A Comparative Analysis of the Fast-Lipschitz Convergence Speed

Martin Jakobsson and Carlo Fischione

Abstract—Fast-Lipschitz optimization is a recently proposed framework useful for an important class of centralized and distributed optimization problems over peer-to-peer networks. The properties of Fast-Lipschitz problems allow to compute the solution without having to introduce Lagrange multipliers, as in most other methods. This is highly beneficial, since multipliers need to be communicated across the network and thus increase the communication complexity of solution algorithms. Although the convergence speed of Fast-Lipschitz optimization methods often outperforms Lagrangian methods in practice, there is not yet a theoretical analysis. This paper provides a fundamental step towards such an analysis. Sufficient conditions for superior convergence of the Fast-Lipschitz method are established. The results are illustrated by simple examples. It is concluded that optimization problems with quadratic cost functions and linear constraints are always better solved by Fast-Lipschitz optimization methods, provided that certain conditions hold on the eigenvalues of the Hessian of the cost function and constraints.

I. INTRODUCTION

Distributed optimization problems are highly useful to take decisions in networked or large-scale systems without relying on a central control unit. Examples include communication networks and autonomous groups of robots. Based only on local information and limited cooperation among nodes or agents, these networks must choose decision variables to maximize a global utility.

Distributed optimization has attracted much attention in the recent years. Several ways to attack the problem have been proposed, such as decomposition, subgradient, and consensus methods. Most methods assume that the global utility can be split into a sum of convex functions. Dual decomposition and subgradient methods can then be used to find the optimal solution [1]–[4]. Since these approaches rely on subgradients or Lagrangian methods, they usually have slow rates of convergence. Recent works have attempted to speed up convergence by considering higher order methods [5], [6]. However, updating the dual variables requires a broadcast and collect iterative message exchanging among all nodes of the network. While this is fine for distributed optimization problems in which the cost of coordination and communication is not an issue, it is not desirable when such a cost is high. This is typically the case for wireless sensor networks. Since communication is one hundred times more expensive than computation [7], the use of Lagrangian methods would quickly deplete batteries of the nodes.

In the consensus approach to distributed optimization, nodes update their decision as a weighted combination of their previous information, and the opinion of their nearby neighbors [8]–[12]. Consensus algorithms have the benefit of not requiring any central coordination, and are robust to failures or changes of the network topology. For this reason, consensus steps often replace gather-and-broadcast procedures as means for updating the dual variables in decomposition methods. The main drawback of consensus consists in the large number of messages that need to be passed among the nodes to reach the agreement. Once again, this can especially be a problem in situations where communication is power-intensive and expensive, bandwidth is scarce, or if the system change faster than the node reach agreement.

A distributed optimization method is given by the by the Interference Function optimization theory [13]–[15], where the unique fixed point of the constraints is the optimal solution. The class of Fast-Lipschitz optimization problems, recently proposed in [7], considers the fixed point solutions of Interference Function optimization in a much more general setting. In Fast-Lipschitz optimization, the cost function is required to be monotonic and the constraints associated to nodes of the network have to satisfy some qualifying conditions. The solution is obtained by a simple parallel and distributed computation approach that applies naturally to peer-to-peer networks, namely in networks with no central coordinator and where every node is a peer of everyone else. In Fast-Lipschitz optimization, the constraint functions are used by the nodes to build contraction mappings that converges geometrically to the optimal solution with a reduced message exchanging among nodes. Although the constraints and objective functions must fulfill certain qualifying conditions, they have the remarkable property of not having to be convex, and the objective function does not have to be separable.

To solve a Fast-Lipschitz problem translates into solving a system of equations. This could be said about most optimization methods, where one typically solves the system of KKT-conditions. Fast-Lipschitz problems, however, need no dual variables, whereby the system of equations to solve is much smaller and less coupled than Lagrangian methods. Therefore, one can intuitively expect that the Fast-Lipschitz framework, when applicable, outperforms traditional methods.

This paper is a first step toward understanding the convergence speed properties of Fast-Lipschitz optimization. An investigation of the Fast-Lipschitz method and a first order Lagrangian method convergence speed is proposed. Both
methods are applied to the same constrained problem, assuming optimal step-length decisions are taken for the Lagrangian method. The convergence speeds, defined in terms of the spectral radii of the iteration matrices, are then bounded and compared. Since the analysis attempts to compare the best convergence speed of the Lagrangian method to the natural convergence speed of Fast-Lipschitz method, the results are conservative. Nevertheless, a sufficient condition for faster Fast-Lipschitz convergence is given. We believe that a theory of distributed optimization with coordination or communication constraints is still in its infancy. This paper gives an important contribution toward such a theory.

The reminder of the paper is organized as follows. Before the main contribution is given in Section III, a more detailed (yet still brief) description of the Fast-Lipschitz problem is given in Section II. Section IV illustrates the analysis by an example. The paper is concluded in Section V, where future work is also highlighted.

A. Notation

Vectors are columns and denoted by lower bold case letters, matrices are denoted with upper bold case letters. The identity matrix, zero matrix and vector of ones are written I, 0, and 1 respectively. The components of a vector \( x \) are denoted \( x_i \) or \([x]_i\). Similarly, the elements of the matrix \( A \) are denoted \( A_{ij} \) or \([A]_{ij}\). The transpose of a vector or matrix is denoted \( ^T \). If \( \mu_i(\mathbf{A}) \) are the eigenvalues of a \( n \times n \) matrix \( \mathbf{A} \), they are ordered in descending order, \( |\mu_1| \geq |\mu_2| \geq \cdots \geq |\mu_n| \). The gradient of a function is defined as \( \nabla f(x)_{ij} = \frac{\partial f_j}{\partial x_i} \). \( \rho(\mathbf{A}) \) denotes the spectral radius of the iteration matrices, \( ||\mathbf{A}||_\infty \) denotes the infinity norm where \( ||\mathbf{A}||_\infty = \max_i \sum_j |A_{ij}| \). All inequalities are intended component-wise, i.e., \( \mathbf{A} \geq \mathbf{B} \) means \( A_{ij} \geq B_{ij} \) for all pairs \( i,j \).

II. Preliminaries

In this section we recall the main results of Fast-Lipschitz theory for distributed optimization problems. We then build on these results to give the core contribution of the paper in the next section.

Definition 1. A problem is said to be on Fast-Lipschitz form if it is written as

\[
\begin{align*}
\max_{x} & \quad f_0(x) \\
\text{s.t.} & \quad x_i \leq f_i(x) \quad \forall i \in \mathcal{I}, \\
& \quad x_i = f_i(x) \quad \forall i \in \mathcal{E}, \\
& \quad x \in \mathcal{D} \subseteq \mathbb{R}^n.
\end{align*}
\]

Here, \( f_0 \) is a possibly vector valued function \( f_0 : \mathcal{D} \rightarrow \mathbb{R}^m \). \( \mathcal{I} \) and \( \mathcal{E} \) are complementary subsets of \( \{1, \ldots, n\} \), and \( f_i \) can be collected as the \( i \)th component of the vector valued function \( \mathbf{f} : \mathcal{D} \rightarrow \mathbb{R}^m \), i.e., \( \mathbf{f}(x) = [f_1(x) \cdots f_n(x)]^T \). The last constraint of problem (1) is a box constraint, meaning \( \mathcal{D} = \{x \in \mathbb{R}^n \mid a \leq x \leq b\} \). \( \mathcal{D} \) does not need to cover the entire feasible region, but must be large enough to contain all fixed points of \( \mathcal{D} \).

Remark 2. The form \( x \leq f(x) \) is general, since any constraint on canonical form, \( g(x) \leq 0 \), can be written \( x \leq x - \beta g(x) \) for some positive constant \( \beta \). An important feature of the Fast-Lipschitz form is that each variable has exactly one constraint (apart from the box \( \mathcal{D} \)).

The optimization problem can be easily associated to a peer-to-peer network when every node \( i \) owns a constraint \( f_i(x) \) and a decision variable \( x_i \), and every node wants to know its optimal decision variable that maximizes the global utility function \( f_0(x) \). These Fast-Lipschitz optimization problems are pervasive over wireless sensor networks, and can be used, e.g., for distributed estimation, distributed detection, and distributed radio power control problem. See [7] for the detail. We now give some useful definitions.

A point \( x^* \) is called Pareto optimal if there exists no other feasible point \( y \) such that \( f_0(y) > f_0(x^*) \). In other words, the objective of a Pareto optimal point need not dominate that of all other points, but no other feasible point will have an objective that dominates that of the Pareto optimum.

Definition 3. A problem is said Fast-Lipschitz when it can be written on Fast-Lipschitz form and, if feasible, admits a unique Pareto optimal solution \( x^* \), uniquely determined by the system of equations \( x^* = f(x^*) \).

A problem in Fast-Lipschitz form is not automatically Fast-Lipschitz. The conditions that, together with Fast-Lipschitz form, guarantee that a problem is Fast-Lipschitz are called qualifying conditions.

Qualifying Condition 1. Let \( \delta(x) = \min_{i,j} |\nabla f_0(x)|_{ij} \) and \( \Delta(x) = \max_{i,j} |\nabla f_0(x)|_{ij} \). For all \( x \in \mathcal{D} \), the following conditions must hold:

\[
\begin{align*}
(0) & \quad \nabla f_0(x) \geq 0 \text{ with non-zero rows} \\
(i.a) & \quad \nabla f(x) \geq 0 \\
(i.b) & \quad ||\nabla f(x)|| < 1 \text{ for some matrix norm} \\
or \quad (ii.a) & \quad \nabla f(x)^2 \geq 0 \text{ e.g., } \nabla f(x) \leq 0 \\
(ii.b) & \quad ||\nabla f(x)||_\infty < \frac{\delta(x)}{\Delta(x)} \\
or \quad (iii.a) & \quad ||\nabla f(x)||_\infty < \frac{\delta(x)}{\delta(x) + \Delta(x)}
\end{align*}
\]

There is an integer \( k \geq 2 \) such that:

\[
\begin{align*}
(iv.a) & \quad (\nabla f(x))^k \geq 0 \\
v(iv.b) & \quad ||\sum_{i=1}^{k-1} \nabla f(x)^i||_\infty < \frac{\delta(x)}{\Delta(x)} \\
v(iv.c) & \quad ||\nabla f(x)|| < 1 \text{ for some norm } ||\cdot||_\infty.
\end{align*}
\]

These conditions ensure that a problem is Fast-Lipschitz by the following theorem [16].

Theorem 4. When problem (1) is feasible, with \( f_0 \) and \( f \) fulfilling the qualifying conditions, the problem is Fast-Lipschitz, i.e., the unique Pareto optimal solution is given by

\[
x^* = f(x^*).
\]
Thus, in Fast-Lipschitz optimization, the optimal solution is given by the constraints at the equality. Many well known algorithms are available to solve systems of equations by distributed algorithms [17]. The simplest Fast-Lipschitz solution method for disturbed optimization is given by the following result, a simplified version of [7, Prop. 3.6]:

**Proposition 5.** Let \( x^0 \in \mathcal{D} \) be an initial guess of the optimal solution to a feasible Fast-Lipschitz problem (1). Then, the sequence

\[
x^{k+1} = f(x^k)
\]

will converge to the optimal point \( x^* = f(x^*) \).

Although the result above is formulated in a centralized setting, it remains true when the system is distributed and asynchronous [7]. In this case, node \( i \) updates its decision variable as \( x_{i}^{k+1} = f_i(x_{i}^{k}) \), where \( x_{i}^{k} \) is the (possibly outdated) global information available to node \( i \) at time \( k \). When \( f_i(x) \) depends only on the decision variables of the neighboring nodes, the communications of these variables is simple. In special situations, \( f_i(x) \) might be given by a local oracle at node \( i \), without any direct communication of decision variables from the other nodes. An example is the radio power optimization problem (e.g., [13]–[15]).

In the following section, we focus on the main result of the paper and provide an analysis of the convergence speed of the Fast-Lipschitz method.

### III. Convergence speed analysis

The characterization of the convergence speed of optimization algorithms is notoriously hard [18]. Ultimately, it reduces to the convergence analysis of, in general, non linear dynamical discrete systems, and it depends on which norm is used to prove convergence. Thus, we restrict ourselves to considering the spectral radius of the matrix associated to the dynamical system. In fact, the spectral radius is sometimes known as the *asymptotic convergence speed*. In the following, we give a lower bound on the spectral radius of Lagrangian methods and we compare it to the Fast-Lipschitz convergence speed.

#### A. Problem formulation

In the analysis, we consider only problems with equality constraints. This technical assumption allows us to use the most general Lagrangian method [18, Sec. 4.4].

Consider a problem on canonical from:

\[
\begin{align*}
\min & \quad f_0(x) \\
\text{s.t.} & \quad h(x) = 0 \in \mathbb{R}^n.
\end{align*}
\]

(5)

This problem can be written on Fast-Lipschitz form as

\[
\begin{align*}
\max & \quad -f_0(x) \\
\text{s.t.} & \quad x = f(x) = x - \beta h(x).
\end{align*}
\]

(6)

We assume that strong duality holds for this problem, so that we can apply Lagrangian methods to compute the optimal solution. Note that Lagrangian duality applies to both convex and non-convex problems, provided that some constraint qualifications hold [18].

In order for problem (6) to be Fast-Lipschitz, \( f(x) \) must satisfy the qualifying conditions and the fixed point \( x^* = f(x^*) \) will be unique. This means there is only one feasible point \( x^* \). The point must fulfill \( h(x^*) = 0 \), and the optimization problem collapses to solving a system of equations. This is precisely what usually gives Fast-Lipschitz optimization an advantage. However, it is not obvious that problem (5) has a unique solution if one does not know that the problem is Fast-Lipschitz. Also, the solution has to be calculated in a distributed manner, wherefore problem (5) is of substantial interest.

#### B. Lagrangian method

In this subsection we characterize the best convergence speed of the Lagrangian method for problem (5). We then compare such a bound to the Fast-Lipschitz method in the following subsection.

For the sake of generality, we consider the most general Lagrangian method available for equality constrained problems, which can thus be applied both to convex and non-convex optimization problems. Such a method is a first order one. It is possible to use higher order Lagrangian methods to get faster convergence. However, one can also claim that higher order methods can be used for solving the system (3), arising in Fast-Lipschitz optimization. Therefore, it is reasonable to restrict ourselves to first order methods.

The Lagrangian of the problem is given by

\[
L(x, \lambda) = f_0(x) + \lambda^T h(x),
\]

where \( \lambda \in \mathbb{R}^n \), \( \lambda \neq 0 \), are the Lagrangian multipliers. The optimal primal-dual variables \( (x^*, \lambda^*) \) must fulfill

\[
\begin{align*}
\nabla_x L(x^*, \lambda^*) &= \nabla f_0(x^*) + \nabla h(x^*)\lambda^* = 0 \\
\nabla_\lambda L(x^*, \lambda^*) &= h(x^*) = 0.
\end{align*}
\]

(7)

This is a system with \( 2n \) equations in \( 2n \) variables, whose solutions gives the optimal solution to the problem. The gradient method solves this by taking steps

\[
\begin{align*}
\alpha x^{k+1} &= x^k - \alpha \nabla_x L(x^k, \lambda^k) \\
\lambda^{k+1} &= \lambda^k + \alpha h(x^k).
\end{align*}
\]

(8a)

(8b)

Strictly speaking, this is a centralized solution method. It is easy to show that any distributed implementation converges slower than this one. For a shorter notation, introduce \( y = [x^T \quad \lambda^T]^T \) and the vector

\[
M_\alpha(y) = \begin{bmatrix}
x - \alpha \nabla_x L(x, \lambda) \\
\lambda + \alpha \nabla_\lambda L(x, \lambda)
\end{bmatrix}
\]

(9)

for some small enough \( \alpha > 0 \). Equations (8) can now be written as \( y^{k+1} = M_\alpha(y^k) \). From (7) it is clear that \( M_\alpha(y^*) = y^* \), where \( y^* = (x^*, \lambda^*) \). If \( M_\alpha(y) \) is contractive in some neighborhood around \( y^* \), any sequence \( y^{k+1} = M_\alpha(y^k) \) within this neighborhood will converge to \( y^* \).

The following proposition is a simplified version of [18, Prop. 4.4.1].

**Proposition 6.** Assume that \( y^* = M_\alpha(y^*) \) and that the eigenvalues of \( \nabla M_\alpha(y^*) \) have magnitudes strictly less than one. Then, for some \( y^{k_0} \) sufficiently close to \( y^* \), the sequence \( y^k \) given by \( y^{k+1} = M_\alpha(y^k) \) will converge to \( y^* \).
The proposition states that the method converges when \( \rho(\nabla M_\alpha(y^*)) < 1 \), and from the proof in [18] it is also clear that convergence becomes faster as the spectral radius becomes smaller. Thus, in order to lower bound the convergence speed of this gradient method, we need the minimal spectral radius of \( \nabla M_\alpha(y^*) \). With this goal in mind, we need two technical results before giving the main result of the subsection, which is Proposition 10.

Taking the gradient of equation (9) gives
\[
\nabla M_\alpha(y^*) = I - \alpha B,
\]
where
\[
B = \begin{bmatrix}
\nabla_{xx}L(x^*, \lambda^*) & \nabla_h(x^*) \\
-\nabla_h(x^*)^T & 0 
\end{bmatrix}
\]
is the gradient of the update direction. Note that \( B \) is a \( m \times m \) matrix, where \( m = 2n \).

**Lemma 7.** Consider \( P(\alpha) = I - \alpha B \) and assume that \( B \) has real, positive eigenvalues \( \mu_1 \geq \mu_2 \geq \cdots \geq \mu_m > 0 \). Then, the spectral radius \( \rho(P(\alpha)) \) is minimized for \( \alpha^* = \frac{2}{\mu_1 + \mu_m} \), and the minimal value is
\[
\rho_{\alpha} = \rho(P(\alpha^*)) = \frac{\mu_1 - \mu_m}{\mu_1 + \mu_m}.
\]

**Remark 8.** For analytical tractability, the lemma assumes that \( B \) has real eigenvalues. The positive eigenvalue assumption is inherited from the Lagrangian method, and necessary for convergence. The case when \( B \) has complex eigenvalues is left for future work.

Let \( \gamma_B = \mu_1 / \mu_m \) be the condition number of \( B \). Then, the smallest spectral radius of \( \nabla M_\alpha \) is
\[
\rho^*_M = \rho_B(\alpha^*) = \frac{\gamma_B - 1}{\gamma_B + 1}.
\]

Note that \( d\rho^*_M / d\gamma_B > 0 \), i.e., \( \rho^*_M \) is an increasing function of \( \gamma_B \) in the considered interval \((1, \infty)\). When \( \gamma_B = 1 \), \( \rho^*_M = 0 \) and the Lagrangian method converges in one iteration. As \( \gamma_B \) grows towards infinity, \( \rho^*_M \) will approach 1 and the convergence will stall.

Note that \( \rho^*_M \) can be lower bounded by choosing appropriately the matrix \( B \), which gives the fastest convergence of the Lagrangian method. Thus, it remains to lower bound \( \rho^*_M \), i.e., lower bound the condition number of \( B \). We need a second technical results before computing such a lower bound.

**Lemma 9.** Let \( P \) be a \( m \times m \) matrix with real, positive eigenvalues. Then,
\[
\gamma_P \geq \frac{\text{trace}(P)}{m \sqrt[\gamma]\text{det}(P)}.
\]

We are now in the position of finding an upper bound for the convergence speed of the Lagrangian method:

**Proposition 10.** Let \( \nabla M_\alpha(y^*) = I - \alpha B \), with
\[
B = \begin{bmatrix}
Q & A \\
-A^T & 0 
\end{bmatrix}.
\]

Consider the minimum spectral radius \( \rho^*_M = \min_\alpha \rho(\nabla M_\alpha(y^*)) \). Suppose that \( B \) has real and positive eigenvalues. Then
\[
\rho^*_M \geq \frac{\text{trace}(Q) - m \sqrt[\gamma]\text{det}(A)^2}{\text{trace}(Q) + m \sqrt[\gamma]\text{det}(A)^2}.
\]

**Proof:** The eigenvalues of \( B \) fulfill the assumptions of Lemma 7, so Eq. (11) gives
\[
\rho^*_M = \frac{\gamma_B - 1}{\gamma_B + 1}.
\]

For the special structure of \( B \), we have \( \text{trace}(B) = \text{trace}(Q) \), and \( \text{det}(B) = \text{det}(A)^2 \) [19]. Lemma 9 gives
\[
\gamma_B \geq \frac{\text{trace}(B)}{m \sqrt[\gamma]\text{det}(B)} = \frac{\text{trace} Q}{m \sqrt[\gamma]\text{det}(A)^2}.
\]

Since \( \rho^*_M \) is increasing in \( \gamma_B \), this implies
\[
\rho^*_M \geq \frac{\gamma_B - 1}{\gamma_B + 1} \geq \frac{\text{trace}(Q) - m \sqrt[\gamma]\text{det}(A)^2}{\text{trace}(Q) + m \sqrt[\gamma]\text{det}(A)^2},
\]
which concludes the proof.

By the previous proposition, we have an upper bound on the convergence speed of the Lagrangian method. We can now compare it to the speed of the Fast-Lipschitz method in the following section.

**C. Fast-Lipschitz Convergence Speed Comparison**

If we know problem (6) is Fast-Lipschitz, we know the solution fulfills \( x^* = f(x^*) \) and that \( f(x) = x - \beta h(x) \) is contractive. Note that this system of equations is half the size of Eq. (7). By the argument of the previous subsection, we are interested in the asymptotic convergence speed \( \rho_f = \rho(\nabla f(x^*)) \). In this sense, the Fast-Lipschitz method is faster than the Lagrangian method if \( \rho_f \) is smaller than the smallest possible \( \rho^*_M \), that is if
\[
\rho_f < \frac{\text{trace}(Q) - m \sqrt[\gamma]\text{det}(A)^2}{\text{trace}(Q) + m \sqrt[\gamma]\text{det}(A)^2} \leq \rho^*_M.
\]

Although this gives a condition that is verifiable, it can be simplified if one allows for a more conservative condition:

**Proposition 11.** Assume \( Q \) given in (12), with \( Q = \nabla^2 f_0(x^*) + \sum_k \nabla^2 f_k(x^*) \lambda^*_k \) and \( A = \nabla h(x^*) \), has real, positive eigenvalues. Then, the Fast-Lipschitz method is faster than the Lagrangian method (8) if
\[
\frac{\text{trace}(Q)}{m} > \frac{1 + \rho_f}{1 - \rho_f}.
\]

**Proof:** We need \( \rho_f < \rho^*_M \) or, from Eq. (13),
\[
\rho_f < \frac{\text{trace}(Q) - m \sqrt[\gamma]\text{det}(A)^2}{\text{trace}(Q) + m \sqrt[\gamma]\text{det}(A)^2}.
\]

This can be rewritten as
\[
\text{trace}(Q) > \frac{1 + \rho_f}{1 - \rho_f} m \sqrt[\gamma]\text{det}(A)^2.
\]

This expression is always defined, since the problem is Fast-Lipschitz, which implies \( \rho_f < 1 \). Recalling that \( f(x) = x - \beta h(x) \), we obtain \( A = \nabla h(x^*) = (I - \nabla f(x^*)) / \beta \), and we can relate \( \text{det}(A) \) to \( \rho_f \) as follows: Let \( \nu_i(\cdot) \) be the \( i \)-th eigenvalue of a matrix. We have
\[
\text{det}(A) = \prod_{i=1}^m \nu_i(A) = \prod_{i=1}^m \left( \frac{1}{\beta} (I - \nabla f(x^*)) \right)
= \frac{1}{\beta^m} \prod_{i=1}^m (1 - \nu_i(\nabla f(x^*))),
\]
which is always defined.
wherefore
\[ \sqrt{\det(A)^2} = \frac{1}{\beta^2} \prod_{i=1}^{m} (1 - \nu_i(\nabla f(x^*))^2) \]
\[ \leq \frac{1}{\beta^2} \prod_{i=1}^{m} (1 + \rho_f)^2 = \frac{(1 + \rho_f)^m}{\beta^2}. \]

This means that inequality (14) holds if
\[ \text{trace}(Q) > \frac{1 + \rho_f}{1 - \rho_f} \frac{m(1 + \rho_f)^2}{\beta^2}, \]
which concludes the proof.

The previous proposition gives a criterion to establish when Fast-Lipschitz optimization is faster than the Lagrangian method. However, Both \( Q = \nabla^2 f_0(x^*) + \sum_k \nabla^2 f_k(x^*) \lambda_k^* \) and \( \rho_f = \rho(\nabla f(x^*)) \) depend on the optimal points \( x^* \) and \( \lambda^* \). If this criterion is to be used in practice to assess if distributed optimization should be done by a Fast-Lipschitz method or by a Lagrangian method, it is only interesting before the problem is actually solved (when \( x^* \) and \( \lambda^* \) are unknown). The spectral radius dependence of \( x^* \) can be circumvented by instead considering \( \max_{x \in D} \rho(\nabla f(x)) \).

Avoiding the dual optimum \( \lambda^* \) is harder, since the only characterisation is \( \lambda^* \neq 0 \). Perhaps a bound could be found through the KKT-conditions, or specific properties of the problem could be used. For example, one could make further assumptions on the problem structure, e.g., as follows:

**Corollary 12.** Consider problem (5). Let \( f_0(x) = -x^T H x - s^T x, \) where \( H \) is a positive definite matrix and \( s \) a vector with positive entries. Let \( h(x) = C x + c, \) where the matrix \( C \) and vector \( c \) both are constant. Suppose that strong duality holds. Then, the Fast-Lipschitz method is faster than the Lagrangian method (8) if
\[ \frac{\text{trace}(H)}{m} > \frac{1}{\beta^2} \frac{(1 + \rho_f)^3}{1 - \rho_f}. \]

In a centralized setting, this linearly constrained quadratic problem is easily solved, wherefore the choice of method is less important. However, distributed applications can still be challenging and requires iterative methods, where the convergence rate, and therefore the choice of method, is of high importance.

In the next section, we illustrate the analysis developed in this section with a numerical example.

**IV. EXAMPLE**

We now show how to apply the results of this paper by a simple numerical example. To circumvent the issue of knowing \( x^* \) and \( \lambda^* \) in advance, we consider the result of Corollary 12. Specifically, we consider the linearly constrained quadratic problem:

\[ \min \quad f_0(x), \]
\[ \text{s.t.} \quad h(x) = 0, \quad x \in D = \{ x : 0 \leq x \leq 1 \}, \]
where
\[ f_0(x) = -\frac{1}{2} x^T \begin{bmatrix} 7 & 3 \\ 3 & 6 \end{bmatrix} x - \begin{bmatrix} 1 & 1 \end{bmatrix} x \]
and
\[ h(x) = \begin{bmatrix} 0.75 & -0.1 \\ -0.05 & 0.8 \end{bmatrix} x - \begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}. \]

We remark that the results of this paper apply to problems of any dimension. We have chosen a simple problem with two decision variables and two constraints for illustrative purposes.

The equivalent optimization problem on Fast-Lipschitz form is
\[ \max -f_0(x), \]
\[ \text{s.t.} \quad x = f(x), \quad x \in D, \]
where
\[ f(x) = x - \beta h(x). \]

We will test two different choices for \( \beta \) — the first corresponds to ignoring the possibility to tune \( \beta \), i.e., setting \( \beta = 1 \), and the second consists in choosing an optimal value. For the first choice, we get
\[ f^{(1)}(x) = x - 1 \cdot h(x) = \begin{bmatrix} 0.25 & 0.1 \\ 0.05 & 0.2 \end{bmatrix} x + \begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}. \]

By the theory of Section II, one can show that the rewritten problem in Fast-Lipschitz form is Fast-Lipschitz. In particular,
\[ \nabla (-f_0(x)) = \begin{bmatrix} 7 x_1 + 3 x_2 + 2 \\ 3 x_1 + 6 x_2 + 2 \end{bmatrix} > 0 \quad \forall x \geq 0, \]
so condition (2a) of the qualifying conditions holds. Furthermore,
\[ \nabla \left( f^{(1)}(x) \right) = \begin{bmatrix} 0.25 & 0.05 \\ 0.1 & 0.2 \end{bmatrix} \geq 0 \quad \forall x, \]
so condition (2b) holds. Finally,
\[ \| \nabla f^{(1)}(x) \|_\infty = \max \{ |0.25|, |0.05|, |0.1|, |0.2| \} = 0.3, \]
so \( (2c) \) is fulfilled, whereby the problem is guaranteed Fast-Lipschitz by cases (0)+(i).

If we instead apply Lemma 7 to the gradient of Eq. (16), we can choose \( \beta = 1.29 \) as the \( \beta \) minimizing \( \rho(\nabla f) \),
\[ f^{(\beta)}(x) = x - 1.29 \cdot h(x) \approx \begin{bmatrix} 0.03 & 0.13 \\ 0.06 & -0.03 \end{bmatrix} x + \begin{bmatrix} 0.52 \\ 0.77 \end{bmatrix}. \]
Again, one can see that the problem is Fast-Lipschitz by checking the qualifying conditions. Condition (2a) still holds, since the objective function is unchanged. Condition (2b) however, no longer applies:
\[ \nabla \left( f^{(\beta)}(x) \right) \approx \begin{bmatrix} 0.03 & 0.06 \\ 0.13 & -0.03 \end{bmatrix} \not\geq 0. \]

Instead, the special structure of \( \nabla \left( f^{(\beta)}(x) \right) \) gives
\[ \nabla \left( f^{(\beta)}(x) \right)^2 \approx \begin{bmatrix} 0.009 & 0 \\ 0 & 0.009 \end{bmatrix} \geq 0, \]
so condition (2d) is fulfilled. It remains to verify condition (2e). The max norm of \( \nabla f^{(\beta)}(x) \) is \( \| \nabla f^{(\beta)}(x) \|_\infty \approx 0.13 + |-0.03| = 0.16 \). For every \( x \), we have
\[ \delta(x) = \min_{i,j} |\nabla f_0|_{ij} = \frac{1}{2} \begin{bmatrix} 7 x_1 + 3 x_2 + 2 \\ 3 x_1 + 6 x_2 + 2 \end{bmatrix} \]
and
\[ \Delta(x) = \max_{i,j} |\nabla f_0|_{ij} = \frac{1}{2} \begin{bmatrix} 7 x_1 + 3 x_2 + 2 \\ 3 x_1 + 6 x_2 + 2 \end{bmatrix}. \]
wherefore
\[
\min_{x \in D} \frac{\Delta(x)}{\delta(x)} = \min_{x \in D} \left\{ \frac{7x_1 + 3x_2 + 2}{3x_1 + 6x_2 + 2}, \frac{3x_1 + 6x_2 + 2}{7x_1 + 3x_2 + 2} \right\} = \frac{5}{9} \approx 0.56
\]
when \( x \) belongs to \( D \). Therefore, we have \( ||\nabla f(\beta)(x)||_\infty < \delta(x)/\Delta(x) \) for all \( x \) in \( D \) and condition (2e) holds. The problem is Fast-Lipschitz by case (0)+(ii).

By now, we have shown that the original problem (15) can also be solved as either one of two Fast-Lipschitz problems. In order to use Corollary 12 we need
\[
\text{trace}(Q) = \text{trace}(\nabla^2 f_0(x)) = \text{trace}\left( \begin{bmatrix} 7 & 3 \\ 3 & 6 \end{bmatrix} \right) = 13,
\]
and the spectral radii of \( \nabla (f^{(1)}(x^*)) \) and \( \nabla (f^{(\beta)}(x^*)) \), which are \( \rho_{\ell,1} = 0.30 \) and \( \rho_{\ell,\beta} \approx 0.10 \) respectively. As the original problem has two variables and two constraints, the dimension of \( B \) is \( m = 4 \).

Corollary 12 now guarantees that both the Fast-Lipschitz methods will outperform the first order Lagrangian method, since
\[
3.25 = \frac{\text{trace}(Q)}{m} > \frac{1}{2} \left( 1 + \rho_{\ell,1} \right)^3 \approx 3.14,
\]
and
\[
3.25 = \frac{\text{trace}(Q)}{m} > \frac{1}{\beta^2} \left( 1 + \rho_{\ell,\beta} \right)^3 \approx 0.88.
\]
To verify these results, the problem is solved by Lagrangian method and the two Fast-Lipschitz methods respectively. The initial primal point was randomly chosen to \( x_0 \approx [0.13 \ 0.08]^T \), and the initial dual point was set to \( \lambda_0 = 0 \). For the Lagrangian method, the optimal step length \( \alpha^* \approx 0.2075 \), given by Lemma 7, was used in each iteration.

The solution converged by the Lagrangian method after 840 iterations within a infinity norm error of \( 10^{-4} \). The Fast-Lipschitz problems were solved by iterating \( x^{k+1} = f(x^k) \) until convergence, once for \( f = f^{(1)} \) and once for \( f = f^{(\beta)} \). The Fast-Lipschitz methods converged after 8 and 4 iterations, respectively, within the error tolerance used for the Lagrangian method.

The results from the three methods are plotted in Figure 1. Also shown, is the lower bound of the Lagrangian method convergence given by Lemma 10. From the figure it should be clear that this bound can be very conservative.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we provided a first fundamental step for the analysis of the convergence speed of the Fast-Lipschitz optimization method over peer-to-peer networks. The method was compared to a general Lagrangian method that can be applied to both convex and non-convex optimization problems. Specifically, we defined the spectral radius of the solution method as the index of the convergence speed. Under the assumption of real eigenvalues of the Hessian associated to the solution method, a general condition showing when Fast-Lipschitz method is faster than Lagrangian methods was established. The condition was applied to problems with quadratic cost function and linear constraints.