 iterations), componentwise StandardScaler preprocessing, and the same ± 0.20 state-increment clipping used in the RC readout. The RC residual-mean network retained its original $(64, 64)$ architecture; no diffusion-generated residuals, KNN risk, or MPC optimization were involved in this offline prediction ablation.

The original prediction-validation archive retained the 20-trajectory error summaries and 100 forecast-window metrics but not the complete original test trajectories and prescribed input sequences. It was therefore not possible to evaluate the new MLP on exactly those archived input sequences. Rather than mixing outcomes from different test sets, we newly generated 20 independent, deterministic, dynamically continuous trajectories (ten excited/feedback and ten unforced), with 70 steps per trajectory, and recomputed all four predictors on the same new observations and recorded future control sequences. The initial-state and excitation-draw protocol for these supplementary trajectories is documented in the accompanying reproducibility script and is not claimed to duplicate the unavailable archived trajectories. Each trajectory provided the same forecast starts $k \in \{10, 20, 30, 40, 50\}$ and freely rolled horizons $H \in \{1, 5, 10, 20\}$, using Equation (31). The RC reservoir was initialized to zero at every forecast start to reproduce the original MPC forecast context. No future true states were used during the rollouts. All methods were trained once with fixed random seeds.

The matched-data prediction results for the four predictor variants on these new, paired test trajectories are reported in Table 4.

Table 4. Supplementary original-protocol MLP comparison on *new, paired* held-out continuous trajectories (20 trajectories, five forecast windows each). Normalized four-state endpoint RMSE; smaller is better. These supplementary test trajectories differ from those used to produce Table 3, so the numerical values should not be combined across tables.

Horizon H (Model-Time Units)	MLP-Only	RC-Only	RC + Mean	RC + Mean + EQ
1 (0.05)	0.00692	0.01732	0.00217	0.00332
5 (0.25)	0.03134	0.10982	0.01261	0.01844
10 (0.50)	0.06409	0.42107	0.03418	0.04221
20 (1.00)	0.16628	1.46163	0.10698	0.11973

In this specific matched-data experiment, MLP-only outperformed RC-only at all four horizons, indicating that the original shuffled-sample RC configuration does not demonstrate an accuracy advantage over a non-recurrent MLP. The RC plus residual-mean predictor achieved lower endpoint errors than the direct MLP under these test conditions, suggesting that the combined predictor, rather than recurrent memory alone, accounts for the favorable accuracy. The MLP comparison is not a claim of universal superiority of either model: it uses one training initialization, a fixed MLP architecture, a newly generated test-trajectory distribution, and no MLP-based MPC closed-loop trials. In particular, these results do not substantiate RC-specific temporal-memory benefits or establish a controller-level advantage over an MLP-driven MPC. The archived original-protocol prediction results in Table 3 remain separate and unchanged.

4.5. Equilibrium-Correction Ablation

Before enforcing Equation (18), the deterministic RC plus mean-residual predictor has a nonzero predicted increment at the physical origin. In an earlier single-run noise-free diagnostic, the uncorrected fixed and adaptive controllers produced terminal normalized errors of 0.047849 and 0.042390 and did not enter the strict sustained-regulation band. The equilibrium-corrected counterpart reduced these errors to 0.000418 and 0.000144 and achieved sustained-regulation times of 2.80 and 1.95 model-time units, respectively. These are individual deterministic diagnostics rather than stability proofs. Subsequent stochastic comparisons use the corrected predictor for both weight strategies so that its benefit is not incorrectly attributed to risk adaptation.

4.6. Representative Four-State Response and Control Input

Figure 7 shows four closed-loop state components under independent Gaussian noise at $\sigma = 0.03$. The solid curves are the means of five paired realizations (1010–1014), and the envelopes denote one sample standard deviation. The adaptive weight strategy tends to reduce the initial pitch and plunge excursion faster, while both controllers maintain small residual fluctuations under continuing process disturbance. These ensemble curves do not by themselves demonstrate that every sample path satisfies the sustained-regulation criterion, which is therefore tabulated separately.

4.7. Robustness Under White and Temporally Correlated Noise

Table 5 summarizes the four matched-noise conditions. Under white noise at $\sigma = 0.03$, the adaptive controller has 1.66% lower normalized ISE, 4.35% less realized input effort, and a mean 5% sustained-regulation time of 1.91 versus 2.16 model-time units for frozen Q/P; both meet the criterion in all five runs. However, adaptive weighting has 1.57% higher raw ISE, emphasizing that normalization changes relative weightings among physical states.

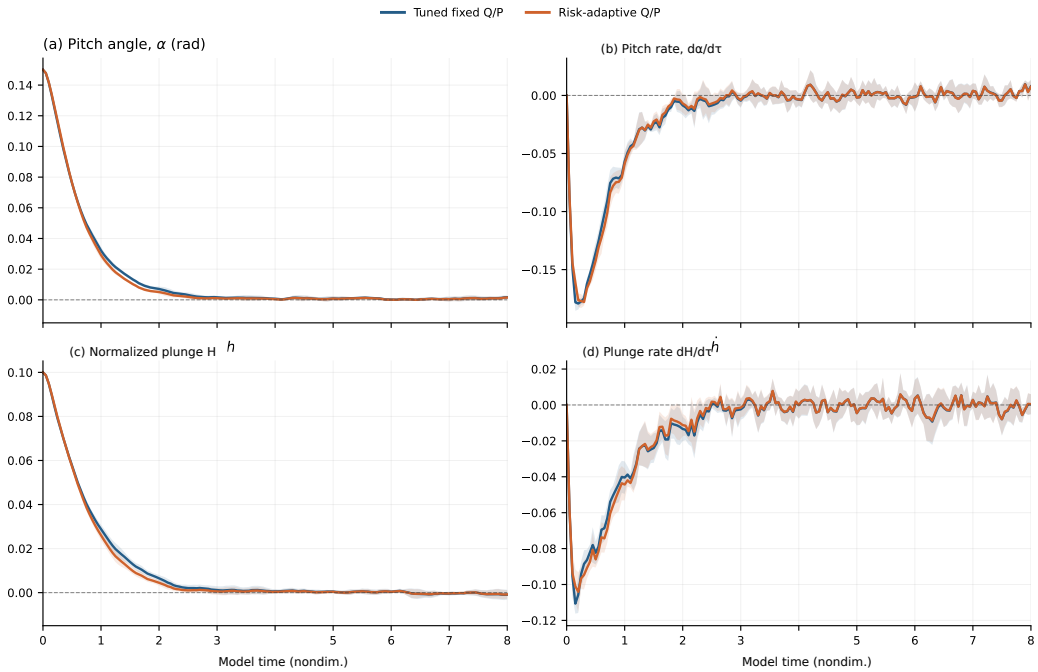


Figure 7. Four-state mean responses in nondimensional model time under white process disturbance ($\sigma = 0.03$): (**a**) pitch angle α; (**b**) pitch rate $d\alpha/d\tau$; (**c**) normalized plunge displacement H; (**d**) normalized plunge rate $dH/d\tau$. The blue and orange curves represent the tuned fixed-Q/P and KNN risk-adaptive Q/P controllers, respectively. Solid curves show the means of five paired realizations (seeds 1010–1014), and shaded regions indicate $\pm$ one standard deviation.

Table 5. Paired numerical comparison under four process noise conditions (five held-out realizations each). All metrics except success counts are five-run means.

Disturbance	Weights	J_N	J_{raw}	E_{RMS}	E_u	5% Success
White $\sigma = 0.03$	Frozen	0.158130	0.038011	0.00980	1.31298	5/5
	Adaptive	0.155507	0.038609	0.00901	1.25580	5/5
White $\sigma = 0.06$	Frozen	0.159585	0.041235	0.01919	2.05702	3/5
	Adaptive	0.158167	0.041675	0.01798	2.04629	5/5
AR(1) $\sigma = 0.03$	Frozen	0.160058	0.039325	0.02204	1.71818	2/5
	Adaptive	0.156537	0.039789	0.01925	1.65890	3/5
AR(1) $\sigma = 0.06$	Frozen	0.169485	0.045858	0.04259	3.64656	1/5
	Adaptive	0.167223	0.045740	0.03877	3.62512	1/5

With white noise at $\sigma = 0.06$, the mean late-window normalized RMS is lower for adaptive weights by 6.31%, and sustained regulation is attained in 5/5 versus 3/5 runs. The normalized ISE difference is smaller (0.89%), and its exploratory paired uncertainty interval includes zero. These five-run results do not establish a general difference in regulation-success probabilities.

With AR(1) disturbances, the temporal persistence considerably increases late-stage errors. At $\sigma = 0.03$, sustained-regulation successes are 2/5 and 3/5 for frozen and adaptive Q/P, respectively, with 12.65% lower mean RMS under adaptive weighting. At $\sigma = 0.06$, both succeed in only 1/5 runs; the mean late-stage RMS remains 8.96% lower for adaptive Q/P, but this does not establish guaranteed regulation under persistent disturbances. Both raw ISE and normalized ISE must be considered: improvement is not uniform across metrics or realizations.

Paired effect sizes and small-sample uncertainty

For each noise condition, the five prespecified seeds 1010–1014 form paired adaptive-minus-frozen comparisons. Table 6 reports empirical mean differences and exploratory 95% paired Student's t intervals for normalized ISE and late-window normalized RMS; all other paired metrics, including raw ISE, effort, and terminal error, are provided in the reproducible CSV. For the reference independent Gaussian setting, the mean normalized-ISE difference is -0.002623 (paired t interval $[-0.003540, -0.001705]$), whereas the mean raw-ISE difference is $+0.000597$ (paired t interval $[+0.000424, +0.000771]$). All five reference-condition seeds favor adaptive weights on normalized ISE, but all five favor frozen weights on raw ISE. Given only five pairs, the parametric t intervals rely on particularly strong distributional assumptions. The exact two-sided sign-flip test gives $p = 0.0625$ for both uniformly signed differences, and no multiple-comparison correction was applied to these exploratory analyses. In particular, neither the parametric intervals nor a five-run sample establish broadly reproducible or generally statistically significant improvements. Regulation-success rates are reported as counts rather than interpreted as precise probabilities; settling times are paired only when both controllers satisfy the stated criterion. Additional independent realizations would be necessary for stronger inference.

Table 6. Exploratory paired mean differences, adaptive-minus-tuned fixed Q/P, based on five independent held-out disturbance seeds per condition. Square brackets show two-sided 95% paired Student's t intervals ($n = 5$); their distributional assumptions are consequential at this sample size. Negative values indicate reductions. Exact two-sided sign-flip tests and bootstrap intervals for each metric, including raw ISE and control effort, are provided in the accompanying CSV.

Disturbance	Δ Normalized ISE [95% CI]	Δ Late-Stage Normalized RMS [95% CI]
White $\sigma = 0.03$	-0.00262 $[-0.00354, -0.00171]$	-0.00079 $[-0.00111, -0.00047]$
White $\sigma = 0.06$	-0.00142 $[-0.00455, +0.00172]$	-0.00121 $[-0.00173, -0.00069]$
AR(1) $\sigma = 0.03$	-0.00352 $[-0.00609, -0.00095]$	-0.00279 $[-0.00351, -0.00207]$
AR(1) $\sigma = 0.06$	-0.00226 $[-0.00802, +0.00350]$	-0.00381 $[-0.00626, -0.00137]$

The late-stage normalized state-error trajectories for all four disturbance conditions are shown in Figure 8.

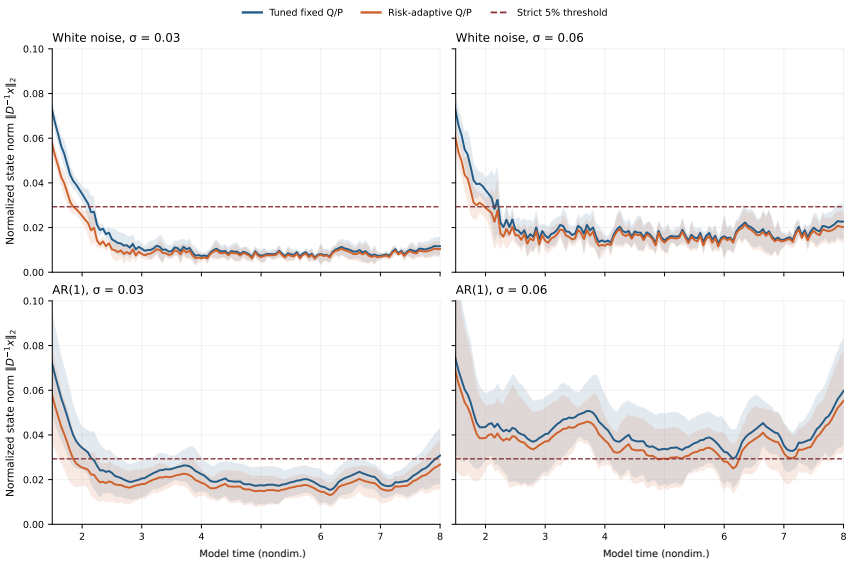


Figure 8. Late-stage normalized state-error responses across two independent Gaussian and two correlated AR(1) noise conditions. Each curve is the five-seed ensemble mean.

4.8. Representative Extended Closed-Loop Time-Horizon Check

A single prespecified closed-loop realization lasting 24 model-time units was evaluated using the actual parameter-perturbed plant at $Q_f = 8$, process noise with $\sigma = 0.03$ and seed 1010, the previously trained equilibrium-consistent RC/residual-mean predictor, and the risk-adaptive Q/P strategy with $R = R_0 = \mathrm{diag}(0.10, 0.10)$. The initial state was $[0.15, 0, 0.10, 0]^\mathsf{T}$, the sampling interval was 0.05 model-time units, and the MPC prediction horizon remained at 20 steps. This extension spans more than two periods of the approximately 9.66-model-time-unit uncontrolled oscillation in Section 2; it is an illustrative trajectory rather than an additional five-seed robustness test. The descriptive performance metrics for this extended realization are given in Table 7.

Table 7. Single realization over 24 model-time units; descriptive results only. The control-effort integral covers 24 model-time units and is not directly comparable with 8 model-time units effort integrals.

Metric	Value
Normalized terminal error	0.01430
Normalized RMS over final 2 model-time units	0.01198
Largest normalized error during model time 8–24	0.01943
Fraction of states outside strict 5% band during model time 8–24	0.00%
Strict 5% sustained regulation by model time 24	yes
Strict 5% sustained-regulation time (model units)	2.15
Control effort $\int_0^{24} u^\mathsf{T} u \, \mathrm{d}\tau$	1.83793
SLSQP successful solves	480/480

Figure 9 shows the four state trajectories, normalized state error, and control-input histories for the extended closed-loop realization.

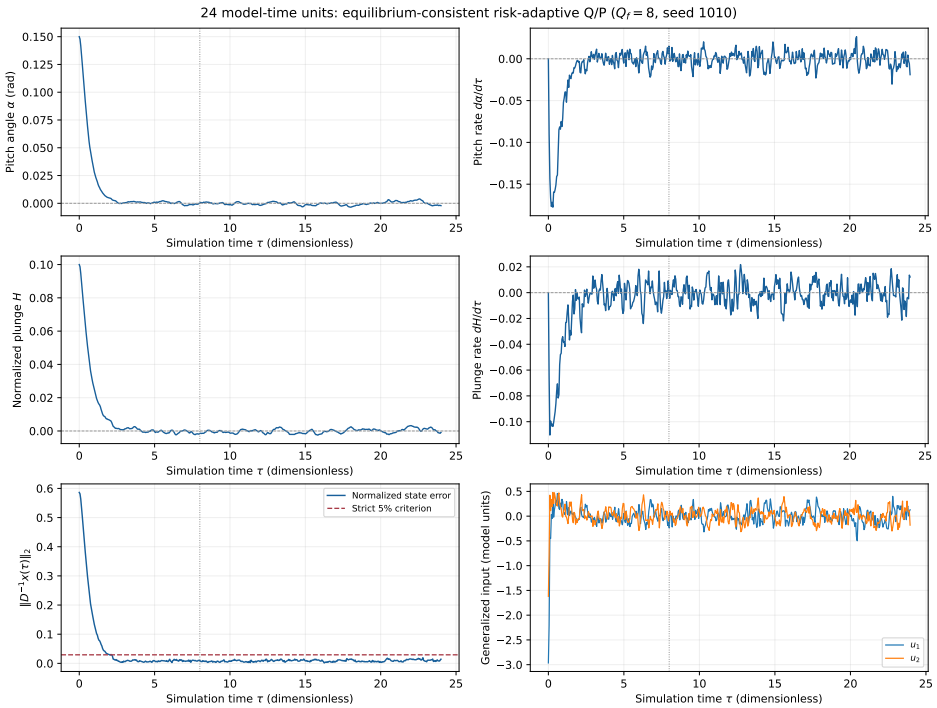


Figure 9. Closed-loop realization over 24 model-time units (adaptive Q/P, seed 1010, white process disturbance $\sigma = 0.03$). Four modeled state components, normalized state error, and two control inputs are shown. The vertical dotted marker at 8 model-time units identifies the length of the systematic ensemble simulations; the dashed horizontal line shows the original strict 5% normalized error threshold. The figure is not an ensemble mean or a stochastic stability proof.

The run remained within the predeclared strict 5% band after 2.15 model-time units, including the entire interval $8 \leq \tau \leq 24$. Its first eight model-time units reproduced the archived seed-1010 states and control inputs exactly. This one realization cannot establish a stochastic success probability or asymptotic stability.

The controller's source code implements the common control penalty $R_0 = 2\,\mathrm{diag}(0.05, 0.05) = \mathrm{diag}(0.10, 0.10)$, not twice the R_0 defined in this manuscript. The first eight model-time units match the archived seed-1010 systematic trajectory, confirming that the actual optimization penalty was not changed for this extension.

4.9. Independent Diagnostic Evaluation of KNN Risk Score

We evaluated the unchanged KNN score on independent, newly generated one-step transitions of the same parameter-perturbed plant. The in-domain (ID) test used the training state-range distribution with independent random draws; a separate out-of-domain (OOD) test expanded the state range by a factor of 1.5 and retained points outside the original state bounds. Both tests used the original training-level process noise, but no test transition was added to the KNN library, residual network, or quantile calibration. The controller and previously reported closed-loop results are unchanged.

The primary diagnostic used one-step RC states initialized to zero (matching the online risk-query convention); a batch-replay predictor was evaluated separately to examine sensitivity to the training/calibration feature-generation procedure. With held-out target state x_{i+1} and predicted origin-consistent mean $\widehat{F}_c(x_i, u_i)$, the realized normalized error and score are

$$\begin{aligned} E_i &= \left\| D^{-1}\left[x_{i+1} - \widehat{F}_c(x_i, u_i)\right] \right\|_2, \\ \rho_i &= \mathrm{tr}\,\Sigma_{\mathrm{KNN}}(c_i), \\ D &= \mathrm{diag}(0.35, 1.80, 0.25, 1.80). \end{aligned} \tag{32}$$

We assessed Spearman's rank correlation $r_s(\rho, E)$, area under the ROC curve (AUROC) for identifying the highest-quartile errors within each test set, and the error ratio between the highest- and lowest-risk quintiles. These measure rank discrimination only. We also report the fraction of held-out errors inside a nominal 95% Gaussian four-dimensional error ellipsoid based on Σ_{KNN}, as a descriptive coverage diagnostic. This Gaussian reference is not a proven distributional model for the residuals or a chance-constraint guarantee. The diagnostic results for both test sets and reservoir-state initialization procedures are summarized in Table 8.

Table 8. Independent diagnostics of the frozen KNN trace score. AUROC 0.5 indicates chance ranking of the test-set top-quartile one-step errors.

Test Set	RC Initialization	$r_s(\rho, E)$	AUROC	High/Low Error Ratio
ID	Zero state	0.039	0.511	1.033
ID	Batch replay	0.073	0.548	1.111
OOD	Zero state	0.204	0.643	2.438
OOD	Batch replay	0.217	0.654	2.448

Figure 10 shows the binned relationship between the raw KNN covariance trace and the errors on the independent test conditions.

The online-like zero-reservoir ID risk–error rank correlation was 0.039, with AUROC 0.511. The empirical Gaussian-reference ellipsoid coverage for the origin-corrected mean was 97.1% in-domain and 55.0% OOD. These results must be interpreted jointly with OOD discrimination and the neighbor-distance diagnostic. Because KNN covariance is computed from in-sample

residual errors in unnormalized output units, neither the scalar trace nor validation-quantile rescaling constitutes calibrated confidence or a bound on realized model error.

As an explicit systematic-bias counterexample, adding a synthetic +0.025 rad constant offset to the held-out pitch next-state label increased mean normalized error from 0.0050 to 0.0704 while leaving the query and raw KNN risk exactly unchanged. This synthetic stress test is not another physical plant experiment; it demonstrates that an unchanged covariance lookup cannot, by construction, directly detect an output-only bias.

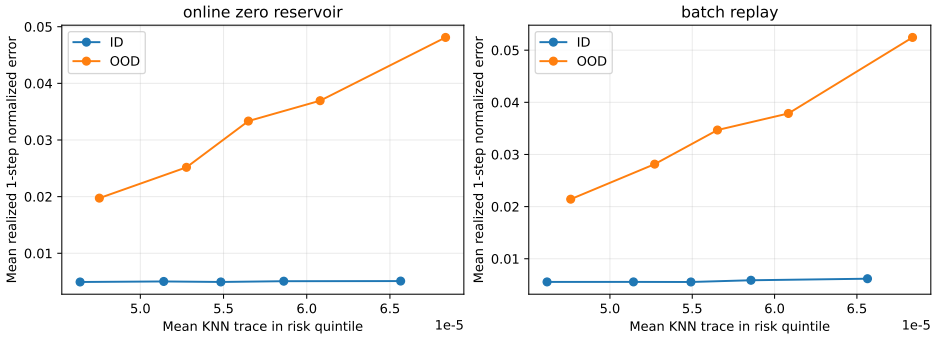


Figure 10. Binned independent-test errors versus raw KNN covariance trace. Bins contain equal numbers of test conditions; this is a discrimination diagnostic rather than a calibration or coverage plot.

4.10. Risk Interpretation, Computational Cost, and Limitations

The KNN score measures local residual-error dispersion, not external noise variance. Independent validation (Section 4.9) finds near-chance in-domain error discrimination (AUROC 0.511 for the online-like zero-reservoir condition) and only moderate discrimination on out-of-range test conditions (AUROC 0.643). Consequently, the observed closed-loop performance advantage over the tuned constant-weight comparator does not establish that the KNN trace is a calibrated or generally reliable predictor of realized model error. That fixed-weight comparison also does not isolate informative KNN risk ordering from the mere use of changing state and terminal weights. Section 4.11 therefore reports an additional exploratory, fully KNN-independent state-feedback Q/P adaptation baseline. Its findings reveal a regulation–effort trade-off, but cannot isolate the informational value of KNN covariance or establish the superiority of either policy. The KNN risk's observed mean also changes little when imposed process-noise intensity increases, so the fixed offline library is not shown to track gust intensity. Forecast accuracy remains sensitive to reservoir initialization because the RC was fitted to shuffled one-step transitions rather than chronological physical sequences.

Across the four systematic disturbance conditions, all 40 closed-loop runs completed with 6400/6400 successful SLSQP status returns and no recorded empirical state-limit violations. Each optimization nevertheless required roughly 9.6–11.0 SI seconds of wall-clock time on average, while the plant simulation advanced by 0.05 dimensionless model-time units per control step. Without a documented reference-frequency conversion, these two intervals cannot be directly compared as physical times; the solver cost and absence of delay modeling preclude any demonstrated real-time feasibility. This study therefore establishes only offline numerical behavior. The later workstation audit is documented above but is not independently matched to all archived solver-time recordings. The AR(1) disturbances are synthetic, the tested operating parameter is $Q_f = 8$, and neither solver success nor a single 24-model-time-unit realization establishes stochastic stability or a general chance-constraint guarantee. Floquet stability of the perturbed plant's unforced orbit also remains unverified.

4.11. Exploratory Ablation Against a Fully KNN-Independent Adaptive-Weight Baseline

A comparison against fixed Q/P matrices alone cannot determine whether the incremental control behavior results from informative KNN residual covariance or simply from changing the state-error weights. To provide a direct, albeit exploratory, comparator, we additionally implemented a state-feedback adaptive-Q/P controller whose scheduling rule contains no KNN neighbor queries, covariance values, KNN risk trajectory, or fitted KNN calibration parameters. This controller used the same equilibrium-corrected RC/residual-mean predictor, MPC horizon, input penalties, input bounds, plant model, and weight-update map as the original KNN-based controller.

For $x_k = [\alpha_k, \dot{\alpha}_k, h_k, \dot{h}_k]^T$, the independent scheduling policy is

$$e_k = \left\| D^{-1} x_k \right\|_2, \quad D = \mathrm{diag}(0.35, 1.80, 0.25, 1.80), \quad \rho_k^{\mathrm{state}} = \mathrm{clip}(0.30 + 0.25 e_k, 0, 1). \tag{33}$$

The coefficient ρ_k^{state} replaces $\rho_{\mathrm{eff},k}$ in Equation (23) so that

$$Q_k = Q_0 + 120 \rho_k^{\mathrm{state}} W, \qquad P_k = P_0 + 280 \rho_k^{\mathrm{state}} W, \tag{34}$$

with unchanged R_0 and S_0. Unlike the previous development-stage KNN-derived temporal templates, this is an entirely KNN-independent state-based controller. The offset 0.30 and gain 0.25 are prescribed exploratory coefficients; they were selected after inspecting a separate earlier pilot (seeds 1030–1031), which suggested that a zero-baseline state-only scheduler rapidly decreased its weight near equilibrium. Neither coefficient was fitted to KNN risk or covariance data, and seeds 1032–1033 below were not used to choose these coefficients. Because model selection was exploratory, the test was not a preregistered or comprehensive adaptive-gain benchmark.

The comparison used the same parameter-perturbed wing-section plant at $Q_f = 8$ and independent Gaussian velocity-channel process noise with $\sigma = 0.03$. Two previously unused paired disturbance realizations (seeds 1032 and 1033) were simulated for 160 MPC updates each ($\Delta t = 0.05$ model-time units), using a 20-step horizon. The fixed-Q/P comparator retained $\rho = 0.24$ as in Equation (25). Within each seed, the three controllers were subjected to identical noise increments, verified by stored hashes. Across the six trajectories, all 960 SLSQP calls returned successful status. To avoid evaluating four identical deterministic-mean objective copies, the additional ablation implementation evaluated a single copy of that same cost. Although algebraically equivalent in exact arithmetic, this was a computational implementation change relative to the archived systematic test, so the new two-seed results were treated as a separate exploratory experiment rather than merged with the prior five-seed tables.

Table 9 shows that the simple KNN-free state scheduler produced a smaller mean normalized ISE than the KNN policy (0.151199 versus 0.155306), whereas KNN weighting used less control effort (1.231068 versus 1.579680) and gave a slightly smaller late-stage normalized RMS error (0.008410 versus 0.008832). Their mean terminal normalized errors were close (0.008856 and 0.008982). The independent state's input effort was approximately 28.3% higher than KNN's, while its mean scheduling coefficient was lower (0.315301 versus 0.450520). Consequently, the experiment did not match input effort and could not establish that any observed differences were caused specifically by KNN covariance information rather than different weight magnitudes and time courses. The rounded raw ISE means of the two adaptive controllers also appear identical at six decimal places; their unrounded values differ and should not be interpreted as exact equality.

The additional ablation was restricted to two paired disturbances under one white-noise amplitude and is reported only descriptively. In particular, no significance, equivalence, cross-disturbance robustness, or KNN-specific superiority was inferred from $n = 2$. It

demonstrates that a genuinely KNN-independent adaptive-weight strategy is feasible and can attain lower normalized ISE at an increased control cost, while leaving the independent incremental contribution of the KNN covariance score unresolved.

Table 9. Exploratory white-noise ($\sigma = 0.03$) closed-loop comparison on two independent paired seeds (1032, 1033). Entries are sample means ($n = 2$ per controller); smaller is better. Normalized and raw metrics have different state scalingw. Neither matched control effort nor statistical significance is claimed.

Metric	KNN Q/P	Independent State Q/P	Fixed Q/P
Normalized ISE	0.155306	0.151199	0.159769
Raw ISE	0.038173	0.038173	0.037469
Terminal normalized error	0.008856	0.008982	0.009012
Terminal raw error	0.013620	0.013857	0.013956
Late-stage normalized RMS	0.008410	0.008832	0.009070
Input effort	1.231068	1.579680	1.254760
Strict 5% settling time (model units)	1.925	2.050	2.150
Mean effective scheduling coefficient	0.450520	0.315301	0.240000

5. Conclusions

This study evaluated an equilibrium-corrected RC/residual-mean predictor with empirically adaptive-weight MPC for nonlinear wing-flutter suppression. The implemented optimizer propagates one deterministic corrected-mean trajectory; diffusion sampling and scenario-count adaptation do not contribute to the reported closed-loop results. Independent one-step and free-running multi-step tests showed substantially smaller forecast error after residual-mean compensation, while reservoir initialization and longer-horizon error accumulation remain limitations. The equilibrium correction preserved the known zero equilibrium of the one-step map and reduced offset in noise-free closed-loop diagnostics.

Against a separately tuned and locked constant-Q/P comparator, KNN-based adaptive weighting reduced normalized ISE by 1.66%, shortened average strict 5% regulation time from 2.16 to 1.91 model-time units, and used 4.35% less input effort in the reference white-noise condition, based on five independent paired seeds. Its mean late-stage normalized RMS was lower in all four original noise conditions, although raw errors and individual realizations did not uniformly improve. The independent KNN diagnostic showed weak in-domain ranking of realized prediction errors (AUROC 0.511), so KNN covariance is interpreted only as an empirical weighting signal, not as calibrated predictive uncertainty.

A new, exploratory two-seed white-noise ablation compared the KNN-weighted controller with a fully KNN-independent state-feedback Q/P scheduler and the fixed-weight comparator. The independent scheduler reduced normalized ISE relative to KNN weighting (0.151199 versus 0.155306) but required more input effort (1.579680 versus 1.231068), and its late-stage normalized RMS was slightly larger (0.008832 versus 0.008410). Therefore, the results reveal a regulation–effort trade-off and do not establish a general incremental benefit attributable uniquely to KNN covariance information. Only one white-noise amplitude and two independent paired seeds were examined in the additional ablation; larger, effort-matched tests would be required for stronger inference. Neither these offline experiments nor the illustrative extended trajectory establish stochastic stability, real-time deployability, or superiority over an independently tuned physics-based MPC. Future work should address chronologically consistent RC training, independently calibrated and effort-matched adaptive-weight baselines, more disturbance realizations, physically validated gust models, and computationally efficient control.

Supplementary Materials: The following supporting information can be downloaded at https://www.mdpi.com/article/10.3390/aerospace13111030/s1: Supplementary File S1: Selected Python and MATLAB scripts, a JSON configuration file, and archived MATCONT continuation data (MAT and CSV files) supporting the numerical analyses.

Author Contributions: Conceptualization, H.Z. and Y.L.; methodology, H.Z., Y.L. and X.L.; software, H.Z. and J.X.; validation, H.Z. and J.X.; formal analysis, H.Z. and X.L.; investigation, H.Z. and J.X.; data curation, H.Z. and J.X.; writing—original draft preparation, H.Z.; writing—review and editing, H.Z., J.X., Y.L. and X.L.; visualization, H.Z. and J.X.; supervision, Y.L. and X.L.; funding acquisition, Y.L. and X.L. All authors have read and agreed to the published version of this manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (grant numbers 12302035, 12572038, and 12372030) , the Fundamental Research Funds for the Central Universities (grant number 30925010411) and Young Science and Technology Talent Support Project of Jiangsu Province (Grant No. JSTJ-2026-219).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Selected code, configuration information, and archived MATCONT continuation data are provided in Supplementary File S1. Additional numerical simulation data generated and analyzed during this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Bisplinghoff, R.L.; Ashley, H.; Halfman, R.L. *Aeroelasticity*; Addison-Wesley: Reading, MA, USA, 2013.
2. Lee, B.H.K.; Price, S.J.; Wong, Y.S. Nonlinear aeroelastic analysis of airfoils: Bifurcation and chaos. *Prog. Aerosp. Sci.* **1999**, *35*, 205–334. https://doi.org/10.1016/S0376-0421(98)00015-3.
3. Hu, H.; Zhao, Y.; Huang, R. Studies on aeroelastic analysis and control of aircraft structures. *Chin. J. Theor. Appl. Mech.* **2016**, *48*, 1–27. (In Chinese) https://doi.org/10.6052/0459-1879-15-423.
4. Huang, R.; Hu, H. Nonlinear aeroservoelastic mechanics of aircraft. *Adv. Mech.* **2021**, *51*, 428–466. (In Chinese)
5. Fonzi, N.; Brunton, S.L.; Fasel, U. Data-driven nonlinear aeroelastic models of morphing wings for control. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **2020**, *476*, 20200079. https://doi.org/10.1098/rspa.2020.0079.
6. Hickner, M.; Fasel, U.; Nair, A.G.; Brunton, B.W.; Brunton, S.L. Data-Driven Unsteady Aeroelastic Modeling for Control. *AIAA J.* **2023**, *61*, 780–792. https://doi.org/10.2514/1.J061518.
7. Kou, J.; Zhang, W. Data-driven modeling for unsteady aerodynamics and aeroelasticity. *Prog. Aerosp. Sci.* **2021**, *125*, 100725. https://doi.org/10.1016/j.paerosci.2021.100725.
8. Li, K.; Kou, J.; Zhang, W. Deep neural network for unsteady aerodynamic and aeroelastic modeling across multiple Mach numbers. *Nonlinear Dyn.* **2019**, *96*, 2157–2177. https://doi.org/10.1007/s11071-019-04915-9.
9. Yao, X.J.; Huang, R.; Hu, H.Y. Data-driven modeling of transonic unsteady flows and efficient analysis of fluid-structure stability. *J. Fluids Struct.* **2022**, *111*, 103549. https://doi.org/10.1016/j.jfluidstructs.2022.103549.
10. Hewing, L.; Wabersich, K.P.; Menner, M.; Zeilinger, M.N. Learning-Based Model Predictive Control: Toward Safe Learning in Control. *Annu. Rev. Control. Robot. Auton. Syst.* **2020**, *3*, 269–296. https://doi.org/10.1146/annurev-control-090419-075625.
11. Berberich, J.; Allgöwer, F. An overview of systems-theoretic guarantees in data-driven model predictive control. *Annu. Rev. Control. Robot. Auton. Syst.* **2025**, *8*, 77–100. https://doi.org/10.1146/annurev-control-030323-024328.
12. Jaeger, H. *The "Echo State" Approach to Analysing and Training Recurrent Neural Networks-With An Erratum Note*; GMD Report 148; German National Research Center for Information Technology: Bonn, Germany, 2001.
13. Lukoševičius, M.; Jaeger, H. Reservoir computing approaches to recurrent neural network training. *Comput. Sci. Rev.* **2009**, *3*, 127–149. https://doi.org/10.1016/j.cosrev.2009.03.005.
14. Ho, J.; Jain, A.; Abbeel, P. Denoising diffusion probabilistic models. In Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–12 December 2020; Volume 33, pp. 6840–6851.
15. Song, J.; Meng, C.; Ermon, S. Denoising diffusion implicit models. In Proceedings of the International Conference on Learning Representations, Online, 3–7 May 2021.

16. Rasul, K.; Seward, C.; Schuster, I.; Vollgraf, R. Autoregressive Denoising Diffusion Models for Multivariate Probabilistic Time Series Forecasting. In Proceedings of the 38th International Conference on Machine Learning, Online, 18–24 July 2021; Volume 139. https://doi.org/10.48550/arXiv.2101.12072.
17. Shen, L.; Kwok, J. Non-autoregressive Conditional Diffusion Models for Time Series Prediction. In Proceedings of the 40th International Conference on Machine Learning, Honolulu, HI, USA, 23–29 July 2023; Volume 202. https://doi.org/10.48550/arXiv.2306.05043.
18. Dyro, R.; Harrison, J.; Sharma, A.; Pavone, M. Particle MPC for Uncertain and Learning-Based Control. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Prague, Czech Republic, 27 September–1 October 2021; pp. 7127–7134. https://doi.org/10.48550/arXiv.2104.02213.
19. Li, D.; Xiang, J. Flutter control of aeroelastic system with freeplay nonlinearity. *J. Beijing Univ. Aeronaut. Astronaut.* **2007**, *33*, 640–643. (In Chinese)
20. Mao, Y.; Liu, X. Data-driven flutter suppression of noised airfoil models: Model identification and robust sliding mode control designed with multi-level control surfaces. *Nonlinear Dyn.* **2025**, *113*, 4235–4252. https://doi.org/10.1007/s11071-024-10463-8.
21. Mayne, D.Q.; Rawlings, J.B.; Rao, C.V.; Scokaert, P.O.M. Constrained model predictive control: Stability and optimality. *Automatica* **2000**, *36*, 789–814. https://doi.org/10.1016/S0005-1098(99)00214-9.
22. Rawlings, J.B.; Mayne, D.Q.; Diehl, M.M. *Model Predictive Control: Theory, Computation, and Design*, 2nd ed.; Nob Hill Publishing: Madison, WI, USA, 2017.
23. Breschi, V.; Chiuso, A.; Formentin, S. Data-driven predictive control in a stochastic setting: A unified framework. *Automatica* **2023**, *152*, 110961. https://doi.org/10.1016/j.automatica.2023.110961.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.