2018 Annual American Control Conference (ACC)
June 27–29, 2018. Wisconsin Center, Milwaukee, USA

Nonlinear Model Predictive Control Reduction Using Truncated Single Shooting

Anson Maitland, Mohit Batra, and John McPhee

Abstract— **We present a new scheme to reduce the computational burden of Nonlinear Model Predictive Controllers (NMPCs). Working with the direct single shooting formulation of NMPC, we identify the timestep as a 'small' parameter and conduct a perturbative analysis of Newton's step. This analysis leads to the introduction of a truncated Lagrangian that can be used to form a new quasi-Newton method with a symbolically reduced Hessian for online optimization. To the authors' knowledge, these methods have not been presented before.**

In this work we leverage the power of symbolic computation to generate efficient controller code for online evaluation. The reduction methods were applied to diesel engine control and electric vehicle cruise control problems and achieved turnaround times more than 2 times faster with no tradeoff in controller performance. These initial results are quite promising for the development of real-time NMPCs and introduce some new avenues for research.

I. Introduction

The computational burden of modern control strategies, such as nonlinear model predictive control (NMPC), has proven to be a major hurdle in their adoption by industry. Thus, the many benefits of these strategies claimed in the literature are waiting to be realized on a commercial scale. For example, it has long been recognized that the electronic control units (ECUs) used in the automotive industry are not capable of satisfactorily solving nonlinear optimization problems within the desired sampling time of an engine, but the benefits of such capability would be revolutionary [1]. The real-time constraint of many systems requires turnaround times (TATs) on the order of milliseconds. This paper provides a step towards meeting this objective by introducing a controller reduction technique applicable to NMPC.

The receding horizon principle behind NMPC is general enough that there exist multiple formulations [2]. In this paper we focus on the single shooting (SS) (sometimes called control parameterization) approach where only the control inputs, and not the states, are exposed as degrees of freedom in the transcribed optimization problem [3]. This approach yields a denser optimization problem as compared to collocation, the more popular approach, which treats plant dynamics as equality constraints. SS has been used in trajectory planning and vehicle collision avoidance problems [4], [5] but has typically been avoided since it suffers from poor stability properties [6]. In its numerical implementation, SS has two stages. The model is first simulated in an inner loop and then the derivatives are computed, through sensitivity or adjoint equations, for the optimizer that runs in an outer loop. Such computations are expensive and this staged approach, though straightforward to implement, is often outperformed by other methods [7]. In this paper we leverage the power of symbolic computation for controller reduction and code generation so that our controllers operate efficiently in a single stage.

To achieve efficient controllers, recent work has focused on collocation with specially designed solvers to take advantage of the shape and sparsity of the underlying equations [8], [9]. In spirit, the work of this paper is opposite to this trend in that we purposefully generate a dense system of minimal dimension. In this way most of the cost is not in the solution but the computation of the system to be solved, which we simplify using optimized code generation and our truncation strategy.

In Section II we review a sample NMPC problem and its formulation using SS. Then in Section III we carry out a perturbative analysis of the resulting problem and introduce the truncated Lagrangian. In Section IV we compress the Lagrangian to yield greater reduction and then in Section V we conduct two numerical studies of our proposed method on the control of a diesel airpath model and an electric vehicle cruise control.

II. Single Shooting in NMPC

Let us consider a simple tracking problem with quadratic cost:

$$\begin{aligned} \arg\min_{\mathbf{u}} J(\mathbf{u}) &= \frac{1}{2}\int_0^{t_f} \|\Delta\mathbf{x}\|^2_{\mathbf{Q}} + \|\mathbf{u}\|^2_{\mathbf{R}}\, dt \\ \text{subject to} \quad \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x},\mathbf{u}), \\ 0 &\leq \mathbf{g}(\mathbf{x},\mathbf{u}), \end{aligned} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{u} \in \mathbb{R}^m$ are the states and control inputs, respectively, $\|\cdot\|^2_{\mathbf{A}}$ denotes the weighted vector norm of matrix $\mathbf{A}$, $\mathbf{Q} \in \mathbb{R}^{n\times n}, \mathbf{R} \in \mathbb{R}^{m\times m}$ are positive definite matrices selected to balance the tracking error and control cost, $\mathbf{x}(0) = \mathbf{x}_0$ is given and $\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_{ref}$ where $\mathbf{x}_{ref}$ is the desired reference trajectory. For a general plant model $\mathbf{f}$ and constraints $\mathbf{g}$ this is an intractable infinite-dimensional optimization problem. NMPC is a feedback control strategy that allows us to find approximate solutions of (1) and is suitable for online implementation. Hence, it has gained much attention by researchers and industry [1], [10].

Using SS, problem (1) is transformed into a sequence of finite dimensional optimization problems. This is done by

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and Maplesoft™.

The authors are with the Department of Systems Design Engineering, University of Waterloo, 200 University Ave W, Waterloo, Canada. Corresponding author: anson.maitland@uwaterloo.ca

first discretizing time using a fixed step size Δt and selecting the prediction horizon H_p and control horizon H_c such that $1 \leq H_c \leq H_p$. Since our problem is autonomous, the finite dimensional optimization problem at each timestep can be written as

$$\begin{aligned} \arg\min_{\overline{\mathbf{u}}} \mathcal{J}(\overline{\mathbf{u}}) &= \frac{1}{2}\left(\sum_{k=0}^{H_p-1} \|\Delta\mathbf{x}(k+1)\|^2_{\mathbf{Q}} + \sum_{k=0}^{H_c-1} \|\mathbf{u}(k)\|^2_{\mathbf{R}}\right) \\ \text{subject to} \quad & 0 \leq \mathbf{g}(\mathbf{x}(k), \mathbf{u}(k)) \quad \text{for } 0 \leq k \leq H_p, \end{aligned} \tag{2}$$

where $\overline{\mathbf{u}} = \begin{pmatrix}\mathbf{u}(0)^T & \cdots & \mathbf{u}(H_c-1)^T\end{pmatrix}^T$ are the discretized controls over the horizon. For computational simplicity the states over the prediction horizon are defined using a one-step method Φ via

$$\mathbf{x}(k+1) = \mathbf{x}(k) + \Delta t \Phi(\mathbf{x}(k), \mathbf{u}(k)), \tag{3}$$

where we define $\mathbf{u}(k) = \mathbf{u}(H_c - 1)$ over the tail, *i.e.* $H_c \leq k \leq H_p$. The simplest one-step method is the forward Euler scheme where $\Phi = \mathbf{f}$. In this problem formulation all states in the prediction horizon are functions of the initial state $\mathbf{x}_0$ and the control inputs $\overline{\mathbf{u}}$. The NMPC runs by measuring the plant to get $\mathbf{x}_0$ then solves (2) for $\overline{\mathbf{u}}^*$ and then implements the control $\mathbf{u}^*(0)$ at each timestep.

Problem (2) is often solved using Newton's method. The Lagrangian of (2) is

$$\mathcal{L}(\overline{\mathbf{u}}) = \Theta(\overline{\mathbf{u}}) + \mathcal{J}(\overline{\mathbf{u}}) \tag{4}$$

where Θ contains any terms used to enforce the constraints *e.g.* any Lagrange multiplier terms and barrier or penalty functions [11]. Typically Θ can be expressed as a sum of the constraints over the prediction horizon and is of the form

$$\Theta(\overline{\mathbf{u}}) = \sum_{i=1}^{c}\sum_{k=0}^{H_p-1} \theta_i(g_i(\mathbf{x}(k), \mathbf{u}(k))) \tag{5}$$

where function θ_i captures the method of handling constraint g_i and c is the total number of constraints.

The solution $\overline{\mathbf{u}}^*$ of (2) is found by finding a minimum of $\mathcal{L}$. This involves repeatedly solving the linear system

$$\mathbf{H}_{\mathcal{L}}(\overline{\mathbf{u}})\Delta\overline{\mathbf{u}}_{NS} = -\nabla\mathcal{L}(\overline{\mathbf{u}})^T \tag{6}$$

while updating $\overline{\mathbf{u}} = \overline{\mathbf{u}} + \Delta\overline{\mathbf{u}}_{NS}$ until convergence, where $\nabla\mathcal{L}$ and $\mathbf{H}_{\mathcal{L}}$ are the gradient and Hessian of $\mathcal{L}$ with respect to $\overline{\mathbf{u}}$, respectively.

For real-time control applications with high sampling rates, the computation of $\overline{\mathbf{u}}^*$ presents a challenge. Many applications do not have the resources to solve problem (2) within the required turnaround time. During each iteration of Newton's method there are two computationally challenging steps: (1) computing $\nabla\mathcal{L}$, $\mathbf{H}_{\mathcal{L}}$ and (2) the linear solve. The former is often the most expensive step. The issue of decreasing the linear solve time has been addressed in [12]. In this paper we present a method that decreases the cost of both steps leading to improved controller turnaround times for NMPC using SS.

III. Perturbative Expansion of Single Shooting

If the model $\mathbf{f}$ is Lipschitz continuous in $\mathbf{x}$ with Lipschitz constant L then $\Delta t = C/L$ for some constant C [13], [14]. Furthermore, if C is small or, if over the control scenario of interest the rate of change of the model does not approach L, then we can identify Δt as a 'small' parameter. Thus we can expect to reduce the computational cost of (6) by conducting a perturbative analysis to truncate the expressions without a significant increase in solution error.

For ease of presentation we take a closer look at a simple one-dimensional system $\dot{x} = f(x,u)$ with $x, u \in \mathbb{R}$ and $H_c = H_p = H$. For notational compactness in this section, we denote $x(k) = x^k, u(k) = u^k$ so that $\overline{u} = \begin{pmatrix}u^0 \cdots u^{H-1}\end{pmatrix}^T$. We begin by examining the cost term $\mathcal{J}$ which can be written as

$$\mathcal{J}(\overline{u}) = \frac{1}{2}\sum_{k=0}^{H-1} Q(\Delta x^{k+1})^2 + R(u^k)^2$$

where $Q, R \in \mathbb{R}$. To obtain $\nabla\mathcal{J}$ and $\mathbf{H}_{\mathcal{J}}$ we need to compute terms $\frac{\partial\mathcal{J}}{\partial u^j}$ and $\frac{\partial^2\mathcal{J}}{\partial u^{j+m}\partial u^j}$ for $m \geq 0$, which require the derivatives of the states with respect to the controls, *e.g.* $\frac{\partial x^{k+1}}{\partial u^j}$. The states in SS are defined recursively by (3) and so the derivatives are also defined recursively, *e.g.*

$$\frac{\partial x^{k+1}}{\partial u^j} = \left(1 + \Delta t \frac{\partial\Phi}{\partial x}\bigg|_k\right)\frac{\partial x^k}{\partial u^j} + \Delta t \frac{\partial\Phi}{\partial u}\bigg|_k \delta_{j,k},$$

where $\frac{\partial\Phi}{\partial x}\big|_k$ denotes $\frac{\partial\Phi}{\partial x}(x^k, u^k)$ and $\delta_{j,k}$ is the Kronecker delta. If we expand the first and second derivatives of the states and then truncate to first order in Δt we uncover the simple expressions

$$\frac{\partial x^{k+1}}{\partial u^j} \approx \begin{cases} 0 & j > k \\ \Delta t \frac{\partial\Phi}{\partial u}\Big|_j & j \leq k \end{cases} \tag{7}$$

and

$$\frac{\partial^2 x^{k+1}}{\partial u^{j+m}\partial u^j} \approx \begin{cases} 0 & j > k \\ \Delta t \frac{\partial^2\Phi}{\partial u^2}\Big|_j \delta_{m,0} & j \leq k \end{cases}. \tag{8}$$

This leads to the following expressions for the derivatives of the cost term to first order in Δt

$$\frac{\partial\mathcal{J}}{\partial u^j} \approx Ru^j + \Delta t \frac{\partial\Phi}{\partial u}\bigg|_j \sum_{k=j}^{H-1} Q\Delta x^{k+1}, \tag{9}$$

$$\frac{\partial^2\mathcal{J}}{\partial u^{j+m}\partial u^j} \approx \delta_{m,0}\left(R + \Delta t \frac{\partial^2\Phi}{\partial u^2}\bigg|_j \sum_{k=j}^{H-1} Q\Delta x^{k+1}\right). \tag{10}$$

Notably (10) indicates that $\mathbf{H}_{\mathcal{J}}$ is diagonally dominant and that all off-diagonal terms are of order Δt^2 or higher.

Turning our attention to the constraint term, if we assume Θ is of the form given by (5), then Θ only weakly couples

the controls from different timesteps. Using similar analysis to the above we find, to first order in Δt,

$$\frac{\partial \Theta}{\partial u^j} \approx \sum_{i=1}^{c} \left(\frac{\partial \theta_i}{\partial g} \frac{\partial g_i}{\partial u}\bigg|_j + \Delta t \frac{\partial \Phi}{\partial u}\bigg|_j \sum_{k=j+1}^{H-1} \frac{\partial \theta_i}{\partial g} \frac{\partial g_i}{\partial x}\bigg|_k \right),$$

$$\begin{aligned}\frac{\partial^2 \Theta}{\partial u^{j+m} \partial u^j} \approx \sum_{i=1}^{c} \Bigg(& \delta_{m,0} \left[\frac{\partial^2 \theta_i}{\partial g^2} \left(\frac{\partial g_i}{\partial u}\bigg|_j \right)^2 + \frac{\partial \theta_i}{\partial g} \frac{\partial^2 g_i}{\partial u^2}\bigg|_j \right. \\ & \left. + \Delta t \frac{\partial^2 \Phi}{\partial u^2}\bigg|_j \sum_{k=j+1}^{H-1} \frac{\partial \theta_i}{\partial g} \frac{\partial g_i}{\partial x}\bigg|_k \right] \\ & + (1-\delta_{m,0}) \Delta t \frac{\partial \Phi}{\partial u}\bigg|_j \left[\frac{\partial^2 \theta_i}{\partial g^2} \frac{\partial g_i}{\partial u}\bigg|_{j+m} \frac{\partial g_i}{\partial x}\bigg|_{j+m} \right. \\ & \left. + \frac{\partial \theta_i}{\partial g} \frac{\partial^2 g_i}{\partial u \partial x}\bigg|_{j+m} \right] \Bigg).\end{aligned}$$

From this we can see that $\mathbf{H}_\Theta$ is also diagonally dominant because off-diagonals of order Δt are present only if some of the constraints in $\mathbf{g}$ involve both states and controls.

These observations can be used to define a truncated Lagrangian whose gradient and Hessian are equal to the original Lagrangian to first order in Δt. Uncovering such an expression allows us to easily generate code for the truncated system using symbolic computing software like MAPLE. For the general problem (2), the truncated Lagrangian is

$$\mathcal{L}^t(\overline{\mathbf{u}}, \overline{\mathbf{v}}) = \Theta^t(\overline{\mathbf{u}}, \overline{\mathbf{v}}) + \mathcal{J}^t(\overline{\mathbf{u}}, \overline{\mathbf{v}}) \tag{11}$$

where

$$\begin{aligned}\Theta^t(\overline{\mathbf{u}}, \overline{\mathbf{v}}) = \sum_{i=1}^{c} \sum_{k=0}^{H_p-1} & \theta_i(g_i(\mathbf{x}(k), \mathbf{u}(k)) \\ & + \Delta t \Phi(\mathbf{x}(k), \mathbf{u}(k))^T \Gamma(\overline{\mathbf{v}}, k)\end{aligned} \tag{12}$$

with $\Gamma(\overline{\mathbf{v}}, k) = \sum_{l=k+1}^{H_p-1} \frac{\partial \theta_i}{\partial g} \frac{\partial g_i}{\partial \mathbf{x}}\Big|_l$ and

$$\begin{aligned}\mathcal{J}^t(\overline{\mathbf{u}}, \overline{\mathbf{v}}) = & \sum_{k=0}^{H_p-1} \Delta t \Phi(\mathbf{x}(k), \mathbf{u}(k))^T \mathbf{Q} X(\overline{\mathbf{v}}, k) \\ & + \frac{1}{2} \sum_{k=0}^{H_c-1} \|\mathbf{u}(k)\|^2_{\mathbf{R}},\end{aligned} \tag{13}$$

with $X(\overline{\mathbf{v}}, k) = \sum_{l=k}^{H_p-1} \Delta \mathbf{x}(l+1)$. It is critical to note that in (12) and (13) the states $\mathbf{x}(k)$ are realized as functions of $\overline{\mathbf{v}}$ not $\overline{\mathbf{u}}$. Further, in expressions Γ, X any arguments $\mathbf{u}$ have been replaced by the dummy variable $\mathbf{v}$, *e.g.* $\frac{\partial g_i}{\partial \mathbf{x}}\Big|_1 = \frac{\partial g_i}{\partial \mathbf{x}}(\mathbf{x}(1), \mathbf{v}(1))$ where $\mathbf{x}(1) = \mathbf{x}_0 + \Delta t \Phi(\mathbf{x}_0, \mathbf{v}(0))$. With this definition of the truncated Lagrangian, $\nabla \mathcal{L}^t$ and $\mathbf{H}_{\mathcal{L}^t}$ taken with respect to the first argument $\overline{\mathbf{u}}$ and evaluated at $\overline{\mathbf{v}} = \overline{\mathbf{u}}$ are equal to $\nabla \mathcal{L}$ and the diagonals of $\mathbf{H}_\mathcal{L}$ to first order in Δt, respectively. The dummy variable $\overline{\mathbf{v}}$ has been introduced to allow for easy symbolic differentiation since we can apply automatic differentiation of $\mathcal{L}^t$ with respect to $\overline{\mathbf{u}}$ followed by the substitution $\overline{\mathbf{v}} \leftarrow \overline{\mathbf{u}}$.

The fact that the truncated system is block diagonal is not only good for an efficient linear solution but can be exploited further. How this can be done is discussed in the next section where we introduce a compression scheme.

IV. Truncated and Compressed Single Shooting

Truncated Single Shooting (TSS) for NMPC can be implemented by replacing $\mathbf{H}_\mathcal{L}(\overline{\mathbf{u}})$ in (6) with $\mathbf{H}_{\mathcal{L}^t}(\overline{\mathbf{u}}, \overline{\mathbf{v}})$ where the derivatives have been taken with respect to the first argument and are evaluated at $\overline{\mathbf{v}} = \overline{\mathbf{u}}$. It was found that also replacing $\nabla \mathcal{L}$ led to worse controller performance with no significant computational savings. In this regard TSS can be seen as a quasi-Newton method for SS with a block diagonal reduced Hessian.

The perturbative analysis of Section III suggests that $\mathbf{H}_\mathcal{L}$ is block diagonally dominant meaning the solutions $\mathbf{u}^*(k)$ for different timesteps k in the horizon are weakly coupled. Since the NMPC algorithm implements $\mathbf{u}^*(0)$ we can take advantage of this and only solve for $\mathbf{u}^*(0)$ with minimal degradation in controller performance. This can be done with or without truncation.

A compressed system is formed by simply changing the input argument of the problem Lagrangian to form a compressed Lagrangian

$$\mathcal{L}^c(\mathbf{u}') = \mathcal{L}\left(\overline{\mathbf{u}}^c\right) \tag{14}$$

where $\overline{\mathbf{u}}^c = \begin{pmatrix} \mathbf{u}'^T & \mathbf{u}^c(1)^T & \cdots & \mathbf{u}^c(H_c-1)^T \end{pmatrix}^T$. The resulting linear system

$$\mathbf{H}_{\mathcal{L}^c}(\mathbf{u}') \Delta \mathbf{u}'_{NS} = -\nabla \mathcal{L}^c(\mathbf{u}')^T \tag{15}$$

where the gradient and Hessian have been derived with respect to $\mathbf{u}'$. Equation (15) is of minimal dimension and its solution $\mathbf{u}'^*$ is often a remarkably good approximation to the actual $\mathbf{u}^*(0)$ when using Newton's method. The compressed system minimizes the expense of the linear solve by reducing the problem dimension from mH_c to m. A SS NMPC that uses (15) instead of (6) is called a compressed single shooting (CSS) NMPC controller, and when compression is combined with truncation we call it truncated compressed single shooting (TCSS).

There is one important design decision to be made when utilizing CSS. Although we need to solve for only the next control step, we need to use some sequence of control inputs over the horizon to compute the gradient and Hessian, that is, we need to choose the $\mathbf{u}^c(k)$ values. Depending on the control objective and the dynamics of the plant, this can be done a number of different ways. One would be to set all $\mathbf{u}^c(k) = \mathbf{u}'$ which is equivalent to simply setting $H_c = 1$. Other options are to use a fixed equilibrium control input or to use $\mathbf{u}'$ from the previous timestep. Many other strategies could be used but they would all require using a fixed rule with values from previous timesteps, precomputed values or guesses since the $\mathbf{u}^c(k)$ values are never updated directly from (15). One detail to note is that, although we include the case $H_c = 1$ in CSS, it differs from most other strategies since the computational complexity of the

derivatives increases as $\mathbf{u}^c(k) = \mathbf{u}'$ for all timesteps k in the prediction horizon.

V. Numerical Studies

In this section we apply truncation and compression to two control scenarios. The first is a diesel airpath (DAP) controller and the second is an electric vehicle cruise control problem.

A. Implementation Details

Our (T/C)SS NMPCs were implemented using Newton's method for online optimization. To apply Newton's method, the functions for the gradient and Hessian were precomputed using MAPLE. In this way each iteration of the solver only involved two computational steps: first, evaluating the gradient and Hessian at the current iteration and second, solving the linear system for the iteration update. Unlike numerical SS methods no simulation stage was necessary. We fixed the number of Newton iterations at 1 and used a precomputed warm start control for the initial step.

Leveraging the power of symbolic computation we were able to handle the recursive definition of the states in SS. This was done by realizing the Lagrangian as a straight line program of symbolic expressions on which we could carry out automatic differentiation followed by code optimization to generate very efficient functions to evaluate the gradient and Hessian.

B. Diesel Airpath Control

The main goal of this controller is to track a reference output trajectory where $\mathbf{y} = \begin{pmatrix} p_{int} & \phi_{EGR} \end{pmatrix}^T$, the intake manifold pressure and EGR rate, are the outputs of interest. Tracking these two outputs allows the engine to meet the driver's demands while satisfying emission requirements. The DAP model under consideration is a 9 state, nonlinear, physics based-model with 3 controls and 2 disturbance inputs. More information on the model and its derivation can be found in [15] and the references therein.

A further objective of the controller is to minimize the rate of change of the controls $\mathbf{u} = \begin{pmatrix} \theta & u_{EGR} & u_{VGT} \end{pmatrix}^T$, throttle command [% closed], EGR valve command [% open], and VGT command [% normalized], while satisfying the control bounds $\theta \in [0, 100], u_{EGR} \in [0, 70]$ and $u_{VGT} \in [40, 80]$. The cost is given by

$$\mathcal{J}(\overline{\mathbf{u}}) = \sum_{k=0}^{H_p-1} \|\Delta \mathbf{y}(k+1)\|^2_{\mathbf{Q}} + \sum_{k=0}^{H_c-1} \|\Delta \mathbf{u}(k)\|^2_{\mathbf{R}}$$

where $\Delta \mathbf{u}(k) = \mathbf{u}(k+1) - \mathbf{u}(k)$, $\mathbf{Q} = \text{diag}\begin{pmatrix} 1 & 10^4 \end{pmatrix}$ and $\mathbf{R} = \text{diag}\begin{pmatrix} 10 & 10 & 10 \end{pmatrix}$. The control bounds were enforced using the penalty function

$$\Theta(\overline{\mathbf{u}}) = \sum_{k=0}^{H_c-1} \|\rho_p(\mathbf{u}(k))\|^2_{\mathbf{P}} \tag{16}$$

where the half-penalty function ρ_p for $p \in \mathbb{N}$ is given by

$$\rho_p(z) = \left(\frac{2z - (z^{\max} + z^{\min})}{z^{\max} - z^{\min}} \right)^p$$

which is evaluated entry-wise for vector-valued arguments. This penalty function was chosen to enforce control bounds of the form $z \in [z^{\min}, z^{\max}]$ due to its behaviour in the limit $p \to \infty$. In our simulations we set $p = 4$ and $\mathbf{P} = \mathbf{I}$. We used a timestep of $\Delta t = 10$ms and prediction horizon $H_p = 10$. The disturbances were input as piecewise constants perturbed every 1.43s.

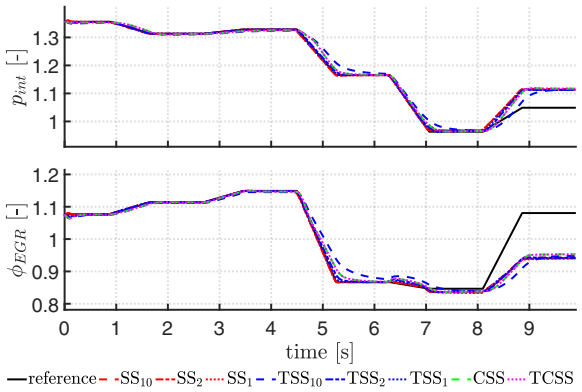
TABLE I
DAP Controller Performance

	Max TAT [ms]	TAT Reduction [×]	$\tilde{J}_- / \tilde{J}_{\text{TCSS}}$ [–]
SS_{10}	0.523	–	0.992
TSS_{10}	0.157	3.34	0.987
SS_2	0.191	–	1.002
TSS_2	0.094	2.04	0.998
SS_1	0.139	–	1.005
TSS_1	0.091	1.52	0.998
CSS	0.134	1.01	1.000
TCSS	0.067	2.06	1

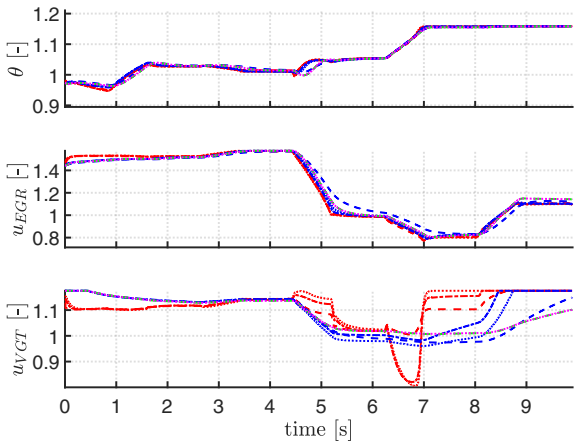
1) Results: For the purposes of comparing controller turnaround times (TATs) and performance we considered 8 different controllers. Those simulations labeled SS_k use the full Lagrangian of the single shooting method (4) and those labeled TSS_k use Hessian of the truncated Lagrangian with $H_c = k$. Labels CSS and TCSS refer to the simulations with compressed and both truncated and compressed Lagrangians, respectively. In our simulations we choose the compression strategy of setting all $\mathbf{u}^c$ values equal to the control input of the previous timestep. All TATs were measured using a MicroAutobox II 1401 real-time computer. The TAT reduction factors are computed with respect to the SS controller with the same control horizon.

In Table I we summarize the results of the DAP control simulations where a measure of the controller performance over the simulation is given by $\tilde{J}_- = \sum_{k=0}^{999} \|\Delta \mathbf{y}(k+1)\|^2_{\mathbf{Q}} + \|\widehat{\mathbf{u}}(k)\|^2_{\mathbf{R}}$. We can see that larger control horizons result in longer TATs but have better performance, as expected. Further, the larger the control horizon the larger the potential there is for a reduction in TAT. As alluded to in Section IV we see that SS_1 and CSS are about equal in TAT which is in line with the observation that one can view SS_1 as a compression strategy. The combination of truncation and compression does give us the best TAT. Interestingly we see that in terms of performance the truncated controllers actually performed slightly better than the non-truncated. This result is not expected but indicates that the reduction in TAT from these strategies came without any expense in terms of controller performance.

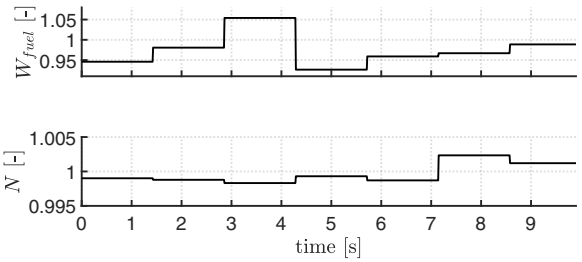
In Fig. 1 we display the simulation results of the different controllers. We can see that all controllers perform reasonably well over the first 8s (when tracking is lost due to a combination of engine regime and controller saturation) and that the CSS and TCSS solutions are effectively equal. Even though the truncated controllers have greater tracking error, they compensate by having more gentle control inputs. This is most notable in the results of u_{VGT}.



(a) Tracking Performance



(b) Control Inputs



(c) Disturbance Inputs

Fig. 1. DAP controller simulation. Note: the displayed values have been normalized and given an offset.

C. Electric Vehicle Cruise Control

The goal of this controller is to track a desired reference velocity given here by the SFTP US06 driving cycle. The vehicle model is a 6 state, stiff, nonlinear model representative of the Toyota RAV4 EV that incorporates tire slip dynamics [16]. The model captures only longitudinal vehicle dynamics and has states $\mathbf{x} = \begin{pmatrix}\omega_m & \omega_w & v & \theta_m & \theta_w & \lambda_t\end{pmatrix}^T$, which are motor angular speed, wheel angular speed, vehicle velocity, motor torsion angle, wheel angle and wheel slip, respectively. Further, there is a single control input, $\mathbf{u} =$ T [N m], motor torque. See [17] for more details about the model and a multi-objective cruise controller with slip-based constraints.

In our simulations we set $H_p = 10$ and $\Delta t = 0.1$ms.

TABLE II
CRUISE CONTROLLER PERFORMANCE

	Max TAT [ms]	TAT Reduction [×]	$\max \lvert v - v_{ref} \rvert$ [m/s]
SS_{10}	0.0240	–	1.184
TSS_{10}	0.0114	2.10	1.184
SS_2	0.0130	–	1.184
TSS_2	0.0062	2.08	1.184
SS_1	0.0120	–	0.925
TSS_1	0.0061	1.97	0.925
CSS	0.0121	1.00	0.925
TCSS	0.0062	1.95	0.925

The cost function is the same as that in (2) with $\mathbf{Q} = \text{diag}\begin{pmatrix}0 & 0 & 2\times10^9 & 0 & 0 & 0\end{pmatrix}$ and $\mathbf{R} = 2\times10^{-3}$. The control constraint $\mathrm{T} \in [-350, 350]$ was enforced using the penalty function (16) with $\mathbf{P} = 10^3$.

1) Results: In Table II we display the results of the cruise control simulations. The overall reduction factor in TAT was about $2\times$ which is inline with our DAP control simulations. The slight differences in TAT reduction observed between our two control scenarios is due to the different underlying models. The DAP model is more computationally complex and thus has longer TATs but also greater TAT reduction using truncation. Unexpectedly, those simulations with shorter control horizons performed better in velocity tracking performance. Again we can conclude that the TAT reduction came at no expense to the controller performance.

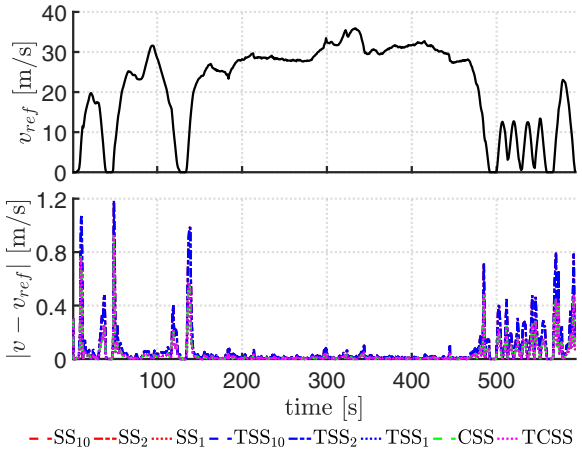
In Fig. 2 we display the results of the cruise controller simulations. The tracking performance of all controllers is poorest when v_{ref} approaches 0 but otherwise performs very well, *e.g.* over the period from 200s to 300s all the controllers are within 0.05% of v_{ref}. As we can see from the zoomed in views the controllers with a single control horizon perform better since the torque response is slightly faster.

VI. CONCLUSIONS

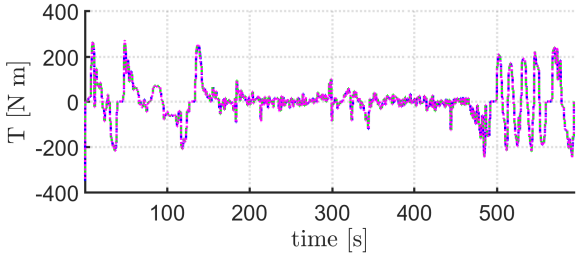
We were able to reduce TATs by half without a reduction in controller performance using the methods presented here. Truncation allowed us to reduce the computational complexity of the Hessian used in Newton's method for optimization and compression yielded further solver reductions. This approach is unique among quasi-Newton methods in that it is not a Hessian approximation but a reduction at the symbolic level of the exact Hessian. In our implementation we used the single shooting approach to NMPC and utilized symbolic computing software to precompute and optimize the required function calls necessary for Newton steps. This allowed us to eliminate a stage of computation typical to purely numerical implementations and to easily realize the truncation and compression schemes.

ACKNOWLEDGMENT

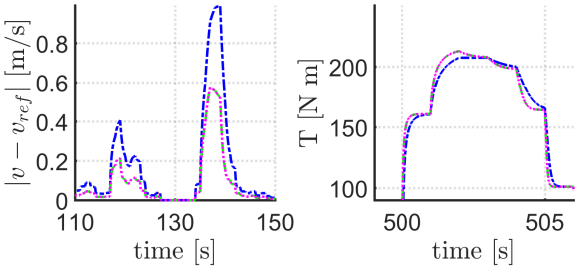
The authors would like to thank Ken Butts of Toyota Motor North America for providing the DAP model and useful feedback. We would also like to thank Kevin Walker from Maplesoft for his assistance with our symbolic computation



(a) Tracking Performance



(b) Control Inputs



(c) Zoomed in Views

Fig. 2. Cruise controller simulations. Note that SS_1, TSS_1, CSS and TCSS are all displayed on top of one other and so too are SS_{10}, TSS_{10}, SS_2 and TSS_2.

setup and Chris Shum for his troubleshooting help with the MicroAutobox II 1401 real-time computer.

References

[1] L. del Re, P. Ortner, and D. Alberer, "Chances and challenges in automotive predictive control," in *Automotive Model Predictive Control*, Springer, 2010.

[2] A. V. Rao, "A survey of numerical methods for optimal control," *Advances in the Astronautical Sciences*, vol. 135, no. 1, pp. 497–528, 2009.

[3] A. Grancharova and T. A. Johansen, *Explicit nonlinear model predictive control: Theory and applications*. Springer Science & Business Media, 2012, vol. 429.

[4] I. Spangelo and O. Egeland, "Trajectory planning and collision avoidance for underwater vehicles using optimal control," *IEEE Journal of Oceanic Engineering*, vol. 19, no. 4, pp. 502–511, 1994.

[5] M. Werling and D. Liccardo, "Automatic collision avoidance using model-predictive online optimization," in *Decision and Control (CDC), 51st Annual Conference on*, IEEE, 2012, pp. 6309–6314.

[6] S. Kameswaran and L. T. Biegler, "Simultaneous dynamic optimization strategies: Recent advances and challenges," *Computers & Chemical Engineering*, vol. 30, no. 10, pp. 1560–1575, 2006.

[7] D. Leineweber, "Efficient reduced sqp methods for the optimization of chemical processes described by large sparse dae models," PhD thesis, Universität Heidelberg, 1999.

[8] C. Kirches, L. Wirsching, S. Sager, and H. G. Bock, "Efficient numerics for nonlinear model predictive control," in *Recent Advances in Optimization and its Applications in Engineering*, Springer, 2010, pp. 339–357.

[9] M. Diehl, H. J. Ferreau, and N. Haverbeke, "Efficient numerical methods for nonlinear mpc and moving horizon estimation," *Nonlinear model predictive control*, vol. 384, pp. 391–417, 2009.

[10] S. J. Qin and T. A. Badgwell, "An overview of industrial model predictive control technology," in *AIChE Symposium Series*, New York, NY: American Institute of Chemical Engineers, 1971-c2002., vol. 93, 1997, pp. 232–256.

[11] S. J. Wright and J. Nocedal, *Numerical optimization*. Springer New York, 1999, vol. 2.

[12] A. Maitland and J. McPhee, "Improving model predictive controller turnaround time using restricted lagrangians," 2017 IEEE 56th Annual Conference on Decision and Control (CDC), 2017.

[13] I. M. Ross, P. Sekhavat, A. Fleming, and Q. Gong, "Optimal feedback control: Foundations, examples, and experimental results for a new approach," *Journal of guidance control and dynamics*, vol. 31, no. 2, pp. 307–321, 2008.

[14] I. M. Ross, Q. Gong, F. Fahroo, and W. Kang, "Practical stabilization through real-time optimal control," in *American Control Conference*, IEEE, 2006, pp. 304–309.

[15] R. Choroszucha, J Sun, and K Butts, "Nonlinear model order reduction for predictive control of the diesel engine airpath," in *American Control Conference*, IEEE, 2016, pp. 5081–5086.

[16] M. Batra, J. McPhee, and N. L. Azad, "Parameter identification for a longitudinal dynamics model based on road tests of an electric vehicle," in *ASME 2016 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, American Society of Mechanical Engineers, 2016, V003T01A026–V003T01A026.

[17] M. Batra, J. McPhee, and N. L. Azad, "Anti-jerk nonlinear model predictive cruise control with slip-based constraints for electric vehicles," in *American Control Conference*, IEEE, 2018, accepted.