Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Strict Checks, Slow Iterations, Large Null Spaces: A Numerical Audit of Fused LEMNA Regression on EMBER

Kottapalli Deepak Varma

kottapalli.deepakvarma2005@gmail.com

R.M.D Engineering College

Research Article

Keywords: Explainable Artificial Intelligence (XAI); Malware Detection; EMBER Dataset; LEMNA; Fused Regression; Alternating Direction Method of Multipliers (ADMM); Numerical Conditioning; Rank Deficiency; Sparse Optimization; Reproducibility

Posted Date: November 10th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11675951/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Strict Checks, Slow Iterations, Large Null Spaces: A Numerical Audit of Fused LEMNA Regression on EMBER

A Bounded Technical Note with Synthetic Controls

Kottapalli Deepak Varma
R.M.D. Engineering College

November 2026

Abstract

LEMNA explains a security classifier by fitting a local mixture of simple regressions under a fused (total-variation) constraint. I adapted it to EMBER features, with 2,381 features and 1,000 samples in each local neighbourhood, and asked whether its regression step can be solved reliably. The fusion graph has 605 edges, so the matrix inside the solver has nullity of at least 776. On two saved inputs, the stacked matrix has a measured numerical rank of 1,020 and 681 out of 2,381 columns. On 16 saved subproblems that had failed my convergence rule, jittered Cholesky, LSMR and LSQR passed the residual tests in 4/16, 4/16 and 2/16 cases. No returned solution stayed inside the TV budget under a zero-tolerance check. For the 16 saved Cholesky outputs I show that this check is stricter than the residual tests: the excess is bounded by $\sqrt{m}$ times the split gap $\|D\beta - z\|_2$, and in all 16 cases it was inside that bound (at most 7.5×10^{-4}). On two inputs, a larger budget did not help, but 1,000 or 2,000 iterations gave a residual certificate, and an independent OSQP solve reached objectives 1.3% and 0.10% below the 200-iteration result. Synthetic tests show that rank deficiency alone need not block convergence. I do not establish a cause, and I report no attack, calibration or official-test experiment.

Keywords: explainable AI, malware classification, EMBER, LEMNA, ADMM, fused regression, rank deficiency, numerical conditioning, reproducibility.

1 Introduction

When a malware detector flags a file, analysts often want to know why. Explainers such as SHAP, LIME and LEMNA try to answer that by showing which features drove a prediction [1, 2, 3]. Before anyone studies whether such explanations are stable or can be manipulated, the explainer itself has to be computed reliably. This note checks that first step for one method.

LEMNA fits a mixture of local linear regressions with a fused-lasso-type penalty [3]. In my EMBER adaptation there are many more coefficients ($d = 2{,}381$) than local samples ($n = 1{,}000$), and the fusion graph constrains only a small part of the coefficient space. The question is simple: can the resulting convex subproblem be solved to a strict standard that I set in advance?

What this note contributes.

1. An audit of the EMBER fusion graph, with a rank bound for the β-update system (Sections 4–5).
2. A comparison of three ways to solve the β update on 16 saved failed subproblems, with the same tolerances, reporting residual convergence and returned-β feasibility separately (Section 7).
3. A bound that links the raw-TV excess to the split gap, checked on all 16 saved Cholesky solutions, plus budget, iteration-cap and reference-solve checks on two inputs (Section 7).

4. A synthetic study that separates rank deficiency from budget tightness and from the strictness of the raw-TV check (Section 8), and a table of the checks that are still open (Section 10).

Scope. This is not an attack study, and it does not reproduce the original LEMNA code. I worked only with feature vectors from public EMBER training data. No PE file was modified or executed, and I did not use the official test set.

2 Background and Related Work

Guo et al. introduced LEMNA for security applications [3]. LIME [2] and SHAP [1] are the two usual comparison methods. Galadima et al. compared LEMNA, LIME and SHAP on EMBER and DREBIN for fidelity, robustness and agreement [4]. Their work judges explanation quality. Mine asks a narrower question: can the LEMNA regression step be solved in my EMBER adaptation? Related work on trust includes sanity checks for Android-malware interpreters [5] and a survey of explainable ML for security [6]. Slack et al. showed that LIME and SHAP can be manipulated [7]. Pierazzi et al. point out that a change in feature space need not be a valid executable [8], so I make no claims at the executable level.

On the optimisation side I use ADMM splitting [9] for the fused-lasso constraint [10], and the iterative least-squares solvers LSQR [11] and LSMR [12]. The dataset is EMBER [13].

3 Problem Formulation

Take one mixture component with sample weights w_i. The β step solves

$$\min_{\beta} \ \frac{1}{2} \frac{\sum_i w_i \, (x_i^\top \beta - y_i)^2}{\sum_i w_i} \quad \text{s.t.} \quad \|D\beta\|_1 \le S, \qquad S = 1.0, \tag{1}$$

where $D \in \mathbb{R}^{m \times d}$ is the fusion (edge-difference) matrix. Let $X_w = \operatorname{diag}\big(\sqrt{w_i / \textstyle\sum_j w_j}\big) X$, and let y_w be the response scaled the same way, so the objective equals $\frac{1}{2}\|X_w\beta - y_w\|_2^2$. ADMM splitting [9] adds a variable $z = D\beta$ with the constraint $\|z\|_1 \le S$, and a scaled dual variable u. The z update projects $D\beta + u$ onto the ℓ_1 ball of radius S. The β update is then the least-squares problem

$$\min_{\beta} \ \left\| \begin{bmatrix} X_w \\ \sqrt{\rho}\, D \end{bmatrix} \beta - \begin{bmatrix} y_w \\ \sqrt{\rho}\, (z - u) \end{bmatrix} \right\|_2^2. \tag{2}$$

I write A for the stacked matrix in Eq. (2). Its normal equations are $G\beta = X_w^\top y_w + \rho D^\top (z - u)$ with $G = X_w^\top X_w + \rho D^\top D$. The original implementation forms G and adds a small diagonal jitter δI before a Cholesky factorisation. LSMR and LSQR work directly on the rectangular stacked system, without jitter.

Acceptance rule. The primal and dual residuals are $r = D\beta - z$ and $s = \rho D^\top (z_k - z_{k-1})$. A *residual certificate* means both pass the absolute-plus-relative tolerances, which I did not change. Because z is projected onto the TV ball and $D\beta$ is not, a residual certificate does not guarantee $\|D\beta\|_1 \le S$. So I also report *joint strict acceptance*: a residual certificate and $\|D\beta\|_1 \le S = 1.0$ for the returned β, with no extra tolerance on TV.

A bound on the excess. Here $z \in \mathbb{R}^m$ with $m = 605$ (one entry per fusion edge), the split is $D\beta = z$, and z is projected so that $\|z\|_1 \le S$. The triangle inequality and Cauchy–Schwarz give

$$\|D\beta\|_1 \ \le \ \|z\|_1 + \|D\beta - z\|_1 \ \le \ S + \sqrt{m}\, \|D\beta - z\|_2. \tag{3}$$

So a returned β can exceed the budget by up to $\sqrt{m}$ times its split gap, even when that gap is within the primal tolerance. A zero-tolerance test $\|D\beta\|_1 \le S$ is therefore stricter than the residual rule, and the two can disagree at convergence without any solver error. I report $\|D\beta\|_1$ only as an extra check on the returned β, not as a standard ADMM stopping rule.

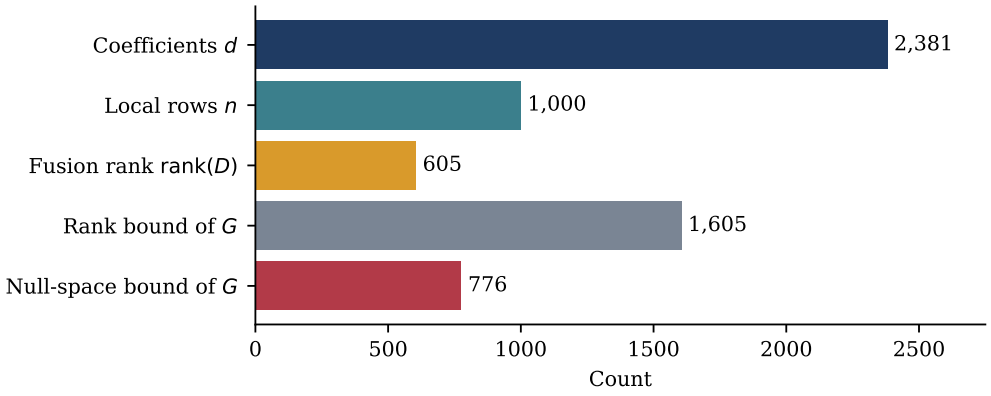


Figure 1: Dimensions that bound the rank of G. The null-space bound of 776 follows from $d-(n+\operatorname{rank} D)$.

4 EMBER Representation and Fusion Graph

I use EMBER 2018, feature version 2, with the official parser ordering, which gives $d = 2{,}381$ features [13]. The fusion edge list has 605 unique undirected edges, with no duplicates and no self-loops. Its non-trivial parts are paths on 256, 256 and 96 vertices (255, 255 and 95 independent edges). The graph touches 608 coordinates and leaves 1,773 isolated, so

$$\operatorname{rank}(D) = 605, \qquad \operatorname{nullity}(D) = 2{,}381 - 605 = 1{,}776.$$

I built the local neighbourhoods in raw feature space by coordinate-median replacement, then applied a `StandardScaler` fitted on model-training records only. This was an implementation pilot on 1,800 training records, not the planned full-study partition.

5 Rank Analysis

The weighted design term $X_w^\top X_w$ has rank at most $n = 1{,}000$, and $\rho D^\top D$ has rank 605. So

$$\operatorname{rank}(G) \le 1{,}000 + 605 = 1{,}605, \qquad \operatorname{nullity}(G) \ge 2{,}381 - 1{,}605 = 776. \tag{4}$$

In exact arithmetic G is singular, and its exact 2-norm condition number is infinite. Figure 1 shows the dimensions. This is a structural bound, not a measured rank. I therefore do not quote a finite condition number for G. Instead, Section 7 (Table 5) reports the numerical rank and the condition number of the retained spectrum of the stacked matrix A on two inputs. Adding δI makes G positive definite, but it acts as a ridge penalty, so the Cholesky branch solves a slightly different problem.

What this does and does not show. If G is rank deficient, the least-squares solution is not unique and the plain normal equations cannot be factorised directly. That does not by itself prove that ADMM cannot converge. Section 8 shows that in a synthetic setting, $n < d$ can make convergence *easier*, because 1,773 coordinates are untouched by the fusion term and can soak up the response.

6 Experimental Setup

The 16 subproblems are the saved inputs from the two failed records of the five-record pilot (7 from record 2 and 9 from record 4). For each one I checked the SHA-256 hash, the required arrays, that all values were finite, and the expected shapes. All solvers used $\rho = 0.1$, $S = 1.0$, absolute tolerance 10^{-6}, relative tolerance 10^{-5} and at most 200 ADMM iterations. The Cholesky branch uses jitter $\delta = 10^{-10}$.

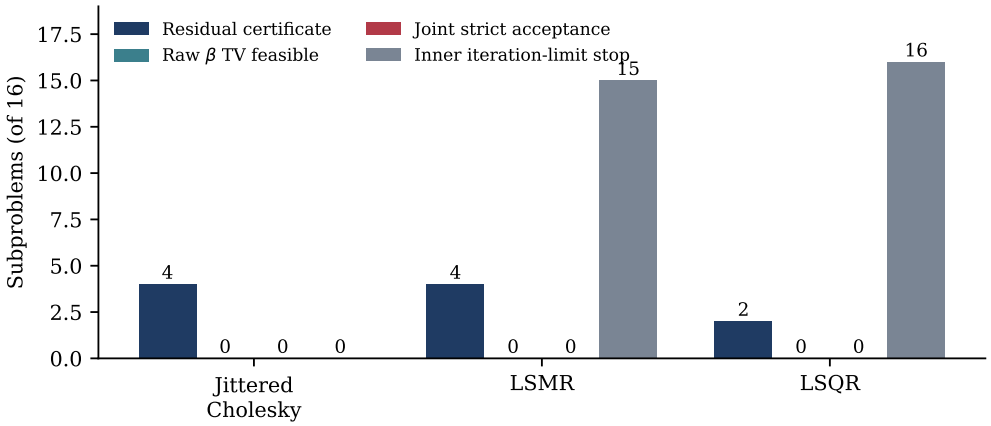


Figure 2: Outcomes by solver. Residual certificates occur in a few cases, but none returns a β within the TV budget, so joint acceptance is zero.

LSMR and LSQR use the rectangular system without jitter, with an inner limit of 300 iterations. I did not retrain any classifier, re-run any explainer or touch any official-test record. Runtimes are medians with minimum and maximum, computed from the saved JSON output.

7 Results

I compare the three ways of solving the β update on the 16 saved subproblems (Tables 1–2, Figures 2–3).

Solver	Residual cert.	Raw β TV feasible	Joint strict	Inner iter.-limit stops
Jittered Cholesky	4/16	0/16	0/16	0/16
LSMR	4/16	0/16	0/16	15/16
LSQR	2/16	0/16	0/16	16/16

Table 1. Outcomes on the 16 persisted failed subproblems.

Solver	Median (s)	Min (s)	Max (s)
Jittered Cholesky	18.40	7.12	26.48
LSMR	315.09	155.49	377.17
LSQR	312.73	278.74	386.20

Table 2. Runtime per subproblem.

Table 1 and Figure 2 show the outcomes, and Table 2 and Figure 3 show the runtimes. Three things stand out. First, no method reached joint strict acceptance. Second, a residual certificate and TV feasibility are different conditions: 4/16 Cholesky runs, 4/16 LSMR runs and 2/16 LSQR runs passed the residual tests, yet none returned a β with $\|D\beta\|_1 \leq 1$. Third, the last inner solve of LSMR and LSQR hit its 300-iteration limit in 15/16 and 16/16 cases, so their failures may come from the inner iteration budget as much as from the problem. Cholesky solves its inner system directly, so it has no inner limit, and it was the fastest; but it solves the jitter-modified system.

How this relates to the five-record pilot. The 16 subproblems come only from the two records that had already failed the original gate. So 0/16 joint acceptance does not contradict the 3/5 pilot pass rate.

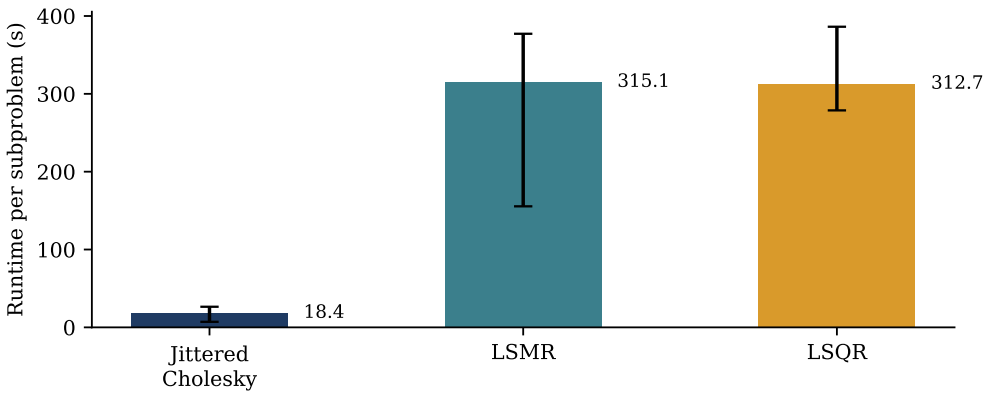


Figure 3: Median runtime with min–max bars. The iterative solvers are roughly 17 times slower than jittered Cholesky, and their final inner solves mostly reach the inner iteration limit.

Split-gap audit on the 16 EMBER subproblems

For all 16 subproblems I took the saved returned β and z arrays. These come from the Cholesky/jitter ADMM branch (the original failure-callback output, with jitter 10^{-10}). The β and z arrays of LSMR and LSQR were not saved, so this audit covers the Cholesky outputs only. I checked whether the raw-TV excess $\|D\beta\|_1 - S$ lies within the bound of Eq. 3, with $S = 1.0$ and $m = 605$. It does in all 16 cases (Table 3). The excess runs from 1.30×10^{-4} to 7.48×10^{-4} (median 2.8×10^{-4}), which is at most 0.075% of S. The split gap runs from 6.8×10^{-6} to 5.0×10^{-5} (median 2.0×10^{-5}), and the bound from 1.7×10^{-4} to 1.2×10^{-3}. The projected variable meets its constraint to rounding error: $\|z\|_1$ ranges from 0.999999999999998 to 1.0000000000000024, and the largest excess over S is 2.4×10^{-15}. The bound follows from $\|z\|_1 \le S$, so passing it is consistent with a correct projection (it is necessary, not sufficient). For Cholesky, the 0/16 raw-TV result is therefore explained by the finite primal gap: the returned β is inside the budget up to solver tolerance. I could not run the same audit for LSMR and LSQR, so their 0/16 raw-TV results are not covered by this check. Nor does it explain the missing residual certificates (12/16 for Cholesky, 12/16 for LSMR and 14/16 for LSQR), which remain the real unresolved failures.

Iteration, budget and reference-solve diagnostics (two subproblems)

I ran three more checks on two of the 16 inputs only: the first saved subproblem of record 2 and the first of record 4 (files `record_2_failed_001` and `record_4_failed_001`). I used the Cholesky branch, $\rho = 0.1$ and tolerances $10^{-6}/10^{-5}$. For the independent reference solve I used CVXPY 1.9.3 with OSQP 1.1.3 on the same weighted objective, with auxiliary variables $t \ge \pm D\beta$ and $\sum_i t_i \le S$; the solver settings are saved in the results files. Results are in Table 4.

Raising S up to 10 did not give a residual certificate within 200 iterations, so a tight budget does not explain the missing certificates on these two inputs. Allowing 1,000 or 2,000 iterations did give a certificate in both cases, which points to slow convergence under the 200-iteration cap rather than non-convergence. The OSQP solves were optimal, with objectives slightly below the 200-iteration Cholesky result. That fits an iterate that has not converged yet, and it may also reflect the jitter, which I did not separate out. These checks cover 2 of 16 inputs and only the Cholesky branch.

Measured numerical rank. For the same two inputs I computed the singular values of the stacked matrix A (not of G, whose condition number would be squared) with SciPy's `gesvd` driver on a single thread. I counted ranks above $\tau = \max(m,n)\,\varepsilon\,\sigma_{\max}$ and above relative thresholds of 10^{-12} and 10^{-10}. All three thresholds gave the same rank (Table 5).

On both inputs the numerical rank is well below the structural bound of 1,605, so the null space has

Subproblem	Excess $\|D\beta\|_1 - S$ ($\times 10^{-4}$)	Bound $\sqrt{m}\,\|D\beta - z\|_2$ ($\times 10^{-4}$)	Excess $\leq$ bound
`record_2_failed_001`	1.60	2.02	yes
`record_2_failed_002`	1.41	1.76	yes
`record_2_failed_003`	1.99	2.47	yes
`record_2_failed_004`	1.58	2.13	yes
`record_2_failed_005`	1.36	1.67	yes
`record_2_failed_006`	1.48	1.99	yes
`record_2_failed_007`	2.25	2.89	yes
`record_4_failed_001`	5.59	6.83	yes
`record_4_failed_002`	7.48	9.11	yes
`record_4_failed_003`	6.04	7.43	yes
`record_4_failed_004`	5.47	6.70	yes
`record_4_failed_005`	5.52	7.01	yes
`record_4_failed_006`	3.39	4.27	yes
`record_4_failed_007`	4.86	6.10	yes
`record_4_failed_008`	1.30	12.25	yes
`record_4_failed_009`	4.85	5.68	yes

Table 3. Bound check on the 16 saved Cholesky/jitter ADMM outputs (16/16 satisfy it).

Diagnostic	Record 2 input	Record 4 input
200 iterations, $S = 1$	no certificate	no certificate
200 iterations, $S \in \{1, 2, 5, 10\}$	none of 4 certified	none of 4 certified
1,000-iteration cap, $S = 1$	certified	certified
2,000-iteration cap, $S = 1$	certified	certified
OSQP reference objective	2.1069×10^{-4}	1.5621×10^{-3}
Cholesky (200 it.) objective minus OSQP	2.76×10^{-6} (1.3%)	1.56×10^{-6} (0.10%)
OSQP status	optimal	optimal

Table 4. Two-subproblem diagnostics (selected subset, not all 16 inputs). The reference problem is the same convex subproblem; $\beta = 0$ is always feasible.

dimension at least 1,361 (record 2) and 1,700 (record 4). The retained part of the spectrum is only moderately conditioned (about 10^5), so the ill-conditioning comes from the null space, not from nearly-zero retained singular values. I did not test why the rank is lower than the bound.

8 Controlled Synthetic Study

Purpose and limits. I cannot vary the saved EMBER subproblems, so I tested three possible causes on synthetic data. This is *not* EMBER data, and it does not prove what happened in the real failures. It only shows which explanations are possible.

Design. Dimension $d = 2{,}381$. A path graph with the same component sizes (256, 256, 96; 605 edges) placed on the first 608 coordinates. A Gaussian design $X \in \mathbb{R}^{n \times d}$, unit weights, and $y = X\beta^\star + 0.1\,\varepsilon$, where $\beta^\star$ is piecewise constant and scaled to total variation $\mathrm{TV}^\star \in \{0.5, 2.0\}$. ADMM uses $\rho = 0.1$, $S = 1.0$, jitter 10^{-10} and Boyd-style absolute and relative residual tolerances (10^{-6}, 10^{-5}) [9]. I used three seeds per setting (two for the longest run). The exact tolerance formula in the EMBER code may

Input	$\sigma_{\max}$	Smallest computed σ	Smallest retained σ	Rank	Condition (retained)
Record 2	32.19	6.1×10^{-18}	1.6×10^{-4}	1,020	2.0×10^{5}
Record 4	14.64	1.2×10^{-19}	4.7×10^{-5}	681	3.1×10^{5}

Table 5. Numerical rank of the stacked matrix A (1605×2381) on two inputs; SVD time about 2.3 s each. Rank is identical at τ, $10^{-12}\sigma_{\max}$ and $10^{-10}\sigma_{\max}$ (rank is threshold-dependent in general). The condition number is $\sigma_{\max}$ divided by the smallest *retained* singular value; the retained values shown are derived from the saved condition numbers, because they were not saved separately. The smallest computed value includes numerical-null directions.

Regime	Iter. cap	Residual cert.	Strict ($\|D\beta\|_1 \le 1$)	Median iters	$\|D\beta\|_1 - S$
$n = 1000$, any TV*	200	3/3	3/3	1	≈ -1
$n = 3000$, TV$^\star = 0.5$	200	3/3	0/3	58	$+2.7 \times 10^{-4}$
$n = 3000$, TV$^\star = 2.0$	200	0/3	0/3	200	$+0.94$
$n = 3000$, TV$^\star = 2.0$	1000	0/2	0/2	1000	$+0.57$ to $+0.59$

Table 6. Synthetic regimes (medians; $n = 3000$, TV$^\star = 0.5$ is identical at 200 and 1000 iterations). With a relaxed check $\|D\beta\|_1 \le 1.001$, the second row passes 3/3.

differ.

Findings.

1. *$n < d$ is trivially solvable here.* With 1,000 rows, the 1,773 isolated coordinates alone can fit y exactly while the path coordinates stay constant. So $\|D\beta\|_1 \approx 0$ and ADMM stops at iteration 1. This answer is degenerate, because all the attribution sits on unfused features. Rank deficiency alone does *not* cause non-convergence.
2. *A strict raw-TV check can fail at convergence.* With $n > d$ and TV$^\star < S$, the budget is still active (fitting noise inflates TV). The residual certificate is met in about 58 iterations, yet $\|D\beta\|_1$ exceeds S by about 3×10^{-4}, and the gap $\|D\beta - z\|$ is about 3×10^{-5}. A strict check with no tolerance rejects solutions that pass the residual rule and are feasible to within solver tolerance.
3. *A tight budget slows ADMM sharply.* When TV* is twice S, no run met the certificate in 200 or 1,000 iterations. TV keeps falling (from about 1.95 at 200 iterations to about 1.58 at 1,000), so this looks like slow convergence, not a hard barrier.

Checking the bound (Eq. 3). For the $n = 3000$, TV$^\star = 0.5$ runs, $\|z\|_1 = S$ to machine precision and the split gap is about 2.5–2.7×10^{-5}. The observed excess $\|D\beta\|_1 - S$ (2.2–2.8×10^{-4}) is below $\sqrt{m}\,\|D\beta - z\|_2$ (6.2–6.5×10^{-4}) in all three runs (Table 7). So the small violation is fully explained by the finite primal residual.

9 Discussion

My evidence supports a narrow claim for the EMBER adaptation. First, for the Cholesky branch the strict zero-tolerance raw-TV check is not informative. On all 16 saved solutions the excess is at most 7.5×10^{-4} and inside the split-gap bound, so its 0/16 result reflects the finite primal gap (Table 3). I could not audit LSMR and LSQR the same way. Second, the unresolved failure is the missing residual certificate: 12/16 Cholesky runs, 12/16 LSMR runs and 14/16 LSQR runs did not meet the primal and dual tolerances within 200 iterations. On the two inputs I tested, a larger budget did not help but a larger iteration cap did (Table 4), so the failures there look like slow ADMM convergence under the cap. The synthetic study makes "rank deficiency causes the failure" hard to defend. The measured ranks (1,020 and 681 of 2,381) show a large null space, but I cannot say whether it contributes to the

Seed	Cert. iter.	$\|z\|_1$	$\|D\beta\|_1$	Excess	$\|D\beta - z\|_2$	$\sqrt{m}\|D\beta - z\|_2$
0	68	1.000000	1.000217	2.17×10^{-4}	2.59×10^{-5}	6.37×10^{-4}
1	51	1.000000	1.000276	2.76×10^{-4}	2.50×10^{-5}	6.16×10^{-4}
2	58	1.000000	1.000266	2.66×10^{-4}	2.65×10^{-5}	6.52×10^{-4}

Table 7. Synthetic check of Eq. 3 ($n = 3000$, $\mathrm{TV}^\star = 0.5$, $S = 1$, $m = 605$). The bound holds in every run.

slowness, because the larger-neighbourhood test that would change the rank was not completed. None of this extends automatically to the other 14 inputs or to LSMR and LSQR, which I did not run beyond 200 iterations.

10 Diagnostic Status and Remaining Checks

Diagnostic	What it tests	Status
Excess TV and the bound of Eq. 3 on the 16 inputs	Whether raw-TV infeasibility is explained by the finite primal gap	Done, 16/16 (Table 3)
Budget sweep in S	Whether a tight $S = 1.0$ explains the missing certificates	Done on 2 inputs (Cholesky): no effect at 200 it.
Iteration cap sweep	Whether failures persist at 1,000–2,000 iterations	Done on 2 inputs (Cholesky): certified; LSMR/LSQR not run
Looser tolerance	Sensitivity to the tolerance	Not run
Independent reference solve (OSQP)	Optimal objective and feasibility of the same convex subproblem; $\beta = 0$ is always feasible	Done on 2 inputs
Numerical rank by SVD of A	Measured rank and conditioning versus the structural bound 1,605	Done on 2 inputs (Table 5)
Larger neighbourhood	Whether increasing n improves measured rank, conditioning and convergence; full column rank must be measured, not assumed	Not completed

11 Limitations

The synthetic study uses Gaussian designs and a path graph on the first 608 coordinates. These differ from real EMBER neighbourhoods (median-replacement masks and a classifier-output response), so the study tests possible mechanisms only. The 16 subproblems come from two records of a small pilot, and I rebuilt them from deterministic training-only settings rather than capturing them in the original run. I did not establish equivalence with the original LEMNA implementation, and the EMBER fusion graph is an adaptation that may not reflect the true feature dependencies. I make no matched held-out comparison of SHAP, LIME and LEMNA quality. The β and z arrays of LSMR and LSQR were not saved, so the split-gap audit covers the Cholesky branch only. Independent review of the adaptation, the jitter and the convergence rule is still pending. An earlier dense-SVD attempt did not finish in the time budget, so the larger-neighbourhood test and the feature ablation were not completed. A later single-threaded SVD took about 2.3 s per input, but I have not rerun those two tests. The numerical rank was measured on two inputs only. The iteration, budget and reference-solve checks cover 2 of the 16 inputs and only

the Cholesky branch, so they may not hold for the other inputs or for LSMR and LSQR. I report no attack, calibration, 30-record or official-test result. A SHA-256 checksum of the official test file was read during an environment audit; no test records were parsed or used.

12 Reproducibility and Availability

Machine-readable results, scripts and pinned dependencies are in an accompanying package. EMBER itself is not redistributed; it must be obtained from its maintainers, and checksums should be verified independently. The synthetic-study scripts accompany this note. Code and results: https://github.com/Deepakvarna02/fused-lemna-ember-audit (commit `b85ff99337342d1c405a8586bb600520e9ee170d`). Archived on Zenodo: https://doi.org/10.5281/zenodo.24070079.

13 Conclusion

For the EMBER fused-LEMNA adaptation I audited, the β-update system is rank deficient in exact arithmetic (nullity at least 776), and no solver I tested met the joint strict rule on the 16 saved failures. On the 16 saved Cholesky solutions, the raw-TV excess is at most 7.5×10^{-4} and inside the split-gap bound, so the zero-tolerance TV check mainly reflects the finite primal gap. Most runs did not reach the residual certificate within 200 iterations. On two inputs, a larger budget did not help, while 1,000 or 2,000 iterations gave a certificate, which suggests slow convergence under the iteration cap. Synthetic controls show that rank deficiency alone does not prevent convergence. On two inputs, the numerical rank of the stacked matrix is 1,020 and 681 of 2,381 columns, far below the structural bound, with a retained-spectrum condition number of about 10^5. The link between this null space and slow convergence, the behaviour of the other 14 inputs and of LSMR and LSQR at larger caps, and independent review are still open. Nothing here supports claims against LEMNA in general.

Appendix A: Pilot Context

Five-record pilot. Records 0, 1 and 3 passed the original strict LEMNA gate; records 2 and 4 failed (about 1,269 s in total). I kept the failures in the record.

Classifier smoke test. On a 720-record training-derived holdout (1,800 records, stratified 60/40), accuracy and ROC-AUC were: Random Forest 0.9264 / 0.9842, LightGBM 0.9625 / 0.9925, MLP 0.9236 / 0.9592. These are small-sample values. They are not evidence of generalisation and not official-test results.

Disclosures

AI assistance: AI tools helped with drafting, editing and code. I am responsible for the scientific claims. **Funding:** The author received no specific funding for this work. **Competing interests:** The author declares no competing interests. **Ethics:** I used only public feature data, and no malware was executed.

References

[1] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30*, pages 4765–4774, 2017.

[2] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proc. ACM SIGKDD*, pages 1135–1144, 2016. doi:10.1145/2939672.2939778.

[3] Wenbo Guo, Dongliang Mu, Jun Xu, Purui Su, Gang Wang, and Xinyu Xing. LEMNA: Explaining deep learning based security applications. In *Proc. ACM SIGSAC Conf. on Computer and Communications Security (CCS)*, pages 364–379, 2018. doi:10.1145/3243734.3243792.

[4] K. Galadima, P. Anthony, R. Ploszek, S. Balogh, P. Zajac, and M. Homola. Evaluating the trustworthiness of LEMNA versus LIME and SHAP in explainable malware detection. In *Proc. 18th Int. Conf. on Agents and Artificial Intelligence (ICAART)*, volume 4, pages 3353–3364, 2026. doi:10.5220/0014429400004052.

[5] Ming Fan, Wenying Wei, Xiaofei Xie, Yang Liu, Xiaohong Guan, and Ting Liu. Can we trust your explanations? sanity checks for interpreters in Android malware analysis. *IEEE Trans. Inf. Forensics Secur.*, 16:838–853, 2021. doi:10.1109/TIFS.2020.3021924.

[6] Azqa Nadeem, Daniël Vos, Clinton Cao, Luca Pajola, Simon Dieck, Robert Baumgartner, and Sicco Verwer. SoK: Explainable machine learning for computer security applications. In *Proc. IEEE European Symp. on Security and Privacy (EuroS&P)*, pages 221–240, 2023. doi:10.1109/EuroSP57164.2023.00022.

[7] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods. In *Proc. AAAI/ACM Conf. on AI, Ethics, and Society (AIES)*, pages 180–186, 2020. doi:10.1145/3375627.3375830.

[8] Fabio Pierazzi, Feargus Pendlebury, Jacopo Cortellazzi, and Lorenzo Cavallaro. Intriguing properties of adversarial ML attacks in the problem space. In *Proc. IEEE Symp. on Security and Privacy (S&P)*, pages 1332–1349, 2020. doi:10.1109/SP40000.2020.00073.

[9] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, 2011. doi:10.1561/2200000016.

[10] Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *J. R. Stat. Soc. Ser. B*, 67(1):91–108, 2005. doi:10.1111/j.1467-9868.2005.00490.x.

[11] Christopher C. Paige and Michael A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Softw.*, 8(1):43–71, 1982. doi:10.1145/355984.355989.

[12] David Chin-Lung Fong and Michael A. Saunders. LSMR: An iterative algorithm for sparse least-squares problems. *SIAM J. Sci. Comput.*, 33(5):2950–2971, 2011. doi:10.1137/10079687X.

[13] Hyrum S. Anderson and Phil Roth. EMBER: An open dataset for training static PE malware machine learning models, 2018. arXiv:1804.04637.