Drift Switch Index (DSI) Validation Report

Report Date : 4 October 2023

Squad Members

Area	Person In Charge
Squad Leader	Harsh Karamchandani (harsh@regentmarkets.com)
Project Manager	Maria Semashko (maria.semashko@regentmarkets.com)
Product Owners	Simo Dafir (simo.dafir@regentmarkets.com) Clément David (clement@deriv.com) Nolan Albanet (nolan@deriv.com)
Backend	Afshin Paydar (afshin.paydar@deriv.com)
Model Validation	Vishal Menon (vishal.menon@deriv.com)

Product Information

Product Description

Product Specifications

- BE specs: https://docs.google.com/document/d/1YAQsODIAgNsGAeVUwAKVsDImx4riL5xTdQA9NAegwBE/edit
- BO specs: https://docs.google.com/document/d/13lLR7P7x-nPd4F37sjsl8GGSPBM4vm_RtD0XRXBnZqE/edit
- R&D: https://drive.google.com/file/d/11cJisiUGQ7QsyWiNZ-R8mNKXMpJgmoEr/view
- Perl Code (10 July 2023): https://github.com/regentmarkets/perl-Feed-Index-DriftSwitch/tree/0618f4ca0dc5ae51821bc6441901f2e0199ea5f1

The Drift Switch Index simulates the financial markets with constant volatilities and variable drift. It is an example of a Markov regime switching model. In particular, it is a geometric brownian motion (GBM) whose drift is driven by a discrete Markov process, causing a positive, negative or neutral drift (trend).

The index is generated every second.

The DSI mimics a typical market where trends occur by speculation. For example:

- Upward trends represent bullish speculation or high confidence in the index
- Neutral trends represent uncertainty in the index
- Downward trends represent bearish speculation or low confidence in the index

Depending on how we tweak the parameters, we can have indices with strong trends (high drift to vol ratio) or noisy indices with weak trends (low drift to vol ratio).

At the moment, the offered indices are:

- DSI10
- DSI20
- DSI30

Initially, there was an MSI (2 ways) as opposed to the current DSI (originally CSI). The idea was scrapped as it wasn't attractive/different enough from the CSI. Slack.

The range of DSIs were released for diversification to minimize risk (PnL variance from riding a trend), to diversify customer strategies and to increase interest: Slack

Construction -- Index

The spot price at time t is defined based on the spot price the timestep before and a state (Markov) process:

$$S_t = S_{t-1} \exp\left[\left(\mu(X_t) - \frac{\sigma^2}{2}\right) dt + \sigma\sqrt{dt}W\right]$$

where

$$\begin{aligned} S &= \text{Spot Price} \\ \mu &= \text{Drift function} \\ X &= \text{Markov state process} \\ W &= \text{Standard Normal Distribution: } \mathcal{N}(0,1) \\ \sigma &= \text{Volatility} \\ dt &= \frac{1}{365 * 86400} \end{aligned}$$

The drift function $\mu(x)$ is defined as:

$$\mu(x) = \begin{cases} \mu & \text{if } x = 0 \\ 0 & \text{if } x = 1 \\ -\mu & \text{if } x = 2 \end{cases}$$

Where μ is a constant positive number.

X_t is a Markov state process on the states $\{0, 1, 2\}$ whose states are generated each second. The transition matrix is as follows:

$$P = \begin{pmatrix} 1-\lambda & \gamma\lambda & \lambda/2 \\ \lambda/2 & 1-\lambda & \lambda/2 \\ \lambda/2 & (1-\gamma)\lambda & 1-\lambda \end{pmatrix}$$

where

$$\begin{aligned} \lambda &= \frac{1}{T} = \text{Base probability of switching states (regimes)} \\ T &= \text{Average duration spent in a regime} \\ \gamma &= \text{Drift correction term} \end{aligned}$$

Note that the elements P_{ij} of the matrix represent the probabilities: $\mathbb{P}(X_t = i \mid X_{t-1} = j)$.

For γ, production implements them as constants. The numbers are derived from the formulas in the appendix.

For DSI10, DSI20, DSI30, the parameters defined are:

Index	Drift μ	Volatility σ	T	γ	Initial Value
DSI10	100	0.1	600 (10 mins)	0.4980997907	10000

DSI20	60	0.1	1200 (20 mins)	0.4977183219	10000
DSI30	35	0.1	1800 (30 mins)	0.4980031156	10000

Construction -- Spread (R&D Section 4)

Unlike other indices developed in Deriv, the DSI does not have a constant bid/ask spread. The purpose of this non-constant spread is to reduce the risk presented by considering a strategy where we take a long position when we know we are in positive drift and short on the negative drift. The following is a summary of the construction without justification. For justification, refer to the docs mentioned in the next section.

The Perfect Strategy

Consider a typical momentum based strategy on the spot price of the DSI, assuming we know the states of the DSI exactly, then we would go long on positive drifts and short on negative drifts and close on a state change. If that is the case, the expected return for a positive regime is:

$$e^{\mu dtT} - 1 \approx \mu dtT$$

A More Reasonable Strategy

Clearly, we cannot use the states as a signal as that information is not public. Say we had a variable representing the probability that the DSI is in a particular state, i.e $x \in [-1, 1]$ such that:

- If x is close to 1, the DSI is likely to be in a positive drift (If $x = 1$, the DSI is a.s positive drift);
- If x is close to -1, the DSI is likely to be in a negative drift (If $x = -1$, the DSI is a.s negative drift);
- If x is close to 0, it is uncertain of the state of the DSI (Could be stationary, or positive/negative drift parameter, yet movement looks stationary)

Then we can define the signal as:

- Long if x is sufficiently positive;
- Short if x is sufficiently negative;
- Close if x switches sides and passes threshold

Then the expected (approximated) PnL here is:

$$PnL(x) = \mu dtT|x|$$

Quantifying x with $diff$

Now, we need to consider a process $diff$ that we can put in place of x such that it predicts the states according to the rules we set for x above. Moreover, any such prediction model must work with limited information, as we cannot assume to have all the state information. To do so, Bayesian inferencing is used to guess the current state based on only an initial state and a recursive scheme.

We expect the given model to return a vector of probabilities for each time t:

$$\begin{pmatrix} \mathbb{P}_0 \\ \mathbb{P}_1 \\ \mathbb{P}_2 \end{pmatrix}$$

Where $\mathbb{P}_{0/1/2}$ is the probability the regime at time t is positive/stationary/negative respectively. Now we try to **infer** the probability that the state at time 1, α_1 is 0 given the current return r_1: $\mathbb{P}(\alpha_1 = 0 \mid R_1 = r_1)$:. Then if R_t denotes the return at time t, they follow the distribution determined by f_α depending on the state α. We can expand $\mathbb{P}(\alpha_1 = 0 \mid R_1 = r_1)$:

$$\mathbb{P}(\alpha_1 = 0 \mid R_1 = r_1) = \frac{f_0(r_1) \cdot \mathbb{P}(\alpha_1 = 0)}{\sum_{i=0}^{2} f_i(r_1) \cdot \mathbb{P}(\alpha_1 = i)}$$

We can do the same for $\mathbb{P}(\alpha_1 = 1 \mid R_1 = r_1)$ and $\mathbb{P}(\alpha_1 = 2 \mid R_1 = r_1)$ to **forecast** the state at $t = 2$ given r_1, $\mathbb{P}(\alpha_2 = 0 \mid R_1 = r_1)$

$$\mathbb{P}(\alpha_2 = 0 \mid R_1 = r_1) = \sum_{\alpha=0}^{2} \mathbb{P}(\alpha_2 = 0 \mid \alpha_1 = i) \cdot \mathbb{P}(\alpha_1 = i \mid R_1 = r_1)$$

We can then get $\mathbb{P}(\alpha_2 = 0 \mid R_1 = r_1, R_2 = r_2)$ and so on with recursion. The algorithm is to do what we did before, alternating between inferencing and forecasting to generate each prediction:

Base case: Set the initial state to the stationary regime

$$\xi_{1|0} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

Then we can define the forecasted probabilities (our objective) for time $t + 1$ based on information from time t and below as:

$$\xi_{t+1|t} = \begin{bmatrix} \mathbb{P}(\alpha_{t=1} = 0 \mid r_t, r_{t-1}, \cdots, r_1) \\ \mathbb{P}(\alpha_{t=1} = 1 \mid r_t, r_{t-1}, \cdots, r_1) \\ \mathbb{P}(\alpha_{t=1} = 2 \mid r_t, r_{t-1}, \cdots, r_1) \end{bmatrix}$$

To obtain each forecasted probability, we use recursion:

$$f_t = \begin{bmatrix} f_0(r_t) \\ f_1(r_t) \\ f_2(r_t) \end{bmatrix}$$

We can define the inferred probabilities from the previous forecasted probabilities as:

$$\xi_{t|t} = \frac{1}{\xi_{t|t-1} f_t} \xi_{t|t-1} \odot f_t$$

Where $\odot$ denotes an element-wise product. We can then compute the next forecasted probability as:

$$\xi_{t+1|t} = P\xi_{t|t}$$

If $\xi_{t+1|t} = \begin{bmatrix} \mathbb{P}^+ & \mathbb{P}^0 & \mathbb{P}^- \end{bmatrix}^T$, we define the signal:

$$diff = \mathbb{P}^+ - \mathbb{P}^-$$

We use $diff$ in place of x, i.e the Pnl is

$$PnL(diff) = \mu dtT|diff|$$

The spread

Modifiying the PnL function above to account for other risk factors, we end up with the spread model:

$$ask(x) = S\left[1 + \max(Perf \cdot \kappa_{ask}x + \epsilon_{ask}, 0)\right] + \frac{pip}{2}\max\left((com_{max} - com_{min})x + com_{min}, 0\right)$$

$$bid(x) = S\left[1 - \max(-Perf \cdot \kappa_{bid}x + \epsilon_{bid}, 0)\right] - \frac{pip}{2}\max\left((com_{max} - com_{min})x + com_{min}, 0\right)$$

Where:

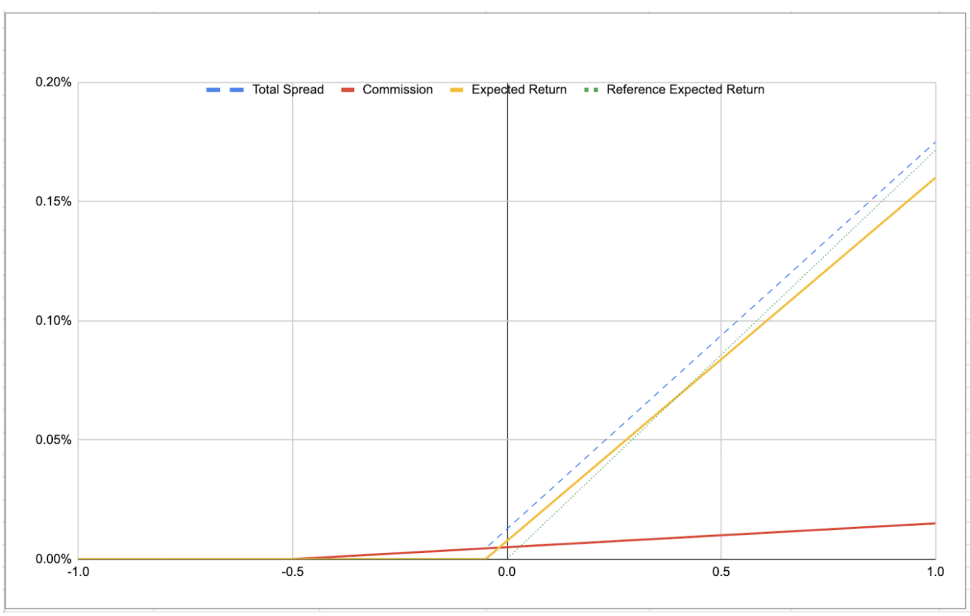
- x: The current $diff$ value;
- S: The current spot price;

Hard-coded constants

- $\kappa_{ask/bid}$: The ask/bid reference performance indicator (About the expected drift of the positive/negative regime);
- $\epsilon_{ask/bid}$: Adjusts the condidence threshold of $diff$, i.e if $\epsilon > 0$, the ask spread will be applied a bit before we know that $diff > 0$ to account for the positive drift early.
- $pip = \$0.01$: The pip size

BO set constants (Dealing)

- $Perf$: The performance of the clients
- com_{min}: The minimum charge as commission, representative of the charge for a stationary regime, i.e when we think we are generating a driftless GBM;
- com_{max}: The maximum charge as commission, representative of the charge to account for PnL Variance and trade concentration risk.



More on the two components here: (DSI BO Tool - explained).

We break down the two components of the spread. Consider S_{ask} and observe the first component:

$$max(Perf\kappa_{ask}x + \epsilon_{ask}, 0)$$

- This refers to the expected PnL function from earlier.
  - We shift it horizontally with $\epsilon_{ask,Ref}$ to account for where (in terms of $diff$) we expect the clients to start winning;
  - We apply $\kappa \sim \mu dtT$ to account for the drift from the PnL function;
  - We apply $\max(\cdot, 0)$ to prevent negative spread.
  - The Dealing team can adjust $Perf$ to account for when customers are under or over-performing.

More details on the idea behind κ and ϵ in section 6.

Now, for the second part:

$$\max\left((com_{max} - com_{min})x + com_{min}, 0\right)$$

- This refers to the commission and and additional charges for other risk factors, similar to other products. Here, the charge has some key points:
  - At $diff = 1$, the extra charge is com_{max}, which accounts for any possible PnL variance / trade concentration risk.
  - At $diff = 0$, the extra charge is com_{min}, which is used for when we think the current state is stationary, i.e a driftles GBM
  - At $diff \leq \frac{-com_{min}}{com_{max} - com_{min}} < 0$, the commission charged is 0

S_{bid} is similar, but flipped on the y-axis and $\kappa_{bid}, \epsilon_{bid}$ differ from $\kappa_{ask}, \epsilon_{ask}$.

The rest are detailed in DSI BO Tool - explained and can be calculated in DSI BO Tool

According to DSI10/20/30 - BO specs, some spread parameters can be set in BackOffice:

- $Perf$, com_{max}, com_{min} will be set by the **Dealing** team regularly in BackOffice
- κ, ϵ will be hard-coded

Hence, there are some constraints:

1. $com_{max} > com_{min}$. By how much depends on the Dealing team;
2. $0 < Perf \leq 2$
3. $com_{max}, com_{min} > 0$

Spread Specifications

DSI10/20/30 - BO Specs has an indicative example of the specs. The κ & ϵ in the picture below are the current values implemented.

Currently, the commission specs for each DSI (Demo 14/09/2023 onwards) are as follows:

```
Demo_Phase_2: # Demo at 14/09/2023 onwards
  DSI10:
    perf: 0.45
    com_min: 36
    com_max: 72

  DSI20:
    perf: 0.37
    com_min: 31
    com_max: 62
```

```
DSI30:
  perf: 0.3
  com_min: 24
  com_max: 48
```

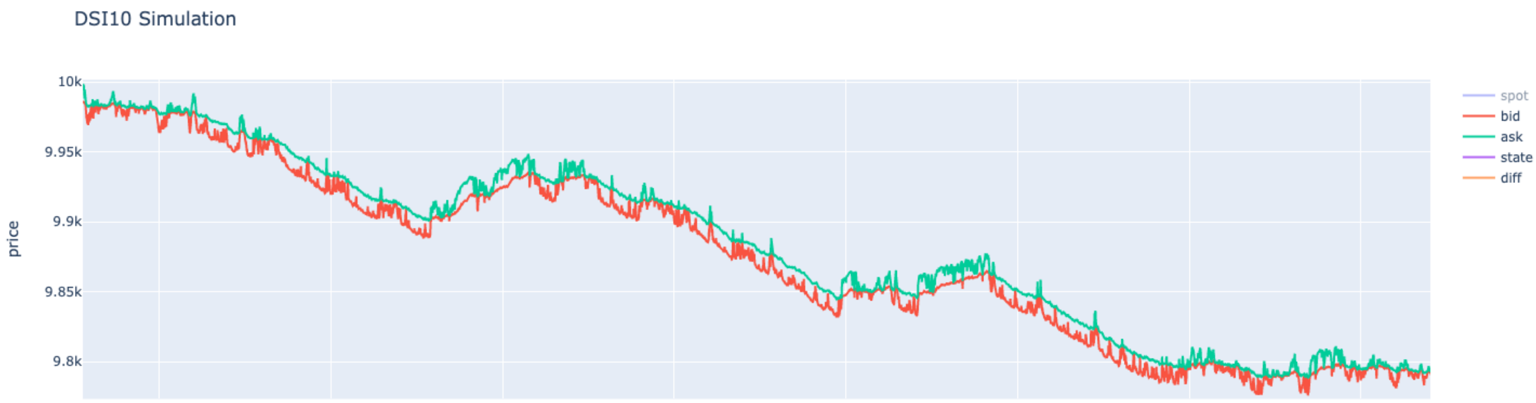
	Hard Coded (BE)			
	Ask Spread		Bid Spread	
	Kappa Ref	Epsilon Ref	Kappa Ref	Epsilon Ref
DSI10	0.1268393717%	0.0001608149%	0.12635732896%	0.00032122775%
DSI20	0.152207353844%	0.000231621965%	0.151512777478%	0.000462954401%
DSI30	0.1331813546891%	0.0001773476251%	0.1326494599%	0.0003545474%

Spread -- Justification

Essentially, the spread above is similar to forward pricing on DSI as detailed in the documents of section 6. Pricing CFDs using the forward prices of the index is a valid method that isn't arbitrageable for the "perfect" strategy described above. Furthermore, this method of bid/ask generation can be generated quickly.

MT5

In MT5, customers will not see the spot, they will see the bid/ask prices (ask feed cannot be graphed by ticks):



Model Validation

Summary

For the validation of DSI indices, we cover the below areas and conclude the outcomes. More details can be found in the respective sections.

Section	Area	Validation	Outcome	Passed?
1	States/Xi storage in Feeddb	The actual states are currently stored in feeddb which poses a potential operational risk. The following dataframes were tested with a sample "perfect" strategy using the state feed on various delays: • Monte Carlo 1 month data (Latest spread parameters) • Monte Carlo 1 week data (Latest spread parameters) • DevServer Data • Demo Data	The sample strategy resulted in high median/mean pnl per trade and high hit/win rate until a reasonably large delay: • DSI10: Profitable up to 2 minutes+ • DSI20: Profitable up to 5 minutes+ • DSI30: Profitable up to 10 minutes+ **Update** (4 October 2023): The view on the state column in `drift_switch_states` will be hidden on Metabase. This significantly restricts access to the states	**Passed**
2	γ derivation and computation	• Validation of derivation of γ • Check that the production constants are correct • Use Monte Carlo to check what happens when γ is not the right value	• Derivation is correct • Perl uses the constants derived by the formulas • Using any other values for γ results in mismatched drift	**Passed**
3	Stationary distribution verification, computation & convergence	• The state distribution for the feeddb/simulations should match the stationary distribution in the long term. • The convergence should also be checked	• Simulation: Generating a 3 month's state process and checking the final distribution shows that the error with the stationary distribution is less than 0.01 • Demo/DevServer: Similar results with max error of 0.03 • Convergence: TV Error is less than 0.05 within a week	**Passed**
4	Long term moments & convergence	• Feed moment data computation and cross-checking with simulation moments • Convergence speed testing	• Feed volatilities match, kurtosis and skewness are sufficiently small, but different • Moments converge approximately within a week	**Passed**
5	Backward engineering the parameters.	• Parameters are backward engineered from the feed data moments; • MLE is attempted	• The result looks fine with the set initial condition and boundary condition; • The MLE algorithm can retrieve μ, σ and the initial state accurately for each of the DSI • T and transition probabilities can be retrieved with varying degrees of success	**Passed**

			(In descending order of accuracy: DSI10, DSI20, DSI30)	
6	Spread formula	• Validation of the proofs & logic behind the spread formula • Spread parameter replication • Demo Feed Bid/Ask replication	• The proofs presented in the document are verified and valid. • The numbers match up with other methods of derivation • Replicated perfectly (minus rounding errors)	**Passed**
7	Correlation	Correlations between one-tick/hourly log returns of DSI10,20,30 are checked. The following indices are used: • Dev Spot • Demo Spot • Demo Phase 1/2 Bid • Demo Phase 1/2 Ask	The log returns are minimally correlated	**Passed**

Data

We import the following data:

- Demo:
  - Full: 02/09/2023 onwards;
  - Phase 1: 02/09/2023 - 13/09/2023;
  - Phase 2: 14/09/2023 onwards
- DevServer: 18/08/2023 - 29/08/2023.

Each dataset starts at midnight UTC. Phase 1 & DevServer end at midnight UTC. The demo data follows the commission specs:

```
Demo_Phase_1: # After Demo launch 02/09/2023 until 13/09/2023
  DSI10:
    perf: 0.6
    com_min: 36
    com_max: 72

  DSI20:
    perf: 0.6
    com_min: 31
    com_max: 62

  DSI30:
    perf: 0.6
    com_min: 24
    com_max: 48

Demo_Phase_2: # Demo at 14/09/2023 onwards
  DSI10:
    perf: 0.45
    com_min: 36
    com_max: 72

  DSI20:
    perf: 0.37
    com_min: 31
    com_max: 62
```

```
DSI30:
  perf: 0.3
  com_min: 24
  com_max: 48
```

The Dev data uses an older version of the spread.

Section 1

The states X_t and the forecast probabilities $\xi_{t|t-1}$ are currently stored in feeddb. Inspecting the feed, new points get updated approximately every 1.5-2 minutes, up until 30s before the time of update. Considering how accessible feeddb is (SQL via Metabase or QABox), storing the values poses an operational risk.

To quantify it, we run a Monte Carlo process on the perfect long/short strategy detailed above. We then repeat it for various delay values (simulating X_t feed retrieval delay) and observe the mean/median pnl per trade and the win/hit rates.

We test on:

- Monte Carlo 1 month data (Latest spread parameters)
- Monte Carlo 1 week data (Latest spread parameters)

We also test on feed data, namely:

- Demo - Phase 1
- Demo - Phase 2
- DevServer ##### Results

We see that both 1 week / 1 month datasets have negligible differences.

1. Win rates drop off significantly faster than the rest of the statistics, they are still profitable per trade, until the rest of the statistics also drop off
2. Each DSI drops off at about
   A. DSI10: 2 minutes
   B. DSI20: 5 minutes
   C. DSI30: 10 minutes
3. Comparing Demo to Dev data, we see that the Dev data is less susceptible to this risk than the Demo data as Demo perf (60%) is significantly lesser than the Dev perf (100%).

NOTE: Swap rates, lot sizes and transaction costs aren't considered (stop loss could be considered, but it is negligible in this scheme)

ALSO: Comparing Demo to Dev data, we see that the Dev data is less susceptible to this risk than the Demo data as Demo perf (60%) is significantly lesser than the Dev perf (100%).

Typically, this would not be an issue, but the states & xi values are stored in feeddb, resulting in possible insider trading. Considering that it is accessible via QAbox and Metabase (at a 2 minute delay), it is not very difficult to take advantage of.

Currently, it exists in Feeddb as an artifact of how the feed is implemented (additionally, it is useful for validation purposes), but we strongly urge the removal of the states/xi at launch.

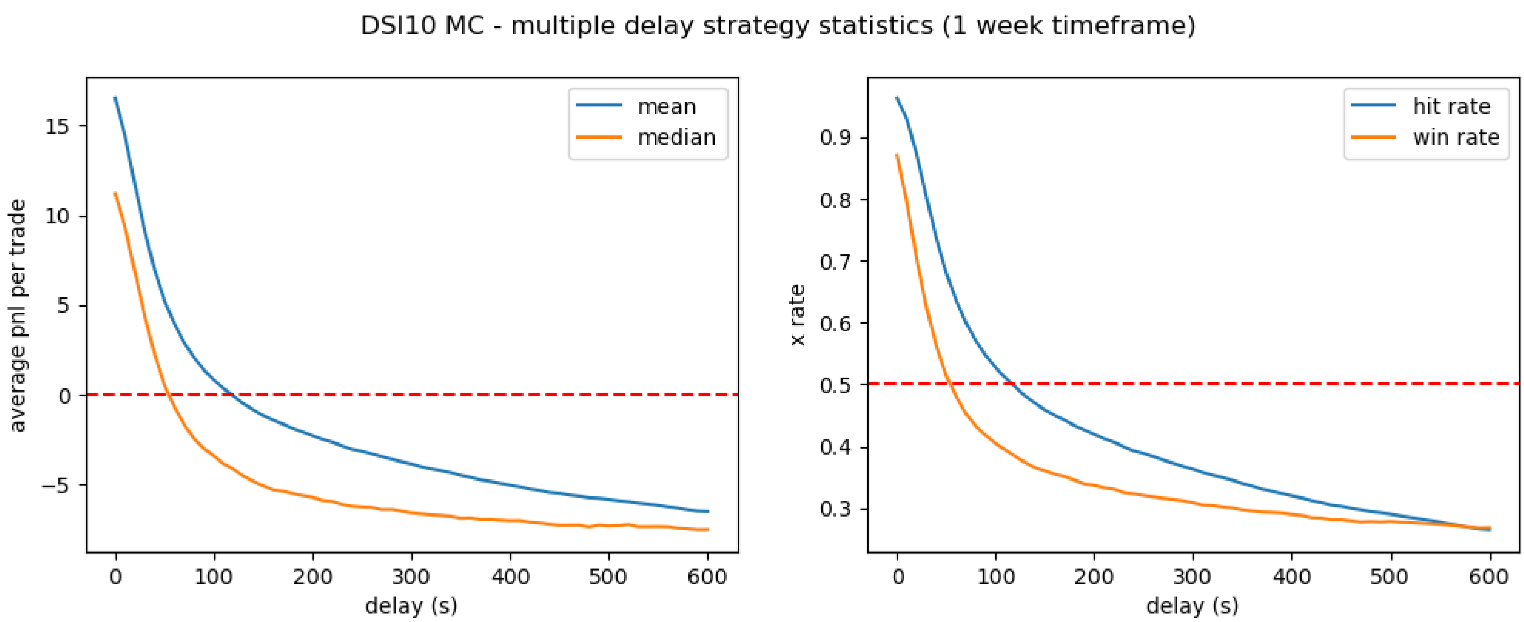
Update (4 October 2023)

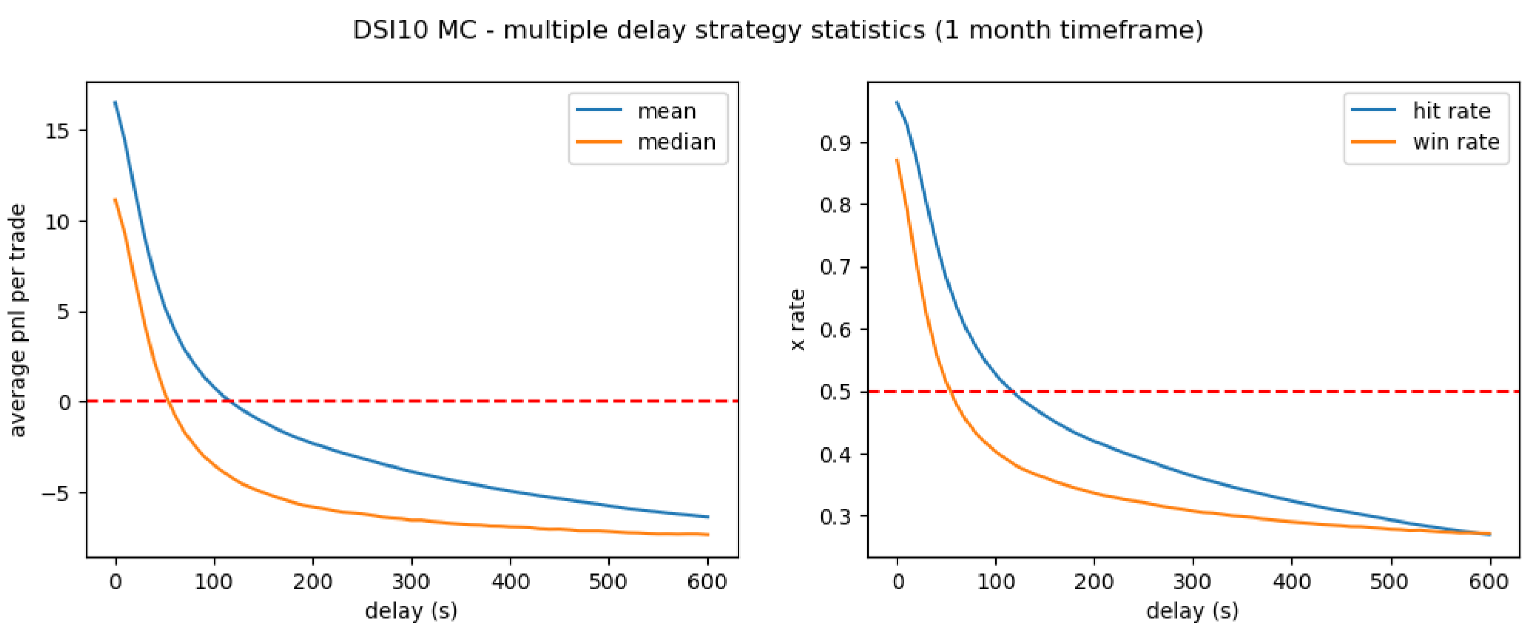
It has been agreed to restrict Metabase access of the `drift_switch_state` table. In particular, the state column will not be accessible via question or SQL in Metabase. This would reduce visibility of the state column significantly, preventing . Note that the xi column will still be visible as

- It is required for Model Validation, $Perf$ calibration and for revenue metrics (DSI Expected Revenue)
- It doesn't pose any risk as the ask/bid spread is just a transformation of the xi values (In particular, `diff = xi[0] - xi[2]`)

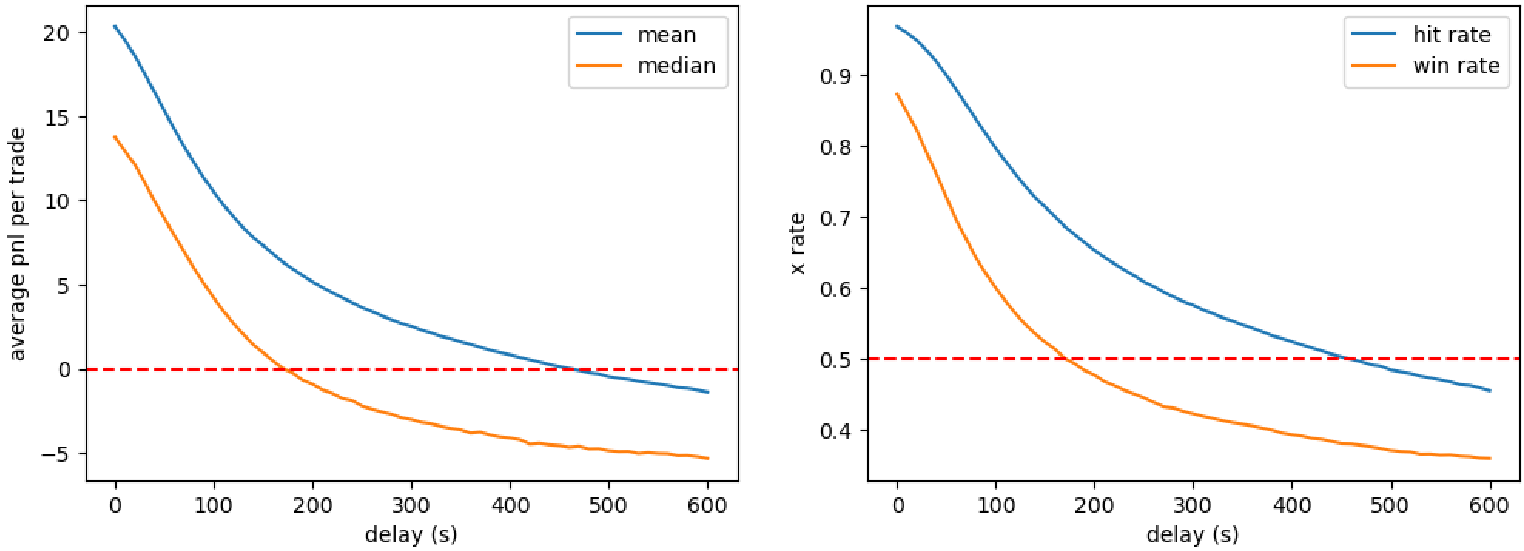
This solution was chosen as the alternative: removing states from being stored outright had the side-effects of requiring a choice of states at a system restart. Regardles of the choice of states, the unpredictability of frequency of restarting was the main reason this alternative was rejected.

As the chosen solution does not require any changes in the backend code, all testing in this document is up-to-date.

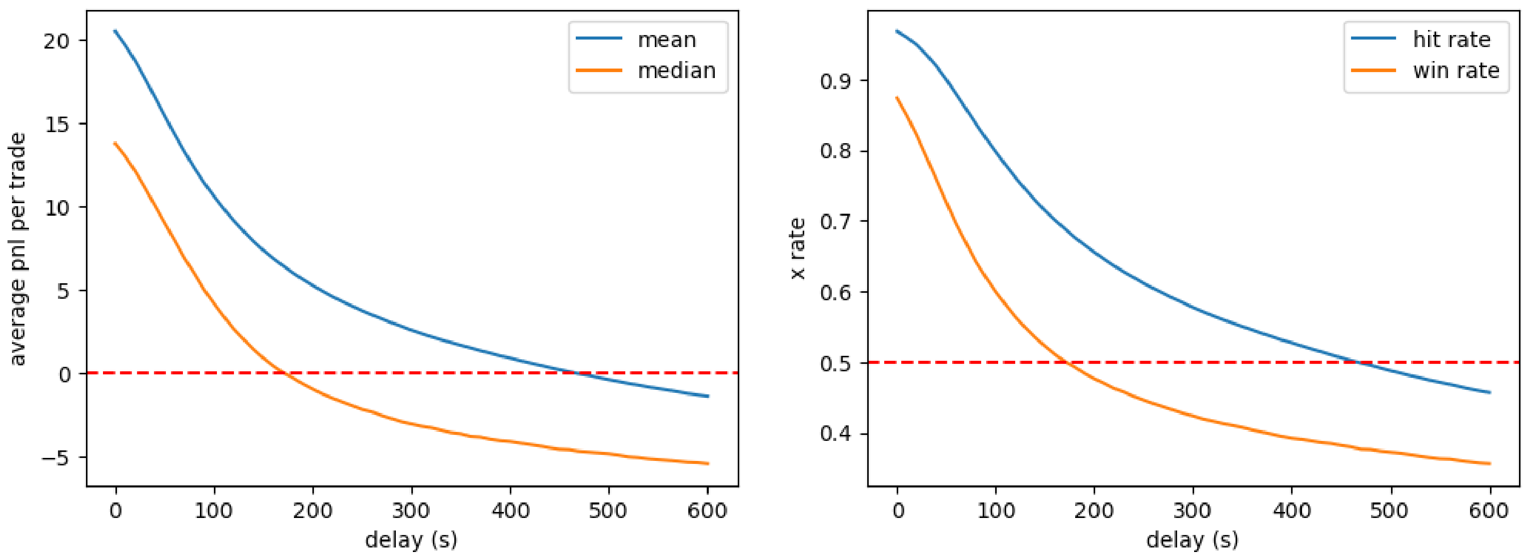




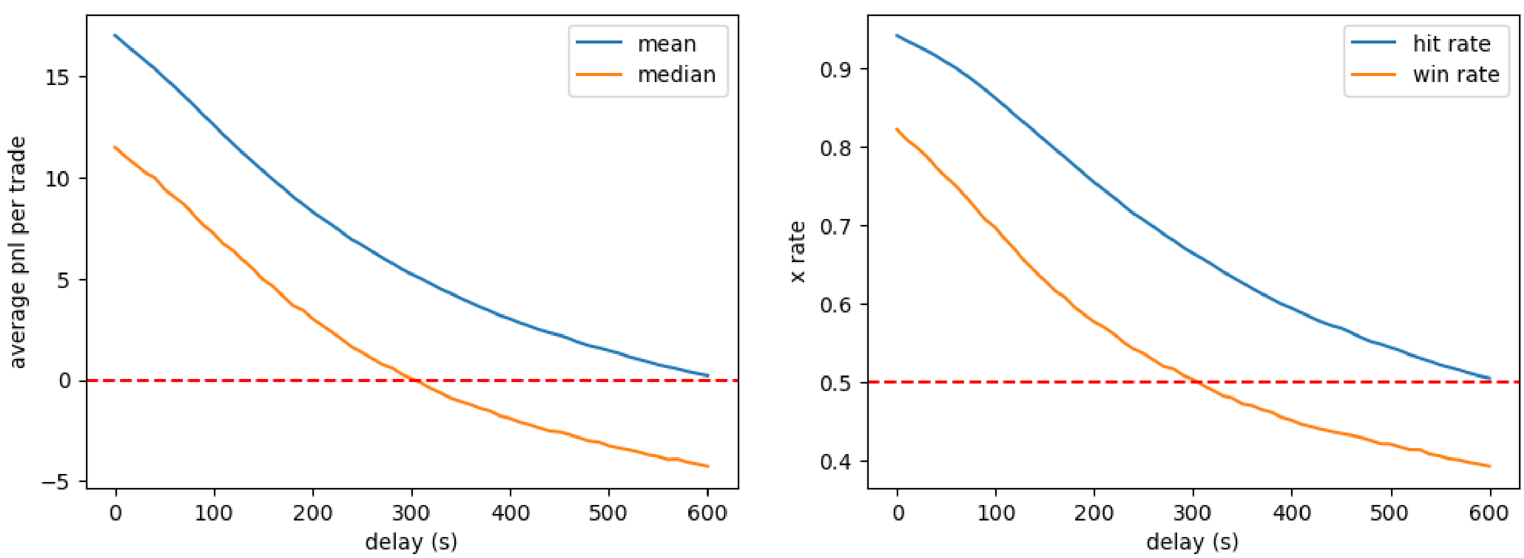
DSI20 MC - multiple delay strategy statistics (1 week timeframe)



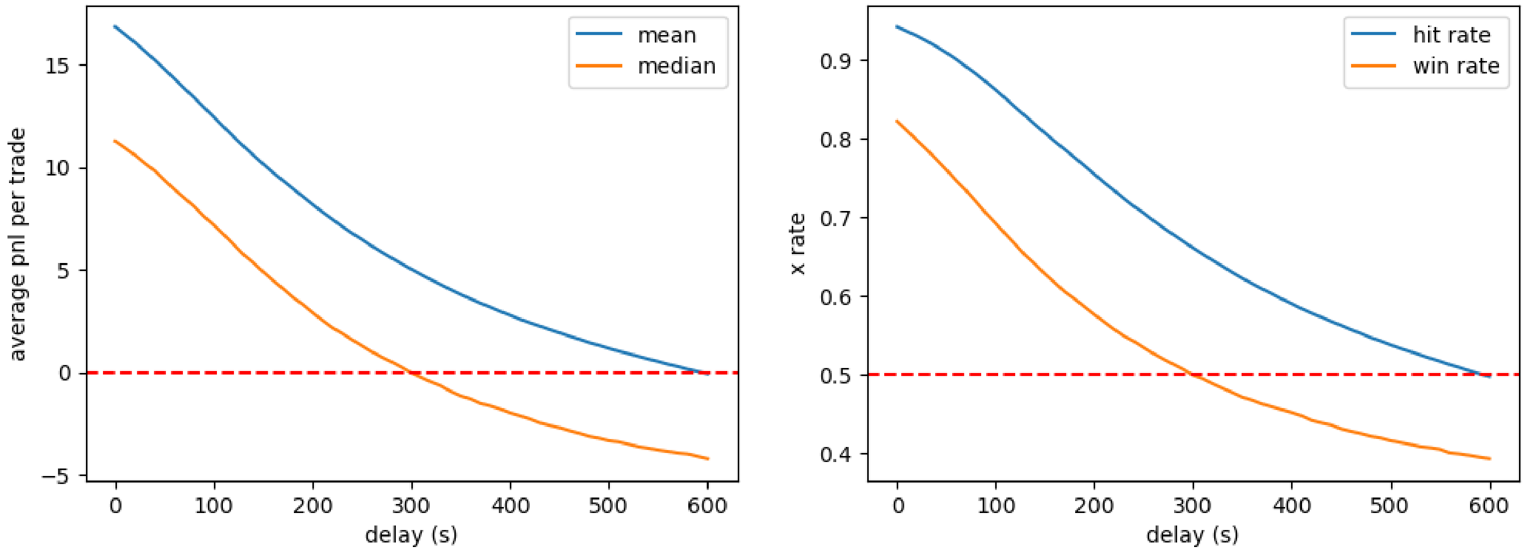
DSI20 MC - multiple delay strategy statistics (1 month timeframe)



DSI30 MC - multiple delay strategy statistics (1 week timeframe)

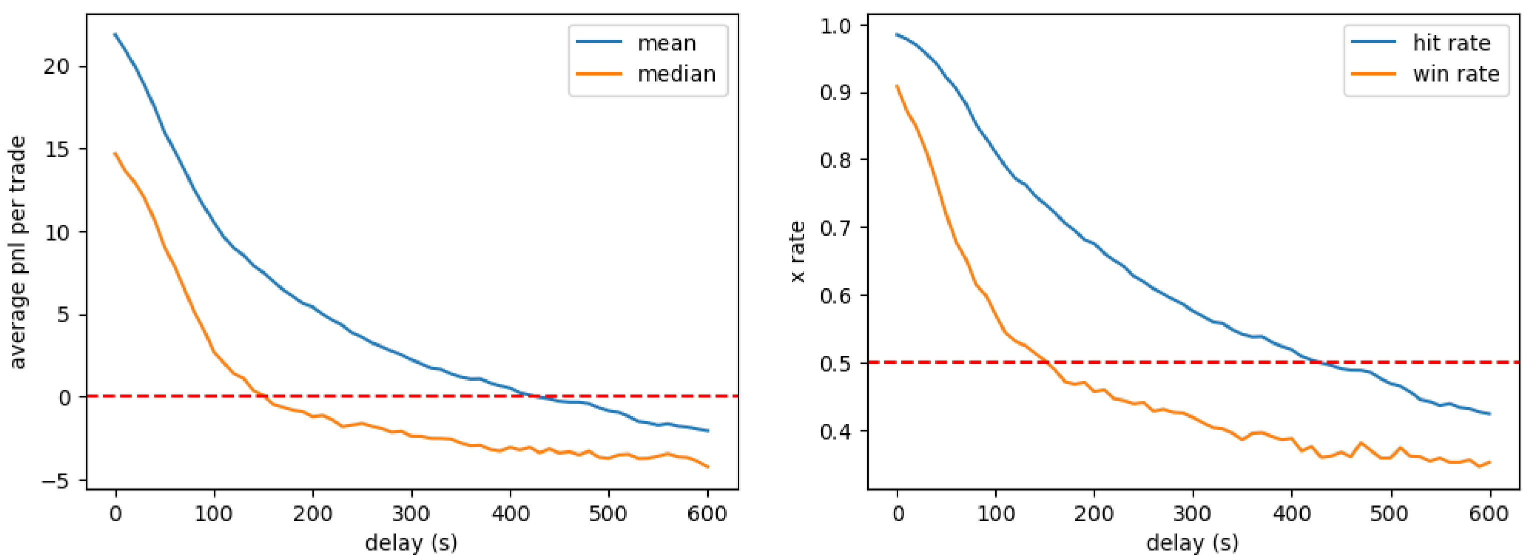


DSI30 MC - multiple delay strategy statistics (1 month timeframe)

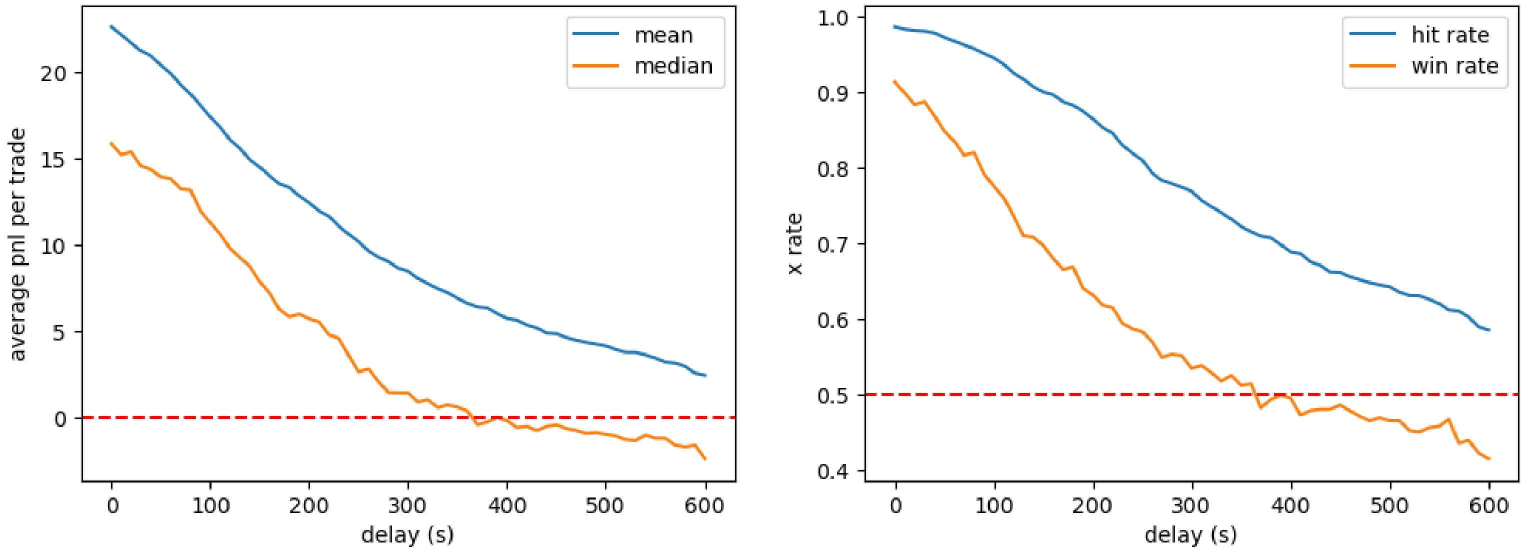



```
Number of trades executed for DSI10 Demo Phase 1: 1073

Number of trades executed for DSI20 Demo Phase 1: 535

Number of trades executed for DSI30 Demo Phase 1: 340
```

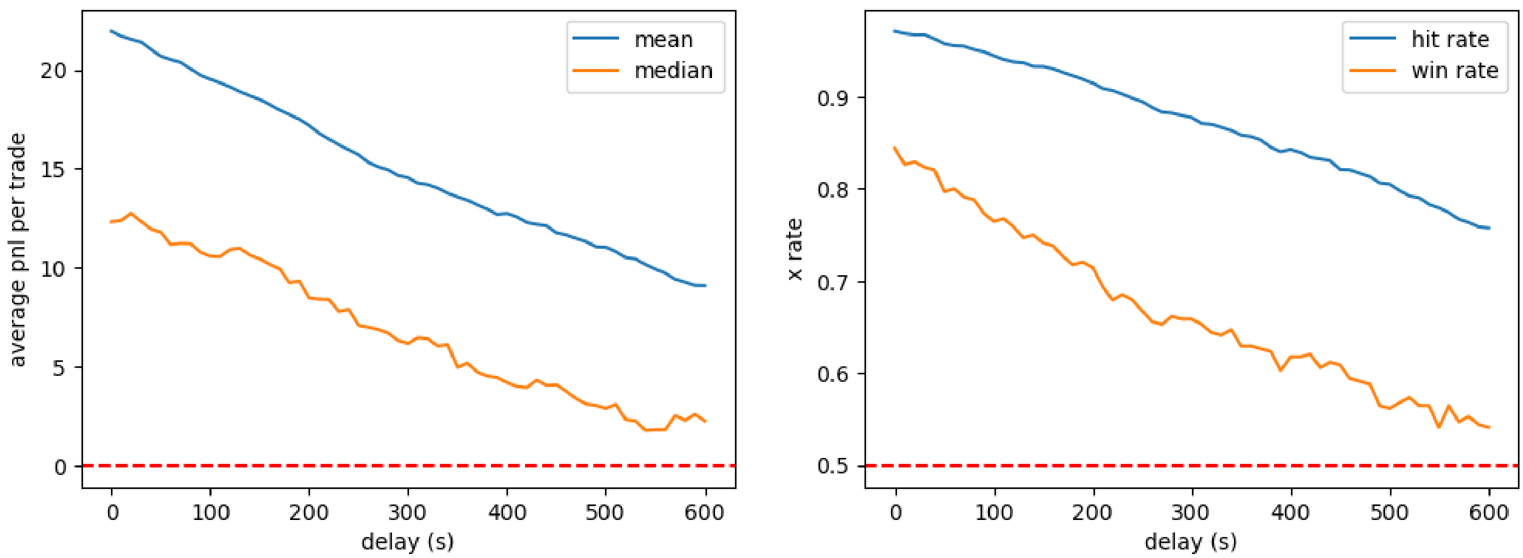
DSI10 MC - multiple delay strategy statistics (Demo Phase 1 (02/09/2023 - 13/09/2023) timeframe)



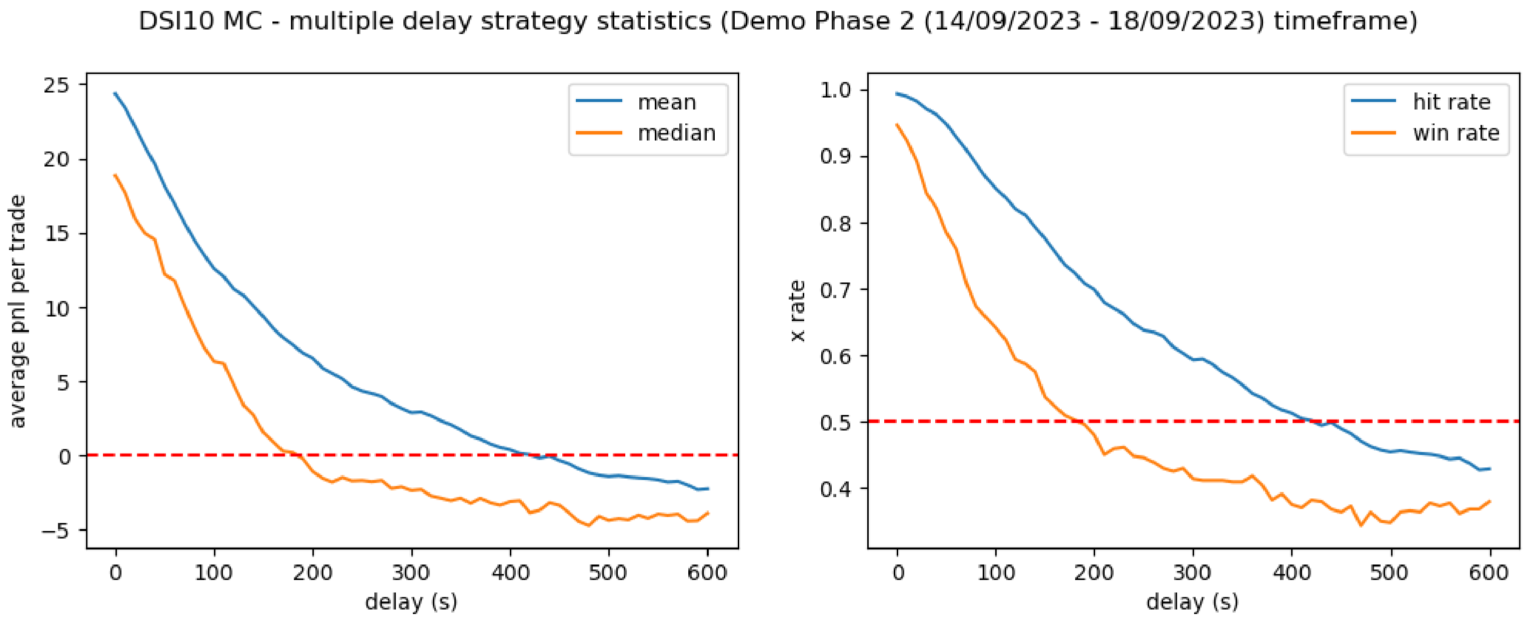
DSI20 MC - multiple delay strategy statistics (Demo Phase 1 (02/09/2023 - 13/09/2023) timeframe)

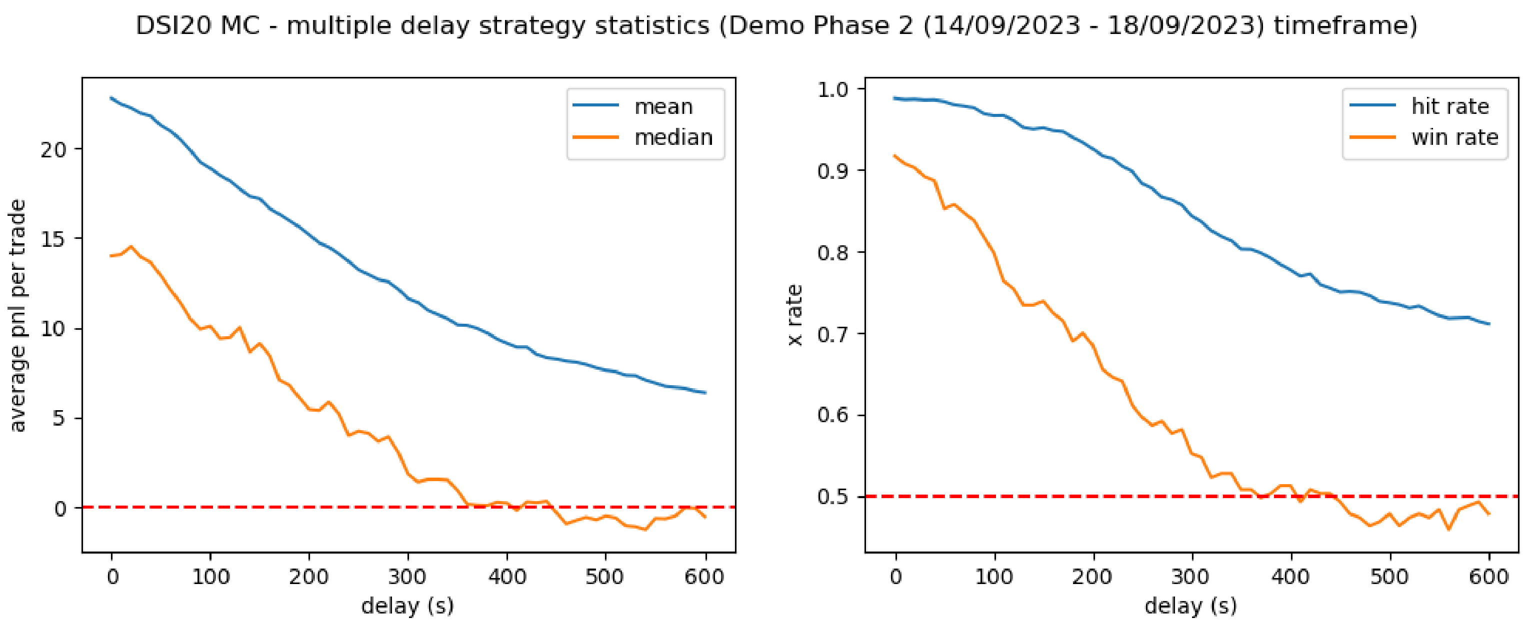


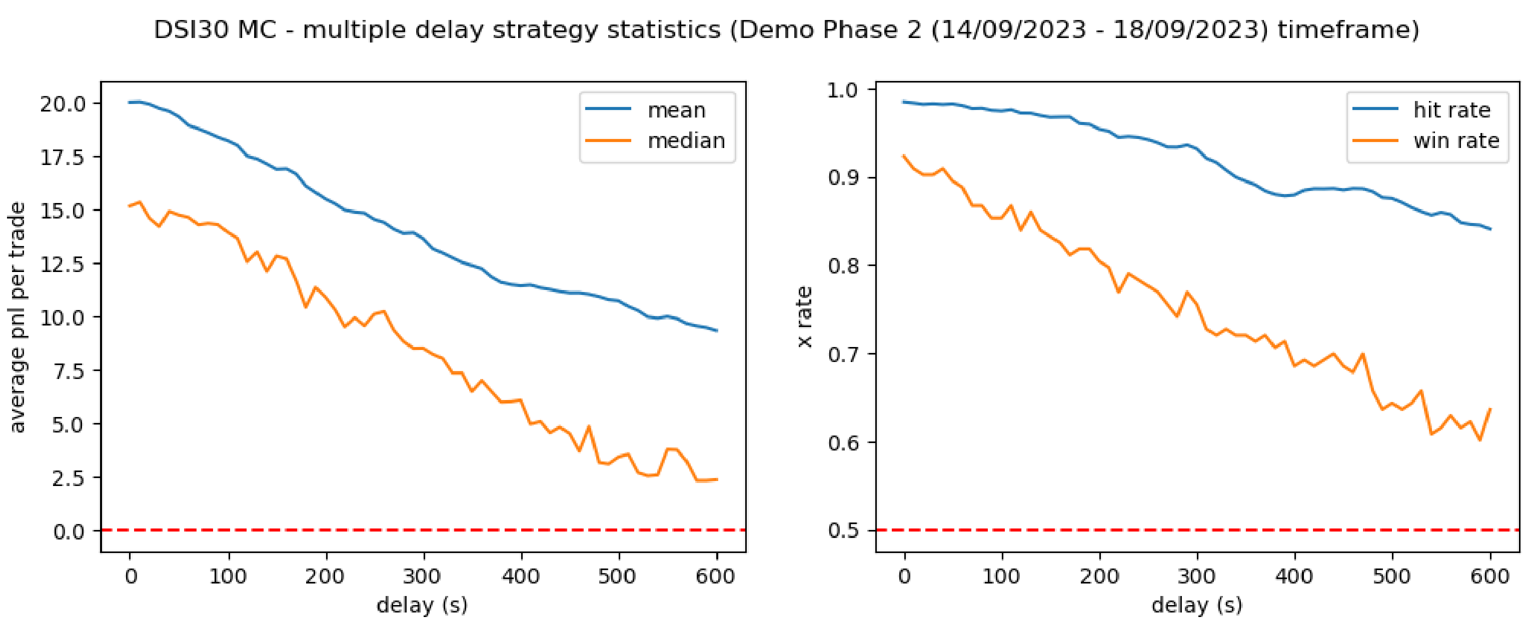
DSI30 MC - multiple delay strategy statistics (Demo Phase 1 (02/09/2023 - 13/09/2023) timeframe)




```
Number of trades executed for DSI10 Demo Phase 2: 440

Number of trades executed for DSI20 Demo Phase 2: 205

Number of trades executed for DSI30 Demo Phase 2: 143
```



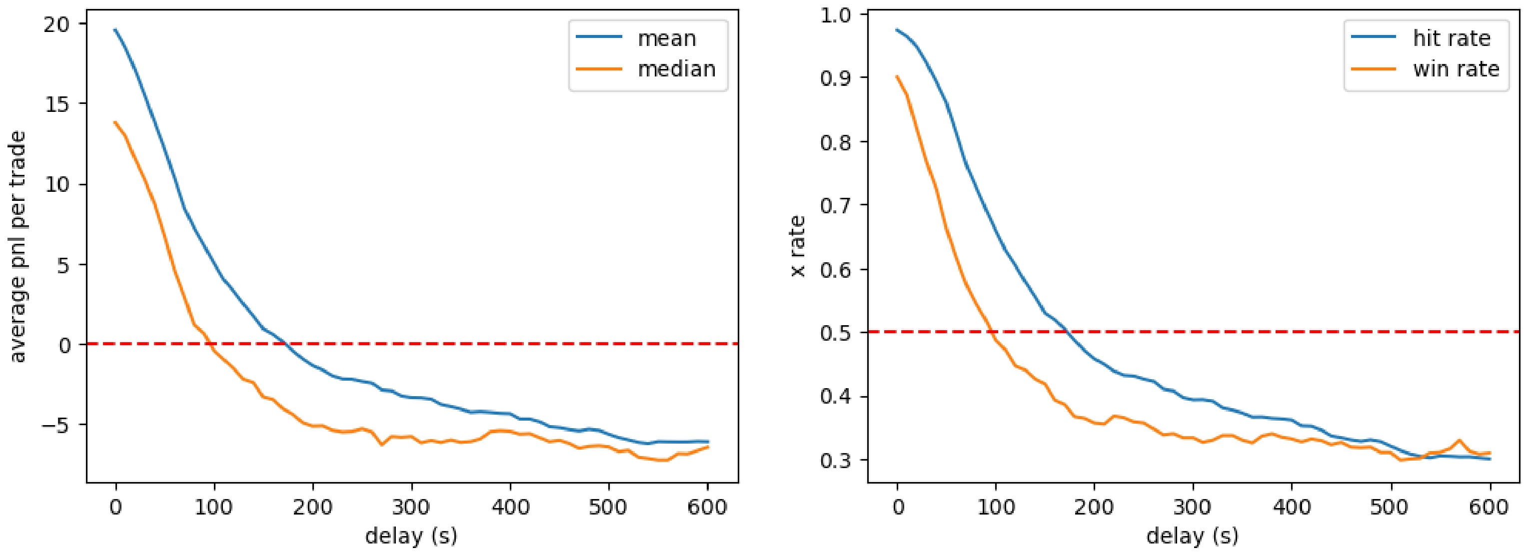




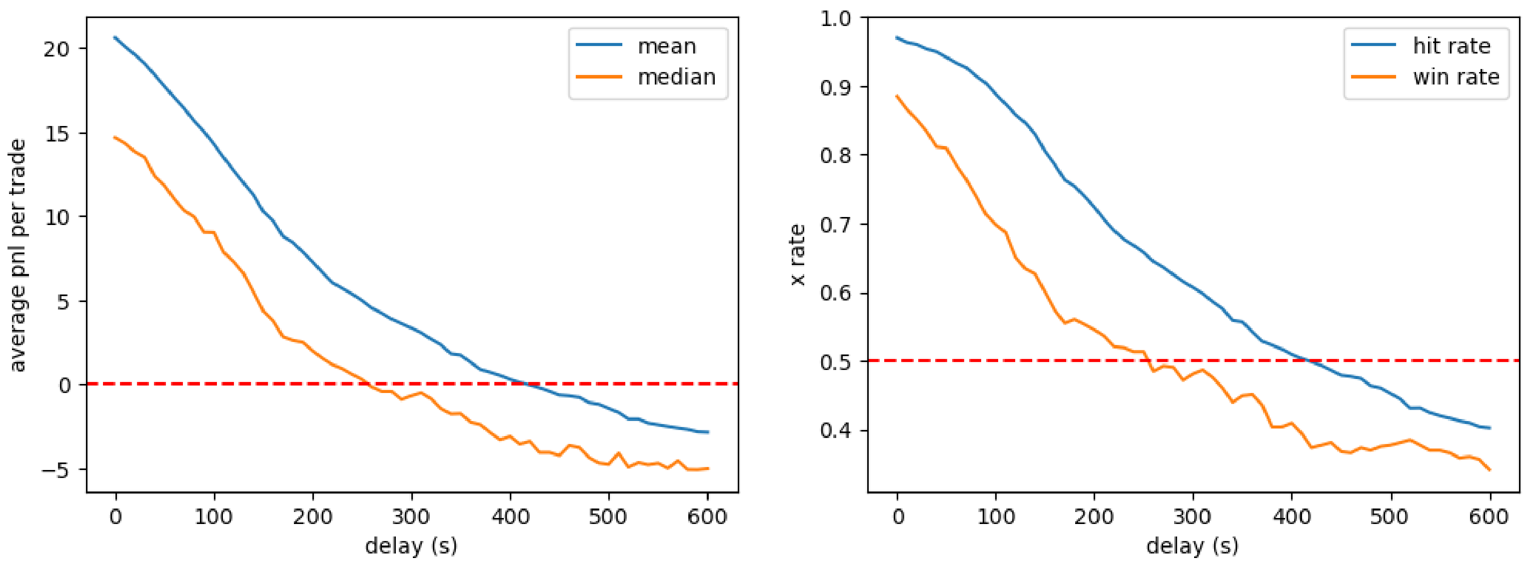

```
Number of trades executed for DSI10 DevServer: 1082

Number of trades executed for DSI20 DevServer: 530
```

```
Number of trades executed for DSI30 DevServer: 356
```

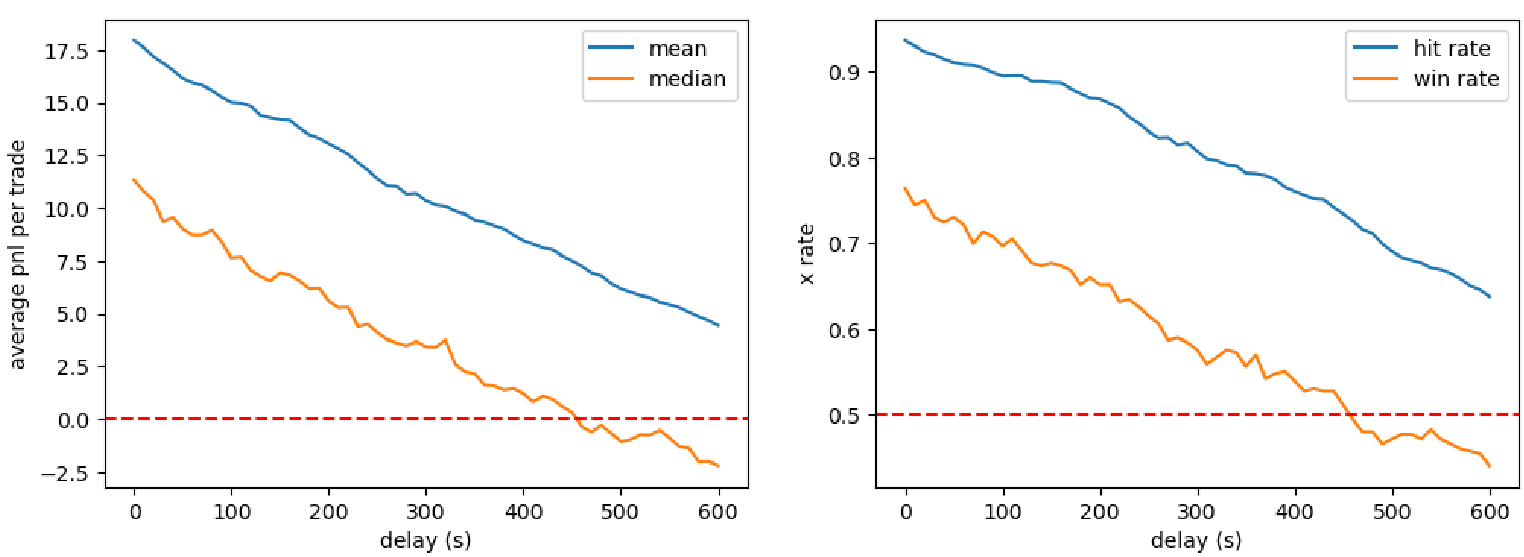
DSI10 MC - multiple delay strategy statistics (DevServer (18/08/2023 - 28/08/2023) timeframe)



DSI20 MC - multiple delay strategy statistics (DevServer (18/08/2023 - 28/08/2023) timeframe)



DSI30 MC - multiple delay strategy statistics (DevServer (18/08/2023 - 28/08/2023) timeframe)



Section 2

Having the correct value for γ is required to ensure that the DSI does not drift to infinity or 0. To verify the value:

1. The derivation/proof in the R&D document (p26-29) is validated
2. The production γ numbers are checked with the formula provided
3. Monte Carlo simulations on the spot are performed for over/underestimated γ, then the mean for each case are compared

Derivation Validation

The derivation is indeed correct. However, the recurrence relations for γ (R&D 9.3.8), critical to the proof were not explicitly proved in the document. The proof has been added to the appendix.

Result

With the help of Nolan on another similar proof, the recurrence relations are valid. Proof is contained in the appendix for completion's sake.

Directly checking γ values

We compute γ for each DSI and compare the calculated value to the production constant

Result

Production constants are within tolerance of the calculated gammas

	Production	Calculated	Relative Error (%)
DSI10	0.49809979	0.49809979	8.97240454e-09
DSI20	0.49771832	0.49771832	1.70345038e-09
DSI30	0.49800312	0.49800312	-7.19538175e-09

Monte Carlo

Let γ be value found in the specs. We perturb by various percentages to check whether setting γ correctly is required. For each DSI and for each perturbation, we simulate 1000x 1 week data samples and check the mean log change from start to finish, i.e $\log(\frac{S_T}{S_0})$ where T is the last tick generated.

The drifts are then compared to the expected drift.

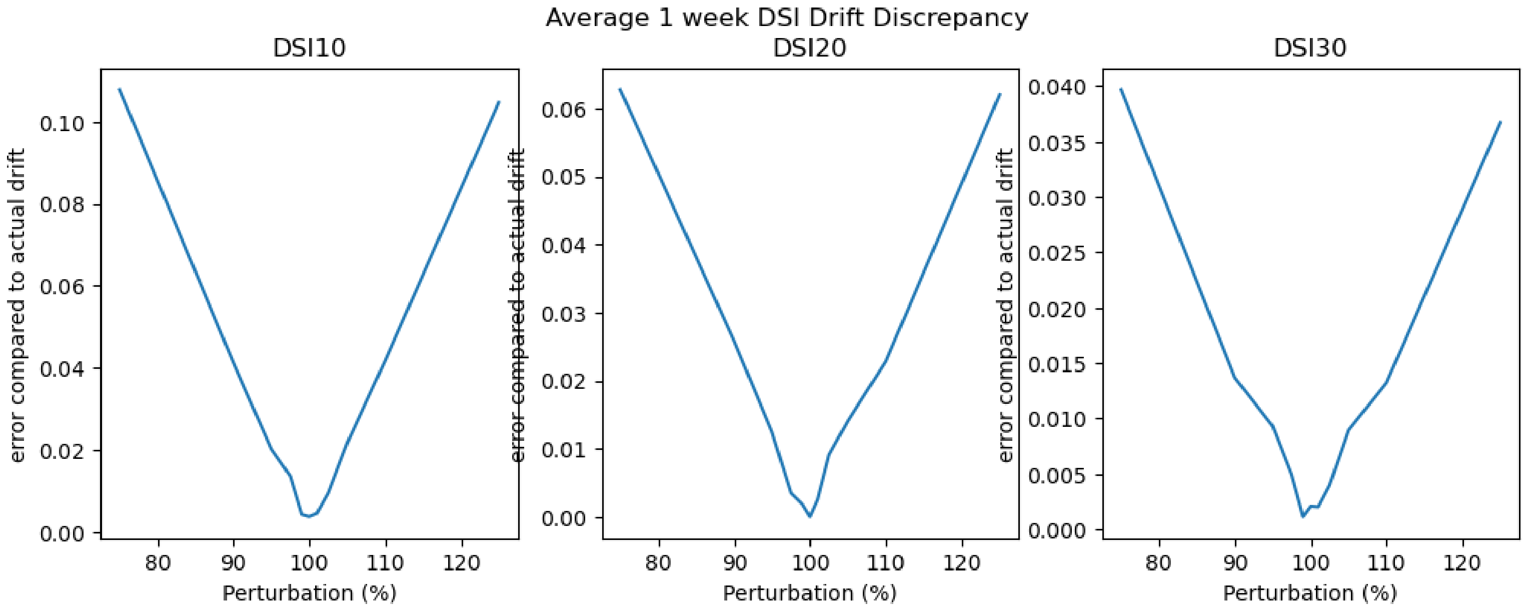
Result

Gamma minimizes between the sample drifts and the expected drift, with some error as 1000 samples may be insufficient.

	Mean Perturbations as % of production gamma				
	75.0%	**90.0%**	**95.0%**	**97.5%**	**99.0%**
DSI10	-0.10958848	-0.04297083	-0.02192016	-0.01529986	-0.00601624
DSI20	-0.06403644	-0.02708110	-0.01368692	-0.00471919	-0.00309488
DSI30	-0.04038091	-0.01438821	-0.01000617	-0.00561569	-0.00185244

Mean Perturbations as % of production gamma

	100.0%	101.0%	102.5%	105.0%	110.0%	125.0%
DSI10	0.00204766	0.00281661	0.00776658	0.01991803	0.04003903	0.10300554
DSI20	-0.00128039	0.00125661	0.00779704	0.01280392	0.02156387	0.06076686
DSI30	-0.00277375	0.00133196	0.00328406	0.00826042	0.01253908	0.03602312



Section 3

Here, calculations on the transition matrix result in the stationary distribution (Refer to appendix):

$$\pi = \begin{bmatrix} \frac{2}{9}(1+\gamma) \\ \frac{1}{3} \\ \frac{2}{9}(2-\gamma) \end{bmatrix}$$

We expect that:

- In the long term, $t \leq T$ for T sufficiently large, the distribution of states $\{S_t\}_{t \leq T}$ should converge to the above distribution.

This should hold for the actual states.

Sanity check

We check that the above distribution is indeed stationary using production gamma.

Result

Applying the transition matrix to the above distribution resulted in the same one, hence it is indeed stationary

```
DSI10: Derived stationary distribution is correct
        stationary: [0.33291106 0.33333333 0.3337556 ],
state * stationary: [0.33291106 0.33333333 0.3337556 ]

DSI20: Derived stationary distribution is correct
        stationary: [0.33282629 0.33333333 0.33384037],
state * stationary: [0.33282629 0.33333333 0.33384037]
```

```
DSI30: Derived stationary distribution is correct
        stationary: [0.33288958 0.33333333 0.33377709],
state * stationary: [0.33288958 0.33333333 0.33377709]
```

Stationary Distribution Monte Carlo

Given state data, we take the empirical distribution of the generated states.

We consider the data:

- Simulated (3 months)
- Demo (2 weeks)
- DevServer (2.5 weeks)

We then consider the total variation (TV) errors of the empirical distribution ν and the stationary distribution π, that is:

$$\|\nu - \pi\|_{TV} = \frac{1}{2} \sum_{x \in \{0,1,2\}} |\nu(x) - \pi(x)|$$

We choose the TV error as it takes accounts of all possible events and chooses the worst case.

Results

- Sim: $\|\nu - \pi\|_{TV} < 0.01$
- Demo: $\|\nu - \pi\|_{TV} < 0.03$
- Dev: $\|\nu - \pi\|_{TV} < 0.03$

Note that this is an aggregate measure, and each component can be quite different, up to 5%

```
--------------------------------------------------------------------------------
DSI10 Sim: Comparison between sampled and actual stationary distributions
           empirical: [0.33988473 0.33565942 0.32445585],
          stationary: [0.33291106 0.33333333 0.3337556 ]
Total variation error: 0.009299753874260325

DSI20 Sim: Comparison between sampled and actual stationary distributions
           empirical: [0.32508818 0.33406683 0.34084499],
          stationary: [0.33282629 0.33333333 0.33384037]
Total variation error: 0.007738115397553785

DSI30 Sim: Comparison between sampled and actual stationary distributions
           empirical: [0.34099031 0.32335747 0.33565222],
          stationary: [0.33288958 0.33333333 0.33377709]
Total variation error: 0.009975864629304138

================================================================================

--------------------------------------------------------------------------------
DSI10 Demo: Comparison between sampled and actual stationary distributions
           empirical: [0.350452   0.33337423 0.31617377],
          stationary: [0.33291106 0.33333333 0.3337556 ]
Total variation error: 0.017581833006066944

DSI20 Demo: Comparison between sampled and actual stationary distributions
           empirical: [0.33320428 0.32539281 0.34140291],
          stationary: [0.33282629 0.33333333 0.33384037]
Total variation error: 0.007940520844572418
```

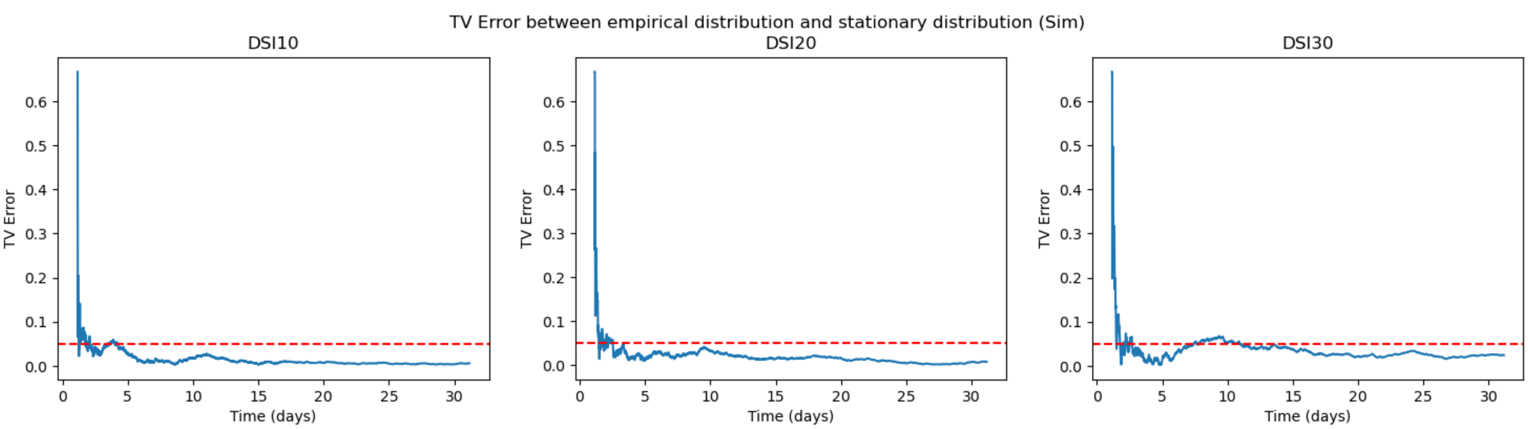
```
DSI30 Demo: Comparison between sampled and actual stationary distributions
            empirical: [0.34598387 0.31057309 0.34344304],
           stationary: [0.33288958 0.33333333 0.33377709]
Total variation error: 0.022760240697919626

================================================================================

--------------------------------------------------------------------------------
DSI10 DevServer: Comparison between sampled and actual stationary distributions
            empirical: [0.34610203 0.32896601 0.32493196],
           stationary: [0.33291106 0.33333333 0.3337556 ]
Total variation error: 0.013190968526982566

DSI20 DevServer: Comparison between sampled and actual stationary distributions
            empirical: [0.36081952 0.33185288 0.30732759],
           stationary: [0.33282629 0.33333333 0.33384037]
Total variation error: 0.027993229652031798

DSI30 DevServer: Comparison between sampled and actual stationary distributions
            empirical: [0.35483318 0.33763716 0.30752966],
           stationary: [0.33288958 0.33333333 0.33377709]
Total variation error: 0.026247422059553577

================================================================================
```

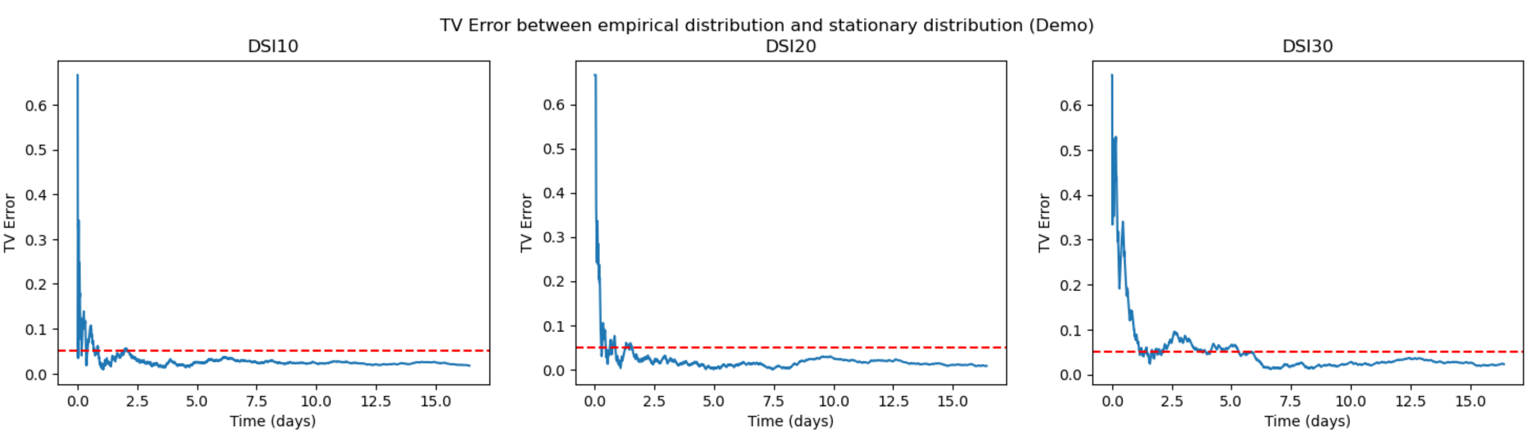
Convergence Testing

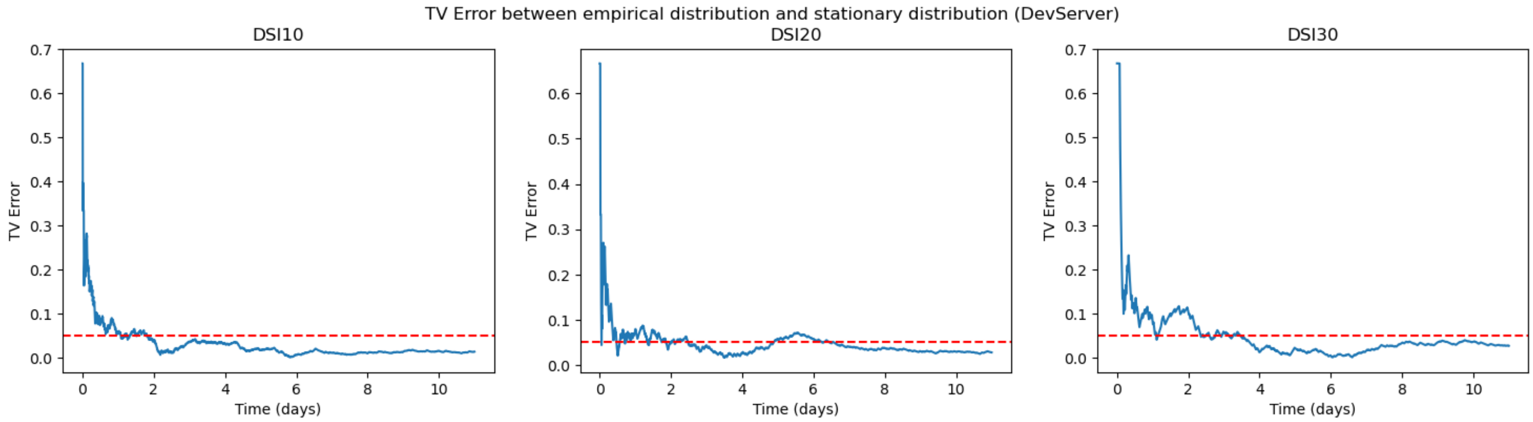
We check the convergence of the empirical distribution by calculating the empirical distribution at each step and checking the TV error. We test the same sets of data as before. We skip the first 100k data points or so, as the cumulative proportion can be very volatile at small sample sizes.

Result

The TV errors tend to enter a 0.05 threshold within a week.







Section 4

Moment cross-checking

We check the moments of real feed data and compare them to the simulation. In particular:

1. Obtain spot feed data from Metabase
2. Compute feed data moments of 1 tick log-returns (Mean, Volatility, Skewness, Kurtosis)
3. Run a 180 day simulation and compute their 1 tick log-return moments
4. Compare 2. and 3.

We expect to find a roughly centered normal-like distribution, i.e

- Skewness: 0
- (Excess) Kurtosis: 0

Results

We see that the Demo & Simulated data are close in volatility, but tend to differ in the other parameters.

Moments			**Mean**			**Volatility**
Data	**Sim**	**Dev**	**Demo**	**Sim**	**Dev**	**Demo**
DSI10	-0.00000002	0.00000003	0.00000014	0.1010398	0.10111771	0.10104511
DSI20	0.00000003	0.0000001	-0.00000003	0.10039703	0.10032422	0.10039828
DSI30	-0.00000001	0.00000009	-0.00000001	0.1001293	0.10017733	0.10009356

Moments			**Skew**			**Kurtosis**
Data	**Sim**	**Dev**	**Demo**	**Sim**	**Dev**	**Demo**
DSI10	-0.00067277	-0.00389256	-0.00372993	-0.00183724	-0.00248631	0.00143173
DSI20	0.0000533	0.00043905	0.00111	-0.00074716	-0.00612898	0.00192054
DSI30	-0.00067841	0.00368372	0.00250332	0.00031672	0.00378606	0.00346608

Error with Sim (%)		**Mean**		**Volatility**
Data	**Dev**	**Demo**	**Dev**	**Demo**
DSI10	228.04836215	696.10416673	-0.07710557	-0.0052574
DSI20	-275.55978188	205.08406846	0.07251995	-0.00124027

DSI30	1512.00158045	0.36792988	-0.04796731	0.03569853

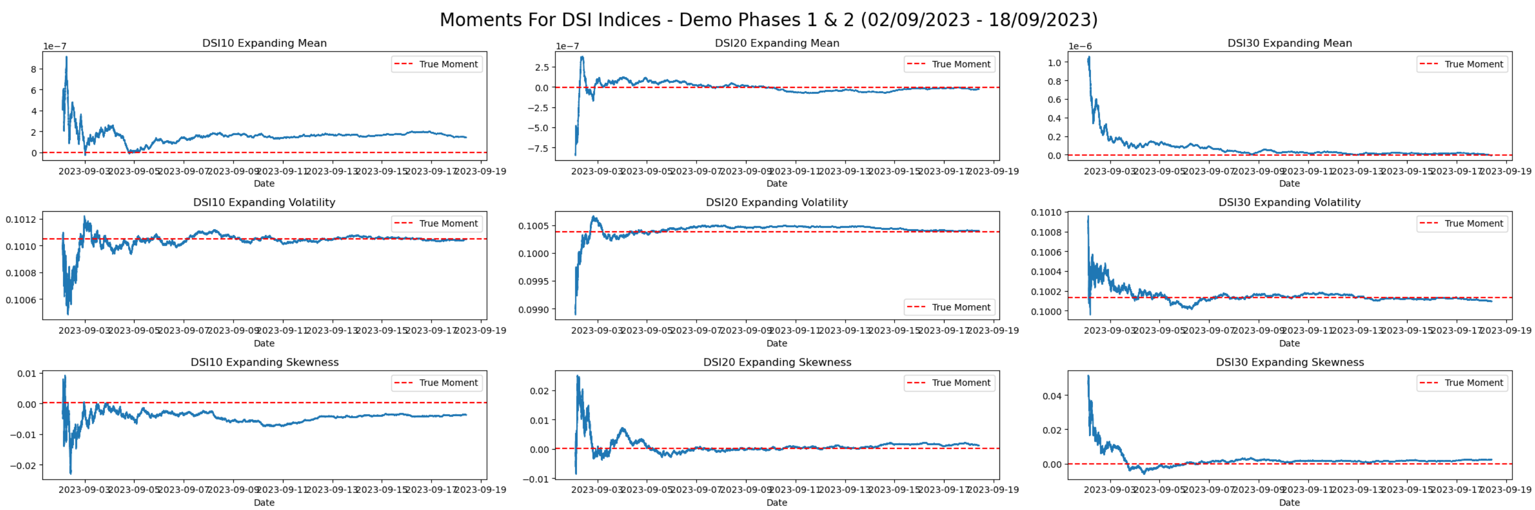
Error with Sim (%)	**Skew**		**Kurtosis**	
Data	**Dev**	**Demo**	**Dev**	**Demo**
DSI10	-478.58353651	-454.41061516	-35.32836057	177.92797236
DSI20	-723.70286035	-1982.4967726	-720.30789543	357.04669462
DSI30	642.99348216	468.99891949	-1095.40134948	-994.37305703

Convergence

We use expanding moments to observe the convergence

Results

We see that each moment converges approximately within a week



Section 5

We backwards engineer the parameters μ, σ and γ from the feed moments. Note that T is not in the closed-form expression of the DSI moments, so it can't be checked via this method (formulas in appendix).

We use the full Demo data for testing.

Results

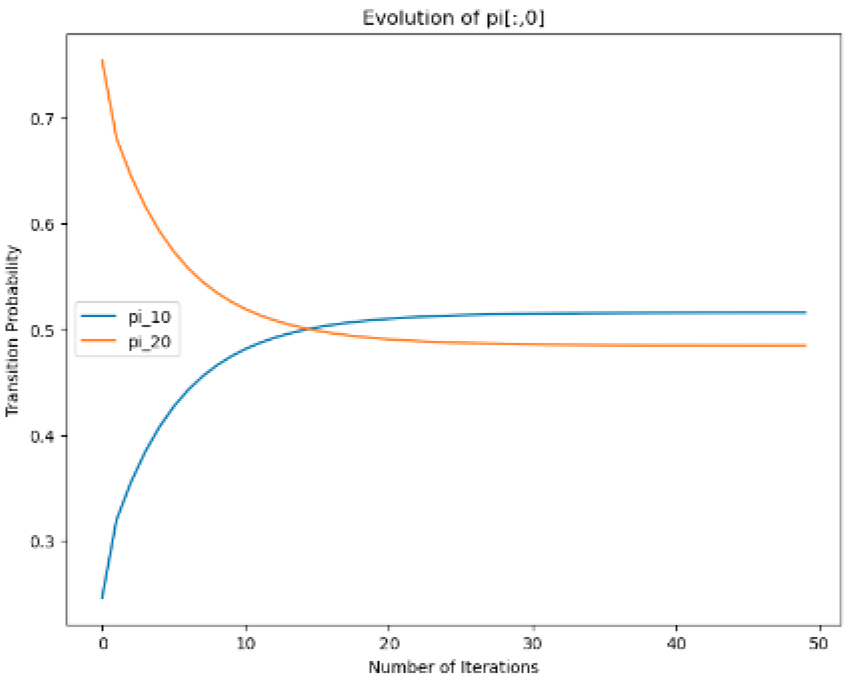
Results highly depend on the initial and boundary conditions, but are close to the actual values

Parameters	**Mu**		**Sigma**		**Gamma**	
Values	**Calculated**	**Actual**	**Calculated**	**Actual**	**Calculated**	**Actual**
DSI10	97.99998129	100	0.10976	0.1	0.52672653	0.49809979
DSI20	58.79999984	60	0.09809322	0.1	0.48823608	0.49771832
DSI30	34.30001209	35	0.09408435	0.1	0.45652895	0.49800312

MLE

We also try the MLE algorithm for the Demo data. The MLE algorithm attempts to infer μ, σ, T, the initial state ξ_0 and the transition probabilities. We run 50 iterations

Note that the transition probability graphs represent the exit distribution assuming the current state is x. For example, the following picture is an example of the case of 0.



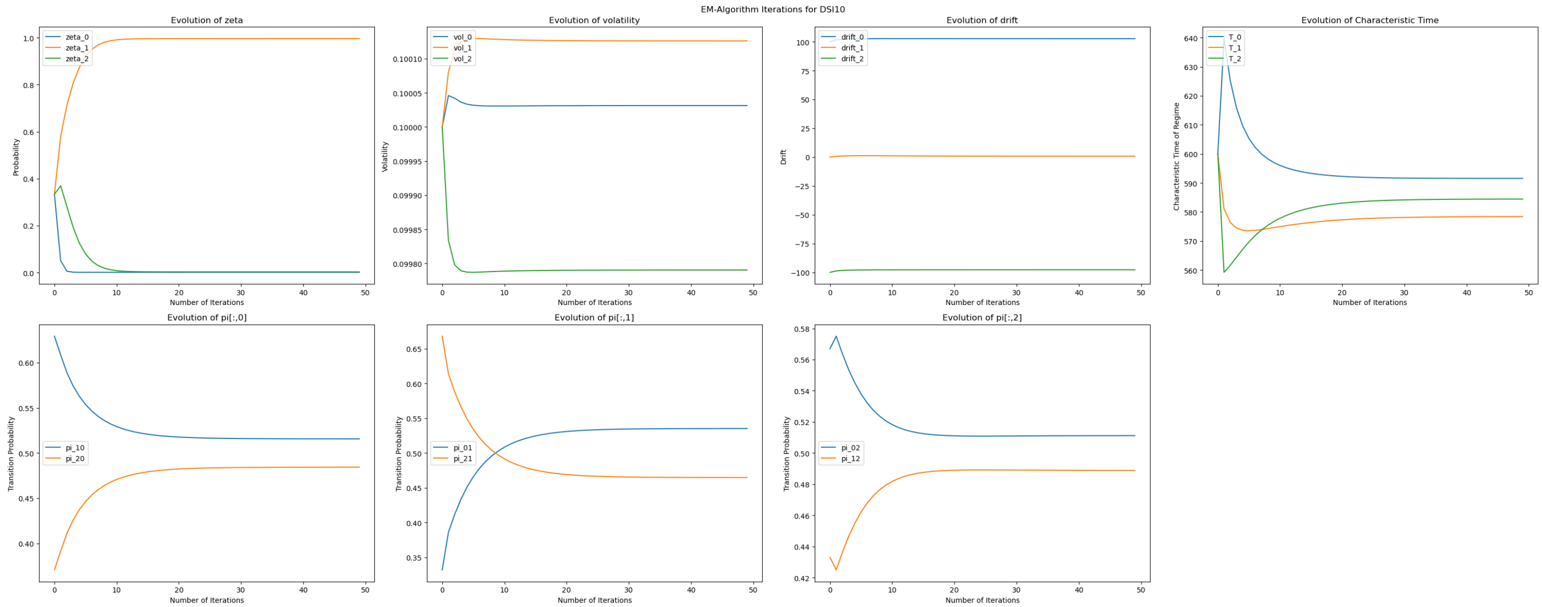
In this case, this implies that we've recovered the probabilities:

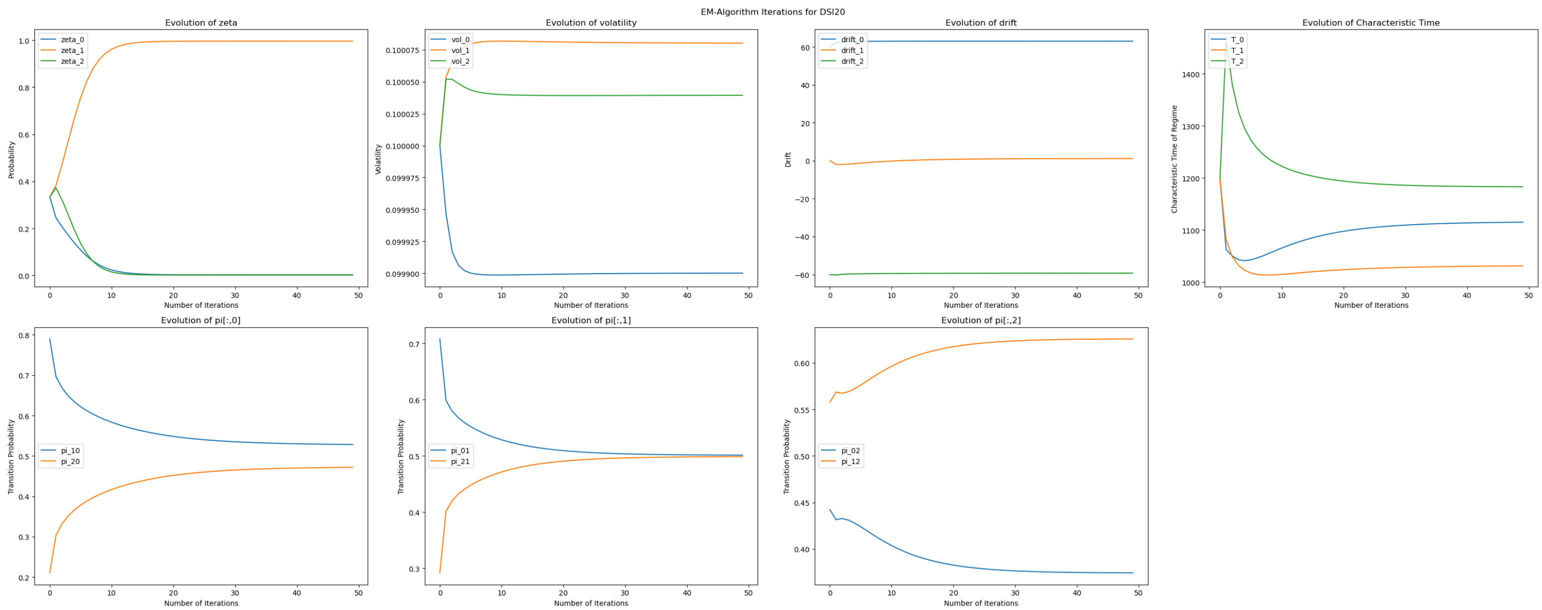
$$\begin{bmatrix} \mathbb{P}[X_t = 1 \mid X_{t-1} = 0] \\ \mathbb{P}[X_t = 2 \mid X_{t-1} = 0] \end{bmatrix} = \begin{bmatrix} \pi_{10} \\ \pi_{20} \end{bmatrix} \approx \begin{bmatrix} 0.51 \\ 0.49 \end{bmatrix}$$

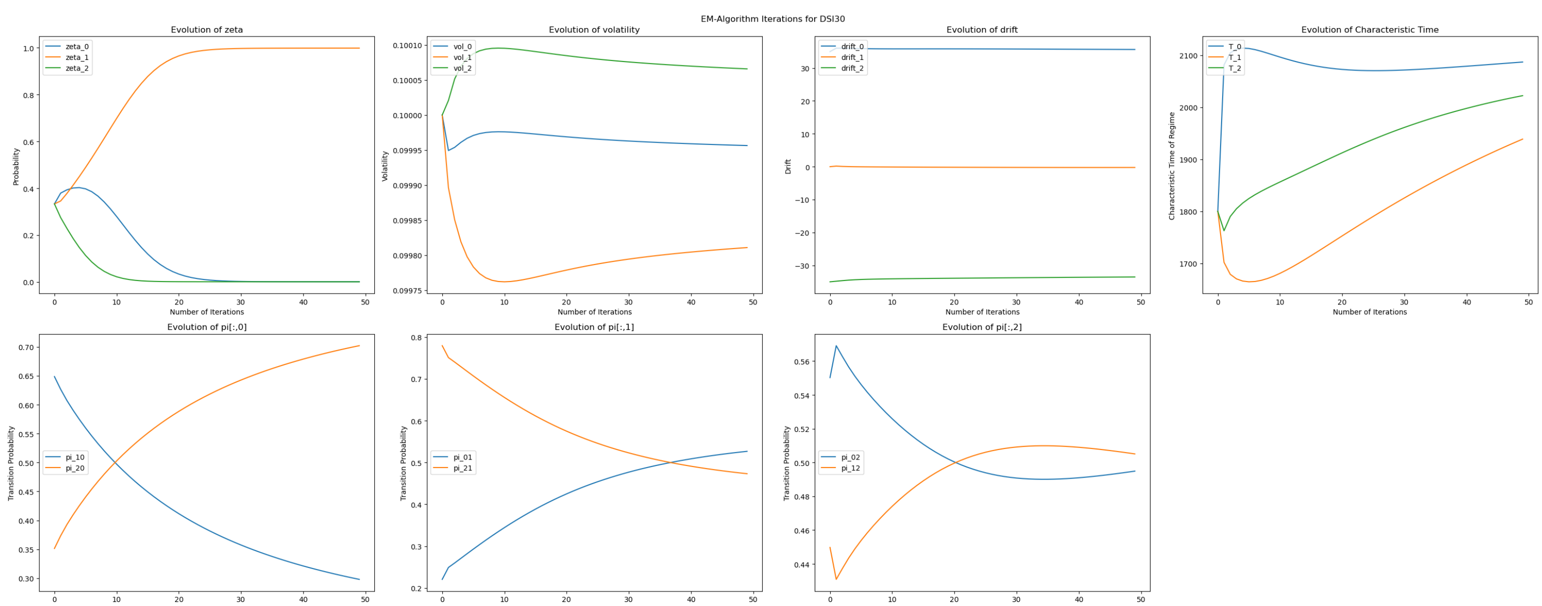
Recalling that the first column of P, i.e the probability distribution of X_t given X_{t-1} has distribution $[1-\lambda, \lambda/2, \lambda/2]^T$, this implies that the actual exit distribution given $X_t = 0$ is $[0.5, 0.5]^T$.

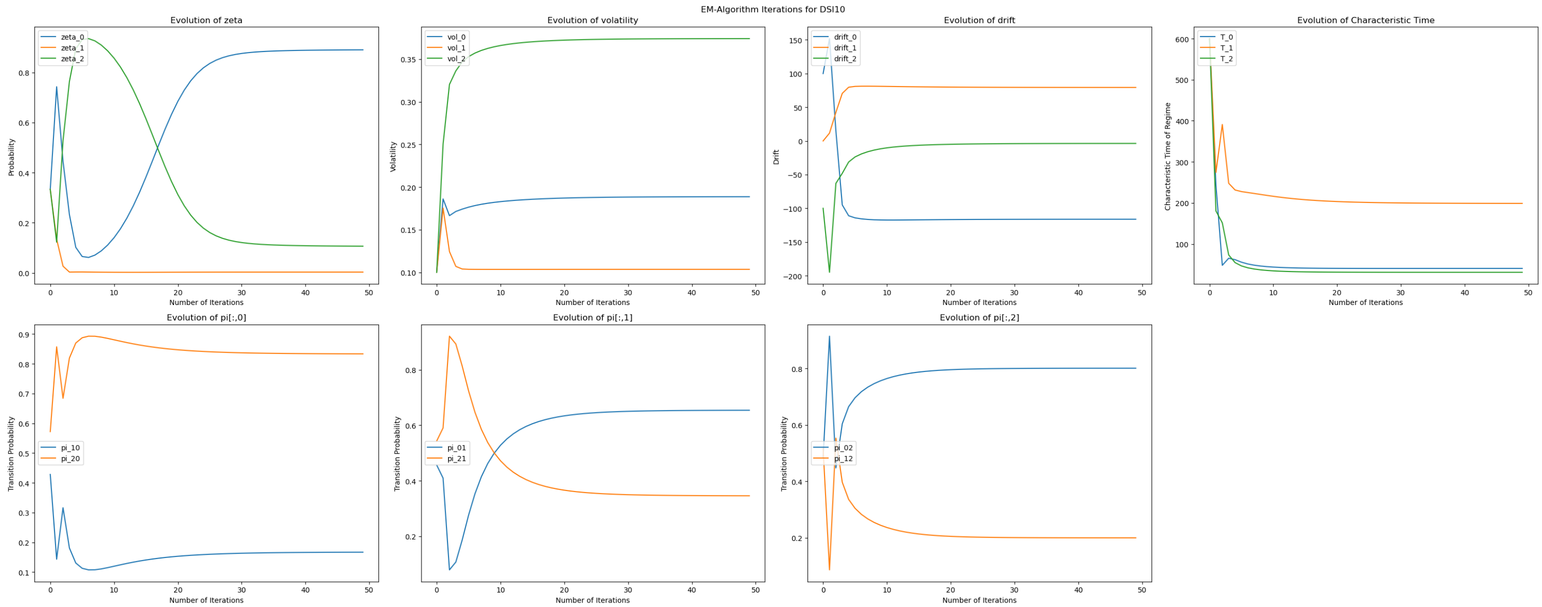
Results

- We see that μ, σ and ξ_0 roughly converge to what we expect for all the DSIs.
- T roughly converges well for DSI10, gets somewhat close for DSI20, and does not converge very well for DSI30. This is likely because there are not enough data points to do so for DSI20 & DSI30 , as DSI20/DSI30 only has 72/48 jumps per day on average and we have only 4 weeks of demo data points at the time of writing
- The transition probabilities for DSI10 are the best approximated, while DSI20 & DSI30 don't converge to the correct values with the current information


EM-Algorithm Iterations for DSI10
Evolution of zeta
zeta_0
zeta_1
zeta_2
Probability
Number of Iterations
Evolution of volatility
vol_0
vol_1
vol_2
Volatility
Evolution of drift
drift_0
drift_1
drift_2
Drift
Evolution of Characteristic Time
T_0
T_1
T_2
Characteristic Time of Regime
Evolution of pi[:,0]
pi_10
pi_20
Transition Probability
Evolution of pi[:,1]
pi_01
pi_21
Evolution of pi[:,2]
pi_02
pi_12


EM-Algorithm Iterations for DSI20
Evolution of zeta
zeta_0
zeta_1
zeta_2
Probability
Number of Iterations
Evolution of volatility
vol_0
vol_1
vol_2
Volatility
Evolution of drift
drift_0
drift_1
drift_2
Drift
Evolution of Characteristic Time
T_0
T_1
T_2
Characteristic Time of Regime
Evolution of pi[:,0]
pi_10
pi_20
Transition Probability
Evolution of pi[:,1]
pi_01
pi_21
Evolution of pi[:,2]
pi_02
pi_12


EM-Algorithm Iterations for DSI30
Evolution of zeta
zeta_0
zeta_1
zeta_2
Probability
Number of Iterations
Evolution of volatility
vol_0
vol_1
vol_2
Volatility
Evolution of drift
drift_0
drift_1
drift_2
Drift
Evolution of Characteristic Time
T_0
T_1
T_2
Characteristic Time of Regime
Evolution of pi[:,0]
pi_10
pi_20
Transition Probability
Evolution of pi[:,1]
pi_01
pi_21
Evolution of pi[:,2]
pi_02
pi_12



Section 6

As mentioned earlier, the spread formula is justified. In particular, if F_t is a forward index for DSI S_t, then we can set:

$$S_{ask} = \max(S_t, F_t)$$

$$S_{bid} = \min(S_t, F_t)$$

Here, these bid/ask prices are:

- Consistent (Any strategy and feed history results in negative expected client pnl)
- Fair (Based on previous bid-ask information, we can find a strategy such that its expected pnl is 0)

on the perfect strategy.

However, the calculations required for the above are rigid and do not leave much room for additional commissions/markups.

That being the case, the current spread model solves this issues, maintaining consistency, while while relaxing fairness enough so we can get a positive expected revenue.

For more details & a more formal justification, refer to the following documents:

1. Forward Indices (Justification of use of forward prices for pricing CFDs)
2. Pricing CFDs with Futures (Systematic method of finding consistent spread models assuming known forward pricing model)
3. Forward Pricing for DSI (Finding forward pricing for DSI)
4. CFD Spread Model for DSI (Consolidation of (3) and justification for using diff)
5. DSI - Expected Revenue (Justification for the expected revenue being solely dependent on the commission terms)

All documents have been validated.

NOTE: There is a recurrence relation in Forward Pricing for DSI whose derivation was added later. For some elaboration on the derivation, refer to the appendix

Spread parameters

According to CFD Spread Model for DSI, the spread formula used is appropriate. The document proves that basing the bid and ask prices on the expected infinite spot, X, is a consistent pricing for CFDs on the DSI.

Spread parameter calculation

We compare the calculations using CFD Spread Model for DSI with the preset parameters in DSI BO Specs.

Result

The calculated κ's and ϵ's match the corresponding values from from DSI BO Tool

Parameters		Bid Kappa		Ask Kappa		Bid Epsilon		Ask Epsilon
Values	**Calculated**	**Reference**	**Calculated**	**Reference**	**Calculated**	**Reference**	**Calculated**	**Reference**
DSI10	0.00126357	0.00126357	0.00126839	0.00126839	0.00000321	0.00000321	0.00000161	0.00000161
DSI20	0.00151513	0.00151513	0.00152207	0.00152207	0.00000463	0.00000463	0.00000232	0.00000232
DSI30	0.00132649	0.00132649	0.00133181	0.00133181	0.00000355	0.00000355	0.00000177	0.00000177

Error from Reference (%)	Bid Kappa	Ask Kappa	Bid Epsilon	Ask Epsilon
DSI10	0.00000001	0.00000004	0.00007197	0.00011067
DSI20	0.0	0.0	0.00000835	0.00002036
DSI30	0.00000011	0.00000007	0.00001352	0.00002925

Bid/Ask Spread Replication

Since the bid/ask spread is variable, we try to replicate the bid/ask spread with our own implementation of the DSI. We apply this to the the following data:

- Demo Phase 1
- Demo Phase 2

Results

The bid/ask prices match, minus rounding errors

```
-------------------------
Test Demo Phase 1 Data:
  Checking DSI10...
  Bid Prices match
  Ask Prices match

  Checking DSI20...
  Bid Prices match
  Ask Prices match

  Checking DSI30...
  Bid Prices match
  Ask Prices match
=========================
```

```
-------------------------
Test Demo Phase 2 Data:
  Checking DSI10...
  Bid Prices match
  Ask Prices match

  Checking DSI20...
  Bid Prices match
  Ask Prices match

  Checking DSI30...
  Bid Prices match
  Ask Prices match
=========================
```

Section 7

We check correlation of the DSIs to see if pair trading is possible. We expect a low correlation. If there is high correlation, there is possible concentration risk. We check for tick-wise and hourly log returns

We compare:

- Feeddb Spots (Dev/Demo Full)
- Feeddb/MT5 Bid (Demo Phase 1/2)
- Feeddb/MT5 Ask (Demo Phase 1/2)

Results

- Feeddb (Dev):
  - Spot log returns have low correlation
- Feeddb (Demo - Full/Phase 1 & 2):
  - Spot log returns have low correlation
  - Bid log returns have low correlation
  - Ask log returns have low correlation

Tick-wise

	logr_spot_DSI10_dev	logr_spot_DSI20_dev	logr_spot_DSI30_dev
logr_spot_DSI10_dev	1.00000000	-0.00192955	0.00031228
logr_spot_DSI20_dev	-0.00192955	1.00000000	-0.00026062
logr_spot_DSI30_dev	0.00031228	-0.00026062	1.00000000

	logr_spot_DSI10_demo_full	logr_spot_DSI20_demo_full	logr_spot_DSI30_demo_full
logr_spot_DSI10_demo_full	1.00000000	-0.00070600	0.00015378
logr_spot_DSI20_demo_full	-0.00070600	1.00000000	-0.00117092
logr_spot_DSI30_demo_full	0.00015378	-0.00117092	1.00000000

	logr_bid_DSI10_demo_1	logr_bid_DSI20_demo_1	logr_bid_DSI30_demo_1
logr_bid_DSI10_demo_1	1.00000000	-0.00108201	0.00162959
logr_bid_DSI20_demo_1	-0.00108201	1.00000000	-0.00208722

logr_bid_DSI30_demo_1	0.00162959	-0.00208722	1.00000000

	logr_ask_DSI10_demo_2	logr_ask_DSI20_demo_2	logr_ask_DSI30_demo_2
logr_ask_DSI10_demo_2	1.00000000	0.00015183	-0.00037784
logr_ask_DSI20_demo_2	0.00015183	1.00000000	-0.00034401
logr_ask_DSI30_demo_2	-0.00037784	-0.00034401	1.00000000

	logr_bid_DSI10_demo_2	logr_bid_DSI20_demo_2	logr_bid_DSI30_demo_2
logr_bid_DSI10_demo_2	1.00000000	-0.00156085	-0.00153604
logr_bid_DSI20_demo_2	-0.00156085	1.00000000	-0.00062528
logr_bid_DSI30_demo_2	-0.00153604	-0.00062528	1.00000000

	logr_ask_DSI10_demo_2	logr_ask_DSI20_demo_2	logr_ask_DSI30_demo_2
logr_ask_DSI10_demo_2	1.00000000	-0.00059276	0.00003137
logr_ask_DSI20_demo_2	-0.00059276	1.00000000	-0.00021295
logr_ask_DSI30_demo_2	0.00003137	-0.00021295	1.00000000

Hourly

	logr_spot_DSI10_dev	logr_spot_DSI20_dev	logr_spot_DSI30_dev
logr_spot_DSI10_dev	1.00000000	-0.08035416	0.00391263
logr_spot_DSI20_dev	-0.08035416	1.00000000	0.03361912
logr_spot_DSI30_dev	0.00391263	0.03361912	1.00000000

	logr_spot_DSI10_demo_full	logr_spot_DSI20_demo_full	logr_spot_DSI30_demo_full
logr_spot_DSI10_demo_full	1.00000000	0.02765958	0.06689022
logr_spot_DSI20_demo_full	0.02765958	1.00000000	-0.02939312
logr_spot_DSI30_demo_full	0.06689022	-0.02939312	1.00000000

	logr_bid_DSI10_demo_1	logr_bid_DSI20_demo_1	logr_bid_DSI30_demo_1
logr_bid_DSI10_demo_1	1.00000000	0.08030117	0.04583827
logr_bid_DSI20_demo_1	0.08030117	1.00000000	-0.03686655
logr_bid_DSI30_demo_1	0.04583827	-0.03686655	1.00000000

	logr_ask_DSI10_demo_2	logr_ask_DSI20_demo_2	logr_ask_DSI30_demo_2
logr_ask_DSI10_demo_2	1.00000000	0.08155024	0.04482290
logr_ask_DSI20_demo_2	0.08155024	1.00000000	-0.03915401
logr_ask_DSI30_demo_2	0.04482290	-0.03915401	1.00000000

	logr_bid_DSI10_demo_2	logr_bid_DSI20_demo_2	logr_bid_DSI30_demo_2
logr_bid_DSI10_demo_2	1.00000000	-0.06823288	0.07062304
logr_bid_DSI20_demo_2	-0.06823288	1.00000000	0.04847695
logr_bid_DSI30_demo_2	0.07062304	0.04847695	1.00000000

	logr_ask_DSI10_demo_2	logr_ask_DSI20_demo_2	logr_ask_DSI30_demo_2
logr_ask_DSI10_demo_2	1.00000000	-0.07365944	0.07828129
logr_ask_DSI20_demo_2	-0.07365944	1.00000000	0.04377123
logr_ask_DSI30_demo_2	0.07828129	0.04377123	1.00000000

Appendix

Deriving the limiting/stationary distribution π

The limiting/stationary distribution is required to calculate the moments of the DSI index. This section will detail the definition and calculations of the stationary distribution.

The limiting distribution is defined as the limit:

$$\pi = \lim_{n \to \infty} P^n x$$

For a distribution vector x. As the transition matrix P is positive recurrent, the limiting distribution is also the stationary distribution. Since this is the case, we can calculate it as follows:

We solve the set of simultaneous equations presented by:

$$P\pi = \pi$$
$$\iff \begin{pmatrix} 1-\lambda & \gamma\lambda & \frac{\lambda}{2} \\ \frac{\lambda}{2} & 1-\lambda & \frac{\lambda}{2} \\ \frac{\lambda}{2} & (1-\gamma)\lambda & 1-\lambda \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \\ p_2 \end{pmatrix} = \begin{pmatrix} p_0 \\ p_1 \\ p_2 \end{pmatrix}$$

w.r.t p_0, p_1, p_2. Yielding:

$$\pi = \begin{bmatrix} \frac{2}{9}(1+\gamma) \\ \frac{1}{3} \\ \frac{2}{9}(2-\gamma) \end{bmatrix}$$

DSI log return MGF

We can calculate the moment-generating function for the log returns:

$$R_t = \log \frac{S_t}{S_{t-1}} = \left(\mu(X_t) - \frac{\sigma^2}{2}\right) dt + \sigma\sqrt{dt}W$$

Indeed, using the independence of X_t and W:

$$\begin{aligned}\mathbb{E}[e^{sR_t}] &= \mathbb{E}\left[e^{s\left(\mu(X_t)-\frac{\sigma^2}{2}\right)dt+\sigma\sqrt{dt}W}\right] \\ &= \mathbb{E}\left[e^{s(\mu(X_t)dt)}\right] \cdot \mathbb{E}\left[e^{s\left(-\frac{\sigma^2}{2}dt\right)}\right] \cdot \mathbb{E}\left[e^{s(\sigma\sqrt{dt}W)}\right]\end{aligned}$$

We use the stationary distribution for the first term:

$$\mathbb{E}\left[e^{s(\mu(X_t)dt)}\right] = \frac{2}{9}(1+\gamma)e^{\mu dt\cdot s} + \frac{1}{3} + \frac{2}{9}(2-\gamma)e^{-\mu dt\cdot s}$$

The second term is an expectation of a constant, and the we can evaluate the third term as usual:

$$\mathbb{E}\left[e^{s(\sigma\sqrt{dt}W)}\right] = e^{\frac{\sigma^2}{2}dt\cdot s}$$

Putting everything together:

$$MGF_{R_t}(s) = \mathbb{E}[e^{sR_t}] = e^{\frac{\sigma^2}{2}dt(s^2-s)}\left(\frac{2}{9}(1+\gamma)e^{\mu dt\cdot s} + \frac{2}{9}(2-\gamma)e^{-\mu dt\cdot s} + \frac{1}{3}\right)$$

Setting $D = \mu(X_t)dt$ and $N = \sigma\sqrt{dt}W - \frac{\sigma^2}{2}dt$, we can rewrite the moment generating function:

$$MGF_{N+D}(s) = MGF_N(s)\cdot MGF_D(s)$$

Where:

$$\begin{aligned} MGF_N(s) &= e^{\frac{\sigma^2}{2}dt(s^2-s)} \\ MGF_D(s) &= \frac{2}{9}(1+\gamma)e^{\mu dt\cdot s} + \frac{2}{9}(2-\gamma)e^{-\mu dt\cdot s} + \frac{1}{3} \end{aligned}$$

DSI log return moments:

We use the MGF to find the moments of R_t using:

$$\mathbb{E}[(R_t)^n] = \frac{d^n}{ds^n}MGF_{N+D}(s)\Big|_{s=0}$$

Calculating the first three moments:

$$\mathbb{E}[R_t] = \left(\frac{2}{9}\mu(2\gamma-1) - \frac{\sigma^2}{2}\right)dt$$

$$\mathbb{E}[R_t^2] = \left(\frac{2}{3}\mu^2 - \frac{2}{9}\mu\sigma^2(2\gamma-1) + \frac{\sigma^4}{4}\right)dt^2 + \sigma^2 dt$$

$$\mathbb{E}[R_t^3] = \left(\frac{2}{9}\mu^3(2\gamma-1) - \frac{\sigma^2}{3}\mu^2 + \frac{2}{9}\mu\sigma^4(2\gamma-1) - \frac{\sigma^6}{8}\right)dt^3 + \left(\frac{2}{9}\mu\sigma^2(2\gamma-1) - \frac{3}{2}\sigma^4\right)dt^2$$

We can write this in terms of moments of $\mu(X_t)$. Note that the moments for $\mu(X_t)$ are:

$$\mathbb{E}[\mu(X_t)^n] = \begin{cases} \frac{2}{9}\mu^n(2\gamma-1) & \text{for } n \text{ odd} \\ \frac{2}{3}\mu^n & \text{for } n \text{ even} \end{cases}$$

Therefore, we rewrite the moments of R_t:

$$\mathbb{E}[R_t] = \left(\mathbb{E}[\mu(X_t)] - \frac{\sigma^2}{2}\right)dt$$

$$\mathbb{E}[R_t^2] = \left(\mathbb{E}[\mu(X_t)^2] - \sigma^2\mathbb{E}[\mu(X_t)] + \frac{\sigma^4}{4}\right)dt^2 + \sigma^2 dt$$

$$\mathbb{E}[R_t^3] = \left(\mathbb{E}[\mu(X_t)^3] - \frac{\sigma^2}{2}\mathbb{E}[\mu(X_t)^2] + \sigma^4\mathbb{E}[\mu(X_t)] - \frac{\sigma^6}{8}\right) dt^3 + \left(\sigma^2\mathbb{E}[\mu(X_t)] - \frac{3}{2}\sigma^4\right) dt^2$$

Forward Pricing Recursive Equations

Previously, in Forward Pricing for DSI, there was no derivation for the recursive relations. It has been already updated to include one. The following below is the same derivation but more detailed for my own understanding.

We define:

- $R \in \{+, 0, -\}$ are the possible regimes;
- $S_t \xrightarrow{R} S_{t+1}$ is the event that $X_t = R$, i.e the step to S_{t+1} follows regime R ($X_t = R$);
- $F_1 = \begin{pmatrix} F_1^+ & F_1^0 & F_1^- \end{pmatrix}$ is the vector of prices of a forward contract on DSI after n iterations. Each component is the forward contract price if the current spot is 1, and if the regime taking $S_0 \to S_1$ is $+$, 0 and $-$ respectively. In the document, it is labeled as the vector of the forward price after 1 iteration. Will also be denoted as B;
- $B = F_1 = \begin{pmatrix} e^{\mu dt} - 1 & 0 & e^{-\mu dt} - 1 \end{pmatrix}^T$ is the vector of prices of a forward contract on DSI if the current spot is 1 after 1 iteration (The 1 tick returns);
- $\mathbb{P}(R_1 \to R_2) = \mathbb{P}(X_1 = R_2 \mid X_0 = R_1)$ is the transition probability $P_{R_2 \leftarrow R_1}$

We consider F_{m+1}^+, for $m \geq 1$. The $0, -$ cases follow similar proofs.

We can rewrite F_{m+1}^+ in terms of the expected future spot, given that we start at 1, and $S_0 \xrightarrow{+} S_1$:

$$1 + F_{m+1}^+ = \mathbb{E}[S_{m+1} \mid S_0 = 1, S_0 \xrightarrow{+} S_1]$$

We condition on each possible $S_1 = s$ and also against the transition from $S_1 \xrightarrow{R} S_2$ for each regime R:

$$\begin{aligned} 1 + F_{m+1}^+ = \int_{\mathbb{R}} \sum_{R \in \{+,0,-\}} & \mathbb{E}[S_{m+1} \mid S_0 = 1, S_0 \xrightarrow{+} S_1, S_1 = s, S_1 \xrightarrow{R} S_2] \\ & \cdot \mathbb{P}(S_1 = s, S_1 \xrightarrow{R} S_2 \mid S_0 = 1, S_0 \xrightarrow{+} S_1)\, ds \end{aligned}$$

We decompose each of the two terms:

The conditional expectation can be simplified:

1. The events $\{S_0 = 1, S_0 \xrightarrow{+} S_1\}$ & $\{S_1 = s, S_1 \xrightarrow{R} S_2 \mid S_0\}$ are independent. Moreover, under the standard filtration for the DSI, we see that the former event is measurable w.r.t the latter event. Using this, we can reduce:

$$\begin{aligned} \mathbb{E}[S_{m+1} \mid S_0 = 1, S_0 \xrightarrow{+} S_1, S_1 = s, S_1 \xrightarrow{R} S_2] &= \mathbb{E}[\mathbb{E}(S_{m+1} \mid S_0 = 1, S_0 \xrightarrow{+} S_1) \mid S_1 = s, S_1 \xrightarrow{R} S_2] \\ &= \mathbb{E}[S_{m+1} \mid S_1 = s, S_1 \xrightarrow{R} S_2] \end{aligned}$$

1. Since S_t is a martingale & $f_t(R) = S_t \xrightarrow{R} S_{t+1}$ is Markov on t, we can move 1 step back:

$$\mathbb{E}[S_{m+1} \mid S_1 = s, S_1 \xrightarrow{R} S_2] = \mathbb{E}[S_m \mid S_0 = s, S_0 \xrightarrow{R} S_1]$$

1. S_t scales with the initial spot, S_0, so:

$$\mathbb{E}[S_m \mid S_0 = s, S_0 \xrightarrow{R} S_1] = s \cdot \mathbb{E}[S_m \mid S_0 = 1, S_0 \xrightarrow{R} S_1]$$

Now for the probability term:

1. $\{S_1 = s\}$ and $\{S_1 \xrightarrow{R} S_2\}$ are independent:

$$\begin{aligned}\mathbb{P}(S_1 = s, S_1 \xrightarrow{R} S_2 \mid S_0 = 1, S_0 \xrightarrow{+} S_1) = \mathbb{P}(S_1 = s \mid S_0 = 1, S_0 \xrightarrow{+} S_1)\\ \cdot \mathbb{P}(S_1 \xrightarrow{R} S_2 \mid S_0 = 1, S_0 \xrightarrow{+} S_1)\end{aligned}$$

1. $\{S_1 \xrightarrow{R} S_2\}$ is independent of $\{S_0 = 1\}$ and therefore the second probability is just a transition probability

$$\begin{aligned}\mathbb{P}(S_1 = s, S_1 \xrightarrow{R} S_2 \mid S_0 = 1, S_0 \xrightarrow{+} S_1) &= \mathbb{P}(S_1 = s \mid S_0 = 1, S_0 \xrightarrow{+} S_1)\\ &\quad \cdot \mathbb{P}(S_1 \xrightarrow{R} S_2 \mid S_0 \xrightarrow{+} S_1)\\ &= \mathbb{P}(S_1 = s \mid S_0 = 1, S_0 \xrightarrow{+} S_1)\\ &\quad \cdot \mathbb{P}(+ \to R)\end{aligned}$$

Reducing equation (1) with the above:

$$\begin{aligned}1 + F^+_{m+1} &= \int_{\mathbb{R}} \sum_{R \in \{+,0,-\}} s \cdot \mathbb{E}[S_m \mid S_0 = 1, S_0 \xrightarrow{R} S_1] \cdot \mathbb{P}(S_1 = s \mid S_0 = 1, S_0 \xrightarrow{+} S_1) \cdot \mathbb{P}(+ \to R)\, ds\\ &= \sum_{R \in \{+,0,-\}} \left(\mathbb{E}[S_m \mid S_0 = 1, S_0 \xrightarrow{R} S_1] \cdot \mathbb{P}(+ \to R) \cdot \int_{\mathbb{R}} s \cdot \mathbb{P}(S_1 = s \mid S_0 = 1, S_0 \xrightarrow{+} S_1)\, ds \right)\\ &= \sum_{R \in \{+,0,-\}} \left(\mathbb{E}[S_m \mid S_0 = 1, S_0 \xrightarrow{R} S_1] \cdot \mathbb{P}(+ \to R) \cdot \mathbb{E}[S_1 \mid S_0 = 1, S_0 \xrightarrow{+} S_1] \right)\end{aligned}$$

Recalling the definition of F^+_m and expanding:

$$\begin{aligned}1 + F^+_{m+1} &= \sum_{R \in \{+,0,-\}} (1 + F^R_m) \cdot (1 + F^+_1) \cdot \mathbb{P}(+ \to R)\\ &= 1 + F^+_1 + \sum_{R \in \{+,0,-\}} F^R_m \cdot (1 + F^+_1) \cdot \mathbb{P}(+ \to R)\\ F^+_{m+1} &= F^+_1 + \sum_{R \in \{+,0,-\}} F^R_m \cdot (1 + F^+_1) \cdot \mathbb{P}(+ \to R)\end{aligned}$$

Repeating for F^0_{m+1}, F^-_{m+1}, we get the same recursive relations as specified in the document.

γ Derivation

γ is derived in the R&D document. The derivation of the recursive expressions for X^R_n are not presented however. To obtain them, we can run a similar process as above. We define the following:

- T_n is the time of the n-th change in regime with $T_0 = 0$;
- R_t is the regime at time t;

- R_{T_n} is the regime after the n-th change.

Recalling from the R&D document that:

- γ_{r_2,r_1} is the probability that the next regime is r_2, given it was r_1 before;
- $e_{R,1}$ is the expected move assuming a start at the regime R until the next regime change, i.e $e_{+,1} = \mathbb{E}[S_{T_1} \mid S_0 = 1, R_{T_0} = +]$;
- $e_{R,n}$ is the expected move assuming a start at the regime R until the n-th regime change, i.e $e_{+,n} = \mathbb{E}[S_{T_n} \mid S_0 = 1, R_{T_0} = +]$;
- $X_{R,n} = e_{R,n} - 1$;
- $x_R = e_{R,1} - 1$.

In particular, from previous computations in the R&D document:

- $x_0 = 0$;
- $x_+ = \frac{\lambda e^{\mu dt}}{1-e^{\mu dt}(1-\lambda)} - 1$;
- $x_- = \frac{\lambda e^{-\mu dt}}{1-e^{-\mu dt}(1-\lambda)} - 1$

NOTE: The e_R's have a different definition than the R&D document (p23), yet are still equivalent as:

1. The e_R in the doc is equivalent to the expected move assuming a regime change instead of just moving into the stationary regime (As any regime change from the positive/negative is just moving into the stationary regime in the 2-ways case)
2. Hence, the e_R in terms of regime change in the 2-ways case is equivalent to the e_R in the 3-ways case, as both indices have equal chances of exiting a regime for any given starting regime.

We consider the positive regime case and expand in a similar manner to the proof before:

$$1 + X_{+,n+1} = \mathbb{E}[S_{T_{n+1}} \mid S_0 = 1, R_{T_0} = +]$$

Which leads to

$$1 + X_{+,n+1} = \int_{\mathbb{R}} \sum_{R \in \{+,0,-\}} \mathbb{E}[S_{T_{n+1}} \mid S_0 = 1, R_{T_0} = +, S_{T_1} = s, R_{T_1} = R] \\ \cdot \mathbb{P}(S_{T_1} = s, R_{T_1} = R \mid S_0 = 1, R_{T_0} = +)\, ds$$

Invoking measurability & independence:

$$1 + X_{+,n+1} = \int_{\mathbb{R}} \sum_{R \in \{+,0,-\}} \mathbb{E}[S_{T_{n+1}} \mid S_{T_1} = s, R_{T_1} = R] \cdot \mathbb{P}(S_{T_1} = s \mid S_0 = 1, R_{T_0} = +) \\ \cdot \mathbb{P}(R_{T_1} = R \mid R_{T_0} = +)\, ds$$

Recognizing that $\mathbb{P}(R_{T_1} = R \mid R_{T_0} = +)$ is just $\gamma_{R,+}$ and applying scaling invariance & independent increments to S_t:

$$1 + X_{+,n+1} = \int_{\mathbb{R}} \sum_{R \in \{+,0,-\}} s \cdot \mathbb{E}[S_{T_n} \mid S_0 = 1, R_{T_0} = R] \cdot \mathbb{P}(S_{T_1} = s \mid S_0 = 1, R_{T_0} = +) \cdot \gamma_{R,+}\, ds$$

Bringing out $\mathbb{E}[S_{T_n} \mid S_0 = 1, R_{T_0} = R]$ of the integral and evaluating it:

$$1 + X_{+,n+1} = \sum_{R \in \{+,0,-\}} \mathbb{E}[S_{T_n} \mid S_0 = 1, R_{T_0} = R] \cdot \mathbb{E}(S_{T_1} = s \mid S_0 = 1, R_{T_0} = +) \cdot \gamma_{R,+}$$

Rewriting in terms of x_R & X_{R_n} and expanding:

$$1 + X_{+,n+1} = \sum_{R \in \{+,0,-\}} (1 + X_{R,n})(1 + x_+) \cdot \gamma_{R,+} = 1 + x_+ + \sum_{R \in \{+,0,-\}} (1 + x_+) \cdot X_{R,n} \cdot \gamma_{R,+}$$

We get our relation:

$$X_{+,n+1} = x_+ + \sum_{R \in \{+,0,-\}} (1 + x_+) \cdot X_{R,n} \cdot \gamma_{R,+}$$

R&D effort needs to be in line with Deriv's vision and mission as formulated by our CEO. Therefore all R&D projects are carefully selected by our C-Level senior management represented by JY and Rakshit and resources for the projects are only allocated after review and shortlisting based on their vision and priorities.

In line with the standards and criterias set out by the CEO, the Model Validation team has validated the product/indices as documented in this report.