PEER REVIEW RECORD

The Constrained LASSO is a Portfolio: A numerical check of the CLA–LASSO identity under constraints

Thomas Schmelzer

Vol. 1 (2026), Art. 10 · https://doi.org/10.68365/jrepstat.2026.10

The decision letters, the reports the authors read, their answers and the editor's signed decisions, as written. The reports come from the journal's review engine. The authors could have kept this record private until they confirmed the proof.

DECISION PATH

2026-10-26 Minor revision
2026-10-26 4 answers
2026-10-26 Accepted

Round 1: Minor revision

SIGNED DECISION

Confirmed and signed by Matthias Templ on 2026-10-26 · ORCID 0000-0002-8638-5276

DECISION LETTER

Thank you for your submission to JRepStat. We are inviting a minor revision. The reviewers' points follow, grouped by weight.

This note verifies numerically that a released portfolio Critical Line Algorithm, wrapped in two thin encodings, computes exact constrained-LASSO solution paths across caps, signs, equality and inequality rows, zero sums and monotone fits, matching LARS, an interior-point solver and three independent path algorithms to rounding, including when there are more predictors than observations. It matters to practitioners who need constrained-LASSO paths that lars_path cannot produce, and to portfolio and statistics readers who can now use one active-set solver across both problems. The reviewers found nothing to fix in the writing itself. We would be glad to see the revised version.

How to read this letter. The decision rests on the items marked required. If a required item seems wrong to you, or cannot be done within this paper, decline it on the revision page and give the reason. A reasoned decline is a legitimate answer, and the editor weighs it. Reviewers can be mistaken. Items marked recommended or optional are advice from the reviewers. Adopt them, decline them with a sentence, or leave them for later work. None of that counts against the paper. The reviewer reports below carry the same marks. The editor who signs this decision may weigh an item differently in the accompanying message, and that message prevails.

REQUIRED (the decision rests on these):

- [Reviewer 2] State the exact objective of the linear program behind the KKT certificate, so a reader can see that a small reported value certifies optimality.
- [Reviewer 3] State that the random designs use fixed seeds recorded in the repository, and add a short determinism attestation: on a fixed platform the discrete outputs (knot counts, which problems are declined, certificate pass or fail) reproduce exactly, while near-rounding residuals vary only in the last digits. Table 4 and Table 5 report knot counts as ranges over three seeds, so the exact figures the text quotes depend on the seeds being pinned.

RECOMMENDED:

- [visualisation] Figure 3 distinguishes its eight largest genera by colour alone against grey, with no direct line labels. Eight categorical hues are hard to separate under common forms of colour vision

deficiency. Draw the eight from a colourblind-safe qualitative palette and label each line at its right-hand end, so the series read without reliance on hue.
- [visualisation] The penalty axis in Figures 1 and 3 reads "0 10^1 10^2" with λ decreasing, which looks like a reversed symlog axis. State the axis transform in the caption (symlog with its linear threshold, reversed) so a reader can place the knots; the units (λ) are labelled, scales are not truncated to exaggerate, and the shaded "beyond the cap" bands in Figure 1 are an honest way to mark the constraint.
- [software-landscape] A single feature table listing which constraint types each package accepts (caps, per-coefficient signs, equality rows, inequality rows, zero sums, p>n, ridge-free) would consolidate what is now spread across Sections 5.5, 5.6 and the related-work paragraph, and would make the placement against comlasso, c-lasso and Gaines et al. legible at a glance. The paper stands without it.
- [Reviewer 1] The diabetes motivating example reads as a modeling endorsement; frame it as an illustration, note the cap u is chosen post hoc, and separate the "low HDL is a risk factor" claim (which [20] supports for incident diabetes) from the inference about disease progression (the outcome here).
- [Reviewer 1] The runtime comparison in Section 5.3 is only against a grid of interior-point solves; add a note on runtime versus the dedicated path algorithms (comlasso, Gaines et al., c-lasso), or state why a cross-language timing is not comparable, so the "when to use it" guidance is complete.
- [Reviewer 1] The "sum to one" example is described "as a synthetic control does", but classic synthetic control also constrains weights to be nonnegative; clarify that this is a negative-weight variant so the analogy is not misread.
- [Reviewer 1] State that the sCD14 and other real-data paths are computational demonstrations and that no biological inference is intended.
- [Reviewer 2] Name the single published test whose path fails the segment check and report what the check returns there.
- [Reviewer 2] Report the hostile and scale tables over more than three seeds, keeping the worst case.
- [Reviewer 2] State the relaxation values of epsilon and note that the certificate, not the extrapolation, guarantees the returned path.
- [Reviewer 2] Clarify how the gap against LARS is read where the minimiser may not be unique, and which five of the Table 2 problems the ported Gaines algorithm matches.
- [Reviewer 3] Document the R toolchain that executes comlasso 0.2. The text says R 4 will not install a package built for R 3, then says the check loads it unchanged, which leaves unclear which R interpreter runs the comparison and whether a package built for R 3 behaves identically when sourced under a later R.
- [Reviewer 3] Report the timing methodology behind the cost ratios in Section 5.3 (number of runs and the spread), instead of "the fastest of several runs".
- [Reviewer 3] Provide a mirror or archival fallback for the sCD14 source (coda4microbiome 0.2.4), since the fetch depends on CRAN retaining that version; the SHA-256 check confirms identity but not availability.

OPTIONAL:

- [statistical-rigor] The simulated and hostile-problem claims rest on three seeds each with worst-case reporting (Tables 4 and 5). Worst-of-three is a conservative summary and no claim depends on a sampling distribution, so this is adequate for the paper's purpose. A larger seed count would let the dependence of the gap on conditioning be characterised as a distribution rather than a single worst case, if a later paper wanted that.
- [Reviewer 1] A predictive comparison (for example cross-validation) showing the constrained fits generalise would strengthen the applied case, though it is beyond the scope of this note.

REVIEWS

Report Reviewer 1, recommends accept

Written by the journal's review engine in the role of the application reviewer, and read by the editor before the decision. How the engine works is on the peer review page.

This is a verification note, and judged as one it does its job well. The underlying identity between the constrained LASSO path and the CLA path is proven elsewhere [14]; the work here is to show that a port-

folio solver, through two encodings (Propositions 1 and 2), actually reproduces the constrained-LASSO knots on a wide spread of constraint types, and to check every path against references that know nothing of portfolios. The test matrix is the strength of the paper. Caps, signs, equality rows, inequality rows, zero sums, monotone orderings, a pinned observation and a sum-to-one constraint are each exercised, on simulated and real data, with LARS as the knot-by-knot reference where it applies and an interior-point solve at segment midpoints elsewhere, backed by a direct optimality check that holds along whole segments rather than only at midpoints. The agreement is at the level of rounding. The hostile cases (near-duplicate columns, correlation 0.99999, p>n with a zero sum) and the deliberate decline of the 1e-6 duplicate are the kind of adversarial testing I want to see in a paper whose whole claim is numerical. The honesty about where references add a ridge, and the point that the CLA needs none for p>n, is both correct and genuinely useful to someone choosing a tool for microbiome-scale zero-sum problems.

My concerns are about the applied framing, not the computation. The opening diabetes example is doing a lot of rhetorical work in Section 1 and Figure 1, and an applied reader may take it as a claim that the capped fit is the better model of the biology. The cap value u = 396.1 is chosen as half the largest least-squares coefficient, which is a post hoc choice, and the robustness sentence (caps up to 637 keep HDL negative) is reassuring but still conditional on this dataset. Separately, the reference [20] supports low HDL as a risk factor for incident diabetes, whereas the response in this dataset is a continuous measure of disease progression. The sign argument slides between those two outcomes. The cleanest fix is to present the example as an illustration of what a cap does to offsetting coefficients, and to qualify the biological reading as suggestive.

On guidance to users, the cost comparison in Section 5.3 is careful and precise, but it compares only against a grid of interior-point solves. The paper also runs comlasso, the Gaines et al. algorithm and c-lasso for accuracy, so a reader deciding between tools will want to know how the CLA compares to those on runtime, not only to an IPM grid. A single paragraph, even one that explains why a fair cross-language timing is not available, would round out the "when to use it" section. The "sum to one" example invokes synthetic control, and since the classic synthetic control estimator also constrains weights to be nonnegative, a one-line clarification that this is the negative-weight variant will stop the analogy being over-read. Finally, the sCD14 path is a good stress test on real compositional data, and a sentence stating that it is a computational demonstration with no biological conclusion attached will keep it from being cited as a finding about inflammation.

The reproducibility posture is strong as described: lockfiles pinning Python 3.12 and package hashes, a single script that writes every number quoted in the text, pinned and checksummed R and Julia environments, a checksum on the fetched sCD14 data that is not redistributed, and a stated tolerance regime that accounts for hardware-dependent last digits. I did not execute the artifacts in this pass, so I assess them as declared and leave the run itself to the reproducibility lens. None of the items below blocks acceptance; they improve how an applied reader will receive the paper.

Numbered suggestions to the authors:

1. RECOMMENDED Reframe the diabetes example in Section 1 and Figure 1 as an illustration of what the cap does to offsetting coefficients. State that u = 396.1 is chosen after seeing the least-squares fit, and separate the HDL risk-factor claim (which [20] supports for incident diabetes) from the inference about progression, which is this dataset's outcome.
2. RECOMMENDED In Section 5.3 and the "when to use it" discussion, add how the CLA compares on runtime to the dedicated path algorithms you already run (comlasso, Gaines et al., c-lasso), or state why a fair cross-language timing is not available, so the cost guidance is not read as comparing only against an IPM grid.
3. RECOMMENDED Clarify in Section 5.1 that the "sum to one" example allows negative weights, unlike the classic synthetic control estimator, so the synthetic-control analogy is not misread.
4. RECOMMENDED Add a sentence where sCD14 is introduced stating that it and the other real-data paths are computational demonstrations and that no biological inference is intended.

5. OPTIONAL A predictive comparison, such as cross-validation of the constrained versus unconstrained fits, would strengthen the applied argument that constraints buy interpretability without cost, though it sits beyond the scope of this note.

Report Reviewer 2, recommends minor

Written by the journal's review engine in the role of the methodology reviewer, and read by the editor before the decision. How the engine works is on the peer review page.

This is a validation note with a clear and narrow job. A theoretical identity between the constrained LASSO path and the Critical Line Algorithm path is proved elsewhere [14]. Here the author poses constrained regressions as portfolios through two encodings, runs them through a released portfolio CLA, and asks whether the resulting paths match independent references. The question is well posed and the design answers it in the right way. The two encodings in Propositions 1 and 2 are stated cleanly and their proofs are correct as given. The split into nonnegative long and short parts, the simplex rescaling with a riskless asset, and the multiplier bookkeeping all follow from the classical bound-penalty equivalence, and the paper is careful to say that the propositions borrow nothing from [14] beyond that equivalence.

The strongest methodological choice is the optimality certificate. Comparing a path only against another numerical solver bounds the error at the solver's own tolerance, and the paper says so. Evaluating the subgradient conditions of problem (1) directly, at every knot and across every whole segment, with the multipliers chosen by a linear program and the residual then evaluated exactly, removes that ceiling. Checking the segment whole rather than at its midpoint alone is the right refinement, since a path can be right at both ends and at the midpoint and still cut across a knot. This is where I want one more sentence. The certificate's force rests on what the linear program optimises. The text says the multipliers are chosen to "satisfy the conditions best", which a reader cannot pin down. If the program minimises the largest violation across all the conditions jointly, subject to the sign constraint on the active-row multipliers, then a small reported value is a genuine optimality certificate and a true violation cannot be hidden. Please state that objective explicitly, because the whole evidential weight of the paper sits on it.

Two further points concern how much a reader can trust the stress results. The hostile and scale tables are drawn with three seeds and report the worst case. Three draws is thin for a claim that an algorithm declines or stays exact on a whole family of ill-conditioned designs. Re-running over more seeds and keeping the worst would make the boundary between "traced exactly" and "declined" look stable rather than incidental. Relatedly, the text claims that one of the published tests contains a path that fails the segment check, which would be a welcome demonstration that the check has teeth. I could not locate which test this is or what the check reports there. Naming it and giving its residual would turn an assertion about the check into evidence for it.

The treatment of degenerate starts is handled honestly as an implementation device, with an extrapolation in epsilon and a certificate afterwards. Because the extrapolation is a two-point Richardson step and its affineness "cannot be checked in advance", the reader should be told which epsilon values are used and reminded that the returned path is guarded by the certificate and not by the extrapolation itself. On uniqueness, the paper is appropriately cautious, testing optimality and not uniqueness where p exceeds n. One consequence deserves a line. Where the minimiser is not unique, LARS and the CLA may trace different paths of minimisers, so a gap against LARS is not by itself a defect. The hostile section already argues "both paths are right" from the conditioning bound, and a short statement folding that logic into the non-unique case, together with naming which five Table 2 problems the ported Gaines algorithm matches, would close the remaining ambiguity.

On reproducibility I can only review what is declared, since the scripts were not provided to run in this pass. As declared the artifacts are unusually complete: lockfiles with versions and hashes, a single script that writes every quoted number, a checksummed download of the microbiome data, legacy R and Julia references pinned and ported with departures marked, and CI. This is the standard a validation paper should meet, and it is met on paper. The competing-interests statement is precise about what the author wrote and what the references are. None of the above changes the paper's conclusion, which the evidence supports. The asks are about letting a reader verify the certificate and the stress claims with the same confidence the author has.

Numbered suggestions to the authors:

1. REQUIRED State the exact objective of the linear program used in the optimality check, specifically that it minimises the largest violation across the KKT conditions of (1) jointly, under the nonnegativity constraint on the active-row multipliers, so that a small reported value certifies optimality and a real violation cannot be masked.
2. RECOMMENDED Name the one published test whose path fails the segment check, and report what the check returns there, so the check's discriminating power is shown rather than asserted.
3. RECOMMENDED Re-run the hostile problems of Table 4 and the scale problems of Table 3 over more than three seeds, keeping the worst case, so the line between "traced exactly" and "declined" reads as stable.
4. RECOMMENDED Give the relaxation values of epsilon used for degenerate starts and add one sentence that the certificate, not the two-point extrapolation, guarantees the returned path.
5. RECOMMENDED Add a sentence on how the gap against LARS is interpreted where the minimiser need not be unique, and name which five of the Table 2 problems the ported Gaines algorithm matches when started at the exact first knot.

Report Reviewer 3, recommends minor

Written by the journal's review engine in the role of the software reviewer, and read by the editor before the decision. How the engine works is on the peer review page.

This is a reproducibility-first paper, and on that axis it is well built. There is a single entry point (`make figures`) that computes every path, runs every check, and writes every figure, table and quoted number, so the text and the code cannot drift apart silently. The environment is captured with lockfiles that pin Python 3.12 and every package by version and hash, the cross-language checks are pinned (Julia 1.12, a checksummed comlasso commit, c-lasso 1.0.11), and the whole thing runs in CI. The non-redistributable sCD14 data are fetched with a SHA-256 check rather than shipped. The competing-interests statement is clear about the author maintaining cvxcla and separates that from the external references. The hardware caveat is handled honestly: residuals near rounding differ in the last digits across CPUs, tolerances are set between 1e-12 and 1e-10, and timings are given only as ratios. This is close to what I would want from a numerical-check submission.

The gap I would close before this is treated as fully reproducible is around randomness. Tables 4 and 5 report knot counts as ranges over three seeds (for example 43–45, 92–228, 50–78), which tells me the discrete structure of the path moves with the seed. For the specific numbers in the text to be reproducible, the seeds have to be fixed and recorded in the repository, and a reader needs to know that. A short determinism statement would settle it: on a fixed platform with the recorded seeds, the knot counts, the set of declined problems, and the certificate outcomes reproduce exactly, and only the near-rounding residuals vary. The paper already supplies the hardware half of this, so the missing half is the seed and discrete-output half.

Two smaller reproducibility points concern the external checks that carry real evidential weight. The comlasso check agrees with the CLA to 4.3e-15, which is convincing, but the code-availability paragraph first says R 4 will not install a package built for R 3 and then says the check loads it unchanged. Which R runs the comparison, and how a package built for R 3 is loaded under it, should be stated so the reader can rebuild that leg of the check. Separately, the cost ratios in Section 5.3 rest on "the fastest of several runs," which is a weak basis for a timing claim even when the claim is secondary to correctness. Reporting how many runs and the spread would make those ratios as defensible as the rest of the numbers.

On the sCD14 fetch, pinning by SHA-256 guards against the file changing but not against it disappearing from CRAN. A mirror or an archived copy, or a sentence naming a fallback, would keep that one real-data check reproducible over time. None of this touches the core correctness evidence, which is internally strong and triangulated across LARS, an interior-point solver and three independent path algorithms, with the KKT conditions evaluated along whole segments rather than at midpoints alone.

Numbered suggestions to the authors:

1. REQUIRED Record the seeds for the random designs (Wide, Correlated, Duplicate, Monotone, Sum to one, and the simulated zero-sum problems) in the repository and say so in the paper, and add a one-sentence determinism attestation that on a fixed platform the knot counts, the declined problems

and the certificate outcomes reproduce exactly while near-rounding residuals vary only in the last digits.

2. RECOMMENDED State which R interpreter runs comlasso 0.2 and how a package built for R 3 is loaded under it, so the comlasso comparison can be rebuilt.
3. RECOMMENDED Give the timing methodology for the cost ratios in Section 5.3 (number of runs and the spread) instead of reporting the fastest of several runs.
4. RECOMMENDED Add a mirror or archival fallback for coda4microbiome 0.2.4 so the sCD14 check survives the CRAN version being removed.

REVISION, RECEIVED 2026-10-26

Authors confirm each required point as done or give their reason for declining it. Further answers are voluntary, because the review compares every revision with the previous version and the next round's reports say what remains open.

> REQUIRED State the exact objective of the linear program behind the KKT certificate, so a reader can see that a small reported value certifies optimality.
>
> **Done.**

> REQUIRED State that the random designs use fixed seeds recorded in the repository, and add a short determinism attestation: on a fixed platform the discrete outputs (knot counts, which problems are declined, certificate pass or fail) reproduce exactly, while near-rounding residuals vary only in the last digits. Table 4 and Table 5 report knot counts as ranges over three seeds, so the exact figures the text quotes depend on the seeds being pinned.
>
> **Done.**

> RECOMMENDED The diabetes motivating example reads as a modeling endorsement; frame it as an illustration, note the cap u is chosen post hoc, and separate the "low HDL is a risk factor" claim (which [20] supports for incident diabetes) from the inference about disease progression (the outcome here).
>
> **Done.** We reframed it slightly to avoid this impression

> OPTIONAL A predictive comparison (for example cross-validation) showing the constrained fits generalise would strengthen the applied case, though it is beyond the scope of this note.
>
> **Declined.** It's indeed beyond the scope of this note

The authors' full response letter, as uploaded, follows as Appendix A.

Round 2: Accepted

SIGNED DECISION

Signed by Matthias Templ on 2026-10-26 · ORCID 0000-0002-8638-5276

All three reviewers recommend acceptance. Both required points of the first round are met: Section 5.1 now states the objective of the linear program behind the optimality certificate, and the availability section records the seeds with a test that checks the discrete results. The editor accepted the final version after the journal reproduced it. One correction remains for the proof, a stray full stop in Section 5.5.

DECISION LETTER

Congratulations, your manuscript has been accepted for publication in JRepStat. Production begins now.

Where it is not on file yet, we still need the following for production.

- a valid ORCID iD for every author
- a captured computational environment (dependencies/lockfile/container)
- a signed determinism attestation for the entry-point command

REVIEWS

Report Reviewer 1, recommends accept

Written by the journal's review engine in the role of the application reviewer, and read by the editor before the decision. How the engine works is on the peer review page.

Of the four recommended points and one optional point I raised, three are adopted and two are answered with a reason.

The two I weighed carefully are the declines. On runtime against the dedicated path algorithms (my point 2), the paper now states in Section 6 why a cross-language timing would measure the languages and the ports rather than the algorithms, and directs the user to Table 6 to choose by the constraints each tool accepts and then to time candidates on their own problem. That is the alternative I named, and it convinces me: comlasso read under R 4, a Julia 0.6 port and a c-lasso grid with a fixed ridge are not comparable stopwatches, and a number would mislead more than it informs. On the predictive comparison (my point 5), the decline as beyond the scope of a verification note is right, and the revised framing of the diabetes and sCD14 examples removes the suggestion that any predictive claim is being made.

The applied framing I was concerned about is now clean. The diabetes example reads as an illustration of what a cap does to offsetting coefficients, with u = 396.1 marked as chosen after seeing the fit, the incident-diabetes risk-factor claim from [22] separated from the progression outcome, and the cap robustness described as a property of these data. The sum-to-one example now distinguishes itself from the nonnegative-weight synthetic control estimator, and sCD14 is labelled a computational demonstration in both the text and Figure 3.

One change the revision made in answer to another reviewer is worth noting as newly visible, since it affects how an applied reader should read the headline claim: the extra seeds moved one p>n draw of Zero-sum, signs to a worst KKT residual of 1.0×10^-9 near an interpolating fit at cond(X) ≈ 6×10^15, and the abstract and Section 5.5 now state this rather than claiming rounding-level residuals there. This is honest and correct, and it does not weaken the applied case. The paper's claim is about where each tool's path ends and whether it declines rather than answers wrongly, and that claim is now stated with the right precision.

I recommend acceptance.

Numbered suggestions to the authors:

(none open)

Report Reviewer 2, recommends accept

Written by the journal's review engine in the role of the methodology reviewer, and read by the editor before the decision. How the engine works is on the peer review page.

Of the one required point and four recommended ones I raised, all are met.

The required point, the objective of the linear program behind the KKT certificate, is now stated in Section 5.1 at the level the evidential weight demands: the program minimises a single number, the largest violation across the optimality conditions of (1) taken jointly under the nonnegativity constraint on the active-row multipliers, with the violation re-evaluated exactly at the multipliers found. The reader can now see why a small reported value certifies optimality to that size and why a real violation cannot be masked. This closes the one point the decision rested on.

The re-run over more seeds did more than tighten a table. Extending Table 3 to three seeds, Tables 4 and 5 to ten, and keeping the worst case moved the boundary between "traced exactly" and "declined", and it exposed three further defects in cvxcla 2.7.0, two in the segment-optimality check and one in the plain homotopy, now fixed in 2.7.1 with regression tests and reported in Section 1 alongside the earlier two. The paper no longer claims rounding-level residuals where a p > n path nears an interpolating fit, and the abstract and Section 5.5 now carry the 10^-9 figure. This is the right outcome of the ask, and the reporting of it is accurate to what the seeds showed.

The remaining three are settled. The segment check now names test_segment_check_sees_between_samples and gives both numbers, the exact optimality residual of zero at the ends and the midpoint against the 0.25 the segment check returns, which demonstrates the check has teeth. The epsilon relaxation schedule (10^-6, 10^-7, 10^-8 times the coefficient scale) is stated in Section 4 with the sentence separating what the extrapolation proposes from what the certificate guarantees. The non-uniqueness reading is folded into Sections 5.4 and 5.6, and the five Table 2 problems

the ported Gaines algorithm matches at the exact first knot are named. One production-level item remains for the proof, a stray full stop in Section 5.5.

Numbered suggestions to the authors:

1. [editor_fix] In Section 5.5, join “with cond(X) near 10^16. and the interior-point solver agrees to 4.2×10^-6” into one sentence, removing the stray full stop before “and”.

Report Reviewer 3, recommends accept

Written by the journal’s review engine in the role of the software reviewer, and read by the editor before the decision. How the engine works is on the peer review page.

The required point is settled. The seeds are now recorded in `python/make_figures.py` (seed 7 for Table 2, 7–9 for Table 3, 7–16 for Tables 4 and 5), the determinism attestation is in the Code and data availability section, and `test_deterministic` runs the Table 3 and Table 4 problems twice and asserts the discrete outputs match. That closes the gap I flagged: the knot counts, the declined problems and the certificate outcomes are now pinned, and the text says which digits are allowed to move.

My three recommended points are also met. The comlasso leg now states R 4.6.1 runs the check and that the R 3.2 package is read with `lazyLoad` from serialized code at a pinned, checksummed commit, with the point that comlasso carries no compiled code, so the loading path is reconstructible. Section 5.3 gives the timing methodology (warm-up run, then 11 timed runs, 3 at the largest size, ratio of medians with the paired-run spread within 8%). The sCD14 fetch now falls back from CRAN to the CRAN archive to a dated Posit snapshot, with the tarball digest pinned per source and the inner data file checked byte for byte, so the real-data check survives a CRAN version removal.

One thing worth recording for the editor rather than the author: the extra seeds the author drew in response to another reviewer surfaced three defects in cvxcla 2.7.0 and shifted the boundary between traced and declined (Duplicate 10^{-6} under the cap now declined on 2 of 10 seeds, and a worst-case p>n residual of 1.0×10^{-9}). The revision reports all of this in the text and the abstract rather than smoothing it over, and the fixes ship in 2.7.1 with regression tests. From the reproducibility angle this strengthens the paper, since the released version now matches the reported numbers.

Numbered suggestions to the authors: (none)

APPENDIX A

The authors' full response letter to round 1

As uploaded with the revision received 2026-10-26. Its own pages follow.

Response to the reviewers

Manuscript: The Constrained LASSO is a Portfolio: A numerical check of the CLA–LASSO identity under constraints **Journal:** Journal of Reproducible Statistics **Decision:** minor revision

I thank the editor and the three reviewers for careful reports. Every required and recommended item is adopted below except one, Reviewer 1's runtime comparison, which is answered with a reason in the text. One recommended item, more seeds, found something new: three defects in `cvxcla`, fixed in release 2.7.1, and two further declined draws. Both are reported under R2.2. Section and table numbers refer to the revised manuscript.

Required

R2 (required). State the exact objective of the linear program behind the KKT certificate. Adopted (Section 5.1). The text now states what the program does. Its unknowns are the multipliers κ of $A\beta = b$ and η of the active rows of $C\beta \le h$. It minimises one number τ, the largest violation of all the optimality conditions taken jointly, subject to $\eta \ge 0$. Each condition $\varphi_i(\kappa, \eta) \le 0$ enters as the row $\varphi_i(\kappa, \eta) \le \tau$. On a segment the conditions at both ends enter one program. The optimum is zero exactly when β is optimal. The violation is then re-evaluated exactly at the multipliers found. A small reported value therefore certifies optimality to that size, and a real violation cannot be masked, since no choice of multipliers brings it below the program's optimum. This is what the code has always done (`best_multipliers` in `python/make_figures.py`); the paper now says so.

R3 (required). Record the seeds and add a determinism attestation. Adopted (Code and data availability; README). Every random design is drawn inside `python/make_figures.py` from a seed recorded there. Seed 7 is used for Table 2, seeds 7–9 for Table 3, and seeds 7–16 for Tables 4 and 5. The paper now states that, on a fixed platform, the knot counts, the set of declined problems and the certificate outcomes reproduce exactly, and that only residuals near rounding vary, in their last digits. A new test, `test_deterministic`, runs the problems of Tables 3 and 4 twice and asserts that the discrete outputs are identical.

Recommended

[visualisation] Figure 3: colourblind-safe palette and direct labels. Adopted. The eight genera are drawn in the Okabe–Ito palette and each line is labelled at its right-hand end ($\lambda = 0$). The labels are spread apart only as far as needed to avoid overlap, so the series can be read without relying on hue.

[visualisation] State the axis transform of Figures 1 and 3. Adopted. Both captions, and the axis label of Figure 3, now say that the horizontal axis is λ decreasing to the right on a symmetric-log scale. It is logarithmic above λ_max/50 and linear below, so that the end of the path at $\lambda = 0$ is shown.

[software-landscape] One feature table. Adopted as Table 6 (Section 6), placed there so that Tables 2–5 keep their numbers. It lists, for `lars_path`, `c-lasso`, comlasso, Gaines et al., CLASSO and `cvxcla`, whether each accepts caps, per-coefficient signs, equality rows, inequality rows, zero sums and $p > n$, and whether it does so for $p > n$ without a ridge. CLASSO's column is taken from its paper, since its code is not released.

R1.1. The diabetes example: an illustration, u chosen post hoc, incident diabetes versus progression. Adopted (Section 1). The example is now introduced as an illustration of what a cap does to offsetting coefficients, "not offered as a better model of the disease". The text says that $u = 396.1$ is chosen after

seeing the fit. It separates the two claims: [20] supports low HDL as a risk factor for *incident* diabetes, while the response here is *progression*, so that finding "suggests a negative sign but does not establish one". The robustness to the cap is described as "a property of these data, not a finding about HDL".

R1.2. Runtime against comlasso, Gaines et al. and c-lasso. Answered with a reason (Section 6), not with timings. None of the three runs here as its authors would run it today. comlasso is R code built for R 3.2 and read under R 4. The algorithm of Gaines et al. is our port of Julia 0.6 code. `c-lasso` returns its own grid with a fixed ridge. A timing across R, Julia and Python would measure the languages and our ports as much as the algorithms. The text says so, and points a user to Table 6 to choose by the constraints each tool accepts, then to time candidates on their own problem.

R1.3. "Sum to one" versus synthetic control. Adopted (Section 5.1). The text now says that, unlike the classic synthetic control estimator (Abadie, Diamond and Hainmueller, 2010, now cited), which also requires nonnegative weights, the weights here may be negative, and that adding the signs would be one more bound.

R1.4. sCD14 and the other real-data paths are computational demonstrations. Adopted, where sCD14 is introduced (Section 5.1) and in the caption of Figure 3: "a computational demonstration on real data. No biological inference is intended, and none should be drawn from its path."

R2.1. Name the test whose path fails the segment check, and what the check returns. Adopted (Section 5.1). It is `test_segment_check_sees_between_samples`. With $p = 1$, $y = 1$ and the cap $\beta \le ½$, the exact path has knots at $\lambda = 1$ and $\lambda = ½$. The single segment from $(\lambda, \beta) = (5/4, 0)$ to $(1/4, ½)$ misses both knots, yet meets the path at its two ends and its midpoint. There the optimality conditions hold exactly (largest violation 0), while the segment check returns 0.25, relative to $\|c\|_\infty$. Both numbers are now written into the text by the script.

R2.2. Tables 3 and 4 over more than three seeds, keeping the worst case. Adopted. One correction first: Table 3 was drawn from a single seed, not three. Table 3 now uses three seeds per size. Tables 4 and 5 now use ten (seeds 7–16). Knot counts are reported as ranges and every other column as the worst case.

The extra seeds did what the reviewer intended and moved the boundary between "traced exactly" and "declined":

- *Duplicate 10^{-6} with the cap* is now declined on 2 of the 10 draws. The cause is the same as without the cap: on those draws both near-duplicate columns become free before the cap binds, and the free block is singular to working precision. The earlier sentence "under the cap they never are" was true of three seeds only and has been replaced. Every path that is traced still meets the optimality conditions to rounding.
- *Correlated* failed on one draw (seed 14). This was not a principled decline but defects in `cvxcla` 2.7.0, which the author maintains. Two were in the check that a bound held along a segment is still optimal. With no active rows its feasibility LP had no variables and raised an error, and its noise floor ignored the benchmark term q, so round-off at $\lambda = 0$, where Σw and q cancel, read as a wrongly held bound. Testing the fix on other platforms exposed a third, in the plain LASSO homotopy, which is not used in the paper: when two events coincide, as when nearly collinear columns swap, it lost the second. All three are fixed in `cvxcla` 2.7.1, which the revision uses, each with a regression test. On seed 14 *Correlated* is now traced and meets the optimality conditions to rounding, and over the ten seeds it reports 43–49 knots and a residual of at most 4.1×10^{-13}. Section 1 reports the three defects as it reports the two fixed in 2.6.0.
- On *Wide* ($p > n$) the worst KKT residual over ten seeds is 1.0×10^{-9}, against 2.7×10^{-13} over three. It occurs on one draw of *Zero-sum, signs*, at the very end of the path ($\lambda \approx 10^{-4}$ λ_max), where the fit

almost interpolates and cond(X) ≈ 6×10^{15}. Everywhere else the residuals stay at rounding, and the interior-point objective is never better than the path's by more than 7.5×10^{-12}, relative. The abstract and Section 5.5 now say this, and no longer claim rounding-level residuals there.

R2.3. The values of ε, and that the certificate guarantees the path. Adopted (Section 4). In `cvxcla` 2.7.1, ε is 10^{-6} times the coefficient scale $\|c\|_\infty$ / $\max_j G_{jj}$, then 10^{-7} and 10^{-8} times it if the certificate fails. A path is accepted at 10^{-9} relative to $\|c\|_\infty$. The text adds: "The extrapolation only proposes a path; the certificate is what guarantees the path returned."

R2.4. The gap against LARS where the minimiser is not unique, and the five Gaines problems. Adopted (Sections 5.4 and 5.6). Where the minimiser is not unique, LARS and the CLA may trace different paths of minimisers. A gap between them is no defect as long as both meet the optimality conditions, and the certificate, not the agreement, is what each path is held to. The five Table 2 problems on which the ported Gaines algorithm, started at the exact first knot, matches the CLA are *Plain*, *Box*, *Cholesterol*, *Zero-sum* and *sCD14*. The Julia check now writes these names into the text.

R3.1. Which R runs comlasso 0.2, and how a package built for R 3 is loaded. Adopted (Code and data availability). R 4.6.1 runs the check, locally and in CI. The package is not installed, since R 4 refuses packages built before R 4.0. The script fetches the zip from the authors' repository at a pinned commit, checks its SHA-256, and reads the functions from the package's serialized code with `lazyLoad`, unchanged. comlasso ships no compiled code, so nothing in it depends on the R 3 build. Its one dependency, Rglpk 0.6.5.1, comes from a dated CRAN snapshot.

R3.2. Timing methodology. Adopted (Section 5.3; README). Each step is now run once to warm up and then timed 11 times (3 times for the p = 500 grid, which takes minutes). The paper quotes the ratio of the medians together with the range of the ratio over paired runs, instead of the fastest of several runs. The ratios barely move: about 90× the path at p = 100, 340× at p = 500, and 14–110× on the constrained problems of Table 2, with every paired run within 8% of the median ratio.

R3.3. Fallback for the sCD14 source. Adopted. `python/fetch_scd14.py` tries CRAN's `src/contrib`, then the CRAN archive, then the Posit Package Manager snapshot of 2026-09-01. Posit repacks the tarball, so each source's tarball digest is pinned separately. The script also checks the SHA-256 of `data/sCD14.rda` inside the tarball, which we verified to be byte-identical across CRAN and Posit. The data that reach the analysis are therefore pinned whichever source serves them.

Optional

[statistical-rigor] More seeds to characterise the gap as a distribution. Partly taken up through R2.2 (ten seeds). Characterising the dependence of the gap on conditioning as a distribution is left, as the editor suggests, to later work.

R1.5. A predictive comparison (cross-validation). Declined as beyond the scope of a verification note, as the reviewer also says. The paper claims nothing about predictive performance, and the revised framing of the diabetes and sCD14 examples (R1.1, R1.4) removes any suggestion that it does.