A Primal-Dual Newton Method for Distributed Quadratic Programming

A Primal-Dual Newton Method for Distributed Quadratic Programming

Emil Klintberg, Sebastien Gros

***Abstract*— This paper considers the problem of solving Quadratic Programs (QP) arising in the context of distributed optimization and optimal control. A dual decomposition approach is used, where the problem is decomposed and solved in parallel, while the coupling constraints are enforced via manipulating the dual variables. In this paper, the local problems are solved using a primal-dual interior point method and the dual variables are updated using a Newton iteration, providing a fast convergence rate. Linear predictors for the local primal-dual variables and the dual variables are introduced to help the convergence of the algorithm. We observe a fast and consistent practical convergence for the proposed algorithm.**

I. Introduction

We consider a strictly convex decomposable Quadratic Program of the form

$$\min_{x} \quad \sum_{k=1}^{N} \frac{1}{2} x_k^T H_k x_k + c_k^T x_x \tag{1a}$$

$$\text{s.t.} \quad \sum_{k=1}^{N} C_k x_k = d \tag{1b}$$

$$x_k \in \mathcal{X}_k \tag{1c}$$

where, for all $k \in \{1, \dots, N\}$, $x_k \in \mathbb{R}^{n_k}$ are the decision variables with $x = [x_1^T, \dots, x_N^T]^T \in \mathbb{R}^n$. Moreover, $H_k \in \mathbb{R}^{n_k \times n_k}$ are positive definite, i.e. for all $k \in \{1, \dots, N\}$, $H_k \succ 0$, the sets $\mathcal{X}_k = \{x_k \in \mathbb{R}^{n_k} | A_k x_k = b_k, F_k x_k \leq e_k\}$ are polyhedral, $A_k \in \mathbb{R}^{l_k \times n_k}$, $b_k \in \mathbb{R}^{l_k}$, $F_k \in \mathbb{R}^{m_k \times n_k}$, $e_k \in \mathbb{R}^{m_k}$, $C_k \in \mathbb{R}^{p \times n_k}$ and $d \in \mathbb{R}^p$ yield the coupling constraints.

Problems in the form (1) arise in many applications of optimization and optimal control. They arise when Nonlinear Programs with decomposable cost functions are solved via Sequential Quadratic Programming (SQP) type methods [1]; or when Model Predictive Control (MPC) is applied to a set of sparsely interconnected subsystems, where the control problem is ultimately formulated as a QP whose structure reflects the distributed nature of the problem; and similarly in Nonlinear Model Predictive Control (NMPC) once the dynamics of the subsystems are discretized via e.g. multiple shooting methods [2], [3].

There are several methods described in the literature to solve (1) centrally, e.g. [4], [5] where the inherent sparsity structure of a distributed problem is exploited. However, centralized methods are not useful once the problem data does not fit into the shared memory. Moreover, when the system is geographically distributed or when subsystems do not want to share sensitive information, it can be highly impractical to centralize the data of the problem. In a distributed framework, a subsystem is aware of the connected subsystems, but has a very limited knowledge of their inner state, hence avoiding both heavy long-distance communication and the sharing of sensitive data.

In this paper, we treat the coupling constraints (1b) using a *Lagrangian relaxation* [6] and decompose (1) into low-dimensional subproblems that can be solved independently. Lagrangian relaxation is used in many different contexts to tackle convex large scale problems, e.g. the authors in [7] propose a coordinate ascent approach to solve matrix problems. In [8], [9], [10] a gradient method, whereas in [11], [12], [13], a fast gradient method is used in order to attain dual optimality. All these methods make use of only first order derivatives to obtain a search direction and thus their theoretical and practical convergence cannot be faster than sublinear. In the context of active-set methods, the authors of [14], [15], [16], [17] overcome this limitation by using second-order derivatives in the dual space. However, the dual Hessian can be singular if a poor initial guess is used for the dual variables, and hence possibly leading to an inconsistent Newton system. In [17], this issue was avoided by relaxing the local inequality constraints with an L_2 penalty.

In [18], it was proved that adding self-concordant barrier terms to the Lagrange function of a generic convex problem render a self-concordant dual function. Hence, in the context of interior point methods, it is possible to use Newton's method to efficiently trace the central path. This result was used in [19], where a method based on inexact solutions of the primal subproblems was suggested.

In this paper, we aim to improve the results of [18], [19] in the context of Quadratic Programming by introducing linear predictors and by using a primal-dual interior method instead of barrier methods. Primal-dual methods are often preferred over barrier methods for their numerical robustness. E.g. in [20], a very fast and robust implementation of the primal-dual interior point method is proposed for solving non-distributed QPs. Likewise, in this paper, the local QPs arising from the Lagrange relaxation are solved locally via a primal-dual interior point method. The local factorisations are re-used to form the dual Hessian alongside linear predictors for the local primal-dual variables and for the dual variables at a negligible computational cost. We show that the predictors improve the convergence of the algorithm. Our method can be viewed as an extension of [20] to a distributed framework.

The paper is organized as follows. In Section II, dual decomposition with second-order information is introduced and a proof showing that relaxing the local inequality constraint with a general smooth and convex function yields a non-singular dual Hessian. Section III presents the proposed relaxation and how to efficiently compute the necessary vec-

tors and matrices. Section IV presents the proposed Newton method. Section V deals with experimental results obtained with the proposed algorithm.

Contribution: This paper contains three contributions. First and second, we show that local matrix factorizations, obtained from taking local primal-dual Newton steps, can be re-used to compute the dual Hessian and linear predictors for the local primal-dual variables and the dual variables. Third, we use this novel ingredients to form a distributed primal-dual interior point algorithm.

II. Dual Decomposition with Second-order Information

In this section, we introduce the second-order dual decomposition approach. We recall the issue of having a singular dual Hessian and show that it can be avoided by relaxing local inequality constraints using any twice differentiable barrier function.

A. Dual decomposition

We introduce the dual variables $\lambda \in \mathbb{R}^p$ corresponding to the coupling constraints (1b) and define the Lagrange function as

$$\mathcal{L}(x,\lambda) = \sum_{k=1}^{N}(\frac{1}{2}x_k^T H_k x_k + c_k^T x_x) + \lambda^T(\sum_{k=1}^{N} C_k x_k - d) \quad (2)$$

Note that $\mathcal{L}(x,\lambda)$ is separable in x, i.e.

$$\mathcal{L}(x,\lambda) = \sum_{k=1}^{N} \mathcal{L}_k(x_k,\lambda) \quad (3)$$

with

$$\mathcal{L}_k(x_k,\lambda) = \frac{1}{2}x_k^T H_k x_k + c_k^T x_k + \lambda^T(C_k x_k - \frac{1}{N}d) \quad (4)$$

The dual function $d(\lambda) = -\min_{x\in\mathcal{X}} \mathcal{L}(x,\lambda)$ can thus be evaluated in parallel as

$$d(\lambda) = -\sum_{k=1}^{N} \min_{x_k\in\mathcal{X}_k} \mathcal{L}_k(x_k,\lambda) \quad (5)$$

Hence, evaluating (5) involves solving local subproblems of the form

$$\begin{array}{ll} \min\limits_{x_k} & \frac{1}{2}x_k^T H_k x_k + c_k^T x_k + \lambda^T C_k x_k \\ \text{s.t.} & A_k x_k = b_k, \quad F_k x_k \leq e_k \end{array} \quad (6)$$

Since (1) is strictly convex, $d(\lambda)$ is convex and continuously differentiable, but not twice differentiable. However, the Hessian of $d(\lambda)$ is a piecewise constant matrix and change with the active-set [17].

The *non-smooth* dual problem then reads

$$\min_{\lambda} d(\lambda) \quad (7)$$

from which solution, the solution to our original problem (1) can be recovered according to strong duality [21].

Strict convexity also implies that the gradient of $d(\lambda)$ is given by the residual of the coupling constraints [22], i.e.

$$\nabla d(\lambda) = -\sum_{k=1}^{N} C_k x_k^*(\lambda) + d \quad (8)$$

where $x_k^*(\lambda) = \arg\min_{x_k\in\mathcal{X}_k} \mathcal{L}_k(x_k,\lambda)$. The dual Hessian is then given by

$$\nabla^2 d(\lambda) = -\sum_{k=1}^{N} C_k \frac{\partial x_k^*(\lambda)}{\partial \lambda} \quad (9)$$

A Newton direction $\Delta\lambda$ in the dual space can then be obtained as a solution to the Newton system

$$\nabla^2 d(\lambda)\Delta\lambda + \nabla d(\lambda) = 0 \quad (10)$$

B. Singularity of the dual Hessian

It has been observed that the dual Hessian (9) can be singular for some λ, consequently making the Newton system inconsistent [17]. This happens when the active local inequality constraints (1c) together with the coupling constraints (1b) are linearly dependent [17], and is thus an inherent drawback when active-set methods are used in the second-order dual decomposition approach. This issue was solved in [17] by relaxing the local inequality constraints with an L_2 penalty. In this paper, we extend the proof provided in [17] by showing that we can achieve a nonsingular dual Hessian in any convex problem by relaxing the local inequality constraints using a twice differentiable and convex barrier function.

Consider a general convex problem, with objective function $f(x)$ and inequality constraints relaxed with a twice differentiable convex barrier function $\phi(x)$. Its dual function, written on a compact form, is

$$d(\lambda,\tau) = -\begin{pmatrix} \min_x & f(x) + \lambda^T C x + \tau\phi(x) \\ \text{s.t.} & Ax = b \end{pmatrix}, \quad (11)$$

where τ will be referred to as the *barrier parameter* and C summarizes all coupling constraints.

Lemma 1: For any λ and $\tau > 0$ the Hessian of the dual function $d(\lambda,\tau)$ is non-singular.

Proof: The dual Hessian is given by

$$\nabla^2_{\lambda\lambda} d(\lambda,\tau) = -C\frac{\partial x^*(\lambda,\tau)}{\partial\lambda} \quad (12)$$

where $\frac{\partial x^*(\lambda,\tau)}{\partial\lambda}$ is the solution to

$$\begin{bmatrix} \nabla^2 f(x) + \tau\nabla^2\phi(x) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial x}{\partial\lambda} \\ \frac{\partial\mu}{\partial\lambda} \end{bmatrix} = -\begin{bmatrix} C^T \\ 0 \end{bmatrix} \quad (13)$$

where μ represents dual variables corresponding to the equality constraints and $\nabla^2 f(x)$ and $\nabla^2\phi(x)$ are evaluated at $x^*(\lambda,\tau)$. If we let N be the null space of A, the dual Hessian is given by

$$\nabla^2_{\lambda\lambda} d(\lambda,\tau) = CN(N^T(\nabla^2 f(x) + \tau\nabla^2\phi(x))N)^{-1}N^T C^T \quad (14)$$

which is never singular provided that the original problem is feasible. ∎

It should however be understood, that this result also follows from [18], where it was formally proved that adding a self-concordant barrier to the Lagrangian yields a self-concordant dual function.

III. Relaxing Constraints and Computing Derivatives

In this section, we reformulate the constraint relaxation strategy to a primal-dual interior point framework. Then we present how the dual Hessian and gradient can be computed efficiently. We also introduce a predictor, that re-uses factorizations from the primal-dual framework, to update variables by exploiting sensitivity information.

A. Constraint relaxation in a primal-dual framework

Consider a local subproblem (6), with inequality constraints relaxed with a self-concordant log-barrier

$$\begin{aligned}\min_{x_k} \quad & \frac{1}{2}x_k^T H_k x_k + c_k^T x_k + \lambda^T C_k x_k \\ & - \tau \sum_{i=1}^{m_k} \log([e_k - F_k x_k]_i) \\ \text{s.t.} \quad & A_k x_k = b_k\end{aligned} \tag{15}$$

where $[e_k - F_k x_k]_i$ represents row i of $e_k - F_k x_k$. The relaxed subproblem (15) results in the following KKT-conditions

$$\begin{aligned}& H_k x_k + c_k + C_k^T \lambda + \tau F_k^T v_k + A_k^T \mu_k = 0 \\ & A_k x_k = b_k, \quad F_k x_k < e_k\end{aligned} \tag{16}$$

where element i of v_k is given by $1/[e_k - F_k x_k]_i$. By introducing the variable $y_k = \tau v_k$, the primal-dual interior-point KKT conditions, equivalent to (16), are

$$\begin{aligned}& r_k(w_k, \lambda, \tau) = \begin{bmatrix} r_{Dk}(w_k, \lambda) \\ r_{Pk}(w_k) \\ r_{Ck}(w_k, \tau) \end{bmatrix} = 0 \\ & y_k > 0, \quad F_k x_k - e_k < 0\end{aligned} \tag{17}$$

where we use the notation $w_k = [x_k^T, \mu_k^T, y_k^T]^T$ for the local primal-dual variables and $r_k(w_k, \lambda, \tau)$ is given by

$$\begin{aligned}r_{Dk}(w_k, \lambda) &= H_k x_k + c_k + C_k^T \lambda + F_k^T y_k + A_k^T \mu_k \\ r_{Pk}(w_k) &= A_k x_k - b_k \\ r_{Ck}(w_k, \tau) &= Y_k(e_k - F_k x_k) - \tau \mathbf{1}\end{aligned} \tag{18}$$

where $Y_k = \mathbf{diag}(y_k)$.

Note that (17) defines the *central path* [21] associated to (6) parametrized by the scalar τ. The solution of (17), for a given λ and τ, is computed by taking local Newton steps Δx_k, $\Delta \mu_k$, Δy_k given by

$$\begin{bmatrix} H_k & A_k^T & F_k^T \\ A_k & 0 & 0 \\ -Y_k F_k & 0 & M_k \end{bmatrix} \begin{bmatrix} \Delta x_k \\ \Delta \mu_k \\ \Delta y_k \end{bmatrix} = - \begin{bmatrix} r_{Dk} \\ r_{Pk} \\ r_{Ck} \end{bmatrix} \tag{19}$$

where $M_k = \mathbf{diag}(e_k - F_k x_k)$, while ensuring $F_k x_k < e_k$ and $y > 0$. Moreover, note that (15) and (17) have the same solution $x_k^*(\lambda, \tau)$.

B. Gradient and Hessian of the dual function

To compute Newton directions $\Delta\lambda$ in the dual space of λ, the dual Hessian and gradient are required. Since the relaxed problem (15) is strictly convex, the gradient and Hessian of its dual problem are obtained from (8) and (9) respectively, evaluated at $x_k^*(\lambda, \tau)$. The sensitivity $\frac{\partial x_k^*(\lambda,\tau)}{\partial \lambda}$ required in (9) is given by the linear system

$$\begin{bmatrix} H_k & A_k^T & F_k^T \\ A_k & 0 & 0 \\ -Y_k^* F_k & 0 & M_k^* \end{bmatrix} \begin{bmatrix} \frac{\partial x_k}{\partial \lambda} \\ \frac{\partial \mu_k}{\partial \lambda} \\ \frac{\partial y_k}{\partial \lambda} \end{bmatrix} = - \begin{bmatrix} C_k^T \\ 0 \\ 0 \end{bmatrix} \tag{20}$$

where Y_k^* and M_k^* represent Y_k and M_k evaluated at $x_k^*(\lambda, \tau)$.

The system (20) requires the factorization of the matrix used in the local Newton steps (19). The factorization used at the last Newton step can therefore be re-used in (20). This procedure provides the exact Hessian via (9) if (17) is solved exactly.

C. Predictor

Updates of the barrier parameter τ move the primal-dual solution of (17) along the central path. In this subsection, we introduce a first-order predictor to update the local primal-dual variables w_k and the dual variables λ, to account for the change of the solution upon a change of τ.

The sensitivity of the local solutions $w_k^*(\lambda, \tau)$ to a change in τ is given by the linear system

$$\begin{bmatrix} H_k & A_k^T & F_k^T \\ A_k & 0 & 0 \\ -Y_k^* F_k & 0 & M_k^* \end{bmatrix} \begin{bmatrix} \frac{\partial x_k}{\partial \tau} \\ \frac{\partial \mu_k}{\partial \tau} \\ \frac{\partial y_k}{\partial \tau} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \mathbf{1} \end{bmatrix} \tag{21}$$

Using the solutions of (20) and (21), the local primal-dual variables can hence be updated following an update of the dual variable λ and/or of the barrier parameter τ using

$$w_k^+ = w_k + \alpha \left(\frac{\partial w_k}{\partial \tau} \Delta\tau + \frac{\partial w_k}{\partial \lambda} \Delta\lambda \right) \tag{22}$$

where $\alpha \in (0, 1]$ is chosen such that $F_k x_k^+ < e_k$ and $y^+ > 0$.

Additionally, the sensitivity of the dual gradient with respect to τ follows from (21), which allows for constructing a predictor for λ upon an update of the barrier parameter τ. Indeed, using

$$\nabla^2_{\lambda\tau} d(\lambda, \tau) = - \sum_{k=1}^{N} C_k \frac{\partial x_k^*(\lambda)}{\partial \tau} \tag{23}$$

after an update $\Delta\tau$ of the barrier parameter, the dual variables λ can be updated using

$$\Delta\lambda = -(\nabla^2_{\lambda\lambda} d(\lambda, \tau))^{-1} \nabla^2_{\lambda\tau} d(\lambda, \tau) \Delta\tau \tag{24}$$

Note that the system (21) requires the same factorization as (20), thus the factorization of (19) used at the last Newton step and re-used in (20), can therefore be re-used also in (21). The procedure is inspired by Mehrotra's predictor-corrector method [23], but is different in the sense that it allows for performing the factorization of (20) on an updated primal-dual point. The local primal-dual updates (22) together with the dual update (24) will be referred to as a *predictor step*.

IV. Algorithm

In this section, we present three different versions of the proposed algorithm. First, a *full convergence* approach similar to a standard barrier method. Second, a method where only one Newton step on λ is performed for each update of τ. Third, a *fast path-following* approach that can be viewed as an extension of [20] to a distributed framework.

A. Full convergence

As a starting point, a basic algorithm is formulated where a sequence of dual problems $\min_\lambda d(\lambda, \tau)$ are solved with decreasing values of τ. We summarize the most important steps in Algorithm 1. The local primal-dual problems are solved to full convergence to find Newton directions for λ, until $\|\nabla_\lambda d(\lambda, \tau)\| \leq \epsilon_1$ is achieved (Step 2-7). The barrier parameter is then reduced and the procedure is restarted.

Note that the last local factorizations of (19) are re-used in Step 4 (to compute $\nabla^2_{\lambda\lambda} d(\lambda, \tau)$) and in Steps 7, 11, 13 (to perform predictor steps). Moreover, Step 3 (solving the local primal-dual systems) and Steps 7, 10, 13 (performing local primal-dual predictor steps) in Algorithm 1 can be performed in a completely distributed fashion.

Algorithm 1: Basic algorithm

Input : $\lambda^{(0)}$, $\tau > 0$, ϵ_1, ϵ_2
1 **while** $\tau > \epsilon_2$ **do**
2 **while** $\|\nabla_\lambda d(\lambda, \tau)\| > \epsilon_1$ **do**
3 Solve the local primal-dual systems, using (19)
4 Compute $\nabla^2_{\lambda\lambda} d(\lambda, \tau)$ and $\nabla_\lambda d(\lambda, \tau)$, using (8) and (9)
5 Solve $\nabla^2_{\lambda\lambda} d(\lambda, \tau)\Delta\lambda = -\nabla_\lambda d(\lambda, \tau)$
6 $\lambda^+ = \lambda + t\Delta\lambda$
7 Perform local primal-dual predictor step (22)
8 **end**
9 Compute $\nabla^2_{\lambda\tau} d(\lambda, \tau)$ in a distributed fashion, using (23)
10 Choose $\alpha \in (0, 1]$ such that $F_k x_k^+ < e_k$ and $y_k^+ > 0$
11 Solve $\nabla^2_{\lambda\lambda} d(\lambda, \tau)\Delta\lambda = -\nabla^2_{\lambda\tau} d(\lambda, \tau)\Delta\tau$
12 $\tau^+ = \tau + \Delta\tau$, $\lambda^+ = \lambda + \Delta\lambda$
13 Perform local primal-dual predictor step (22)
14 **end**

The algorithm is terminated when primal feasibility is achieved up to ϵ_1 and $x^*(\lambda, \tau)$ is no more than $\epsilon_2 m$-suboptimal [21]. Algorithm 1 is tested in Sec. V.

B. Path-following

Algorithm 1 can be modified by updating τ at every update of the dual variables λ, i.e. without checking the condition $\|\nabla_\lambda d(\lambda, \tau)\| \leq \epsilon_1$. The procedure is summarized in Algorithm 2.

Note that Algorithm 2 is divided into two phases: first, a centring phase brings the variables to the neighborhood of the central path, then a path-following is performed. The barrier parameter τ should be updated carefully enough, so the iterates are not leaving the region of quadratic convergence. Algorithm 2 is tested in Sec. V.

Algorithm 2: Path-following

Input : $\lambda(0)$, $\tau > 0$, ϵ_1, ϵ_2, ϵ_3
1 **Centering**: Perform Steps 2-8 of Alg. 1
2 **while** $\|\nabla_\lambda d(\lambda, \tau)\| > \epsilon_3$ *or* $\tau > \epsilon_2$ **do**
3 Solve the local primal-dual systems, using (19)
4 Compute $\nabla^2_{\lambda\lambda} d(\lambda, \tau)$, $\nabla_\lambda d(\lambda, \tau)$ and $\nabla^2_{\lambda\tau} d(\lambda, \tau)$, using (8), (9) and (23)
5 Solve $\nabla^2_{\lambda\lambda} d(\lambda, \tau)\Delta\lambda = -\nabla_\lambda d(\lambda, \tau) - \nabla^2_{\lambda\tau} d(\lambda, \tau)\Delta\tau$
6 $\tau^+ = \tau + \Delta\tau$, $\lambda^+ = \lambda + \Delta\lambda$
7 Choose α such that $F_k x_k^+ < e_k$ and $y_k^+ > 0$
8 Perform local primal-dual predictor step (22)
9 **end**

C. Fast path-following

Algorithm 2 can be reduced to a full central path neighborhood method [24]. In the path-following phase, instead of iterating the local primal-dual problems to full convergence, only one Newton iteration is performed before updating λ. Note that since (17) is not solved to full convergence, an *inexact* dual gradient $\nabla_\lambda \hat{d}(\lambda, \tau)$ and dual Hessian $\nabla^2_{\lambda\lambda} \hat{d}(\lambda, \tau)$ are used to compute the updates in the dual space.

The algorithm requires a very limited additional computational burden compared to a primal-dual interior point method deployed on a single central QP of a size comparable to a subproblem. This suggests that the algorithm could be used to develop a fast distributed QP solver.

Even though we do not state a proof of convergence, note that thanks to the predictor of Step 8 in Algorithm 2, r_k is, at a first-order approximation, unaffected by the updates of λ and τ. The local Newton steps then reduce $\|r_k\|$ while the updates on λ improve $\|\nabla_\lambda \hat{d}(\lambda, \tau)\|$ with limited disturbance on the local residual $\|r_k\|$.

V. Experimental Results

To show numerical features of the proposed algorithms randomly generated problems are used, with $N = 50$ subproblems, $n = 1000$ variables in total, $p = 50$ coupling constraints, 1000 inequality constraints and 750 equality constraints. All algorithms were implemented in Python using the Message Passing Interface (MPI) protocol.

To compare the performance of the algorithms, the total amount of *local Newton iterations* are counted, i.e. the number of factorizations of the local KKT matrices that needs to be computed. The proposed stopping criterion was $\|\nabla_\lambda d(\lambda, \tau)\|_\infty < 10^{-6}$ and $\tau < 10^{-6}$.

A. Algorithm 1

In Algorithm 1, many unnecessary iterations are performed for achieving the full convergence of both the local and the dual problems when a large τ is used, thus hindering the overall convergence of the algorithm.

The convergence is illustrated in Figure 1 for the proposed problem. Here 24 Newton iterations on the dual variables λ and an average of 43.38 total local iterations per subproblem are needed to solve the problem to the desired accuracy.

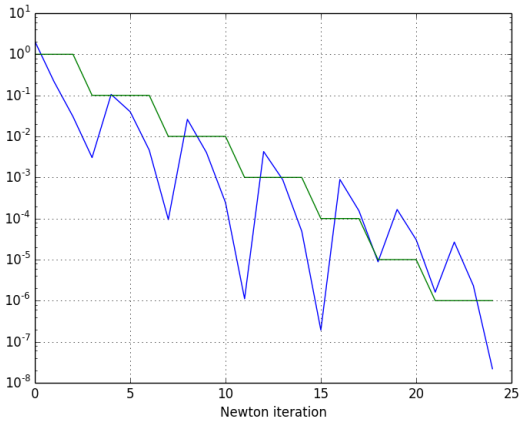

Fig. 1. Illustration of the convergence of the basic algorithm. The blue line represents $\|\nabla_\lambda d(\lambda,\tau)\|$ and the green line represents τ.

B. Fast path-following vs path-following

Numerical tests indicate that both the path-following methods require significantly fewer iterations than Algorithm 1. It is also, in general, possible to update the barrier parameter τ quite aggressively before the convergence is seriously hindered. Moreover, for reasonable update rules of τ, the path-following method often only takes one local iteration before updating λ, which then collapses into the fast path-following method. This indicates that the predictor step is a good warm start for the local Newton iterations.

The convergence of the fast path-following method for 50 randomly generated problems, with the previously described dimensions, is visualized in Figure 2. A very consistent convergence is obtained on all the tested problems.

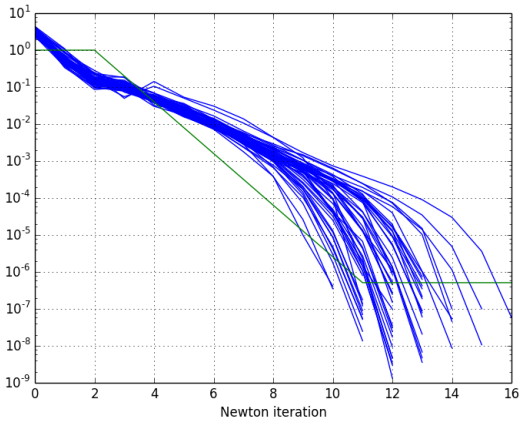

Fig. 2. Illustration of the convergence of the fast path-following. The blue lines represent $\|\nabla_\lambda \hat{d}(\lambda,\tau)\|$ and the green line represents τ.

C. Benefits of predictor step

This subsection deals with the effects of the predictor step presented in Section III-C. Numerical tests indicate that the step is useful in all three methods, here is however the fast path-following method used as an example.

The performance of the fast path-following algorithm with and without the predictor step, with a quite conservative update of τ, is visualized in Figure 3 and Figure 4 respectively.

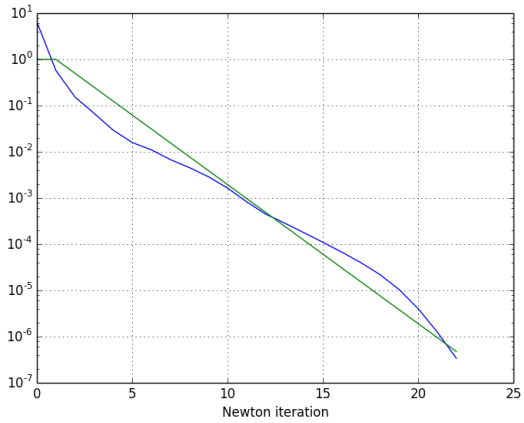

Fig. 3. Illustration of the convergence of the fast path-following algorithm with predictor steps. The blue line represents $\|\nabla_\lambda \hat{d}(\lambda,\tau)\|$ and the green line represents τ.

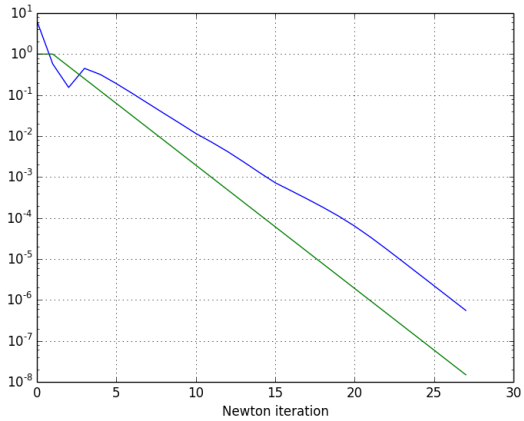

Fig. 4. Illustration of the convergence of the fast path-following algorithm without predictor steps. The blue line represents $\|\nabla_\lambda \hat{d}(\lambda,\tau)\|$ and the green line represents τ.

First, it should be observed that even for a modest update rate of the barrier parameter, the predictor step helps achieving a better convergence. The relative difference is in general bigger for more aggressive update rules, but it should be mentioned that even without the predictor step, the algorithm does most often converge even for extreme updates of τ.

VI. Conclusions and Future Work

In this paper, we consider the problem of solving distributed Quadratic Programs efficiently and improve on the results of [12], [19] by introducing linear predictors exploiting sensitivity information. In the proposed algorithm, a primal-dual interior point method is used to solve the local subproblems, and a Newton iteration is used to update the

dual variables. Predictors for the local primal-dual variables and the dual variables are introduced to improve the convergence. The resulting algorithm shows fast, consistent and robust practical convergence on the problems tested. Since local factorizations are re-used to form the dual Hessian and the predictors, the proposed algorithm requires a very limited additional computation burden when compared to a primal-dual interior point method deployed on a single Quadratic Program. This suggests that the proposed algorithm could be used to develop a fast distributed QP solver.

Future work will consider the possibility of designing an adaptive update of the barrier parameter and studying formally the convergence of the algorithm. The communication burden involved in the proposed algorithm will also be investigated.

REFERENCES

[1] M. Powell, "Algorithms for nonlinear constraints that use Lagrangian functions," *Mathematical Programming*, vol. 14, no. 3, pp. 224–248, 1978.

[2] H. Bock and K. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," in *Proceedings 9th IFAC World Congress Budapest*. Pergamon Press, 1984, pp. 242–247.

[3] C. Savorgnan, C. Romani, A. Kozma, and M. Diehl, "Multiple shooting for distributed systems with applications in hydro electricity production," *Journal of Process Control*, vol. 21, pp. 738–745, 2011.

[4] E. Gertz and S. Wright, "Object-Oriented Software for Quadratic Programming," *ACM Transactions on Mathematical Software*, vol. 29, no. 1, pp. 58–81, 2003.

[5] IBM Corp., *IBM ILOG CPLEX V12.1, User's Manual for CPLEX*, 2009.

[6] C. Lemaréchal, "Lagrangian relaxation," in *Computational Combinatorial Optimization*, ser. Lecture Notes in Computer Science, M. Jünger and D. Naddef, Eds. Springer Berlin / Heidelberg, 2001, vol. 2241, pp. 112–156. [Online]. Available: http://dx.doi.org/10.1007/3-540-45586-8_4

[7] R. W. Cottle, S. G. Duvall, and K. Zikan, "A lagrangean relaxation algorithm for the constrained matrix problem," *Naval Research Logistics Quarterly*, vol. 33, no. 1, pp. 55–76, 1986. [Online]. Available: http://dx.doi.org/10.1002/nav.3800330106

[8] L. S. Lasdon, *Optimization theory for Large Systems*. Dover, 1970.

[9] P. Tseng, "Dual ascent methods for problems with strictly convex costs and linear constraints: A unified approach," *SIAM Journal on Control and Optimization*, vol. 28, no. 1, pp. 214–242, 1990. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/0328011

[10] P. Giselsson and A. Rantzer, "Distributed model predictive control with suboptimality and stability guarantees," in *Decision and Control (CDC), 2010 49th IEEE Conference on*, 2010, pp. 7272–7277.

[11] S. Richter, M. Morari, and C. Jones, "Towards computational complexity certification for constrained MPC based on Lagrange Relaxation and the fast gradient method," in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, dec. 2011, pp. 5223 –5229.

[12] I. Necoara, V. Nedelcu, and I. Dumitrache, "Parallel and distributed optimization methods for estimation and control in networks," *Journal of Process Control*, vol. 21, no. 5, pp. 756 – 766, 2011, special Issue on Hierarchical and Distributed Model Predictive Control. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S095915241000257X

[13] P. Giselsson, M. D. Doan, T. Keviczky, B. D. Schutter, and A. Rantzer, "Accelerated gradient methods and dual decomposition in distributed model predictive control," *Automatica*, vol. 49, no. 3, pp. 829 – 833, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109813000101

[14] H. Ferreau, A. Kozma, and M. Diehl, "A parallel active-set strategy to solve sparse parametric quadratic programs arising in MPC," in *Proceedings of the 4th IFAC Nonlinear Model Predictive Control Conference, Noordwijkerhout, The Netherlands*, 2012.

[15] J. V. Frasch, M. Vukov, H. Ferreau, and M. Diehl. (2013) A dual Newton strategy for the efficient solution of sparse quadratic programs arising in SQP-based nonlinear MPC. Optimization Online 3972. [Online]. Available: http://www.optimization-online.org/DB_HTML/2013/07/3972.html

[16] A. Kozma, J. V. Frasch, and M. Diehl, "A Distributed Method for Convex Quadratic Programming Problems Arising in Optimal Control of Distributed Systems," in *Proceedings of the 52nd Conference on Decision and Control (CDC)*, 2013.

[17] A. Kozma, E. Klintberg, S. Gros, and M. Diehl, "An improved distributed dual Newton-CG method for convex quadratic programming problems," in *American Control Conference*, 2014.

[18] I. Necoara and J. Suykens, "An interior-point Lagrangian decomposition method for separable convex optimization," *J. Optim. Theory and Appl.*, vol. 143, no. 3, pp. 567–588, 2009.

[19] D. Q. Tran, I. Necoara, C. Savorgnan, and M. Diehl, "An inexact perturbed path-following method for lagrangian decomposition in large-scale separable convex optimization," *SIAM Journal on Optimization*, vol. 23, pp. 95–125, 2013.

[20] A. Domahidi, A. Zgraggen, M. Zeilinger, M. Morari, and C. Jones, "Efficient Interior Point Methods for Multistage Problems Arising in Receding Horizon Control," in *IEEE Conference on Decision and Control (CDC)*, Maui, HI, USA, Dec. 2012, pp. 668 – 674.

[21] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge: University Press, 2004.

[22] D. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: Numerical methods*. Prentice Hall, 1989.

[23] S. Mehrotra, "On the Implementation of a Primal-Dual Interior Point Method," *SIAM Journal on Optimization*, vol. 2, no. 4, pp. 575–601, 1992.

[24] J. Nocedal and S. Wright, *Numerical Optimization*, 2nd ed., ser. Springer Series in Operations Research and Financial Engineering. Springer, 2006.