



Conformance is not correctness.

Autonomy Platform · Public Blog

P E R S P E C T I V E

© 2026 Azirella Ltd. All rights reserved worldwide.

[Read more at azirella.com/blog](https://azirella.com/blog)

TL;DR

Tim Varelmann is right that a solver says more than I allowed: a bound, a gap, an infeasibility status, every statement narrow and every one true. But he defines his own guarantee as a policy that "will not deviate from its specification", and concedes that a model is wrong against the world rather than against its specification, which together say the guarantee does not cover the part that can be wrong. A gap handed to a human with no instrument to measure it is not a guarantee but a transfer of liability, and the instrument he himself named, conformal prediction, belongs one level up: a standing measurement of whether the specification still matches the world.

KEY POINTS

- A solver's optimality gap measures distance inside the model you wrote, and read as confidence it points the wrong way: by the optimizer's curse, the harder the search, the more reliably it finds the input estimate that flatters.
- The architectural guarantee covers conformance to the specification, including where the specification has departed from the world, so it is not the property that earns mathematics the core of a high-stakes decision.
- Planning already names boundaries on the horizon dimension through buckets, horizons and time fences, and names none at all on the parameter dimension: the nine-day lead time enters S&OP and the scheduler with identical confidence.

- Outside its regime a specification does not acquire noise, it acquires a bias whose sign is known in advance: by Jensen's inequality a plan built on averages reports a floor the real cost will almost never reach.

Dr. Tim Varelmann [replied to my piece on optimization and overconfidence](#), and he is right about something I got wrong, so let me start there.

I wrote that a solver returns one plan, labelled optimal, with no probability attached. It returns more than that. If it proves optimality, it has proved something real: no better solution exists for the question the model asked. If it stops at a time limit, it reports a bound and a gap, so the plan is provably within that distance of the best possible. If the numerics are shaky, the status says so. And if the model is infeasible there is no plan at all, which is also an answer, and an honest one. Every statement a solver makes is narrow, and every one is true. My sentence did not allow for any of it, and I withdraw it.

None of those statements is a probability or a likelihood. That is where I think the argument survives, and the rest of this is why.

The one number it does report points the wrong way

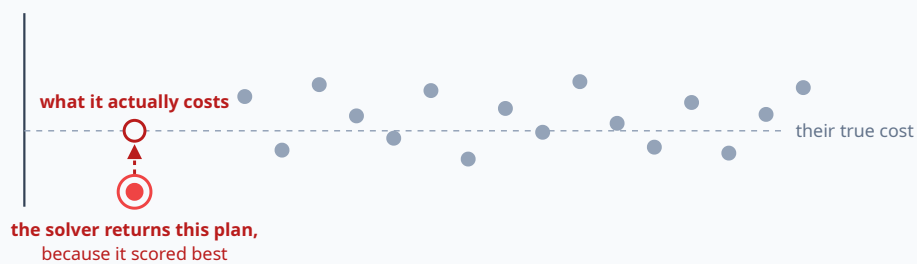
The optimality gap tells you how far your plan is from the best solution of the model you wrote. It says nothing about the distance between that model and the world.

Read as a confidence statement, it actually points backwards, and the reason is worth being precise about, because it is not the one people expect.

A solver does not evaluate one plan. It evaluates an enormous number of them and returns whichever scored best. Every one of those scores was computed from estimates: this lane's transit time, that line's yield, this supplier's cost. The estimates are honest, too high as often as too low, and not one of them is biased. But the plan that wins is disproportionately the plan whose errors happened to run in its favour, because scoring best on a noisy measure is partly a matter of being good and partly a matter of being flattered. The solver never chose the estimates. It chose the plan they flattered.

The solver does not choose the estimates. It chooses the plan they flattered.

estimated cost of each plan the solver evaluated



Every estimate is honest, too high as often as too low. The plan that wins is the one whose errors ran in its favour, because that is what winning on a noisy score means.

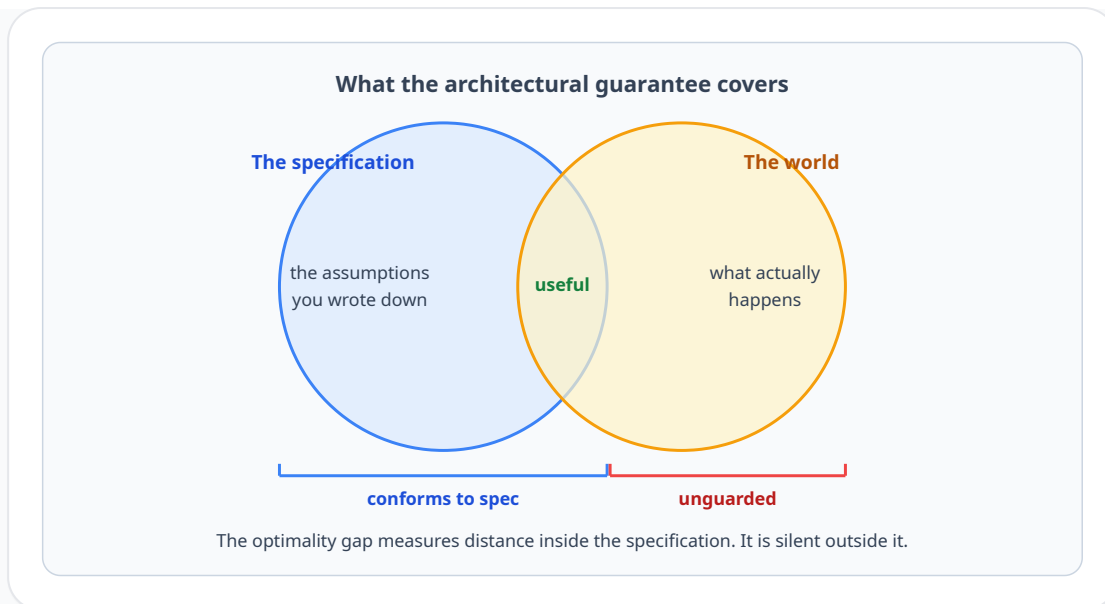
Search harder and you search more plans, so you find a better winner, not a better plan.
No estimate was biased. The selection was.

This is the optimizer's curse (Smith and Winkler, 2006), and the uncomfortable part is the direction it runs. The more plans the optimizer searches, and the harder it searches them, the more thoroughly it mines the estimates for a favourable error. A tighter gap means a more exhaustive search of a space scored on those estimates, so it means more exposure, not less. The one number a solver reports about its own confidence gets worse as the search gets better.

Conformance, not correctness

Tim states the guarantee precisely, in his own words. His planning policy "will not deviate from its specification." That is conformance.

He also writes that when a model is wrong, "it is wrong against the world, never against its specification, and you debug it by finding where the specification departs from reality." I agree with every word. But put the two sentences side by side and they say something he does not draw out: the guarantee does not cover the part that can be wrong.



This is my whole objection in one picture. The architectural guarantee is a guarantee about the left-hand circle. It holds with full force over the region where the specification has departed from reality, because conformance to a wrong specification is still conformance. A feasible plan built on a capacity figure that is fifteen percent too high is still an invented answer. It is just an invented answer in a tidier format, and it is harder to catch precisely because it is internally consistent. Nothing in the solver's return status distinguishes that plan from a good one.

A transfer of liability

So what happens to the part the guarantee does not cover? Tim hands it to the modeller. Completeness of the model is the modeller's job; using the technology correctly is the modeller's responsibility. "A technology that demands skilled modeling is not an overconfident technology."

As a division of labour that is unremarkable, and I want to be fair to it. A compiler guarantees conformance to your source, not that your program is right, and nobody minds. Every engineering discipline assigns the correctness of the specification to the engineer.

But the claim on the table is larger than a division of labour. It is that mathematics belongs at the core of high-stakes decisions *because* its guarantee is architectural, where a learned model's is merely statistical. If that guarantee stops at the specification boundary, it is not the property that earns the core position. A gap handed to a human who has no instrument to measure it is not a guarantee. It is a transfer of liability.

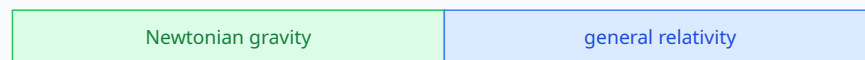
"Some are useful" is a claim, and a claim needs a test

George Box's line that all models are wrong but some are useful is nearly fifty years old, and it is almost always quoted as a shrug: an all-purpose licence for any model at all. It is not a shrug. The first half is a concession. The second half is a claim, and a claim needs a test.

Physics is the discipline that takes the second half seriously. Newtonian gravity is wrong, and precisely so: it is the low-speed, weak-field limit of general relativity, and it drops terms relativity keeps. It is also the right model for designing a bridge, where the dropped terms sit far below anything you could measure. Move to a GPS satellite, where those same terms put you kilometres out within a day, and Newton stops being useful and relativity starts. Neither is correct. Each is the useful one inside its own band, and physics is unusual in knowing where the band ends and saying so out loud.

A useful model is a model with a stated boundary

PHYSICS

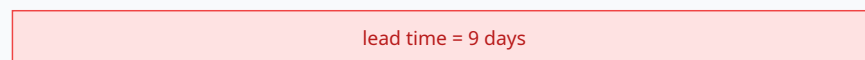


a bridge: the dropped terms are unmeasurable

a GPS satellite: they decide the answer

Newton is the low-speed, weak-field limit of relativity. Physics knows where it stops holding.

A SUPPLY CHAIN SPECIFICATION



a calm week

peak season

the port closes

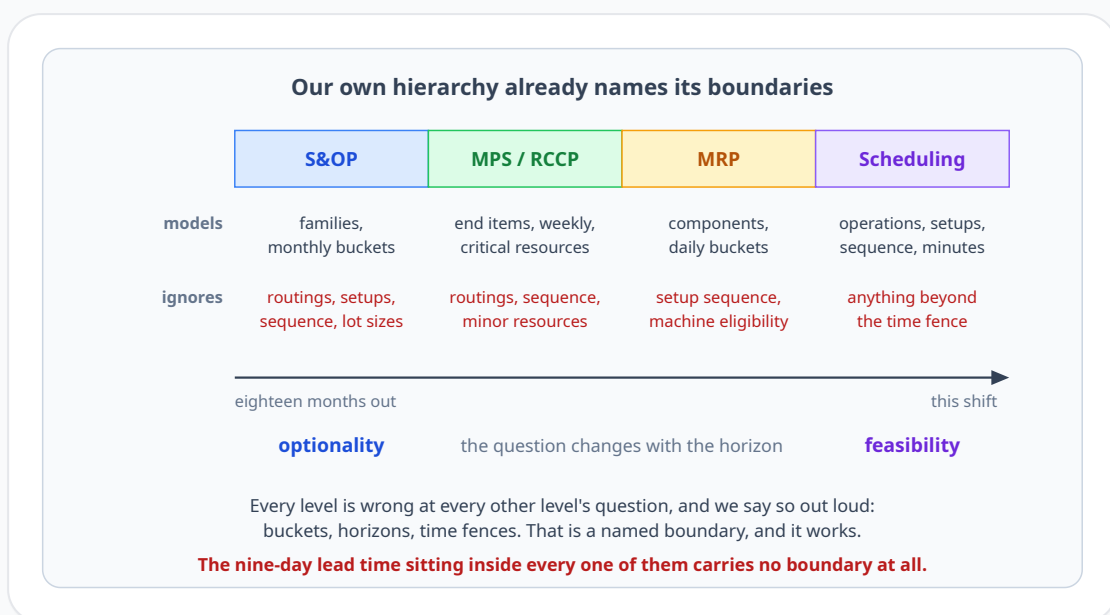
One number, equal confidence everywhere. The boundary exists; nobody wrote it down.

A supply chain specification has boundaries too. The nine-day lead time holds in a calm week, holds less well in peak season, and means nothing at all the week the port closes. The difference is that almost nobody writes the boundary down. So the same number is applied with equal confidence in every condition, and the model fails silently in one of them. The specification is not wrong in the way a bug is wrong. It is wrong in the way Newton is wrong: perfectly serviceable until you leave the regime, and silent about where the regime ends.

Our own hierarchy already knows this

Before I push this at Tim any further, I should admit that supply chain planning worked it out long ago, for one dimension at least.

Nobody schedules a shift eighteen months out, and nobody runs S&OP in five-minute buckets. S&OP models families in monthly buckets and ignores routings, setups, sequence and lot sizes. Rough-cut capacity planning checks the master schedule against the critical resources and ignores the rest. MRP works in components and daily buckets. A finite-capacity scheduler models the setup matrix, machine eligibility and the sequence, and ignores anything beyond the time fence.



Every one of those models is wrong at every other one's question, and we say so out loud. The bucket size is a stated boundary. The planning horizon is a stated boundary. The time fence is a stated boundary. That is precisely the practice I just praised physics for, and it has been standard in planning for decades.

It also says something about this debate. Tim and I have both been arguing as though there were one right core for the whole stack, and the hierarchy says there is not, because the two ends want opposite things. At eighteen months the question is optionality: what am I keeping open, what am I qualifying, where am I positioning inventory. A point-valued optimum means very little there, because the inputs are distributions months wide and none of the options have expired yet. At this shift the question is feasibility: does this fit, in this order, on this machine. That is where a solver is at its strongest, and where "infeasible" as a return status is exactly the answer you want and nothing else reliably gives it.

The strategic level is Newton's law of gravity. Execution is Einstein's relativity. At the strategic horizon the corrections a detailed model would add, the exact setup sequence, the particular machine, the hour of the day, sit far below the resolution of the decision being taken, so the coarse model is not a compromise, it is the right model. At execution those corrections dominate. Strategic planning is the bridge; execution is the satellite.

I want to mark where that analogy stops, because the break is the interesting part. Relativity is correct at every scale and merely unnecessary at most of them; you could design a bridge with it and get the right answer at an absurd price. A finite-capacity schedule at eighteen months is not unnecessary, it is false, because the inputs it needs do not exist yet. Our models do not become redundant outside their band. They become wrong. Which makes naming the boundary more important for us than it is for physics, not less.

And here is the gap. We name boundaries on the horizon dimension with real discipline, and we name none at all on the parameter dimension. The nine-day lead time sits inside every level of that hierarchy. It carries no bucket, no horizon, no fence, and no statement of the conditions under which nine is the right number. It goes into S&OP and into the scheduler with exactly the same confidence, and neither of them is told when to stop believing it.

The specification does not fail at random

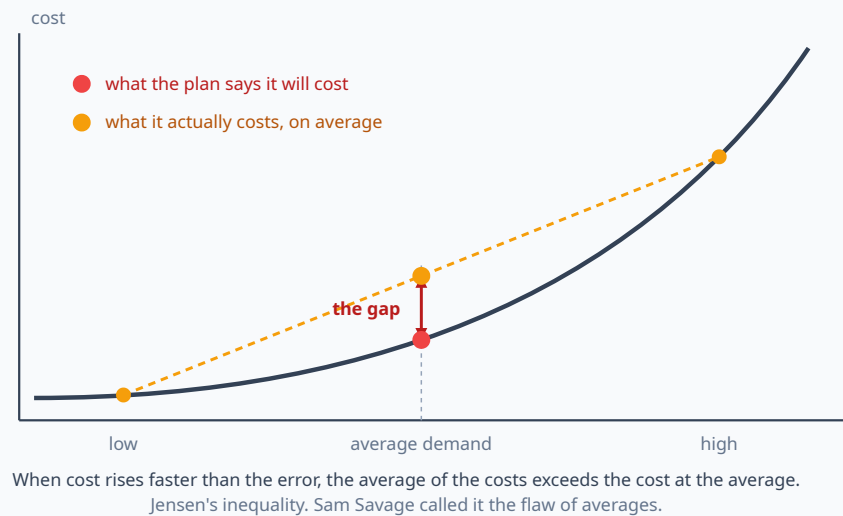
Here is the part I did not make enough of the first time, and it is the sharpest thing I can say against treating a deterministic optimum as a neutral estimate.

When a plan leaves its regime, it does not just acquire noise. It acquires a bias, and the sign of the bias is known in advance.

Take any quantity where the cost of being wrong rises faster than the error itself. That is the normal case in a supply chain, and it is Taleb's definition of fragility: while the shortfall is smaller than your buffer it costs almost nothing, and once it is larger than your buffer every extra unit is paid for at the worst price available, expedited freight, overtime, a spot buy, a lost order. Plot that cost against demand and you get a curve that starts flat and bends upwards.

Now plan at the average.

The flaw of averages: the plan reports the lower number



The plan reports the cost at the average demand. What you actually pay, averaged over the demands that really occur, is the average of the costs, and on a curve that bends upwards that is the higher number. Always. This is Jensen's inequality, and Sam Savage named its everyday form the flaw of averages (2009): plans based on average conditions are wrong on average.

So a deterministic optimization planning at the average lead time, the average demand and the average yield does not report its expected cost. It reports a floor the real cost will almost never reach. The solver conformed perfectly to the specification it was given, proved optimality against it, and returned a number that is biased in a direction you could have predicted before you ran it.

That is what conformance buys you and what it does not. The model is exactly right about itself and systematically optimistic about the world, and no status code in the return value will tell you which.

Why I agree with Tim about LLMs

Tim's original argument was that every large language model is System 1: fast, fluent, never short of an answer. I agree, and the regime framing is the reason I agree more strongly than I did.

A language model has no regime at all. It does not hold a wrong model of the world. It holds a model of text about the world, which is why it is fluent everywhere and accountable nowhere. There is no band to leave because there was never a band. An

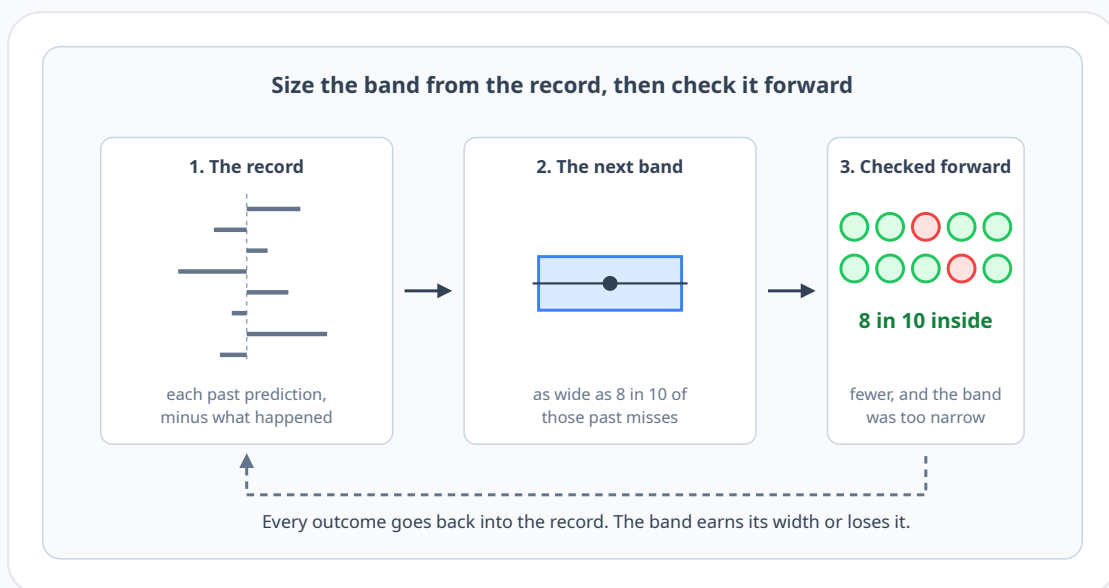
LLM has no business anywhere on the decision axis, and every use in explaining a decision that something else already made.

The promise of a world model, which in our language is a digital twin, is not that it is correct. It is that it carries mechanism: bills of materials, capacities, lead times, flows, the structural logic Tim is right to defend. A model with mechanism can be wrong in a locatable place, and it can be told where it stops. That is Tim's own debuggability argument, and I think it points at a simulator as much as at a solver.

The instrument exists, and Tim named it

Here is where we converge, and it is worth saying how far the convergence goes. Tim's answer to uncertainty is to modify the deterministic formulation (a capacity factor, a percentile input, a change penalty) and then tune those parameters with an outer simulation over realistic scenarios, offline, so the online solve stays fast. When I read his DecisionCAMP paper in August I concluded that his outer simulation and our training loop are the same loop, and I still think so. He has now said the same thing from his side. We crossed the same river from opposite banks.

What is still missing from both halves of the architecture is the instrument, and Tim named it himself: conformal prediction. Compare what you predicted with what happened, keep the misses, and size the band on the next prediction from them. If eight in ten of the past misses were smaller than a given size, a band that wide will hold about eight times in ten. No assumption that errors are normal, no assumption that the model is right. A worse model simply earns a wider band.



Tim endorses this for sizing the inputs, and calls it best practice. I would put it one level up. Not just a better distribution going in, but a standing measurement of whether the specification still matches the world, attached to every decision and checked forward against every outcome. That is the instrument the modeller is currently expected to do without. It is also the only thing I know of that tells you that you have left the regime, while you can still do something about it.

What belongs at the core

AI on the outside, mathematics at the core. I have always agreed with the shape. Where I would push Tim is on what makes a core trustworthy.

Reproducibility tells you the same inputs give the same answer. Conformance tells you the answer obeys the specification. Neither tells you the specification was right, and neither tells you when it stopped being right.

So what I want at the core is not the solver, and not the network either. It is whatever keeps being checked against what actually happened.

One thing I should say plainly, since it is the same standard I am holding Tim to. We have built that layer. We have not run it long enough in production to call it evidence. Until the outcomes say otherwise, it is architecture.

Tim, thank you for a conversation I will keep thinking about too.

See Autonomy in action

Walk through how Autonomy models, executes, monitors, and governs supply chain decisions with autonomous AI agents.

[See It Live](#)