

FrontierGuard DEA

Toward Auditable and Reproducible Efficiency, Productivity, and Network Analysis

Vamsi Chinda

Technical Report 0.1

Software version 0.1.0a1

7 August 2026

Non-peer-reviewed alpha-preview technical report

OPEN RESEARCH SOFTWARE / MIT LICENSE /
[GITHUB.COM/CHINDAVAMSI/FRONTIERGUARD-DEA](https://github.com/chindavamsi/frontierguard-dea)



Status statement. *This document is a technical report, not a peer-reviewed journal article. It describes the design, current implementation, validation evidence, limitations, and research programme of FrontierGuard DEA version 0.1.0a1. The package is an alpha research preview, and its implemented scientific contracts remain provisional except the general acyclic-network model, which is experimental. Neither this report nor the local test suite establishes parity with every DEA implementation, statistical validity for an empirical study, or correctness for every admissible dataset.*

Suggested citation before a DOI is assigned

Chinda, V. (2026). *FrontierGuard DEA: Toward auditable and reproducible efficiency, productivity, and network analysis (Technical Report 0.1; software version 0.1.0a1). FrontierGuard DEA.*
<<https://github.com/chindavamsi/frontierguard-dea>>

When an archival DOI becomes available, the DOI-bearing record and the exact software release identifier should replace the provisional citation above.

The repository includes the rendered [PDF edition](#). Rebuild it from this source in the locked development environment:

```
uv sync --frozen --all-groups
uv run --frozen python scripts/render_whitepaper_pdf.py
```

Abstract

Data envelopment analysis (DEA) encompasses a large family of nonparametric models for estimating relative efficiency and productivity. Reproducible use of these models requires more than solving a linear programme: analysts must make explicit choices about orientation, returns to scale, score normalization, reference technology, slack treatment, zero values, infeasibility, alternate optima, numerical tolerances, and the interpretation of intermediate production flows. Those choices are often distributed across software calls, informal notes, and solver output, making analyses difficult to inspect or reproduce.

FrontierGuard DEA is an open-source Python project intended to make these choices explicit in data contracts, model classes, result schemas, and post-solve evidence. Version 0.1.0a1 implements provisional radial envelopment, weighted additive, directional-distance, slack-based, free-disposal-hull, radial super-efficiency, multiplier and selected cross-efficiency, assurance region, economic-efficiency, adjacent-period Malmquist, scale-efficiency, and serial relational-network routines. It also includes an experimental relational model for graph-defined acyclic production networks. Results preserve solver status, score conventions, data fingerprints, reference intensities, targets, and independently recomputed feasibility or identity residuals. A reporting layer can export strict JSON, tidy CSV, a method card, and a checksum manifest.

The report defines the package's present scientific boundary and explains why an audit residual is useful but not sufficient evidence of methodological correctness. It presents the architecture, model families, network decompositions, local validation strategy, and a reproducible example. It also sets out a research roadmap centred on a public cross-language conformance corpus, independently reviewed numerical oracles, real-user studies, and version-specific software citation. FrontierGuard is not presented as a complete collection of DEA methods or as superior to established software. Its proposed contribution is a common, inspectable workflow in which model conventions and numerical evidence travel with the result.

Keywords: data envelopment analysis; efficiency measurement; productivity; network DEA; multi-stage DEA; reproducible research; scientific software; linear programming

1. Problem and motivation

DEA estimates the relative efficiency of decision-making units (DMUs) by constructing a piecewise frontier from observed inputs and outputs. Its intellectual foundations include Farrell's decomposition of productive efficiency (Farrell, 1957), the constant-returns formulation of Charnes, Cooper, and Rhodes (Charnes et al., 1978), and the variable-returns formulation of Banker, Charnes, and Cooper (Banker et al., 1984).

The apparent simplicity of a scalar efficiency score can conceal consequential modelling decisions:

- input contraction and output expansion use different raw radial factors;
- constant, variable, non-increasing, and non-decreasing returns to scale define different technologies;
- first-phase radial efficiency does not generally exhaust input and output slacks;
- efficiency is relative to the declared reference technology, which may be different from the evaluated sample;
- non-radial and directional models use score conventions that should not be silently converted into radial scores;
- alternate optimal intensity or multiplier vectors can produce the same objective value;
- a valid score may coexist with an undefined stage ratio in a network model;
- solver tolerances and data scaling affect the numerical evidence used to accept a solution; and
- descriptive frontier estimates do not, by themselves, carry causal or probabilistic interpretations.

The methodological literature has expanded across radial, non-radial, productivity, network, environmental, robust, and inferential approaches (Cook and Seiford, 2009; Emrouznejad and Yang, 2018). Software necessarily implements subsets of this literature and adopts particular conventions. The resulting diversity is valuable, but it also makes movement between tools difficult when assumptions and outputs are not represented in a common, machine-readable form.

FrontierGuard addresses a narrower problem: **how can a DEA computation retain enough information for another researcher to identify the mathematical contract, reconstruct key feasibility conditions, and determine which validation gates have and have not been satisfied?** It does not attempt to replace methodological judgement, independent replication, or domain-specific study design.

2. State of the field and scope of the contribution

2.1 From radial frontiers to structured production systems

Several strands of DEA research motivate the package structure:

Strand	Representative foundation	Consequence for software design
Radial efficiency	Farrell (1957), Charnes et al. (1978), Banker et al. (1984)	Orientation, returns to scale, score convention, slacks, and reference set must be explicit.

Strand	Representative foundation	Consequence for software design
Pareto--Koopmans and additive models	Charnes et al. (1985)	Non-radial inefficiency should retain its own units and efficient value.
Directional distance	Chambers et al. (1996)	Improvement direction is data, not an implicit option; signed distance can matter.
Slack-based efficiency	Tone (2001)	Fractional scores, positivity requirements, and Charnes--Cooper recovery require dedicated auditing.
Non-convex technology	Deprins et al. (1984)	Free Disposal Hull (FDH) should not be approximated by a convex DEA formulation.
Ranking and multiplier restrictions	Andersen and Petersen (1993) , Sexton et al. (1986) , Thompson et al. (1990)	Infeasibility, alternate multipliers, cross-appraisal, and weight assumptions must be visible.
Productivity change	Caves et al. (1982) , Färe et al. (1994)	Period alignment, four distance terms, index direction, and decomposition identities must be recorded.
Network production	Kao and Hwang (2008) , Tone and Tsutsui (2009) , Kao (2014)	Intermediate products and stage connections should be modelled directly rather than flattened without explanation.
Statistical and robust frontiers	Simar and Wilson (1998) , Cazals et al. (2002) , Simar and Wilson (2007)	Deterministic scores are not uncertainty estimates; bootstrap and partial-frontier procedures require separate contracts.

These strands are not interchangeable. For example, a two-stage production network is not the same object as a statistical second-stage regression, and a second optimization for radial slacks is not either of those. FrontierGuard uses separate public types for such distinctions.

2.2 Existing software and the intended interoperability role

Established DEA capabilities are distributed across R, Python, Julia, MATLAB, and specialist applications. FrontierGuard's dated [ecosystem inventory](#) records an initial documentation review of selected packages. That inventory is a project-maintained comparison, not an independently reviewed systematic study. It is expected to change as packages add features and as FrontierGuard's cross-language tests identify convention differences.

Version 0.1.0a1 does not claim:

- implementation of every published DEA method;
- feature parity with every established package;
- superior accuracy or performance;
- a novel DEA estimator merely because models share an API; or
- independent validation from tests created in the same development process.

The proposed software contribution is instead the combination of:

1. immutable, named data contracts;
2. model-specific classes with explicit conventions;
3. solver-neutral status and result schemas;
4. independent post-solve reconstruction of feasibility and identities;
5. deterministic fingerprints and evidence bundles; and
6. a method-level stability registry that distinguishes provisional,

experimental, planned, and theory-gated work.

Whether that combination produces practically better reproducibility must be tested with external users and cross-language replications; that evaluation is part of the research roadmap, not a conclusion of this report.

3. Release scope and scientific status

3.1 Stability terminology

FrontierGuard applies stability labels to individual mathematical contracts, not merely to the package:

Label	Meaning in this report
Provisional	Implemented, locally tested, and checked against analytical or published numerical fixtures, but the complete cross-language and external-review gate has not been met.
Experimental	Implemented for investigation, while the API, formulation, interpretation, or independent oracle requires broader review.
Planned	Identified for future work but not available as an implemented method.
Gate	Primary theory, conventions, or independent-oracle work is required before implementation should be advertised.
Stable	Reserved for a method that has completed the project's full release criteria. No method has this status in version 0.1.0a1.

The machine-readable registry records, for each implemented family, a method identifier, equation version, label, stability, optimization class, data contract, score name, efficient value, orientations, returns-to-scale options, citations, compatible result models, and known limitations.

3.2 Implemented model families

The following table summarizes the release scope. The authoritative, more granular status is maintained in the [feature matrix](#).

Family	Current contract	Status
Radial envelopment	Input/output orientation; CRS, VRS, NIRS, and NDRS; optional lexicographic slack phase; targets, peers, and separate reference data	Provisional
Weighted additive	Explicit non-negative weights or unit-normalized weights; efficient inefficiency value zero	Provisional
Directional distance	Shared or DMU-specific non-negative contraction/expansion directions; signed directional factor	Provisional
Slack-based measure	Non-oriented canonical SBM using an exact Charnes–Cooper linear transformation	Provisional
Free Disposal Hull	Exact input- and output-oriented observed-peer enumeration	Provisional
Scale diagnostics	CRS/VRS scale efficiency and IRS/CRS/DRS classification using NIRS comparison	Provisional
Radial super-efficiency	Self-excluded reference technology with explicit infeasibility policy	Provisional
Multiplier and cross-efficiency	CRS multiplier model, one solver-selected cross-efficiency matrix, and Type-I assurance-region weight ratios	Provisional
Economic efficiency	Cost, revenue, technical, and allocative efficiency using common or DMU-specific prices	Provisional

Family	Current contract	Status
Productivity	Adjacent-period four-distance Malmquist index with efficiency-change and technical-change decomposition	Provisional
Serial network	CRS relational model for an arbitrary number of serial stages with a telescoping product identity	Provisional
General acyclic network	Graph-defined CRS relational model with external stage flows, directed links, and stage-gap decomposition	Experimental
Evidence export	Strict JSON, tidy CSV, Markdown method card, and SHA-256 manifest	Provisional

RAM, BAM, Russell measures, undesirable-output models, network SBM, dynamic network DEA, global and Malmquist–Luenberger indexes, bootstrap inference, partial frontiers, second-stage double bootstrap, window analysis, fuzzy DEA, inverse DEA, and large-scale sparse or parallel execution are not implemented contracts in this release.

4. Architecture

4.1 Design principles

The implementation follows six working principles:

1. **Make conventions data.** Orientation, returns to scale, slack strategy, reference technology, directions, price matrices, and network topology are explicit inputs.
2. **Fail before optimization when a contract is violated.** Missing, non-finite, negative, malformed, or incompatible data are not silently imputed, translated, or clipped.
3. **Separate mathematical construction from the solver.** Models generate framework-neutral programmes; adapters normalize backend status and diagnostics.
4. **Audit the returned solution independently.** Feasibility and identities are reconstructed from the input data and returned values rather than accepted solely because a solver reports an optimum.
5. **Preserve provenance.** Results retain solver identity, data and reference fingerprints, method conventions, tolerance, and audit residuals.
6. **Gate claims per method.** Source presence does not by itself make a method stable or release-ready.

These principles are consistent with general recommendations for reproducible computational research and scientific software ([Sandve et al., 2013](#); [Wilson et al., 2014](#)).

4.2 Component flow

```

flowchart LR
    A["DEADData / SerialNetworkData / NetworkData"] --> B["Contract validation"]
    B --> C["Model-specific programme builder"]
    C --> D["LP or exact finite enumeration"]
    D --> E["Solver adapter"]
    E --> F["Independent feasibility and identity audit"]
    F --> G["Immutable result and method metadata"]
    G --> H["Tables and evidence bundle"]

```

The package is divided into:

- data contracts for cross-sectional, panel, serial-network, and graph-network problems;
- model modules for radial, additive, directional, SBM, FDH, multiplier, super-efficiency, economic, productivity, and network families;
- a solver protocol and two HiGHS-based adapters;
- immutable result objects with table and strict-dictionary views;
- a machine-readable method registry; and
- reporting and command-line diagnostics.

The default numerical stack is NumPy, pandas, and SciPy. The zero-configuration adapter uses SciPy's HiGHS interface ([Virtanen et al., 2020](#)). An optional direct highspy adapter exposes sparse matrices, duals, reduced costs, and deterministic solver controls. HiGHS includes modern simplex methods described in, among other work, [Huangfu and Hall \(2018\)](#). A shared backend suite is intended to verify that adapters preserve model conventions.

4.3 Data contracts and fingerprints

DEADData stores non-negative input and output matrices together with unique DMU, input, and output names. SerialNetworkData adds an ordered sequence of intermediate-link matrices and stage names. NetworkData represents named stages and directed intermediate links and rejects cyclic graphs in the current contract.

All matrices are copied, validated, and made read-only. Each data object exposes a deterministic SHA-256 fingerprint derived from matrix shapes, data types, values, names, and, where applicable, topology. A fingerprint is an integrity and identity aid; it is not a substitute for archiving the data, documenting its provenance, or checking semantic equivalence.

4.4 Solver and result boundary

Model builders produce minimization-form LinearProgram objects. A LinearSolver returns a normalized status, primal vector, objective, backend status, and message. Public results record the normalized status and solver identity. The direct HiGHS adapter can retain richer optimization diagnostics, but optional backend features must not silently alter the scientific convention.

Result arrays are copied into immutable representations. Depending on the method, results may contain:

- normalized efficiency and raw radial factors;
- reference intensities or multiplier weights;
- input and output slacks;
- targets and peer sets;
- stage efficiencies or stage-gap contributions;
- economic or productivity decomposition components;
- solver status and identity;

- data, reference, or price fingerprints; and
- scale-adjusted audit residuals and the acceptance tolerance.

Strict JSON serialization uses `null` where a mathematically undefined value must be represented; it does not emit non-standard NaN tokens.

5. Mathematical contracts

This section summarizes the implemented conventions. Full equations and edge conditions are maintained in [METHODS.md](#).

5.1 Radial envelopment

Let X in $\mathbb{R}^{(n \times m)}_+$ and Y in $\mathbb{R}^{(n \times s)}_+$ define a reference technology. For evaluated DMU o , the input-oriented programme is

$$\min_{\theta} \theta$$

subject to

$$\begin{aligned} X^T \lambda &\leq \theta x_o, \\ Y^T \lambda &\geq y_o, \\ \lambda &\geq 0. \end{aligned}$$

The output-oriented programme maximizes ϕ subject to

$$\begin{aligned} X^T \lambda &\leq x_o, \\ Y^T \lambda &\geq \phi y_o, \\ \lambda &\geq 0. \end{aligned}$$

FrontierGuard reports input technical efficiency as θ , the raw output expansion factor as ϕ , and normalized output technical efficiency as $1/\phi$. It therefore does not label both raw objectives "efficiency."

Returns-to-scale restrictions are:

$$\begin{aligned} \text{CRS: } &\text{no intensity-sum restriction,} \\ \text{VRS: } &1^T \lambda = 1, \\ \text{NIRS: } &1^T \lambda \leq 1, \\ \text{NDRS: } &1^T \lambda \geq 1. \end{aligned}$$

An optional second phase fixes the optimal radial factor and maximizes input and output slacks. The default slack objective is normalized by observed variable magnitudes. A raw-sum strategy is available explicitly, and the slack phase can be disabled. The second phase cannot change the first-phase radial score.

5.2 Additive, directional, and slack-based measures

The weighted additive formulation maximizes

$$a^T s^- + b^T s^+$$

subject to

$$X^T \lambda + s^- = x_o,$$

$$Y^T \lambda - s^+ = y_o,$$

the chosen returns-to-scale restriction, and non-negative variables. Its reported quantity is an inefficiency measure with efficient value zero. Unit-normalized slack weights are invariant to positive unit changes; raw weights retain their declared units.

Given non-negative input-contraction direction g_x and output-expansion direction g_y , the directional-distance model maximizes β subject to

$$X^T \lambda \leq x_o - \beta g_x,$$

$$Y^T \lambda \geq y_o + \beta g_y.$$

At least one directional component must be positive for each evaluated DMU. The signed β is preserved when a separately supplied reference technology places an observation beyond the frontier along the selected direction. FrontierGuard does not yet implement the undesirable-output technology associated with environmental applications such as [Chung et al. \(1997\)](#).

For strictly positive evaluated inputs and outputs, the non-oriented SBM score is

$$\rho = (1 - 1 / m \sum_i s_i^- / x_{(io)}) / (1 + 1 / s \sum_r s_r^+ / y_{(ro)}).$$

The implementation solves the exact Charnes--Cooper linear transformation, recovers values in the original scale, and reconstructs the fractional score. It rejects zero evaluated denominators rather than inserting an undocumented epsilon or translating the data.

5.3 FDH, scale, and super-efficiency

FDH removes the convexity assumption and compares an evaluated DMU with one observed reference peer plus free disposability. FrontierGuard enumerates feasible peers directly. For input orientation, a peer j with $y_j \geq y_o$ has radial factor

$$\theta_j = \max_i x_{(ji)} / x_{(io)},$$

and the minimum feasible factor is selected. Output orientation analogously maximizes

$$\phi_j = \min_r y_{(jr)} / y_{(or)}$$

over peers satisfying $x_j \leq x_o$. Zero evaluated components receive explicit feasibility handling.

Scale efficiency follows the common normalized convention

$$SE_o = E_o^{(CRS)} / E_o^{(VRS)}.$$

When CRS and VRS scores differ, equality of NIRS and VRS scores identifies decreasing returns; otherwise the result is classified as increasing returns.

Radial super-efficiency excludes the evaluated DMU from its reference set (Andersen and Petersen, 1993). Input scores above one are retained. VRS and restricted-returns super-efficiency can be infeasible, so the caller must choose fail-fast behavior or explicit missing-result recording. This model is not super-SBM.

5.4 Multiplier, cross-efficiency, and assurance regions

The input-oriented CRS multiplier programme normalizes virtual input:

$$\max_{(u, v)} u^T y_o$$

subject to

$$\begin{aligned} v^T x_o &= 1, \\ u^T y_j - v^T x_j &\leq 0 \text{ for all } j, \\ u, v &\geq 0. \end{aligned}$$

For one solver-selected optimal multiplier system (u_o, v_o) , the cross-efficiency assigned by appraiser o to DMU j is

$$E_{(oj)} = (u_o^T y_j) / (v_o^T x_j).$$

Multiplier optima need not be unique. Consequently, version 0.1.0a1 reports one solver-selected cross-efficiency matrix and does not claim a unique ranking. Aggressive and benevolent secondary objectives are not implemented.

Type-I assurance regions express selected input-weight or output-weight ratios as

$$L \leq w_p / w_q \leq U.$$

They are compiled as linear restrictions without numerical division. Bounds are interpreted in original data units even when the numerical programme scales columns internally. These restrictions express substantive assumptions and can change efficiency scores; they are not merely solver settings.

5.5 Cost, revenue, and allocative efficiency

Given input prices w_o , minimum cost is

$$C_o^* = \min_{(\lambda)} w_o^T X^T \lambda$$

subject to output feasibility and the selected returns-to-scale technology. FrontierGuard reports

$$\begin{aligned} CE_o &= (C_o^*) / (w_o^T x_o), \\ AE_o &= CE_o / TE_o^{(input)}. \end{aligned}$$

Given output prices p_o , maximum revenue is

$$R_o^* = \max_{\lambda} p_o^T Y^T \lambda$$

subject to input feasibility and the selected technology, with

$$\begin{aligned} RE_o &= (p_o^T y_o) / (R_o^*), \\ AE_o &= RE_o / TE_o^{(output)}. \end{aligned}$$

Prices can be common vectors or DMU-specific matrices. Observed cost or revenue must be positive for the corresponding ratio to be defined. These decompositions follow the technical/economic distinction developed from [Farrell \(1957\)](#) and subsequent production-efficiency work such as [Färe et al. \(1985\)](#). Profit efficiency is not inferred from these ratios because zero or negative profit requires a different normalization and boundedness analysis.

5.6 Adjacent-period Malmquist productivity

Let $e_a(b)$ denote normalized radial efficiency for an observation from period a evaluated against period b 's technology. For adjacent periods t and $t + 1$, FrontierGuard computes

$$\begin{aligned} M = & \\ & [\\ & (e_{t+1|t}) / e_{t|t}) \\ & \times \\ & (e_{t+1|t+1}) / (e_{t|t+1}) \\ &]^{(1/2)}, \end{aligned}$$

with

$$EC = (e_{t+1|t+1}) / e_{t|t},$$

and

$$\begin{aligned} TC = & \\ & [\\ & (e_{t+1|t}) / (e_{t+1|t+1}) \\ & \times \\ & e_{t|t} / (e_{t|t+1}) \\ &]^{(1/2)}. \end{aligned}$$

The implementation computes ratios in log space and checks $M = EC \times TC$. Values above one denote productivity growth under this convention. CRS is the scale-inclusive total-factor-productivity specification. The explicit VRS option performs the analogous four-distance calculation against VRS technologies but should not be interpreted as the CRS scale-inclusive index. Global Malmquist and Malmquist--Luenberger indexes are separate future methods.

5.7 Serial relational networks

For a K -stage serial CRS process, external inputs x enter the first stage, intermediate products $z^{(1)}, \dots, z^{(K-1)}$ connect adjacent stages, and final outputs y leave the final stage. One multiplier system is shared across all stages. For evaluated DMU o , the programme normalizes

$$v^T x_o = 1$$

and maximizes

$$u^T y_o.$$

For every reference DMU j , stage constraints include

$$w_1^T z^{(1)}_j - v^T x_j \leq 0,$$

$$w_k^T z^{(k)}_j - w_{(k-1)}^T z^{(k-1)}_j \leq 0$$

$$(k = 2, \dots, K-1),$$

and

$$u^T y_j - w_{(K-1)}^T z^{(K-1)}_j \leq 0.$$

Stage efficiencies are ratios of adjacent weighted flows. When every selected stage denominator is positive, they telescope:

$$E_{\text{(overall)}} = \text{product}_{(k=1)}^K E_k.$$

The implementation independently checks this closure identity. This contract is related to relational network decompositions in the literature, including the two-stage formulation of [Kao and Hwang \(2008\)](#). FrontierGuard's arbitrary-stage software interface is an engineering generalization; this report does not claim that the underlying mathematical principle is new.

5.8 Experimental graph-defined acyclic networks

NetworkData represents stages as nodes and intermediate flows as directed links. A stage may have external inputs and outputs. Each link has one producer, one consumer, and one shared multiplier. For stage k and reference DMU j , define virtual input and output:

$$D_{(kj)} = v_k^T x^k_j +$$

$$\text{sum}_{(l \text{ in } \text{In}(k))} w_l^T z_{(l j)},$$

$$N_{kj} = u_k^T y^k_j + \sum_{l \in \text{Out}(k)} w_l^T z_l(j).$$

The centralized CRS model normalizes total external virtual input for the evaluated DMU, maximizes total external virtual output, and requires

$$N_{kj} \leq D_{kj}$$

for every stage and reference DMU. Internal link values cancel when the stage constraints are summed, producing the audited gap identity

$$1 - E_o = \sum_k (D_{ko} - N_{ko}).$$

The result reports stage ratios N_{ko}/D_{ko} and gap contributions $D_{ko} - N_{ko}$. A stage ratio is undefined when its selected denominator is zero even though the overall efficiency and gap decomposition remain defined. Stage weights and ratios may also be non-unique under alternate optima.

Network DEA includes multiple competing formulations and decompositions (Kao, 2014). The graph-defined model is therefore explicitly experimental. It requires published branched-network oracles, comparison with alternative network formulations, and external methodological review before any stability or novelty claim would be appropriate. It is not a network SBM or dynamic-network model.

6. Audited evidence

6.1 What an audit does

After a solver returns a candidate optimum, a model-specific auditor reconstructs key conditions using the original data and returned solution. Depending on the model, the reconstructed checks include:

- non-negativity of intensities, multipliers, and slacks;
- input and output feasibility;
- returns-to-scale restrictions;
- objective and reported-score consistency;
- target reconstruction from reference intensities;
- preservation of the phase-one radial factor;
- SBM fractional-score reconstruction;
- cost/revenue and technical/allocative identities;
- Malmquist $M = EC \times TC$ closure;
- serial-network product closure; and
- graph-network stage-gap closure.

Residuals are scaled by problem magnitudes so that a fixed absolute unit does not determine acceptance. Each optimal result records the maximum recomputed residual and the tolerance against which it was checked. A solution over the declared tolerance raises an error instead of being relabelled as valid.

6.2 What an audit does not do

An audit is not:

- an independent implementation of the estimator;
- proof that the encoded programme matches the primary publication;
- proof of global optimality beyond the solver and optimization class;
- evidence that the input and output variables represent a valid production technology;
- a bootstrap interval, sensitivity analysis, or robustness assessment;
- protection against the same conceptual mistake appearing in implementation, test, and documentation; or
- peer review.

This boundary is essential. AI-assisted source code and AI-assisted tests, in particular, cannot be treated as independent validation of one another.

6.3 Evidence bundles

For compatible results, `write_evidence_bundle` writes:

File	Contents
<code>result.json</code>	Strict, sorted, machine-readable result with method identifier and provenance fields
<code>scores.csv</code>	Tidy primary result table
<code>METHOD_CARD.md</code>	Equation version, stability, optimization class, score convention, citations, solver, fingerprint, residual, and known limitations
<code>manifest.json</code>	Schema and software versions plus SHA-256 checksums and byte sizes for the other files

The writer fails on a non-empty destination by default. With explicit overwrite permission, it replaces only the known FrontierGuard bundle files and preserves unrelated files. The manifest makes accidental change detectable, but long-term reproducibility still requires an archived software release, environment information, source data, and study-specific analysis code.

7. Validation status

7.1 Validation strategy

The project uses four layers:

1. **Contract tests** for shapes, finiteness, non-negativity, names, immutability, configurations, status mapping, and serialization.
2. **Algebraic tests** for primal feasibility, targets, decompositions, duality relationships, and model-specific identities.
3. **Metamorphic tests** for theoretically valid unit changes, permutations, dominated or duplicate references, and scaling properties.
4. **Independent oracles** intended to combine hand calculations, published

numerical cases, distinct mathematical formulations, and cross-language results.

A different solver applied to the same programme builder is a backend conformance check, not an independent method oracle.

7.2 Local release snapshot

On 26 July 2026, the following command was executed from the repository:

```
uv run pytest --cov=frontierguard --cov-branch --cov-report=term-missing
```

The observed local result under Python 3.12.8 was:

- 231 tests collected and passed;
- 3,698 measured statements;
- 970 measured branches; and
- 90.30% combined branch-aware coverage, above the configured 90% alpha-preview

threshold.

This is a dated execution snapshot, not an assurance that another checkout or environment will produce the same coverage. Continuous-integration records and the immutable release commit should be consulted when the version is tagged.

7.3 Remaining validation gaps

The current limitations of the evidence are at least as important as the test count:

Evidence	Status at report 0.1
Input validation, solver status, core feasibility, serialization	Covered by local automated tests
Analytical and selected published numerical fixtures	Present for implemented families, with depth varying by method
Invariance and decomposition properties	Present for selected theoretically valid cases
Default/direct HiGHS adapter conformance	Locally tested
Systematic R/Python/Julia cross-language corpus	Not complete
Independently reproduced package-wide benchmark	Not complete
External DEA methodologist review	Not complete
Real-user studies	Not complete
Large-n, sparse, parallel, and memory benchmarks	Not complete
Statistical coverage or uncertainty validation	Not applicable to the current deterministic release; future work

For this reason, the feature matrix uses "provisional" rather than "stable."

8. Reproducible example

The example below fits an input-oriented VRS radial model to three DMUs. It is deliberately small enough for its result to be inspected.

8.1 Environment

Once the v0.1.0a1 tag and release artifact exist, a source-based environment can be prepared with:

```
git clone https://github.com/chindavamsi/frontierguard-dea.git
cd frontierguard-dea
git checkout v0.1.0a1
uv sync --all-groups
```

Until that tag is published, `git checkout v0.1.0a1` is prospective and the current commit hash must be recorded instead. An archival DOI, when created, should identify the exact release used.

8.2 Analysis

```
from pathlib import Path

import pandas as pd
import frontierguard as fg

frame = pd.DataFrame(
    {
        "dmu": ["A", "B", "C"],
        "staff": [1.0, 2.0, 3.0],
        "cases": [1.0, 1.0, 2.0],
    }
)

data = fg.DEADData.from_frame(
    frame,
    inputs=["staff"],
    outputs=["cases"],
    dmu="dmu",
)

model = fg.RadialDEA(
    orientation="input",
    rts="vrs",
    slacks="normalized",
)
result = model.fit(data)

print(result.to_frame())
print(result.peers("B"))
print(result.targets("B"))
print(result.audit_residual)

fg.write_evidence_bundle(
    result,
    Path("evidence") / "radial-vrs-example",
    method_id="radial.envelopment",
)
```

The score table produced in the validation environment was:

	technical_efficiency	radial_factor	status
dmu			
A	1.0	1.0	optimal
B	0.5	0.5	optimal
C	1.0	1.0	optimal

DMU B uses DMU A as a peer with intensity 1.0 and has the target $staff = 1.0$, $cases = 1.0$. The independently recomputed residual was zero for all three fits in this environment, against a recorded audit tolerance of 10^{-7} .

The interpretation is conditional: relative to these three observations, this VRS technology, the input orientation, and the declared variables, B could radially contract its input by 50% while preserving its output. The example does not show that B is inefficient in an absolute, causal, or statistically significant sense.

9. Limitations and threats to validity

9.1 Scientific scope

- DEA measures performance relative to a constructed reference technology.

Variable selection, sample composition, heterogeneity, measurement error, and outliers can materially alter the frontier.

- The current release is deterministic. It does not report sampling

uncertainty, bias correction, or confidence intervals. Bootstrap and robust partial-frontier methods require separate implementation and validation ([Simar and Wilson, 1998](#); [Cazals et al., 2002](#)).

- Second-stage regression is not implemented. Naive inference based on

estimated DEA scores can be invalid; future work must follow an explicit inferential contract such as the issues addressed by [Simar and Wilson \(2007\)](#).

- A relative efficiency score is not a causal effect, a welfare judgement, or a complete management recommendation.

9.2 Data and model boundary

- Common data contracts require finite, non-negative matrices and do not impute missing values.

- Canonical SBM requires strictly positive evaluated inputs and outputs.

- Negative-data, weak-disposability, and undesirable-output technologies are not implemented.

- Serial and graph-defined relational network models are CRS in this release.

- Network SBM, dynamic carry-overs, shared-resource variants, and undesirable links are not implemented.

- The general DAG model is experimental and restricted to acyclic graphs.

- Cross-efficiency uses one solver-selected optimum; it does not resolve alternate multiplier systems with aggressive or benevolent secondary objectives.

- VRS and restricted-returns radial super-efficiency may be infeasible.

- The productivity module implements adjacent-period Malmquist calculations, not global Malmquist, window DEA, or Malmquist--Luenberger methods.

9.3 Numerical and software boundary

- Local residual checks can detect inconsistency in a returned result but

cannot prove the formulation correct.

- Alternate optima can change peers, multipliers, stage weights, and cross-efficiency entries while preserving primary objective values.
- Dense, one-DMU-at-a-time programme construction has not been established as suitable for very large datasets.
- Bitwise equality across platforms and solver versions is not guaranteed; research comparisons should use declared tolerances and inspect feasibility and identities.
- The API and evidence schema may change before version 1.0.
- Cross-language validation and external methodological review remain incomplete.

9.4 Claims and generalizability

This report is authored within the same project that produced the software. Its descriptions are therefore subject to developer confirmation bias. Independent replication, adversarial test cases, methodologist review, and real-user studies are required. No journal acceptance, citation impact, or practical superiority should be inferred from the existence of this report.

10. Research and release roadmap

The roadmap is ordered by evidence, not by the number of methods.

10.1 Immediate release foundation

For version 0.1.0a1 and its successor releases:

- bind package metadata, CITATION.cff, CodeMeta, the report, and the tagged source to one version;
- publish build artifacts only after clean installation and release tests;
- archive each cited release and record its version DOI and concept DOI;
- preserve the source in a long-term software archive;
- publish the API and mathematical conventions;
- keep the AI-use disclosure and human contribution roles current; and
- make release checks and known failures visible.

Software should be cited as a versioned research output rather than only as a project homepage ([Smith et al., 2016](#)).

10.2 Cross-language conformance research

The principal near-term research deliverable is a public benchmark corpus:

1. define compact, citable datasets for each convention;
2. calculate rational or hand-worked expected values where feasible;
3. run established R, Python, and Julia implementations from frozen environments;

4. compare objectives, normalized scores, targets, feasibility, duality, and identities rather than assuming alternate intensities must match;
5. classify disagreements as convention, data handling, numerical tolerance, implementation defect, or unresolved;
6. publish code, raw outputs, normalized comparison tables, and environment locks; and
7. submit the protocol and conclusions for independent review.

This study can evaluate a defensible research question: whether explicit contracts and audit evidence improve the reproducibility of DEA results across software ecosystems.

10.3 Beta evidence

Before a beta claim, the project seeks:

- at least two independently reproducible applied studies;
- external review by a DEA methodologist and an optimization specialist;
- installation and usability tests by researchers outside the project;
- performance and memory benchmarks with representative small, medium, and large technologies;
- stable evidence-bundle semantics; and
- public issue and release histories that document how findings changed the software.

10.4 Method expansion

Method families will be added only after their mathematical and validation contracts are defined. Priorities include:

- RAM, BAM, Russell, and additional non-radial measures;
- environmental and undesirable-output directional models;
- network SBM, dynamic networks, and carry-overs;
- global and environmental productivity indexes;
- bootstrap, partial-frontier, and second-stage inference;
- window, metafrontier, and robust-sensitivity analysis; and
- sparse, batched, and parallel execution.

These are research intentions, not features of version 0.1.0a1.

11. Data, code, and materials availability

The project source, tests, executable examples, and current documentation are available at <https://github.com/chindavamsi/frontierguard-dea> under the MIT License. This report describes software version 0.1.0a1. Until the release is tagged and archived, the repository URL alone is not an immutable scholarly record.

No new empirical dataset is analysed in this report. The numerical example is fully specified in Section 8. Small analytical and published-example fixtures used by the package are exposed through source tests and dataset loaders; their provenance should be checked in the corresponding tests and method documentation before reuse.

For an archival publication, the availability statement should be updated with:

- the Git commit hash;
- the signed or otherwise protected release tag;
- the software version DOI and concept DOI;
- the white-paper DOI, if archived separately;
- the environment lockfile checksum; and
- the cross-language benchmark DOI when that corpus exists.

The BibTeX source for this report is [references.bib](#). Reference metadata and DOI strings were checked against Crossref records or publisher metadata on 26 July 2026.

12. Authorship, contributions, funding, and interests

12.1 CRediT contributor statement

Vamsi Chinda: Conceptualization (lead); Methodology (lead); Software (lead and accountable maintainer); Validation (lead and accountable maintainer); Writing -- original draft (lead); Writing -- review and editing (lead); Project administration (lead).

Vamsi Chinda is the sole author and accountable maintainer.

The role statement records accountable human responsibility and should be updated when additional people make substantive contributions. Authorship of future papers must follow the policy of the selected venue and should not be assigned for honorary, promotional, or purely administrative participation.

12.2 Funding

This work received no external funding.

12.3 Competing interests

The author declares no competing interests.

13. Generative-AI use disclosure

AI tools assisted with code and documentation; all outputs were reviewed and validated by the author.

Generative AI materially assisted the FrontierGuard project and this report. OpenAI Codex was used during project work, including on 26 July 2026, for research and project planning, repository scaffolding, and drafting or reviewing source code, tests, documentation, release metadata, reference metadata, and this white-paper manuscript.

The exact serving model and version were not preserved in repository metadata for all development sessions. This report therefore does not assert a model identifier that cannot be verified. The product, date, task scope, affected artifacts, and validation performed should be logged for future sessions when that information is exposed.

For this report, AI assistance included:

- organizing the manuscript and proposing explanatory language;
- summarizing implemented contracts from source and project documentation;
- drafting equations, tables, limitations, and the reproducible example;
- discovering candidate scholarly references;
- checking DOI metadata against Crossref or primary publisher records; and

- checking internal citation keys and Markdown structure.

Human accountability is not delegated to the AI system. Vamsi Chinda is the author, creator, and accountable maintainer. The human author is responsible for deciding the scientific objectives, model conventions, architecture, release scope, accepted wording, and publication claims; reviewing and accepting or rejecting AI-assisted changes; checking formulations against primary literature; running and interpreting the quality checks; and verifying numerical results through independent implementations where available.

AI systems are not authors or contributors and cannot provide consent, take responsibility, disclose conflicts, or independently validate their own output. AI-assisted code and AI-assisted tests are treated as untrusted until human review and independent evidence are complete. The living project disclosure is maintained in [AI_USAGE.md](#).

14. Conclusion

FrontierGuard DEA version 0.1.0a1 is an alpha research preview intended to make DEA model contracts, solver provenance, and mathematical evidence inspectable within one Python workflow. It already spans several classical, economic, productivity, and relational-network families, but breadth is not evidence of completeness or correctness. The general graph-network model is experimental, implemented families remain provisional, and cross-language and external validation are unfinished.

The immediate research value of the project is therefore its testable proposition: explicit conventions, deterministic provenance, independent post-solve audits, and versioned evidence bundles may make efficiency research easier to reproduce and challenge. Establishing that proposition requires a public conformance corpus, independent replication, real-user studies, and peer review. Those activities-not an unqualified claim of being "complete" or "best"-define the route from this technical report to a credible research contribution.

References

- Farrell, M. J. (1957). The measurement of productive efficiency. *Journal of the Royal Statistical Society. Series A (General)*, 120(3), 253-290. <<https://doi.org/10.2307/2343100>>
- Charnes, A., Cooper, W. W., & Rhodes, E. (1978). Measuring the efficiency of decision making units. *European Journal of Operational Research*, 2(6), 429-444. <[https://doi.org/10.1016/0377-2217\(78\)90138-8](https://doi.org/10.1016/0377-2217(78)90138-8)>
- Banker, R. D., Charnes, A., & Cooper, W. W. (1984). Some models for estimating technical and scale inefficiencies in data envelopment analysis. *Management Science*, 30(9), 1078-1092. <<https://doi.org/10.1287/mnsc.30.9.1078>>
- Deprins, D., Simar, L., & Tulkens, H. (1984). Measuring labor-efficiency in post offices. In M. Marchand, P. Pestieau, & H. Tulkens (Eds.), *The performance of public enterprises: Concepts and measurement* (pp. 243-267). North-Holland.
- Charnes, A., Cooper, W. W., Golany, B., Seiford, L., & Stutz, J. (1985). Foundations of data envelopment analysis for Pareto-Koopmans efficient empirical production functions. *Journal of Econometrics*, 30(1-2), 91-107. <[https://doi.org/10.1016/0304-4076\(85\)90133-2](https://doi.org/10.1016/0304-4076(85)90133-2)>
- Färe, R., Grosskopf, S., & Lovell, C. A. K. (1985). *The measurement of efficiency of production*. Kluwer-Nijhoff. <<https://doi.org/10.1007/978-94-015-7721-2>>
- Sexton, T. R., Silkman, R. H., & Hogan, A. J. (1986). Data envelopment analysis: Critique and extensions. *New Directions for Program Evaluation*, 1986(32), 73-105. <<https://doi.org/10.1002/ev.1441>>
- Thompson, R. G., Langemeier, L. N., Lee, C.-T., Lee, E., & Thrall, R. M. (1990). The role of multiplier bounds in efficiency analysis with application to Kansas farming. *Journal of Econometrics*, 46(1-2), 93-108. <[https://doi.org/10.1016/0304-4076\(90\)90049-Y](https://doi.org/10.1016/0304-4076(90)90049-Y)>

- Andersen, P., & Petersen, N. C. (1993). A procedure for ranking efficient units in data envelopment analysis. *Management Science*, 39(10), 1261-1264. <<https://doi.org/10.1287/mnsc.39.10.1261>>
- Chambers, R. G., Chung, Y., & Färe, R. (1996). Benefit and distance functions. *Journal of Economic Theory*, 70(2), 407-419. <<https://doi.org/10.1006/jeth.1996.0096>>
- Chung, Y. H., Färe, R., & Grosskopf, S. (1997). Productivity and undesirable outputs: A directional distance function approach. *Journal of Environmental Management*, 51(3), 229-240. <<https://doi.org/10.1006/jema.1997.0146>>
- Tone, K. (2001). A slacks-based measure of efficiency in data envelopment analysis. *European Journal of Operational Research*, 130(3), 498-509. <[https://doi.org/10.1016/S0377-2217\(99\)00407-5](https://doi.org/10.1016/S0377-2217(99)00407-5)>
- Caves, D. W., Christensen, L. R., & Diewert, W. E. (1982). The economic theory of index numbers and the measurement of input, output, and productivity. *Econometrica*, 50(6), 1393-1414. <<https://doi.org/10.2307/1913388>>
- Färe, R., Grosskopf, S., Norris, M., & Zhang, Z. (1994). Productivity growth, technical progress, and efficiency change in industrialized countries. *American Economic Review*, 84(1), 66-83. <<https://www.jstor.org/stable/2117971>>
- Kao, C., & Hwang, S.-N. (2008). Efficiency decomposition in two-stage data envelopment analysis: An application to non-life insurance companies in Taiwan. *European Journal of Operational Research*, 185(1), 418-429. <<https://doi.org/10.1016/j.ejor.2006.11.041>>
- Tone, K., & Tsutsui, M. (2009). Network DEA: A slacks-based measure approach. *European Journal of Operational Research*, 197(1), 243-252. <<https://doi.org/10.1016/j.ejor.2008.05.027>>
- Cook, W. D., & Seiford, L. M. (2009). Data envelopment analysis (DEA)-thirty years on. *European Journal of Operational Research*, 192(1), 1-17. <<https://doi.org/10.1016/j.ejor.2008.01.032>>
- Kao, C. (2014). Network data envelopment analysis: A review. *European Journal of Operational Research*, 239(1), 1-16. <<https://doi.org/10.1016/j.ejor.2014.02.039>>
- Emrouznejad, A., & Yang, G.-L. (2018). A survey and analysis of the first 40 years of scholarly literature in DEA: 1978-2016. *Socio-Economic Planning Sciences*, 61, 4-8. <<https://doi.org/10.1016/j.seps.2017.01.008>>
- Simar, L., & Wilson, P. W. (1998). Sensitivity analysis of efficiency scores: How to bootstrap in nonparametric frontier models. *Management Science*, 44(1), 49-61. <<https://doi.org/10.1287/mnsc.44.1.49>>
- Cazals, C., Florens, J.-P., & Simar, L. (2002). Nonparametric frontier estimation: A robust approach. *Journal of Econometrics*, 106(1), 1-25. <[https://doi.org/10.1016/S0304-4076\(01\)00080-X](https://doi.org/10.1016/S0304-4076(01)00080-X)>
- Simar, L., & Wilson, P. W. (2007). Estimation and inference in two-stage, semi-parametric models of production processes. *Journal of Econometrics*, 136(1), 31-64. <<https://doi.org/10.1016/j.jeconom.2005.07.009>>
- Sandve, G. K., Nekrutenko, A., Taylor, J., & Hovig, E. (2013). Ten simple rules for reproducible computational research. *PLoS Computational Biology*, 9(10), e1003285. <<https://doi.org/10.1371/journal.pcbi.1003285>>
- Wilson, G., Aruliah, D. A., Brown, C. T., Chue Hong, N. P., Davis, M., Guy, R. T., Haddock, S. H. D., Huff, K. D., Mitchell, I. M., Plumbley, M. D., Waugh, B., White, E. P., & Wilson, P. (2014). Best practices for scientific computing. *PLoS Biology*, 12(1), e1001745. <<https://doi.org/10.1371/journal.pbio.1001745>>
- Smith, A. M., Katz, D. S., Niemeyer, K. E., & FORCE11 Software Citation Working Group. (2016). Software citation principles. *PeerJ Computer Science*, 2, e86. <<https://doi.org/10.7717/peerj-cs.86>>
- Huangfu, Q., & Hall, J. A. J. (2018). Parallelizing the dual revised simplex method. *Mathematical Programming Computation*, 10(1), 119-142. <<https://doi.org/10.1007/s12532-017-0130-5>>
- Virtanen, P., Gommers, R., Oliphant, T. E., et al. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3), 261-272. <<https://doi.org/10.1038/s41592-019-0686-2>>