CLSP: Linear Algebra Foundations of a Modular Two-Step Convex Optimization-Based Estimator for Ill-Posed Problems

Ilya Bolotov *

Posted Date: 29 October 2025

doi: 10.20944/preprints202509.2192.v2

Keywords: convex optimization; modular estimators; least-squares; generalized inverse; regularization; normalized RMSE; Monte Carlo simulation

Article

CLSP: Linear Algebra Foundations of a Modular Two-Step Convex Optimization-Based Estimator for Ill-Posed Problems

Ilya Bolotov [1,†]

Prague University of Economics and Business, Faculty of International Relations, W. Churchill Sq. 1938/4, 130 67 Prague 3 – Žižkov, Czech Republic; ilya.bolotov@vse.cz or ekonobol@gmail.com

† Current address: Prague University of Economics and Business.

Abstract

This paper develops the linear-algebraic foundations of the Convex Least Squares Programming (CLSP) estimator and constructs its modular two-step convex optimization framework, capable of addressing ill-posed and underdetermined problems. After reformulating a problem in its canonical form, $\mathbf{A}^{(r)}\mathbf{z}^{(r)} = \mathbf{b}$, Step 1 yields an iterated (if $r > 1$) minimum-norm least-squares estimate $\widehat{\mathbf{z}}^{(r)} = (\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbf{b}$ on a constrained subspace defined by a symmetric idempotent $\mathbf{Z}$ (reducing to the Moore-Penrose pseudoinverse when $\mathbf{Z} = \mathbf{I}$). The optional Step 2 corrects $\widehat{\mathbf{z}}^{(r)}$ by solving a convex program, which penalizes deviations using a Lasso/Ridge/Elastic net-inspired scheme parameterized by $\alpha \in [0, 1]$ and yields $\widehat{\mathbf{z}}^{*}$. The second step guarantees a unique solution for $\alpha \in (0, 1]$ and coincides with the Minimum-Norm BLUE (MNBLUE) when $\alpha = 1$. This paper also proposes an analysis of numerical stability and CLSP-specific goodness-of-fit statistics, such as partial R^2, normalized RMSE (NRMSE), Monte Carlo t-tests for the mean of NRMSE, and condition-number-based confidence bands. The three special CLSP problem cases are then tested in a 50,000-iteration Monte Carlo experiment and on simulated numerical examples. The estimator has a wide range of applications, including interpolating input-output tables and structural matrices.

Keywords: convex optimization; modular estimators; least-squares; generalized inverse; regularization; normalized RMSE; Monte Carlo simulation

MSC: 90C25, 65F22, 65F20, 65F35, 65C05

JEL Classification: C13, C61, C63, C15

1. Introduction

As existing literature on optimization — for example, (Nocedal and Wright [1], pp. 1–9) and (Boyd and Vandenberghe [2], pp. 1–2) — points out, constrained optimization plays an important role not only in natural sciences, where it was first developed by Euler, Fermat, Lagrange, Laplace, Gauss, Cauchy, Weierstrass, and others, but also in economics and its numerous applications, such as econometrics and operations research, including managerial planning at different levels, logistics optimization, and resource allocation. The history of modern constrained optimization began in the late 1940s to 1950s, when economics, already heavily reliant on calculus since the marginalist revolution of 1871–1874 ([3], pp. 43–168), and the emerging discipline of econometrics, standing on the shoulders of model fitting ([4], pp. 438–457), internalized a new array of quantitative methods referred to as programming after the U.S. Army's logistic jargon. Programming — including classical, linear, quadratic, and dynamic variants — originated with Kantorovich in economic planning in the USSR and was independently discovered and developed into a mathematical discipline by Dantzig in the United States (who introduced the bulk of terminology and the simplex algorithm) [5] ([6],

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

pp. 1–51). Since the pivotal proceedings published by Koopmans [7], programming became an integral part of mathematical economics [8–10], in both microeconomic analysis (an overview is provided in (Intriligator and Arrow [11], pp. 76–91)) and macroeconomic modeling, such as Leontief's input-output framework outlined in (Koopmans [7], pp. 132–173) and, in an extended form, in (Dorfman et al. [12], pp. 204–264).

Calculus, model fitting, and programming are, however, subject to strong assumptions, including function smoothness, full matrix ranks, well-posedness, well-conditionedness, feasibility, non-degeneracy, and non-cycling (in the case of the simplex algorithm), as summarized by (Nocedal and Wright [1], pp. 304–354, 381–382) and (Intriligator and Arrow [11], pp. 15–76). These limitations have been partially circumvented by developing more strategy-based than analysis-based, computationally intensive constraint programming algorithms; consult Frühwirth and Abdennadher [13] and Rossi et al. [14] for an overview. Still, specific cases of constrained optimization problems in econometric applications, such as incorporating the textbook two-quarter business-cycle criterion into a general linear model [15] or solving underdetermined linear programs to estimate an investment matrix [16], remain unresolved within the existing methodologies and algorithmic frameworks. In response, this text develops a modular two-step convex programming methodology to formalize such problems as a linear system of the form $\mathbf{Az} = \mathbf{b}$, including linear(ized) constraints $\mathbf{Cx} + \mathbf{Sy}^* = \mathbf{b}_{1:k}$, with $k \leq m$, where $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a partitioned matrix, $\mathbf{C} \in \mathbb{R}^{k\times p}$ and $\mathbf{S} \in \mathbb{R}^{k\times(n-p)}$ are its submatrices, $\mathbf{b} \in \mathbb{R}^m$ is a known vector, and $\mathbf{z} \in \mathbb{R}^n$ is the vector of unknowns, which includes both the target solution subvector $\mathbf{x} \in \mathbb{R}^p$ and an auxiliary slack-surplus subvector $\mathbf{y}^* \in \mathbb{R}^{n-p}$, introduced to accommodate inequalities in constraints.

The methodology (Convex Least Squares Programming, or CLSP, framework) comprises selectively omittable and repeatable actions for enhanced flexibility — and consists of *two steps*: (a) the first, grounded in the theory of (constrained) pseudoinverses as summarized in Rao and Mitra [17], Ben-Israel and Greville [18], Lawson and Hanson [19], and Wang et al. [20], is iterable to refine the solution and serves as a mandatory baseline in cases where the second step becomes infeasible (due to mutually inconsistent constraints) — as exemplified by Whiteside et al. [21] for regression-derived constraints and by Blair [22,23] in systems with either too few or too many effective constraints; — and (b) the second, grounded in the theory of regularization and convex programming as summarized in Tikhonov et al. [24], Gentle [4], Nocedal and Wright [1], and Boyd and Vandenberghe [2], provides an optional correction of the first step, drawing on the logic of Lasso, Ridge, and Elastic Net to address ill-posedness and compensate for constraint violations or residual approximation errors resulting from the first-step minimum-norm LS estimate. Hence, it can be claimed that the proposed framework is grounded in the general algorithmic logic of Wolfe [25] and Dax [26], as further formalized by Osborne [27], who were among the first to introduce an additional estimation step into simplex-based solvers, and is comparable to the algorithms of Übi [28,29,30,31,32], the closest and most recent elaborations on the topic.

The aim of this work is to define the linear algebra foundations (the theoretical base) of the CLSP framework, present supporting theorems and lemmas, develop tools for analyzing numerical stability of matrix $\mathbf{A}$, and discuss the goodness of fit of vector $\widehat{\mathbf{z}}^*$, illustrating them on simulated examples. Topics, such as the maximum likelihood estimation of $\widehat{\mathbf{z}}^*$ and corresponding information criteria (Akaike and Bayesian), are beyond its present scope. The text is organized as follows. Section 2 summarizes the historical development and recent advances in convex optimization, motivating the formulation of a new methodology. Section 3 presents the formalization of the estimator, while Section 4 develops a sensitivity analysis for $\mathbf{C}$. Section 5 introduces goodness-of-fit statistics, including the normalized root mean square error (NRMSE) and its sample-mean test. Section 6 presents special cases, and Section 7 reports the results of a Monte Carlo experiment and solves simulated problems via CLSP-based Python 3 modules. Section 8 concludes with a general discussion. Throughout, the following notation is used: bold uppercase letters (e.g., $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{G}, \mathbf{M}, \mathbf{S}$) denote matrices; bold lowercase letters (e.g., $\mathbf{b}, \mathbf{x}, \mathbf{y}, \mathbf{z}$) denote column vectors; italic letters (e.g., $i, j, k, m, n, p, \lambda, \omega$) denote scalars; the transpose

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

of a matrix is denoted by the superscript $\top$ (e.g., $\mathbf{X}^\top$); the inverse by -1 (e.g., $\mathbf{Q}^{-1}$); generalized inverses are indexed using curly braces (e.g., $\mathbf{G}^{\{1,2\}}$); the Moore-Penrose pseudoinverse is denoted by dagger (e.g., $\mathbf{A}^\dagger$); ℓ_p norms, where $1 \leq p \leq \infty$, are denoted by double bars (e.g., $\|\cdot\|_2$); and condition numbers by $\kappa(\cdot)$. All functions, scalars, vectors, and matrices are defined over the real numbers ($\mathbb{R}$).

2. Historical and Conceptual Background of the CLSP Framework

The methodology of convex optimization — formally, $\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$, subject to convex inequality constraints $g_i(\mathbf{x}) \leq 0$ and affine equality constraints $h_j(\mathbf{x}) = 0$, as defined by (Boyd and Vandenberghe [2], pp. 7, 127–129), — has evolved over more than two centuries through several milestones, including generalized inverses and linear and (convex) quadratic programming, each relaxing the strong assumptions of its predecessors. As documented in the seminal works of (Rao and Mitra [17], pp. vii–viii) and (Ben-Israel and Greville [18], pp. 4–5, 370–374), the first *pseudoinverses* (original term, now typically reserved for $\mathbf{A}^\dagger$) or *generalized inverses* (current general term) emerged in the theory of integral operators — introduced by Fredholm in 1903 and further developed by Hurwitz in 1912 — and, implicitly, in the theory of differential operators by Hilbert in 1904, whose work was subsequently extended by Myller in 1906, Westfall in 1909, Bounitzky in 1909, Elliott in 1928, and Reid in 1931. The cited authors attribute their first application to matrices to Moore in 1920 under the term *general reciprocals* [33] (though some sources suggest he may have formulated the idea as early as 1906), with independent formulations by Siegel in 1937 and Rao in 1955, and generalizations to singular operators by Tseng in 1933, 1949, and 1956, Murray and von Neumann in 1936, and Atkinson in 1952–1953. A theoretical consolidation occurred with Bjerhammar's [34] rediscovery of Moore's formulation, followed by Penrose's [35,36] introduction of four conditions defining the unique least-squares minimum-norm generalized inverse $\mathbf{G} \equiv \mathbf{A}^\dagger$, which, such that $\mathbf{G} \in \mathbb{R}^{n\times m}$, can be expressed in Equation (1):

$$\text{(i) } \mathbf{AGA} = \mathbf{A} \quad \text{(ii) } \mathbf{GAG} = \mathbf{G} \quad \text{(iii) } (\mathbf{AG})^\top = \mathbf{AG} \quad \text{(iv) } (\mathbf{GA})^\top = \mathbf{GA} \tag{1}$$

In econometrics, it led to conditionally unbiased (minimum bias) estimators [37] and a redefinition of the Gauss-Markov theorem [38] in the 1960s, superseding earlier efforts [39].

Further contributions to the theory, calculation, and diverse applications of $\mathbf{G}$ were made in the 1950s–early 1970s by Rao and Mitra [17,40] (a synthesis of the authors' previous works from 1955–1969), Ben-Israel and Greville [18], Greville [41,42,43,44], Cline [45], Cline and Greville [46], Golub and Kahan [47], Ben-Israel and Cohen [48], and Lewis and Newman [49] (the most cited sources on the topic in the Web of Science database, although the list is not exhaustive; consult (Rao and Mitra [17], pp. vii–viii) for an extended one), which, among others, introduced the concept of a $\{\cdot\}$-generalized inverse $\mathbf{G} \in \mathbf{A}^{\{\cdot\}}$, where $\mathbf{A}^{\{\cdot\}} \subseteq \mathbb{R}^{n\times m}$ is the space of the $\{\cdot\}$-generalized inverses of $\mathbf{A} \in \mathbb{R}^{m\times n}$, — hereafter, $\{i,j\}$-inverses for $i > 1$, $\{i,j,k\}$-inverses, Drazin inverses, and higher-order $\mathbf{G}$ are disregarded because of their inapplicability to the CLSP framework — by satisfying a subset or all of the (i)–(iv) conditions in Equation (1), as described in Table 1. Formally, for $\mathbf{G}^{\{1\}} \in \mathbf{A}^{\{1\}}$, $\mathbf{I}_n \in \mathbb{R}^{n\times n}$, $\mathbf{I}_m \in \mathbb{R}^{m\times m}$, and arbitrary matrices $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{n\times m}$, any $\{1\}$-inverse, including all $\{1,j\}$-inverses and $\mathbf{A}^\dagger$, can be expressed as $\mathbf{G} = \mathbf{G}^{\{1\}} + (\mathbf{I}_n - \mathbf{G}^{\{1\}}\mathbf{A})\mathbf{U} + \mathbf{V}(\mathbf{I}_m - \mathbf{AG}^{\{1\}})$, out of which only $\mathbf{G} \equiv \mathbf{A}^\dagger$ and a constrained $\mathbf{G} \equiv \mathbf{G}^{\{1,2\}} \in \mathbb{R}^{n\times m}$ — here, the Bott-Duffin inverse [50], expressed in modern notation as $\mathbf{A}^\dagger_\mathbf{S} = \mathbf{P}^L_\mathbf{S}(\mathbf{P}^L_\mathbf{S}\mathbf{A}\mathbf{P}^R_\mathbf{S})^\dagger\mathbf{P}^R_\mathbf{S}$, where $\mathbf{P}^L_\mathbf{S} \in \mathbb{R}^{m\times m}$ and $\mathbf{P}^R_\mathbf{S} \in \mathbb{R}^{n\times n}$ are orthogonal (perpendicular) projectors onto the rows and columns defined by an orthonormal matrix $\mathbf{S}$ — can be uniquely defined ($\mathbf{A}^\dagger_\mathbf{S}$ under certain conditions) and qualify as minimum-norm unbiased estimators in the sense of Chipman [37] and Price [38].

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

Table 1. Selected types of generalized inverses $\mathbf{G} \in \mathbf{A}^{\{\cdot\}}$, where $\mathbf{A}^{\{\cdot\}} \subseteq \mathbb{R}^{n\times m}$, $\mathbf{x} \in \mathbb{R}^n$, and $\mathbf{b} \in \mathbb{R}^m$.

Type of Inverse	Terminology	Properties
$\{1\}$	Equation Solving Inverse	$\mathbf{G} \in \mathbf{A}^{\{1\}}$ iff $\mathbf{AGA} = \mathbf{A}$. Hence, $\mathbf{Gb}$ is a solution to $\mathbf{Ax} = \mathbf{b}$ for all $\mathbf{b} \in \mathcal{R}(\mathbf{A})$.
$\{1,2\}$	Reflexive Inverse	$\mathbf{G} \in \mathbf{A}^{\{1,2\}}$ iff $\mathbf{AGA} = \mathbf{A}$ and $\mathbf{GAG} = \mathbf{G}$. Each $\{1,2\}$-inverse defines a direct sum $\mathcal{N}(\mathbf{A}) \oplus \mathcal{R}(\mathbf{G}) = \mathbb{R}^n$, $\mathcal{R}(\mathbf{A}) \oplus \mathcal{N}(\mathbf{G}) = \mathbb{R}^m$. For complementary subspaces $(\mathcal{N}, \mathcal{R})$, $\mathbf{G}_{\mathcal{N},\mathcal{R}}$ is unique, with $\mathcal{R}(\mathbf{G}_{\mathcal{N},\mathcal{R}}) = \mathcal{R}$ and $\mathcal{N}(\mathbf{G}_{\mathcal{N},\mathcal{R}}) = \mathcal{N}$.
$\{1,3\}$	Least Squares Inverse	$\mathbf{G} \in \mathbf{A}^{\{1,3\}}$ iff $\mathbf{AGA} = \mathbf{A}$ and $(\mathbf{AG})^\top = \mathbf{AG}$. Hence, $\mathbf{Gb}$ is the least-squares solution minimizing the norm $\|\mathbf{Ax} - \mathbf{b}\|_2^2$.
$\{1,4\}$	Minimum Norm Inverse	$\mathbf{G} \in \mathbf{A}^{\{1,4\}}$ iff $\mathbf{AGA} = \mathbf{A}$ and $(\mathbf{GA})^\top = \mathbf{GA}$. Hence, $\mathbf{Gb}$ is the minimum-norm solution of $\mathbf{Ax} = \mathbf{b}$ for all $\mathbf{b} \in \mathcal{R}(\mathbf{A})$.
$\{1,2,3,4\}$	Moore-Penrose Inverse	The unique $\mathbf{G} \equiv \mathbf{A}^\dagger \in \mathbf{A}^{\{1,2,3,4\}}$ is the least-squares minimum-norm solution.

Source: Adapted from Campbell and Meyer (2009, Table 6.1, p. 96) ([51], pp. 91–119) as a summary of (Rao and Mitra [17], pp. 44–71), Rao and Mitra [40], (Ben-Israel and Greville [18], pp. 40–51), and (Wang et al. [20], pp. 10-19, 33-42).

Starting from the early 1960s, Cline [45], (Rao and Mitra [17], pp. 64–71), Meyer [52], (Ben-Israel and Greville [18], pp. 175–200), Hartwig [53], (Campbell and Meyer [51], pp. 53–61), Rao and Yanai [54], Tian [55], Rakha [56], (Wang et al. [20], pp. 193–210), and Baksalary and Trenkler [57], among others, extended the formulas for $\mathbf{A}^\dagger$ to partitioned matrices, including the general row-wise case, $\mathbf{A}_1 \in \mathbb{R}^{m_1\times n}$, $\mathbf{A}_2 \in \mathbb{R}^{m_2\times n}$, $m_1 + m_2 = m$ — and, equivalently, column-wise one, $\mathbf{A}_1 \in \mathbb{R}^{m\times n_1}$, $\mathbf{A}_2 \in \mathbb{R}^{m\times n_2}$, and $n_1 + n_2 = n$ (Equation (2), used in the numerical stability analysis of the CLSP estimator for $\mathbf{C}_{\text{canon}} = [\mathbf{C}\ \ \mathbf{S}]$ in the decomposition $\mathbf{A} = \mathbf{BC}_{\text{canon}} + \mathbf{R}$ given $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{C}_{\text{canon}} \in \mathbb{R}^{k\times n}$, and $\mathbf{B} = \mathbf{AC}_{\text{canon}}^\dagger = \mathbf{A}[\mathbf{C}\ \ \mathbf{S}]^\dagger \in \mathbb{R}^{m\times k}$):

$$\begin{bmatrix}\mathbf{A}_1 & \mathbf{A}_2\end{bmatrix}^\dagger = \left(\begin{bmatrix}\mathbf{A}_1 & \mathbf{A}_2\end{bmatrix}^\top \begin{bmatrix}\mathbf{A}_1 & \mathbf{A}_2\end{bmatrix}\right)^\dagger \begin{bmatrix}\mathbf{A}_1 & \mathbf{A}_2\end{bmatrix}^\top = \begin{bmatrix}\mathbf{A}_1^\top\mathbf{A}_1 & \mathbf{A}_1^\top\mathbf{A}_2 \\ \mathbf{A}_2^\top\mathbf{A}_1 & \mathbf{A}_2^\top\mathbf{A}_2\end{bmatrix}^\dagger \begin{bmatrix}\mathbf{A}_1^\top \\ \mathbf{A}_2^\top\end{bmatrix} \tag{2}$$

where $\operatorname{rank}([\mathbf{A}_1\ \ \mathbf{A}_2]) \le \operatorname{rank}(\mathbf{A}_1) + \operatorname{rank}(\mathbf{A}_2)$, with equality iff $\mathcal{R}(\mathbf{A}_1) \cap \mathcal{R}(\mathbf{A}_2) = \{\mathbf{0}\}$, and strict inequality iff $\mathcal{R}(\mathbf{A}_1) \cap \mathcal{R}(\mathbf{A}_2) \neq \{\mathbf{0}\}$. For the original definitions and formulations, consult (Rao and Mitra [17], pp. 64–66) and (Ben-Israel and Greville [18], pp. 175–200).

To sum up, $\mathbf{G}$ (especially $\mathbf{A}^\dagger$), SVD, and regularization techniques — namely, Lasso (based on the ℓ_1-norm), Tikhonov regularization (hereafter, referred to as Ridge regression, based on the ℓ_2-norm), and Elastic Net (a convex combination of ℓ_1 and ℓ_2 norms) ([4], pp. 477–479) — have become an integral part of estimation (model-fitting), where $\mathbf{A}^\dagger$ can be defined as a *minimum-norm best linear unbiased estimator (MNBLUE)*, reducing to the classical *BLUE* in (over)determined cases. These methods have been applied in (a) natural sciences since the 1940s ([24], pp. 68–156) [34,50] ([58], pp. 85–172) [59–61], and (b) econometrics since the 1960s [37,38] ([19], pp. 36–157, 174–232) ([62], pp. 61–106) ([63], pp. 37–338) in both unconstrained and equality-constrained forms, thereby relaxing previous rank restrictions.

The incorporation of *inequality constraints* into optimization problems during the 1930s–1950s (driven by the foundational contributions of Kantorovich [64], Koopmans [7], and Dantzig [6], along with the later authors [5]) marked the next milestone in the development of convex optimization under the emerging framework of programming, namely for linear cases (LP) — formally, $\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathbb{R}^n} \mathbf{c}^\top\mathbf{x}$ subject to $\mathbf{Ax} \le \mathbf{b}$, $\mathbf{x} \ge \mathbf{0}$ (with the dual $\boldsymbol{\lambda}^* = \arg\min_{\boldsymbol{\lambda}\in\mathbb{R}^m} \mathbf{b}^\top\boldsymbol{\lambda}$ subject to $\mathbf{A}^\top\boldsymbol{\lambda} \ge \mathbf{c}$), where $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{c} \in \mathbb{R}^n$ — and convex quadratic ones

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

(QP) — formally, $\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^n}\left\{\frac{1}{2}\mathbf{x}^\top\mathbf{Q}\mathbf{x}+\mathbf{c}^\top\mathbf{x}\right\}$ subject to $\mathbf{A}\mathbf{x} \leq \mathbf{b}$ (with the dual $\boldsymbol{\lambda}^* = \arg\max_{\boldsymbol{\lambda}\in\mathbb{R}^m_+}\left\{-\frac{1}{2}(\mathbf{A}^\top\boldsymbol{\lambda}+\mathbf{c})^\top\mathbf{Q}^{-1}(\mathbf{A}^\top\boldsymbol{\lambda}+\mathbf{c})-\mathbf{b}^\top\boldsymbol{\lambda}\right\}$), where $\mathbf{Q}\in\mathbb{R}^{n\times n}$ is symmetric positive definite, $\mathbf{c}\in\mathbb{R}^n$, $\mathbf{A}\in\mathbb{R}^{m\times n}$, and $\mathbf{b}\in\mathbb{R}^m$ — ([2], pp. 146–213) (for a classification of programming, consult Figure 1) ([6], pp. 1–31) ([1], pp. 355–597), with Dantzig's simplex and Karmarkar's interior-point algorithms being efficient solvers [65,66].

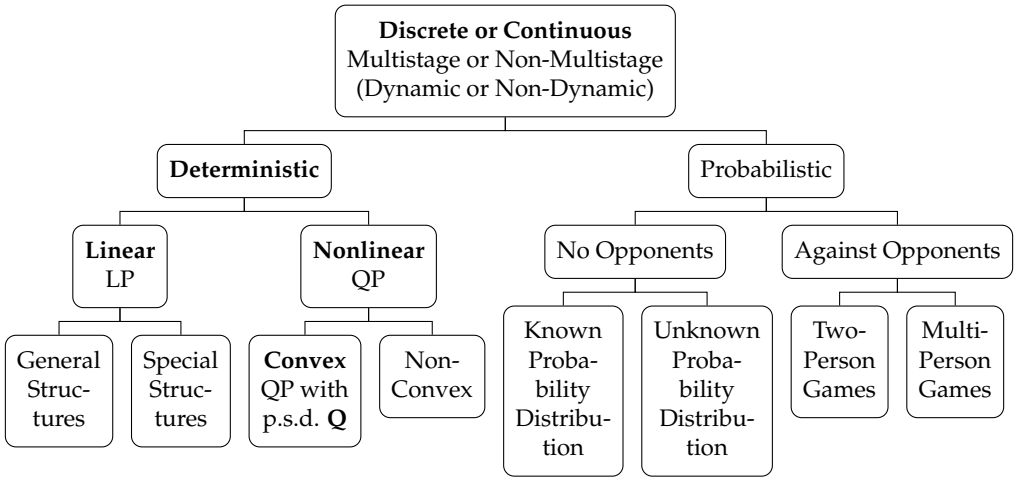

Figure 1. LP and QP in programming. Adapted from Dantzig (1990, Figure 1.4.1, p. 8) ([6], pp. 1–11).

In the late 1950s–1970s, the application of LP and QP extended to LS problems (which can be referred to as a *major generalization of mainstream convex optimization methods*; consult (Boyd and Vandenberghe [2], pp. 291–349) for a comprehensive modern overview) with the pioneering works of Wagner [67] and Sielken and Hartley [68], expanded by Kiountouzis [69] and Sposito [70], who first substituted LS with LP-based (ℓ_1-norm) least absolute (LAD) and (ℓ_∞-norm) least maximum (LDP) deviations and derived unbiased (ℓ_p-norm, $1 < p < \infty$) estimators with non-unique solutions; whilst, in the early 1990s, Stone and Tovey [66] demonstrated algorithmic equivalence between LP algorithms and iteratively reweighted LS. Judge and Takayama [71] and Mantel [72] reformulated (multiple) LS with inequality constraints as QP, and (Lawson and Hanson [19], pp. 144–147) introduced, among others, the non-negative least squares (NNLS) method. These and further developments are reflected in the second step of the CLSP framework, where $\widehat{\mathbf{z}} = \mathbf{G}\mathbf{b}$ is corrected under a regularization scheme inspired by Lasso ($\alpha = 0$), Ridge ($\alpha = 1$), and Elastic Net ($0 < \alpha < 1$) regularization, where $\widehat{\mathbf{z}}^* = \arg\min_{\mathbf{z}\in\mathbb{R}^n}\left\{(1-\alpha)\|\mathbf{z}-\widehat{\mathbf{z}}\|_1 + \alpha\|\mathbf{z}-\widehat{\mathbf{z}}\|_2^2\right\}$, subject to $\mathbf{A}\mathbf{z} = \mathbf{b}$.

The final frontier that motivates the formulation of a new framework is the class of *ill-posed problems* in Hadamard's sense ([4], pp. 143, 241), i.e., systems with no (i) feasibility, $\mathbf{x} \notin \mathcal{F} := \{\mathbf{x} \,|\, \forall_{i,j}\, g_i(\mathbf{x}) \leq 0,\ h_j(\mathbf{x}) = 0\}$, (ii) uniqueness, $\exists\, \mathbf{x}_1 \neq \mathbf{x}_2 : \mathbf{A}\mathbf{x}_1 = \mathbf{A}\mathbf{x}_2$, (iii) stability, $\forall\varepsilon > 0,\ \exists\delta > 0 :\ \|\mathbf{b}-\mathbf{b}_\delta\| < \delta \Rightarrow \left\|\mathbf{A}^\dagger\mathbf{b} - \mathbf{A}^\dagger\mathbf{b}_\delta\right\| \geq \varepsilon$, or formulation — as a well-posed LS, LP, or QP problem — of solution under mainstream assumptions, where $\mathbf{A}\in\mathbb{R}^{m\times n}$, $\mathbf{x}\in\mathbb{R}^n$, and $\mathbf{b}\in\mathbb{R}^m$, such as the ones described in Bolotov [15,16]: namely, (a) underdetermined, $\operatorname{rank}(\mathbf{A}) < \min(m,n)$, and/or ill-conditioned linear systems, $\kappa(\mathbf{A}) = \|\mathbf{A}\|_2\|\mathbf{A}^\dagger\|_2 \gg 1$, $\mathbf{A}\in\mathbb{R}^{m\times n}$, in cases of LS and LP, where solutions are either non-unique or highly sensitive to perturbations [22,73]; and (b) degenerate, $\#\{\text{active constraints at } \mathbf{x}\} > \dim(\mathbf{x}) = n$, with cycling behavior, $c^\top\mathbf{x}^{(k+1)} = c^\top\mathbf{x}^{(k)}$, and $\mathbf{x}^{(k+1)} \neq \mathbf{x}^{(k)}$, $\mathbf{x}\in\mathbb{R}^n$, in all LP and QP problems, where efficient methods, such as the simplex algorithm, may fail to converge [27,74]. Such problems have been addressed since the 1940s–1970s by (a) regularization (i.e., a method of perturbations) and (b) problem-specific algorithms ([2], pp. 455–630). For LS cases, the list includes, among others, Lasso, Ridge regularization, and Elastic Net ([4], pp. 477–479), constrained LS methods [75], restricted LS [76], LP-based sparse recovery approaches [73], Gröbner bases [77], as well as derivatives of eigenvectors and Nelson-type, BD, QR- and SVD-based algorithms [78]. For LS and QP — among others, Wolfe's regularization-based technique for simplex [25] and its modifications [27,74], Dax's LS-based steepest-

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

descent algorithm [26], a primal-dual NNLS-based algorithm (LSPD) [79], as well as modifications of the ellipsoid algorithm [80]. A major early attempt to unify (NN)LS, LP, and QP within a single framework to solve both well- and ill-posed convex optimization problems was undertaken by Übi [28,29,30,31,32], who proposed a series of problem-specific algorithms (VD, INE, QPLS, and LS1). Figure 2 compares all the mentioned methods by functionality.

Problem Class/ Method		LS, based on $\mathbf{A}^{-1}$	LS, based on $\mathbf{A}^{\dagger}$ or $\mathbf{A}_S^{\dagger}$	LP/QP	Regular-ization	Problem-specific
Well-posed unconstrained or equality-constrained	Full-rank	●	●	●	●	●
	Overdetermined	●	●	●	●	●
	Underdetermined		●	●	●	●
Well-posed with inequality constraints	Full-rank			●	●	●
	Overdetermined			●	●	●
	Underdetermined			●	●	●
Ill-posed unconstrained or constrained	Full-rank				●	●
	Overdetermined				●	●
	Underdetermined				●	●

Figure 2. Applicability of selected types of convex optimization methods across problem classes.

To sum up, based on the citation and reference counts in modern literature mapping software (here, Litmaps), as well as in-group relevance (i.e., the number of citations within the collection) of the above-cited works (consult Figure 3), this text incorporates the seminal studies and an almost exhaustive representation of prior research on the topic in question.

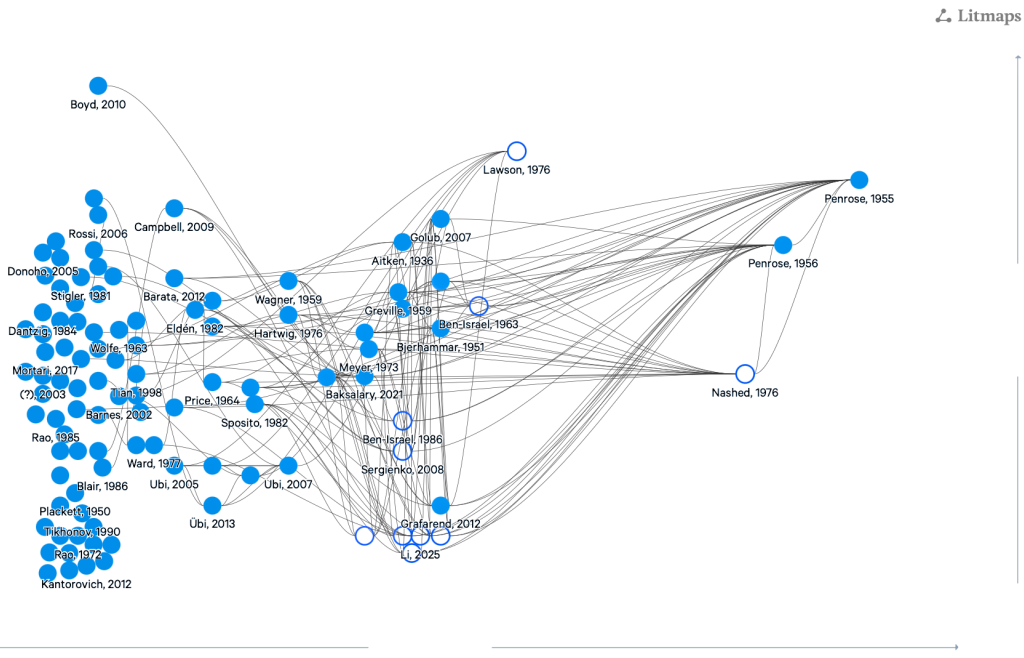

Figure 3. Map of citation and reference counts for the above-cited sources, where hollow circles are suggested items. Monographs not indexed in the database are denoted as **(?)**, with Lawson and Hanson being included in the bibliography as a 1995 reprint [19]. Adapted from Litmaps (Shared Citations & References), available at https://www.litmaps.com/ (accessed on September 30, 2025).

3. Construction and Formalization of the CLSP Estimator

To accommodate the above-described problem classes, a *modular two-step CLSP* consists of a first (compulsory) step — minimum-norm LS estimation, based on $\mathbf{A}^{\dagger}$ or $\mathbf{A}_S^{\dagger}$ (hereafter denoted as $\mathbf{A}_Z^{\dagger}$ to match $\mathbf{z}$), — followed by a second (optional) step, a regularization-inspired correction. This structure

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

ensures that CLSP *is able to yield a unique solution* under a strictly convex second-step correction and *extends the classical LS framework in both scope and precision*, providing a generalized approach that reduces the reliance on problem-specific algorithms [28–32]. The estimator's algorithmic flow is illustrated in Figure 4 and formalized below.

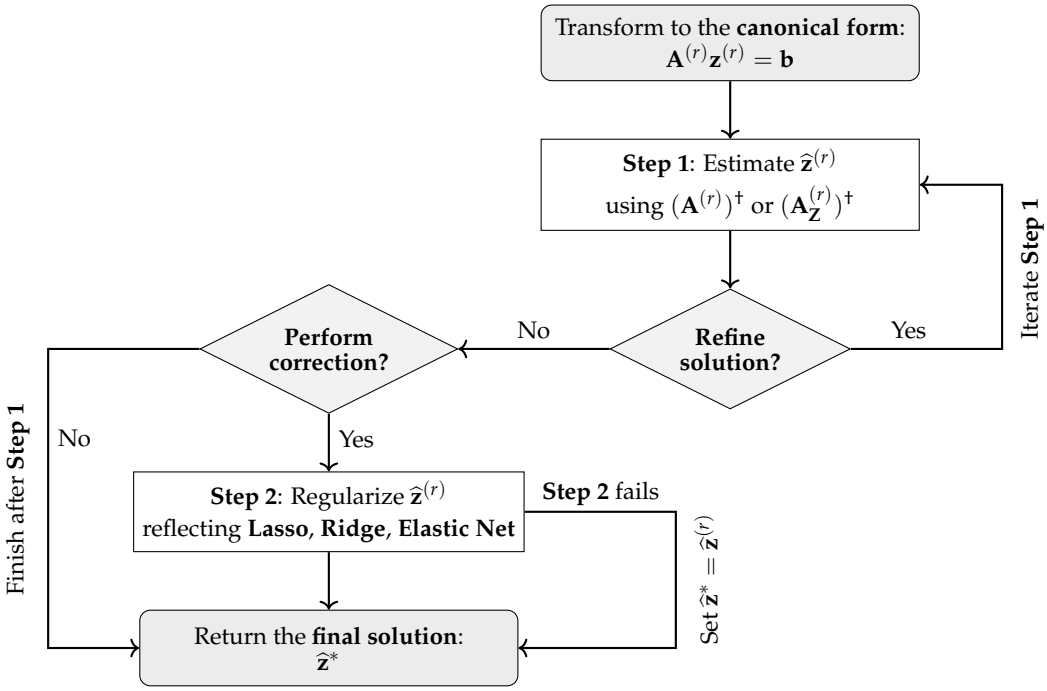

Figure 4. Algorithmic flow of the CLSP Estimator. Decision nodes (diamonds) indicate modularity.

The first process block in the CLSP algorithmic flow denotes the transformation of the initial problem — $\widehat{\mathbf{x}}_M^* = \text{Proj}_{\mathbf{x}_M}\left\{\mathbf{Q}_X[\mathbf{x}_M^{\mathbf{x}_L}] \in \mathbb{R}^p : \forall_{1\le i\le s}\, g_i(\mathbf{x}_M, \mathbf{x}_L) \gtreqless \gamma_i,\ \forall_j\, h_j(\mathbf{x}_M, \mathbf{x}_L) = \eta_j\right\}$, where $\text{Proj}_{\mathbf{x}_M}$ is the projection onto the $\mathbf{x}_M$ coordinates, $\mathbf{x}_M \in \mathbb{R}^{p-l}$ is the vector of model (target) variables to be estimated ($\widehat{\mathbf{x}}_M^*$), $\mathbf{x}_L \in \mathbb{R}^l$, $l \ge 0$, is a vector of latent variables that appear only in the constraint functions $g_i(\cdot)$ and $h_j(\cdot)$, such that $\mathbf{Q}_X[\mathbf{x}_M^{\mathbf{x}_L}] = \mathbf{x} \in \mathbb{R}^p$, $\mathbf{Q}_X$ is the permutation matrix, and $g_i(\cdot) \gtreqless \gamma_i$ and $h_j(\cdot) = \eta_j$ are the linear(ized) inequality and equality constraints, $\forall_{i,j}\gamma_i, \eta_j \in \mathbf{b}$ — to the **canonical form** $\mathbf{A}^{(r)}\mathbf{z}^{(r)} = \mathbf{b}$ (a term first introduced in LP ([6], pp. 75–81)), where (a) $\mathbf{A}^{(r)} \in \mathbb{R}^{m\times n}$ is the block *design matrix* consisting of a $\forall_i g_i(\cdot)$-and-$\forall_j h_j(\cdot)$ *constraints matrix* $\mathbf{C} \in \mathbb{R}^{k\times p}$, a *model matrix* $\mathbf{M} = \mathbf{Q}_M^L \begin{bmatrix} \mathbf{M}_{\text{partial}} \\ \mathbf{0} \end{bmatrix} \mathbf{Q}_M^R \in \mathbb{R}^{(m-k)\times p}$, a sign *slack matrix* $\mathbf{S} = \mathbf{Q}_S^L \begin{bmatrix} \mathbf{S}_{s\subseteq \forall_{\sigma_1,\ldots,\sigma_s\in\{-1,\,+1\}}} \ \text{diag}(\sigma_1,\ldots,\sigma_s) \\ \mathbf{0} \end{bmatrix} \mathbf{Q}_S^R \in \mathbb{R}^{k\times(n-p)}$, and either a *zero matrix* $\mathbf{0}$ (if $r = 1$) or a reverse-sign *slack matrix* $\mathbf{S}_{\text{residual}} = \left[\mathbf{0} \ \ \text{diag}\left(-\text{sign}\left((\mathbf{b} - \mathbf{A}^{(r-1)}\widehat{\mathbf{z}}^{(r-1)})_{k+1:m}\right)\right)\right] \in \mathbb{R}^{(m-k)\times(n-p)}$ (if $r > 1$) (see Equation (3)):

$$\mathbf{A}^{(r)} = \begin{bmatrix} \mathbf{C} & \mathbf{S} \\ \mathbf{M} & \begin{cases} \mathbf{0}, & \text{initial iteration}, \quad r = 1 \\ \mathbf{S}_{\text{residual}}, & \text{subsequent iterations}, \quad r > 1 \end{cases} \end{bmatrix} \tag{3}$$

and (b) $\mathbf{z}^{(r)} = \begin{bmatrix} \mathbf{x}^{(r)} \\ (\mathbf{y}^{(r)})^* \end{bmatrix} \in \mathbb{R}^n$ is *the full vector of unknowns*, comprising (model) $\mathbf{x}^{(r)} \in \mathbb{R}^p$ and (slack) $\left(\mathbf{y}^{(r)}\right)^* = \begin{bmatrix} \mathbf{y}^{(r)} \\ \mathbf{y}_{\text{residual}}^{(r)} \end{bmatrix} \in \mathbb{R}^{n-p}$, $y_i^{(r)} \ge 0$ and $y_j^{(r)} = 0$ — an equivalent problem ([2], pp. 130–132), where $g_i(\cdot)^a = g_i(\cdot) + \mathbf{S}_{ii}y_i^{(r)} = 0$ and $h_j(\cdot)^a = h_j(\cdot)$. The constraint functions $g_i(\cdot)^a$ and $h_j(\cdot)^a$ must be linear(ized), so that $\mathbf{C}, \mathbf{S} \subseteq \mathbf{A}^{(r)} \Leftrightarrow \mathbf{C}\mathbf{x}^{(r)} + \mathbf{S}\left(\mathbf{y}^{(r)}\right)^* = \mathbf{b}_{1:k}$, $k \le m$.

In practical implementations, for problems with $(m > m_E) \vee (n > n_E)$, where E is an estimability limit, a blockwise estimation can be performed through the partitioning $\mathbf{A}^{(r)} = \left\{\mathbf{A}_{subset,\, i_E, j_E}^{(r)}\right\}$,

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

$m_{\text{subset}} = m_{\text{reduced}} - 1$, $n_{\text{subset}} = n_{\text{reduced}} - 1$, $i_E \in \{1, \ldots, \left\lceil \frac{m}{m_{\text{subset}}} \right\rceil\}$, and $j_E \in \{1, \ldots, \left\lceil \frac{n}{n_{\text{subset}}} \right\rceil\}$, leading to a total of $\left(\left\lceil \frac{m}{m_{\text{subset}}} \right\rceil \cdot \left\lceil \frac{n}{n_{\text{subset}}} \right\rceil\right)$ matrices $\mathbf{A}^{(r)}_{\text{reduced}} \in \mathbb{R}^{m_{\text{reduced}} \times n_{\text{reduced}}}$, composed of $\mathbf{A}^{(r)}_{\text{subset}}$ and $\sum a^{(r)}_{i,j}$, $i \notin \mathcal{I}_{\text{subset}} \subset \{1, \ldots, m\}$, $j \notin \mathcal{J}_{\text{subset}} \subset \{1, \ldots, n\}$, and a total of $\left(\left\lceil \frac{m}{m_{\text{subset}}} \right\rceil\right)$ vectors $\mathbf{b}_{\text{r.}} \in \mathbb{R}^{m_{\text{reduced}}}$ (as formalized in Equation (4)):

$$\mathbf{A}^{(r)}_{\text{reduced}} = \mathbf{Q}^{\mathcal{I}}_{A} \begin{bmatrix} \mathbf{A}^{(r)}_{\mathcal{I}_{\text{subset}}, \mathcal{J}_{\text{subset}}} & \sum_{j \notin \mathcal{J}_{\text{subset}}} a^{(r)}_{i,j} \\ \sum_{i \notin \mathcal{I}_{\text{subset}}} a^{(r)}_{i,j} & \sum_{i \notin \mathcal{I}_{\text{subset}}} \sum_{j \notin \mathcal{J}_{\text{subset}}} a^{(r)}_{i,j} \end{bmatrix} \mathbf{Q}^{\mathcal{J}}_{A}, \quad \mathbf{b}_{\text{r.}} = \mathbf{Q}^{\mathcal{I}}_{b} \begin{bmatrix} \mathbf{b}_{\mathcal{I}_{\text{subset}}} \\ \sum_{i \notin \mathcal{I}_{\text{subset}}} b_i \end{bmatrix} \tag{4}$$

$\sum a^{(r)}_{i,j}$ are then treated as slack rows/columns, i.e., incorporated into $\mathbf{S}$, to act as balancing adjustments since the reduced model, by construction, tends to inflate the estimates $\widehat{a}_{i_E, j_E}$. However, full compensation of information loss, caused by aggregation, is infeasible; hence, estimable (smaller) full models are always preferred over reduced (per partes) estimation.

The second process block, i.e., the **first step of the CLSP estimator**, denotes obtaining an (iterated if $r > 1$) estimate $\widehat{\mathbf{z}}^{(r)} = \mathbf{G}^{(r)}\mathbf{b}$ (alternatively, $\left(\left\lceil \frac{m}{m_{\text{subset}}} \right\rceil \cdot \left\lceil \frac{n}{n_{\text{subset}}} \right\rceil\right)$-times $\widehat{\mathbf{z}}^{(r)}_{\text{reduced}} = \mathbf{G}^{(r)}_{\text{reduced}}\mathbf{b}_{\text{r.}}$ with $\widehat{\mathbf{z}}^{(r)}_{i_E, j_E} = \left\{\widehat{\mathbf{z}}^{(r),\,(i_E, j_E)}_{\text{reduced},\,1:m_{\text{subset}},1:n_{\text{subset}}}\right\}$) through the (constrained) Bott-Duffin inverse $\mathbf{G}^{(r)} \equiv (\mathbf{A}^{(r)}_{\mathbf{Z}})^{\dagger} \in \mathbb{R}^{n \times m}$ [50] ([20], pp. 49–64), defined on a subspace specified by a symmetric idempotent matrix $\mathbf{Z} \in \mathbb{R}^{n \times n}$ (regulating solution-space exclusion through binary entries 0/1, such that $\mathbf{P}^{L}_{\mathbf{Z}} = \mathbf{Z}$ restricts the domain of estimated variables to the selected subspace, while $\mathbf{P}^{R}_{\mathbf{Z}} = \mathbf{I}_m$ leaves the data vector, i.e., the input $\mathbf{b}$, unprojected). $\mathbf{G}^{(r)} \equiv (\mathbf{A}^{(r)}_{\mathbf{Z}})^{\dagger}$ reduces to the Moore-Penrose pseudoinverse $\mathbf{G}^{(r)} \equiv (\mathbf{A}^{(r)})^{\dagger} \in \mathbb{R}^{n \times m}$ when $\mathbf{Z} = \mathbf{I}_n$, and equals the standard inverse $\mathbf{G}^{(r)} \equiv (\mathbf{A}^{(r)})^{-1}$ iff $\mathbf{Z} = \mathbf{I}_n \wedge \text{rank}(\mathbf{A}^{(r)}) = m = n$ (the application of a left-sided Bott-Duffin inverse to $\mathbf{A}^{(r)}\mathbf{z}^{(r)} = \mathbf{b}$ is given by Equation (5)):

$$\widehat{\mathbf{z}}^{(r)} = (\mathbf{A}^{(r)}_{\mathbf{Z}})^{\dagger}\mathbf{b} = \left[\left(\mathbf{Z}(\mathbf{A}^{(r)})^{\top}\mathbf{A}^{(r)}\mathbf{Z}\right)^{\dagger}\mathbf{Z}(\mathbf{A}^{(r)})^{\top}\right]\mathbf{b}, \quad \mathbf{Z}^2 = \mathbf{Z} = \mathbf{Z}^{\top} \tag{5}$$

The number of iterations (r) increases until a termination condition is met, such as error $\left|\frac{1}{\sqrt{m}}\left\|(\mathbf{b} - \mathbf{A}^{(r)}\widehat{\mathbf{z}}^{(r)})\right\|_2 - \frac{1}{\sqrt{m}}\left\|(\mathbf{b} - \mathbf{A}^{(r-1)}\widehat{\mathbf{z}}^{(r-1)})\right\|_2\right| < \varepsilon$ or $r > r_{\text{limit}}$, whichever occurs first, — implemented in CLSP-based software, — although any goodness-of-fit metric can be employed. The described condition, maximizing the fit quality, — especially *under a missing second step* — is, however, heuristic, with a formal proof of convergence lying beyond the scope of this work. In practical implementations, $\widehat{\mathbf{z}}^{(r)}$ is efficiently computed using the SVD (see **Appendix A.1**) with optional Ridge regularization to stabilize a nearly singular system.

The third process block, i.e., the **second step of the CLSP estimator**, denotes obtaining a corrected final solution $\widehat{\mathbf{z}}^{*}$ (alternatively, $\left(\left\lceil \frac{m}{m_{\text{subset}}} \right\rceil \cdot \left\lceil \frac{n}{n_{\text{subset}}} \right\rceil\right)$-times $\widehat{\mathbf{z}}^{*}_{\text{reduced}}$ with $\widehat{\mathbf{z}}^{*}_{i_E, j_E} = \left\{\widehat{\mathbf{z}}^{*,\,(i_E, j_E)}_{\text{reduced},\,1:m_{\text{subset}},1:n_{\text{subset}}}\right\}$) — unless the algorithmic flow terminates after Step 1, in which case $\widehat{\mathbf{z}}^{*} = \widehat{\mathbf{z}}^{(r)}$ — by penalizing deviations from the (iterated if $r > 1$) estimate $\widehat{\mathbf{z}}^{(r)}$ reflecting Lasso ($\alpha = 0 \Leftrightarrow \widehat{\mathbf{z}}^{*} = \widehat{\mathbf{z}}^{(r)} + \widehat{\mathbf{u}}$, where $\widehat{\mathbf{u}} = \arg\min_{\mathbf{u}} \|\mathbf{u}\|_1$, s.t. $\mathbf{A}^{(r)}(\widehat{\mathbf{z}}^{(r)} + \mathbf{u}) = \mathbf{b}$), Ridge ($\alpha = 1 \Leftrightarrow \widehat{\mathbf{z}}^{*} = \widehat{\mathbf{z}}^{(r)} + (\mathbf{A}^{(r)})^{\dagger}\mathbf{A}^{(r)}\widehat{\mathbf{u}}$, where $(\mathbf{A}^{(r)})^{\dagger}\mathbf{A}^{(r)} = \mathbf{P}_{\mathcal{R}(\mathbf{A}^{\top})}$, a projector equal to $\mathbf{I}$ iff $\mathbf{A}^{(r)}$ is of full column rank, and $\widehat{\mathbf{u}} = \arg\min_{\mathbf{u}} \|\mathbf{u}\|_2^2$, s.t. $\mathbf{A}^{(r)}(\widehat{\mathbf{z}}^{(r)} + \mathbf{u}) = \mathbf{b}$), and Elastic Net ($0 < \alpha < 1 \Leftrightarrow \widehat{\mathbf{z}}^{*} = \widehat{\mathbf{z}}^{(r)} + (1 - \alpha)\widehat{\mathbf{u}} + \alpha(\mathbf{A}^{(r)})^{\dagger}\mathbf{A}^{(r)}\widehat{\mathbf{u}}$, where $\widehat{\mathbf{u}} = \arg\min_{\mathbf{u}}\{(1 - \alpha)\|\mathbf{u}\|_1 + \alpha\|\mathbf{u}\|_2^2\}$, s.t. $\mathbf{A}^{(r)}(\widehat{\mathbf{z}}^{(r)} + \mathbf{u}) = \mathbf{b}$) ([4], pp. 477–479) ([2], pp. 306–310) (the combination of Lasso, Ridge, and Elastic Net corrections of $\widehat{\mathbf{z}}^{*}$ is given by Equation (6)):

$$\widehat{\mathbf{z}}^{*} = \arg\min_{\mathbf{z} \in \mathbb{R}^n}\left\{(1 - \alpha)\left\|\mathbf{z} - \widehat{\mathbf{z}}^{(r)}\right\|_1 + \alpha\left\|\mathbf{z} - \widehat{\mathbf{z}}^{(r)}\right\|_2^2\right\}, \; \alpha \in [0,\, 1], \quad \text{s.t. } \mathbf{A}^{(r)}\mathbf{z} = \mathbf{b} \tag{6}$$

where the parameter α may be selected arbitrarily or determined from the input $\mathbf{b}$ using an appropriate criterion, such as minimizing prediction error via cross-validation, satisfying model-selection heuristics (e.g., AIC or BIC), or optimizing residual metrics under known structural constraints. Alternatively, α may be fixed at benchmark values, e.g., $\alpha \in \{0,\, \frac{1}{2},\, 1\}$, or based on an error rule, e.g.,

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

$\alpha = \min\left(\frac{\left(\frac{1}{\sqrt{m}}\left\|(\mathbf{b}-\mathbf{A}^{(r)}\widehat{\mathbf{z}}^*)\right\|_2\right)_{\alpha=0}}{\left(\frac{1}{\sqrt{m}}\left\|(\mathbf{b}-\mathbf{A}^{(r)}\widehat{\mathbf{z}}^*)\right\|_2\right)_{\alpha=0}+\left(\frac{1}{\sqrt{m}}\left\|(\mathbf{b}-\mathbf{A}^{(r)}\widehat{\mathbf{z}}^*)\right\|_2\right)_{\alpha=1}+\varepsilon}, 1\right)$, $\varepsilon \to 0^+$, — in CLSP-based software. In practical implementations, $\widehat{\mathbf{z}}^*$, obtained from solving a convex optimization problem, is efficiently computed with the help of numerical solvers.

To sum up, the definition of the process blocks ensures that the CLSP estimator yields a unique final solution $\widehat{\mathbf{z}}^*$ under regularization weight $\alpha \in (0,\ 1]$ owing to the strict convexity of the objective function; see Theorem 1 (although Theorem 1 is mathematically straightforward, it is included for completeness because its corollaries establish the conditions for uniqueness and convergence that underlie special cases). For $\alpha = 0$, the solution remains convex but may not be unique unless additional structural assumptions are imposed. In the specific case of $\alpha = 1$, the estimator reduces to the Minimum-Norm Best Linear Unbiased Estimator (MNBLUE), or, alternatively, to a block-wise $m_{\text{subset}} \times n_{\text{subset}}$-MNBLUE across $\left(\left\lceil\frac{m}{m_{\text{subset}}}\right\rceil \cdot \left\lceil\frac{n}{n_{\text{subset}}}\right\rceil\right)$ reduced problems (for a proof, consult Theorem 2 and its corollaries). It is, however, important to note that — when compared to the classical BLUE — the CLSP, as an MNBLUE estimator, by definition, is not constrained by the same Gauss-Markov conditions and remains well-defined under rank-deficient or underdetermined systems ($m < n$) by introducing the minimum-norm condition as a substitute for the homoscedasticity and full-rank assumptions of the original theorem. Therefore, the MNBLUE represents a generalized or, for ill-posed cases, a problem-class-specific extension of the BLUE, applicable to *systems which violate one or more Gauss-Markov requirements* while still *preserving* (a) *linearity*, (b) *unbiasedness*, and (c) *minimum variance* (as demonstrated in Theorem 2, although a formal theoretical treatment of the MNBLUE is beyond the scope of this work).

Theorem 1. *Let* $\widehat{\mathbf{z}}^{(r)} = (\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbf{b}$ *be the r-iterated estimate obtained in the first step of the CLSP algorithm, and let* $\widehat{\mathbf{z}}^*$ *be the second-step correction obtained through convex programming by solving* $\widehat{\mathbf{z}}^* = \arg\min_{\mathbf{z}\in\mathbb{R}^n}\left\{(1-\alpha)\left\|\mathbf{z}-\widehat{\mathbf{z}}^{(r)}\right\|_1 + \alpha\left\|\mathbf{z}-\widehat{\mathbf{z}}^{(r)}\right\|_2^2\right\}$, $\alpha \in (0,\ 1]$, *s.t.* $\mathbf{A}^{(r)}\mathbf{z} = \mathbf{b}$ *(i.e., the regularization parameter excludes a pure Lasso-type correction), then the final solution* $\widehat{\mathbf{z}}^*$ *is unique.*

Proof. Let $\widehat{\mathbf{z}}^{(r)} = (\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbf{b}$ denote the linear estimator obtained via the Bott-Duffin inverse, defined on a subspace determined by a symmetric idempotent matrix $\mathbf{Z}$, and producing a conditionally unbiased estimate over that subspace. The Bott-Duffin inverse $(\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}$ is given by $(\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger} = \left(\mathbf{Z}(\mathbf{A}^{(r)})^{\top}\mathbf{A}^{(r)}\mathbf{Z}\right)^{\dagger}\mathbf{Z}(\mathbf{A}^{(r)})^{\top}$, and the estimate $\widehat{\mathbf{z}}^{(r)}$ is unique if $\operatorname{rank}(\mathbf{A}^{(r)}\mathbf{Z}) = \dim(\mathcal{R}(\mathbf{Z}))$. In this case, $\widehat{\mathbf{z}}^{(r)}$ is the unique minimum-norm solution in $\mathcal{R}(\mathbf{Z})$. In all other cases, the solution set is affine and given by $\widehat{\mathbf{z}}^{(r)} = (\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbf{b} + \left[\ker(\mathbf{A}^{(r)}\mathbf{Z}) \cap \mathcal{R}(\mathbf{Z})\right]$, where the null-space component represents degrees of freedom not fixed by the constraint $\mathbf{A}^{(r)}\mathbf{Z}$, hence $\widehat{\mathbf{z}}^{(r)}$ is not unique. At the same time, the second-step estimate $\widehat{\mathbf{z}}^*$ is obtained by minimizing the function $(1-\alpha)\left\|\mathbf{z}-\widehat{\mathbf{z}}^{(r)}\right\|_1 + \alpha\left\|\mathbf{z}-\widehat{\mathbf{z}}^{(r)}\right\|_2^2$ over the affine (hence convex and closed) constraint set $\mathcal{F}^{(r)} = \left\{\mathbf{z}\in\mathbb{R}^n \;\middle|\; \mathbf{A}^{(r)}\mathbf{z} = \mathbf{b}\right\}$. Under $\alpha \in (0,\ 1]$, the quadratic term $\alpha\left\|\mathbf{z}-\widehat{\mathbf{z}}^{(r)}\right\|_2^2$ contributes a positive-definite Hessian $2\alpha\,\mathbf{I} \succ 0$ to the objective function, making it a strictly convex parabola-like curvature, and, given $\mathcal{F}^{(r)} \neq \emptyset$, the minimizer $\mathbf{z}^*$ exists and is unique. Therefore, the CLSP estimator with $\alpha \in (0,\ 1]$ yields a unique $\widehat{\mathbf{z}}^*$. □

Corollary 1. *Let the final solution be* $\widehat{\mathbf{z}}^* = \widehat{\mathbf{z}}^{(r)} = (\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbf{b}$, *where* $(\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}$ *denotes the Bott-Duffin inverse on a subspace defined by a symmetric idempotent matrix* $\mathbf{Z}$. *Then (one-step)* $\widehat{\mathbf{z}}^*$ *is unique iff* $\operatorname{rank}(\mathbf{A}^{(r)}\mathbf{Z}) = \dim(\mathcal{R}(\mathbf{Z}))$. *Else, the solution set is affine and given by* $\widehat{\mathbf{z}}^* \in (\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbf{b} + \left[\ker(\mathbf{A}^{(r)}\mathbf{Z}) \cap \mathcal{R}(\mathbf{Z})\right]$, *which implies that the minimizer* $\mathbf{z}^*$ *is not unique, i.e.,* $\#\arg\min(\cdot) > 1$.

Corollary 2. *Let* $\widehat{\mathbf{z}}^*$ *be the final solution obtained in the second step of the CLSP estimator by solving* $\widehat{\mathbf{z}}^* = \arg\min_{\mathbf{z}\in\mathbb{R}^n}\left\{\left\|\mathbf{z}-\widehat{\mathbf{z}}^{(r)}\right\|_1\right\}$, *s.t.* $\mathbf{A}^{(r)}\mathbf{z} = \mathbf{b}$ *(i.e., the regularization parameter is set to* $\alpha = 0$,

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

corresponding to a Lasso correction). Then the solution $\widehat{\mathbf{z}}^*$ *exists and the problem is convex. The solution is unique iff the following two conditions are simultaneously satisfied: (1) the affine constraint set* $\mathcal{F}^{(r)} = \left\{ \mathbf{z} \in \mathbb{R}^n \;\middle|\; \mathbf{A}^{(r)}\mathbf{z} = \mathbf{b} \right\}$ *intersects the subdifferential of* $\|\mathbf{z} - \widehat{\mathbf{z}}^{(r)}\|_1$ *at exactly one point, and (2) the objective function is strictly convex on* $\mathcal{F}^{(r)}$*, which holds when* $\mathcal{F}^{(r)}$ *intersects the interior of at most one orthant in* $\mathbb{R}^n$*. In all other cases, the minimizer* $\mathbf{z}^*$ *is not unique, and the set of optimal solutions forms a convex subset of the feasible region* $\mathcal{F}^{(r)}$*; that is, the final solution* $\widehat{\mathbf{z}}^* \in \arg\min_{\mathbf{z}\in\mathcal{F}^{(r)}} \left\|\mathbf{z} - \widehat{\mathbf{z}}^{(r)}\right\|_1$*, where* $\arg\min(\cdot)$ *is convex and* $\#\arg\min(\cdot) > 1$*.*

Theorem 2. *Let* $\widehat{\mathbf{z}}^{(r)} = (\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbf{b}$ *be the r-iterated estimate obtained in the first step of the CLSP algorithm, and let* $\widehat{\mathbf{z}}^*$ *be the second-step correction obtained through convex programming by solving* $\widehat{\mathbf{z}}^* = \arg\min_{\mathbf{z}\in\mathbb{R}^n} \left\{ \left\|\mathbf{z} - \widehat{\mathbf{z}}^{(r)}\right\|_2^2 \right\}$*, s.t.* $\mathbf{A}^{(r)}\mathbf{z} = \mathbf{b}$ *(i.e., the regularization parameter is* $\alpha = 1$*, corresponding to a Ridge correction), then* $\widehat{\mathbf{z}}^*$ *is the Minimum-Norm Best Linear Unbiased Estimator (MNBLUE) of* $\mathbf{z}$ *under the linear(ized) constraints set by the canonical form of the CLSP.*

Proof. Let $\widehat{\mathbf{z}}^{(r)} = (\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbf{b}$ denote the linear estimator obtained via the Bott-Duffin inverse, defined on a subspace determined by a symmetric idempotent matrix $\mathbf{Z}$, and producing a conditionally unbiased estimate over that subspace. The Bott-Duffin inverse is $(\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger} = \left(\mathbf{Z}(\mathbf{A}^{(r)})^{\top}\mathbf{A}^{(r)}\mathbf{Z}\right)^{\dagger}\mathbf{Z}(\mathbf{A}^{(r)})^{\top}$ and the estimate $\widehat{\mathbf{z}}^{(r)}$ is unique if $\operatorname{rank}(\mathbf{A}^{(r)}\mathbf{Z}) = \dim(\mathcal{R}(\mathbf{Z}))$. Given the linear model derived from the canonical form $\mathbf{b} = \mathbf{A}^{(r)}\mathbf{z}^{(r)} + \boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon}$ are residuals with $\mathbb{E}[\boldsymbol{\varepsilon}] = \mathbf{0}$, it follows that $\mathbb{E}[\widehat{\mathbf{z}}^{(r)}] = \mathbb{E}[(\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbf{b}] = (\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbb{E}[\mathbf{b}] = (\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbf{A}^{(r)}\mathbf{z}^{(r)}$. Substituting $(\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}$ yields $\mathbb{E}[\widehat{\mathbf{z}}^{(r)}] = \left(\mathbf{Z}(\mathbf{A}^{(r)})^{\top}\mathbf{A}^{(r)}\mathbf{Z}\right)^{\dagger}\mathbf{Z}(\mathbf{A}^{(r)})^{\top}\mathbf{A}^{(r)}\mathbf{z}^{(r)}$. By the generalized projection identity $(\mathbf{A}_{\mathbf{Z}}^{(r)})^{\dagger}\mathbf{A}^{(r)}\mathbf{Z} = \mathbf{Z}$, one obtains $\mathbb{E}[\widehat{\mathbf{z}}^{(r)}] = \mathbf{Z}\mathbf{z}^{(r)}$, which proves conditional unbiasedness on $\mathcal{R}(\mathbf{Z})$ (and full unbiasedness if $\mathbf{Z} = \mathbf{I}$). Subsequently, the second-step estimate $\widehat{\mathbf{z}}^*$ is obtained by minimizing the squared Euclidean distance between $\widehat{\mathbf{z}}^{(r)}$ and $\mathbf{z}$, subject to the affine constraint $\mathbf{A}^{(r)}\mathbf{z} = \mathbf{b}$, which corresponds, in geometric and algebraic terms, to an orthogonal projection of $\widehat{\mathbf{z}}^{(r)}$ onto the affine (hence convex and closed) subspace $\mathcal{F}^{(r)} = \left\{ \mathbf{z} \in \mathbb{R}^n \;\middle|\; \mathbf{A}^{(r)}\mathbf{z} = \mathbf{b} \right\}$. By Theorem 1, the unique minimizer $\mathbf{z}^*$ exists and is given explicitly by $\mathbf{z}^* = \widehat{\mathbf{z}}^{(r)} + (\mathbf{A}^{(r)})^{\dagger}\left(\mathbf{b} - \mathbf{A}^{(r)}\widehat{\mathbf{z}}^{(r)}\right)$. This expression satisfies the first-order optimality condition of the convex program and ensures that $\mathbf{z}^*$ is both feasible and closest (in the examined ℓ_2-norm) to $\widehat{\mathbf{z}}^{(r)}$ among all solutions in $\mathcal{F}^{(r)}$. Therefore, $\mathbf{z}^*$ is (1) linear (being an affine transformation of a linear estimator), (2) unbiased (restricted to the affine feasible space), (3) minimum-norm (by construction), and (4) unique (by strict convexity). By the generalized Gauss-Markov theorem ([17], (Chapter 7, Definition 4, p. 139) ([18], (Chapter 8, Section 3.2, Theorem 2, p. 287), it follows that $\widehat{\mathbf{z}}^*$ is the *Minimum-Norm Best Linear Unbiased Estimator (MNBLUE)* under the set of linear(ized) constraints $\mathbf{A}^{(r)}\mathbf{z} = \mathbf{b}$ — i.e., the one with the smallest dispersion (in the Löwner sense) among the class of unbiased minimum second-norm least-squares estimators $\widehat{\mathbf{z}}^* = \arg\min_{\mathbf{z}} \|\mathbf{b} - \mathbf{A}\mathbf{z}\|_2^2 = \mathbf{G}^{\{1,3,4\}}\mathbf{b} + \boldsymbol{\varepsilon}$, where $\boldsymbol{E}(\boldsymbol{\varepsilon}) = 0$ and $\operatorname{Var}(\boldsymbol{\varepsilon}) = 0$ given the Ridge-inspired convex programming correction (residual sterilization), $\widehat{\mathbf{z}}^* = \arg\min_{\mathbf{z}\in\mathbb{R}^n} \left\{ \left\|\mathbf{z} - \widehat{\mathbf{z}}^{(r)}\right\|_2^2 \right\}$, s.t. $\mathbf{A}^{(r)}\mathbf{z} = \mathbf{b}$, ensuring strict satisfaction of the problem constraints $\mathbf{A}^{(r)}\mathbf{z} = \mathbf{b}$, *under the assumption of a feasible Step 2.* □

Corollary 3. *Let the canonical system* $\mathbf{A}^{(r)}\mathbf{z}^{(r)} = \mathbf{b}$ *be consistent and of full column rank,* $\mathbf{A}^{(r)} \in \mathbb{R}^{m\times n}$*,* $m \geq n$*, and* $\operatorname{rank}(\mathbf{A}^{(r)}) = n$*. Suppose that the CLSP algorithm terminates after Step 1 and that* $\mathbf{Z} = \mathbf{I}_n$*, such that* $\widehat{\mathbf{z}}^* = \widehat{\mathbf{z}}^{(r)} = (\mathbf{A}^{(r)})^{\dagger}\mathbf{b} = ((\mathbf{A}^{(r)})^{\top}\mathbf{A}^{(r)})^{-1}(\mathbf{A}^{(r)})^{\top}\mathbf{b}$*. Then, provided the classical linear model* $\mathbf{b} = \mathbf{A}^{(r)}\mathbf{z}^{(r)} + \boldsymbol{\varepsilon}$*, where* $\mathbb{E}[\boldsymbol{\varepsilon}] = \mathbf{0}$ *and* $\operatorname{Cov}(\boldsymbol{\varepsilon}) = \sigma^2\mathbf{I}_m$*, holds, the CLSP estimator is equivalent to OLS while* $\widehat{\mathbf{z}}^*$ *is the Best Linear Unbiased Estimator (BLUE).*

Corollary 4. *Let* $\widehat{\mathbf{z}}_{reduced}^{(r)} = \mathbf{A}^{\dagger}_{\mathbf{Z}_{reduced},reduced}\mathbf{b}_{r.}$ *be the r-iterated estimate obtained via the Bott-Duffin inverse, defined on a subspace determined by a symmetric idempotent matrix* $\mathbf{Z}_{reduced}$*, and producing a conditionally unbiased estimate over that subspace. Subsequently, the estimate* $\widehat{\mathbf{z}}_{reduced}^*$ *is obtained by minimizing the squared*

Euclidean distance between $\widehat{\mathbf{z}}^{(r)}_{reduced}$ *and* $\mathbf{z}_{reduced}$*, subject to the affine constraint* $\mathbf{A}^{(r)}_{reduced}\mathbf{z}_{reduced} = \mathbf{b}_{r.}$*, such that* $\widehat{\mathbf{z}}^{*}_{i_E,j_E} = \left\{\widehat{\mathbf{z}}^{*,(i_E,j_E)}_{reduced,\,1:m_{subset},1:n_{subset}}\right\}$*. Then each* $\widehat{\mathbf{z}}^{*}_{reduced}$ *is the Minimum-Norm Best Linear Unbiased Estimator (MNBLUE) under the linear(ized) constraints defined by the canonical form of each reduced system of the CLSP, and* $\widehat{\mathbf{z}}^{*}$ *is a block-wise* $m_{\text{subset}} \times n_{\text{subset}}$*-MNBLUE corresponding to each of the* $\left(\left\lceil \frac{m}{m_{subset}} \right\rceil \cdot \left\lceil \frac{n}{n_{subset}} \right\rceil\right)$ *reduced problems.*

4. Numerical Stability of the Solutions $\widehat{\mathbf{z}}^{(r)}$ and $\widehat{\mathbf{z}}^{*}$

The numerical stability of solutions, potentially affecting solver convergence in practical implementations, at each step of the CLSP estimator — both the (iterated if $r > 1$) first-step estimate $\widehat{\mathbf{z}}^{(r)}$ (alternatively, $\widehat{\mathbf{z}}^{(r)}_{i_E,j_E} = \left\{\widehat{\mathbf{z}}^{(r),(i_E,j_E)}_{\text{reduced},\,1:m_{\text{subset}},1:n_{\text{subset}}}\right\}$) and the final solution $\widehat{\mathbf{z}}^{*}$ (alternatively, $\widehat{\mathbf{z}}^{*}_{i_E,j_E} = \left\{\widehat{\mathbf{z}}^{*,(i_E,j_E)}_{\text{reduced},\,1:m_{\text{subset}},1:n_{\text{subset}}}\right\}$) — depends on the condition number of the design matrix, $\kappa(\mathbf{A}^{(r)})$, which, given its block-wise formulation in Equation (3), can be analyzed as a function of the *"canonical" constraints matrix* $\mathbf{C}_{\text{canon}} = [\mathbf{C}\ \ \mathbf{S}] \in \mathbb{R}^{k\times n}$ as a fixed, r-invariant, part of $\mathbf{A}^{(r)}$. For both full and reduced problems, **sensitivity analysis** of $\kappa(\mathbf{A}^{(r)})$ **with respect to constraints** (i.e., rows) in $\mathbf{C}_{\text{canon}}$, especially focused on their reduction (i.e., dropping), is performed based on the decomposition (Equations (7) and (8)):

$$\mathbf{A}^{(r)} = \mathbf{B}^{(r)}\mathbf{C}_{\text{canon}} + \mathbf{R}^{(r)} = \mathbf{B}^{(r)}\begin{bmatrix}\mathbf{C} & \mathbf{S}\end{bmatrix} + \mathbf{R}^{(r)}, \quad \mathbf{R}^{(r)} = \mathbf{A}^{(r)}(\mathbf{I} - \mathbf{P}_{\mathcal{R}([\mathbf{C}\ \mathbf{S}]^\top)}) \tag{7}$$

where $\mathbf{B}^{(r)} = \mathbf{A}^{(r)}\mathbf{C}^{\dagger}_{\text{canon}} \in \mathbb{R}^{m\times k}$, denotes the constraint-induced, r-variant, part of $\mathbf{A}$, and, when the rows of $\mathbf{A}^{(r)}$ lie entirely within $\mathcal{R}(\mathbf{C}_{\text{canon}})$, i.e., $\mathbf{A}^{(r)} = \mathbf{A}^{(r)}\mathbf{P}_{\mathcal{R}([\mathbf{C}\ \mathbf{S}]^\top)} \Leftrightarrow \mathbf{R}^{(r)} = \mathbf{0}$,

$$\begin{aligned}
\kappa(\mathbf{A}^{(r)}) &\leq \kappa(\mathbf{B}^{(r)}) \cdot \kappa(\mathbf{C}_{\text{canon}}) \\
\kappa(\mathbf{A}^{(r)}) + \Delta\kappa_A &\leq \left(\kappa(\mathbf{B}^{(r)}) + \Delta\kappa_B\right) \cdot (\kappa(\mathbf{C}_{\text{canon}}) + \Delta\kappa_C) \\
&= \kappa(\mathbf{B}^{(r)}) \cdot \kappa(\mathbf{C}_{\text{canon}}) + \Delta\kappa_B \cdot \kappa(\mathbf{C}_{\text{canon}}) + \kappa(\mathbf{B}^{(r)}) \cdot \Delta\kappa_C + \Delta\kappa_B \cdot \Delta\kappa_C
\end{aligned} \tag{8}$$

i.e., a change in $\kappa(\mathbf{A}^{(r)})$ consists of a (lateral) constraint-side effect, $\kappa(\mathbf{B}^{(r)}) \cdot \Delta\kappa_C$, induced by structural modifications in $\mathbf{C}_{\text{canon}}$, an (axial) model-side effect, $\Delta\kappa_B \cdot \kappa(\mathbf{C}_{\text{canon}})$, reflecting propagation into $\mathbf{B}^{(r)}$ — where $\mathbf{B}^{(r)} = \mathbf{A}^{(r)}\mathbf{C}^{\dagger}_{\text{canon}}$ and $\mathbf{A}^{(r)} = f\left(\mathbf{C}_{\text{canon}}, \mathbf{M}^{(r)}_{\text{canon}}\right)$, $\mathbf{M}^{(r)}_{\text{canon}} = g\left(\mathbf{M}, \boldsymbol{\varepsilon}^{(r-1)}\right)$, and $\boldsymbol{\varepsilon}^{(r-1)} = \mathbf{b} - \mathbf{A}^{(r-1)}\widehat{\mathbf{z}}^{(r-1)}$ (for a "partitioned" decomposition of $\mathbf{B}^{(r)}$ given the structure of $\mathbf{C}^{\dagger}_{\text{canon}}$, see **Appendix A.2**), — and the (non-linear) term, $\Delta\kappa_B \cdot \Delta\kappa_C$.

The condition number of $\mathbf{C}_{\text{canon}} \in \mathbb{R}^{k\times n}$, $\kappa(\mathbf{C}_{\text{canon}})$, the change of which determines the constraint-side effect, $\kappa(\mathbf{B}^{(r)}) \cdot \Delta\kappa_C$, can itself be analyzed with the help of pairwise angular measure between row vectors $\mathbf{c}_i^\top, \mathbf{c}_j^\top \in \mathbf{C}_{\text{canon}}$, where $i, j \in \{1, \ldots, k\}$. For any two $\mathbf{c}_i^\top, \mathbf{c}_j^\top \in \mathbb{R}^{1\times n}$ (alternatively, in a centered form, $\tilde{\mathbf{c}}_i^\top = \mathbf{c}_i^\top - \bar{c}_i$, $\tilde{\mathbf{c}}_j^\top = \mathbf{c}_j^\top - \bar{c}_j \in \mathbb{R}^{1\times n}$, with $\bar{c}_i = \frac{1}{n}\sum_{k=1}^{n} c_{i,k}$, $\bar{c}_j = \frac{1}{n}\sum_{k=1}^{n} c_{j,k}$), $i \neq j$, the cosine of the angle between them $\cos(\theta_{i,j})$ (alternatively, the Pearson correlation coefficient $\rho = \cos(\tilde{\theta}_{i,j})$), captures their angular alignment — values close to ± 1 indicate near-collinearity and potential ill-conditioning, whereas values near 0 imply near-orthogonality and improved numerical stability (for the pairwise definition and its potential aggregated measures, consult Equations (9) and (10)):

$$\forall_{\mathbf{c}_i^\top, \mathbf{c}_j^\top \in \mathbf{C}_{\text{canon}},\, i\neq j} \ \cos(\theta_{i,j}) = \frac{\mathbf{c}_i^\top \cdot \mathbf{c}_j}{\|\mathbf{c}_i^\top\|_2 \cdot \|\mathbf{c}_j^\top\|_2} \in [-1, 1] \tag{9}$$

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

The pairwise angular measure can be aggregated across j (i.e., for each row $\mathbf{c}_i^\top$) and jointly across i, j (i.e., for the full set of $\binom{k}{2} = \frac{k(k-1)}{2}$ unique row pairs of $\mathbf{C}_{\text{canon}}$), yielding a scalar summary statistic of angular alignment, e.g., the *Root Mean Square Alignment* (RMSA):

$$\text{RMSA}_i = \sqrt{\frac{1}{k-1}\sum_{\substack{j=1\\ j\neq i}}^{k}\cos^2(\theta_{i,j})}, \quad \text{RMSA} = \sqrt{\frac{2}{k(k-1)}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\cos^2(\theta_{i,j})} \tag{10}$$

where RMSA_i captures the average angular alignment of constraint i with the rest, while RMSA evaluates the overall constraint anisotropy of $\mathbf{C}_{\text{canon}}$. As with individual cosines $\cos(\theta_{i,j})$ (alternatively, $\rho = \cos(\tilde{\theta}_{i,j})$), values near 1 suggest collinearity and potential numerical instability, while values near 0 reflect angular dispersion and reduced condition number $\kappa(\mathbf{C}_{\text{canon}})$. The use of squared cosines $\cos^2(\theta_{i,j})$ (alternatively, $\rho^2 = \cos^2(\tilde{\theta}_{i,j})$) in contrast with other metrics (e.g., absolute values) maintains consistency with the Frobenius norms in $\kappa(\mathbf{C}_{\text{canon}})$ and assigns greater penalization to near-collinear constraints in $\mathbf{C}_{\text{canon}}$. Furthermore, a combination of RMSA_i, $i \in \{1, \ldots, k\}$, and the above-described *marginal effects*, obtained by excluding row i from $\mathbf{C}_{\text{canon}}$ — namely, $\kappa(\mathbf{B}^{(r)}) \cdot \Delta\kappa_C^{(-i)}$, $\Delta\kappa_B^{(-i)} \cdot \kappa(\mathbf{C}_{\text{canon}})$, and $\Delta\kappa_B^{(-i)} \cdot \Delta\kappa_C^{(-i)}$ if $\mathbf{R}^{(r)} = \mathbf{0}$ — together with the corresponding changes in the *goodness-of-fit statistics* from the solutions $\widehat{\mathbf{z}}^{(r)(-i)}$ and $\widehat{\mathbf{z}}^{*(-i)}$ (consult Section 5), can be visually presented over either all i, or a filtered subset, such that $\text{RMSA}_i \geq$ threshold, as depicted in Figure 5.

Constraint	**RMSA_i (total: ω)**	$\Delta\kappa_C^{(-i)}$	$\Delta\kappa_B^{(-i)}$	$\Delta\kappa_A^{(-i)}$	$\lvert\text{GoF}^{(-i)}\rvert$
##	----------	$\pm\lambda$	$\pm\lambda$	$\pm\lambda$	$\lvert \pm\lambda \rvert$
##	----------	$\pm\lambda$	$\pm\lambda$	$\pm\lambda$	$\lvert \pm\lambda \rvert$
##	----------	$\pm\lambda$	$\pm\lambda$	$\pm\lambda$	$\lvert \pm\lambda \rvert$
##	----------	$\pm\lambda$	$\pm\lambda$	$\pm\lambda$	$\lvert \pm\lambda \rvert$

Figure 5. Schematic correlogram of constraint rows i with high alignment ($\text{RMSA}_i \geq$ threshold). Each row encodes directional alignment and symbolic marginal effects on model conditioning and fit.

To sum up, in the CLSP estimator, a feasible solution $\widehat{\mathbf{z}}^*$ (in limiting cases, $\widehat{\mathbf{z}}^* \equiv \widehat{\mathbf{z}}^{(r)}$), by the theoretical definition of the Bott-Duffin (Moore-Penrose) inverse (consult Section 2), always exists for the canonical form $\mathbf{A}^{(r)}\mathbf{z}^{(r)} = \mathbf{b}$ (therefore, there is no threshold value $\kappa(\mathbf{A}^{(r)}) > \kappa(\mathbf{A}^{(r)})_{\text{threshold}}$ that would imply nonexistence of solution). Still, ill-conditioning of the design matrix $\mathbf{A}^{(r)}$ might hinder the *software-dependent computation of* (a) *the SVD in Step 1* and (b) *the convex optimization problem in Step 2* on an **ad hoc basis**, thereby manifesting the *ill-posedness of the problem*. The proposed decomposition and RMSA_i (RMSA) metric, quantifying the anisotropy of $\mathbf{C}_{\text{canon}}$, provide a means to analyze the origin of instability and optimize the constraint structure of $\mathbf{A}^{(r)}$ — e.g., by mitigating the (numerical) issues discussed in Whiteside et al. [21] and Blair [22,23] — before re-estimation can be performed.

5. Goodness of Fit of the Solutions $\widehat{\mathbf{z}}^{(r)}$ and $\widehat{\mathbf{z}}^*$

Given the hybrid structure of the CLSP estimator — comprising a least-squares-based (iterated if $r > 1$) initial estimate $\widehat{\mathbf{z}}^{(r)}$ (or $\widehat{\mathbf{z}}^{(r)}_{i_E,j_E} = \left\{\widehat{\mathbf{z}}^{(r),(i_E,j_E)}_{\text{reduced},\,1:m_{\text{subset}},1:n_{\text{subset}}}\right\}$), followed by a regularization-inspired final solution $\widehat{\mathbf{z}}^*$ (or $\widehat{\mathbf{z}}^*_{i_E,j_E} = \left\{\widehat{\mathbf{z}}^{*,(i_E,j_E)}_{\text{reduced},\,1:m_{\text{subset}},1:n_{\text{subset}}}\right\}$) — and the potentially ill-posed or underdetermined nature of the problems it addresses, standard **goodness-of-fit methodology** of classical regression — such as explained variance measures (e.g., the coefficient of determination, R^2), error-based metrics (e.g., Root Mean Square Error, RMSE, for comparability with the above-described RMSA), hypothesis tests (e.g., F-tests in the analysis of variance and t-tests at the coefficient level), or confidence intervals (e.g., based on Student's t- and the normal distribution) — are not universally applicable (with the exception of RMSE, these measures are *valid only under classical Gauss-Markov assumptions, i.e. for overdetermined problems where* $\widehat{\mathbf{z}}^* \equiv \widehat{\mathbf{z}}^{(r)}$). The same limitation applies to model-

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

selection criteria, such as R^2_{adjusted}, AIC, and BIC, that presuppose the existence of a maximizable likelihood function and well-defined (i.e., greater than zero) degrees of freedom, not satisfied in the examined class of problems (equally, the formulation of a likelihood function for the two-step CLSP estimator lies beyond the scope of this work). This text, therefore, discusses the applicability of *selected alternative statistics, most of which are robust to underdeterminedness*: partial R^2 (i.e., the adaptation of R^2 to the CLSP structure), normalized RMSE, t-test for the mean of the NRMSE, and a diagnostic interval based on the condition number of $\mathbf{A}^{(r)}$ — implemented in CLSP-based software (see Equations (11)–(18)).

For the *explained variance measures*, the block-wise formulation of the full vector of unknowns $\mathbf{z}^{(r)} = \begin{bmatrix} \mathbf{x}^{(r)} \\ (\mathbf{y}^{(r)})^* \end{bmatrix} \in \mathbb{R}^n$, — obtained from $\mathbf{x}^{(r)} = \mathbf{Q}_X \begin{bmatrix} \mathbf{x}^{(r)}_M \\ \mathbf{x}^{(r)}_L \end{bmatrix} \in \mathbb{R}^p$ (where $\mathbf{x}^{(r)}_M \in \mathbb{R}^{p-l}$ is the vector of model (target) variables and $\mathbf{x}^{(r)}_L \in \mathbb{R}^l$, $l \geq 0$, is a vector of latent variables present only in the constraints) and a vector of slack variables $\left(\mathbf{y}^{(r)}\right)^* \in \mathbb{R}^{n-p}$, — the design matrix $\mathbf{A}^{(r)} = \begin{bmatrix} \mathbf{C}_{\text{canon}} \\ \mathbf{M}^{(r)}_{\text{canon}} \end{bmatrix} \in \mathbb{R}^{m \times n}$, where $\mathbf{M}^{(r)}_{\text{canon}} = [\mathbf{M} \;\; \{\mathbf{0},\, r = 1;\, \mathbf{S}_{\text{residual}},\, r > 1\}] \in \mathbb{R}^{(m-k)\times n}$ is the "canonical" model matrix, and the input $\mathbf{b} = \begin{bmatrix} \mathbf{b}_C \\ \mathbf{b}_M \end{bmatrix} \in \mathbb{R}^m$, all necessitate a **"partial" R^2** (hereafter, the term "partial" will be reserved for statistics related to vector $\mathbf{x}^* \subseteq \mathbf{z}$) to isolate the estimated variables $\widehat{\mathbf{x}}^*$, where, given the vector of ones $\mathbf{1} = \{1, \ldots, 1\}^\top$,

$$R^2_{\text{partial}} = 1 - \frac{\|\mathbf{b}_M - \mathbf{M}\widehat{\mathbf{x}}^*\|_2^2}{\left\|\mathbf{b}_M - \bar{b}_M \mathbf{1}\right\|_2^2}, \quad \bar{b}_M = \frac{1}{m-k} \sum_{i=1}^{m-k} b_{M,i} \tag{11}$$

provided that $\text{rank}(\mathbf{A}^{(r)}) = n$, $m \geq n$, i.e., applicable strictly to (over)determined problems. If $\mathbf{C} = \emptyset \Leftrightarrow k = 0 \wedge \mathbf{y} = \emptyset \Leftrightarrow n = p$, then $\mathbf{A}^{(r)} \equiv \mathbf{M}$ and R^2_{partial} reduces to the classical R^2. Overall, R^2_{partial} has limited use for the CLSP estimator and is provided for completeness.

Next, for the *error-based metrics*, independent of $\text{rank}(\mathbf{A}^{(r)})$ and thus more robust across underdetermined cases (in contrast to R^2_{partial}), a Root Mean Square Error (RMSE) can be defined for both (a) the full solution $\widehat{\mathbf{z}}^*$, via $\text{RMSE} = \frac{1}{\sqrt{m}} \left\| \mathbf{b} - \mathbf{A}^{(r)} \widehat{\mathbf{z}}^* \right\|_2$, serving as an overall measure of fit, and (b) its variable component $\widehat{\mathbf{x}}^*$, via $\text{RMSE}^{(r)}_{\text{partial}} = \frac{1}{\sqrt{m-k}} \|\mathbf{b}_{k+1:m} - \mathbf{M}\widehat{\mathbf{x}}^*\|_2$, quantifying the fit quality with respect to the estimated target variables only (in alignment with R^2_{partial}). Then, to enable cross-model comparisons and especially its use in hypothesis testing (see below), RMSE must be normalized — typically by the standard deviation of the reference input, $\sigma_{\mathbf{b}}$ or $\sigma_{\mathbf{b}_M}$, — yielding the **Normalized Root Mean Square Error (NRMSE)** comparable across datasets and/or models (e.g., in the t-test for the mean of the NRMSE):

$$\text{NRMSE} = \frac{\frac{1}{\sqrt{m}} \left\| \mathbf{b} - \mathbf{A}^{(r)} \widehat{\mathbf{z}}^* \right\|_2}{\sigma_{\mathbf{b}}}, \quad \text{NRMSE}_{\text{partial}} = \frac{\frac{1}{\sqrt{m-k}} \|\mathbf{b}_{k+1:m} - \mathbf{M}\widehat{\mathbf{x}}^*\|_2}{\sigma_{\mathbf{b}_M}} \tag{12}$$

The standard deviation is preferred in statistical methodology over the alternatives (e.g., max-min scaling or range-based metrics) because it expresses residuals in units of their variability, producing a scale-free measure analogous to the standard normal distribution. It is also more common in practical implementations (such as Python's `sklearn` or MATLAB).

Also, for the *hypothesis tests*, classical ANOVA-based F-tests and coefficient-level t-tests — relying on variance decomposition and residual degrees of freedom — are applicable exclusively to the least-squares-based (iterated if $r > 1$) first-step estimate $\widehat{\mathbf{z}}^{(r)}$ and to the final solution $\widehat{\mathbf{z}}^* = \widehat{\mathbf{z}}^{(r)}$, given a strictly overdetermined $\mathbf{A}^{(r)}\mathbf{z}^{(r)} = \mathbf{b}$, i.e., $\text{rank}(\mathbf{A}^{(r)}) = n$, $m > n$ (an OLS case), and under the assumption of homoscedastic and normally distributed residuals. Then, in the case of the F-tests, the test statistics follow the distributions (a) $F \sim \mathcal{F}(q, m-n)$ and (b) $F_{\text{partial}} \sim \mathcal{F}(q_{\text{partial}}, m-n)$, a Wald-type test on a linear restriction — with the degrees of freedom $q = n - 1$ and $q_{\text{partial}} = p - 1$ if $\mathbf{x}^{(r)}$ is a vector of coefficients from a linear(ized) model with an intercept and $q = n$ and $q_{\text{partial}} = p$

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

otherwise, — yielding (a) $H_0 : \widehat{\mathbf{z}}^{(r)} = \mathbf{0}$ and (b) $H_0^{\text{partial}} : \widehat{\mathbf{x}}^{(r)} = \mathbf{R}\widehat{\mathbf{z}}^{(r)} = \mathbf{0}$, where $\mathbf{R} \in \mathbb{R}^{p \times n}$ is a restriction matrix:

$$F = \frac{\frac{1}{q}\left\|\mathbf{A}^{(r)}\widehat{\mathbf{z}}^{(r)}\right\|_2^2}{\frac{1}{m-n}\left\|\mathbf{b} - \mathbf{A}^{(r)}\widehat{\mathbf{z}}^{(r)}\right\|_2^2}, \quad F_{\text{partial}} = \frac{\frac{1}{q_{\text{partial}}}(\mathbf{R}\widehat{\mathbf{z}}^{(r)})^\top \left[\mathbf{R}((\mathbf{A}^{(r)})^\top \mathbf{A}^{(r)})^{-1}\mathbf{R}^\top\right]^{-1}(\mathbf{R}\widehat{\mathbf{z}}^{(r)})}{\frac{1}{m-n}\left\|\mathbf{b} - \mathbf{A}^{(r)}\widehat{\mathbf{z}}^{(r)}\right\|_2^2} \tag{13}$$

In the case of the t-tests, (a) $\forall_{i\in\{1,\ldots,n\}}\ t_i \sim t_{m-n}$ (n test statistics) and (b) $\forall_{i\in\{1,\ldots,p\}}\ t_{\text{partial},i} \equiv t_i$ (p test statistics), given that the partial result $\widehat{\mathbf{x}}^{(r)}$ or $\widehat{\mathbf{x}}^* = \widehat{\mathbf{x}}^{(r)}$ is a subvector of length p of $\widehat{\mathbf{z}}^{(r)}$ (also $\widehat{\mathbf{z}}^* = \widehat{\mathbf{z}}^{(r)}$), yielding (a) $\forall_{i\in\{1,\ldots,n\}}\ H_0 : \widehat{z}_i^{(r)} = 0$ and (b) $\forall_{i\in\{1,\ldots,p\}}\ H_0^{\text{partial}} : \widehat{x}_i^{(r)} = 0$:

$$\forall_{i\in\{1,\ldots,n\}}\ t_i = \frac{\widehat{z}_i^{(r)}}{\left(\frac{1}{\sqrt{m-n}}\left\|\mathbf{b} - \mathbf{A}^{(r)}\widehat{\mathbf{z}}^{(r)}\right\|_2\right) \cdot \sqrt{\left[((\mathbf{A}^{(r)})^\top \mathbf{A}^{(r)})^{-1}\right]_{ii}}}, \quad \forall_{i\in\{1,\ldots,p\}}\ t_{\text{partial},i} \equiv t_i \tag{14}$$

An alternative hypothesis test — robust to both $\text{rank}(\mathbf{A}^{(r)})$ and the step of the CLSP algorithm, i.e., valid for the final solution $\widehat{\mathbf{z}}^*$ (in contrast to the classical ANOVA-based F-tests or coefficient-level t-tests) — can be a ***t*-test for the mean of the NRMSE**, comparing the observed $\text{NRMSE}_{\text{obs}}$ (as μ_0), both full and partial, to the mean of a simulated sample $\text{NRMSE}_{\text{sim}}^{(\tau)}$, generated via *Monte Carlo simulation*, typically from a uniformly (a structureless flat baseline) or normally (a "canonical" choice) distributed random input $\mathbf{b}^{(\tau)} \sim \mathcal{U}(\mathbf{0},\ \mathbf{I}_m)$ or $\mathbf{b}^{(\tau)} \sim \mathcal{N}(\mathbf{0},\ \mathbf{I}_m)$ — both the distributions being standard (the choice of distributions is, therefore, analogous to employing standard weakly or non-informative priors, with the exception of Jeffrey's prior, in Bayesian inference), — for $\tau = 1, \ldots, T$, where T is the sample size, yielding $t \sim t_{T-1}$ with $H_0 : \text{NRMSE}_{\text{obs}} = \overline{\text{NRMSE}}_{\text{sim}}$ and $H_1 : \text{NRMSE}_{\text{obs}} \neq \overline{\text{NRMSE}}_{\text{sim}}$ for the two-sided test and $H_1 : \text{NRMSE}_{\text{obs}} \gtrless \overline{\text{NRMSE}}_{\text{sim}}$ for the one-sided one:

$$t = \frac{\overline{\text{NRMSE}}_{\text{sim}} - \text{NRMSE}_{\text{obs}}}{\sqrt{\frac{1}{T(T-1)}\sum_{\tau=1}^{T}\left(\text{NRMSE}_{\text{sim}}^{(\tau)} - \overline{\text{NRMSE}}_{\text{sim}}\right)^2}}, \quad \overline{\text{NRMSE}}_{\text{sim}} = \frac{1}{T}\sum_{\tau=1}^{T}\text{NRMSE}_{\text{sim}}^{(\tau)} \tag{15}$$

justified when $\mathbf{b}^{(\tau)}$ is normally distributed or approximately normally distributed based on the Lindeberg-Lévy Central Limit Theorem (CLT), i.e., when $T > 30$ (as a rule of thumb) for $\mathbf{b}^{(\tau)} \sim \mathcal{U}(\mathbf{0},\ \mathbf{I}_m)$. Then, H_0 denotes a good fit for $\widehat{\mathbf{z}}^*$ in the sense that $\text{NRMSE}_{\text{obs}}$ does not significantly deviate from the simulated distribution, i.e., H_0 should not be rejected for the CLSP model to be considered statistically consistent. In practical implementations (such as Python, R, Stata, SAS, Julia, and MATLAB/Octave), T typically ranges from 50 to 50,000.

Finally, for the *confidence intervals*, classical formulations for (a) $z_i^{(r)}$, $i \in \{1, \ldots, n\}$, and (b) $x_i^{(r)}$, $i \in \{1, \ldots, p\}$, — based on Student's t- and, asymptotically, the standard normal distribution, $t \sim t_{m-n}$ and $z \sim \mathcal{N}(0,\ 1)$, — are equivalently exclusively applicable to the least-squares-based (iterated if $r > 1$) first-step estimate $\widehat{\mathbf{z}}^{(r)}$ and to the final solution $\widehat{\mathbf{z}}^* = \widehat{\mathbf{z}}^{(r)}$ under a strictly overdetermined $\mathbf{A}^{(r)}\mathbf{z}^{(r)} = \mathbf{b}$, i.e., $\text{rank}(\mathbf{A}^{(r)}) = n$, $m > n$ (an OLS case), and assuming homoscedastic and normally distributed residuals. Then, provided $\alpha \in (0,\ 1)$, such as $\alpha \in \{0.01,\ 0.05,\ 0.1\}$, the confidence intervals for (a) $z_i^{(r)}$ and (b) $x_i^{(r)}$ are:

$$\begin{aligned} \forall_{i\in\{1,\ldots,n\}}\ z_i^{(r)} &\in \widehat{z}_i^{(r)} \pm t_{m-n,1-\alpha/2} \cdot \frac{1}{\sqrt{m-n}}\left\|\mathbf{b} - \mathbf{A}^{(r)}\widehat{\mathbf{z}}^{(r)}\right\|_2 \cdot \sqrt{\left[((\mathbf{A}^{(r)})^\top \mathbf{A}^{(r)})^{-1}\right]_{ii}}, \\ \forall_{i\in\{1,\ldots,p\}}\ x_i^{(r)} &\equiv z_i^{(r)} \end{aligned} \tag{16}$$

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

given that the partial result $\widehat{\mathbf{x}}^{(r)}$ or $\widehat{\mathbf{x}}^* = \widehat{\mathbf{x}}^{(r)}$ is a subvector of length p of $\widehat{\mathbf{z}}^{(r)}$ or $\widehat{\mathbf{z}}^* = \widehat{\mathbf{z}}^{(r)}$. For $n > 30$, the distribution of the test statistic, $t_{m-n,1-\alpha/2}$, approaches a standard normal one, $Z_{1-\alpha/2}$, based on the CLT, yielding an alternative notation for the confidence intervals:

$$\begin{aligned} \forall_{i\in\{1,\ldots,n\}}\ z_i^{(r)} &\in \widehat{z}_i^{(r)} \pm Z_{1-\alpha/2} \cdot \frac{1}{\sqrt{m-n}} \left\| \mathbf{b} - \mathbf{A}^{(r)}\widehat{\mathbf{z}}^{(r)} \right\|_2 \cdot \sqrt{\left[((\mathbf{A}^{(r)})^\top \mathbf{A}^{(r)})^{-1} \right]_{ii}}, \\ \forall_{i\in\{1,\ldots,p\}}\ x_i^{(r)} &\equiv z_i^{(r)} \end{aligned} \tag{17}$$

An alternative "confidence interval" — equivalently robust to both $\operatorname{rank}(\mathbf{A}^{(r)})$ and the step of the CLSP algorithm, i.e., valid for the final solution $\widehat{\mathbf{z}}^*$ (in contrast to the classical Student's t- and, asymptotically, the standard normal distribution-based formulations) — can be *constructed deterministically via condition-weighted bands*, relying on componentwise numerical sensitivity. Let the residual vector $\mathbf{b} - \mathbf{A}^{(r)}\widehat{\mathbf{z}}^*$ be a (backward) perturbation in $\mathbf{b}$, i.e., $\Delta\mathbf{b} = \mathbf{b} - \mathbf{A}^{(r)}\widehat{\mathbf{z}}^*$. Then, squaring both sides of the classical first-order inequality $\frac{\|\Delta\widehat{\mathbf{z}}^*\|_2}{\|\widehat{\mathbf{z}}^*\|_2} \leq \kappa(\mathbf{A}^{(r)}) \cdot \frac{\|\Delta\mathbf{b}\|_2}{\|\mathbf{b}\|_2}$ yields $\frac{\sum_{i=1}^n (\Delta\widehat{z}_i^*)^2}{\sum_{i=1}^n (\widehat{z}_i^*)^2} \leq \kappa(\mathbf{A}^{(r)})^2 \cdot \frac{\left\|\mathbf{b}-\mathbf{A}^{(r)}\widehat{\mathbf{z}}^*\right\|_2^2}{\|\mathbf{b}\|_2^2}$, where $\kappa(\mathbf{A}^{(r)}) = \|\mathbf{A}^{(r)}\|_2 \|(\mathbf{A}^{(r)})^\dagger\|_2 = \frac{\sigma_{\max}(\mathbf{A}^{(r)})}{\sigma_{\operatorname{rank}(\mathbf{A}^{(r)})}(\mathbf{A}^{(r)})}$ — $\sigma_{\max}$ and $\sigma_{\operatorname{rank}(\mathbf{A}^{(r)})}$ being the biggest and smallest singular values of $\mathbf{A}^{(r)}$ — iff $\sigma_{\operatorname{rank}(\mathbf{A}^{(r)})} > 0 \wedge \|\mathbf{b}\|_2 > 0$. Under a uniform relative squared perturbation (a heuristic allocation of error across components as a simplification, not an assumption) of $\mathbf{z}^*$, rearranging terms and taking square roots of both sides gives $\forall_{i\in\{1,\ldots,n\}}\ |\Delta\widehat{z}_i^*| \leq |\widehat{z}_i^*| \cdot \kappa(\mathbf{A}^{(r)}) \cdot \frac{\left\|\mathbf{b}-\mathbf{A}^{(r)}\widehat{\mathbf{z}}^*\right\|_2}{\|\mathbf{b}\|_2}$ and a condition-weighted "confidence" band:

$$\forall_{i\in\{1,\ldots,n\}}\ z_i^* \in \widehat{z}_i^* \cdot \left(1 \pm \kappa(\mathbf{A}^{(r)}) \cdot \frac{\left\| \mathbf{b} - \mathbf{A}^{(r)}\widehat{\mathbf{z}}^* \right\|_2}{\|\mathbf{b}\|_2} \right), \quad \forall_{i\in\{1,\ldots,p\}}\ x_i^* \equiv z_i^* \tag{18}$$

which is a **diagnostic interval based on the condition number** for $\mathbf{z}^*$, consisting of (1) a canonical-form system conditioning, $\kappa(\mathbf{A}^{(r)})$, (2) a normalized model misfit, $\frac{\left\|\mathbf{b}-\mathbf{A}^{(r)}\widehat{\mathbf{z}}^*\right\|_2}{\|\mathbf{b}\|_2}$, and (3) the "scale" of the final solution, $\widehat{\mathbf{z}}^*$, without probabilistic assumptions. A perturbation in one or more of these three components, e.g., caused by a change in RMSA resulting from dropping selected constraints (consult Section 4), will affect the limits of $\mathbf{z}^*$. Under a perfect fit, the interval collapses to $z_i^* \in \widehat{z}_i^* \pm 0$ and, for $\kappa(\mathbf{A}^{(r)}) \gg 1$, tends to be very wide. Overall, in practical implementations, where the squared perturbations may violate the uniformity simplification, an aggregated (e.g., average) width of the diagnostic interval for vectors $\mathbf{z}^*$ and $\mathbf{x}^*$ becomes more informative, as it represents an *adjusted goodness-of-fit statistic* — normalized error weighted by the condition number of the design matrix $\mathbf{A}^{(r)}$.

To sum up, the CLSP estimator requires a goodness-of-fit framework that reflects both its algorithmic structure and numerical stability. Classical methods remain informative but are valid only under strict overdetermination ($m > n$), full-rank design matrix $\mathbf{A}^{(r)}$, and distributional assumptions (primarily, homoscedasticity and normality of residuals). In contrast, the proposed alternatives — (a) NRMSE and $\text{NRMSE}_{\text{partial}}$, (b) t-tests for the mean of NRMSE with the help of Monte Carlo sampling, and (c) condition-weighted confidence bands — are robust to underdeterminedness and ill-posedness, making them preferrable in practical implementations (e.g., in existing CLSP-based software for Python, R, and Stata).

6. Special Cases of CLSP Problems: APs, CMLS, and LPRLS/QPRLS

The structure of the *design matrix* $\mathbf{A}^{(r)}$ in the CLSP estimator, as defined in Equation (3) (consult Section 3), allows for accommodating a selection of special-case problems, out of which three cases are covered (each case will use modified notation for i, j, m, and p), but the list may not be exhaustive. Among others, the CLSP estimator is, given its ability to address ill-posed or underdetermined problems under linear(ized) constraints, efficient in addressing what can be referred to as **allocation problems (APs)** (or, for flow variables, **tabular matrix problems, TMs**) — i.e., in most

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

cases, underdetermined problems involving matrices of dimensions $m \times p$ to be estimated, subject to known row and column sums, with the degrees of freedom (i.e., nullity) equal to $n - \text{rank}(\mathbf{A}^{(r)}) \geq mp + s^* - (\frac{m}{i} + \frac{p}{j}) - q$, where $0 \leq s^* \leq (\frac{m}{i} + \frac{p}{j}) + q$ is the number of active (non-zero) slack variables and $0 \leq q \leq \text{rank}(\mathbf{M}_{\text{partial}}) \leq \min(k, mp)$ is the number of known model (target) variables (e.g., a zero diagonal), — whose design matrix, $\mathbf{A}^{(r)} \in \mathbb{R}^{(\frac{m}{i}+\frac{p}{j}+k)\times n}$, comprises a (by convention) *row-sum-column-sum constraints matrix* $\mathbf{C} = \left[\left(\left(\mathbf{I}_{\frac{m}{i}} \otimes \mathbf{1}_i\right) \otimes \mathbf{1}_p\right)^\top \middle| \left(\mathbf{1}_m \otimes \left(\mathbf{I}_{\frac{p}{j}} \otimes \mathbf{1}_j\right)\right)^\top\right]^\top \in \mathbb{R}^{(\frac{m}{i}+\frac{p}{j})\times mp}$ (where $\mathbf{1} = \{1, \ldots, 1\}$), a *model matrix* $\mathbf{M} = \mathbf{Q}_M^L \begin{bmatrix} \mathbf{M}_{\text{partial}} \\ \mathbf{0} \end{bmatrix} \mathbf{Q}_M^R \in \mathbb{R}^{k\times mp}$ — in a trivial case, $\mathbf{M}_{\text{partial}} \subseteq \mathbf{I}_{mp}$ — a sign *slack matrix* $\mathbf{S} = \mathbf{Q}_S^L \begin{bmatrix} \mathbf{S}_{s\subseteq\forall_{\sigma_1,\ldots,\sigma_s\in\{-1,\,+1\}}} & \text{diag}(\sigma_1,\ldots,\sigma_s) \\ \mathbf{0} \end{bmatrix} \mathbf{Q}_S^R \in \mathbb{R}^{(\frac{m}{i}+\frac{p}{j})\times(n-mp)}$, and either a *zero matrix* $\mathbf{0}$ (if $r = 1$) or a (standard) reverse-sign *slack matrix* $\mathbf{S}_{\text{residual}} \in \mathbb{R}^{k\times(n-mp)}$ (provided $r > 1$) (as given by Equation (19) extending Equation (3)):

$$\mathbf{A}^{(r)} = \begin{bmatrix} \begin{bmatrix} \left(\mathbf{I}_{\frac{m}{i}} \otimes \mathbf{1}_i\right) \otimes \mathbf{1}_p \\ \mathbf{1}_m \otimes \left(\mathbf{I}_{\frac{p}{j}} \otimes \mathbf{1}_j\right) \end{bmatrix} & \mathbf{S} \\ \mathbf{M} & \begin{cases} \mathbf{0}, & \text{initial iteration}, \quad r = 1 \\ \mathbf{S}_{\text{residual}}, & \text{subsequent iterations}, \quad r > 1 \end{cases} \end{bmatrix} \tag{19}$$

where $\frac{m}{i}$ and $\frac{p}{j}$ denote groupings of the m rows and p columns, respectively, into i and j homogeneous blocks (when no grouped row or column sums are available, $i = j = 1$); with real-world examples including: (a) input-output tables (national, inter-country, or satellite); (b) structural matrices (e.g., trade, country-product, investment, or migration); (c) financial clearing and budget-balancing; and (d) data interpolations (e.g., quarterly data from annual totals). Given the available literature in 2025, the first pseudoinverse-based method of estimating input-output tables was proposed in Pereira-López et al. [81] and the first APs (TMs)-based study was conducted in Bolotov [16], attempting to interpolate a world-level (in total, 232 countries and dependent territories) "bilateral flow" matrix of foreign direct investment (FDI) for the year 2013, based on known row and column totals from UNCTAD data (aggregate outward and inward FDI). The estimation employed a minimum-norm least squares solution under APs (TMs)-style equality constraints, based on the Moore-Penrose pseudoinverse of a "generalized Leontief structural matrix", rendering it equivalent to the initial (non-corrected) solution in the CLSP framework, $\widehat{\mathbf{z}}^{(1)} = \mathbf{A}_{\mathbf{Z}}^{(1)\dagger}\mathbf{b}$. As a rule of thumb, AP (TM)-type problems are rank deficient for $m > 2 \wedge p > 2$ with the CLSP being a *unique* (if $\alpha \in (0,\ 1]$) and an *MNBLUE* (if $\alpha = 1$) estimator (consult Section 3).

In addition to the APs (TMs), the CLSP estimator is also efficient in addressing what can be referred to as **constrained-model least squares (CMLS)** (or, more generally, **regression problems, RPs**) — i.e., (over)determined problems involving vectors of dimension p to be estimated with (standard) OLS degrees of freedom, subject to, among others, linear(ized) data matrix $\mathbf{D} \in \mathbb{R}^{k\times p}$-related (in)equality constraints, $\forall_{i\in\{1,\ldots,t\}}\ \mathbf{C}_i \equiv \mathbf{T}_i\mathbf{D} \in \mathbb{R}^{m_i\times p}$, where $\mathbf{T}_i \in \mathbb{R}^{m_i\times k}$ is a transformation matrix of $\mathbf{D}$, such as the i-th difference or shift (lag/lead), and $\mathbf{C}_i\mathbf{x} \gtreqless \gamma_i$, — whose design matrix, $\mathbf{A}^{(r)} \in \mathbb{R}^{(\sum_{i=1}^u m_i+k)\times n}$, consists of a *u-block constraints matrix* $\mathbf{C} = \begin{bmatrix} \mathbf{C}_1 \\ \cdots \\ \mathbf{C}_u \end{bmatrix} \in \mathbb{R}^{(\sum_{i=1}^u m_i)\times p}$ ($\forall_{i\in\{1,\ldots,u\}}\ \mathbf{C}_i \in \mathbb{R}^{m_i\times p}$), $u \geq t$, a data matrix as the *model matrix* $\mathbf{M} \equiv \mathbf{D} \in \mathbb{R}^{k\times p}$, a (standard) sign *slack matrix* $\mathbf{S} \in \mathbb{R}^{(\sum_{i=1}^u m_i)\times(n-p)}$, and

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

either a *zero matrix* $\mathbf{0}$ (if $r = 1$) or a (standard) reverse-sign *slack matrix* $\mathbf{S}_{\text{residual}} \in \mathbb{R}^{k\times(n-p)}$ (when $r > 1$) (as given by Equation (20) extending Equation (3), with $\mathbf{M}$ substituted by data matrix $\mathbf{D}$):

$$\mathbf{A}^{(r)} = \begin{bmatrix} \begin{bmatrix} \mathbf{C}_1 \\ \dots \\ \mathbf{C}_u \end{bmatrix} & \mathbf{S} \\ \mathbf{D} & \begin{cases} \mathbf{0}, & \text{initial iteration}, \quad r = 1 \\ \mathbf{S}_{\text{residual}}, & \text{subsequent iterations}, \quad r > 1 \end{cases} \end{bmatrix} \tag{20}$$

with real-world examples including: (a) previously infeasible textbook-definition econometric models of economic (both micro and macro) variables (e.g., business-cycle models), (b) additional constraints applied to classical econometric models (e.g., demand analysis). Given the available literature in 2025, the first studies structurally resembling CMLS (RPs) were conducted in Bolotov [15,82], focusing on the decomposition of the long U.S. GDP time series into trend (y_t) and cyclical (c_t) components — using exponential trend, moving average, Hodrick-Prescott filter, and Baxter-King filter — under constraints on its first difference, based on the U.S. National Bureau of Economic Research (NBER) delimitation of the U.S. business cycle. c_t was then modeled with the help of linear regression (OLS), based on an n-th order polynomial with extreme values smoothed via a factor of $\frac{1}{10^{n-i}}$ and simultaneously penalized via an n-th or $(n+1)$-th root, $c_t = \beta_0 + \beta_1 \int \left(\sqrt[\{n,\, n+1\}]{\prod_{i=1}^{i=n}(t - t_i) \cdot \frac{1}{10^{n-i}}} \right) dt + \varepsilon_t$, where β_0, β_1 are the model parameters, $\forall_{1\le i\le n}\, t_i$ are the values of the (externally given) stationary points, and ε_t is the error. The order of the polynomial and the ad-hoc selection of smoothing and penalizing factors, however, render such a method inferior to the CLSP. Namely, the *unique* (if $\alpha \in (0,\ 1]$) and an *MNBLUE* (if $\alpha = 1$) CLSP estimator (consult Section 3) allows (a) presence of inequality constraints and (b) their ill-posed formulations.

Finally, the CLSP estimator can be used to address (often unsolvable using classical solvers) underdetermined and/or ill-posed — caused by too few and/or mutually inconsistent or infeasible constraints, as in the sense of Whiteside et al. [21] and Blair [22,23], — LP and QP problems, an approach hereafter referred to as **linear/quadratic programming via regularized least squares (LPRLS/QPRLS)**: CLSP substitutes the original objective function of the LP/QP problem with the canonical form $\mathbf{A}^{(r)}\mathbf{z}^{(r)} = \mathbf{b}$ (i.e., focusing solely on the problem's constraints, without distinguishing between the LP and QP cases) with $\widehat{\mathbf{x}}^*_M = \text{Proj}_{\mathbf{x}^{(r)}_M}\left\{ \mathbf{Q}_X \begin{bmatrix} \mathbf{x}^{(r)}_M \\ \mathbf{x}^{(r)}_L \end{bmatrix} \in \mathbb{R}^p : \forall_{1\le i\le s}\, g_i(\mathbf{x}^{(r)}_M, \mathbf{x}^{(r)}_L) \gtreqless \gamma_i,\ \forall_j\, h_j(\mathbf{x}^{(r)}_M, \mathbf{x}^{(r)}_L) = \eta_j \right\} \subseteq \mathbf{z}^*$ being the solution of the original problem — where $\mathbf{Q}_X$ is a permutation matrix, and $g_i(\cdot) \gtreqless \gamma_i$ and $h_j(\cdot) = \eta_j$ represent the linear(ized) inequality and equality constraints, $\forall_{i,j}\gamma_i, \eta_j \in \mathbf{b}$ — and the degrees of freedom being equal — under the classical formalization of a primal LP/QP problem (consult Section 2) — to $n - \text{rank}(\mathbf{A}^{(r)}) \ge (p + s) - (m_{ub} + m_{eq} + m_{nn}) = p - m_{eq}$, where p is the length of $\mathbf{x}^{(r)}_M$, $s = m_{ub} + m_{nn}$ is the number of introduced slack variables, while $m_{ub} \ge 0$, $m_{eq} \ge 0$, and $m_{nn} \equiv p$ denote the numbers of upper-bound, equality, and non-negativity constraints, respectively (under the standard assumption that all model variables are constrained to be non-negative). In the LPRLS/QPRLS, the design matrix, $\mathbf{A}^{(r)} \in \mathbb{R}^{(m_{ub}+m_{eq}+m_{nn})\times n}$, by the definition of LP/QP, is r-invariant and consists of a block constraints matrix $\mathbf{C} = \begin{bmatrix} \mathbf{C}_{ub} \\ \mathbf{C}_{eq} \\ \mathbf{C}_{nn} \end{bmatrix} \in \mathbb{R}^{(m_{ub}+m_{eq}+m_{nn})\times p}$, where $\mathbf{C}_{ub} \in \mathbb{R}^{m_{ub}\times p}$, $\mathbf{C}_{eq} \in \mathbb{R}^{m_{eq}\times p}$, and $\mathbf{C}_{nn} \equiv \mathbf{I}_p \in \mathbb{R}^{p\times p}$, a (standard) sign *slack matrix* $\mathbf{S} \in \mathbb{R}^{(m_{ub}+m_{eq}+m_{nn})\times(n-p)}$, and $\mathbf{M}_{\text{canon}} = \emptyset$ (as given by Equation (21) extending Equation (3), under the condition $\mathbf{M} = \mathbf{S}_{\text{residual}} = \emptyset$):

$$\mathbf{A}^{(r)} = \mathbf{A}^{(1)} = \begin{bmatrix} \begin{bmatrix} \mathbf{C}_{ub} \\ \mathbf{C}_{eq} \\ \mathbf{C}_{nn} \end{bmatrix} & \mathbf{S} \end{bmatrix} \tag{21}$$

Given the available literature in 2025, the first documented attempts to incorporate LS into LP and QP included, among others, Dax's LS-based steepest-descent algorithm [26] and the primal-dual NNLS-

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

based algorithm (LSPD) [79], whose structural restrictions — in terms of well-posedness and admissible constraints — can be relaxed through LPRLS/QPRLS, still guaranteeing a *unique* (if $\alpha \in (0,\ 1]$) and an *MNBLUE* (if $\alpha = 1$) solution (consult Section 3).

To sum up, the CLSP framework can be applied to three special classes of problems — allocation (APs (TMs)), constrained-model least squares (CMLS (RPs)), and regularized linear or quadratic programming (LPRLS/QPRLS) — differing only in the nature of their constraint blocks, treatment of slack or residual components, and rank properties of their design matrix $\mathbf{A}^{(r)}$, while the estimation method and goodness-of-fit statistics remain identical. The three examined cases correspond to the overwhelming majority of potential real-world uses of the estimator, whereas custom (non-described) problem types are relatively scarce. The AP (TM) case is of particular methodological importance as a means of estimating, among others, input-output tables (without reliance on historical observations).

7. Monte Carlo Experiment and Numerical Examples

This work will finally demonstrate the capability of the CLSP estimator to undergo large-scale Monte Carlo experiments across its special cases — namely, the AP (TM), CMLS (RP), and LPRLS/QPRLS problems (illustrated below on the example of the APs (TMs)) — which can be replicated to assess its explanatory power and to calibrate key parameters (r and α) for real-world applications in future research. However, the construction of a complete econometric framework — requiring not only the results of such calibration, but also the formal derivation of its maximum likelihood function (for model selection) and an extension of the Gauss-Markov theorem (to explicitly incorporate MNBLUE cases) — lies beyond the scope of this theoretical text, whose aim is to establish the linear-algebraic foundations of the estimator and to discuss its numerical stability and potential measures of its goodness of fit. Therefore, the Monte Carlo experiment is complemented by simulated numerical examples serving as a proof of the estimator's practical implementability and the estimability of its diagnostic metrics (i.e., RMSA, NRMSE, t-tests, and diagnostic intervals).

Proceeding to the Monte Carlo experiment, the standardized form of $\mathbf{C}$ in the APs (TMs), i.e., $\mathbf{C} = f(m, i, p, j)$, with given $\mathbf{S}$ and r, enables **large-scale simulation of the distribution of NRMSE** — a robust goodness-of-fit metric for the CLSP — through *repeated random trials* under varying matrix dimensions $m \times p$, row and column group sizes i and j, and composition of $\mathbf{M}$, inferring asymptotic percentile means and standard deviations for practical implementations (e.g., the t-test for the mean). This text presents the results of a simulation study on the distribution of the NRMSE from the first-step estimate $\widehat{\mathbf{z}}^{(1)}$ (implementable in standard econometric software without CLSP modules) for $i = 1$, $j = 1$, $\mathbf{S} = \emptyset$, $\mathbf{Z} = \mathbf{I}_{m+p+k} \Leftrightarrow (\mathbf{A}_{\mathbf{Z}}^{(1)})^{\dagger} = (\mathbf{A}^{(1)})^{\dagger}$, and $r = 1$, conducted in Stata/MP (set to version 14.0) with the help of Mata's 128-bit floating-point cross-product-based `quadcross()` for greater precision and SVD-based `svsolve()` with a strict tolerance of `c("epsdouble")` for increased numerical stability, and a random-variate seed `123456789` (see **Appendix B.1** containing a list of dependencies [83] and the Stata .do-file). In this 50,000-iterations Monte Carlo simulation study, spanning matrix dimensions from 1×2 and 2×1 up to 50×50, random normal input vectors $\mathbf{b} \sim \mathcal{N}(\mathbf{0},\ \mathbf{I}_{m+p+k})$ were generated for each run, applied with and without zero diagonals (i.e., if, under $l = 0 \Leftrightarrow \mathbf{x}_L = \emptyset$, $\mathbf{x}_M = \mathbf{x} = \mathbf{z}_{1:mp} \in \mathbb{R}^{mp}$ is reshaped into a $(m \times p)$-matrix $\mathbf{X}_M = \mathbf{X} \in \mathbb{R}^{m\times p}$, then $\forall_{i\in\{1,\ldots,\min(m,p)\}}\ \mathbf{X}_{ii} = 0$). Thus, a total of 249.9 million observations ($= (50 \cdot 50 - 1) \cdot 2 \cdot 50{,}000$) was obtained via the formula $\text{NRMSE} = \frac{1}{\sqrt{m}} \|\mathbf{b} - \mathbf{A}^{(1)} \left((\mathbf{A}^{(1)})^{\dagger} \mathbf{b} \right) \|_2 / \sigma_{\mathbf{b}}$, resulting in 4,998 aggregated statistics of the asymptotic distribution, assuming convergence: `mean`, `sd`, `skewness`, `kurtosis`, `min`, `max`, as well as `p1`-`p99`, as presented in Figure 6 (depicting the intensity of its first two moments, `mean` and `sd`) and Table 2 (reporting a subset of all thirteen statistics for $m, p \bmod 10 = 0$).

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

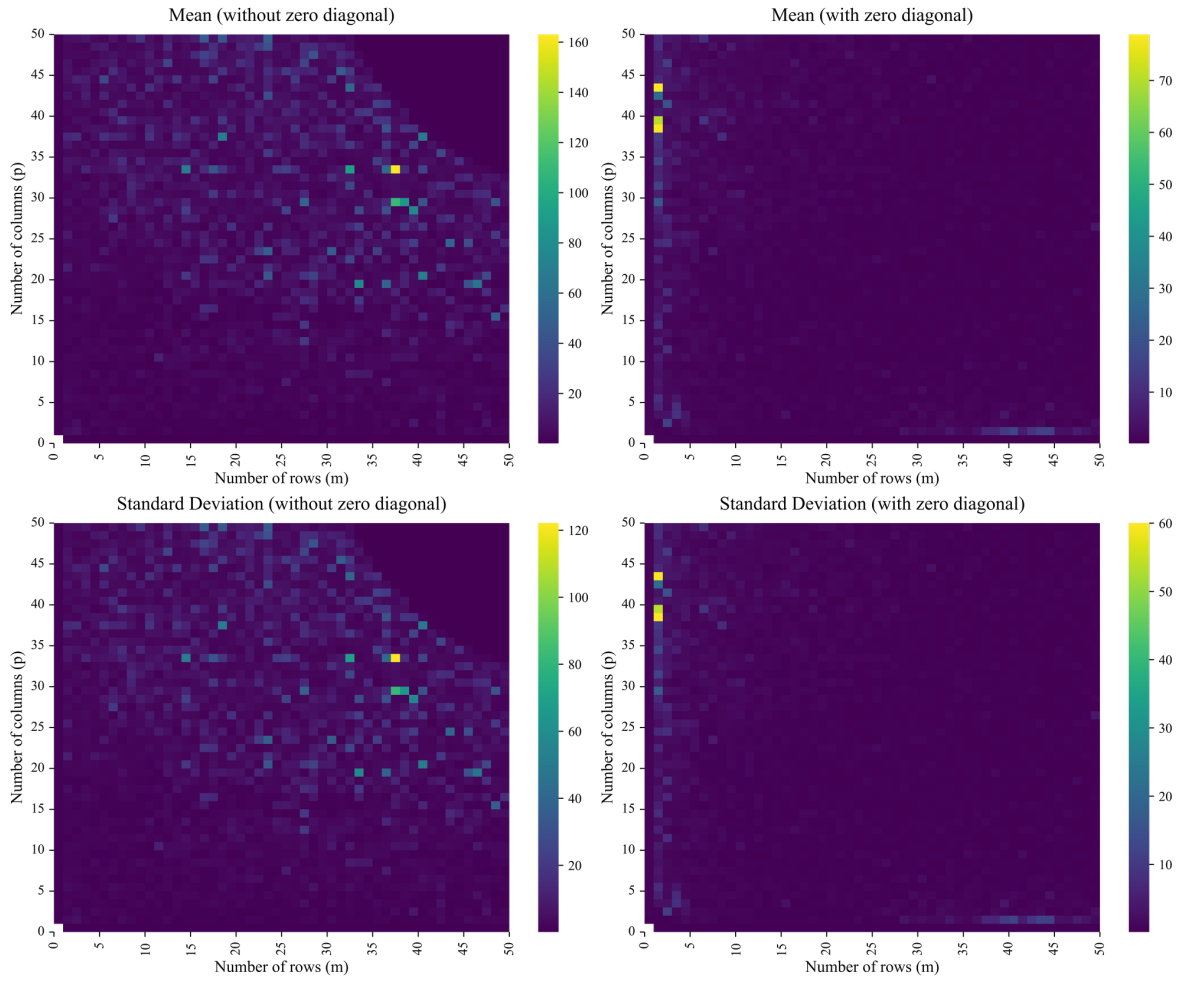

Figure 6. Heatmaps (based on `seaborn`'s `viridis` colormap) of means, $\texttt{mean} = \frac{1}{50{,}000}\sum_{i=1}^{50{,}000}\text{NRMSE}_i$, and standard deviations, $\texttt{sd} = \sqrt{\frac{1}{50{,}000-1}\sum_{i=1}^{50{,}000}(\text{NRMSE}_i - \texttt{mean})^2}$, computed from the NRMSE of $\widehat{\mathbf{z}}^{(1)} = (\mathbf{A}^{(1)})^{\dagger}\mathbf{b}$, where $\mathbf{x} = \text{vec}(\mathbf{X} \in \mathbb{R}^{m\times p},\ 1 \leq m, p \leq 50) \subseteq \mathbf{z}$, under two structural variants: with and without a zero diagonal ($\forall_{i\in\{1,\ldots,\min(m,p)\}}\ \mathbf{X}_{ii} = 0$). Color reflects the magnitude of the statistic, with lighter shades indicating higher values. The figure was prepared using `matplotlib` in Python 3.

From the results of the Monte Carlo experiment, it is observable that: (1) the NRMSE from the first-step estimate $\widehat{\mathbf{z}}^{(1)}$ and its distributional statistics exhibit increasing stability and boundedness as matrix dimensions $m \times p$ increase — specifically, for $m, p \gg 1$, both the `mean` and `sd` of NRMSE tend to converge, indicating improved estimator performance (e.g., under no diagonal restrictions, at $m = p = 5$, $\texttt{mean} = 1.84$ and $\texttt{sd} = 1.29$, while at $m = p = 45$, $\texttt{mean} = 0.21$ and $\texttt{sd} = 0.16$); (2) the zero-diagonal constraints ($\forall_{i\in\{1,\ldots,\min(m,p)\}}\ \mathbf{X}_{ii} = 0$) reduce the degrees of freedom and lead to a much more uniform and dimensionally stable distribution of NRMSE across different matrix sizes (e.g., at $m = p = 5$, `mean` drops to 1.17 and `sd` to 0.85, while at $m = p = 45$, `mean` rises slightly to 0.26 and `sd` to 0.19); and (3) the distribution of NRMSE exhibits mild right skew and leptokurtosis — less so for the zero-diagonal case: under no diagonal restriction, the average `skewness` is 0.97 with a coefficient of variation of 38.16% (i.e., > 0.00), and the average `kurtosis` is 4.15 with a variation of 304.61%; whereas under the zero-diagonal constraint, `skewness` averages 0.95 with only 8.57% variation, and `kurtosis` averages 3.76 with 25.94% variation (i.e., > 3.00). To sum up, the larger and less underdetermined (i.e., in an AP (TM) problem, the larger the $\mathbf{M}^{(1)}$ block of $\mathbf{A}^{(1)}$) the canonical system $\mathbf{A}^{(1)}\mathbf{z}^{(1)} = \mathbf{b}$, the lower the estimation error (i.e., the mean NRMSE) and its variance, and, provided a stable limiting law for NRMSE exists, the skewness and kurtosis from the simulations drift toward 0 and 3, respectively, consistent with — though not proving — a convergence toward a standard-normal distribution. This leads to a straightforward conclusion that the CLSP estimator's

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

Step 1 performance improves monotonically with increasing problem size and rank, indicating that larger and better-conditioned systems yield more stable normalized errors, thereby allowing future research to concentrate on the calibration of the parameters r and, on inclusion of Step 2, α.

Table 2. Subset (at $m, p \bmod 10 = 0$) of the summary statistics for NRMSE from the Simulation Study.

m	p	mean	sd	skewness	kurtosis	min	max	p1	p5	p25	p50	p75	p95	p99
Without Zero Diagonal														
10	10	000.81	000.59	000.82	003.24	000.00	003.52	000.01	000.06	000.33	000.70	001.18	001.94	002.47
10	20	001.53	001.14	000.92	003.60	000.00	008.10	000.02	000.12	000.62	001.31	002.21	003.70	004.80
10	30	005.30	003.96	000.95	003.69	000.00	027.25	000.09	000.41	002.13	004.51	007.65	012.95	016.90
10	40	007.56	005.67	000.97	003.77	000.00	046.23	000.12	000.61	003.02	006.45	010.90	018.59	024.18
10	50	025.08	018.95	000.99	003.84	000.00	127.63	000.39	001.98	009.99	021.10	036.31	061.64	080.75
20	10	000.57	000.43	000.93	003.60	000.00	002.73	000.01	000.04	000.23	000.49	000.83	001.39	001.79
20	20	007.90	005.85	000.89	003.48	000.00	040.71	000.13	000.64	003.18	006.77	011.48	019.17	024.57
20	30	004.10	003.03	000.92	003.61	000.00	021.59	000.06	000.33	001.68	003.50	005.92	009.92	012.87
20	40	005.85	004.37	000.95	003.71	000.00	031.73	000.09	000.47	002.36	005.00	008.44	014.27	018.65
20	50	008.52	006.36	000.94	003.63	000.00	043.37	000.14	000.68	003.44	007.22	012.36	020.73	026.90
30	10	002.89	002.16	000.97	003.78	000.00	015.48	000.05	000.23	001.16	002.47	004.17	007.05	009.19
30	20	002.85	002.11	000.92	003.58	000.00	014.28	000.05	000.23	001.16	002.43	004.12	006.88	008.93
30	30	004.44	003.32	000.94	003.66	000.00	022.66	000.07	000.35	001.78	003.76	006.43	010.85	014.08
30	40	008.73	006.50	000.95	003.69	000.00	048.55	000.14	000.69	003.56	007.47	012.61	021.13	027.85
30	50	005.61	004.22	000.96	003.69	000.00	031.96	000.09	000.44	002.25	004.75	008.10	013.70	017.94
40	10	000.55	000.41	000.98	003.80	000.00	003.10	000.01	000.04	000.22	000.47	000.79	001.35	001.76
40	20	004.67	003.48	000.93	003.66	000.00	025.54	000.07	000.35	001.87	003.97	006.74	011.34	014.83
40	30	001.39	001.04	000.95	003.69	000.00	006.70	000.02	000.11	000.56	001.18	002.01	003.40	004.39
40	40	000.27	000.20	000.96	003.71	000.00	001.48	000.00	000.02	000.11	000.23	000.39	000.67	000.87
40	50	000.29	000.21	000.95	003.69	000.00	001.58	000.00	000.02	000.11	000.24	000.41	000.70	000.91
50	10	003.74	002.82	000.99	003.84	000.00	023.34	000.06	000.29	001.50	003.16	005.37	009.20	012.13
50	20	001.65	001.24	000.95	003.68	000.00	008.17	000.03	000.13	000.67	001.40	002.39	004.05	005.26
50	30	005.24	003.90	000.94	003.69	000.00	028.36	000.08	000.42	002.12	004.47	007.56	012.74	016.54
50	40	000.47	000.35	000.97	003.73	000.00	002.40	000.01	000.04	000.19	000.40	000.67	001.15	001.49
50	50	000.23	000.17	000.94	003.64	000.00	001.21	000.00	000.02	000.09	000.20	000.33	000.56	000.73
With Zero Diagonal														
10	10	000.71	000.52	000.85	003.37	000.00	003.20	000.01	000.06	000.29	000.62	001.04	001.71	002.20
10	20	000.69	000.51	000.92	003.60	000.00	003.54	000.01	000.05	000.28	000.59	001.00	001.67	002.16
10	30	001.93	001.44	000.95	003.74	000.00	010.17	000.03	000.15	000.78	001.64	002.77	004.69	006.13
10	40	004.65	003.50	000.97	003.76	000.00	023.68	000.07	000.37	001.86	003.92	006.72	011.43	014.89
10	50	001.94	001.47	000.98	003.84	000.00	012.42	000.03	000.15	000.77	001.64	002.81	004.75	006.25
20	10	000.70	000.52	000.92	003.59	000.00	003.70	000.01	000.06	000.28	000.60	001.02	001.70	002.20
20	20	000.70	000.52	000.93	003.57	000.00	003.68	000.01	000.05	000.28	000.60	001.01	001.71	002.21
20	30	000.42	000.31	000.94	003.68	000.00	002.24	000.01	000.03	000.17	000.36	000.61	001.02	001.34
20	40	000.27	000.20	000.94	003.64	000.00	001.27	000.00	000.02	000.11	000.23	000.39	000.65	000.85
20	50	002.06	001.55	000.97	003.81	000.00	011.72	000.03	000.16	000.83	001.75	002.98	005.04	006.64
30	10	000.45	000.34	000.95	003.67	000.00	002.40	000.01	000.04	000.18	000.38	000.65	001.10	001.43
30	20	000.42	000.32	000.95	003.71	000.00	002.16	000.01	000.03	000.17	000.36	000.61	001.03	001.34
30	30	000.45	000.34	000.96	003.76	000.00	002.53	000.01	000.03	000.18	000.38	000.65	001.10	001.43
30	40	000.34	000.26	000.97	003.73	000.00	001.88	000.01	000.03	000.14	000.29	000.50	000.85	001.10
30	50	000.57	000.43	000.96	003.75	000.00	003.21	000.01	000.05	000.23	000.48	000.82	001.39	001.81
40	10	000.34	000.26	000.96	003.76	000.00	001.83	000.01	000.03	000.14	000.29	000.49	000.84	001.08
40	20	000.33	000.25	000.98	003.77	000.00	001.68	000.01	000.03	000.13	000.28	000.47	000.80	001.05
40	30	000.63	000.47	000.97	003.80	000.00	003.36	000.01	000.05	000.25	000.53	000.90	001.52	001.99
40	40	000.38	000.28	000.98	003.82	000.00	002.23	000.01	000.03	000.15	000.32	000.54	000.92	001.20
40	50	000.30	000.23	000.97	003.78	000.00	001.76	000.00	000.02	000.12	000.26	000.44	000.75	000.98
50	10	000.61	000.46	000.97	003.74	000.00	003.16	000.01	000.05	000.24	000.52	000.88	001.48	001.95
50	20	000.17	000.13	000.97	003.76	000.00	000.87	000.00	000.01	000.07	000.14	000.24	000.41	000.54
50	30	000.32	000.24	000.97	003.82	000.00	002.10	000.00	000.03	000.13	000.28	000.47	000.80	001.03
50	40	000.61	000.46	000.95	003.67	000.00	003.04	000.01	000.05	000.24	000.52	000.89	001.49	001.95
50	50	000.68	000.51	000.96	003.70	000.00	003.76	000.01	000.05	000.27	000.58	000.98	001.66	002.16

Source: self-prepared.

For an illustration of the implementability of the estimator for the APs (TMs), CMLS (RPs), and LPRLS/QPRLS, the author's cross-platform Python 3 module, `pyclsp` (version $\geq$ 1.6.0, available on PyPI) [84], — together with its interface co-modules for the APs (TMs), `pytmpinv` (version $\geq$ 1.2.0, available on PyPI) [85], and the LPRLS/QPRLS, `pylppinv` (version $\geq$ 1.3.0, available on PyPI) [86] (a co-module for the CMLS (RPs) is not provided, as the variability of $\forall_{i\in\{1,\ldots,t\}}\ \mathbf{C}_i \equiv \mathbf{T}_i\mathbf{D}$ within the $\mathbf{C} = \begin{bmatrix}\mathbf{C}_1\\ \cdots \\ \mathbf{C}_u\end{bmatrix}$, $u \geq t$, block of $\mathbf{A}^{(r)}$ prevents efficient generalization), — is employed

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

for a sample (dataset) of i random elements $\mathcal{X}_i$ (scalars, vectors, or matrices depending on context) and their transformations, drawn independently from the standard normal distribution, i.e., $\{\mathcal{X}_1, \ldots, \mathcal{X}_i\} \overset{\text{iid}}{\sim} \mathcal{N}(0,\, 1)$, under the mentioned random-variate seed `123456789` (configured for consistency with the above-described Monte Carlo experiment). Thus, using the cases of (a) a (non-negative symmetric) input-output table and (b) a (non-negative) zero-diagonal trade matrix as two **AP (TM) numerical examples**, this text simulates problems similar to the ones addressed in Pereira-López et al. [81] and Bolotov [16]: an underdetermined $\mathbf{X} \in \mathbb{R}^{m \times p}$ is estimated — from row sums, column sums, and k known values of two randomly generated matrices (a) $\mathbf{X}_{\text{true}} = 0.5 \cdot \left(|\mathcal{X}_1| + |\mathcal{X}_1|^\top\right)$ and (b) $\mathbf{X}_{\text{true},i,j} = (|\mathcal{X}_1|)_{i,j}\left(1 - \mathbb{1}_{\{i=j \leq \min(m,p)\}}\right)$, $(\mathcal{X}_1)_{i,j} \overset{\text{iid}}{\sim} \mathcal{N}(0,\, 1)$, subject to (a) $\mathbf{X}_{i,j} \geq 0$ and (b) $\mathbf{X}_{i,j} \geq 0 \wedge \forall_{i \in \{1,\ldots,\min(m,p)\}}\, \mathbf{X}_{ii} = 0$ — via CLSP assuming $\mathbf{x} = \text{vec}(\mathbf{X}) \subseteq \mathbf{z}$, $\mathbf{M} \equiv \left(\mathbf{I}_{mp}\right)_{\mathcal{I}}$, $\mathbf{b} = \begin{bmatrix} \mathbf{b}_{\text{row sums}} \\ \mathbf{b}_{\text{column sums}} \\ \mathbf{b}_{\text{known values}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}_{\text{true}}\mathbf{1} \\ \left(\mathbf{1}^\top \mathbf{X}_{\text{true}}\right)^\top \\ \left(\text{vec}(\mathbf{X}_{\text{true}})\right)_{\mathcal{I}} \end{bmatrix}$, and $\mathcal{I} \sim \mathcal{U}\{1, \ldots, mp\} \wedge |\mathcal{I}| = k$. The Python 3 code, based on `pytmpinv` and two other modules, installable by executing `pip install matplotlib numpy pytmpinv==1.2.0`, for (a) $m = p = 20$ and $k = 0.1 \cdot mp = 40$ and (b) $m = p = 40$ — estimated with the help of $m_{\text{subset}} \times n_{\text{subset}}$-MNBLUE across $\left(\left\lceil \frac{m}{m_{\text{subset}}} \right\rceil \cdot \left\lceil \frac{n}{n_{\text{subset}}} \right\rceil\right) = \left(\left\lceil \frac{40}{20-1} \right\rceil \cdot \left\lceil \frac{40}{20-1} \right\rceil\right) = 9$ reduced models $m_{\text{reduced}} = p_{\text{reduced}} \leq 20$ assuming an estimability constraint of $mp \leq 400$ — and $k = 0.2 \cdot mp = 320$, with a non-iterated ($r = 1$), unique ($\alpha > 0$), and MNBLUE ($\alpha = 1$) (two-step) CLSP solution, is implemented in (a) Listing 1 and (b) Listing 2 (e.g., to be executed in a Jupyter Notebook).

```
import numpy             as     np
from   tmpinv            import tmpinv
import matplotlib.pyplot as     plt

seed   = 123456789
rng    = np.random.default_rng(seed)

# sample (dataset)
m, p   = 20, 20                                          # matrix dimensions (m x p)
X_true = np.abs(rng.normal(size=(m, p)))                 # symmetric non-negative X_true
X_true = (X_true + X_true.T) / 2.0
idx    = rng.choice(m * p, size=max(1, int(0.1 * (m * p))),  # random numbers from
                    replace=False)                       # [0,(m * p) - 1], 10% of total

# model
M      = np.eye(m * p)[idx,:]                            # unit matrix
b_row  = X_true.sum(axis=1)                              # row sums (length m)
b_col  = X_true.sum(axis=0)                              # column sums (length p)
b_val  = X_true.reshape(-1, 1)[idx]                      # column vector
bounds = (0, None)                                       # non-negativity
result = tmpinv(
             M=M, b_row=b_row, b_col=b_col, b_val=b_val,
             m=m, p=p, bounds=bounds, symmetric=True,
             r=1,                                        # a solution without refinement
             alpha=1.0                                   # a unique MNBLUE estimator
         )

# helpers
def fmt(v):
    v = float(v)
    return f"{v:.4e}" if abs(v) >= 1e6 or (v != 0 and abs(v) < 1e-4) else f"{v:.6f}"

# results
print("true X:")
print(np.round(np.asarray(X_true), 4))
print("X_hat:")
print(np.round(result.x,           4))

print("\nDegrees of freedom:")
```

```
dof    = result.model.A.shape[1] - np.linalg.matrix_rank(result.model.A)
p_null = round(dof / result.model.A.shape[1] * 100, 2)
print("␣␣nullity␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣:", dof              )
print("␣␣percentage␣of␣total␣unknowns:", str(p_null) + "%")

print("\nNumerical␣stability:")
print("␣␣kappaC␣:", fmt(result.model.kappaC))
print("␣␣kappaB␣:", fmt(result.model.kappaB))
print("␣␣kappaA␣:", fmt(result.model.kappaA))
corr = result.model.corr()
plt.figure(figsize=(8, 4))
plt.grid(True, linestyle="--", alpha=0.6)
plt.bar(range(len(corr["rmsa_i"])), corr["rmsa_i"])
plt.xlabel("Constraint␣index")
plt.ylabel("dRMSA␣(row␣deletion␣effect)")
plt.title(f"CLSP␣Correlogram␣(Total␣RMSA␣=␣{result.model.rmsa:.2f})")
plt.tight_layout()
plt.show()

print("\nGoodness-of-fit:")
x_true = X_true.flatten()
x_hat  = result.x.flatten()
ss_res = np.sum((x_true - x_hat) ** 2)
ss_tot = np.sum((x_true - np.mean(x_true)) ** 2)
print("␣␣R2_user_defined␣␣␣␣␣␣:", fmt(1 - ss_res / ss_tot))
print("␣␣NRMSE␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣:", fmt(result.model.nrmse))
print("␣␣Diagnostic␣band␣(min):", fmt(np.min(result.model.x_lower)))
print("␣␣Diagnostic␣band␣(max):", fmt(np.max(result.model.x_upper)))
print("␣␣Monte␣Carlo␣t-test:")
for kw, val in result.model.ttest(sample_size=30,                    # NRMSE sample
                                  seed=seed, distribution="normal",  # seed and distribution
                ).items():
    print(f"␣␣␣␣{kw}:␣{float(val):.6f}")
```

Listing 1. Simulated numerical example for a symmetric input-output-table-based AP (TM) problem.

In case (a) (see Listing 1 for code), the number of model (target) variables $\#\widehat{\mathbf{x}}^*_M$ is $mp = 400$ with a nullity of $\dim(\ker(\mathbf{A}^{(1)})) = n - \operatorname{rank}(\mathbf{A}^{(1)}) = 321$, corresponding to 80.25% of the total unknowns. Given the simulation of $\mathbf{X}_{\text{true}} \in \mathbb{R}^{20\times 20}$, a matrix unknown in real-world applications, — i.e., CLSP is used to estimate the elements of an existing matrix from its row sums, $\mathbf{X}_{\text{true}}\mathbf{1} \in \mathbb{R}^{20}$, column sums, $\left(\mathbf{1}^\top \mathbf{X}_{\text{true}}\right)^\top \in \mathbb{R}^{20}$, and a randomly selected 10% of its entries, $(\operatorname{vec}(\mathbf{X}_{\text{true}}))_{\mathcal{I}} \in \mathbb{R}^{40}$, — the model's goodness of fit can be measured by a user-defined $R^2 \approx \max\left(0,\ 1 - \frac{\left\|\operatorname{vec}(\mathbf{X}_{\text{true}}) - \widehat{\mathbf{x}}^*_M\right\|_2^2}{\left\|\operatorname{vec}(\mathbf{X}_{\text{true}}) - \mathbf{1}\frac{1}{400}\sum_{i=1}^{400}(\operatorname{vec}(\mathbf{X}_{\text{true}})_i)\right\|_2^2}\right) \approx 0.278256$ (i.e., CLSP achieves an improvement of $\Delta R^2 = 0.178256$ over a hypothetical naïve predictor reproducing the known 10% entries of $\mathbf{X}_{\text{true},\mathcal{I}}$ and yielding a $R^2 \approx 0.1$, but still a modest value of R^2, i.e. a relatively large error, NRMSE = 12.649111) with $\widehat{\mathbf{x}}^*_M$ lying within wide condition-weighted diagnostic intervals $\widehat{\mathbf{x}}^*_M \in \left[\min(\widehat{\mathbf{x}}^*_{M,\text{lower}}),\ \max(\widehat{\mathbf{x}}^*_{M,\text{upper}})\right] = \left[-8.0847 \times 10^{15},\ 8.0847 \times 10^{15}\right]$ reflecting the ill-conditioning of the strongly rectangular $\mathbf{A}^{(1)} \in \mathbb{R}^{80\times 400}$, $\kappa(\mathbf{A}^{(1)}) = 4.9968 \times 10^{14}$, and only the left-sided Monte Carlo-based t-test for the mean of the NRMSE (on a sample of 30 NRMSE values obtained by substituting $\mathbf{b}$ with $\left(\mathbf{b}^{(\tau)}\right)_i \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$) suggesting consistency with expectations (i.e., with the p-value ≈ 1). In terms of numerical stability, $\kappa(\mathbf{C}_{\text{canon}}) = 4.8023 \times 10^{14} \approx \kappa(\mathbf{A}^{(1)})$ with a low $\text{RMSA}_{i\in\{1,\ldots,40\}} = \text{RMSA} = 0.0400$, as confirmed by the correlogram produced in `matplotlib`, indicating a well-chosen constraints matrix, even given the underdeterminedness of the model (which also prevents a better fit).

Peer-reviewed version available at Mathematics 2025, 13, 3476; doi:https://doi.org/10.3390/math13213476

```
import numpy  as     np
from   tmpinv import tmpinv

seed   = 123456789
rng    = np.random.default_rng(seed)

# sample (dataset)
m, p   = 40, 40                                          # matrix dimensions (m x p)
X_true = np.abs(rng.normal(size=(m, p)))                 # non-negative X_true
X_true = X_true * (1 - np.eye(m, p))                     # zero diagonal
idx    = rng.choice(m * p, size=max(1, int(0.2 * (m * p))),  # random numbers from
                    replace=False)                       # [0,(m * p) - 1], 20% of total

# model
M      = np.eye(m * p)[idx,:]                            # unit matrix
b_row  = X_true.sum(axis=1)                              # row sums (length m)
b_col  = X_true.sum(axis=0)                              # column sums (length p)
b_val  = X_true.reshape(-1, 1)[idx]                      # column vector
bounds = (0, None)                                       # non-negativity
result = tmpinv(
             M=M, b_row=b_row, b_col=b_col, b_val=b_val,
             m=m, p=p, zero_diagonal=True, reduced=(20,20),  # reduced models of (20, 20)
             bounds=bounds,
             r=1,                                        # a solution without refinement
             alpha=1.0                                   # a unique MNBLUE estimator
         )

# helpers
def fmt(v):
    v = float(v)
    return f"{v:.4e}" if abs(v) >= 1e6 or (v != 0 and abs(v) < 1e-4) else f"{v:.6f}"

# results
print("true␣X:")
print(np.round(np.asarray(X_true), 4))
print("X_hat:")
print(np.round(result.x,           4))

print("\nDegrees␣of␣freedom:")
dof    = np.array([CLSP.A.shape[1] - np.linalg.matrix_rank(CLSP.A)
                   for CLSP in result.model])
p_null = np.array([round(dof[i] / result.model[i].A.shape[1] * 100, 2)
                   for i in range(len(dof))])
print("␣␣nullity␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣:", np.min(dof).item(),               "-",
                                        np.max(dof).item()              )
print("␣␣percentage␣of␣total␣unknowns:", str(np.min(p_null).item()) + "%", "-",
                                        str(np.max(p_null).item()) + "%")

print("\nNumerical␣stability␣(min-max␣across␣models):")
kappaC = np.array([CLSP.kappaC for CLSP in result.model])
kappaB = np.array([CLSP.kappaB for CLSP in result.model])
kappaA = np.array([CLSP.kappaA for CLSP in result.model])
print("␣␣kappaC␣:", fmt(np.min(kappaC)), "-", fmt(np.max(kappaC)))
print("␣␣kappaB␣:", fmt(np.min(kappaB)), "-", fmt(np.max(kappaB)))
print("␣␣kappaA␣:", fmt(np.min(kappaA)), "-", fmt(np.max(kappaA)))

print("\nGoodness-of-fit␣(min-max␣across␣models):")
x_true = X_true.flatten()
x_hat  = result.x.flatten()
mask   = np.isfinite(x_true) & np.isfinite(x_hat)
if mask.any():
    ss_res = np.sum((x_true[mask] - x_hat[mask])**2)
    ss_tot = np.sum((x_true[mask] - np.mean(x_true[mask]))**2)
```

Peer-reviewed version available at Mathematics 2025, 13, 3476; doi:https://doi.org/10.3390/math13213476

```
nrmse      = np.array(      [CLSP.nrmse                          for CLSP in result.model])
x_lower    = np.concatenate([np.array(CLSP.x_lower).reshape(-1) for CLSP in result.model])
x_upper    = np.concatenate([np.array(CLSP.x_upper).reshape(-1) for CLSP in result.model])
print("␣␣R2_user_defined␣␣␣␣␣␣:", fmt(1 - ss_res / ss_tot                                ))
print("␣␣NRMSE␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣:", fmt(np.min(nrmse)),      "-", fmt(np.max(nrmse)     ))
print("␣␣Diagnostic␣band␣(min):", fmt(np.min(x_lower)),    "-", fmt(np.max(x_lower)   ))
print("␣␣Diagnostic␣band␣(max):", fmt(np.min(x_upper)),    "-", fmt(np.max(x_upper)   ))
print("␣␣Monte␣Carlo␣t-test:")
ttests     = [CLSP.ttest(sample_size=30, seed=seed,
                         distribution="normal")                for CLSP in result.model]
keys       = ttests[0].keys()
for kw in keys:
    val    = np.array([t[kw] for t in ttests], dtype=float)
    print(f"␣␣␣␣{kw}:␣{fmt(np.min(val))}␣-␣{fmt(np.max(val))}")
```

Listing 2. Simulated numerical example for a (non-negative) trade-matrix-based AP (TM) problem.

In case (b) (see Listing 2 for code), the corresponding number of model (target) variables in each of the $\left(\left\lceil\frac{40}{19}\right\rceil \cdot \left\lceil\frac{40}{19}\right\rceil\right) = 9$ reduced design submatrices $\mathbf{A}^{(1)}_{\text{subset}} \subset \mathbf{A}^{(1)}_{\text{reduced}}$, $\#\widehat{\mathbf{x}}^{*}_{M,\,\text{reduced}}$, is $m_{\text{subset}} p_{\text{subset}} \leq 361$ — where $m_{\text{subset}} \leq 20 - 1 = 19$ and $p_{\text{subset}} \leq 20 - 1 = 19$, one row and one column being reserved for $\sum_{j \notin \mathcal{J}} a^{(1)}_{i,j}$ and $\sum_{i \notin \mathcal{I}} a^{(1)}_{i,j}$, which enter $\mathbf{A}^{(1)}_{\text{reduced}}$ as $\mathbf{S} = \begin{bmatrix} \mathbf{I}_{m - m_{\text{subset}}} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{p - p_{\text{subset}}} \end{bmatrix}$ and the vector of slack variables $\left(\widehat{\mathbf{y}}^{*}_{\text{reduced}}\right)^{*} = \begin{bmatrix} \widehat{\mathbf{y}}^{*}_{\text{reduced}} \\ \widehat{\mathbf{y}}^{*}_{\text{residual, reduced}} \end{bmatrix} \subset \widehat{\mathbf{z}}^{*}_{\text{reduced}}$, so that $\mathbf{C}_{\text{reduced}} \mathbf{x}_{\text{reduced}} + \mathbf{S}_{\text{reduced}} \mathbf{y}^{*}_{\text{reduced}} = \mathbf{b}_{\text{r.},\,1:k}$ (i.e., the slack matrix compensates for the unaccounted row and column sums in the reduced models as opposed to the full one, such as case (a)) — with a nullity $\dim(\ker(\mathbf{A}^{(1)}_{\text{reduced}})) = n_{\text{reduced}} - \text{rank}(\mathbf{A}^{(1)}_{\text{reduced}}) \in [2,\ 290]$ (per reduced model), corresponding to 25.00%–72.68% of the total unknowns (per reduced model) — to compare, a full model, under the same inputs and no computational constraints (i.e., $mp \nleq m_E p_E$), would have a nullity $\dim(\ker(\mathbf{A}^{(1)})) = n - \text{rank}(\mathbf{A}^{(1)}) = 1168$, corresponding to 73.00% of the total unknowns. In the examined example, — based on $\mathbf{X}_{\text{true}} \mathbf{1} \in \mathbb{R}^{40}$, $\left(\mathbf{1}^{\top} \mathbf{X}_{\text{true}}\right)^{\top} \in \mathbb{R}^{40}$, and a randomly selected 20% of entries of the true matrix $\mathbf{X}_{\text{true}} \in \mathbb{R}^{40 \times 40}$, — the reduced-model block solution's goodness of fit could not be efficiently measured by a user-defined $R^2 \approx \max\left(0,\ 1 - \frac{\left\|\text{vec}(\mathbf{X}_{\text{true}}) - \text{vec}\left(\left\{\widehat{\mathbf{x}}^{*,\,(i_E, j_E)}_{M,\,\text{reduced},\,1:m_{\text{subset}},\,1:n_{\text{subset}}}\right\}\right)\right\|_2^2}{\left\|\text{vec}(\mathbf{X}_{\text{true}}) - \mathbf{1} \frac{1}{1600} \sum_{i=1}^{1600} (\text{vec}(\mathbf{X}_{\text{true}})_i)\right\|_2^2}\right) \approx 0$ (i.e., the block matrix constructed from $\left(\left\lceil\frac{40}{19}\right\rceil \cdot \left\lceil\frac{40}{19}\right\rceil\right) = 9$ reduced-model estimates led to $R^2 \approx \max(0,\ -5.435377)$, but to an error proportionate to the one in case (a), $\text{NRMSE}_{\text{reduced}} \in [3.464102,\ 15.748016]$ (per reduced model)) — in contrast, a full model would achieve $R^2 \approx \max\left(0,\ 1 - \frac{\left\|\text{vec}(\mathbf{X}_{\text{true}}) - \widehat{\mathbf{x}}^{*}_{M}\right\|_2^2}{\left\|\text{vec}(\mathbf{X}_{\text{true}}) - \mathbf{1} \frac{1}{1600} \sum_{i=1}^{1600} (\text{vec}(\mathbf{X}_{\text{true}})_i)\right\|_2^2}\right) \approx 0.278427$, but at a cost of a greater error, $\text{NRMSE} = 29.427878$ — with $\widehat{\mathbf{x}}^{*}_{M,\,\text{reduced}}$ lying within strongly varying condition-weighted diagnostic intervals $\widehat{\mathbf{x}}^{*}_{M,\,\text{reduced}} \in \left[\min(\widehat{\mathbf{x}}^{*}_{M,\,\text{lower, reduced}}),\ \max(\widehat{\mathbf{x}}^{*}_{M,\,\text{upper, reduced}})\right]$, where $\min(\widehat{\mathbf{x}}^{*}_{M,\,\text{lower, reduced}}) \in [-330.638753,\ -2.5009 \times 10^{-19}]$ (per reduced model) and $\max(\widehat{\mathbf{x}}^{*}_{M,\,\text{upper, reduced}}) \in [-2.5456 \times 10^{-19},\ 336.547225]$ (per reduced model), and varying results of Monte Carlo-based t-tests for the mean of the $\text{NRMSE}_{\text{reduced}}$ (on a sample of 30 $\text{NRMSE}_{\text{reduced}}$ values obtained by substituting $\mathbf{b}_{\text{r.}}$ with $\left(\mathbf{b}^{(\tau)}_{\text{r.}}\right)_i \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$), where p-values range 4.8022×10^{-09}–1.000000 (left-sided), 0.000000–1.000000 (two-sided), and 3.7021×10^{-20}–1.000000 (right-sided) (per reduced model) — alternatively, a full model would lead to wider condition-weighted diagnostic intervals $\widehat{\mathbf{x}}^{*}_{M} \in \left[\min(\widehat{\mathbf{x}}^{*}_{M,\,\text{lower}}),\ \max(\widehat{\mathbf{x}}^{*}_{M,\,\text{upper}})\right] = \left[-1.5357 \times 10^{16},\ 1.5357 \times 10^{16}\right]$ (i.e., reflecting the ill-conditioning of the strongly rectangular $\mathbf{A}^{(1)} \in \mathbb{R}^{400 \times 1600}$, $\kappa(\mathbf{A}^{(1)}) = 2.2196 \times 10^{14}$) and only the left-sided Monte Carlo-based t-test for the mean of the NRMSE (on a sample of 30 NRMSE values obtained by substituting $\mathbf{b}$ with $\left(\mathbf{b}^{(\tau)}\right)_i \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$) suggesting consistency with expectations (i.e., with the p-value ≈ 1). In terms of numerical stability, $\kappa(\mathbf{C}_{\text{canon, reduced}}) \in [2.236068,\ 6.244998] < \kappa(\mathbf{A}^{(1)}_{\text{reduced}}) \in [4.509742,\ 9.325735]$ (per reduced model), which indicates well-conditioning of all the reduced models

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

— conversely, in a full model, $\kappa(\mathbf{C}_{\text{canon}}) = 1.0907 \times 10^{14} < \kappa(\mathbf{A}^{(1)})$ (therefore, a full model ensures an overall better fit but a lower fit quality, i.e., a trade-off).

As the **CMLS (RP) numerical example**, this text addresses a problem similar to the one solved in Bolotov [15,82]: a coefficient vector $\boldsymbol{\beta}$ from a (time-series) linear regression model in its classical (statistical) notation $y_t = \mathbf{x}_t\boldsymbol{\beta} + \epsilon_t$, s.t. $\sum_{j=1}^{i+1} \beta_j = c \wedge \forall_{t\in\mathbf{P}}\ \Delta y_t = 0 \wedge \forall_{t\in\mathbf{P}}\ \Delta^2 y_t \gtreqless 0$ with $\Delta^n y_t$ denoting n-th order differences (i.e., the discrete analogue of $\frac{d^n y_t}{dt^n}$) — where y_t is the dependent variable, $\mathbf{x}_t = [1, x_{t,1}, \ldots, x_{t,i}]$ is the vector of regressors with a constant, ϵ_t is the model's error (residual), c is a constant, and $\mathbf{P} \subsetneqq \{t_j : j = 1, \ldots, k\}$ is the set of stationary points — is estimated on a simulated sample (dataset) with the help of CLSP assuming $\mathbf{x}_M \equiv \boldsymbol{\beta} \in \mathbb{R}^p$ and $\mathbf{D} \equiv \mathbf{X} = [\mathbf{1}, \mathbf{x}_1, \ldots, \mathbf{x}_i] \in \mathbb{R}^{k\times p}$, $p = i + 1$. The Python 3 code, based on `numpy` and `pyclsp` modules, installable by executing `pip install numpy pyclsp==1.6.0`, for $k = 500$, $p = 6$, $c = 1$, $\mathbf{y} = \mathbf{D}\boldsymbol{\beta}_{\text{true}} + \boldsymbol{\epsilon}$, where $\forall_{j\in\{2,\ldots,6\}}\ \mathbf{D}_{\cdot,j} \equiv \mathcal{X}_j$, $(\mathcal{X}_j)_i \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$, $\mathcal{X}_j \in \mathbb{R}^{500}$, $\boldsymbol{\beta}_{\text{true}} = \frac{\mathcal{X}_\beta}{\mathbf{1}^\top \mathcal{X}_\beta}$, $(\mathcal{X}_\beta)_i \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$, $\mathcal{X}_\beta \in \mathbb{R}^6$, $\boldsymbol{\epsilon} \equiv \mathcal{X}_\epsilon$, $(\mathcal{X}_\epsilon)_i \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$, $\mathcal{X}_\epsilon \in \mathbb{R}^{500}$, and $\mathbf{P} = \{t : \Delta\mathbf{y} = 0\}$, with a non-iterated ($r = 1$), unique ($\alpha > 0$), and MNBLUE ($\alpha = 1$) two-step CLSP solution (consistent with the ones from cases (a) and (b) for APs (TMs)), is implemented in Listing 3 (e.g., for a Jupyter Notebook).

```
import   numpy as     np
from     clsp  import CLSP

seed     = 123456789
rng      = np.random.default_rng(seed)

# sample (dataset)
k        = 500                                          # number of observations in D
p        = 6                                            # number of regressors
c        = 1                                            # sum of coefficients
D        = np.empty((k, p))
D[:, 0 ] = 1.0                                          # constant
D[:, 1:] = rng.normal(size=(k, p - 1))                  # D.,j ~ N(0,1), 2 <= j <= p
b_true   = rng.normal(size=p)                           # b_true ~ N(0,1)
b_true   = (b_true / b_true.sum()) * c
e        = rng.normal(size=(k, 1))                      # e ~ N(0,1)
y        = (D @ b_true).reshape(-1, 1) + e              # y_t = D @ b_true + e_t

# model
b        = np.vstack([
               np.asarray([c]),                         # c (the sum of coefficients)
               np.zeros((k - 2, 1)),                    # zeros
               np.zeros((k - 1, 1)),                    # zeros
               y                                        # values of y_t
           ])
C        = np.vstack([
               np.ones((1, p)),                         # a row of ones
               np.diff(D, n=2, axis=0),                 # the 2nd differences
               np.diff(D, n=1, axis=0)                  # the 1st differences
           ])
S        = np.block([
               [np.zeros((  1, k-2))],                  # a zero vector
               [np.diag(-np.sign(np.diff(y.ravel(), n=2)))], # a diagonal sign matrix
               [np.zeros((k-1, k-2))],                  # a zero matrix
           ])
model    = CLSP().solve(
               problem="cmls", b=b, C=C, S=S, M=D,
               r=1,                                     # a solution without refinement
               alpha=1.0                                # a unique MNBLUE estimator
           )

# results
print("true␣beta␣(x_M):")
```

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

```
print(np.round(np.asarray(b_true).flatten(), 4))

print("beta␣hat␣(x_M␣hat):")
print(np.round(model.x.flatten(), 4))

print("\nNumerical␣stability:")
print("␣␣kappaC␣:", round(model.kappaC, 4))
print("␣␣kappaB␣:", round(model.kappaB, 4))
print("␣␣kappaA␣:", round(model.kappaA, 4))

print("\nGoodness-of-fit:")
print("␣␣R2_partial␣␣␣␣␣␣␣␣␣␣:", round(model.r2_partial,    6))
print("␣␣NRMSE_partial␣␣␣␣␣␣␣␣:", round(model.nrmse_partial, 6))
print("␣␣Diagnostic␣band␣(min):", np.round(np.min(model.x_lower), 4))
print("␣␣Diagnostic␣band␣(max):", np.round(np.max(model.x_upper), 4))
print("␣␣Monte␣Carlo␣t-test:")
for kw, val in model.ttest(sample_size=30,                    # NRMSE_partial sample
                           seed=seed, distribution="normal", # seed and distribution
                           partial=True).items():
    print(f"␣␣␣␣{kw}:␣{float(val):.6f}")
```

Listing 3. Simulated numerical example for a (time-series) stationary-points-based CMLS (RP) problem.

Compared to the true values $\boldsymbol{\beta}_{\text{true}} = [0.1687,\ -0.3593,\ 0.5969,\ -0.0628,\ -0.1742,\ 0.8307]$, the CMLS (RP) estimate is $\widehat{\boldsymbol{\beta}} = \widehat{\mathbf{x}}^*_M = [0.2024,\ -0.1121,\ 0.2095,\ -0.0396,\ -0.0296,\ 0.2611]$ with a modest $R^2_{\text{partial}} = 0.293328$ (i.e., a greater error, $\text{NRMSE}_{\text{partial}} = 0.840638$), moderate condition-weighted diagnostic intervals $\widehat{\mathbf{x}}^*_M \in [-29.7780,\ 30.3003]$, and only the right-sided Monte Carlo-based t-test for the mean of the $\text{NRMSE}_{\text{partial}}$ (on a sample of 30 $\text{NRMSE}_{\text{partial}}$ values obtained by substituting $\mathbf{b}$ with $\left(\mathbf{b}^{(\tau)}\right)_i \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$) — in the example under scrutiny, $\text{NRMSE}_{\text{partial}}$ is preferable to NRMSE due to $\widehat{\mathbf{y}}^* \gg \widehat{\mathbf{x}}^*$ — suggesting consistency with expectations (i.e., with the p-value ≈ 1). Similarly, in terms of numerical stability, $\kappa(\mathbf{C}_{\text{canon}}) = 138.6109 > \kappa(\mathbf{A}^{(1)}) = 127.8287$, indicating that the constraint block is ill-conditioned, most likely, because of the imposed data- (rather than theory-) based definition of stationary points, which rendered Step 2 infeasible ($\widehat{\mathbf{x}}^*_M = \widehat{\mathbf{x}}^{(1)}_M$) (limiting the fit quality).

Finally, in the case of a **LPRLS/QPRLS numerical example**, this text simulates potentially ill-posed LP (QP) problems, similar to the ones addressed in Whiteside et al. [21] and Blair [22,23]: a solution $\widehat{\mathbf{x}}^* \gtrless 0$ in its classical LP (QP) notation $\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathbb{R}^p} \mathbf{c}^\top\mathbf{x}$ ($\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^p}\left\{\frac{1}{2}\mathbf{x}^\top\mathbf{Q}\mathbf{x} + \mathbf{c}^\top\mathbf{x}\right\}$), s.t. $\mathbf{A}\mathbf{x} \leq \mathbf{b} \equiv \mathbf{A}_{ub}\mathbf{x} \leq \mathbf{b}_{ub} \wedge \mathbf{A}_{eq}\mathbf{x} = \mathbf{b}_{eq}$ — where $\mathbf{Q} \in \mathbb{R}^{p\times p}$ is a symmetric positive definite matrix, $\mathbf{c} \in \mathbb{R}^p$, $\mathbf{A} = \begin{bmatrix}\mathbf{A}_{ub}\\ \mathbf{A}_{eq}\end{bmatrix} \in \mathbb{R}^{(m_{ub}+m_{eq})\times p}$, and $\mathbf{b} = \begin{bmatrix}\mathbf{b}_{ub}\\ \mathbf{b}_{eq}\end{bmatrix} \in \mathbb{R}^{m_{ub}+m_{eq}}$ — is estimated from two randomly generated (coefficient) matrices $\mathbf{A}_{ub} \equiv \mathcal{X}_1$, $(\mathcal{X}_1)_{i,j} \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$, and $\mathbf{A}_{eq} \equiv \mathcal{X}_2$, $(\mathcal{X}_2)_{i,j} \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$, and two (right-hand-side) vectors $\mathbf{b}_{ub} \equiv \mathcal{X}_3$, $(\mathcal{X}_3)_i \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$, and $\mathbf{b}_{eq} \equiv \mathcal{X}_4$, $(\mathcal{X}_4)_i \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$, via CLSP assuming $\mathbf{C} = \begin{bmatrix}\mathbf{A}_{ub}\\ \mathbf{A}_{eq}\end{bmatrix}$, $\mathbf{S} = \mathbf{Q}^L_S\begin{bmatrix}\mathbf{S}_s \subseteq \mathbf{I}_{m_{ub}}\\ \mathbf{0}\end{bmatrix}\mathbf{Q}^R_S$, and $\mathbf{b} = \begin{bmatrix}\mathbf{b}_{ub}\\ \mathbf{b}_{eq}\end{bmatrix}$, where $\mathbf{Q}^L_S$ and $\mathbf{Q}^R_S$ are permutation matrices, while omitting $\mathbf{Q}$ and $\mathbf{c}$. The Python 3 code, based on `numpy` and `pylppinv` modules, installable by executing `pip install numpy pylppinv==1.3.0`, for $m_{ub} = 50$, $m_{eq} = 25$, and $p = 500$, with a non-iterated ($r = 1$), unique ($\alpha > 0$), and MNBLUE ($\alpha = 1$) (two-step) CLSP solution (analogously consistent with the ones from cases (a) and (b) for APs (TMs) and the one from CMLS (RPs)), is implemented in Listing 4 (e.g., for a Jupyter Notebook).

```
import numpy  as     np
from   lppinv import lppinv

seed   = 123456789
rng    = np.random.default_rng(seed)

# sample (dataset)
A_ub   = rng.normal(size=(50, 500))                    # underdetermined LP/QP matrix
A_eq   = rng.normal(size=(25, 500))                    # underdetermined LP/QP matrix
```

Peer-reviewed version available at Mathematics 2025, 13, 3476; doi:https://doi.org/10.3390/math13213476

```
b_ub   = rng.normal(size=(50,   1))                          # may be inconsistent with A_ub
b_eq   = rng.normal(size=(25,   1))                          # may be inconsistent with A_eq
model  = lppinv(
             A_ub=A_ub, A_eq=A_eq, b_ub=b_ub, b_eq=b_eq,
             non_negative=False,                             # allow negative values
             r=1,                                            # a solution without refinement
             alpha=1.0                                       # a unique MNBLUE estimator
         )

# results
print("x␣hat␣(x_M␣hat):")
print(np.round(model.x.flatten(), 4))

print("\nDegrees␣of␣freedom:")
dof    = model.A.shape[1] - np.linalg.matrix_rank(model.A)
p_null = round(dof / model.A.shape[1] * 100, 2)
print("␣␣nullity␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣:", dof              )
print("␣␣percentage␣of␣total␣unknowns:", str(p_null) + "%")

print("\nNumerical␣stability:")
print("␣␣kappaC␣:", round(model.kappaC, 4))
print("␣␣kappaB␣:", round(model.kappaB, 4))
print("␣␣kappaA␣:", round(model.kappaA, 4))

print("\nGoodness-of-fit:")
print("␣␣NRMSE␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣:", round(model.nrmse,              6))
print("␣␣Diagnostic␣band␣(min):", np.round(np.min(model.x_lower), 4))
print("␣␣Diagnostic␣band␣(max):", np.round(np.max(model.x_upper), 4))
print("␣␣Monte␣Carlo␣t-test:")
for kw, val in model.ttest(sample_size=30,                   # NRMSE sample
                           seed=seed, distribution="normal", # seed and distribution
                ).items():
    print(f"␣␣␣␣{kw}:␣{float(val):.6f}")
```

Listing 4. Simulated numerical example for a underdetermined and potentially infeasible LP (QP) problem.

The nullity (i.e., underdeterminedness) of the CLSP design matrix, $\mathbf{A}^{(1)}$, is $\dim(\ker(\mathbf{A}^{(1)})) = n - \operatorname{rank}(\mathbf{A}^{(1)}) = 475$, corresponding to 86.36% of the total unknowns, accompanied by a greater (relatively high) error, NRMSE $= 12.247449$, moderate condition-weighted diagnostic intervals $\widehat{\mathbf{x}}_M^* \in [-1.2919,\ 1.3918]$, and only the right-sided Monte Carlo-based t-test for the mean of the NRMSE (on a sample of 30 NRMSE values obtained by substituting $\mathbf{b}$ with $\left(\mathbf{b}^{(\tau)}\right)_i \overset{\text{iid}}{\sim} \mathcal{N}(0,\ 1)$) suggesting consistency with expectations (i.e., with the p-value ≈ 1). Similarly, in terms of numerical stability, $\kappa(\mathbf{C}_{\text{canon}}) = 2.2168 = \kappa(\mathbf{A}^{(1)}) = 2.2168$, indicating that the constraint block is well-conditioned despite a strongly rectangular $\mathbf{A}^{(1)} \in \mathbb{R}^{75\times 550}$. Hence, CLSP can be efficiently applied to AP (TM), CMLS (RP), and LPRLS/QPRLS special cases using Python 3, and the reader may wish to experiment with the code in Listings 1–4 by relaxing its uniqueness (setting $\alpha = 0$) or the MNBLUE characteristic (setting $\alpha \in [0,\ 1)$).

To sum up, while the simulated numerical examples for the APs (TMs), CMLS (RPs), and LPRLS/QPRLS problems successfully *illustrate the theoretical assumptions and numerical behavior of the CLSP estimator* (as described in Sections 3–5 of this work), they remain limited by simplification and a priori construction. First, each estimation was performed under a fixed $r = 1$ (i.e., without iterative refinement) and a unique MNBLUE configuration ($\alpha = 1$), thus omitting the calibration of these parameters. Second, in the AP (TM) case with 10% known values and a non-zero diagonal, similar to the CMLS (RP) one, the estimator achieved a modest $R^2 \approx 0.3$, consistent with the degree of underdeterminedness, while the reduced models (e.g., zero-diagonal, block-decomposed) performed substantially worse due to aggregation and blockwise estimation (backing the recommendation to prefer full models over reduced ones). Finally, the estimator, given its formalization, is *computationally demanding*, with the SVD-based Step 1 complexity of $O(\max(m,n)\min(m,n)^2)$ for $\mathbf{A}^{(r)} \in \mathbb{R}^{m\times n}$ in each iteration (in

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

CLSP-based software built on Python's `SciPy`, R, and Stata) and convex-programming-based Step 2 complexity of $O(k\,nnz(\mathbf{A}^{(r)}))$ or $O(n^3)$, where k is the number of solver iterations and $nnz(\cdot)$ is the number of non-zero elements in $\mathbf{A}^{(r)} \in \mathbb{R}^{m\times n}$, (in CLSP-based software built on Python's `CVXPY` and R's `CVXR`), not including eventual repeated estimations for RMSA_i (in which $\mathbf{A}^{(r)}$ is reduced by one row) and Monte Carlo-based t-test — to exemplify, a 20×20 AP (TM) problem with 10% known values and a non-zero diagonal has an $\mathbf{A}^{(r)} \in \mathbb{R}^{80\times 400}$ requiring r-times 2.56 million floating-point operations in Step 1 and up to 320 million under the SCS solver in Step 2, both eventually repeated twenty times to calculate RMSA_i and at least thirty times (per CLT) for Monte Carlo-based t-test. Despite its complexity, the CLSP framework remains practically viable and is ready for parameter calibration with subsequent application to real-world problems.

8. Discussion and Conclusions

The proposed modular two-step Convex Least Squares Programming (CLSP) estimator represents an attempt to unify historical efforts developed mainly in the 1960s–1990s to address ill-posed problems in both LS and LP (QP) — namely, the mentioned Wolfe [25], Stoer [75], Waterman [76], Dax [26], Osborne [27], George and Osborne [74], Barnes et al. [79], Donoho and Tanner [73], Grafarend and Awange [77], Bei et al. [80], and Qian et al. [78] — by correcting (1) the estimation, based on a Moore-Penrose pseudoinverse $\mathbf{A}^{\dagger}$ (which has been applied in an ad hoc manner in the field, particularly in econometrics in the 2010s; consult Pereira-López et al. [81] and Bolotov [16]) and a Bott-Duffin inverse $\mathbf{A}^{\dagger}_{\mathbf{Z}}$ (based on the literature research, this text may be the first to consider a constrained generalized inverse to generalize an $\mathbf{A}^{\dagger}$-based estimator) with the help of a (2) regularization-inspired correction ([4], pp. 477–479), achieving greater fit quality in terms of ensuring strict satisfaction of problem constraints (previously infeasible under an $\mathbf{A}^{\dagger}$-based solution), while yielding a (potentially) unique and MNBLUE solution (by setting $\alpha = 1$). At the same time, CLSP is well grounded in the algorithmic logic of the works in question, such as Wolfe [25] and Dax [26], as further formalized by, e.g., Osborne [27]. However, the primary strength of the framework in question lies in its "generality": in contrast to the (individual) problem-specific algorithms, such as the ones developed by Übi [28,29,30,31,32] — routines combining (NN)LS, LP, and QP within a single framework to solve both well- and ill-posed convex optimization problems, — CLSP provides a unified estimator with a proper methodology for numerical stability and goodness-of-fit assessment, centered around RMSA and NRMSE. A secondary strength is its practical "versatility": through at least three classes of special cases — APs (TMs), CMLS (RPs), and LPRLS/QPRLS, — it allows estimation of (rectangular) input-output tables (national, inter-country, or satellite), structural matrices (such as trade, country-product, investment, or migration data), and previously infeasible textbook-definition models of economic variables and/or LP (QP) problems. Furthermore, the inclusion of $\mathbf{A}^{\dagger}_{\mathbf{Z}}$ allows probabilistic and Bayesian applications, since its subspace restrictions can be interpreted in terms of random variables, prior distributions, as well as Monte Carlo-based inference. To conclude, CLSP has already been fully implemented in Python 3 (`pyclsp`, `pytmpinv`, and `pylppinv`) and is to be ported into R and Stata in the future, making it a readily available alternative to purely theoretical or not freely accessible algorithms or proprietary packages, and applicable in a real-world setting (such as econometric research, replacing [15,16,81] with the two-step corrected estimation).

Author Contributions: The manuscript was prepared by a sole author.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The numerical examples used in the manuscript are based on simulated data. The Python 3.10 and Stata 14 code for replication purposes is provided in Listings 1–4 and B.1.

Peer-reviewed version available at Mathematics 2025, 13, 3476; doi:https://doi.org/10.3390/math13213476

Acknowledgments: All arguments, structure, and conclusions in this manuscript are the author's own. During the preparation of the manuscript, the author recurred to ChatGPT-5 (OpenAI) solely to improve clarity and language, including refinement of style, formal precision in proofs of theorems, and assistance with interpretation of data and results. No content was adopted without the author's review and editing, and the author claims full responsibility for the final content of the manuscript.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript (in order of their appearance in the sections):

CLSP	Convex Least Squares Programming
LS	Least Squares
OLS	Ordinary Least Squares
NNLS	Non-Negative Least Squares
SVD	Singular Value Decomposition
LP	Linear Programming
QP	Quadratic Programming
Lasso	Least Absolute Shrinkage and Selection Operator
Ridge	Ridge Regression (Tikhonov regularization)
MNBLUE	Minimum-Norm Best Linear Unbiased Estimator
BLUE	Best Linear Unbiased Estimator
RMSA	Root Mean Square Alignment
RMSE	Root Mean Square Error
NRMSE	Normalized Root Mean Square Error
ANOVA	Analysis of Variance
CLT	Lindeberg-Lévy Central Limit Theorem
APs	Allocation Problems
TMs	Tabular Matrix Problems
UNCTAD	UN Trade and Development
CMLS	Constrained-Model Least Squares
RPs	Regression Problems
NBER	U.S. National Bureau of Economic Research
LPRLS	Linear Programming via Regularized Least Squares
QPRLS	Quadratic Programming via Regularized Least Squares
iid	independent and identically distributed

Appendix A

This appendix presents (a) an equation of the Bott-Duffin inverse, $(\mathbf{A}^{(r)}_{\mathbf{Z}})^{\dagger}$, based on the singular value decomposition (SVD), as well as (b) a partitioned decomposition of $\mathbf{B}^{(r)}$ (consult Equations (A1)–(A3) and, for the history of all three concepts, Sections 2 and 4).

Appendix A.1

Let $\mathbf{A}^{(r)} \in \mathbb{R}^{m\times n}$ be a real matrix, and let $\mathbf{Z} \in \mathbb{R}^{n\times n}$ be a symmetric idempotent matrix (i.e., $\mathbf{Z}^2 = \mathbf{Z} = \mathbf{Z}^{\top}$) defining a subspace restriction. Then, the Bott-Duffin inverse $(\mathbf{A}^{(r)}_{\mathbf{Z}})^{\dagger} \in \mathbb{R}^{n\times m}$ is given by the formula $(\mathbf{A}^{(r)}_{\mathbf{Z}})^{\dagger} = \left(\mathbf{Z}(\mathbf{A}^{(r)})^{\top}\mathbf{A}^{(r)}\mathbf{Z}\right)^{\dagger}\mathbf{Z}(\mathbf{A}^{(r)})^{\top}$, as defined in Equation (5), where the projected Gram matrix $\mathbf{Z}(\mathbf{A}^{(r)})^{\top}\mathbf{A}^{(r)}\mathbf{Z}$ is computable via its SVD:

$$\mathbf{Z}(\mathbf{A}^{(r)})^{\top}\mathbf{A}^{(r)}\mathbf{Z} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^{\top} \tag{A1}$$

where $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{n\times n}$ are orthogonal (since the projected Gram matrix is symmetric positive semidefinite, $\mathbf{U} = \mathbf{V}$), and $\mathbf{\Sigma} = \operatorname{diag}(\sigma_1, \ldots, \sigma_{\text{rank}}, 0, \ldots, 0) \in \mathbb{R}^{n\times n}$ contains the singular values. Hence, for $\mathbf{\Sigma}^\dagger = \operatorname{diag}(\sigma_1^{-1}, \ldots, \sigma_{\text{rank}}^{-1}, 0, \ldots, 0)$, $\mathbf{U}\mathbf{\Sigma}^\dagger\mathbf{U}^\top$ substitutes $\left(\mathbf{Z}(\mathbf{A}^{(r)})^\top\mathbf{A}^{(r)}\mathbf{Z}\right)^\dagger$:

$$(\mathbf{A}_{\mathbf{Z}}^{(r)})^\dagger = \left(\mathbf{Z}(\mathbf{A}^{(r)})^\top\mathbf{A}^{(r)}\mathbf{Z}\right)^\dagger\mathbf{Z}(\mathbf{A}^{(r)})^\top = \left(\mathbf{U}\mathbf{\Sigma}^\dagger\mathbf{U}^\top\right)\mathbf{Z}(\mathbf{A}^{(r)})^\top \tag{A2}$$

Appendix A.2

Let $\operatorname{rank}([\mathbf{C}\ \ \mathbf{S}]) \leq \operatorname{rank}(\mathbf{C}) + \operatorname{rank}(\mathbf{S})$ in the analysis of numerical stability of the first estimate $\widehat{\mathbf{z}}^{(r)}$, as defined in Equation (2). Then a decomposition of $\mathbf{B}^{(r)}$ reveals how the interaction of constraints $\mathbf{C}$ and slack structure $\mathbf{S}$ forms the curvature of the solution space:

$$\begin{aligned}\mathbf{B}^{(r)} &= \mathbf{A}^{(r)}\begin{bmatrix}\mathbf{C} & \mathbf{S}\end{bmatrix}^\dagger \\ &= f\left(\begin{bmatrix}\mathbf{C} & \mathbf{S}\end{bmatrix}, g\left(\mathbf{M}, \mathbf{b} - \mathbf{A}^{(r-1)}\widehat{\mathbf{z}}^{(r-1)}\right)\right)\cdot\begin{bmatrix}\mathbf{C}^\top\mathbf{C} & \mathbf{C}^\top\mathbf{S} \\ \mathbf{S}^\top\mathbf{C} & \mathbf{S}^\top\mathbf{S}\end{bmatrix}^\dagger\begin{bmatrix}\mathbf{C}^\top \\ \mathbf{S}^\top\end{bmatrix}\end{aligned} \tag{A3}$$

While the block $\mathbf{C}^\top\mathbf{C}$ captures the contribution of the constraints and $\mathbf{S}^\top\mathbf{S}$ reflects the curvature introduced by slack variables, the off-diagonal blocks $\mathbf{C}^\top\mathbf{S}$ and $\mathbf{S}^\top\mathbf{C}$ encode cross-interactions between constraints and slacks: if these off-diagonal blocks vanish, the effects are orthogonal and additive (in the column space sense since $\mathbf{C}^\top\mathbf{S} = 0$ implies $\mathcal{R}(\mathbf{C}) \perp \mathcal{R}(\mathbf{S})$); otherwise, $\mathbf{C}$ is orthogonalized with respect to $\mathcal{R}(\mathbf{S})$, which is effected by $(\mathbf{I} - \mathbf{S}\mathbf{S}^\dagger)\mathbf{C}$. Hence, the presence of $\mathbf{S}$ in $[\mathbf{C}\ \ \mathbf{S}]$ can improve or worsen the ill-posedness of the canonized problem: namely, $\mathbf{C}^\top\mathbf{S} \neq 0$ tends to worsen the condition number $\kappa([\mathbf{C}\ \ \mathbf{S}])$.

Appendix B

This appendix presents Stata code, i.e., a `.do`-file, for the Monte Carlo experiment, compatible with version 14.0 and above (Stata/MP with at least four CPUs is recommended for replication — specifically, 50,000 iterations require substantial CPU time and memory).

Stata/MP 14.0 dependencies:

TITLE

`'SUMMARIZEBY'`: module to use `statsby` functionality with `summarize`

DESCRIPTION/AUTHOR(S)

This routine combines the `summarize` command with `statsby`. It iteratively collects `summarize` results for each variable with `statsby` and saves the result to memory or to a Stata `.dta` file.

KW: data management
KW: summarize
KW: statsby

Requires: Stata version 8

Distribution–Date: 20221012
Author: Ilya Bolotov, Prague University of Economics and Business
Support: email `ilya.bolotov@vse.cz`

INSTALLATION FILES

`summarizeby.ado`
`summarizeby.sthlp`

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

(type `ssc install summarizeby` to install)

Appendix B.1

```
*! version 1.0.2  07oct2022
version 14.0

**# Bookmark #1
* ==== CHECK FOR MISSING SOFTWARE ========================================== *

**# Bookmark #2
* ==== CONFIGURATION ======================================================== *
clear all
tempfile tmpf DTA_FILE                                  // temporary file
tempname t                                              // temporary frame
glo date : di %tdCYND daily("$S_DATE", "DMY")           // date
glo date = cond("`1'" != "", "`1'", "$date")
glo compiled_by "Ilya Bolotov, MBA, Ph.D."
// load a pre-save (if applicable)
cap use "AP____DATA_pre-save.dta", clear
// create folder and/or open log
cap log using "AP____DATA_$date.smcl", name(data) replace

**# Bookmark #3
* ==== MATA CODE ============================================================ *
sca `t' = clock(c(current_date) + " " + c(current_time), "DMYhms")
// Experiment, QUAD PRECISION
mata: d = (1::50)                                       // dimensions of `a'
mata: n = 50000                                         // iterations

loc RS        real scalar
loc RV        real colvector
loc RM        real matrix
loc SS        string scalar
loc TM        transmorphic matrix
loc PF        pointer(`RM' function) scalar
loc VV        void

loc version   "version 14:"

mata:
mata  set matastrict on

`RS' appinv_dist(`RS' r, `RS' c,`RS' n, |`RS' f_d, `RS' f_n, `RS' tol,
                 `RS' seed, `PF' dist)                          /* solver  */
{
    `RV' e, i
    `RM' a, b
    `TM' tmp, f

    // general configuration
    f_d   = f_d   != . ? f_d  : 0                   /* flag: zero-diagonal   */
    f_n   = f_n   != . ? f_n  : 0                   /* flag: uniform|normal  */
    tol   = tol   != . ? tol  : c("epsdouble")      /* tolerance             */
    seed  = seed  != . ? seed : 123456789           /* MC seed, distribution */
    dist  = dist       ? dist : (f_n ? &appinv_rnormal() : &appinv_runiform())

    // build the dataset (if the observation doesn't exist)
    if (st_nvar())                                                      ///
    if (sum(rowsum(st_data(., ("r","c","zerod")) :== (r,c,f_d)) :== 3))  ///
    return(0)
    else (r,c,f_d)
    // TM problem, `a` and `b`
    a = I(r)#J(1,c,1)\J(1,r,1)#I(c)\
```

Peer-reviewed version available at *Mathematics* **2025**, *13*, 3476; doi:https://doi.org/10.3390/math13213476

```
        (f_d ? (tmp=(I(min((r, c)))#(1,J(1,c,0)))[.,(1..min((r,c))*c)]),
               J(rows(tmp),r*c-cols(tmp),0)
             : J(0,r*c,.))                          /* diagonal of ‘a’ -> 0  */
    /* format: %e[2].0fc                                                     */
    f = (tmp=trunc(floatround(log10(n)))) + trunc(floatround(tmp / 3)) + 2
    printf("\n{txt}Simulations ({res}%"+ strofreal(f - 1)+".0fc{txt})\n", n)
    printf("----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5\n")
    /* perform MC with n simulations                                         */
    rseed(seed)
    for (i=1;i<(n+1);i++) {
        e = e\sqrt(quadcross((tmp=(b=(*dist)(r+c + (f_d ? min((r,c)) : 0), 1)) -
              a * svsolve(a, b, ., tol)), tmp) / r / variance(b))
        if (! mod(i, 5)) { printf("."); displayflush(); }
        if (! mod(i, 50))  printf("%"+strofreal(f)+".0fc\n", i)
    }
    /**/
    (void) stata("preserve",                                              1)
    (void) st_dropvar(.)                        /* e -> summarizeby      */
    (void) st_addvar("double", "NRMSE",                                   1)
    (void) st_addobs(rows(e),                                             1)
    (void) st_store(., ("NRMSE"),                                         e)
    (void) stata("summarizeby, clear d",                                  1)
    (void) st_addvar("long", ("r","c","zerod","rank","nvar"))
    (void) st_store(st_nobs(), ("r","c","zerod","rank","nvar"),
                               (r,c,f_d,rank(a),(f_d ? r*c - min((r,c)) : r*c)))
    (void) stata("save ‘tmpf’, replace",                                  1)
    /* add variables (if none are declared)                               */
    (void) stata("restore",                     /* pre-save result    */ 1)
    (void) stata("append using ‘tmpf’",                                   1)
    (void) stata("save AP____DATA_pre-save.dta, replace",                 1)
    return(1)
}

‘RM’ appinv_runiform(‘RS’ r, ‘RS’ c) return(runiform(r,c))         /* dummy   */
‘RM’ appinv_rnormal(‘RS’ r, ‘RS’ c) return(rnormal(r,c,0,1))

for(i=1;i<=rows((tmp=d#J(rows(d),1,1),J(rows(d),1,1)#d));i++) {    /* start   */
    (void) appinv_dist(tmp[i,1], tmp[i,2], n, /* zerod */ 0, /* normal */ 1)
    (void) appinv_dist(tmp[i,1], tmp[i,2], n, /* zerod */ 1, /* normal */ 1)
}

end
rm "AP____DATA_pre-save.dta"                          // delete pre-save
sca ‘t’ = clock(c(current_date) + " " + c(current_time), "DMYhms") - ‘t’
di as res "Completed in ‘=round(‘t’/1000, .001)’s"     // print timestamp

**# Bookmark #7
* ==== FINALIZE DATA ========================================================= *
// drop obsolete
drop if r == 1 & c == 1

// order and sort
order variable N r c zerod nvar
sort  r c zerod

// add data and variable labels
la data "Monte Carlo: mean NRMSE in general/zero-diagonal TM models, N = 50,000"
****
la var variable "variable name"
la var N        "observations"
la var r        "rows"
la var c        "columns"
la var zerod    "0 general TM, 1 zero-diagonal TM"
la var nvar     "variables in ‘a‘"
```

Peer-reviewed version available at Mathematics 2025, 13, 3476; doi:https://doi.org/10.3390/math13213476

```
la var rank     "rank of ‘a‘"

// add data and variable notes
notes: Generated with random-number seed 123456789, descriptive /*
     */statistics collected with the help of summarizeby (SSC)
notes: Compiled by $compiled_by on TS

// save
compress                                                // compress data
datasignature set, reset
****
save ‘DTA_FILE’, replace                                // cloud-proof saving
copy ‘DTA_FILE’ "AP____DATA_$date.dta", replace
****
preserve
keep r c zerod mean sd p*
mata: fputmatrix((tmp=fopen("AP____DATA_$date.mmat", "w")), st_data(.,.))
mata: fclose(tmp)

**# Bookmark #8
* ==== CLEAN UP ============================================================ *
cap log close data                                      // close log
clear all
```

Listing B.1. Stata code from the Monte Carlo experiment dedicated to mapping the distribution of NRMSE.

References

1. Nocedal, J.; Wright, S.J. *Numerical Optimization*, 2 ed.; Springer Series in Operations Research, Springer: New York, NY, 2006; p. 664. https://doi.org/10.1007/978-0-387-40065-5.
2. Boyd, S.; Vandenberghe, L. *Convex Optimization*, 1 ed.; Cambridge University Press: Cambridge, 2004; p. 727. https://doi.org/10.1017/CBO9780511804441.
3. Sydsæter, K.; Hammond, P.; Seierstad, A.; Strøm, A. *Further Mathematics for Economic Analysis*, 2 ed.; FT Prentice Hall: Harlow; München, 2011; p. 616.
4. Gentle, J.E. *Matrix Algebra: Theory, Computations and Applications in Statistics*, 3 ed.; Springer Texts in Statistics, Springer: Cham, 2024; p. 725. https://doi.org/10.1007/978-3-031-42144-0.
5. Dantzig, G.B. Reminiscences about the Origins of Linear Programming. *Memoirs of the American Mathematical Society* **1984**, *48*, 1–11. https://doi.org/10.1090/memo/0298.
6. Dantzig, G.B. *Linear Programming and Extensions*, 1 (reprint of 1963) ed.; Princeton Landmarks in Mathematics and Physics, Princeton University Press: Princeton, NJ, 1998; p. 656.
7. Koopmans, T.C. *Activity Analysis of Production and Allocation: Proceedings of a Conference*, 1 ed.; Wiley: New York, NY, 1951; p. 404.
8. Allen, R.G.D. *Mathematical Economics*, 2 (reprint of 1959) ed.; Palgrave Macmillan: London, 1976; p. 812. https://doi.org/10.1007/978-1-349-81547-0.
9. Lancaster, K. *Mathematical Economics*, 1 ed.; Dover Publications: New York, NY, 1987; p. 411.
10. Intriligator, M.D. *Mathematical Optimization and Economic Theory*, 1 (reprint of 1971) ed.; Classics in Applied Mathematics, SIAM: Philadelphia, PA, 2002; p. 508. https://doi.org/10.1137/1.9780898719215.
11. Intriligator, M.D.; Arrow, K.J. *Handbook of Mathematical Economics*, 1 ed.; Handbooks in Economics, North-Holland: Amsterdam; New York, NY, 1981; p. 378.
12. Dorfman, R.; Samuelson, P.A.; Solow, R.M. *Linear Programming and Economic Analysis*, 1 (reprint of 1958) ed.; Dover Books on Advanced Mathematics, Dover Publications: New York, NY, 1987; p. 525.
13. Frühwirth, T.; Abdennadher, S. *Essentials of Constraint Programming*, 1 ed.; Cognitive Technologies, Springer: Berlin; Heidelberg, 2003; p. 144. https://doi.org/10.1007/978-3-662-05138-2.
14. Rossi, F.; van Beek, P.; Walsh, T., Eds. *Handbook of Constraint Programming*, 1 ed.; Vol. 2, *Foundations of Artificial Intelligence*, Elsevier: Amsterdam; Boston, MA, 2006; p. 955.
15. Bolotov, I. Modeling of Time Series Cyclical Component on a Defined Set of Stationary Points and Its Application on the U.S. Business Cycle. In Proceedings of the The 8th International Days of Statistics and Economics. Melandrium, Sep 11–13 2014, pp. 151–160.

16. Bolotov, I. Modeling Bilateral Flows in Economics by Means of Exact Mathematical Methods. In Proceedings of the The 9th International Days of Statistics and Economics. Melandrium, Sep 10–12 2015, pp. 199–208.
17. Rao, C.R.; Mitra, S.K. *Generalized Inverse of Matrices and Its Applications*, 1 ed.; Wiley Series in Probability and Mathematical Statistics, Wiley: New York, NY, 1971; p. 240. https://doi.org/10.1002/SERIES1345.
18. Ben-Israel, A.; Greville, T.N.E. *Generalized Inverses: Theory and Applications*, 2 (reprint of 2003) ed.; CMS Books in Mathematics, Springer: New York, NY, 2006; p. 420. https://doi.org/10.1007/b97366.
19. Lawson, C.L.; Hanson, R.J. *Solving Least Squares Problems*, 1 (reprint of 1974) ed.; Classics in Applied Mathematics, SIAM: Philadelphia, PA, 1995; p. 337. https://doi.org/10.1137/1.9781611971217.
20. Wang, G.; Wei, Y.; Qiao, S. *Generalized Inverses: Theory and Computations*, 1 ed.; Vol. 53, *Developments in Mathematics*, Springer: Singapore, 2018; p. 397. https://doi.org/10.1007/978-981-13-0146-9.
21. Whiteside, M.M.; Choi, B.; Eakin, M.; Crockett, H. Stability of Linear Programming Solutions Using Regression Coefficients. *Journal of Statistical Computation and Simulation* **1994**, *50*, 131–146. https://doi.org/10.1080/00949659408811605.
22. Blair, C. Random Linear Programs with Many Variables and Few Constraints. *Mathematical Programming* **1986**, *34*, 62–71. https://doi.org/10.1007/bf01582163.
23. Blair, C. Random Inequality Constraint Systems with Few Variables. *Mathematical Programming* **1986**, *35*, 135–139. https://doi.org/10.1007/bf01580644.
24. Tikhonov, A.N.; Goncharskiy, A.V.; Stepanov, V.V.; Yagola, A.G. *Chislennyye metody resheniya nekorrektnykh zadach*, 2 ed.; Nauka: Moscow, 1990; p. 232.
25. Wolfe, P. A Technique for Resolving Degeneracy in Linear Programming. *Journal of the Society for Industrial and Applied Mathematics* **1963**, *11*, 205–211. https://doi.org/10.1137/0111016.
26. Dax, A. Linear Programming via Least Squares. *Linear Algebra and Its Applications* **1988**, *111*, 313–324. https://doi.org/10.1016/0024-3795(88)90067-5.
27. Osborne, M.R. Degeneracy: Resolve or Avoid? *Journal of the Operational Research Society* **1992**, *43*, 829–835. https://doi.org/10.2307/2583102.
28. Übi, E. Exact and Stable Least Squares Solution to the Linear Programming Problem. *Central European Journal of Mathematics* **2005**, *3*, 228–241. https://doi.org/10.2478/BF02479198.
29. Übi, E. On Stable Least Squares Solution to the System of Linear Inequalities. *Open Mathematics* **2007**, *5*, 373–385. https://doi.org/10.2478/s11533-007-0003-7.
30. Übi, E. A Numerically Stable Least Squares Solution to the Quadratic Programming Problem. *Open Mathematics* **2008**, *6*, 171–178. https://doi.org/10.2478/s11533-008-0012-1.
31. Übi, E. Mathematical Programming via the Least-Squares Method. *Open Mathematics* **2010**, *8*. https://doi.org/10.2478/s11533-010-0049-9.
32. Übi, E. Linear Inequalities via Least Squares. *Proceedings of the Estonian Academy of Sciences* **2013**, *62*, 238–248. https://doi.org/10.3176/proc.2013.4.04.
33. Dresden, A. The Fourteenth Western Meeting of the American Mathematical Society. *Bulletin of the American Mathematical Society* **1920**, *26*, 385–397. https://doi.org/10.1090/S0002-9904-1920-03322-7.
34. Bjerhammar, A. Rectangular Reciprocal Matrices, With Special Reference to Geodetic Calculations. *Bulletin géodésique* **1951**, *20*, 188–220. https://doi.org/10.1007/BF02526278.
35. Penrose, R. A Generalized Inverse for Matrices. *Mathematical Proceedings of the Cambridge Philosophical Society* **1955**, *51*, 406–413. https://doi.org/10.1017/S0305004100030401.
36. Penrose, R. On Best Approximate Solutions of Linear Matrix Equations. *Mathematical Proceedings of the Cambridge Philosophical Society* **1956**, *52*, 17–19. https://doi.org/10.1017/S0305004100030929.
37. Chipman, J.S. On Least Squares with Insufficient Observations. *Journal of the American Statistical Association* **1964**, *59*, 1078–1111. https://doi.org/10.2307/2282625.
38. Price, C.M. The Matrix Pseudoinverse and Minimal Variance Estimates. *SIAM Review* **1964**, *6*, 115–120. https://doi.org/10.1137/1006029.
39. Plackett, R.L. Some Theorems in Least Squares. *Biometrika* **1950**, *37*, 149–157. https://doi.org/10.2307/2332158.
40. Rao, C.R.; Mitra, S.K., Generalized Inverse of a Matrix and Its Applications. In *Theory of Statistics: Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume I*, 1 ed.; Le Cam, L.M.; Neyman, J.; Scott, E.L., Eds.; University of California Press: Berkeley, CA, 1972; pp. 601–620.
41. Greville, T.N.E. The Pseudoinverse of a Rectangular Matrix and Its Statistical Applications. In Proceedings of the Annual Meeting of the American Statistical Association. American Statistical Association, Dec 27–30 1958, pp. 116–121.

42. Greville, T.N.E. The Pseudoinverse of a Rectangular or Singular Matrix and Its Application to the Solution of Systems of Linear Equations. *SIAM Review* **1959**, *1*, 38–43. https://doi.org/10.1137/1001003.
43. Greville, T.N.E. Some Applications of the Pseudoinverse of a Matrix. *SIAM Review* **1960**, *2*, 15–22. https://doi.org/10.1137/1002004.
44. Greville, T.N.E. Note on Fitting of Functions of Several Independent Variables. *Journal of the Society for Industrial and Applied Mathematics* **1961**, *9*, 109–115. https://doi.org/10.1137/0109011.
45. Cline, R.E. Representations for the Generalized Inverse of a Partitioned Matrix. *Journal of the Society for Industrial and Applied Mathematics* **1964**, *12*, 588–600. https://doi.org/10.1137/0112050.
46. Cline, R.E.; Greville, T.N.E. An Extension of the Generalized Inverse of a Matrix. *SIAM Journal on Applied Mathematics* **1970**, *19*, 682–688. https://doi.org/10.1137/0119069.
47. Golub, G.H.; Kahan, W. Calculating the Singular Values and Pseudo-Inverse of a Matrix. *SIAM Journal on Numerical Analysis* **1965**, *2*, 205–224. https://doi.org/10.1137/0702016.
48. Ben-Israel, A.; Cohen, D. On Iterative Computation of Generalized Inverses and Associated Projections. *SIAM Review* **1966**, *8*, 410–419. https://doi.org/10.1137/0703035.
49. Lewis, T.O.; Newman, T.G. Pseudoinverses of Positive Semidefinite Matrices. *SIAM Journal on Applied Mathematics* **1968**, *16*, 701–703. https://doi.org/10.1137/0116057.
50. Bott, R.; Duffin, R.J. On the Algebra of Networks. *Transactions of the American Mathematical Society* **1953**, *74*, 99–109. https://doi.org/10.2307/1990850.
51. Campbell, S.L.; Meyer, C.D. *Generalized Inverses of Linear Transformations*, 1 (reprint of 1979) ed.; Classics in Applied Mathematics, SIAM: Philadelphia, PA, 2009; p. 272. https://doi.org/10.1137/1.9780898719048.
52. Meyer, Carl D., J. Generalized Inverses and Ranks of Block Matrices. *SIAM Journal on Applied Mathematics* **1973**, *25*, 597–602. https://doi.org/10.1137/0125057.
53. Hartwig, R.E. Block Generalized Inverses. *Archive for Rational Mechanics and Analysis* **1976**, *61*, 197–251. https://doi.org/10.1007/bf00281485.
54. Rao, C.R.; Yanai, H. Generalized Inverses of Partitioned Matrices Useful in Statistical Applications. *Linear Algebra and Its Applications* **1985**, *70*, 105–113. https://doi.org/10.1016/0024-3795(85)90046-1.
55. Tian, Y. The Moore-Penrose Inverses of m x n Block Matrices and Their Applications. *Linear Algebra and Its Applications* **1998**, *283*, 35–60. https://doi.org/10.1016/s0024-3795(98)10049-6.
56. Rakha, M.A. On the Moore-Penrose Generalized Inverse Matrix. *Applied Mathematics and Computation* **2004**, *158*, 185–200. https://doi.org/10.1016/j.amc.2003.09.004.
57. Baksalary, O.M.; Trenkler, G. On Formulae for the Moore-Penrose Inverse of a Columnwise Partitioned Matrix. *Applied Mathematics and Computation* **2021**, *403*, 1–10. https://doi.org/10.1016/j.amc.2020.125913.
58. Albert, A.E. *Regression and the Moore-Penrose Pseudoinverse*, 1 ed.; Mathematics in Science and Engineering, Academic Press: New York, NY, 1972; p. 180.
59. Dokmanić, I.; Kolundžija, M.; Vetterli, M. Beyond Moore-Penrose: Sparse Pseudoinverse. In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP). IEEE, May 26–31 2013, pp. 6526–6530. https://doi.org/10.1109/ICASSP.2013.6638923.
60. Baksalary, O.M.; Trenkler, G. The Moore-Penrose Inverse: A Hundred Years on a Frontline of Physics Research. *European Physical Journal H* **2021**, *46*, 1–10. https://doi.org/10.1140/epjh/s13129-021-00011-y.
61. Mortari, D. Least-Squares Solution of Linear Differential Equations. *Mathematics* **2017**, *5*, 48. https://doi.org/10.3390/math5040048.
62. Getson, A.J.; Hsuan, F.C. *{2}-Inverses and Their Statistical Application*, 1 (reprint of 1988) ed.; Lecture Notes in Statistics, Springer: New York, NY, 2012; p. 110. https://doi.org/10.1007/978-1-4612-3930-7.
63. Björck, A. *Numerical Methods for Least Squares Problems*, 1 ed.; SIAM: Philadelphia, PA, 1996; p. 408.
64. Kantorovich, L.V. *Matematicheskiye metody organizatsii i planirovaniia proizvodstva*, reprint ed.; Izdatel'skiy dom S.-Peterb. gos. un-ta: Saint Petersburg, 2012; p. 96.
65. Shamir, R. The Efficiency of the Simplex Method: A Survey. *Management Science* **1987**, *33*, 301–334. https://doi.org/10.1287/mnsc.33.3.301.
66. Stone, R.E.; Tovey, C.A. The Simplex and Projective Scaling Algorithms as Iteratively Reweighted Least Squares Methods. *SIAM Review* **1991**, *33*, 220–237. https://doi.org/10.1137/1033049.
67. Wagner, H.M. Linear Programming Techniques for Regression Analysis. *Journal of the American Statistical Association* **1959**, *54*, 206–212. https://doi.org/10.1080/01621459.1959.10501506.
68. Sielken, R.L.; Hartley, H.O. Two Linear Programming Algorithms for Unbiased Estimation of Linear Models. *Journal of the American Statistical Association* **1973**, *68*, 639–641. https://doi.org/10.1080/01621459.1973.10481398.

69. Kiountouzis, E.A. Linear Programming Techniques in Regression Analysis. *Applied Statistics* **1973**, *22*, 69. https://doi.org/10.2307/2346304.
70. Sposito, V.A. On Unbiased Lp Regression Estimators. *Journal of the American Statistical Association* **1982**, *77*, 652–653. https://doi.org/10.1080/01621459.1982.10477865.
71. Judge, G.G.; Takayama, T. Inequality Restrictions in Regression Analysis. *Journal of the American Statistical Association* **1966**, *61*, 166–181. https://doi.org/10.2307/2283052.
72. Mantel, N. Restricted Least Squares Regression and Convex Quadratic Programming. *Technometrics* **1969**, *11*, 763–773. https://doi.org/10.2307/1266897.
73. Donoho, D.L.; Tanner, J. Sparse Nonnegative Solution of Underdetermined Linear Equations by Linear Programming. *Proceedings of the National Academy of Sciences of the United States of America* **2005**, *102*, 9446–9451. https://doi.org/10.1073/pnas.0502269102.
74. George, K.; Osborne, M.R. On Degeneracy in Linear Programming and Related Problems. *Annals of Operations Research* **1993**, *46-47*, 343–359. https://doi.org/10.1007/BF02023104.
75. Stoer, J. On the Numerical Solution of Constrained Least-Squares Problems. *SIAM Journal on Numerical Analysis* **1971**, *8*, 382–411. https://doi.org/10.1137/0708038.
76. Waterman, M.S. A Restricted Least Squares Problem. *Technometrics* **1974**, *16*, 135–136. https://doi.org/10.1080/00401706.1974.10489160.
77. Grafarend, E.W.; Awange, J.L., Algebraic Solutions of Systems of Equations. In *Linear and Nonlinear Models*, 1 ed.; Springer: Berlin; Heidelberg, 2012; pp. 527–569. https://doi.org/10.1007/978-3-642-22241-2_15.
78. Qian, J.; Andrew, A.L.; Chu, D.; Tan, R.C.E. Methods for Solving Underdetermined Systems. *Numerical Linear Algebra with Applications* **2018**, *25*, 17. https://doi.org/10.1002/nla.2127.
79. Barnes, E.; Chen, V.; Gopalakrishnan, B.; Johnson, E.L. A Least-Squares Primal-Dual Algorithm for Solving Linear Programming Problems. *Operations Research Letters* **2002**, *30*, 289–294. https://doi.org/10.1016/s0167-6377(02)00163-3.
80. Bei, X.; Chen, N.; Zhang, S. Solving Linear Programming with Constraints Unknown. In Proceedings of the 42nd International Colloquium, ICALP 2015. Springer, Jul 6–10 2015, Vol. 9134, pp. 129–142. https://doi.org/10.1007/978-3-662-47672-7_16.
81. Pereira-López, X.; Fernández-Fernández, M.; Carrascal-Incera, A. Rectangular Input-output Models by Moore-Penrose Inverse. *Revista Electrónica De Comunicaciones Y Trabajos De ASEPUMA* **2014**, *15*, 13–24.
82. Bolotov, I. The Problem of Relationships Between Conditional Statements and Arithmetic Functions. *Mundus Symbolicus* **2012**, *20*, 5–12.
83. Bolotov, I. SUMMARIZEBY: Stata Module to Use Statsby Functionality With Summarize (version 1.1.2), 2022.
84. Bolotov, I. PYCLSP: Modular Two-Step Convex Optimization Estimator for Ill-Posed Problems (version 1.3.0), 2025.
85. Bolotov, I. PYTMPINV: Tabular Matrix Problems via Pseudoinverse Estimation (version 1.2.0), 2025.
86. Bolotov, I. PYLPPINV: Linear Programming via Pseudoinverse Estimation (version 1.3.0), 2025.