2022 IEEE 61st Conference on Decision and Control (CDC)
December 6-9, 2022. Cancún, Mexico

# Data-Based Actuator Selection for Optimal Control Allocation

Filippos Fotiadis[1], Kyriakos G. Vamvoudakis[1], Zhong-Ping Jiang[2]

2022 IEEE 61st Conference on Decision and Control (CDC) | 978-1-6654-6761-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/CDC51059.2022.9992848

***Abstract*— In this work, we consider an actuator redundant system, i.e., a system with more actuators than the number of effective control inputs, and bring together connections between control allocation, actuator selection, and learning. In this kind of systems, the actuator commands can be chosen to meet a given control objective while still having leftover degrees of freedom to use towards minimizing the overall actuation energy. We show that this energy can be further minimized by optimally selecting the actuators themselves, which we perform in two different scenarios; first, in the case where the control objective is not known beforehand; and second, in the case where the control objective is defined to be a stabilizing state feedback controller. To relax the requirement for knowledge of the system's plant matrix, we compose a novel learning mechanism based on policy iteration, which computes the anti-stabilizing solution to an associated algebraic Riccati equation using trajectory data. Simulations are performed that demonstrate our approach.**

## I. Introduction

A system is defined to be actuator redundant when the number of actuators installed on it is greater than the number of high-level control inputs available for design [1], [2]. In these redundant systems, there exist an infinite number of realizations for the actuator commands that can yield a desired control policy or objective, hence leaving the designer with leftover degrees of freedom that can be utilized to optimize other, unrelated specifications. The problem of optimally allocating the additional degrees of freedom is commonly known as the control allocation (CA) problem, and comprehensive surveys regarding it are given in [3]–[5].

A popular CA specification in actuator redundant systems is the minimization of the closed-loop actuation energy. The solution to this problem yields an analytic expression that involves the generalized inverse of the actuation matrix [6], though one may need to resort to numerical methods in constrained cases that consider saturation limits [7]. The aforementioned results make clear that the actuation matrix of the system directly affects the minimum energy that is attainable in CA. Consequently, in this work, we are motivated to study the problem of actuator selection for optimal CA, where also the dynamics of the system's plant matrix could be unknown.

Much attention has been given lately to the actuator selection problem, in which one is tasked with picking the actuators of the system so that a metric of controllability or resilience is optimized. Examples of metrics used include Gramians-related functions [8]–[11], which are associated with the minimum energy required to drive a linear system to a given state; functions that account for security-aware actuator selection [12]–[14]; and metrics that quantify linear-quadratic optimality [15]–[17]. The aforementioned works, though effective in their purpose, did not consider a system with model uncertainty nor focused on the CA problem.

Learning and estimation are often used as tools to deal with unknown dynamics in control systems, and they were recently used in the context of actuator selection with convergence guarantees. Particularly, by using learning methods, a Gramian-based metric of controllability was provably optimized online in [18] without knowledge of the system dynamics. However, the problem of learning-based actuator selection for optimal control allocation, as well as the consideration of redundant systems, was not considered in [18].

*Contributions of this paper:* Unlike the aforementioned works, here we focus on the actuator selection problem in the context of CA as well as learning. Since the minimum energy solution of the CA problem is directly dependent on the actuators of the system, we define novel actuator selection metrics quantifying this energy and consider two different scenarios; first, we assume an agnostic setup where the desired control policy in CA is unknown; and second, we suppose that the desired control policy in CA has a known, linear in the state structure. In the latter case, the actuator selection metric directly depends on the system's plant matrix through a dual LE defined over the transposition of the plant. Nevertheless, we relax this requirement of system knowledge by expressing the solution to this LE as the anti-stabilizing solution to an algebraic Riccati equation (ARE). This ARE is subsequently solved using a modified version of learning-based policy iteration (PI), which conventionally can only find stabilizing solutions to AREs instead.

*Notation:* The operators $\otimes$ and $\oplus$ will denote the Kronecker product and sum, respectively. Given a matrix $Z \in \mathbb{R}^{n\times m}$, $\text{vec}(Z) = [Z_{1,1}\ Z_{1,2}\ \dots\ Z_{1,m}\ Z_{2,1}\ Z_{2,2}\ \dots\ Z_{n,m}]^{\mathrm{T}}$ will denote the vectorized form of a matrix, whereas $\text{vec}^{-1}$ will perform the inverse of this operation. Additionally, if $Z$ is square and symmetric, we define as $\text{vech}(Z) = [Z_{1,1}\ Z_{1,2}\ \dots\ Z_{1,n}\ Z_{2,2}\ Z_{2,3}\ \dots\ Z_{n,n}]^{\mathrm{T}}$ the half vectorized form of $Z$, and as $\text{vecs}(Z) = [Z_{1,1}\ 2Z_{1,2}\ \dots\ 2Z_{1,n}\ Z_{2,2}\ 2Z_{2,3}\ \dots\ Z_{n,n}]^{\mathrm{T}}$ the scaled half vectorized form of $Z$.

## II. Problem Formulation and Preliminaries

Consider a continuous-time linear system of the form

$$\dot{x}(t) = Ax(t) + Gv(t),\ x(0) = x_0,\ t \geqslant 0, \qquad (1)$$

[1]F. Fotiadis and K. G. Vamvoudakis are with the School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA, USA. Email: {ffotiadis, kyriakos}@gatech.edu.

[2]Z.-P. Jiang is with the Control and Networks Lab, Department of Electrical and Computer Engineering, Tandon School of Engineering, New York University, Brooklyn, NY 11201 USA. Email: zjiang@nyu.edu.

This work was supported in part, by ARO under grant No. W911NF-19 − 1 − 0270, by NSF under grant Nos. CAREER CPS-1851588, S&AS-1849198, and SATC-1801611, and by the Onassis Foundation-Scholarship ID: F ZQ 064 − 1/2020 − 2021.

where $x(t) \in \mathbb{R}^n$ denotes the state with initial condition $x_0 \in \mathbb{R}^n$, $v(t) \in \mathbb{R}^m$ is the control policy, and $A \in \mathbb{R}^{n\times n}$, $G \in \mathbb{R}^{n\times m}$ are the system's state and input matrices, respectively. It is assumed that $v$ is a control policy that is predetermined. For example, $v$ could be a state feedback control law designed for appropriate pole placement of the closed-loop system or a linear quadratic regulator.

### A. Actuator Redundancy

In the problem of control allocation, it is assumed that there exist multiple actuator commands that can effectively yield the closed-loop behavior prescribed by the policy $v$. More specifically, let the control policy $v$ satisfy the following mapping from actuator commands to control inputs:

$$v(t) = Bu(t),\ \forall t \geqslant 0, \tag{2}$$

where $B \in \mathbb{R}^{m\times k}$ is a matrix whose columns comprise the actuators of the system, and $u(t) \in \mathbb{R}^k$ is a vector containing each actuator's commands. Then, if $k > m$ and $B$ has full row rank, the equation (2) has an infinite number of solutions with respect to $u(t)$. Any systems with these properties are called *actuator redundant* [1], [2], because they have more actuators than the minimum needed to construct the policy $v$ from the actuator commands $u$.

**Definition 1.** [2] The system (1) is called actuator redundant if its mapping (2) from actuator commands to control inputs satisfies $k > m$ and $\text{rank}(B) = m$. □

### B. Optimal Control Allocation

Given that the system is actuator redundant, one can choose $u$ so that (2) holds, while still having leftover degrees of freedom over further modifying $u$. For example, consider the control input $u(t) = B^\dagger v(t) + B_\perp z(t),\ t \geqslant 0$, where $B^\dagger = B^{\mathrm{T}}(BB^{\mathrm{T}})^{-1}$ is the Moore–Penrose inverse of $B$, $B_\perp = I - B^\dagger B$ is the null-space projection matrix of $B$, and $z(t) \in \mathbb{R}^n$ is an arbitrary signal. Then, no matter how $z(t)$ is chosen, it always holds that $v(t) = Bu(t)$ (see [2] for the proof). As such, it is natural to search for an actuator command vector $u(t)$ that not only achieves the control policy requirement (2), but also has some form of minimum energy. These two objectives are not mutually exclusive, owing exactly to the actuator redundancy described above.

Let $W = W(B) \in \mathbb{R}^{k\times k}$ be a positive definite, diagonal matrix that assigns a weight to each actuator, i.e., to each column of $B$. Then, the problem can be described pointwise in time $t \geqslant 0$ by the constrained optimization [1]:

$$\begin{aligned} \min_{u(t)} \quad & L(u(t)) = u^{\mathrm{T}}(t)Wu(t) \\ \text{s.t.} \quad & v(t) = Bu(t). \end{aligned} \tag{3}$$

Following [1], the solution to this problem is given by

$$u^\star(t) = W^{-1}B^{\mathrm{T}}(BW^{-1}B^{\mathrm{T}})^{-1}v(t). \tag{4}$$

Note that per Definition 1, the inverse here always exists for actuator redundant systems. Hence, plugging the optimal actuator command (4) in (3) yields the constrained minimum value of the weighted energy:

$$L_B^\star(t) := L(u^\star(t)) = v^{\mathrm{T}}(t)(BW^{-1}B^{\mathrm{T}})^{-1}v(t). \tag{5}$$

### C. Actuator Placement and Learning for Allocation

The purpose of this paper is to bring connections between control allocation, actuator selection, and learning. Specifically, notice that the minimum allocation energy (5) directly depends on the actuator matrix $B$. Hence, if one can choose the actuators that will be used by the system, then the minimum energy (5) can be further optimized, but this time with respect to $B$.

Towards this end, let $B$ be decomposed as $B = [B_1\ B_2]$, where $B_1 \in \mathbb{R}^{m\times k_1}$, $B_2 \in \mathbb{R}^{m\times k_2}$ and $k_1 + k_2 = k$. In this decomposition, $B_1$ represents the part of $B$ that is fixed, and $B_2$ represents the part of $B$ that is free to be selected. Particularly, we define $B_1 = \begin{bmatrix}\beta_1 & \beta_2 & \dots & \beta_{k_1}\end{bmatrix}$ and $B_2 = \begin{bmatrix}b_1 & b_2 & \dots & b_{k_2}\end{bmatrix}$, where $\beta_j \in \mathbb{R}^m$, $j = 1,\dots,k_1$, are fixed actuator columns, and $b_i \in \mathbb{R}^m$, $i = 1,\dots,k_2$, are columns each of which corresponds to an actuator to be selected. Let now $\mathcal{S} = \{s_1,\ \dots,\ s_N\}$ be the set of available actuator columns $s_i \in \mathbb{R}^m$, where $N \geqslant k_2$. Then, the problem to be solved in this work is to choose the columns of the free matrix $B_2$ by solving the optimization:

$$\begin{aligned} \min_{\mathcal{B}\subseteq\mathcal{S}} \quad & f(\mathcal{B}), \\ \text{s.t.} \quad & \text{card}(\mathcal{B}) = k_2, \\ & \mathcal{B} = \{b_1, b_2,\ \dots, b_{k_2}\}, \end{aligned} \tag{6}$$

where $f : 2^{\mathcal{S}} \to \mathbb{R}$ is a function quantifying the optimality of $\mathcal{B}$ with respect to (5), and is to be defined in the following sections based on the characteristics of $v(t)$, $\forall t \geqslant 0$. Since such functions inherently require knowledge of the system's plant matrix $A$, a subsequent purpose of this work is to construct learning algorithms to evaluate those in a data-based manner.

Before proceeding, we require the following assumption, which ensures that system (1)-(2) is indeed actuator redundant per Definition 1.

**Assumption 1.** The matrix $B_1 \in \mathbb{R}^{m\times k_1}$ has full row rank, $k_1 \geqslant m$ and $k_2 > 0$. □

## III. Actuator Selection without prior Information on the Policy $v$

We will initially consider a general setup in which we want to select the system's actuators to reduce the energy in (5), but where the nominal control input $v$ is neither fixed nor known beforehand. It can be seen that a direct optimization of $L_B^\star$ with respect to $B$ is not possible in this scenario, because $L_B^\star$ depends on the undetermined value $v(t)$, $\forall t \geqslant 0$. Nevertheless, an "average" optimization approach can be applied instead, where the actuators are chosen to minimize (5) evenly across all possible directions of $v(t)$ [9], [10]. Particularly, one can choose $f$ to be equal to

$$f_e(\mathcal{B}) = \text{tr}\left((BW^{-1}B^{\mathrm{T}})^{-1}\right). \tag{7}$$

Albeit elegant, the choice (7) entails a computational hurdle; the matrix $(BW^{-1}B^{\mathrm{T}})^{-1}$ can be shown to be equivalent to an inverse of a weighted Gramian, and the trace of such matrices is notorious for being hard to optimize [19].

**Proposition 1.** *Let $E \in \mathbb{R}^{n\times n}$ be the null matrix. Then, $BW^{-1}B^{\mathrm{T}}$ is a weighted controllability Gramian of the control pair $(E, B)$.*

*Proof.* One possible weighted controllability Gramian for the control pair $(E, B)$ is given by

$$\int_0^1 e^{E\tau}BW^{-1}B^{\mathrm{T}}e^{E^{\mathrm{T}}\tau}\mathrm{d}\tau = \int_0^1 BW^{-1}B^{\mathrm{T}}\mathrm{d}\tau = BW^{-1}B^{\mathrm{T}},$$

where we used the fact that $e^{E\tau} = I$ for any $\tau \geqslant 0$. ∎

Relaxations are usually considered to tackle this issue of computational complexity. For example, let $E \in \mathbb{R}^{m\times m}$ be a positive definite matrix. Then, motivated by the inequality

$$\frac{\mathrm{tr}(E^{-1})}{m} \geqslant \frac{m}{\mathrm{tr}(E)} \tag{8}$$

a relaxation to the problem of minimizing $\mathrm{tr}(E^{-1})$ is to maximize $\mathrm{tr}(E)$ instead [20], [21]. Accordingly, this relaxation in our setup is equivalent to choosing $f$ in (6) as:

$$f_{\mathrm{er}}(\mathcal{B}) = -\mathrm{tr}(BW^{-1}B^{\mathrm{T}}). \tag{9}$$

Indeed, such a relaxation is sufficient to reduce the computational complexity to polynomial levels. Particularly, let us denote $W = W(B)$ as $W = \mathrm{diag}\begin{bmatrix} w_{\beta_1} & \dots & w_{\beta_{k_1}} & w_{b_1} & \dots & w_{b_{k_2}}\end{bmatrix}$, where we note again that $W$ is a function of $\mathcal{B}$ (or $B$) which assigns a positive weight to each actuator. Then:

$$\begin{aligned}\mathrm{tr}(BW^{-1}B^{\mathrm{T}}) &= \mathrm{tr}\left(\sum_{j=1}^{k_1}\frac{1}{w_{\beta_j}}\beta_j\beta_j^{\mathrm{T}} + \sum_{i=1}^{k_2}\frac{1}{w_{b_i}}b_ib_i^{\mathrm{T}}\right)\\ &= \sum_{j=1}^{k_1}\frac{1}{w_{\beta_j}}\left\|\beta_j\right\|^2 + \sum_{i=1}^{k_2}\frac{1}{w_{b_i}}\left\|b_i\right\|^2.\end{aligned} \tag{10}$$

As it is evident from the equation above, optimizing $\mathrm{tr}(BW^{-1}B^{\mathrm{T}})$ is equivalent to sorting the values $w_s^{-1}\left\|s\right\|^2$ for all $s \in \mathcal{S}$, and choosing $\mathcal{B}$ so that it contains the $k_2$ elements of $\mathcal{S}$ with the largest such values. Hence, the complexity of solving (6) with $f$ chosen as in (9) is $\mathcal{O}(N\log N)$. It should be noted here that the first summation term in (10) is constant, since $\beta_j$, $j = 1, \dots, k_1$, are fixed.

## IV. Actuator Selection with Prior Information on the Policy $v$

In this section, we consider a more specific setup, where the control input $v$ is assumed to have a known structure. Particularly, we assume that:

$$v(t) = Kx(t), \tag{11}$$

where $K \in \mathbb{R}^{m\times n}$ is a known constant gain such that the closed-loop matrix $A + GK$ is Hurwitz. Such a feedback gain could be designed either using knowledge of the system matrices $A, G$, or by exploiting data and employing model-free learning methods [22], [23]. Additionally, we assume that the initial state $x_0 \in \mathbb{R}^n$ has known variance, given by:

$$\mathbb{E}\left[x_0x_0^{\mathrm{T}}\right] = V \succ 0. \tag{12}$$

In the aforementioned setup, the constrained minimum value of the weighted energy (5) will be given by:

$$L_B^\star(t) = u^{\star\mathrm{T}}(t)Wu^\star(t) = x^{\mathrm{T}}(t)K^{\mathrm{T}}(BW^{-1}B^{\mathrm{T}})^{-1}Kx(t). \tag{13}$$

Notice that the optimal weighted energy (13) now has a much more informative structure than (5), where the unknown evaluation of the function $v$ was involved. Because of this, not only can one choose actuators to optimize (13) "on average", but we can actually minimize (13) directly over an infinite horizon on $t \geqslant 0$. Specifically, we can choose:

$$\begin{aligned}f_p(\mathcal{B}) &= \mathbb{E}\left[\int_0^\infty L_B^\star(t)\mathrm{d}t\right] = \mathbb{E}\left[\int_0^\infty u^{\star\mathrm{T}}(t)Wu^\star(t)\mathrm{d}t\right]\\ &= \mathbb{E}\left[\int_0^\infty x^{\mathrm{T}}(t)K^{\mathrm{T}}(BW^{-1}B^{\mathrm{T}})^{-1}Kx(t)\mathrm{d}t\right].\end{aligned} \tag{14}$$

### A. Model-Based Computation of the Cost

While (14) involves trajectories of the state and an expectation, it can still be written statically as a function of the model matrices $A$, $G$, as stated in the following theorem.

**Theorem 1.** *Consider the system* (1) *under the control policy* (11)*, and with initial state variance given by* (12)*. Then*

$$\mathbb{E}\left[\int_0^\infty x^{\mathrm{T}}(t)K^{\mathrm{T}}(BW^{-1}B^{\mathrm{T}})^{-1}Kx(t)\mathrm{d}t\right] = \mathrm{tr}(QR) = \mathrm{tr}(PV)$$

*where $R = K^{\mathrm{T}}(BW^{-1}B^{\mathrm{T}})^{-1}K$, and $P$, $Q \in \mathbb{R}^{n\times n}$ are symmetric matrices, with $P \succeq 0$, $Q \succ 0$, satisfying the LEs:*

$$(A + GK)^{\mathrm{T}}P + P(A + GK) + R = 0, \tag{15}$$

$$(A + GK)Q + Q(A + GK)^{\mathrm{T}} + V = 0. \tag{16}$$

*Proof.* The proof is omitted due to space limitations. ∎

According to Theorem 1, optimizing the cost (14) has once again an increased computational complexity, due to the appearance of the matrix inverse $(BW^{-1}B^{\mathrm{T}})^{-1}$ through $R$. Fortunately, however, Theorem 1 also states that the solution of just one LE is required to evaluate $f$ at all points in $2^{\mathcal{S}}$. Specifically, if $f$ is chosen as in (14), then its realization is

$$f_p(\mathcal{B}) = \mathrm{tr}(QR) \tag{17}$$

where we can see from (16) that $Q$ is completely independent of $\mathcal{B}$. Therefore, although a brute-force algorithm to solve (6) may require a large number of iterations, the per iteration complexity will remain at relatively low levels as it will not involve the iterative solution of the LE (16).

For the same reason, it would be in one's interest to avoid using the alternate form $f(\mathcal{B}) = \mathrm{tr}(PV)$ provided by Theorem 1, because $P$ is the solution of an LE that depends directly on $\mathcal{B}$. Hence, a brute force algorithm that would evaluate $f(\mathcal{B}) = \mathrm{tr}(PV)$ across all possible points in $2^{\mathcal{S}}$ would have to solve the LE (15) at each iteration, which would then lead to increased per-iteration complexity.

### B. Model-Based Computation of the Cost with Relaxation

Consider now the scenario where $m = n$, so that $G$ and $K$ are square matrices. Additionally, let us assume, since $K$ is square, that it is also invertible. Then we are once again able to use the inequality (8) to motivate the same relaxation employed in the previous section; in lieu of minimizing $\mathrm{tr}(QR) = \mathrm{tr}(R^{1/2}QR^{1/2})$, we can instead minimize:

$$f_{pr}(\mathcal{B}) = -\mathrm{tr}((R^{1/2}QR^{1/2})^{-1}) = -\mathrm{tr}(R^{-1}Q^{-1}). \tag{18}$$

The inverse of $Q$ here exists owing to Theorem 1, while the inverse of $R$ exists due to Assumption 1 and the non-singularity of $K$. Further analysis of (18) yields:

$$\begin{aligned}\mathrm{tr}(R^{-1}Q^{-1}) &= \mathrm{tr}(K^{-1}BW^{-1}B^{\mathrm{T}}K^{-\mathrm{T}}Q^{-1}) \\ &= \mathrm{tr}(W^{-1}B^{\mathrm{T}}K^{-\mathrm{T}}Q^{-1}K^{-1}B) \\ =\sum_{j=1}^{k_1}\frac{1}{w_{\beta_j}}\beta_j^{\mathrm{T}}K^{-\mathrm{T}}Q^{-1}K^{-1}\beta_j &+ \sum_{i=1}^{k_2}\frac{1}{w_{b_i}}b_i^{\mathrm{T}}K^{-\mathrm{T}}Q^{-1}K^{-1}b_i.\end{aligned}$$

Hence, we can now notice that the employed relaxation once again brings the complexity down to polynomial levels; to optimize $f_{pr}$, one only needs to sort the values $\frac{1}{w_s}s^{\mathrm{T}}K^{-\mathrm{T}}Q^{-1}K^{-1}s$ for all $s \in \mathcal{S}$, and choose $\mathcal{B}$ to contain the $k_2$ elements of $\mathcal{S}$ with the largest such values. Therefore, the complexity of solving (6) with $f$ as in (18) is $\mathcal{O}(N\log N)$, and only one LE of the form (16) needs to be solved.

## V. Partially Model-Free Cost Computation

### A. Cost Expression Through the Anti-Stabilizing Solution to an ARE

In this section, we proceed to derive a learning algorithm that will allow for the evaluation of the costs (17)-(18) without knowledge of the system's drift matrix $A$. To this end, note that these costs depend on $A$ only through the matrices $Q$ and $Q^{-1}$. Therefore, if we manage to learn/estimate $Q$ or $Q^{-1}$ using measured data, our purpose for partially model-free evaluation of the costs (17)-(18) will have been fulfilled.

Unlike $P$, which satisfies the nominal LE (15) for the plant $A+GK$, the matrix $Q$ is a solution to the dual LE (16) instead; that is, the order of the transpose operator in (16) is switched when compared to (15). Consequently, to learn $Q$ using measured data and describe (16) in a model-free manner, trajectories from the transpose plant $A^{\mathrm{T}}$ are needed. To bypass this issue, and since $Q$ is positive definite and thus invertible, we pre- and post-multiply the LE (16) by $Q^{-1}$ and turn it into the following ARE for the plant $A+GK$:

$$(A+GK)^{\mathrm{T}}X + X(A+GK) + XVX = 0, \tag{19}$$

where $X_a = Q^{-1}$ is a solution to (19).

Various methods for solving AREs model-free exist in the literature [22], [24], which are essentially a learning-based formulation of procedures known as policy iteration (PI) or Kleinman's algorithm [25], [26]. Accordingly, it is tempting to use these methods to solve (19) in a learning-based manner and compute $Q^{-1}$ without knowledge of $A$. However, there is a pitfall in the ARE (19) that would cause the implementations of [22], [24] to fail to find $Q^{-1}$; these methods can only find the *stabilizing* solution to an ARE.

**Definition 2.** Consider the general form of an ARE:

$$\bar{A}^{\mathrm{T}}\bar{X} + \bar{X}\bar{A} + \bar{Q} - \bar{X}\bar{\Sigma}\bar{X} = 0 \tag{20}$$

where $\bar{A}, \bar{X}, \bar{Q}, \bar{\Sigma} \in \mathbb{R}^{n\times n}$, and $\bar{X}, \bar{Q}, \bar{\Sigma}$ are symmetric.

1) A solution $\bar{X}_s \in \mathbb{R}^{n\times n}$ to the ARE (20) is called stabilizing with respect to $\bar{A}$ if $\bar{A} - \bar{\Sigma}\bar{X}_s$ is Hurwitz.
2) A solution $\bar{X}_a \in \mathbb{R}^{n\times n}$ to the ARE (20) is called anti-stabilizing with respect to $\bar{A}$ if all eigenvalues of $\bar{A} - \bar{\Sigma}\bar{X}_a$ have strictly positive real parts. □

**Algorithm 1** PI to compute $Q^{-1}$

1: Let $i = 0$, $\epsilon > 0$. Start with a matrix $Y_0 \in \mathbb{R}^{n\times n}$ such that $-(A+GK) - Y_0$ is Hurwitz.
2: **repeat**
3: Compute $X_i$ by solving the LE:

$$\begin{aligned}-(A+GK+Y_i)^{\mathrm{T}}X_i - X_i(A+GK+Y_i) \\ + Y_i^{\mathrm{T}}V^{-1}Y_i = 0.\end{aligned} \tag{22}$$

4: Compute $Y_{i+1}$ as $Y_{i+1} = VX_i$ and set $i = i+1$.
5: **until** $\|Y_i - Y_{i-1}\|_{\mathrm{F}} < \epsilon$

As stated next, the matrix $Q^{-1}$ is not the stabilizing solution to (19), hence the learning-based PI methods of [22], [24] would fail to compute it; in fact, $Q^{-1}$ is the anti-stabilizing solution of (19).

**Lemma 1.** *Consider the ARE* (19)*, where $X \in \mathbb{R}^{n\times n}$ is the variable to be solved for. Then:*

1) *$X_s = 0$ is a stabilizing solution to* (19) *with respect to $A+GK$.*
2) *$X_a = Q^{-1}$ is an anti-stabilizing solution to* (19) *with respect to $A+GK$.*

*Proof.* The proof is omitted due to space limitations. ■

### B. PI for the ARE's Anti-Stabilizing Solution

Although $Q^{-1}$ is not the stabilizing solution of (19) with respect to $A+GK$, it can prove handful that it has been characterized as the anti-stabilizing one. Particularly, due to this property, the matrix $Q^{-1}$ can be shown to be a stabilizing solution to an alternate ARE instead.

**Lemma 2.** *$X_a = Q^{-1}$ is the stabilizing solution of*

$$-(A+GK)^{\mathrm{T}}X - X(A+GK) - XVX = 0. \tag{21}$$

*with respect to $-(A+GK)$.*

*Proof.* $Q^{-1}$ is a solution to (21) due to the fact that (21) is just the negation of (19). In addition, $-(A+GK) - VQ^{-1}$ is Hurwitz owing to Lemma 1, hence $Q^{-1}$ is stabilizing with respect to $-(A+GK)$. ■

The matrix $Q^{-1}$ has now been characterized as a stabilizing solution of (21) with respect to $-(A+GK)$. Therefore, can now use the PI procedure to compute it in an iterative fashion, which we could not have done directly on (19) with respect to $A+GK$. Although the PI algorithm is inherently model-based as seen from Algorithm 1, it is the first step towards computing $Q^{-1}$ in a data-based fashion [22], [24].

Notice now that the constant term in the ARE (21) is zero, hence the observability assumptions imposed in [22], [24], [26] for the constant term do not hold. However, the convergence of Algorithm 1 is still guaranteed.

**Theorem 2.** *Consider the sequence of matrices $\{X_i\}_{i\in\mathbb{N}}$, $\{Y_i\}_{i\in\mathbb{N}}$ generated by Algorithm 1. Then, for all $i \in \mathbb{N}$ :*

1) *$-(A+GK+Y_i)$ is Hurwitz,*
2) *$Q^{-1} \preceq X_{i+1} \preceq X_i$,*
3) *$\lim_{i\to\infty} X_i = Q^{-1}$.*

*Proof.* The proof is omitted due to space limitations. ■

Like all PI-based algorithms, Algorithm 1 requires that the initial matrix $Y_0$ is stabilizing, i.e., that $-(A+GK)-Y_0$ is Hurwitz. However, in the present framework, such a stabilizing matrix can be constructed by just the mere knowledge of a lower bound to the minimum eigenvalue of $A$.

**Theorem 3.** *Let $Y_0 = -GK - \alpha I$, where $\alpha < \alpha^\star = min_{i\in\{1,\dots,n\}} Re(\lambda_i(A))$. Then, $-(A+GK)-Y_0$ is Hurwitz.*

*Proof.* The proof is omitted due to space limitations. ∎

### C. Learning-Based PI for the ARE's Anti-Stabilizing Solution

The analysis of the previous section allowed us to successively approximate the anti-stabilizing solution $Q^{-1}$ of (19) using PI. In view of this result, we now proceed to derive a data-based formulation for Algorithm 1 which does not require knowledge of the system's matrix $A$. Notice that if we pre- and post-multiply (22) by $x(t)$, then:

$$\begin{aligned}&-x^{\mathrm{T}}(t)(A^{\mathrm{T}}X_i+X_iA)x(t)-x^{\mathrm{T}}(t)(GK+Y_i)^{\mathrm{T}}X_ix(t)\\&-x^{\mathrm{T}}(t)X_i(GK+Y_i)x(t)+x^{\mathrm{T}}(t)Y_i^{\mathrm{T}}V^{-1}Y_ix(t)=0,\end{aligned}$$

or equivalently:

$$\begin{aligned}&-\frac{\mathrm{d}}{\mathrm{d}t}(x^{\mathrm{T}}(t)X_ix(t))+x^{\mathrm{T}}(t)X_iGv(t)+(Gv(t))^{\mathrm{T}}X_ix(t)\\&-x^{\mathrm{T}}(t)X_i(GK+Y_i)x(t)-((GK+Y_i)x(t))^{\mathrm{T}}(t)X_ix(t)\\&\qquad+x^{\mathrm{T}}(t)Y_i^{\mathrm{T}}V^{-1}Y_ix(t)=0.\end{aligned}$$

Note that here, $v$ could be any signal generating the trajectory data of $x$, and not necessarily given by (11). If we let $T>0$ and integrate over $[t,\ t+T]$ then:

$$\begin{aligned}&x^{\mathrm{T}}(t)X_ix(t)-x^{\mathrm{T}}(t+T)X_ix(t+T)+\int_t^{t+T}\big(x^{\mathrm{T}}(\tau)X_iGv(\tau)\\&+(Gv(\tau))^{\mathrm{T}}X_ix(\tau)-x^{\mathrm{T}}(\tau)X_i(GK+Y_i)x(\tau)\\&-((GK+Y_i)x(\tau))^{\mathrm{T}}X_ix(\tau)+x^{\mathrm{T}}(\tau)Y_i^{\mathrm{T}}V^{-1}Y_ix(\tau)\big)\mathrm{d}\tau=0.\end{aligned}\tag{23}$$

Using Kronecker algebra, equation (23) can be written as:

$$\Theta_i^{\mathrm{T}}(t)\mathrm{vec}(X_i)=\Phi_i(t)\tag{24}$$

where

$$\begin{aligned}&\Theta_i(t)=\delta_{\mathrm{xx}}(t)+(G\otimes I_n)J_{\mathrm{vx}}(t)+(I_n\otimes G)J_{\mathrm{xv}}(t)\\&\qquad-((GK+Y_i)\oplus(GK+Y_i))J_{\mathrm{xx}}(t),\\&\Phi_i(t)=-J_{\mathrm{xx}}^{\mathrm{T}}(t)\mathrm{vec}(Y_i^{\mathrm{T}}V^{-1}Y_i),\\&\delta_{\mathrm{xx}}(t)=x(t)\otimes x(t)-x(t+T)\otimes x(t+T),\\&J_{\mathrm{vx}}(t)=\int_t^{t+T}v(\tau)\otimes x(\tau)\mathrm{d}\tau,\ J_{\mathrm{xv}}(t)=\int_t^{t+T}x(\tau)\otimes v(\tau)\mathrm{d}\tau,\\&J_{\mathrm{xx}}(t)=\int_t^{t+T}x(\tau)\otimes x(\tau)\mathrm{d}\tau.\end{aligned}$$

The redundant parameters in (24), which are owed to the symmetry of $X_i$, can be eliminated by rewriting (24) as:

$$\bar{\Theta}_i^{\mathrm{T}}(t)\mathrm{vecs}(X_i)=\Phi_i(t)\tag{25}$$

where $\bar{\Theta}(t)=\mathrm{vech}(\mathrm{vec}^{-1}\Theta_i(t))$. Note that the matrix $A$ is not involved in (25). Hence, by gathering data along the trajectories of (1) in the form of matrices $J_{\mathrm{xx}}(t_k)$, $J_{\mathrm{xv}}(t_k)$, $J_{\mathrm{vx}}(t_k)$, $\delta_{\mathrm{xx}}(t_k)$, where $t_k>0$ is a sampling instant for $k\in\{0,\dots,K\}$, $K\in\mathbb{N}$, we can write the following using (25):

$$\sum_{k=0}^{K}\bar{\Theta}_i(t_k)\bar{\Theta}_i^{\mathrm{T}}(t_k)\mathrm{vecs}(X_i)=\sum_{k=0}^{K}\bar{\Theta}_i(t_k)\Phi_i(t_k)\tag{26}$$

which can be solved for $X_i$ given that the corresponding data matrices are sufficiently rich, which we assume next.

**Assumption 2.** There exists $K_0\in\mathbb{N}$, such that if $K\geqslant K_0$ then $\sum_{k=0}^{K}\bar{\Theta}_i(t_k)\bar{\Theta}_i^{\mathrm{T}}(t_k)$ is non-singular, for all $i\in\mathbb{N}$. □

Given Assumption (2), the solution to (26) is given by:

$$\mathrm{vecs}(X_i)=\left(\sum_{k=0}^{K}\bar{\Theta}_i(t_k)\bar{\Theta}_i^{\mathrm{T}}(t_k)\right)^{-1}\sum_{k=0}^{K}\bar{\Theta}_i(t_k)\Phi_i(t_k).\tag{27}$$

This gives rise to the learning-based PI Algorithm 2 for computing the anti-stabilizing solution $Q^{-1}$ to (19). Consequently, we can now solve the actuator selection problem (6) with cost functions (17)-(18) in a data-based manner.

We summarize the convergence of Algorithm 2 next.

**Algorithm 2** Learning-Based PI to Compute $Q^{-1}$

1: Let $i=0$, $\epsilon>0$. Start with a matrix $Y_0\in\mathbb{R}^{n\times n}$ such that $-(A+GK)-Y_0$ is Hurwitz.
2: **repeat**
3: Compute $X_i$ from (27).
4: Compute $Y_{i+1}$ as $Y_{i+1}=VX_i$ and set $i=i+1$.
5: **until** $\|Y_i-Y_{i-1}\|_{\mathrm{F}}<\epsilon$

**Theorem 4.** *Let Assumption 2 hold. Consider the sequence of matrices $\{X_i\}_{i\in\mathbb{N}}$, $\{Y_i\}_{i\in\mathbb{N}}$ generated by Algorithm 2. Then, for all $i\in\mathbb{N}$ :*

*1) $-(A+GK+Y_i)$ is Hurwitz,*
*2) $Q^{-1}\leq X_{i+1}\leq X_i$,*
*3) $\lim_{i\to\infty}X_i=Q^{-1}$.*

*Proof.* Given Assumption 2, equation (26) has a unique solution that is given by (27). Additionally, any matrix $X_i$ satisfying (22) also satisfies (26), for all $i\in\mathbb{N}$. Hence, the proof follows directly from Theorem 2. ∎

## VI. Simulations

We consider the linearized model of an electric vertical take-off and landing (eVTOL) aircraft [27], with $n=9$ states and $m=5$ high-level control inputs. Its input matrix is given by $G=\{e_4,e_5,e_6,e_7,e_9\}$, where $e_i\in\mathbb{R}^9$ is a unit vector with $i$-th entry equal to unity. The set $\mathcal{S}=\{s_1,s_2,\dots,s_{14}\}$ of available actuators is shown at the top of next page, while a full description of the plant matrix $A$ can be found in [27].

The initial set of actuators is $B_1=[s_1\ s_2\ s_3\ s_{13}\ s_{14}]$, and we want to augment it with $k_2=6$ additional actuators from $\mathcal{S}$. To do this, we solve (6), where the cost function $f$ is given by (17), the matrix $K$ is a stabilizing gain, the variance matrix is $V=0.1I_9$, and the weight for each actuator is $w_s=1$, for all $s\in\mathcal{S}$. To learn the matrix $Q$ of the cost (17) without knowing $A$, Algorithm 2 is applied after gathering data from the system for a period of 9 seconds using the control input $v(t)=Kx(t)+[1\ 1\ 1\ 1\ 1]^{\mathrm{T}}0.01\sum_{j=1}^{100}\sin(\omega_jt)$, where $\omega_j$ are frequencies imposing exploration and are randomly chosen over $[-50,\ 50]$ for all $j\in\{1,\dots,100\}$.

$$\mathcal{S} = \left\{ \begin{pmatrix} 0.6825 \\ 0.8888 \\ -0.1503 \\ -0.3510 \\ -0.6197 \end{pmatrix}, \begin{pmatrix} 10^{-4} \\ 10^{-4} \\ 0 \\ 10^{-4} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 10^{-4} \\ 0 \\ 10^{-4} \\ 0 \end{pmatrix}, \begin{pmatrix} -0.6825 \\ 0.8888 \\ 0.1503 \\ -0.3510 \\ -0.6197 \end{pmatrix}, \begin{pmatrix} -10^{-4} \\ 10^{-4} \\ 0 \\ 10^{-4} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 10^{-4} \\ 0 \\ 10^{-4} \\ 0 \end{pmatrix}, \begin{pmatrix} 1.2330 \\ -0.2469 \\ -0.0419 \\ -0.0332 \\ -0.4254 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 10^{-4} \\ 10^{-4} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 10^{-4} \\ 10^{-4} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ 10^{-4} \\ 0 \end{pmatrix}, \begin{pmatrix} -1.2330 \\ -0.2469 \\ 0.0419 \\ -0.0332 \\ -0.4254 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -10^{-4} \\ 10^{-4} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -10^{-4} \\ 10^{-4} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ 10^{-4} \\ 0 \end{pmatrix} \right\}$$

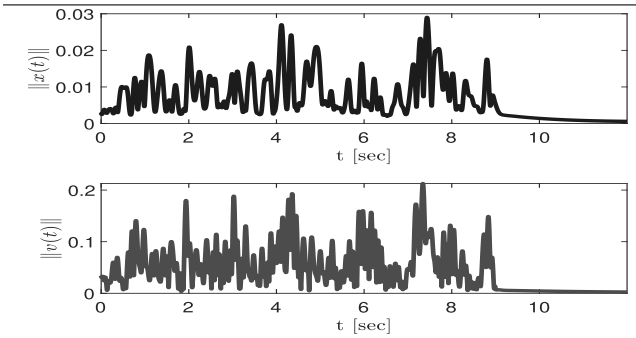


Fig. 1. Evolution of $\|x(t)\|$ and $\|v(t)\|$ in the data-gathering phase.

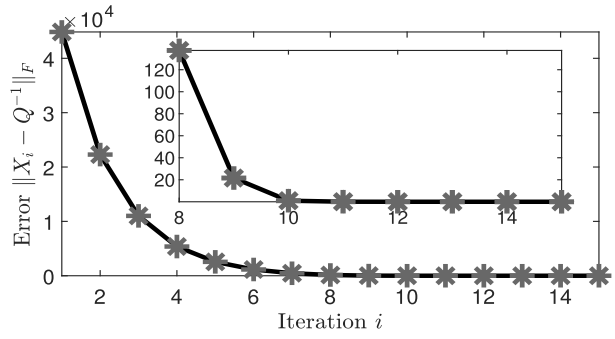


Fig. 2. Evolution of the error norm $\left\|X_i - Q^{-1}\right\|_{\mathrm{F}}$, $i \in \mathbb{N}$.

The evolution of the norm of the trajectories during the data-gathering phase is shown in Figure 1, while Figure 2 shows the distance of the sequence $\{X_i\}_{i\in\mathbb{N}}$ from the matrix $Q^{-1}$. It can be seen that after 15 iterations, this distance practically vanishes, and $X_{15} \approx Q^{-1}$. Subsequently, the matrix $X_{15}^{-1}$ is used as a substitute of $Q$ and the actuator placement optimization problem (6) is solved. The optimal solution is found to be $\mathcal{B}^\star = \{s_2, s_3, s_4, s_6, s_7, s_{11}\}$.

## VII. Conclusion

We considered an actuator redundant system and performed actuator selection for optimal control allocation. To render this procedure model-free, we designed a data-based algorithm to compute the costs of the corresponding optimization problems without knowledge of the system's plant matrix. Future work will include an extension to a completely adaptive system, where both learning, actuator selection, and control allocation take place in real-time.

## References


[1] O. Härkegård and S. T. Glad, "Resolving actuator redundancy—optimal control vs. control allocation," *Automatica*, vol. 41, no. 1, pp. 137–144, 2005.

[2] P. Kolaric, V. G. Lopez, and F. L. Lewis, "Optimal dynamic control allocation with guaranteed constraints and online reinforcement learning," *Automatica*, vol. 122, p. 109265, 2020.

[3] K. A. Bordignon, *Constrained control allocation for systems with redundant control effectors*. Virginia Polytechnic Institute and State University, 1996.

[4] T. A. Johansen and T. I. Fossen, "Control allocation—a survey," *Automatica*, vol. 49, no. 5, pp. 1087–1103, 2013.

[5] M. W. Oppenheimer, D. B. Doman, and M. A. Bolender, "Control allocation for over-actuated systems," in *2006 14th Mediterranean Conference on Control and Automation*, pp. 1–6, IEEE, 2006.

[6] L. Zaccarian, "Dynamic allocation for input redundant control systems," *Automatica*, vol. 45, no. 6, pp. 1431–1438, 2009.

[7] T. A. Johansen, T. I. Fossen, and P. Tøndel, "Efficient optimal constrained control allocation via multiparametric programming," *Journal of guidance, control, and dynamics*, vol. 28, no. 3, pp. 506–515, 2005.

[8] F. Pasqualetti, S. Zampieri, and F. Bullo, "Controllability metrics, limitations and algorithms for complex networks," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 1, pp. 40–52, 2014.

[9] T. H. Summers, F. L. Cortesi, and J. Lygeros, "On submodularity and controllability in complex dynamical networks," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 1, pp. 91–101, 2015.

[10] V. Tzoumas, M. A. Rahimian, G. J. Pappas, and A. Jadbabaie, "Minimal actuator placement with bounds on control effort," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 1, pp. 67–78, 2015.

[11] M. Siami, A. Olshevsky, and A. Jadbabaie, "Deterministic and randomized actuator scheduling with guaranteed performance bounds," *IEEE Transactions on Automatic Control*, 2020.

[12] J. Milošević, A. Teixeira, K. H. Johansson, and H. Sandberg, "Actuator security indices based on perfect undetectability: Computation, robustness, and sensor placement," *IEEE Transactions on Automatic Control*, vol. 65, no. 9, pp. 3816–3831, 2020.

[13] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Robust and adaptive sequential submodular optimization," *IEEE Transactions on Automatic Control*, 2020.

[14] M. Pirani, E. Nekouei, H. Sandberg, and K. H. Johansson, "A game-theoretic framework for security-aware sensor placement problem in networked control systems," in *2019 American Control Conference (ACC)*, pp. 114–119, IEEE, 2019.

[15] T. Summers, "Actuator placement in networks using optimal control performance metrics," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 2703–2708, IEEE, 2016.

[16] N. Darivandi, K. Morris, and A. Khajepour, "An algorithm for lq optimal actuator location," *Smart materials and structures*, vol. 22, no. 3, p. 035001, 2013.

[17] M.-A. Belabbas, "Geometric methods for optimal sensor design," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 472, no. 2185, p. 20150312, 2016.

[18] F. Fotiadis and K. G. Vamvoudakis, "Learning-based actuator placement for uncertain systems," in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 90–95, IEEE, 2021.

[19] T. Summers and M. Kamgarpour, "Performance guarantees for greedy maximization of non-submodular set functions in systems and control," *arXiv preprint arXiv:1712.04122*, 2017.

[20] A. Olshevsky, "On a relaxation of time-varying actuator placement," *IEEE Control Systems Letters*, vol. 4, no. 3, pp. 656–661, 2020.

[21] T. H. Summers and J. Lygeros, "Optimal sensor and actuator placement in complex dynamical networks," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 3784–3789, 2014.

[22] Y. Jiang and Z.-P. Jiang, "Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics," *Automatica*, vol. 48, no. 10, pp. 2699–2704, 2012.

[23] K. G. Vamvoudakis, "Q-learning for continuous-time linear systems: A model-free infinite horizon optimal control approach," *Systems & Control Letters*, vol. 100, pp. 14–20, 2017.

[24] D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F. L. Lewis, "Adaptive optimal control for continuous-time linear systems based on policy iteration," *Automatica*, vol. 45, no. 2, pp. 477–484, 2009.

[25] M. Abu-Khalaf and F. L. Lewis, "Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network hjb approach," *Automatica*, vol. 41, no. 5, pp. 779–791, 2005.

[26] D. Kleinman, "On an iterative technique for riccati equation computations," *IEEE Transactions on Automatic Control*, vol. 13, no. 1, pp. 114–115, 1968.

[27] E. C. Suiçmez, "Full Envelope Nonlinear Controller Design for a Novel Electric VTOL (eVTOL) Air-taxi via INDI Approach Combined with CA," in *Dissertation, Middle East Technical University*, 2021.