60802 Dynamic Time Sync Error –
Error Model & Monte Carlo Method Analysis

David McCall & Kevin Stanton (Intel)

November 2021 IEEE 802 Plenary – 802.1 TSN – IEEE/IEC 60802

Abstract

- Industrial Automation Systems require microsecond-accurate time across long daisy-chains of devices using IEEE Std. 802.1AS™-2020 as specified by IEEE/IEC 60802.
- Simulated protocol and system parameters have thus far either been judged impractical or have failed to meet the time-accuracy requirement.
- An analysis of how errors accumulate suggested that a Monte Carlo method analysis could support fast iteration of potential scenarios and deliver insights into cause and effect.
  - See 60802-McCall-et-al-Time-Sync-Error-Model-0921-v03.pdf
- In this contribution we:
  - Describe a Monte Carlo method analysis programmed in Rstudio
  - Compare the analysis' results to results from previous time series simulations
  - Present a detailed analysis of sensitivities and trade offs
  - Recommend approaches to achieve the stated goals and propose next steps

Content

- Background & Recap
- Which Errors to Model & How They Add Up
- Monte Carlo Method Analysis Overview (RStudio)
- Comparison with Time Series Simulations
  - 7-Sigma Limit
- Error Analysis
  - Graphical Representation
  - Sensitivities & Trade Offs
- Recommendations & Next Steps

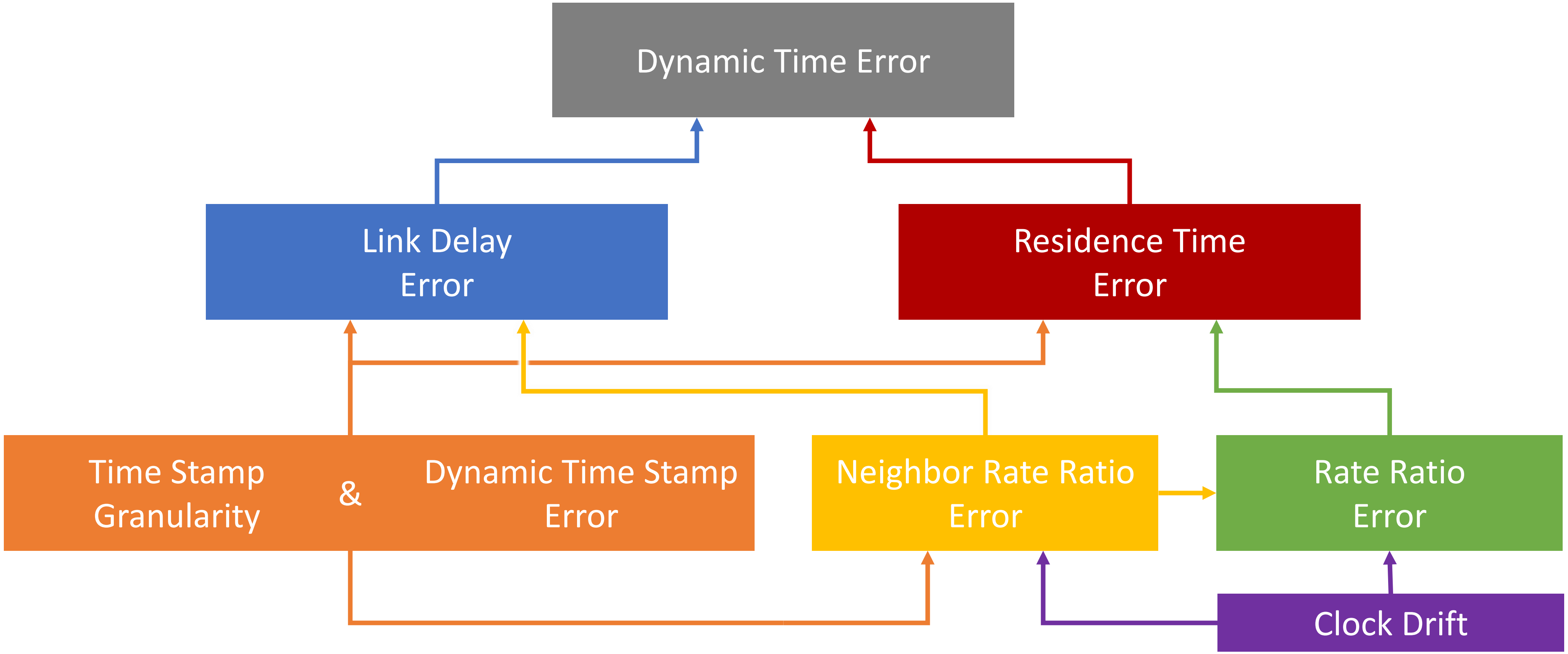
Background & Recap

In addition to the abstract…

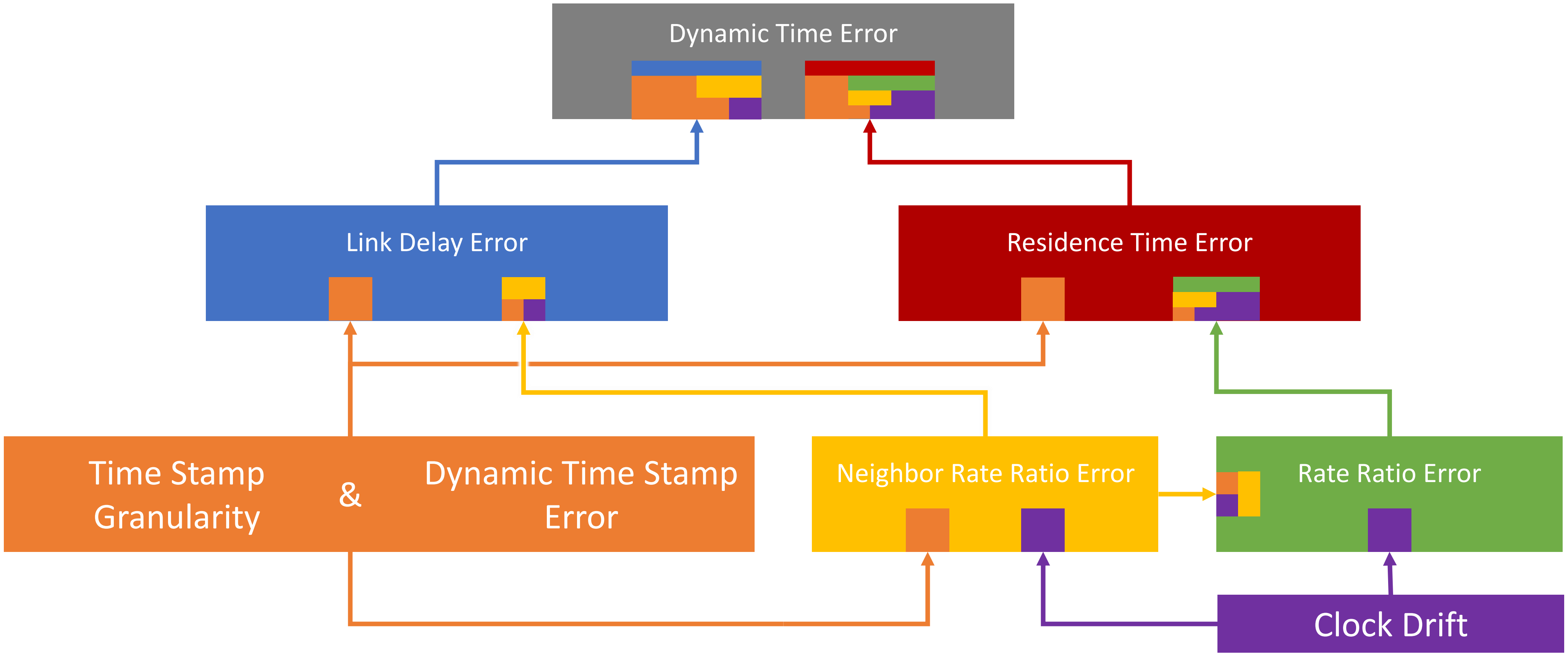
- The Monte Carlo analysis is intended as an addition to the toolbox, not an alternative to Time Series simulation.
- If successful it should provide…
  - The ability to iterate much faster
  - Greater insight into the source of errors and how they accumulate
  - Greater confidence that when selecting parameters to achieve a desired goal
  - Input into future Time Series simulations

Which Errors to Model & How They Add Up

Time Sync – Errors to Model



Time Sync – How Errors Add Up



All errors in this analysis are caused by either Clock Drift or Timestamp Errors

Monte Carlo Method Analysis

Overview of the modelling approach and analysis tool built in RStudio

Monte Carlo Method Analysis

- Inputs
- Errors, Formulae & Observations
- R & RStudio
  - Script code availability
- Demo

Input Errors

Error	Distribution	Default Min	Default Max	Equation	Unit
$clockDrift_{GM}$ $(ClockDrift_{GMmin}$ $\&\ ClockDrift_{GMmax})$	X% Stable (zero) (100-X)% Uniform	-0.6	+0.6	$clockDrift_{GM} \sim U(clockDrift_{GMmin}, clockDrift_{GMmax})$	ppm/s
$clockDrift$	X% Stable (zero) (100-X)% Uniform	-0.6	+0.6	$clockDrift \sim U(-clockDrift, +clockDrift)$	ppm/s
$TSGE_{TX}$	Uniform	-4	+4	$TSGE_{TX} \sim U(-TSGETX, +TSGETX)$	ns
$TSGE_{RX}$	Uniform	-4	+4	$TSGE_{RX} \sim U(-TSGERX, +TSGERX)$	ns
$DTSE_{TX}$	Uniform	-2	+2	$DTSE_{TX} \sim U(-DTSETX, +DTSETX)$	ns
$DTSE_{RX}$	Uniform	-1	+1	$DTSE_{RX} \sim U(-DTSERX, +DTSERX)$	ns

Formulae below take into account the unit of the input value.
Formulae in the main section of the September presentation did not.

Input Parameters

Error	Default Value	Unit
pDelayInterval	1,000	ms
pDelayTurnaround	10	ms
residenceTime	10	ms

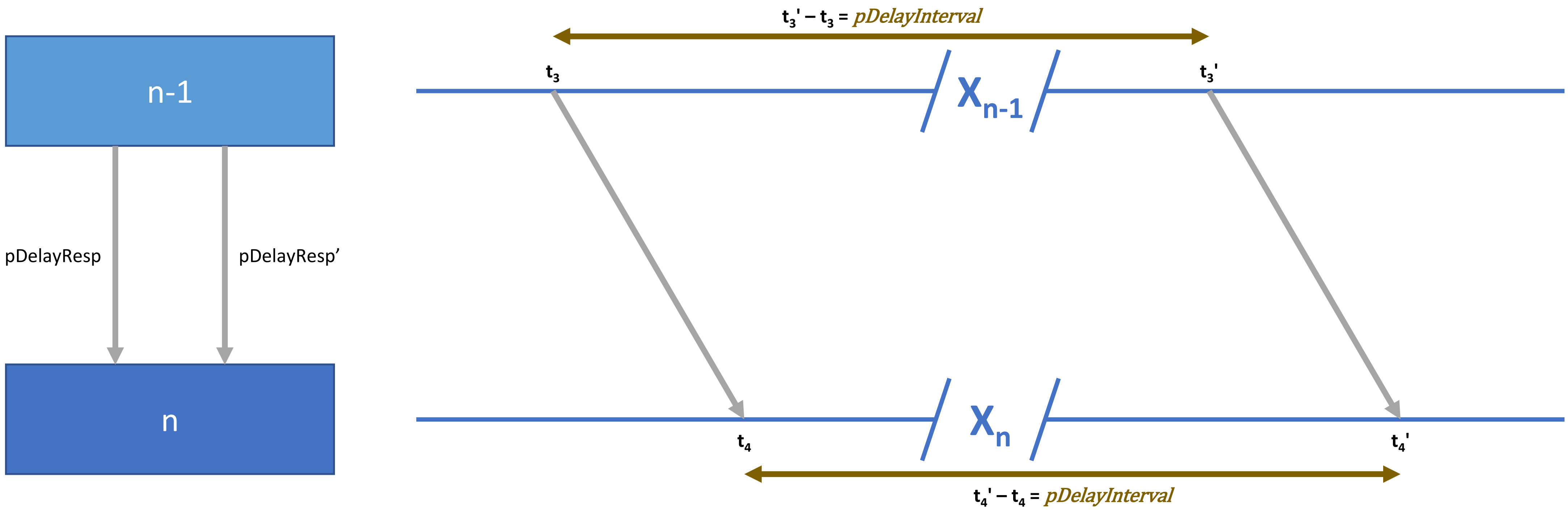
Input Correction Factor

Error	Default Value	Unit
$meanLinkDelay_{errorCorrection}$	0	Value (0-1)
$driftRate_{errorCorrection}$	0	Value (0-1)
$pDelayRespSync_{correction}$	0	Value (0-1)
$mNRRsmoothingN$	1	Number (1+)

All formulae in this analysis are ultimately composed of one of these inputs:

Input Errors (Clock Drift or Timestamp Errors)
Input Parameters
Input Correction Factors

Measured Neighbor Rate Ratio (mNRR)

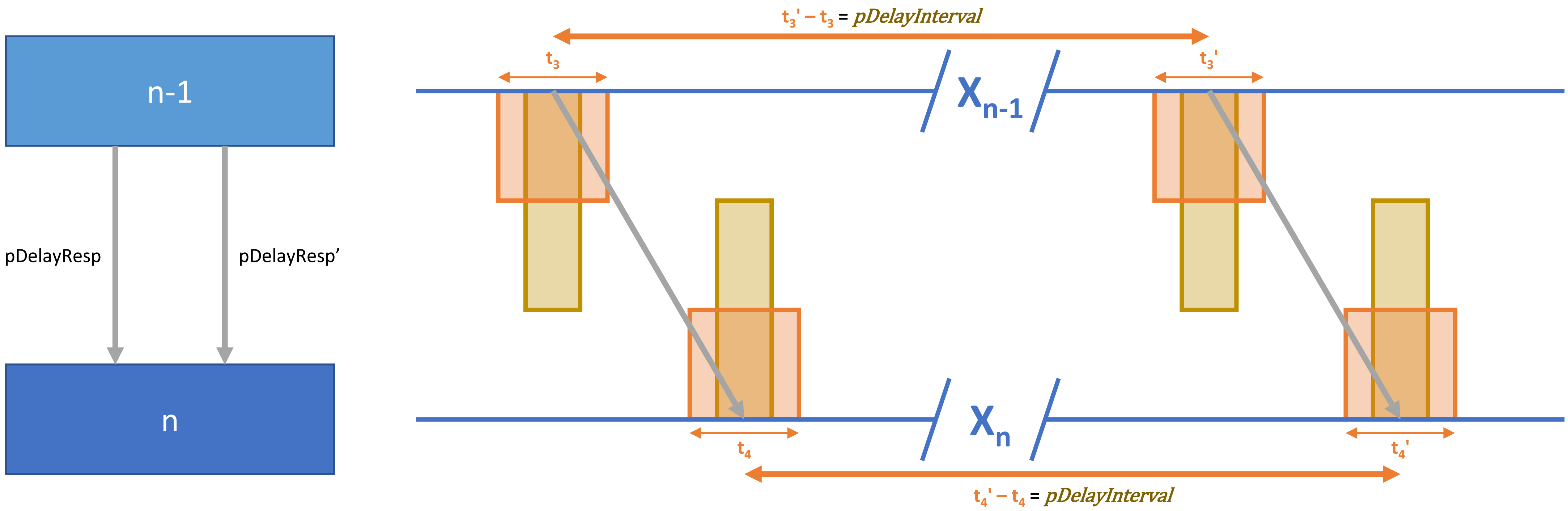


$$mNRR = \left(\frac{(t_4{}' - t_4)}{(t_3{}' - t_3)}\right)$$

$$mNRR_{error} = mNRRe_{rrorTS} + mNRRe_{rrorCD}$$

ppm

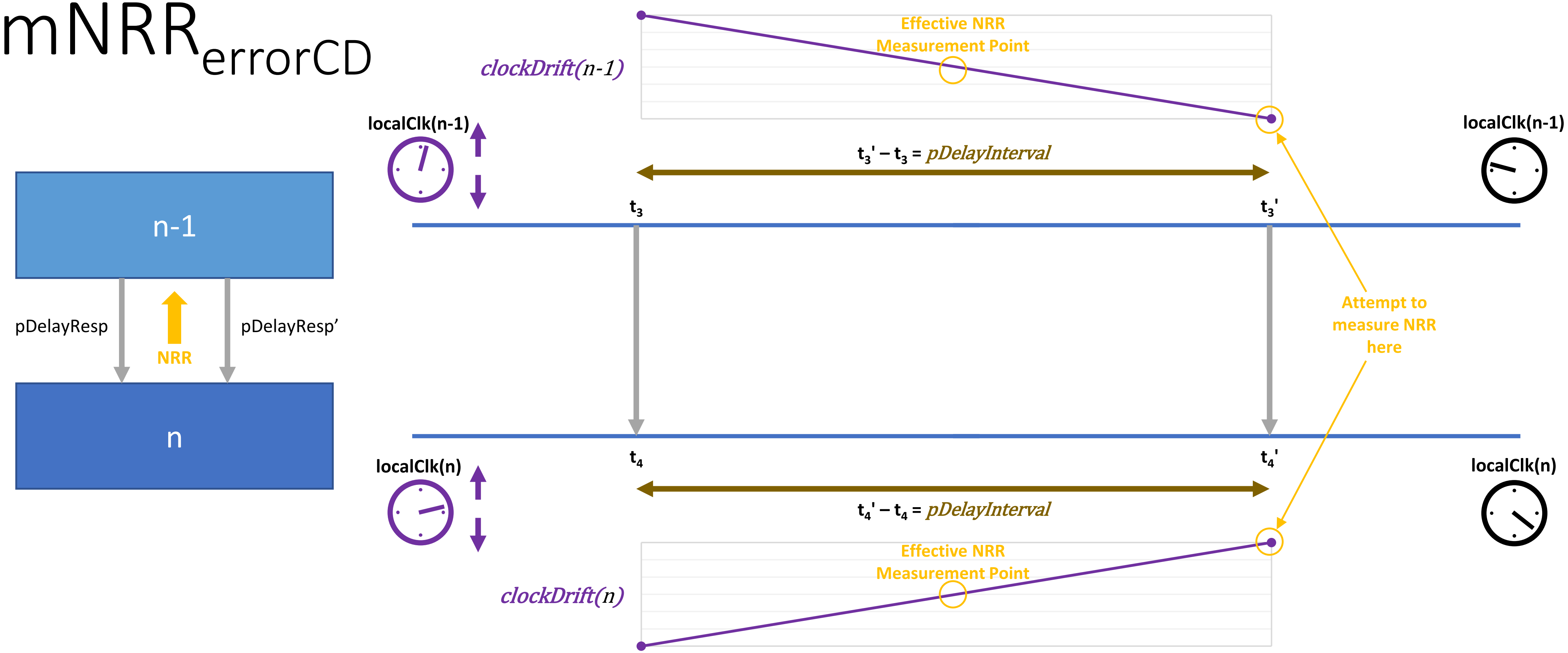
$mNRR_{errorTS}$



$$mNRR_{errorTS} = \left(\frac{(t_{4PDerror} - t_{4PDerrorPrevious}) - (t_{3PDerror} - t_{3PDerrorPrevious})}{\boldsymbol{pDelayInterval} \times \boldsymbol{mNRRsmoothingN}}\right)$$

ppm

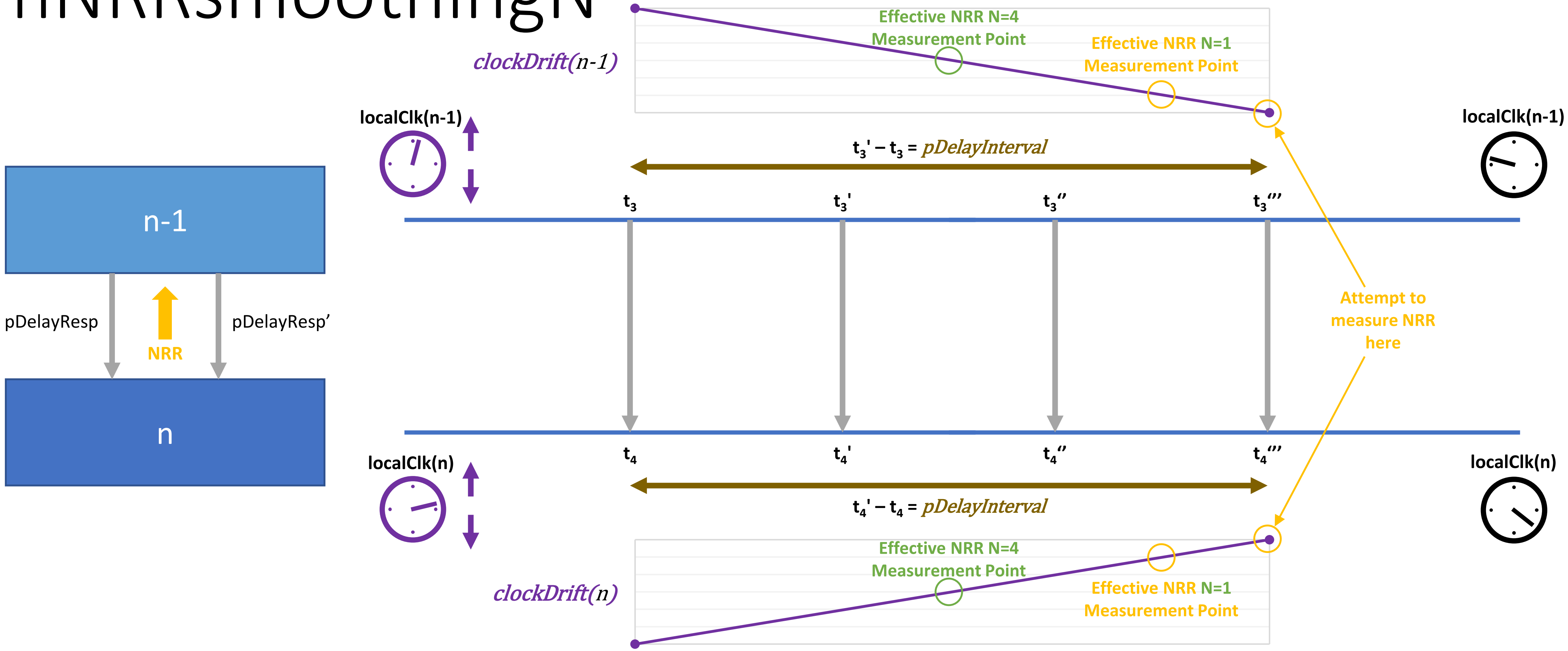
$mNRR_{errorCD}$



$$mNRR_{errorDrift}(n) = (1 - \boldsymbol{driftRate}_{errorCorrection}) \times \boldsymbol{mNRRsmoothingN} \times \left(\frac{\boldsymbol{pDelayInterval}}{2 \times 10^3}\right)(\boldsymbol{clockDrift}(n-1) - \boldsymbol{clockDrift}(n))$$

ppm

mNRRsmoothingN



$$mNRR_{errorDrift}(n) = (1 - driftRate_{errorCorrection}) \times mNRRsmoothingN \times \left(\frac{pDelayInterval}{2 \times 10^3}\right)\left(clockDrift(n-1) - clockDrift(n)\right)$$

ppm

Formulae –mNRR$_{error}$

$$mNRR_{error} = mNRR_{errorCD} + mNRRe_{rrorTS}$$ **ppm**

$$mNRR_{errorCD}(n) = (1 - \boldsymbol{driftRate}_{errorCorrection}) \times \boldsymbol{mNRRsmoothingN} \times \left(\frac{\boldsymbol{pDelayInterval}}{\boldsymbol{2 \times 10^3}}\right)(\boldsymbol{clockDrift}(n-1) - \boldsymbol{clockDrift}(n))$$ **ppm**

$$mNRR_{errorTS} = \left(\frac{(t_{4PDerror} - t_{4PDerrorPrevious}) - (t_{3PDerror} - t_{3PDerrorPrevious})}{\boldsymbol{pDelayInterval} \times \boldsymbol{mNRRsmoothingN}}\right)$$ **ppm**

$$t_{3PDerrorPrevious} = \boldsymbol{TSGE_{TX}} + \boldsymbol{DTSE_{TX}} \qquad t_{4PDerrorPrevious} = \boldsymbol{TSGE_{RX}} + \boldsymbol{DTSE_{RX}}$$ **ns**

The factors $t_{3PDerror}$ and $t_{4PDerror}$ are the same as in the $pDelay_{error}$ calculation.
(Same value. No need for new random numbers, as the same pDelayResp message is used to measure NRR.)

Observations of $\text{mNRR}_{\text{error}}$ Behaviour

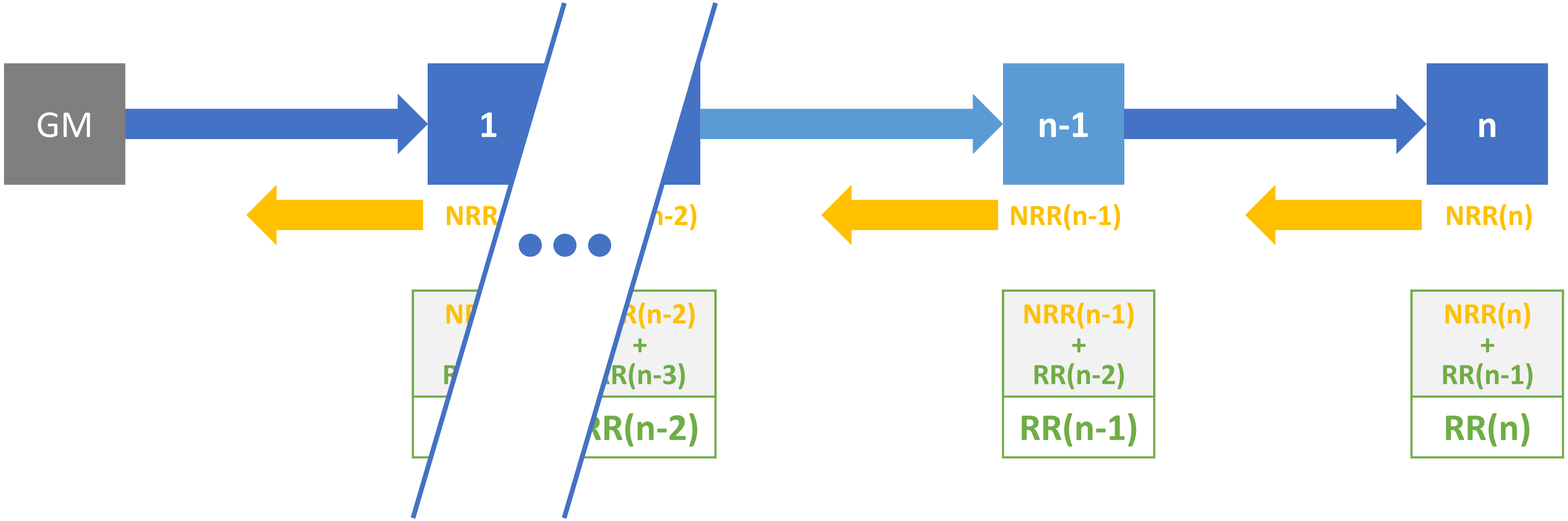
- Errors in mNRR do not accumulate along the chain of nodes
- An mNRR error due to clock drift at one node will tend to be reversed at the next node.
  - This does not apply for mNRR errors due to clock drift at the GM

$$mNRR_{errorCD}(n) = \left(\frac{\boldsymbol{pDelayInterval}}{\mathbf{2 \times 10^3}}\right)\left(\boldsymbol{clockDrift}(n-1) - \boldsymbol{clockDrift}(n)\right)$$

$$mNRR_{errorCD}(n+1) = \left(\frac{\boldsymbol{pDelayInterval}}{\mathbf{2 \times 10^3}}\right)\left(\boldsymbol{clockDrift}(n) - \boldsymbol{clockDrift}(n+1)\right)$$

- mNRR errors due to Timestamp errors are independent of each other. DTE errors at one node due to this component will not tend to be reversed at the next node.
- mNRR errors due to clock drift are modelled as a combination of two uniform distributions (clock drifts at n and n-1) and will have a simple probability distribution
  - Errors due to Timestamp errors are modelled as combinations of eight uniform distributions and will be more complex.

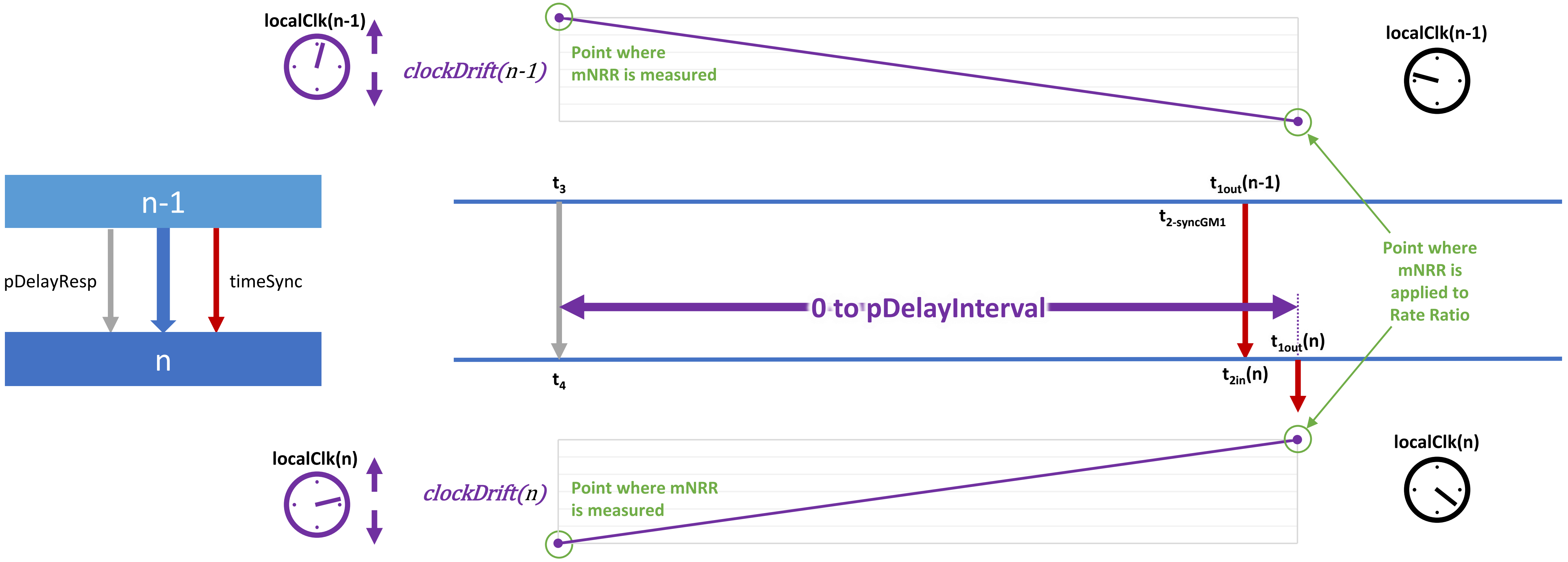
Rate Ratio



$$RR(n) = RR(n-1) + mNRR(n)$$

$$RR_{error}(n) = RR_{error}(n-1) + mNRR_{error}(n) + RRe_{rrorCD}(n)$$

$RR_{errorCD}$



$$RR_{errorCD}(n) = (1 - \boldsymbol{driftRate}_{errorCorrection}) \times \frac{delay_{mNRR\_Sync}}{10^3}(\boldsymbol{clockDrift}(n-1) - \boldsymbol{clockDrift}(n))$$ **ppm**

Formulae – RR_{error}

$$RR_{error}(n) = RR_{error}(n-1) + mNRRe_{rror}(n) + RR_{errorCD}(n)$$ **ppm**

$$RR_{errorCD}(n) = (1 - \mathbf{driftRate}_{\mathbf{errorCorrection}}) \times \frac{delay_{mNRR\_Sync}}{10^3}\left(\mathbf{clockDrift}(n-1) - \mathbf{clockDrift}(n)\right)$$ **ppm**

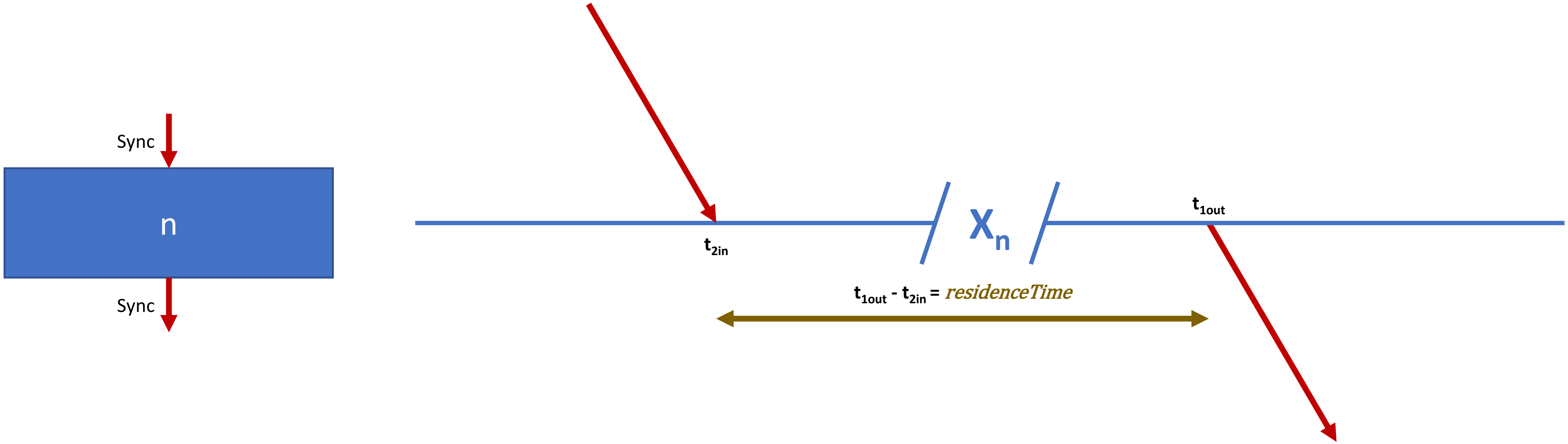
$$delay_{mNRR\_Sync} \sim U(0, (1 - \mathbf{pDelayRespSync}_{\mathbf{correction}})\mathbf{pDelayInterval})$$ **ms**

Does not include any effect of changing clock drift (ppm/s²) during Residence Time. See speaker notes in RR_{error} section for details.

Observations of RR_{error} Behaviour

- $mNRR_{error}$ feeds directly into RR_{error} where they accumulate
  - Since $mNRR_{error}$ due to clock drift at one node tends to reverse at the next node, the component of RR_{error} due to this will tend not to increase along a chain of devices.
    - This does not apply for error due to GM clock drift as it only appears in $mNRR_{error}$ at the first hop.
  - $mNRR_{errors}$ due to timestamp errors are much less likely to cancel out so RR_{error} from this source is more likely to increase along a chain of devices.
- RR_{error} due to clock drift during the delay between measurement of mNRR and when it is applied to Rate Ratio could cancel out…but only if the delay one node is the same as at the next…which is unlikely.
  - $RR_{errorCD}$ is therefore much more likely to increase along a chain of devices than the clock drift component of $mNRR_{error}$
- RR_{error} due to clock drift can be reduced by decreasing pDelayInterval; reduces range of $delay_{mNRRSync}$
  - Similar effect if pDelay messaging can be aligned to occur just before Sync message; modelled using $pDelayRespSync_{correction}$
  - Note: see earlier in this presentation for the effect reducing pDelayInterval has on $mNRR_{error}$

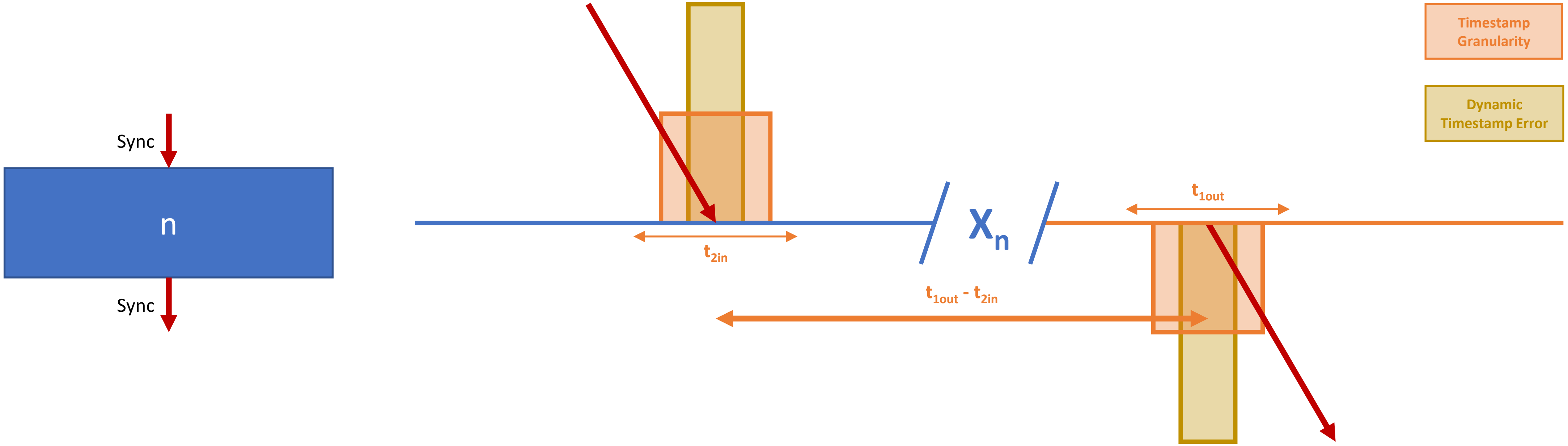
Residence Time



$$residenceTime = RR(t_{1out} - t_{2in})$$ **ns**

$$residenceTime_{error} = residenceTimee_{rrorTS} + residenceTimee_{rrorRR}$$ **ns**

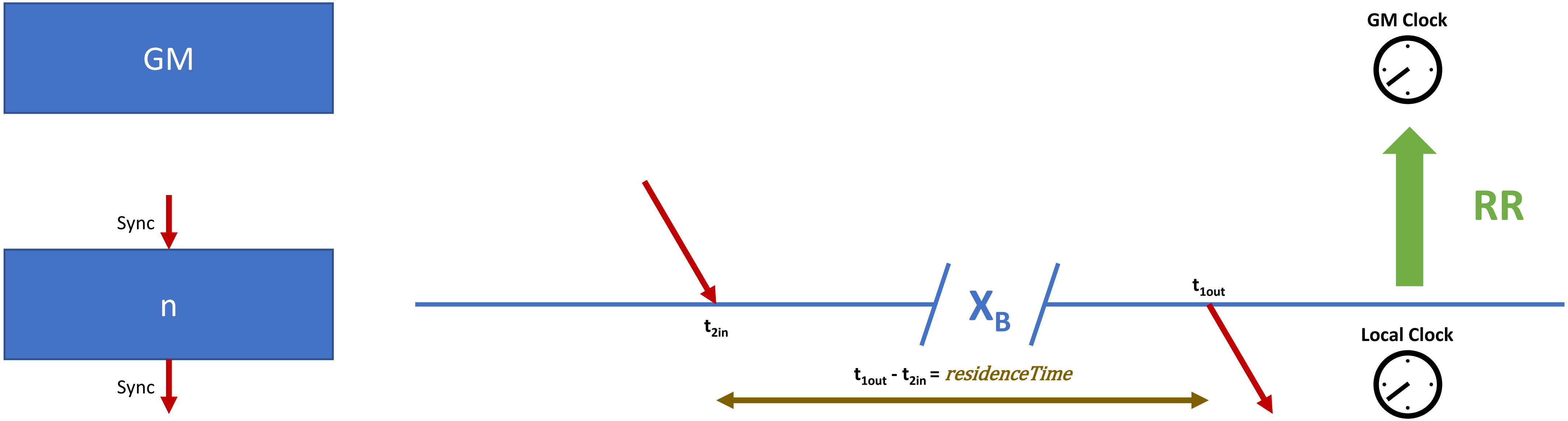
$RT_{errorTS}$



$$residenceTime_{errorTS} = t_{1Serror} - t_{2Serror}$$

ns

$residenceTime_{errorRR}$



$$residenceTime_{errorRR} = \frac{RR_{error}}{10^6}(\boldsymbol{residenceTime} \times 10^6) \quad \textbf{ns}$$

$$= RRe_{rror} \times \boldsymbol{residenceTime} \quad \textbf{ns}$$

Formulae – residenceTime$_{error}$

$$residenceTime_{error} = residenceTimee_{rrorTS} + residenceTimee_{rrorRR}$$ **ns**

$$residenceTime_{errorTS} = t_{1Serror} - t_{2Serror}$$ **ns**

$$t_{2Serror} = \boldsymbol{TSGE_{RX}} + \boldsymbol{DTSE_{RX}} \qquad t_{1Serror} = \boldsymbol{TSGE_{TX}} + \boldsymbol{DTSE_{TX}}$$ **ns**

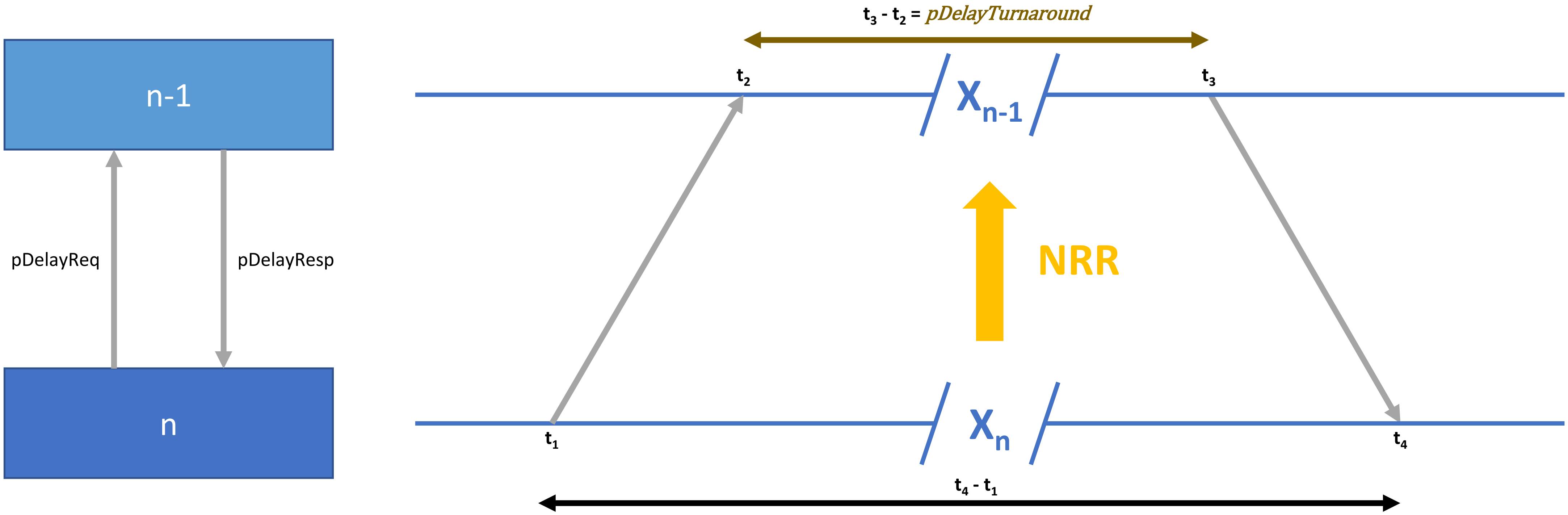
$$residenceTime_{errorRR} = \frac{RR_{error}}{10^6}(\boldsymbol{residenceTime} \times 10^6 + residenceTime_{errorTS})$$ **ns**

$$= RRe_{rror} \times \left(\boldsymbol{residenceTime} + \frac{residenceTime_{errorTS}}{10^6}\right)$$ **ns**

Observations of $\text{residenceTime}_{\text{error}}$ Behaviour

- $\text{residenceTime}_{\text{error}}$ due to timestamp error is independent of Residence Time.
  - Reducing Residence Time will have no effect on this source of error.
- $\text{residenceTime}_{\text{error}}$ due to RR_{error} is proportional to Residence Time
  - Reducing Residence Time can reduce this source of error.
- Since larger RR_{error} is more likely further along a chain of devices, the amount of Residence Time Error at each node is also likely to increase.
  - For example: the component of DTE due to Residence Time Error after 100 hops is more likely to be larger from errors in nodes 51-100 than from nodes 1-50.

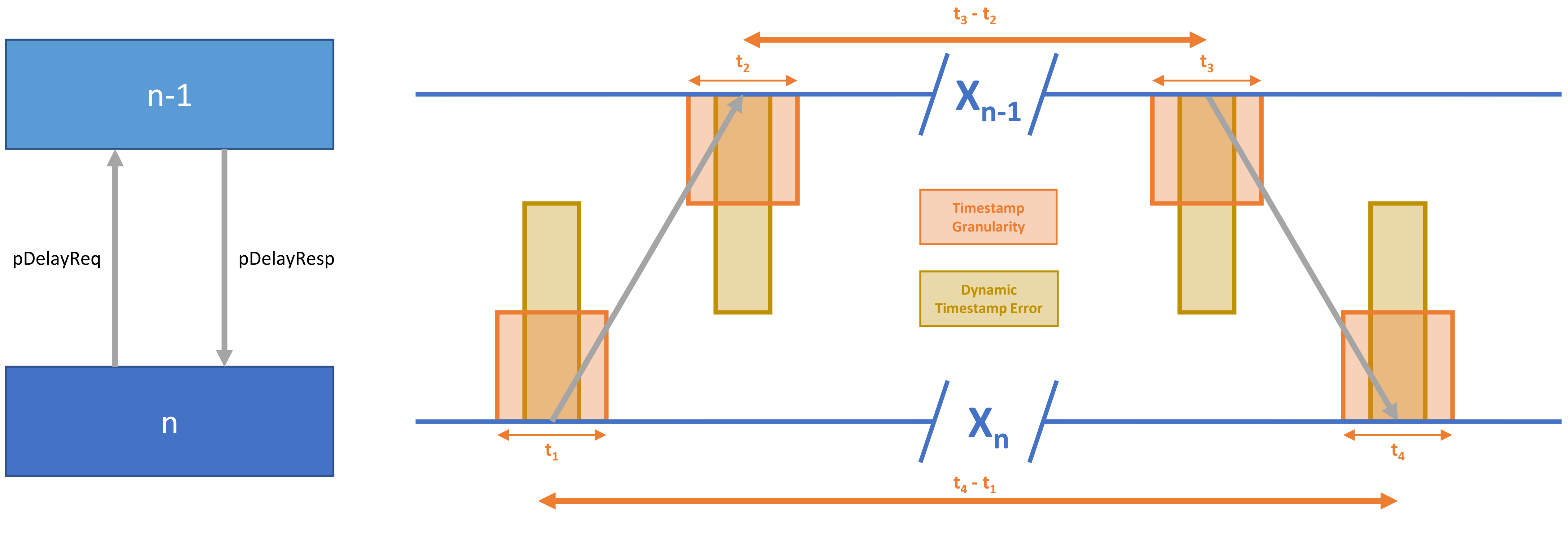
meanLinkDelay



$$meanLinkDelay = RR\left(\frac{(t_4 - t_1) - NRR(t_3 - t_2)}{2}\right)$$ **ns**

$$meanLinkDelay_{error} = (1 - \boldsymbol{meanLinkDelay_{errorCorrection}})(pDelay_{errorTS} + pDelay_{errorNRR} + pDelay_{errorRR})$$ **ns**

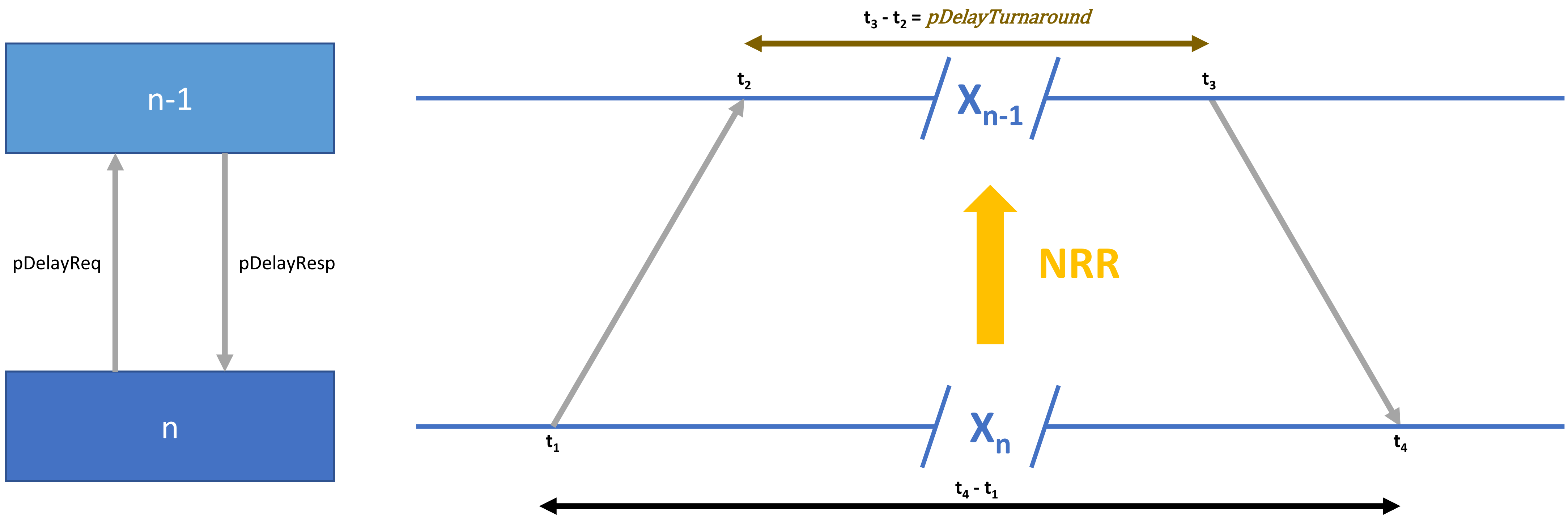
$pDelay_{errorTS}$



$$pDelay_{errorTS} = \frac{(t_{4PDerror} - t_{1PDerror}) - (t_{3PDerror} - t_{2PDerror})}{2}$$

ns

$\text{pDelay}_{\text{errorNRR}}$



$$pDelay_{errorNRR} = mNRR_{error}\left(\frac{pDelayTurnaround}{2}\right)$$

ns

Formulae – meanLinkDelay$_{error}$

$$meanLinkDelay_{error} = (1 - \boldsymbol{meanLinkDelay_{errorCorrection}})(pDelay_{errorTS} + pDelay_{errorNRR} + pDelay_{errorRR}) \quad \mathbf{ns}$$

$$pDelay_{errorTS} = \frac{(t_{4PDerror} - t_{1PDerror}) - (t_{3PDerror} - t_{2PDerror})}{2} \quad \mathbf{ns}$$

$$t_{1PDerror} = \boldsymbol{TSGE_{TX}} + \boldsymbol{DTSE_{TX}} \qquad t_{2PDerror} = \boldsymbol{TSGE_{RX}} + \boldsymbol{DTSE_{RX}} \quad \mathbf{ns}$$

$$t_{3PDerror} = \boldsymbol{TSGE_{TX}} + \boldsymbol{DTSE_{TX}} \qquad t_{4PDerror} = \boldsymbol{TSGE_{RX}} + \boldsymbol{DTSE_{RX}} \quad \mathbf{ns}$$

$$pDelay_{errorNRR} = \frac{mNRR_{error}}{10^6}\left(\frac{\boldsymbol{pDelayTurnaround} \times 10^6}{2}\right) \quad \mathbf{ns}$$

$$= mNRR_{error}\left(\frac{\boldsymbol{pDelayTurnaround}}{2}\right) \quad \mathbf{ns}$$

$$pDelay_{errorRR} = \frac{RR_{error}}{10^6}(\boldsymbol{pDelay} + (1 - \boldsymbol{pDelay_{errorCorrection}})(pDelay_{errorTS} + pDelaye_{rrorNRR})) \quad \mathbf{ns}$$

Observations of meanLinkDelay$_{error}$ Behaviour

- meanLinkDelay$_{error}$ due to timestamp error is independent of pDelayTurnaround.
  - Reducing pDelayTurnaround will have no effect on this source of error.
- meanLinkDelay$_{error}$ due to mNRR$_{error}$ is proportional to pDelayTurnaroud
  - Reducing pDelayTurnaround can reduce this source of error.
- The actual Link Delay is not a significant source of error
  - Only needs to be included as part of pDelay$_{errorRR}$…which is small enough to ignore for the purposes of this analysis

Formulae – Top Level

$$DTE(x) = \sum_{n=1}^{x} (meanLinkDelay_{error}(n) + residenceTimeerror(n))$$ **ns**

$$meanLinkDelay_{error} = (1 - \boldsymbol{meanLinkDelay_{errorCorrection}})(pDelay_{errorTS} + pDelay_{errorNRR} + pDelay_{errorRR})$$ **ns**

$$residenceTime_{error} = residenceTimee_{rrorTS} + residenceTimee_{rrorRR}$$ **ns**

The current analysis does not include factors shown in grey.
***x* is number of hops.**

Special Cases

$$clockDrift(0) = clockDrift_{GM}$$

For the first hop ($n = 1$), $n - 1 = \mathbf{0}$, i.e. the first device in chain, which is the GM.

$$residenceTime_{error}(x) = 0$$

For the last hop ($n = x$), the Sync message is not passed on so there is no Residence Time Error.

Analysis Carried Out Using R & RStudio

- R is available here: https://www.r-project.org/
  - Open source license: various, but mostly GNU GPL v2 and GPL v3
- RStudio is available here: https://www.rstudio.com/products/rstudio/download/
  - Open source license: GNU Affero GPL v3
- Model uses write.csv and write.table functions, which are part of R.Utils package
  - Install in RStudio by typing…

    ```
    install.packages("R.utils")
    ```

    …in the console window.

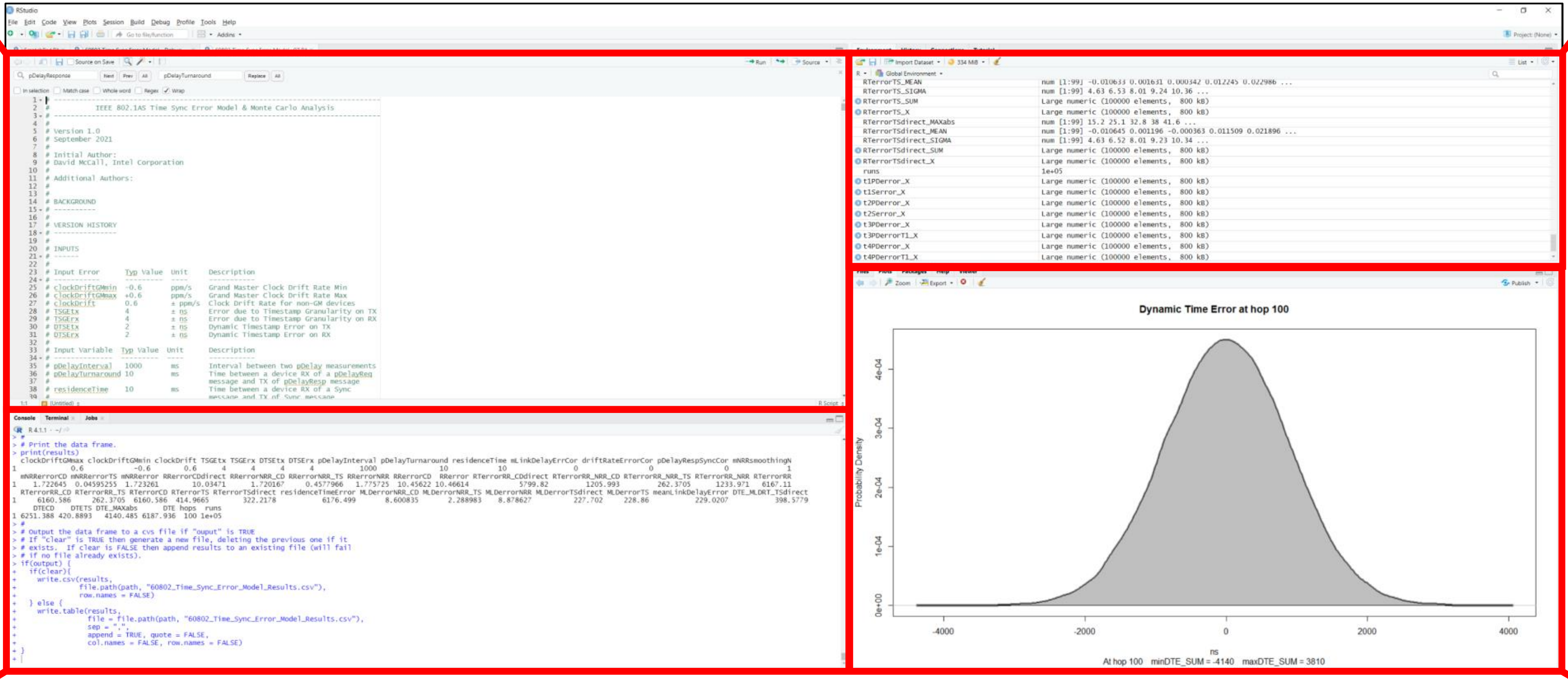
Availability of Analysis Script

- Intention is to make the script code available under an open source license
  - Probably BSD 3-Clause
  - Other licenses are an option; feedback welcome.
  - Timing TBD. Target is before the end of the year.
- Current plan is to simply make the script code available, not to set up an open source project (e.g. GitHub)
  - I someone wants to

RStudio

Script

Environment Variables



Console

Plots

RStudio Script Code Summary

- Configuration (Output? Hops? Runs? More charts? Seed value?)
- Inputs (see above)
- Initialize tracking vectors
- Hop 1
  - Calculate main values that contribute to DTE
  - Also calculate values of error components for analysis
  - Calculate MAXabs, MEAN and SIGMA for all main & component values and record in tracking vectors
- Loop: Hops 2+
  - Mostly the same as Hop 1, but errors accumulate where appropriate.
- Plot Charts

Demo

Lenovo Thinkpad T480

Intel(R) Core(TM) i5-8350U CPU @ 1.70GHz   1.90 GHz

16GB RAM

(While running Webex & background corporate apps)

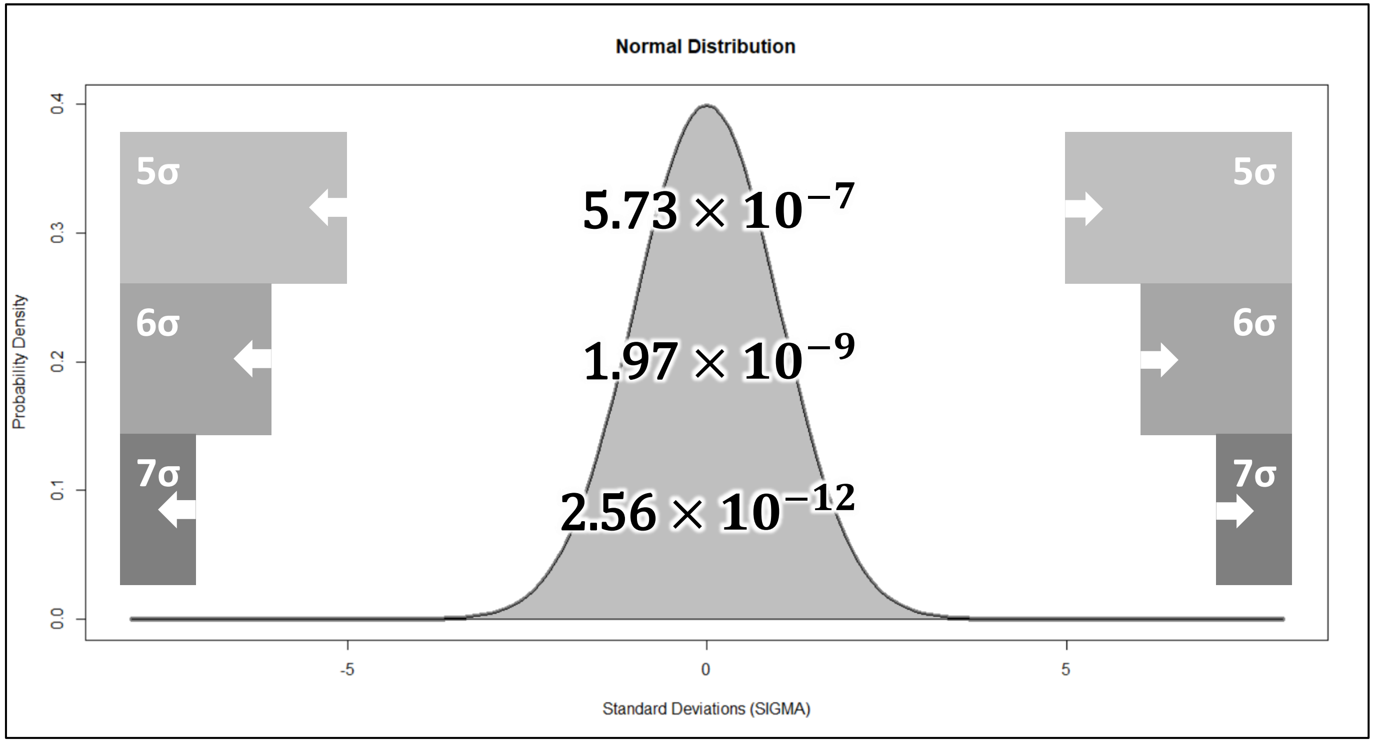
Results

- Monte Carlo analysis of errors allows for many "runs" in very little time.
  - 100hops & 100,000 runs in <30 seconds
  - Calculating hops takes <15 seconds; rest of time spent generating plots
  - Add approx. 10 seconds to generate additional detailed plots
- Next step: determine if the results are useful.
  - Compare to previous Time Series Simulation

Comparison with Time Series Simulations

Values to Compare Against Time Series max|DTE|

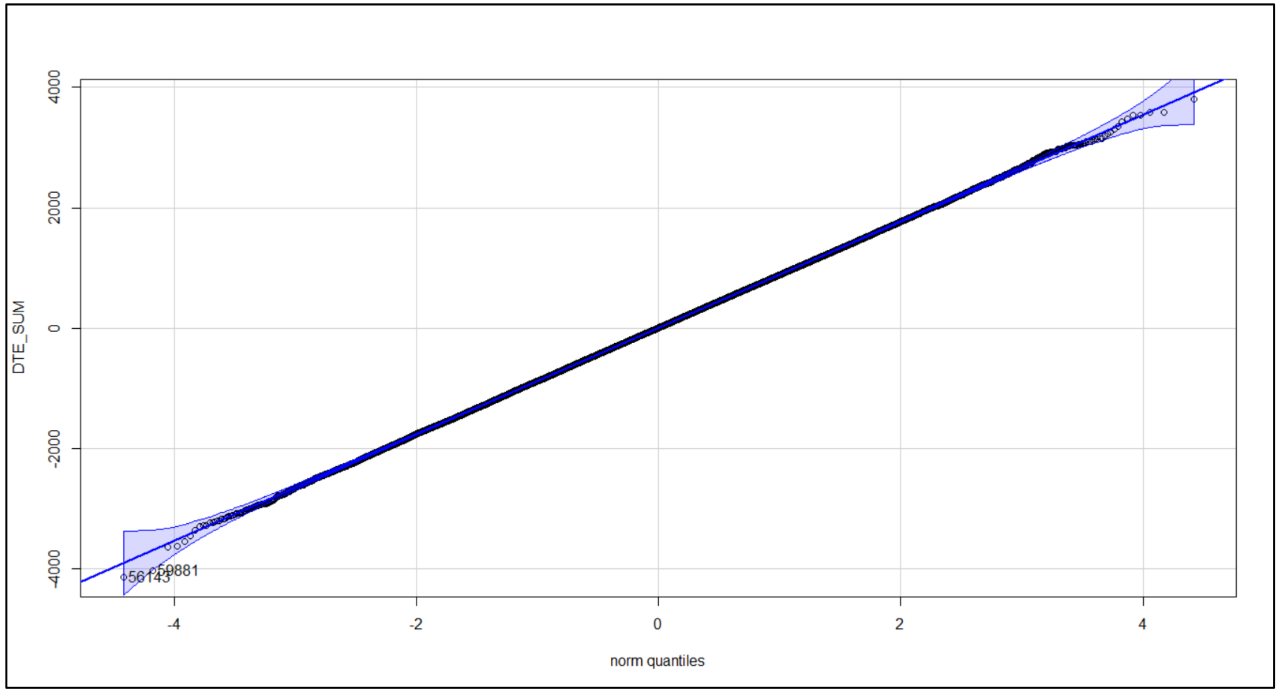
- Maximum Absolute Value of Dynamic Time Error (**max|DTE|**)
- A multiple of SIGMA (σ) for DTE at hop 100, based on probability of exceeding it…*



*** Only valid if data forms a normal (Gaussian) distribution**

Dynamic Time Error – Normal Distribution?

- Use of SIGMA to calculate probability of exceeding a value is only valid if the data forms a normal distribution.
- Quantile-Quantile Plot of DTE at hop 100 (100,000 runs)
  - 802.1AS default parameters
  - Clock Drift: ±0.6 ppm/s
  - TSGE & DTSE: ±4ns
  - Data should lie along a straight line (with some variance at extremes expected)
- Result: **YES**, data forms normal distribution.



Which SIGMA?

- Probability of **|DTE|** > ?-SIGMA over a period of time?

125ms Sync Interval	
8	per second
480	per minute
28,800	per hour
691,200	per day
252,460,800	per year

? x SIGMA	Average Time Before Exceeding
5σ	2.5 days
6σ	2 years
7σ	1,548 years

- Conclusion: use 7σ
  - Revisit if we can’t achieve goals using 7σ
  - Lower multiple (6σ?) would be appropriate for constant time error
  - Lower multiple for dynamic time error might be justified due to combination with constant time error

Enough runs for measurement of SIGMA?

- Increasing number of runs; track max|DTE| and 7
  - Same seed value…but structure of model means first 100 runs when analysing 1,000 runs are not the same as when analysing 100 runs.

Runs	max|DTE| (ns)	7σ of DTE (ns at hop 100)
100	2,400	6,230
1,000	3,710	6,380
10,000	3,610	6,100
100,000	4,140	6,190
1,000,000	4,380	6,190

- 100,000 runs is a good compromise between speed and accuracy
  - Give up very little in terms of accuracy & runs in

Custom Script to Match Time Series Simulation

- The Time Series Simulation includes two factors that must be specifically modelled differently than the main model.
- Dynamic Time Error is +8ns or -8ns with 50% probability of either.
  - Not realistic. Not recommended for main model.
- mNRR is determined by caculating NRR using most recent and N^{th} prior pDelayResponse messages (included in the main model via mNRRsmoothingN) **and then** taking median value of previous N calculations (not included in the main model).
  - N is an odd number; values of 11 and 7 have been used.
  - Assuming clocks drift linearly, this will result in the $\frac{N+1}{2}$ previous value being used; e.g. if N=11, 6^{th} previous value.
    - This results in an additional clock drift between effective measurement and mNRR of $\left(\frac{N+1}{2} - 1\right) = \left(\frac{N-1}{2}\right)$
  - This has no effect on $mNRR_{errorTS}$, but increases $mNRR_{errorCD}$ by an additional factor of N-1.
    - Modelled by changing the effect of mNRRsmoothingN on mNRRerrorCD from…

```
* mNRRsmoothingN
```

…to…

```
* ((mNRRsmoothingN * 2) – 1)
```

  - Taking median of previous N calculations only has a negative effect; it only adds an additional source of error and mitigates no existing error. Therefore, not recommended for main model (or use in practical systems).

Comparison with Time Series Simulation

Case	residenceTime (ms)	TSGE (±ns)	DTSE (±ns)	Temp Factor	mNRR Smoothing	Time Series max|DTE| @100 hops (ns)	% Diff 100% Temp → 10% Temp	Monte Carlo max|DTE| @100 hops (ns)	Monte Carlo max7sigma @100 hops (ns)	TS → MC max|DTE|	TS → MC 7Sigma
Single Replication											
1	1	4	8	100%	No	2717		1715	2729	-36.9%	0.4%
2	1	2	8	100%	No	2891		1666	2649	-42.4%	-8.4%
3	4	4	8	100%	No	6023		5967	8983	-0.9%	49.1%
4	4	2	8	100%	No	6155		5848	8722	-5.0%	41.7%
5	10	4	8	100%	No	13618		14470	21722	6.3%	59.5%
6	10	2	8	100%	No	13252		14212	21089	7.2%	59.1%
7	1	4	8	10%	No	3058	13%	1715	2729	-43.9%	-10.8%
8	1	2	8	10%	No	2777	-4%	1666	2649	-40.0%	-4.6%
9	4	4	8	10%	No	6341	5%	5967	8983	-5.9%	41.7%
10	4	2	8	10%	No	6045	-2%	5848	8722	-3.3%	44.3%
11	10	4	8	10%	No	13942	2%	14470	21722	3.8%	55.8%
12	10	2	8	10%	No	12766	-4%	14212	21089	11.3%	65.2%
13	1	0	0	100%	No	32		13	19	-59.4%	-40.6%
14	1	0	8	100%	No	2841		1860	2630	-34.5%	-7.4%
15	1	4	0	100%	No	733		458	759	-37.5%	3.5%
16	1	4	8	100%	Yes	579		711	1080	22.8%	86.5%
17	1	2	8	100%	Yes	547		670	1049	22.5%	91.7%
18	4	4	8	100%	Yes	658		962	1466	46.2%	122.8%
19	4	2	8	100%	Yes	627		919	1426	46.6%	127.4%
20	10	4	8	100%	Yes	909		1653	2555	81.8%	181.1%
21	10	2	8	100%	Yes	856		1607	2489	87.7%	190.8%
22	1	4	8	10%	Yes	584	1%	711	1080	21.8%	84.9%
23	1	2	8	10%	Yes	601	10%	670	1049	11.5%	74.5%
24	4	4	8	10%	Yes	690	5%	962	1466	39.4%	112.5%
25	4	2	8	10%	Yes	787	26%	919	1426	16.8%	81.2%
26	10	4	8	10%	Yes	1441	59%	1653	2555	14.7%	77.3%
27	10	2	8	10%	Yes	1376	61%	1607	2489	16.8%	80.9%
300 Replications											
16 (28)	1	4	8	100%	Yes	815	40.8%	711	1080	-12.7%	32.5%
18 (29)	4	4	8	100%	Yes	1121	78.8%	962	1466	-14.2%	30.8%
22 (30)	1	4	8	10%	Yes	784	34.2%	711	1080	-9.3%	37.8%
27 (31)	10	2	8	10%	Yes	2202	60.0%	1607	2489	-27.0%	13.0%

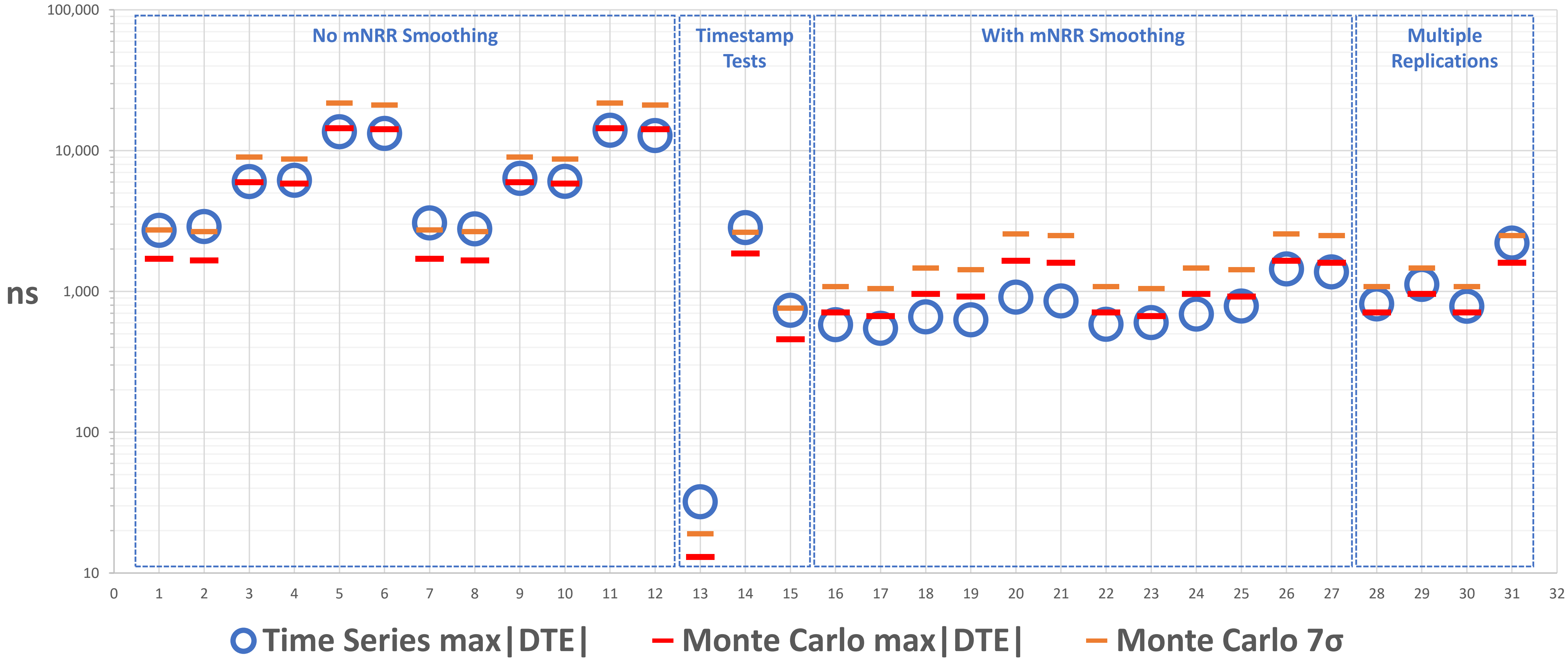
No mNRR Smoothing

Timestamp Tests

With mNRR Smoothing

Multiple Replications

Comparison with Time Series Simulation



Comparison with Time Series Simulation

- Monte Carlo results broadly align with Time Series results
- Where results deviate the most, they are still usefully close and the Monte Carlo results move in the same direction and by similar amounts as the Time Series results when input parameters change
- The closest matches are between Monte Carlo results and results from multiple replications of the Time Series simulation
- Monte Carlo analysis is definitely good enough for investigating approaches to minimising DTE

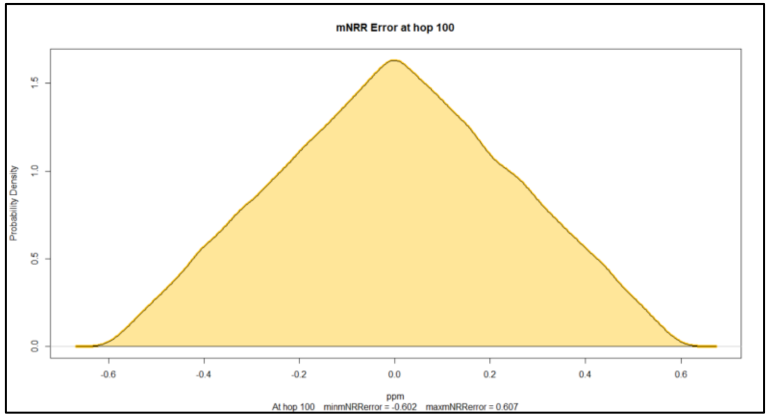
Error Analysis

Graphical Representations

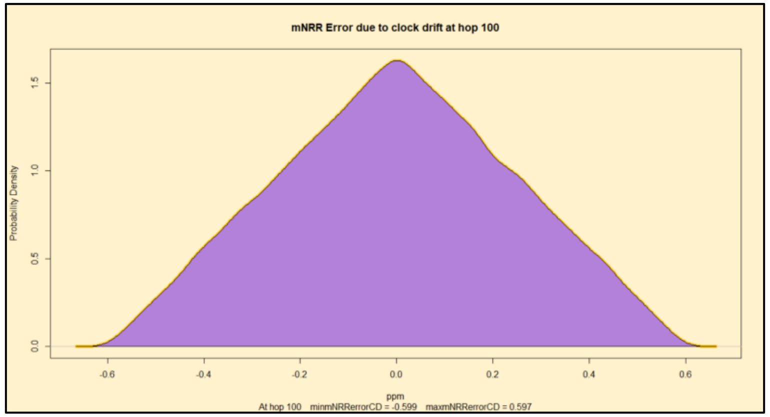
- The Monte Carlo Analysis can generate **a lot** of data
- Prioritising what to data to look at and how to analyse it will help reach useful conclusions quickly
- With that in mind, here is an overview of the range of data…
  - Main Model
    (not the Time Series Match version)

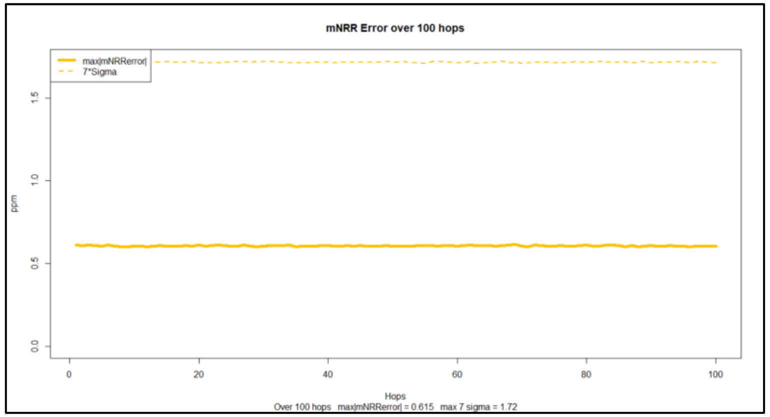
Input Errors		
GM Clock Drift Max	+0.6	ppm
GM Clock Drift Min	-0.6	ppm
Clock Drift (non-GM)	0.6	±ppm
Timestamp Granularity TX	4	±ns
Timestamp Granularity RX	4	±ns
Dynamic Time Stamp Error TX	4	±ns
Dynamic Time Stamp Error RX	4	±ns
Input Parameters		
pDelay Interval	1000	ms
pDelay Response Time	10	ms
residenceTime	10	ms
Correction Factors		
Mean Link Delay	0	%
Drift Rate	0	%
pDelayResponse → Sync	0	%
mNRR Smoothing	1	
Configuration		
Hops	100	
Runs	100,000	

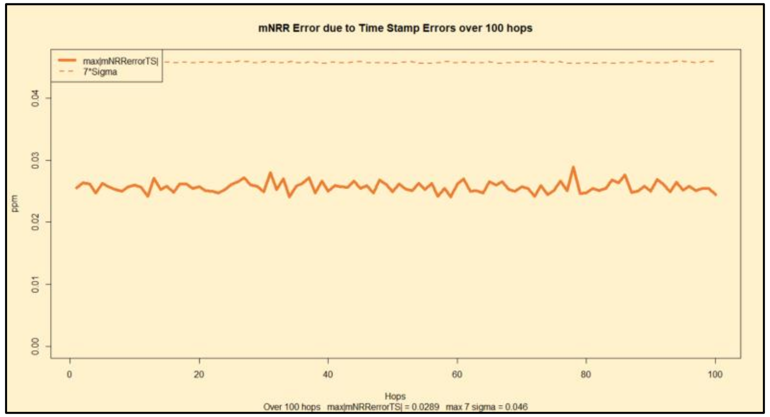
mNRR Error

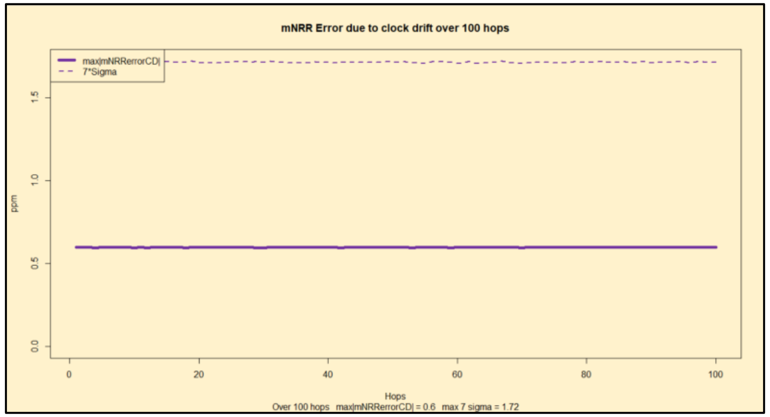

mNRR Error at hop 100
Probability Density
0.0
0.5
1.0
1.5
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
ppm
At hop 100 minmNRRerror = -0.602 maxmNRRerror = 0.607


mNRR Error due to Time Stamp Errors at hop 100
Probability Density
0
10
20
30
40
50
60
-0.02
-0.01
0.00
0.01
0.02
ppm
At hop 100 minmNRRerrorTS = -0.0244 maxmNRRerrorTS = 0.0239

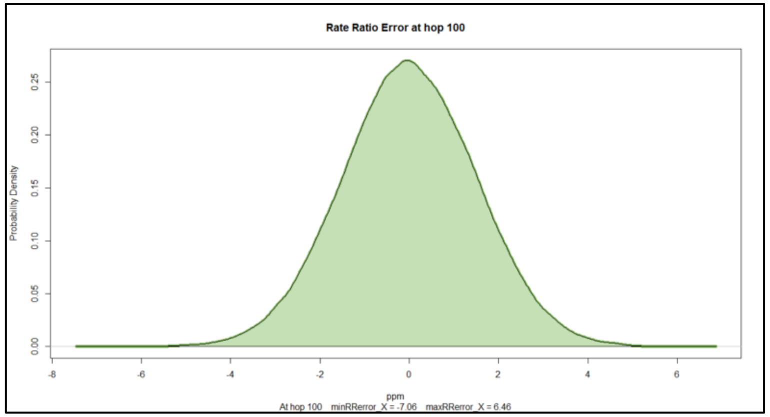

mNRR Error due to clock drift at hop 100
Probability Density
0.0
0.5
1.0
1.5
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
ppm
At hop 100 minmNRRerrorCD = -0.599 maxmNRRerrorCD = 0.597

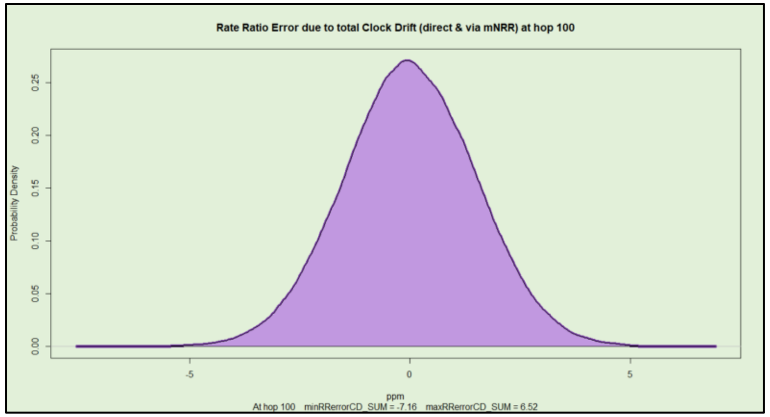

mNRR Error over 100 hops
max|mNRRerror|
7*Sigma
ppm
0.0
0.5
1.0
1.5
0
20
40
60
80
100
Hops
Over 100 hops max|mNRRerror| = 0.615 max 7 sigma = 1.72

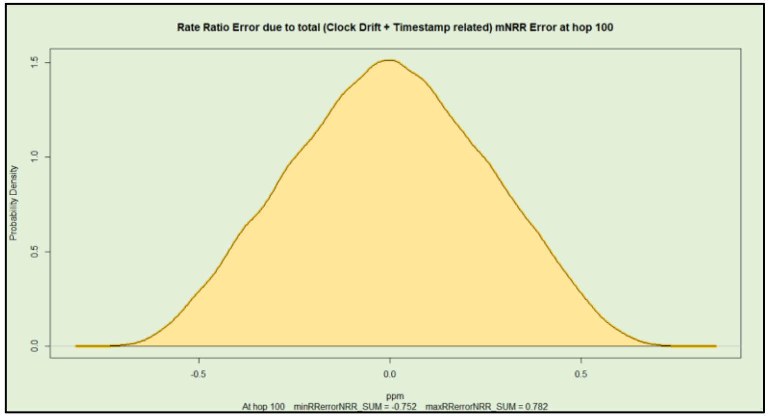

mNRR Error due to Time Stamp Errors over 100 hops
max|mNRRerrorTS|
7*Sigma
ppm
0.00
0.01
0.02
0.03
0.04
0
20
40
60
80
100
Hops
Over 100 hops max|mNRRerrorTS| = 0.0289 max 7 sigma = 0.046

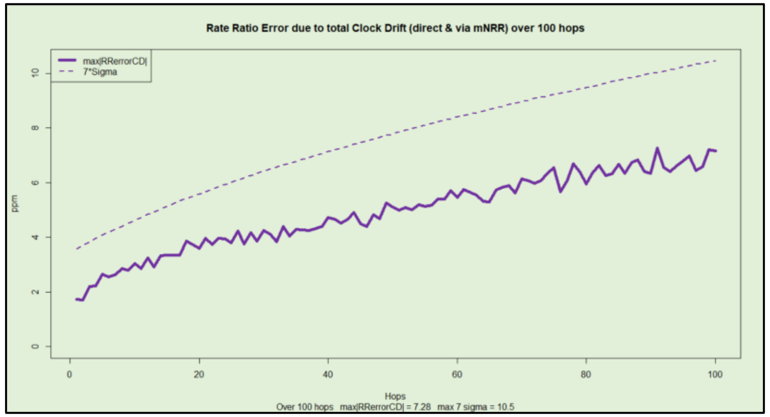

mNRR Error due to clock drift over 100 hops
max|mNRRerrorCD|
7*Sigma
ppm
0.0
0.5
1.0
1.5
0
20
40
60
80
100
Hops
Over 100 hops max|mNRRerrorCD| = 0.6 max 7 sigma = 1.72

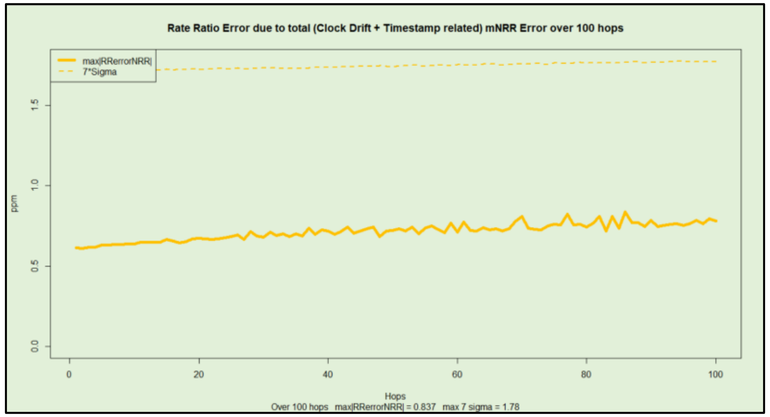
Rate Ratio Error – Timestamp & Clock Drift



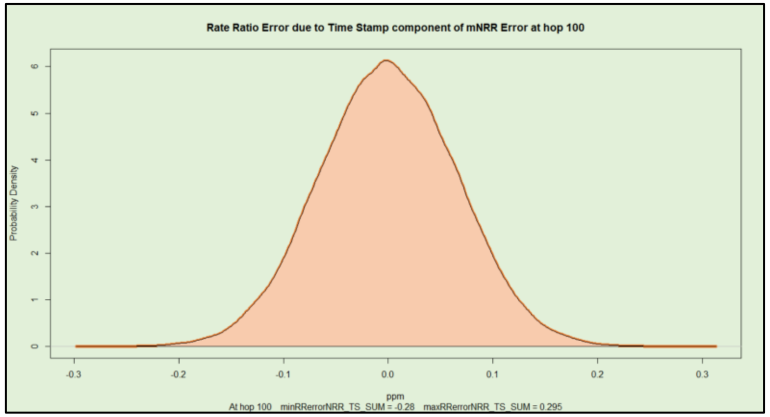


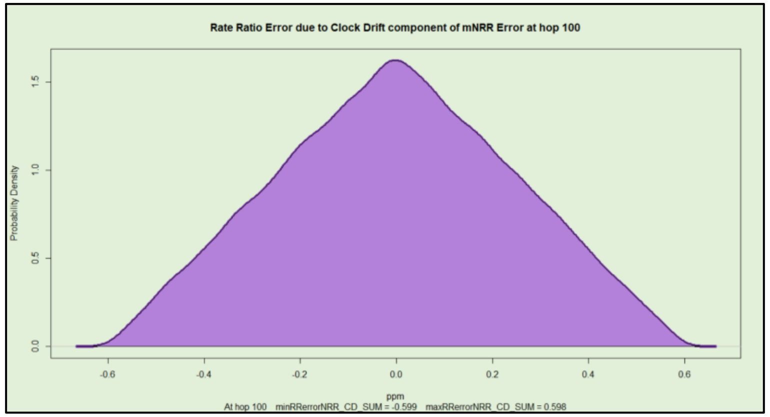


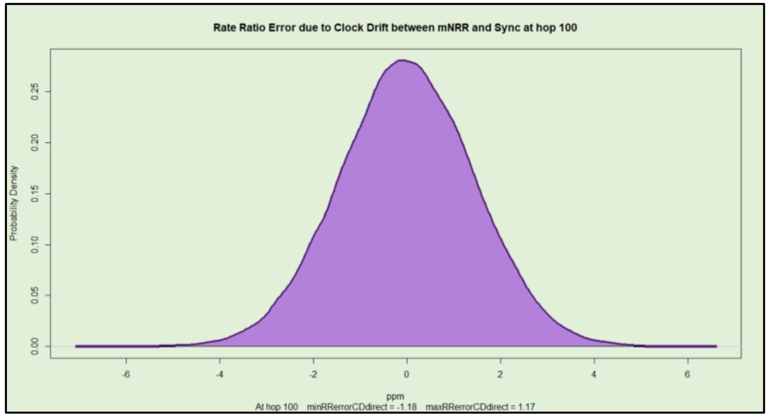


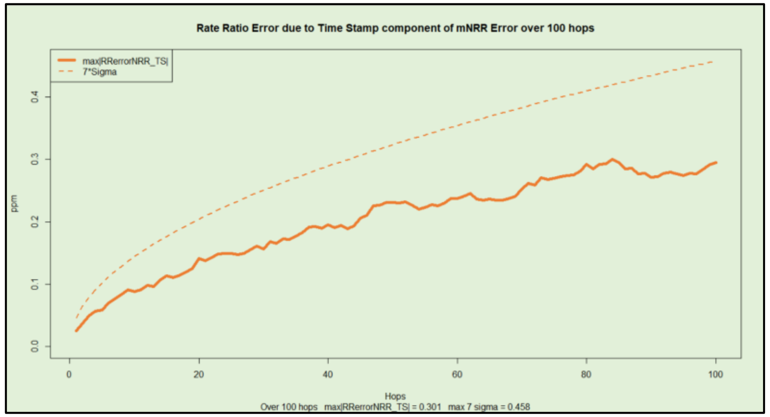


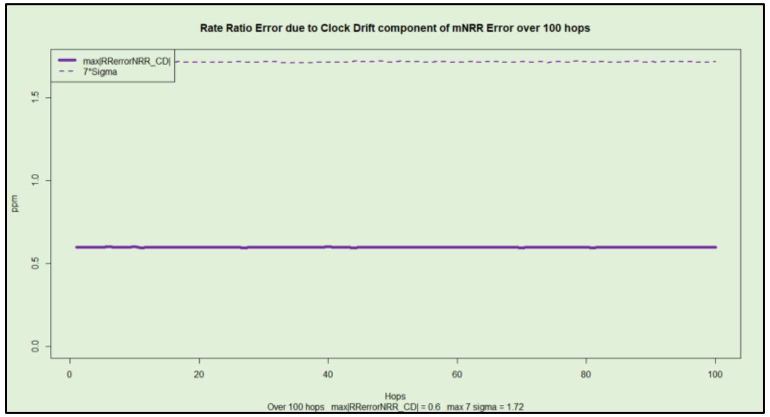
Rate Ratio Error – mNRR & Direct Clock Drift

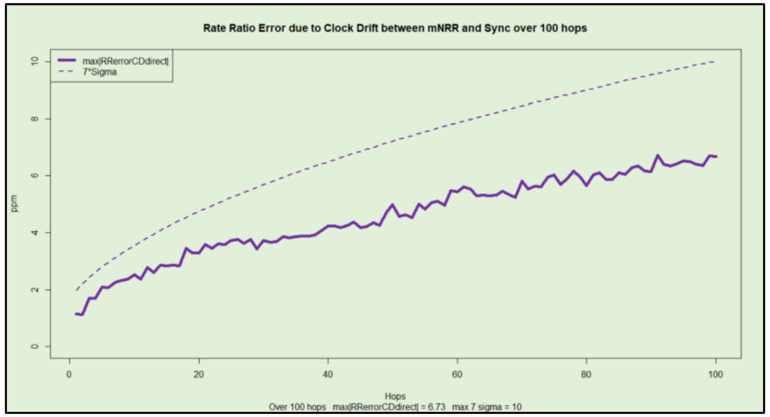




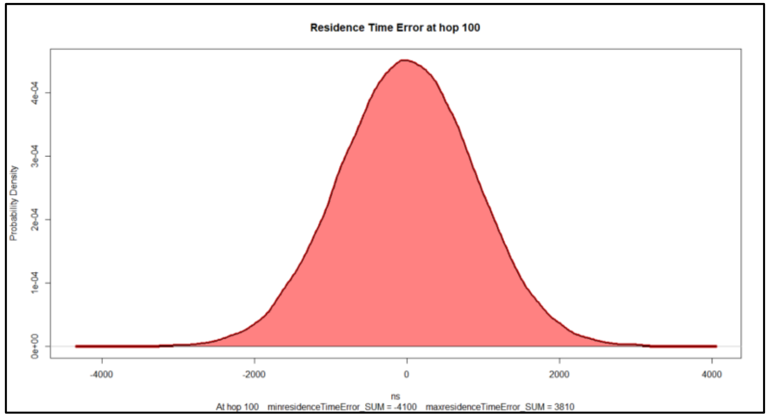


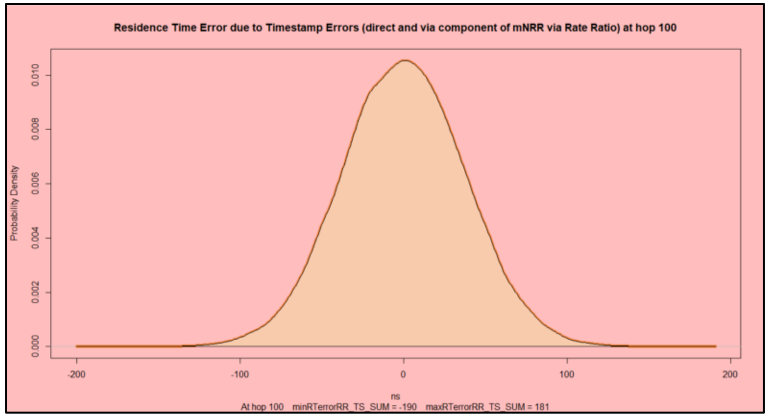


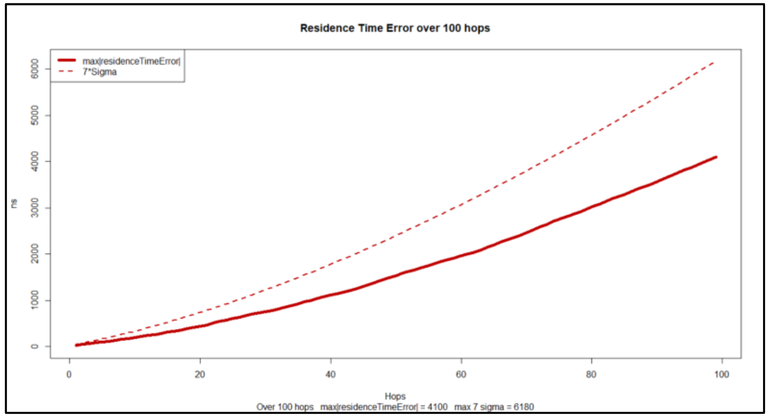


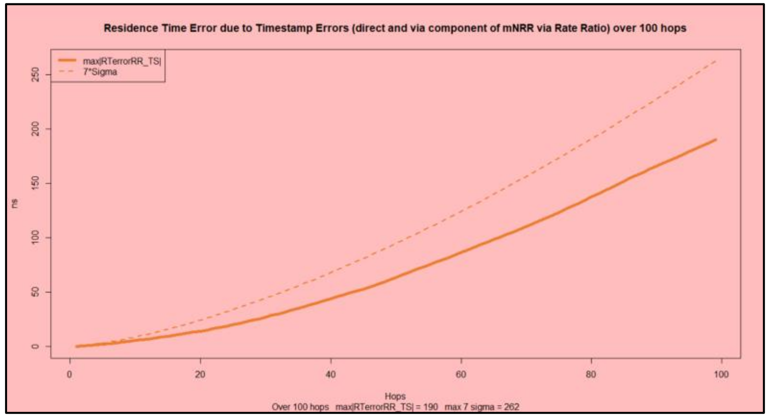


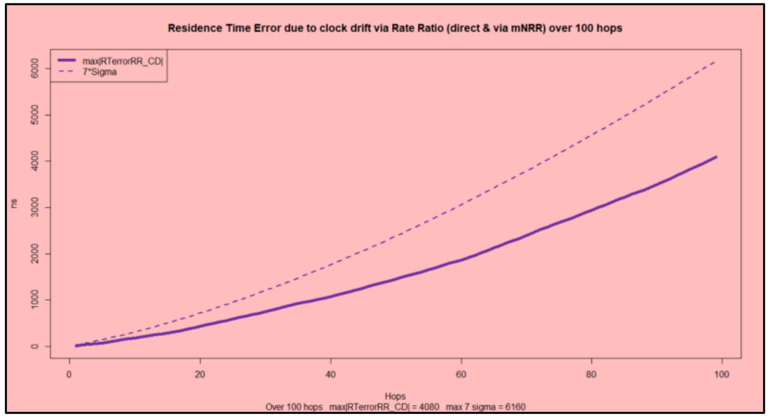
Residence Time Error – Timestamp & Clock Drift



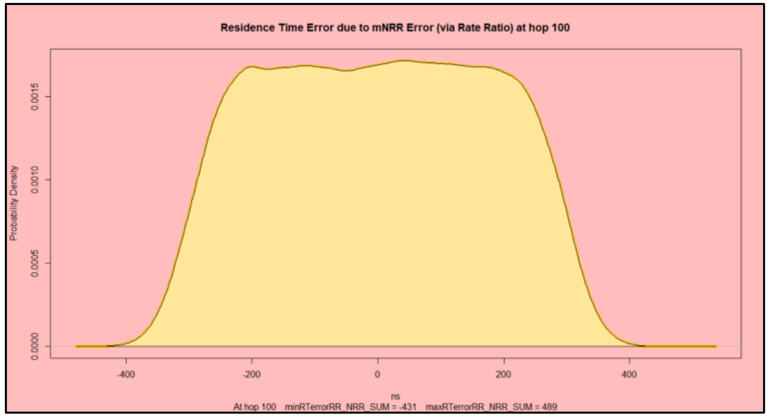


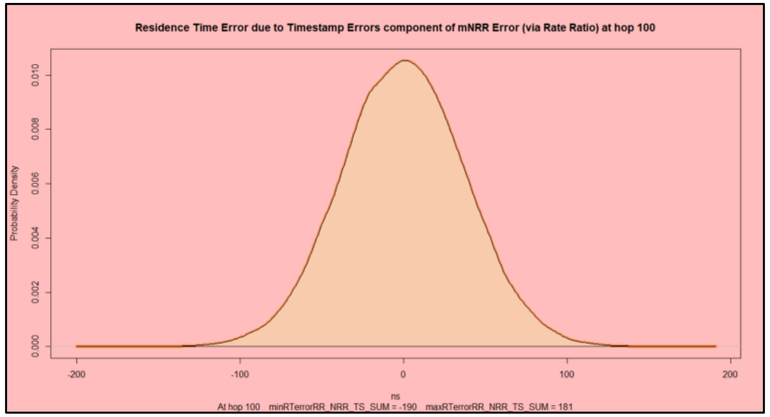


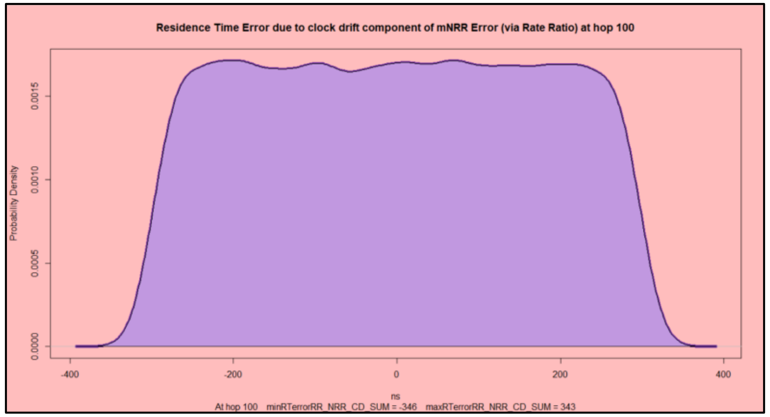


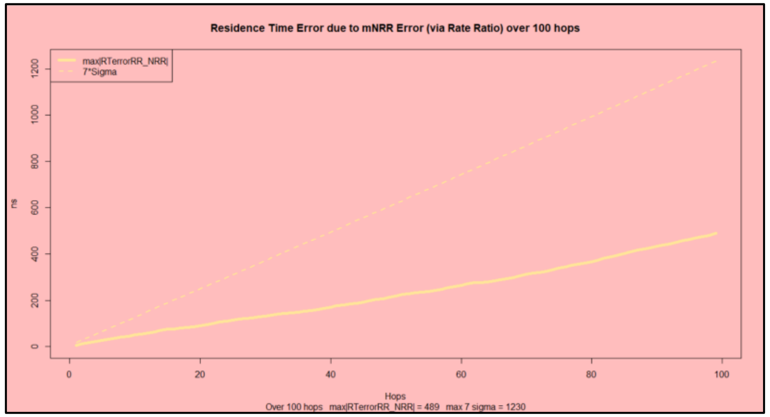


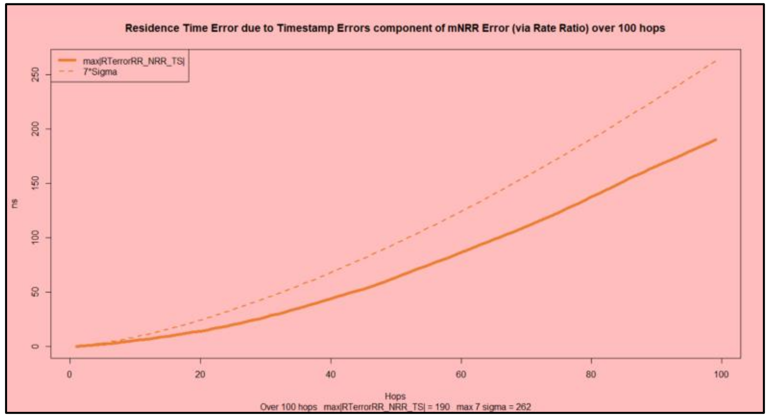
Residence Time Error – mNRR

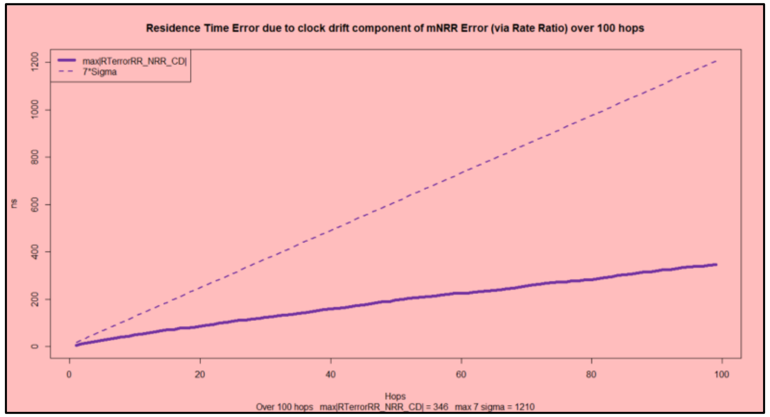




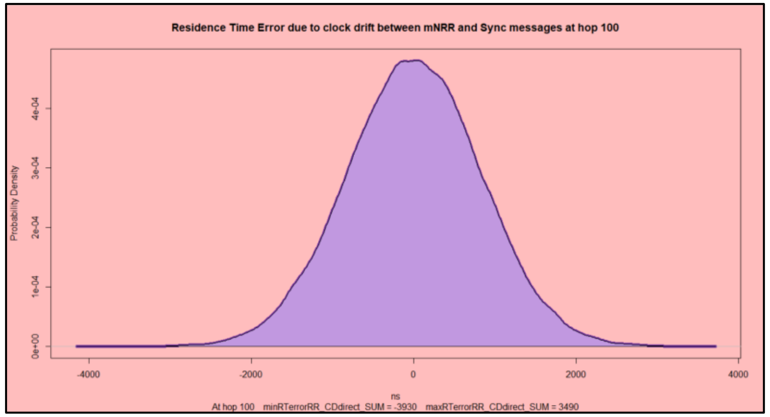


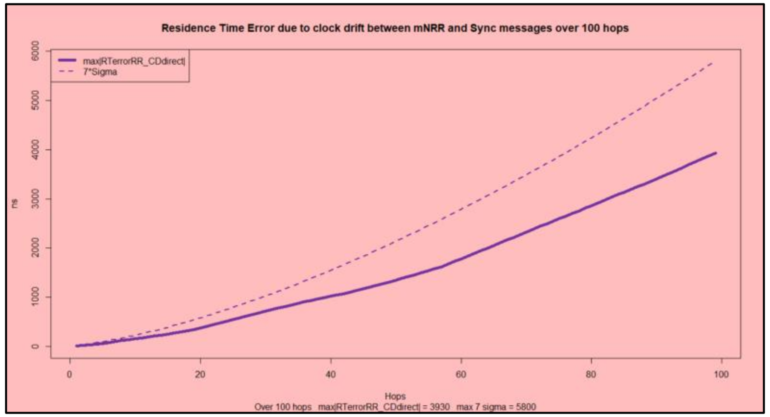




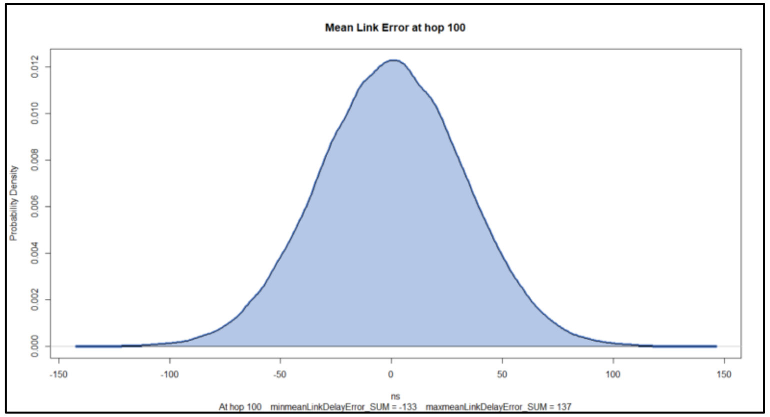


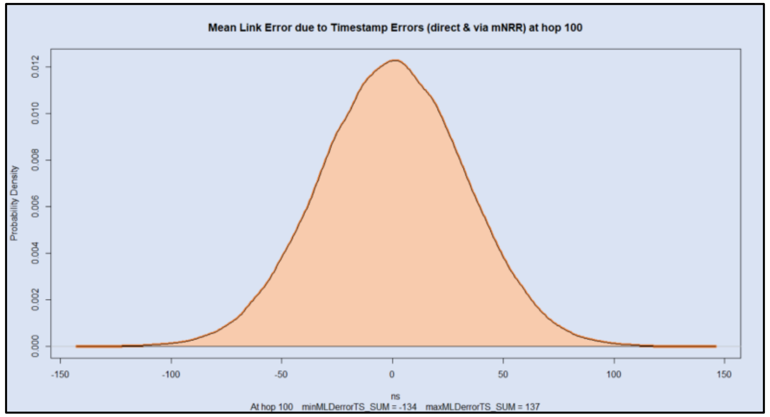
Residence Time – Direct Clock Drift

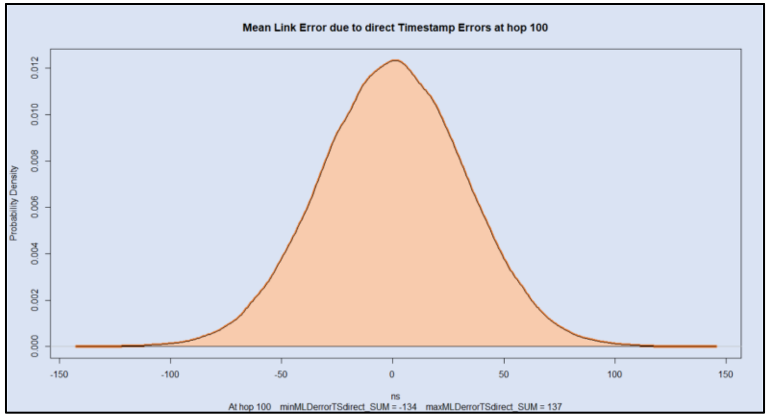


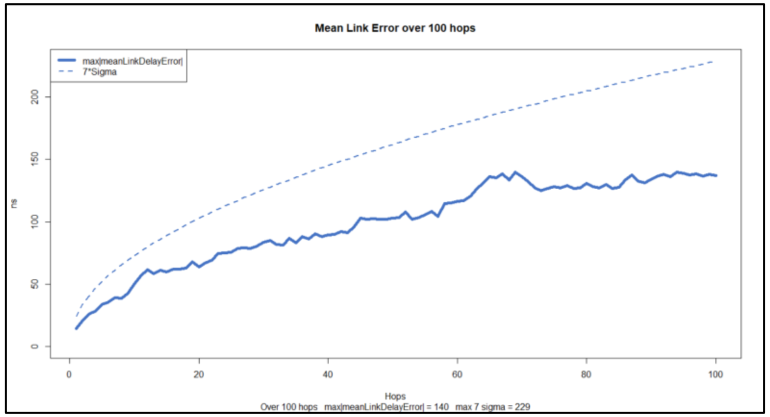


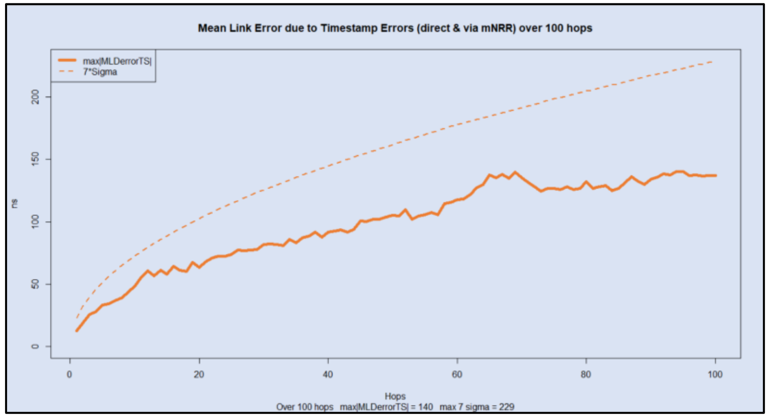
Mean Link Delay Error –
Total & Direct Timestamp Errors

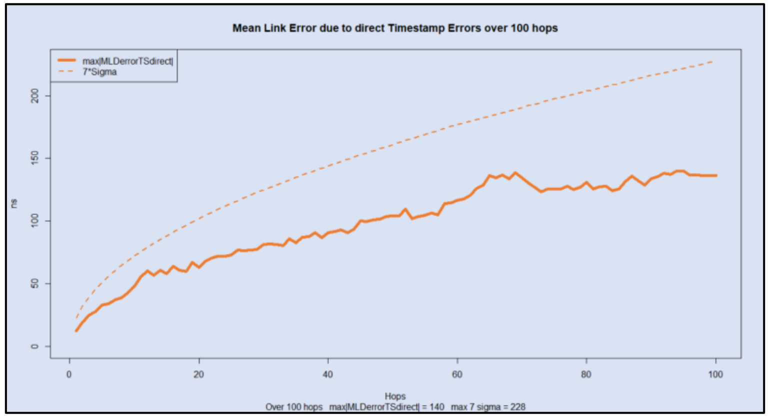




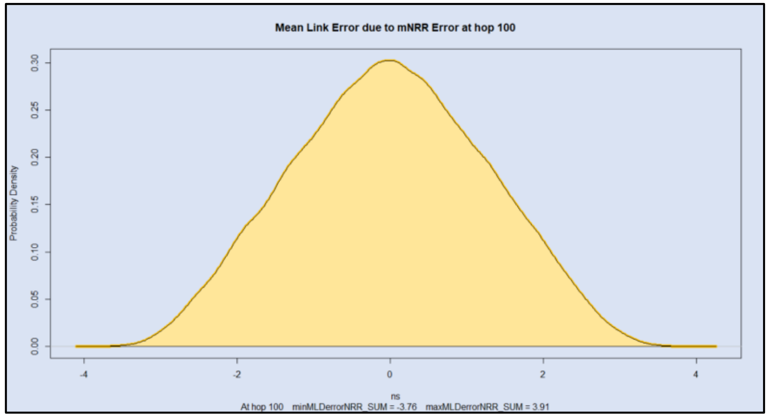


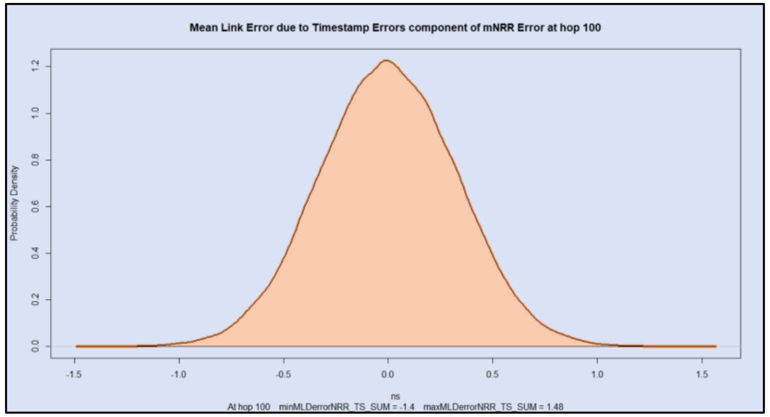


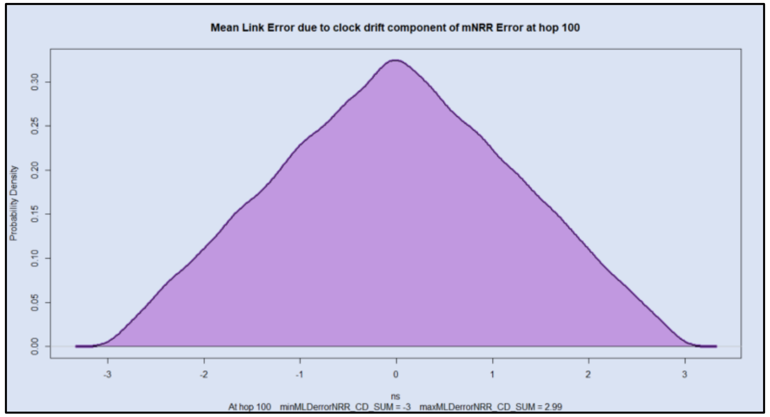


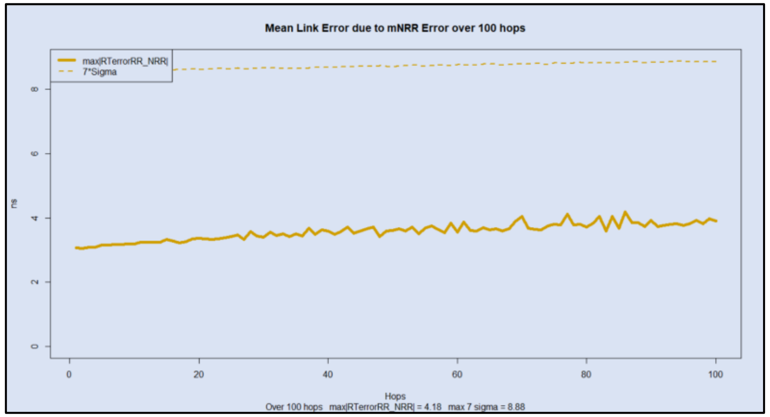


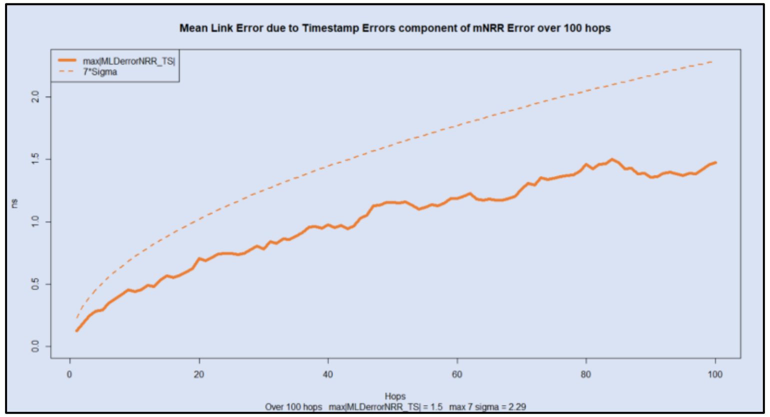
Mean Link Delay Error – mNRR Error

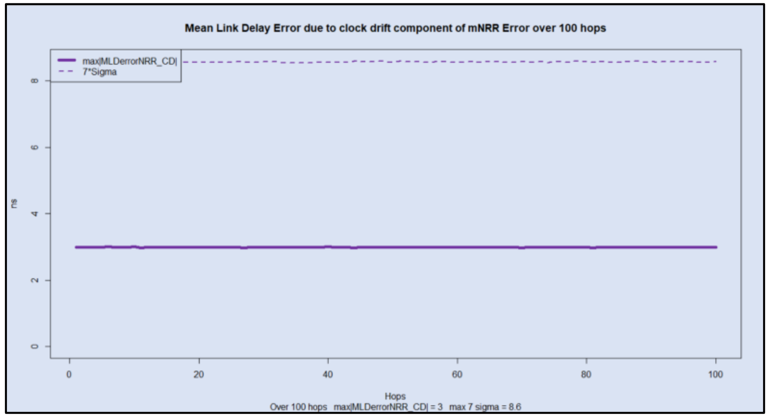




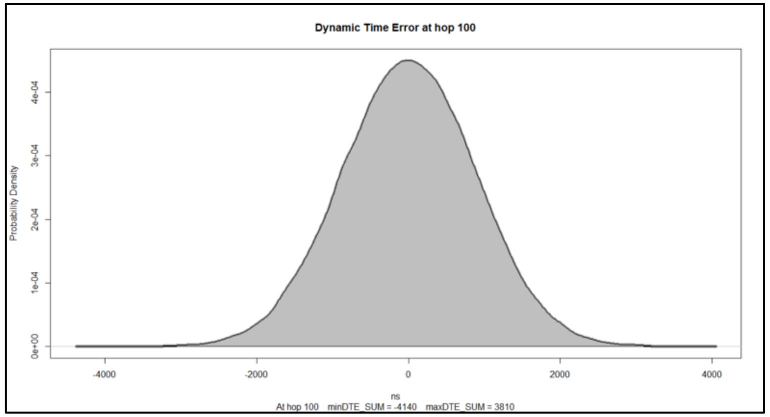


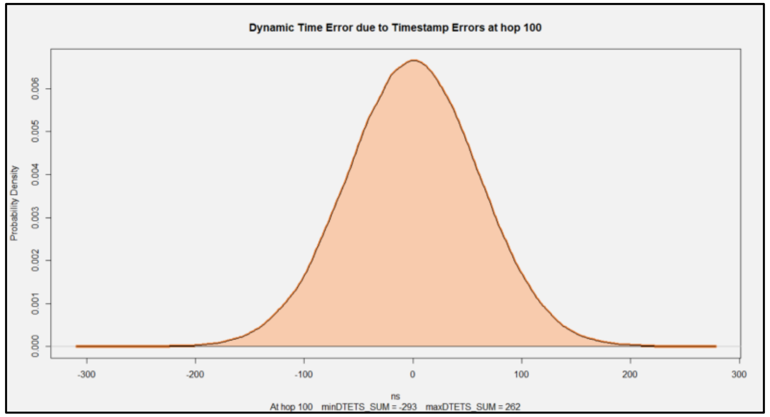


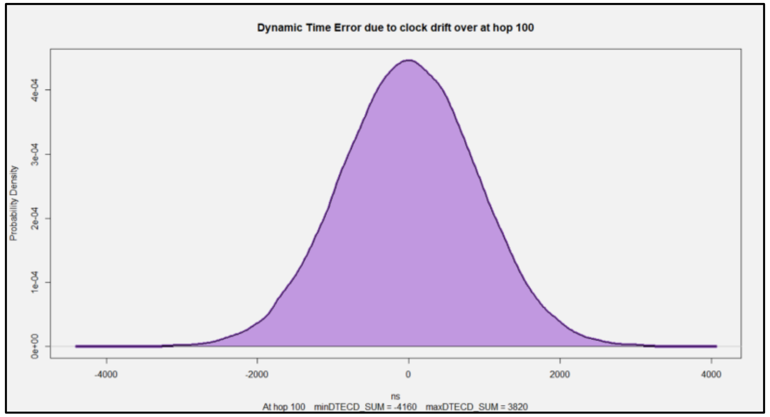


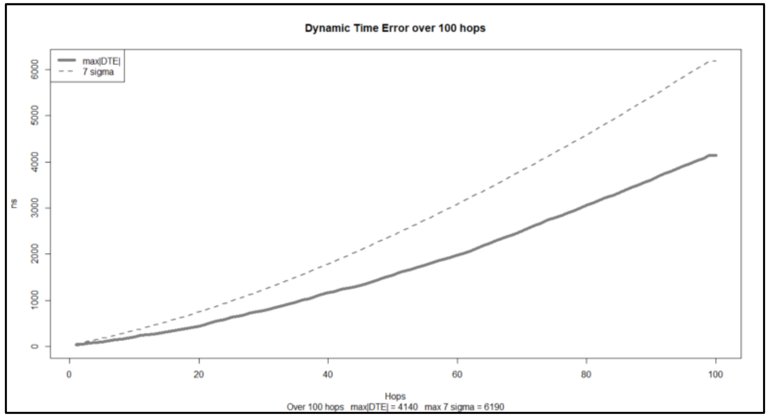


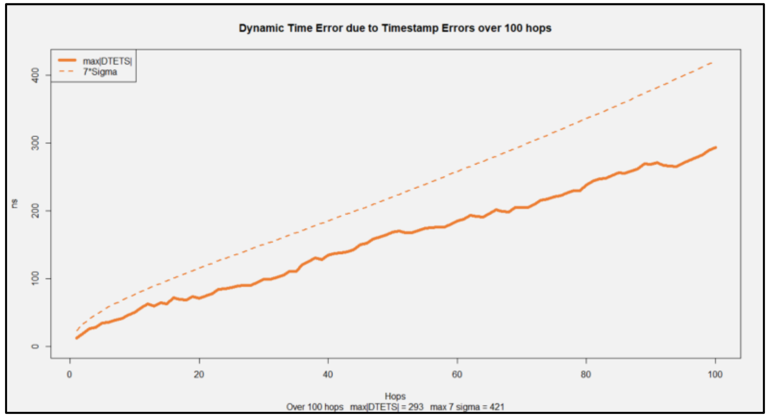
Dynamic Time Error

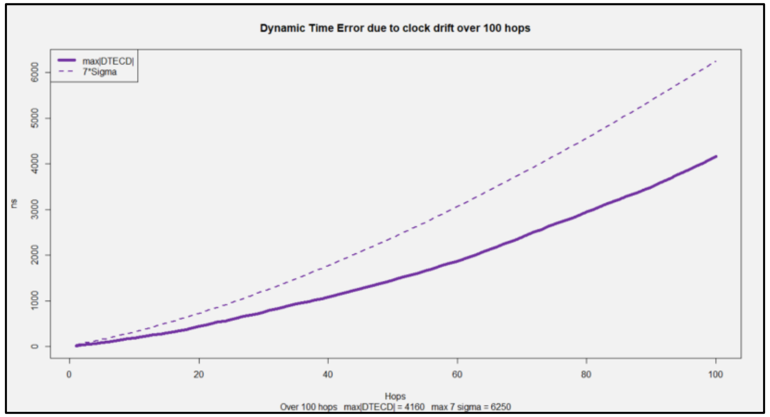




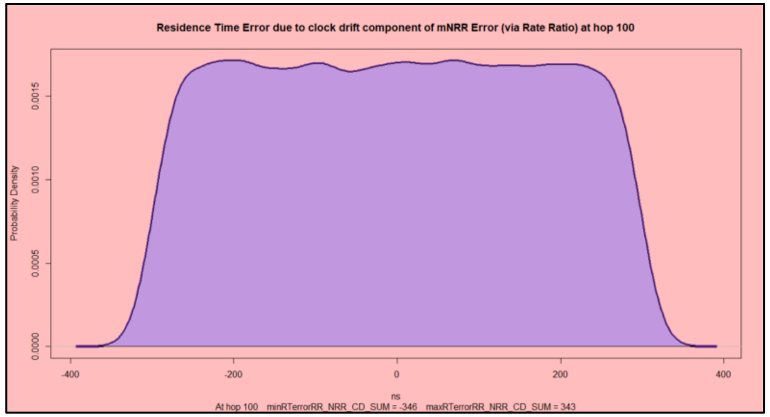




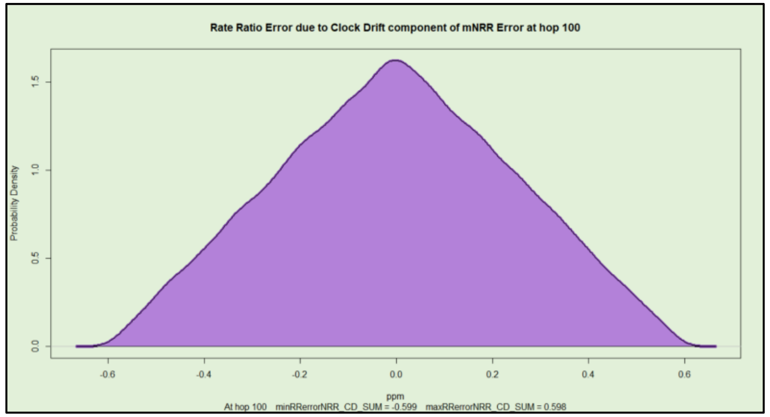


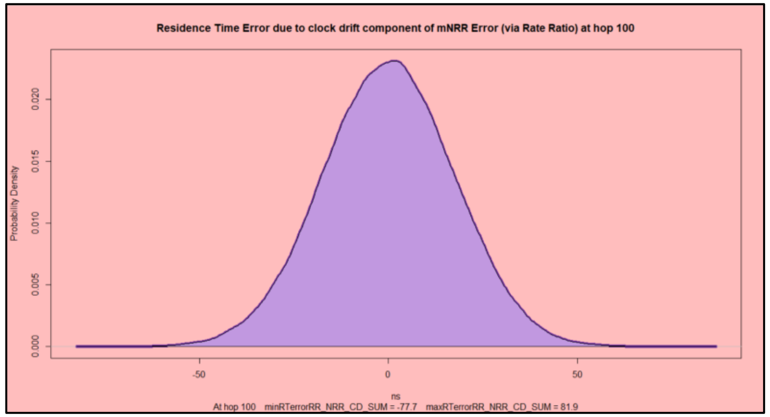


Effect of GM Clock Drift on Residence Time Error and Rate Ratio Error

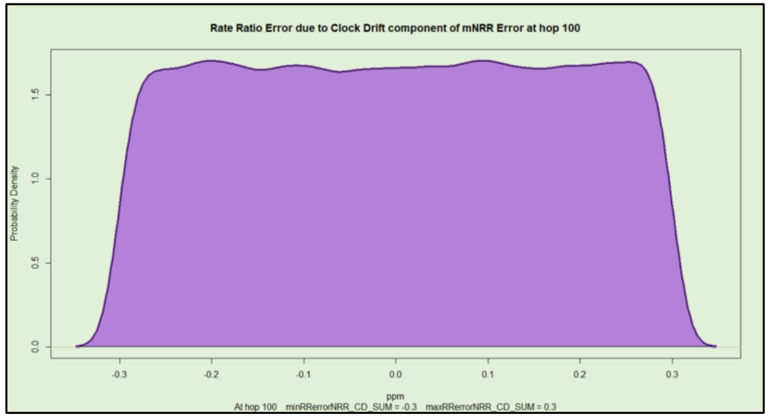


GM Clock Drift ±0.6 ppm/s

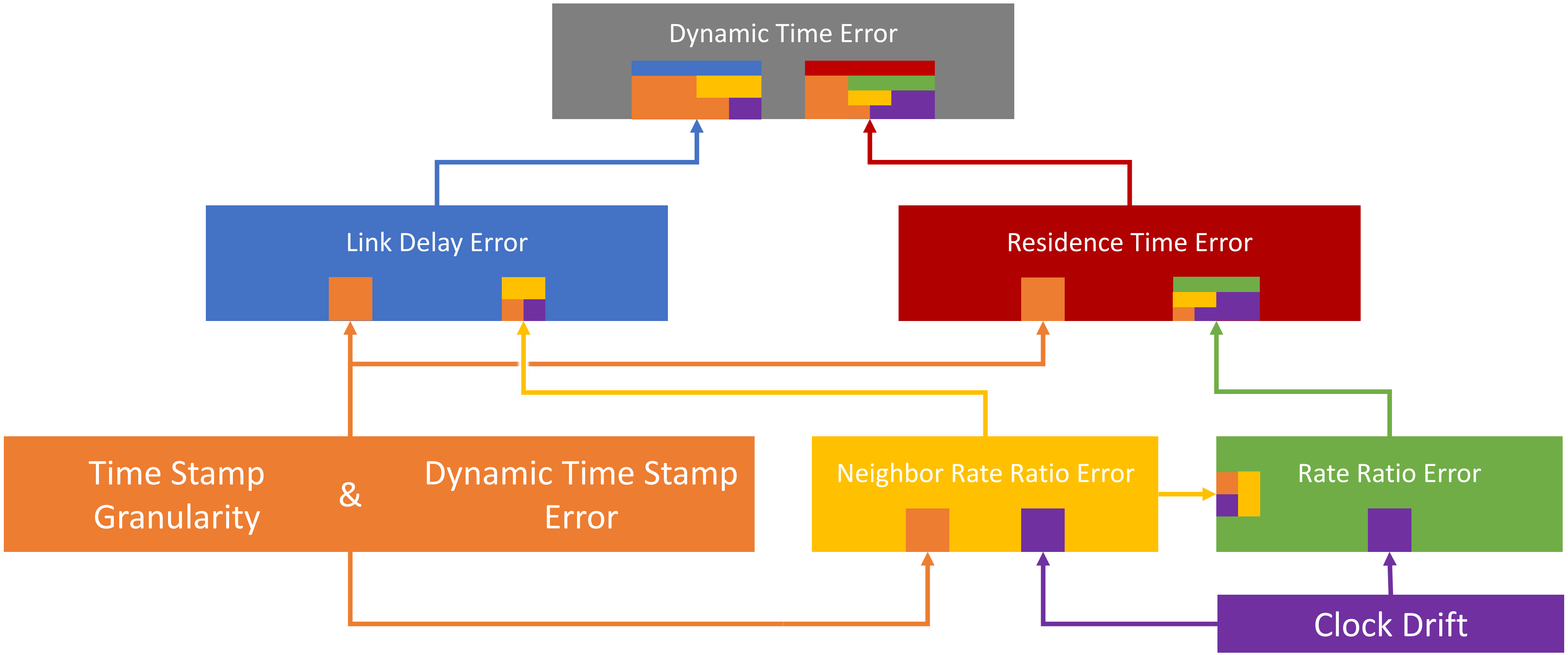




GM Clock Drift 0 ppm/s



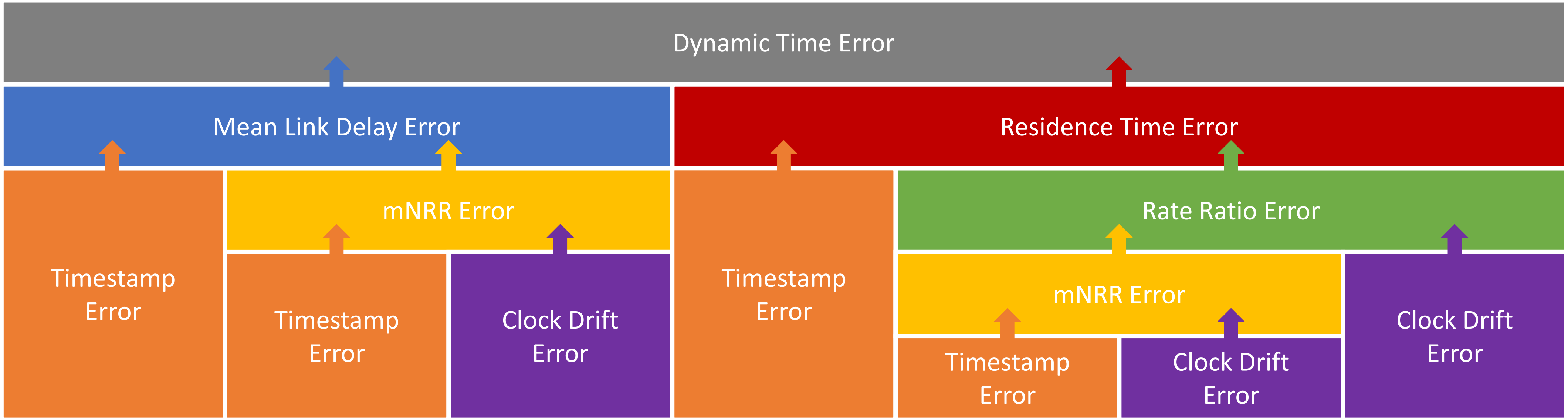
Time Sync – How Errors Add Up



All errors in this analysis are caused by either Clock Drift or Timestamp Errors

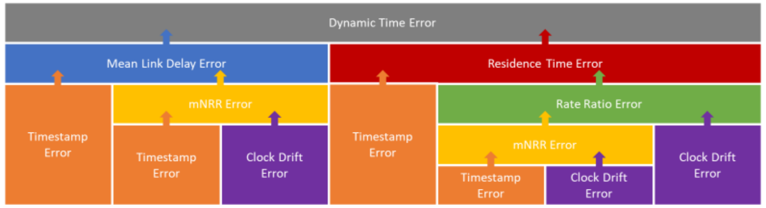
Graphical Representation of Error Accumulation

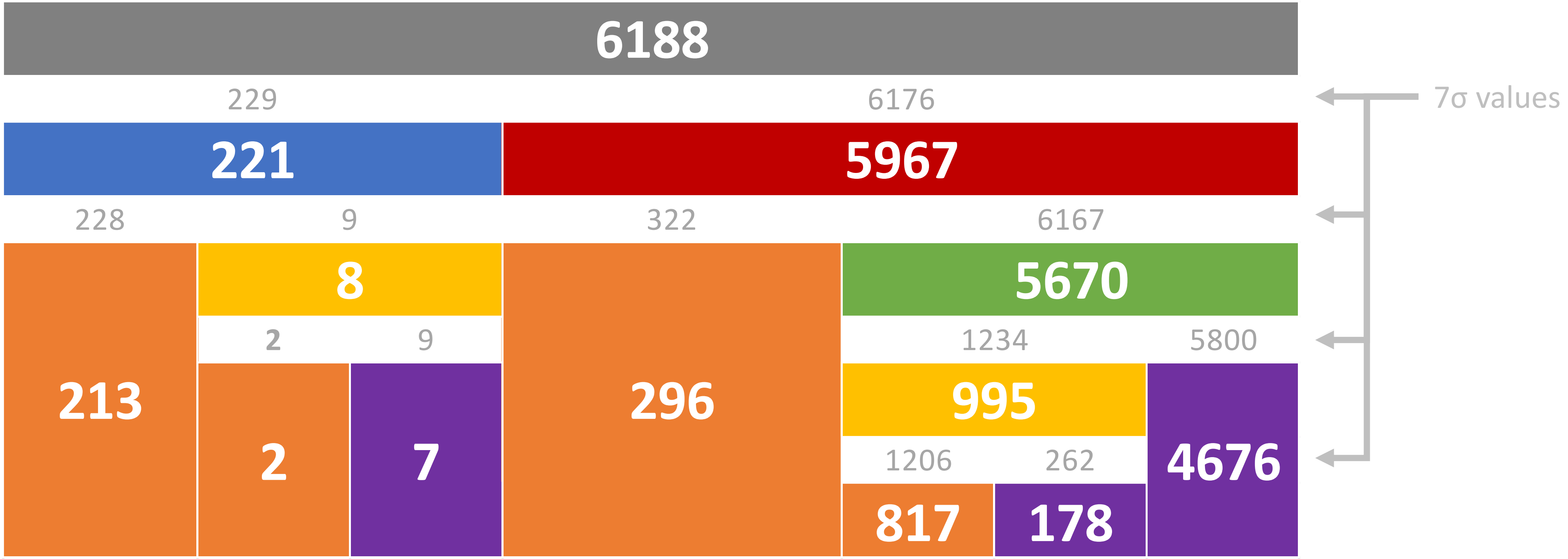
- Each error breaks down into two parts



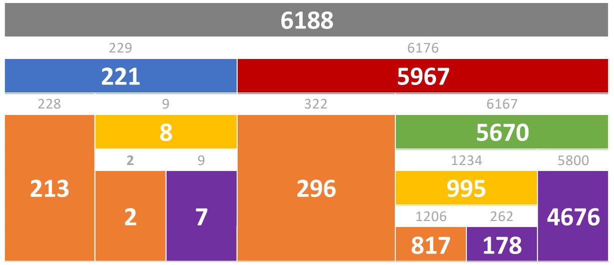
- The relative weight of each part can be judged by their 7σ values

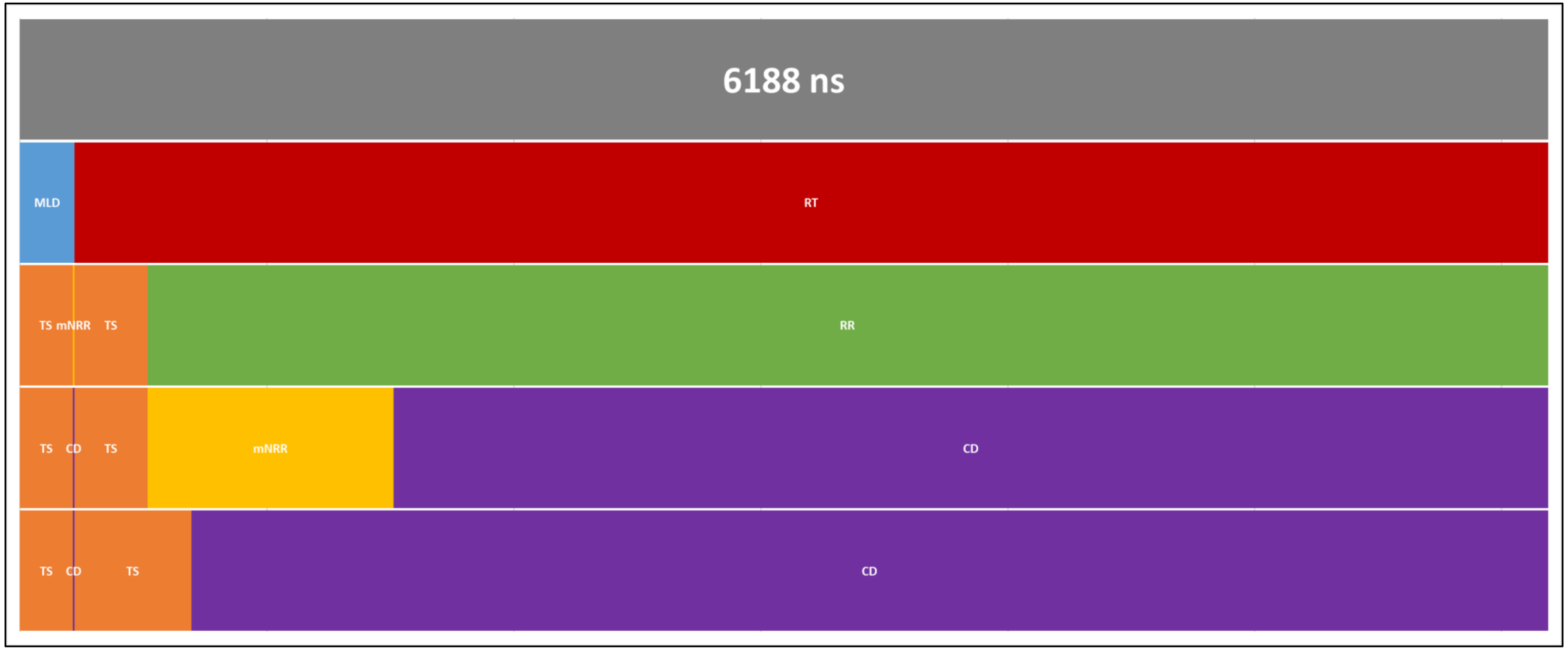
Graphical Representation of Error Accumulation





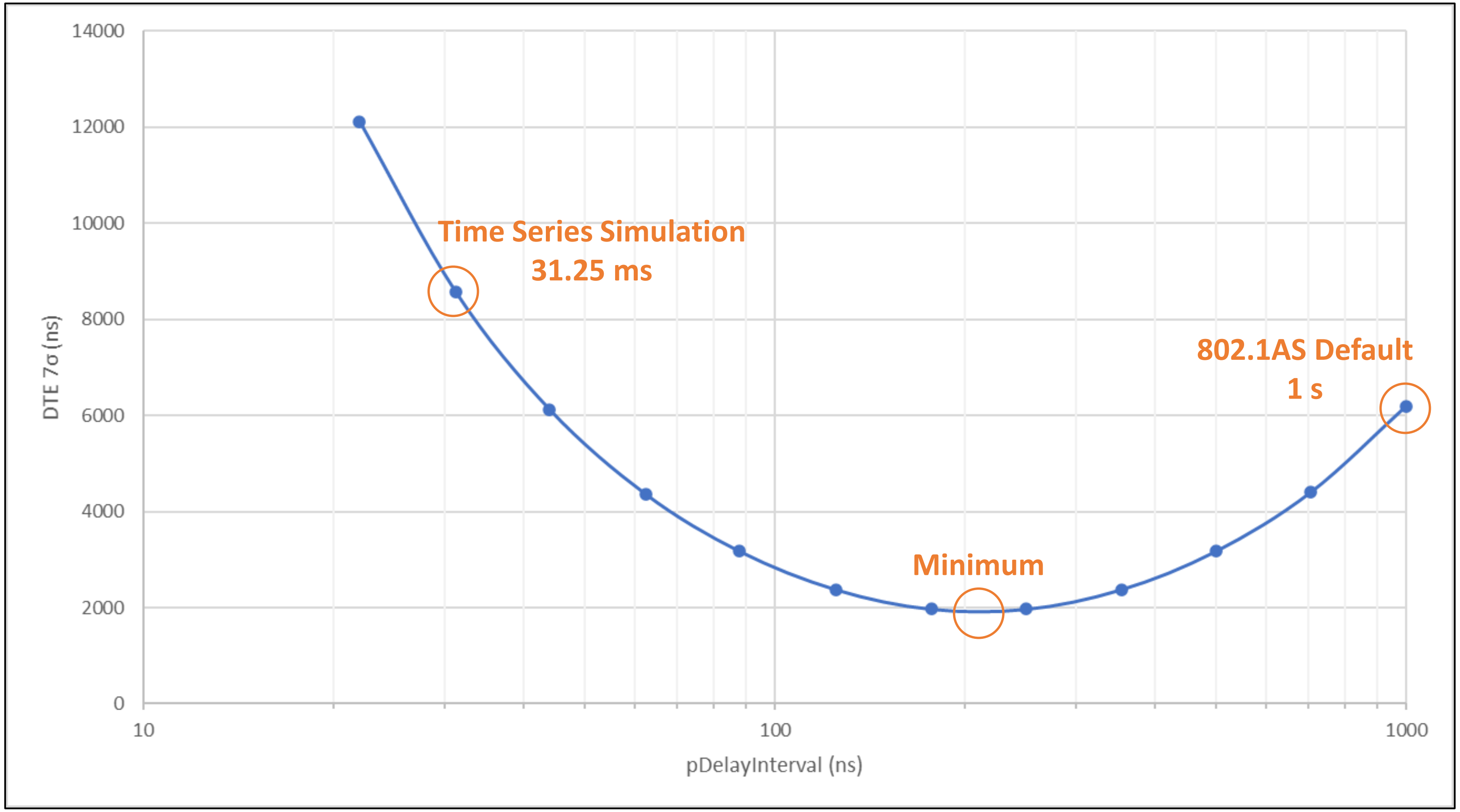
Graphical Representation of Error Accumulation





pDelayInterval Sensitivity Analysis

pDelayInterval Sensitivity Analysis



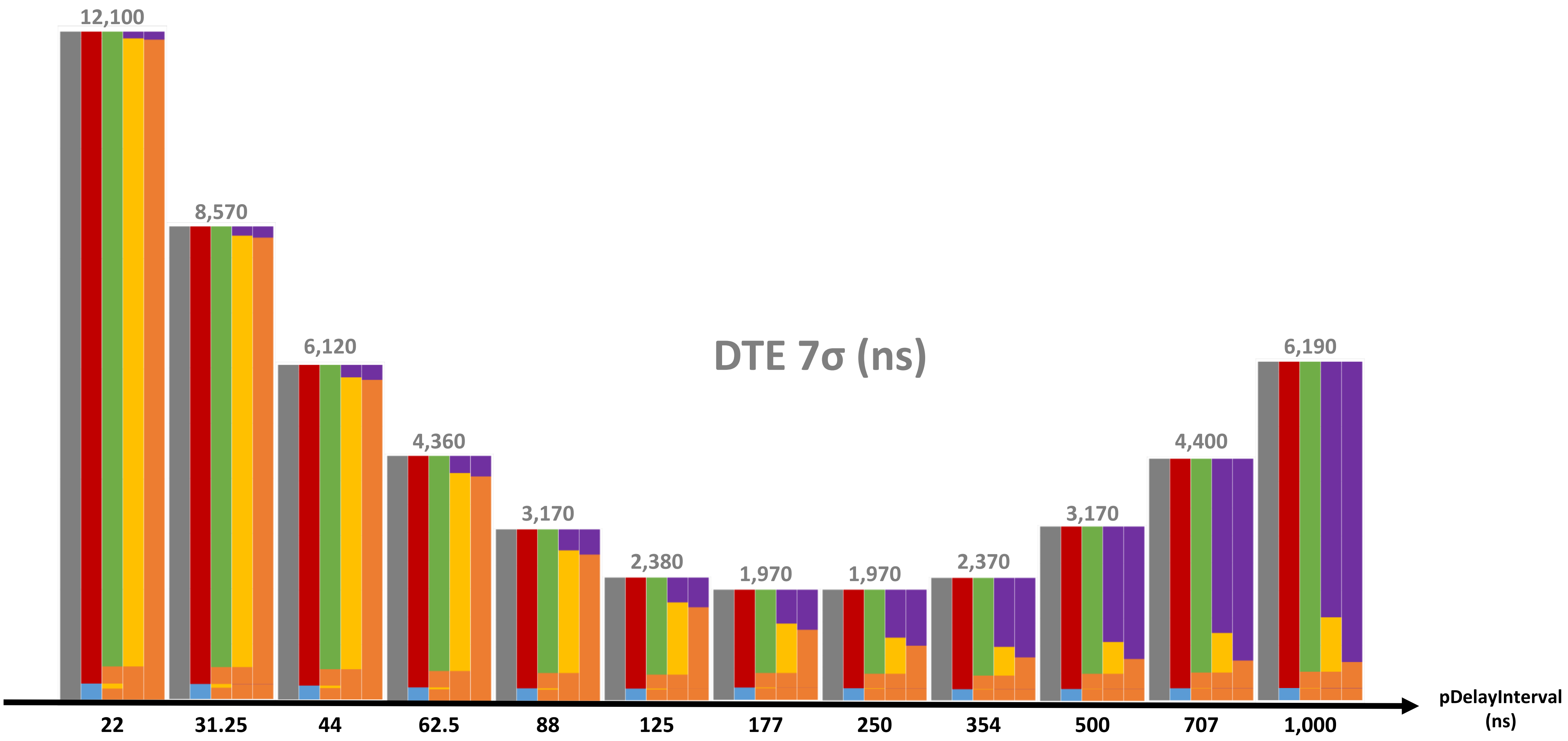
Input Errors		
GM Clock Drift Max	+0.6	ppm
GM Clock Drift Min	+0.6	ppm
Clock Drift (non-GM)	0.6	±ppm
Timestamp Granularity TX	4	±ns
Timestamp Granularity RX	4	±ns
Dynamic Time Stamp Error TX	4	±ns
Dynamic Time Stamp Error RX	4	±ns
Input Parameters		
pDelay Interval	Variable	ms
pDelay Response Time	10	ms
residenceTime	10	ms
Correction Factors		
Mean Link Delay	0	%
Drift Rate	0	%
pDelayResponse → Sync	0	%
mNRR Smoothing	1	
Configuration		
Hops	100	
Runs	100,000	

Vary This

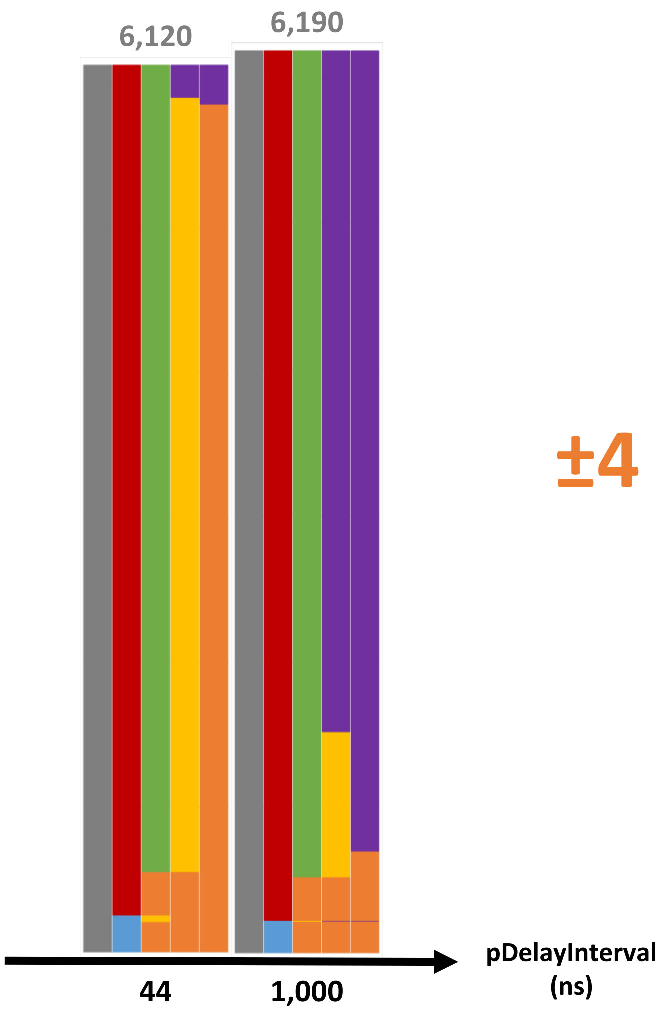
Minimum is at 250ms…

…for this set of parameters.

pDelayInterval Sensitivity Analysis



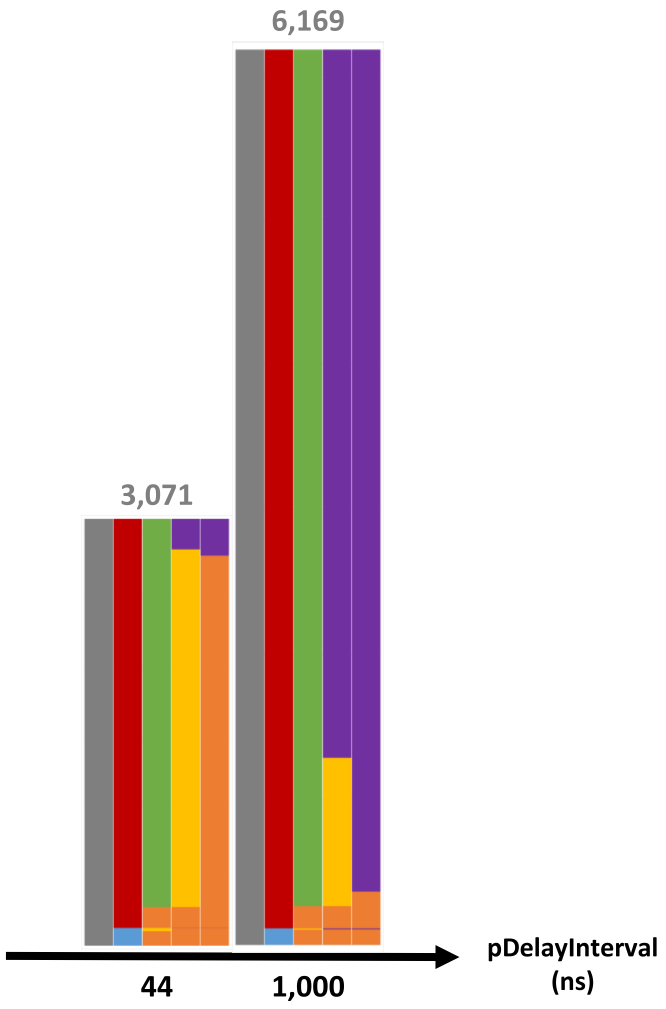
pDelayInterval Sensitivity Analysis



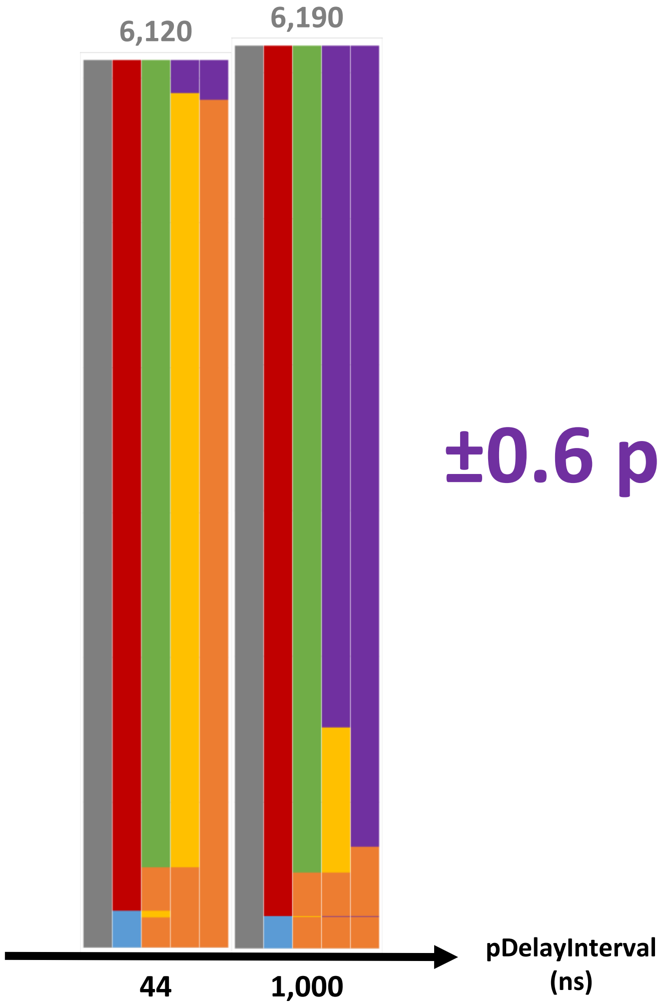
±4 ns

TSGE & DTSE

±2 ns

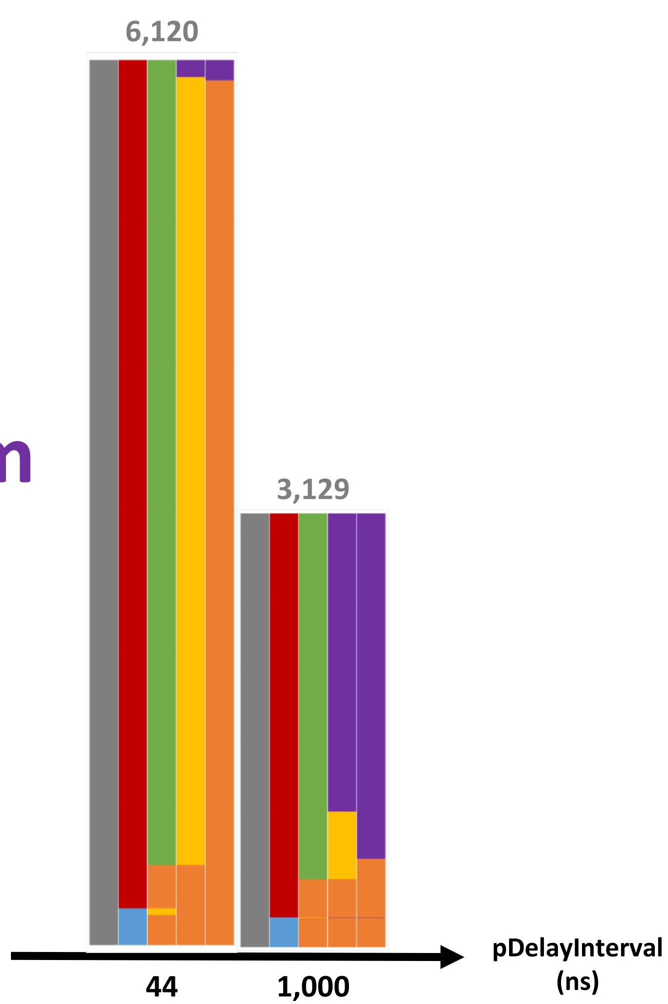


pDelayInterval Sensitivity Analysis

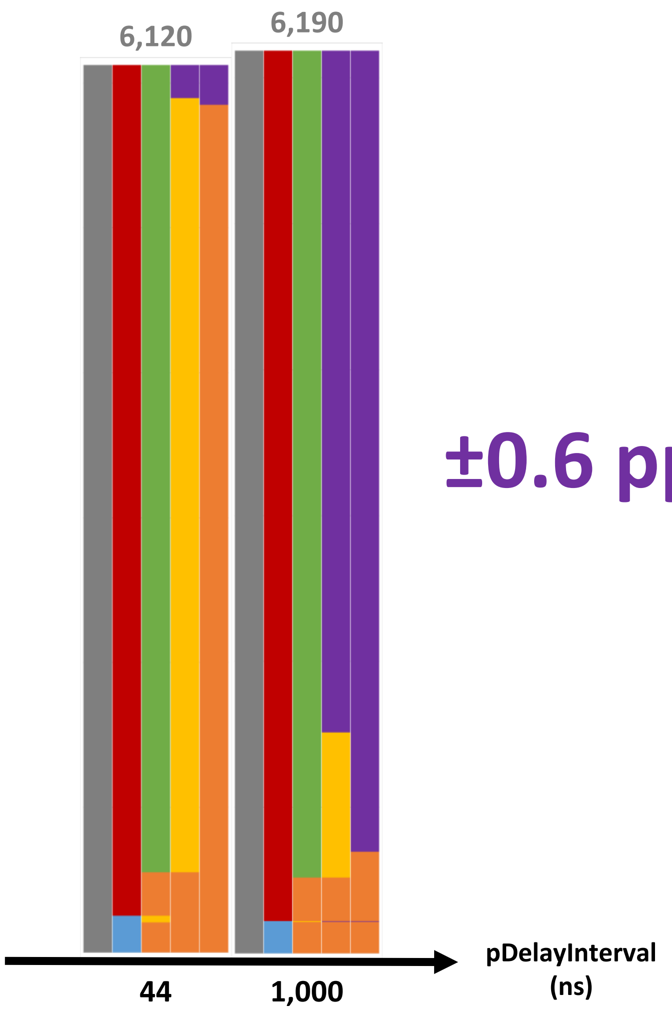


±0.6 ppm **Clock Drift** **±0.3 ppm**

GM & Others



pDelayInterval Sensitivity Analysis



±0.6 ppm

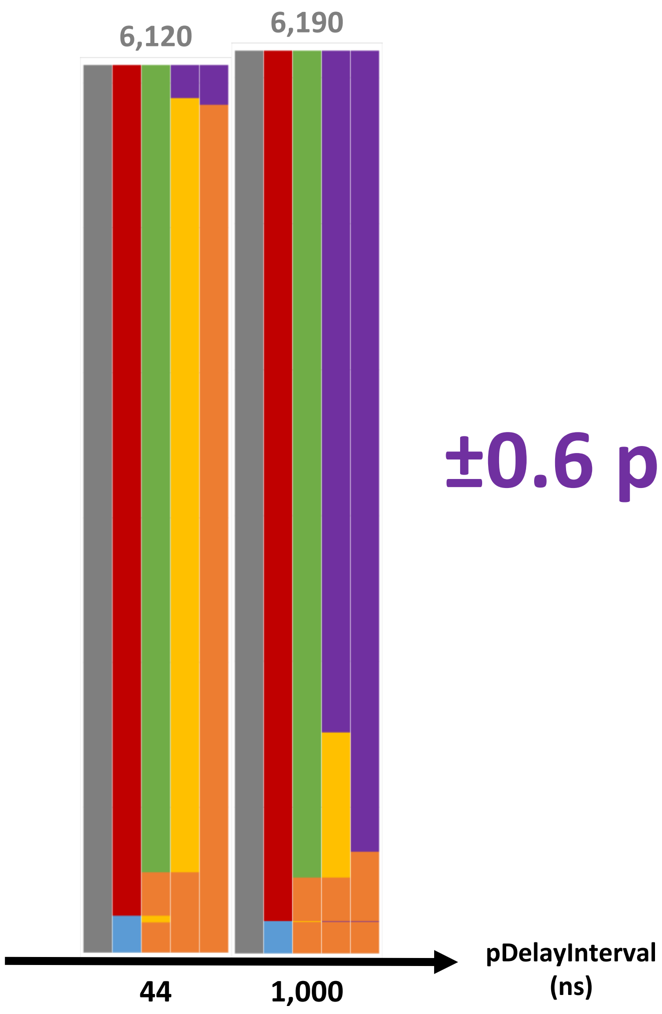
Clock Drift

GM Only

Others remain at ±0.6 ppm

±0.3 ppm

pDelayInterval Sensitivity Analysis



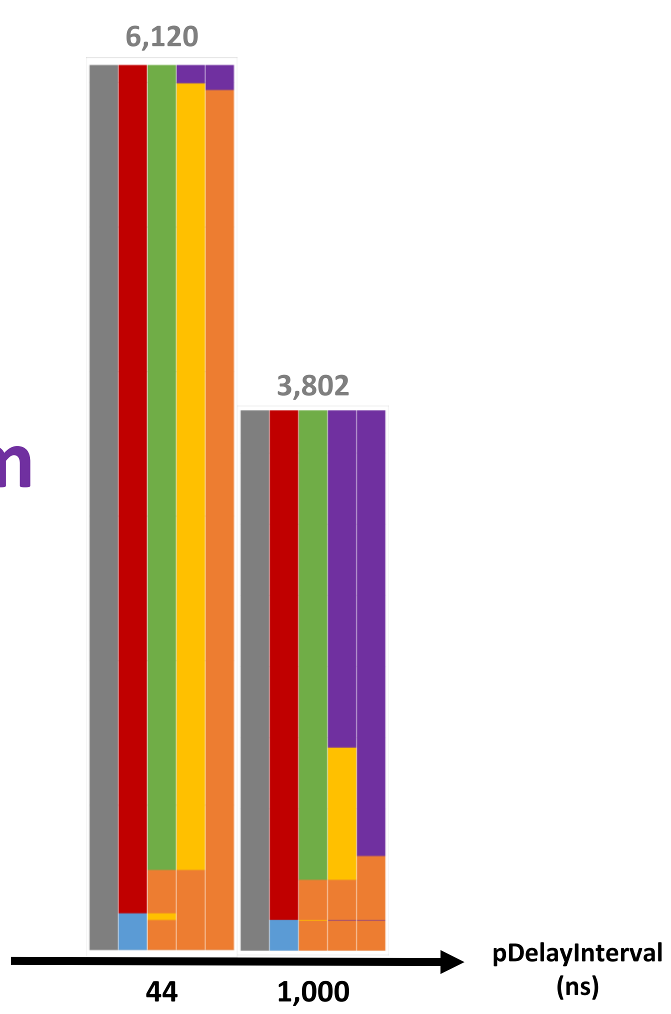
±0.6 ppm

Clock Drift

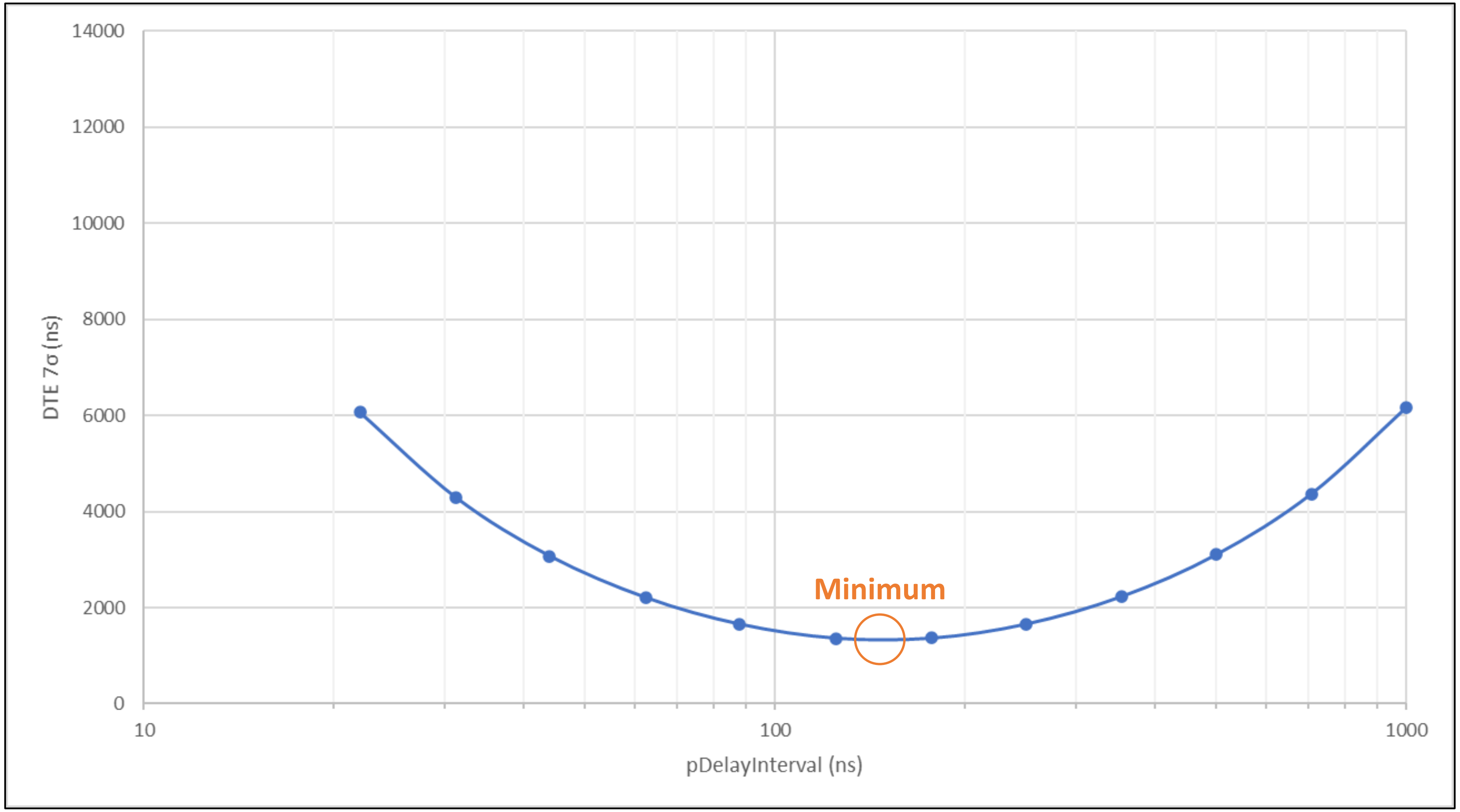
Others Only

GM remains at ±0.6 ppm

±0.3 ppm



pDelayInterval Sensitivity Analysis with Lower Timestamp Error

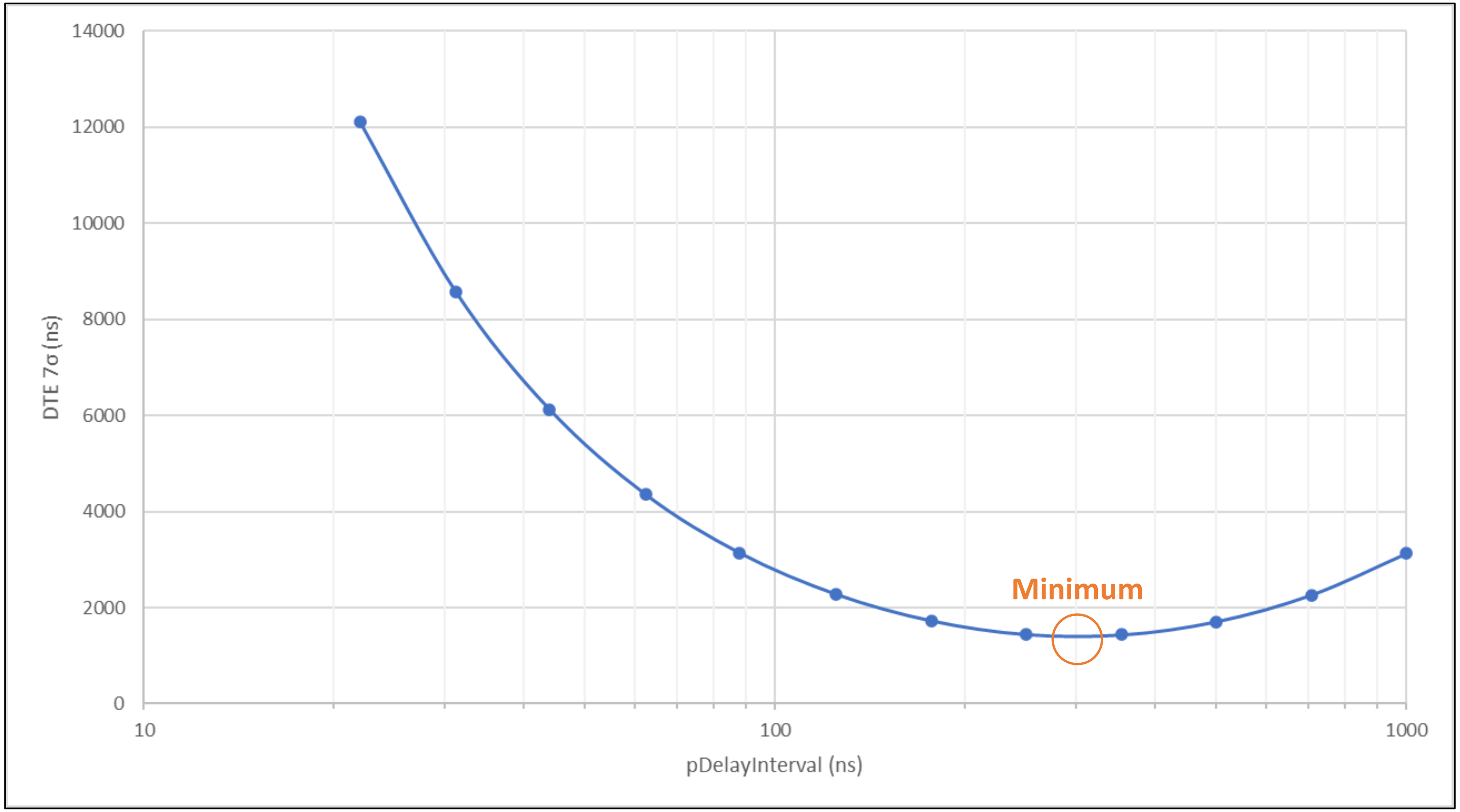


Input Errors		
GM Clock Drift Max	+0.6	ppm
GM Clock Drift Min	+0.6	ppm
Clock Drift (non-GM)	0.6	±ppm
Timestamp Granularity TX	2	±ns
Timestamp Granularity RX	2	±ns
Dynamic Time Stamp Error TX	2	±ns
Dynamic Time Stamp Error RX	2	±ns
Input Parameters		
pDelay Interval	Variable	ms
pDelay Response Time	10	ms
residenceTime	10	ms
Correction Factors		
Mean Link Delay	0	%
Drift Rate	0	%
pDelayResponse → Sync	0	%
mNRR Smoothing	1	
Configuration		
Hops	100	
Runs	100,000	

Minimum is at approx 150ms…

…for this set of parameters.

pDelayInterval Sensitivity Analysis with Lower Clock Drift



Input Errors		
GM Clock Drift Max	+0.3	ppm
GM Clock Drift Min	+0.3	ppm
Clock Drift (non-GM)	0.3	±ppm
Timestamp Granularity TX	4	±ns
Timestamp Granularity RX	4	±ns
Dynamic Time Stamp Error TX	4	±ns
Dynamic Time Stamp Error RX	4	±ns
Input Parameters		
pDelay Interval	Variable	ms
pDelay Response Time	10	ms
residenceTime	10	ms
Correction Factors		
Mean Link Delay	0	%
Drift Rate	0	%
pDelayResponse → Sync	0	%
mNRR Smoothing	1	
Configuration		
Hops	100	
Runs	100,000	

Minimum is at approx 300ms…

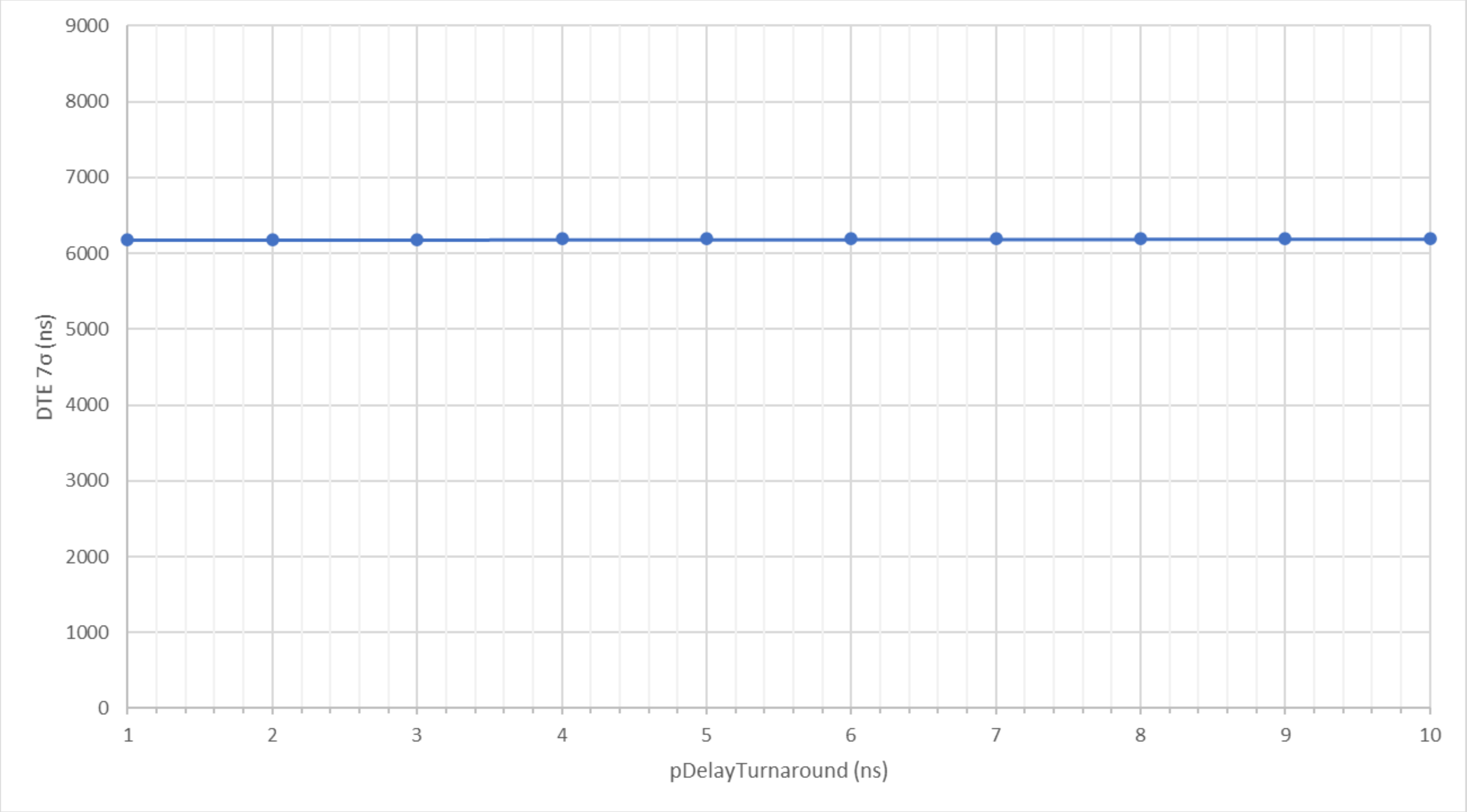
…for this set of parameters.

pDelayInterval Sensitivity - Conclusion

- Choice of pDelayInterval can have a large impact on DTE.
- pDelay interval can be optimised but the optimal choice depends on other parameters and sources of error.
- Monte Carlo Analysis is an effective tool for investigating this further.

pDelayTurnaround Sensitivity Analysis

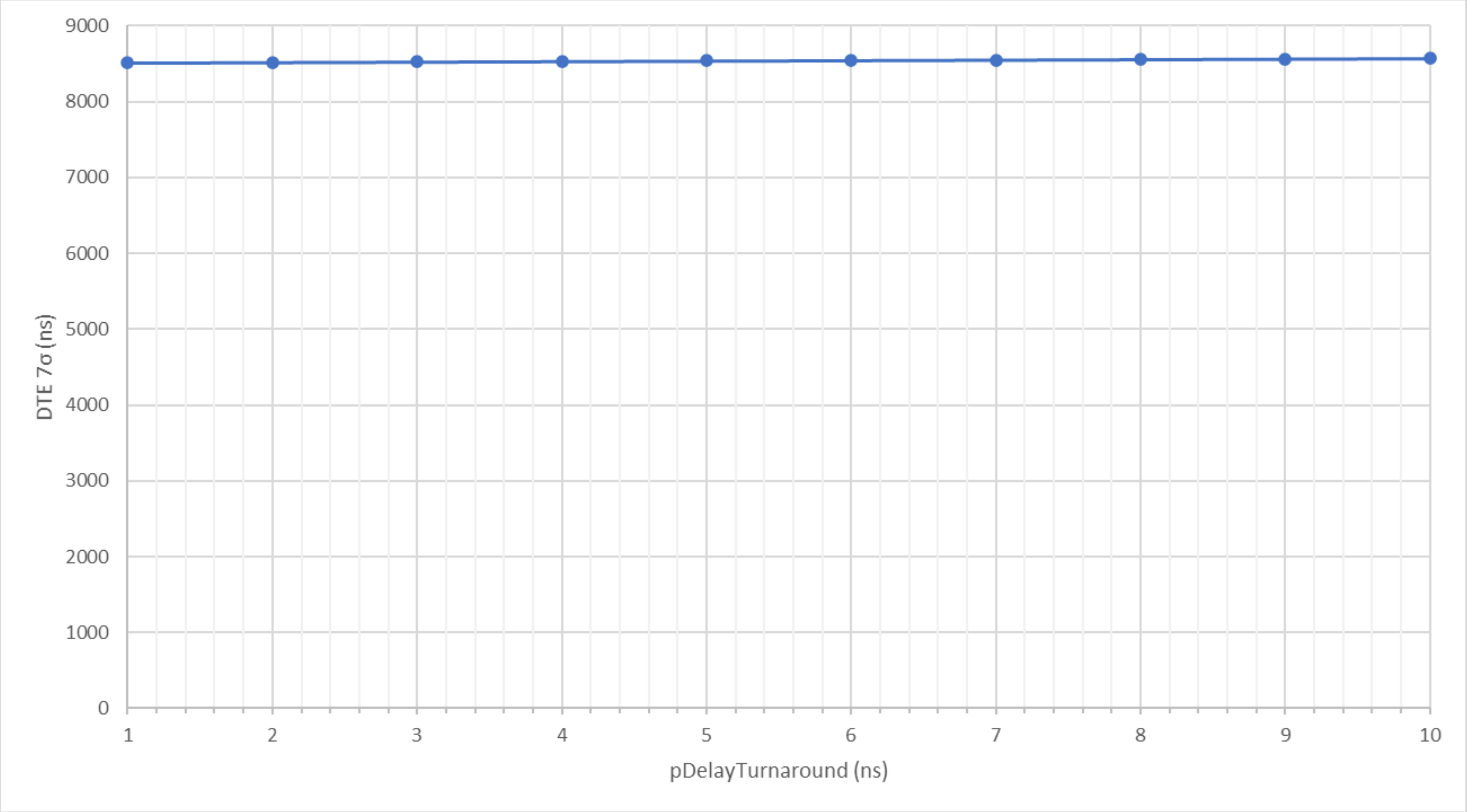
pDelayTurnaround Sensitivity – 1 s pDelay Interval



Input Errors		
GM Clock Drift Max	+0.6	ppm
GM Clock Drift Min	+0.6	ppm
Clock Drift (non-GM)	0.6	±ppm
Timestamp Granularity TX	4	±ns
Timestamp Granularity RX	4	±ns
Dynamic Time Stamp Error TX	4	±ns
Dynamic Time Stamp Error RX	4	±ns
Input Parameters		
pDelay Interval	**1000**	ms
pDelay Response Time	**Variable**	ms
residenceTime	10	ms
Correction Factors		
Mean Link Delay	0	%
Drift Rate	0	%
pDelayResponse → Sync	0	%
mNRR Smoothing	1	
Configuration		
Hops	100	
Runs	100,000	

DTE is not sensitive to pDelayTurnaround when pDelayInterval is high.

pDelayTurnaround Sensitivity – 31.25 ms pDelay Interval

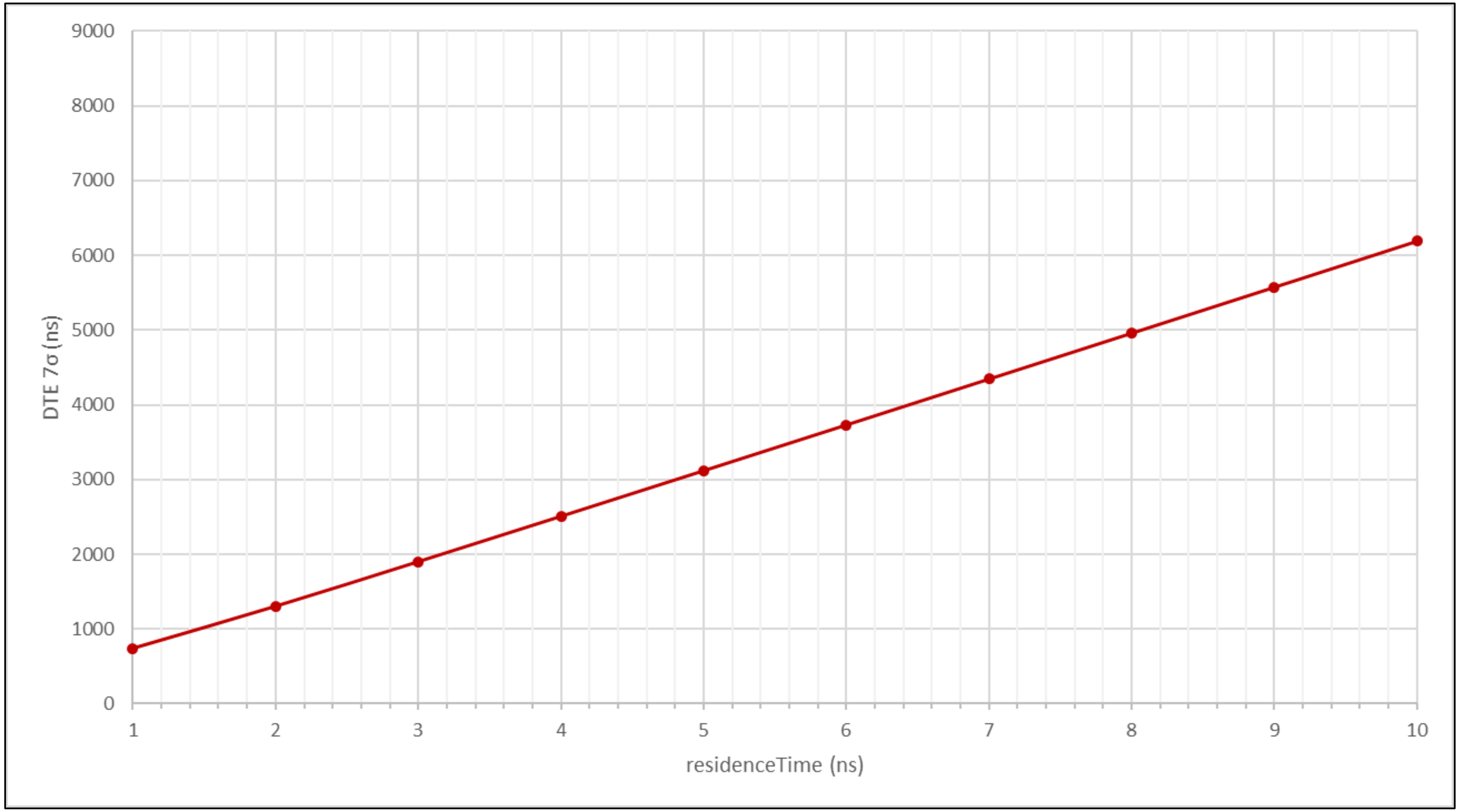


Input Errors		
GM Clock Drift Max	+0.6	ppm
GM Clock Drift Min	+0.6	ppm
Clock Drift (non-GM)	0.6	±ppm
Timestamp Granularity TX	4	±ns
Timestamp Granularity RX	4	±ns
Dynamic Time Stamp Error TX	4	±ns
Dynamic Time Stamp Error RX	4	±ns
Input Parameters		
pDelay Interval	**31.25**	ms
pDelay Response Time	**Variable**	ms
residenceTime	10	ms
Correction Factors		
Mean Link Delay	0	%
Drift Rate	0	%
pDelayResponse → Sync	0	%
mNRR Smoothing	1	
Configuration		
Hops	100	
Runs	100,000	

And DTE is not sensitive to pDelayTurnaround when pDelayInterval is low.

Residence Time Sensitivity Analysis

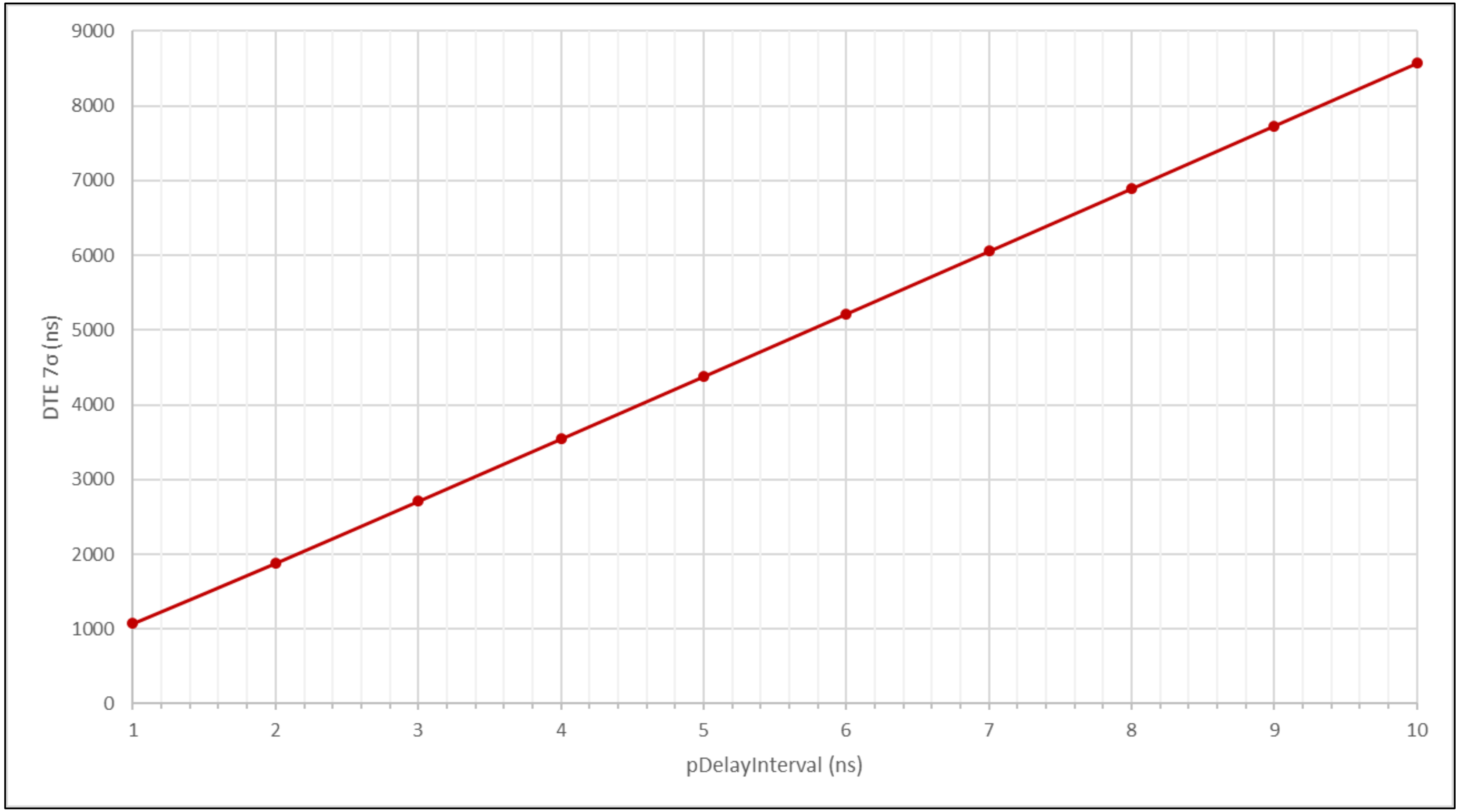
residenceTime Sensitivity – 1 s pDelay Interval



Input Errors		
GM Clock Drift Max	+0.6	ppm
GM Clock Drift Min	+0.6	ppm
Clock Drift (non-GM)	0.6	±ppm
Timestamp Granularity TX	4	±ns
Timestamp Granularity RX	4	±ns
Dynamic Time Stamp Error TX	4	±ns
Dynamic Time Stamp Error RX	4	±ns
Input Parameters		
pDelay Interval	**1000**	ms
pDelay Response Time	10	ms
residenceTime	**Variable**	ms
Correction Factors		
Mean Link Delay	0	%
Drift Rate	0	%
pDelayResponse → Sync	0	%
mNRR Smoothing	1	
Configuration		
Hops	100	
Runs	100,000	

Lower Residence Time is better.

residenceTime Sensitivity – 31.25 ms pDelay Interval

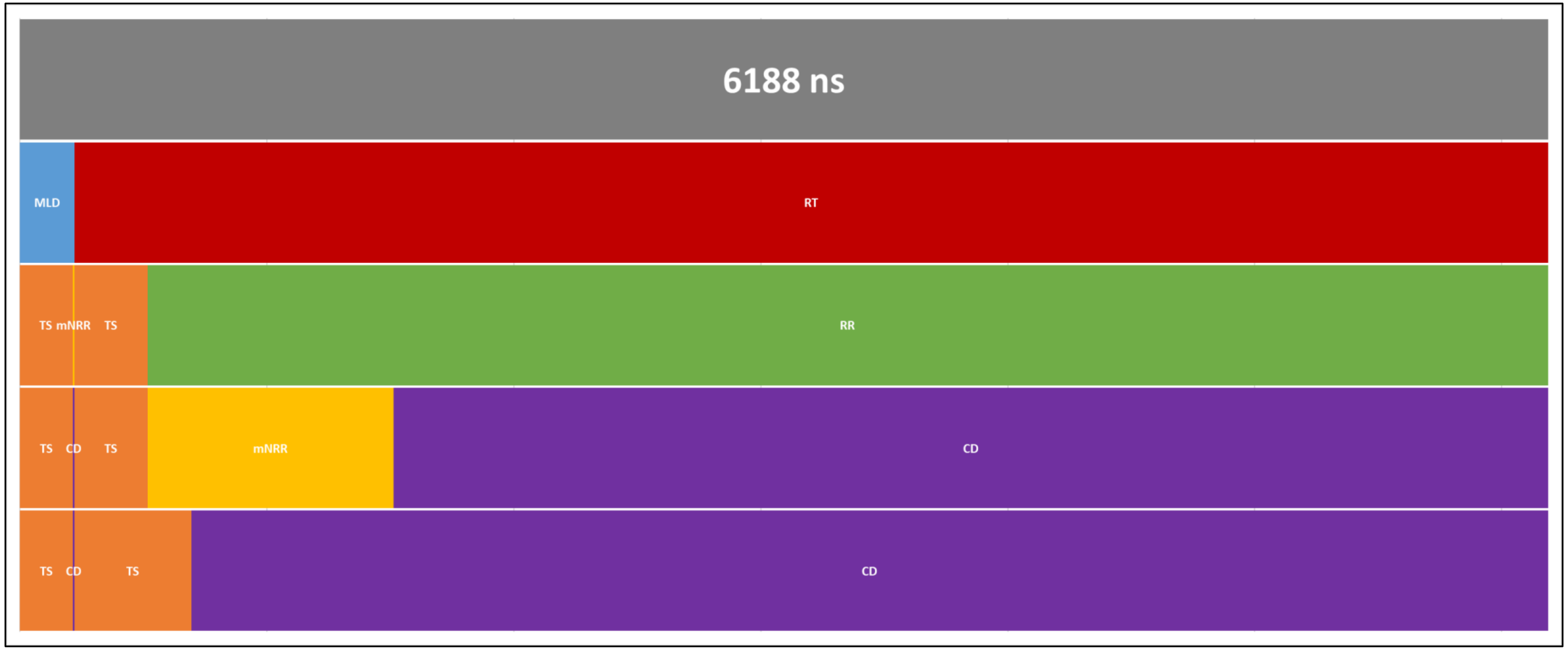


Input Errors		
GM Clock Drift Max	+0.6	ppm
GM Clock Drift Min	+0.6	ppm
Clock Drift (non-GM)	0.6	±ppm
Timestamp Granularity TX	4	±ns
Timestamp Granularity RX	4	±ns
Dynamic Time Stamp Error TX	4	±ns
Dynamic Time Stamp Error RX	4	±ns
Input Parameters		
pDelay Interval	**31.25**	ms
pDelay Response Time	10	ms
residenceTime	**Variable**	ms
Correction Factors		
Mean Link Delay	0	%
Drift Rate	0	%
pDelayResponse → Sync	0	%
mNRR Smoothing	1	
Configuration		
Hops	100	
Runs	100,000	

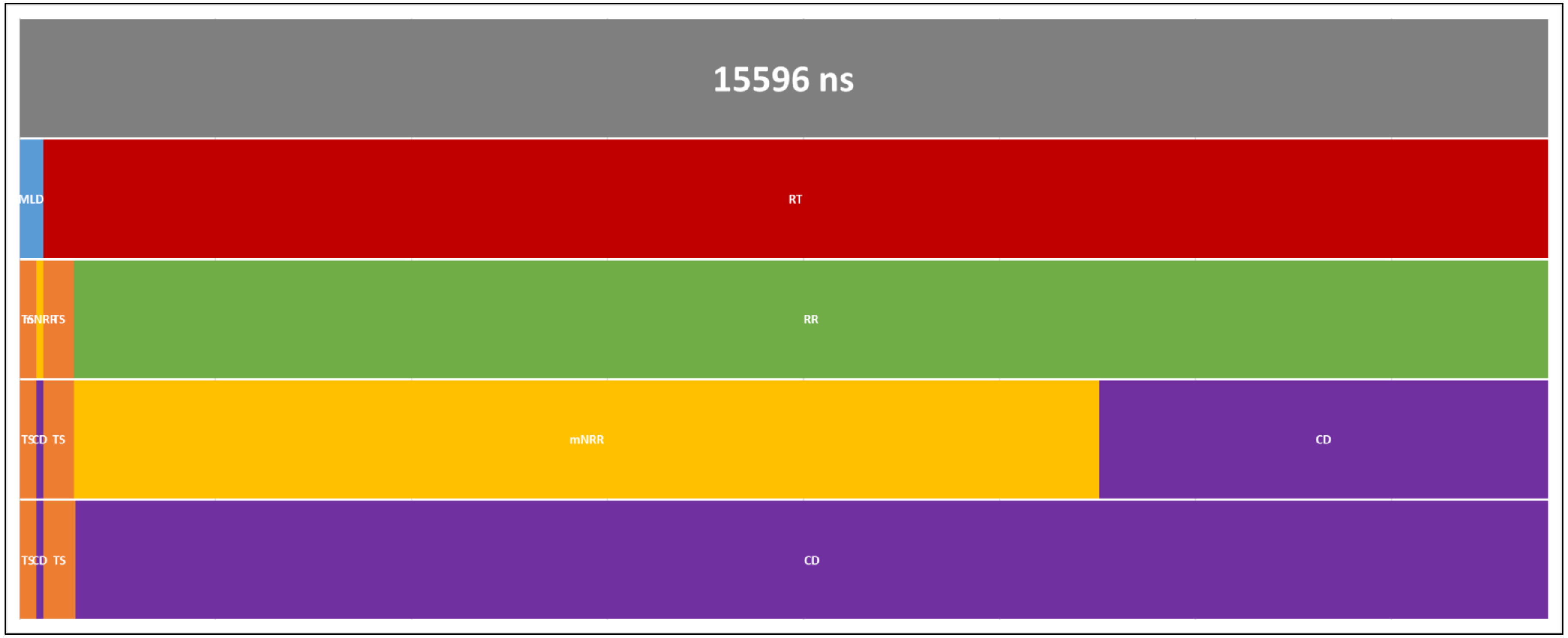
Lower Residence Time is better…
…proportional to the amount of error coming from Residence Time (vs Mean Link Delay)

Effect of Error Correction Measures

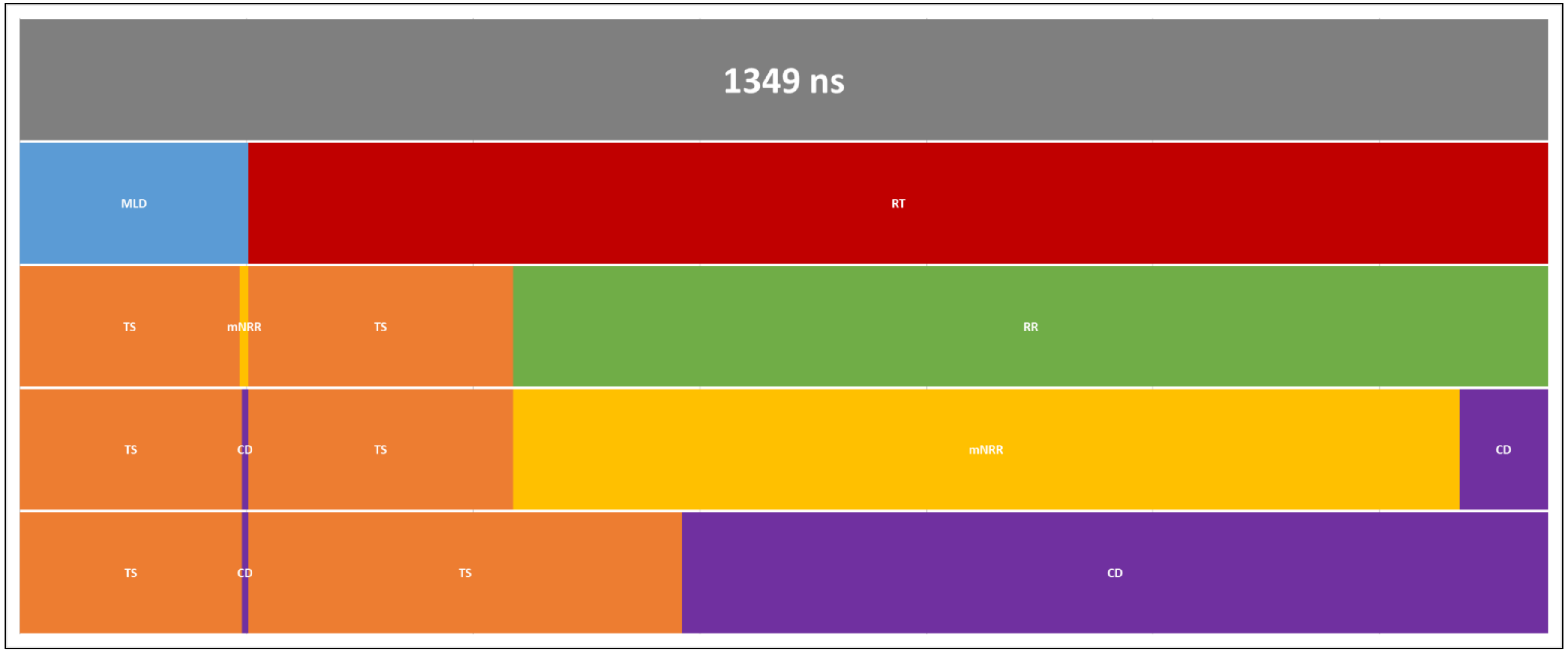
Default Values – No Error Correction Factors



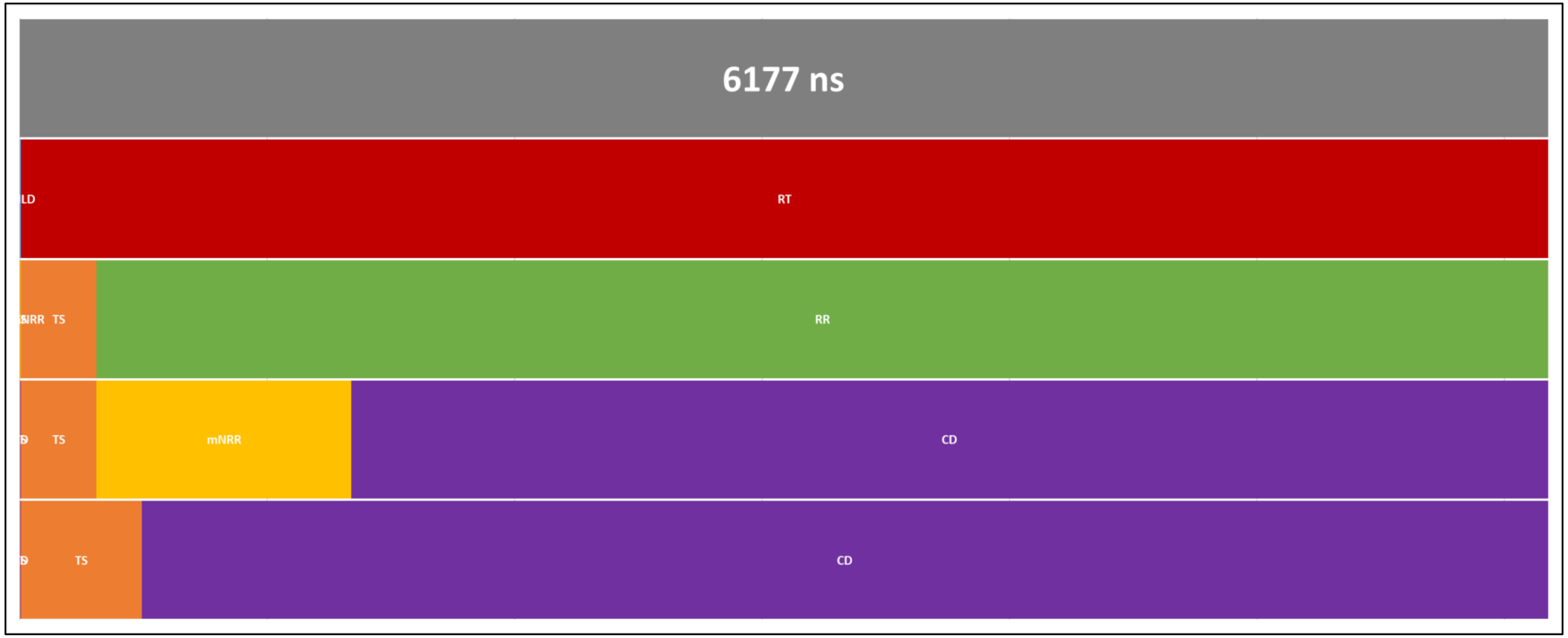
mNRRsmoothingN = 11



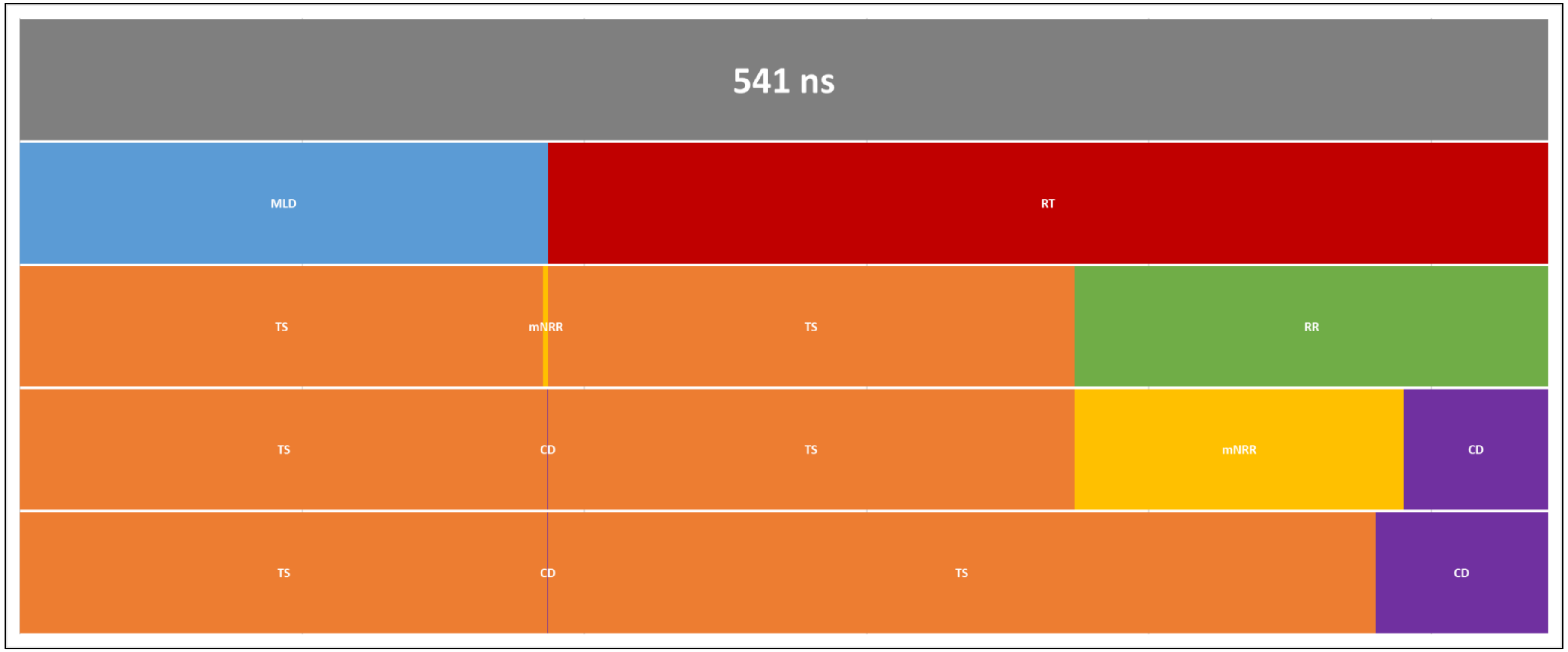
$\text{pDelayRespSync}_{\text{correction}} = 98\%$



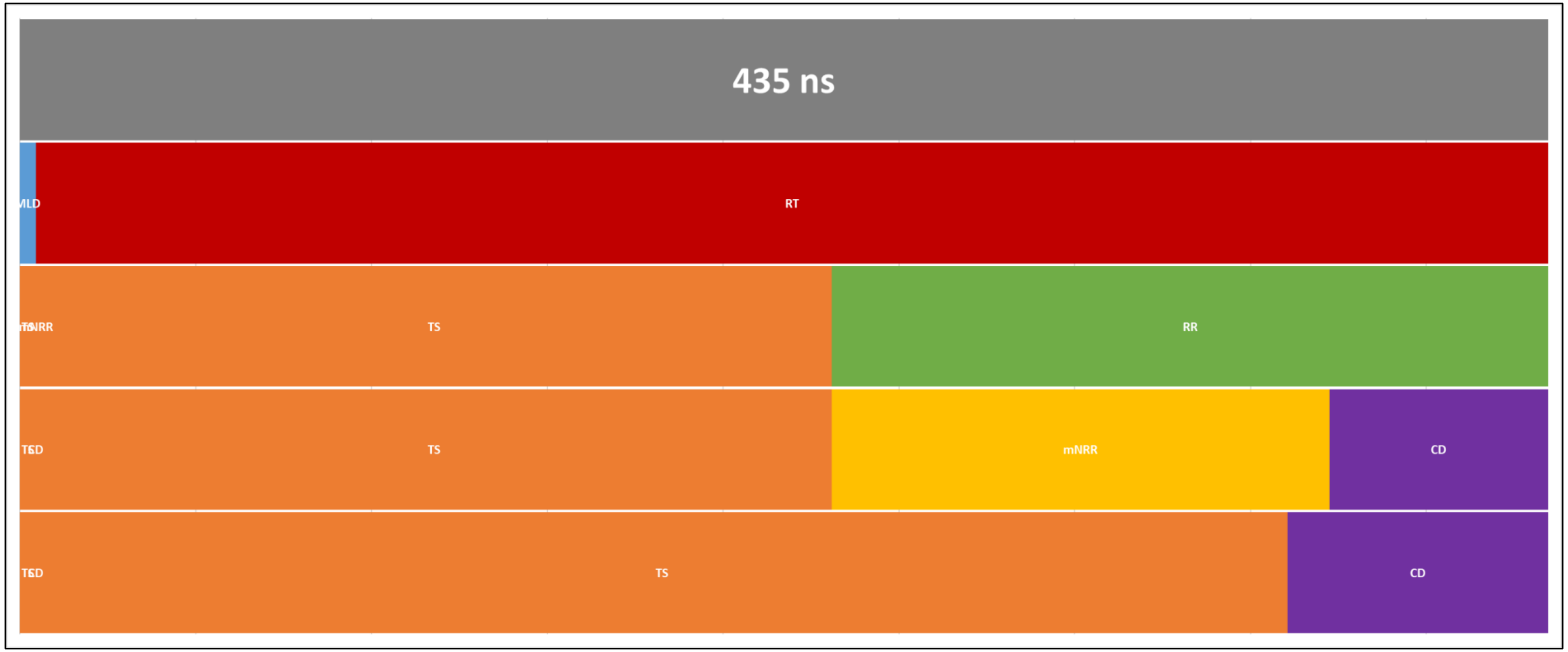
$\mathrm{mLinkDelayError}_{\mathrm{correction}} = 98\%$



driftRateError$_{correction}$ = 98%



mLinkDelayError$_{correction}$ & driftRateError$_{correction}$ = 98%



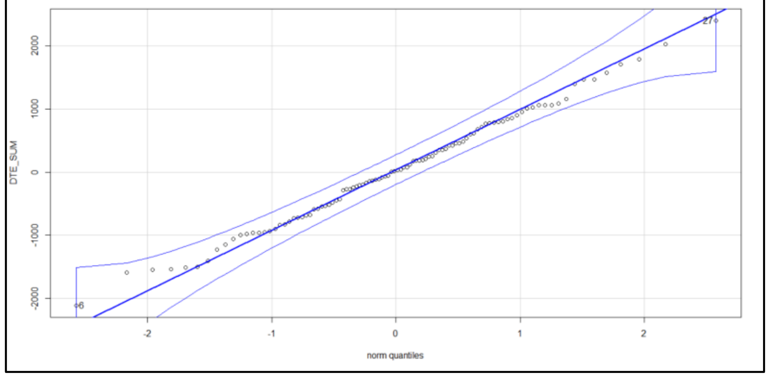
Recommendation & Next Steps

- Carry out Time Series simulations to validate
  - Effect of varying input parameters and sources of error.
  - Effect of applying correction factors
- Focus on averaging Mean Link Delay & Clock Drift Compensation
  - Mean Link Delay averaging should be straightforward, but startup may be an issue.
    - Can only be investigated via Time Series simulation
  - Clock Drift Compensation can be estimated by simply reducing Clock Drift
- I am planning another contribution in December on techniques for compensating for Clock Drift

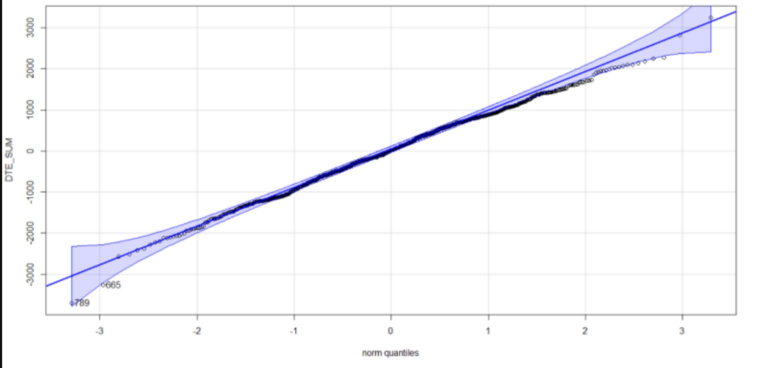
Backup Material

QQ Plots for Different Numbers of Runs

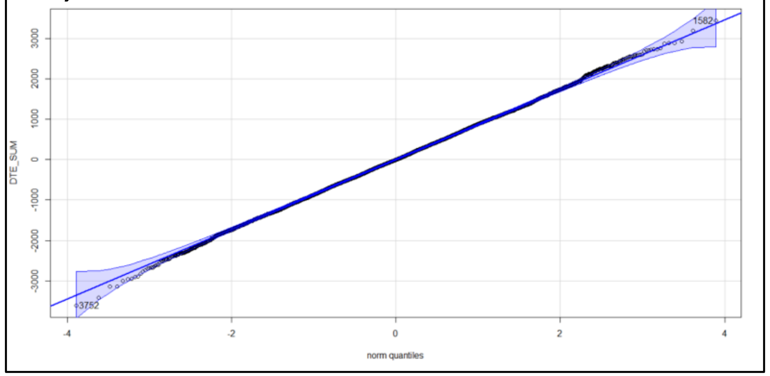
100 Runs



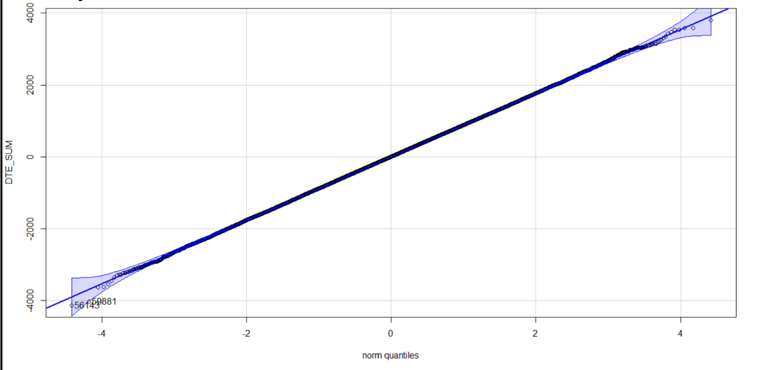
1,000 Runs



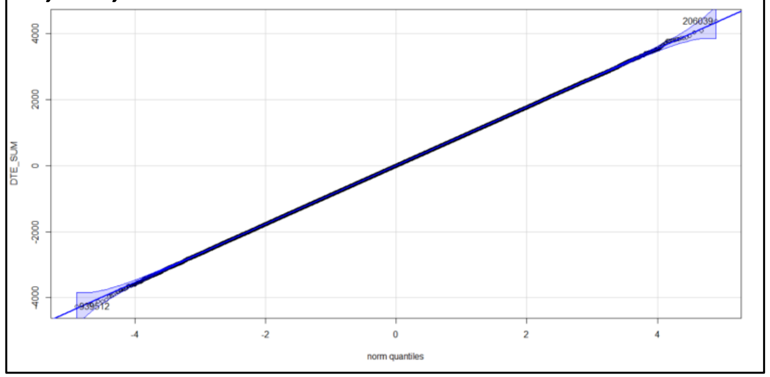
10,000 Runs



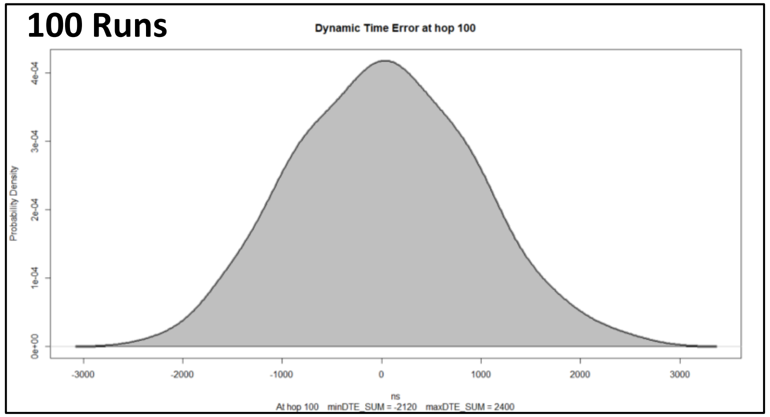
100,000 Runs

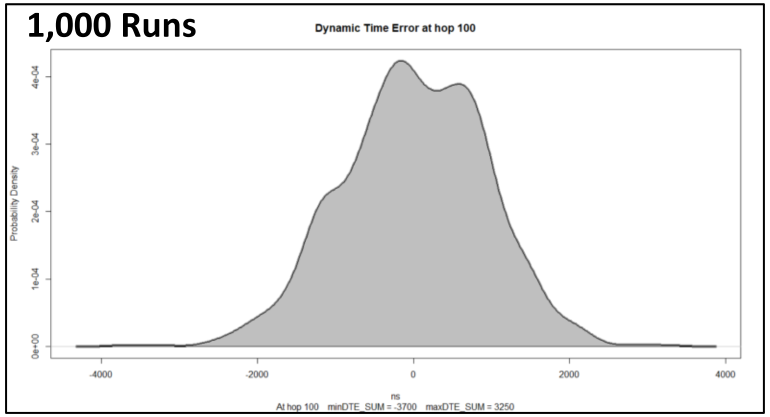


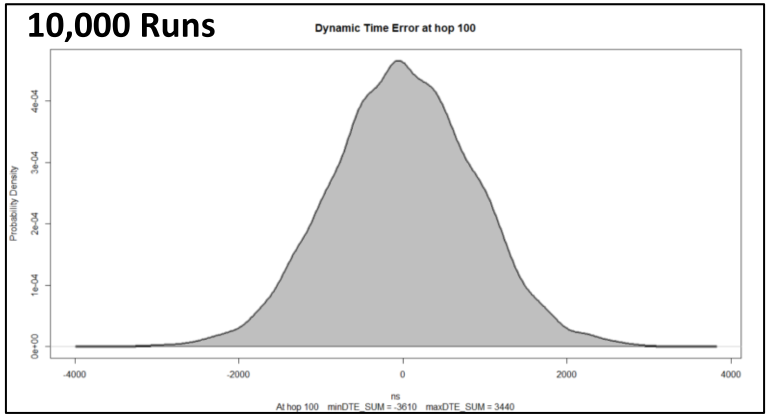
1,000,000 Runs

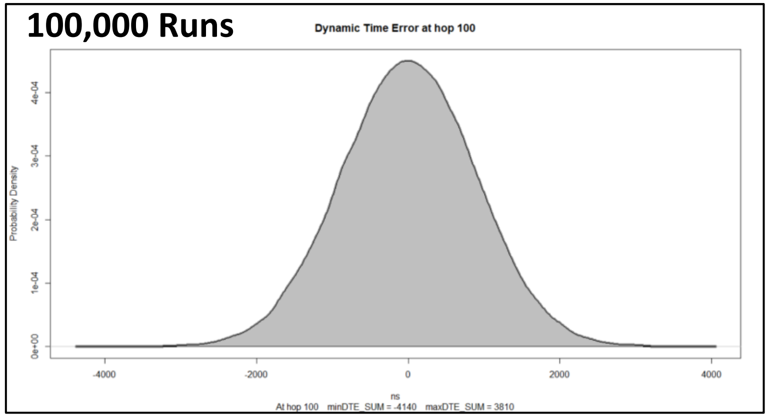


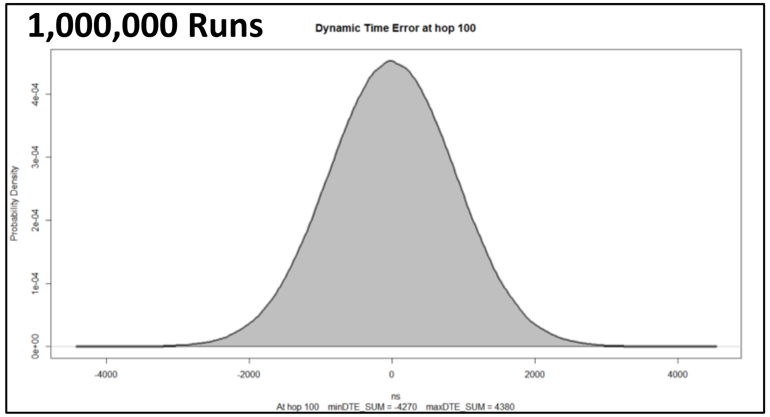
DTE Probability Density at Hop 100 for Different Numbers of Runs



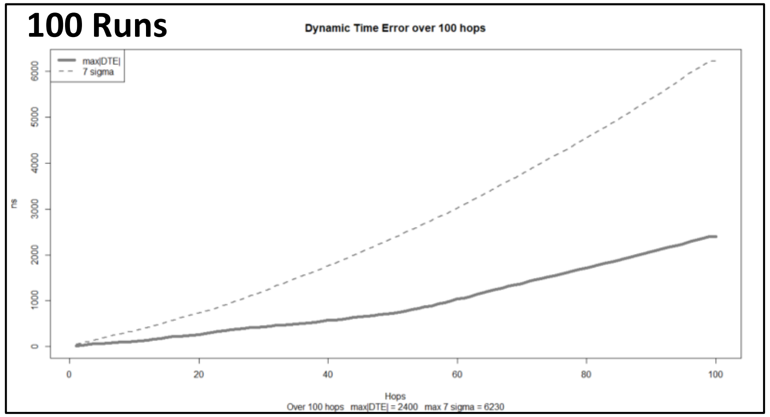


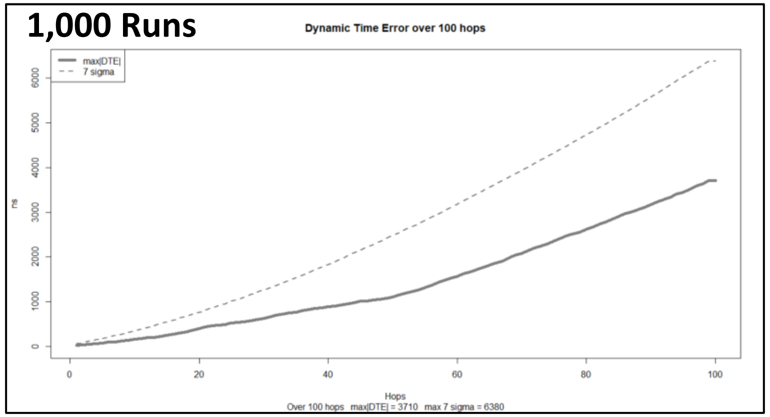


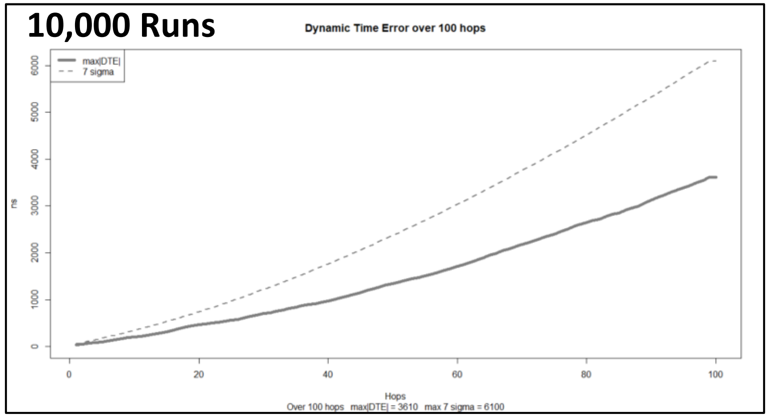


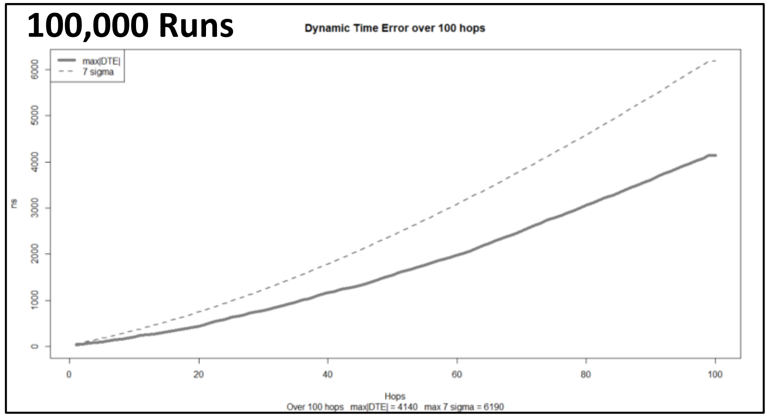


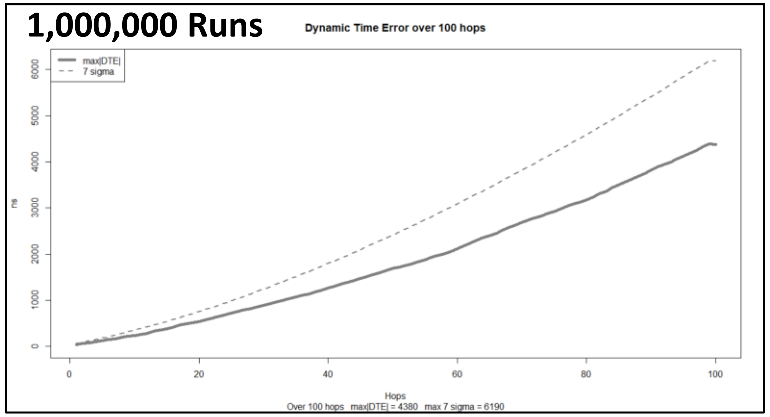
max|DTE| & 7σ of DTE Across 100 Hops for Different Numbers of Runs



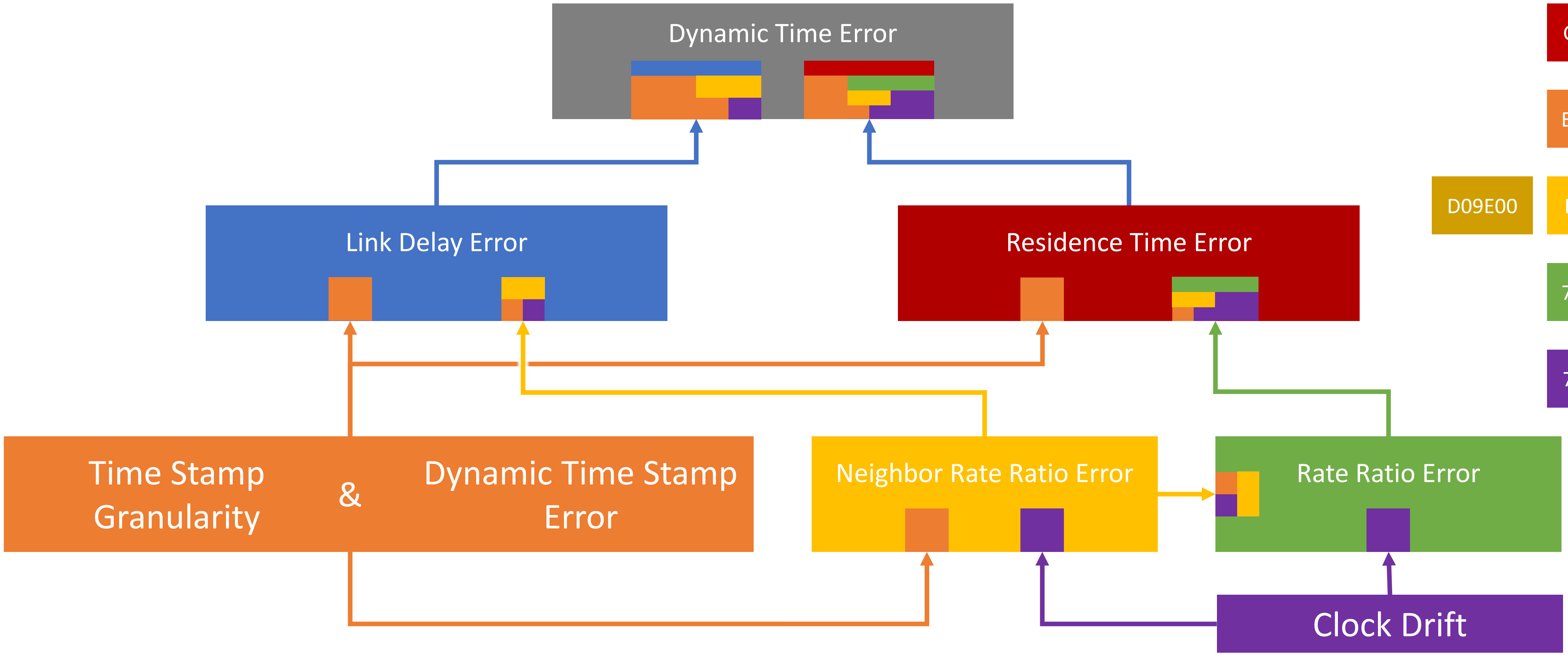








Graphics Colour Palette



	Lines	Areas	Backgrounds
	7F7F7F	BFBFBF	F2F2F2
	4472C4	B4C7E7	DAE3F3
	C00000	FF8181	FFBDBD
	ED7D31	F8CBAD	FBE5D6
D09E00	FFC000	FFE699	FFF2CC
	70AD47	C5E0B4	E2F0D9
	7030A0	C198E0	DFC9EF