Constellation and The Unknown (Part 3)

Is Selective Censoring Happening on Solana?

This is one of the most difficult questions to answer.

There is no meaningful way to lets everyone agree on when a transaction first became available to a leader. The only observable is when a tx is executed on-chain.

This makes basically impossible to answer direct questions like

> Did this validator receive the transaction earlier and intentionally delay it?

However, there is another, still useful, question

> Do some validators execute certain swaps later than we would expect relative to comparable swaps and comparable blocks?

This report asks whether some Solana validators execute specific swaps later than expected under a transparent on-chain model.

Because we do not observe when a transaction first became available to a leader, we cannot directly prove that a validator intentionally delayed it. We therefore study a weaker but measurable object: whether the final on-chain placement of certain swaps is systematically later than expected relative to comparable swaps and comparable blocks.

We construct a latent-displacement model. The model converts non-vote transaction order into coarse timing chunks, treats each swap's true availability time as hidden, and estimates whether a validator's observed swap placements require additional latent delay relative to the class baseline.

We combine three diagnostics: directional displacement, normalized likelihood improvement, and KL divergence from the baseline distribution. A candidate validator-class pair must be late-directed, distributionally unusual, better explained by an additional-delay model, and robust across multiple model widths.

The evidence identifies a subset of validator-class pairs with leader-specific excess latent displacement. This is consistent with additional delay, but it does not prove intentional censorship. The result should be interpreted as a prioritized candidate set for further investigation.

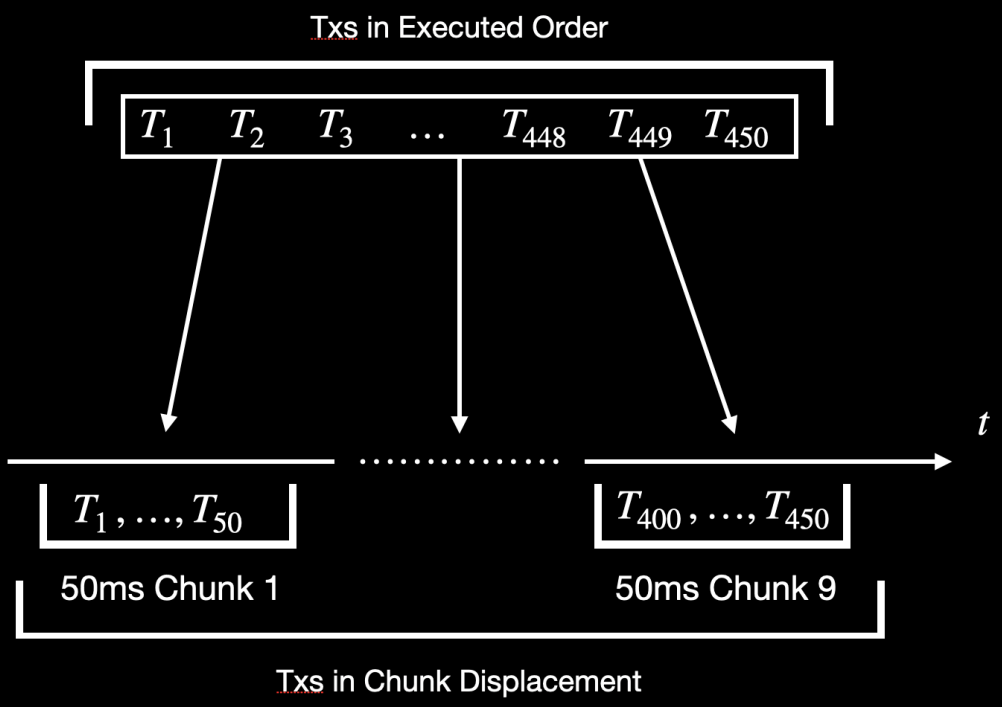
The Latent Displacement Model

If we identify each transaction with an event, and we assume that the mean rate of these event arrival is fixed (the chain TPS), we can model transactions arrival in any given window Δ with a Poisson distribution.

Clearly, Poisson assumes independence between occurrences, which in the case of specific swaps may not being accurate. Still, the block contains a sparse category of transactions, meaning we can still use this approximation if we consider the whole block structure.

That is, let's consider a generic slot s and divide it in B_s chunks of $\Delta = 50\,ms$.

So if a slot has around 450 non-vote transactions, the model divides it into roughly nine chunks.



It is worth noting that, this is a way to convert transaction order into a coarse timing coordinate under some assumptions:

> Given the number of non-vote transactions in this slot, where would this swap fall if we divided the non-vote flow into equal 50ms-sized chunks?

That gives each swap an observed chunk position.

Under our Poisson assumption, with 1000 TPS, the expected number of them in each chunk would be

$$\lambda = TPS \cdot \frac{\Delta}{1000} = 50\,.$$

If we denote with M_s the number of non-vote txs in the slot, we can compute the number of chunks as

$$B_s = \left\lceil \frac{M_s}{\lambda} \right\rceil .$$

It follows that, the observed chunk of transaction i can be determined as

$$Y_i = \left\lceil B_s \frac{r_i}{M_s} \right\rceil ,$$

where r_i is the order of the considered transaction.

At this point we focus on a given class of swaps, and assign to each of the an observed position into a chunk.

Here we consider swaps against BisonFi SOL-USDC market, maintaining the two directions as distinct observables.

Fixed the category, we can compute the entropy of swap displacement for each leader given a sample of slots. That is, denoting the leader with l an the class with C, we can define the empirical distribution of swaps as

$$q_C^l(y) = \frac{n_y^C}{\sum_{h=1}^{B_s} n_h} ,$$

where n_i^C represents the number of swaps in each chunk of a given class C. From this we can define the entropy of the distribution as

$$H(q_C^l) = -\sum_{y=1}^{B} q_C^l(y) \log q_C^l(y) ,$$

which has a useful interpretation:

- If the swaps concentrate in few chunks, the entropy will be small
- If the swaps spreads on all chunks, the entropy will be high.

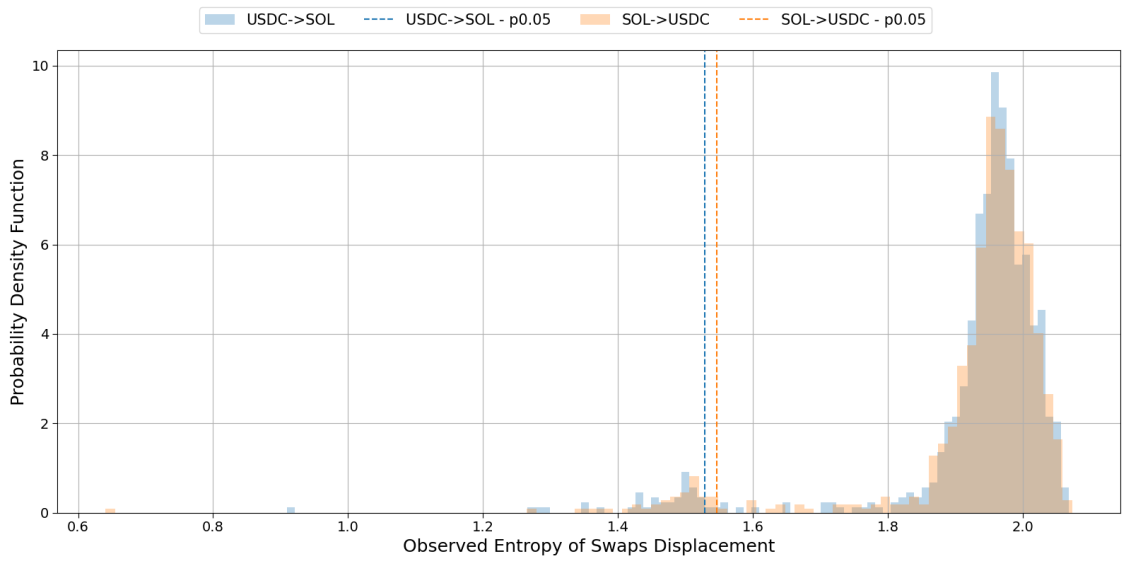
It is worth noting that the entropy alone doesn't tell us if a tx has been delayed, just that txs are concentrated.

Our first observation is that, under our chunk assumption we observe a long tail in the entropy distribution. Precisely, we observe a major bulk of the entropy distributed around 1.8-2.0, with a small bulk of entropy distributed between 1.4 and 1.6.

This is an interesting result since it's telling us that

> Majority of validators are showing high entropy, meaning fair spread of swaps among chunks. A subset of validators shows a high localization (i.e. low entropy).

The share of slots produced by the validators in the p0.5 tail (i.e. showing "concentrated swaps") is ~9% over total slots for both markets. All validators in the "anomalous region" were running the same client during the sampling phase.



This still doesn't help us identifying if the validators in the tail are actually delaying swap transactions. For this we need a model since we can't directly measure when the transaction was made available.

The Hidden Variable Model

Until now, we defined a set of observables. However, we don't know what is the true chunk in which the transaction became available to the leader, which this is not directly measurable on-chain.

Precisely, for each swap, there is some hidden moment when it became available to the leader. Maybe it was triggered by a market signal. Maybe by a routing decision. Maybe by a private strategy. We do not observe that.

Let's consider the following *Gedankenexperiment*.

Imagine a set of packages arriving at a warehouse. We do not know when each package arrived at the loading dock. We only see when it was scanned into the system.

If a package is scanned late, there are two possible explanations:

1. it arrived late
2. it arrived earlier but waited before being scanned

With only scan records, we cannot prove which one happened for a single package.

However, we can look across many packages.

If packages of a certain type are repeatedly scanned later than similar packages under one warehouse manager, that is a signal. It is not proof of intentional delay, but it is evidence of systematic late placement.

Here we can assume that, the hidden trigger could have occurred in any chunk of the slot. This is a maximum-ignorance assumption. It does not claim that triggers are truly uniform in reality. It simply gives us a neutral starting point when no external trigger data is available.

Then the swap is executed some number of chunks after that hidden trigger.

That difference is the swap's latent displacement.

Precisely, let's define this (not observed) latent arrival chunk as

$$G_i \sim \text{Uniform}\{1,\dots,B_s\}\,.$$

We can define the latent displacement as

$$D_i = Y_i - G_i\,, \qquad D_i \geq 0\,.$$

For clear reasons, the model doesn't allow execution before arrival.

Once the transaction is made available to the leader, we can imagine that it's executed in some non-negative displacement determined by some half-normal distribution. That is, the baseline kernel over non-negative displacement d is

$$k_0(d;\sigma) \propto \exp\left(-\frac{d^2}{2\sigma^2}\right), \qquad d \in \{0,1,2,\dots\}\,.$$

Intuitively, a small σ makes the baseline expected transaction to be executed close to their latent arrival chunk. A large σ, on the other hand, allows broader baseline displacement.

At this point, we can model a delay in execution via an exponential tilt

$$k_\alpha(d;\sigma) \propto k_0(d;\sigma)\exp\left(\alpha d\right) = \exp\left(-\frac{d^2}{2\sigma^2} + \alpha d\right)\,.$$

The parameter α tilts the mass towards larger displacement. Higher α means longer latent displacement.

It is worth mentioning that raw α is not directly comparable across σ since the effect scale clearly depends on σ.

A useful way to think about it is

> σ controls how forgiving the baseline is about normal displacement.

If σ is small, the model expects swaps to be executed close to the chunk where they became available. If a swap appears later, the model treats that as more surprising.

If σ is large, the model is more forgiving. It allows swaps to naturally appear several chunks after their hidden arrival point without immediately calling that unusual.

So σ is not "the delay." It is the width of the baseline expectation.

At this point, the probability of observing execution chunk y, given $B = b$, $G \sim \text{Uniform}\{1,\dots,b\}$, and $D \sim k_\alpha$ is

$$p_\alpha(y \mid b,\sigma) = \frac{\sum_{d=0}^{y-1} k_\alpha(d;\sigma)}{\sum_{d=0}^{b-1}(b-d)k_\alpha(d;\sigma)}\,,$$

where the numerator accounts for all displacements compatible with $Y = y$, while the denominator normalizes over all valid (G, D) combinations that keep $Y \leq b$.

We can define then the log-likelihood for each trade class C and σ

$$\mathscr{L}_{C,\sigma}(\alpha) = \sum_{i \in C} \log p_\alpha(Y_i \,|\, B_i, \sigma)\,.$$

From here we can compute the baseline delay as

$$\hat{\alpha}_{C,\sigma} = \arg\max_{\alpha} \mathscr{L}_{C,\sigma}(\alpha)\,.$$

For each leader l and class C, the leader likelihood is

$$\mathscr{L}_{l,C,\sigma}(\alpha) = \sum_{i \in (l,C)} \log p_\alpha(Y_i \,|\, B_i, \sigma)\,,$$

so we can estimate the delay specific for the leader as

$$\hat{\alpha}_{l,C,\sigma} = \arg\max_{\alpha \geq \alpha_{\hat{C},\sigma}} \mathscr{L}_{l,C,\sigma}(\alpha)\,.$$

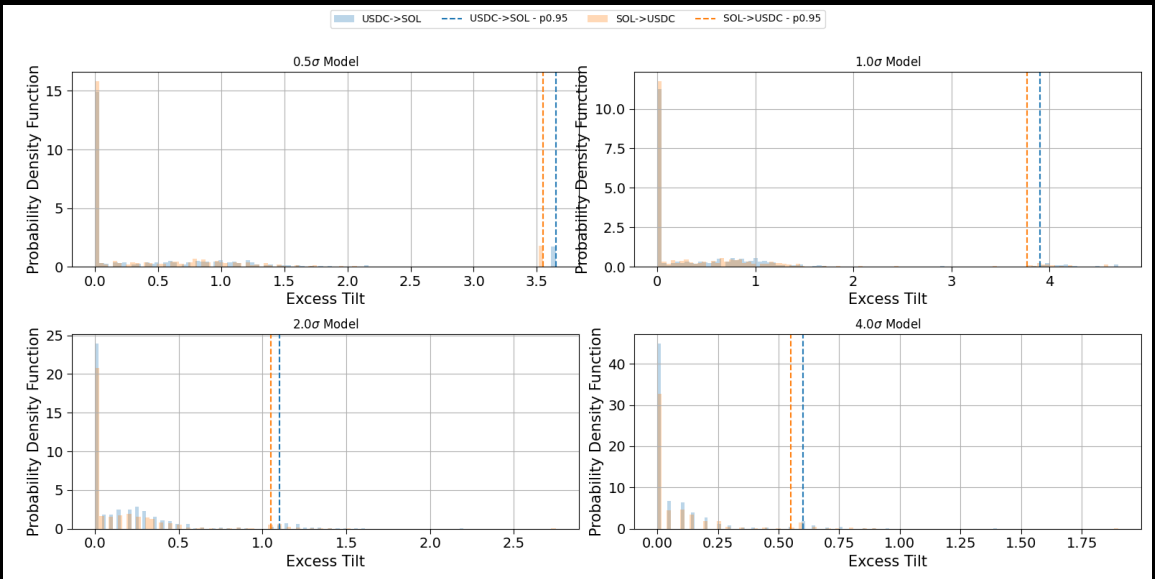
This non-negativity constraint means that the fit only searches for additional delay relative to the class baseline. The excess tilt is

$$\delta_{l,C,\sigma} = \hat{\alpha}_{l,C,\sigma} - \hat{\alpha}_{C,\sigma}\,.$$

So the model first learns the normal placement pattern for the class. Then it asks if the specific leader look more delayed than the class baseline.

This is a relative test.

A leader is not called late because its swaps are late in absolute terms. It is called late only if its swaps are later than expected for that same class.



We can see that the distribution has the majority of the mass around 0, with small tails. Again, around ~9% of the slots belong to leader that show an excess tilt above p95.

It is worth noting that, by construction $\delta_{l,C,\sigma} \geq 0$. Therefore, the excess tilt tells us whether the best delayed leader-specific model moves away from the class baseline. However, it alone is not sufficient as an effect-size measure.

To test the goodness of fit we employed the likelihood-ratio statistic

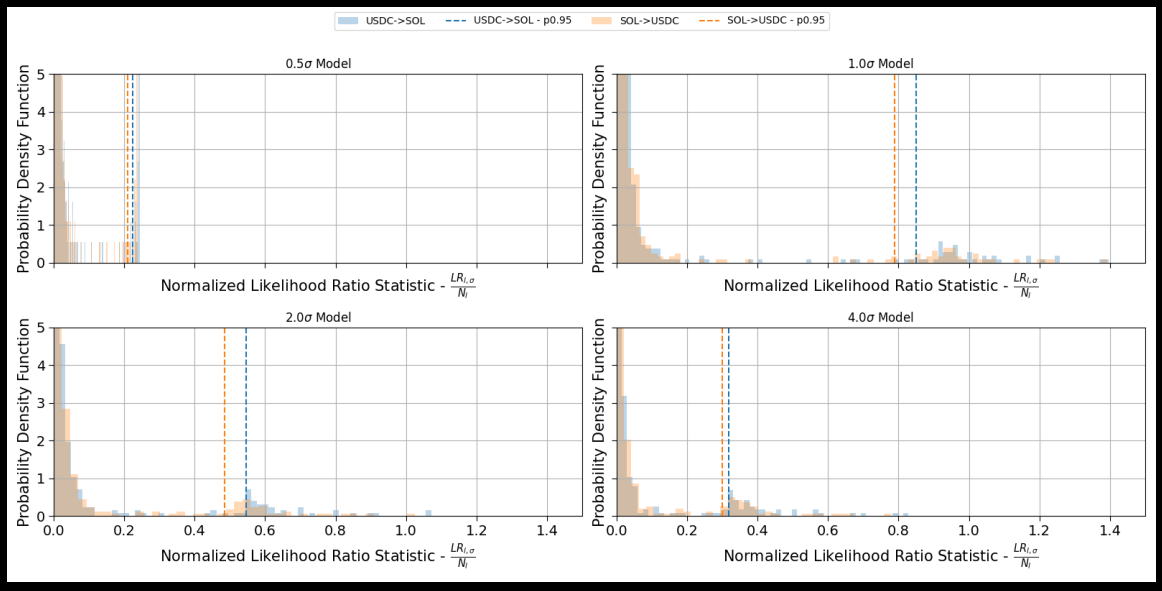
$$LR_{l,\sigma} = 2\left[\mathscr{L}_{l,\sigma}(\hat{\alpha}_{l,\sigma}) - \mathscr{L}_{l,\sigma}(\hat{\alpha}_{\sigma})\right] .$$

Since LR grows with sample size, the analysis also uses the normalized variant

$$\frac{LR_{l,\sigma}}{N_l},$$

where N_l is the number of observed swaps in the leader class group.

The normalized likelihood-ratio measures the per-observation improvement obtained by the leader-specific delayed model relative to the class baseline. Since $LR = 2\Delta\mathscr{L}$, a value $LR/N \sim 1$corresponds to an average log-likelihood gain of 0.5 nats per observed trade. Equivalently, the delayed model assigns the observed chunk positions a geometric mean probability about $e^{0.5} \sim 1.65$ times larger than the class baseline.



We see stability of the anomalous group on all tested values of displacement dispersion. This suggests that the low-entropy subgroup is not merely a visualization artifact: a subset of the same validators also appears anomalous under the latent-displacement likelihood model. We therefore treat this subgroup as a candidate set for further investigation, not as proof of censorship.

By considering the share of slots proposed by the validators showing an anomalous (> p95) normalized likelihood ratio test, we see a 13% share for the USDC->SOL market and a ~10% for SOL->USDC market. Again, all validators in the “anomalous region” were running the same client during the sampling phase.

Mean Posterior Directional Displacement

The tests done so far measure model fit without providing information about directional displacement. For this reason, we define a posterior displacement residual.

Our modelling helps as to compare implied displacement with what the class baseline would have expected. So, for each swap we can define displacement residuals

- positive if it looks more delayed than baseline
- negative if it looks less delayed than baseline.

Positive values of this estimator mean the leader-class pair appears later than expected under the class baseline. Negative values mean relative early placement compared with the class baseline.

Formalizing that, for an observed chunk $Y = y$, the posterior mean displacement under parameter α is

$$m_\alpha(y, b; \sigma) = \mathbb{E}[D \mid Y = y, B = b] = \frac{\sum_{d=0}^{y-1} d\, k_\alpha(d; \sigma)}{\sum_{d=0}^{y-1} k_\alpha(d; \sigma)}.$$

The expected posterior displacement under the baseline is

$$\bar{m}_{\hat{\alpha}}(b; \sigma) = \sum_{y=1}^{b} p_{\hat{\alpha}}(y \mid b, \sigma) m_{\hat{\alpha}}(y, b; \sigma).$$

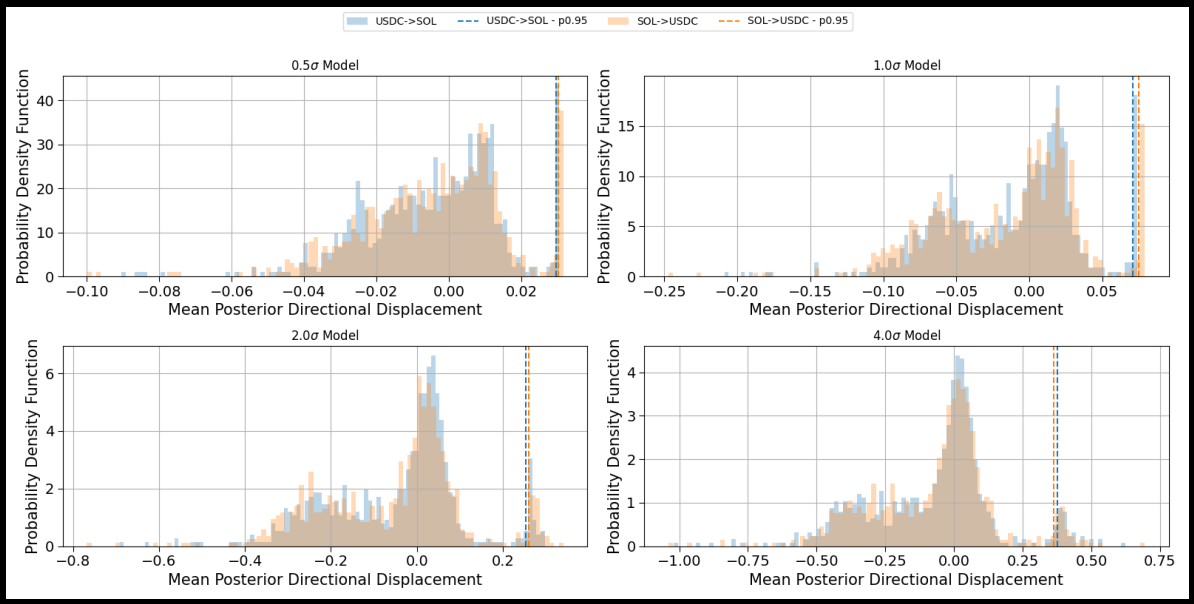
Thus, for each trade i we can define the trade level residual

$$u_i = m_\alpha((Y_i, B_i; \sigma) - \bar{m}_{\hat{\alpha}}(B_i; \sigma).$$

Aggregating by leader (for a fixed class)

$$U_{l,\sigma}^{mean} = \frac{1}{N_{l,\sigma}} \sum_i u_i,$$

we get our mean posterior displacement residual.



The strict models ($\sigma = 0.5$ and $\sigma = 1.0$) test whether delay-like behaviour appears under a local displacement baseline. A small σ means that the baseline kernel expects execution to occur close to the latent arrival chunk. If most validators remain near or below 0 then the data do not support a broad network-wide delay.

Instead, the evidence is concentrated in a positive upper tail.

At this point the path is clear.

A validator-class pair that only appears anomalous at $\sigma = 0.5$ is weaker. A pair that remains anomalous at both $\sigma = 1$ and $\sigma = 2$, and preferably also at either $\sigma = 0.5$ or $\sigma = 4$, is much more credible.

Distribution Discrepancy Measure

The directional displacement statistic tells us whether a validator-class pair looks later or earlier than the class baseline.

This is useful, but it is not the full story.

A validator may have a positive directional displacement because many swaps are slightly shifted later, or because a small number of swaps are concentrated in an unusual part of the block. These two cases have different interpretations. In the first case, the whole distribution is gently moved. In the second case, the distribution has a strange shape. If we only look at direction, we might miss this distinction.

This is why we also need a measure of distributional deviation. The question is not only:

> Are the swaps later than expected?

but also:

> Does the whole observed chunk distribution look different from what the baseline model predicts?

Let's consider the following *Gedankenexperiment*.

Let's consider again the warehouse example. Suppose we observe when packages are scanned.

The directional displacement tells us whether packages under a certain warehouse manager are scanned later on average. But this average alone does not tell us whether all packages are slightly late, or whether most packages are normal and a smaller group is scanned in a highly unusual time window.

To distinguish these cases, we need to compare the full observed scan-time distribution against the expected distribution.

In our setting, this means comparing the observed distribution of swap chunks under a validator with the chunk distribution predicted by the class baseline. If the two distributions are close, then even a small positive displacement may not be very informative. If they are far apart, then the validator-class pair is not only directionally shifted; it is also distributionally unusual.

We use Kullback-Leibler (KL) divergence for this purpose. KL does not say whether the swaps are late or early. It only says how different the observed distribution is from the baseline. Therefore, KL should not be interpreted alone as a delay signal. Instead, it complements the directional statistic.

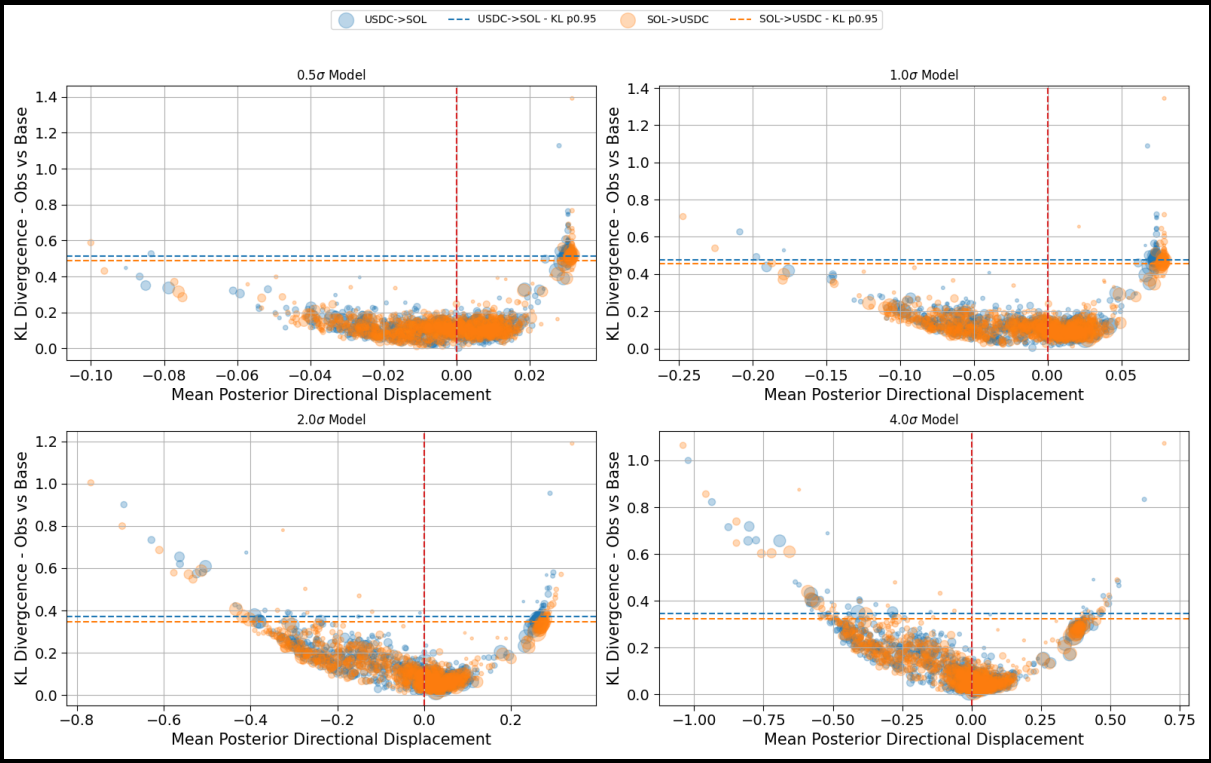
Formally, for a validator l and a class C, we can define the baseline joint distribution

$$q^{base}_{l,C,\sigma}(b,y) = q_{l,C}(b)p_{\hat{\alpha}_{C,\sigma}}(y\,|\,b,\sigma)\,.$$

From this, we can define the KL divergence as

$$D_{KL}\left(q_{l,C}(b,y)\,||\,q^{base}_{l,C,\sigma}(b,y)\right) = \sum_{b,y} q_{l,C}(b,y)\log\frac{q_{l,C}(b,y)}{q^{base}_{l,C,\sigma}(b,y)}\,.$$

This choice is important: the KL comparison does not penalize a validator merely because it has a different empirical distribution of B. Instead, it conditions on the validator's observed B-mix and asks whether, given B, the observed chunk positions Y look different from the baseline model.



What makes it useful is the combination. If a validator has positive directional displacement, then its swaps look later than expected. If it also has high KL, then the pattern is not only later; it is also distributionally different from the baseline. And if the delayed model also improves the likelihood per trade, then the evidence becomes more coherent: the validator's swaps are late-directed, unusually distributed, and better explained by an additional-delay model.

Detecting the Anomaly

At this point, we have all ingredient to detect which validators show an excess latent displacements.

The three diagnostics used in the final anomaly rule measure different aspects of the same phenomenon:

1. **Mean Posterior Directional Displacement:** It helps us to identify the class of validator with measured excess directional displacement with respect to the baseline
2. **Normalized Likelihood Ratio:** It tells us whether the additional-delay model improves the fit after accounting for sample size.
3. **KL Divergence:** It tells us whether the observed chunk distribution differs in shape from the class baseline.

A leader-class row is therefore considered anomalous only if all three diagnostics are simultaneously high relative to comparable rows.

We used as thresholds the 95th percentiles of these tree quantities.

We finally take the intersection of validators showing anomalous behaviour across all parameter models. Taking the intersection across model widths strengthens the evidence because it reduces dependence on a single hyperparameter choice. If a validator-class pair remains anomalous when the displacement kernel is made both stricter and more permissive, the result is less likely to be a numerical artifact of one σ.

The evidence does not establish intentional censorship. The arrival time of each swap remains latent, and the model cannot prove that a leader had access to a transaction before executing it.

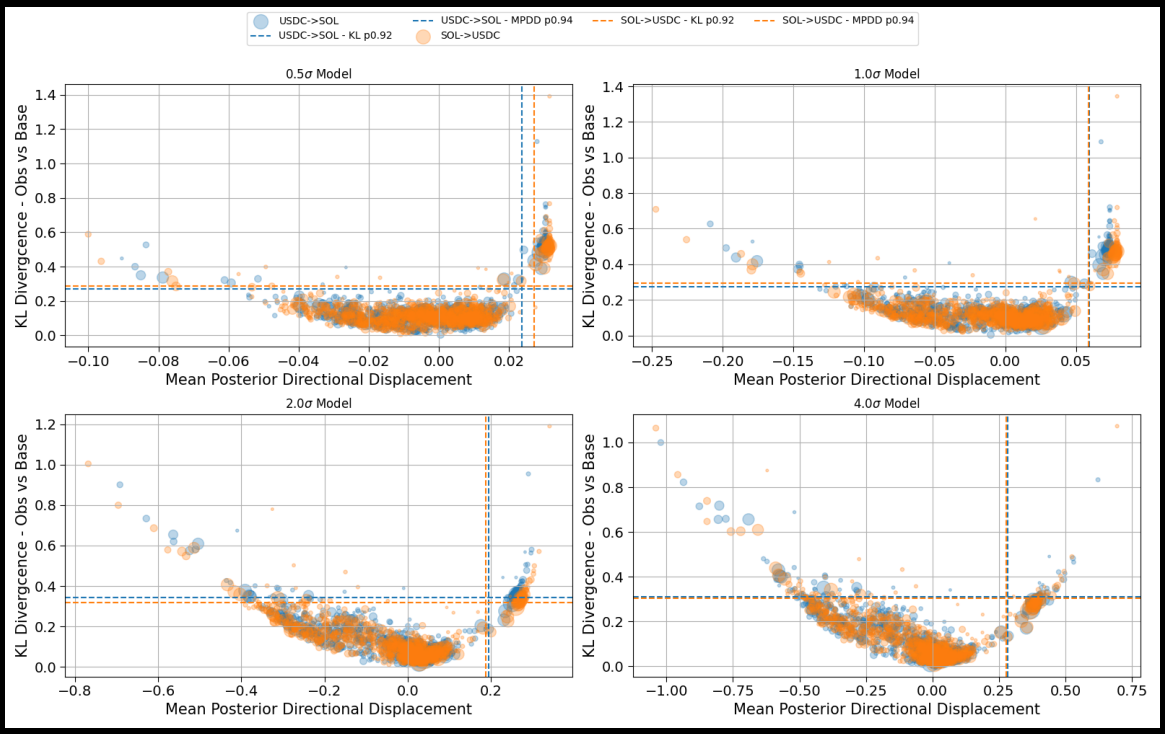
However, after constructing a non-vote transaction timing coordinate and fitting a class-relative latent-displacement model, we find a subset of 37 validator-class pairs whose observed swap placements are simultaneously late-directed, distributionally anomalous, and better explained by a one-sided additional-delay alternative. All validators in the anomalous region are running the same client.

It is important to mention that not all validators running this client show this feature, and some of them appears to be in the right opposite regime. This pattern is consistent with a configuration-dependent, rather than a property shared by all validators running the same client. However, the current analysis cannot identify the mechanism.

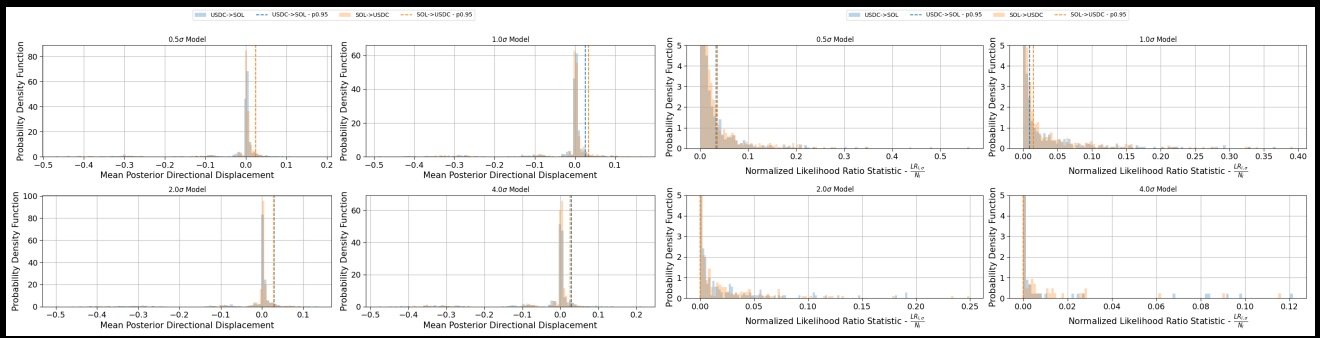
We interpret these as candidate cases of leader-specific excess latent displacement, warranting further investigation.

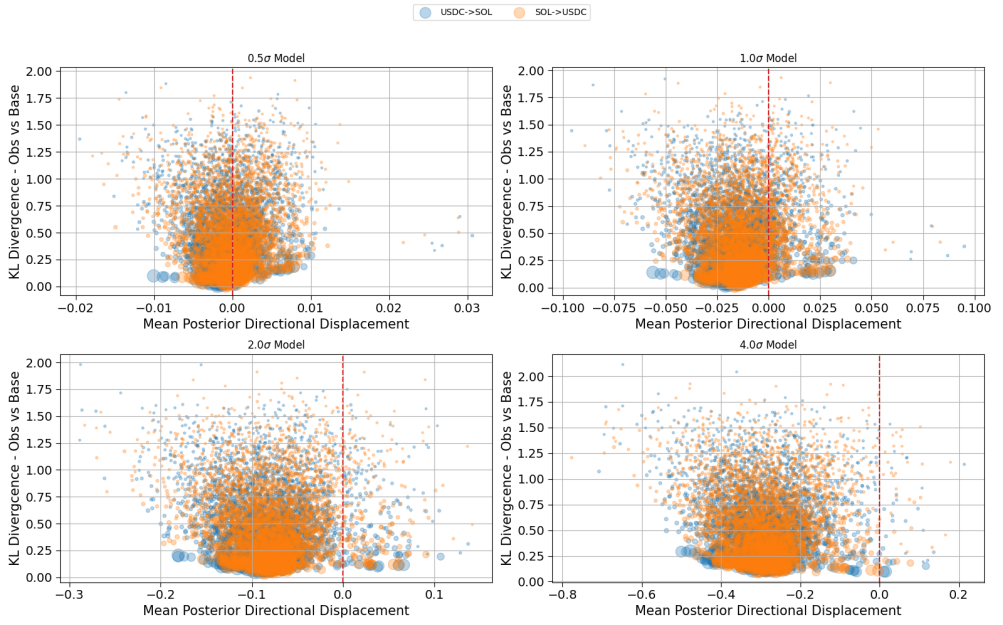
We now relax the assumption of anomalous being above 95th percentile and consistent across all model parameters.

Precisely we focus on the $\sigma = 1$ model, and isolate the cluster of validators detached from the global distribution. Here we get 48 validators, 1 slot lagger, UMi1…, 3 rev strat, 1 another client. Thus again the majority of anomalous behaviour were from one single client.



We repeated the analysis using another propAMM as class baseline, and we don't see the same tail behaviour.





Conclusions

We cannot see the moment when a swap first reached the leader. That means we cannot look at a finalized block and directly say: this validator received the transaction earlier and intentionally delayed it.

But we can still ask whether the final placement of swaps looks normal.

The model in this report does exactly that. It builds a timing coordinate from non-vote transaction order, learns a baseline placement pattern for each swap class, and then checks whether some validators place those swaps later than expected.

The answer is not that all validators look suspicious. They do not. Most validator-class pairs remain close to baseline. The interesting signal is in the tail: a subset of validators whose swaps are later-directed, distributionally unusual, and better explained by an additional-delay model.

> That is not proof of censorship. It is a candidate set.

The practical value of this model is that it narrows the search. Instead of asking whether censorship is happening somewhere on Solana, we can ask a much sharper question:

> Why do these validators, for these swap classes, produce this late-placement pattern?

Answering that requires the next layer of evidence: first-seen measurements.

Until then, the right conclusion is careful but meaningful:

> the on-chain data reveal a subset of validators with swap placement patterns consistent with additional latent delay. This is not a verdict, but it is a strong reason to investigate further.