Online appendix to
"fereg: Fast high-dimensional fixed-effects regressions in Stata"

Johannes F. Schmieder
Department of Economics, Boston University

November 2026

B Absorption methods: exact numerical rules

This appendix accompanies section 5 of the article and describes version 0.6.0 of `fereg`. Appendix B.1 lists the methods that `solver(auto)` selects, appendix B.2 states the accuracy check and the stopping targets of the iterative methods, and the remaining sections give the exact rules of each method, including the methods that can be requested through `solver()` (appendix B.8). Equations and sections numbered without a letter prefix are those of the article. Throughout, τ denotes the value of `tolerance()`.

B.1 Methods selected by `solver(auto)`

Table B.1 summarizes the methods described in sections 5.2 and 5.4 to 5.7 of the article, with the target at which each method stops its own iterations and what happens if it declines or its result fails the check.

B.2 Accuracy check and stopping targets

Let v be a variable, $r = v - D\hat{\alpha}$ a candidate residual reconstructed from candidate effects $\hat{\alpha}$, and $s = \max(\|W^{1/2}v\|, \|W^{1/2}r\|)$, where $\|\cdot\|$ is the Euclidean length. For group g of absorbed term ℓ, let $D_{\ell g}$ hold the columns of D that belong to the group, restricted to the observations in the group: an intercept column and one column for each slope variable. Let $W_{\ell g}$ and $r_{\ell g}$ be the corresponding weights and residuals, and let $Q_{\ell g}$ be a matrix with orthonormal columns that span the columns of $W_{\ell g}^{1/2} D_{\ell g}$. The check statistic is

$$c(r) = \max_{\ell, g} \frac{\|Q_{\ell g}' W_{\ell g}^{1/2} r_{\ell g}\|}{s},$$

and `fereg` accepts the residuals if $c(r) \leq \tau$ for the outcome and for every covariate. If $s = 0$, only a zero projection is accepted. For a group of an intercept-only effect, $\|Q_{\ell g}' W_{\ell g}^{1/2} r_{\ell g}\| = \sqrt{n_g}\,|\bar{r}_g|$,

Table B.1: Methods selected by `solver(auto)`

Design	Method	Iteration target	If it declines
`fereg`			
One absorbed term	Subtract group means, or fit each group's own regression on its slopes (section 5.2)	Exact in one pass	Repeat the pass
Two to five categorical effects	Eliminate the effect with the most groups; conjugate gradients on the reduced system, preconditioned by its diagonal and, after a stall, by block-Jacobi with exact or multigrid solves (section 5.4)	$\tau/10$	Full system
Six or more categorical effects	Conjugate gradients on the full system, preconditioned by its diagonal and, if predicted slow, by additive Schwarz over pairs of effects (section 5.5)	$\tau/10$	Established routes (appendix B.4)
Two terms with slopes	Eliminate the larger term; conjugate gradients on the other, preconditioned by its group blocks (section 5.6)	$\tau/10$	Full system, then alternating projections
Three or more terms with slopes	Conjugate gradients on the full system, preconditioned by group blocks and, if slow, by Schwarz	$\tau/10$	Alternating projections
`feppml`, within each iteration			
One absorbed term	As for `fereg`	Exact in one pass	As for `fereg`
First iterations of a fresh fit	Project the working response and covariates with the elimination; solve the covariates' normal equations	As for `fereg`	Joint steps
Two or more absorbed terms	Eliminate the term with the most coefficients; conjugate gradients for the remaining effects and the coefficient update together (section 5.7)	$\tau_k/4$	Absorb each variable separately; error if that fails
Two categorical effects	As above, with a multigrid preconditioner that is built when needed and reused across iterations	$\tau_k/4$	As above

Notes: τ is the tolerance set by `tolerance()`. The iteration target is the value at which a method stops its own iterations, relative to the variable's length. Linear absorptions must then pass (6) at τ; the smaller targets leave a margin for reconstruction error. A declined variable continues with the next method within the remaining iterations, and the result reports a certified fallback. For `feppml`, τ_k tightens from 10^{-6} to $\min(10^{-12}, \tau/10)$. Joint steps require reconstructed fixed-effect moments at $\tau_k/2$ and independently computed covariate scores at τ_k; the strict and final weighted fits absorb every variable separately with the methods of `fereg` and check them at the strict tolerance (section 5.7). Joint steps require at least one covariate and sufficient workspace. Optional preconditioners are built only if their workspace fits in memory. An error stops the command without posting results.

which gives (6). The statistic is computed from the original data and weights, not from quantities produced inside a solver.

The iteration targets in table B.1 are smaller than τ. Evaluating the check requires reconstructing r from the estimated effects and a full pass over the data, which is too expensive to repeat at every step. An iterative method instead tracks its progress with an internal measure that it updates almost for free. For conjugate gradients on the reduced system (7), the measure is the vector of weighted residual sums of the remaining groups, updated by the recurrence rather than recomputed. For conjugate gradients on the full system, it is the residual of the normal equations, and for LSMR an estimate of it. These measures follow the check closely but not exactly: a running estimate drifts away from the true residual as rounding errors accumulate, a measure for the system as a whole does not bound the worst group, and reconstructing r adds rounding errors of its own. Aiming at $\tau/10$ uses additional iterations to reduce the risk that the reconstructed result fails the check. The cost depends on the conditioning of the problem, and the margin does not guarantee acceptance. A result that fails the check is retried as described below or passed to the next method. The targets affect only when a method stops, not the standard it must meet.

B.3 Two to five categorical effects

In the notation of (7), the worker effects follow from the remaining effects as $a_1 = N_1^{-1}(D_1'Wv - Ca_2)$, and the residual is $\tilde{v} = v - D_1a_1 - D_2a_2$. With two effects, let c_j' be row j of C and $n_j = \sum_k C_{jk}$ the size of worker j. Then

$$S = \sum_j L_j, \qquad L_j = \operatorname{diag}(c_j) - \frac{c_j c_j'}{n_j}.$$

Each L_j is the Laplacian of a complete graph on the firms at which worker j is observed, with weight $C_{jk}C_{jl}/n_j$ on the link between firms k and l; its rows sum to zero because $\sum_k C_{jk} = n_j$. With three to five effects, N_2 and C have one block for each remaining effect, and the products with S are formed from the distinct cells of each worker's remaining groups. Products with S are computed without forming it.

Conjugate gradients runs on up to eight variables in lockstep with the preconditioner $M = \operatorname{diag}(S)$. A variable stops when the weighted residual sums $b - Sa_2$ of the remaining groups have length at most $\tau/10$ times the weighted length of the variable after removing its worker means; in weighted fits, an estimate of the error in the coefficients must also be at most τ times the root mean square of the variable. A variable whose measure stops falling at the level of rounding error returns to its best iterate if that iterate's measure is at most $\tau/2$ times the variable's length, and declines otherwise. The iterations stall if a variable's measure does not halve within 40 steps (within 8 steps once its iterate would be accepted), and the method runs at most 1,500 steps.

A stall or the step limit switches the variables to block-Jacobi preconditioning and continues from their current iterate. The preconditioner is block diagonal, with one block B_b for each remaining effect b, approximating the inverse of the block S_{bb} of S. A block with at most 256 groups, or at

most 1,024 when its dense solve costs no more than a pass of the reduced operator, is factored exactly by a Cholesky factorization with one group fixed in each connected set. A larger block is solved approximately by one cycle of combinatorial multigrid (Koutis et al., 2011) on the graph of S_{bb}: a worker observed in more than five groups of the block is an auxiliary vertex joined to each of them, whose elimination gives exactly that worker's term L_j, and other workers contribute the links of L_j directly. If multigrid builds no hierarchy, the block is preconditioned by its diagonal. An escalated variable stops at a tenth of its original target. Later solves of the same design start with block-Jacobi. The blocks are built for the current weights and only if their workspace fits in memory.

The worker effects follow by back-substitution, and one pass over the data forms $\tilde{v}$ and recomputes the weighted residual sums of every remaining group, which must pass the check at τ; the worker sums vanish by construction. A result that fails the check is projected again from its residual. A stall after escalation, a refused workspace, or a numerical breakdown returns the untouched variable to the full-system method of appendix B.4. If that also declines, two effects continue with the established conjugate gradients on (7) with a diagonal preconditioner, which a structural rule may replace by multigrid when the iterations stall, and three or more effects with alternating projections.

B.4 Six or more categorical effects

Conjugate gradients on the full system solves $D'WD\,x = D'Wv$ for up to eight variables in lockstep, starting from the group means of the largest effect, with $M = \mathrm{diag}(D'WD)$. After 24 steps, `fereg` projects from the recent rate of progress the number of steps each variable still needs. If a projection exceeds 100 steps, or if the iterations stall or exhaust their budget, the variables continue from their current iterate with the additive Schwarz preconditioner

$$M^{-1} = \sum_i R_i'\, A_i^{\dagger}\, R_i,$$

provided that its workspace fits in memory; otherwise they continue with the diagonal. Here i runs over the connected sets of the graphs formed by the largest effect and each other effect, R_i selects the groups in connected set i, and $A_i = R_i D'WDR_i'$ is the corresponding block of $D'WD$ (Toselli and Widlund, 2005; Fischer and Schröder, 2026). $A_i^{\dagger}$ eliminates the effect with more groups exactly and solves the Laplacian that remains: by a dense Cholesky factorization when it is small or dense, by an envelope Cholesky factorization when that costs at most 128 operations per entry, and otherwise by the approximate Cholesky factorization AC(2) of Gao et al. (2026) with a fixed seed. A term proportional to the constant vector of each connected set keeps $A_i^{\dagger}$ positive definite. The corrections are added without weights and without a coarse correction. Later solves of the same design start with this preconditioner.

The iterations stop when the residual of the normal equations falls below $\tau/10$ times its starting length, and the result is accepted only if the reconstructed residuals pass the check for every group. The method may use half of the limit set by `iterate()`. If it declines, the variables continue with

the established routes within the rest of the limit: two effects as in appendix B.3, and three or more effects with alternating projections. With exactly three effects and unit weights, a short pilot of alternating projections, bounded by the overlap of each pair of effects, may come first.

B.5 Absorbed slopes and alternating projections

With slopes, each group's coefficients form a block of $D'WD$ whose size is the number of the term's channels: its intercept, if any, and its slope variables. With exactly two terms, the larger term's group blocks are eliminated exactly, and conjugate gradients solves for the other term's coefficients, preconditioned by the exact inverses of its group blocks of the reduced system. With three or more terms, or when this declines, conjugate gradients runs on the full system with the inverse group blocks as preconditioner. A block whose channels are dependent within the group, such as a slope that is constant in the group, acts as a pseudo-inverse on its range. When the iterations are predicted to be slow, the preconditioner is replaced by additive Schwarz: each pair of the largest term with another term is one subdomain over both terms' channels, whose larger term's group blocks are eliminated exactly and whose remaining system is solved exactly when it has at most 128 coefficients and by its group blocks otherwise. Both methods stop at $\tau/10$ and are checked at τ. If they decline, alternating projections finish the fit. They use symmetric sweeps and compute the check every four sweeps.

B.6 Final residuals

When residuals, saved fixed effects, or predictions that include the absorbed effects are requested, `fereg` computes and checks the final residuals with $\min(\tau, 10^{-14})$ in place of τ. Otherwise, with `solver(auto)` and the default memory plan, the final residual is formed from the certified absorptions of the outcome and covariates without a new solve and checked directly; explicit methods, the low-memory plan, and a failed check absorb it again.

B.7 Poisson regression

Let $f = \min(10^{-12}, \tau/10)$. In iteration k of iteratively reweighted least squares, `feppml` absorbs with tolerance

$$\tau_k = \max\{f,\ \min(t_k,\, p_k,\, \tau_{k-1})\}, \qquad \tau_0 = p_1 = 10^{-6},$$

where $t_k = \exp\{(1-\lambda_k)\log 10^{-6} + \lambda_k \log f\}$ with $\lambda_k = \min\{(k-1)/7, 1\}$ tightens the tolerance over eight iterations. For $k > 1$, the progress target p_k equals f once the previous accepted iteration satisfies both outer convergence criteria; otherwise it equals $0.1\max(\Delta_{k-1}, \sigma_{k-1})$, bounded to $[f, 10^{-6}]$, where Δ is the relative change in the deviance and σ is the largest normalized score. Once the outer criteria hold, `feppml` solves the weighted regression with tolerance f and resumes iterating if the estimates move. Before this strict solve, `feppml` waits while the fit is still contracting: with m_k the larger of the relative changes in the linear predictor and in the coefficients between the

last two accepted iterations, it enters the strict solve once $m_k^2/m_{k-1} \leq \min(\tau, 10^{-8})$, or at once if $m_k/m_{k-1} > 1/2$, if fewer than three iterations remain, or after eight such deferrals.

A fresh fit with two to five categorical effects and at least one covariate, in the fast memory plan, first takes plain iterations. Each projects the working response and the covariates with the elimination of appendix B.3, starting from the previous iteration's residuals plus the change in the working response, and solves the normal equations of the projected covariates. Only full steps that do not increase the deviance are accepted; a damped step, a covariate collinear with the fixed effects, or a rank or numerical problem discards these iterations and restores the starting values. They stop once a step changes the deviance by at most $\tau/100$, and at most $k_{\max} - 2$ are taken, where $k_{\max}$ is the limit set by `iterate()`. If the last step meets the deviance criterion and the scores pass their independent check, it counts as the first of the two converged iterations; only the linear predictor is passed on.

In later ordinary iterations with at least two absorbed terms and at least one covariate, `feppml` eliminates the term with the most coefficients exactly and solves for the remaining fixed-effect coefficients and the coefficient update δ jointly by conjugate gradients, starting from the previous iteration's solution. The preconditioner is block diagonal, with the current weighted group sizes $\tilde{N}$ for the fixed effects and the Cholesky factor of $\tilde{X}'H\tilde{X}$ for δ, where $\tilde{X}$ holds the covariates after the eliminated effect is removed and $H = \mathrm{diag}(h_i)$. The recurrence stops at $\tau_k/4$. Reconstructing the residual from the original inputs must then satisfy the fixed-effect moment check at $\tau_k/2$, and normalized scores for the covariates, computed separately from the original inputs, must be at most τ_k. This joint-step check does not require separate absorptions of every covariate. The strict and final weighted fits do compute those absorptions, with the elimination of appendix B.3 for two to five categorical effects and the full system of appendix B.4 for six or more, and apply the accuracy check of appendix B.2 at the strict tolerance f.

With exactly two categorical effects, a joint solve that reaches step 64 builds a multigrid preconditioner P for the current reduced matrix, or rebuilds one built under earlier weights, and restarts from its current iterate. Later iterations apply $\tilde{N}^{1/2}P\tilde{N}^{1/2}$ to the fixed-effect block, with $\tilde{N}$ evaluated at the current weights. Before the strict solve, `feppml` attempts to rebuild a preconditioner built under earlier weights with the current weights. If none exists, `feppml` solves for the first variable on the largest connected set without one and builds one if that solve needs more than 32 steps. The strict solve uses the preconditioner for connected sets with at least 64 groups. A preconditioner is built or used only if its workspace fits in memory; if a rebuild does not fit, the earlier preconditioner can be kept. `feppml` attempts at most three constructions per regression, including those refused for lack of memory.

With one absorbed term, `feppml` uses the exact calculation of section 5.2. If the model has no covariates, or if the joint solve does not fit in memory, `feppml` absorbs the working response and the covariates separately with the methods of `fereg`. If the joint solve fails its check, `feppml` repeats the iteration once by absorbing every variable separately. If that also fails, the command stops with an error.

B.8 Methods requested through `solver()`

For `fereg`, `solver(cg)` with `reduce(eliminate)` or `reduce(full)` runs the methods of appendices B.3 to B.5; `precondition(auto)` escalates as described there, and a named preconditioner never escalates. `solver(lsmr)` runs modified LSMR (Fong and Saunders, 2011) on $W^{1/2}D$, one variable at a time or up to eight in lockstep, with $M \approx D'WD$ given by its diagonal (or the group blocks with slopes), the identity, or the additive Schwarz preconditioner of appendix B.4; `precondition(auto)` starts with the diagonal and switches to Schwarz as for conjugate gradients. Because its stopping tests are proxies, each LSMR pass is followed by the check, and a pass that fails it continues from its iterate at a tighter target, for at most four passes. Both methods stop at $\tau/10$ and are checked at τ. `solver(cg, precondition(cmg))` builds one graph of the two largest categorical effects, with up to 64 further fixed-effect columns, and solves all variables by multigrid-preconditioned conjugate gradients; if the multigrid preconditioner cannot be built or applied, it continues with conjugate gradients on the operator of alternating projections and reports the route `map, accelerate(cg)`.

A request that its method cannot serve continues on a certified route within the remaining iterations and the same memory limit. For `solver(lsmr)` and `solver(cg)` on categorical effects, the next routes are the automatic elimination and full system without the formulation that failed, then the established routes; with slopes, the full system with the automatic preconditioner, then alternating projections; for multigrid requests outside their support, the automatic routes. The stored results report the request in `e(fe_requested)`, the route that ran in `e(fe_route)`, and `e(fe_fallback)` and `e(fe_fallback_reason)`. A request that exhausts `iterate()` stops with an error.

For `feppml`, explicit `solver(lsmr)` requires at least two categorical intercept effects, no absorbed slopes, and the fast memory plan. Each iteration solves its weighted least-squares problem by one modified LSMR recurrence on $W^{1/2}[X\ D]$, with a block-diagonal preconditioner: the covariates' weighted cross-product factor and, for the fixed effects, the inverse group sizes, the identity, or the additive Schwarz preconditioner of appendix B.4 built for the current weights. With `precondition(auto)`, a recurrence that stalls or misses its check continues with the Schwarz block, and one that still fails hands the iteration to separate projections of the covariates and the working response; later iterations start where the earlier ones ended. `e(fe_formulation)` reports which of these ran. Explicit `solver(cg)` uses the multigrid graph and requires two or three categorical effects, the third with at most 64 non-base indicators. An iteration that explicit methods do not certify, and the later ones, run on the automatic route, and the final weighted fit always does; requests outside these requirements run `solver(auto)`, with the fallback reported.

B.9 Two convergence stress tests

These small, identified examples illustrate stopping-rule failures under deliberate numerical stress.[1]

[1] Comparisons use `fereg` 0.5.0, `reghdfe` 6.14.1, `ppmlhdfe` 2.3.4 and `fixest` 0.14.2 on one Mac with one worker. Scripts and the full grids, including warnings and refusals, are in `replication/certification_rhs_scale/`.

OLS and covariate units. For each $g = 1, \ldots, 100$, connect to $h = g$ and $h = g + 1$, wrapping 101 to 1. Each pair contains all four combinations of $u, v \in \{-1, 1\}$, giving 800 observations. Set $y = 2u + v$ and $x = s(A_g + B_h + u)$, where

$$A_g = 100[\cos(4\pi g/100) + \sin(10\pi g/100)],$$
$$B_h = 100[\sin(2\pi h/100) + \cos(6\pi h/100) + \sin(14\pi h/100)].$$

Absorbing g and h leaves su and $2u + v$, so the exact slope in common units is $s\hat{\beta} = 2$ and RSS is 800; an explicit-dummy SVD confirms this on the stored inputs. Changing s from 1 to 10^{-4} moves default `fixest`'s slope from 1.999999975 to 1.992224802 without an omission or convergence warning. At the smaller scale, its demeaned covariate gives 3.59×10^{-6} in (6), about 359 times the default threshold. Both `fereg` and `reghdfe` recover 2; `fixest` also does so with `fixef.tol=1e-11`.

PPML and small positive outcomes. Each of 100 groups contains $x = (0, 0, 1, 1)$ and $y = (0.5, 1.5, 0.5\delta, 1.5\delta)$, with $\delta = 10^{-10}$. The model $\mu_i = \exp(\alpha_g + \beta x_i)$ has finite optimum $\alpha_g = 0$ and $\beta = \log \delta = -23.02585093$; all outcomes are positive. Default `ppmlhdfe` and `fixest` report convergence at -21.27668584 and -21.20153855, while `feppml` recovers the optimum. After optimizing the group effects conditional on each returned slope, independently recomputed normalized scores are 1.77×10^{-5} and 1.87×10^{-5}, respectively, exceeding 10^{-8}. `ppmlhdfe` warns about low standardized outcomes. Its `tolerance(1e-12)` gives -23.02479761; `fixest` with `glm.tol=1e-12`, `fixef.tol=1e-11` and `glm.iter=100` gives -23.02584963. Neither example measures the frequency of such errors in empirical applications.

C Computational performance: additional results

This appendix accompanies section 7 of the article. It describes the benchmark protocol (appendix C.1), reports the achieved numerical deviations and the tighter-setting timings (appendix C.2), summarizes the variation between repeated times (appendix C.3), and lists every simulated configuration at both settings (appendix C.4). Two historical results of version 0.5.0 follow: the configurations of table 5 on a newer node (appendix C.5) and the dependence of coefficient accuracy on `tolerance()` (appendix C.6).

C.1 Benchmark protocol

The campaign ran on November 10, 2026 on the Linux nodes of Boston University's Shared Computing Cluster, with the pre-release build `80b6768e` of `fereg`, whose numerical runtime is that of version 0.6.0. Its 1,297 comparison blocks ran in 779 allocations, each of which reserved one whole 28-core node; the scheduler placed them on 47 nodes, whose processor models the evidence does not record. The schedule, the inputs, and the rules for retaining results were fixed before the run: every scheduled fit was attempted once, including larger datasets after a timeout, and no fit

was repeated or replaced. An earlier run of the same schedule with a previous build was stopped after two failures of `fereg` (a tighter PPML fit and two saved-effect refits); after their repair, the complete schedule was run again, and none of the earlier observations are used here.

In each block, `fereg` and one comparator each ran at their default and tighter settings (table C.1), each fit in a fresh process with the data already in memory, on the same assigned cores, with the order of the commands rotating across repetitions. Before timing, each process fit the same kind of model, with the same number of fixed effects, type of standard errors, and saved output, to a small separate dataset, so that compilation in Julia and first-use costs elsewhere fall outside the measured time. Every package was limited to the assigned cores through its own setting: `set processors` in Stata, `setFixest_nthreads()` in R, `RAYON_NUM_THREADS` for `PyFixest`, the number of Julia threads for `FixedEffectModels.jl`, `GLFixedEffectModels.jl`, and the Julia session of `reghdfejl`, with the linear-algebra libraries of every language on one thread. Stata was licensed for four cores, so the Stata commands ran only with one, two, and four. `fixest` 0.14.2 ran in R 4.5.2 and `PyFixest` in Python 3.12, with the pinned environments of the earlier measurements; `reghdfejl` used its own Julia 1.12.7 runtime and pinned package versions, with its Stata–Julia bridge built from the pinned source.

Each package computed the requested type of standard errors with its own default conventions: IID errors for the linear models and robust errors for PPML unless noted, and errors clustered by worker where noted. For IID errors, the Stata commands used `vce(unadjusted)`, or `vce(ols)` for `areg`. The command limit was 3,600 seconds for every command, within a process limit of 4,500 seconds. Memory is the operating system's record of the peak resident set size of the estimating process, reset before and read after the timed command. Individual observations and predictions stayed in temporary storage on each node; the numerical comparisons below were computed there from the complete arrays before deletion. The replication archive keeps the compact results and per-configuration aggregates of the deviations; the per-fit deviations, coefficients, and standard errors are retained outside it, identified by their hashes.

Table C.1: Default and tighter stopping settings

Package	Default	Tighter
`fereg`, `feppml`	package defaults	`tolerance(1e-12)`
`reghdfe`	package defaults	`tolerance(1e-12)`
`ppmlhdfe`	package defaults	outer `tolerance(1e-12)`; default inner absorption
`areg`	package defaults	`tolerance(1e-12)`
`reghdfejl`	package defaults	`tolerance(1e-12)`
`fixest`	package defaults	fixed effects 10^{-11}; PPML iterations 10^{-10}
`PyFixest`	adaptive `within` default	fixed effects 10^{-11}; PPML iterations 10^{-10}
`FixedEffectModels.jl`	package defaults	tolerance 10^{-10}
`GLFixedEffectModels.jl`	package defaults	centering, deviance and step tolerances 10^{-12}

Notes: The tighter settings change only the stopping rules; standard errors, degrees-of-freedom conventions, and separation rules stay at each package's defaults. Equal numbers do not define a common error criterion across packages.

Of the 5,188 timed fits, 5,178 completed and 10 reached the limit: all three tighter `reghdfe` fits of the sparse three-effect design with ten million observations, and 7 of the 15 tighter `GLFixedEffectModels.jl` fits on the Veneto PPML panel. In 130 completed fits the numerical capture was flagged. In 115 fits with saved effects, the saved predictions did not satisfy an independent check of the model's first-order conditions at the package's settings; these are default fits of `fixest`, `PyFixest`, `FixedEffectModels.jl`, `GLFixedEffectModels.jl`, `reghdfe`, and `reghdfejl` and default and tighter fits of `ppmlhdfe`. In the other 15, tighter `GLFixedEffectModels.jl` fits reported a log likelihood of $-\infty$, which the predictions cannot reproduce. `GLFixedEffectModels.jl` evaluates the Poisson density, which assigns zero probability to the non-integer outcomes of the benchmark data, and reports this log likelihood in all its fits; its estimates are unaffected, and the comparisons below use the PPML objective evaluated from its predictions. All these fits keep their timings and numerical captures. All 2,594 timed fits of `fereg` completed and passed the campaign's acceptance checks.

C.2 Numerical agreement and tighter stopping settings

Default stopping settings differ in both their numerical values and the quantities they monitor. We therefore report numerical differences separately from running times. Every fit is compared with the tighter `fereg` fit of the same block, in every repetition (tables C.2 and C.3). Coefficients are compared in units of the reference standard errors; standard errors and covariance matrices are compared as reported, with each package's default inference. Predictions are compared on the intersection of the retained samples, and objectives only when the samples coincide. Sample sizes, sample intersections, and omitted coefficients are retained in the numerical evidence.

At default settings, `fereg` stayed within 4.1×10^{-10} standard errors of its tighter fit in every coefficient, and its standard errors within a relative 1.3×10^{-10}. At tighter settings, the coefficients of every comparator except `areg` and `GLFixedEffectModels.jl` were within 2.4×10^{-8} standard errors of the reference; the differences of `areg`, of order 10^{-4} standard errors, are unchanged at 10^{-12}, and those of `GLFixedEffectModels.jl` shrink only slightly. The standard-error differences are conventions. In the PPML models of `fixest` and `PyFixest`, the covariance matrices differ from `feppml`'s by a common factor that equals $N/(N-p)$, with p the number of coefficients and absorbed fixed effects, to within 10^{-4} in the worker–firm design with one million observations and on the Veneto panel: about 1.27 and 1.31, so that the standard errors are up to 12.5% and 14.4% larger. `areg`'s clustered standard errors are up to 11.9% larger, those of the other linear commands differ by at most 0.04%, and those of `GLFixedEffectModels.jl` by at most 0.5%.

Samples differ only in the gravity design. `feppml` and `ppmlhdfe` retain 960,000 observations and omit the separated control; `fixest` and `PyFixest` retain 970,000 and report a coefficient for it; `GLFixedEffectModels.jl` retains all 1,000,000. The coefficients on the common controls agree to within 10^{-7} standard errors, and no objective is compared across these samples.

Table C.4 summarizes the tighter-setting timings; each comparator's tighter fit is divided by the tighter `fereg` fit of the same block. Tightening `fereg`'s tolerance to 10^{-12} raised its time by

Table C.2: Coefficient and covariance deviations from tighter `fereg`

		Max. $\lvert\Delta\beta\rvert/s$		Max. $\lvert\Delta s\rvert/s$		$\lVert\Delta V\rVert_F/\lVert V\rVert_F$	
Package	Fits	Default	Tighter	Default	Tighter	Default	Tighter
Simulations, linear							
`fereg`	732	4.1×10^{-10}	ref.	1.3×10^{-10}	ref.	3.6×10^{-10}	ref.
`reghdfe`	112/109	1.9×10^{-9}	3.1×10^{-10}	6.4×10^{-11}	4.6×10^{-13}	1.7×10^{-10}	1.2×10^{-12}
`reghdfejl`	112/112	9.0×10^{-8}	3.5×10^{-10}	3.7×10^{-4}	3.7×10^{-4}	7.5×10^{-4}	7.5×10^{-4}
`areg`	97/97	4.7×10^{-4}	4.7×10^{-4}	0.119	0.119	0.253	0.253
`fixest`	137/137	6.1×10^{-5}	1.7×10^{-9}	3.6×10^{-4}	3.6×10^{-4}	7.2×10^{-4}	7.2×10^{-4}
`PyFixest`	137/137	3.1×10^{-10}	3.5×10^{-10}	3.6×10^{-4}	3.6×10^{-4}	7.2×10^{-4}	7.2×10^{-4}
`FixedEffectModels.jl`	137/137	2.3×10^{-4}	3.5×10^{-10}	3.7×10^{-4}	3.7×10^{-4}	7.5×10^{-4}	7.5×10^{-4}
Simulations, PPML							
`fereg`	439	5.3×10^{-11}	ref.	4.0×10^{-15}	ref.	3.8×10^{-15}	ref.
`ppmlhdfe`	91/91	1.7×10^{-10}	6.8×10^{-11}	8.6×10^{-9}	1.6×10^{-10}	1.4×10^{-8}	3.1×10^{-10}
`fixest`	116/116	8.6×10^{-7}	1.1×10^{-10}	0.125	0.125	0.266	0.266
`PyFixest`	116/116	1.8×10^{-8}	1.8×10^{-8}	0.125	0.125	0.266	0.266
`GLFixedEffectModels.jl`	116/116	9.5×10^{-7}	8.3×10^{-7}	5.0×10^{-3}	5.0×10^{-3}	0.0101	0.0101
Veneto, OLS							
`fereg`	72	6.8×10^{-12}	ref.	1.0×10^{-14}	ref.	1.2×10^{-14}	ref.
`reghdfe`	9/9	5.4×10^{-9}	4.6×10^{-9}	3.9×10^{-13}	4.5×10^{-13}	2.0×10^{-13}	1.3×10^{-13}
`reghdfejl`	9/9	4.5×10^{-8}	9.7×10^{-11}	1.2×10^{-7}	1.2×10^{-7}	2.4×10^{-7}	2.4×10^{-7}
`areg`	9/9	2.5×10^{-4}	2.5×10^{-4}	3.3×10^{-7}	3.3×10^{-7}	6.5×10^{-7}	6.5×10^{-7}
`fixest`	15/15	1.6×10^{-6}	2.4×10^{-8}	1.3×10^{-9}	5.5×10^{-11}	2.6×10^{-9}	4.3×10^{-11}
`PyFixest`	15/15	8.9×10^{-11}	1.1×10^{-10}	1.7×10^{-13}	1.9×10^{-13}	1.2×10^{-13}	1.4×10^{-13}
`FixedEffectModels.jl`	15/15	9.2×10^{-5}	9.9×10^{-11}	1.3×10^{-7}	1.2×10^{-7}	1.8×10^{-7}	2.4×10^{-7}
Veneto, PPML							
`fereg`	54	1.1×10^{-12}	ref.	3.1×10^{-15}	ref.	1.2×10^{-15}	ref.
`ppmlhdfe`	9/9	3.2×10^{-11}	1.5×10^{-11}	7.1×10^{-10}	5.4×10^{-11}	5.7×10^{-10}	4.6×10^{-11}
`fixest`	15/15	5.4×10^{-8}	1.2×10^{-10}	0.144	0.144	0.309	0.309
`PyFixest`	15/15	1.4×10^{-10}	1.5×10^{-10}	0.144	0.144	0.309	0.309
`GLFixedEffectModels.jl`	15/8	6.6×10^{-10}	2.7×10^{-12}	1.1×10^{-7}	9.1×10^{-8}	1.8×10^{-7}	1.8×10^{-7}

Notes: Every repetition of every comparison block. The reference is tighter `fereg` (10^{-12}) in the same block; for `fereg` the default fit is compared with it (ref.: the reference itself). This measures pairwise agreement, not independent estimation error. Counts give default/tighter fits with a numerical capture; timed-out fits have none. Entries are absolute maxima. s and V are the reference standard errors and covariance matrix; coefficients are compared on the common estimable slopes. With package-default inference, standard-error and covariance differences mainly reflect small-sample conventions, which a tighter setting does not change. Coefficient-level deviations, maximum absolute covariance differences and best-fitting covariance scalars are retained in the numerical evidence.

Table C.3: Predictions, objectives and retained samples

		Rel. RMSE		Reported $\lvert\Delta\ell\rvert/N$		Replayed $\lvert\Delta\ell\rvert/N$		
Package	Pred. fits	Default	Tighter	Default	Tighter	Default	Tighter	Diff. samples
Simulations, linear								
`fereg`	228	8.5×10^{-14}	ref.	3.0×10^{-15}	ref.	1.2×10^{-16}	ref.	0
`reghdfe`	36/35	9.7×10^{-5}	1.2×10^{-8}	6.5×10^{-13}	2.2×10^{-15}	1.1×10^{-8}	2.3×10^{-16}	0/0
`reghdfejl`	36/36	3.3×10^{-6}	4.4×10^{-10}	1.8×10^{-11}	2.9×10^{-16}	1.4×10^{-11}	3.3×10^{-16}	0/0
`areg`	0/0	—	—	2.3×10^{-7}	2.3×10^{-7}	—	—	0/0
`fixest`	41/41	2.1×10^{-4}	5.4×10^{-7}	5.0×10^{-8}	3.5×10^{-13}	5.0×10^{-8}	3.5×10^{-13}	0/0
`PyFixest`	45/45	3.7×10^{-7}	3.0×10^{-7}	7.2×10^{-15}	3.3×10^{-16}	1.6×10^{-13}	1.1×10^{-13}	0/0
`FixedEffectModels.jl`	45/45	2.3×10^{-4}	3.8×10^{-8}	8.1×10^{-8}	2.0×10^{-15}	6.4×10^{-8}	1.6×10^{-15}	0/0
Simulations, PPML								
`fereg`	135	3.0×10^{-14}	ref.	0	ref.	4.7×10^{-16}	ref.	0
`ppmlhdfe`	27/27	2.0×10^{-5}	3.3×10^{-7}	1.2×10^{-12}	2.5×10^{-14}	6.3×10^{-9}	8.8×10^{-14}	0/0
`fixest`	34/34	2.1×10^{-6}	4.2×10^{-9}	3.1×10^{-11}	5.1×10^{-15}	3.1×10^{-11}	7.0×10^{-16}	5/5
`PyFixest`	36/36	1.4×10^{-7}	2.3×10^{-8}	3.4×10^{-12}	1.2×10^{-15}	3.4×10^{-12}	2.1×10^{-15}	5/5
`GLFixedEffectModels.jl`	36/36	4.1×10^{-5}	1.1×10^{-7}	—	—	2.1×10^{-10}	1.7×10^{-14}	5/5
Veneto, OLS								
`fereg`	24	9.8×10^{-16}	ref.	2.1×10^{-17}	ref.	2.6×10^{-18}	ref.	0
`reghdfe`	3/3	2.0×10^{-6}	1.3×10^{-9}	1.5×10^{-15}	2.9×10^{-16}	9.1×10^{-11}	5.5×10^{-17}	0/0
`reghdfejl`	3/3	8.5×10^{-8}	8.2×10^{-12}	1.5×10^{-13}	5.3×10^{-18}	1.6×10^{-13}	5.3×10^{-18}	0/0
`areg`	0/0	—	—	9.8×10^{-9}	9.8×10^{-9}	—	—	0/0
`fixest`	5/5	6.6×10^{-7}	8.0×10^{-10}	9.8×10^{-12}	5.3×10^{-18}	9.8×10^{-12}	7.9×10^{-18}	0/0
`PyFixest`	5/5	1.3×10^{-9}	1.3×10^{-9}	5.3×10^{-17}	5.3×10^{-18}	4.5×10^{-17}	4.8×10^{-17}	0/0
`FixedEffectModels.jl`	5/5	5.4×10^{-6}	1.1×10^{-9}	9.6×10^{-10}	7.9×10^{-18}	6.7×10^{-10}	2.6×10^{-17}	0/0
Veneto, PPML								
`fereg`	18	2.2×10^{-15}	ref.	0	ref.	6.8×10^{-16}	ref.	0
`ppmlhdfe`	3/3	1.8×10^{-5}	3.3×10^{-8}	1.2×10^{-13}	1.1×10^{-13}	9.7×10^{-8}	3.0×10^{-13}	0/0
`fixest`	5/5	1.7×10^{-7}	1.6×10^{-11}	3.9×10^{-12}	8.0×10^{-14}	3.8×10^{-12}	6.8×10^{-15}	0/0
`PyFixest`	5/5	2.6×10^{-7}	2.6×10^{-7}	6.4×10^{-14}	6.1×10^{-14}	4.8×10^{-12}	4.8×10^{-12}	0/0
`GLFixedEffectModels.jl`	5/3	3.8×10^{-7}	1.9×10^{-10}	—	—	1.8×10^{-11}	1.4×10^{-15}	0/0

Notes: Every repetition of every comparison block. The reference is tighter `fereg` (10^{-12}) in the same block; for `fereg` the default fit is compared with it (ref.: the reference itself). This measures pairwise agreement, not independent estimation error. Counts give default/tighter fits with a numerical capture; timed-out fits have none. Entries are absolute maxima. Predictions come from saved-effect fits or, for unsaved timing fits, from a separate saved-effect refit in the first repetition; relative RMSE divides the RMSE by the RMS reference prediction on the common sample. Objectives are the RSS for OLS and the full Poisson log likelihood for PPML, per observation; reported values come from each package, replayed values from its saved predictions, and neither is compared across different samples. `GLFixedEffectModels.jl` reports a log likelihood of $-\infty$ for non-integer outcomes (—). Diff. samples counts fits whose retained sample differs from `fereg`'s; all are in the gravity design.

19% on average (geometric mean of the within-block ratios). The comparators' geometric-mean ratios are larger than at default settings, except those of `ppmlhdfe` and of `PyFixest` in PPML. A comparator was faster than `fereg` in three configurations: `FixedEffectModels.jl` with one fixed effect and `PyFixest` in the PPML designs with one fixed effect and with three dense effects. Tighter `GLFixedEffectModels.jl` completed on the Veneto PPML panel only with four or more cores, and with sixteen in two of three blocks; tighter `reghdfe` did not complete the sparse three-effect design with ten million observations. Table C.7 lists every configuration.

C.3 Variation of repeated times

Table C.5 summarizes how far the repeated times behind each median spread. Because each block ran in its own allocation, the repetitions of a configuration ran on different nodes, and the ranges include differences between nodes. `fereg`'s repeated times differed by a median of 9% of their median; the range exceeded 15% in 91 of 289 groups and reached 161%. The largest ranges come from two nodes, which hosted 97 of `fereg`'s default fits. There its times were a median 1.6 times its medians in the same configurations, against at most 1.1 times for the comparators; the evidence does not identify the cause. Blocks on these nodes lower the within-block ratios. All these observations are retained in the medians. The within-block ratios of the article pair each comparator fit with a `fereg` fit on the same node, so that such node differences largely cancel; ratios that differ by less than these ranges should nevertheless not be read as differences between the commands.

Table C.4: Tighter-setting time ratios relative to tighter `fereg`

	Linear regression			PPML		
Package	Completed	Range	Geo. mean	Completed	Range	Geo. mean
Panel A. Simulated configurations						
`reghdfe`	23 of 24	6.0–124	33.0	—	—	—
`ppmlhdfe`	—	—	—	19 of 19	8.1–84.1	23.6
`areg`	21 of 21	2.2–19.7	9.44	—	—	—
`reghdfejl`	24 of 24	4.0–77.2	17.2	—	—	—
`fixest`	29 of 29	1.8–69.6	7.35	24 of 24	1.2–143	4.78
`PyFixest`	29 of 29	1.4–15.9	4.37	24 of 24	0.88–6.2	2.79
`FixedEffectModels.jl`	29 of 29	0.73–21.4	5.34	—	—	—
`GLFixedEffectModels.jl`	—	—	—	24 of 24	1.1–92.8	9.93
Panel B. Veneto panels						
`reghdfe`	3 of 3	131–149	138.6	—	—	—
`ppmlhdfe`	—	—	—	3 of 3	50.4–59.1	54.4
`areg`	3 of 3	11.5–12.5	12.0	—	—	—
`reghdfejl`	3 of 3	37.5–67.2	52.7	—	—	—
`fixest`	5 of 5	10.9–22.9	15.8	5 of 5	15.6–30.7	22.0
`PyFixest`	5 of 5	3.7–9.0	6.09	5 of 5	4.9–10.8	7.50
`FixedEffectModels.jl`	5 of 5	21.7–32.7	24.3	—	—	—
`GLFixedEffectModels.jl`	—	—	—	3 of 5	174–206	190.7

Notes: Tighter stopping settings of table C.1; each comparator's tighter fit is divided by the tighter `fereg` fit (10^{-12}) of the same block. Completed counts configurations with at least one completed time among those scheduled for the package; a configuration's ratio is the median, over its comparison blocks, of the comparator's time divided by `fereg`'s time in the same block. Range and geometric mean are taken over the completed configurations. Values below one favor the comparator. Stopping criteria differ across packages, so equal settings need not deliver equal accuracy. A configuration in which every fit reached the 3,600-second limit is not completed.

Table C.5: Variation of the repeated times

	Default settings				Tighter settings			
Command	Groups	Median	90th pct.	Largest	Groups	Median	90th pct.	Largest
Simulated designs								
`fereg`	247	9.2	68.6	161.0	247	9.7	64.1	147.8
`reghdfe`	24	3.7	10.1	13.2	23	3.7	7.2	12.8
`ppmlhdfe`	19	11.0	27.0	188.5	19	11.4	26.4	187.9
`areg`	21	2.5	8.4	28.5	21	2.7	8.3	26.1
`reghdfejl`	24	19.6	119.0	145.6	24	12.5	41.6	114.4
`fixest`	53	5.1	30.7	47.5	53	5.5	31.3	62.5
`PyFixest`	53	4.4	16.3	45.8	53	4.8	18.1	36.9
`FixedEffectModels.jl`	29	11.7	36.3	121.6	29	9.8	49.5	73.1
`GLFixedEffectModels.jl`	24	5.3	11.5	12.7	24	5.5	12.5	29.2
Veneto panels								
`fereg`	42	5.0	29.1	36.9	42	7.0	25.6	30.7
`reghdfe`	3	0.9	2.3	2.3	3	0.5	0.8	0.8
`ppmlhdfe`	3	6.8	8.9	8.9	3	6.2	8.2	8.2
`areg`	3	1.5	6.6	6.6	3	1.3	9.5	9.5
`reghdfejl`	3	9.9	49.4	49.4	3	11.5	52.5	52.5
`fixest`	10	9.9	27.2	27.2	10	11.7	23.5	23.5
`PyFixest`	10	11.0	21.8	21.8	10	12.0	17.8	17.8
`FixedEffectModels.jl`	5	6.2	12.9	12.9	5	10.9	17.3	17.3
`GLFixedEffectModels.jl`	5	3.6	13.5	13.5	2	15.0	15.0	15.0

Notes: Range of repeated times in percent of their median. A group is one configuration and one comparator family whose five (three above one million observations and on the Veneto panels) comparison blocks all completed; `fereg`'s groups are its fits in the same blocks. The blocks ran in separate allocations, generally on different nodes, so the ranges include differences between nodes as well as between runs.

C.4 All simulated configurations

Table C.6 lists every simulated configuration of the article's comparisons with `fereg`'s median time and each comparator's within-block time ratio at default stopping settings, and table C.7 does the same at tighter settings. The counts and ranges stated in section 7 can be checked against them.

Table C.6: All simulated configurations at default stopping settings

Configuration	`fereg` seconds	`reghdfe`	`areg`	`reghdfejl`	`fixest`	`PyFixest`	Julia
Linear regression							
Worker–firm	0.56	26.6	12.1	11.4	3.61	4.02	2.95
Worker–firm, saved	0.74	22.5		10.3	3.61	12.7	4.00
Worker–firm, robust	0.67	14.4	10.3	9.52	3.62	3.41	2.63
Worker–firm, robust, saved	0.84	20.2		9.07	3.62	11.3	3.62
Worker–firm, clustered	0.66	23.1	10.4	9.76	3.29	3.96	2.63
Worker–firm, clustered, saved	0.86	20.0		10.9	3.22	11.2	3.54
Worker–firm, 1 core	1.25	22.6	10.5	6.09	3.27	4.03	1.32
Worker–firm, 2 cores	0.77	25.1	10.4	8.00	3.24	4.34	2.44
Worker–firm, 8 cores	0.41				4.18	4.11	3.52
Worker–firm, 16 cores	0.36				4.99	3.93	3.72
Worker–firm, 16 cores, clustered, saved	0.58				4.30	15.5	5.57
Two effects, dense, 16 cores	0.31				4.41	2.41	2.32
Three effects, sparse, 16 cores	0.82				10.8	2.63	10.3
Worker–firm, 4 controls	0.26	24.8	12.2	10.1	3.34	2.92	2.50
Worker–firm, 32 controls	0.91	33.3	13.5	13.3	3.61	4.60	2.91
Clustered mobility	0.61	74.0	17.0	32.0	8.21	4.26	6.55
One effect	0.55	5.90	2.15	3.91	1.82	1.38	0.74
Two effects, dense	0.54	11.9	7.48	5.07	2.48	1.66	1.35
Two effects, sparse	0.59	95.3	17.2	45.2	15.6	4.30	6.75
Three effects, dense	0.63	14.6	7.22	5.76	2.34	1.68	1.37
Three effects, sparse	1.35	97.6	9.06	44.2	17.3	2.48	6.52
Worker–firm, 100,000 obs.	0.10	12.1	4.34	11.8	2.55	1.93	3.70
Worker–firm, 300,000 obs.	0.27	15.9	6.75	8.33	2.42	2.48	3.16
Worker–firm, 3,000,000 obs.	1.49	36.9	15.8	14.0	4.25	5.39	3.45
Worker–firm, 10,000,000 obs.	4.79	48.8	16.9	17.6	5.07	8.46	4.10
Three effects, sparse, 100,000 obs.	0.18	39.3	5.35	18.0	7.68	1.86	6.18
Three effects, sparse, 300,000 obs.	0.43	68.0	8.23	20.1	11.3	2.47	8.85
Three effects, sparse, 3,000,000 obs.	3.81	120	10.2	46.8	22.5	2.90	7.48
Three effects, sparse, 10,000,000 obs.	12.1	204	10.7	54.0	27.8	4.61	8.38
Poisson regression (PPML)							
Worker–firm	2.52	33.1			3.20	3.90	8.65
Worker–firm, saved	2.90	29.3			3.06	5.93	7.96
Worker–firm, clustered	2.61	32.3			3.00	3.93	8.43
Worker–firm, clustered, saved	2.93	29.2			2.87	5.91	7.56
Worker–firm, 1 core	6.37	32.1			3.06	3.89	3.11
Worker–firm, 2 cores	3.76	30.1			3.07	3.96	6.44
Worker–firm, 8 cores	1.83				2.59	3.91	12.1
Worker–firm, 16 cores	1.72				3.10	3.52	12.7
Worker–firm, 16 cores, clustered, saved	2.12				2.74	6.48	10.5
Two effects, dense, 16 cores	1.56				2.41	1.76	4.27
Three effects, sparse, 16 cores	7.33				11.9	1.42	34.2
Worker–firm, 4 controls	1.09	22.0			3.00	3.54	5.79
Worker–firm, 32 controls	4.57	40.5			3.34	3.85	9.65
Clustered mobility	4.39	69.7			7.19	2.42	16.1

Configuration	`fereg` seconds	`reghdfe`	`areg`	`reghdfejl`	`fixest`	`PyFixest`	Julia
One effect	2.41	10.6			1.64	1.39	1.54
Two effects, dense	2.49	17.7			1.77	1.44	2.79
Two effects, sparse	4.30	103			11.5	2.83	23.1
Three effects, dense	2.97	16.9			1.82	1.32	2.94
Three effects, sparse	15.1	59.3			9.61	1.27	16.4
Gravity, with zeros	4.82	17.9			2.67	3.04	6.03
Worker–firm, 100,000 obs.	0.28	26.7			3.54	3.20	5.72
Worker–firm, 300,000 obs.	0.85	29.0			2.84	3.20	5.98
Worker–firm, 3,000,000 obs.	7.51	33.9			4.24	4.70	10.3
Worker–firm, 10,000,000 obs.	37.5	23.4			3.32	4.35	9.20

Notes: Default stopping settings and package-default standard errors. `fereg` seconds are the median of its fits in all comparison blocks of the configuration; each other column gives the median, over that package's blocks, of its time divided by the `fereg` time in the same block. Unless noted, one million observations, four cores, 16 controls, and IID (linear) or robust (PPML) standard errors; "clustered" and "saved" denote clustered standard errors and saved fixed effects. For PPML, the `reghdfe` column reports `ppmlhdfe` and the Julia column `GLFixedEffectModels.jl`; for linear models, the Julia column reports `FixedEffectModels.jl`. Blank: not run. Brackets give completed/planned fits when some fits reached the limit.

Table C.7: All simulated configurations at tighter stopping settings

Configuration	`fereg` seconds	`reghdfe`	`areg`	`reghdfejl`	`fixest`	`PyFixest`	Julia
Linear regression							
Worker–firm	0.58	31.6	13.7	15.8	4.62	5.23	4.41
Worker–firm, saved	0.75	28.4		13.9	4.41	13.6	5.15
Worker–firm, robust	0.68	16.0	11.7	13.1	4.41	4.49	3.86
Worker–firm, robust, saved	0.85	25.1		12.2	4.23	12.2	4.61
Worker–firm, clustered	0.67	28.5	11.7	13.7	4.18	5.04	3.82
Worker–firm, clustered, saved	0.88	25.1		14.1	3.81	12.0	4.54
Worker–firm, 1 core	1.29	28.0	12.5	7.32	4.52	5.57	2.38
Worker–firm, 2 cores	0.79	30.8	11.1	11.1	4.58	5.85	3.91
Worker–firm, 8 cores	0.42				4.90	5.10	5.02
Worker–firm, 16 cores	0.37				5.36	5.20	5.39
Worker–firm, 16 cores, clustered, saved	0.59				4.61	15.9	6.85
Two effects, dense, 16 cores	0.31				3.87	2.40	2.28
Three effects, sparse, 16 cores	1.12				27.9	2.42	19.7
Worker–firm, 4 controls	0.27	29.0	15.9	12.2	4.04	3.82	3.36
Worker–firm, 32 controls	0.94	39.5	15.2	17.9	4.64	6.02	4.36
Clustered mobility	0.66	92.8	15.9	49.6	15.0	4.62	12.9
One effect	0.56	5.95	2.17	3.99	1.80	1.36	0.73
Two effects, dense	0.54	12.0	8.17	4.99	2.36	1.68	1.36
Two effects, sparse	0.63	123	15.9	77.2	59.7	5.07	15.7
Three effects, dense	0.63	14.6	7.18	6.00	2.47	1.67	1.43
Three effects, sparse	1.79	97.4	7.71	57.9	53.5	2.69	13.7
Worker–firm, 100,000 obs.	0.11	13.9	4.68	12.9	2.35	2.44	4.26
Worker–firm, 300,000 obs.	0.27	18.6	7.57	10.1	3.08	3.21	4.16
Worker–firm, 3,000,000 obs.	1.53	45.0	18.0	19.3	5.94	7.16	5.99
Worker–firm, 10,000,000 obs.	5.17	60.3	19.7	24.5	7.26	10.6	7.61
Three effects, sparse, 100,000 obs.	0.22	45.2	4.34	21.2	20.9	1.87	9.48
Three effects, sparse, 300,000 obs.	0.53	74.3	6.19	28.3	31.1	2.45	14.9

Configuration	`fereg` seconds	`reghdfe`	`areg`	`reghdfejl`	`fixest`	`PyFixest`	Julia
Three effects, sparse, 3,000,000 obs.	5.31	124	7.21	59.2	61.3	2.94	16.0
Three effects, sparse, 10,000,000 obs.	17.2	>3,600[a]	7.86	62.0	69.6	4.44	21.4
Poisson regression (PPML)							
Worker–firm	3.97	24.2			3.26	3.20	9.28
Worker–firm, saved	4.37	23.0			3.17	4.59	8.94
Worker–firm, clustered	3.41	28.8			3.78	3.79	10.9
Worker–firm, clustered, saved	3.74	26.8			3.60	5.34	9.93
Worker–firm, 1 core	11.3	21.1			3.00	2.98	3.19
Worker–firm, 2 cores	6.28	21.3			3.17	3.11	6.56
Worker–firm, 8 cores	2.61				2.74	3.42	14.6
Worker–firm, 16 cores	2.19				3.60	3.36	17.0
Worker–firm, 16 cores, clustered, saved	2.43				3.40	6.22	15.2
Two effects, dense, 16 cores	2.03				1.90	1.46	4.45
Three effects, sparse, 16 cores	8.29				69.3	2.04	92.8
Worker–firm, 4 controls	1.20	23.0			4.45	3.93	9.50
Worker–firm, 32 controls	9.54	22.8			2.60	2.36	8.32
Clustered mobility	5.87	61.7			19.4	2.58	28.0
One effect	3.77	8.09			1.16	0.88	1.10
Two effects, dense	3.92	11.1			1.17	1.03	2.41
Two effects, sparse	5.76	84.1			143	3.24	50.1
Three effects, dense	4.44	11.2			1.39	0.99	2.70
Three effects, sparse	17.8	59.9			67.5	1.78	43.9
Gravity, with zeros	6.12	14.0			3.16	3.92	9.47
Worker–firm, 100,000 obs.	0.43	19.5			3.21	2.87	6.19
Worker–firm, 300,000 obs.	1.27	22.0			2.85	2.83	6.81
Worker–firm, 3,000,000 obs.	11.8	25.7			4.97	3.90	12.4
Worker–firm, 10,000,000 obs.	40.5	25.7			5.99	5.35	16.5

Notes: Tighter stopping settings (table C.1) and package-default standard errors. `fereg` seconds are the median of its tighter fits in all comparison blocks of the configuration; each other column gives the median, over that package's blocks, of its time divided by the tighter `fereg` time in the same block. Unless noted, one million observations, four cores, 16 controls, and IID (linear) or robust (PPML) standard errors; "clustered" and "saved" denote clustered standard errors and saved fixed effects. For PPML, the `reghdfe` column reports `ppmlhdfe` and the Julia column `GLFixedEffectModels.jl`; for linear models, the Julia column reports `FixedEffectModels.jl`. Blank: not run. Brackets give completed/planned fits when some fits reached the limit. [a] Did not finish within the 3,600-second limit per command.

C.5 Historical: version 0.5.0 on a newer node

The earlier version of this article reported a campaign of `fereg` version 0.5.0 on November 2–3, 2026, with harmonized inference settings, command limits of 3,600 seconds for `reghdfe` and `ppmlhdfe` and 600 seconds for the other commands, and every model and design of a series on one node. That campaign also repeated the configurations of table 5 on one whole 32-core Intel Xeon 6737P node (Granite Rapids), with four cores per command (table C.8). Its times and ratios are retained as a historical hardware comparison; none is combined with the measurements of the article.

On Granite Rapids, `fereg` took 0.49 seconds in the baseline linear worker–firm model and `feppml` 2.22 seconds in PPML. `reghdfe` and `ppmlhdfe` took 15.4 and 21.1 times as long, `areg` 10.6

times, `fixest` 2.6 and 2.1 times, and `PyFixest` 2.2 and 2.1 times. In the same campaign's main comparisons on older nodes, the ratios for the Stata commands were 18.0 and 16.1. Each ratio compares commands within one host; three to five repetitions do not measure variation across hosts.

C.6 Historical: coefficient accuracy and `tolerance()`

The accuracy check (6) bounds the group means of the residuals, not the error in the coefficients (section 5.1). Table C.9 measures the coefficients directly for version 0.5.0. It fits three OLS datasets at tolerances from 10^{-12} to 10^{-6} and compares each fit with a fit of the same data at 10^{-14}: the sparse three-effect and clustered-mobility designs of the simulations and the Veneto largest connected set. At the default tolerance of 10^{-8}, the largest coefficient change was about 1.3×10^{-10} of its standard error, and at 10^{-7} about 8.4×10^{-9}. At 10^{-6}, the largest change reached 2×10^{-6} standard errors, on the Veneto sample, and in the sparse design with three effects the check of the covariate scores (section 6.4) failed, so the command stopped with an error instead of posting estimates. The error message names the loose tolerance and suggests a smaller one. The reported standard errors were unchanged in every completed fit. Tighter tolerances add few iterations down to 10^{-12} (in the sparse design with three effects, 33 instead of 23 at the default), but the reference fit at 10^{-14} needed 526. The default-versus-tighter comparisons of table C.2 give the corresponding evidence for the benchmarked build, against a reference at 10^{-12}.

Table C.8: Version 0.5.0 on a Granite Rapids node (historical)

Inference and output	Median seconds (times as long as `fereg`)			Peak memory (GiB)		
	`fereg`	`reghdfe`/`ppmlhdfe`	`areg`	`fereg`	`reghdfe`/`ppmlhdfe`	`areg`
Linear regression						
IID	0.49	7.55 (15.4×)	5.21 (10.6×)	0.62	0.76	1.07
Robust	0.54	7.61 (14.1×)	5.46 (10.1×)	0.63	0.76	1.07
Clustered by worker	0.54	7.66 (14.1×)	5.59 (10.3×)	0.64	0.76	1.08
IID, saved effects	0.60	8.36 (13.9×) [1/5]	—	0.71	0.76	—
Poisson regression (PPML)						
Robust	2.22	46.9 (21.1×)	—	1.30	1.27	—
Clustered by worker	2.25	46.0 (20.5×)	—	1.32	1.27	—
Robust, saved effects	2.38	47.5 (19.9×) [1/5]	—	1.30	1.27	—

Notes: Historical measurements of `fereg` version 0.5.0 on November 3, 2026, all on one 32-core Intel Xeon 6737P node (Granite Rapids), with the configurations of table 5 and four cores per command. Medians of five fresh-process runs; parentheses divide the comparator's median by `fereg`'s median on this node. These runs used the earlier campaign's harmonized inference settings and limits (appendix C.5); they are not comparable with the article's times and ratios. `areg` cannot save individual fixed effects (—). [1/5]: one of five planned runs; after a saved-prediction check failed, that campaign skipped the remaining repetitions.

Table C.9: Coefficient accuracy and `tolerance()` in version 0.5.0

Data	`tolerance()`						
	10^{-14}	10^{-12}	10^{-10}	10^{-9}	10^{-8}	10^{-7}	10^{-6}
Panel A. Largest coefficient change, in standard errors							
Three effects, sparse	ref.	1×10^{-11}	8×10^{-12}	1×10^{-11}	5×10^{-11}	1×10^{-11}	stopped[a]
Clustered mobility	ref.	5×10^{-12}	2×10^{-11}	1×10^{-11}	3×10^{-11}	2×10^{-10}	2×10^{-8}
Veneto, largest connected	ref.	2×10^{-10}	5×10^{-10}	3×10^{-10}	1×10^{-10}	8×10^{-9}	2×10^{-6}
Panel B. Iterations							
Three effects, sparse	526	33	28	26	23	21	stopped[a]
Clustered mobility	24	21	18	16	14	13	11
Veneto, largest connected	51	43	36	32	28	23	19

Notes: Untimed linear regressions with `fereg` version 0.5.0 and four cores (November 2–3, 2026). Each fit is compared with the fit of the same data at a tolerance of 10^{-14} (ref.): panel A gives the largest absolute change of a coefficient divided by its standard error in that reference fit. The simulated designs have one million observations and 16 controls; the Veneto sample is that of table 7. The default is 10^{-8}. Iterations count the iterations of the absorption method.
[a] The accuracy check (6) failed, and the command stopped with an error instead of posting estimates.

References

Alexander Fischer and Kristof Schröder. Graph preconditioning for high-dimensional fixed effects regression. Working paper, draft of June 2026, GitHub commit bd77e87, 2026. URL https://github.com/py-econometrics/within-paper. Accessed October 28, 2026.

David Chin-Lung Fong and Michael A. Saunders. LSMR: An iterative algorithm for sparse least-squares problems. *SIAM Journal on Scientific Computing*, 33(5):2950–2971, 2011. doi: 10.1137/10079687X.

Yuan Gao, Rasmus Kyng, and Daniel A. Spielman. AC(k): Robust solution of Laplacian equations by randomized approximate Cholesky factorization. *SIAM Journal on Scientific Computing*, 48(3):A1643–A1684, 2026. doi: 10.1137/24M1673577.

Ioannis Koutis, Gary L. Miller, and David Tolliver. Combinatorial preconditioners and multilevel solvers for problems in computer vision and image processing. *Computer Vision and Image Understanding*, 115(12): 1638–1646, 2011. doi: 10.1016/j.cviu.2011.05.013.

Andrea Toselli and Olof B. Widlund. *Domain Decomposition Methods—Algorithms and Theory*, volume 34 of *Springer Series in Computational Mathematics*. Springer, Berlin, 2005. doi: 10.1007/b137868.