Optimal Linearization via Quadratic Programming

Junjie Shen[1] and Dennis Hong[1]

***Abstract*—The technique of linearization for nonlinear systems around some operating point has been widely used for analysis and synthesis of the system behavior within a certain operating range. Conventional linearization methods include the analytical linearization (AL) method using the Jacobian matrix, the result of which usually works only for a sufficiently small region, as well as the numerical linearization (NL) method based on small perturbation, the accuracy of which is usually not guaranteed. In this letter, we propose an optimal linearization method via quadratic programming (OLQP). We start with uniform data sampling within the neighborhood of the operating point based on the nonlinear ordinary differential equation (ODE). We then find the best linear model that fits to these sample points with a QP formulation. The OLQP solution is derived in closed form with proved convergence to the AL solution. Two examples of nonlinear systems are investigated in terms of linearization and results are compared among these linearization methods, which has shown the proposed OLQP method features a great balance between model accuracy and computational complexity. Moreover, the OLQP method offers additional options in controller design by tuning its parameters.**

***Index Terms*—Optimization and optimal control, performance evaluation and benchmarking.**

I. INTRODUCTION

ALMOST all systems in reality are nonlinear. However, there are much more well-established analysis and synthesis tools for linear systems due to simplicity. The technique of linearization is accordingly developed and widely used to approximate the nonlinear system by a corresponding linear model so that linear system theories can be readily applied to the nonlinear system. This approach of studying nonlinear systems has been proved effective in many applications, e.g., stability analysis of equilibrium point [1]. We herein review the existing linearization methods.

A. Review of Linearization Methods

Consider a nonlinear system governed by a nonlinear ordinary differential equation (ODE):

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}), \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the vector of state variables, $\boldsymbol{u} \in \mathbb{R}^m$ is the vector of control inputs, and $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a nonlinear function. Suppose the function $\boldsymbol{f}$ is continuously differentiable at some operating point $(\boldsymbol{x}_o, \boldsymbol{u}_o) \in \mathbb{R}^n \times \mathbb{R}^m$, then the system can be linearized about this point using the following methods:

1) Analytical Linearization (AL)

Define deviation variables $\Delta\boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}_o$ and $\Delta\boldsymbol{u} = \boldsymbol{u} - \boldsymbol{u}_o$. The AL solution of the original nonlinear system (1) about the operating point $(\boldsymbol{x}_o, \boldsymbol{u}_o)$ is thus given by

$$\Delta\dot{\boldsymbol{x}} = \boldsymbol{A}\Delta\boldsymbol{x} + \boldsymbol{B}\Delta\boldsymbol{u}, \tag{2}$$

where

$$\boldsymbol{A} = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{(\boldsymbol{x},\boldsymbol{u})=(\boldsymbol{x}_o,\boldsymbol{u}_o)} \in \mathbb{R}^{n\times n} \tag{3a}$$

and

$$\boldsymbol{B} = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right|_{(\boldsymbol{x},\boldsymbol{u})=(\boldsymbol{x}_o,\boldsymbol{u}_o)} \in \mathbb{R}^{n\times m} \tag{3b}$$

are the constant Jacobian matrices. The AL solution captures the most accurate dynamic information around the operating point in a linear manner. However, the range within which the linear approximation is valid is unknown and usually small [2]. In addition, it is sometimes tedious to compute it out in terms of symbolic calculations. Fortunately, we have automatic differentiation (AD) [3] tool which can numerically evaluate the exact AL solution.

2) Numerical Linearization (NL)

Starting from $\boldsymbol{x}_o$, two other state vectors can be created, e.g., $\boldsymbol{x}_i^+ = \boldsymbol{x}_o + h\boldsymbol{e}_i$ and $\boldsymbol{x}_i^- = \boldsymbol{x}_o - h\boldsymbol{e}_i$, $i = 1, \ldots, n$, where $\boldsymbol{e}_i \in \mathbb{R}^n$ is the standard basis vector with its ith entry equal to 1 and 0 for the rest, and $h \in \mathbb{R}$ is a small positive perturbation. That is, the ith component of the state vector $\boldsymbol{x}_i^\pm$ is perturbed from $\boldsymbol{x}_o$ by $\pm h$. By further setting $\boldsymbol{u} = \boldsymbol{u}_o$, their time-derivatives can be calculated from (1) as $\dot{\boldsymbol{x}}_o$, $\dot{\boldsymbol{x}}_i^+$, and $\dot{\boldsymbol{x}}_i^-$, respectively. The ith column of the matrix $\boldsymbol{A}$ of (3a), denoted as $\boldsymbol{a}_i$, can thus be approximated by

$$\boldsymbol{a}_i \approx \left(\dot{\boldsymbol{x}}_i^+ - \dot{\boldsymbol{x}}_o\right)/h \tag{4a}$$

for a forward-difference approximation (FDA), or

$$\boldsymbol{a}_i \approx \left(\dot{\boldsymbol{x}}_o - \dot{\boldsymbol{x}}_i^-\right)/h \tag{4b}$$

for a backward-difference approximation (BDA), or

$$\boldsymbol{a}_i \approx \left(\dot{\boldsymbol{x}}_i^+ - \dot{\boldsymbol{x}}_i^-\right)/(2h) \tag{4c}$$

for a central-difference approximation (CDA) [4], [5]. The matrix $\boldsymbol{B}$ of (3b) can also be approximated in a similar way by perturbing the control inputs one after another while fixing $\boldsymbol{x} = \boldsymbol{x}_o$. It is really straightforward to apply the NL method but the accuracy is usually not guaranteed.

Manuscript received February, 22, 2020; revised April, 20, 2020; accepted June, 5, 2020.

This letter was recommended for publication by Editor Dezhen Song upon evaluation of the Associate Editor and Reviewers' comments.

[1]Junjie Shen and Dennis Hong are with the Robotics and Mechanisms Laboratory, the Department of Mechanical and Aerospace Engineering, University of California, Los Angeles, CA 90095, USA junjieshen@ucla.edu dennishong@ucla.edu

Digital Object Identifier (DOI): see top of this page.

3) Statistical Linearization (SL)

SL method [6], [7] determines the constant matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ jointly by minimizing the expectation

$$E\left[\|\boldsymbol{f}(\boldsymbol{z}_o+\boldsymbol{\xi})-\boldsymbol{f}(\boldsymbol{z}_o)-\boldsymbol{W}\boldsymbol{\xi}\|_2^2\right] \tag{5a}$$

with respect to $\boldsymbol{W}=[\boldsymbol{A} \;\; \boldsymbol{B}]$, where $\boldsymbol{z}_o=\left[\boldsymbol{x}_o^T,\boldsymbol{u}_o^T\right]^T$, $\|.\|_2$ is the Euclidean norm, and $\boldsymbol{\xi}$ is a vector of random variables with zero mean and covariance matrix $\Xi=E\left[\boldsymbol{\xi}\boldsymbol{\xi}^T\right]$. The unique optimal solution is given by

$$\boldsymbol{W}^*=E\left[\left(\boldsymbol{f}\left(\boldsymbol{z}_o+\boldsymbol{\xi}\right)-\boldsymbol{f}\left(\boldsymbol{z}_o\right)\right)\boldsymbol{\xi}^T\right]\Xi^{-1}. \tag{5b}$$

The random vector $\boldsymbol{\xi}$ under consideration is usually jointly Gaussian, e.g., with zero mean and variance h for each entry. Although the SL solution is on average closer to the given nonlinear system, it gets much more computationally expensive in terms of expectation calculation, not to mention the tremendous number of dimensions for practical systems.

4) Least Squares Optimal Linearization (LSOL)

LSOL method [2], [8], [9] determines the constant matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ jointly by minimizing the integral

$$\int_{\boldsymbol{Z}}\|\boldsymbol{f}(\boldsymbol{z}_o+\boldsymbol{z})-\boldsymbol{f}(\boldsymbol{z}_o)-\boldsymbol{W}\boldsymbol{z}\|_2^2\,\mathrm{d}\boldsymbol{z} \tag{6a}$$

with respect to $\boldsymbol{W}=[\boldsymbol{A} \;\; \boldsymbol{B}]$ over some finite region of interest $\boldsymbol{Z}$ around $\boldsymbol{z}_o$, e.g., a hypercube of edge $2h$, where $\boldsymbol{z}=\left[\Delta\boldsymbol{x}^T,\Delta\boldsymbol{u}^T\right]^T$. The unique optimal solution is given by

$$\boldsymbol{W}^*=\left(\int_{\boldsymbol{Z}}(\boldsymbol{f}(\boldsymbol{z}_o+\boldsymbol{z})-\boldsymbol{f}(\boldsymbol{z}_o))\boldsymbol{z}^T\mathrm{d}\boldsymbol{z}\right)\left(\int_{\boldsymbol{Z}}\boldsymbol{z}\boldsymbol{z}^T\mathrm{d}\boldsymbol{z}\right)^{-1}. \tag{6b}$$

One can consider the LSOL method similar to the SL method but with a uniform distribution instead. Compared with SL, LSOL performs better in terms of competent model accuracy yet with less computational complexity.

5) Optimal Linearization via Domain Densification (OLDD)

OLDD method [10] determines the constant matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ jointly by minimizing the summation

$$\sum_{\boldsymbol{z}_s\in\boldsymbol{S}}\|\boldsymbol{f}(\boldsymbol{z}_o+\boldsymbol{z}_s)-\boldsymbol{f}(\boldsymbol{z}_o)-\boldsymbol{W}\boldsymbol{z}_s\|_2^2 \tag{7a}$$

with respect to $\boldsymbol{W}=[\boldsymbol{A} \;\; \boldsymbol{B}]$ for a given finite set $\boldsymbol{S}$ of points $\boldsymbol{z}_s=\left[\Delta\boldsymbol{x}_s^T,\Delta\boldsymbol{u}_s^T\right]^T$ around $\boldsymbol{z}_o$. The set $\boldsymbol{S}$ is constructed from an h-dense curve via domain densification and the unique optimal solution is given by

$$\boldsymbol{W}^*=\left(\sum_{\boldsymbol{z}_s\in\boldsymbol{S}}(\boldsymbol{f}(\boldsymbol{z}_o+\boldsymbol{z}_s)-\boldsymbol{f}(\boldsymbol{z}_o))\,\boldsymbol{z}_s^T\right)\left(\sum_{\boldsymbol{z}_s\in\boldsymbol{S}}\boldsymbol{z}_s\boldsymbol{z}_s^T\right)^{-1} \tag{7b}$$

as a multiple linear regression problem. The OLDD method expresses optimal linearization as a parameter identification problem, which makes it in high dimensions more tractable than using multiple integrals in the LSOL method. However, domain densification is unable to collect the points evenly over the region of interest, i.e., model accuracy is mediocre and yet dependent on the order of the states and controls.

6) Other Methods

In this letter, we are only interested in finding the best linear approximation to a nonlinear system with the same states and controls around some operating point, and thus the following linearization methods are out of scope:

- *Trajectory-Based Optimal Linearization* [11], [12], [13] optimizes the linear approximation to a particular solution of (1) from a given initial condition to the final state. That is, a trajectory with respect to time is linearized instead of a region of states and controls.
- *Feedback Linearization* [14], [15] method algebraically transforms a nonlinear system into a linear one with completely different dynamic interpretation. In addition, the number of dimensions usually increases if the nonlinear ODE is not an affine function of control inputs.
- *System Identification* [16], [17] technique estimates the model parameters by minimizing the error between the model output and the measured response, which is thus similar to the trajectory-based optimal linearization and only works for the stable equilibrium point at best.

B. Motivation & Contribution

The conventional AL method only captures the linear term in a Taylor series expansion of a nonlinear function around the operating point. Therefore, the resulting linear model usually works only for sufficiently small variations of states and controls. However, in some practical applications [2], [10], linear analysis and synthesis are desired to be applied to a much larger region of interest. The NL, SL, LSOL, and OLDD methods can be readily adapted to different ranges of states and controls by manipulating the parameter h, but they are subject to the trade-off between model accuracy and computational complexity.

Inspired by the previous work, this letter now presents a novel optimal linearization method via quadratic programming (OLQP), which features a great balance between model accuracy and computational complexity. Over specified ranges of states and controls around the operating point, the OLQP method finds the best linear approximation to a given nonlinear function by fitting the linear model to the data points uniformly sampled within the region, which makes it more accurate than the AL, NL, and OLDD methods in predicting the nonlinear behavior, while less computationally expensive than the SL and LSOL methods. In addition, the OLQP method consists with the AL method in that as the region of interest gets smaller, its solution is proved to converge to the AL solution. Therefore, AL can be viewed as a special case of OLQP and it can be used for Jacobian estimation. Lastly, it has shown that the OLQP method offers additional options in controller design by tuning its parameters.

The rest of this letter is organized as follows. Section II illustrates the proposed OLQP method. Section III benchmarks OLQP method against other linearization methods using the rigid-body aircraft model. Section IV investigates the control system behavior based on the OLQP linear model for the well-known cart-pole system. Section V concludes the letter with potential future directions.

II. Optimal Linearization via Quadratic Programming

This section details the proposed OLQP method. We start with the uniform data sampling strategy, then formulate linear approximation to an optimization problem, later convert it into a QP problem, derive its optimal solution, afterwards prove its convergence to the AL solution, and finally simplify the calculation for most practical systems of interest.

A. Data Sampling

We can observe in the linear model (2) that the state vector $\boldsymbol{x}$ and control input $\boldsymbol{u}$ are decoupled. Accordingly, let's first define the neighborhood of $(\boldsymbol{x}_o, \boldsymbol{u}_o)$ separately by

$$\boldsymbol{O} = \{(\boldsymbol{x}, \boldsymbol{u}) \mid \boldsymbol{u} = \boldsymbol{u}_o,\ \|\boldsymbol{x} - \boldsymbol{x}_o\|_\infty \leq h\}, \tag{8a}$$
$$\boldsymbol{Q} = \{(\boldsymbol{x}, \boldsymbol{u}) \mid \boldsymbol{x} = \boldsymbol{x}_o,\ \|\boldsymbol{u} - \boldsymbol{u}_o\|_\infty \leq h\}, \tag{8b}$$

where $\|\cdot\|_\infty$ is the infinity norm selecting the largest absolute value among all the entries of a vector. $\boldsymbol{O}$ and $\boldsymbol{Q}$ essentially capture the state and control space around $(\boldsymbol{x}_o, \boldsymbol{u}_o)$ bounded by a hypercube of edge $2h$. The edges for all the states and controls do not need to be set equal in general. Here we are just making it comparable with other methods.

To make the problem finite dimensional, one simple data sampling strategy is that we can uniformly collect points from $\boldsymbol{O}$ and $\boldsymbol{Q}$, which yields

$$\boldsymbol{R} = \{(\boldsymbol{x}, \boldsymbol{u}) \mid \boldsymbol{u} = \boldsymbol{u}_o,\ \boldsymbol{x}^{(i)} = \boldsymbol{x}_o^{(i)} - h + (j-1)\Delta h,\ i = 1, \ldots, n,\ j = 1, \ldots, N\}, \tag{9a}$$
$$\boldsymbol{S} = \{(\boldsymbol{x}, \boldsymbol{u}) \mid \boldsymbol{x} = \boldsymbol{x}_o,\ \boldsymbol{u}^{(i)} = \boldsymbol{u}_o^{(i)} - h + (j-1)\Delta h,\ i = 1, \ldots, m,\ j = 1, \ldots, N\}, \tag{9b}$$

respectively, where Δh is the resolution, $N = 2h/\Delta h + 1 \geq 2$ is thus the number of points on the edge, and $(\cdot)^{(i)}$ denotes for the ith entry. Note that there are a total number of N^n points sampled in $\boldsymbol{R}$ and N^m points in $\boldsymbol{S}$. Again, the resolutions do not need to be set equal in general.

Finally, two sets of data points are created. The first set is

$$\boldsymbol{T} = \{(\Delta\boldsymbol{x}, \Delta\boldsymbol{u}, \Delta\dot{\boldsymbol{x}}) \mid \Delta\boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}_o,\ \Delta\boldsymbol{u} = \boldsymbol{u} - \boldsymbol{u}_o,\ \Delta\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\boldsymbol{x}_o, \boldsymbol{u}_o),\ (\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{R}\}, \tag{10a}$$

and the second set is

$$\boldsymbol{V} = \{(\Delta\boldsymbol{x}, \Delta\boldsymbol{u}, \Delta\dot{\boldsymbol{x}}) \mid \Delta\boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}_o,\ \Delta\boldsymbol{u} = \boldsymbol{u} - \boldsymbol{u}_o,\ \Delta\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\boldsymbol{x}_o, \boldsymbol{u}_o),\ (\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{S}\}, \tag{10b}$$

which will eventually be utilized in the OLQP method.

B. Problem Formulation

Given the nonlinear system (1) with some operating point $(\boldsymbol{x}_o, \boldsymbol{u}_o)$ as well as the linear model (2) around that point, the goal is to determine the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ such that the difference between them is minimized over the region of interest. Define the difference

$$\boldsymbol{d} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\boldsymbol{x}_o, \boldsymbol{u}_o) - \boldsymbol{A}\Delta\boldsymbol{x} - \boldsymbol{B}\Delta\boldsymbol{u}. \tag{11}$$

For every element $(\Delta\boldsymbol{x}, \Delta\boldsymbol{u}, \Delta\dot{\boldsymbol{x}})_k \in \boldsymbol{T}$, $k = 1, \ldots, N^n$, the difference is reduced to

$$\boldsymbol{d}_k = \Delta\dot{\boldsymbol{x}}_k - \boldsymbol{A}\Delta\boldsymbol{x}_k, \tag{12a}$$

while for each element $(\Delta\boldsymbol{x}, \Delta\boldsymbol{u}, \Delta\dot{\boldsymbol{x}})_l \in \boldsymbol{V}$, $l = 1, \ldots, N^m$, the difference is reduced to

$$\boldsymbol{d}_l = \Delta\dot{\boldsymbol{x}}_l - \boldsymbol{B}\Delta\boldsymbol{u}_l. \tag{12b}$$

Using the squared Euclidean norm $\|\boldsymbol{d}\|_2^2 = \boldsymbol{d}^T\boldsymbol{d}$ as a measure of the difference, the minimization of the total difference J can be formulated as

$$\begin{aligned}\underset{\boldsymbol{A},\boldsymbol{B}}{\text{minimize}} \quad J &= \sum_{k=1}^{N^n} \|\boldsymbol{d}_k\|_2^2 + \sum_{l=1}^{N^m} \|\boldsymbol{d}_l\|_2^2 \\ &= \sum_{k=1}^{N^n} \|\Delta\dot{\boldsymbol{x}}_k - \boldsymbol{A}\Delta\boldsymbol{x}_k\|_2^2 + \sum_{l=1}^{N^m} \|\Delta\dot{\boldsymbol{x}}_l - \boldsymbol{B}\Delta\boldsymbol{u}_l\|_2^2,\end{aligned} \tag{13}$$

which can essentially be decoupled in terms of $\boldsymbol{A}$ and $\boldsymbol{B}$.

C. QP Formulation

A typical formulation for a mathematical QP problem can be written as follows:

$$\begin{aligned}\underset{\boldsymbol{z}}{\text{minimize}} \quad & \frac{1}{2}\boldsymbol{z}^T\boldsymbol{P}\boldsymbol{z} + \boldsymbol{q}^T\boldsymbol{z} + c \\ \text{subject to} \quad & \boldsymbol{G}\boldsymbol{z} \preceq \boldsymbol{w},\end{aligned} \tag{14}$$

where $c \in \mathbb{R}$, $\boldsymbol{z}, \boldsymbol{q} \in \mathbb{R}^n$, $\boldsymbol{w} \in \mathbb{R}^m$, $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ is symmetric positive semidefinite, and $\boldsymbol{G} \in \mathbb{R}^{m\times n}$ [18]. If the problem is unconstrained and $\boldsymbol{q} \in \mathcal{R}(\boldsymbol{P})$, it is simple enough to have the well-known analytical solution $\boldsymbol{z}^* = -\boldsymbol{P}^\dagger\boldsymbol{q}$, where $\boldsymbol{P}^\dagger$ is the pseudo-inverse of $\boldsymbol{P}$. We will now show (13) is essentially a QP problem.

Let's first rewrite (12a) as

$$\boldsymbol{d}_k = \Delta\dot{\boldsymbol{x}}_k - \Delta\boldsymbol{X}_k\boldsymbol{a}, \tag{15a}$$

where

$$\Delta\boldsymbol{X}_k = \begin{bmatrix} \Delta\boldsymbol{x}_k^T & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \Delta\boldsymbol{x}_k^T & \ddots & \vdots \\ \vdots & \ddots & \ddots & \boldsymbol{0} \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & \Delta\boldsymbol{x}_k^T \end{bmatrix} \in \mathbb{R}^{n\times n^2}, \tag{15b}$$
$$\boldsymbol{a} = \mathbf{vec}\left(\boldsymbol{A}^T\right) \in \mathbb{R}^{n^2}. \tag{15c}$$

That is, $\Delta\boldsymbol{X}_k$ is a block diagonal matrix with n blocks of $\Delta\boldsymbol{x}_k^T$ and $\boldsymbol{a}$ is the vectorization of the matrix $\boldsymbol{A}^T$. Similarly, (12b) can be rewritten as

$$\boldsymbol{d}_l = \Delta\dot{\boldsymbol{x}}_l - \Delta\boldsymbol{U}_l\boldsymbol{b}, \tag{16a}$$

where

$$\Delta\boldsymbol{U}_l = \begin{bmatrix} \Delta\boldsymbol{u}_l^T & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \Delta\boldsymbol{u}_l^T & \ddots & \vdots \\ \vdots & \ddots & \ddots & \boldsymbol{0} \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & \Delta\boldsymbol{u}_l^T \end{bmatrix} \in \mathbb{R}^{n\times mn}, \tag{16b}$$
$$\boldsymbol{b} = \mathbf{vec}\left(\boldsymbol{B}^T\right) \in \mathbb{R}^{mn}. \tag{16c}$$

Substituting (15a) and (16a) into the cost function J yields

$$J = \sum_{k=1}^{N^n} \|\Delta\dot{\boldsymbol{x}}_k - \Delta\boldsymbol{X}_k\boldsymbol{a}\|_2^2 + \sum_{l=1}^{N^m} \|\Delta\dot{\boldsymbol{x}}_l - \Delta\boldsymbol{U}_l\boldsymbol{b}\|_2^2$$

$$
\begin{aligned}
&= \boldsymbol{a}^T \underbrace{\left(\sum_{k=1}^{N^n} \Delta \boldsymbol{X}_k^T \Delta \boldsymbol{X}_k \right)}_{\frac{1}{2}\boldsymbol{P}_a} \boldsymbol{a} + \underbrace{\left(-2 \sum_{k=1}^{N^n} \Delta \dot{\boldsymbol{x}}_k^T \Delta \boldsymbol{X}_k \right)}_{\boldsymbol{q}_a^T} \boldsymbol{a} \\
&\quad + \boldsymbol{b}^T \underbrace{\left(\sum_{l=1}^{N^m} \Delta \boldsymbol{U}_l^T \Delta \boldsymbol{U}_l \right)}_{\frac{1}{2}\boldsymbol{P}_b} \boldsymbol{b} + \underbrace{\left(-2 \sum_{l=1}^{N^m} \dot{\boldsymbol{x}}_l^T \Delta \boldsymbol{U}_l \right)}_{\boldsymbol{q}_b^T} \boldsymbol{b} \\
&\quad + \underbrace{\sum_{k=1}^{N^n} \Delta \dot{\boldsymbol{x}}_k^T \Delta \dot{\boldsymbol{x}}_k + \sum_{l=1}^{N^m} \Delta \dot{\boldsymbol{x}}_l^T \Delta \dot{\boldsymbol{x}}_l}_{\tilde{c}} \\
&= \underbrace{\begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{bmatrix}^T}_{\tilde{\boldsymbol{z}}^T} \underbrace{\begin{bmatrix} \frac{1}{2}\boldsymbol{P}_a & \boldsymbol{0} \\ \boldsymbol{0} & \frac{1}{2}\boldsymbol{P}_b \end{bmatrix}}_{\frac{1}{2}\tilde{\boldsymbol{P}}} \underbrace{\begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{bmatrix}}_{\tilde{\boldsymbol{z}}} + \underbrace{\begin{bmatrix} \boldsymbol{q}_a \\ \boldsymbol{q}_b \end{bmatrix}^T}_{\tilde{\boldsymbol{q}}^T} \underbrace{\begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{bmatrix}}_{\tilde{\boldsymbol{z}}} + \tilde{c}.
\end{aligned} \tag{17}
$$

Therefore, (13) is equivalent to

$$\underset{\tilde{\boldsymbol{z}}}{\text{minimize}} \quad J = \frac{1}{2} \tilde{\boldsymbol{z}}^T \tilde{\boldsymbol{P}} \tilde{\boldsymbol{z}} + \tilde{\boldsymbol{q}}^T \tilde{\boldsymbol{z}} + \tilde{c}, \tag{18}$$

which is a QP problem defined by (14) with no constraint.

D. Optimal Solution

With the uniform data sampling strategy as suggested in Section II-A, it is guaranteed that $\boldsymbol{P}_a \succ \boldsymbol{0}$ and $\boldsymbol{P}_b \succ \boldsymbol{0}$, which is followed that $\tilde{\boldsymbol{P}} \succ \boldsymbol{0}$ as well. The positive definiteness of the matrices will be verified later in this subsection. As a result, the optimal solution is determined to be

$$\tilde{\boldsymbol{z}}^* = -\tilde{\boldsymbol{P}}^{-1} \tilde{\boldsymbol{q}} \tag{19}$$

for (18) with

$$\boldsymbol{a}^* = -\boldsymbol{P}_a^{-1} \boldsymbol{q}_a, \tag{20a}$$

$$\boldsymbol{b}^* = -\boldsymbol{P}_b^{-1} \boldsymbol{q}_b. \tag{20b}$$

The OLQP solution $\boldsymbol{A}^*$ and $\boldsymbol{B}^*$ can be further constructed from (15c) and (16c), respectively. Specifically, let's first write out (20a) in detail:

$$
\begin{aligned}
\boldsymbol{a}^* &= \left(\sum_{k=1}^{N^n} \Delta \boldsymbol{X}_k^T \Delta \boldsymbol{X}_k \right)^{-1} \left(\sum_{k=1}^{N^n} \Delta \boldsymbol{X}_k^T \Delta \dot{\boldsymbol{x}}_k \right) \\
&= \begin{bmatrix} \boldsymbol{\Sigma}_a^{-1} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Sigma}_a^{-1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \boldsymbol{0} \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{\Sigma}_a^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \\ \vdots \\ \boldsymbol{\mu}_n \end{bmatrix},
\end{aligned} \tag{21a}
$$

where

$$\boldsymbol{\Sigma}_a = \sum_{k=1}^{N^n} \Delta \boldsymbol{x}_k \Delta \boldsymbol{x}_k^T, \tag{21b}$$

$$\boldsymbol{\mu}_i = \sum_{k=1}^{N^n} \Delta \dot{\boldsymbol{x}}_k^{(i)} \Delta \boldsymbol{x}_k, \quad i = 1, \ldots, n. \tag{21c}$$

The matrix $\boldsymbol{A}^*$ is then constructed as

$$
\begin{aligned}
\boldsymbol{A}^* &= \begin{bmatrix} \boldsymbol{\mu}_1^T \boldsymbol{\Sigma}_a^{-1} \\ \vdots \\ \boldsymbol{\mu}_n^T \boldsymbol{\Sigma}_a^{-1} \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^{N^n} \Delta \dot{\boldsymbol{x}}_k^{(1)} \Delta \boldsymbol{x}_k^T \\ \vdots \\ \sum_{k=1}^{N^n} \Delta \dot{\boldsymbol{x}}_k^{(n)} \Delta \boldsymbol{x}_k^T \end{bmatrix} \boldsymbol{\Sigma}_a^{-1} \\
&= \left(\sum_{k=1}^{N^n} \begin{bmatrix} \Delta \dot{\boldsymbol{x}}_k^{(1)} \\ \vdots \\ \Delta \dot{\boldsymbol{x}}_k^{(n)} \end{bmatrix} \Delta \boldsymbol{x}_k^T \right) \boldsymbol{\Sigma}_a^{-1} \\
&= \left(\sum_{k=1}^{N^n} \Delta \dot{\boldsymbol{x}}_k \Delta \boldsymbol{x}_k^T \right) \boldsymbol{\Sigma}_a^{-1}.
\end{aligned} \tag{22}
$$

The matrix $\boldsymbol{\Sigma}_a$ of (21b) can actually be further simplified to

$$\boldsymbol{\Sigma}_a = \sum_{k=1}^{N^n} \textbf{diag} \left(\left(\Delta \boldsymbol{x}_k^{(1)} \right)^2, \ldots, \left(\Delta \boldsymbol{x}_k^{(n)} \right)^2 \right) \tag{23}$$

due to the symmetry of the uniform data sampling strategy, i.e., all off-diagonal entries cancel out. Moreover, based on (9a) and (10a) with $\Delta h = 2h/(N-1)$, each diagonal entry

$$
\begin{aligned}
&\sum_{k=1}^{N^n} \left(\Delta \boldsymbol{x}_k^{(i)} \right)^2 \\
=& N^{n-1} \sum_{j=1}^{N} \left(-h + (j-1)\, \Delta h \right)^2 \\
=& N^{n-1} \left(N h^2 - 2h \Delta h \frac{N(N-1)}{2} + \Delta h^2 \frac{N(N-1)(2N-1)}{6} \right) \\
=& \frac{h^2 N^n (N+1)}{3(N-1)}, \quad i = 1, \ldots, n,
\end{aligned} \tag{24}
$$

which implies $\boldsymbol{\Sigma}_a \succ \boldsymbol{0}$ when $N \geq 2$, and thus so is $\boldsymbol{P}_a \succ \boldsymbol{0}$. Substituting (23) with (24) into (22) yields a simplified form

$$\boldsymbol{A}^* = \frac{3(N-1)}{h^2 N^n (N+1)} \sum_{k=1}^{N^n} \Delta \dot{\boldsymbol{x}}_k \Delta \boldsymbol{x}_k^T \tag{25a}$$

and similarly,

$$\boldsymbol{B}^* = \frac{3(N-1)}{h^2 N^m (N+1)} \sum_{l=1}^{N^m} \Delta \dot{\boldsymbol{x}}_l \Delta \boldsymbol{u}_l^T, \tag{25b}$$

which essentially eliminates the matrix inversion.

E. Summary of OLQP Method

Given the nonlinear system (1) with some operating point $(\boldsymbol{x}_o, \boldsymbol{u}_o)$ as well as the linear model (2) around that point, the proposed OLQP method follows:

- Step 1: Determine the region of interest around $(\boldsymbol{x}_o, \boldsymbol{u}_o)$, $\boldsymbol{O}$ of (8a) and $\boldsymbol{Q}$ of (8b), with the parameter h for the size of the region.
- Step 2: Uniformly sample points within the region of interest to construct the sets $\boldsymbol{R}$ of (9a) and $\boldsymbol{S}$ of (9b), with the parameter N for the resolution.
- Step 3: Create two new sets $\boldsymbol{T}$ of (10a) and $\boldsymbol{V}$ of (10b) based on $\boldsymbol{R}$ and $\boldsymbol{S}$, respectively.
- Step 4: Compute the optimal linear model $\boldsymbol{A}^*$ of (25a) and $\boldsymbol{B}^*$ of (25b) using $\boldsymbol{T}$ and $\boldsymbol{V}$, respectively.

Note that for actual implementation, we don't need to create the sets $\boldsymbol{T}$ and $\boldsymbol{V}$ exactly. To reduce memory storage, once the contribution of one sample point is involved, we do not need to have it anymore.

F. Convergence to AL Solution

We will now prove the OLQP solution, the matrices $\boldsymbol{A}^*$ of (25a) and $\boldsymbol{B}^*$ of (25b), actually converge to the AL solution, $\boldsymbol{A}$ of (3a) and $\boldsymbol{B}$ of (3b), when h goes to zero.

Based on (22) with (23), any entry a^*_{ir} in $\boldsymbol{A}^*$ of row i and column r is determined to be

$$a^*_{ir} = \frac{\sum_{k=1}^{N^n} \Delta\dot{\boldsymbol{x}}_k^{(i)} \Delta\boldsymbol{x}_k^{(r)}}{\sum_{k=1}^{N^n} \left(\Delta\boldsymbol{x}_k^{(r)}\right)^2}. \tag{26}$$

Taking the limit to (26) as h goes to zero yields

$$\lim_{h\to 0} a^*_{ir} = \lim_{h\to 0} \frac{\sum_{k=1}^{N^n} \Delta\dot{\boldsymbol{x}}_k^{(i)} \Delta\boldsymbol{x}_k^{(r)}}{\sum_{k=1}^{N^n} \left(\Delta\boldsymbol{x}_k^{(r)}\right)^2}. \tag{27}$$

For any k and r with $\Delta\boldsymbol{x}_k^{(r)} \neq 0$ we have

$$\begin{aligned}\lim_{h\to 0} \frac{\Delta\dot{\boldsymbol{x}}_k^{(i)} \cancel{\Delta\boldsymbol{x}_k^{(r)}}}{\left(\Delta\boldsymbol{x}_k^{(r)}\right)^{\cancel{2}}} &= \lim_{h\to 0} \frac{\boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_o)^{(i)} - \boldsymbol{f}(\boldsymbol{x}_o, \boldsymbol{u}_o)^{(i)}}{\boldsymbol{x}_k^{(r)} - \boldsymbol{x}_o^{(r)}} \\ &= \left.\frac{\partial \boldsymbol{f}^{(i)}}{\partial \boldsymbol{x}^{(r)}}\right|_{(\boldsymbol{x},\boldsymbol{u})=(\boldsymbol{x}_o,\boldsymbol{u}_o)},\end{aligned} \tag{28}$$

which leads (27) to

$$\lim_{h\to 0} \frac{\sum_{k=1}^{N^n} \Delta\dot{\boldsymbol{x}}_k^{(i)} \Delta\boldsymbol{x}_k^{(r)}}{\sum_{k=1}^{N^n} \left(\Delta\boldsymbol{x}_k^{(r)}\right)^2} = \left.\frac{\partial \boldsymbol{f}^{(i)}}{\partial \boldsymbol{x}^{(r)}}\right|_{(\boldsymbol{x},\boldsymbol{u})=(\boldsymbol{x}_o,\boldsymbol{u}_o)} \tag{29}$$

as well, and the limit on the right-hand side is exactly the entry a_{ir} in $\boldsymbol{A}$ of (3a). Note that (29) holds based on (28) due to the following lemma:

Consider 4 sequences α_n, β_n, $\gamma_n \neq 0$, $\delta_n \neq 0$ with $n \in \mathbb{N}$. If $\lim_{n\to\infty} \alpha_n/\gamma_n = \lim_{n\to\infty} \beta_n/\delta_n = \rho$ as well as $\gamma_n + \delta_n \neq 0$, then $\lim_{n\to\infty} (\alpha_n + \beta_n) / (\gamma_n + \delta_n) = \rho$. The proof is trivial.

Essentially, (29) is equivalent to

$$\lim_{h\to 0} \boldsymbol{A}^* = \boldsymbol{A} \tag{30a}$$

of (3a) and similarly,

$$\lim_{h\to 0} \boldsymbol{B}^* = \boldsymbol{B} \tag{30b}$$

of (3b), which proves the convergence.

G. Simplification for Most Practical Systems

So far we have developed the OLQP method using a QP formulation and proved its convergence to the AL solution, which is sufficient to be implemented on any general nonlinear system. Nevertheless, the proposed method can be further simplified for most practical systems of interest.

Consider the equations of motion taking the form:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{F}\boldsymbol{u}, \tag{31}$$

where $\boldsymbol{q}$ is the vector of generalized coordinates, $\boldsymbol{M}(\boldsymbol{q})$ stands for the inertia matrix, the vector $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ captures the Coriolis, centrifugal, and gravitational forces, and the matrix $\boldsymbol{F}$ defines how the control input $\boldsymbol{u}$ enters the model. We can further convert (31) into its state-space form as (1), where the state vector $\boldsymbol{x} = \left[\boldsymbol{q}^T, \dot{\boldsymbol{q}}^T\right]^T$ and

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{M}(\boldsymbol{q})^{-1} \left(\boldsymbol{F}\boldsymbol{u} - \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\right) \end{bmatrix}. \tag{32}$$

We observe that $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u})$ is an affine function of the control input $\boldsymbol{u}$. For any well-defined operating point $(\boldsymbol{x}_o, \boldsymbol{u}_o) = \left(\left[\boldsymbol{q}_o^T, \dot{\boldsymbol{q}}_o^T\right]^T, \boldsymbol{u}_o\right)$, the AL solution is thus structured as

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix} \quad \text{and} \quad \boldsymbol{B} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{B}_2 \end{bmatrix}, \tag{33a}$$

where

$$\boldsymbol{A}_{21} = \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{q}}\right|_{(\boldsymbol{x},\boldsymbol{u})=(\boldsymbol{x}_o,\boldsymbol{u}_o)}, \quad \boldsymbol{A}_{22} = \left.\frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{q}}}\right|_{(\boldsymbol{x},\boldsymbol{u})=(\boldsymbol{x}_o,\boldsymbol{u}_o)}, \tag{33b}$$

$$\boldsymbol{B}_2 = \boldsymbol{M}(\boldsymbol{q}_o)^{-1}\boldsymbol{F}, \tag{33c}$$

and $\boldsymbol{I}$ is the identity matrix. Since the matrix $\boldsymbol{B}$ is already in closed form, there is no need to consider it in the QP anymore, i.e., simply set $\boldsymbol{B}^* = \boldsymbol{B}$ of (33a). In addition, the system can be reduced to

$$\ddot{\boldsymbol{q}} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}), \tag{34a}$$

with the linear model around the operating point

$$\Delta\ddot{\boldsymbol{q}} = \begin{bmatrix} \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{q} \\ \Delta\dot{\boldsymbol{q}} \end{bmatrix} + \boldsymbol{B}_2 \Delta\boldsymbol{u}, \tag{34b}$$

where $\Delta\boldsymbol{q} = \boldsymbol{q} - \boldsymbol{q}_o$. The proposed OLQP method still works here since there is no strict requirement for $\boldsymbol{A}$ to be a square matrix. Once $[\boldsymbol{A}^*_{21} \quad \boldsymbol{A}^*_{22}]$ is computed, the OLQP solution $\boldsymbol{A}^*$ can be constructed from (33a) with $\boldsymbol{B}^* = \boldsymbol{B}$.

III. Example of Rigid-Body Aircraft Model

In this section, the rigid-body aircraft model is studied for the longitudinal motion. The system is linearized around the equilibrium condition and results are compared among the linearization methods.

A. Modeling

The following equations describe a rigid-body aircraft in the longitudinal direction [2]:

$$m\dot{V} = T\cos\alpha - D - mg\sin\gamma, \tag{35a}$$

$$mV\dot{\gamma} = T\sin\alpha + L - mg\cos\gamma, \tag{35b}$$

$$\dot{\alpha} = q - \dot{\gamma}, \tag{35c}$$

$$I_{yy}\dot{q} = M - x_c L\cos\alpha - x_c D\sin\alpha, \tag{35d}$$

where the state vector $\boldsymbol{x} = [V, \gamma, \alpha, q]^T$, V is the airspeed, γ is the flight path angle, α is the angle of attack, q is the pitch rate; the control input $\boldsymbol{u} = [T, \delta_c]^T$, T is the thrust, δ_c is the canard deflection; the variable $L = 0.5C_L\rho V^2 S$ is the lift force, $D = 0.5C_D\rho V^2 S$ is the drag force, $M = 0.5C_M\rho V^2 S\bar{c}$ is the pitching moment; the parameter m is the aircraft mass, g is the gravitational acceleration, I_{yy} is the moment of inertia about the y-axis, ρ is the air density, S is the reference area, $\bar{c}$ is the mean aerodynamic chord, $x_c = -0.0465\bar{c}$ is the distance between the aircraft aerodynamic center and the center of mass; for $\alpha > 0$, the aerodynamic coefficients are given by

$C_L = \sum A_i\alpha^i + (1/2.235)\,(\delta_c+\alpha)\sum C_i\alpha^i$, $C_D = \sum B_i\alpha^i$, $C_M = (\delta_c+\alpha)\sum C_i\alpha^i - 1.5q\bar{c}/V$, $i = 0,\ldots,5$. Table I summarizes all the parameters. The equilibrium point of interest is given by $(\boldsymbol{x}_e, \boldsymbol{u}_e) = \left([100, 0, 0.0754, 0]^T, [12781, -0.124]^T\right)$.

TABLE I
AIRCRAFT PARAMETERS

$m = 10,617$ kg $\rho = 1.225$ kg/m^3	$g = 9.81$ m/s^2 $S = 57.7$ m^2	$I_{yy} = 77,095$ kg·m^2 $\bar{c} = 4.4$ m
$A_0 = 0.00933$	$B_0 = 0.02323$	$C_0 = 0.28933$
$A_1 = 3.58977$	$B_1 = 0.03809$	$C_1 = -0.15349$
$A_2 = 4.40752$	$B_2 = 1.64156$	$C_2 = 0.75441$
$A_3 = -16.98693$	$B_3 = 1.65442$	$C_3 = -1.50691$
$A_4 = 13.38188$	$B_4 = -2.30301$	$C_4 = 1.07489$
$A_5 = -3.34885$	$B_5 = 0.55977$	$C_5 = -0.25771$

B. Linearization & Jacobian Estimation

The AL solution, or the Jacobian, is determined to be

$$\boldsymbol{A} = \begin{bmatrix} -2.401\times10^{-2} & -9.81 & -10.406 & 0 \\ 1.944\times10^{-3} & 0 & 1.382 & 0 \\ -1.944\times10^{-3} & 0 & -1.382 & 1 \\ 0 & 0 & 9.622 & -1.331 \end{bmatrix}, \tag{36a}$$

$$\boldsymbol{B} = \begin{bmatrix} 9.392\times10^{-5} & 0 \\ 7.093\times10^{-8} & 4.192\times10^{-2} \\ -7.093\times10^{-8} & -4.192\times10^{-2} \\ 0 & 5.795 \end{bmatrix}, \tag{36b}$$

around the equilibrium condition, with eigenvalues $\lambda(\boldsymbol{A}) = \{-4.460, 1.755, -0.0161 \pm 0.152j\}$. We are first interested in how well the proposed OLQP method can estimate the AL solution. Let's define the difference between them

$$\boldsymbol{D} = \left[\begin{array}{cc} \boldsymbol{A} & \boldsymbol{B} \end{array}\right] - \left[\begin{array}{cc} \boldsymbol{A}^* & \boldsymbol{B}^* \end{array}\right] \tag{37}$$

with the Frobenius norm $\|\boldsymbol{D}\|_F = \sqrt{\mathrm{tr}\,(\boldsymbol{D}^T\boldsymbol{D})}$ as a measure of the difference. Note that $\boldsymbol{A}^*$ and $\boldsymbol{B}^*$ are actually functions of the parameters h and N, and thus so is $\boldsymbol{D}$, i.e., $\boldsymbol{A}^*(h,N)$, $\boldsymbol{B}^*(h,N)$, and $\boldsymbol{D}(h,N)$. Fig. 1 shows the difference function $\|\boldsymbol{D}(h,N)\|_F$ for the aircraft example. It is verified that the OLQP solution approaches the AL solution as h goes to zero, which is proved in Section II-F. In addition, we can see that when h is sufficiently small, increasing N, i.e., improving the resolution of data sampling, does not help too much in enhancing the accuracy.

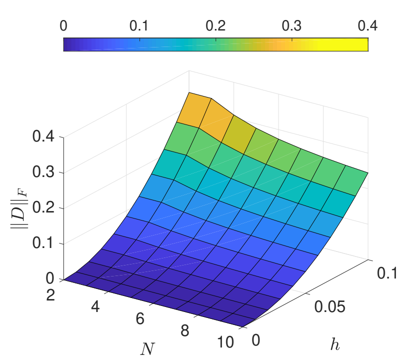

Fig. 1. Plot of $\|\boldsymbol{D}\|_F$ as a function of h and N. The difference $\|\boldsymbol{D}\|_F$ vanishes when h approaches zero. Moreover, when h is sufficiently small, N does not contribute too much.

Fig. 2 compares how close other linearization solutions are to the AL solution. The FDA and BDA solutions are very close to each other while their accuracy is terrible even for small value of h. The CDA, LSOL, OLDD and OLQP solutions are all very close to the AL solution even for large value of h wherein the LSOL solution is the best at estimating the AL solution for the aircraft example. However, it takes around 40 seconds for the LSOL method according to Table II, which further compares the average running time of each method for computing the linear model 100 times on an Intel Core i7-7700HQ@2.80 GHz quad-core laptop. It is clear that the proposed OLQP method is a great choice for Jacobian estimation in consideration of both accuracy and computational complexity. Note that the OLDD method depends on the order of the states and controls so only two arbitrary cases are considered; MATLAB's *dlgradient* function is used for AD calculation; the SL method is not involved because it cannot handle the problem dimension and nonlinearity.

TABLE II
AVERAGE RUNNING TIME

OLQP ($N=2$)	7.9 ms	FDA	7.3 ms	LSOL	40.4 s
OLQP ($N=6$)	15.6 ms	BDA	7.3 ms	OLDD	18.6 ms
QLQP ($N=10$)	56.6 ms	CDA	7.7 ms	AL (AD)	265 ms

C. Model Accuracy

We are also interested in how close the OLQP linear model is to the original nonlinear system in Section III-A, i.e., how well the OLQP method can predict the nonlinear behavior, compared against other linearization methods. Since the equilibrium point of interest is unstable, the following linear quadratic regulator (LQR) controller is designed for all the systems, which is based on the AL linear model (36a) and (36b), with $\boldsymbol{Q} = \mathbf{diag}(10^{-4}, 1, 10, 1)$ and $\boldsymbol{R} = \mathbf{diag}(1, 10)$. Numerical simulations are performed for three different values of $\Delta\alpha$ with initial condition $\boldsymbol{x}(0) = \boldsymbol{x}_e + [0, 0, \Delta\alpha, 0]^T$, as shown in Fig. 3. As $\Delta\alpha$ increases, the prediction of AL model becomes worse, as expected. On the other hand, the LSOL and OLQP models achieve an overall better description of the nonlinear system than other methods. Specifically, when $\Delta\alpha = 0.1$ rad, the response of AL model is almost identical with the nonlinear model; when $\Delta\alpha = 0.4$ rad, they are on about the same level of closeness; when $\Delta\alpha = 0.6$ rad,

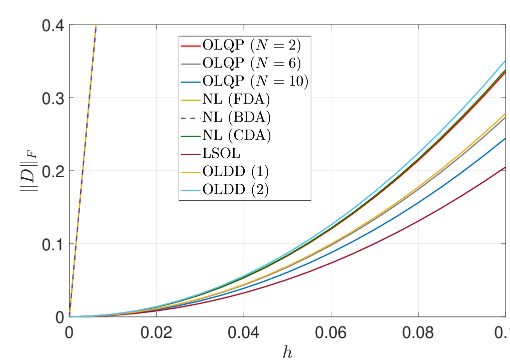

Fig. 2. Comparison of Jacobian estimation among the proposed OLQP, NL, LSOL, and OLDD methods with varying h. The OLQP solution with $N = 10$ is the second closest for the aircraft example.

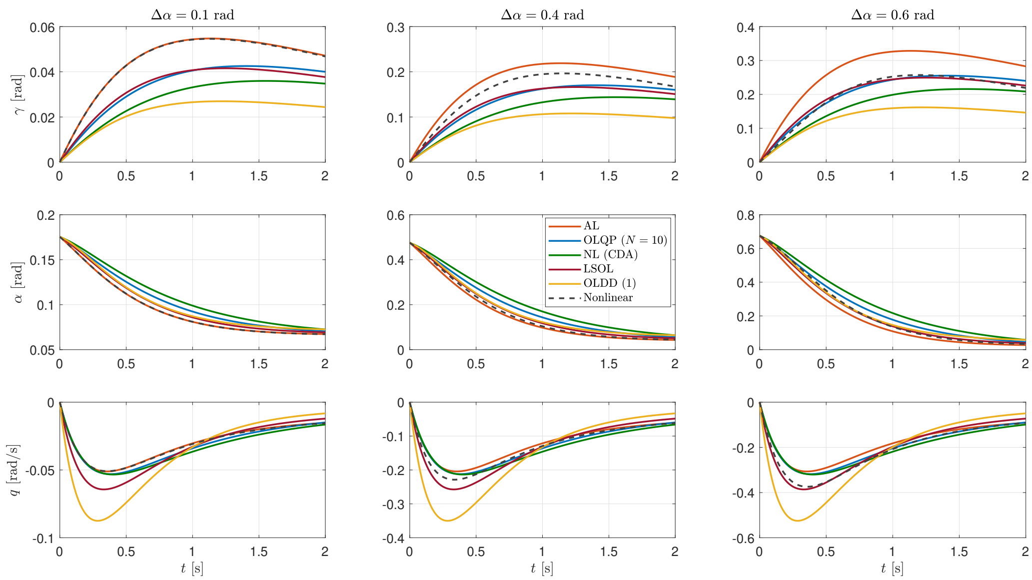

Fig. 3. Comparison of model accuracy among AL, OLQP, NL, LSOL, and OLDD methods with $h = 0.4$. Numerical simulations are performed for all the models with three different initial conditions. The LSOL and OLQP models achieve an overall better description of the nonlinear system than other methods.

the LSOL and OLQP models are closer. This makes sense because the AL model is only valid for a small region, while the LSOL and OLQP models capture more nonlinear dynamic information over a specified larger region in an optimal manner. For this example, the parameter h is fixed equal to 0.4 and the OLQP model works better when the states are far away from the equilibrium. Imaging if we have a varying h and thus a varying OLQP model which depends on the location of states and controls, a better description of nonlinear behavior can be expected, which might be worth working on in the future.

IV. Example of Cart-Pole System

In this section, the well-known cart-pole system is investigated. Linearization around the unstable equilibrium point is carried out with both the AL method and proposed OLQP method. LQR controller is further designed based on the two linear models. Corresponding results are compared.

A. Modeling

The equations of motion take the form as (31), where

$$\boldsymbol{M}(\boldsymbol{q}) = \begin{bmatrix} m_1 + m_2 & m_2 l \cos\theta \\ m_2 l \cos\theta & m_2 l^2 \end{bmatrix}, \tag{38a}$$

$$\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \begin{bmatrix} -m_2 l \dot{\theta}^2 \sin\theta \\ -m_2 g l \sin\theta \end{bmatrix}, \ \boldsymbol{F}\boldsymbol{u} = \begin{bmatrix} u \\ 0 \end{bmatrix}, \tag{38b}$$

$\boldsymbol{q} = [x, \theta]^T$ is the vector of generalized coordinates, x is the position of the cart, θ is the angle of the pole, and the control force is given by u; the masses of the cart and pole are given by m_1 and m_2, respectively, and the length of the pole and acceleration due to gravity are l and g, respectively. We can further convert it into its state-space form as (1), where the state vector $\boldsymbol{x} = [\theta, \dot{x}, \dot{\theta}]^T$. Note that the position x actually does not contribute to the dynamics at all, and thus the total number of states is reduced to 3. The unstable equilibrium point of interest is given by $(\boldsymbol{x}_e, u_e) = \big([0, 0, 0]^T, 0\big)$.

B. Linearization

Given the operating point $(\boldsymbol{x}_e, u_e)$ and the parameters $m_1 = m_2 = 1$ kg, $l = 10$ m, $g = 1$ m/s^2, the AL solution around this point is determined to be

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0.2 & 0 & 0 \end{bmatrix} \quad \text{and} \quad \boldsymbol{B} = \begin{bmatrix} 0 \\ 1 \\ -0.1 \end{bmatrix}, \tag{39}$$

with eigenvalues $\lambda(\boldsymbol{A}) = \{0, \pm\sqrt{5}/5\}$. The OLQP solution is computed as $\boldsymbol{A}^*(h, N)$ and $\boldsymbol{B}^* = \boldsymbol{B}$ from Section II-G.

C. Control

Since $(\boldsymbol{A}, \boldsymbol{B})$ is verified to be controllable and assume all the states can be measured directly, LQR controller can be designed based on the linear models to stabilize the pendulum around the upright configuration. Note that the optimal control $u = -\boldsymbol{K}\boldsymbol{x}$, where the gain matrix $\boldsymbol{K}$ is computed by MATLAB's *lqr* command with the weighting matrices $\boldsymbol{Q} = \mathbf{diag}(1, 1, 1)$ and $R = 1$.

To evaluate the controller performance based on the OLQP linear model, two aspects are investigated. On the one hand, the settling time t_s for the angle θ, i.e., $|\theta(t)| \leq \pi/1800$ for all $t \geq t_s$ with a fixed initial condition $\boldsymbol{x}(0) = [\pi/9, 0, 0]^T$, is considered as the main property quantifying the system transient response; on the other hand, the maximum feasible value for the initial angle $\theta(0)_{\max}$, i.e., starting from which the pendulum can still be stabilized, is regarded as the main property reflecting the system robustness.

Fig. 4 shows the simulation results of the OLQP method with varying h and fixed $N = 5$. First, it is clear that for small value of h, the LQR controllers of the AL and OLQP methods behave similarly to each other, due to the two linear models close to each other. When h becomes larger, as may be expected, the two linear models get further away from each other. Specifically for the OLQP method, it essentially captures the most information of the original nonlinear system

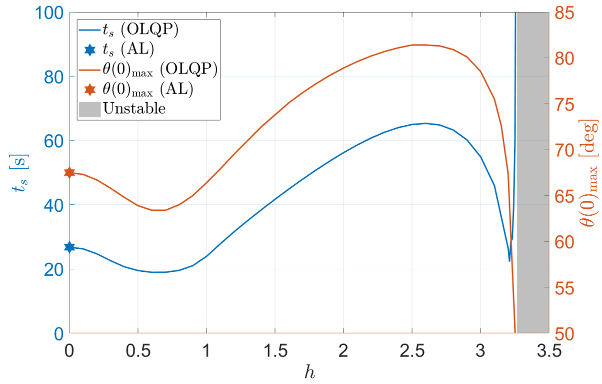

Fig. 4. Simulation results of the settling time t_s and maximum initial angle $\theta(0)_{\max}$ with varying h and fixed $N = 5$. The closed-loop system is unstable when $h > 3.265$, i.e., the OLQP linear model fails to describe the nonlinear system accurately.

over some region of states and controls around the operating point, and determines the best linear approximation over the entire region in an average sense. The effect, for the cart-pole system, is a trade-off between the system transient response and robustness when that region gets larger, i.e., h gets larger. The closed-loop system starts with an improvement in the transient response as t_s decreases and yet a deterioration in the robustness as $\theta(0)_{\max}$ decreases as well. Later, the transient response gets worse while the robustness turns better. Finally, the system becomes unstable when that region gets too further away from the operating point, i.e., the OLQP linear model fails to accurately describe the original nonlinear system.

To sum up, the effect that changing the parameter h in the proposed OLQP method will result in a trade-off between the system transient response and robustness, indicates that the OLQP method actually offers extra options in designing the controller. It is also reasonable to involve the parameter N when tuning the controller since it will affect the OLQP solution as well.

V. Conclusion

In this letter, an optimal linearization method via quadratic programming (OLQP) is presented. It starts with the uniform data sampling over a specified region of states and controls around the operating point based on the nonlinear ordinary differential equation (ODE). The best linear model that fits to these sample points is then found via a quadratic programming (QP) formulation.

Compared with other existing linearization methods, the proposed OLQP method features a great balance between model accuracy and computational complexity. The OLQP method is also consistent with the analytical linearization (AL) method in that as the region of interest becomes smaller around the operating point, the OLQP solution is proved to converge to the AL solution. Therefore, AL can be viewed as a special case of OLQP and it can be used for Jacobian estimation. Moreover, the OLQP method offers additional options in controller design since the change in its parameters has shown a trade-off between the closed-loop system transient response and robustness. Last but not least, the OLQP method is applicable to a much larger class of nonlinear functions than the AL method since its process only involves summations instead of derivatives, i.e., the function does not even need to be continuously differentiable at the point of interest.

Many interesting research topics have crossed our mind based on the OLQP method. For example, how to better describe the nonlinear system with a state-varying OLQP linear model, as mentioned in the end of Section III-C. Another one is that conventional trajectory stabilizer requires a linear time-varying approximation of the system around the trajectory. That is, the system is almost linearized at each point along the entire trajectory. What if we wisely split the state and control space into several regions and apply the OLQP method for each region? How can we do that and how will the system behave differently? We are looking forward to applying the OLQP method to more examples in the future.

References

[1] D. Arrowsmith and C. Place, *Dynamical Systems: Differential Equations, Maps, and Chaotic Behaviour*. Chapman Hall/CRC Mathematics Series, Taylor & Francis, 1992.

[2] Y. Zhao, “Least squares optimal linearization,” *Journal of Guidance Control Dynamics*, vol. 17, pp. 990–997, Sep 1994.

[3] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, “Automatic differentiation in machine learning: a survey,” *Journal of Machine Learning Research*, vol. 18, no. 153, pp. 1–43, 2018.

[4] A. Antoniou and W.-S. Lu, *Practical Optimization: Algorithms and Engineering Applications*. Springer Publishing Company, Incorporated, 1st ed., 2007.

[5] J. Persson and L. Soder, “Comparison of three linearization methods,” in *Proceedings 16th Power Systems Computation Conference (PSCC)*, pp. 1478–1484, 2008.

[6] D. REARDON and W. E. LEITHEAD, “Statistical linearization: a comparative study,” *International Journal of Control*, vol. 52, no. 5, pp. 1083–1105, 1990.

[7] C.-A. Lin and V. H. L. Cheng, “Statistical linearization for multi-input/multi-output nonlinearities,” *Journal of Guidance, Control, and Dynamics*, vol. 14, no. 6, pp. 1315–1318, 1991.

[8] Y. Zhao, “Concepts of Least Squares Optimal Linearization,” in *Proceedings of the AIAA Guidance & Control Conference*, pp. 1186–1195, 1992.

[9] A. V. Kamyad, H. H. Mehne, and A. H. Borzabadi, “The best linear approximation for nonlinear systems,” *Appl. Math. Comput.*, vol. 167, p. 1041–1061, Aug. 2005.

[10] K. Bousson, J. Silva, and A. Cardoso, “Optimal linearization of nonlinear control models,” in *44th AIAA Aerospace Sciences Meeting and Exhibit*, January 2006.

[11] T. Benouaz and O. Arino, “Least square approximation of a nonlinear ordinary differential equation,” *Computers & Mathematics with Applications*, vol. 31, no. 8, pp. 69 – 84, 1996.

[12] F. Belkhouche, “Trajectory-based optimal linearization for nonlinear autonomous vector fields,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 52, pp. 127–138, Jan 2005.

[13] P. D. Orzechowski, “Optimal linearization method of nonlinear state equation as an example of model of ion recombination in electrical arc,” in *Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments IV* (R. S. Romaniuk, ed.), vol. 6159, pp. 992 – 997, International Society for Optics and Photonics, SPIE, 2006.

[14] R. W. Brockett, “Feedback invariants for nonlinear systems,” *IFAC Proceedings Volumes*, vol. 11, no. 1, pp. 1115–1120, 1978.

[15] J. Slotine and W. Li, *Applied Nonlinear Control*. Prentice-Hall International Editions, Prentice-Hall, 1991.

[16] L. Ljung, *System Identification (2nd Ed.): Theory for the User*. USA: Prentice Hall PTR, 1999.

[17] K. J. Keesman, *System Identification - An Introduction*. Springer, London, 2011.

[18] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.