INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2025, VOL. 71, NO. 4, PP. 1–10
Manuscript received September 4, 2025; revised October 2025. doi: 10.24425/ijet.2025.155472

Parallel and distributed Machine Learning on Augmented Lagrangian Algorithms

Anthony Nwachukwu, and Andrzej Karbowski

***Abstract*—Constrained optimization is central to large-scale machine learning, particularly in parallel and distributed environments. This paper presents a comprehensive study of augmented Lagrangian–based algorithms for such problems, including classical Lagrangian relaxation, the method of multipliers, the Alternating Direction Method of Multipliers (ADMM), Bertsekas' algorithm, Tatjewski's method, and the Separable Augmented Lagrangian Algorithm (SALA). We develop a unified theoretical framework, analyze convergence properties and decomposition strategies, and evaluate these methods on two representative classes of tasks: regularized linear systems and K-means clustering. Numerical experiments on synthetic and real-world datasets show that Bertsekas' method consistently achieves the best balance of convergence speed and solution quality, while ADMM offers practical scalability under decomposition but struggles in high-dimensional or ill-conditioned settings. Tatjewski's method benefits significantly from partitioning, whereas the classical Augmented Lagrangian approach proves computationally inefficient for large-scale problems. These findings clarify the trade-offs among augmented Lagrangian algorithms, highlighting Bertsekas' method as the most effective for distributed optimization and providing guidance for algorithm selection in large-scale machine learning applications.**

***Keywords*—Augmented Lagrangian, Optimization, Machine Learning, Alternating Direction Method of Multipliers, Parallel Computing, Convex and Non-convex Optimization, ADMM, Distributed Computing, Clustering, Support Vector Machine, Regression**

I. Introduction

MODERN machine learning (ML) tasks frequently involve optimizing high-dimensional models under explicit constraints, such as parameter bounds, resource limitations, or fairness criteria. Examples include risk minimization with regularization, network flow, and structured prediction. To meet the demands of scale and privacy, data and computations are often distributed across multiple processors or agents, requiring parallel algorithms for constrained optimization. In such distributed settings, classical single-machine solvers are typically inadequate due to communication bottlenecks.

Augmented Lagrangian techniques address these challenges by combining Lagrange multipliers with quadratic penalties. This formulation mitigates duality gaps while preserving separability, making it well-suited for parallel implementations. A prominent example is the Alternating Direction Method of Multipliers (ADMM), which integrates dual decomposition with augmented penalties to enable independent updates across machines, followed by consistency enforcement. ADMM has become popular in distributed ML for its simplicity and effectiveness.

Beyond ADMM, other algorithms extend this approach for distributed optimization. Ordinary Lagrangian relaxation (dual decomposition) suffers from duality gaps and slow convergence. The classic method of multipliers improves convergence but requires coupled updates, limiting scalability. Bertsekas' algorithm introduces damped multiplier updates to improve separability and convergence speed. Tatjewski's method similarly refines decomposition with scaled multipliers. The Separable Augmented Lagrangian Algorithm (SALA) further exploits primal reformulation and resource-based splitting for parallel efficiency.

To provide a rigorous foundation for this work, we consider the constrained optimization problem:

$$\min_{x} \quad f(x) \tag{1}$$

$$\text{s.t.} \quad h(x) = 0 \tag{2}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $h : \mathbb{R}^n \to \mathbb{R}^m$ with $m \leq n$. Our focus is on parallelizable algorithms for solving (1), (2) when the structure is separable, enabling distributed computation.

In this paper, we present a comprehensive comparison of six methods for solving such problems in a distributed setting: ordinary Lagrangian relaxation, the classical augmented Lagrangian (multiplier) method, ADMM, Bertsekas' algorithm, Tatjewski's method, and the SALA scheme. We examine their theoretical properties, including convergence behavior and duality gap management, and discuss the practical implications of their update rules, step-size adjustments, and penalty parameters. Each method's ability to decompose global objectives into local subproblems is critically analyzed in the context of parallel ML tasks.

Our analysis is complemented by numerical experiments on distributed clustering and sparse recovery problems, which serve to illustrate the practical trade-offs and validate the theoretical considerations discussed. Through this unified treatment and comparative evaluation, we aim to clarify the roles of these augmented Lagrangian algorithms within the broader landscape of distributed ML optimization and to inform the

A. Nwachukwu and A. Karbowski are with Faculty of Electronics and Information Technology, Warsaw University of Technology, Warsaw, Poland (e-mail: anthonychukwuemeka.nwachukwu@gmail.com, andrzej.karbowski@pw.edu.pl).

design of scalable and efficient parallel routines for real-world applications.

II. Review of Augmented Lagrangian-Based Algorithms

The Lagrangian method, introduced by Arrow et al. [1], reformulates constrained optimization as:

$$L(x,\lambda) = f(x) + \lambda' h(x), \tag{3}$$

where λ is the vector of Lagrange multipliers. The dual function follows as:

$$q(\lambda) = \min_x L(x,\lambda). \tag{4}$$

While effective for convex problems, this approach struggles with nonconvexity due to duality gaps. Hestenes [2] and Powell [3] improved this by adding a quadratic penalty:

$$L_\rho(x,\lambda) = f(x) + \lambda h(x) + \frac{\rho}{2}\|h(x)\|^2. \tag{5}$$

For separable problems, the augmented Lagrangian takes the form:

$$L_\rho(x,\lambda) = \sum_{i=1}^{N}\left[f_i(x_i) + \lambda^T h_i(x_i)\right] + \frac{\rho}{2}\left\|\sum_{i=1}^{N} h_i(x_i)\right\|^2, \tag{6}$$

introducing non-separability. Bertsekas [4] restored separability by introducing an auxiliary variable s:

$$L_\rho(x,\lambda,s) = \sum_{i=1}^{N}\left[f_i(x_i) + \frac{\rho}{2}\|s_i - x_i\|^2 + \lambda^T h_i(x_i)\right]. \tag{7}$$

Tanikawa and Mukai [5] refined this with an additional penalty term:

$$\begin{aligned} L_{\rho\beta}(x,s) = \sum_{i=1}^{N}\Big[f_i(x_i) &+ \frac{\rho}{2}\|s_i - x_i\|^2 \\ &+ \left(\lambda(s)^T + \beta h(s)^T M(s)\right) h_i(x_i)\Big]. \end{aligned} \tag{8}$$

Nwachukwu and Karbowski [6] extended Bertsekas' method by scaling λ updates with $\beta \ll 1$:

$$x_i^{k+1} = \arg\min_{x_i \in X_i} L_{\rho_i}(x_i, \lambda^k, s_i^k), \tag{9}$$

$$s_i^{k+1} = \xi s_i^k + (1-\xi) x_i^{k+1}, \tag{10}$$

$$\lambda^{k+1} = \lambda^k + \beta\rho h(x^{k+1}). \tag{11}$$

Tatjewski [7] modified the quadratic term for alternative decomposition:

$$\begin{aligned} L_\rho(x,\lambda,s) = \sum_{i=1}^{N}\Bigg[&f_i(x_i) + \lambda^T h_i(x_i) \\ &+ \frac{\rho}{2}\left\|\sum_{j\neq i} h_j(s_j) + h_i(x_i)\right\|^2\Bigg]. \end{aligned} \tag{12}$$

Another widely used approach is the Alternating Direction Method of Multipliers (ADMM) [8], which efficiently handles large-scale problems with the decision vector (x,z) and the constraint $Ax + Bz = c$:

$$x^{k+1} := \arg\min_x L_\rho(x, z^k, \lambda^k), \tag{13}$$

$$z^{k+1} := \arg\min_z L_\rho(x^{k+1}, z, \lambda^k), \tag{14}$$

$$\lambda^{k+1} := \lambda^k + \rho(Ax^{k+1} + Bz^{k+1} - c). \tag{15}$$

However, ADMM requires convexity [9], [10], making it unsuitable for our problem. A better alternative is the Separated Augmented Lagrangian Algorithm (SALA) by Hamdi et al. [11]–[13], which introduces auxiliary variables s_i:

$$\min_{x\in X,\, s} \sum_{i=1}^{N} f_i(x_i), \quad \text{s.t. } h_i(x_i) = s_i, \quad \sum_{i=1}^{N} s_i = 0. \tag{16}$$

The SALA augmented Lagrangian is:

$$\begin{aligned} L_\rho(x,s,\lambda) = \sum_{i=1}^{N}\Big[&f_i(x_i) + \lambda^T(h_i(x_i) - s_i) \\ &+ \frac{\rho}{2}\|h_i(x_i) - s_i\|^2\Big]. \end{aligned} \tag{17}$$

with updates:

$$(x^{k+1}, s^{k+1}) = \arg\min_{x,s} L_\rho(x,s,\lambda^k), \tag{18}$$

$$\lambda^{k+1} = \lambda^k + \frac{\rho_k}{N}\sum_{i=1}^{N} h_i(x_i^{k+1}), \tag{19}$$

$$\rho_{k+1} = \alpha\rho_k. \tag{20}$$

Using ADMM principles, SALA enables parallelizable updates while maintaining consistency, making it well-suited for our application.

III. Solution of Regularized Linear Systems with Augmented Lagrangian Algorithms

In various fields such as signal processing, machine learning, numerical optimization, and high-dimensional statistics, the need to obtain stable and computationally efficient solutions from linear systems has driven the study of regularized approaches [14], [15]. Consider a general linear system of the form $Ax = b$, where $A \in \mathbb{R}^{m\times n}$, $x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, and m, n are the number of samples and features respectively. Regardless of whether the system is underdetermined, overdetermined, or exactly determined, practical considerations often necessitate regularizing the solution to enhance stability, prevent overfitting, or deal with ill-conditioning in the data [16], [17].

Among the most widely used strategies is ℓ_2-regularization, introduced in the context of ridge regression by Hoerl and Kennard [18], which discourages large solution norms and improves numerical conditioning without necessarily enforcing sparsity. Unlike the ℓ_0 pseudo-norm or its convex surrogate ℓ_1, the ℓ_2 norm promotes smoothness and penalizes large coefficients, yielding well-behaved solutions that are particularly attractive in noisy or high-dimensional regimes.

To formalize this, let $P \subseteq \{1,\ldots,n\}$ index the decision variables and $M = \{1,\ldots,m\}$ index the constraints. Let

$a_i \in \mathbb{R}^{|P|}$ denote the feature vector of the i-th observation, and let $y_i \in \mathbb{R}$ denote the associated response. The regularized recovery problem can then be expressed as the following constrained ℓ_2-minimization problem:

$$\min_x \quad \sum_{j\in P} x_j^2, \quad \text{s.t.} \sum_{j\in P} a_{ij}x_j = y_i, \quad \forall i \in M \tag{21}$$

Here, $x_j \in \mathbb{R}$ for all $j \in P$ are the optimization variables, and the equality constraints enforce exact satisfaction of the observed data. The objective function imposes an ℓ_2-norm penalty, which encourages solutions with small magnitudes and improved robustness to noise and multicollinearity. This formulation underlies classical techniques such as Tikhonov regularization and ridge regression, and plays a central role in contemporary optimization-based methods for solving linear inverse problems.

The problem formulation in (21) lends itself naturally to decomposition techniques that are well-suited for large-scale and distributed optimization. This section presents several decomposition approaches for solving the regularized linear systems, each grounded in the framework of augmented Lagrangian methods and dual ascent strategies.

The key idea underlying these methods is to exploit the separability of the objective function and structure of the constraints to design efficient iterative schemes. In particular, variants of the classical Lagrangian method are considered, as well as augmented Lagrangian formulations including the Multiplier Method, Bertsekas Method, Tatjewski Method, and the SALA (Separated Augmented Lagrangian Algorithm) version of the Alternating Direction Method of Multipliers (ADMM).

These methods enable the decoupling of variables and facilitate parallel or distributed updates, making them attractive for high-dimensional optimization problems commonly encountered in compressed sensing and machine learning.

A. The Lagrangian

The Lagrangian of Problem (21) can be written as

$$\begin{aligned} L(x,\lambda) &= \sum_{j\in P}\left[x_j^2 + \sum_{i\in M}\lambda_i\Big(a_{ij}x_j - \frac{y_i}{|P|}\Big)\right] \\ &= \sum_{j\in P} L_j(x_j,\lambda) \end{aligned} \tag{22}$$

where $L_j(x_j,\lambda) = x_j^2 + \sum_{i\in M}\lambda_i\Big(a_{ij}x_j - \frac{y_i}{|P|}\Big)$. At iteration $k+1$, with $\lambda_i \in \mathbb{R}$ and $\rho_k > 0$, the dual variables are updated according to the following rule:

$$x_j^{k+1} = \min_{x_j} L_j(x_j,\lambda^k), \quad \forall j \in N \tag{23}$$

$$\lambda_i^{k+1} = \lambda_i^k + \rho_k\Big(\sum_{j\in P} a_{ij}x_j^{k+1} - y_i\Big), \quad \forall i \in M \tag{24}$$

B. The Multiplier Method

The classical augmented Lagrangian method introduces a quadratic penalty term to improve the convergence properties of the basic Lagrangian scheme. The augmented Lagrangian of Problem (21) becomes:

$$\begin{aligned} L_\rho(x,\lambda) = &\sum_{j\in P} x_j^2 + \sum_{i\in M}\lambda_i\Big(\sum_{j\in P} a_{ij}x_j - y_i\Big) \\ &+ \frac{\rho}{2}\sum_{i\in M}\Big(\sum_{j\in P} a_{ij}x_j - y_i\Big)^2 \end{aligned} \tag{25}$$

At iteration $k+1$, with $\lambda_i \in \mathbb{R}$, and $\rho_k > 0$, the dual variables are updated according to the following rule:

$$\begin{aligned} x^{k+1} &= \min_x L_\rho(x,\lambda^k) \\ \lambda_i^{k+1} &= \lambda_i^k + \rho_k\Big(\sum_{j\in P} a_{ij}x_j^{k+1} - y_i\Big), \quad \forall i \in M \end{aligned} \tag{26}$$

C. The Bertsekas Method

The Bertsekas augmented Lagrangian method leverages a coordinate-wise structure, decoupling the objective further across variables x_j. The resulting formulation is:

$$\begin{aligned} &L_\rho(x,x^s,\lambda) \\ &= \sum_{j\in P}\left\{x_j^2 + \sum_{i\in M}\lambda_i\Big(a_{ij}x_j - \frac{y_i}{|P|}\Big) + \frac{\rho}{2}\Big(x_j - x_j^s\Big)^2\right\} \\ &= \sum_{j\in P} L_{\rho_j}(x_j,x_j^s,\lambda) \end{aligned} \tag{27}$$

Here, x_j^s represents a surrogate or anchor value from the previous iteration. At iteration $k+1$, with $\lambda_i \in \mathbb{R}$, $\rho_k > 0$, and $\zeta = [0,1)$ the dual variables are updated according to the following rule:

$$x_j^{k+1} = \min_{x_j} L_{\rho_j}(x_j, x_j^{s^k}, \lambda^k), \quad \forall j \in N \tag{28}$$

$$x_j^{s^{k+1}} = \zeta x_j^{s^k} + (1-\zeta)x_j^{k+1}, \quad \forall j \in N \tag{29}$$

$$\lambda_i^{k+1} = \lambda_i^k + \rho_k\Big(\sum_{j\in P} a_{ij}x_j^{k+1} - y_i\Big), \quad \forall i \in M \tag{30}$$

D. The Tatjewski Method

The Tatjewski method incorporates a more refined surrogate mechanism by coupling updates of x_j with fixed values of the remaining variables, leading to a partially separable augmented Lagrangian:

$$\begin{aligned} L_\rho(x,x^s,\lambda) = &\sum_{j\in P}\left\{x_j^2 + \sum_{i\in M}\lambda_i\Big(a_{ij}x_j - \frac{y_i}{|P|}\Big)\right. \\ &\left. + \frac{\rho}{2}\sum_{i\in M}\Big(a_{ij}x_j - \frac{y_i}{|P|} + \sum_{l\in P, l\neq j} a_{il}x_l^s\Big)^2\right\} \\ = &\sum_{j\in P} L_{\rho_j}(x_j,x_j^s,\lambda) \end{aligned} \tag{31}$$

This technique permits effective iterative refinement and is particularly useful in parallel implementations. At iteration $k+$

1, with $\lambda_i \in \mathbb{R}$, $\rho_k > 0$, and $\zeta = [0, 1)$, the dual variables are updated according to the following rule:

$$x_j^{k+1} = \min_{x_j} L_{\rho_j}(x_j, x_j^{s^k}, \lambda^k), \quad \forall j \in N \tag{32}$$

$$x_j^{s^{k+1}} = \zeta x_j^{s^k} + (1-\zeta)x_j^{k+1}, \quad \forall j \in N \tag{33}$$

$$\lambda_i^{k+1} = \lambda_i^k + \rho_k\Big(\sum_{j\in P} a_{ij}x_j^{k+1} - y_i\Big), \quad \forall i \in M \tag{34}$$

E. The ADMM (SALA Version)

The Separated Augmented Lagrangian Algorithm (SALA) introduces auxiliary variables s_{ij} to explicitly split the affine constraints (see equation (21)). The corresponding augmented Lagrangian is given by:

$$\begin{aligned} L_\rho(x, s, \lambda) &= \sum_{j\in P}\Bigg\{x_j^2 + \sum_{i\in M}\lambda_i\Big(a_{ij}x_j - \frac{y_i}{|P|} - s_{ij}\Big) \\ &\quad + \frac{\rho}{2}\sum_{i\in M}\Big(a_{ij}x_j - \frac{y_i}{|P|} - s_{ij}\Big)^2\Bigg\} \\ &= \sum_{j\in P} L_{\rho_j}(x_j, s_j, \lambda) \end{aligned} \tag{35}$$

At iteration $k+1$, the algorithm proceeds with the following updates:

$$x_j^{k+1} = \arg\min_{x_j} L_{\rho_j}(x_j, s_j^k, \lambda^k), \quad j \in P \tag{36}$$

$$r_i^{k+1} = \sum_{j\in P} a_{ij}x_j^{k+1} - y_i, \quad i \in M \tag{37}$$

$$s_{ij}^{k+1} = a_{ij}x_j^{k+1} - \frac{y_i}{|P|} - \frac{r_i^{k+1}}{|P|}, \quad j \in P, i \in M \tag{38}$$

$$\lambda_i^{k+1} = \lambda_i^k + \frac{\rho_k}{|P|}r_i^{k+1}, \quad i \in M \tag{39}$$

$$\rho_{k+1} = \alpha\,\rho_k, \quad \alpha \geq 1 \tag{40}$$

This method allows efficient decoupling and parallel updates of the variables while maintaining primal feasibility via residual tracking.

IV. K-Means Clustering with Augmented Lagrangian Algorithms

In various fields such as data mining, machine learning, image analysis, and bioinformatics, the need to uncover interpretable and computationally efficient patterns in large datasets has driven the study of *clustering methods* [19]. Among these, K-means clustering [20], [21] is one of the most widely used techniques for partitioning data into homogeneous groups. The primary objective is to identify cluster centers that minimize intra-cluster variation, yielding a compact data representation. Recent advances have explored reformulating K-means clustering using continuous optimization frameworks, where augmented Lagrangian algorithms provide efficient strategies for handling the nonconvex constraints inherent in clustering problems [22], [23].

K-means clustering is a technique to divide a dataset into n clusters. Given a dataset matrix $A = [a_{iq}]_{m\times Q}$ containing samples $A_i = [a_{i1}, a_{i2}, \ldots, a_{iQ}], i = 1, \ldots, m$ in $\mathbb{R}^Q$, the objective is to identify n cluster centers $\phi_1, \phi_2, \ldots, \phi_n$, such that the total squared distance between each point A_i and the closest cluster center ϕ_j is minimized. This can be formulated as:

$$\min_{x,\phi}\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}\left(\frac{1}{2}\sum_{q=1}^{Q}(a_{iq} - \phi_{jq})^2\right) \tag{41}$$

subject to:

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, \ldots, m, \tag{42}$$

$$x_{ij} \in \{0, 1\}, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n \tag{43}$$

where x_{ij} is an indicator whether the i-th sample belongs to the j-th cluster. The algorithm proceeds iteratively with two main steps: first, each point A_i is assigned to the nearest cluster, then, the cluster centers ϕ_j are updated as the mean of the points assigned to each cluster. This process repeats until convergence.

However, K-Means can sometimes produce empty or very small clusters, especially when applied to high-dimensional datasets. To address this, a constrained version of K-Means was introduced in [24]. This approach modifies the optimization problem by adding constraints that ensure that each cluster has at least τ_j points. The paper also suggested solving (41)-(43) in x for fixed ϕ, then solving (41) in ϕ for fixed x. The new optimization problem is defined as:

At iteration $k + 1$:

- **Cluster Assignment**: Let x_{ij}^k be a solution to the following problem with ϕ_j^k fixed.

$$\min_{x}\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}\left(\frac{1}{2}\sum_{q=1}^{Q}\left(a_{iq} - \phi_{jq}^k\right)^2\right) \tag{44}$$

  subject to:

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, \ldots, m, \tag{45}$$

$$\sum_{i=1}^{m} x_{ij} \geq \tau_j, \quad j = 1, \ldots, n, \tag{46}$$

$$x_{ij} \in \{0, 1\}, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n \tag{47}$$

- **Cluster Center Update**: The cluster centers are updated $\forall j = 1, \ldots, n;\ q = 1, \ldots, Q$, as:

$$\phi_{jq}^{k+1} = \begin{cases} \frac{\sum_{i=1}^m x_{ij}^k a_{iq}}{\sum_{i=1}^m x_{ij}^k}, & \text{if } \sum_{i=1}^m x_{ij}^k > 0 \\ \phi_{jq}^k, & \text{otherwise} \end{cases}, \tag{48}$$

Stop when $\phi_{jq}^{k+1} = \phi_{jq}^k$ for all j, q, else increment k by 1 and go to step 1.

The challenge with the algorithm is that it is impossible to decompose it due to the coupling constraint, $\sum_{i=1}^m x_{ij} \geq \tau_j$. To overcome this, the Bertsekas Decomposition method is used. It will be convenient to introduce for every training example i the admissible set X_i resulting from the equations (45), (47). This set for each training example i will be defined as:

$$X_i = \left\{ \begin{array}{l} \sum_{j=1}^{n} x_{ij} = 1 \\ x_{ij} \in \{0,1\}, \quad j = 1,\ldots,n \end{array} \right\} \tag{49}$$

The constrained K-Means clustering problem, as formulated in (44)–(47), admits a structure that is well-suited for decomposition techniques, particularly in the context of large-scale and distributed optimization. While the objective function is partially separable across data points, the assignment constraints introduce coupling through the cardinality conditions $\sum_{i=1}^{m} x_{ij} \geq \tau_j$. This blend of separability and structured coupling motivates using augmented Lagrangian and dual decomposition methods to solve the resulting mixed-integer optimization problem efficiently.

To address the coupling constraints and enable scalable computation, consider a family of decomposition strategies rooted in classical and modern dual optimization frameworks. These include the Multiplier Method, the Bertsekas Decomposition technique, the Tatjewski formulation and the SALA (Separated Augmented Lagrangian Algorithm) variant of the Alternating Direction Method of Multipliers (ADMM). Each approach introduces dual variables associated with the cluster-level constraints and iteratively updates the primal variables x and the cluster centers ϕ while coordinating through dual ascent or primal-dual updates. The augmented Lagrangian terms serve both to penalize constraint violations and to improve convergence stability.

These decomposition techniques are especially attractive in distributed environments where data points are stored across multiple compute nodes. By decoupling subproblems, typically across data samples, they enable parallel updates of the assignment variables, subject to coordination via dual variables.

A. The Lagrangian

The Lagrangian of Problem (44)-(47) can be written as

$$\begin{aligned} &L(x,\phi,\mu) \\ &= \sum_{i\in M}\sum_{j\in P}\left[\sum_{q\in Q} x_{ij}\left(a_{iq}-\phi_{jq}\right)^2 + \mu_j\Big(\frac{\tau}{|M|} - x_{ij}\Big)\right] \\ &= \sum_{i\in M}\sum_{j\in P} L_{ij}(x_{ij},\phi_j,\mu_j) \end{aligned} \tag{50}$$

where

$$\begin{aligned} &L_{ij}(x_{ij},\phi_j,\mu_j) \\ &\qquad = \sum_{q\in Q} x_{ij}\left(a_{iq}-\phi_{jq}\right)^2 + \mu_j\Big(\frac{\tau}{|M|} - x_{ij}\Big) \end{aligned} \tag{51}$$

At iteration $k+1$, with $\mu_j^k \in R$, and $\rho > 0$,

$$x_i^{k+1} = \arg\min_{x_i\in X_i} \sum_{j\in P} L_{ij}(x_{ij},\phi_j^k,\mu_j^k) \tag{52}$$

$$\phi_{jq}^{k+1} = \begin{cases} \frac{\sum_{i=1}^m x_{ij}^{k+1} a_{iq}}{\sum_{i=1}^m x_{ij}^{k+1}}, & \text{if } \sum_{i=1}^m x_{ij}^{k+1} > 0 \\ \phi_{jq}^k, & \text{otherwise} \\ \forall j \in P;\ q \in Q \end{cases} \tag{53}$$

$$\mu_j^{k+1} = \max\left\{0, \mu_j^k + \rho\Big(\tau - \sum_{i\in M} x_{ij}\Big)\right\}, \quad \forall j \in P \tag{54}$$

B. The Multiplier Method

The Augmented Lagrangian of Problem (44)-(47) can be written as

$$\begin{aligned} &L_\rho(x,\phi,\mu) \\ &= \sum_{j\in P}\left[\sum_{i\in M}\sum_{q\in Q} x_{ij}\left(a_{iq}-\phi_{jq}\right)^2 + \mu_j\Big(\tau_j - \sum_{i\in M} x_{ij}\Big)\right] \\ &+ \frac{\rho}{2}\sum_{j\in P}\Big(\tau_j - \sum_{i\in M} x_{ij}\Big)^2 \end{aligned} \tag{55}$$

At iteration $k+1$,

$$x^{k+1} = \arg\min_{x_i\in X_i, \forall i\in M} L_\rho(x,\phi^k,\mu^k) \tag{56}$$

$$\phi_{jq}^{k+1} = \begin{cases} \frac{\sum_{i=1}^m x_{ij}^{k+1} a_{iq}}{\sum_{i=1}^m x_{ij}^{k+1}}, & \text{if } \sum_{i=1}^m x_{ij}^{k+1} > 0 \\ \phi_{jq}^k, & \text{otherwise} \\ \forall j \in P;\ q \in Q \end{cases} \tag{57}$$

$$\mu_j^{k+1} = \max\left\{0, \mu_j^k + \rho\Big(\tau - \sum_{i\in M} x_{ij}\Big)\right\}, \quad \forall j \in P \tag{58}$$

C. The Bertsekas Method

The Bertsekas Augmented Lagrangian method for problem (44)-(47) has this form:

$$\begin{aligned} &L_\rho(x,\phi,x^s,\phi^s,\mu) \\ &= \sum_{i\in M}\sum_{j\in P}\left\{\sum_{q\in Q} x_{ij}\left(a_{iq}-\phi_{jq}\right)^2 + \mu_j\Big(\frac{\tau}{|M|} - x_{ij}\Big)\right. \\ &\left. + \frac{\rho}{2}\left[\left(x_{ij}-x_{ij}^s\right)^2 + \frac{1}{|M|}\sum_{q\in Q}\left(\phi_{jq}-\phi_{jq}^s\right)^2\right]\right\} \\ &= \sum_{i\in M}\sum_{j\in P} L_{\rho_{ij}}(x_{ij},\phi_j,x_{ij}^s,\mu_j) \end{aligned} \tag{59}$$

where

$$\begin{aligned} &L_{\rho_{ij}}(x_{ij},\phi_j,x_{ij}^s,\mu_j) \\ &\qquad = \sum_{q\in Q} x_{ij}\left(a_{iq}-\phi_{jq}\right)^2 + \mu_j\Big(\frac{\tau}{|M|} - x_{ij}\Big) \\ &\qquad + \frac{\rho}{2}\left(x_{ij}-x_{ij}^s\right)^2 \end{aligned} \tag{60}$$

At iteration $k+1$, with $x_{ij}^s \in \{0,1\}$, $\phi_j^s \in R$,

$$x_i^{k+1} = \arg\min_{x_i\in X_i} \sum_{j\in P} L_{\rho_{ij}}(x_{ij},\phi_j^k,x_{ij}^{s^k},\mu_j^k), \quad \forall i \in M \tag{61}$$

$$\phi_{jq}^{k+1} = \begin{cases} \frac{\rho\phi_{jq}^k + \sum_{i=1}^m x_{ij}^{k+1} a_{iq}}{\rho + \sum_{i=1}^m x_{ij}^{k+1}}, & \text{if } \sum_{i=1}^m x_{ij}^{k+1} > 0 \\ \phi_{jq}^k, & \text{otherwise} \\ \forall j \in P;\ q \in Q \end{cases} \tag{62}$$

$$x_{ij}^{s^{k+1}} = \xi_k x_{ij}^{s^k} + (1-\xi_k) x_{ij}^{k+1}, \quad \forall i \in M;\ j \in P \tag{63}$$

$$\phi_{jq}^{s^{k+1}} = \xi_k \phi_{jq}^{s^k} + (1-\xi_k) \phi_{jq}^{k+1}, \quad \forall i \in M;\ j \in P \tag{64}$$

$$\mu_j^{k+1} = \max\left\{0, \mu_j^k + \rho\Big(\tau - \sum_{i\in M} x_{ij}\Big)\right\}, \quad \forall j \in P \tag{65}$$

D. The Tatjewski Method

The Tatjewski Augmented Lagrangian method for Problem (44)-(47) has this form

$$\begin{aligned} &L_\rho(x,\phi,x^s,\mu) \\ &\quad= \sum_{i\in M}\sum_{j\in P}\Bigg\{\sum_{q\in Q} x_{ij}\left(a_{iq}-\phi_{jq}\right)^2+\mu_j\Big(\frac{\tau}{|M|}-x_{ij}\Big) \\ &\quad+\frac{\rho}{2}\Big(\tau_j-x_{ij}-\sum_{l\in M,l\neq i}x^s_{lj}\Big)^2\Bigg\} \\ &\quad= \sum_{i\in M}\sum_{j\in P} L_{\rho_{ij}}(x_{ij},\phi_j,x^s_{ij},\mu_j) \end{aligned} \tag{66}$$

where

$$\begin{aligned} &L_{\rho_{ij}}(x_{ij},\phi_j,x^s_{ij},\mu_j) \\ &\quad= \sum_{q\in Q} x_{ij}\left(a_{iq}-\phi_{jq}\right)^2+\mu_j\Big(\frac{\tau}{|M|}-x_{ij}\Big) \\ &\quad+\frac{\rho}{2}\Big(\tau_j-x_{ij}-\sum_{l\in M,l\neq i}x^s_{lj}\Big)^2 \end{aligned} \tag{67}$$

At iteration $k+1$,

$$x_i^{k+1}=\arg\min_{x_i\in X_i}\sum_{j\in P}L_{\rho_{ij}}(x_{ij},\phi_j^k,x_{ij}^{s^k},\mu_j^k),\quad \forall i\in M \tag{68}$$

$$\phi_{jq}^{k+1}=\begin{cases}\frac{\sum_{i=1}^m x_{ij}^{k+1}a_{iq}}{\sum_{i=1}^m x_{ij}^{k+1}}, & \text{if } \sum_{i=1}^m x_{ij}^{k+1}>0\\ \phi_{jq}^k, & \text{otherwise}\\ \forall j\in P;\, q\in Q\end{cases} \tag{69}$$

$$x_{ij}^{s^{k+1}}=\xi_k x_{ij}^{s^k}+(1-\xi_k)x_{ij}^{k+1},\quad \forall i\in M;\, j\in P \tag{70}$$

$$\mu_j^{k+1}=\max\left\{0,\mu_j^k+\rho\Big(\tau-\sum_{i\in M}x_{ij}\Big)\right\},\quad \forall j\in P \tag{71}$$

E. The ADMM (SALA Version)

Adapting ADMM SALA (18)-(20) to solve Problem (44)-(47), we have,

$$\begin{aligned} &L_\rho(x,\phi,s,\mu) \\ &\quad= \sum_{i\in M}\sum_{j\in P}\Bigg[\sum_{q\in Q} x_{ij}\left(a_{iq}-\phi_{jq}\right)^2 \\ &\quad+\mu_j\Big(\frac{\tau}{|M|}-x_{ij}-s_{ij}\Big)+\frac{\rho}{2}\Big(\frac{\tau}{|M|}-x_{ij}-s_{ij}\Big)^2\Bigg] \\ &\quad= \sum_{i\in M}\sum_{j\in P} L_{\rho_{ij}}(x_{ij},\phi_j,s_{ij},\mu_j) \end{aligned}$$

where

$$\begin{aligned} L_{\rho_{ij}}(x_{ij},\phi_j,s_{ij},\mu_j) &= \sum_{q\in Q} x_{ij}\left(a_{iq}-\phi_{jq}\right)^2 \\ &+\mu_j\Big(\frac{\tau}{|M|}-x_{ij}-s_{ij}\Big)+\frac{\rho}{2}\Big(\frac{\tau}{|M|}-x_{ij}-s_{ij}\Big)^2 \end{aligned} \tag{72}$$

At iteration $k+1$,

$$x_i^{k+1}=\arg\min_{x_i\in X_i}\sum_{j\in P}L_{\rho_{ij}}(x_{ij},\phi_j^k,s_{ij}^k,\mu_j^k),\quad \forall i\in M \tag{73}$$

$$s_{ij}^{k+1}=\frac{\tau}{|M|}-x_{ij}^{k+1}-\frac{\mu_j^{k+1}}{\rho},\quad \forall j\in P, i\in M \tag{74}$$

$$\phi_{jq}^{k+1}=\begin{cases}\frac{\sum_{i=1}^m x_{ij}^{k+1}a_{iq}}{\sum_{i=1}^m x_{ij}^{k+1}}, & \text{if } \sum_{i=1}^m x_{ij}^{k+1}>0\\ \phi_{jq}^k, & \text{otherwise}\\ \forall j\in P;\, q\in Q\end{cases} \tag{75}$$

$$\mu_j^{k+1}=\max\left\{0,\mu_j^k+\rho\Big(\tau-\sum_{i\in M}x_{ij}\Big)\right\},\quad \forall j\in P \tag{76}$$

V. Experiments

All implementations and numerical experiments were conducted using Python 3.12.0. The optimization models were formulated with the Pyomo modeling framework and solved using the Gurobi optimizer. Computational experiments were performed on a machine equipped with an AMD Ryzen 5 4600H processor (3.00 GHz, Radeon Graphics), 32 GB of RAM, and a 512 GB SSD, running a 64-bit Windows 10 Pro operating system.

The considered four optimization algorithms, ADMM, Bertsekas, Tatjewski's method, and the classical Augmented Lagrangian were implemented with and without variable partitioning. All problems, except for the Augmented Lagrangian, were implemented using decomposition. The "No Partition" configuration corresponds to a single-processor (serial) execution, whereas "12 Partitions" denotes parallelization with 12 processors, one assigned to each partition.

The individual results were first aligned to a unified time axis to enable consistent comparison across datasets with potentially different time indices. A comprehensive timeline was constructed by taking the union of all time points in the datasets. Each dataset was then merged onto this unified timeline using nearest-neighbor matching via an as-of merge (***pandas.merge_asof*** *performs a merge by nearest key rather than exact matches, aligning rows based on the closest preceding key in a sorted dataset. It is especially useful for time-series data to join on nearest timestamps without requiring exact equality*), ensuring that for each time point in the reference axis, the closest available record from each dataset was selected. This approach preserves temporal coherence while allowing synchronized evaluation of multiple time series, even in non-uniform or asynchronous sampling intervals.

A. Regularized Linear Systems

1) Dataset Description: The experiments are conducted on three linear systems designed to reflect varying data structures and matrix conditions. Two are derived from an image inpainting task using randomized diagonal measurement operators, and the third is based on a biomedical dataset reformulated as a compressed feature recovery problem. Each dataset adheres to the standard linear model $Ax = y$, with a known ground truth $x^\star$.

The first dataset simulates a diagonal sensing problem in grayscale image recovery. A 16×16 grayscale image is vectorized into $x^\star \in \mathbb{R}^{256}$, and a diagonal measurement matrix $A \in \mathbb{R}^{256\times256}$ is constructed with entries drawn independently from a uniform distribution over $[0, 1)$. The corresponding observation vector is computed as $y = Ax^\star$, resulting in a well-scaled and positive semi-definite system. This formulation captures moderate conditioning and serves as a stable reference for evaluating solution quality.

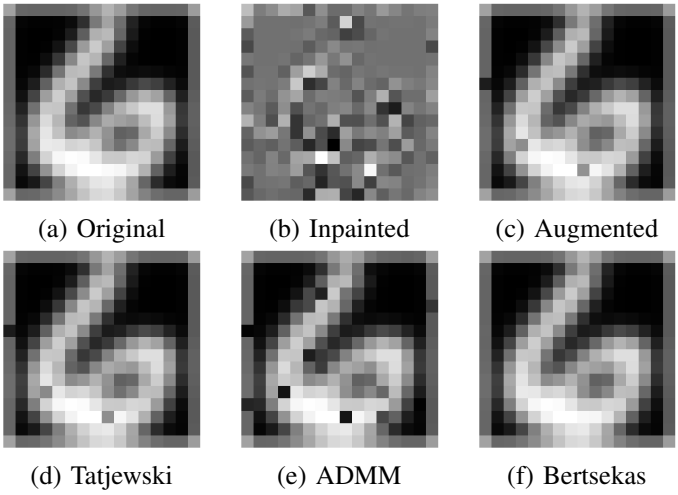
The second dataset uses the same underlying image and construction but replaces the uniform distribution with a standard Gaussian distribution for the diagonal entries of A. This results in a more ill-conditioned system where entries can be both positive and negative, potentially with large magnitude. The measurement vector $y = Ax^\star$, therefore, reflects a noisier and more variable scaling of the original image, posing greater challenges for algorithmic recovery under unstable and zero-mean multiplicative transformations.

The third dataset is derived from the UCI Breast Cancer Wisconsin (Diagnostic) dataset [25] and formulated as a compressed sensing task. After standardizing the data, a single feature vector $x^\star \in \mathbb{R}^{30}$ is selected from the training partition. A sensing matrix $A \in \mathbb{R}^{10\times30}$ is generated with independent standard Gaussian entries scaled by $1/\sqrt{10}$. The measurement vector $y = Ax^\star$ thus represents a low-dimensional projection of the original biomedical profile. This setting is representative of practical dimensionality reduction problems in clinical data, where reconstruction must be achieved from limited and noisy observations.

2) Results and Discussion: Table I summarizes the performance of four optimization algorithms, ADMM, Bertsekas, Tatjewski's method, and the classical Augmented Lagrangian, across three datasets with and without variable partitioning. Across the Gaussian and Uniform datasets, which feature large and ill-conditioned linear systems, Bertsekas' method consistently achieves the best trade-off between convergence and computational efficiency. It converges in all settings (Fig. 1) and does so significantly faster when the problem is decomposed into 12 partitions. ADMM and Tatjewski both fail to converge within the 5000-second limit in the unpartitioned setting but improve under decomposition, highlighting the benefits of problem structure exploitation. The Augmented Lagrangian method does not converge in time for either of the datasets and lacks implementation under decomposition.

On the Cancer dataset, which involves a significantly smaller and better-conditioned system, all algorithms (except Augmented Lagrangian) converge rapidly in both partitioned and unpartitioned forms. Here, Bertsekas again records the lowest runtime, while ADMM and Tatjewski exhibit modestly higher computational cost. The Augmented Lagrangian method produces a suboptimal objective value and performs poorly relative to the others.

These results indicate that dual ascent approaches, particularly Bertsekas' method, are robust across system scales and benefit markedly from decomposition. In contrast, classical methods like ADMM and Augmented Lagrangian suffer in high-dimensional settings without partitioning, with the latter also being sensitive to problem scaling.



(a) Original (b) Inpainted (c) Augmented

(d) Tatjewski (e) ADMM (f) Bertsekas

Fig. 1. Image Recovery under Gaussian Additive Noise and no partitions

B. K-Means Clustering

1) Dataset Description: To evaluate the performance and generalizability of the proposed algorithms, experiments were conducted on both synthetic and real-world datasets representing a diverse range of clustering challenges. All datasets were processed to ensure comparability and reproducibility, with dimensionality reduction applied where appropriate.

Four synthetic datasets were created using **make_blobs** from the Scikit-learn library [26], which generates isotropic Gaussian clusters commonly used for clustering evaluation. The datasets vary in the number of samples, features, and clusters to simulate different levels of complexity as shown below:

- **High-Dimensional Multi-Cluster (Synthetic-HD-MC)**: 1,000 samples with 100 features grouped into 15 clusters.
- **Low-Dimensional Few-Cluster (Synthetic-LD-FC)**: 5,000 samples with 10 features and 5 clusters.
- **2D Multi-Cluster Visualization Set (Synthetic-2D-MC)**: 1,000 samples in 2D space with 15 clusters, suitable for visual inspection.
- **2D Few-Cluster Visualization Set (Synthetic-2D-FC)**: 5,000 samples in 2D space with 5 clusters.

These synthetic datasets help assess clustering performance under controlled distributions, with cluster separability influenced by feature dimensionality and the number of clusters.

Two real-world datasets, **ISIC 2019** and **MedQuAD**, were also used to test the algorithms in more practical, noisy scenarios.

The ISIC 2019 dataset [27]–[29], the most widely available publicly available collection of quality-controlled dermatology skin images, was used to test clustering models with image data. The dataset contains 25331 dermoscopic images of skin lesions, each associated with ground-truth diagnoses (benign, malignant) and clinical metadata. These standardized images provided a diverse and high-quality foundation for generating features customized to clustering tasks in the medical imaging domain.

To prepare the images for clustering, they were converted to grayscale to simplify the data while retaining essential visual

TABLE I
PERFORMANCE COMPARISON OF OPTIMIZATION ALGORITHMS FOR REGULARIZED LINEAR SYSTEMS ACROSS DIFFERENT DATASETS

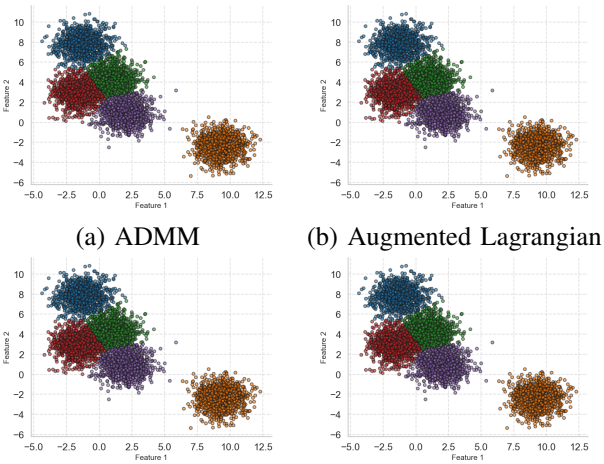
Dataset	Algorithm	No Partition			12 Partitions		
		Objective	Time(s)	Status	Objective	Time(s)	Status
Gaussian	ADMM	59.01	5001	Time Out	61.4	3396	Converged
	Bertsekas	61.41	4880	Converged	61.41	938	Converged
	Tatjewski	61.03	5010	Time Out	61.41	2372	Converged
	Aug Lagrangian	61.05	5001	Time Out	-	-	-
Uniform	ADMM	57.04	5000	Time Out	60.67	2916	Converged
	Bertsekas	60.87	3095	Converged	60.87	724	Converged
	Tatjewski	59.78	5009	Time Out	60.87	1983	Converged
	Aug Lagrangian	59.92	5003	Time Out	-	-	-
Cancer	ADMM	54.61	150	Converged	54.61	55	Converged
	Bertsekas	54.61	119	Converged	54.61	55	Converged
	Tatjewski	54.61	309	Converged	54.61	95	Converged
	Aug Lagrangian	54.61	150	Converged	-	-	-

features. Each image was then resized for uniformity, converted into arrays, and flattened into one-dimensional vectors. These flattened arrays served as the feature set for the clustering models. By preprocessing ISIC 2019 images this way, the clustering analysis could focus on the underlying patterns and relationships in the data, enabling a robust comparison of clustering approaches across this rich medical imaging dataset.

- **ISIC Lesion Embeddings (ISIC-PCA10)**: Derived from the ISIC 2019 Challenge dataset [27]–[29], a publicly available collection of quality-controlled dermatology skin images. 2,000 image feature vectors were selected, reduced to 10 principal components.
- **ISIC Low-Dimensional Variant (ISIC-PCA2)**: The same set as above but reduced to 2 PCA components, creating a more compressed representation to test performance under extreme dimensionality reduction.
- **MedQuAD QA Representations (MedQA-PCA20)**: Based on the MedQuAD dataset [30], which contains medically relevant question-answer pairs. To generate the features for the clustering problem, each row of the "answers" column of the MedQuAD dataset was processed using BioClinicalBERT [31], a pre-trained language model explicitly designed for clinical text to create numerical embeddings. These embeddings, which capture the semantic essence of the text, were then used as the feature set for the clustering models. 1,000 instances were sampled, and PCA was applied to reduce the feature space to 20 components.
- **MedQuAD Minimal Representation (MedQA-PCA2)**: A simplified variant with only 2 PCA components to evaluate performance in a very low-dimensional semantic space.

The ISIC dataset serves as a representative for medical image data, while MedQuAD captures structured medical text, providing a rich testbed for evaluating clustering robustness across modalities and dimensionalities.

2) Results and Discussion: Tables II and III summarize the performance of four optimization algorithms for K-means clustering on synthetic and real-world datasets, a typical solution is presented in Fig. 2.



(a) ADMM (b) Augmented Lagrangian

(c) Bertsekas (d) Tatjewski

Fig. 2. K-means clustering for the Synthetic-LD-FC dataset

Synthetic Datasets: Across all synthetic datasets, the algorithms converged to identical objective values and clustering scores, confirming the correctness and equivalence of the underlying formulations. However, notable differences emerged in computational efficiency.

The **Bertsekas algorithm** achieved the **best overall runtime**, converging quickly with low computational cost across all datasets, and proved effective for both low- and high-dimensional clustering. **ADMM** was slightly slower but remained consistent and reliable; partitioning further reduced runtime (e.g., 43s $\rightarrow$ 31s on Synthetic-2D-FC). **Tatjewski's method** was slow in the non-partitioned setting but gained substantial efficiency when partitioned (e.g., 5× speedup on Synthetic-2D-MC), demonstrating good scalability. In contrast, the **Augmented Lagrangian** approach scaled poorly: although it converged on small datasets, it often timed out in partitioned settings due to synchronization overhead, limiting its practicality.

Real-World Datasets: The findings on real-world data mirrored those on synthetic data, though the impact of dimen-

TABLE II
PERFORMANCE COMPARISON OF OPTIMIZATION ALGORITHMS FOR K-MEANS CLUSTERING ON SYNTHETIC DATASETS

		No Partition				8 Partitions			
Dataset	**Algorithm**	**Objective**	Score	**Time**(s)	Status	**Objective**	Score	**Time**(s)	Status
Synthetic-LD-FC	ADMM	24807.09	0.74	29	Converged	24807.09	0.74	25	Converged
	Bertsekas	24807.09	0.74	26	Converged	24807.09	0.74	21	Converged
	Tatjewski	24807.09	0.74	797	Converged	24807.09	0.74	197	Converged
	Aug Lagrangian	24807.09	0.74	1157	Converged	-	-	-	-
Synthetic-HD-MC	ADMM	346173.75	0.55	29	Converged	346173.75	0.55	23	Converged
	Bertsekas	346173.75	0.55	64	Converged	346173.75	0.55	44	Converged
	Tatjewski	346173.75	0.55	161	Converged	346173.75	0.55	61	Converged
	Aug Lagrangian	346173.75	0.55	109	Converged	-	-	-	-
Synthetic-2D-MC	ADMM	1064.55	0.45	38	Converged	1064.55	0.45	28	Converged
	Bertsekas	1064.55	0.45	20	Converged	1064.55	0.45	19	Converged
	Tatjewski	1064.55	0.45	177	Converged	1064.55	0.45	60	Converged
	Aug Lagrangian	1064.55	0.45	512	Converged	-	-	-	-
Synthetic-2D-FC	ADMM	4505.94	0.55	43	Converged	4505.94	0.55	31	Converged
	Bertsekas	4505.94	0.55	27	Converged	4505.94	0.55	21	Converged
	Tatjewski	4505.94	0.55	978	Converged	4505.94	0.55	297	Converged
	Aug Lagrangian	4506.03	0.55	4115	Time Out	-	-	-	-

TABLE III
PERFORMANCE COMPARISON OF OPTIMIZATION ALGORITHMS FOR K-MEANS CLUSTERING ON REAL WORLD DATASETS

		No Partition				8 Partitions			
Dataset	**Algorithm**	**Objective**	Score	**Time**(s)	Status	**Objective**	Score	**Time**(s)	Status
ISIC-PCA10	ADMM	234036414.61	0.17	107	Converged	234036414.61	0.17	50	Converged
	Bertsekas	234036414.61	0.17	88	Converged	234036414.61	0.17	61	Converged
	Tatjewski	234036414.61	0.17	1699	Converged	234036414.61	0.17	385	Converged
	Aug Lagrangian	237761989.80	0.17	4108	Time Out	-	-	-	-
ISIC-PCA2	ADMM	80010359.93	0.37	56	Converged	80010359.93	0.37	35	Converged
	Bertsekas	80010359.93	0.37	42	Converged	80010359.93	0.37	25	Converged
	Tatjewski	80010359.93	0.37	859	Converged	80010359.93	0.37	200	Converged
	Aug Lagrangian	80064769.15	0.37	4060	Time Out	-	-	-	-
MedQA-PCA20	ADMM	1289.70	0.14	251	Converged	1289.70	0.14	74	Converged
	Bertsekas	1289.70	0.14	35	Converged	1289.70	0.14	24	Converged
	Tatjewski	1289.70	0.14	251	Converged	1289.70	0.14	74	Converged
	Aug Lagrangian	1289.70	0.14	708	Converged	-	-	-	-
MedQA-PCA2	ADMM	57.95	0.36	55	Converged	57.95	0.36	33	Converged
	Bertsekas	57.95	0.36	36	Converged	57.95	0.36	23	Converged
	Tatjewski	57.95	0.36	715	Converged	57.95	0.36	103	Converged
	Aug Lagrangian	57.95	0.36	1258	Converged	-	-	-	-

sionality reduction (via PCA) made performance distinctions more pronounced.

Bertsekas' algorithm again stood out for its speed, completing MedQA-PCA2 and ISIC-PCA2 in under 25 seconds while preserving optimal objective values and clustering scores. **ADMM** remained robust, balancing runtime and reliability, and scaled well to larger datasets such as ISIC-PCA10, where partitioning reduced runtime from 107s to 50s. **Tatjewski** also benefited greatly from partitioning (e.g., 1699s $\rightarrow$ 385s on ISIC-PCA10), confirming its usefulness in distributed settings despite a higher per-iteration cost. In contrast, the **Augmented Lagrangian** approach was inefficient for high-dimensional or partitioned data, often failing to converge within time limits or offering no advantage over simpler methods.

Overall, **Bertsekas' method emerged as the most efficient**, delivering fast and reliable convergence across all datasets and settings. **ADMM** was a strong second, offering consistent performance and good scalability. **Tatjewski** lagged in speed but benefited markedly from partitioning, making it viable for parallel environments. In contrast, the **Augmented Lagrangian** method showed limited utility due to its sensitivity to problem decomposition and coordination overhead.

VI. CONCLUSIONS

This study has conducted a comprehensive comparative analysis of four optimisation algorithms: ADMM, Bertsekas' method, Tatjewski's method, and the classical Augmented Lagrangian, across a diverse set of problem instances, including regularised linear systems and K-means clustering, using both synthetic and empirical datasets.

Among the evaluated methods, **Bertsekas' method** consistently demonstrated superior performance. It achieved rapid convergence, exhibited strong resilience in ill-conditioned settings, and scaled effectively when decomposition was applied. These attributes render it particularly well suited to large-scale and distributed optimisation scenarios where coordination efficiency and robust convergence are essential.

ADMM also showed dependable performance across tasks, especially under decomposition. While generally slower than

Bertsekas' method, its convergence behaviour remained stable and reliable, making it a viable option where decomposition is supported and computational budgets are less restrictive.

Tatjewski's method performed less favourably in non-decomposed settings, but benefited significantly from problem partitioning, indicating its potential applicability in parallelised or decentralised optimisation environments.

The **classical Augmented Lagrangian method** was evaluated in its standard, non-decomposed form, as the algorithm does not support decomposition by design. Its performance was comparatively limited, particularly on high-dimensional or complex problem instances. Notably, a number of trials failed to reach convergence within the allocated optimisation time. These cases do not necessarily indicate divergence, but rather suggest that, within practical time constraints, the method may be less efficient than alternatives.

These findings highlight the significant benefits of distributed optimization, particularly in improving the efficiency of computationally intensive methods like Tatjewski and Augmented Lagrangian. These insights contribute to the broader understanding of optimization in machine learning and provide a foundation for future research into adaptive and hybrid optimization strategies that further enhance efficiency in distributed environments.

REFERENCES

[1] K. Arrow, H. Azawa, K. M. R. Collection, L. Hurwicz, H. Uzawa, H. Chenery, S. Johnson, and S. Karlin, *Studies in Linear and Non-linear Programming*, ser. Stanford mathematical studies in the social sciences. Stanford University Press, 1958. [Online]. Available: https://books.google.pl/books?id=TkRlnQEACAAJ

[2] M. R. Hestenes, "Multiplier and gradient methods," *Journal of Optimization Theory and Applications*, vol. 4, no. 5, pp. 303–320, nov 1969. [Online]. Available: https://doi.org/10.1007/bf00927673

[3] M. J. D. Powell, "A method for nonlinear constraints in minimization problems," *Optimization*, pp. 283–298, 1969. [Online]. Available: https://ci.nii.ac.jp/naid/20000922074/en/

[4] D. P. Bertsekas, "Convexification procedures and decomposition methods for nonconvex optimization problems," *Journal of Optimization Theory and Applications*, vol. 29, no. 2, pp. 169–197, oct 1979. [Online]. Available: https://doi.org/10.1007/bf00937167

[5] A. Tanikawa and H. Mukai, "A new technique for nonconvex primal-dual decomposition of a large-scale separable optimization problem," *IEEE Transactions on Automatic Control*, vol. 30, no. 2, pp. 133–143, 1985.

[6] A. C. Nwachukwu and A. Karbowski, "Solution of the simultaneous routing and bandwidth allocation problem in energy-aware networks using augmented Lagrangian-based algorithms and decomposition," *Energies*, vol. 17, no. 5, p. 1233, mar 2024. [Online]. Available: http://dx.doi.org/10.3390/en17051233

[7] P. Tatjewski, "New dual-type decomposition algorithm for nonconvex separable optimization problems," *Automatica*, vol. 25, no. 2, pp. 233–242, mar 1989. [Online]. Available: https://doi.org/10.1016/0005-1098(89)90076-9

[8] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Computers & Mathematics with Applications*, vol. 2, no. 1, pp. 17–40, 1976. [Online]. Available: https://doi.org/10.1016/0898-1221(76)90003-1

[9] S. Boyd, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010. [Online]. Available: https://doi.org/10.1561/2200000016

[10] J. Eckstein and D. P. Bertsekas, "On the Douglas—Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming*, vol. 55, no. 1-3, pp. 293–318, apr 1992. [Online]. Available: https://doi.org/10.1007/bf01581204

[11] A. Hamdi, P. Mahey, and J. P. Dussault, "A new decomposition method in nonconvex programming via a separable augmented Lagrangian," in *Recent Advances in Optimization*, P. Gritzmann, R. Horst, E. Sachs, and R. Tichatschke, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 90–104.

[12] A. Hamdi, "Two-level primal-dual proximal decomposition technique to solve large scale optimization problems," *Applied Mathematics and Computation*, vol. 160, no. 3, p. 921 – 938, 2005.

[13] A. Hamdi and S. K. Mishra, "Decomposition methods based on augmented Lagrangians: A survey," *Springer Optimization and Its Applications*, vol. 50, p. 175 – 203, 2011.

[14] A. N. Tikhonov and V. Y. Arsenin, *Solutions of Ill-posed Problems*. New York: Wiley, 1977.

[15] P. C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*. Philadelphia: SIAM, 1998.

[16] A. N. Tikhonov, "Ill-posed problems of linear algebra and a stable method for their solution," *Soviet Math. Dokl.*, vol. 5, pp. 1035–1038, 1965.

[17] G. H. Golub, P. C. Hansen, and D. P. O'Leary, "Tikhonov regularization and total least squares," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 1, pp. 185–194, 1999.

[18] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[19] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[20] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[21] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1. University of California Press, 1967, pp. 281–297.

[22] E. G. Birgin, J. M. Martínez, and M. Raydan, "Practical augmented Lagrangian methods for constrained optimization," *SIAM Journal on Optimization*, vol. 14, no. 2, pp. 500–523, 2005.

[23] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York: Springer, 2006.

[24] P. Bradley, K. Bennett, and A. Demiriz, "Constrained K-means clustering," Microsoft Research, Redmond, Tech. Rep. MSR-TR-2000-65, 2000, Microsoft Research Technical Report. [Online]. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr-2000-65.pdf

[25] W. H. Wolberg and O. L. Mangasarian, "Breast cancer Wisconsin (diagnostic)," 1993.

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[27] P. Tschandl, C. Rosendahl, and H. Kittler, "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific Data*, vol. 5, no. 1, 8 2018. [Online]. Available: http://dx.doi.org/10.1038/sdata.2018.161

[28] N. C. F. Codella, D. Gutman, M. E. Celebi, B. Helba, M. A. Marchetti, S. W. Dusza, A. Kalloo, K. Liopyris, N. Mishra, H. Kittler, and A. Halpern, "Skin lesion analysis toward melanoma detection: A challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC)," 2017. [Online]. Available: https://arxiv.org/abs/1710.05006

[29] M. Combalia, N. C. F. Codella, V. Rotemberg, B. Helba, V. Vilaplana, O. Reiter, C. Carrera, A. Barreiro, A. C. Halpern, S. Puig, and J. Malvehy, "BCN20000: Dermoscopic lesions in the wild," 2019. [Online]. Available: https://arxiv.org/abs/1908.02288

[30] A. Ben Abacha and D. Demner-Fushman, "A question-entailment approach to question answering," *BMC Bioinform.*, vol. 20, no. 1, pp. 511:1–511:23, 2019. [Online]. Available: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-3119-4

[31] E. Alsentzer, J. R. Murphy, W. Boag, W. Weng, D. Jin, T. Naumann, and M. B. A. McDermott, "Publicly available clinical BERT embeddings," *CoRR*, vol. abs/1904.03323, 2019. [Online]. Available: http://arxiv.org/abs/1904.03323