EE364B Midterm Project Report: Embedded MPC

Philippe Weingertner
pweinger@stanford.edu

1 Background

We study methods for accelerating Model Predictive Control (MPC) solutions to make them suitable for real-time robotics applications. For embedded devices, reliability, determinism, and real-time performances are of paramount importance [1]. Embedded MPC solutions correspond to a growing commercial market with companies like odys, and embotech. We study how to exploit structure in the MPC problem [2, 3, 4] to get fast MPC solutions.

2 Problem Statement

We formulate a convex-embedded MPC problem as follows:

$$
\begin{array}{lll}
\min_{x,u} & \frac{1}{2}\sum_{k=0}^{N-1}\left(x_{k+1}^T Q x_{k+1} + u_k^T R u_k\right) + \frac{1}{2} x_N Q_N x_N & \\
\text{s.t.} & x_{k+1} = Ax_k + Bu_k + f & k = 0, \ldots, N-1 \\
 & Fx_k + Gu_k \leq d & k = 0, \ldots, N-1 \\
 & F_N x_N \leq d_N & \\
 & x_0 = x_{\text{init}} &
\end{array}
$$

A, B, f correspond to linearized dynamics, $x_k \in \mathbb{R}^n, u_k \in \mathbb{R}^m, Q, Q_N \in S_{++}^n, R \in S_{++}^m, A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times m}, f \in \mathbb{R}^n, F, F_N \in \mathbb{R}^{p \times n}, G \in \mathbb{R}^{p \times m}, d, d_N \in \mathbb{R}^p$

We investigate the optimization of this typical MPC problem. We review accelerated Dual Gradient-Projection Algorithms (DGPA) for embedded Linear MPC [5] and propose a related Dual Newton Projection Algorithm. We will compare the proposed algorithm with existing DGPA algorithms and some existing Interior Point based solvers.

3 Problem treatment

We rewrite the problem in the following form

$$
\begin{array}{ll}
\underset{z}{\text{minimize}} & \frac{1}{2} z^T H z \\
s.t. & A_e z = b_e \\
 & A_i z \leq b_i
\end{array}
$$

With $z = (x_0, u_0, \ldots, x_{N-1}, u_{N-1}, x_N) \in \mathbb{R}^{N(m+n)+n}$

$$
H = \begin{bmatrix} W & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 \\ 0 & 0 & W & 0 \\ 0 & 0 & 0 & Q_N \end{bmatrix}, W = \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix}, A_i = \begin{bmatrix} F & G & 0 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & 0 & \cdots & 0 \\ 0 & 0 & \cdots & F & G & \vdots \\ 0 & 0 & \cdots & 0 & 0 & F_N \end{bmatrix}, b_i = \begin{bmatrix} d \\ \vdots \\ d \\ d_N \end{bmatrix}
$$

$$A_e = \begin{bmatrix} I & 0 & 0 & \cdots & \cdots & 0 \\ -A_0 & -B_0 & I & 0 & \cdots & 0 \\ 0 & -A_1 & -B_1 & I & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & -A_{N-1} & -B_{N-1} & I \end{bmatrix}, b_e = \begin{bmatrix} x_{init} \\ f \\ f \\ \vdots \\ f \end{bmatrix}$$

H is of dimension $(N(m+n)+n) \times (N(m+n)+n)$, W is $(m+n) \times (m+n)$, A_e is $((N+1)n) \times (N(m+n)+n)$, b_e is $(N+1)n$, A_i is $((N+1)p) \times (N(m+n)+n)$, b_i is $(N+1)p$. We write the Lagrangian of our problem $L(z,\lambda) = \frac{1}{2}z^T Hz + \lambda^T (A_i z - b_i)$ with $\lambda \geq 0$. The dual function is

$$g(\lambda) = \inf_{z | A_e z = b_e} \frac{1}{2} z^T Hz + \lambda^T (A_i z - b_i) \tag{1}$$

The inner problem we have to solve, given some λ, is:

$$\begin{array}{ll} \underset{z}{\text{minimize}} & \frac{1}{2}z^T Hz + \lambda^T (A_i z - b_i) \\ s.t. & A_e z = b_e \end{array}$$

Which is a QP problem with linear equality constraints. It can be solved by applying the KKT conditions. The Lagrangian of the sub problem is

$$L_{inner}(z,\nu) = \frac{1}{2}z^T Hz + \lambda^T (A_i z - b_i) + \nu^T (A_e z - b_e)$$

The KKT conditions lead to

$$\nabla_z L_{inner}(z,\nu) = 0, \nabla_\nu L_{inner}(z,\nu) = 0$$

$\nabla_z L_{inner}(z,\nu) = Hz + A_i^T \lambda + A_e^T \nu$

$\nabla_\nu L_{inner}(z,\nu) = A_e z - b_e$

We solve the KKT system $\begin{bmatrix} H & A_e^T \\ A_e & 0 \end{bmatrix} \begin{bmatrix} z \\ \nu \end{bmatrix} = \begin{bmatrix} -A_i^T \lambda \\ b_e \end{bmatrix}$ by elimination. H is a diagonal matrix.

$$\begin{cases} Hz + A_e^T \nu = -A_i^T \lambda \\ A_e z = b_e \end{cases} \Leftrightarrow \begin{cases} z = -H^{-1}\left(A_i^T \lambda + A_e^T \nu\right) \\ A_e z = b_e \end{cases} \Leftrightarrow \begin{cases} z = -H^{-1}\left(A_i^T \lambda + A_e^T \nu\right) \\ -A_e H^{-1}\left(A_i^T \lambda + A_e^T \nu\right) = b_e \end{cases}$$

$$\begin{cases} z = -H^{-1}\left(A_i^T \lambda + A_e^T \nu\right) \\ -A_e H^{-1} A_e^T \nu = b_e + A_e H^{-1} A_i^T \lambda \end{cases} \Leftrightarrow \begin{cases} \left(A_e H^{-1} A_e^T\right) \nu = -b_e - A_e H^{-1} A_i^T \lambda \\ z = -H^{-1}\left(A_e^T \nu + A_i^T \lambda\right) \end{cases}$$

We note $H_A = A_e H^{-1} A_e^T$. As $H \succ 0$, $A_e H^{-1} A_e^T$ is Positive Definite. We get an analytical solution:

$$\begin{cases} \nu = -H_A^{-1}\left(b_e + A_e H^{-1} A_i^T \lambda\right) \\ z = -H^{-1} A_e^T \nu - H^{-1} A_i^T \lambda \end{cases}$$

$$z^*(\lambda) = -H^{-1} A_e^T \left(-H_A^{-1}\left(b_e + A_e H^{-1} A_i^T \lambda\right)\right) - H^{-1} A_i^T \lambda \tag{2}$$

We can pre-compute H_A^{-1} by Cholesky factorization in $\mathcal{O}\left(((N+1)n)^3\right)$ as $H_A \in \mathbb{R}^{(N+1)n \times (N+1)n}$. In the formula of $z^*(\lambda)$, all terms not dependent on λ, can also be pre-computed and cached. Thus we get:

$$z^*(\lambda) = H^{-1} A_e^T H_A^{-1} b_e - H^{-1} A_e^T A_e H^{-1} A_i^T \lambda - H^{-1} A_i^T \lambda$$

$$z^*(\lambda) = H^{-1} A_e^T H_A^{-1} b_e - \left(H^{-1} A_e^T A_e H^{-1} A_i^T + H^{-1} A_i^T\right) \lambda$$

$$z^*(\lambda) = Z_0 + Z_1 \lambda \tag{3}$$

With

$$Z_0 = H^{-1} A_e^T H_A^{-1} b_e, Z_1 = -\left(H^{-1} A_e^T A_e H^{-1} A_i^T + H^{-1} A_i^T\right) \tag{4}$$

Solving the inner sub problem is very fast: a matrix multiplication followed by a matrix addition.

For the dual function $g(\lambda) = \inf\limits_{z|A_e z = b_e} \frac{1}{2} z^T H z + \lambda^T (A_i z - b_i)$, we get

$$g(\lambda) = \frac{1}{2} z^* (\lambda)^T H z^* (\lambda) + \lambda^T (A_i z^* (\lambda) - b_i) \tag{5}$$

We maximize $g(\lambda)$ subject to $\lambda \geq 0$. Thus the update rule is:

$$\lambda_{k+1} \leftarrow (\lambda_k + \alpha \nabla_\lambda g(\lambda_k))_+ \tag{6}$$

$$\lambda_{k+1} \leftarrow \left(\lambda_k + \frac{1}{L} (A_i z^* (\lambda_k) - b_i)\right)_+$$

Where $\alpha = \frac{1}{L}$ with L a Lipschitz constant we will derive later below. We can summarize a first version of the algorithm, a dual projected gradient as:

Algorithm 1 Dual Projected Gradient generic

Initialize λ_0
for $k = 0$ to $K_{\max}$ **do**
    $z_k = \arg\min\limits_{z \in \{z | A_e z = b_e\}} L(z, \lambda_k)$
    $\lambda_{k+1} = \left(\lambda_k + \frac{1}{L} \nabla_\lambda L(z_k, \lambda_k)\right)_+$
end for

Algorithm 2 Dual Projected Gradient analytical

Initialize λ_0
Set $H_A = A_e H^{-1} A_e^T$
Compute H_A^{-1}
Set $Z_0 = H^{-1} A_e^T H_A^{-1} b_e$
Set $Z_1 = -\left(H^{-1} A_e^T A_e H^{-1} A_i^T + H^{-1} A_i^T\right)$
for $k = 0$ to $K_{\max}$ **do**
    $z_k = Z_0 + Z_1 \lambda_k$
    $\lambda_{k+1} = \left(\lambda_k + \frac{1}{L} (A_i z_k - b_i)\right)_+$
end for

The convergence rate of this algorithm is $\mathcal{O}\left(\frac{1}{k}\right)$, as demonstrated in Nesterov [6] section 2.1.5.

A method from Nesterov in 1983 [7] enables a much faster $\mathcal{O}\left(1/k^2\right)$ convergence rate. These methods are thoroughly treated in Nesterov [7] section 2.2. The main results are concisely summarized in Bertsekas [8] section 6.10.2: Gradient Projection with Extrapolation.The method has the form:

$$\begin{aligned} y_k &= x_k + \beta_k (x_k - x_{k-1}) && \text{extrapolation step} \\ x_{k+1} &= P_X (y_k - \alpha \nabla f(y_k)) && \text{gradient projection step} \end{aligned}$$

with $\beta_k \in (0, 1)$. With the proper choice of β_k, the method has iteration complexity $\mathcal{O}\left(1/k^2\right)$. β_k can be chosen such that $\beta_k = \frac{\theta_k (1 - \theta_{k-1})}{\theta_{k-1}}$ where the sequence $\{\theta_k\}$ satisfies $\theta_0 = \theta_1 \in (0, 1]$, and $\frac{1 - \theta_{k+1}}{\theta_{k+1}^2} \leq \frac{1}{\theta_k^2}, \theta_k \leq \frac{2}{k+2}$. One possible choice is $\beta_k = \begin{cases} 0 & \text{if } k = 0 \\ \frac{k-1}{k+2} & \text{if } k = 1, 2, \ldots \end{cases}$, $\theta_k = \begin{cases} 1 & \text{if } k = -1 \\ \frac{2}{k+2} & \text{if } k = 0, 1, \ldots \end{cases}$. Another choice, potentially faster, proposed by Tseng [9] is $\theta_{k+1} = \frac{\sqrt{\theta_k^4 + 4\theta_k^2} - \theta_k^2}{2}$ used with the same formula $\beta_k = \frac{\theta_k (1 - \theta_{k-1})}{\theta_{k-1}}$. The latter formulas are applied in Patrinos and Bemporad [5].

Algorithm 3 Accelerated Dual Projected Gradient Patrinos and Bemporad [5]

Initialize λ_0
Initialize $\theta_0 = \theta_{-1} = 1$
Set $H_A = A_e H^{-1} A_e^T$
Compute H_A^{-1}
Set $Z_0 = H^{-1} A_e^T H_A^{-1} b_e$
Set $Z_1 = -\left(H^{-1} A_e^T A_e H^{-1} A_i^T + H^{-1} A_i^T\right)$
Set $L = \left\| A_i H^{-1} A_i^T \right\|_2$
for $k = 0$ to $K_{\max}$ **do**
    $\beta_k = \frac{\theta_k (1-\theta_{k-1})}{\theta_{k-1}}$
    $\omega_k = \lambda_k + \beta_k \left(\lambda_k - \lambda_{k-1}\right)$
    $z_k = Z_0 + Z_1 \omega_k$
    $\lambda_{k+1} = \left(\omega_k + \frac{1}{L}\left(A_i z_k - b_i\right)\right)_+$
    $\theta_{k+1} = \frac{\sqrt{\theta_k^4 + 4\theta_k^2} - \theta_k^2}{2}$
end for

These algorithms have a convergence rate of $\mathcal{O}\left(1/k^2\right)$ assuming the gradient step size is $\alpha = 1/L$ where L is a Lipschitz constant such that $\left\|\nabla f(x) - \nabla f(y)\right\| \leq \|x - y\|, \forall x, y \in X$ with $f : \mathbb{R}^n \rightarrow \mathbb{R}$ a differentiable convex function defined over a closed convex set X.

One of the key question is how to derive L ? In Giselsson [10] the notions of strong convexity and Lipschitz continuity of the gradient of convex functions are generalized to account for different curvatures in different directions. For differentiable and convex functions $f : \mathbb{R}^n \rightarrow \mathbb{R}$ with a Lipschitz continuous gradient with constant L, the author generalizes the notion of Lipschitz continuity $f(x_1) \leq f(x_2) + \nabla f(x_2)^T (x_1 - x_2) + \frac{L}{2}\|x_1 - x_2\|_2^2$ to $f(x_1) \leq f(x_2) + \nabla f(x_2)^T (x_1 - x_2) + \frac{1}{2}\|x_1 - x_2\|_L^2$ where $\|x\|_L = \sqrt{x^T L x}$ with $L \in \mathbb{S}_{++}^n$and the notion of strong convexity $f(x_1) \geq f(x_2) + \nabla f(x_2)^T (x_1 - x_2) + \frac{\sigma}{2}\|x_1 - x_2\|_2^2$ to $f(x_1) \geq f(x_2) + \nabla f(x_2)^T (x_1 - x_2) + \frac{1}{2}\|x_1 - x_2\|_H^2$. These two inequalities provide lower and upper quadratic bounds of the function f. By setting $\mathbf{L} = L\mathbf{I}$ and $\mathbf{H} = \sigma\mathbf{I}$ we retrieve the original definitions. We replace the constants L and σ by matrices $\mathbf{L}$ and $\mathbf{H}$, which enable gradient updates with different step sizes in different directions. This should provide faster convergence. In Giselsson [10] section 6, an improved fast dual gradient method for solving the problem $\begin{array}{ll} \text{minimize} & \frac{1}{2} y^T H y \\ \text{subject to} & Ay = b \\ & By \leq d \end{array}$ is provided. The author introduces dual variables μ for $By \leq d$ which are updated iteratively by $\mu_k = \operatorname{prox}_g^{\mathbf{L}_\mu}\left(v^k + \mathbf{L}_\mu^{-1}\left(By^k - d\right)\right)$ where $g = I_{\mathcal{Y}}$ is an indicator function on the set $\mathcal{Y} = \{y \mid By \leq d\}$. By restricting $\mathbf{L}_\mu$ to be diagonal, this update takes the form:

$$\mu^k = \max\left(0, v^k + \mathbf{L}_\mu^{-1}\left(By^k - d\right)\right) \tag{7}$$

In the paper different methods to pre-compute the matrix $\mathbf{L}_\mu$ are presented. One option is to choose $\boldsymbol{L_\mu}$ by solving a semi-definite program: $\begin{array}{ll} \text{minimize} & \operatorname{tr} \mathbf{L}_\mu \\ \text{subject to} & \mathbf{L}_\mu \succeq BH^{-1}B^T, \mathbf{L}_\mu \text{ diagonal} \end{array}$

The main contribution of our paper is to provide a method related to the above family of Improved fast Dual Gradient Methods, which are accelerated first-order methods. However, we further exploit the structure of our problem and introduce an explicit second-order iteration step to update the dual variables. Considering the dual function in 5 and substituting with 3, we get:

$$g(\lambda) = \frac{1}{2}\left(Z_0 + Z_1\lambda\right)^T H \left(Z_0 + Z_1\lambda\right) + \lambda^T \left(A_i\left(Z_0 + Z_1\lambda\right) - b_i\right)$$

We recognize a quadratic function in λ with Hessian $H_g = \frac{1}{2} Z_1^T H Z_1 + A_i Z_1$. We change the first-order gradient update rule of the dual variable in 6 by a second-order Newton step update

$$\lambda_{k+1} \leftarrow \left(\lambda_k + H_g^{-1} \nabla g\left(\lambda_k\right)\right)_+ \tag{8}$$

Where

$$H_g^{-1} = \left(\frac{1}{2} Z_1^T H Z_1 + A_i Z_1\right)^{-1} \tag{9}$$

can be pre-computed. The Newton update can be related to 7 where H_g^{-1} is used in place of $\mathbf{L}_\mu^{-1}$. Thus, after exploiting the problem structure, we propose the following algorithm where the most expensive computations are performed before iterating:

Algorithm 4 Dual Projected Newton method

Initialize λ_0
Set $H_A = A_e H^{-1} A_e^T$
Compute H_A^{-1}
Set $Z_0 = H^{-1} A_e^T H_A^{-1} b_e$
Set $Z_1 = -\left(H^{-1} A_e^T A_e H^{-1} A_i^T + H^{-1} A_i^T\right)$
Set $H_g = \frac{1}{2} Z_1^T H Z_1 + A_i Z_1$
Compute H_g^{-1}
for $k = 0$ to $K_{\max}$ **do**
    $z_k = Z_0 + Z_1 \lambda_k$
    $\lambda_{k+1} = \left(\lambda_k + H_g^{-1} \left(A_i z_k - b_i\right)\right)_+$
end for

4 Experiments and Results

The proposed algorithm is evaluated by applying it to the AFTI-16 aircraft model as in Giselsson [10] and Patrinos and Bemporad [5]. This problem is part of the tutorial example in the MPC toolbox in MATLAB.

References

[1] Carlo Alberto Pascucci, Alberto Bemporad, Samir Bennani, and Max Rotunno. Iaa-aas-dycoss 2-14-1407 embedded mpc for space applications. In 2014.

[2] Yang Wang and Stephen Boyd. Fast model predictive control using online optimization. *IEEE Transactions on Control Systems Technology*, 18(2):267–278, 2010. DOI: `10.1109/TCST.2009.2017934`.

[3] Juan L. Jerez, Paul J. Goulart, Stefan Richter, George A. Constantinides, Eric C. Kerrigan, and Manfred Morari. Embedded online optimization for model predictive control at megahertz rates. *IEEE Transactions on Automatic Control*, 59(12):3238–3251, 2014. DOI: `10.1109/TAC.2014.2351991`.

[4] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, USA, 2004. ISBN: 0521833787.

[5] Panagiotis Patrinos and Alberto Bemporad. An accelerated dual gradient-projection algorithm for embedded linear model predictive control. *IEEE Transactions on Automatic Control*, 59(1):18–33, 2014. DOI: `10.1109/TAC.2013.2275667`.

[6] Yurii Nesterov. *Lectures on Convex Optimization*. Springer Publishing Company, Incorporated, 2nd edition, 2018. ISBN: 3319915770.

[7] Yurii Nesterov. A method for solving the convex programming problem with convergence rate o(1/k^2). *Proceedings of the USSR Academy of Sciences*, 269:543–547, 1983.

[8] Dimitri P. Bertsekas. Convex optimization theory. In 2009.

[9] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. *submitted to SIAM Journal on Optimization*, January 2008.

[10] Pontus Giselsson. Improved fast dual gradient methods for embedded model predictive control. *IFAC Proceedings Volumes*, 47:2303–2309, 2014.