Barometric Praxinomics - Technical Outreach

From: **Joseph.Maxwell**
Joseph.Maxwell02@proton.me

To: **Joseph.Maxwell02@proton.me**
Joseph.Maxwell02@proton.me

On: **Thursday, 28 May 2026 at 16:33:12**

Technical Specimen: Barometric Praxinomics in Use

A practical descriptor-reliability audit for materials modelling

Prepared by: Joseph Maxwell
Context: Systemica Engineering / GIL–EDK / Barometric Praxinomics
Purpose: Demonstrate how the measurement framework is used in practice on a real tabular materials dataset.

1. Executive summary

This document demonstrates a practical use case of Barometric Praxinomics as a measurement layer for systems under pressure.

The example used here is a materials machine-learning workflow for shear modulus prediction. Instead of asking only which descriptors rank highly in a model, the method asks whether those descriptors remain reliable when modelling conditions are disturbed.

The practical aim is simple:

identify which descriptors are stable enough to deserve further technical attention, and which should remain caution-labelled as fragile, target-adjacent, unstable, or artefact-prone.

This matters because materials machine-learning models are often used for both prediction and interpretation. A feature may rank highly in one fitted model while behaving inconsistently under changes in data split, perturbation level, model family, or preprocessing condition. The materials-facing paper frames this directly: descriptor reliability can be inspected separately from baseline importance, and stable descriptors should be treated as candidates for follow-up rather than as proof of physical truth.

The method therefore provides a practical evidence layer between:

raw model output

→ descriptor reliability audit

→ cautious follow-up decision

It does not claim to replace experiments, domain expertise, DFT, physical validation, or materials judgement.

It provides a structured way to decide what is worth investigating next.

2. Problem being addressed

A standard machine-learning workflow might produce:

model score

feature ranking

feature importance plot

This is useful, but incomplete.

In materials settings, descriptors often carry physical meaning. Columns such as density, band gap, formation energy, bulk modulus, elastic anisotropy and Poisson ratio are not neutral labels. They can easily be read as physical explanations.

The risk is that a descriptor becomes trusted because it ranked highly once.

The practical problem is:

predictive importance is not the same as descriptor reliability.

The current behavioural feature-auditing paper states the shift clearly: static ranking tells us what the model used at one point, while a behavioural trace asks what happens when modelling conditions change. The audit records drift, variability, recurrence, rank divergence and negative-control behaviour.

So the core question becomes:

Did the descriptor remain reliable under pressure,

or did it only look important under one comfortable model condition?

3. Method overview

The method is a perturbation-based descriptor reliability audit.

It uses a local CSV, selects a numeric target, trains baseline models, perturbs input descriptors, recomputes feature influence, and records how descriptor behaviour changes.

The MVP walkthrough describes the working demo as a minimal behavioural feature-audit run on a local CSV. It uses numeric descriptor columns, fits Ridge and Random Forest baselines, applies Gaussian noise and feature dropout, recomputes importance,

calculates descriptor drift D_j, calculates descriptor variability S_j, aggregates repeated-seed results, and exports evidence objects including stability tables, audit decision tables, checkpoint tables, plots and reports.

The simplified workflow is:

1. Load materials CSV

2. Select target property

3. Remove non-modelling columns

4. Apply optional physical sanity gate

5. Train baseline models

6. Perturb descriptors

7. Recompute feature influence

8. Measure descriptor drift and variability

9. Aggregate across seeds

10. Classify descriptor behaviour

11. Export tables, plots and decision-support summaries

The important point is that the method produces an audit trace, not just a model score.

4. Dataset used: shear modulus case

The worked example uses Shear.csv.

The target column is:

shear

The modelling descriptors include:

formation_energy_per_atom

density

band_gap

total_magnetization

bulk

elastic_anisotropy

poisson

The behavioural audit paper describes the input as a tabular materials dataset with rows as material entries, descriptors such as density, band gap, formation energy, bulk modulus, elastic anisotropy and Poisson ratio, and the target column shear. The modelling view drops the index column and formula string, while preserving them in the data-quality report.

The dataset is useful because it is not artificially clean. The raw file contains physically suspect values, including non-positive bulk modulus values and Poisson ratio values outside common physical bounds. These are reported rather than silently hidden.

That makes the case more realistic.

Real engineering and materials datasets often contain awkward values, partial artefacts, target-adjacent descriptors, and uncertain physical meaning. The audit layer is designed to make these issues visible before the model story becomes overconfident.

5. What is measured

The audit measures descriptor behaviour under perturbation.

The two core descriptor metrics are:

D_j = descriptor drift

S_j = descriptor variability

Plain-language definitions:

D_j = how far a descriptor’s influence moves away from baseline

S_j = how inconsistent that influence becomes across perturbation conditions

The MVP walkthrough states that the demo calculates descriptor drift D_j and descriptor variability S_j, then aggregates repeated-seed results and assigns audit decisions.

The audit can also record:

baseline importance

rank divergence

negative-control behaviour

recurrence across runs

first-unstable perturbation level

dynamic response profile

barcode state

target-adjacent caution

These measurements answer different questions.

A descriptor may have high baseline importance but high drift.
A descriptor may have moderate importance but strong stability.
A descriptor may be stable as a predictor but too target-adjacent to interpret independently.
A descriptor may behave well only after physical sanity gating.
A descriptor may appear stable at one perturbation level but drift under a dynamic sweep.

This is why a single feature ranking is not enough.

6. Engine behaviour on the shear dataset

The shear-origin MVP produced an important practical lesson: the interpretation changed depending on calibration and audit pressure.

In the raw uncalibrated run, the behavioural layer overreacted and marked all descriptors as volatile. After hierarchy-aware pre-audit and within-model importance normalisation, the fixed-level calibrated run became more interpretable. Under default thresholds, all seven descriptors classified as stable, but stricter profiles still downgraded some descriptors, especially bulk and density. The dynamic sweep then clarified that the fixed-level result was not the whole story.

This is exactly what the audit is supposed to do.

It prevents two bad extremes:

1. blindly trusting the static ranking

2. blindly rejecting descriptors because one raw audit pass looked unstable

Instead, the audit produces a more useful statement:

This descriptor is useful under these conditions,

sensitive under these other conditions,

and should be retained, rechecked, caution-labelled, or rejected accordingly.

7. Static ranking vs behavioural audit

In the shear case, static ranking alone encourages attention to mechanically close descriptors such as bulk, poisson, and elastic_anisotropy.

That is not wrong. These descriptors are likely predictive.

But because they are mechanically close to shear modulus, they are also target-adjacent. Their predictive usefulness does not automatically make them independent physical explanations.

The materials-facing paper makes this explicit: bulk modulus, Poisson ratio and elastic anisotropy are mechanically close to shear modulus. Their stability as predictors is useful, but they should not be interpreted as independent causal explanations without domain-specific validation.

So the practical audit decision is not:

bulk is important, therefore bulk explains shear

It is:

bulk is useful as a predictor,

but target-adjacent and pressure-sensitive,

so retain it with caution and avoid standalone physical interpretation.

That is a materially better decision.

8. Dynamic sweep: stability is not binary

A fixed perturbation level can make stability look like a yes/no label.

The dynamic sweep improves this.

Instead of asking:

is this descriptor stable?

it asks:

stable up to what audit pressure?

where does drift begin?

does instability appear gradually or sharply?

does the physics gate change the response?

The behavioural audit paper says the dynamic sweep reframes stability as a function over perturbation pressure rather than a binary property attached to a feature.

This matters in real-world use because technical decisions are rarely binary.

A descriptor may be acceptable for screening but not for explanation.
A descriptor may be retained for prediction but not for physical interpretation.
A descriptor may be stable under default thresholds but require recurrence before

promotion.
A descriptor may be useful only as part of an interaction model.

This is why the audit produces decision states rather than only labels.

9. Descriptor hierarchy as an audit hypothesis

The materials version can use a supplied descriptor hierarchy.

Example grouping:

Energetic / electronic:

formation_energy_per_atom

band_gap

total_magnetization

Structural geometry:

density

Mechanical / target-adjacent:

bulk

elastic_anisotropy

poisson

The materials paper treats this as a constraint-chain audit structure. Energetic and electronic descriptors provide upstream context, structural geometry descriptors describe packing and lattice-related structure, and mechanical-response descriptors sit closer to the target. The paper is clear that this hierarchy is not assumed as proven causal structure from one dataset. It is used to ask better audit questions.

The practical question is:

Does hierarchy position change descriptor behaviour under audit pressure?

If yes, the hierarchy is useful for follow-up.

If no, it has not earned practical value.

This is important for academic and industrial users because it keeps domain knowledge

in the loop without pretending the hierarchy is already proven.

10. Barcode decision support

The upgraded audit produces barcode-style decision states.

These are not physical certificates.

They are structured state-memory summaries for descriptor behaviour.

The materials paper describes barcode checkpoints as structured state-memory objects tied to a declared target, model family and evidence context. The barcode is not a physical certificate; it is a decision-support object linked back to the audit trace.

Example decision states:

invariant_core

stable_influence

conditional_stable

gradual_drift

fragile_proxy

volatile_collapse

interaction_dominant

rejected

The behavioural audit paper provides a state-to-action map. For example, invariant_core can be prioritised for follow-up if the scientific context supports it, conditional_stable should be retained with caution and checked through recurrence/dynamic sweep, fragile_proxy should not be used explanatorily, and volatile_collapse should be rejected as explanation.

This makes the method practical.

It turns a complex audit into follow-up guidance.

11. Example interpretation table

A simplified interpretation from the shear-origin audit would look like this:

Descriptor	Role	Audit behaviour	Practical decision
formation_energy_per_atom	energetic stability context	robust across current sweep	retain for follow-up
band_gap	electronic response proxy	conditional stable	recheck under recurrence

total_magnetization	magnetic/electronic context	conditional stable	retain as interaction-risk support
density	structural geometry / packing proxy	pressure-sensitive	re-audit and compare gated/raw response
bulk	mechanical target-adjacent response	useful but pressure-sensitive	retain predictor, avoid standalone explanation
elastic_anisotropy	mechanical anisotropy	target-adjacent stable/caution	retain with caution
poisson	mechanical target-adjacent response	conditional stable/caution	retain with caution

The exact state can change with thresholds, physical gating and dynamic sweep settings.

That is not a weakness.

That is the point.

The audit is showing which interpretations are robust and which depend on the audit conditions.

12. What this enables in practice

For an academic user, this enables:

more defensible descriptor interpretation

clearer separation between prediction and explanation

structured evidence before physical follow-up

better reporting of uncertainty and instability

For an industrial user, this enables:

risk reduction before using model explanations

identification of fragile proxies

clearer audit trails

decision support for which descriptors to keep, test, quarantine or reject

For a small consultancy or technical support business, this becomes a service pattern:

Client has tabular technical/model data

→ run descriptor reliability audit

→ produce stability map and decision table

→ flag fragile/high-risk interpretations

→ recommend follow-up tests or modelling refinements

This is where the Access to Work relevance comes in.

The technical work exists, but delivering it as a service requires significant non-technical execution: client communication, file handling, follow-up emails, meeting notes, task tracking, scheduling, evidence-pack preparation, formatting, version control, and repeated administrative coordination.

Those are exactly the kinds of disability-related work barriers that can prevent a viable technical method from becoming a viable business process.

13. Real-life delivery workflow

A practical delivery workflow would look like this.

Stage 1: Intake

Receive client dataset

Confirm target column

Confirm descriptor columns

Confirm excluded fields

Confirm domain constraints

Agree audit scope

Output:

intake note

data dictionary

scope confirmation

risk flags

Stage 2: Pre-audit

Drop non-modelling columns

Check numeric columns

Apply optional physical sanity rules

Log suspect rows

Mark target-adjacent descriptors

Check redundancy/relevance

Assign full/caution/quarantine lanes

Output:

pre-audit report

suspect-row summary

descriptor lane table

Stage 3: Perturbation run

Train baseline models

Run repeated seeds

Apply perturbation levels

Apply dropout/noise tests

Run optional negative controls

Recompute descriptor influence

Output:

model response table

descriptor drift table

descriptor variability table

Stage 4: Dynamic sweep

Increase audit pressure

Track first drift onset

Track volatility onset

Compare raw vs gated runs

Compare model-family response

Output:

dynamic sweep summary

first-unstable threshold table

response heatmap

Stage 5: Decision-support output

Classify descriptor states

Generate barcode checkpoint table

Generate stability map

Produce client-facing interpretation

Separate prediction-use from explanation-use

Output:

technical report

descriptor decision table

figures

action recommendations

appendix/audit bundle

Stage 6: Follow-up

Discuss results with client

Identify descriptors for physical follow-up

Identify fragile proxies

Recommend further runs

Prepare revised audit if needed

Output:

meeting summary

next-step plan

follow-up quote or work package

14. Practical wins from the current engine

A key design feature of this workflow is that it has been built for low-friction use.

Because of my health conditions, including fatigue, pain, ADHD-related executive dysfunction and fluctuating working capacity, the system could not depend on me manually holding the whole modelling process in working memory. The engine therefore had to be designed around a simple operational loop:

load data

→ check data

→ choose run type

→ run audit

→ read report

→ inspect full diagnostic files if needed

This is not only an accessibility feature. It is also an engineering advantage.

A useful audit engine should not require the user to manually reconstruct what happened during the run. It should produce a readable summary, preserve the detailed diagnostic files, and keep enough checkpointed state that the run can be inspected, repeated or resumed without relying on memory.

The current MVP already follows this principle. The walkthrough describes the demo as taking a local CSV, selecting a numeric target, fitting Ridge and Random Forest baselines, applying Gaussian noise and feature dropout, recomputing descriptor importance, calculating descriptor drift and variability, aggregating repeated-seed results, and exporting a full evidence bundle. The exported outputs include stability tables, baseline-versus-audit decision tables, barcode checkpoint tables, phase-density projections, plots and report files.

That means the user does not only receive a final answer. They receive a diagnostic pack.

A typical output bundle can include:

summary report

descriptor stability table

baseline-vs-audit decision table

barcode checkpoint table

phase-density projection

dynamic perturbation trace

descriptor threshold table

plots and figures

data-quality warnings

This gives two levels of use.

First, a fast user-level read:

what happened?

which descriptors were stable?

which descriptors need caution?

what should be followed up?

Second, a technical inspection layer:

what rows were removed?

which descriptors drifted?

where did instability begin?

did the physics gate change the result?

what evidence supports the decision?

That split is important for real-world use. A client, supervisor or assessor can read the report first, while a technical reviewer can inspect the supporting files if they want to check the run.

15. Checkpointing and scalability

The checkpoint system is also important for large datasets.

Large modelling runs can become difficult to manage on smaller devices because they generate many intermediate states, repeated splits, perturbation levels, feature-importance traces and diagnostic outputs. Instead of storing everything as one uncontrolled memory-heavy object, the engine stores structured checkpoint artefacts and compressed decision states.

This is the purpose of the barcode layer. The materials paper describes barcode checkpoints as structured state-memory objects for descriptor behaviour, tied to a declared target, model family and evidence context. It is not a physical certificate; it is a decision-support object linked back to the full audit trace.

In practice, this means the system can preserve the important run state without requiring the whole process to remain live in memory.

The benefit is practical:

large run can be split into segments

checkpoint states can be written during the run

diagnostic files preserve intermediate evidence

failed or interrupted runs can be inspected

future versions can restart from checkpoints

summary states can be reviewed without loading the full trace

This is especially relevant for Materials Project-style datasets. The next step is to move from the current tabular shear example into larger Materials Project runs, including the planned 3D/spatial mode.

The v2 spatial specification extends the same governance philosophy into 3D crystal structures. Instead of only perturbing tabular descriptors, it introduces spatial perturbations such as atomic coordinate jitter, vacancy dropout and supercell scaling to test whether a model remains physically valid under structural change.

That is the logical next stage:

v1 / tabular mode:

audit descriptor reliability under perturbation

v2 / spatial mode:

audit physical robustness of 3D structure models under spatial perturbation

The v2.1 notes frame this as a dual-modality setup: v1.6.3 audits the logic of property relationships, while v2.1 audits the physicality of 3D atomic structure through vacancy and extensivity checks.

For Access to Work, the point is not that the 3D engine is already a finished commercial product. The point is that there is a clear technical development path from a working tabular MVP toward a more advanced spatial audit workflow.

16. Speed and messy-data handling

The current shear dataset run is also practically useful because it is fast enough to be realistic.

On the current MVP, the shear-origin run can be completed in under five minutes on a normal working setup. That matters because the tool is not only theoretical. It can be used iteratively.

A fast run time means the user can:

test a dataset

inspect warnings

adjust gates

rerun with a physics sanity check

compare raw versus gated outputs

prepare a diagnostic pack

without needing a large compute cluster for the initial audit.

The shear case also shows that the engine can detect messy data rather than pretending it is not there. The behavioural audit paper records that the raw Shear.csv contained 15 rows with non-positive bulk modulus, 8 rows with Poisson ratio below -1, and 89 rows with Poisson ratio above 0.5. These were reported rather than silently hidden.

That is a significant practical win.

A normal modelling workflow might clean the data and move on. This audit workflow preserves the fact that the dataset contained physically suspect values, then allows raw and gated behaviour to be compared.

That is more useful for engineering judgement because it shows whether the interpretation depends on the messy rows.

In the shear-origin case, the optional physics gate removed 99 suspect rows and left 1939 rows for sensitivity checking. The behavioural paper reports that the fixed-level calibrated MVP preserved the same overall stable pattern, while the dynamic sweep showed that the raw and physics-gated runs differed in descriptor behaviour.

That is exactly the kind of diagnostic result the engine is meant to produce.

Not just:

model score = X

But:

the dataset contains these physical issues;

the raw run behaves like this;

the gated run behaves like this;

the descriptor decision changes here;

therefore this interpretation should be retained, cautioned, rechecked or rejected.

17. Practical next steps

The next development step is to package the tool into a more repeatable external workflow.

The near-term path is:

1. Keep the tabular MVP stable.

2. Use shear as the worked example.

3. Add a Materials Project dataset as a second validation-style run.

4. Improve the report template.

5. Formalise the checkpoint/restart logic.

6. Package the diagnostic outputs into a client-ready evidence pack.

7. Develop the 3D/spatial audit as the next technical extension.

The 3D mode is the natural next extension because it tests a different class of failure.

The tabular engine asks:

does descriptor importance remain reliable under perturbation?

The spatial engine asks:

does a 3D materials model remain physically reliable under spatial perturbation, vacancy dropout and supercell scaling?

The v2 port document describes this directly: v1.2 tests whether a model's mathematical logic breaks when array distributions shift, while v2.0 tests whether a model's physical logic breaks when atoms jitter, point defects occur, or boundary conditions scale up.

That gives the development roadmap a clean structure:

current usable engine:

tabular descriptor reliability audit

near-term outreach specimen:

shear dataset evidence pack

next validation step:

Materials Project tabular dataset

next technical extension:

3D spatial perturbation engine

This is also where Access to Work support becomes practically important. The technical runtime can produce the audit files, but turning those outputs into consistent external deliverables requires admin, communication, version control, file organisation, report formatting, follow-up tracking and client-ready packaging.

That is the real-world delivery bottleneck.

The support need is not to invent the method. The support need is to make the method deliverable, repeatable and communicable.

Claim ceiling

This specimen does not claim:

the method proves causality

stable descriptors are physically true

the shear dataset validates broad materials generality

the barcode layer is certification

the runtime replaces experiments or expert judgement

It claims:

descriptor importance can be stress-tested

descriptor reliability can be separated from static ranking

unstable descriptors should not be over-interpreted

audit traces can support better follow-up decisions

the method can be packaged as a practical technical service

That is the correct claim.

The paper's own limitations say the current evidence layer is still small, descriptor-space perturbations are not physical simulations, repeated cached intakes are not one large independent dataset, barcode state memory is decision support rather than certification, and stable descriptors are not claimed as causal.

That restraint is what makes the method credible.

Closing statement

This technical specimen shows a practical route from a modelling problem to a usable service.

The starting problem is common:

a model gives a feature ranking,

but the user needs to know whether that ranking is reliable enough to guide interpretation.

The proposed solution is a perturbation-based descriptor reliability audit.

The shear-modulus example demonstrates how the method can:

load a real materials dataset

preserve awkward data-quality issues

train baseline models

stress-test descriptors

measure drift and variability

compare raw and calibrated runs

identify target-adjacent caution

produce dynamic sweep summaries

export decision-support states

The useful outcome is not a claim that one descriptor “wins”.

The useful outcome is a better decision surface:

what to retain

what to recheck

what to caution-label

what not to interpret physically

what deserves further follow-up

That is the practical value of the engine.

It turns feature importance from a static ranking into an auditable behavioural trace.

Add-on

Why Access to Work support matters

This workflow is realistic, but it contains many execution steps that are not purely technical.

The most disabling parts are likely to be:

admin tracking

email follow-up

document packaging

deadline monitoring

version control

client communication

meeting scheduling

action-list maintenance

evidence-bundle organisation

financial/admin paperwork

The technical method relies on careful sequencing. If communication, admin or working-memory load collapses, the technical work becomes harder to deliver consistently even if the modelling capability exists.

The support needed is therefore not someone to invent the method.

It is support to make the method deliverable.

A support worker or business administration assistant could help by:

tracking client requests

maintaining a delivery checklist

organising files and outputs

preparing draft emails

logging follow-up actions

formatting reports

maintaining project folders

checking deadlines

helping convert technical outputs into client-ready packs

That support would directly improve the viability of the business because the service depends on turning technical analysis into repeatable deliverables.