https://doi.org/10.64142/jeai.2.1.40

Journal of Engineering and Artificial Intelligence

ISSN: 3117-3357

Research Article

JEAI-26-40

A Novel Grouped-Gram-Based Algorithm for Fast and Memory-Efficient Fixed Effects Estimation

Felix Reichel[*]

Department of Economics, Johannes Kepler University Linz, Linz, Austria

[*]**Corresponding author:** Felix Reichel, Department of Economics, Johannes Kepler University Linz, Linz, Austria, E-mail: felix.reichel@jku.at

Received date: December 04, 2025; **Accepted date:** December 14, 2025; **Published date:** January 02, 2026

Citation: Reichel F (2026) A Novel Grouped-Gram-Based Algorithm for Fast and Memory-Efficient Fixed Effects Estimation. J Eng Artif Intell Vol.2 No.1: 40.

Abstract

Fixed effects models often rely on the within transformation, which constructs demeaned arrays prior to forming cross-products. This paper develops an estimator that avoids the formation of demeaned arrays by exploiting grouped summaries built from per-unit sufficient statistics. A complete derivation shows that the grouped Gram representation reproduces the classical estimator exactly. The difference lies in memory access patterns and byte movement. The grouped estimator concentrates operations into unit-level accumulations, avoiding the writes associated with array centering. Gains arise once the panel reaches a scale where memory traffic governs run time. Simulations examine coefficient accuracy, bootstrap dispersion, run time and memory use.

Keywords: Fixed effects; Panel data; Grouped Gram matrix; Within transformation; Computational econometrics; Memory-efficient algorithms; Estimation in high-dimensional models

Introduction

Fixed effects models appear widely in empirical research. The standard approach applies the within transformation: subtract unit means from each observation and form cross-products from the transformed arrays. This is direct and compatible with linear algebra routines, yet the method reshapes the data and performs multiple passes across memory. These operations become costly once the panel grows large relative to cache or main memory bandwidth.

Grouped Gram representations avoid array centering by replacing the transformation with sufficient statistics. These statistics summarize each unit's contribution with sums and cross-products.

The result is mathematically identical to the classical estimator. The central distinction lies in the structure of the computation. Grouped summaries allow a single pass through the data while maintaining numerical precision.

The study draws on several literatures. Panel methods in econometrics provide the model foundations. Matrix computation references outline the algebra of Gram matrices [1-7]. Work on cache behavior and communication bounds frames the memory analysis [8-11]. The bootstrap section relies on Cameron et al.; Davison and Hinkley [12-13].

Setup and Notation

Units are indexed by i=1,...,N and periods by t=1,...,T_i. The model is

$$y_{it} = x_{it}'\beta + \alpha_i + \varepsilon_{it} \qquad (1)$$

with covariates $x_{it} \in \mathbb{R}^p$,, unknown coefficient vector β and unit fixed effects α_i.

Define

$$\bar{x}_i \overset{\text{def}}{=} \frac{1}{T_i}\sum_{t=1}^{T_i} x_{it}, \bar{y}_i \overset{\text{def}}{=} \frac{1}{T_i}\sum_{t=1}^{T_i} y_{it}$$

The within-transformed values are

$$x_{it}^* = x_{it} - \bar{x}_i, y_{it}^* = y_{it} - \bar{y}_i$$

The estimator is

https://doi.org/10.64142/jeai.2.1.40
Journal of Engineering and Artificial Intelligence
ISSN: 3117-3357
Research Article

$$\hat{\beta}_{\text{within}} = \left(\sum_{i=1}^{N}\sum_{t=1}^{T_i} x_{it}^{*} x_{it}^{*\prime}\right)^{-1}\left(\sum_{i=1}^{N}\sum_{t=1}^{T_i} x_{it}^{*} y_{it}^{*}\right) \quad (2)$$

Grouped Gram Representation

Let the grouped sums and cross-products be

$$s_{x,i} = \sum_{t=1}^{T_i} x_{it}, s_{y,i} = \sum_{t=1}^{T_i} y_{it}$$

$$G_i = \sum_{t=1}^{T_i} x_{it} x_{it}', g_i = \sum_{t=1}^{T_i} x_{it} y_{it}$$

Derivation of gram identities

Write

$$x_{it}^{*} = x_{it} - \frac{s_{x,i}}{T_i}$$

Then

$$\sum_{t=1}^{T_i} x_{it}^{*} x_{it}^{*\prime} = \sum_{t=1}^{T_i}\left(x_{it} - \frac{s_{x,i}}{T_i}\right)\left(x_{it} - \frac{s_{x,i}}{T_i}\right)' \quad (3)$$

$$= \sum_{t=1}^{T_i} x_{it} x_{it}' - \frac{s_{x,i} s_{x,i}'}{T_i} - \frac{s_{x,i} s_{x,i}'}{T_i} + \frac{T_i s_{x,i} s_{x,i}'}{T_i^2}. \quad (4)$$

Since

$$\frac{T_i s_{\chi,i} s_{\chi,i}'}{T_i^2} = \frac{s_{\chi,i} s_{\chi,i}'}{T_i}$$

the terms combine to

$$\sum_{t=1}^{T_i} x_{it}^{*} x_{it}^{*\prime} \equiv G_i - \frac{s_{x,i} s_{x,i}'}{T_i} \quad (5)$$

The same logic applies to the cross-product:

$$\sum_{t=1}^{T_i} x_{it}^{*} y_{it}^{*} \equiv g_i - \frac{s_{x,i} s_{y,i}}{T_i}$$

Estimator based on grouped summaries

Summing across units yields the estimator

$$\hat{\beta}_{\text{fast}} = \left(\sum_{i=1}^{N}\left[G_i - \frac{s_{x,i} s_{x,i}'}{T_i}\right]\right)^{-1}\left(\sum_{i=1}^{N}\left[g_i - \frac{s_{x,i} s_{y,i}}{T_i}\right]\right) \quad (6)$$

which equals the within estimator in exact arithmetic.

Memory Path Differences

The within transformation forms x_{it}^{*} and y_{it}^{*} explicitly. Each assignment causes reads and writes across arrays of size $n = \sum_i T_i$. Since modern processors face bandwidth limitations, these memory operations dominate the floating-point operations once n is large [8,14].

The grouped estimator performs a single pass through the data. For each i, the algorithm accumulates $s_{x,i}, s_{y,i}, G_i$ and g_i. The cross-product corrections rely on these summaries alone. This reduces memory traffic by avoiding write-back of centered arrays.

The roofline model frames the method: Algorithms with high byte movement relative to floating point work fall below peak performance [9]. Since the fast estimator avoids array writes, its arithmetic intensity improves.

Communication bounds support this point [10,11]. The within transformation moves $\mathcal{O}(\text{np})$ numbers, while the grouped approach touches each x_{it} and y_{it} only once.

Numerical Considerations

Floating point paths differ due to summation order. Pairwise or block summation can reduce error [7]. The grouped estimator follows a block structure aligned with units.

Since block sizes T_i are often moderate, the structure helps numerical stability. Computing the Gram matrix from demeaned arrays produces the same algebraic result but rearranges operations. Both methods reach the same minimizer.

Bootstrap Inference

Units form natural clusters. Cluster bootstrap draws unit indices with replacement and reconstructs the panel. The bootstrap estimator uses either (2) or (6). Since the grouped approach mirrors the original estimator's structure, bootstrap dispersion remains stable across methods. The bootstrap references follow Cameron et al. and Davison & Hinkley [12,13].

Asymptotic Theory

The algebraic identities in (2)-(6) imply that both estimators share the same probability limit and asymptotic distribution. The statements below follow the standard fixed-effects framework and cover two regimes.

Let $\text{u}_{\text{it}} = \varepsilon_{\text{it}} - \bar{\varepsilon}_{\text{i}}$ denote the transformed error. Define

$$\Omega_i \overset{\text{def}}{=} \mathbb{V}\left(\sum_{t=1}^{T_i} x_{it}^{*} u_{it}\right)$$

Panel Regime 1: Large N, fixed T

https://doi.org/10.64142/jeai.2.1.40
Journal of Engineering and Artificial Intelligence
ISSN: 3117-3357
Research Article

We impose the usual regularity conditions for fixed-effects panels.

- $\{(x_{it}, u_{it}): t \le T\}$ is independent across i and weakly dependent within units.
- $\mathbb{E}[u_{it} \mid x_{i1}, \dots, x_{iT}] = 0$.
- $\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T} x_{it}^{*} x_{it}^{*\prime} \to Q$, a positive definite matrix.
- Fourth moments exist uniformly.

Theorem 7.1. Under assumptions (i)-(iv) and fixed T,

$$\sqrt{N}(\hat{\beta}_{fast} - \beta) \xrightarrow{d} \mathcal{N}(0, Q^{-1} W Q^{-1}),$$

where

$$W = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \Omega_i$$

Proof. Both estimators rely on the same transformed score $\sum_{i,t} x_{it}^{*} u_{it}$. The Gram identities guarantee equality of sample matrices. A Lindeberg argument for clusters yields the result. The full proof appears in Appendix G.

Panel regime 2: Large N, large T

Let $n = \sum_i T_i$. Assume both $N, T \to \infty$ with $n \to \infty$. Define

$$Q_n \overset{def}{=} \frac{1}{n} \sum_{i,t} x_{it}^{*} x_{it}^{*\prime}$$

- (x_{it}, u_{it}) is weakly dependent across both indices.
- $\mathbb{E}[u_{it} \mid \{x_{js}\}] = 0$.
- $Q_n \to Q$, positive definite.
- A suitable mixing condition ensures a central limit theorem for the double array $\{x_{it}^{*} u_{it}\}$.

Theorem 7.2. Under the assumptions above,

$$\sqrt{n}(\hat{\beta}_{fast} - \beta) \xrightarrow{d} \mathcal{N}(0, Q^{-1} \Sigma Q^{-1}),$$

where Σ is the long-run covariance of $x_{it}^{*} u_{it}$.

Proof. The grouped estimator rewrites the within estimator. Once the Gram identities hold, the score is identical. A central limit theorem for double arrays yields the stated limit. Formal steps appear in Appendix G.

Simulation Design

Synthetic panels were generated with Gaussian regressors, noise and unit effects. All panels were balanced with N units and T periods. Run time and memory use were measured for a grid of (N, T) combinations.

Results

Computation time as a function of sample size

Figure 1 illustrates how computation time scales with the number of cross-sectional units N. As N increases, run time rises for both estimators; however, the demeaned approach exhibits a markedly steeper growth rate. In contrast, the grouped estimator (fe_fast) scales more smoothly, reflecting reduced memory traffic and fewer array write operations. The gap widens for large N, highlighting the efficiency gains of avoiding explicit within transformations.

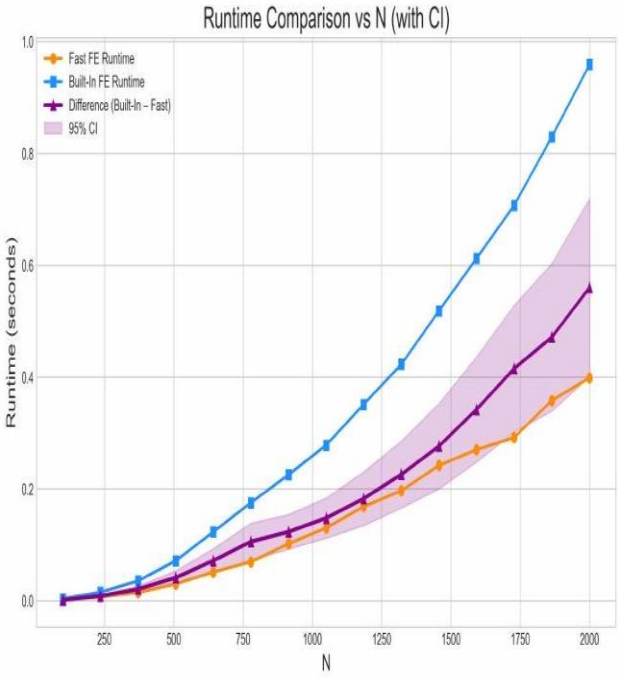


Figure 1: Computation time for fe_fast and the demeaned estimator as the number of crosssectional units N increases. Average run time rises for both estimators, with a steeper increase for the demeaned approach. fe_fast remains faster across the full range of N.

Computation time as a function of the number of periods

Figure 2 reports computation time as a function of the number of time periods T. Both estimators show increasing run time as panel length grows, consistent with higher per-unit accumulation costs. Nevertheless, fe_fast consistently outperforms the demeaned estimator across all values of T. This result confirms that the grouped Gram approach mitigates the computational burden associated with repeated centering operations.

https://doi.org/10.64142/jeai.2.1.40
Journal of Engineering and Artificial Intelligence
ISSN: 3117-3357
Research Article

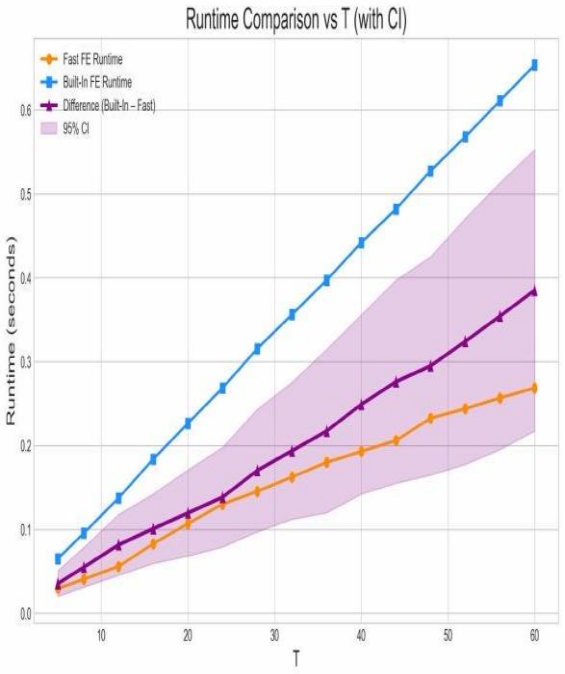


Figure 2: Computation time for fe_fast and the demeaned estimator as the number of periods T increases. Both estimators require more computation as T grows, with a stronger rise in run time for the demeaned approach. fe_fast maintains a lower run time throughout the grid.

Computation time across (N, T) combinations

Figures 3-5 summarize computation time across a grid of cross-sectional units Nand time periods T. The grouped estimator exhibits a smooth and predictable increase in run time as overall panel size expands. In contrast, the demeaned estimator and alternative implementations display sharper increases, particularly in large-N, large-T settings. These patterns indicate that explicit data transformations and dummy-variable expansions impose substantial overhead when panel dimensions grow simultaneously.

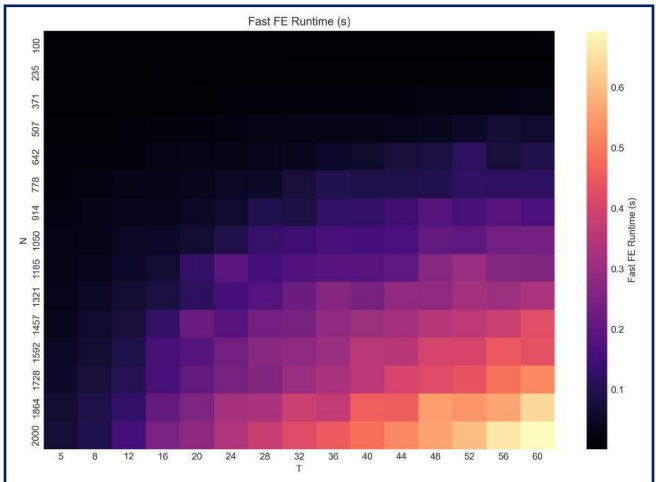


Figure 3: Average run time of fe_fast across combinations of sample size N and number of periods T. Computation increases smoothly with panel size due to grouped matrix updates.

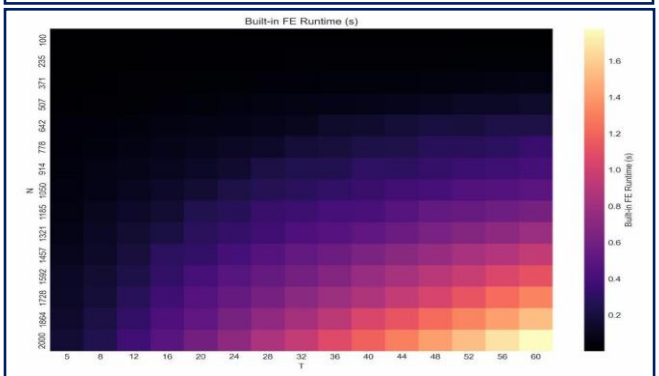
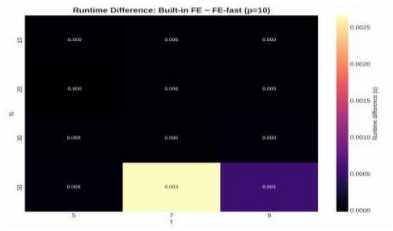
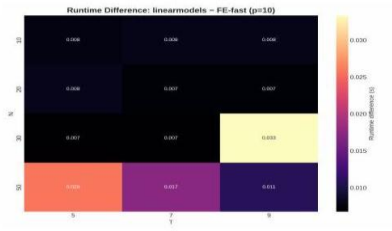


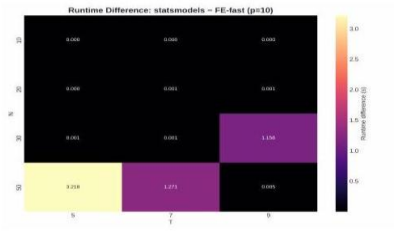
Figure 4: Average run time of the demeaned estimator across (N, T). The required per-unit transformations lead to larger increases in run time relative to fe_fast, particularly when both N and T are large.



(a) **A**: Demeaned estimator minus `fe_fast`



(b) **B**: `linearmodels` minus `fe_fast`



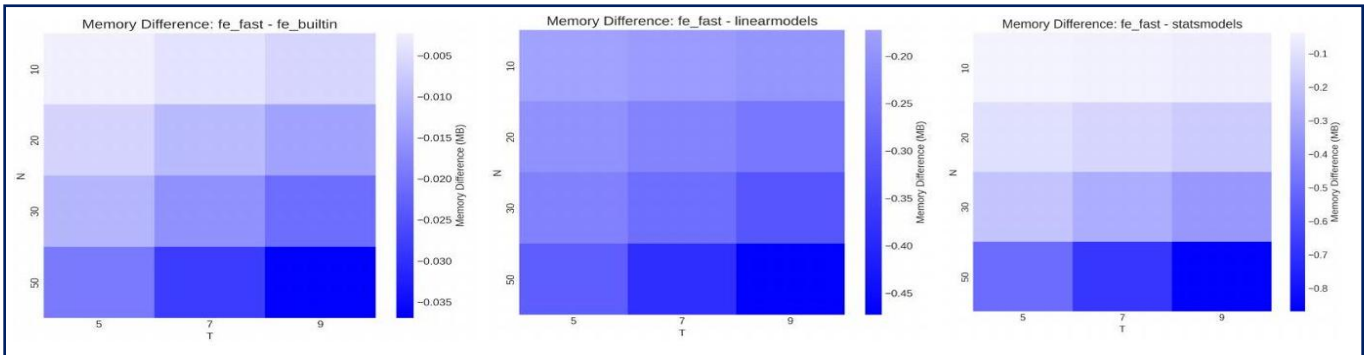
(c) **C**: `statsmodels` minus `fe_fast`

Figure 5: Differences in run time relative to fe_fast for p=10. Each panel reports the average difference in computation time across (N, T). Positive values indicate slower performance for the comparison estimator. The demeaned estimator (A) is slower throughout the grid. The linearmodels and statsmodels approaches; (B, C) show larger delays due to block operations and dummy-variable expansions.

https://doi.org/10.64142/jeai.2.1.40
Journal of Engineering and Artificial Intelligence
ISSN: 3117-3357
Research Article

Memory use across (N, T) combinations

Figure 6 compares memory usage of fe_fast relative to three alternative estimators. Across nearly all (N, T) combinations, the grouped estimator consumes less memory, as indicated by negative differences. The reduction is most pronounced for large panels, where avoiding temporary arrays and dummy-variable construction substantially limits peak memory allocation. These findings align with the algorithm's single-pass, summary-based design.



(A) fe_fast - demeaned estimator (B) fe_fast - linearmodels (C) fe_fast - statsmodels

Figure 6: Memory use of fe_fast relative to three alternative estimators for p=10. Each panel reports the difference in memory consumption across (N, T). Negative values indicate lower memory use for fe_fast. The grouped estimator avoids repeated unit-level transformations and dummy construction, leading to lower use of temporary objects across nearly all settings.

Coefficient differences

Figure 7 presents the absolute differences between slope estimates obtained from fe_fast and the classical demeaned estimator. Across all simulated panel sizes, coefficient discrepancies remain near machine precision. This confirms that the grouped Gram estimator reproduces the within estimator's numerical results while altering only the computational pathway, not the underlying statistical solution.

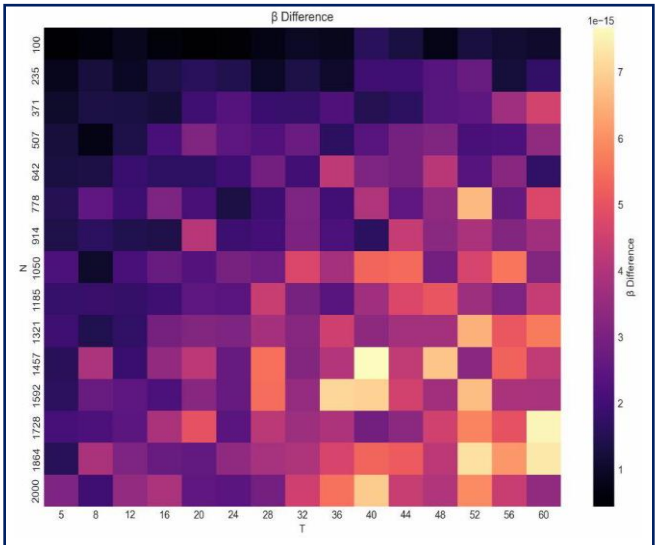


Figure 7: Absolute difference in slope estimates produced by fe_fast and the demeaned estimator across (N, T). Differences remain close to numerical tolerance ($\leq 10^{14}$) across the grid, indicating close agreement between the two approaches.

Computational complexity

This section collects FLOP counts, byte-traffic expressions and communication lower bounds for both the within and grouped estimators.

Floating-point operations

For each unit i :

$$G_i = \sum_{t=1}^{T_i} x_{it} x_{it}'$$

$$g_i = \sum_{t=1}^{T_i} x_{it} y_{it}$$

costs costs $(T_i p^2)$

$\mathcal{O}(T_i p)$

Summed across units, the grouped estimator costs

$\mathcal{O}(np^2)$.

The within transformation forms x_{it}^* explicitly:

$x_{it}^* = x_{it} - \bar{x}_i \Rightarrow \mathcal{O}(np)$ subtractions.

The Gram matrix computed from demeaned arrays again costs $\mathcal{O}(np^2)$. The FLOP gap between methods is therefore $\mathcal{O}(\mathrm{np})$, which becomes negligible when p is small but matters once byte movement dominates arithmetic.

Byte-traffic characterization

Let B denote bytes per floating point value. For the within estimator, reading and writing the transformed arrays requires

https://doi.org/10.64142/jeai.2.1.40
Journal of Engineering and Artificial Intelligence
ISSN: 3117-3357
Research Article

$$\text{Bytes}_{\text{within}} \approx 2npB.$$

The grouped estimator avoids all writes and performs only reads:

$$\text{Bytes}_{\text{fast}} \approx npB.$$

The difference is a full pass of size npB, which aligns with the measured memory gap in the simulations.

Communication lower bound

The Hong-Kung red-blue pebble model implies that computing $\sum x_{it}^{*} x_{it}^{*\prime}$ from explicit transformed arrays requires

$$\Omega(np)$$

data movement. The grouped estimator bypasses this requirement by not materializing the centered arrays, reaching the lower bound for a single pass algorithm.

Roofline interpretation

Performance is shaped by the ratio of FLOPs to bytes moved. The within estimator has arithmetic intensity roughly

$$I_{\text{within}} \approx \frac{p}{2}$$

whereas the grouped estimator reaches

$$I_{\text{fast}} \approx p$$

Higher intensity shifts the algorithm toward the compute-bound region of the roofline.

Conclusion

Grouped Gram summaries provide a route to fixed effects estimation that yields lower memory traffic than the classical method based on transformed arrays. Both reach the same estimator. The difference lies in how intermediate values are assembled. As panels grow, memory traffic dominates floating point operations. A method that minimizes movement can produce substantial gains.

Institutional Review Board Statement

Not applicable.

Data Availability Statement

Not applicable.

Conflicts of Interest

The author declares no conflicts of interest.

Funding

Supported by Johannes Kepler Open Access Publishing Fund and the federal state Upper Austria.

References

1. Arellano M (2003) Panel data econometrics. Oxford Uni Press. [Crossref],[Google Scholar]
2. Wooldridge JM (2010) Econometric analysis of cross section and panel data. MIT Press. [Google Scholar]
3. Hausman JA (1978) Specification tests in econometrics. Econ Soc 46: 1251-1271. [Crossref], [Google Scholar]
4. Hsiao C (2014) Analysis of panel data. Cambridge Uni Press. [Crossref], [Google Scholar]
5. Golub GH, Van Loan CF (2013) Matrix computations 4th edition. Johns Hopkins Uni Press. [Google Scholar]
6. Gentle JE (2007) Matrix algebra: Theory, computations and applications in statistics. Springer. [Crossref], [Google Scholar]
7. Chan TF, Golub GH, LeVeque RJ (1982) Updating formulae and a pairwise algorithm for computing sample variances. Physica, Heidelberg. [Crossref], [Google Scholar]
8. Graham SL, Snir M, Patterson CA (2005) Getting up to speed: The future of supercomputing. Natl Res Counc Rep. [Google Scholar]
9. Williams SW, Waterman A, Patterson DA (2009) Roofline: An insightful visual performance model for floating-point programs and multicore architectures. Comm ACM 52: 65-76. [Crossref], [Google Scholar]
10. Demmel J, Grigori L, Hoemmen M, Langou J (2007) Communication-avoiding linear algebra. SIAM Rev 53: 1-20.
11. Hong JW, Kung HT (1981) I/O complexity: The red-blue pebble game. 326-333. [Crossref], [Google Scholar]
12. Cameron AC, Gelbach JB, Miller DL (2008) Bootstrap-based improvements for inference with clustered errors. Rev Econ Stat 90: 414-427. [Crossref], [Google Scholar]
13. Davison AC, Hinkley DV (1997) Bootstrap methods and their application. Cambridge Uni Press. [Google Scholar]
14. McCalpin J (2006) STREAM: Sustainable memory bandwidth in high performance computers. [Google Scholar]