Active Internal Flow Control Using Adaptive Model Predictive Control with Online, Closed-Loop System Identification

Jacob C. Vander Schaaf*, Krzysztof J. Fidkowski†, and Dennis S. Bernstein‡
University of Michigan, Ann Arbor, Michigan, 48109

This paper uses predictive cost adaptive control (PCAC) for active flow control. PCAC is a discrete-time, indirect adaptive control technique based on model predictive control. Unlike model-based control, PCAC uses no prior model, and, unlike deep and reinforcement learning, PCAC uses no preflight data or offline learning. For rapid online multiple-input, multiple-output (MIMO) system identification, PCAC uses recursive least squares (RLS) with variable-rate forgetting. To determine control inputs, PCAC uses quadratic programming for receding horizon optimization, which allows PCAC to enforce physically meaningful magnitude and rate-saturation constraints. The present paper applies PCAC to 2D internal flow-control problems under "cold-start" conditions. The following flow-control scenarios are considered: 1) control of streamwise velocity within the boundary layer in a duct, and 2) control of streamwise velocity within the boundary layer of an axial cylinder in a duct.

I. Introduction

Active flow control remains a scientifically challenging problem with significant technological benefits for flight vehicles [1–11]. The present paper focuses on active flow control, which uses a feedback control algorithm to determine the desired actuator action based on sensor data. For model-based feedback control, available models may possess errors that degrade closed-loop performance [12, 13]. These errors include imprecise physical models and parameter values as well as errors due to time discretization and model reduction. Controller-synthesis techniques used for flow control include optimal control [14, 15], robust control [12], and machine-learning methods [10, 16].

The present paper focuses on data-driven control, where closed-loop system identification is performed during operation in response to actual, changing conditions with no prior modeling information. In particular, the present paper focuses on predictive cost adaptive control (PCAC) as an alternative to retrospective cost adaptive control considered in [17]. PCAC is a discrete-time, indirect adaptive control technique based on model predictive control. Unlike model-based control, PCAC uses no prior model, and, unlike deep and reinforcement learning, PCAC uses no preflight data or offline learning. For rapid online multiple-input, multiple-output (MIMO) system identification, PCAC uses recursive least squares (RLS) with variable-rate forgetting. To determine control inputs, PCAC uses quadratic programming for receding horizon optimization, which allows PCAC to enforce physically meaningful magnitude and rate-saturation constraints. In order to assess the performance of PCAC, we consider two flow control problems. In particular, we first consider control of streamwise velocity within a laminar boundary layer in a duct, followed by a more difficult scenario involving the boundary layer of an axial cylinder in a duct.

II. Review of Predictive Cost Adaptive Control

As shown in Figure 1, predictive cost adaptive control (PCAC) combines online identification with output-feedback model predictive control (MPC). PCAC uses no a priori modeling information aside from a suitable model order for system identification, nor does it use probing signals.

*Ph.D. Candidate, Department of Aerospace Engineering, jacobcvs@umich.edu.
†Professor, Department of Aerospace Engineering, kfid@umich.edu.
‡Professor, Department of Aerospace Engineering, dsbaero@umich.edu.

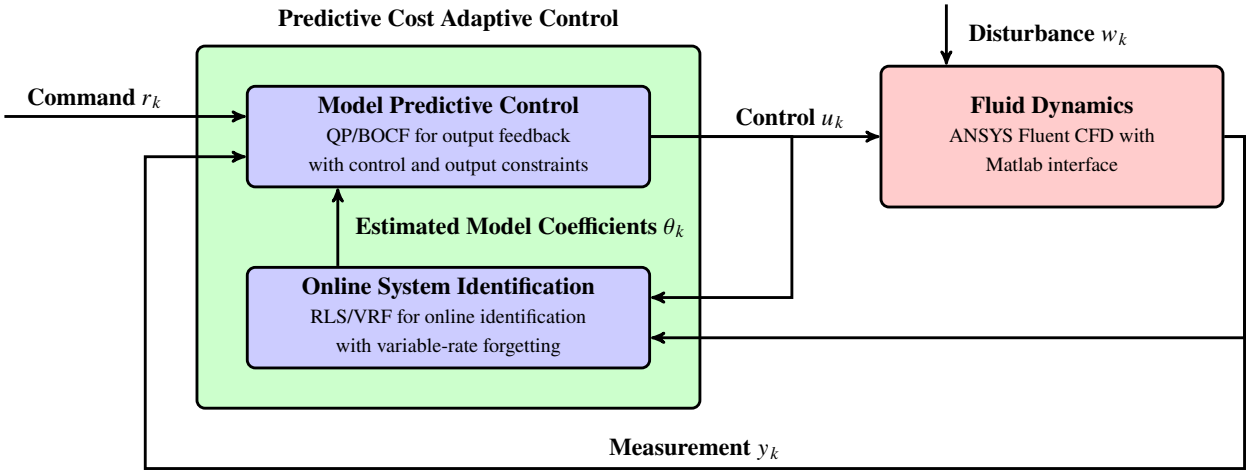

Fig. 1 PCAC block diagram. The online, closed-loop system identification is based on recursive least squares (RLS) with variable-rate forgetting (VRF). The model predictive control (MPC) algorithm, which is based on quadratic programming (QP), uses the estimated model coefficients θ_k to form a block-observable canonical form (BOCF) state-space model, which is used by QP to determine the control input u_k.

A. Online Identification

To perform online closed-loop system identification, we first consider the MIMO input-output model

$$\hat{y}_k = -\sum_{i=1}^{\hat{n}} \hat{F}_i y_{k-i} + \sum_{i=1}^{\hat{n}} \hat{G}_i u_{k-i}, \tag{1}$$

where $k \geq 0$ is the step, $\hat{n} \geq 1$ is the identification data window, $\hat{F}_i \in \mathbb{R}^{p \times p}$ and $\hat{G}_i \in \mathbb{R}^{p \times m}$ are the estimated model coefficients, and $u_k \in \mathbb{R}^m$, $y_k \in \mathbb{R}^p$, and $\hat{y}_k \in \mathbb{R}^p$ are the inputs, outputs, and predicted outputs. We estimate $\hat{F}_k$ and $\hat{G}_k$ online using recursive least squares with variable-rate forgetting (RLS/VRF), by minimizing the cost function [18]

$$J_k(\hat{\theta}) = \sum_{i=0}^{k} \frac{\rho_i}{\rho_k} z_i^{\mathrm{T}}(\hat{\theta}) z_i(\hat{\theta}) + \frac{1}{\rho_k} (\hat{\theta} - \theta_0)^{\mathrm{T}} P_0^{-1} (\hat{\theta} - \theta_0), \tag{2}$$

where $\rho_k \triangleq \prod_{j=0}^{k} \lambda_j^{-1} \in \mathbb{R}$, $\lambda_k \in (0, 1]$ is the forgetting factor, $P_0 \in \mathbb{R}^{[\hat{n}p(m+p)] \times [\hat{n}p(m+p)]}$ is positive definite, $\theta_0 \in \mathbb{R}^{[\hat{n}p(m+p)]}$ is the initial estimate of the coefficient vector, and the performance variable $z_i(\hat{\theta}) \in \mathbb{R}^p$ is defined as

$$z_k(\hat{\theta}) = y_k - \phi_k \hat{\theta}. \tag{3}$$

The vector $\hat{\theta} \in \mathbb{R}^{[\hat{n}p(m+p)]}$ of coefficients to be estimated is

$$\hat{\theta} \triangleq \mathrm{vec} \begin{bmatrix} \hat{F}_1 & \cdots & \hat{F}_{\hat{n}} & \hat{G}_1 & \cdots & \hat{G}_{\hat{n}} \end{bmatrix} = \mathrm{vec} \begin{bmatrix} \hat{\theta}_{\hat{F}} & \hat{\theta}_{\hat{G}} \end{bmatrix},$$

where $\hat{\theta}_{\hat{F}}$ and $\hat{\theta}_{\hat{G}}$ are the estimates of the denominator and numerator coefficients, defined by

$$\hat{\theta}_{\hat{F}} \triangleq \mathrm{vec} \begin{bmatrix} \hat{F}_1 & \cdots & \hat{F}_{\hat{n}} \end{bmatrix}, \tag{4}$$

$$\hat{\theta}_{\hat{G}} \triangleq \mathrm{vec} \begin{bmatrix} \hat{G}_1 & \cdots & \hat{G}_{\hat{n}} \end{bmatrix}. \tag{5}$$

With the regressor matrix $\phi_k \in \mathbb{R}^{p \times [\hat{n}p(m+p)]}$ defined by

$$\phi_k \triangleq \begin{bmatrix} -y_{k-1}^{\mathrm{T}} & \cdots & -y_{k-\hat{n}}^{\mathrm{T}} & u_{k-1}^{\mathrm{T}} & \cdots & u_{k-\hat{n}}^{\mathrm{T}} \end{bmatrix} \otimes I_p,$$

the global minimizer $\theta_{k+1} \triangleq \operatorname{argmin}_{\hat{\theta}} J_k(\hat{\theta})$ of (2) is

$$L_k = \lambda_k^{-1} P_k, \tag{6}$$

$$P_{k+1} = L_k - L_k \phi_k^{\mathrm{T}} (I_p + \phi_k L_k \phi_k^{\mathrm{T}})^{-1} \phi_k L_k, \tag{7}$$

$$\theta_{k+1} = \theta_k + P_{k+1} \phi_k^{\mathrm{T}} (y_k - \phi_k \theta_k). \tag{8}$$

The variable-rate forgetting factor λ_k is given by [19]

$$\lambda_k = \frac{1}{1 + \eta g(z_{k-\tau_{\mathrm{d}}}, \ldots, z_k) \mathbf{1}[g(z_{k-\tau_{\mathrm{d}}}, \ldots, z_k)]}, \tag{9}$$

where $\mathbf{1}\colon \mathbb{R} \to \{0, 1\}$ is the unit step function, and

$$g(z_{k-\tau_{\mathrm{d}}}, \ldots, z_k) \triangleq \sqrt{\frac{\tau_{\mathrm{n}}}{\tau_{\mathrm{d}}} \frac{\left(\Sigma_{\tau_{\mathrm{n}}}(z_{k-\tau_{\mathrm{n}}}, \ldots, z_k) \Sigma_{\tau_{\mathrm{d}}}(z_{k-\tau_{\mathrm{d}}}, \ldots, z_k)^{-1}\right)}{c}} - \sqrt{f},$$

where $\eta > 0$ and $p \leq \tau_{\mathrm{n}} < \tau_{\mathrm{d}}$ represent numerator and denominator window lengths. $\Sigma_{\tau_{\mathrm{n}}}$ and $\Sigma_{\tau_{\mathrm{d}}}$ are the sample variances of the respective window lengths, and the threshold constant f is described in [20, 21]. The constant c, based on the windows lengths is described in [20]. The estimator coefficients $\hat{\theta}$ can be written in the block observable canonical form with matrices $\hat{A}_k$, $\hat{B}_k$, and $\hat{C}_k$ given by

$$\hat{A}_k \triangleq \begin{bmatrix} -\hat{F}_{1,k} & I_p & \cdots & \cdots & 0_{p\times p} \\ \vdots & 0_{p\times p} & \ddots & & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0_{p\times p} \\ \vdots & \vdots & & \ddots & I_p \\ -\hat{F}_{\hat{n},k} & 0_{p\times p} & \cdots & \cdots & 0_{p\times p} \end{bmatrix}, \quad \hat{B}_k \triangleq \begin{bmatrix} \hat{G}_{1,k} \\ \hat{G}_{2,k} \\ \vdots \\ \hat{G}_{\hat{n},k} \end{bmatrix}, \tag{10}$$

$$\hat{C}_k \triangleq \begin{bmatrix} I_p & 0_{p\times p} & \cdots & 0_{p\times p} \end{bmatrix}. \tag{11}$$

B. Model Predictive Control

Model predictive control (MPC) uses a model of the system to optimize the future performance over a time horizon. The optimization yields a sequence of controls, the first of which is implemented, and the procedure is repeated at subsequent steps. By performing constrained optimization, MPC enforces constraints on the control input and plant outputs [22–25]. At step k, PCAC uses the identified model $\hat{A}_k$, $\hat{B}_k$, and $\hat{C}_k$. As in [26], receding-horizon optimization is performed using quadratic programming (QP), which is a convex optimization technique. This optimization determines the control input u_{k+1} at the next time step, while also attempting to satisfy constraints on the state and control input

To describe QP-based MPC, let $\mathcal{R}_{k,\ell} \triangleq \begin{bmatrix} r_{k+1}^{\mathrm{T}} \cdots r_{k+\ell}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{\ell p_{\mathrm{t}}}$ be the vector of future commands over the horizon ℓ, let $Y_{1|k,l}$ be the corresponding l-step predicted output for a sequence of l future controls, $U_{1|k,l}$, and let $Y_{\mathrm{t},1|k,\ell} \triangleq C_{\mathrm{t},\ell} Y_{1|k,\ell}$ be the ℓ-step predicted output, where $C_{\mathrm{t},\ell} \triangleq I_\ell \otimes C_{\mathrm{t}} \in \mathbb{R}^{\ell p_{\mathrm{t}} \times \ell p}$, $\otimes$ is the Kronecker product, and $C_{\mathrm{t}} y_{i|k}$ computes the tracking outputs from $y_{i|k}$. Let $C_\ell \triangleq I_\ell \otimes (C C_{\mathrm{c}}) \in \mathbb{R}^{\ell n_{\mathrm{c}} \times \ell p}$, where $C_{\mathrm{c}} y_{i|k}$ creates the constrained outputs from $y_{i|k}$, let $\mathcal{D}_\ell \triangleq 1_\ell \otimes \mathcal{D} \in \mathbb{R}^{\ell n_{\mathrm{c}}}$, and define the sequence of differences of control inputs as

$$\Delta U_{1|k,\ell} \triangleq \begin{bmatrix} (u_{1|k} - u_k)^{\mathrm{T}} & \cdots & (u_{\ell|k} - u_{\ell-1|k})^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{\ell m}. \tag{12}$$

The QP-based MPC optimization problem is then given by

$$\min_{U_{1|k,\ell}} \left(Y_{\mathrm{t},1|k,\ell} - \mathcal{R}_{k,\ell}\right)^{\mathrm{T}} Q \left(Y_{\mathrm{t},1|k,\ell} - \mathcal{R}_{k,\ell}\right) + \Delta U_{1|k,\ell}^{\mathrm{T}} R \Delta U_{1|k,\ell}, \tag{13}$$

subject to

$$C_\ell Y_{1|k,\ell} + \mathcal{D}_\ell \le 0_{\ell n_c}, \tag{14}$$

$$U_{\min} \le U_{1|k,\ell} \le U_{\max}, \tag{15}$$

$$\Delta U_{\min} \le \Delta U_{1|k,\ell} \le \Delta U_{\max}, \tag{16}$$

where $Q \triangleq \begin{bmatrix} \bar{Q} & 0_{(\ell-1)p_t \times p_t} \\ 0_{p_t \times (\ell-1)p_t} & \bar{P} \end{bmatrix} \in \mathbb{R}^{\ell p_t \times \ell p_t}$ is the output weighting, $\bar{Q} \in \mathbb{R}^{(\ell-1)p_t \times (\ell-1)p_t}$ is the cost-to-go output weighting, $\bar{P} \in \mathbb{R}^{p_t \times p_t}$ is the terminal output weighting, $R \in \mathbb{R}^{\ell m \times \ell m}$ is the control-move-size weighting, $U_{\min} \triangleq 1_\ell \otimes u_{\min} \in \mathbb{R}^{\ell m}$, $U_{\max} \triangleq 1_\ell \otimes u_{\max} \in \mathbb{R}^{\ell m}$, $\Delta U_{\min} \triangleq 1_\ell \otimes \Delta u_{\min} \in \mathbb{R}^{\ell m}$, and $\Delta U_{\max} \triangleq 1_\ell \otimes \Delta u_{\max} \in \mathbb{R}^{\ell m}$.

III. Simulation Setup

All examples in this paper were computed using ANSYS Fluent. Fluent is interfaced with the MATLAB-based PCAC control code using a custom Fluent user-defined function (UDF). At each timestep k, the UDF obtains measurements y_k from the flow field and provides them to PCAC. PCAC uses y_k to compute the performance variable z_k, which it uses to compute the control inputs u_k. The UDF then sends u_k to Fluent, which applies the control input by modifying the associated actuator boundary conditions.

IV. Application of PCAC to Boundary Layer Control

In this section, the flow-control objective is to specify the streamwise velocity at a specified location within a viscous boundary layer. An idealized flow-velocity sensor is assumed to be placed at the location of interest to measure the streamwise velocity y_k, and PCAC specifies the flow velocity u_k at the duct inlet to achieve the streamwise velocity setpoint r_k. As in all of the examples in this paper, no model of the fluid dynamics is assumed to be available to PCAC. Instead, PCAC uses recursive least squares with variable-rate forgetting to perform online closed-loop system identification.

Example 1. *Streamwise-velocity setpoint command following in a 2D duct boundary layer.* We consider a 2D duct with width 0.5 m and length 1 m. A streamwise-velocity setpoint is commanded at the location of a velocity sensor, which is positioned halfway along the duct and within the boundary layer. The left boundary condition is the velocity inlet with uniform velocity profile specified by the control input, and the right boundary condition is a constant pressure outlet. A no-slip wall boundary condition is enforced at the bottom of the duct, and a symmetry boundary condition is enforced at the midheight line of the duct. The velocity sensor measures the streamwise flow velocity y_k at the sensor location. The sensor data are used by PCAC to compute the command-following error z_k between the current measurement and the setpoint. The control input specified by PCAC is the uniform inlet velocity. The fluid computation is incompressible, unsteady, viscous, and laminar with dynamic viscosity $\mu = 1.7894 \times 10^{-5}$ kg/(m-s).

The flow-field velocity in the duct is initialized to be 0 m/s at all locations. The flow state is advanced forward in time with a time step of 0.01 s using first-order implicit time stepping. PCAC samples the velocity sensor with a matching time step of 0.01 s. To investigate the response of PCAC, we vary the tunable weightings on the performance variable z_k in the MPC cost function while fixing the weighting on the change Δu_k in the control input. In particular, the intermediate and terminal weightings Q and P, respectively, on z_k are set to 100, 50, 10, and 0.1. The weighting R on Δu_k is fixed at 0.1. Figure 2 shows the sensor measurements and the control inputs for the chosen values of Q and P. Note that, as Q and P are increased, the command-following performance improves at the expense of larger transients when the controller is enabled. Figure 3 shows the sensor measurements, the control inputs, the estimated model coefficients, and the forgetting factor for the weightings $Q = P = 10$ and $R = 0.1$, which balance command-following performance with overshoot.

As an independent check of the simulation and asymptotic performance, we use the classical Blasius laminar boundary-layer analysis, which describes the boundary layer over a flat plate, to approximately determine the inlet velocities that produce the setpoint velocities at the sensor location. In particular, the Blasius differential equation is given by

$$f''' + \frac{1}{2} f f'' = 0, \tag{17}$$

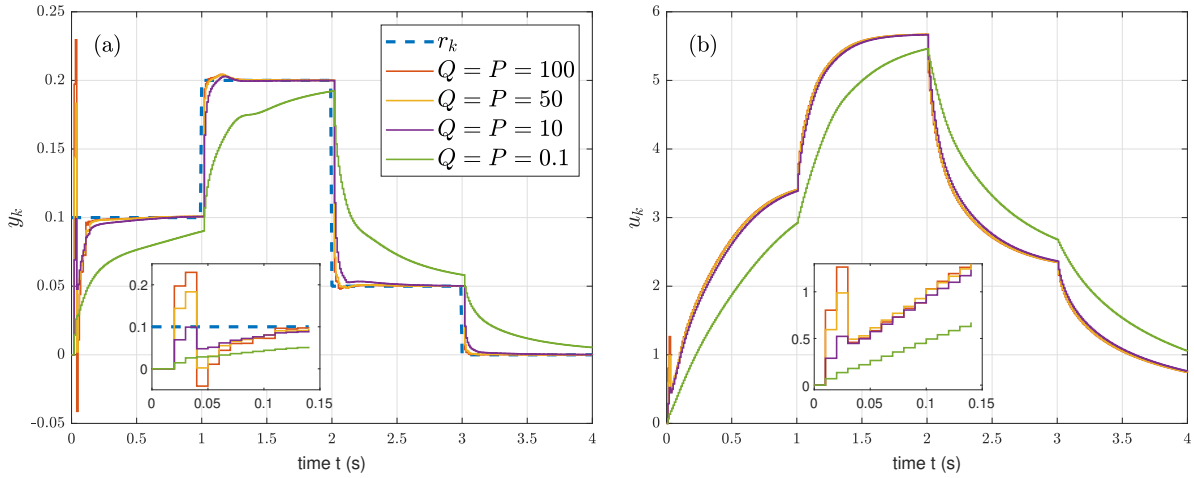

Fig. 2 Example 1: With PCAC starting at time $t = 0$, (a) shows the streamwise velocity y_k at the sensor location. Note that, as the intermediate and terminal weightings Q and P increase, the initial transient due to learning increases, whereas the subsequent response improves. (b) shows the specified duct-inlet velocity u_k which changes with each change in setpoint. The inset plots show the initial transient response.

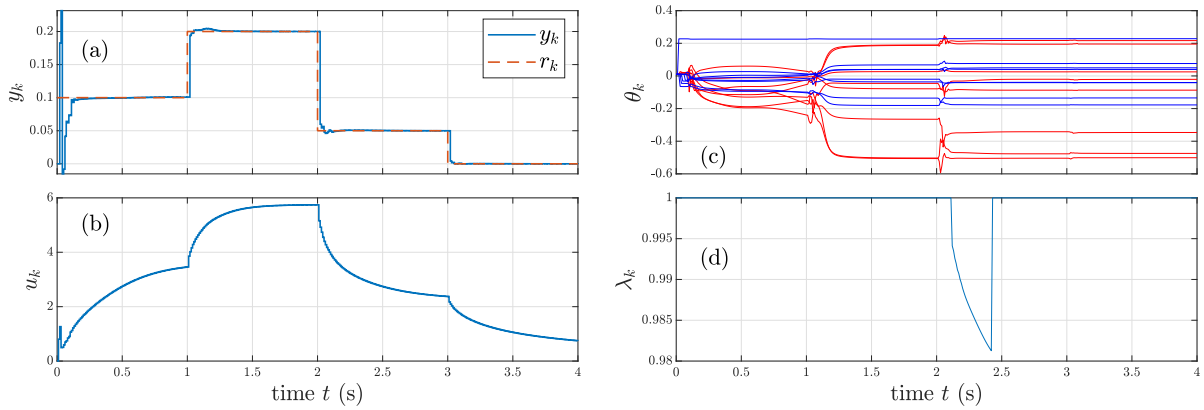

Fig. 3 Example 1: With PCAC starting at time $t = 0$ and using $Q = P = 10$ and $R = 0.1$, (a) shows that the streamwise velocity at the sensor location converges to the setpoint after about 0.25 s. (b) shows the duct-inlet velocities specified by PCAC. (c) shows the time histories of the numerator (blue) and denominator (red) coefficients of the identified model. (d) shows the variable-rate forgetting factor. Forgetting is active between 2 s and 2.5 s, where $\lambda_k < 1$.

where the nondimensional streamfunction f is a function of the similarity variable η, which describes the location of interest within the boundary layer and is defined by [27, p. 234]

$$\eta = y\sqrt{\frac{u_\infty}{\nu x}}, \tag{18}$$

where y is the distance from the plate, u_∞ is the free-stream velocity, $\nu = \mu/\rho$ is the kinematic viscosity, and x is the distance along the plate. For the boundary conditions

$$f(0) = 0, \quad f'(0) = 0, \quad f'(\infty) = 1, \tag{19}$$

solving (17) yields the normalized velocity profile in the boundary layer. The velocity u at the sensor location is defined using this profile and is given by

$$u = f'(\eta)u_\infty. \tag{20}$$

To determine the inlet velocity that results in the setpoint velocity at the sensor location, we substitute the setpoint r_k for the velocity u at sensor location, and we substitute the inlet velocity u_k for the free-stream velocity u_∞. Then, using an iterative method, we solve for the inlet velocity that corresponds to the setpoint. Figure 4 shows the relationship between the setpoint velocity at the sensor location and the required inlet velocity obtained from the Blasius solution.

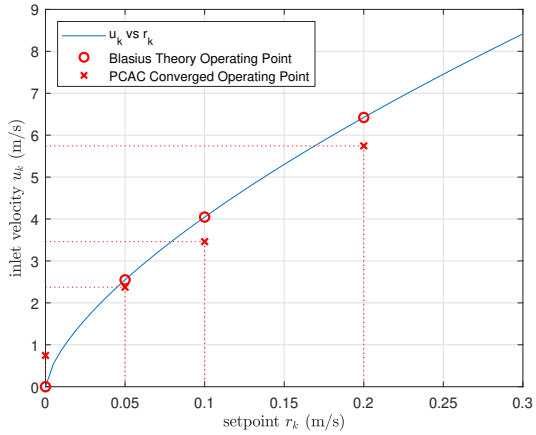

Fig. 4 Example 1: Required inlet velocity versus the setpoint at the sensor location computed by the Blasius flat plate laminar boundary layer theory. The red dots indicates the setpoints in Example 1 and the Blasius theory inlet velocity. The red crosses indicate the same setpoints and the inlet velocities requested by PCAC. These operating conditions are close, however a small difference is present, partially due to the application of a flat plate theory to a duct and due to low Reynolds number flow.

The above procedure yields the inlet velocities 2.55, 4.05, and 6.42 m/s, which are close to the converged inlet velocities 2.37, 3.46, and 5.74 m/s obtained from PCAC. Note that PCAC achieves these values with no prior model of the fluid dynamics. An exact match is not expected since the duct simulation does not adhere to flat-plate laminar boundary-layer theory. Additionally, at the sensor location, the Reynolds numbers Re_x based on x are 8.12×10^4, 1.18×10^5, and 1.97×10^5 using the converged inlet velocities. Note that the Blasius solution becomes more accurate as the Reynolds numbers increases.

It can be seen in the PCAC results in Figure 3 that the streamwise velocity measurement converges to the setpoint in a few time steps, whereas the PCAC-specified inlet velocity continues to change. Since the flow is approximately incompressible, changes in the duct-inlet velocity are immediately propagated to the entire duct, and thus the velocity at the sensor location changes immediately with changes in the duct-inlet velocity. While the change in the inlet velocity is immediately evident at the sensor location, the boundary layer still takes time to develop with respect to the new bulk velocity. This process is visible in Figure 5, where the boundary profile at the sensor location is plotted at different time instances.

In order to maintain the setpoint velocity at the sensor location within the boundary layer as the boundary layer develops, the bulk velocity needs to change. This is evident in Figure 6, where the velocity at the sensor location is plotted as the duct-inlet velocity undergoes step changes. At each setpoint change, a fast timescale change in the measured velocity due to the change in bulk velocity is visible in the first 0.25 s, while a slow timescale change in the measured velocity due to the boundary-layer development is visible over the remaining 1.75 s.

System performance with PCAC is compared with performance with PI control in Figure 7. Here, the PI controller is tuned for a fast convergence rate while avoiding larger overshoot than present with PCAC. Due to the initial large change in measured velocity, the magnitude of the PI gains are limited to prevent large overshoot. However, these small gains lead to slow convergence to the velocity setpoint as the boundary layer develops.

◇

Example 2. *Streamwise velocity setpoint in the boundary layer of an axial cylinder in a 2D duct.* Reconsider the geometry in Example 1 with the addition of an axial cylinder with a radius of 0.1 m. This geometry is shown in Figure 8. In this example, the x-velocity of the flow at a location within the boundary layer on the leading surface of the axial cylinder is commanded with a harmonic setpoint. QP performs its optimization with knowledge of the future setpoint.

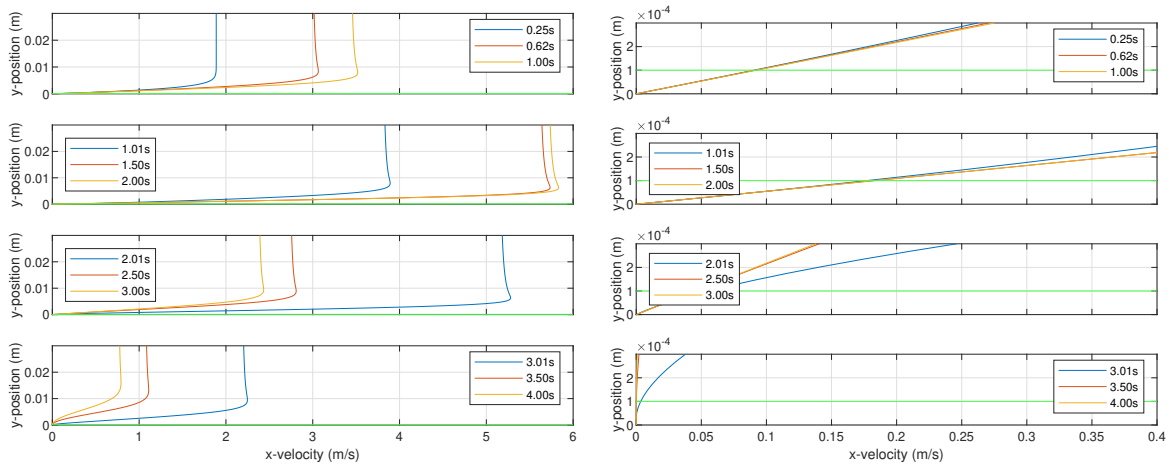

Fig. 5 Example 1: Boundary layer profiles at the sensor longitudinal position for different instants in time. The horizontal green line indicates the sensor height. The left plots show the full boundary layer profile while the right plots are zoomed in the vicinity of the sensor height. The rows (top to bottom) correspond to the velocity setpoints: 0.1, 0.2, 0.05, and 0.0 m/s. The right plots show that the velocity at the sensor location remains nearly constant for each setpoint, while the left plots show large variation in the boundary-layer profile as the boundary layer develops.

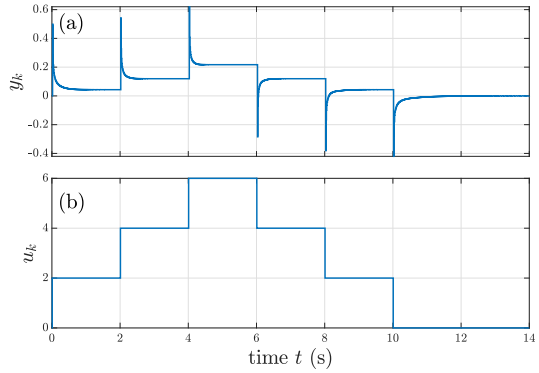

Fig. 6 Example 1: Sensor flow-velocity measurements in response to step changes in duct-inlet velocity.

This case is a more complex version of Example 1 where Blasius theory no longer applies since the setpoint is time varying instead of steady and because the geometry is no longer a flat plate.

The flow field is initialized to have 0 m/s velocity at all locations within the duct. The fluid computation is incompressible, unsteady, viscous, and turbulent using the k-ω turbulence model. The dynamic viscosity is $\mu = 1.7894 \times 10^{-5}$ kg/(m-s). At the inlet, the turbulent intensity is 5% and the turbulent viscosity ratio is 10. The flow state is advanced forward in time with a time step of 0.01 s using first-order implicit time stepping. PCAC samples the velocity sensor with a matching time step of 0.01 s. To investigate the response of PCAC, we vary the identified model order $\hat{n}$ in the RLS system identification. In particular, $\hat{n}$ is set to $2, 4, 8, 16$, and 32. All other PCAC parameters remain fixed. Figure 9 shows the sensor measurements and the control inputs for the chosen values of $\hat{n}$. Figure 10 shows the sensor measurements, the control inputs, the estimated model coefficients, and the forgetting factor for the model order $\hat{n} = 64$, which gives the best command following performance. Without any prior model of the fluid dynamics, PCAC is able to follow the harmonic velocity command.

◇

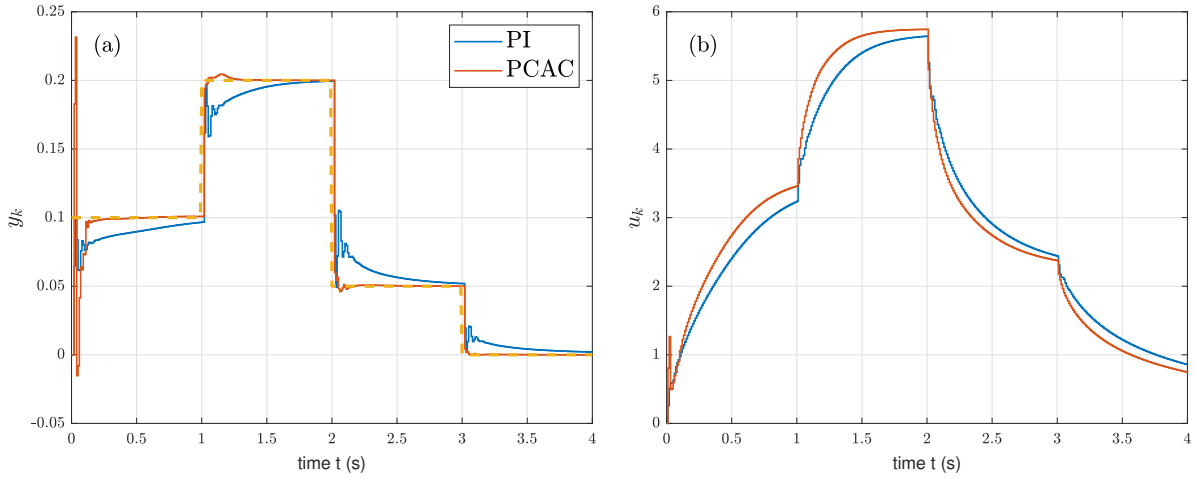

Fig. 7 Example 1: (a) shows the sensor measurements and setpoint using PI and PCAC. (b) shows the inlet velocities. PCAC produces a larger initial transient as it identifies the system dynamics, but subsequently yields faster convergence to the setpoint with less oscillation.

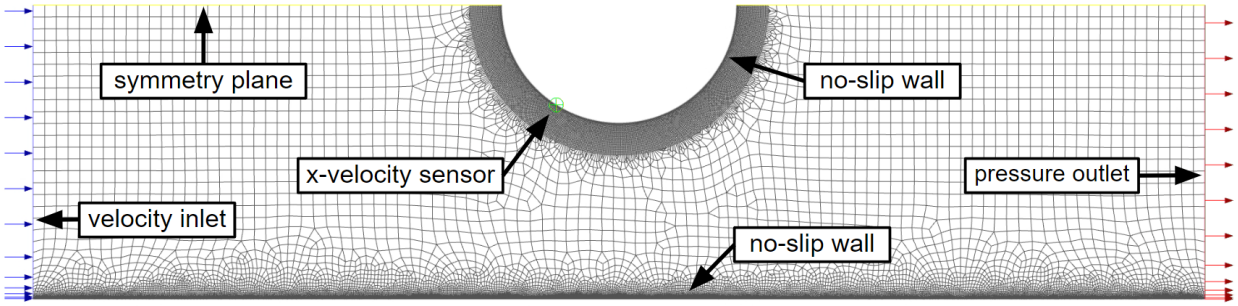

Fig. 8 Example 2: Axial cylinder within duct mesh and boundary conditions. A symmetry condition is applied at the duct midheight line. The x-velocity sensor location is indicated by the green cross.

V. Conclusions

This paper investigated the performance of active flow control using predictive cost adaptive control (PCAC). For all of the examples considered in this paper, PCAC was implemented as a "cold-start" technique, where no prior modeling information was assumed to be available. Future research will extend this study to 3D fluid dynamics in order to further ascertain the effectiveness of PCAC on realistic active flow control problems.

VI. Acknowledgments

This research was supported by ONR grants N00014-22-1-2457 and N00014-24-1-2285.

References

[1] Bewley, T. R., "Flow control: new challenges for a new renaissance," *Progress in Aerospace sciences*, Vol. 37, No. 1, 2001, pp. 21–58.

[2] Aamo, O. M., and Krstic, M., *Flow Control by Feedback: Stabilization and Mixing*, Springer, 2003.

[3] Gunzburger, M. D., *Perspectives in Flow Control and Optimization*, SIAM, 2002.

[4] Gad-el Hak, M., *Flow Control: Passive, Active, and Reactive Flow Management*, Cambridge University Press, 2000.

[5] Joslin, R. D., and Miller, D. N. (eds.), *Fundamentals and Applications of Modern Flow Control*, AIAA, 2009.

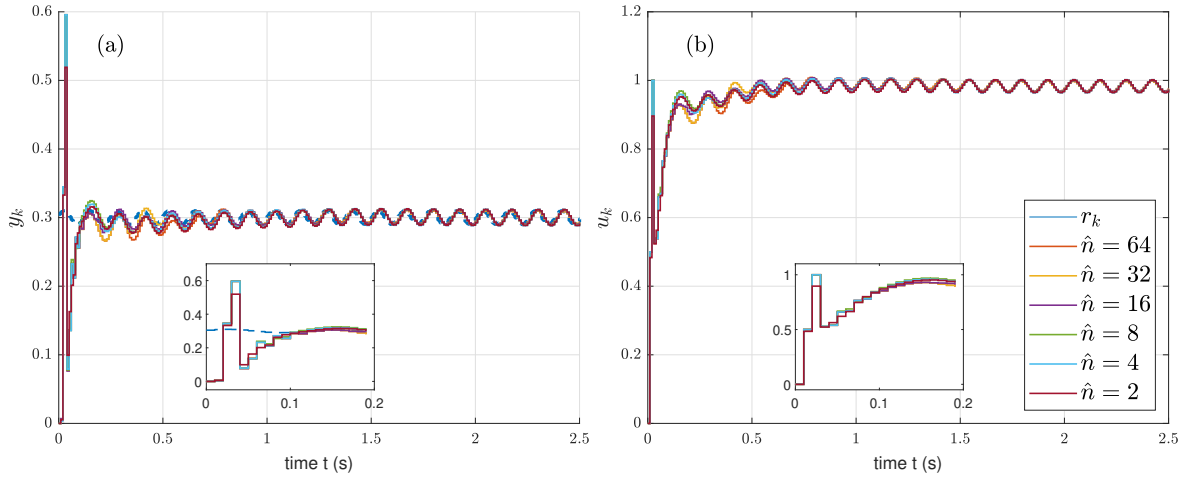

Fig. 9 Example 2: With PCAC starting at time $t = 0$, (a) shows the streamwise velocity y_k at the sensor location. Note that, as $\hat{n}$ is increased, the command-following performance and initial transient improves. (b) shows the specified duct-inlet velocity u_k. The inset plots show the initial transient response.

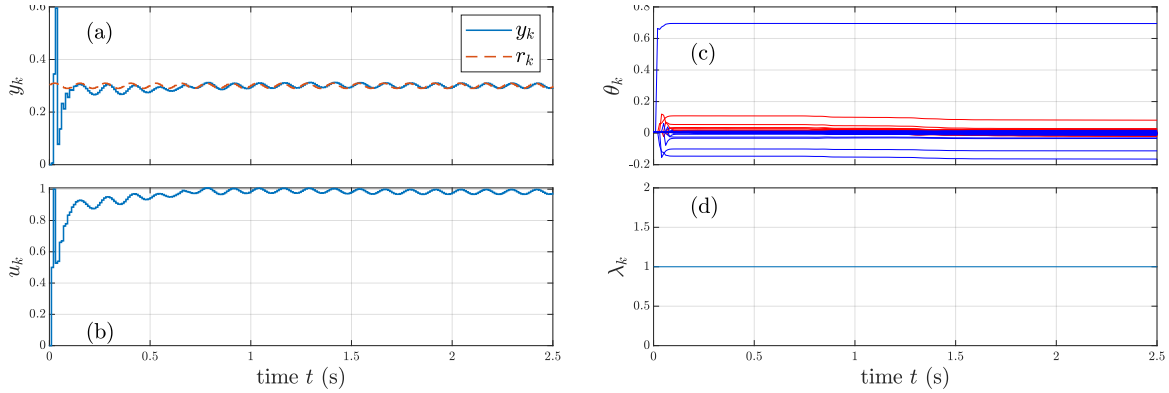

Fig. 10 Example 2: With PCAC starting at time $t = 0$ and using $\hat{n} = 64$, (a) shows that the measured velocity at the sensor location approaches the harmonic setpoint. (b) shows the duct-inlet velocities specified by PCAC. (c) shows the time histories of the numerator (blue) and denominator (red) coefficients of the identified model. (d) shows the variable-rate forgetting factor. Forgetting is inactive in this case.

[6] Barbu, V., *Stabilization of Navier–Stokes Flows*, Springer, 2011.

[7] Cattafesta III, L. N., and Sheplak, M., "Actuators for active flow control," *Ann. Rev. Fluid Mech.*, Vol. 43, 2011, pp. 247–272.

[8] Luhar, M., Sharma, A. S., and McKeon, B. J., "Opposition control within the resolvent analysis framework," *J. Fluid Mech.*, Vol. 749, 2014, p. 597–626.

[9] Brunton, S. L., and Noack, B. R., "Closed-loop turbulence control: Progress and challenges," *Appl. Mech. Rev.*, Vol. 67, No. 5, 2015.

[10] Duriez, T., Brunton, S. L., and Noack, B. R., *Machine Learning Control—Taming Nonlinear Dynamics and Turbulence*, Springer, 2017.

[11] Wang, J., and Feng, L., *Flow Control Techniques and Applications*, Cambridge University Press, 2019.

[12] Bewley, T. R., and Liu, S., "Optimal and robust control and estimation of linear paths to transition," *J. Fluid Mech.*, Vol. 365, 2001, pp. 305–349.

[13] Mushtaq, T., Seiler, P., and Hemati, M. S., "Feedback control of transitional flows: A framework for controller verification using quadratic constraints," *AIAA Aviation Forum*, virtual, 2021. AIAA-2021-2825.

[14] Joshi, S. S., Speyer, J. L., and Kim, J., "A Systems Theory Approach to the Feedback Stabilization of Infinitesimal and Finite-Amplitude Disturbances in Plane Poiseuille Flow," *J. Fluid Mech.*, Vol. 332, 1997, pp. 157–184.

[15] Joshi, S. S., Speyer, J. L., and Kim, J., "Finite Dimensional Optimal Control of Poiseuille Flow," *J. Guid. Contr. Dyn.*, Vol. 22, 1999, pp. 340–348.

[16] Gautier, N., Aider, J.-L., Duriez, T., Noack, B., Segond, M., and Abel, M., "Closed-loop separation control using machine learning," *J. Fluid Mech.*, Vol. 770, No. 5, 2015, pp. 442–457.

[17] Vander Schaaf, J., Lu, Q., Fidkowski, K. J., and Bernstein, D. S., "Data-Driven Retrospective Cost Adaptive Flow Control," *2024 AIAA SciTech Forum*, Orlando, FL, 2024, pp. 1–22. AIAA 2024-1935.

[18] Bruce, A. L., Goel, A., and Bernstein, D. S., "Convergence and consistency of recursive least squares with variable-rate forgetting," *Automatica*, Vol. 119, 2020, p. 109052.

[19] Mohseni, N., and Bernstein, D. S., "Recursive least squares with variable-rate forgetting based on the F-test," *Proc. Amer. Contr. Conf.*, 2022, pp. 3937–3942.

[20] Mohseni, N., and Bernstein, D. S., "Recursive Least Squares with Variable-Rate Forgetting Based on the F-Test," *Proc. Amer. Contr. Conf.*, 2022, pp. 3937–3942.

[21] McKeon, J. J., "F approximations to the distribution of Hotelling's T_0^2," *Biometrika*, Vol. 61, No. 2, 1974, pp. 381–383.

[22] Kwon, W., and Han, S., *Receding Horizon Control: Model Predictive Control for State Models*, Springer, 2006.

[23] Rawlings, J. B., Mayne, D. Q., and Diehl, M. M., *Model Predictive Control: Theory, Computation, and Design*, Nob Hill, 2020.

[24] Tao, J., Ma, L., and Zhu, Y., "Improved control using extended non-minimal state space MPC and modified LQR for a kind of nonlinear systems," *ISA Trans.*, Vol. 65, 2016, pp. 319–326.

[25] Eren, U., Prach, A., Koçer, B. B., Raković, S. V., Kayacan, E., and Açıkmeşe, B., "Model predictive control in aerospace systems: Current state and opportunities," *J. Guid. Contr. Dyn.*, Vol. 40, No. 7, 2017, pp. 1541–1566. https://doi.org/10.2514/1.G002507.

[26] Nguyen, T. W., Islam, S. A. U., Bernstein, D. S., and Kolmanovsky, I. V., "Predictive Cost Adaptive Control: A Numerical Investigation of Persistency, Consistency, and Exigency," *IEEE Contr. Sys. Mag.*, Vol. 41, 2021, pp. 64–96.

[27] White, F. M., *Viscous Fluid Flow (2nd ed.)*, McGraw Hill, 1991.