Scaling Reproducibility: An AI-Assisted Workflow for Large-Scale Replication and Reanalysis*

Yiqing Xu (Stanford) Leo Yang Yang (HKBU)

First version: February 17, 2026
This version: March 24, 2026

Abstract

Computational reproducibility is central to scientific credibility, yet verifying published results at scale remains costly. We develop an AI-assisted workflow for automated full-paper replication—retrieving materials, reconstructing environments, executing code, and matching outputs to point estimates reported in regression tables. We define a universe of all empirical and quantitative papers from the three top political science journals (2010–2025) and measure stated data availability using automated extraction. For a stratified sample of 384 studies, we apply the workflow to conduct full-paper replication, totaling 3,382 empirical models. We find that journal verification requirements, combined with data archiving mandates, drive reproducibility: the full-paper reproducibility rate rises from 29.6% before DA-RT adoption to 79.8% after, and conditional on accessible replication packages, 94.4% of papers are fully reproducible (237/251). As a secondary application, we apply standardized IV diagnostics to 92 studies (215 specifications), illustrating how automated execution enables systematic reanalysis across heterogeneous empirical settings.

Keywords: reproducibility, replication, research transparency, open science, AI-assisted workflows, agentic AI, causal inference

*Yiqing Xu, Assistant Professor, Department of Political Science, Stanford University. Email: yiqingxu@stanford.edu. Leo Yang Yang, Research Assistant Professor, Department of Accountancy, Economics and Finance, School of Business, Hong Kong Baptist University, Kowloon, Hong Kong SAR. Email: leoyang@hkbu.edu.hk. The authors used Claude Code and ChatGPT as research and writing assistants in preparing this manuscript. All interpretations, conclusions, and any errors remain solely the responsibility of the authors.

1. Introduction

Reproducibility is fundamental to research credibility and cumulative scientific progress. In empirical social science, reproducible analyses allow researchers to verify published claims, scrutinize identifying assumptions, and assess the practical relevance of new methodological developments. Access to real-world data and code has therefore become important both for assessing research credibility and for advancing methodology through systematic reanalysis.

Institutional norms have expanded the availability of replication packages—bundles of data, code, and documentation deposited by authors to enable reproduction of published results. Leading journals in political science and economics now require authors to post replication packages, and some conduct replication checks before publication. Yet availability alone does not ensure reproducibility at scale. Replication packages vary widely in environments, structure, and execution logic, so reproducing results across many papers remains costly and fragile. The bottleneck is operational: executing heterogeneous packages in a standardized and auditable manner requires substantial effort.

This paper develops an AI-assisted workflow for automated full-paper replication to address this execution bottleneck. The workflow reconstructs computational environments, executes code, and verifies published results. Its purpose is to determine whether existing empirical analyses can be reproduced reliably and at scale, not to introduce new estimators or diagnostics. Once data and code are standardized through this process, they can be readily reused, lowering the cost for methodological researchers to develop and apply new methods on a common empirical basis.

A central design principle is the separation of scientific reasoning from computational execution. Researchers specify the empirical targets to be reproduced—model specifications and reported quantities of interest as defined by the paper's narrative and replication package. Conditional on these inputs, replication reduces to execution-oriented tasks: acquiring replication packages, reconstructing environments, locating and running specifica-

tions, extracting analysis datasets, and harmonizing outputs. Throughout this paper, we use *reproducibility* to mean re-execution of existing code and data (National Academies of Sciences, Engineering, and Medicine, 2019), distinguishing it from *replicability*—obtaining similar findings in new studies. Earlier literature sometimes uses "replication" for the former (King, 1995).[1]

Using this workflow, we study a large set of top political science publications. We define the universe as all empirical and quantitative papers in the *American Political Science Review* (APSR), *American Journal of Political Science* (AJPS), and *Journal of Politics* (JOP) from 2010 to 2025, totaling 3,464 papers. For this universe, we measure stated data availability using automated extraction of availability statements. Combined with replication outcomes, this allows us to compare trends in stated availability and actual reproducibility over time and relate them to changes in journal policies, particularly the introduction of verification requirements.

We then apply the workflow to two tasks. First, we conduct full-paper replication for a stratified sample of 384 studies (8 papers per journal-year). For each paper, we execute the complete codebase and attempt to reproduce point estimates reported in regression tables, excluding figures, descriptive tables, and simulation results. This defines a consistent unit of evaluation across heterogeneous designs and software environments. The estimated full-paper reproducibility rate rises from 29.6% before DA-RT adoption to 79.8% after, using journal-specific implementation dates (APSR: 2016; AJPS and JOP: 2015). This increase is driven by journal policies requiring public data archiving and, especially, in-house or third-party verification. Conditional on accessible materials, 237 of 251 papers (94.4%) are fully reproducible and 239 of 251 (95.2%) are fully or largely reproducible regardless of publication

[1]The terminology varies across fields. In political science, King (1995) used "replication" to encompass both re-execution of existing analyses and new studies testing the same hypothesis. Economics has a similar tradition (Vilhuber, 2020). We follow the distinction codified by the National Academies (National Academies of Sciences, Engineering, and Medicine, 2019) and adopted by recent work in both economics (Vilhuber, 2020) and political science (Alvarez and Heuberger, 2022): *reproducibility* denotes re-execution of the original code and data, while *replicability* denotes obtaining consistent findings with new data. The compound "replication package"—the standard term for the bundle of code and data deposited by authors—retains the older usage; the property we measure when re-executing those packages is reproducibility.

year—failures trace almost entirely to missing or restricted data, not to code quality.

Second, we apply the workflow to instrumental variable (IV) designs. Building on Lal et al. (2024), we analyze 92 studies and extract 215 specifications for standardized diagnostics, illustrating how automated execution enables systematic analysis across heterogeneous empirical settings. Among the 215 specifications, the workflow achieves an 82% end-to-end autonomous success rate (55 of 67 original papers); all failures reflect unavailable replication archives rather than execution errors. Conditional on accessible materials, the workflow reproduces benchmark estimates exactly and completes all diagnostic tests.

This level of reliability is achieved through an adaptive, version-controlled process. Replication packages span multiple programming languages, estimation commands, directory structures, and coding conventions, and many failure modes arise only when new materials are encountered. Recurring issues are encoded as generalized rules in the execution layer and versioned across runs. Coverage expands across versions, while numerical behavior remains fixed within each version. The Supplementary Materials document these failure patterns and the corresponding adjustments.

In political science, King (1995) argued that scholarly norms should require sufficient data and code to reproduce published results. This vision was formalized through the Data Access and Research Transparency (DA-RT) initiative and the 2014 Journal Editors' Transparency Statement (JETS), building on public data repositories—principally the Harvard Dataverse (King, 2007), ICPSR, and the Open Science Framework—that provided the infrastructure for deposit and retrieval. The three journals in our corpus adopted verification requirements at different times (Data Access and Research Transparency (DA-RT), 2014; Key, 2016), providing institutional variation that allows us to assess whether these policies are associated with improved reproducibility in practice.

These transparency norms made possible a series of large-scale reanalysis projects (Hainmueller, Mummolo and Xu, 2019; Lal et al., 2024; Chiu et al., 2026), in which much of the effort was devoted to reconstructing environments and harmonizing replication packages

rather than developing new methods. Those experiences suggest that, once empirical targets are specified, most remaining work is procedural and therefore automatable. The present workflow operationalizes this insight.

More broadly, this project connects to long-standing concerns about research credibility, from Leamer's call to "take the con out of econometrics" (Leamer, 1983) to the credibility revolution (Angrist and Pischke, 2010; Torreblanca et al., 2026) and the replication crisis in psychology (Open Science Collaboration, 2015). These discussions emphasize verifiable analysis and expose the consequences of fragile research pipelines.

The main contribution of this paper is twofold. First, we make computational reproducibility measurable at scale and document that journal transparency policies—particularly verification requirements—are associated with substantially higher reproducibility rates in political science. Second, we show that automated execution enables standardized diagnostics across published studies, illustrated with IV designs, and can facilitate methodological research by providing reusable data and outputs.

The scope of this study is empirical social science, with data drawn from top political science journals where norms for sharing replication packages are relatively strong. The workflow requires usable code and data and does not recover results when packages are missing or fundamentally flawed. The findings therefore reflect what is feasible under current best practices. Although the analysis focuses on political science, the framework is general and can be extended to other fields with established transparency norms.

2. AI-Assisted Reproducibility Workflow

This section describes the AI-assisted workflow used to conduct full-paper replication and design-specific diagnostics. The workflow targets the execution bottleneck identified in the introduction: executing heterogeneous replication packages in a standardized and auditable manner. It does not automate methodological reasoning or introduce new statistical procedures. Instead, it standardizes and accelerates execution when usable data and code are

available.

Design principles. Large-scale reproducibility involves a basic tension between heterogeneity and determinacy. Replication materials vary widely across studies in programming language (Stata, R, and Python), directory structure, naming conventions, and documentation quality. At the same time, reproducibility requires determinacy: for a fixed pipeline version and fixed inputs, numerical outputs must not depend on ad hoc decisions, platform-specific defaults, or stochastic behavior.

The workflow resolves this tension through three separation principles. First, adaptive coordination is separated from fixed computation. An LLM orchestrator routes tasks, interprets failures, and selects among predefined recovery steps. All numerical work—data preparation, model estimation, and diagnostic computation—is executed by version-controlled program code. For a fixed pipeline version and fixed inputs, the workflow produces identical numerical outputs and retains a complete audit trail. Second, scientific reasoning is separated from execution. Researchers define the empirical targets and, where applicable, the diagnostic templates. The workflow handles the operational tasks: acquiring replication packages, reconstructing environments, executing code, and harmonizing outputs. Third, execution is separated from verification. Running an author's code successfully does not by itself establish that the paper's reported results have been reproduced. The workflow therefore treats code execution and result verification as distinct phases, with an explicit gate between them.

Architecture. The system is organized as a three-layer architecture. At the top layer, a large language model (LLM), Claude, serves as an orchestrator that dispatches tasks, interprets errors, and determines how the pipeline proceeds. The middle layer consists of structured skill descriptions that define each stage's input–output contract and record previously resolved failure patterns. At the bottom layer, deterministic agent code and statistical scripts perform all file operations and computation. The orchestrator controls

task routing but is excluded from numerical estimation and inference. Figure 1 illustrates the architecture.

Execution proceeds through three phases. In Phase A (acquisition and execution), the system extracts metadata and replication links from the paper's PDF, downloads the replication package from public repositories, prepares the code for automated execution by resolving path dependencies and environment assumptions, and executes the complete codebase. All estimation commands encountered during execution are instrumented to capture structured output, including coefficients, standard errors, sample sizes, and clustering. The output is a comprehensive log of every regression the code produces.

In Phase B (reproducibility verification), the system extracts all coefficients reported in the paper's regression tables and compares them against the code-generated outputs. Matching uses precision-aware rounding: each code coefficient is rounded to the same number of decimal places as the paper-reported value, then checked for exact equality. Optimal assignment across paper–code coefficient pairs ensures that each reported value is matched to the most appropriate code output. Phase B concludes with a reproducibility verdict: papers classified as fully or largely reproducible proceed to Phase C; others enter a bounded fix loop (up to three iterations) before the verdict is finalized.

In Phase C (diagnostic evaluation), the system identifies the research design used in each reproduced specification—currently supporting IV and difference-in-differences designs—and applies the corresponding diagnostic template. For IV designs, this includes first-stage F-statistics, Anderson–Rubin tests, bootstrap confidence intervals, jackknife sensitivity, and OLS comparisons, implemented through the `ivDiag` package (Lal and Xu, 2024). Phase C concludes with a verification check confirming that all diagnostics completed without error. The system then compiles a standardized report combining the reproducibility assessment from Phase B and the diagnostic results from Phase C.

Each phase produces explicit intermediate artifacts—structured logs, extracted datasets, matching reports, diagnostic outputs—written to disk. Agents share no hidden state. This

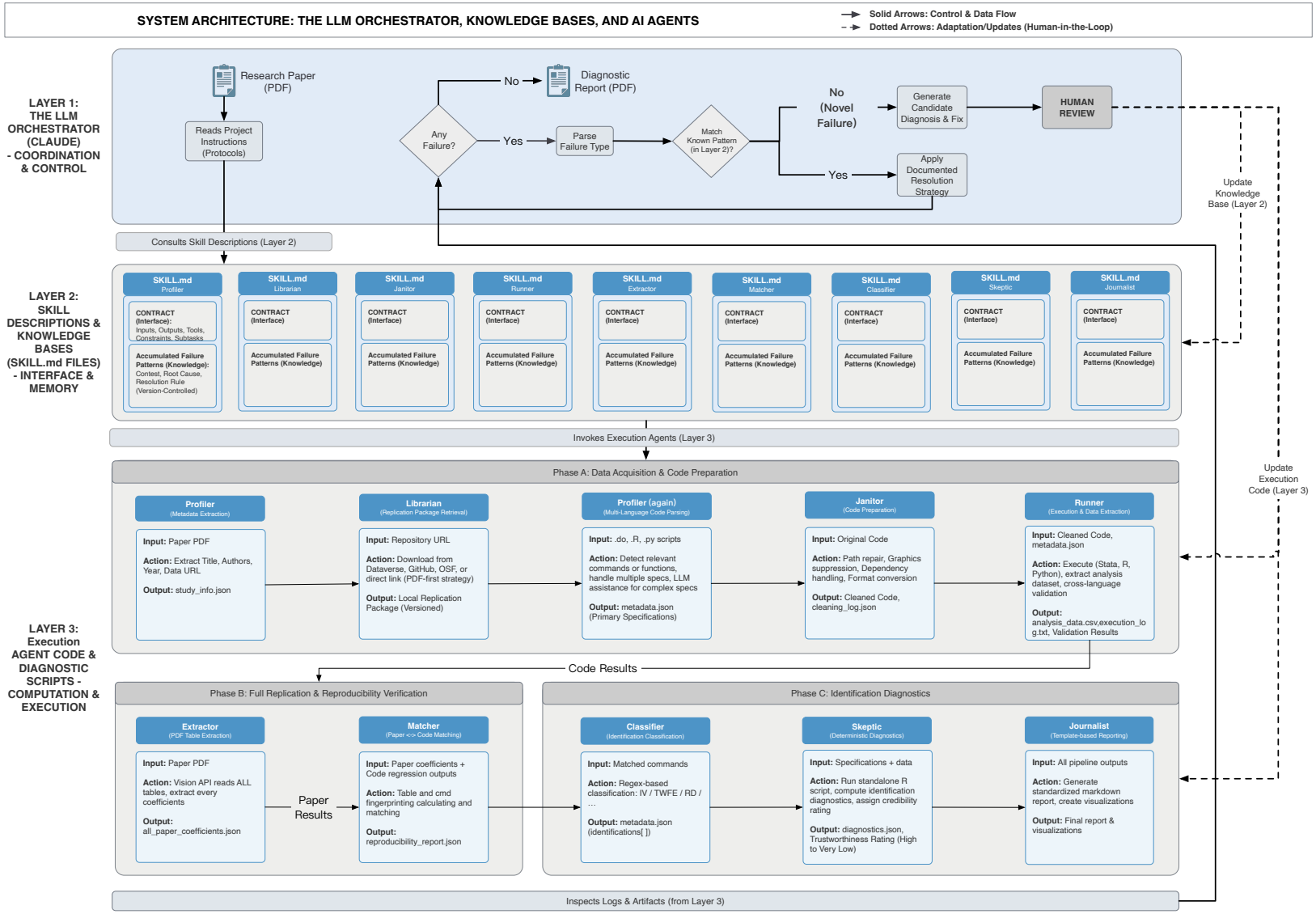


Figure 1. Overview of the AI-assisted workflow for reproducibility. The system is organized as a three-layer architecture. A top-layer LLM orchestrator routes tasks and interprets errors but does not perform estimation. The middle layer defines structured input–output contracts and records resolved failure patterns. The bottom layer consists of deterministic agent code and statistical scripts in R, Stata, and Python. Execution proceeds through three phases: acquisition and execution (Phase A), reproducibility verification (Phase B), and diagnostic evaluation (Phase C), with gates separating each phase.

modularity ensures that every step is inspectable, and execution can resume from any phase without restarting the pipeline.

Adaptation. The workflow was developed iteratively through repeated encounters with diverse replication packages. Replication materials span multiple programming languages, estimation commands, directory structures, and coding conventions, and many failure modes arise only when new materials are encountered. When a recurring failure pattern is identified, it is encoded as a generalized rule in the execution layer and version-controlled between runs. Coverage expands across pipeline versions, while numerical behavior remains fixed within each version. The executor is kept design-agnostic: extending to new research designs requires specifying a new diagnostic template rather than modifying the underlying architecture.

3. Empirical Corpus and Evaluation Design

We apply the workflow described in Section 2 to publications from three leading political science journals. The analysis proceeds at three levels. First, we define a universe of empirical and quantitative papers and measure stated data availability at the population level. Second, we draw a stratified sample and attempt full-paper replication. Third, we apply a fixed diagnostic template to a corpus of IV designs. Each level narrows the scope and increases the cost per paper. The first requires only text classification; the second requires executing replication packages; the third requires additional design-specific diagnostics. Table 1 summarizes the two evaluation tasks.

3.1. Policy Context

The three journals adopted transparency requirements at different times and through different mechanisms. The key distinction is between *availability* policies, which require authors to deposit data and code in a public repository, and *verification* policies, which require that

Table 1. Empirical Task Summary

	Full-Paper Replication	**IV Diagnostics**
Journals	APSR, AJPS, JOP	APSR, AJPS, JOP
Period	2010–2025	2010–2025
Universe	3,464 empirical, quant papers	67 + 25 IV studies
Sample	384 (8 per journal-year)	92 papers
Models/Specifications	3,382	215
Sampling design	Stratified, inverse-probability-weighted	Lal et al. (2024) + forward extension
Evaluation unit	All main-text models	Up to 3 IV specs per paper
Matching criterion	Matching point estimates	Exact 2SLS point estimate
Verdict categories	Fully / largely / partially / not reproducible	Reproduced / not reproduced
Benchmark verification	Human review of failures	Manual (original 67)

Notes. The two tasks are applied independently; the 92 IV studies were not subjected to full-paper replication. For the IV corpus, the original 67 studies are from Lal et al. (2024); the 25 additional studies (2023–2025) use identical inclusion criteria.

deposited materials be executed and checked before publication.

The discipline-wide catalyst was the 2014 JETS, organized through the DA-RT initiative. Twenty-seven political science journals—including all three in our corpus—committed to requiring that "cited data are available at the time of publication through a trusted digital repository" by January 15, 2016. JETS formalized a norm; implementation varied by journal.

AJPS moved earliest. It introduced a mandatory data archiving policy in 2012 and announced a formal replication policy in March 2015 (Jacoby, 2015). That same year, AJPS contracted with the Odum Institute for third-party verification of accepted manuscripts—one of the earliest systematic external verification programs in the social sciences. Under this policy, corresponding authors of accepted manuscripts must upload replication files to the AJPS Dataverse, and publication is contingent on successful verification. AJPS thus combined availability and verification before the JETS deadline.

JOP transitioned during 2015, coinciding with a publisher change to the University of Chicago Press and an editorial transition. By mid-2015, published JOP articles carried a standardized statement that replication packages were available in the JOP Dataverse—a

shift from earlier practice, when authors were asked only to note the location of packages without mandatory deposit (Key, 2016). JOP adopted in-house verification around early 2021, when published articles began stating that the analysis had been "successfully replicated by the JOP replication analyst."

APSR signed JETS in 2014 and revised its submission guidelines effective January 2016, but enforcement accelerated under a new editorial team beginning in August 2016. Authors of conditionally accepted manuscripts must now submit a reproducibility package to the APSR Dataverse, and the journal reviews packages to reproduce tables and figures prior to publication.

These policy shifts define three regimes: (i) pre-policy, when data sharing was voluntary or weakly enforced; (ii) post-availability, when journals required data deposit but did not verify reproducibility; and (iii) post-verification, when journals required that submitted code and data reproduce the published results as a condition for acceptance. The timing of each transition differs by journal.

3.2. Defining the Population

We accessed articles published in the APSR, AJPS, and JOP between 2010 and 2025. Each article was classified by an LLM as empirical and quantitative or not, based on full text, following the criteria in Torreblanca et al. (2026).[2] This classification identifies papers whose primary analysis is based on quantitative data from real-world sources, excluding purely theoretical, qualitative, or simulation-based studies. After applying the current screened political-science inclusion filter, the resulting universe contains 3,464 papers: 902 from APSR, 955 from AJPS, and 1,607 from JOP.

For each paper in the universe, we extract the data availability statement from the full text and record two indicators: (i) whether the paper includes a stated commitment to make replication packages available, and (ii) whether the statement provides a link to a public

[2]For this task, we used Claude Sonnet 4 Batch API; model identifier `claude-sonnet-4-20250514`.

repository where the packages can be located. These indicators measure stated availability at the population level. Whether the packages can actually be downloaded and executed is assessed only for the replication sample described below.

3.3. Replication Sample and Reproducibility Criterion

From the universe of 3,464 empirical papers, we draw a stratified sample of 384 studies: 8 papers per journal-year across 48 cells. Because the number of empirical papers varies across journal-years, we apply inverse probability weights to ensure that sample-level estimates reflect the composition of the universe.

For each sampled paper, we execute the authors' complete codebase and compare every regression coefficient it produces against every coefficient reported in the paper's regression tables. Point estimates are the most precisely defined and consistently reported quantities across heterogeneous studies; exact coefficient agreement implies that the full specification—controls, fixed effects, sample restrictions, and data transformations—was correctly reproduced. We exclude descriptive tables, simulation results, and figures: descriptive tables lack the standardized structure of regression output, and figures would require computer vision to extract numerical targets. Standard errors and other uncertainty measures are captured for downstream diagnostics but are not used as matching criteria, as they are reported inconsistently across papers and can depend on bootstrap seeds. Each code-generated coefficient is rounded to the same number of decimal places as the paper-reported value, then checked for exact equality. Code-generated and paper-reported coefficients are then matched via optimal one-to-one assignment.

We classify a paper as *fully reproducible* if 100% of point estimates in main-text regression tables are matched, *largely reproducible* if more than 80% but fewer than 100% are matched, *partially reproducible* if 50–80% are matched, and *not reproducible* otherwise.

3.4. IV Diagnostic Corpus

As a secondary application, we apply the same workflow to a corpus of IV designs. The starting point is the 67 IV studies analyzed in Lal et al. (2024), drawn from the same three journals (2010–2022) and satisfying a common set of design restrictions: linear IV models with a single endogenous regressor. The original project manually verified benchmark 2SLS point estimates for each study, which serve as fixed ground truth.

We extend this corpus by incorporating 25 additional IV studies published between 2023 and 2025 under identical inclusion criteria. For each study, the workflow replicates up to three IV specifications, including the baseline and key robustness variants. The total number of evaluated specifications increases from 70 to 215.

Unlike the full-paper task, the IV evaluation targets only the selected specifications rather than all main-text tables (Table 1). Each specification is verified through a three-step chain: (i) the authors' code reproduces the paper's coefficients, (ii) an independent R re-estimation matches the code output, and (iii) only fully verified specifications proceed to diagnostic analysis. For the 25 newly incorporated studies, we verify that the automated pipeline reproduces the reported 2SLS estimate.

4. Demonstration

We trace a single study through the full workflow to illustrate both tasks described in Section 3. We select Rueda (2017), which was previously analyzed in Lal et al. (2024), because it exemplifies a rigorous empirical design and provides a clear illustration of how the workflow operates: a clean instrumental variable strategy, a well-organized Stata replication package with clear documentation, and results spanning both IV and non-IV specifications across multiple tables, even though our workflow is capable of handling more complex cases. The paper studies the relationship between polling station size and vote buying in Colombia, instrumenting average polling station size with the maximum size set by election authori-

ties. The replication code is written in Stata—the dominant environment in our evaluation corpus. The workflow processes this paper in two steps: full-paper replication and, for the IV specifications, design-specific diagnostics.

Full-paper replication. Before applying design-specific diagnostics, the workflow attempts to reproduce all coefficients reported in the paper's regression tables. It executes the complete replication package—25 Stata scripts in this case—and compares every code-generated coefficient against the corresponding paper-reported value using precision-aware rounding and optimal assignment.

Table 2 summarizes the results. Of the 35 coefficients reported across five main-text tables, 31 are exactly reproduced, yielding an overall match rate of 88.6%. The paper is classified as largely reproducible. This verdict reflects the restricted LAPOP microdata rather than any issue with the authors' code; all coefficients that can be evaluated are reproduced exactly.

TABLE 2. FULL-PAPER REPLICATION RESULTS FOR RUEDA (2017)

Table	Coefficients	Matched	Reasons for Mismatches
Table 2	5	5	—
Table 3	6	6	—
Table 4	3	0	Missing LAPOP microdata
Table 5	16	16	—
Table 6	5	4	Missing LAPOP microdata
Total	**35**	**31**	**88.6% matched**

Notes: Table 1 in the original paper reports summary statistics and is excluded from the replication assessment. Each coefficient reported in the remaining main-text tables is compared against the corresponding code-generated value using precision-aware rounding. Table 4 cannot be reproduced because `final_LAPOP.dta` is missing from the public replication package. The unmatched coefficient in Table 6 also depends on LAPOP data. Overall verdict: largely reproducible.

All four unmatched coefficients trace to a single cause: the LAPOP survey microdata file (`final_LAPOP.dta`) is not included in the public replication package. Table 4 relies entirely on this file and cannot be reproduced; one specification in Table 6 also depends on it. The remaining 31 coefficients, spanning OLS, IV, and reduced-form estimates across four tables,

are reproduced exactly. The LAPOP microdata are subject to third-party access restrictions, and their exclusion from the public replication package is consistent with standard practice for restricted-use survey data. Automating the retrieval and integration of such externally hosted datasets is a natural extension of the workflow.

IV Diagnostic analysis. The workflow identifies all eight IV specifications reported in the main-text tables of Rueda (2017): four with outcome `e_vote_buying` and four with outcome `sum_vb`, each using a progressively richer set of controls. All eight share the same instrument (`lz_pob_mesa_f`), treatment (`lm_pob_mesa`), and clustering variable (`muni_code`).

For the IV specifications identified in Table 5, the workflow applies a three-step verification chain before proceeding to diagnostics: (i) the authors' code reproduces the paper's coefficients, (ii) an independent R re-estimation matches the code output, and (iii) only fully verified specifications receive diagnostic analysis. All eight specifications in Rueda (2017) pass this chain.

TABLE 3. DIAGNOSTIC RESULTS FOR RUEDA (2017): ALL EIGHT MAIN-TEXT IV SPECIFICATIONS

	Spec 1	Spec 2	Spec 3	Spec 4	Spec 5	Spec 6	Spec 7	Spec 8
Outcome	`e_vote_buying`	`e_vote_buying`	`e_vote_buying`	`e_vote_buying`	`sum_vb`	`sum_vb`	`sum_vb`	`sum_vb`
#Covariates	2	4	7	7	2	4	7	7
Instrument strength								
Effective F	8,598	8,310	817	170	1,955	1,908	204	60
2SLS estimate								
Coefficient	−0.984	−0.709	−1.459	−2.989	−0.699	−0.447	−2.242	−5.644
Std. error	0.142	0.111	0.463	1.247	0.235	0.125	1.300	3.650
p-value	<0.001	<0.001	0.002	0.017	0.003	<0.001	0.085	0.122
N	4,352	4,352	4,347	1,071	1,069	1,069	1,069	296
Clusters	1,098	1,098	1,098	742	632	632	632	261
Robust inference								
tF $p < 0.05$	Yes	Yes	Yes	Yes	Yes	Yes	No	No
Boot-c 95% CI	[−1.28, −0.74]	[−0.93, −0.51]	[−2.49, −0.69]	[−5.76, −0.77]	[−1.16, −0.31]	[−0.69, −0.22]	[−4.72, −0.07]	[−13.95, −0.09]
Boot-t incl. 0?	No	No	No	No	No	No	Yes	Yes
Sensitivity (jackknife)								
Range	[−0.99, −0.96]	[−0.74, −0.70]	[−1.49, −1.30]	[−3.13, −2.08]	[−0.71, −0.49]	[−0.46, −0.39]	[−2.27, −0.94]	[−6.50, −1.96]
Max Δ%	2.0	3.9	10.8	30.4	29.7	12.3	58.1	65.2
OLS comparison								
OLS coefficient	−0.675	−0.441	−0.598	−1.227	−0.925	−0.824	−0.984	−1.906
2SLS/OLS ratio	1.5	1.6	2.4	2.4	0.8	0.5	2.3	3.0

Notes: All eight specifications use instrument `lz_pob_mesa_f`, treatment `lm_pob_mesa`, and cluster on `muni_code`. Specifications 1–4 use outcome `e_vote_buying`; 5–8 use `sum_vb`. Within each outcome, specifications add progressively more controls (2, 4, 7, 7 covariates; Specs 4 and 8 use reduced samples). Bootstrap tests use 1,000 iterations with cluster-level resampling. Jackknife removes one municipality cluster at a time.

The diagnostic template, described in Section C in the Supplementary Materials, is then

applied to each verified specification. Table 3 reports the results. All eight specifications display strong first stages. The effective F-statistics range from 60 to 8,598, well above the conventional cutoff of 10, and no model is flagged for weak instruments. Diagnostic details—bootstrap confidence intervals, jackknife sensitivity, and OLS comparisons—are reported in Table 3. The workflow applies the same criteria uniformly and reports the results without substantive interpretation; the full diagnostic output is included in the standardized report. The reanalysis does not assess the credibility of the core identification assumptions. When there is a single instrument for a single endogenous regressor, these assumptions are not directly testable, and the diagnostics cannot adjudicate their validity.

Standardized report. The workflow compiles all outputs—replication results, verification chain, and diagnostic analyses—into a standardized report. The report includes an executive summary listing reproducibility verdicts and diagnostic summaries, followed by specification-level diagnostics with coefficient comparison plots, first-stage F-statistics, bootstrap confidence intervals, and jackknife sensitivity analyses. The format is identical across papers, facilitating cross-study comparison.

The full pipeline—from PDF ingestion and package retrieval to report generation—completed in less than 4 minutes of wall-clock time. Diagnostic computation accounted for most of the runtime, and no human intervention was involved. A sample diagnostic report for Rueda (2017) is included in the Supplementary Materials (Section E).

5. Main Findings

This section reports three sets of findings from deploying the AI-assisted workflow at scale. First, we document trends in stated data and code availability across the three journals over the study period. Second, we report full-paper replication results for the stratified sample of 384 papers, including inverse-probability-weighted reproducibility rates and a diagnostic classification of non-reproducible cases. Third, we present results from the IV diagnostic

corpus, showing that the automated pipeline reproduces the core patterns documented in Lal et al. (2024). Across the political-science replication tasks, the workflow processed 3,382 targeted models and executed 2,012 code scripts with no human intervention.

5.1. Data and Code Availability

Before examining replication outcomes, we summarize the transparency environment in which the workflow operates. For each paper in the universe of 3,464 empirical articles, the Profiler agent extracts whether the paper includes a data availability statement and whether a repository URL is provided. These indicators measure *stated* availability at the population level. For the stratified replication sample ($n = 384$), we go further and verify whether replication packages are actually downloadable and whether they contain usable data.

Figure 2 presents both perspectives. The upper panel plots population-level stated availability by journal. Availability is near zero for all three journals before 2012. For APSR, the sharp increase coincides with the adoption of DA-RT around 2016, with availability climbing steeply after 2019–2020 as the journal adopted in-house verification. For AJPS and JOP, stated availability has been high since the early 2010s. By the end of the study period, stated availability exceeds 90% in all three journals.

The lower panel provides a more intensive validation using the replication sample. Three lines track progressively stricter definitions of availability: stated availability, actually downloadable replication packages, and packages containing usable (non-restricted) data. The gap between these lines is revealing. Even for AJPS and JOP, where stated availability appears high early on, *consistent* data availability—where downloadable packages reliably contain usable data—only emerges after the adoption of DA-RT around 2015–2016. The shaded area between the downloadable and usable-data lines captures papers with available replication packages but entirely restricted data. In the post-DA-RT period, restricted data emerges as the primary remaining barrier to reproducibility: authors comply with archiving require-

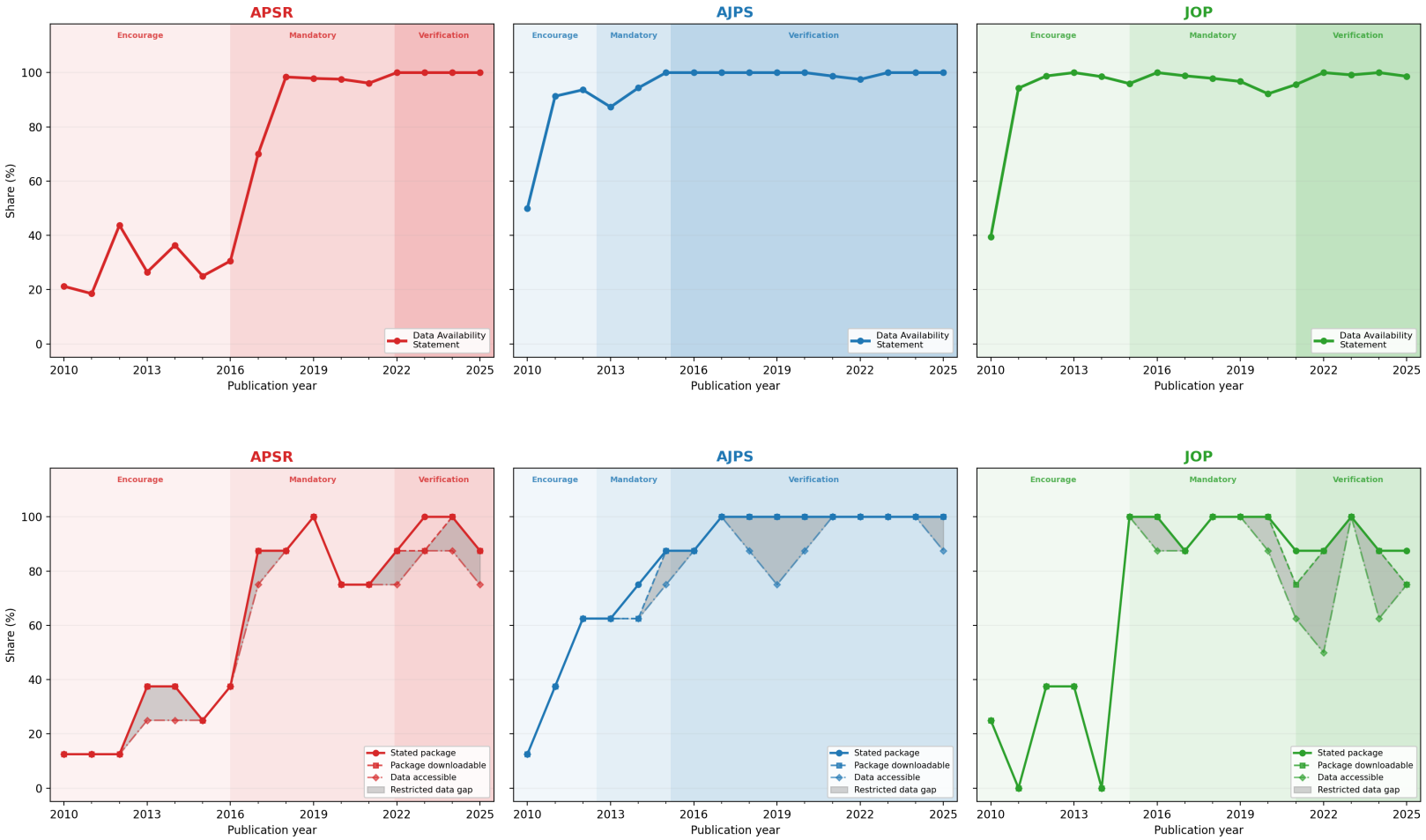


Figure 2. Replication package availability by journal over time. *Upper panel:* Population-level statistics from 3,464 empirical articles. Each dot represents the share of papers with a stated replication package by journal (APSR, AJPS, JOP) in a given year. Shaded regions indicate policy regimes: pre-policy (no mandatory data archiving), post-availability (mandatory deposit), and post-verification (mandatory reproducibility checks). *Lower panel:* Sample-level validation from the stratified replication sample ($n = 384$). Solid lines show papers with stated data availability; long-dashed lines with squares show papers whose replication packages are actually downloadable; short-dashed lines with diamonds show papers with usable data for at least one model. The shaded area between the second and third lines represents papers with an available replication package but entirely restricted data.

ments by posting code and non-restricted data, but key analyses depend on proprietary, confidential, or third-party datasets that cannot be included. Addressing this gap—through secure remote execution, synthetic data, or coordinated access agreements—is an important frontier for the field.

These trajectories reflect a broader institutional transformation. Journal mandates were effective in part because authors could deposit packages in established public repositories—principally the Harvard Dataverse, ICPSR, and the Open Science Framework—that provide persistent access, DOI-based citation, and programmatic retrieval. The infrastructure preceded the mandates and made compliance feasible at scale. The implication for reproducibility is direct: without retrievable replication packages, reproduction cannot begin. Papers published before mandatory archiving are far less likely to have accessible packages, and this shapes the results reported in the next subsection.

5.2. Full-Paper Replication at Scale

We now turn to the main empirical task: full-paper replication of 384 studies sampled from the three journals across 2010–2025. For each paper, the workflow attempts to download the replication package, execute all code, extract every coefficient from the paper's regression tables, and match code outputs to paper-reported values.

Of the 384 sampled papers, 275 (71.6%) have replication files available in a public repository; the remaining 109 have no replication package posted. Among papers with available files, 251 (65.4% of the full sample) have data that can be accessed and code that can be executed; 24 papers are blocked by restricted or proprietary data. Of the 251 papers with accessible data and executable code, 237 (94.4%) are classified as fully reproducible (Figure 3).

The dominant source of non-replication is the absence of replication packages, not code failure: missing packages (109) and restricted data (24) together account for 133 of the 147

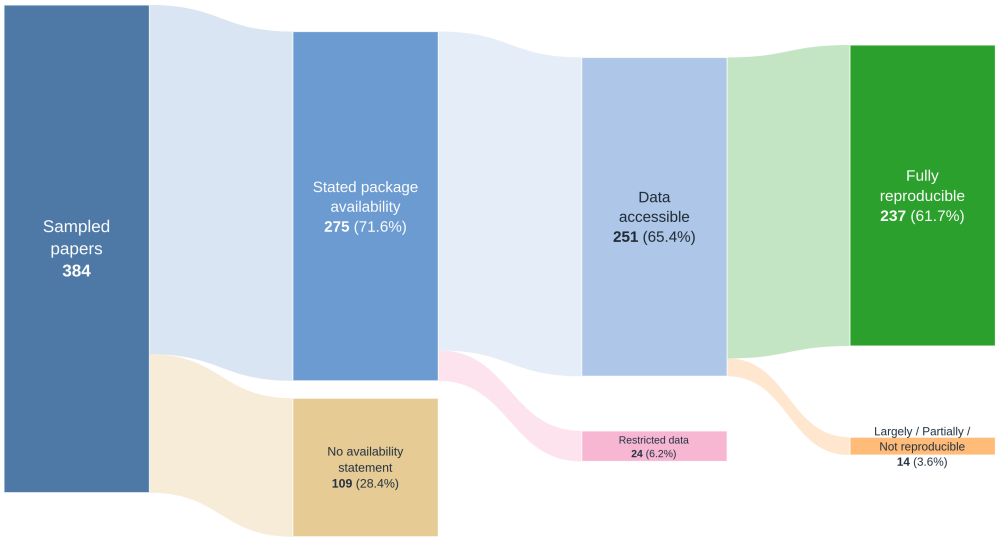


Figure 3. Replication pipeline funnel. Flow from 384 sampled papers to final reproducibility verdicts. Papers exit the pipeline at three stages: no replication package provided (109), restricted or inaccessible data (24), and below fully reproducible (14). Among papers with accessible data and executable code, 237 of 251 (94.4%) are fully reproducible.

papers that are not fully reproducible.[3] Only 14 papers with accessible materials receive a verdict below fully reproducible: 2 are largely reproducible, 9 are partially reproducible, and 3 are not reproducible—see Table S2 in the Supplementary Materials for a detailed breakdown. The adaptive execution layer contributes materially to coverage by repairing common author-side packaging errors that would otherwise block execution, but it cannot overcome packages that were never posted.

Reproducibility by journal and year. Figure 4 disaggregates these outcomes by journal and publication year. The upper panel shows paper-level verdicts: papers published before DA-RT adoption are dominated by missing replication packages (tan bars), while papers published after are predominantly fully reproducible (green bars). The timing of this shift closely tracks the adoption of mandatory data archiving and verification policies documented in Figure 2, strongly suggesting that institutional policy change—not secular trends in author

[3]That said, our workflow was able to fix repairable coding errors in 10 studies; these are classified as fully reproducible. See Table S3 in the Supplementary Materials.

Figure 4. Reproducibility by journal over time. *Upper panel:* Stacked bars show paper-level verdict composition for each journal-year cell ($n = 8$ papers per cell). Green: fully reproducible. Tan: no replication package provided. Other colors: largely, partially, or not reproducible, and restricted data. *Lower panel:* Model-level reproducibility conditional on data availability. For papers with downloadable, non-restricted replication packages, bars show the number of models attempted in each journal-year cell. Solid shading indicates replicated models; lighter shading indicates non-replicated models.

behavior—drove the improvement. The lower panel shifts from paper-level verdicts to model-level outcomes, restricting attention to papers with downloadable, non-restricted replication packages. For each journal-year cell, bars show the number of models the workflow attempted to replicate, with solid shading for replicated models and lighter shading for non-replicated ones. The contrast between panels is informative: while the upper panel shows substantial non-replication driven by missing data, the lower panel reveals that conditional on data availability, the share of non-reproducible models is very small across all journals and time periods.

Population-level estimates. Table 4 quantifies these patterns using inverse-probability-weighted rates, with journal-specific DA-RT implementation dates as cutoffs (APSR: 2016; AJPS and JOP: 2015). Because the number of empirical papers varies across journal-years—from 27 to 177—the inverse-probability-weighting (IPW) estimator reweights each cell to reflect its share of the publication universe. Across all years, the estimated pooled reproducibility rate is 67.2% with a 95% CI of [62.9%, 71.4%]. The rate is substantially higher in the post-DA-RT period: 79.8% ([74.6%, 84.9%]) compared to 29.6% ([22.1%, 37.0%]) before DA-RT adoption.

In the post-DA-RT period, the three journals show broadly similar rates: AJPS at 84.4%, JOP at 81.2%, and APSR at 73.0%. AJPS, which introduced third-party verification earliest (2015), has the highest rate; APSR (73.0%), which adopted verification only around 2021, lags despite requiring data deposit since 2016. Pre-DA-RT rates vary more widely—AJPS at 47.0% versus APSR at 16.9% and JOP at 20.6%—reflecting AJPS's earlier culture of data sharing even before formal mandates. The timing of the pre-to-post shift closely tracks verification adoption at each journal, consistent with the conclusion that active verification—not merely archiving mandates—drove the improvement in reproducibility.

Runtime. The workflow processes the full sample of 384 papers, and processing is fully parallelizable. Runtime varies substantially by outcome category. For papers where no public

Table 4. Reproducibility Rates by Journal and DA-RT Period

Journal		All Years	Pre-DA-RT	Post-DA-RT
APSR (≥2016)	Rate	61.2%	16.9%	73.0%
	95% CI	[53.1, 69.2]	[7.1, 26.7]	[63.2, 82.9]
	Papers sampled	128	48	80
AJPS (≥2015)	Rate	71.8%	47.0%	84.4%
	95% CI	[64.9, 78.7]	[32.6, 61.4]	[77.0, 91.9]
	Papers sampled	128	40	88
JOP (≥2015)	Rate	67.8%	20.6%	81.2%
	95% CI	[60.8, 74.7]	[8.9, 32.3]	[73.0, 89.5]
	Papers sampled	128	40	88
Pooled	Rate	67.2%	29.6%	79.8%
	95% CI	[62.9, 71.4]	[22.1, 37.0]	[74.6, 84.9]
	Papers sampled	384	128	256

Notes. Each journal-year cell contains $n = 8$ sampled papers. Weights are inverse sampling probabilities: $w_{jt} = N_{jt}/n_{jt}$, where N_{jt} is the number of empirical articles in journal j, year t. The IPW estimates the share of fully reproducible papers in the publication universe. 95% confidence intervals use the ratio estimator variance (Supplementary Materials, Section A.1). Cluster bootstrap CIs (5,000 resamples of journal-year cells) yield consistent intervals (Table S1). "Pre-DA-RT" and "Post-DA-RT" use journal-specific cutoffs aligned with each journal's adoption of mandatory data archiving or verification: APSR ≥2016, AJPS ≥2015, JOP ≥2015.

replication package is found, the pipeline terminates after profiling and retrieval attempts, with a median wall-clock time under 2 minutes. For papers that enter full execution—downloading materials, running code, extracting and matching coefficients—the median end-to-end time is under 10 minutes, with the most complex codebases requiring up to roughly one hour. Most wall-clock time is spent downloading replication packages and executing authors' code; LLM calls account for a small share of total computation. Once the execution layer is stabilized, the marginal cost per additional paper is low. By comparison, manual full-paper replication at this scale would require person-years of effort.

5.3. IV Diagnostic Results

As a secondary application, we evaluate the workflow on the IV diagnostic corpus described in Section 3. This corpus provides a validation anchor: the 67 original studies in Lal et al. (2024) have manually verified benchmark estimates, allowing direct assessment of the workflow's execution reliability.

For the original 67 papers, the workflow autonomously retrieves, executes, and analyzes 55 (82%). All 12 failures occur at the material retrieval stage—replication archives for these pre-2020 papers are no longer publicly available. Using archived materials retained from the earlier project, the workflow reproduces all 67 papers: conditional on accessible data and code, specification extraction, execution, and diagnostic analysis succeed in every case. Performance generalizes to the 25 newly incorporated papers, all of which succeed end to end, yielding 215 successfully processed specifications across the combined corpus (Table S4 in the Supplementary Materials). The high conditional success rate reflects the adaptive nature of the execution layer, the strong compliance with verification requirements among post-DA-RT publications, and the deterministic nature of the diagnostic template.

Empirical patterns. We apply the diagnostic template from Lal et al. (2024) to the extended corpus of 215 specifications (92 papers). Figure 5 mirrors Figure 5 in Lal et al. (2024). The core empirical finding is preserved: 2SLS estimates systematically exceed OLS estimates in magnitude, and the 2SLS-to-OLS ratio is negatively correlated with first-stage strength among observational designs ($p < 0.001$, standard errors clustered at the study level) but not among experimental designs ($p = 0.391$).

The extended sample confirms the original finding: 2SLS magnitudes systematically exceed OLS (median ratio 3.0), and the discrepancy grows as first-stage strength weakens among observational designs, consistent with bias amplification under violations of instrument validity. The pattern is absent among experimental designs. These results reinforce the argument in Lal et al. (2024) that many observational IV estimates rest on fragile identification assumptions. The key difference is scale and speed: whereas the original study required approximately four years of manual data collection, processing, and reanalysis, the automated workflow processed the expanded corpus within days.

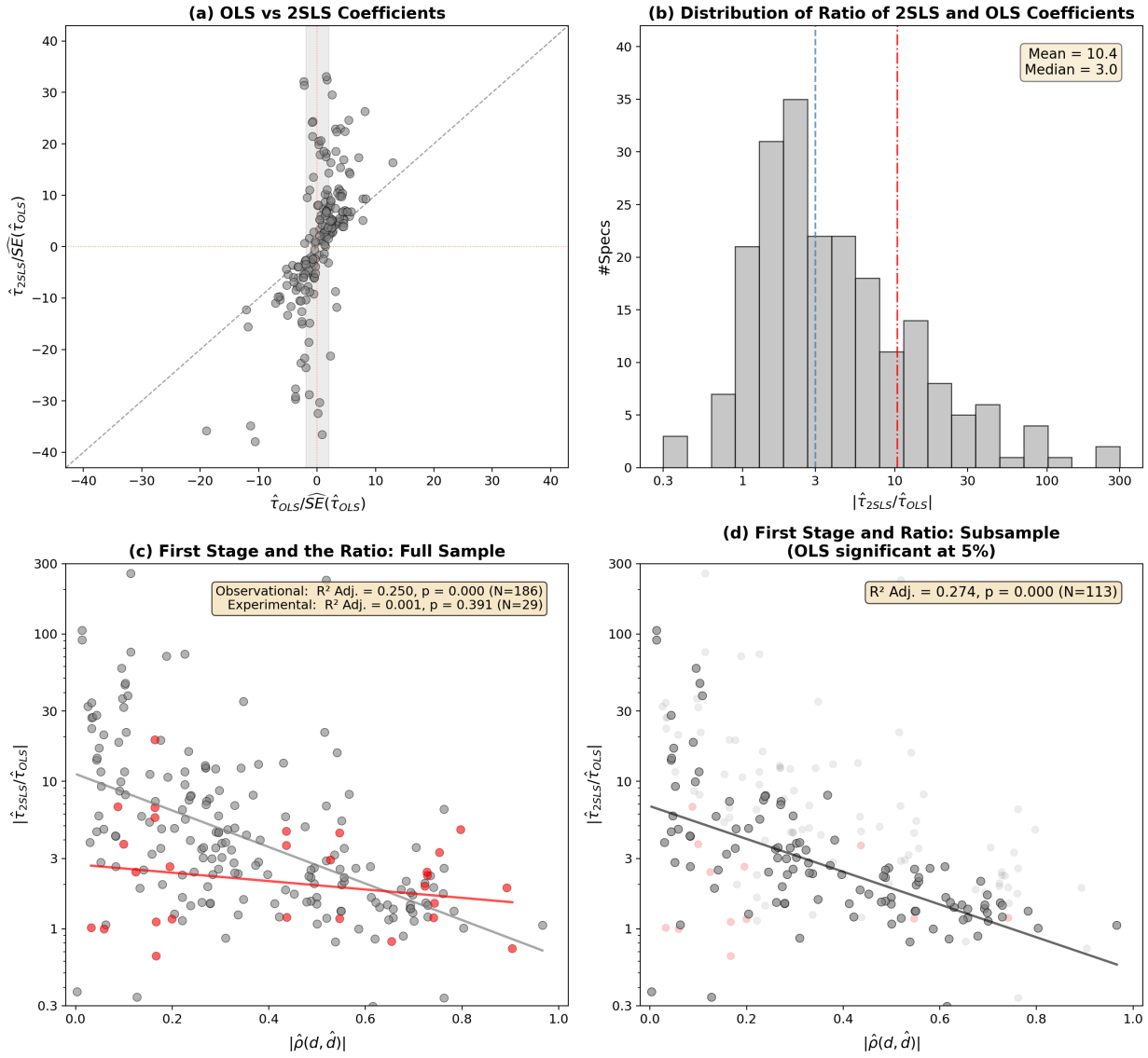


Figure 5. Relationship between OLS and 2SLS estimates. This figure replicates Figure 5 in Lal et al. (2024) using 215 specifications (versus 70 in the original). Panel (a) rescales both coefficients by the reported OLS standard errors; the shaded region corresponds to the interval $[-1.96, 1.96]$. Panel (b) presents the distribution of the log absolute ratio $|\hat{\tau}_{\text{2SLS}}/\hat{\tau}_{\text{OLS}}|$. Panels (c) and (d) examine how first-stage strength, measured by $|\hat{\rho}(d, \hat{d})|$, relates to the magnitude of the 2SLS-to-OLS ratio. Gray markers denote observational designs and red markers denote experiment-based instruments.

6. Discussion

This paper shows that computational reproducibility in political science can be verified at scale using an AI-assisted workflow, and that journal verification requirements are associ-

ated with substantially higher reproducibility rates. Among 384 sampled papers from three top journals over 2010–2025, the estimated population-level reproducibility rate rises from 29.6% before DA-RT adoption to 79.8% after. The workflow also enables standardized diagnostics at scale, reproducing and extending the IV analysis of Lal et al. (2024) across 215 specifications in days rather than years. Together, these results demonstrate that automated verification is feasible, that existing transparency policies have worked, and that the remaining frontier is expanding data access.

These results rest on institutional infrastructure that took decades to build. King (1995) argued that scholarly norms should require sufficient data and code to reproduce published results. The DA-RT initiative and the 2014 JETS formalized that argument into journal policy. Public repositories—the Harvard Dataverse (King, 2007), ICPSR, the Open Science Framework—provided the deposit and retrieval layer that made compliance feasible. AJPS's contract with the Odum Institute for third-party verification (2015), and the AEA Data Editor system in economics (2019) (Vilhuber, 2020), demonstrated that active verification, not merely archiving, is what drives compliance. Our results show, at scale, that these investments paid off: the pre-to-post DA-RT shift accounts for the bulk of the difference in reproducibility rates. Making data and code publicly available is an important first step; verification infrastructure ensures that posted materials actually work. The present workflow can substantially reduce the cost of providing that verification.

The workflow scales evaluation rather than defining evaluative standards: it relies on diagnostic templates specified by human experts and applies them uniformly across studies. Human-led replication initiatives—journal data editor programs, research transparency organizations such as OSF and BITSS, the Institute for Replication (I4R), and independent replication communities—remain indispensable for evaluating identification strategies, measurement choices, and research design.

All results are conditional on the existence and accessibility of replication packages. The workflow does not take the extra step of fetching data that may be publicly available but not

included in the replication package, such as public-use surveys or government datasets hosted on external platforms. When such data are absent from the package, the paper is classified as having inaccessible packages, even though a researcher willing to locate and download the data separately could potentially reproduce the results. The high conditional reproducibility rate (94.4% among papers with accessible packages) therefore characterizes the quality of self-contained packages, not the reproducibility of the literature as a whole. Similarly, the matching criterion targets point estimates only; extending verification to figures, uncertainty estimates, and other reported quantities is a natural next step. The workflow also does not currently handle studies that are highly computationally intensive, such as large-scale simulation studies and Bayesian analyses requiring extended MCMC sampling. These studies can in principle be incorporated with additional computational resources and are a natural target for future extension.

The workflow can be extended along three dimensions. First, we plan to apply it to additional research designs for which structured corpora already exist, including two-way fixed effects studies (Chiu et al., 2026) and heterogeneous treatment effect analyses using linear interaction models (Hainmueller, Mummolo and Xu, 2019; Liu, Liu and Xu, 2025). Second, the scope of replication can be deepened beyond benchmark specifications to reconstruct complete tables and figures, verify internal coherence across reported results, and automate diagnostics such as cluster jackknife, leave-one-out influence, and alternative inference procedures. Third, the workflow can integrate into the research and publication pipeline—assisting authors in preparing replication packages at submission, and enabling journals or third-party verification services to conduct reproducibility checks more routinely. A companion study extending this framework across six fields is in preparation.

Beyond these extensions, the broader implications concern how verification, reporting, and methodological development are organized.

Lowering the cost of verification dramatically. The most immediate implication is a substantial reduction in the marginal cost of verification. When replication packages are available, recomputing estimates and applying standardized diagnostics becomes considerably less expensive than under current practice. This does not resolve disputes about identification or theory, but it changes incentives. At present, systematic verification is rare because its cost often exceeds its expected benefit. As that cost falls, more journals may find it feasible to require in-house or third-party reproducibility checks as a condition for acceptance. Authors, anticipating a higher likelihood of auditing, may adopt more disciplined coding practices and address influence, clustering, and resampling concerns ex ante. Verification thus becomes more closely integrated into the publication process rather than applied only after controversies arise.

Standardizing diagnostic reporting. Uniform diagnostic protocols can also reshape reporting norms. If weak-instrument tests, robust inference procedures, and sensitivity analyses are implemented automatically and summarized in standardized formats, discretion in how robustness is selected and presented declines. Journals may require structured diagnostic summaries alongside main results, much as data-availability statements have become routine. Referees may increasingly expect influence diagnostics and alternative clustering checks as part of the baseline empirical presentation. Over time, graduate training may adapt to treat such diagnostics as integral components of empirical analysis rather than supplementary exercises.

Enabling large-scale reanalysis and accelerating methodological research. Harmonized analysis datasets and structured metadata enable large-scale reanalysis under consistent criteria. Each of our previous large-scale reanalysis projects—Hainmueller, Mummolo and Xu (2019), Lal et al. (2024), and Chiu et al. (2026)—took three to four years of sustained effort. Much of that time was spent harmonizing replication packages, clarifying benchmark estimands, and standardizing robustness checks across heterogeneous applications. With the

present workflow, many of these steps can be automated, substantially reducing the time required to conduct comparable large-scale reanalyses.

In the near term, this infrastructure may support more frequent and systematic reassessment of empirical literatures. Research groups, professional associations, or journals could periodically revisit published findings using updated diagnostic standards without incurring multi-year coordination costs. As harmonized corpora accumulate, empirical claims may compete not only on substantive grounds but also on demonstrated stability under shared diagnostics.

A large quantity of harmonized data will have profound implications for methodological research. In computer science, benchmark datasets such as ImageNet (Deng et al., 2009), MS COCO (Lin et al., 2014), SQuAD (Rajpurkar et al., 2016), and GLUE (Wang et al., 2018) structured progress by providing common evaluation environments. Researchers could compare algorithms under identical tasks and metrics, which facilitated cumulative improvement. Analogously, a large collection of harmonized empirical datasets with standardized diagnostic outputs can serve as a benchmark platform for causal and statistical methods. Methodologists could evaluate new estimators and inference procedures across diverse real-world applications rather than relying primarily on stylized simulations. By lowering the cost of empirical validation, the workflow may help shift methodological research toward cumulative comparison under shared empirical settings.

Taken together, these implications suggest that agentic AI workflows for reproducibility, with humans in the loop, can function as research infrastructure. They do not replace researchers' substantive judgment, but make systematic evaluation easier to conduct and harder to avoid. By lowering the cost of verification, standardizing diagnostics, and accelerating methodological development, they may help make transparency and cumulative scrutiny part of routine empirical practice.

References

Alvarez, R. Michael and Simon Heuberger. 2022. "How (Not) to Reproduce: Practical Considerations to Improve Research Transparency in Political Science." *PS: Political Science & Politics* 55(1):149–154.

Angrist, Joshua D and Jörn-Steffen Pischke. 2010. "The credibility revolution in empirical economics: How better research design is taking the con out of econometrics." *Journal of Economic Perspectives* 24(2):3–30.

Berge, Laurent. 2023. *fixest: Fast Fixed-Effects Estimation*. R package version X.X.X.
URL: *https://CRAN.R-project.org/package=fixest*

Blair, Graeme, Jasper Cooper, Alexander Coppock, Macartan Humphreys and Luke Sonnet. 2024. *estimatr: Fast Estimators for Design-Based Inference*. R package version 1.0.4.
URL: *https://CRAN.R-project.org/package=estimatr*

Chiu, Albert, Xingchen Lan, Ziyi Liu and Yiqing Xu. 2026. "Causal Panel Analysis under Parallel Trends: Lessons from a Large Reanalysis Study." *American Political Science Review* 120(1):245–266.

Data Access and Research Transparency (DA-RT). 2014. "Journal Editors' Transparency Statement (JETS).".
URL: *https://www.dartstatement.org/2014-journal-editors-statement-jets*

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li and Li Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE pp. 248–255.

Hainmueller, Jens, Jonathan Mummolo and Yiqing Xu. 2019. "How much should we trust estimates from multiplicative interaction models? Simple tools to improve empirical practice." *Political Analysis* 27(2):163–192.

Jacoby, William G. 2015. "Guidelines for Preparing Replication Files, Version 1.0.". American Journal of Political Science.
URL: *https://ajps.org/ajps-replication-policy/*

Key, Ellen M. 2016. "How Are We Doing? Data Access and Replication in Political Science." *PS: Political Science & Politics* 49(2):268–272.

King, Gary. 1995. "Replication, Replication." *PS: Political Science & Politics* 28(3):444–452.

King, Gary. 2007. "An Introduction to the Dataverse Network as an Infrastructure for Data Sharing." *Sociological Methods & Research* 36(2):173–199.

Lal, Apoorva, Mackenzie Lockhart, Yiqing Xu and Ziwen Zu. 2024. "How much should we trust instrumental variable estimates in political science? Practical advice based on 67 replicated studies." *Political Analysis* 32(4):521–540.

Lal, Apoorva and Yiqing Xu. 2024. *ivDiag: Estimation and Diagnostic Tools for Instrumental Variables Designs*. R package version 1.0.6.
URL: *https://CRAN.R-project.org/package=ivDiag*

Leamer, Edward E. 1983. "Let's take the con out of econometrics." *The American Economic Review* 73(1):31–43.

Lee, David S., Justin McCrary, Marcelo J. Moreira and Jack Porter. 2022. "Valid t-Ratio Inference for IV." *American Economic Review* 112(10):3260–3290.

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision (ECCV)*. Springer pp. 740–755.

Liu, Jiehan, Ziyi Liu and Yiqing Xu. 2025. "A Practical Guide to Estimating Conditional Marginal Effects: Modern Approaches." *arXiv preprint arXiv:2504.01355* .

Montiel Olea, José Luis and Carolin Pflueger. 2013. "A Robust Test for Weak Instruments." *Journal of Business & Economic Statistics* 31(3):358–369.

National Academies of Sciences, Engineering, and Medicine. 2019. *Reproducibility and Replicability in Science*. Washington, DC: The National Academies Press.

Open Science Collaboration. 2015. "Estimating the reproducibility of psychological science." *Science* 349(6251):aac4716.

Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. ACL pp. 2383–2392.

Rueda, Miguel R. 2017. "Small Aggregates, Big Manipulation: Vote Buying Enforcement and Collective Monitoring." *American Journal of Political Science* 61(1):163–177.
URL: *https://doi.org/10.1111/ajps.12260*

Torreblanca, Carolina, William Dinneen, Guy Grossman and Yiqing Xu. 2026. "The Credi-

bility Revolution in Political Science.".
URL: *https://arxiv.org/abs/2601.11542*

Vilhuber, Lars. 2020. "Reproducibility and Replicability in Economics." *Harvard Data Science Review* 2(4).

Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy and Samuel R. Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP*. ACL pp. 353–355.

Supplementary Materials

Scaling Reproducibility: An AI-Assisted Workflow for Large-Scale Replication and Reanalysis

A. Additional Tables and Results

A.1. IPW Estimator and Variance Formula

The IPW-weighted reproducibility rate reweights each journal-year cell by its share of the publication universe. Let $j = 1, \ldots, J$ index journals and $t = 1, \ldots, T$ index years. For each cell (j, t), let N_{jt} denote the number of empirical articles in the universe, n_{jt} the number sampled, and k_{jt} the number classified as fully reproducible. Define $\hat{p}_{jt} = k_{jt}/n_{jt}$ and let $N = \sum_{j,t} N_{jt}$. The estimator is

$$\hat{R} = \frac{\sum_{j,t} N_{jt} \hat{p}_{jt}}{\sum_{j,t} N_{jt}} = \sum_{j,t} \frac{N_{jt}}{N} \hat{p}_{jt}. \tag{A1}$$

Equivalently, with inverse probability weights $w_{jt} = N_{jt}/n_{jt}$,

$$\hat{R} = \frac{\sum_{j,t} w_{jt} k_{jt}}{\sum_{j,t} w_{jt} n_{jt}}. \tag{A2}$$

Under stratified simple random sampling without replacement within each cell, the variance is

$$\mathrm{Var}(\hat{R}) = \sum_{j,t} \left(\frac{N_{jt}}{N} \right)^2 \left(1 - \frac{n_{jt}}{N_{jt}} \right) \frac{S_{jt}^2}{n_{jt}}, \tag{A3}$$

where S_{jt}^2 is the finite-population variance of the binary reproducibility indicator in cell (j, t). For a binary outcome, a plug-in estimator is

$$\widehat{\mathrm{Var}}(\hat{R}) = \sum_{j,t} \left(\frac{N_{jt}}{N} \right)^2 \left(1 - \frac{n_{jt}}{N_{jt}} \right) \frac{\hat{p}_{jt}(1 - \hat{p}_{jt})}{n_{jt} - 1}, \tag{A4}$$

with standard error $\widehat{\mathrm{SE}}(\hat{R}) = \sqrt{\widehat{\mathrm{Var}}(\hat{R})}$.

Confidence intervals in Table 4 use the Wald interval $\hat{R} \pm 1.96 \times \widehat{\mathrm{SE}}(\hat{R})$, clipped to $[0, 1]$. As a robustness check, we also compute cluster bootstrap confidence intervals by resampling journal-year cells with replacement (5,000 replicates, percentile method). Table S1 reports the bootstrap intervals; they are similar to the analytic intervals across all subgroups.

TABLE S1. ESTIMATED REPRODUCIBILITY RATES: CLUSTER BOOTSTRAP CONFIDENCE INTERVALS

Journal		All Years	Pre-DA-RT	Post-DA-RT
APSR (≥2016)	Rate	61.2%	16.9%	73.0%
	95% CI	[46.9, 70.9]	[12.5, 21.2]	[64.9, 79.1]
	Papers sampled	128	48	80
AJPS (≥2015)	Rate	71.8%	47.0%	84.4%
	95% CI	[59.3, 82.9]	[28.0, 62.5]	[76.5, 92.2]
	Papers sampled	128	40	88
JOP (≥2015)	Rate	67.8%	20.6%	81.2%
	95% CI	[52.6, 80.4]	[5.1, 35.1]	[71.3, 91.3]
	Papers sampled	128	40	88
Pooled	Rate	67.2%	29.6%	79.8%
	95% CI	[58.8, 74.4]	[18.0, 40.9]	[74.4, 85.4]
	Papers sampled	384	128	256

Notes. Point estimates are identical to Table 4. Confidence intervals use the percentile method from 5,000 cluster bootstrap resamples of journal-year cells (seed 20260322). "Pre-DA-RT" and "Post-DA-RT" use journal-specific cutoffs: APSR ≥2016, AJPS ≥2015, JOP ≥2015. Bootstrap intervals are wider for individual journals (5–11 clusters per period) but closely match the analytic intervals for pooled estimates.

A.2. Diagnostic Classification

Tables S2 and S3 provide a paper-level classification of all 25 papers that received a verdict below fully reproducible at any point during processing, as summarized in Section 5.2. Table S2 lists the 15 papers whose gaps remain unresolved—due to missing author data, pipeline limitations, or ambiguous code-to-table mappings. Table S3 lists 10 papers where the workflow's adaptive execution repaired author-side packaging errors (missing dependencies, deprecated syntax, broken paths), recovering them to fully reproducible or near-fully reproducible status.

Table S2. Papers Remaining Below Fully Reproducible

Paper	**Verdict**	**Category**	**Why not fully reproducible**
Author-caused unresolved gaps			
Paper 1	Partial	Missing author data	A main-text table requires a data file not included in the released package; only appendix or intermediate files are provided.
Paper 2	Partial	Restricted author data	The README states that the full analysis depends on restricted registry data; only a subset of results is reproducible from released summary statistics.
Paper 3	Partial	Missing data object	A main-text table depends on a data object not shipped in the public package; the released file contains only pre-computed summaries.
Paper 4	Partial	Missing code/spec.	The released package exposes a richer model specification but not the code needed to reproduce a reduced model in the main table.
Paper 5	Partial	Missing author data	The package omits a survey source file required for one of the main-text tables.
Paper 6	Partial	Paper/code divergence	The shipped code runs and produces model output, but the remaining gap is between paper-reported values and the code-generated coefficients.
Paper 7	Not repl.	Missing data inputs	One set of main-text tables replicates, but remaining tables require cleaned input files absent from the released package.
Paper 8	Restricted	Proprietary data	Key variables were purchased from a commercial provider and excluded from the public release, preventing construction of the central exposure measures.

Continued on next page

Table S2 *(continued)*

Paper	Verdict	Category	Why not fully reproducible
Paper 9	Restricted	Permissioned data	The replication package requires variables from a government agency that are not included in the release.
Paper 10	Largely	Corrupt data file	The main-text do-file aborts because a required data file is unreadable, blocking remaining table columns before estimation begins.
Pipeline-side deep semantic cases			
Paper 11	Not repl.	Deep semantic recon.	Raw models run and some transformed quantities are visible, but most remaining paper rows depend on ambiguous post-estimation transforms that cannot be exposed safely without deeper reconstruction.
Paper 12	Partial	Rendered-table recon.	The missing table columns come from rendered output; recovered model families are not on the same displayed scale as the paper columns.
Paper 13	Partial	Incomplete rendered recovery	Printed regression-table output recovers some missing columns but does not provide a safe one-to-one reconstruction of the remaining table family.
Paper 14	Largely	Matching-pipeline recon.	The missing matched-model columns require a saved matched dataset and weighting artifact that are absent; the script does not complete the computationally intensive matching stage in batch.
Ambiguous cases			
Paper 15	Not repl.	Appendix vs. main-table mapping	The available code prints an appendix matrix rather than the paper's main table, and there is no safe local proof that the appendix outputs correspond to the unmatched main-text rows.

TABLE S3. AUTHOR-SIDE PACKAGING ERRORS REPAIRED BY THE WORKFLOW

Paper	Verdict	Category	What was repaired
Paper 16	Fully	Missing dependencies	The package assumed user-written Stata dependencies without a working local setup; installing them recovered the core tables.
Paper 17	Fully	Fragile output layer	Rendering commands stopped execution before results were fully exposed; bypassing those blockers recovered the paper.
Paper 18	Fully	Missing Stata dependency	The released script called a user-written command without shipping or installing it; isolating the failing script and running the remaining regressions recovered the paper.
Paper 19	Fully	Missing Stata dependencies	Legacy user-written commands were required for the last two main-text models; locating and installing them recovered the missing table columns.
Paper 20	Fully	Missing data & deprecated output	The package referenced a missing data file and relied on deprecated output assumptions; isolating the broken script recovered the viable analysis chain.
Paper 21	Fully	Deprecated syntax	Deprecated loop constructs and brittle setup paths blocked the main analysis pipeline; repairing those issues recovered the paper.
Paper 22	Fully	Missing Stata dependency	The script depended on a user-written package for auxiliary output; bypassing the failing script preserved the substantive regression pipeline.
Paper 23	Fully	Missing sub-script	The master do-file called a sub-script absent from the released package; running the surviving scripts directly recovered all main-text outputs.
Paper 24	Fully	Column-name bug	The released analysis script expected a column name not present in the shipped data; correcting the mismatch recovered the main table models.
Paper 25	Partial	Fragile RD bookkeeping	Legacy post-estimation bookkeeping after valid RD estimates crashed the loop; stabilizing those blocks allowed most substantive models to run.

A.3. IV Pipeline Performance

Table S4 reports stage-level and end-to-end success rates for the IV diagnostic corpus described in Section 5.3. Because failures are cumulative, the table reports both paper-level and specification-level outcomes.

TABLE S4. IV CORPUS: STAGE-LEVEL SUCCESS RATES

A. *Original Sample* (67 papers)	Input	Success	Failure	Success rate
Material retrieval (Librarian)	67	55	12	82%
Specification extraction (Profiler)	67	67	0	100%
Code execution (Runner)	67	67	0	100%
Diagnostic analysis (Skeptic)	67	67	0	100%
End-to-end success rate				**55 / 67 = 82%**
Success rate given data*				**67 / 67 = 100%**

B. *Expanded Sample* (25 papers)	Input	Success	Failure	Success rate
Material retrieval (Librarian)	25	25	0	100%
Specification extraction (Profiler)	25	25	0	100%
Code execution (Runner)	25	25	0	100%
Diagnostic analysis (Skeptic)	25	25	0	100%
End-to-end success rate				**25 / 25 = 100%**

C. *All 215 Specifications*	Input	Success	Failure	Success rate
Specification extraction (Profiler)	215	215	0	100%
Code execution (Runner)	215	215	0	100%
Diagnostic analysis (Skeptic)	215	215	0	100%
End-to-end success rate				**215 / 215 = 100%**

Notes. Panels A and B report paper-level success: a paper succeeds at a given stage if at least one specification passes. Panel C reports specification-level success. "Failure" at the material retrieval stage reflects that, among the 67 original papers, replication packages for 12 are no longer publicly available online and were manually supplied from archived copies.

B. System Architecture and Workflow

This section describes the three-layer system architecture and details of the AI workflow.

B.1. Three-Layer Architecture

The AI workflow described in the main text, which is adaptive in orchestration and deterministic in computation, is implemented through a three-layer system. The layers are ordered by control flow: the LLM orchestrator governs coordination, skill descriptions mediate task specification and accumulated knowledge, and deterministic agent code executes all operations whose outputs must be numerically reproducible.

Layer 1: The LLM Orchestrator. The orchestrator is an LLM (Claude) that manages the pipeline lifecycle. It reads a project-level instruction file that specifies global protocols for stage ordering, error handling, logging, and knowledge updates. For each stage, it consults the relevant skill description, prepares the required inputs, invokes the corresponding agent, and inspects the resulting logs and artifacts.

When a stage fails, the orchestrator parses structured log output to determine the failure type. If the error matches a previously recorded pattern in the relevant knowledge base, it applies the documented resolution strategy by modifying inputs or dispatching auxiliary steps. If the failure is novel, it generates a candidate diagnosis and proposed fix, which is subject to human review before being incorporated into the system. The orchestrator does not perform statistical estimation, transform datasets, or modify numerical routines. Its role is strictly coordinative: it decides *which* component runs and *how* to respond to execution outcomes, but it never enters the computational path that determines numerical results.

Layer 2: Skill descriptions and knowledge bases. Each agent is associated with a structured natural-language file (`SKILL.md`) that functions as both a formal interface specification and a persistent knowledge base.

The first component of the file defines the agent's contract: required inputs, expected outputs, permissible tools, execution constraints, and the sequence of subtasks. This specification ensures that the orchestrator interacts with the agent in a controlled and predictable manner.

The second component records accumulated failure patterns encountered during development and evaluation. Each entry documents the context in which a failure occurred, the root cause, and the generalized resolution rule. These entries are written in structured form to promote consistency across updates. When a new class of failure is resolved, the corre-

sponding rule is added to the relevant skill file. This process expands the system's coverage without modifying deterministic computation within a given pipeline version. Because skill files are version-controlled, each run is associated with a fixed and inspectable knowledge state.

Layer 3: Deterministic agent code and diagnostic scripts. The bottom layer consists of deterministic program code that executes all file operations and statistical procedures. Each agent is implemented as an independent Python class responsible for a specific stage of the pipeline (e.g., metadata extraction, repository retrieval, specification parsing, code preparation, execution, or reporting). Agents operate only on explicit inputs and produce explicit outputs written to disk. They share no internal state.

Statistical estimation and diagnostic procedures are executed by explicit scripts in `R`, `Stata`, and `Python`. In particular, the full diagnostic suite is implemented in a standalone R script (`diagnostics_core.R`) that consumes exported analysis datasets and produces structured diagnostic outputs. This script calls established statistical packages, including `estimatr` (Blair et al., 2024), `fixest` (Berge, 2023), `boot`, and `ivDiag` (Lal and Xu, 2024). Given the same inputs and the same pipeline version, this layer produces identical numerical results across runs.

Information flows downward as instructions and dispatch decisions (Layer 1 to Layer 3) and upward as logs, intermediate artifacts, and error messages (Layer 3 to Layer 1). Adaptation occurs only through controlled updates to Layer 2 (skill descriptions) and, when necessary, to deterministic agent code between versions. Within a fixed version, numerical outputs depend exclusively on deterministic code and exported datasets.

This separation allows the system to remain adaptive in coverage while preserving computational determinacy in each execution. Human oversight operates at the boundary between Layers 1 and 2: proposed updates to knowledge bases or deterministic routines are reviewed before being committed. The result is an architecture that evolves across versions yet remains reproducible within version.

B.2. Agents and Pipeline Stages

The workflow is implemented through a set of specialized agents, each responsible for a distinct stage of the pipeline. The agents serve both the full-paper replication task (Phases A and B) and the design-specific diagnostic task (Phase C). The IV diagnostic template is described separately in Section C.

Phase A: Acquisition and execution. These tasks are distributed across four agents. The *Profiler* extracts metadata and repository links from the published PDF, prioritizing URLs embedded in the paper over keyword-based search. The *Librarian* retrieves the replication package from public repositories (Dataverse, GitHub, OSF, or direct links). The *Janitor* prepares the code for automated execution: path dependencies are resolved, deprecated packages are handled, and environment assumptions are standardized. The *Runner* executes the complete codebase and captures all estimation outputs—coefficients, standard errors, sample sizes, and clustering information—in structured form.

For the IV diagnostic task, the workflow additionally identifies the target 2SLS specifications in the code using language-specific parsing across Stata, R, and Python. Up to three primary specifications per study are selected for diagnostic analysis. The analysis dataset for each specification is extracted and, when the original code is in Stata, independently re-estimated in R to verify cross-language consistency.

Phase B: Reproducibility verification. The *Extractor* extracts all coefficients reported in the paper's regression tables using a vision-capable language model that reads rendered table images. The *Matcher* compares extracted coefficients against code-generated outputs through precision-aware rounding and optimal one-to-one assignment, producing a per-table match rate and an overall reproducibility verdict. Phase B applies only to the full-paper replication task.

Phase C: Diagnostic evaluation. For papers that pass the reproducibility check and have a supported research design, the *Skeptic* applies a standardized diagnostic template. For IV designs, this includes first-stage F-statistics, Anderson–Rubin tests, bootstrap confidence intervals, jackknife sensitivity, and OLS comparisons, implemented through established statistical packages (Lal and Xu, 2024; Blair et al., 2024; Berge, 2023). All diagnostic procedures are deterministic and version-controlled. The *Journalist* compiles results into a standardized report with specification-level diagnostics and summary indicators.

B.3. Adaptive Execution

Replication packages span multiple programming languages, directory structures, and coding conventions. Many failure modes arise only when new materials are encountered. The workflow addresses this heterogeneity through a structured adaptation mechanism.

When the workflow encounters a recurring failure class—such as path mismatches, deprecated dependencies, or nonstandard syntax—the resolution is encoded as a generalized rule in the deterministic execution layer rather than as a paper-specific patch. Each resolution is documented in a structured knowledge base associated with the relevant pipeline stage, recording the failure context, root cause, fix, and scope of applicability. These knowledge bases are version-controlled: coverage expands across pipeline versions, while numerical behavior remains fixed within each version.

Adaptation proceeds through two channels. Well-defined, recurring failure classes are encoded as deterministic code rules that apply automatically to future papers. Patterns better captured as contextual guidance are recorded in the knowledge base and used by the orchestrator to inform task routing and error recovery. In both cases, updates occur between runs and are subject to human review before being committed. The empirical inventory of resolved issue classes is reported in Section D.

C. IV Diagnostic Template

This section briefly summarizes the statistical diagnostics applied to each IV specification. The full motivation and interpretation are detailed in Lal et al. (2024). The pipeline implements the same diagnostic template.

Instrument strength. Instrument strength is assessed using first-stage F-statistics. The effective F-statistic (Montiel Olea and Pflueger, 2013) serves as the primary indicator. Following common practice, $F < 10$ triggers a weak-instrument warning.

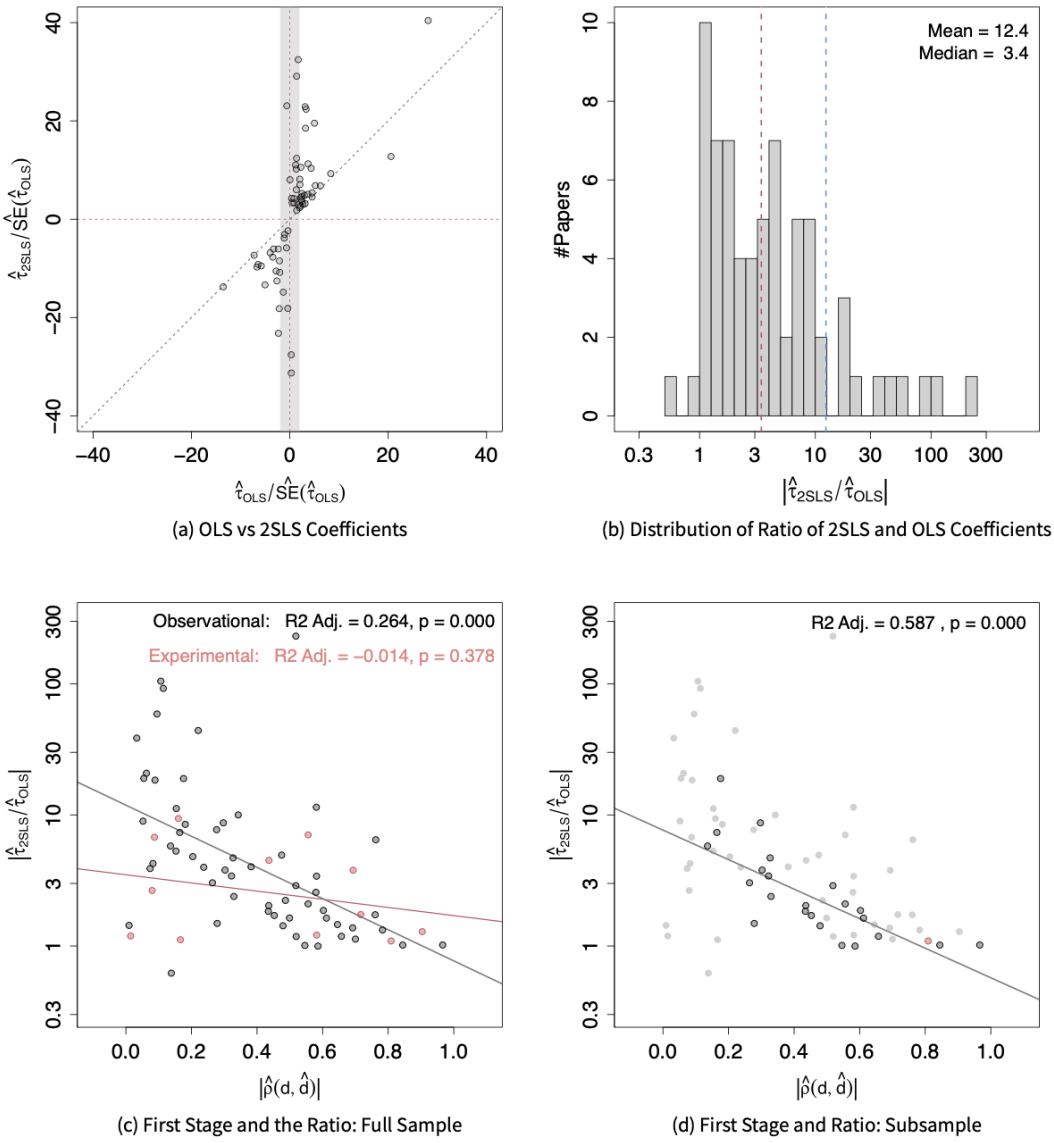
Robust inference. Inference robustness is evaluated using:

- The Anderson–Rubin (AR) test, which remains valid under weak instruments.
- Bootstrap confidence intervals (including cluster bootstrap when clustering is used).
- The tF procedure (Lee et al., 2022) in single-instrument cases, which adjusts critical values as a function of the first-stage F.

Sensitivity analysis. A leave-one-out jackknife procedure assesses the influence of individual observations or clusters on the IV estimate. Large changes relative to the baseline estimate trigger warnings.

2SLS–OLS comparison. An OLS model (without instruments) with the same outcome-treatment-covariates specification is estimated and compared to the 2SLS estimate.

Original findings. For comparison purposes, we reproduce Figure 5 of Lal et al. (2024), which is open access, below.



(a) OLS vs 2SLS Coefficients

(b) Distribution of Ratio of 2SLS and OLS Coefficients

(c) First Stage and the Ratio: Full Sample

(d) First Stage and Ratio: Subsample

Figure 5. Relationship between OLS and 2SLS estimates. In subfigure (a), both axes are normalized by reported OLS SE estimates with the gray band representing the $[-1.96, 1.96]$ interval. Subfigure (b) displays a histogram of the logarithmic magnitudes of the ratio between reported 2SLS and OLS coefficients. Subfigures (c) and (d) plot the relationship between $|\hat{\rho}(d, \hat{d})|$ and the ratio of 2SLS and OLS estimates. Gray and red circles represent observational and experimental studies, respectively. Subfigure (d) highlights studies with statistically significant OLS results at the 5% level, claimed as part of the main findings.

D. Empirical Inventory and Performance

This section documents the full set of implementation-level issues encountered and resolved during development and evaluation of the AI-assisted reproduction pipeline across 92 IV studies. Each entry records a distinct execution pattern, the agent(s) adjusted, and the corresponding resolution. The detailed inventory is organized into ten classes. Together, they provide a concrete account of the recurring irregularities that arise when replication packages are executed in a standardized, automated environment.

D.1. Classes of Implementation Variation

Across the 92 studies, we identified and resolved failures spanning ten broad classes. Table S5 summarizes these classes, affected stages, and typical issues of failure patterns.

Table S5. Classes of variation encountered and resolved across 92 IV studies

Class	Stage(s)	Representative pattern
Path and environment	Janitor, Runner	Absolute paths, global macros, sub-file mismatches
Software and language	Profiler, Runner, Skeptic	Multi-language codebases, operator translation
Data format and encoding	Janitor, Runner, Skeptic	`.tab` ambiguity, factor encoding, quoting rules
Variable name mismatch	Skeptic, Profiler	Typos, prefix matches, unresolved macros
Stata syntax dialect	Janitor	`#delimit` modes, merge syntax, `e(sample)` handling
Model specification structure	Skeptic, Profiler	Panel FE, subset conditions, bandwidth rules
Runtime resource constraints	Skeptic	Memory limits, jackknife timeouts
Graphics and interactivity	Janitor	Graphics devices, output table packages
Data acquisition	Librarian	Deprecated links, incorrect datasets
Code injection logic	Janitor	Delimiter mode, line-number drift

These classes span the full pipeline: data acquisition, code parsing, environment preparation, execution, cross-language validation, and diagnostic computation. Some patterns reflect software dialect differences (e.g., Stata version changes); others reflect repository-specific formats or naming conventions. Many failures were silent and detectable only through cross-stage consistency checks.

The detailed inventory that follows lists each resolved issue, its manifestation, the deter-

ministic repair implemented, and the responsible agent. The purpose is transparency rather than exhaustiveness of narrative: readers can trace each capability to a concrete failure pattern and code modification.

D.2. Pipeline Performance

Performance gains are substantial and empirically verifiable. The end-to-end success rate increased from approximately 63% in the initial implementation to 92.5% in the current version. These improvements were developed and evaluated on the same benchmark corpus; performance on previously unseen corpora may differ. Nevertheless, the resolved failure modes primarily reflect language-level and software-level conventions rather than idiosyncratic features of individual papers, suggesting that a meaningful portion of the gains should generalize.

The single largest improvement followed the adoption of a "PDF-first retrieval architecture," in which the system first analyzes the published PDF to infer the likely location of the replication package before attempting data acquisition. This design substantially reduced incorrect dataset downloads and increased the success rate from 62.9% to 88.6%.

Approximately 40% of resolved issue classes were first encountered during evaluation rather than anticipated ex ante. This pattern illustrates the limits of purely rule-based design. A fixed script can implement predefined rules, but it cannot account for patterns that were not foreseen. Here, new patterns were incorporated into the deterministic code base after diagnosis and review, expanding coverage across subsequent studies.

D.3. Detailed Inventory of Resolved Issues

Below we present an class-by-class inventory of resolved implementation issues.

Class 1: Path and environment variability. Replication code is typically written for a specific directory structure and operating system. When executed in a standardized workspace, these assumptions often fail. The patterns below record the path and environment issues encountered and the corresponding repairs.

Pattern	Manifestation	Resolution	Agent
Absolute paths	`cd "C:\Users\john\..."`, `setwd("/home/author/...")`	Regex detection; replace with relative paths or comment out	Janitor

Pattern	Manifestation	Resolution	Agent
Global path macros	Stata `global datadir "C:\..."` followed by `$datadir` references	Inline macro substitution before commenting out the `global` definition	Janitor
`use` command variants	Quoted paths, extensionless filenames, digit-prefixed names, macro-embedded paths	Multiple regex patterns covering all observed variants	Janitor
Sub-file reference mismatches	`do script.do` when actual filename is `script_rep.do`	Fuzzy stem-matching: search for candidates when exact match fails	Janitor
Subdirectory output files	`analysis_data.csv` generated in a subdirectory, not the expected root	Recursive `rglob()` fallback search	Runner
Platform-specific software paths	macOS vs. Linux Stata installation paths	Platform-aware path detection	Runner

Class 2: Software and language variability. The papers in the corpus use Stata, R, and Python, and several combine multiple languages within a single replication package. Supporting cross-language execution required systematic expansion of parsing and translation rules. The table below lists the software- and language-related issues resolved during development.

Pattern	Manifestation	Resolution	Agent
Multi-language codebases	Stata data preparation + R analysis in a single replication package	Profiler parses `.do`, `.R`, and `.py` files simultaneously	Profiler
Diverse IV commands	`ivreg2`, `ivregress`, `reghdfe`, `ivprobit`, `ivtobit`, `rdrobust fuzzy`, manual 2SLS	Expanding IV command pattern list; detecting manual two-stage implementations	Profiler
Stata → R variable translation	`l4.margin_index2` becomes `l4margin_index2` or `l4_margin_index2` in CSV	Three-tier resolution: exact → dot-stripped → underscore-separated → recompute from base	Runner, Skeptic, R core
Factor variable expansion	`i.year` → dummy columns `_Iyear_2000`, `_Iyear_2001`, etc.	Detect `i.` prefix; match expanded dummy patterns	Runner, Skeptic

Pattern	Manifestation	Resolution	Agent
Time-series operators	`L.`, `L4.`, `F2.`, `D.`, compound `L2D.var`	Dual-side parsing chains in Python (Runner/Skeptic) and R (`diagnostics_core.R`)	Runner, Skeptic, R core
Truncated variable names	Stata truncates to 32 characters with `~` (e.g., `incumbvotesmajor~t`)	Prefix-plus-tilde pattern matching	Runner
R formula objects with `update()`	Base formula `f <- y ~ x` modified by `update(f, . ~ . + z)`	Extract formula dictionary; apply `update()` rules to expand	Profiler
R interaction shorthand	`A * B` implying `A + B + A:B`	Detect `*` patterns; add explicit `A:B` to control list	Profiler

Class 3: Data format and encoding variability. Replication materials are distributed in multiple data formats and encoding conventions. Differences in file types, string encodings, and export formats required explicit handling to ensure consistent downstream computation. The following entries summarize the issues encountered.

Pattern	Manifestation	Resolution	Agent
`.tab` format ambiguity	Dataverse stores as `.tab` (TSV); Stata expects `.dta` (binary)	Detect `.tab` files; convert to CSV or rewrite `use` commands	Janitor
Factor-encoded strings	Column stored as `"0: Not low-education"` instead of numeric 0/1	Python preprocessing to convert to numeric before R diagnostics	Skeptic
Lost variables in Dataverse export	`pctcath` variable missing from `.tab` export	Detect and flag as data-source limitation	Skeptic
R backtick quoting	Column `_computed_outcome` quoted as `` `_computed_outcome` `` by R internals	Four-way lookup: raw → backtick → strip → fuzzy (9 call sites)	R core
Destructive `gen` safeguard	`capture drop X; gen X = expr` deletes `X` when `expr` fails	Backup-restore pattern: rename → gen → restore on failure	Janitor
Format conversion (`.dta` → CSV)	Stata binary format unreadable by R/Python	Automatic pandas-based conversion	Runner

Class 4: Variable name mismatches. Variable names extracted from code do not always match column names in the exported analysis dataset. Discrepancies arise from

typos, truncation, macro expansion, encoding differences, or computed expressions. The patterns below record the matching rules added to reconcile these differences.

Pattern	Manifestation	Resolution	Agent
Single-character typos	`lcri_euac1_r` vs. `lcri_euc1_r`	Four-tier resolution: exact → case-insensitive → Levenshtein ≤ 2 → prefix match	Skeptic
Missing suffixes	`serfperc` vs. `serfperc1`	Levenshtein distance matching	Skeptic
Prefix matches	`incumbvotes` vs. `incumbvotesmajorpercent`	Bidirectional prefix detection	Skeptic
Computed expressions	`zero1(infeels-outfeels)` is a function expression, not a column name	Expression detection + precomputation (min-max normalize, log transform)	Skeptic
Unresolved Stata macros	Metadata contains `$controls_z2` or `'controls_z2'` literally	Scan `command` field for `$xxx` references; expand using macro dictionary	Profiler
Unsplit cluster variables	`"ccode year"` → R's `make.names()` produces `"ccode.year"`	Space-based splitting normalization for cluster variables	Skeptic
Unresolved macro passthrough	`$Z`, `$C` passed to R as literal strings	Guard: detect unresolved `$macro` patterns before invoking R	Skeptic
Case inconsistencies	Variable names differ only in capitalization	Case-insensitive matching (second tier of resolution chain)	Skeptic

Class 5: Stata syntax variants. Stata syntax differs across versions and programming styles. The pipeline encountered variations in delimiter modes, legacy commands, macro conventions, and wrapper structures. The following entries summarize the syntax-related adjustments implemented in the Janitor and related agents.

Pattern	Manifestation	Resolution	Agent
`#delimit ;` mode	Semicolons replace newlines as statement terminators; multi-line commands	Delimiter state tracking; treat content between semicolons as one statement	Janitor
`#delimit` abbreviations	`#d`, `#delim`, `#delimi` are all valid	Extended regex: `#d(?:e(?:l(?:i(?:m(?:i(?:t)?)?)?)?)?)?`	Janitor
Old `merge` syntax	`merge cow year using "file.dta"` (pre-Stata 11)	Auto-detection and conversion to `merge m:1 ...`	Janitor

Pattern	Manifestation	Resolution	Agent
Weight specifications	`[aweight=w]`, `[pweight=w]` embedded in IV commands	Separate weight parsing before variable-list extraction	Runner
`e(sample)` filtering	`ivreg2 ... if e(sample)` references prior estimation's sample	Full-panel export with `janitor_esample` flag column; R filters post-lag-computation	Janitor, R core
`capture noisily` side effects	Wrapping failed commands corrupts `e(sample)` and `e(b)`	Comment out non-target IV commands entirely instead of wrapping	Janitor
`e(sample)` restoration	After `capture noisily` failure, `e(sample)` is invalid	Backward scan to find estimation command that set `e(sample)`; inject re-estimation	Janitor
User-defined commands	`edvreg` as custom `ivreg2` wrapper	Maintain list of known user-defined commands; detect `program define` blocks	Janitor
Wrapper commands	`parmby "ivreg2 ...", ...`, `bootstrap`, `jackknife`	Wrapper exclusion list: preserve IV commands inside wrappers from commenting	Janitor
Multi-line command end detection	In `#delimit ;` mode, commands span multiple lines until `;`	Extended `_find_end_of_stata_cmd()` to scan forward to semicolon in delimiter mode	Janitor

Class 6: Statistical model specification variability. The IV designs in the corpus differ in fixed effects, sub-sample conditions, clustering structures, weights, and model types. Supporting these variations required additional parsing and normalization rules. The table below lists the specification-related issues resolved.

Pattern	Manifestation	Resolution	Agent
Panel fixed effects	`xtivreg2, fe` requires `xtset panel time`	Detect `xtset` in metadata; add FE columns to IV formula	Skeptic
Stata `if` conditions	`if year >= 2000 & region == "east"`	Extract condition; pass through to R `--if_condition` parameter	Skeptic
R inline subsetting	`data = df[df$var == 0, ]`	Manual addition of `subset()` to metadata `data_prep`	Skeptic

Pattern	Manifestation	Resolution	Agent
RDD bandwidth subsetting	`rdrobust` computes bandwidth at runtime	Compute bandwidth from execution log; hardcode as numeric subset condition	Skeptic
Endogenous variable interactions	`log(providers) * as.factor(year)`	Identified as structural limitation; Journalist flags as "Non-Linear Model Approximation"	Journalist
Non-linear IV models	`ivprobit`, `ivtobit`	Journalist generates information box noting linear approximation	Journalist
Weighted regression	Stata `[pweight=w]`, R `weights = sample.size`	Cross-software weights extraction with fallback regardless of language	Skeptic
Multi-equation output	`ivprobit`/`ivtobit` produce two-line headers	Multi-line header detection and merging helper functions	Runner

Class 7: Runtime resource constraints. The datasets in the corpus range from small cross-sections to large panels exceeding one million observations. These differences create variation in memory use and runtime. The following entries summarize the constraints encountered and the safeguards implemented.

Pattern	Manifestation	Resolution	Agent
Large-dataset timeout/OOM	1.26M rows (Ritter 2016); 115K rows (Lelkes 2017)	`MAX_OBS = 100,000` sampling cap with fixed seed for reproducibility	R core
Cluster jackknife memory exhaustion	Large cluster count causes OOM	Fallback to observation-level jackknife (200 observations)	Skeptic
Code execution timeout	Some scripts exceed 600s	Configurable hard timeout (`--timeout`)	Runner
Pre-computed lag column NAs	Full-panel export has extra NAs in lag columns outside estimation sample	Compare NA counts: if pre-computed has more NAs than base, recompute from base	R core

Class 8: Graphics and interactive commands. Many replication packages assume an interactive environment with an available graphics device and user input. In a batch execution setting, such commands cause interruptions or failures. The table below records the patterns identified and the corresponding handling rules.

Pattern	Manifestation	Resolution	Agent
Common graphics commands	`graph twoway`, `histogram`, `plot()`, `ggplot()`, `plt.show()`	Regex-based commenting	Janitor
Rare Stata graphics	`cibplot`, `marginsplot`, `binscatter`, `spmap`	Expanded graphics command list	Janitor
Interactive commands	`pause`, `View()`, `browser()`, `input()`, `breakpoint()`	Comment out all interactive commands	Janitor
Output table packages	`modelsummary()`, `stargazer()`, `texreg()`	Comment out (unnecessary for data extraction; may fail in batch mode)	Janitor
Over-commenting false positives	Function named `estimate_plot_data` incorrectly flagged as graphics	Match complete function call patterns, not substrings	Janitor

Class 9: Data acquisition variability. Replication materials are hosted on multiple platforms with different API formats, URL conventions, and availability guarantees. Supporting these platforms required explicit handling of retrieval formats and error cases. The following entries document the acquisition-related issues encountered.

Pattern	Manifestation	Resolution	Agent
Multiple hosting platforms	Harvard Dataverse, GitHub, OSF, journal websites	Multi-platform API support	Librarian
Wrong dataset retrieval	Keyword search returns unrelated datasets with similar titles	PDF-first architecture: prioritize URLs from paper over search	Librarian
API format errors	Trailing slash in Dataverse API URL causes 404	Correct API URL formatting	Librarian
Deprecated R packages	`rgdal`, `rgeos`, `maptools` retired from CRAN in 2023	Comment out `library()` calls for known deprecated packages	Janitor
Incompatible R packages	`ri` package incompatible with R 4.x	Flagged as unfixable	—
Expired repository URLs	Repository taken offline or never publicly released	Multi-tier retrieval with manual-supply fallback	Librarian

Class 10: Code injection logic variability. To extract analysis datasets, the Janitor inserts export commands into the original scripts. The correct insertion depends on delimiter modes, multi-line commands, and surrounding control flow. The entries below summarize

the injection-related patterns resolved during development.

Pattern	Manifestation	Resolution	Agent
Injection in `#delimit ;` regions	Injected code uses newline terminators; surrounding code uses semicolons	Detect active delimiter mode; wrap injection with `#delimit cr`/`#delimit ;`	Janitor
Multi-line command truncation	Export block inserted between lines of a multi-line command	Scan forward to complete command end before inserting	Janitor
Line-number drift	Insertions shift all subsequent line numbers; index-based tracking breaks	Content-based detection (backward scan from marker) instead of index tracking	Janitor
Full-panel vs. `e(sample)` export	Some specifications need full panel for lag computation; others need only the estimation sample	Detect `esample_mode`; choose export strategy accordingly	Janitor
`parmest` block handling	Code between `parmest` and `restore` is interdependent	Comment out entire block as a unit	Janitor

E. Example of Diagnose Report

Diagnostics Report: Small Aggregates, Big Manipulation: Vote Buying Enforcement and Collective Monitoring

Miguel R. Rueda — *American Journal Of Political Science* — (2017)

Generated: 2026-03-25 17:52:01

Contents

This report evaluates the robustness of causal identification strategies used in this study, including: Instrumental Variables (IV).

1 Part 1: Reproducibility Assessment

Replication-first check: We execute the authors' replication code and compare every coefficient it produces against every coefficient reported in the paper. Matching uses **precision-aware rounding**: each code coefficient is rounded to the same number of decimal places as the paper-reported value, then checked for exact equality. A coefficient is marked pass when the rounded values agree. Optimal one-to-one assignment across all paper–code pairs maximizes total match quality.

PASS **Overall Verdict:** LARGELY_REPRODUCIBLE (88.6% matched).

Summary: 31 of 35 main-text coefficients matched.

DATA NOTE 4 model(s) (TABLE 6 ((5)), Table 4 ((1)), Table 4 ((2)), Table 4 ((3))) depend on LAPOP survey microdata (`final_LAPOP.dta`), which are subject to third-party access restrictions. Their exclusion from the public replication package is consistent with standard practice for restricted-use survey data. These models cannot be verified from the publicly available materials.

Table 2 ✓ 3/3 models matched

	(1)	(2)	(3)
Polling place size	-1.929 ✓	-1.544 ✓	-1.116 ✓
	(0.363)	(0.232)	(0.251)
Armed group			-0.325 ✓
			(0.123)
Electorate size			-0.37 ✓
			(0.029)

TABLE 3 ✓ 6/6 models matched

	(1)	(2)	(3)	(4)	(5)	(6)
Polling place size	-1.627 ✓	-0.93 ✓	0.963 ✓	-4.445 ✓	-0.448 ✓	-1.383 ✓
	(0.496)	(0.437)	(1.524)	(1.573)	(0.184)	(0.79)

Table 4 × 0/3 models matched

	(1)	(2)	(3)
Polling place size	-0.349 ×	-0.364 ×	-3.123 ×

Table 5 ✓ 8/8 models matched

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
Polling place size	-0.984 ✓	-0.709 ✓	-1.46 ✓	-2.965 ✓	-0.699 ✓	-0.447 ✓	-2.242 ✓	-5.644 ✓
	(0.142)	(0.11)	(0.463)	(1.239)	(0.235)	(0.124)	(1.294)	(3.593)
Ruled-based size	0.796 ✓	0.798 ✓	0.656 ✓	0.581 ✓	0.784 ✓	0.781 ✓	0.853 ✓	0.725 ✓
	(0.009)	(0.009)	(0.023)	(0.044)	(0.018)	(0.018)	(0.06)	(0.094)

TABLE 6 ∼ 4/5 models matched

	(1)	(2)	(3)	(4)	(5)
Polling place size	-0.892 ✓	-0.985 ✓	-2.59 ✓	-1.117 ✓	-0.324 ×
	(0.384)	(0.752)	(6.29)	(0.907)	

Execution Summary: Scripts run = 25, scripts failed = 0.

2 Executive Summary

Spec	Method	Outcome (Y)	Treatment (D)	Instrument (Z)	Eff. F
Table 5 (1)	IV	e_vote_buying	lm_pob_mesa	lz_pob_mesa_f	8598.3
Table 5 (2)	IV	e_vote_buying	lm_pob_mesa	lz_pob_mesa_f	8310.0
Table 5 (3)	IV	e_vote_buying	lm_pob_mesa	lz_pob_mesa_f	817.3
Table 5 (4)	IV	e_vote_buying	lm_pob_mesa	lz_pob_mesa_f	169.8
Table 5 (5)	IV	sum_vb	lm_pob_mesa	lz_pob_mesa_f	1954.9
Table 5 (6)	IV	sum_vb	lm_pob_mesa	lz_pob_mesa_f	1907.6
Table 5 (7)	IV	sum_vb	lm_pob_mesa	lz_pob_mesa_f	203.9
Table 5 (8)	IV	sum_vb	lm_pob_mesa	lz_pob_mesa_f	60.0

Primary Specification (Table 5 (1)): PASS No diagnostic indicators flagged.
Main Finding: The 2SLS estimate is **-0.9835** ($p < 0.001$), which is **statistically significant** at the 5% level.

Table 5 (2) (e_vote_buying): PASS No diagnostic indicators flagged.

Table 5 (3) (e_vote_buying): PASS No diagnostic indicators flagged.

Table 5 (4) (e_vote_buying):
DIAGNOSTIC FLAG Removing one cluster changes estimate by 30.4% (influential unit: 8001)

Table 5 (5) (sum_vb):
DIAGNOSTIC FLAG Removing one cluster changes estimate by 29.7% (influential unit: 11001)

Table 5 (6) (sum_vb): PASS No diagnostic indicators flagged.

Table 5 (7) (sum_vb):
DIAGNOSTIC FLAG Effect not significant in Anderson-Rubin test (p = 0.075) DIAGNOSTIC FLAG Removing one cluster changes estimate by 58.1% (influential unit: 11001)

Table 5 (8) (sum_vb):
DIAGNOSTIC FLAG Effect not significant in Anderson-Rubin test (p = 0.115) DIAGNOSTIC FLAG Removing one cluster changes estimate by 65.2% (influential unit: 11001)

3 Instrumental Variables Analysis

3.1 Study Design

This study uses **instrumental variable (IV) analysis** to answer a causal question. Here's how to understand the key components:

Paper: *Small Aggregates, Big Manipulation: Vote Buying Enforcement and Collective Monitoring*

3.1.1 Variables

Role	Variable	Description
Outcome 1	`e_vote_buying`	The incidence of voters receiving material benefits in exchange for their votes. (4 specs)
Outcome 2	`sum_vb`	The total number of reported vote buying incidents across various polling stations. (4 specs)
Treatment (D)	`lm_pob_mesa`	The size of the population at polling stations, influencing vote buying dynamics.
Instrument (Z)	`lz_pob_mesa_f`	The published electoral results for small voting groups, facilitating vote buying enforcement.
Cluster	`muni_code`	Clustering unit for standard errors

Controls: 2 variables (`lpopulation`, `lpotencial`)

3.2 Specification Verification

Three-level verification chain: We verify each specification through three stages:

1. **Paper → Code:** Can the authors' code reproduce the paper's coefficients?
2. **Code → R:** Does our R re-estimation match the code output?
3. **Chain status:** Only fully-verified specifications receive full diagnostic analysis.

3.2.1 Step 1: Paper → Code

PASS 8/8 paper coefficients matched to code output.

3.2.2 Verification Chain Summary

Spec	Outcome	Paper	Code	R (2SLS)	Paper↔Code	Code↔R	Chain
Table 5 (1)	`e_vote_buying`	-0.9840	-0.9835	-0.9835	**PASS**	**Excellent**	**PASS** Verified
Table 5 (2)	`e_vote_buying`	-0.7090	-0.7092	-0.7092	**PASS**	**Excellent**	**PASS** Verified
Table 5 (3)	`e_vote_buying`	-1.4600	-1.4600	-1.4594	**PASS**	**Excellent**	**PASS** Verified
Table 5 (4)	`e_vote_buying`	-2.9650	-2.9650	-2.9886	**PASS**	**Excellent**	**PASS** Verified
Table 5 (5)	`sum_vb`	-0.6990	-0.6992	-0.6992	**PASS**	**Excellent**	**PASS** Verified
Table 5 (6)	`sum_vb`	-0.4470	-0.4471	-0.4471	**PASS**	**Excellent**	**PASS** Verified
Table 5 (7)	`sum_vb`	-2.2420	-2.2420	-2.2420	**PASS**	**Excellent**	**PASS** Verified
Table 5 (8)	`sum_vb`	-5.6440	-5.6436	-5.6436	**PASS**	**Excellent**	**PASS** Verified

⋆ = primary specification

Reading this table: **Paper** = coefficient from published PDF. **Code** = coefficient from executing the authors' replication code. **R (2SLS)** = coefficient from our independent R re-estimation. **Chain** = **PASS** Verified when both Paper↔Code and Code↔R agree, confirming the diagnostics below analyze the correct specification.

3.3 Table 5 (1): e_vote_buying

Outcome (Y): `e_vote_buying` | **Treatment (D):** `lm_pob_mesa` | **Instrument (Z):** `lz_pob_mesa_f`

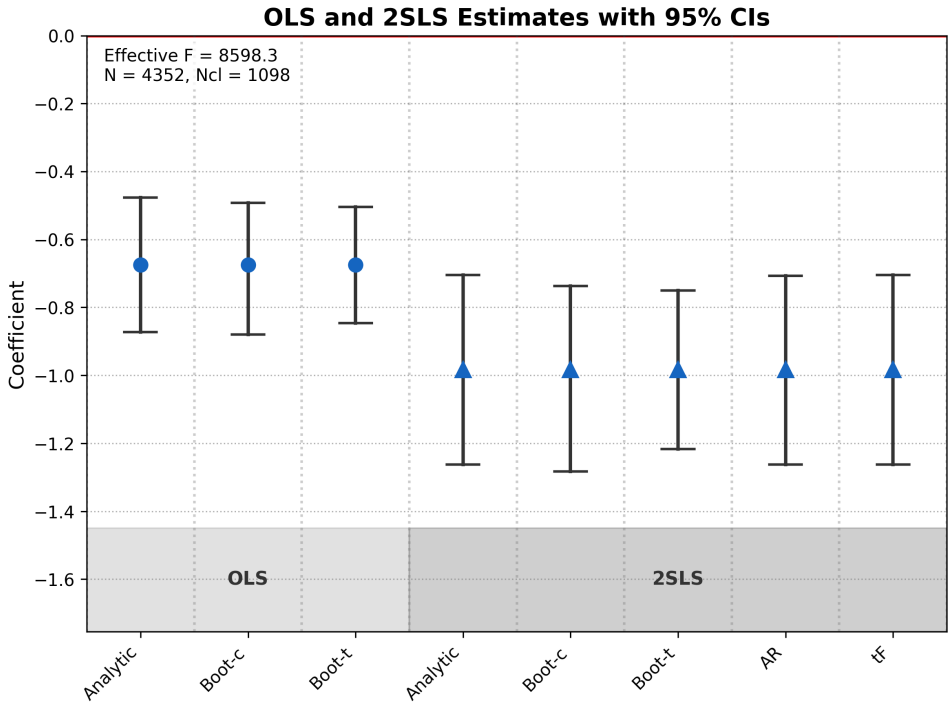


Figure 1: OLS and 2SLS estimates with 95% CIs for e_vote_buying

3.3.1 Instrument Strength

For IV to work, the instrument must **strongly predict** the treatment. A "weak instrument" leads to unreliable estimates.

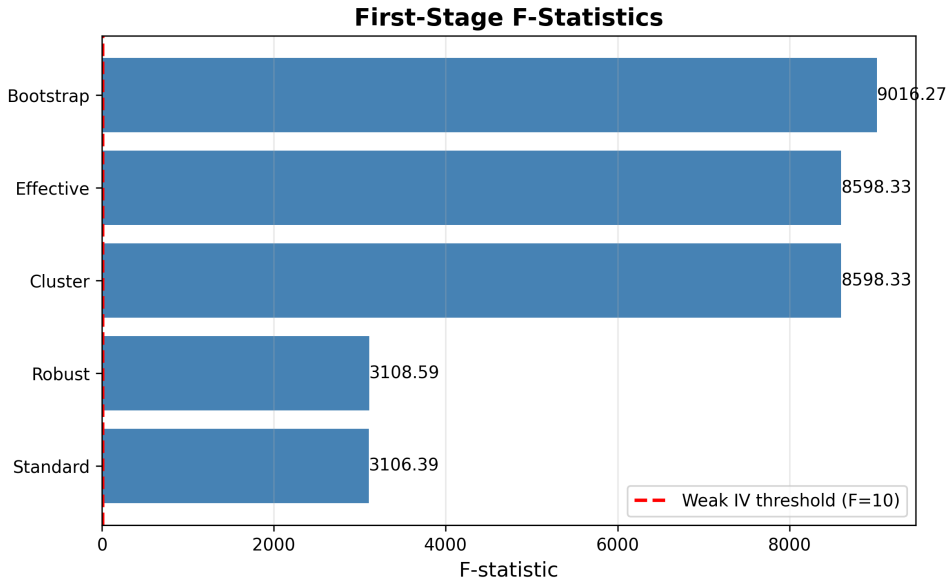


Figure 2: First-stage F-statistics for e_vote_buying

Statistic	Value	Assessment
F-effective	8598.33	**PASS** Strong
F-standard	3106.39	—
F-cluster	8598.33	Cluster-robust
F-robust	3108.59	HC-robust
F-bootstrap	9016.27	Bootstrap-robust

First-Stage Parameter	Value
First-stage coef ($\hat{\pi}$)	0.7957
First-stage SE	0.0086
First-stage ρ (correlation coefficient)	0.6455

PASS **The instrument is strong** (F = 8598.33). Standard IV inference should be reliable.

3.3.2 Robust Inference

Here, we gauge the **uncertainties** of the 2SLS estimates.

Statistic	Value
Coefficient	-0.9835
Standard Error	0.1424
p-value	< 0.001
95% CI	[-1.2626, -0.7044]
N	4352
N clusters	1098

A one-unit increase in treatment is associated with a 0.9835 decrease in the outcome (p < 0.001). **Statistically significant at 5%.**

Anderson-Rubin Test (weak-IV robust): p < 0.001 → **PASS** **Significant**
AR 95% CI: [-1.2626, -0.7073] (bounded)
tF Procedure (Lee et al. 2022): |t| = 6.91 vs critical t = 1.96 → **PASS** **Significant**
tF 95% CI: [-1.2626, -0.7044]

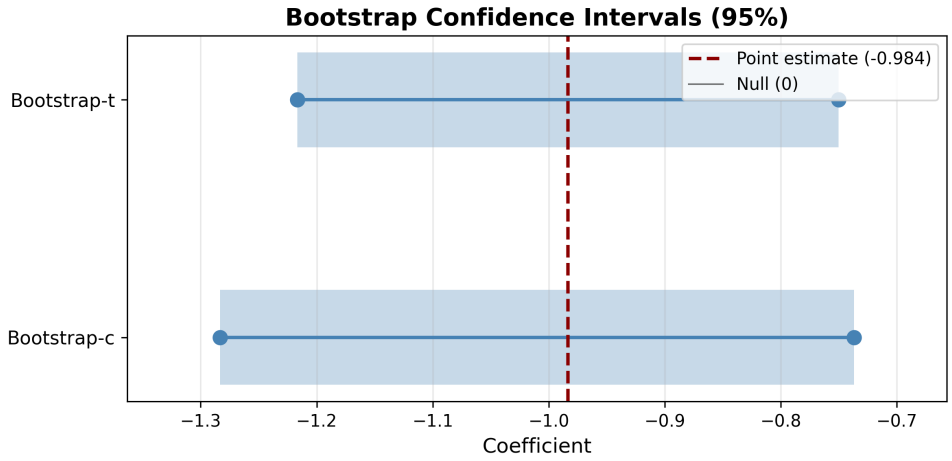


Figure 3: Bootstrap confidence intervals for e_vote_buying

Method	95% CI	Includes Zero?
Bootstrap-c	[-1.2834, -0.7370]	No
Bootstrap-t	[-1.2167, -0.7503]	No

PASS Bootstrap CI **excludes zero** — effect is significant.

3.3.3 Sensitivity Analysis

The **jackknife** method removes each observation/cluster one at a time and re-estimates the effect. Stable results = robust findings.

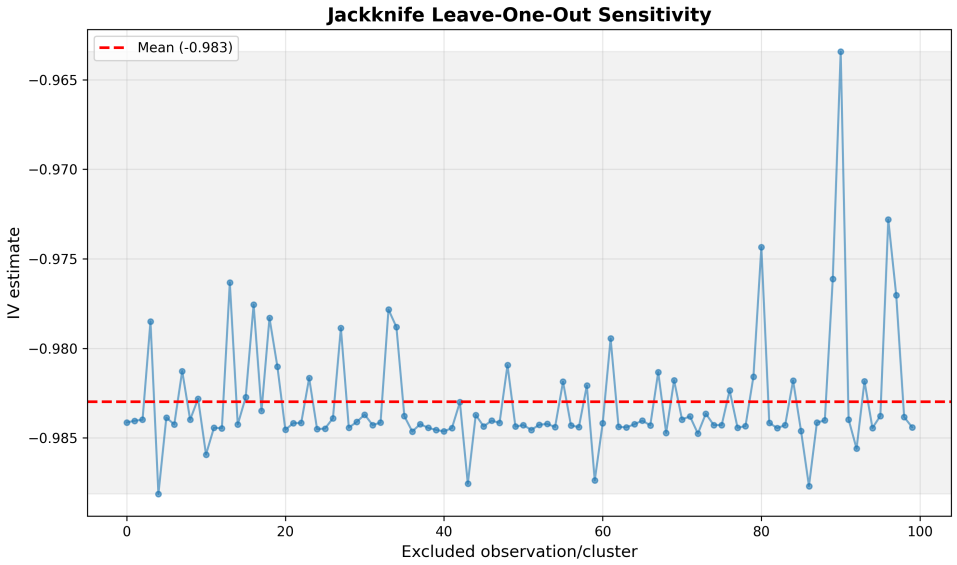


Figure 4: Jackknife leave-one-out sensitivity for e_vote_buying

Statistic	Value
Mean estimate	-0.9830
Range	[-0.9881, -0.9634]
Std. deviation	0.0033
Most influential unit	11001 (Δ = 0.0201)

PASS **Robust** — only 2.5% variation across leave-one-out samples.

3.3.4 IV vs. OLS Comparison

Comparing 2SLS estimate to the naive OLS estimate.

Method	Coefficient	Ratio
OLS	-0.6750	—
2SLS	**-0.9835**	**1.5x**

PASS The 2SLS estimate and the naive OLS estimate are **similar** (ratio = 1.46) — little evidence of bias.

3.4 Table 5 (2): e_vote_buying

Outcome (Y): e_vote_buying | **Treatment (D):** lm_pob_mesa | **Instrument (Z):** lz_pob_mesa_f

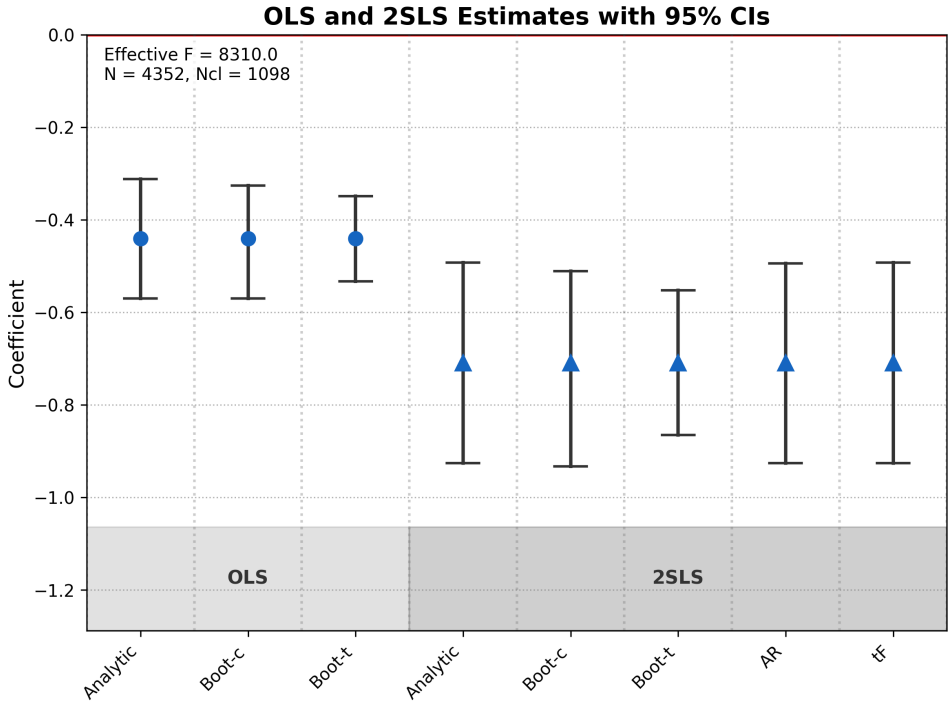


Figure 5: OLS and 2SLS estimates with 95% CIs for e_vote_buying

3.4.1 Instrument Strength

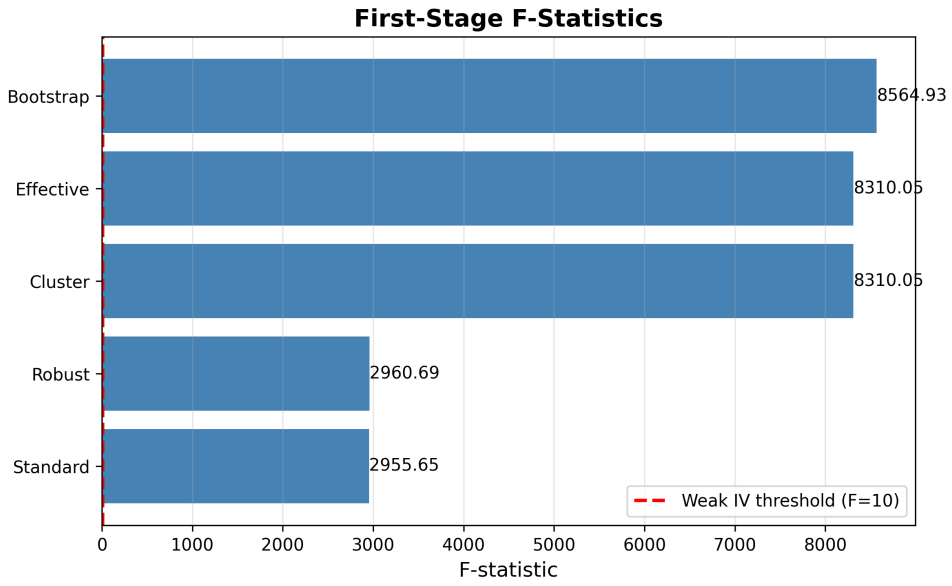


Figure 6: First-stage F-statistics for e_vote_buying

Statistic	Value	Assessment
F-effective	8310.05	**PASS** Strong
F-standard	2955.65	—
F-cluster	8310.05	Cluster-robust
F-robust	2960.69	HC-robust
F-bootstrap	8564.93	Bootstrap-robust

First-Stage Parameter	Value
First-stage coef ($\hat{\pi}$)	0.7978
First-stage SE	0.0088
First-stage ρ (correlation coefficient)	0.6362

PASS **The instrument is strong** (F = 8310.05). Standard IV inference should be reliable.

3.4.2 Robust Inference

Statistic	Value
Coefficient	-0.7092
Standard Error	0.1106
p-value	< 0.001
95% CI	[-0.9259, -0.4925]
N	4352
N clusters	1098

A one-unit increase in treatment is associated with a 0.7092 decrease in the outcome (p < 0.001). **Statistically significant at 5%.**

Anderson-Rubin Test (weak-IV robust): p < 0.001 → **PASS** **Significant**
AR 95% CI: [-0.9259, -0.4947] (bounded)
tF Procedure (Lee et al. 2022): |t| = 6.41 vs critical t = 1.96 → **PASS** **Significant**
tF 95% CI: [-0.9259, -0.4925]

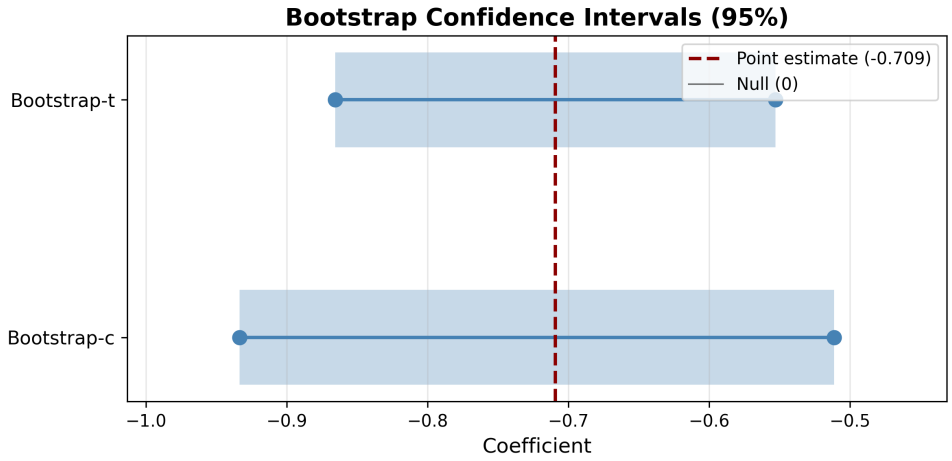


Figure 7: Bootstrap confidence intervals for e_vote_buying

Method	95% CI	Includes Zero?
Bootstrap-c	[-0.9335, -0.5111]	No
Bootstrap-t	[-0.8656, -0.5527]	No

PASS Bootstrap CI **excludes zero** — effect is significant.

3.4.3 Sensitivity Analysis

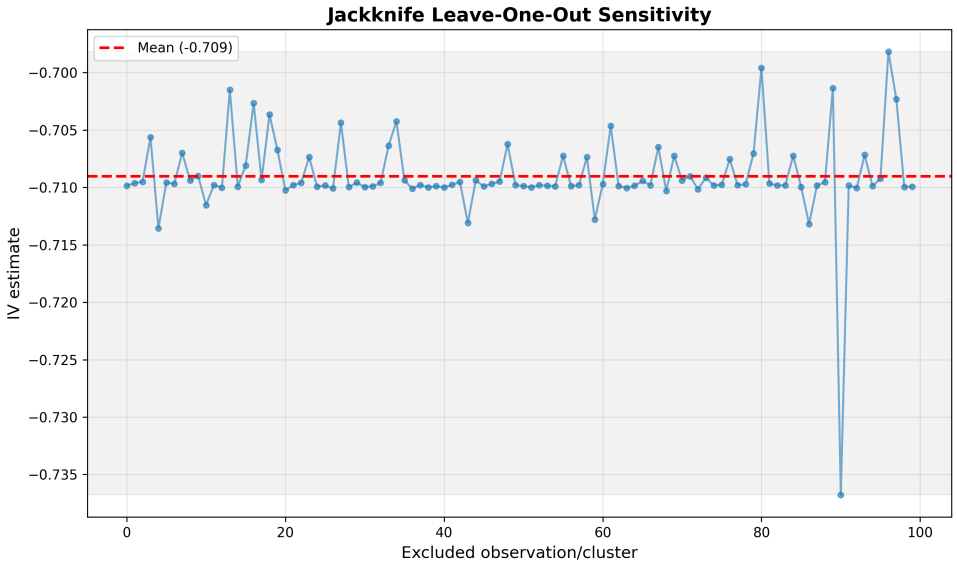


Figure 8: Jackknife leave-one-out sensitivity for e_vote_buying

Statistic	Value
Mean estimate	-0.7090
Range	[-0.7368, -0.6982]
Std. deviation	0.0038
Most influential unit	11001 (Δ = 0.0276)

PASS **Robust** — only 5.4% variation across leave-one-out samples.

3.4.4 IV vs. OLS Comparison

Method	Coefficient	Ratio
OLS	-0.4412	—
2SLS	**-0.7092**	**1.6x**

PASS The 2SLS estimate and the naive OLS estimate are **similar** (ratio = 1.61) — little evidence of bias.

3.5 Table 5 (3): e_vote_buying

Outcome (Y): `e_vote_buying` | **Treatment (D):** `lm_pob_mesa` | **Instrument (Z):** `lz_pob_mesa_f`

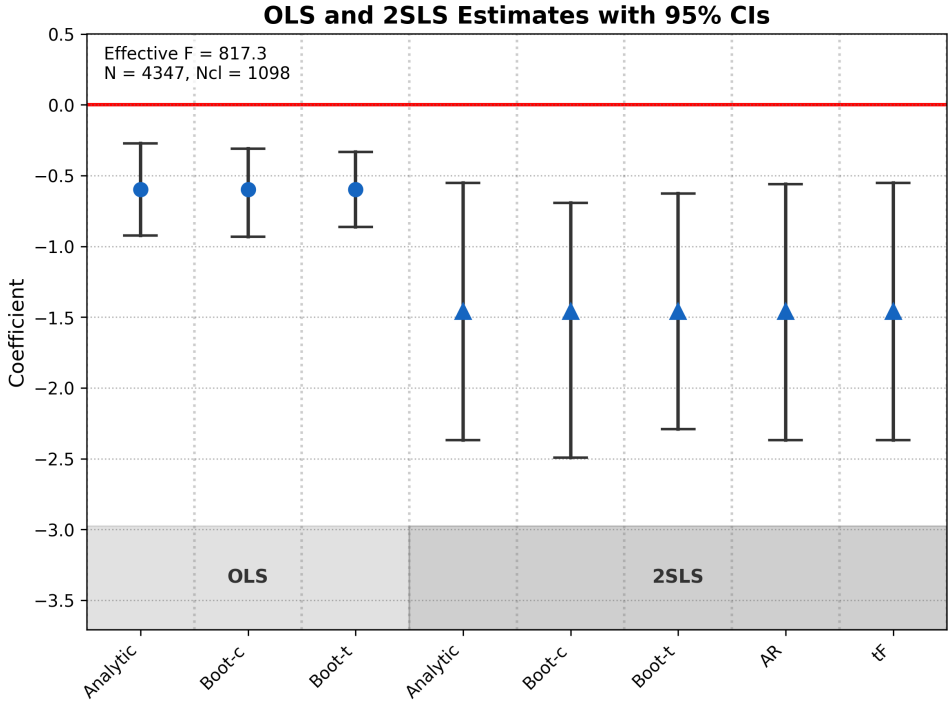


Figure 9: OLS and 2SLS estimates with 95% CIs for e_vote_buying

3.5.1 Instrument Strength

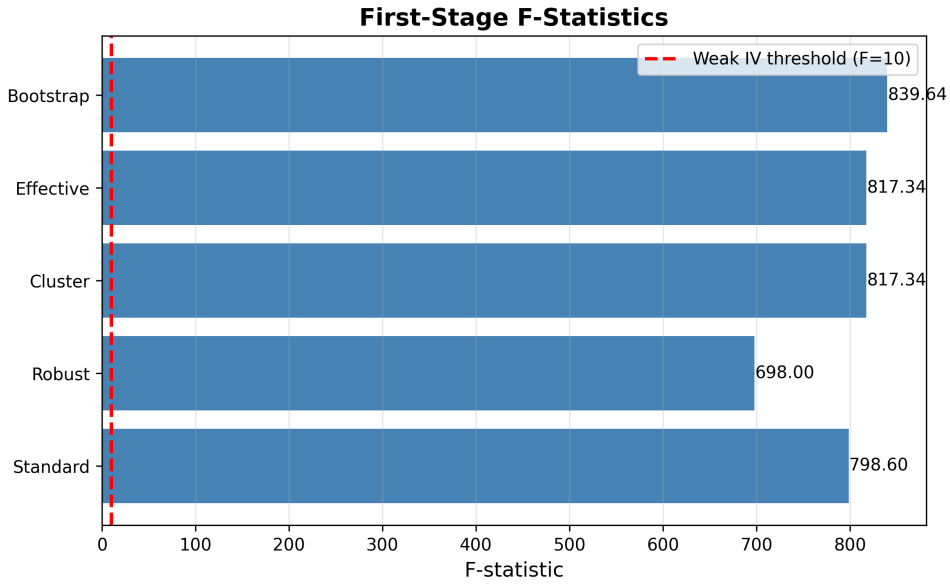


Figure 10: First-stage F-statistics for e_vote_buying

Statistic	Value	Assessment
F-effective	817.34	**PASS** Strong
F-standard	798.60	—
F-cluster	817.34	Cluster-robust
F-robust	698.00	HC-robust
F-bootstrap	839.64	Bootstrap-robust

First-Stage Parameter	Value
First-stage coef ($\hat{\pi}$)	0.6572
First-stage SE	0.0230
First-stage ρ (correlation coefficient)	0.3943

PASS **The instrument is strong** (F = 817.34). Standard IV inference should be reliable.

3.5.2 Robust Inference

Statistic	Value
Coefficient	-1.4594
Standard Error	0.4633
p-value	0.0016
95% CI	[-2.3676, -0.5513]
N	4347
N clusters	1098

> A one-unit increase in treatment is associated with a 1.4594 decrease in the outcome (p = 0.0016). **Statistically significant at 5%.**

Anderson-Rubin Test (weak-IV robust): p = 0.0016 → **PASS** **Significant**
AR 95% CI: [-2.3676, -0.5606] (bounded)
tF Procedure (Lee et al. 2022): |t| = 3.15 vs critical t = 1.96 → **PASS** **Significant**
tF 95% CI: [-2.3676, -0.5513]

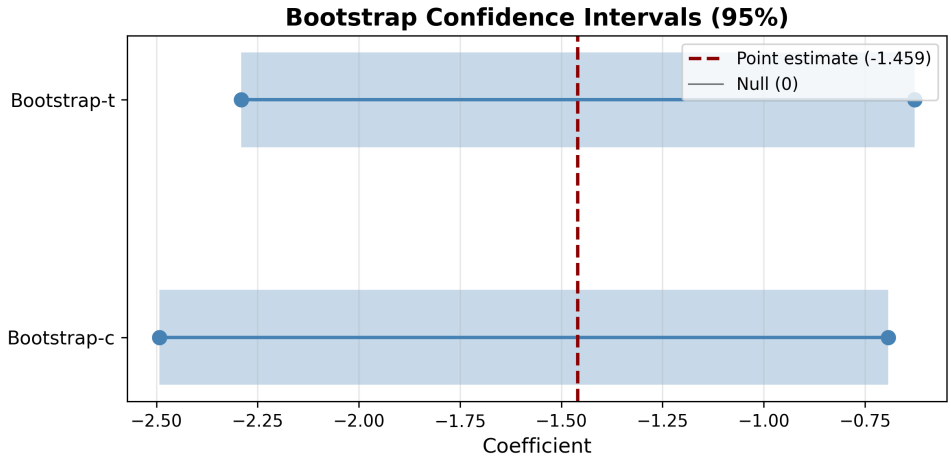


Figure 11: Bootstrap confidence intervals for e_vote_buying

Method	95% CI	Includes Zero?
Bootstrap-c	[-2.4924, -0.6930]	No
Bootstrap-t	[-2.2910, -0.6279]	No

PASS Bootstrap CI **excludes zero** — effect is significant.

3.5.3 Sensitivity Analysis

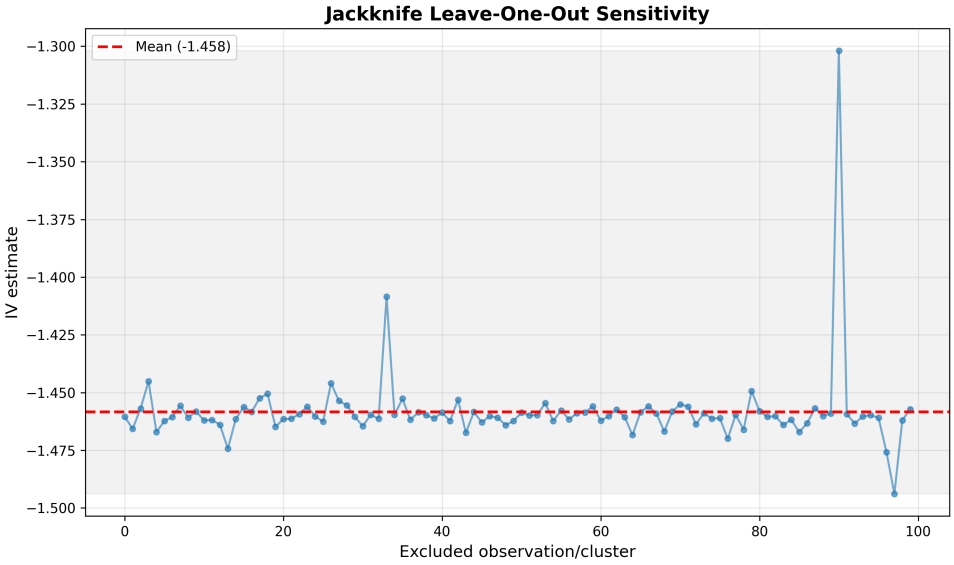


Figure 12: Jackknife leave-one-out sensitivity for e_vote_buying

Statistic	Value
Mean estimate	-1.4584
Range	[-1.4939, -1.3019]
Std. deviation	0.0176
Most influential unit	11001 (Δ = 0.1575)

PASS **Robust** — only 13.2% variation across leave-one-out samples.

3.5.4 IV vs. OLS Comparison

Method	Coefficient	Ratio
OLS	-0.5981	—
2SLS	**-1.4594**	**2.4x**

The 2SLS estimate is 2.4x larger than the naive OLS estimate, suggesting moderate endogeneity correction.

3.6 Table 5 (4): e_vote_buying

Outcome (Y): `e_vote_buying` | **Treatment (D):** `lm_pob_mesa` | **Instrument (Z):** `lz_pob_mesa_f`

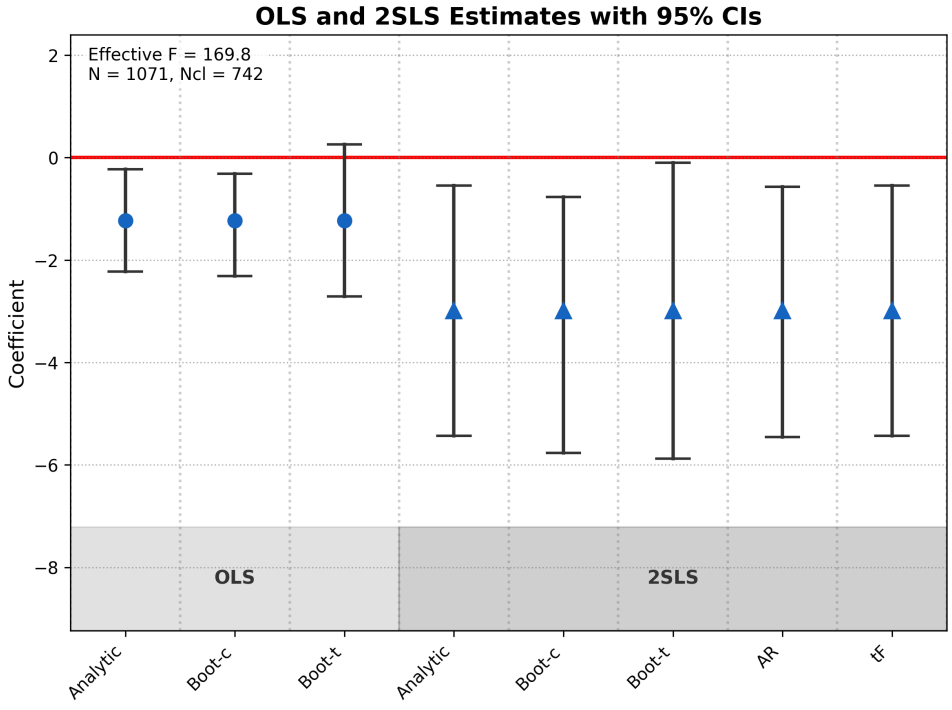


Figure 13: OLS and 2SLS estimates with 95% CIs for e_vote_buying

3.6.1 Instrument Strength

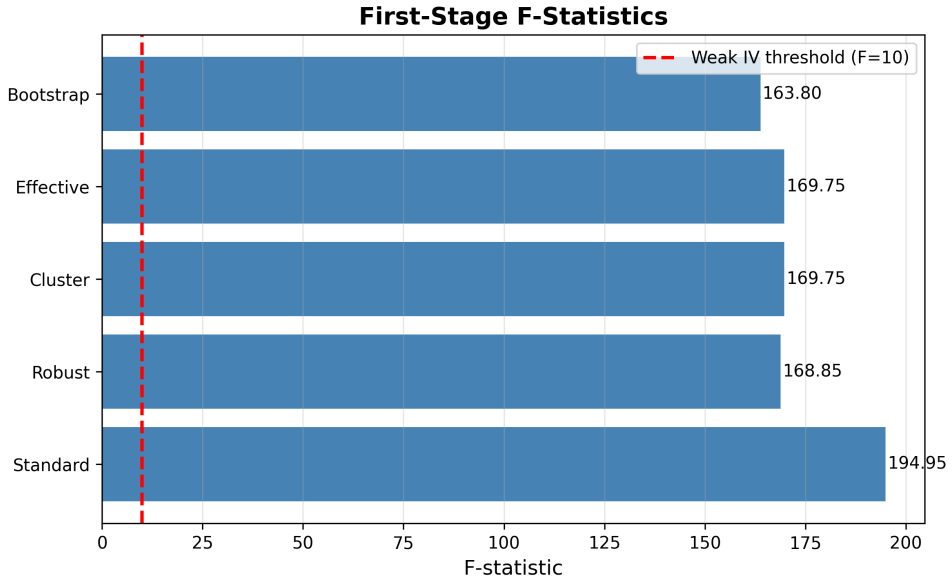


Figure 14: First-stage F-statistics for e_vote_buying

Statistic	Value	Assessment
F-effective	169.75	**PASS** Strong
F-standard	194.95	—
F-cluster	169.75	Cluster-robust
F-robust	168.85	HC-robust
F-bootstrap	163.80	Bootstrap-robust

First-Stage Parameter	Value
First-stage coef ($\hat{\pi}$)	0.5823
First-stage SE	0.0447
First-stage ρ (correlation coefficient)	0.3938

PASS **The instrument is strong** (F = 169.75). Standard IV inference should be reliable.

3.6.2 Robust Inference

Statistic	Value
Coefficient	-2.9886
Standard Error	1.2469
p-value	0.0165
95% CI	[-5.4326, -0.5446]
N	1071
N clusters	742

> A one-unit increase in treatment is associated with a 2.9886 decrease in the outcome (p = 0.0165). **Statistically significant at 5%.**

Anderson-Rubin Test (weak-IV robust): p = 0.0168 → **PASS** **Significant**
AR 95% CI: [-5.4575, -0.5696] (bounded)
tF Procedure (Lee et al. 2022): |t| = 2.40 vs critical t = 1.96 → **PASS** **Significant**
tF 95% CI: [-5.4326, -0.5446]

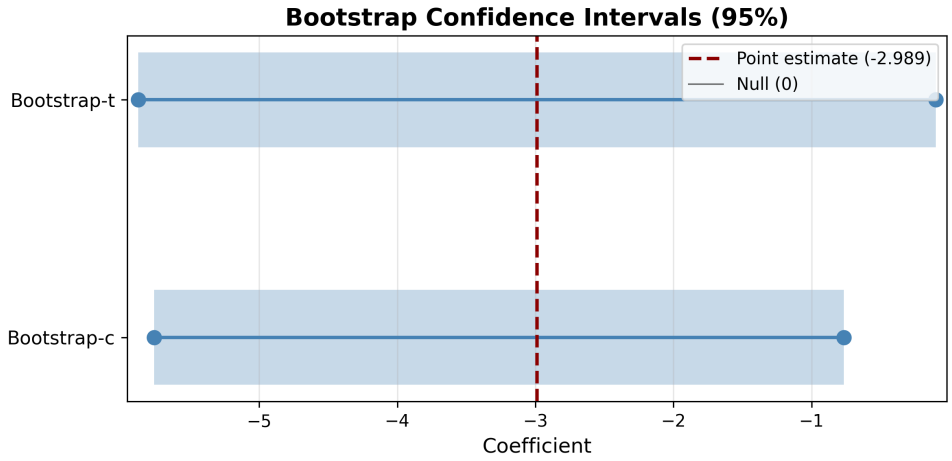


Figure 15: Bootstrap confidence intervals for e_vote_buying

Method	95% CI	Includes Zero?
Bootstrap-c	[-5.7622, -0.7659]	No
Bootstrap-t	[-5.8769, -0.1002]	No

PASS Bootstrap CI **excludes zero** — effect is significant.

3.6.3 Sensitivity Analysis

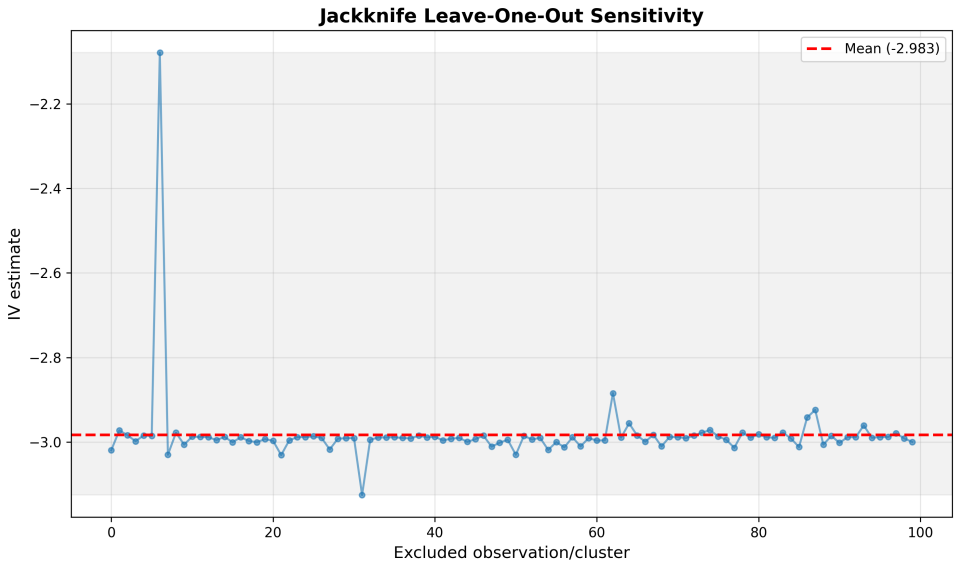


Figure 16: Jackknife leave-one-out sensitivity for e_vote_buying

Statistic	Value
Mean estimate	-2.9828
Range	[-3.1252, -2.0791]
Std. deviation	0.0941
Most influential unit	8001 (Δ = 0.9095)

NOTE Moderately sensitive — 35.0% variation across leave-one-out samples.

3.6.4 IV vs. OLS Comparison

Method	Coefficient	Ratio
OLS	-1.2275	—
2SLS	**-2.9886**	**2.4x**

The 2SLS estimate is 2.4x larger than the naive OLS estimate, suggesting moderate endogeneity correction.

3.7 Table 5 (5): sum_vb

Outcome (Y): `sum_vb` | **Treatment (D):** `lm_pob_mesa` | **Instrument (Z):** `lz_pob_mesa_f`

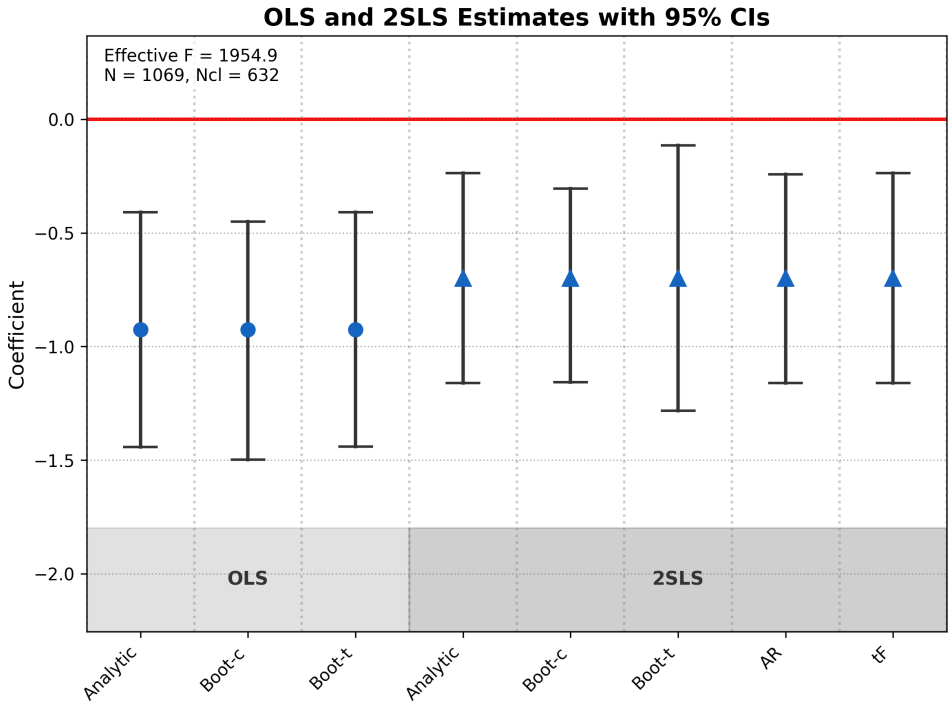


Figure 17: OLS and 2SLS estimates with 95% CIs for sum_vb

3.7.1 Instrument Strength

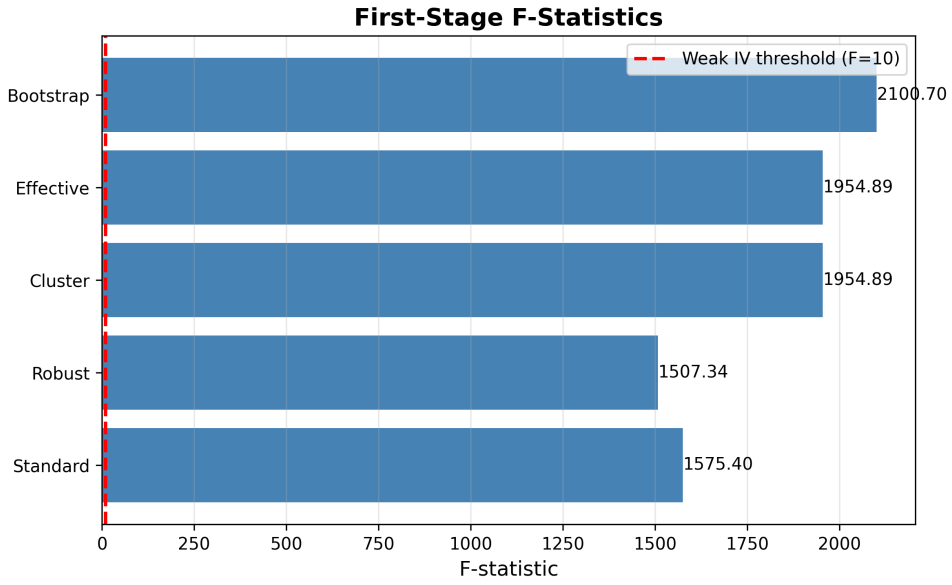


Figure 18: First-stage F-statistics for sum_vb

Statistic	Value	Assessment
F-effective	1954.89	**PASS** Strong
F-standard	1575.40	—
F-cluster	1954.89	Cluster-robust
F-robust	1507.34	HC-robust
F-bootstrap	2100.70	Bootstrap-robust

First-Stage Parameter	Value
First-stage coef ($\hat{\pi}$)	0.7842
First-stage SE	0.0177
First-stage ρ (correlation coefficient)	0.7724

PASS **The instrument is strong** (F = 1954.89). Standard IV inference should be reliable.

3.7.2 Robust Inference

Statistic	Value
Coefficient	-0.6992
Standard Error	0.2354
p-value	0.0030
95% CI	[-1.1605, -0.2378]
N	1069
N clusters	632

> A one-unit increase in treatment is associated with a 0.6992 decrease in the outcome (p = 0.0030). **Statistically significant at 5%.**

Anderson-Rubin Test (weak-IV robust): p = 0.0027 → **PASS** **Significant**
AR 95% CI: [-1.1605, -0.2425] (bounded)
tF Procedure (Lee et al. 2022): |t| = 2.97 vs critical t = 1.96 → **PASS** **Significant**
tF 95% CI: [-1.1605, -0.2378]

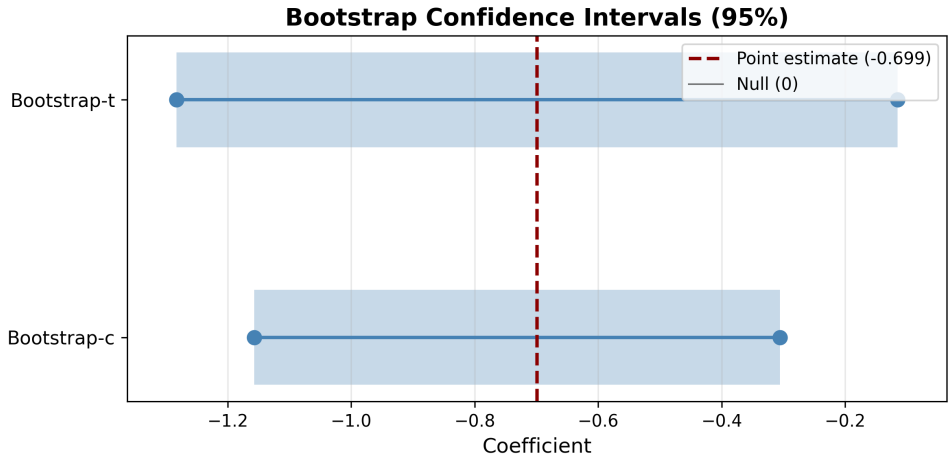


Figure 19: Bootstrap confidence intervals for sum_vb

Method	95% CI	Includes Zero?
Bootstrap-c	[-1.1569, -0.3059]	No
Bootstrap-t	[-1.2829, -0.1154]	No

PASS Bootstrap CI **excludes zero** — effect is significant.

3.7.3 Sensitivity Analysis

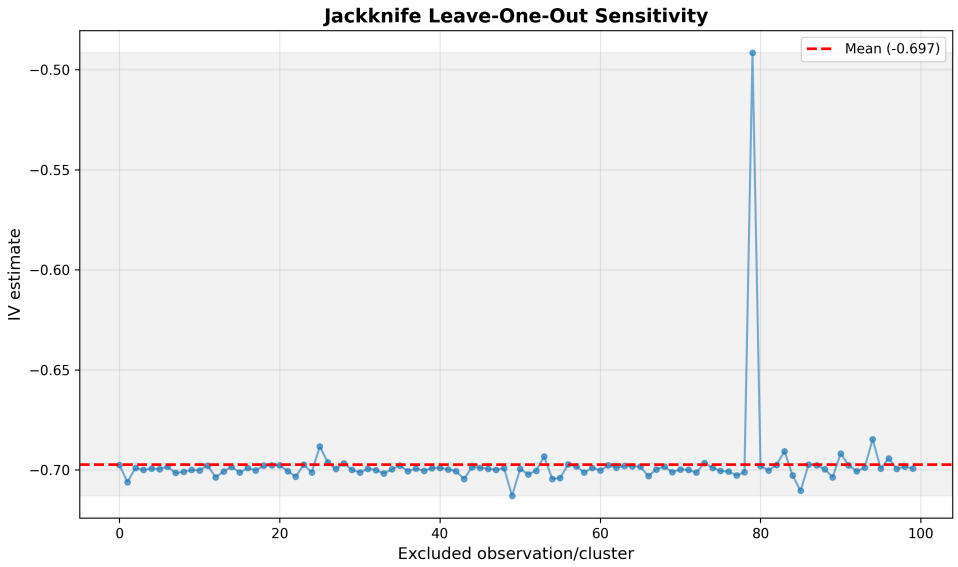


Figure 20: Jackknife leave-one-out sensitivity for sum_vb

Statistic	Value
Mean estimate	-0.6974
Range	[-0.7130, -0.4915]
Std. deviation	0.0211
Most influential unit	11001 (Δ = 0.2076)

NOTE Moderately sensitive — 31.7% variation across leave-one-out samples.

3.7.4 IV vs. OLS Comparison

Method	Coefficient	Ratio
OLS	-0.9253	—
2SLS	**-0.6992**	**0.8x**

PASS The 2SLS estimate and the naive OLS estimate are **similar** (ratio = 0.76) — little evidence of bias.

3.8 Table 5 (6): sum_vb

Outcome (Y): `sum_vb` | **Treatment (D):** `lm_pob_mesa` | **Instrument (Z):** `lz_pob_mesa_f`

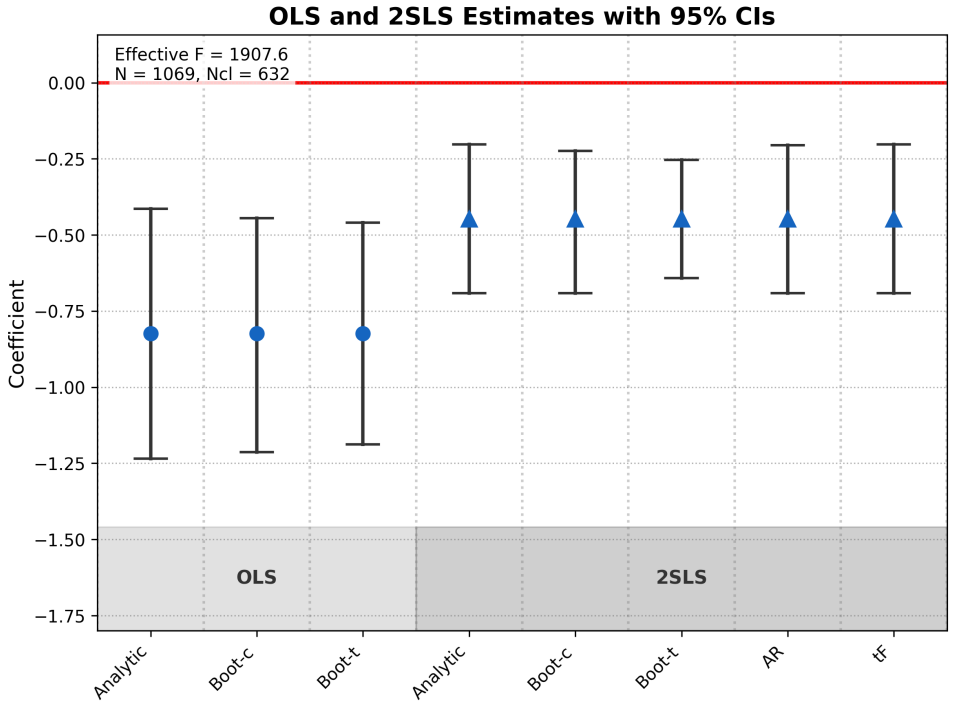


Figure 21: OLS and 2SLS estimates with 95% CIs for sum_vb

3.8.1 Instrument Strength

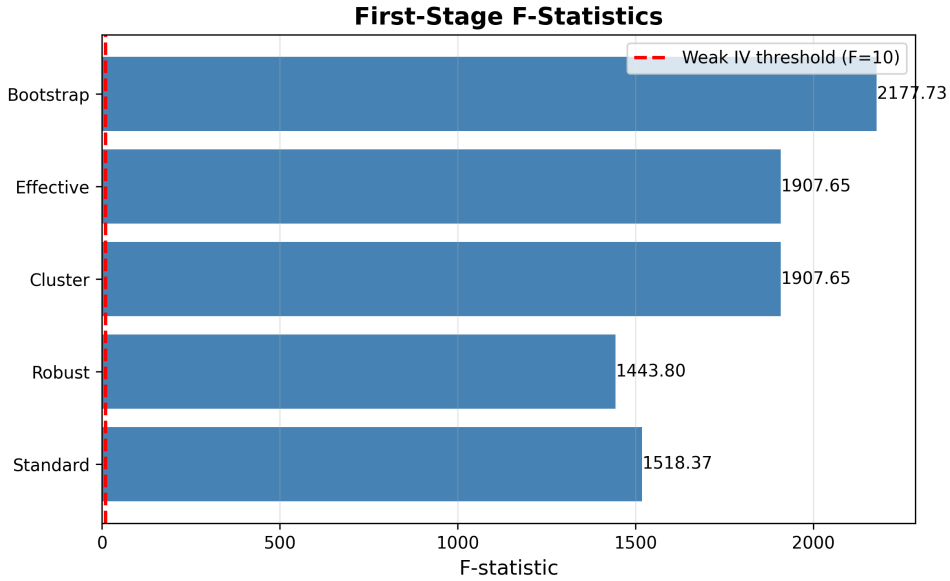


Figure 22: First-stage F-statistics for sum_vb

Statistic	Value	Assessment
F-effective	1907.65	**PASS** Strong
F-standard	1518.37	—
F-cluster	1907.65	Cluster-robust
F-robust	1443.80	HC-robust
F-bootstrap	2177.73	Bootstrap-robust

First-Stage Parameter	Value
First-stage coef ($\hat{\pi}$)	0.7808
First-stage SE	0.0179
First-stage ρ (correlation coefficient)	0.7669

PASS **The instrument is strong** (F = 1907.65). Standard IV inference should be reliable.

3.8.2 Robust Inference

Statistic	Value
Coefficient	-0.4471
Standard Error	0.1245
p-value	< 0.001
95% CI	[-0.6912, -0.2031]
N	1069
N clusters	632

A one-unit increase in treatment is associated with a 0.4471 decrease in the outcome (p < 0.001). **Statistically significant at 5%.**

Anderson-Rubin Test (weak-IV robust): p < 0.001 → **PASS** **Significant**
AR 95% CI: [-0.6912, -0.2056] (bounded)
tF Procedure (Lee et al. 2022): |t| = 3.59 vs critical t = 1.96 → **PASS** **Significant**
tF 95% CI: [-0.6912, -0.2031]

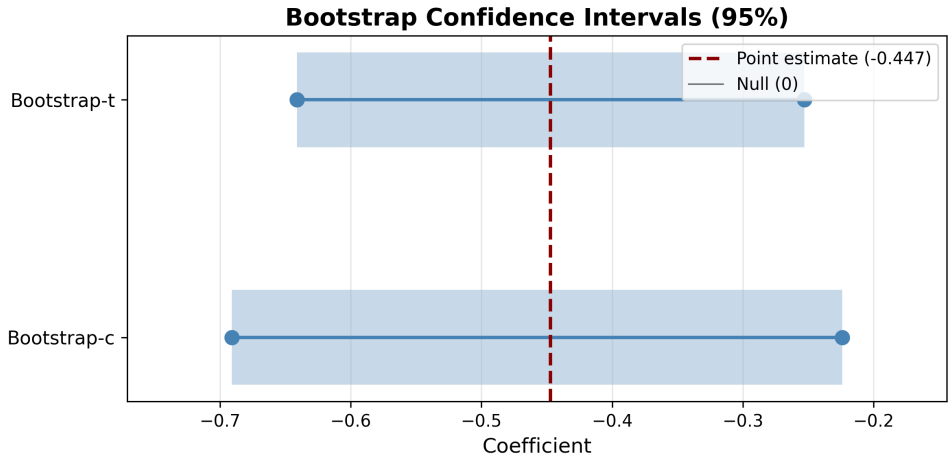


Figure 23: Bootstrap confidence intervals for sum_vb

Method	95% CI	Includes Zero?
Bootstrap-c	[-0.6911, -0.2240]	No
Bootstrap-t	[-0.6412, -0.2531]	No

PASS Bootstrap CI **excludes zero** — effect is significant.

3.8.3 Sensitivity Analysis

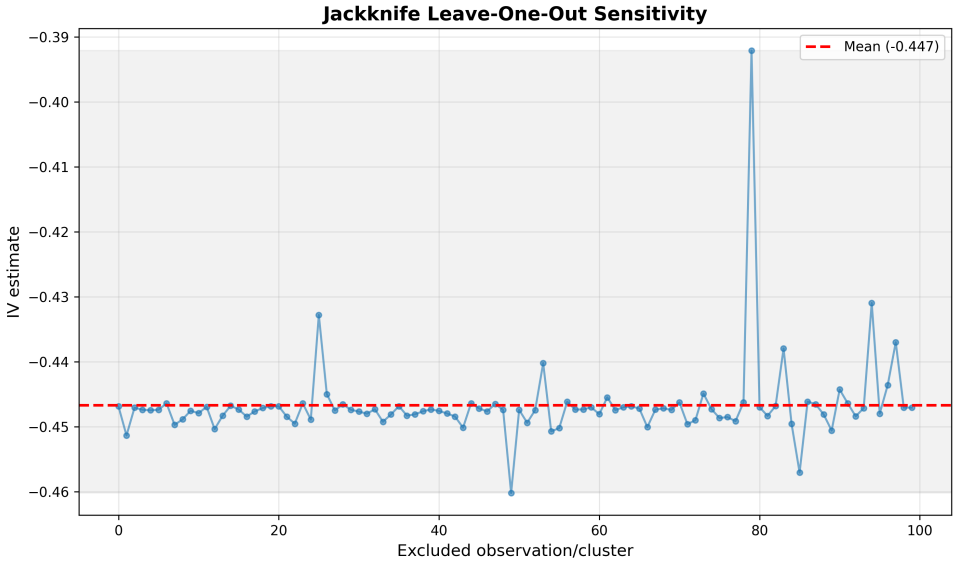


Figure 24: Jackknife leave-one-out sensitivity for sum_vb

Statistic	Value
Mean estimate	-0.4467
Range	[-0.4602, -0.3921]
Std. deviation	0.0065
Most influential unit	11001 (Δ = 0.0550)

PASS **Robust** — only 15.2% variation across leave-one-out samples.

3.8.4 IV vs. OLS Comparison

Method	Coefficient	Ratio
OLS	-0.8242	—
2SLS	**-0.4471**	**0.5x**

PASS The 2SLS estimate and the naive OLS estimate are **similar** (ratio = 0.54) — little evidence of bias.

3.9 Table 5 (7): sum_vb

Outcome (Y): `sum_vb` | **Treatment (D):** `lm_pob_mesa` | **Instrument (Z):** `lz_pob_mesa_f`

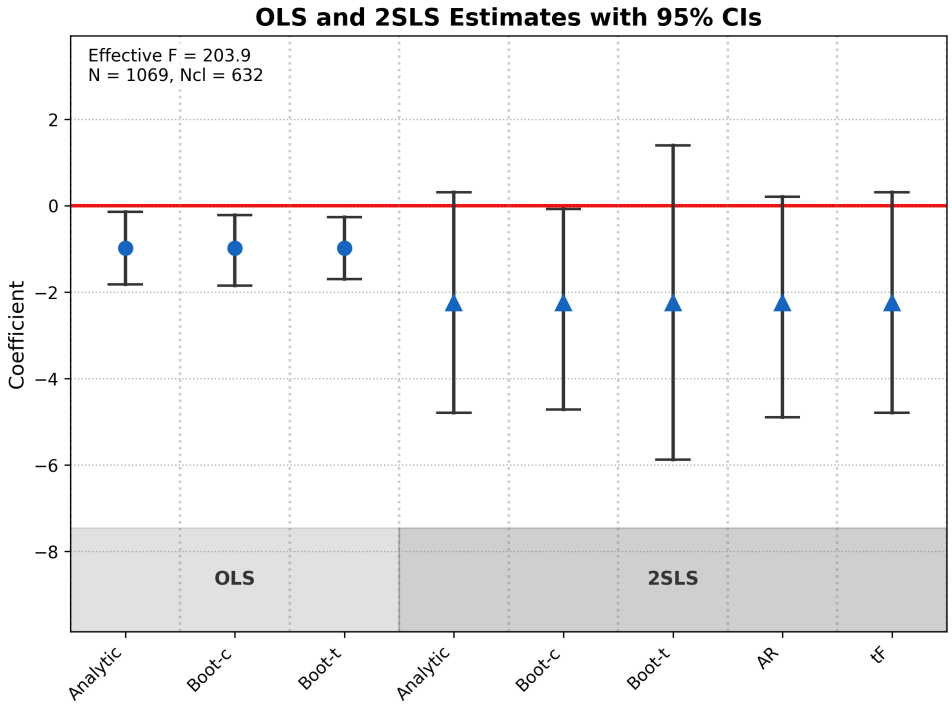


Figure 25: OLS and 2SLS estimates with 95% CIs for sum_vb

3.9.1 Instrument Strength

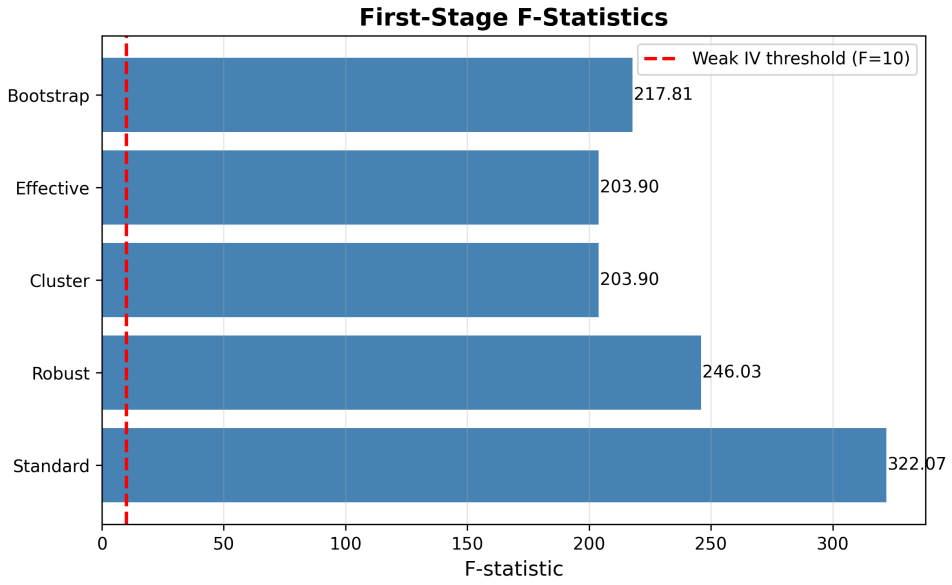


Figure 26: First-stage F-statistics for sum_vb

Statistic	Value	Assessment
F-effective	203.90	**PASS** Strong
F-standard	322.07	—
F-cluster	203.90	Cluster-robust
F-robust	246.03	HC-robust
F-bootstrap	217.81	Bootstrap-robust

First-Stage Parameter	Value
First-stage coef ($\hat{\pi}$)	0.8526
First-stage SE	0.0597
First-stage ρ (correlation coefficient)	0.4827

PASS **The instrument is strong** (F = 203.90). Standard IV inference should be reliable.

3.9.2 Robust Inference

Statistic	Value
Coefficient	-2.2420
Standard Error	1.2998
p-value	0.0845
95% CI	[-4.7895, 0.3055]
N	1069
N clusters	632

> The effect is **not statistically significant (p = 0.0845 > 0.05)**.

Anderson-Rubin Test (weak-IV robust): p = 0.0747 → **NOTE** Not significant
AR 95% CI: [-4.8935, 0.2015] (bounded)
tF Procedure (Lee et al. 2022): |t| = 1.73 vs critical t = 1.96 → **NOTE** Not significant
tF 95% CI: [-4.7895, 0.3055]

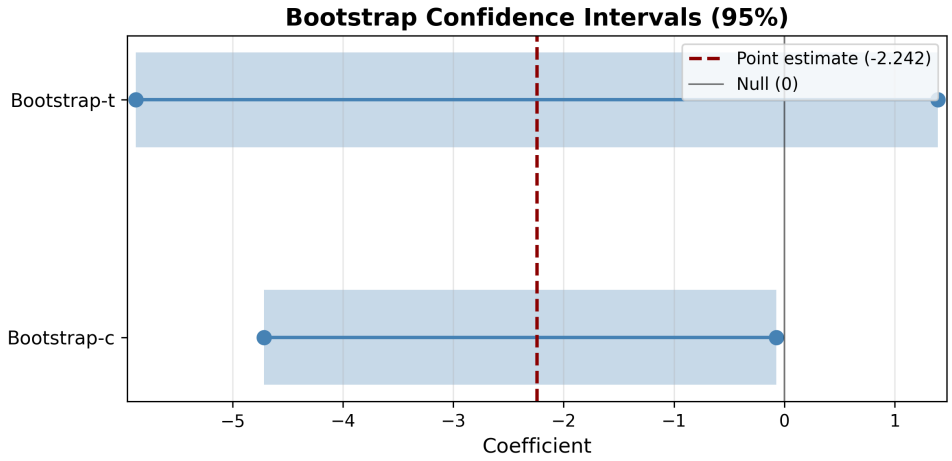


Figure 27: Bootstrap confidence intervals for sum_vb

Method	95% CI	Includes Zero?
Bootstrap-c	[-4.7151, -0.0729]	No
Bootstrap-t	[-5.8756, 1.3916]	Yes

NOTE Bootstrap results mixed — percentile CI excludes zero but studentized CI includes zero (or vice versa).

3.9.3 Sensitivity Analysis

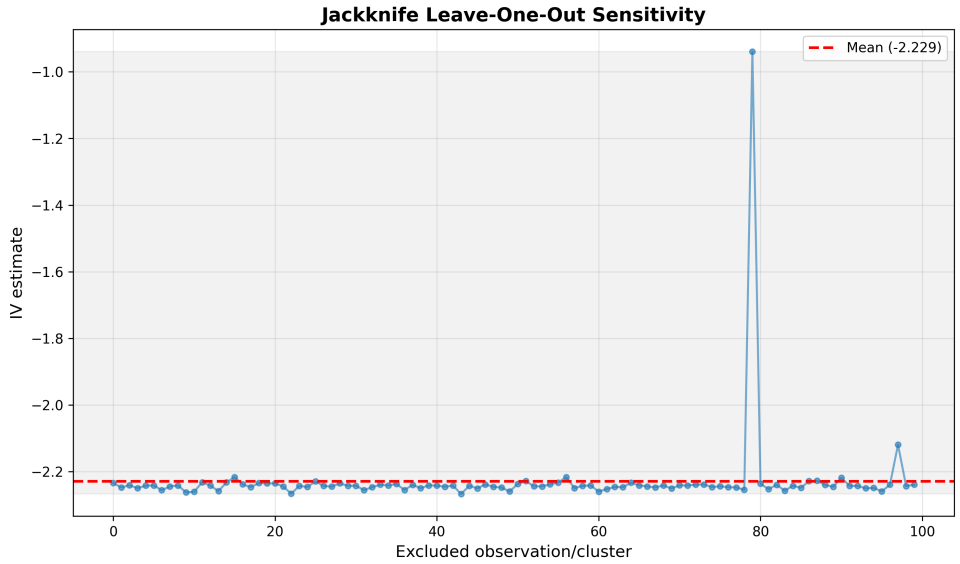


Figure 28: Jackknife leave-one-out sensitivity for sum_vb

Statistic	Value
Mean estimate	-2.2293
Range	[-2.2674, -0.9393]
Std. deviation	0.1312
Most influential unit	11001 (Δ = 1.3027)

NOTE Sensitive — 59.2% variation across leave-one-out samples.

3.9.4 IV vs. OLS Comparison

Method	Coefficient	Ratio
OLS	-0.9841	—
2SLS	**-2.2420**	**2.3x**

The 2SLS estimate is 2.3x larger than the naive OLS estimate, suggesting moderate endogeneity correction.

3.10 Table 5 (8): sum_vb

Outcome (Y): `sum_vb` | **Treatment (D):** `lm_pob_mesa` | **Instrument (Z):** `lz_pob_mesa_f`

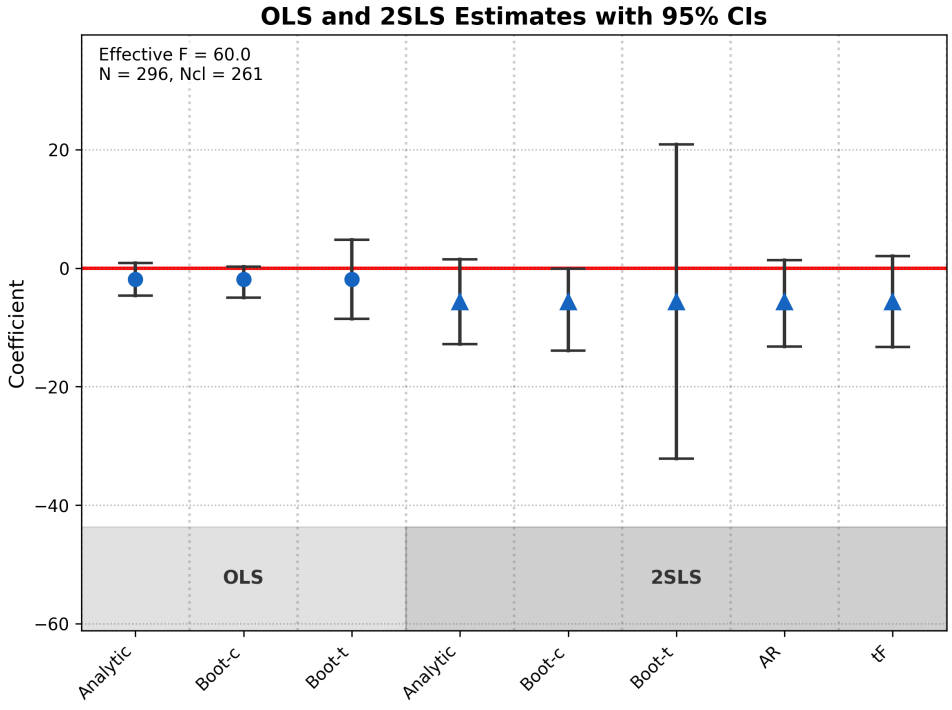


Figure 29: OLS and 2SLS estimates with 95% CIs for sum_vb

3.10.1 Instrument Strength

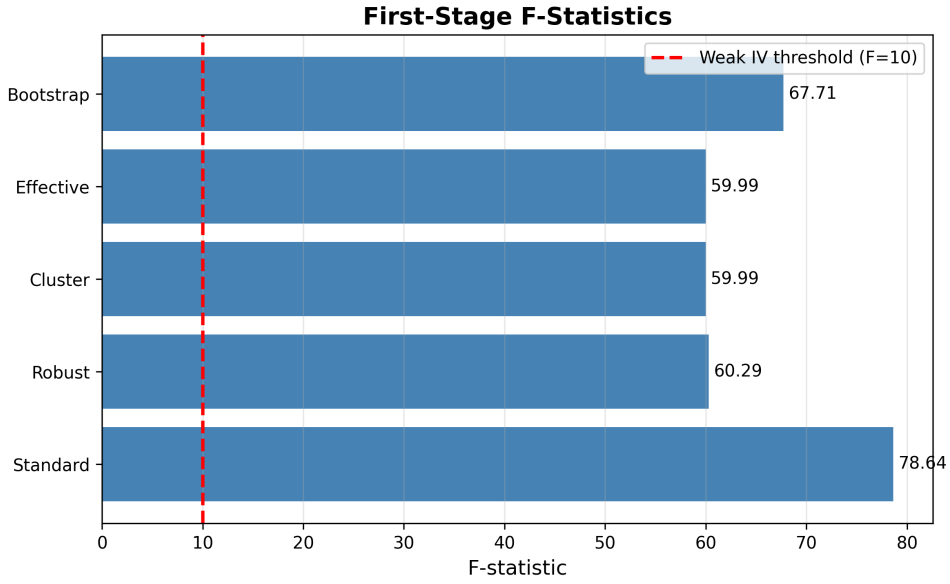


Figure 30: First-stage F-statistics for sum_vb

Statistic	Value	Assessment
F-effective	59.99	**PASS** Strong
F-standard	78.64	—
F-cluster	59.99	Cluster-robust
F-robust	60.29	HC-robust
F-bootstrap	67.71	Bootstrap-robust

First-Stage Parameter	Value
First-stage coef ($\hat{\pi}$)	0.7254
First-stage SE	0.0937
First-stage ρ (correlation coefficient)	0.4638

PASS **The instrument is strong** (F = 59.99). Standard IV inference should be reliable.

3.10.2 Robust Inference

Statistic	Value
Coefficient	-5.6436
Standard Error	3.6500
p-value	0.1221
95% CI	[-12.7976, 1.5104]
N	296
N clusters	261

The effect is **not statistically significant (p = 0.1221 > 0.05)**.

Anderson-Rubin Test (weak-IV robust): p = 0.1154 → NOTE Not significant
AR 95% CI: [-13.2356, 1.3644] (bounded)
tF Procedure (Lee et al. 2022): |t| = 1.55 vs critical t = 2.10 → NOTE Not significant
tF 95% CI: [-13.2921, 2.0050]

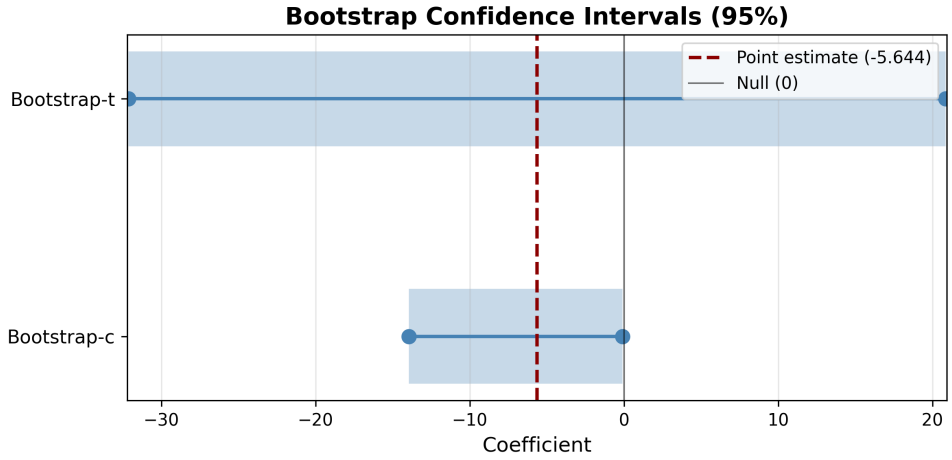


Figure 31: Bootstrap confidence intervals for sum_vb

Method	95% CI	Includes Zero?
Bootstrap-c	[-13.9454, -0.0873]	No
Bootstrap-t	[-32.1472, 20.8600]	Yes

NOTE Bootstrap results mixed — percentile CI excludes zero but studentized CI includes zero (or vice versa).

3.10.3 Sensitivity Analysis

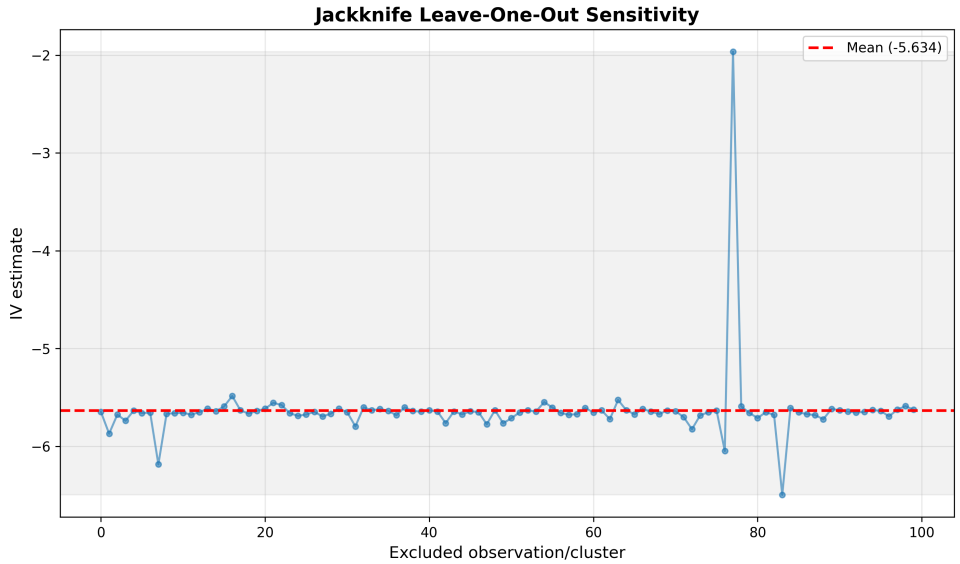


Figure 32: Jackknife leave-one-out sensitivity for sum_vb

Statistic	Value
Mean estimate	-5.6340
Range	[-6.4962, -1.9641]
Std. deviation	0.3893
Most influential unit	11001 (Δ = 3.6795)

NOTE Sensitive — 80.3% variation across leave-one-out samples.

3.10.4 IV vs. OLS Comparison

Method	Coefficient	Ratio
OLS	-1.9058	—
2SLS	**-5.6436**	**3.0x**

The 2SLS estimate is 3.0x larger than the naive OLS estimate, suggesting moderate endogeneity correction.

3.11 Diagnostic Summary

This study was evaluated across **8 specifications**. 4 specification(s) had diagnostic indicators flagged; see the specification-level sections above for details.

NOTE **Some diagnostic indicators were flagged.** The following diagnostics reported values outside standard thresholds:

- Standard and robust inference results (AR test, bootstrap CIs) are reported above
- Sensitivity analysis details are provided in the specification-level sections

4 Technical Appendix

4.1 Methods Used in This Report

Method	Purpose	Reference
F-statistic (effective)	Test instrument strength	Olea & Pflueger (2013)
Anderson-Rubin test	Weak-IV robust inference	Anderson & Rubin (1949)
Bootstrap CI (percentile)	Robust confidence intervals	Efron (1979)
Bootstrap CI (studentized)	More accurate small-sample CI	Hall (1992)
Jackknife	Sensitivity to influential obs	Quenouille (1956)
tF procedure	Weak-IV robust critical values	Lee et al. (2022)

4.2 Key References

- **Lal et al. (2024).** "How Much Should We Trust Instrumental Variable Estimates in Political Science?" *Comprehensive guide to IV diagnostics.*
- **Stock & Yogo (2005).** "Testing for Weak Instruments." *Established the $F \geq 10$ rule of thumb.*
- **Olea & Pflueger (2013).** "A Robust Test for Weak Instruments." *Effective F-statistic for heteroskedastic errors.*
- **Lee et al. (2022).** "Valid t-ratio Inference for IV." *tF procedure for weak instrument inference.*

Report generated by Journalist Agent (Replication & Diagnostics Workflow)