// magnetiz.ai / field_notes / diagnostic_stack_v1

The AI GTM Engineer's Diagnostic Stack

Three metrics for the new motion.

AUTHOR	Jason Derr
ORG	Magnetiz.ai
VERSION	1.0.0
AUDIENCE	GTM operators, AI GTM engineers, hiring managers
READ_TIME	~12 minutes
STATUS	SHIPPED
COMPANION_TO	gtm_engineer_playbook_v3

// index

What's inside

A diagnostic for the metrics that broke when AI joined the motion. Built on real work, not theory.

Companion to *The GTM Engineer's Playbook v3*. **01.** The setup *(p. 03) Why the metrics that ran GTM in 2022 don't tell you what's happening now.* **02.** Three layers, three metrics, one dashed arrow *(p. 04) The trilogy and the leverage thesis.* **03.** PPO *(p. 05) Pipeline per operator. Operator-as-supervisor productivity.* **04.** PQR *(p. 08) Pipeline-to-quota ratio. Pipeline maturity as a forecast.* **05.** QRR *(p. 11) Qualified reply rate. Reply quality as a separable layer.* **06.** Closing the dashed arrow *(p. 14) Three practices to make the contingent loop close.*

// chapter 01

The setup

Why the metrics that ran GTM in 2022 don't tell you what's happening now.

Every GTM metric the industry agreed on was built for a world where one rep generated one rep's worth of output. CAC, LTV, win rate, pipeline coverage, NRR. Each one assumed a human-only motion where pipeline grew through volume and signal quality could be inferred from reply rate.

That world is gone.

In the AI-augmented motion, one operator supervises a stack of agents. Pipeline grows when systems catch deviations the team would have missed by eyeballing a Monday report. Reply rate is now actively misleading because AI-generated outreach gets more replies and a higher proportion of them are noise. **The metrics didn't get worse. They got disconnected from what's actually producing or destroying revenue.**

This playbook is three metrics for the new motion. Each one solves a problem the canon doesn't see. Each one comes with the build that makes it work in production.

Operator productivity at the human-plus-agent unit. (PPO) Pipeline maturity over time, not just coverage at a moment. (PQR) Reply quality as a separable layer from reply volume. (QRR)

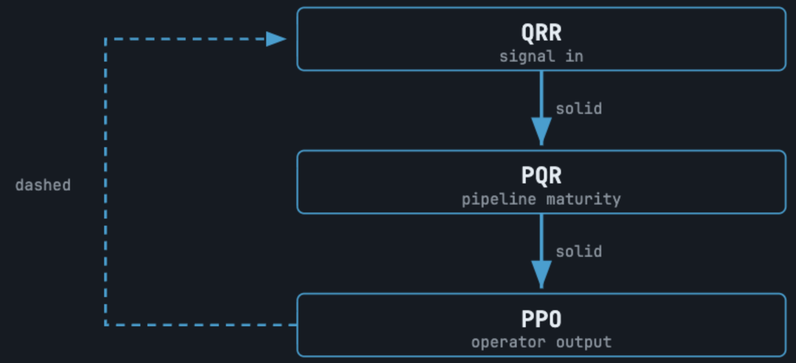
The stack is the diagnostic. The build under each metric is the leverage.

// chapter 01 / continued

Three layers, three metrics, one dashed arrow

The motion has three layers that determine whether a quarter ships. Top of funnel is signal quality. Middle of funnel is pipeline maturity. Operator level is throughput per human-plus-agent unit.

Each layer has its own diagnostic metric. Each metric has a build that makes it real-time and actionable, not just visible in a weekly review.



Two of these arrows are mechanical. Better top-of-funnel signal quality directly enriches downstream pipeline. Clearer pipeline maturity directly improves operator prioritization. Instrument the layer, the cascade follows.

The third arrow is contingent. Operator productivity feeds back into signal quality only if freed capacity gets reinvested into experimentation rather than absorbed into more of the same volume. Most orgs default to volume. **Closing this loop deliberately is what separates a GTM motion that compounds from one that plateaus.**

The chapters teach the metrics and the builds. The closer teaches how to push against funnel gravity.

// metric 01

PPO

01

Pipeline per operator. Operator-as-supervisor productivity in the AI-augmented motion.

// what this metric measures

Most teams measure productivity the way they did when one rep meant one human input. Meetings per SDR. ARR per AE. Dials per day. Each one assumed one human did one human's work.

In 2026 that's the wrong unit. An operator running three AI outbound sequences and a deal-coaching agent looks identical to a non-augmented rep on the dashboard. Same titles, same comp plan, same activity reports. Their leverage scales with the agent stack they supervise. The dashboard can't see it.

Worse, the legacy metrics actively punish the operator who shifts work to agents. Their human-input volume drops. Their output rises. Comp, ranking, and headcount decisions get made off numbers that no longer reflect what the unit is producing.

PPO without an agent denominator is a 2022 metric pretending to be a 2026 one.

PPO is qualified pipeline per operator per week, where the operator denominator is the human plus the AI workflows they supervise. Two cuts against the same opportunity table, both calculated against that same operator denominator. PPO-creation captures stage 2+ opportunities created per operator-week. PPO-maturation captures stage 4+ pipeline value moved per operator-week. Same join, different filters, two diagnostic stories.

An AI-augmented SDR might triple PPO-creation while PPO-maturation stays flat. That's the right shape for that role. An AI-augmented AE supervising deal-coaching agents might show the inverse. The two cuts make the operator visible in the unit they actually work in.

// metric 01 // build

PPO

// the build

A weekly batch query that joins your opportunity table to ownership and aggregates by operator and week, where operator is the unit you're actually compensating (the human plus the agent stack they supervise). Two views off the same join. PPO-creation counts stage 2+ opportunities created. PPO-maturation sums stage 4+ pipeline value moved.

The artifact that proves the metric is the before/after curve. Instrument PPO across a cohort. Introduce one AI workflow. Measure the lift over 6-8 weeks. Supervision moves from anecdote to measured curve. The curve shape is leadership's case for workflow ROI and the operator's case for comp.

```
-- ppo.sql
-- creation: stage 2+ opps created per operator-week
with creation as (
  select OwnerId as operator,
         date_trunc('week', CreatedDate) as week,
         count(*) as opps_created
  from Opportunity
  where StageName not in (
    'Prospecting','Qualification')
  group by 1, 2
),
-- maturation: value moved into stage 4+ per operator-week
maturation as (
  select o.OwnerId as operator,
         date_trunc('week', h.CreatedDate) as week,
         sum(o.Amount) as value_moved
  from OpportunityFieldHistory h
  join Opportunity o on o.Id = h.OpportunityId
  where h.Field = 'StageName'
    and h.NewValue in (
      'Proposal','Negotiation','Closed Won')
  group by 1, 2
)
select operator, week, opps_created, value_moved
from creation
full outer join maturation using (operator, week)
```

// metric 01 // unlocks + diagnostic

// what this unlocks

- Workflow ROI you can show to a CFO
- Compensation tied to actual output, not activity
- Capacity planning that asks the right question first (max PPO before max headcount)
- A teachable skill called supervision

// diagnostic

- Can you point to one operator whose PPO went up after they adopted an AI workflow? By how much, over what window?
- When you compare PPO-creation to PPO-maturation per operator, do they tell different stories?
- How many operators on your team supervise more than 2 agents today? What's their PPO vs the team median?
- If you 2x'd headcount tomorrow with the same agent stack, would PPO go up, down, or stay flat?

// metric 02

PQR

02

Pipeline-to-quota ratio. Pipeline maturity as a forecast, not a coverage check.

// what this metric measures

Most teams measure pipeline coverage as a single moment. Open pipeline at each stage, divided by quota. 4x feels safe. 5x feels comfortable. The team forecasts off that number until two weeks before quarter-end, when reality catches up.

The problem isn't the math. It's the snapshot. Coverage at a single moment tells you nothing about whether what's in the pipe will mature in time to close. You can have 5x coverage with most of it in stage 1 and still miss because stage 1 doesn't convert to closed-won in 8 weeks.

PQR fixes this by adding two things to coverage. First, it calculates against cumulative pipeline by stage. Stage 4+ PQR sums everything from stage 4 forward (stage 4 + stage 5 + stage 6 + closed won) and divides by quota. That captures pipeline maturity, not just pipeline volume.

Second, it gets calculated daily and snapshotted against history. That turns the metric from a snapshot into a forecast.

PQR coverage without snapshots is forecasting theater.

Nikko Lubrano built the original PQR concept at Hightouch, citing Tomasz Tunguz's "Sales Sandwich" framework as the inspiration. What follows is the early-warning layer that turns the metric from a measurement into an alarm system.

// metric 02 // build

PQR

// the build

Step one is the live calculation. Aggregate cumulative pipeline by stage from your opportunity table, divided by quarterly quota. Build the same query for every stage cut you care about (stage 1+, stage 2+, etc.). Now you have a real-time view of maturity coverage by stage.

Step two is daily snapshots. The live query gives you coverage at a moment. The forecasting power comes from watching the shape over time. dbt snapshot models are the cleanest path. They take a daily picture of your opportunity table and add a snapshot date column. Query the snapshot like you'd query the live table, just filter by date.

Step three is the early-warning layer. Compare today's PQR shape to the same day-in-quarter from the last 6-8 quarters. Significant deviations (>1.5σ from historical mean) post to a Slack channel with historical analog context. The analog matters. "Last time S2+ PQR deviated this much on day 35, we missed quota by 12%" is an alert someone acts on. A bare deviation number is decoration.

```
# pqr_anomaly.py
# detect PQR deviations on the daily snapshot stream

baseline = pqr_history.filter(
    day_in_quarter=current_day,
    stage_cut='stage4+',
    quarters=last_8
).agg(mean=mean(pqr), sd=std(pqr))

current = pqr_today.filter(stage_cut='stage4+')
z = (current.pqr - baseline.mean) / baseline.sd

if abs(z) > 1.5:
    analog = pqr_history.find_closest(
        pqr=current.pqr,
        day_in_quarter=current_day,
        stage_cut='stage4+',
    )
    slack.post(
        channel='#pqr-alerts',
        deviation=z,
        analog_quarter=analog.quarter,
        analog_outcome=analog.eoq_miss_pct,
    )
```

// metric 02 // unlocks + diagnostic

// what this unlocks

- Forecast accuracy within ±5% by end of month 1 instead of end of quarter
- Lead time of 2+ weeks ahead of standard pipeline reviews
- Conversation shifts from "are we behind" to "which intervention works"
- Quarterly retros become evidence-based (which deviation patterns predicted which misses)

// diagnostic

- Can your team spot a PQR deviation 2 weeks before quarter-end?
- When current-quarter shape diverges from historical baseline, does anyone get pinged?
- Are your PQR alerts read-only, or do they trigger interventions?
- Can you point to a quarter where the snapshot layer caught a miss in time to fix it?

// metric 03

QRR

03

Qualified reply rate. Reply quality as a separable layer from reply volume.

// what this metric measures

Most teams measure reply rate. Sends in the denominator, replies in the numerator. Sequence A gets 8% reply rate. Sequence B gets 12%. B wins. The team optimizes against that number.

In 2026 the number is misleading. AI-generated outreach gets more replies than human-written outreach. A higher proportion of those replies are noise. "Wrong person." "Take me off this list." "Out of office until next month." Auto-responders. Reply rate goes up. Pipeline doesn't.

The team is optimizing a metric that's no longer connected to outcome. Worse, every iteration makes the disconnect deeper. Better-replying templates get prioritized regardless of what kind of replies they generate.

QRR is the new MQL.

Inbound figured this out a decade ago. Lead volume isn't the goal. Marketing-qualified leads are. The same evolution is overdue on the outbound side.

QRR is qualified replies divided by sends. The numerator depends on a classifier that tags every reply across eight categories (genuine interest, soft interest, wrong person, referral, unsubscribe, hostile, auto-responder, noise). Qualified is the first three. Everything else is routing, cleanup, or signal that the targeting is off.

This metric isn't standardized yet. Reply rate is canonical. Qualified reply rate is what most teams need but few measure. Naming it formally is a position. So is shipping the classifier that makes it computable.

// metric 03 // build

QRR

// the build

Step one is the classifier. Take 200-300 sample replies from your existing outbound and label them by hand across the eight categories. Use those labels to build a few-shot prompt for Claude or GPT, or fine-tune a smaller model if your volume justifies it. Either approach works. The classifier reads each new reply and outputs a category in under a second.

Step two is the metric. Once the classifier is running, QRR is one division. Qualified replies (genuine interest + soft interest + referral) over total sends, calculated weekly per sequence, per segment, per rep.

Step three is the feedback loops. The classifier is more valuable than the metric. Every classified reply is a labeled data point. Pipe positive classifications back into the signal layer to reinforce what's working. Pipe negative classifications (wrong person, hostile) into ICP refinement. Pipe auto-responder rates into deliverability monitoring. The classifier is the artifact; the metric is one of several things you do with it.

```
# qrr_classifier.py
# label every reply, then compute QRR per slice

CATEGORIES = [
    'genuine_interest', 'soft_interest',
    'wrong_person', 'referral',
    'unsubscribe', 'hostile',
    'auto_responder', 'noise',
]

QUALIFIED = {
    'genuine_interest',
    'soft_interest',
    'referral',
}

def classify(reply_text):
    return claude.classify(
        text=reply_text,
        categories=CATEGORIES,
        examples=labeled_examples,
    )

def qrr(slice_filter, week):
    sends = outbound.filter(slice_filter, week).count()
    replies = inbound.filter(slice_filter, week)
    qualified = sum(
        1 for r in replies
        if classify(r.text) in QUALIFIED
    )
    return qualified / sends
```

// metric 03 // unlocks + diagnostic

// what this unlocks

- Sequence quality scoring beyond reply rate
- Empirical ICP refinement (segments with highest QRR are your real ICP)
- A/B testing copy with a signal that correlates to pipeline
- AI drafting agent supervision (is QRR climbing or decaying as the agent runs?)
- A reusable classifier you can plug into routing, scoring, and reporting

// diagnostic

- What % of your outbound replies are genuinely interested vs. routing or noise?
- When you A/B test sequences, are you measuring against reply rate or against QRR?
- Which segments of your ICP produce the highest QRR? If you don't know, your ICP isn't validated.
- Can you point to a sequence that beat reply rate but lost on QRR?

// chapter 04

Closing the dashed arrow

04

The third arrow doesn't close itself.

// the leverage thesis revisited

Two of the three arrows in the diagnostic stack are mechanical. Better top-of-funnel signal quality enriches downstream pipeline. Clearer pipeline maturity improves operator prioritization. Instrument the layer, the cascade follows.

The third arrow is contingent. Operator productivity feeds back into signal quality only if freed capacity gets reinvested into experiments that test new signal hypotheses. Most orgs default to volume. Capacity gets absorbed into more sequences, more touches, more meetings booked. The system stays at its baseline signal quality and the compounding never happens.

The default has gravity. Reps optimize for activity numbers because that's what comp plans reward. Managers optimize for pipeline targets because that's what boards measure. Nobody on the team has a job description that says "make the signal better." The dashed arrow stays dashed.

This closer is three operational practices that close it. Each one is small, concrete, and runnable in the next two weeks. None of them require headcount or budget. They require the deliberate choice most teams don't make.

// chapter 04 // continued

Three practices

Practice 01. Protect 10-15% of operator capacity for signal experiments.

Calendar block. Recurring. Priority of a customer meeting, not "if there's time leftover." A 40-hour operator protects 4-6 hours weekly for testing signal hypotheses (a new signal source, a new ICP slice, a new enrichment approach, a new opener). One experiment per cycle is the minimum bar. Without protected time, the work falls back to the default.

Practice 02. Tag every experiment and measure outcomes against QRR.

Reply rate will lie here for the same reason QRR exists. An experiment that drives reply rate up but QRR flat isn't a win. Tag each experiment in your outbound system, run the classifier against the replies, and after 4-6 weeks of data the experiment either improved QRR or it didn't. Bury the ones that didn't.

Practice 03. Quarterly retro on signal quality.

90-minute meeting, every quarter. Looks at every experiment that ran, what QRR shifted, and what to retire vs scale. The output is a one-page update to the signal library and the ICP definition. Without the retro, experiments accumulate but the system never gets smarter.

// chapter 04 // closing read

The diagnostic stack measures three things. PPO catches operator productivity at the human-plus-agent unit. PQR catches pipeline maturity before the quarter is gone. QRR catches reply quality where reply rate fails.

The first two work because the metrics make the layers visible. The third works because the operators build the loop deliberately.

Most teams will instrument PPO and PQR and stop there. The teams that compound will close the dashed arrow.