2024 American Control Conference (ACC)
July 8-12, 2024. Toronto, Canada

Data-Driven Model Predictive Control of Airfoil Flow Separation

Jacob C. Vander Schaaf, Qizhi Lu, Krzysztof J. Fidkowski, and Dennis S. Bernstein

***Abstract*— This paper numerically investigates the performance of active flow control based on adaptive model predictive control without prior analytical modeling of any kind. Recursive least squares with variable-rate forgetting is used for online closed-loop system identification. The identified model is then used for receding-horizon optimization based on quadratic programming. This technique is applied to an airfoil model simulated using computational fluid dynamics. Two cases are considered. In the first case, the tangential component of the flow velocity is commanded at the location of a single flow-velocity sensor. In the second case, an array of flow-velocity sensors is used to estimate the location of the flow-separation point, which is used to command the flow-separation point.**

I. Introduction

Active feedback flow control uses sensors, actuators, and feedback control algorithms to modify the fluid dynamics to enhance aerodynamic performance [1]–[3]. Although the physics of fluid dynamics is well understood, the models used to design control laws may possess errors that degrade closed-loop performance [4], [5]. These errors arise from model approximation, parameter errors, and discretization errors. Active feedback flow-control techniques include optimal control [6], robust control [4], and machine-learning methods [7], [8]. The present paper focuses on adaptive control techniques, where the controller learns and adapts during operation in response to actual, changing conditions with minimal prior modeling information. The present paper focuses on predictive cost adaptive control (PCAC).

PCAC is an indirect adaptive control technique based on model predictive control (MPC) [9], [10]. For online, closed-loop system identification, PCAC uses recursive least squares (RLS) with variable-rate forgetting [11]–[15]. At each time step, RLS-based system identification updates the coefficients of a SISO or MIMO input-output model, where the model order is a hyperparameter specified by the user. For model predictive control, the receding-horizon optimization can be performed by either the backward-propagating Riccati equation [16], [17] or quadratic programming.

In this paper, we apply PCAC to flow-control problems involving flow separation. This scenario represents a fundamental challenge in aerodynamics, where, under high angle of attack, the flow becomes detached from the airfoil, thus leading to potentially catastrophic loss of lift. The present paper considers jet-velocity actuation near the leading edge of the airfoil in conjunction with velocity sensors along the upper surface. Since PCAC includes online system identification for learning, and since the initial model is zero, the controller adapts from zero gains; hence, PCAC operates under "cold-start" conditions. The user only needs to specify hyperparameters for learning and adaptation.

Jacob C. Vander Schaaf, Qizhi Lu, Krzysztof J. Fidkowski, and Dennis S. Bernstein are with the Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, USA. {jacobcvs, luqseven, kfid, dsbaero}@umich.edu

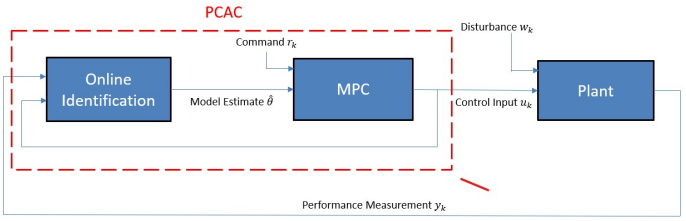

Fig. 1: PCAC block diagram.

The approach of the present paper differs from flow-control techniques that depend on prior modeling information and offline training. For example, by exciting a flow-simulation model, the approach of [18] is based on a neural network trained on 100,000–800,000 data points, which, for each Reynolds number, corresponds to about 3 hours of simulated flow time. In contrast, the application of PCAC to the flow-separation examples in the present paper uses only online data from the closed-loop system, without probing signals, and with re-identification occurring automatically as the operating conditions, such as the Reynolds number, angle of attack, and setpoint change.

We begin by commanding the x-velocity at the location of a single sensor near the upper surface of an airfoil, where x denotes the horizontal component. Next, we extend this case by commanding a sequence of x-velocities over time and by introducing disturbances in the freestream velocity and airfoil angle of attack. In these studies, we assess whether the flow is attached or separated at the sensor location, and we qualitatively move the flow-separation point forward or aft by commanding various x-velocities at the sensor location. Next, we use an array of x-velocity sensors to estimate the flow-separation point. Using this estimate, we command the location of the flow-separation point directly. We extend this case by commanding a sequence of flow-separation points over time and by introducing an angle-of-attack disturbance.

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.

979-8-3503-8265-5/$31.00 ©2024 AACC

A. Online Identification

To perform online 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 [12]

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

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

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

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

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

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

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 [15]

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

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

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

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 [19], [20]. The constant c, based on the windows lengths is described in [19]. 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 its performance over a future finite interval of time. 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 facilitates the enforcement of constraints on the state and control input. At step k, PCAC uses the identified model $\hat{A}_k$, $\hat{B}_k$, and $\hat{C}_k$. As in [9], the 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 $\mathcal{C}_\ell \triangleq I_\ell \otimes (\mathcal{C} C_{\mathrm{c}}) \in \mathbb{R}^{\ell n_{\mathrm{c}}\times \ell p}$, 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

$$\mathcal{C}_\ell Y_{1|k,\ell} + \mathcal{D}_\ell \leq 0_{\ell n_{\mathrm{c}}}, \tag{14}$$

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

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

where $Q \triangleq \begin{bmatrix}\bar{Q} & 0_{p_{\mathrm{t}}\times p_{\mathrm{t}}} \\ 0_{p_{\mathrm{t}}\times p_{\mathrm{t}}} & \bar{P}\end{bmatrix} \in \mathbb{R}^{\ell p_{\mathrm{t}}\times \ell p_{\mathrm{t}}}$ is the output weighting, $\bar{Q} \in \mathbb{R}^{(\ell-1)p_{\mathrm{t}}\times(\ell-1)p_{\mathrm{t}}}$ is the cost-to-go output weighting, $\bar{P} \in \mathbb{R}^{p_{\mathrm{t}}\times p_{\mathrm{t}}}$ 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 involve the NACA 4412 airfoil of chord length 1 m. The fluid simulations are 2D incompressible unsteady RANS using the SA turbulence model computed in Ansys Fluent. The discretization is second-order finite-volume in space, with the mesh shown in Figure 2, and first-order implicit in time, with a time step of 0.01 s. In the following examples, the nominal freestream velocity is 15 m/s and remains constant unless otherwise stated. This nominal freestream velocity corresponds to $Re = 1.03e6$.

Fluent is interfaced with the MATLAB-based PCAC control code using a custom Fluent user-defined function (UDF). With matching time step 0.01 s, the interface obtains a measurement y_k from the flow field, and provides it to PCAC. PCAC uses y_k to compute the performance variable z_k, which it uses to compute the requested control-jet velocity u_k. The interface then sends u_k to Fluent, which applies the requested control-jet velocity as the boundary condition at the location of the control jet.

A positive control-jet velocity indicates blowing, whereas a negative control-jet velocity indicates suction. The control-jet location and direction are shown in Figure 3. In the following examples, two different sensor architectures are used to obtain the measurement y_k. In Section IV, a single velocity sensor, as shown in Figure 3, measures the x-component of the velocity. In Section V, an array of velocity sensors is used to estimate the flow-separation point.

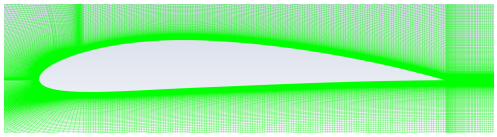

Fig. 2: Partial view of the mesh around the NACA 4412 airfoil. The full mesh of 196, 300 nodes extends to approximately 8 chord lengths in all directions.

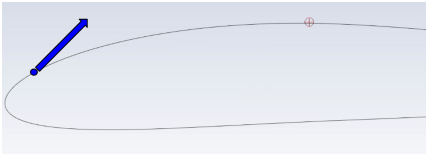

Fig. 3: Leading half of the NACA 4412 airfoil with the sensor used in Section IV (red crosshair) and the control-jet location and direction (blue dot and arrow). The blue arrow indicates the direction of positive control-jet flow. In Section V, an array of sensors (not shown) is used.

IV. Airfoil x-velocity Control

This section uses PCAC to control the x-velocity of the flow at the sensor location above the NACA 4412 airfoil. A single, idealized velocity sensor, shown in Figure 3, measures the flow velocity in the x-direction of the airfoil-fixed frame.

Flow separation along the airfoil occurs where the flow velocity within the boundary layer reverses direction. By measuring the x-velocity of the flow, which is approximately tangential to the airfoil surface, we can determine whether the flow is attached or separated at the sensor location. Furthermore, by using feedback control to command a positive x-velocity at the sensor location, we ensure that the flow at the sensor location is attached.

Example 1. *Near-airfoil, x-velocity setpoint.* The objective of this example is to command the x-velocity at the sensor location. The angle of attack of the airfoil is 20 deg, and the freestream velocity is 15 m/s, both of which are held constant. We apply adaptive control to command an x-velocity setpoint at the sensor location of $r_k = 8$ m/s. The adaptive controller is enabled at 3 s to allow the flow around the airfoil to develop. In the absence of control, the x-velocity at the sensor location is approximately -2 m/s, indicating separated flow. Figure 4 shows the time history of the x-velocity measurement y_k, the x-velocity setpoint r_k, the requested control-jet velocity u_k, and the estimated coefficient vector θ_{m}. Under closed-loop control, Figure 4 shows that the x-velocity converges to the commanded setpoint approximately 7 s after the controller is enabled. ◇

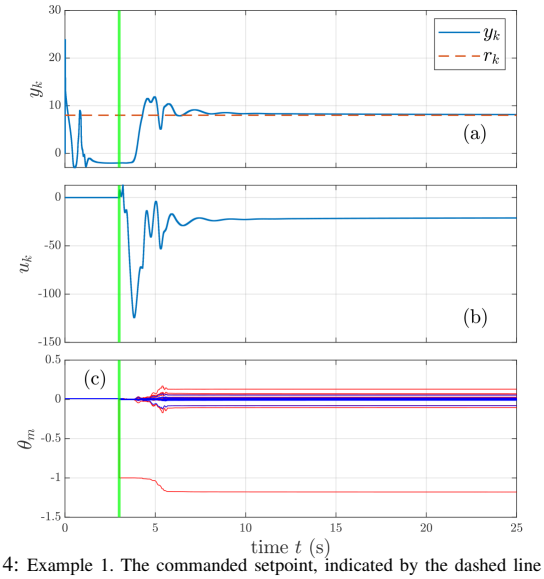

Fig. 4: Example 1. The commanded setpoint, indicated by the dashed line in (a), is 8 m/s. The adaptive controller is enabled at 3 s, as indicated by the vertical green line. (a) shows that, approximately 10 s after the controller is enabled, the x-velocity measurement converges to the commanded setpoint. (b) shows the corresponding requested control-jet velocity, which converges to approximately -21 m/s, indicating suction. (c) shows the time history of the estimated coefficient vector θ_{m}.

Example 2. *Sequence of near-airfoil, x-velocity setpoints.* This example extends Example 1 by specifying a sequence of x-velocity setpoints. The angle of attack of the airfoil is 20 deg, and the freestream velocity is 15 m/s, both of which are held constant. We apply adaptive control to the airfoil with the sequence of setpoints $r_k = 8$, 10, and 12 m/s before stepping back to 8 m/s. The adaptive controller is enabled at 3 s to allow the flow around the airfoil to develop. Figure 5 shows the time history of the x-velocity measurement y_k, the x-velocity setpoint r_k, the requested control-jet velocity u_k, and the estimated coefficient vector θ_{m}. Under closed-loop control, Figure 5 shows that, in approximately 10 s, the x-velocity converges with a small setpoint error to its initial setpoint and, after each change in setpoint, takes approximately 3 s to converge to the new setpoint. A small setpoint error is present when the setpoint is 12 m/s.

Figure 6 shows x-velocity contours at $t = 2.8, 19.8, 39.8,$ and 59.8 s. These contours show that the flow is separated at the sensor location in the absence of control, and, in the controlled flow, that the flow-separation point is moved further toward the trailing edge of the airfoil as the velocity setpoint at the sensor location is increased.

By assessing whether the flow is separated at the sensor location and commanding different x-velocities, we have rudimentary control of the flow-separation point. ◇

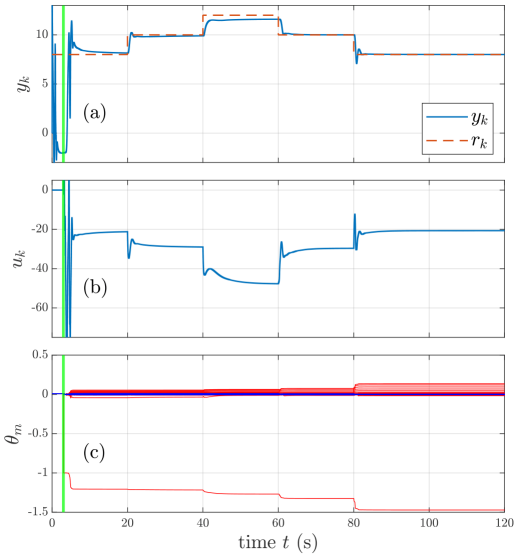

Fig. 5: Example 2. The commanded setpoint, indicated by the dashed line in (a), is initially 8 m/s varies stepwise as shown in (a). The adaptive controller is enabled at 3 s, as indicated by the vertical green line. (a) shows that, approximately 15 s after the controller is enabled, the x-velocity measurement converges to the commanded setpoint. After each change in setpoint, it takes approximately 3 s to converge to the new commanded setpoint. A command-following bias occurs when the setpoint is 12 m/s. (b) shows the corresponding requested control-jet velocity. This velocity is higher in magnitude when a larger setpoint velocity is requested. (c) shows the time history of the estimated coefficient vector θ_{m}.

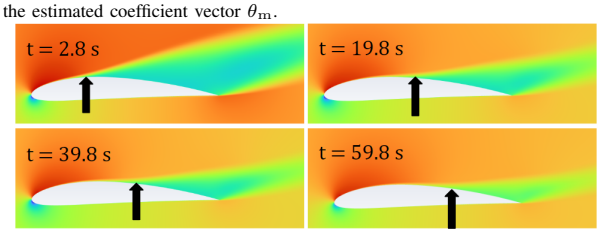

Fig. 6: Example 2. x-velocity contours at $t = 2.8, 19.8, 39.8,$ and 59.8 s. The approximate flow-separation point is indicated by the black arrow. At $t = 2.8$ s, the flow is uncontrolled. At $t = 19.8, 39.8,$ and 59.8 s, the x-velocity setpoints are $8, 10,$ and 12 m/s, respectively. When the flow is uncontrolled, it is separated at the sensor location. In the controlled flow, the flow is attached at the sensor location, and the flow-separation point moves aft as the x-velocity setpoint is increased.

Example 3. *Near-airfoil, x-velocity setpoint with variable freestream velocity.* In this example, we repeat Example 1, however, we now introduce a disturbance by varying the freestream velocity, imposed on the airfoil farfield boundary. Figure 7 shows that the freestream velocity begins at the nominal value of 15 m/s and is ramped up to 20 and then 25 m/s before being ramped back down to 15 m/s. At these three velocities, the Reynolds number is 1.03, 1.37, 1.71, and 1.03 million, respectively. The angle of attack of the airfoil is 20 deg. We apply adaptive control to the airfoil with an x-velocity setpoint of $r_k = 8$ m/s. The adaptive controller is enabled at 3 s to allow the flow around the airfoil to develop. Figure 7 shows the time history of the x-velocity measurement y_k, the x-velocity setpoint r_k, the requested control-jet velocity u_k, and the estimated coefficient vector θ_{m}. Under closed-loop control, Figure 7 shows that the measured x-velocity is brought to the commanded velocity setpoint and remains near the velocity setpoint despite the changing freestream velocity. ◇

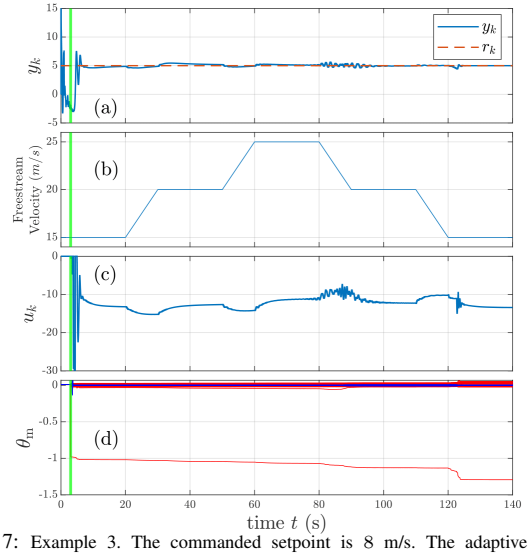

Fig. 7: Example 3. The commanded setpoint is 8 m/s. The adaptive controller is enabled at 3 s, indicated by the vertical green line. (a) shows that the x-velocity measurement converges to the commanded setpoint. As the freestream velocity changes, an oscillation is present as the controller re-adapts. This oscillation damps out after the re-adaptation and is largest when the freestream velocity is decreasing. (b) shows the freestream velocity, which ramps up and down between three different values. This freestream profile replicates an aircraft speeding up and slowing down in flight. (c) shows the corresponding requested control-jet velocity. With each change in freestream velocity, PCAC is able to converge to the correct control-jet velocity to maintain the desired flow velocity at the sensor location. (d) shows the time history of the estimated coefficient vector θ_{m}.

Example 4. *Near-airfoil, x-velocity setpoint with variable angle of attack.* In this example, we repeat Example 1, however we introduce a disturbance by varying the airfoil angle of attack as shown in Figure 8. The freestream velocity is 15 m/s. We apply adaptive control to the airfoil with an x-velocity setpoint of $r_k = 8$ m/s. The adaptive controller is enabled at 3 s to allow the flow around the airfoil to develop. Figure 7 shows the time history of the x-velocity measurement y_k, the x-velocity setpoint r_k, the requested control-jet velocity u_k, and the estimated coefficient vector θ_{m}. Under closed-loop control, Figure 7 shows that the measured x-velocity is brought to the setpoint velocity and remains at the commanded velocity despite the changing angle of attack. ◇

V. Flow-Separation-Point Control

Example 2 shows that, by commanding the x-velocity at the sensor location, the flow-separation point can be moved toward the trailing edge of the airfoil, thereby decreasing the tendency of the airfoil to stall. In this section, the goal is to specify the location of the flow-separation point rather than qualitatively moving it toward the trailing edge. To do this, we extend the examples of Section IV by introducing an array of 39 x-velocity sensors along the upper surface of the airfoil at evenly spaced locations along the chord in the range $x \in [0.2157, 0.9804]$ m, where $x = 0$ m denotes the leading edge and $x = 1$ m denotes the trailing edge. By measuring the x-velocity at multiple locations, the flow-separation point can be estimated as the position of the forward-most sensor reporting negative x-velocity. In each example, the location of the commanded flow-separation point may or may not coincide with one of the sensors in the array. At the cost of a larger number of sensors, the improved resolution of the sensor array allows the location of the flow-separation point

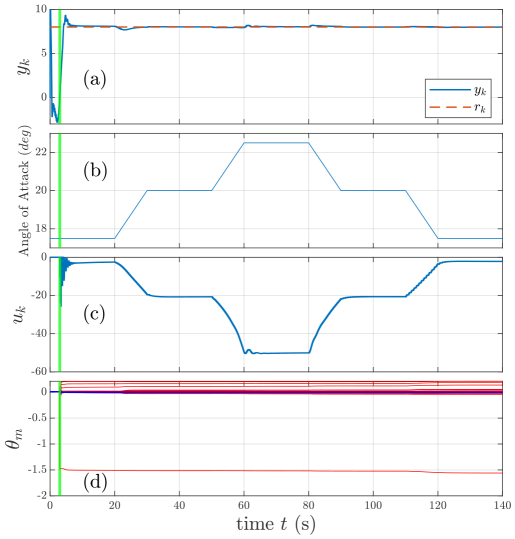

Fig. 8: Example 4. The commanded setpoint is 8 m/s. The adaptive controller is enabled at 3 s, as indicated by the vertical green line. (a) shows that the x-velocity measurement converges to and remains near the commanded setpoint despite the changing airfoil angle of attack. With each change in angle of attack, a small transient is present as the controller re-adapts. (b) shows the angle of attack, which ramps up and down between three different values. This angle of attack profile replicates an aircraft pitching up and back down in flight. (c) shows the corresponding requested control-jet velocity. With each change in angle of attack, PCAC is able to converge to the correct control-jet velocity to maintain the desired flow velocity at the sensor location. As the angle of attack is increased, the adaptive controller must request larger magnitude control-jet velocities to maintain the desired flow velocity. (d) shows the time history of the estimated coefficient vector θ_{m}.

to be estimated more accurately.

Example 5. *Single separation-point setpoint.* In this example, we specify a single separation-point setpoint given as a location on the airfoil. The angle of attack of the airfoil is 20 deg, and the freestream velocity is 15 m/s, both of which are constant. In the absence of control, the separation-point estimate is $x = 0.333$ m. We apply adaptive control to the airfoil with the setpoint $x = 0.7$ m. The adaptive controller is enabled at 3 s to allow the flow around the airfoil to develop. Figure 9 shows the time history of the separation-point estimate y_k, the separation-point setpoint r_k, the requested control-jet velocity u_k, and the estimated coefficient vector θ_{m}. Under closed-loop control, Figure 9 shows that, in approximately 7 s, the estimated separation point moves from $x = 0.333$ m to approximately $x = 0.7$ m. Since no sensor is present at $x = 0.7$ m, the separation-point estimate cannot be $x = 0.7$ m. ◇

Example 6. *Sequence of separation-point setpoints.* In this example, we extend Example 5 by specifying a sequence of separation-point setpoints. The angle of attack of the airfoil is 20 deg, and the freestream velocity is 15 m/s, both of which are constant. We apply adaptive control to the airfoil with the sequence of setpoints $x = 0.5, 0.6, 0.7,$ and $x = 0.5$ m. The adaptive controller is enabled at 3 s to allow the flow around the airfoil to develop. Figure 10 shows the time history of the separation-point estimate y_k, the separation-point setpoint r_k, the requested control-jet velocity u_k, and the estimated coefficient vector θ_{m}. Under closed-loop control, Figure 10 shows that, in approximately 6 s, the separation-point estimate is moved from $x = 0.333$ m to $x = 0.5$ m. At this commanded separation-point setpoint, oscillation is present. After each change in command, it takes

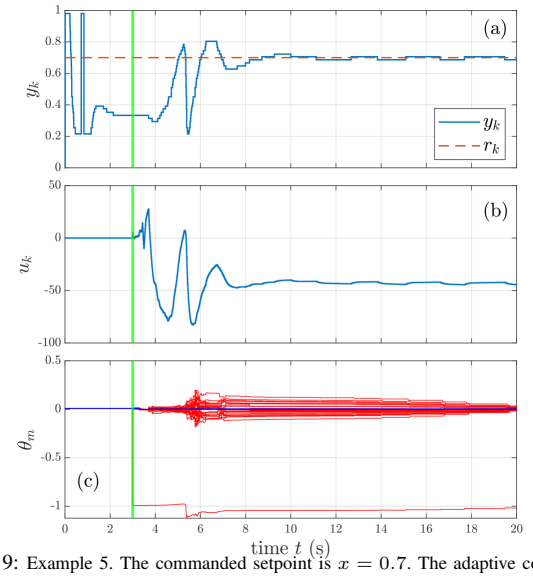

Fig. 9: Example 5. The commanded setpoint is $x = 0.7$. The adaptive controller is enabled at 3 s, as indicated by the vertical green line. (a) shows that, approximately 7 s after the controller is enabled, the separation-point estimate oscillates around the setpoint $x = 0.7$. This oscillation is due to the fact that the closest sensors to the setpoint are located at $x = 0.68$ and $x = 0.712$. (b) shows the corresponding requested control-jet velocity, which oscillates around -43 m/s, indicating suction. (c) shows the time history of the estimated coefficient vector θ_{m}.

approximately 4 s for the separation-point estimate to move to the new separation-point setpoint. When the setpoint is $x = 0.6$ m and $x = 0.7$ m, the previous oscillation is gone, but a small steady state error is apparent. When the setpoint is returned to $x = 0.5$ m, the oscillation returns. ◇

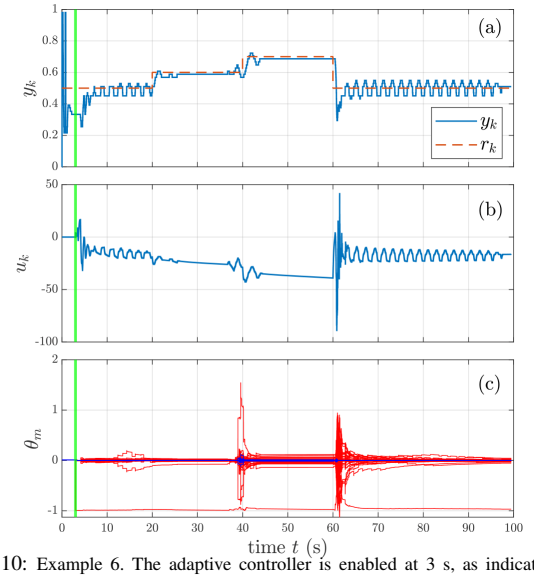

Fig. 10: Example 6. The adaptive controller is enabled at 3 s, as indicated by the vertical green line. The commanded setpoint, indicated by the orange dashed line, is initially $x = 0.5$ m, and changes at 20 s intervals. (a) shows that, approximately 6 s after the controllers is enabled, the separation-point estimate oscillates around the setpoint $x = 0.5$ m. After each change in setpoint, it takes approximately 4 s for the separation-point estimate to move to the new setpoint. (b) shows the corresponding requested control-jet velocities, which are larger in magnitude when the setpoint is closer to the trailing edge of the airfoil. (c) shows the time history of the estimated coefficient vector θ_{m}.

Example 7. *Separation-point setpoint with time-varying angle of attack.* In this example, we extend Example 5 by specifying a single separation-point setpoint and varying the airfoil angle of attack as shown in Figure 11. The freestream velocity is 15 m/s. We apply adaptive control to the airfoil with the setpoint $x = 0.7$ m; the adaptive controller is

enabled at 3 s. Figure 11 shows the time history of the separation-point estimate y_k, the separation-point setpoint r_k, the requested control-jet velocity u_k, and the estimated coefficient vector θ_{m}. Under closed-loop control, Figure 11 shows that, in approximately 11 s, the separation-point estimate is moved from $x = 0.333$ m to approximately $x = 0.7$ m. As the airfoil angle of attack is changed, transients appear as the controller re-adapts to the new angle of attack. Figure 12 shows the time history of the RLS forgetting factor λ_{id}. When $\lambda_{\mathrm{id}} = 1$, no forgetting is taking place. As operating conditions change, the forgetting factor decreases, and RLS automatically discounts past data as it updates the estimated coefficient vector. ◇

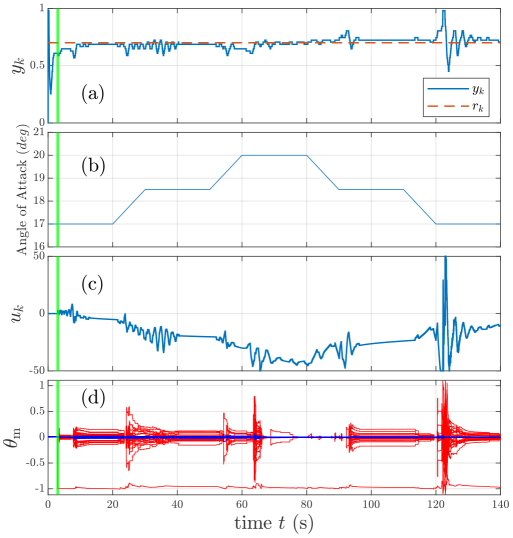

Fig. 11: Example 7. The adaptive controller is enabled at 3 s, as indicated by the vertical green line. The setpoint is $x = 0.7$ m. (a) shows that the separation-point estimate oscillates around the setpoint over the duration of the run, despite the changing airfoil angle of attack. (b) shows the airfoil angle of attack, which indicates an airfoil pitching up and back down. (c) shows the corresponding requested control-jet velocities, which are larger in magnitude when the airfoil is at a higher angle of attack. (d) shows the time history of the estimated coefficient vector θ_{m}.

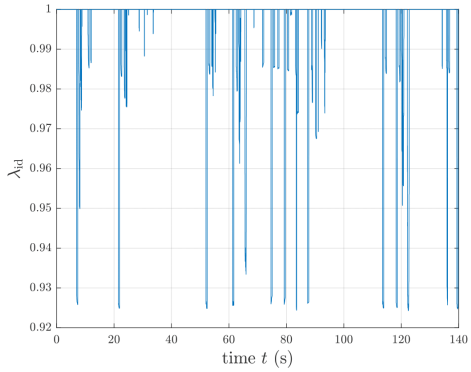

Fig. 12: Example 7. RLS forgetting factor. Forgetting occurs when the forgetting factor drops below 1. This occurs primarily when the airfoil angle of attack changes.

VI. Conclusions

This paper used predictive cost adaptive control (PCAC) to control the flow-separation point on an airfoil. The simulation was implemented with 2D computational fluid dynamics using Ansys Fluent. PCAC used no a priori modeling of the fluid dynamics and no open-loop data collection; only hyperparameters for the closed-loop system identification and quadratic programming were specified. This technique is motivated by the need for controlling flow under rapidly changing conditions and without the benefit or use of offline data collection and model pre-training. Using a single control jet and a single flow-velocity sensor, PCAC attached the flow at the sensor location and moved the flow-separation point aft using larger x-velocity setpoints. For the case of an array of flow-velocity sensors, PCAC moved the flow-separation point to the location of the specified flow sensor. These numerical results suggest the feasibility of flow control without the need for prior modeling. Future work will focus on the system behavior during initialization, particularly the ability of PCAC to maintain stability as the system is being identified. Additionally, the effect of measurement uncertainty will be examined to ensure the feasibility of the method with non-idealized sensors.

References

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

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

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

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

[5] K. Taira et al, "Modal Analysis of Fluid Flows: An Overview," *AIAA J.*, vol. 55, pp. 4013–4041, 2017.

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

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

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

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

[10] T. W. Nguyen, I. V. Kolmanovsky, and D. S. Bernstein, "Sampled-data output-feedback model predictive control of nonlinear plants using online linear system identification," in *Proc. Amer. Contr. Conf.*, 2021, pp. 4682–4687.

[11] S. A. U. Islam and D. S. Bernstein, "Recursive least squares for real-time implementation," *IEEE Contr. Syst. Mag.*, vol. 39, no. 3, pp. 82–85, 2019.

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

[13] A. Goel, A. L. Bruce, and D. S. Bernstein, "Recursive least squares with variable-direction forgetting: Compensating for the loss of persistency," *IEEE Contr. Sys. Mag.*, vol. 40, no. 4, pp. 80–102, 2020.

[14] A. L. Bruce, A. Goel, and D. S. Bernstein, "Necessary and sufficient regressor conditions for the global asymptotic stability of recursive least squares," *Sys. Contr. Lett.*, vol. 157, pp. 1–7, 2021, article 105005.

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

[16] W. H. Kwon and A. E. Pearson, "On feedback stabilization of time-varying discrete linear systems," *IEEE Trans. Autom. Contr.*, vol. AC-23, no. 3, pp. 479–481, 1978.

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

[18] K. Bieker, S. Peitz, S. L. Brunton, J. N. Kutz, and M. Dellnitz, "Deep model predictive flow control with limited sensor data and online learning," *Theor. Comp. Fluid Dyn.*, vol. 34, no. 4, p. 577–591, 2020.

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

[20] J. J. McKeon, "F approximations to the distribution of Hotelling's T_0^2," *Biometrika*, vol. 61, no. 2, pp. 381–383, 08 1974.