


Article

A Method for Transforming Non-Convex Optimization Problem to Distributed Form

Oleg O. Khamisov ¹, Oleg V. Khamisov ^{1,*}, Todor D. Ganchev ^{2,*}  and Eugene S. Semenkin ^{3,*}

¹ Department of Applied Mathematics, Melentiev Energy Systems Institute, 664033 Irkutsk, Russia; cygx151@gmail.com

² Department of Computer Science and Engineering, Technical University of Varna, 9010 Varna, Bulgaria

³ Scientific and Educational Center “Artificial Intelligence Technologies”, Baumann Moscow State Technical University, 105005 Moscow, Russia

* Correspondence: khamisov@isem.irk.ru (O.V.K.); tganchev@tu-varna.bg (T.D.G.); e.semenkin@emtc.ru (E.S.S.)

Abstract: We propose a novel distributed method for non-convex optimization problems with coupling equality and inequality constraints. This method transforms the optimization problem into a specific form to allow distributed implementation of modified gradient descent and Newton’s methods so that they operate as if they were distributed. We demonstrate that for the proposed distributed method: (i) communications are significantly less time-consuming than oracle calls, (ii) its convergence rate is equivalent to the convergence of Newton’s method concerning oracle calls, and (iii) for the cases when oracle calls are more expensive than communication between agents, the transition from a centralized to a distributed paradigm does not significantly affect computational time. The proposed method is applicable when the objective function is twice differentiable and constraints are differentiable, which holds for a wide range of machine learning methods and optimization setups.

Keywords: distributed optimization; non-convex optimization; gradient descent; Newton’s method

MSC: 68W15; 68Q85



Citation: Khamisov, O.O.; Khamisov, O.V.; Ganchev, T.D.; Semenkin, E.S.

A Method for Transforming Non-Convex Optimization Problem to Distributed Form. *Mathematics* **2024**, *12*, 2796. <https://doi.org/10.3390/math12172796>

Academic Editor: Jüri Majak

Received: 29 June 2024

Revised: 23 August 2024

Accepted: 29 August 2024

Published: 9 September 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In modern society, digital technologies play an essential role in organizing our work and daily routine. Ubiquitous computing and digital technologies enable us to solve a wide range of complex problems in such important fields as ecology [1,2] and medicine [3,4].

The complexity of creating decision support systems in a digital environment requires the use of advanced technologies for designing and optimizing intelligent information processing systems. For example, within a holistic approach to integrating computational intelligence systems and human expert knowledge [5], it is possible to automatically design machine learning models with self-tuning adaptive stochastic optimization algorithms [6,7]. In this case, the processed data can remain on the problem owner’s servers, which ensures trust and allows us to remain within a federated approach to learning, and the resulting models will be interpretable and explainable [8]. However, very large-scale optimization problems arising under such conditions require special computations decomposition methods [9]. At the same time, in many cases, optimization problems of this kind have properties that allow the use of rigorous mathematical methods in their solution, which makes it possible to effectively use hybrid approaches [10]. In this regard, along with the improvement in adaptive methods of computational intelligence, the evolution of traditional optimization methods is of great importance. The aim of such advancements could be focused on adaption to problems of extremely high dimensionality and federated learning through the decentralization of work and to use such properties of theirs as guaranteed

convergence to the optimum at high speed, which is fundamentally important in the tasks under consideration.

Specifically, in the following, we propose a novel decentralized optimization method for non-convex optimization problems with a separable objective function and coupling equality and inequality constraints. Under standard assumptions for distributed optimization [11,12], the problem has to be solved by a set of agents communicating over a connected graph. These agents are expected to communicate synchronously and can transmit real-valued numbers to adjacent agents. Communications are performed synchronously, and communication delays and packet losses are ignored. In addition to standard conditions, agents cannot share their decision variables and objective functions as they are considered to be private information.

Decentralized optimization already proved to be an essential instrument in a wide variety of applications. Namely, application in optimal transport [13] including coordination of mobile autonomous agents [14,15] and railway traffic [16,17], power systems control [18,19] with demand response [20] as well as data analysis in sensor networks [21,21]. Finally, decentralized optimization gains increasing popularity in federated learning [22] and support vector machines [23].

1.1. Related Work

While decentralized optimization has many applications of practical relevance, most of the corresponding results are dedicated to convex or strictly convex optimization. A comprehensive survey covering these areas can be found in [24]. The majority of these methods can be separated into primal [13,25], dual [26] or ADMM-based approaches [27,28]. Additionally, there exists a set of works utilizing the primal–dual approach [29,30].

The literature dedicated to non-convex or non-linear constraints is significantly more scarce. One of the proposed approaches is the application of SQP with the inner ADMM method [31]. However, in this work, coupling constraints are linear. Lagrangian methods for polynomial objective and equality constraints are presented in [32], and non-linear coupling constraints are considered. However, each constraint is associated with one of the agents and coupling is present only with the variables of adjacent agents.

Finally, there exist several works that consider convex optimization problems with separable objective functions such that decision variables and corresponding summands of objective functions are considered private for each agent and cannot be exchanged with other participants [19,33]

1.2. Contribution

Here, we propose a novel distributed optimization algorithm for non-convex optimization problems with equality and inequality constraints. It is assumed that the objective function is twice differentiable and the constraints are differentiable. In addition, communications are significantly less time consuming than oracle calls. Under these assumptions, we show that

1. The proposed method can be applied to any optimization problem with non-convex separable objective function and coupling constraints.
2. Its convergence rate is equivalent to the convergence of Newton's method with respect to oracle calls.
3. Decision variables, cost and constraint functions are not exchanged between agents.

The theoretical results are supported by numerical experiments.

1.3. Paper Organization

The remainder of this article is organized as follows. Section 2 introduces the problem statement. Section 3 is dedicated to the reformulation of optimization problems with equality constraints. Section 4 outlines the distributed gradient descent algorithm. In Section 5, a problem with equality and inequality constraint and its equivalent formulation

is presented. Section 6 outlines the distributed Newton method. Finally, Section 7 presents the Arrow–Hurwicz method and a numerical example.

1.4. Notations

Let $\mathbf{1}_K$ denote the vector of ones of the size K ; I^q is the identity $q \times q$ matrix. Operator vec is the vectorization operator for a matrix A ; $\ker A$ is the matrix kernel. For two matrices A and B , the Kronecker product is denoted by $A \otimes B$, and $\text{diag}(A, B)$ means the extended matrix of the corresponding size with A and B as the blocks on the main diagonal. For a twice differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}$, ∇f and $\nabla^2 f$ are the gradient and Hessian matrix, respectively. For a vector function $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$, its Jacobian matrix is denoted by Jg . For a vertex (agent) i in a communication graph, a set of adjacent vertices is defined by $\text{Adj}(i)$.

2. Problem Statement

Let us consider a non-convex optimization problem with a coupling objective function and constraints that must be solved by a multi-agent connected network with N vertices (agents). The optimization problem has the following form:

$$\min_x \left\{ F(x) = \sum_{i=1}^N f^i(x^i) \right\}, \tag{1a}$$

subject to

$$G(x) = \sum_{i=1}^N g^i(x^i) = 0, \tag{1b}$$

$$H(x) = \sum_{i=1}^N h^i(x^i) \leq 0. \tag{1c}$$

Here, the vector of objective variables $x \in \mathbb{R}^n$ is separated into subvectors of local variables $x^i \in \mathbb{R}^{n_i}, i \in \{1, \dots, N\}$, $n_1 + n_2 + \dots + n_N = n$. All functions $f^i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}, g^i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}^{\hat{m}}$ and $h^i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}^{\hat{m}}, i \in \{1, \dots, N\}$ are smooth.

Here, we postulate that the Problem (1) has to be solved in the distributed way, which brings the following perspective. It is assumed that each subvector x^i belongs to an agent i and cannot be shared with the other agents, and the agent network is defined by a connected graph with the Laplacian matrix $L \in \mathbb{R}^{N \times N}$. Every node in the graph represents some agent. Communication in the network is allowed only between neighboring vertices (agents). An agent i is characterized by its objective function f^i , equality constraint function g^i and inequality constrain function h^i . The objective function F is separable, so the main difficulty in deriving a distributed version of problem (1) is given by constraints (1b) and (1c). They are coupling (non-local) even though the constraint functions G and H are also separable.

Subsequently, we set the goal to develop an algorithm and present a reduction, which, for an arbitrary problem (1), creates an auxiliary problem, such that

1. The auxiliary problem has the same solution as (1);
2. Methods of gradient descent, gradient projection and quasi-Newton methods will operate as distributed optimization methods when applied to the auxiliary problem without any modification (1):

3. Optimization Problem with Equality Constraints

Let us first introduce a simplified version of the problem (1);

$$\min_{x \in \mathbb{R}^n} \left\{ F(x) = f^1(x^1) + f^2(x^2) + \dots + f^N(x^N) \right\}, \tag{2a}$$

subject to

$$g_1^1(x^1) + g_1^2(x^2) + \dots + g_1^N(x^N) = 0, \tag{2b}$$

$$g_2^1(x^1) + g_2^2(x^2) + \dots + g_2^N(x^N) = 0, \tag{2c}$$

$$\dots$$

$$g_{\tilde{m}}^1(x^1) + g_{\tilde{m}}^2(x^2) + \dots + g_{\tilde{m}}^N(x^N) = 0, \tag{2d}$$

i.e., the problem defined as in (1) without the inequality constraints. We introduce into consideration an auxiliary vector $y \in \mathbb{R}^{N\tilde{m}}$ consisting of subvectors $y^i \in \mathbb{R}^{\tilde{m}}$, $y^i = (y_1^i, y_2^i, \dots, y_{\tilde{m}}^i)^\top$, $i = 1, \dots, N$. Each subvector y^i is connected to the constraint function g^i of the vertex i . Consider the j -th scalar equality constraint from (2):

$$\sum_{i=1}^N g_j^i(x^i) = 0. \tag{3}$$

It can be reformulated to the following form:

$$\begin{aligned} g_j^1(x^1) + \sum_{i=1}^N L_{1i}y_j^i &= 0, \\ g_j^2(x^2) + \sum_{i=1}^N L_{2i}y_j^i &= 0, \\ &\vdots \\ g_j^N(x^N) + \sum_{i=1}^N L_{Ni}y_j^i &= 0, \end{aligned} \tag{4}$$

where L_{si} , $s, i = 1, \dots, N$ are elements of the Laplacian matrix L . Such a transition is performed for all \tilde{m} equality constraints. Thus, in the further consideration, we use the following notation:

$$\tilde{g} = \text{vec}(g^1, g^2, \dots, g^N), \tag{5}$$

$$\tilde{L} = (L \otimes I^{\tilde{m}}), \tag{6}$$

where $L \otimes I^{\tilde{m}}$ is the Kronecker product of the Laplacian matrix L and the identity matrix $I^{\tilde{m}}$. The interpretation of representation (5) is given in Figure 1.

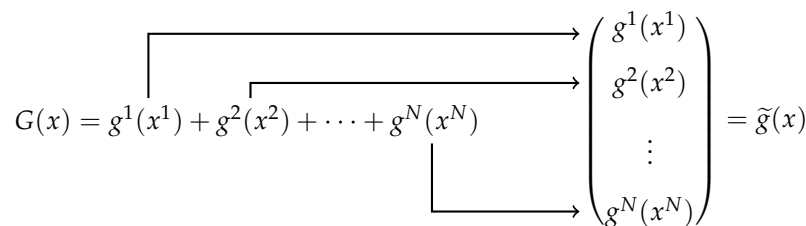


Figure 1. The agent network with information available at each node.

Repeating the transition from (3) to (4) for all $j = 1, \dots, \tilde{m}$ yields the following reformulation of problem (2):

$$\min_{(x,y) \in \mathbb{R}^n \times \mathbb{R}^{N\tilde{m}}} F(x), \tag{7a}$$

$$\tilde{g}(x) + \tilde{L}y = 0. \tag{7b}$$

Lemma 1. The following statements are correct:

1. Problem (2) is feasible if and only if problem (7) is feasible.
2. A pair (x^*, y^*) is the solution to problem (7) if and only if x^* is the solution to (2).

Proof. 1. Consider system (4) as a system of linear equations with respect to variables $(y_j^1, y_j^2, \dots, y_j^N)$ for fixed x and right-hand side vector $(-g_j^1(x^1), -g_j^2(x^2), \dots, -g_j^N(x^N))$.

According to the Fredholm Alternative [34] and due to the symmetricity of L , this system is consistent if and only if

$$\text{vec}\left(g_j^1(x^1), \dots, g_j^N(x^N)\right) \perp \ker L, \tag{8}$$

where $\ker L = \{v \in \mathbb{R}^N : Lv = 0\}$ is the kernel of the Laplacian matrix L . Since the agent graph is connected, $\ker L = \{v \in \mathbb{R}^N : v = \rho \mathbf{1}_N, \rho \in \mathbb{R}, \rho \neq 0\}$. Then, (8) is equivalent to

$$\text{vec}\left(g_j^1(x^1), \dots, g_j^N(x^N)\right)^\top \mathbf{1}_N = \sum_{i=1}^N g_j^i(x^i) = 0. \tag{9}$$

Repeating this consideration for all $j = 1, \dots, m$, we find that problems (2) and (7) are feasible simultaneously.

2. The correctness of the second statement follows from the fact that both problems have the same objective function. \square

Let us consider the main property of system (4). Since L is the Laplacian matrix, this system can be rewritten in the following form:

$$\begin{aligned} g_j^1(x^1) + \sum_{i \in \text{Adj}(1)} (y_j^1 - y_j^i) &= 0, \\ g_j^2(x^2) + \sum_{i \in \text{Adj}(2)} (y_j^2 - y_j^i) &= 0, \\ &\vdots \\ g_j^N(x^N) + \sum_{i \in \text{Adj}(N)} (y_j^N - y_j^i) &= 0. \end{aligned} \tag{10}$$

In order to evaluate the ℓ -th constraint in (10), it is necessary to know the local variables x^ℓ , y_j^ℓ , local function g_j^ℓ and variables y_j^i from the neighboring vertices $i \in \text{Adj}(\ell)$ only. This is the main advantage of system (10) in comparison to constraint (3), for which it is necessary to know information from all vertices of the agent network.

If we fix vector y , for example, $y = \tilde{y}$, then, due to the separability of function F (see (2a)) and property (10), optimization with respect to the remaining vector x in problem (7) can be performed separately, i.e., each agent ℓ independently solves the corresponding problem:

$$\min_{x^\ell} f^\ell(x^\ell), \tag{11}$$

$$g_j^\ell(x^\ell) = - \sum_{i \in \text{Adj}(\ell)} (\tilde{y}_j^\ell - \tilde{y}_j^i), \quad j = 1, \dots, \tilde{m}. \tag{12}$$

Assume that problems (11) and (12) are solvable for all $\ell = 1, \dots, N$, and \tilde{x}^ℓ is the corresponding solutions. The vector $\tilde{x} = \text{vec}(\tilde{x}^1, \dots, \tilde{x}^N)$ provides the solution of problem (7) for fixed $y = \tilde{y}$.

4. Gradient Descent in Variables y

Variables y are called communication variables. If we set $\tilde{y}^\ell = y^{0,\ell} = 0$, $\ell = 1, \dots, N$, and problems (11) and (12) are solvable with $x^{0,\ell}$ as the corresponding solutions, then the pair (x^0, y^0) is a feasible starting point for problem (7), and $F^0 = F(x^0)$ is a starting objective function record value.

Let us write down the Lagrange function for problem (7) as

$$V(x, y, \lambda) = \sum_{i=1}^N f^i(x^i) + \lambda^\top (\tilde{g}(x) + \tilde{L}y), \tag{13}$$

where $\lambda \in \mathbb{R}^{N\tilde{m}}$ is the vector of Lagrange multipliers consisting of subvectors $\lambda^i = (\lambda_1^i, \lambda_2^i, \dots, \lambda_{\tilde{m}}^i)^\top$, $i = 1, \dots, N$. Each subvector λ^i corresponds to constraint vector-function g^i of the agent i . The corresponding necessary optimality conditions

$$\frac{\partial V}{\partial x} = \text{vec}(\nabla f^1(x^1), \dots, \nabla f^N(x^N)) + J\tilde{g}(x)^\top \lambda = 0, \tag{14a}$$

$$\frac{\partial V}{\partial y} = \tilde{L}\lambda = 0, \tag{14b}$$

$$\frac{\partial V}{\partial \lambda} = \tilde{g}(x) + \tilde{L}y = 0, \tag{14c}$$

where $J\tilde{g}(x) = \text{diag}(Jg^1(x^1), Jg^2(x^2), \dots, Jg^N(x^N))$ is the Jacobian of $\tilde{g}(x)$. If we again fix $y = y^0$ and solve the corresponding problem (7) for $y = y^0$, obtaining the corresponding primal solution x^0 and dual solution λ^0 , then conditions (14a) and (14c) will be satisfied with $x = x^0$ and $\lambda = \lambda^0$. Condition (14b) can be violated $\frac{\partial V}{\partial y} = \tilde{L}\lambda^0 \neq 0$, since we do not perform optimization in y . Hence, we correct y^0 by the gradient descent step in y

$$y^1 = y^0 - \rho_0 \tilde{L}\lambda^0.$$

In general, we obtain the following recalculation formula for y^k :

$$y^{k+1} = y^k - \rho_k \tilde{L}\lambda^k. \tag{15}$$

From (14b), we have the following element-wise representation at step k due to the structure of the Laplacian matrix L and the structure of vector λ :

$$\frac{\partial V(x^k, y^k, \lambda^k)}{\partial y_j^i} = \sum_{s \in \text{Adj}(i)} (\lambda_j^{k,i} - \lambda_j^{k,s}), \quad j = 1, \dots, \tilde{m}. \tag{16}$$

Therefore, the calculation of $\frac{\partial V}{\partial y_j^i}$ satisfies the distributed form, since λ_j^i is a local dual variable and all λ_j^k are dual variables from adjacent agents.

The main computational scheme for solving problem (7) is presented in Algorithm 1. We assume here that problem (7) is solvable for $y = 0$.

Since we do not make any convexity assumptions, Algorithm 1 is suggested for finding a strict local saddle point in the Lagrange function in the sense of [35]. In [36], it is pointed out that values ρ at Step 4 must be chosen small enough in order to achieve convergence to a strict local saddle point. One of the recommended choices for ρ_k is the following: $\rho_0 = 1$, $\rho_k = \frac{1}{\sqrt{k}}$ for $k > 1$.

Example 1. Consider the problem with $N = 4$, $n_i = 1$, $i = 1, \dots, 4$, $\tilde{m} = 1$ and

$$f^1(x^1) = (x^1)^4 + 3(x^1)^3 - (x^1)^2 - 3x^1, \quad f^2(x^2) = -\sin(x^2) + 0.1(x^2)^2,$$

$$f^3(x^3) = (x^3)^2 - 4, \quad f^4(x^4) = x^4,$$

$$g^1(x^1) = (x^1)^3 - 8, \quad g^2(x^2) = (x^2)^2 - 7x^2 + 10, \quad g^3(x^3) = x^3 - 1, \quad g^4(x^4) = (x^4)^2 - 9.$$

The network is described by the Laplacian matrix

$$L = \begin{pmatrix} 2 & -1 & 0 & -1 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ -1 & 0 & -1 & 2 \end{pmatrix}.$$

The starting solution of the corresponding problem (7) with $y = y^0 = 0$ is the following $x^0 = (-2, 2, 1, -3)^\top$, $\lambda^0 = (0.417, 0.272, 2.000, 0.167)^\top$, $F(x^0) = -12.509$. The tolerance $\varepsilon = 0.1$.

The ε -solution was obtained after 294 iterations of Algorithm 1: $x^{294} = (-2.131, -1.954, 0.098, -2.877)^\top$, $F(x^{294}) = -13.972$, $\rho_k = \frac{1}{\sqrt{k}}$. The optimal solution $x^* = (-2.182, 1.831, -0.093, -2.678)^\top$, $F(x^*) = -14.013$. Algorithm 1 generated a sequence of feasible points with decreasing objective function values.

Algorithm 1 Gradient descent method (for agent ℓ)

Input: $f^\ell, g^\ell, \varepsilon > 0$.

Output: $x^{\ell,*}$

Algorithm steps:

Step 1. Set $y^{0,\ell} = 0$ and $k = 0$;

Step 2. Obtain $y^{k,i}$, $i \in \text{Adj}(\ell)$ from neighboring agents;

Step 3. Solve problem (11) and (12) for $\tilde{y}^\ell = y^{k,\ell}$, $\tilde{y}^i = y^{k,i}$, $i \in \text{Adj}(\ell)$. Let $x^{k,\ell}$, $\lambda^{k,\ell}$ be the corresponding primal and dual solutions;

Step 4. Obtain $\lambda^{k,i}$, $i \in \text{Adj}(\ell)$ from neighboring agents;

Step 5. If

$$\left| \frac{\partial V(x^k, y^k, \lambda^k)}{\partial y_j^\ell} \right| = \left| \sum_{s \in \text{Adj}(\ell)} (\lambda_j^{k,\ell} - \lambda_j^{k,s}) \right| < \varepsilon \quad \forall j = 1, \dots, \tilde{m},$$

then go to Step 8;

Step 6. Calculate $y^{k+1,\ell}$:

$$y^{k+1,\ell} = y^{k,\ell} - \rho_k \frac{\partial V(x^k, y^k, \lambda^k)}{\partial y_j^\ell};$$

Step 7. Set $k = k + 1$ and go to Step 2;

Step 8. Stop: $x^{k,\ell}$ is an ε -stationary point of problem (2).

5. Problem with Equality and Inequality Constraints

Similarly to the equality constraints, let us introduce vector-function $\hat{h} : \mathbb{R}^n \rightarrow N\hat{m}$:

$$\hat{h} = \begin{pmatrix} h^1(x^1) \\ \vdots \\ h^N(x^N) \end{pmatrix} \tag{17}$$

and expansion of the Laplacian matrix

$$\hat{L} = L \otimes I^{\hat{m}}. \tag{18}$$

Then, the new optimization problem has the form

$$\min_{x \in \mathbb{R}^n, y \in \mathbb{R}^{N\hat{m}}, z \in \mathbb{R}^{N\hat{m}}} \sum_{i=1}^N f^i(x^i), \tag{19a}$$

$$\tilde{g}(x) + \tilde{L}y = 0, \tag{19b}$$

$$\hat{h}(x) + \hat{L}z \leq 0. \tag{19c}$$

Firstly, let us prove the following lemma.

Lemma 2. *The following statements are correct:*

1. Problem (19) is feasible if problem (1) is feasible.
2. Triplet (x^*, y^*, z^*) is the solution to problem (19) if and only if x^* is the solution to (1).

Proof. In Lemma 1, it was shown that linear constraints in both problems have the same solution in x . Let us now consider inequality constraints in both problems. As before, we consider the constraint j from (1c)

$$\sum_{i=1}^N h_j^i(x^i) \leq 0 \tag{20}$$

and the set of corresponding constraints $c(j)$ from (19c)

$$\text{vec}\left(h_j^1(x^1), \dots, h_j^N(x^N)\right) + Lz \leq 0. \tag{21}$$

The sum of the rows of L is zero. Thus, the sum of the inequalities (21) yields (20). Let us now show that if for some x equation (21) is correct, then there always exists x such that (20) holds. Vector $\text{vec}\left(h_j^1(x^1), \dots, h_j^N(x^N)\right)$ can always be decomposed using some orthogonal basis $\mathbf{1}_N, q^2, \dots, q^N$:

$$\text{vec}\left(h_j^1(x^1), \dots, h_j^N(x^N)\right) = \alpha \mathbf{1}_N + \sum_{i=2}^N \beta_i q^i = \alpha \mathbf{1}_N + q. \tag{22}$$

Vector q is orthogonal to $\mathbf{1}_N$ and, consequently, is orthogonal to $\ker L$. Thus, there always exists z such that $Lz = -q$. Substitution of such z into left-hand side of (21) gives

$$\text{vec}\left(h_j^1(x^1), \dots, h_j^N(x^N)\right) + Lz = \alpha \mathbf{1}_N + q + Lz = \alpha \mathbf{1}. \tag{23}$$

Additionally,

$$\sum_{i=1}^N h_j^i(x^i) = \text{vec}\left(h_j^1(x^1), \dots, h_j^N(x^N)\right)^\top \mathbf{1} = (\alpha \mathbf{1}_N + q)^\top \mathbf{1} = \alpha. \tag{24}$$

Thus, from (20) $\alpha \leq 0$ and (21), since this statement holds for all $j \in \{1, \dots, \hat{m}\}$, the lemma is proven. \square

6. Newton’s Method

Here we adapt a Newton-type approach to distributed optimization [37]. In order to carry this out, we have to derive algorithms for the initial problem (1) and the distributed problem (19) in parallel. Due to the similar structure of these problems, all variables and functions of the initial problem will be denoted with an upper index c , which stands for centralized.

Let us introduce Lagrange functions for problem (19):

$$V(x, y, \lambda, \mu) = f(x) + \lambda^\top (\tilde{g}(x) + \tilde{L}y) + \mu^\top (\hat{h}(x) + \hat{L}z). \tag{25}$$

The corresponding Karush–Kuhn–Tucker conditions have the form

$$\frac{\partial V}{\partial x} = \nabla f(x) + (J\tilde{g}(x))^\top \lambda + (J\hat{h}(x))^\top \mu = 0, \tag{26a}$$

$$\frac{\partial V}{\partial y} = \tilde{L}\lambda = 0 \tag{26b}$$

$$\frac{\partial V}{\partial \mu} = \hat{L}z = 0, \tag{26c}$$

$$\tilde{g}(x) + \tilde{L}y = 0, \tag{26d}$$

$$(\hat{h}(x) + \hat{L}z)_i \mu_i = 0, \mu \geq 0. \tag{26e}$$

In order to replace the complimentary slackness conditions with equations suitable for the Newton method, the complementarity function $\psi : \mathbb{R}^2 \rightarrow \mathbb{R}$ is introduced. It has the following property: $\psi(x, y) = 0$ if and only if $x \geq 0, y \geq 0$ and $xy = 0$. It can be chosen in multiple ways. Here, we use the following form:

$$\psi(x, y) = \begin{cases} y, & \text{if } x > 0 \text{ or } y \geq 0, \\ -x, & \text{otherwise.} \end{cases} \tag{27}$$

Then, the KKT conditions (26) can be replaced with

$$\phi(x, y, z, \lambda, \mu) = 0, \tag{28}$$

where

$$\phi(x, y, z, \lambda, \mu) = \begin{pmatrix} \frac{\partial V}{\partial x} \\ \tilde{L}\lambda \\ \tilde{L}\mu \\ \tilde{g}(x) + \tilde{L}y \\ \psi(\mu, \hat{h}(x) + \hat{L}z) \end{pmatrix}. \tag{29}$$

Next, we introduce diagonal matrices $\mathcal{A}(x, \mu)$ with elements

$$\mathcal{A}_{ii}(x, \mu) = \begin{cases} 1, & \psi_i(\mu, \hat{h}(x) + \hat{L}z) = \hat{h}(x) + \hat{L}z, \\ 0, & \text{otherwise;} \end{cases}$$

and $\mathcal{B}(x, \mu)$ with elements

$$\mathcal{B}_{ii}(x, \mu) = \begin{cases} 1, & \psi_i(\mu, \hat{h}(x)) = -\mu, \\ 0, & \text{otherwise.} \end{cases}$$

Then,

$$\Phi(x, y, z, \lambda, \mu) = \begin{pmatrix} \frac{\partial^2 V}{\partial x^2} & 0 & 0 & (J\tilde{g}(x))^\top & (J\hat{h}(x))^\top \\ 0 & 0 & 0 & \tilde{L} & 0 \\ 0 & 0 & 0 & 0 & \hat{L} \\ J\tilde{g}(x) & \tilde{L} & 0 & 0 & 0 \\ \mathcal{A}J\hat{h}(x) & 0 & \mathcal{B}\hat{L} & 0 & \mathcal{B} \end{pmatrix} \tag{30}$$

and the values $x^{k+1}, y^{k+1}, z^{k+1}, \lambda^{k+1}, \mu^{k+1}$, corresponding to the k -th Newton iteration step, are calculated as the solution to the following system:

$$\Phi^k(\text{vec}(x, y, z, \lambda, \mu) - \text{vec}(x^{k+1}, y^{k+1}, z^{k+1}, \lambda^{k+1}, \mu^{k+1})) = \phi^k, \tag{31}$$

where

$$\Phi^k = \Phi(x^k, y^k, z^k, \lambda^k, \mu^k), \quad \phi^k = \phi(x^k, y^k, z^k, \lambda^k, \mu^k). \tag{32}$$

Let us introduce the same Newton step equations for initial problem (1). For the Lagrange function, we have

$$V^c(x^c, \lambda^c, \mu^c) = f(x^c) + \lambda^{c\top} g(x^c) + \mu^{c\top} h(x^c) \tag{33}$$

and the corresponding parameters have the form

$$\phi^c(x^c, \lambda^c, \mu^c) = \begin{pmatrix} \frac{\partial V^c}{\partial x^c} \\ G(x^c) \\ \psi(\mu^c, H(x^c)) \end{pmatrix}, \tag{34a}$$

$$\Phi^c(x^c, \lambda^c, \mu^c) = \begin{pmatrix} \frac{\partial^2 V^c}{\partial x^{c2}} & (JG(x))^T & (JH(x))^T \\ JG(x^c) & 0 & 0 \\ \mathcal{A}(x^c, \mu^c)JH(x^c) & 0 & \mathcal{B}(x^c, \mu^c) \end{pmatrix}. \tag{34b}$$

Finally, with

$$\Phi^{c,k} = \Phi(x^{c,k}, \lambda^{c,k}, \mu^{c,k}), \phi^{c,k} = \phi(x^{c,k}, \lambda^{c,k}, \mu^{c,k}) \tag{35}$$

for a given Newton step, we have

$$\Phi^{c,k} \left(\text{vec}(x, y, z, \lambda, \mu) - \text{vec}(x^{c,k+1}, y^{c,k+1}, z^{c,k+1}, \lambda^{c,k+1}, \mu^{c,k+1}) \right) = \phi^{c,k}. \tag{36}$$

Let us now prove the following result.

Theorem 1. *If the following conditions hold*

1. $x^0 = x^{c,0}$;
2. $\lambda_{c(i)}^0 = \mathbf{1}\lambda_i^{c,0}/N$ for $i \in \{1, \dots, \tilde{m}\}$;
3. $\mu_{c(i)}^0 = \mathbf{1}\mu_i^{c,0}/N$ for $i \in \{1, \dots, \hat{m}\}$;
4. For all $i \in \{1, \dots, \hat{m}\}$ $z_{c(i)}^0$ is solution of the following optimization problem

$$\min_{t \in \mathbb{R}, z_{c(i)} \in \mathbb{R}^N} \frac{1}{2} t^\top t, \tag{37a}$$

$$\widehat{g}_{c(i)}(x^{0,i}) + Lz_{c(i)} + t = 0, \tag{37b}$$

then the convergence of (31) coincides with the convergence of (36).

Proof. If, for the iteration k , conditions 1–4 hold, then, from (31), we arrive at a system of linear equations. Using Δ to denote the difference between variables at the k -th iteration, Equation (31) can be rewritten as

$$\frac{\partial^2 V}{\partial x^2} \Delta x + (JG(x^k))^T \Delta \lambda + (JH(x^k))^T \Delta \mu = \frac{\partial V}{\partial x}, \tag{38a}$$

$$\widetilde{L} \Delta \lambda = \widetilde{L} \lambda^k, \tag{38b}$$

$$\widehat{L} \Delta \mu = \widehat{L} \mu^k, \tag{38c}$$

$$J\widetilde{g}(x^k) \Delta x + \widetilde{L} \Delta y = \widetilde{g}(x^k) + \widetilde{L} y^k, \tag{38d}$$

$$\mathcal{A}(x^k, \mu^k) \widehat{h}(x^k) \Delta x + \mathcal{A}(x^k, \mu^k) \widehat{L} \Delta z + \mathcal{B}(x^k, \mu^k) \mu^k = \psi(\mu^k, \widehat{h}(x^k) + \widehat{L} z^k). \tag{38e}$$

This set of equations describes a stationary point in the optimization problem

$$\min_{\Delta x \in \mathbb{R}^n, \Delta y \in \mathbb{R}^{\tilde{m}N}, \Delta z \in \mathbb{R}^{\hat{m}N}} \frac{1}{2} \Delta x^\top \frac{\partial^2 V}{\partial x^2} \Delta x + \Delta x^\top \frac{\partial V}{\partial x}, \tag{39a}$$

$$J\widetilde{g}(x^k) \Delta x + \widetilde{L} \Delta y = \widetilde{g}(x^k) + \widetilde{L} y^k, \tag{39b}$$

$$\left(J\widehat{h}(x^k) \Delta x + \widehat{L} \Delta z \right)_{\mathcal{J}} = \left(\widehat{h}(x^k) + \widehat{L} z^k \right)_{\mathcal{J}'}, \tag{39c}$$

where $\mathcal{J} = \{i \in \{1, \dots, \hat{m}N\} \mid \mathcal{A}_{ii}(x^k, \mu^k) = 1\}$.

Likewise, for the centralized Equation (36), we can obtain a similar optimization problem:

$$\min_{\Delta x^c \in \mathbb{R}^n} \frac{1}{2} \Delta x^{c\top} \frac{\partial^2 V^c}{\partial x^{c2}} \Delta x^c + \Delta x^{c\top} \frac{\partial V^c}{\partial x^c}, \tag{40a}$$

$$J\widetilde{g}(x^{c,k}) \Delta x^c = \widetilde{g}(x^{c,k}), \tag{40b}$$

$$\left(J\hat{h}(x^{c,k})\Delta x^c \right)_I = \left(\hat{h}(x^{c,k}) \right)_I, \tag{40c}$$

where $I = \{i \in \{1, \dots, \tilde{m}\} \mid \mathcal{A}a_{ii}(x^{c,k}, \mu^{c,k}) = 1\}$. Let us now show that (39) is an expansion of problem (40). Consider objective functions in both problems, which have first and second derivatives of the corresponding Lagrange functions. For the first derivative, we have

$$\frac{\partial V}{\partial x} = \nabla f(x^k) + (J\tilde{g}(x^k))^\top \lambda + (J\hat{h}(x^k))^\top \mu. \tag{41}$$

Note that due to condition 2,

$$\left((J\tilde{g}(x^k))^\top \lambda^k \right)_i = \sum_{k=1}^{\tilde{m}} \sum_{j \in c(i)} \frac{\partial g^k}{\partial x^i} \lambda_k^j = \sum_{k=1}^{\tilde{m}} \sum_{j \in c(i)} \frac{\partial g^k}{\partial x^i} \frac{\lambda_k^c}{\tilde{m}} = \lambda_k^c \sum_{k=1}^{\tilde{m}} \sum_{j \in c(i)} \frac{\partial g^k}{\partial x^i} = \left((Jg(x^k))^\top \lambda^{k,c} \right). \tag{42}$$

The same equality holds for $((J\hat{h}(x^k))^\top \mu^k)$. Thus,

$$\frac{\partial V}{\partial x} = \frac{\partial V^c}{\partial x^c} \tag{43}$$

and, consequently,

$$\frac{\partial^2 V}{\partial x^2} = \frac{\partial^2 V^c}{\partial x^{c2}}. \tag{44}$$

As a result, the objective functions in problems (40) and (39) are equivalent. Let us now consider the relation between sets J and I . Firstly, we focus on the optimization problem (37). It can be shown that in its optimum, $z_{c(i)}$ and t are chosen so that $t_i = t_j$, since it is the only case, where $t \in \ker L$. Thus, for each i , the corresponding inequality constraints from (40) and (39),

$$\sum_{j=1}^N g_i^j(x^{c,j}) \leq 0 \tag{45}$$

and

$$\tilde{g}_{c(i)}(x) + Lz_{c(i)} \leq 0 \tag{46}$$

are all active or inactive simultaneously. Thus,

$$\mathcal{J} = \bigcup_{i \in I} c(i). \tag{47}$$

As a result, problem (39) is an expansion of problem (40), and according to Lemma 1, has the same solution in x . Moreover, it means that for $k + 1$, item 1 of the lemma is satisfied. Let us now demonstrate that the values $x^{k+1}, y^{k+1}, z^{k+1}, \lambda^{k+1}, \mu^{k+1}$ satisfy items 2–4 of the Lemma. From (38b), for each $i \in \{1, \dots, \tilde{m}\}$, all components of $\Delta \lambda_{c(i)}^{k+1}$ are equal to each other, which gives item 2. The same approach applies for all $\mu_{c(i)}^{k+1}, i \in I$, and for all other $i \mu_c^{k+1} i = 0$, which means that item 3 holds. Finally, for $z_{c(i)}^{k+1}, i \in I$, the corresponding constraint is active, and problem (37) is solved with $t = 0$. For all other i , we have $z_{c(i)}^{k+1}$, and therefore, item 4 holds. \square

Corollary 1. *The convergence speed of Newton’s method applied to problem (19) is equal to the convergence speed of Newton’s method applied to problem (1).*

Corollary 2. *Assume that*

- *functions f, g and h are twice differentiable in some neighborhood of the solution x^* of problem (1), and their second derivatives are Lipschitz-continuous in the neighborhood of the x^* .*
- *the constraints’ gradients are linear independent in the optimum (linear independence constraint qualification);*

- solution x^* has unique corresponding dual variables $\lambda^{c,*}$ and $\mu^{c,*}$ in problem (1);
- for x^* , $\lambda^{c,*}$ and $\mu^{c,*}$ we have

$$u^\top \frac{\partial^2 L^c}{\partial x^2}(x^*, \lambda^{c,*}, \mu^{c,*}) > 0 \quad \forall u \in K_+(x^*) \setminus \{0\}, \tag{48}$$

where

$$K_+(x^*) = \{u \in \ker(Jh(x^*)) \mid (Jg(x^*))_i u = 0 \quad \forall i : g_i(x^*) = 0 \text{ and } \mu_i^{c,*} > 0\}. \tag{49}$$

Then, in problem (19), for any starting point (x, λ, μ) sufficiently close to (x^*, λ^*, μ^*) , where $\lambda_{c(i)}^0 = \mathbf{1}\lambda_i^{c,0}/N$ for $i \in \{1, \dots, \tilde{m}\}$ and $\mu_{c(i)}^0 = \mathbf{1}\mu_i^{c,0}/N$ for $i \in \{1, \dots, \hat{m}\}$, Algorithm 2 converges to the solution with quadratic rate.

This corollary result is based on the estimation of Newton’s method convergence rate for problem (1) given in [37].

Finally, Algorithm 2 requires the exchange of information only in steps 6 and 8. However, during this step, the gradient descent method is used with its distributed implementation shown in the previous section. Thus, operations in Algorithm 2 are performed in the distributed form.

Algorithm 2 Newton’s method

Input: $f, g, L, \varepsilon > 0$.

Output: x^*

Algorithm steps:

Step 1. Set $y^0 = 0$;

Step 2. For each $i \in \{1, \dots, N\}$ set $z_{c(i)}$ as a solution of (37) using gradient descent;

Step 3. Set $\lambda^0 = 0$ and $\mu^0 = 0$;

Step 4. Set $k = 0$;

Step 5. Solve optimization problem (39) using the gradient descent method;

Step 6. Assign

$$\begin{pmatrix} x^{k+1} \\ y^{k+1} \\ z^{k+1} \\ \lambda^{k+1} \\ \mu^{k+1} \end{pmatrix} = \begin{pmatrix} \Delta x^k \\ \Delta y^k \\ \Delta z^k \\ \Delta \lambda^k \\ \Delta \mu^k \end{pmatrix} + \begin{pmatrix} x^k \\ y^k \\ z^k \\ \lambda^k \\ \mu^k \end{pmatrix} \tag{50}$$

Step 7. For all inactive constraints $i \in \{1, \dots, \hat{m}\}$, solve problem (37) using gradient descent and assign its solution to $z_{c(i)}^{k+1}$;

Step 8. If $\|x^k - x^{k+1}\| > \varepsilon$, then set $k = k + 1$ and go back to step 5;

Step 9. Stop: x^{k+1} is an ε -stationary point.

Example 2. Consider the initial problem with the following components: $N = 4, \tilde{m} = 1, \hat{m} = 0, n_1 = n_2 = n_3 = n_4 = 1, f^1(x^1) = (x_1^1 - 5)^2, f^2(x^2) = (x_1^2 - 4)^2, f^3(x^3) = (x_1^3 - 3)^2, f^4(x^4) = (x_1^4 - 2)^2, g^1(x^1) = (x_1^1)^2 - 3, g^2(x^2) = (x_1^2)^2 - 3, g^3(x^3) = (x_1^3)^2 - 3, g^4(x^4) = (x_1^4)^2 - 3$. The network is described by the Laplacian matrix (as in example 1):

$$L = \begin{pmatrix} 2 & -1 & 0 & -1 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ -1 & 0 & -1 & 2 \end{pmatrix}.$$

Starting points $x_1^{1,0} = 2, x_1^{2,0} = 2, x_1^{3,0} = 2, x_1^{4,0} = 2, y_1^{1,0} = 0, y_1^{2,0} = 0, y_1^{3,0} = 0, y_1^{4,0} = 0, \lambda^{1,0} = 0.4, \lambda_2^{1,0} = 0.4, \lambda^{3,0} = 0.4, \lambda_1^{4,0} = 0.4$. The tolerance $\varepsilon = 0.01$. Then, in four iterations,

Algorithm 2 finds an ε -optimal solution with components $x_1^{1,4} = 2.357, x_1^{2,4} = 1.886, x_1^{3,4} = 1.414, x_1^{4,4} = 0.943, y_1^{1,4} = -1.083, y_1^{2,4} = -0.472, y_1^{3,4} = 0.694, y_1^{4,4} = 0.861, \lambda_1^{1,4} = \lambda_1^{2,4} = \lambda_1^{3,4} = \lambda_1^{4,4} = 1.121$. The objective function value $F(x^4) = 15.088$.

7. Application of the Arrow–Hurwicz Algorithm to Problems with Inequality Constraints

We consider here problem (19) without equality constraints

$$\min_{x \in \mathbb{R}^n, z \in \mathbb{R}^{N\hat{m}}} \sum_{i=1}^N f^i(x^i), \tag{51a}$$

$$\hat{h}(x) + \hat{L}z \leq 0. \tag{51b}$$

The Lagrange function is given by

$$V(x, z, \mu) = f(x) + \mu^\top (\hat{h}(x) + \hat{L}z). \tag{52}$$

Assume that starting vectors (x^0, z^0, μ^0) are given. The Arrow–Hurwicz algorithm can be described by the following relations ([38]):

$$x^{k+1} = x^k - \rho \frac{\partial V(x^k, z^k, \mu^k)}{\partial x}, \tag{53}$$

$$z^{k+1} = z^k - \rho \frac{\partial V(x^k, z^k, \mu^k)}{\partial z}, \tag{54}$$

$$\mu^{k+1} = \max\{0, \mu^k + \rho h(x^k)\}. \tag{55}$$

As was shown above, the computational scheme (53)–(55) with V defined in (52) has the distributed form. The algorithm stops when $\|x^k - x^{k+1}\| \leq \varepsilon$, where $\varepsilon > 0$ is the tolerance. The following strategy of choosing the step size ρ was used. We set the initial value $\rho = 1$. If, during the iterations, the deviation $h(x^k)$ is becoming large enough, for example, greater than the practically chosen value \bar{h} , then ρ is reset: $\rho = \frac{\rho}{2}$, and the algorithms restart from the initial starting set (x^0, z^0, μ^0) . The explanation of such a restarting is the following. If the deviation $\hat{h}(x^k)$ is big enough, then the point x^k is too far from the feasible domain in contrast to starting point x^0 , which is assumed to be chosen close enough to the feasible domain. The first and main reason of using this algorithm is due to the fact that it provides a minimization procedure in the non-convex case. The second reason consists of the following. In some neighborhood of the point of minimum, the Lagrange function is usually locally convex, and if the algorithm managed to get into this neighborhood, then it determines this point of minimum.

Example 3. Problem (51) has the following components: $N = 4, \hat{m} = 2, n_1 = 2, n_2 = 1, n_3 = 3, n_4 = 2, f_1(x_1^1, x_2^1) = (x_1^1)^2(x_2^1)^2, f_2(x_1^2) = 2 \sin((x_1^2 - 3)x_1^2), f_3(x_1^3, x_2^3, x_3^3) = (x_1^3)^2 + (x_2^3)^2 + (x_3^3)^2, f_4(x_1^4, x_2^4) = (x_1^4 - x_2^4 - 1)^2, h_1^1(x_1^1, x_2^1) = x_1^1 + x_2^1 - 3, h_2^1(x_1^1, x_2^1) = (x_1^1)^2 + (x_2^1)^2 - 5, h_1^2(x_1^2) = (x_1^2)^2 - 4, h_1^3(x_1^3, x_2^3, x_3^3) = x_1^3 + x_2^3 + x_3^3 - 3, h_2^3(x_1^3, x_2^3, x_3^3) = x_3^3 - 2, h_1^4(x_1^4, x_2^4) = x_1^4 - x_2^4, h_2^4(x_1^4, x_2^4) = (x_1^4)^2 + (x_2^4)^2 - 1$.

The interpretation of the agent network is shown in Figure 2. The Laplacian matrix is

$$L = \begin{pmatrix} 2 & -1 & 0 & -1 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ -1 & 0 & -1 & 2 \end{pmatrix}.$$

All components of the starting vectors x^0, z^0, μ^0 were equal to 2. Tolerance $\varepsilon = 0.001$. After 69 iterations, the algorithm determined point $x^{69} = (-0.162, -0.162, 0.676, 0, 0, 0, -0.119, 1.119)$, which happened to be optimal, $y^{69} = (3.108, 3.296, 2.525, 2.466, 1.940, 1.446, 0.427, 0.791)$, $F(x^{69}) = -1.999$.

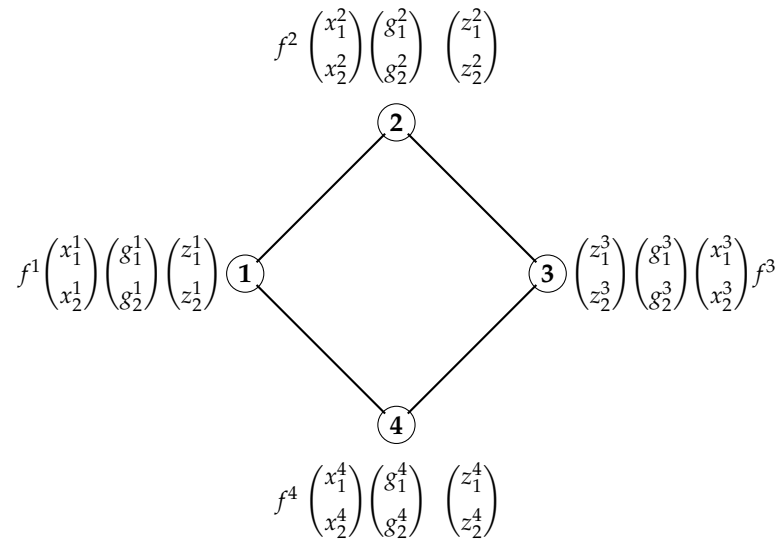


Figure 2. The agent network with information available at each node.

8. Conclusions

A novel approach for the decentralized solution of non-convex optimization problems was proposed. It is based on the reformulation of optimization problems to a specific form that allows the distributed implementation of modified gradient descent and Newton’s methods. The main strength of the modified Newton’s method is in having the same number of oracle callers as a standard Newton’s method applied to the initial problem formulation. Thus, in the cases when oracle calls are more expensive than communication between agents, the transition from centralized to distributed paradigm does not significantly affect computational time. Moreover, if the convergence speed for Newton’s method in application to centralized problems is quadratic, the same speed will remain for the modified decentralized algorithm.

Such properties of the proposed approach are extremely useful in solving optimization problems that arise when automating the design of decision support systems and digital twins based on a holistic approach that uses mathematical and machine learning models in conjunction with human expert knowledge [5], especially in the context of ubiquitous computing and digitalization [1].

Author Contributions: Conceptualization, O.O.K., O.V.K. and E.S.S.; methodology, O.O.K., O.V.K. and E.S.S.; validation, O.O.K., O.V.K., T.D.G. and E.S.S.; formal analysis O.V.K., T.D.G. and E.S.S.; investigation, E.S.S.; writing—original draft preparation, O.O.K. and O.V.K.; writing—review and editing, O.V.K., T.D.G. and E.S.S.; supervision, E.S.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Ganchev, T.D. Chapter 8—Ubiquitous computing and biodiversity monitoring. In *Advances in Ubiquitous Computing*; Neustein, A., Ed.; Advances in Ubiquitous Sensing Applications for Healthcare; Academic Press: Cambridge, MA, USA, 2020; pp. 239–259. [CrossRef]

2. Ganchev, T.; Markova, V.; Valcheva-Georgieva, I.; Dobrev, I. Assessment of pollution with heavy metals and petroleum products in the sediments of Varna Lake. *Rev. Bulg. Geol. Soc.* **2022**, *83*, 3–9. [[CrossRef](#)]
3. Nandal, A.; Zhou, L.; Dhaka, A.; Ganchev, T.; Nait-Abdesselam, F. *Machine Learning in Medical Imaging and Computer Vision*; IET: London, UK, 2024.
4. Markova, V.; Ganchev, T.; Filkova, S.; Markov, M. MMD-MSD: A Multimodal Multisensory Dataset in Support of Research and Technology Development for Musculoskeletal Disorders. *Algorithms* **2024**, *17*, 187. [[CrossRef](#)]
5. Semenkin, E. Computational Intelligence Algorithms based Comprehensive Human Expert and Data driven Model Mining for the Control, Optimization and Design of Complicated Systems. *Int. J. Inf. Technol. Secur.* **2019**, *11*, 63–66.
6. Semenkin, E.; Semenkina, M. Artificial neural networks design with self-configuring genetic programming algorithm. In Proceedings of the Bioinspired Optimization Methods and Their Applications, Maribor, Slovenia, 17–18 November 2012; pp. 291–300.
7. Akhmedova, S.; Semenkina, M.; Stanovov, V.; Semenkin, E. Semi-supervised Data Mining Tool Design with Self-tuning Optimization Techniques. In *Proceedings of the Informatics in Control, Automation and Robotics: 14th International Conference, ICINCO 2017 Madrid, Spain, 26–28 July 2017*; Revised Selected Papers; Springer: Berlin/Heidelberg, Germany, 2020; pp. 87–105.
8. Sherstnev, P.; Semenkin, E. Application of evolutionary algorithms for the design of interpretable machine learning models in classification problems. *Control Syst. Inf. Technol.* **2022**, *22*, 17–20. [[CrossRef](#)]
9. Vakhnin, A.; Sopov, E.; Semenkin, E. On Improving Adaptive Problem Decomposition Using Differential Evolution for Large-Scale Optimization Problems. *Mathematics* **2022**, *10*, 4297. [[CrossRef](#)]
10. Krutikov, V.; Gutova, S.; Tovbis, E.; Kazakovtsev, L.; Semenkin, E. Relaxation Subgradient Algorithms with Machine Learning Procedures. *Mathematics* **2022**, *10*, 3959. [[CrossRef](#)]
11. Nedic, A.; Ozdaglar, A. Distributed Subgradient Methods for Multi-Agent Optimization. *IEEE Trans. Autom. Control* **2009**, *54*, 48–61. [[CrossRef](#)]
12. Metelev, D.; Beznosikov, A.; Rogozin, A.; Gasnikov, A.; Proskurnikov, A. Decentralized optimization over slowly time-varying graphs: Algorithms and lower bounds. *Comput. Manag. Sci.* **2024**, *21*, 8. [[CrossRef](#)]
13. Nedić, A.; Olshevsky, A.; Shi, W. Achieving Geometric Convergence for Distributed Optimization Over Time-Varying Graphs. *Siam J. Optim.* **2017**, *27*, 2597–2633. [[CrossRef](#)]
14. Tang, Y.; Deng, Z.; Hong, Y. Optimal Output Consensus of High-Order Multiagent Systems With Embedded Technique. *IEEE Trans. Cybern.* **2019**, *49*, 1768–1779. [[CrossRef](#)] [[PubMed](#)]
15. Jadbabaie, A.; Lin, J.; Morse, A. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Trans. Autom. Control* **2003**, *48*, 988–1001. [[CrossRef](#)]
16. Luan, X.; Schutter, B.; Meng, L.; Corman, F. Decomposition and distributed optimization of real-time traffic management for large-scale railway networks. *Transp. Res. Part Methodol.* **2020**, *141*, 72–97. [[CrossRef](#)]
17. Luan, X.; Schutter, B.; van den Boom, T.; Corman, F.; Lodewijks, G. Distributed optimization for real-time railway traffic management. *IFAC-PapersOnLine* **2018**, *51*, 106–111. [[CrossRef](#)]
18. Khamisov, O.O. Direct disturbance based decentralized frequency control for power systems. In Proceedings of the 2017 IEEE 56th Annual Conference on Decision and Control (CDC), Melbourne, Australia, 12–15 December 2017; pp. 3271–3276. [[CrossRef](#)]
19. Khamisov, O.O.; Chernova, T.; Bialek, J.W. Comparison of two schemes for closed-loop decentralized frequency control and overload alleviation. In Proceedings of the 2019 IEEE Milan PowerTech, Milan, Italy, 23–27 June 2019; pp. 1–6. [[CrossRef](#)]
20. Motta, V.N.; Anjos, M.; Gendreau, M. Optimal allocation of demand response considering transmission system congestion. *Comput. Manag. Sci.* **2020**, *20*, 2023.
21. Rabbat, M.; Nowak, R. Distributed optimization in sensor networks. In Proceedings of the Third International Symposium on Information Processing in Sensor Networks, IPSN 2004, Berkeley, CA, USA, 26–27 April 2004; pp. 20–27. [[CrossRef](#)]
22. Sadiev, A.; Borodich, E.; Beznosikov, A.; Dvinskikh, D.; Chezhegov, S.; Tappenden, R.; Takac, M.; Gasnikov, A. Decentralized personalized federated learning: Lower bounds and optimal algorithm for all personalization modes. *Euro J. Comput. Optim.* **2022**, *10*, 100041. [[CrossRef](#)]
23. Forero, P.A.; Cano, A.; Giannakis, G.B. Consensus-Based Distributed Support Vector Machines. *J. Mach. Learn. Res.* **2010**, *11*, 1663–1707.
24. Gorbunov, E.; Rogozin, A.; Beznosikov, A.; Dvinskikh, D.; Gasnikov, A., Recent Theoretical Advances in Decentralized Distributed Convex Optimization. In *High-Dimensional Optimization and Probability: With a View Towards Data Science*; Nikeghbali, A., Pardalos, P.M., Raigorodskii, A.M., Rassias, M.T., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 253–325. [[CrossRef](#)]
25. Kovalev, D.; Salim, A.; Richtarik, P. Optimal and Practical Algorithms for Smooth and Strongly Convex Decentralized Optimization. In Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–12 December 2020; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., Eds.; Curran Associates, Inc.: New York, NY, USA, 2020; Volume 33, pp. 18342–18352.
26. Kovalev, D.; Shulgin, E.; Richtarik, P.; Rogozin, A.V.; Gasnikov, A. ADOM: Accelerated Decentralized Optimization Method for Time-Varying Networks. In Proceedings of the 38th International Conference on Machine Learning, Virtual Event, 18–24 July 2021; Meila, M., Zhang, T., Eds.; PMLR: Westminster, UK, 2021; Volume 139, Proceedings of Machine Learning Research; pp. 5784–5793.

27. Wang, Z.; Ong, C.J.; Hong, G.S. Distributed Model Predictive Control of linear discrete-time systems with coupled constraints. In Proceedings of the 2016 IEEE 55th Conference on Decision and Control (CDC), Las Vegas, NV, USA, 12–14 December 2016; pp. 5226–5231. [\[CrossRef\]](#)
28. Erseghe, T. Distributed Optimal Power Flow Using ADMM. *IEEE Trans. Power Syst.* **2014**, *29*, 2370–2380. [\[CrossRef\]](#)
29. Yarmoshik, D.; Rogozin, A.; Khamisov, O.O.; Dvurechensky, P.; Gasnikov, A. Decentralized Convex Optimization Under Affine Constraints for Power Systems Control. In Proceedings of the 21st International Conference Mathematical Optimization Theory and Operations Research, Petrozavodsk, Russia, 2–6 July 2022; Pardalos, P., Khachay, M., Mazalov, V., Eds.; Springer: Cham, Switzerland, 2022; pp. 62–75.
30. Khamisov, O.O. Distributed continuous-time optimization for convex problems with coupling linear inequality constraints. *Math. Optim. Theory Oper. Res.* **2024**, *21*, 1619–6988. [\[CrossRef\]](#)
31. Stomberg, G.; Engelmann, A.; Faulwasser, T. Decentralized non-convex optimization via bi-level SQP and ADMM. In Proceedings of the 2022 IEEE 61st Conference on Decision and Control (CDC), Cancun, Mexico, 6–9 December 2022; pp. 273–278. [\[CrossRef\]](#)
32. Hours, J.H.; Jones, C.N. A Parametric Nonconvex Decomposition Algorithm for Real-Time and Distributed NMPC. *IEEE Trans. Autom. Control* **2016**, *61*, 287–302. [\[CrossRef\]](#)
33. Zhao, C.; Topcu, U.; Li, N.; Low, S. Design and Stability of Load-Side Primary Frequency Control in Power Systems. *IEEE Trans. Autom. Control* **2014**, *59*, 1177–1189. [\[CrossRef\]](#)
34. Shores, T. *Applied Linear Algebra and Matrix Analysis*; Springer : New York, NY, USA, 2007.
35. Evtushenko, Y. *Numerical Optimization Technique*; Springer: New York, NY, USA, 1985.
36. Hadley, G. *Nonlinear and Dynamic Programming*; Addison-Wesley Publishing Company Inc.: Boston, MA, USA, 1964.
37. Izmailov, A.F.; Solodov, M.V. *Newton-Type Methods for Optimization and Variational Problems*; Springer: Cham, Switzerland, 2014.
38. Minoux, M. *Programmation Mathématique: Théorie et Algorithmes*; Dunod: Paris, France, 1983.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.