On the Uses of Linear-Quadratic Methods in Solving Nonlinear Dynamic Optimization Problems with Direct Transcription

IMECE2020-23885

Daniel R. Herber\textsuperscript{1} \quad Athul K. Sundarrajan\textsuperscript{2}

\textsuperscript{1} Assistant Professor
\textsuperscript{2} Graduate Student, \textbf{Presenting Author}

Colorado State University, Department of Systems Engineering
\texttt{daniel.herber@colostate.edu}
\texttt{Athul.Sundarrajan@colostate.edu}
Introduction
Introduction

- Dynamic optimization (DO) and optimal control are the optimization of systems of equations with time-varying behavior [Betts 2010]
- Because of time dependent elements, the solution must satisfy the infinite points in a given time horizon $t \in [t_o, t_f]$
- There are two family of methods that are used to solve DO problems: direct numerical methods and indirect methods
- Indirect methods use optimality conditions to find the optimal dynamics [Bryson and Ho 1975]
- Direct methods transcribe the infinite-dimensional problem to a finite-dimensional optimization problem [Rao 2010]
- Here we focus on a direct method solution strategy known as Direct Transcription (DT) [Betts 2010; Rao 2010]
→ Problem Description

\[
\min_{x=[u,\xi,p]} \quad o = \int_{t_0}^{t_f} \ell(t, u, \xi, p) \, dt + m(p, \xi_0, \xi_f)
\]  
(1a)

subject to:

\[
\dot{\xi}(t) - f(t, u, \xi, p) = 0
\]  
(1b)

\[
h(t, u, \xi, p, \xi_0, \xi_f) = 0
\]  
(1c)

\[
g(t, u, \xi, p, \xi_0, \xi_f) \leq 0
\]  
(1d)

where:

\[
\xi_0 = \xi(t_0), \quad \xi_f = \xi(t_f)
\]  
(1e)

• Consider the elements in a general **Nonlinear Dynamic Optimization (NLDO)** problem
  • The optimization variables are controls \(u\), states \(\xi\), and parameters \(p\)
  • The states \(\xi(t)\) and open-loop controls \(u(t)\) are time varying quantities
  • \(o(\cdot)\) is the objective term composed of the Lagrange term \(\ell(\cdot)\) and the Mayer term \(m(\cdot)\)
  • \(f(\cdot)\) is the explicit first order state derivative function
  • \(h(\cdot)\) and \(g(\cdot)\) are the additional equality and inequality constraints
Direct Transcription

\[
\min_{x=[\text{vec}(U), \text{vec}(\Xi), p]} v^{DT} (t, U, \Xi, p, \Xi_1, \Xi_n) \quad (2a)
\]

subject to:

\[
\begin{align*}
\mathbf{h}^{DT} &= \left[ h^{DT}_\zeta (t, U, \Xi, p) \\
&\quad h^{DT}_h (t, U, \Xi, p, \Xi_1, \Xi_n) \right] = 0 \quad (2b)
\end{align*}
\]

\[
\mathbf{g}^{DT} (t, U, \Xi, p, \Xi_1, \Xi_n) \leq 0 \quad (2c)
\]

- DT methods transcribe the infinite-dimensional DO problem to a finite dimensional problem that can be solved with standard NLP solvers [Betts 2010]
  - The **objective function** in the continuous problem is approximated using numerical **quadrature** methods
  - The **dynamic** state equation is transcribed through different **collocation** methods
  - Path constraints are enforced at each time point
  - Initial and final value constraints can be expressed as they were
Linear-Quadratic Dynamic Optimization (LQDO) is a special case of the (NLDO) with a quadratic objective function subject to linear constraints [Herber 2017]. This implies linear dynamic models. Through DT, the given LQDO problem is converted into a large sparse quadratic programming (QP). Many efficient tools have been developed to solve the QPs.
Advantages of LQDO Problems

\[
\begin{aligned}
\min_x & \quad f'x + \frac{1}{2}x'Hx \\
\text{subject to:} & \quad A_hx - b_h = 0 \\
& \quad A_gx - b_g \leq 0
\end{aligned}
\]  

(4a)

(4b)

(4c)

- Structure of a given optimization problem can be used to increase the efficiency and robustness of the techniques used to solve the problem.

- If the objective function of the QP problem is convex and the feasible set is nonempty, then there exists a **global solution** to the problem.
Research Questions

• Many DO problems have both LQ and nonlinear elements yet there are not many studies that look at efficient implementation of LQ elements
• There are efficient tools for LQDO problems
• We know that the problem structure contributes to their efficiency

1 So how can we incorporate these elements to solve NLDO problems?
2 If a NLDO has both LQ and NL elements, can we take advantage of this structure?
3 Even if there are no LQ elements, is there a way to still use LQ elements and solve the original NLDO problem?
Solving NLDO using LQDO Elements
Direct Incorporation of LQ Elements

- For a given NLDO problem, we can treat the LQ elements differently than the general nonlinear elements.
- This can be done by manually specifying which problem elements are LQ or automated methods.

\[
\begin{align*}
\min_{x=[u,\xi,p]} & \quad \int_{t_0}^{t_f} \left[ \ell^{OL} (\cdot) + \ell^{NL} (\cdot) \right] dt + m^{OL} (\cdot) + m^{NL} (\cdot) \\
\text{subject to:} & \quad \dot{\xi}(t) - f (\cdot) = \begin{bmatrix} \dot{\xi}^{OL}(t) - f^{OL} (\cdot) \\ \dot{\xi}^{NL}(t) - f^{NL} (\cdot) \end{bmatrix} = 0 \\
& \quad h (\cdot) = \begin{bmatrix} h^{OL} (\cdot) \\ h^{NL} (\cdot) \end{bmatrix} = 0 \\
& \quad g (\cdot) = \begin{bmatrix} g^{OL} (\cdot) \\ g^{NL} (\cdot) \end{bmatrix} \leq 0
\end{align*}
\]
Direct Incorporation of LQ Elements (continued)

There are several potential advantages for using this strategy:

- NLP solvers use first and second derivative information
- For LQ elements, the hessian is zero, and the jacobian is constant
- Identifying and pre-computing these values for LQ elements will help reduce the problem complexity and cost

\[
\mathcal{L}(\mathbf{x}, \lambda, \sigma) = v(\mathbf{x}) + \lambda \cdot \mathbf{h}(\mathbf{x}) + \sigma \cdot \mathbf{g}(\mathbf{x})
\]  \hspace{1cm} (6)

\[
H^L_x(\mathbf{x}) = \sum_j \lambda^O_L j \cdot \left[ H^O_L(j) (\mathbf{x}) \right] + \sum_j \lambda^N_L j \cdot \left[ H^N_L(j) (\mathbf{x}) \right] + \cdots
\]  \hspace{1cm} (7)
Two-Level Methods

- Two-level optimization methods partition the problem variables into upper- and lower-level variables $x^u$ and $x^l$ [Bard 1988]

$$\min_{x^u} o(x^u, x^l)$$

subject to: $x^u \in \Omega$ (8a)

$x^l \in L(x^u)$ (8b)

- For a candidate variable $x^u$ provided by the upper-level problem, the lower-level problem involves finding the optimal value for $x^l$

- Here we focus on NLDO problems where fixing the values of $p$ yields a lower-level LQDO problem that solves for $\xi(t)$ and $u(t)$ [Herber and Allison 2019]
Quasilinearization

- Quasilinearization is a method that solves NLDO problems by expanding them about a reference trajectory $\bar{x}$ using a Taylor series and sequentially solving QPs [Jaddu 2002]
  - Nonlinear constraints are expanded till the linear:
    \[ e(x) \approx e^L(x, \bar{x}) = e(\bar{x}) + J^e_x(\bar{x}) [x - \bar{x}] \] (9)
  - Nonlinear objective terms are expanded till quadratic:
    \[ e(x) \approx e^O(x, \bar{x}) = e(\bar{x}) + [J^e_x(\bar{x})] [x - \bar{x}] + \frac{1}{2} [x - \bar{x}] H^e_x(\bar{x}) [x - \bar{x}] \] (10)
  - The reference LQDO problem is solved and the solution from the previous iteration used as the reference trajectory for the next iteration
  - This process continues until the difference between successive iterations is small
3

Case Studies
All the case studies included in the paper are solved using a free and open-source MATLAB-based tool, *DTQP* https://github.com/danielrherber/dt-qp-project

- In this problem, we want to optimally transfer the states of a container crane subject to linear path constraints
- All parts except for $\dot{\xi}_6$ are Linear Quadratic elements

\[
\begin{align*}
\min_{u, \xi} & \quad \frac{1}{2} \int_0^{t_f} \left[ \xi_3^2 + \xi_6^2 + \rho \left( u_1^2 + u_2^2 \right) \right] dt \\
\text{subject to:} & \quad \dot{\xi} = \begin{bmatrix} \xi_4 \\ \xi_5 \\ \xi_6 \\ u_1 + c_4 \xi_3 \\ u_2 \\ - \left[ u_1 + c_5 \xi_3 + 2\xi_5 \xi_6 \right] / \xi_2 \end{bmatrix} \\
\xi(0) & = (0, 22, 0, 0, -1, 0) \\
\xi(t_f) & = (10, 14, 0, 2.5, 0, 0) \\
-c_6 & \leq \xi_4 \leq c_6, \quad c_7 \leq \xi_5 \leq c_7 \\
-c_1 & \leq u_1 \leq c_1, \quad c_2 \leq u_2 \leq c_3
\end{align*}
\]
Container Crane Results

- The problem was tested using **quasilinearization** and the nonlinear optimization was solved using an **interior point (IP)** algorithm.

### Quasilinearization Results for Container Crane

![Quasilinearization Results](image-url)
Container Crane Results (continued)

- **Quasilinearization was generally faster**
- When LQ problem elements were directly incorporated in IP method, the solution time was 15 – 165% faster

<table>
<thead>
<tr>
<th>Method</th>
<th>$v$</th>
<th>Iter.</th>
<th>$T_{sym}$</th>
<th>$T_{int}$</th>
<th>$T_{opt}$</th>
<th>$T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>QLIN-TR20</td>
<td>0.0380</td>
<td>3</td>
<td>0.57</td>
<td>0.007</td>
<td>0.03</td>
<td>0.03</td>
</tr>
<tr>
<td>IP-SD-OLQ-TR20</td>
<td>0.0380</td>
<td>20</td>
<td>1.42</td>
<td>0.002</td>
<td>0.08</td>
<td>0.08</td>
</tr>
<tr>
<td>IP-SD-NL-TR20</td>
<td>0.0380</td>
<td>20</td>
<td>2.17</td>
<td>0.003</td>
<td>0.16</td>
<td>0.16</td>
</tr>
<tr>
<td>IP-FD-OLQ-TR20</td>
<td>0.0380</td>
<td>22</td>
<td>0.00</td>
<td>0.003</td>
<td>0.22</td>
<td>0.22</td>
</tr>
<tr>
<td>IP-FD-NL-TR20</td>
<td>0.0380</td>
<td>22</td>
<td>0.00</td>
<td>0.001</td>
<td>0.24</td>
<td>0.24</td>
</tr>
<tr>
<td>QLIN-TR200</td>
<td>0.0375</td>
<td>3</td>
<td>0.56</td>
<td>0.010</td>
<td>0.31</td>
<td>0.32</td>
</tr>
<tr>
<td>IP-SD-OLQ-TR200</td>
<td>0.0375</td>
<td>29</td>
<td>1.41</td>
<td>0.002</td>
<td>0.41</td>
<td>0.41</td>
</tr>
<tr>
<td>IP-SD-NL-TR200</td>
<td>0.0375</td>
<td>29</td>
<td>2.17</td>
<td>0.003</td>
<td>0.58</td>
<td>0.59</td>
</tr>
<tr>
<td>IP-FD-OLQ-TR200</td>
<td>0.0375</td>
<td>30</td>
<td>0.00</td>
<td>0.003</td>
<td>0.88</td>
<td>0.89</td>
</tr>
<tr>
<td>IP-FD-NL-TR200</td>
<td>0.0375</td>
<td>29</td>
<td>0.00</td>
<td>0.002</td>
<td>0.95</td>
<td>0.95</td>
</tr>
<tr>
<td>QLIN-TR2000</td>
<td>0.0375</td>
<td>3</td>
<td>0.56</td>
<td>0.033</td>
<td>8.23</td>
<td>8.26</td>
</tr>
<tr>
<td>IP-SD-OLQ-TR2000</td>
<td>0.0375</td>
<td>35</td>
<td>1.42</td>
<td>0.008</td>
<td>7.21</td>
<td>7.21</td>
</tr>
<tr>
<td>IP-SD-NL-TR2000</td>
<td>0.0375</td>
<td>35</td>
<td>2.18</td>
<td>0.004</td>
<td>8.25</td>
<td>8.26</td>
</tr>
<tr>
<td>IP-FD-OLQ-TR2000</td>
<td>0.0375</td>
<td>39</td>
<td>0.00</td>
<td>0.003</td>
<td>13.92</td>
<td>13.92</td>
</tr>
<tr>
<td>IP-FD-NL-TR2000</td>
<td>0.0375</td>
<td>35</td>
<td>0.00</td>
<td>0.003</td>
<td>13.95</td>
<td>13.95</td>
</tr>
<tr>
<td>QLIN-PS10</td>
<td>0.0376</td>
<td>4</td>
<td>0.55</td>
<td>0.009</td>
<td>0.03</td>
<td>0.04</td>
</tr>
<tr>
<td>IP-SD-OLQ-PS10</td>
<td>0.0376</td>
<td>24</td>
<td>1.42</td>
<td>0.002</td>
<td>0.08</td>
<td>0.08</td>
</tr>
<tr>
<td>IP-SD-NL-PS10</td>
<td>0.0376</td>
<td>28</td>
<td>2.17</td>
<td>0.003</td>
<td>0.21</td>
<td>0.21</td>
</tr>
<tr>
<td>IP-FD-OLQ-PS10</td>
<td>0.0376</td>
<td>29</td>
<td>0.00</td>
<td>0.003</td>
<td>0.25</td>
<td>0.26</td>
</tr>
<tr>
<td>IP-FD-NL-PS10</td>
<td>0.0376</td>
<td>31</td>
<td>0.00</td>
<td>0.002</td>
<td>0.31</td>
<td>0.31</td>
</tr>
</tbody>
</table>
Failure of Quasilinearization: Van der Pol

- Variations of the Van der Pol oscillator problem have been studied as a convergent problem using quasilinearization [Jaddu 2002]
- When some parameters are introduced [Azad and Alexander-Ramos 2019], the method **fails to converge** using quasilinearization
- The parameters candidates for successive iterations oscillates between two trajectories
- An accurate solution can be obtained using an IP method

Results for a Variation of the Van der Pol Problem with Parameters
Simple Suspension

- This is a simplified quarter-car vehicle suspension problem as discussed in [Herber and Allison 2019]

**Simple Suspension Problem Formulation**

\[
\begin{align*}
\min_{u, \xi, p} & \quad \int_{0}^{t_f} \left[ w_1 \xi_1^2 + w_2 \xi_4^2 + w_3 u^2 \right] dt \\
\text{subject to:} & \\
\dot{\xi}(t) &= \begin{bmatrix}
0 \\
-k_t m_{us}/4 \\
0 \\
0
\end{bmatrix} + \begin{bmatrix}
0 \\
\frac{1}{m_{us}/4} \\
0 \\
0
\end{bmatrix} \xi(t) + \cdots \\
&+ \begin{bmatrix}
0 \\
-\frac{1}{m_{us}/4} \\
0 \\
\frac{1}{m_s/4}
\end{bmatrix} u(t) + \begin{bmatrix}
\frac{1}{c_t m_{us}/4} \\
0 \\
0
\end{bmatrix} \dot{z}_0(t) \\
\xi(0) &= 0 \\
|\xi_3(t)| &\leq r_{\max} \\
p_{\min} &\leq p \leq p_{\max}
\end{align*}
\]
Simple Suspension Results

- Tests were conducted using all the methods including the **two-level fixed-parameter (TLFP)** and **all-at-once (IP)** strategies.
- Optimal value for the **damper constant is 101.12 Ns/m** and **21.558 kN/m for the spring constant**.
Simple Suspension Results (continued)

<table>
<thead>
<tr>
<th>Method</th>
<th>$v$</th>
<th>Iter.</th>
<th>$T_{sym}$</th>
<th>$T_{int}$</th>
<th>$T_{opt}$</th>
<th>$T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>TLFP-TR200</td>
<td>1.977</td>
<td>17</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>1.10</td>
</tr>
<tr>
<td>IP-SD-OLQ-TR200</td>
<td>1.977</td>
<td>12</td>
<td>6.46</td>
<td>0.003</td>
<td>0.3</td>
<td>0.31</td>
</tr>
<tr>
<td>IP-SD-NL-TR200</td>
<td>1.977</td>
<td>12</td>
<td>6.47</td>
<td>0.003</td>
<td>0.3</td>
<td>0.33</td>
</tr>
<tr>
<td>IP-FD-OLQ-TR200</td>
<td>2.048</td>
<td>598</td>
<td>0.00</td>
<td>0.003</td>
<td>21.8</td>
<td>21.80</td>
</tr>
<tr>
<td>IP-FD-NL-TR200</td>
<td>2.048</td>
<td>598</td>
<td>0.00</td>
<td>0.003</td>
<td>21.9</td>
<td>21.92</td>
</tr>
<tr>
<td>TLFP-TR2000</td>
<td>1.996</td>
<td>22</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>9.18</td>
</tr>
<tr>
<td>IP-SD-OLQ-TR2000</td>
<td>1.996</td>
<td>13</td>
<td>6.49</td>
<td>0.005</td>
<td>1.6</td>
<td>1.63</td>
</tr>
<tr>
<td>IP-SD-NL-TR2000</td>
<td>1.996</td>
<td>13</td>
<td>6.51</td>
<td>0.004</td>
<td>1.7</td>
<td>1.74</td>
</tr>
<tr>
<td>IP-FD-OLQ-TR2000</td>
<td>1.997</td>
<td>599</td>
<td>0.00</td>
<td>0.006</td>
<td>216.7</td>
<td>216.68</td>
</tr>
<tr>
<td>IP-FD-NL-TR2000</td>
<td>1.997</td>
<td>599</td>
<td>0.00</td>
<td>0.004</td>
<td>216.4</td>
<td>216.43</td>
</tr>
<tr>
<td>QLIN-TRX</td>
<td>—</td>
<td>did not converge</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- TLFP approach is nearly **20-24× faster** than the IP method when using finite-differences.
- This example highlights the effectiveness of **TLFP** methods in the **absence of symbolic derivatives** or other efficient derivative methods.
- The difference between OLQ and NL options is **relatively small**, as only two of the state derivative functions were OLQ elements, and the objective function was nonlinear.
- This problem **did not converge** when using quasilinearization.
References


Questions?

On the Uses of Linear-Quadratic Methods in Solving Nonlinear Dynamic Optimization Problems with Direct Transcription
IMECE2020-23885

Daniel R. Herber
Colorado State University
daniel.herber@colostate.edu
www.engr.colostate.edu/~drherber

Athul K. Sundarrajan
Colorado State University
Athul.Sundarrajan@colostate.edu

https://github.com/danielrherber/dt-qp-project
~/examples/IMECE2020-23885