New 60802 dTE_R Time Series Simulation Results for Comparison with Monte Carlo Simulation Results

Geoffrey M. Garner
Huawei (Consultant)

gmgarner@alum.mit.edu

IEEE 802.1 TSN TG
2022.03.10

Introduction – 1

- References [1], and previous presentations in the series (by the same author), present simulation results for max|dTE$_R$| (i.e., maximum absolute value of dynamic time error relative to the grandmaster (GM)) using a Monte Carlo approach that does not involve time-domain simulation
  - The model is approximate, but has the advantage of running several orders of magnitude faster than time domain simulations using probabilistic models
- In the discussion of [1] and preceding presentations, it was decided to compare corresponding results obtained using the Monte Carlo simulator and the time-domain simulator
- The specific simulation cases are summarized in slide 6 of [1], which is reproduced on the next slide here for convenience.
- The current presentation provides max|dTE$_R$| results for cases A, B, D, E, and F, obtained using the time-domain simulator, for comparison with results obtained using the Monte Carlo approach simulator
- The time domain results are based on 300 multiple, independent replications for each case

Introduction - 2

Proposed Time Series Simulations - Details (copied from [1])

		Errors			Parameter			Correction Factors	
Case	Reason	**Clock Drift Model – 40°C ↔ +85°C Hold for 30s at Each** (Each node's position in cycle distributed at random across 100% of Cycle)	Timestamp Granularity (ns)	Dynamic Timestamp Error (±ns)	pDelay Interval (ms)	Residence Time (ms)	pDelay Turnaround Time (ms)	Mean Link Delay Averaging	mNRR Smoothing Factor N
A	Baseline with previous assumptions	Ramp Rate 1°C / s (Cycle of 310 s)	8	8	31.25	1	1	Off	1
B	Verify optimised pDelayInterval		8	4	1000	10	10		
C					250	10	10		
D					31.25	10	10		
E	Verify effect of reduced Timestamp Error (reduced DTE when pDelay Interval is low, i.e. 31.25ms)		4	2	31.25	10	10		
F	Verify effect of reduced Clock Drift (reduced DTE when pDelay Interval is high, i.e. 1000ms)	Ramp Rate 0.5°C / s (Cycle of 560 s)	8	4	1000	10	10		

Timestamp Granularity and Dynamic Timestamp Error are uniform distributions
Sync Interval: 125ms
pDelay Interval variation is +0-30%; uniform distribution
Sync Interval variation is ±10%; gamma distribution with 90% probability of landing in the ±10% range
Note: 8ns Timestamp Granularity in Time Series Simulation is equivalent to ±4ns Timestamp Granularity Error in Monte Carlo Analysis
No difference between base (PHY related) propagation delay for pDelay and Sync messages

Introduction – 2

- The time domain simulator computes the time history of dTE at each node (PTP Instance) in a chain consisting of a GM, followed by 100 PTP Instances (nodes)
- Both unfiltered and filtered (by a PLL with specified parameters as given in the following slides) are computed at each node
- dTE_R relative to the GM (i.e., the first node) is computed by interpolating the time histories at each node to a common set of times, and then computing the difference at corresponding times
- Details of the simulation model are given in [2], [3], and [4], and references cited in those presentations.
- Some of the details, and the assumptions, are given on the following slides (some of which are copied from [4])

Model for Variable Sync Interval – 1

- IEEE Std 802.1AS-2020 requires in 10.7.2.3 (an analogous requirement is in 9.5.9.2 of IEEE Std 1588-2019):

When the value of syncLocked is FALSE, time-synchronization messages shall be transmitted such that the value of the arithmetic mean of the intervals, in seconds, between message transmissions is within ± 30% of $2^{currentLogSyncInterval}$. In addition, a PTP Port shall transmit time-synchronization messages such that at least 90% of the inter-message intervals are within ± 30% of the value of $2^{currentLogSyncInterval}$. The interval between successive time-synchronization messages should not exceed twice the value of $2^{portDS.logSyncInterval}$ in order to prevent causing a syncReceiptTimeout event. The PortSyncSyncSend state machine (see 10.2.12) is consistent with these requirements, i.e., the requirements here and the requirements of the PortSyncSyncSend state machine can be met simultaneously.

NOTE 1—A minimum number of inter-message intervals is necessary in order to verify that a PTP Port meets these requirements. The arithmetic mean is the sum of the inter-message interval samples divided by the number of samples. For more detailed discussion of statistical analyses, see Papoulis [B25].

Model for Variable Sync Interval – 2

- The above requirements do not specify the actual probability distribution; however, it was decided to model the Sync Intervals as being gamma-distributed
  - The gamma distribution is often used to model inter-message times in networks
  - The same model was used in simulations for the PTP Telecom Time Profile with full timing support from the network (ITU-T Rec. G.8275.1), see 11.2 and Eqs. (11-1) through (11-10) of [7])
- While both 802.1AS-2020 and 1588-2019 both allow variation in the duration of the Sync intervals up to $\pm$ 30% of the mean Sync interval, it was decided after the discussion of [4] to consider variations of $\pm\beta$, with β = 10%
- The shape and scale parameters of the gamma distribution are chosen such that the distribution has the desired mean and that 90% of the probability mass is within β of the mean
- The resulting gamma distribution has a shape parameter of 270.5532; the details of how this parameter is obtained and how the samples of the gamma distribution are generated are given in [4]

Model for Variable Pdelay Interval - 1

- IEEE Std 802.1AS-2020 has the following NOTE in 11.5.2.2 (it refers to the requirement in 9.5.13.2 of IEEE Std 1588-2019):

NOTE 3—The MDPdelayReq state machine ensures that the times between transmission of successive Pdelay_Req messages, in seconds, are not smaller than $2^{\text{currentLogPdelayReqInterval}}$. This is consistent with IEEE Std 1588-2019, which requires that the logarithm to the base 2 of the mean value of the interval, in seconds, between Pdelay_Req message transmissions is no smaller than the interval computed from the value of the portDS.logMinPdelayReqInterval member of the data set of the transmitting PTP Instance. The sending of Pdelay_Req messages is governed by the LocalClock and not the synchronized time (i.e., the estimate of the Grandmaster Clock time). Since the LocalClock frequency can be slightly larger than the Grandmaster Clock frequency (e.g., by 100 ppm, which is the specified frequency accuracy of the LocalClock; see B.1.1), it is possible for the time intervals between successive Pdelay_Req messages to be slightly less than $2^{\text{currentLogPdelayReqInterval}}$ when measured relative to the synchronized time.

- However, the actual requirement in 9.5.13.2 of IEEE 1588 is:

Subsequent Pdelay_Req messages shall be transmitted such that the value of the arithmetic mean of the intervals, in seconds, between Pdelay_Req message transmissions is not less than the value of $0.9 \times 2^{\text{portDS.logMinPdelayReqInterval}}$.

- This requirement will be satisfied even if the LocalClock is 100 ppm fast due to the factor of 0.9 (frequency offsets resulting from the temperature profile and frequency stability model of [3] are less than 100 ppm)

Model for Variable Pdelay Interval – 2

- IEEE 802.1AS and IEEE 1588-2019 do not specify the distribution for the Pdelay interval, nor do they specify the maximum amount that the actual intervals can exceed $2^{\text{portDS.logMinPdelayReqInterval}}$
- For the simulations, it was decided to use a uniform distribution over the range $[P, 1.3P]$, where P is $2^{\text{portDS.logMinPdelayReqInterval}}$

Assumptions for Temperature Profile ([1] – [4])

- The temperature history is assumed to vary between – 40°C and +85°C, at a rate of 1°C /s in cases A, B, C, D, and E (slide 3), and 0.5 °C /s in case F (slide 3); this takes 125 s or 250 s, respectively
- When the temperature is increasing and reaches +85°C, it remains at +85°C for 30 s
- The temperature then decreases from +85°C to – 40°C at a rate of 1°C /s (cases A, B, C, D, E) and 0.5 °C /s (case F); this takes 125 s or 250 s, respectively
- The temperature then remains at – 40°C for 30 s
- The temperature then increases to +85°C at a rate of 1°C /s or 0.5 °C /s ; this takes 125 s or 250 s, respectively
- The duration of the entire cycle (i.e., the period) is therefore 310 s (cases A – E), or 560 s (case F)

Assumptions for Frequency Stability due to Temperature Variation

- The dependence of frequency offset on temperature is assumed to be as described in [4] and [5] of Reference [4] here
  - Specifically, the values a_0, a_1, a_2, and a_3 computed in [4] will be used in the cubic polynomial fit, and the resulting frequency offset will be multiplied by 1.1 (i.e., a margin of 10% will be used).
- The frequency stability data that this polynomial fit is based on is contained in the Excel spreadsheet attached to [5] of Reference [4] here
  - This data was provided by the author of [4] of Reference [5] here
- The time variation of frequency offset will be obtained from the cubic polynomial frequency dependence on temperature, and the temperature dependence on time described in the previous slide
  - The time variation of phase/time error at the LocalClock entity will be obtained by integrating the above frequency versus time waveform
  - The time variation of frequency drift rate at the LocalClock entity will be obtained by differentiating the above frequency versus time waveform

Assumptions on Relative Time Offsets of Phase Error Histories at Each Node

- Choose the phase of the LocalClock time error waveform at each node randomly in the range [0,T], at initialization, where T is the period of the phase and frequency variation waveforms (i.e., 310 s or 560 s)

Other Assumptions - 1

- Additional assumptions
  - Mean Sync interval: 125 ms
  - Mean Pdelay interval: 31.25 ms (cases A, D, E); 1000 ms (cases B, F); 250 ms (case C)
  - Timestamp granularity: 8 ns (modeled by truncating to next lower multiple of 8 ns)
  - Residence time: 1 ms (case A); 10 ms (cases B – F)
  - Pdelay turnaround time is the same as the residence time for the respective case
  - Dynamic timestamp error is taken to have a uniform distribution over ±e, where e is 8 ns (case A), 4 ns (cases B, C, D, F), and 2 ns (case E)
- Other assumptions are taken from slide 3 above or, if not indicated on slide 3, then from [4], and are summarized on the following slides
- Note that only cases A, B, D, E, F are simulated; case C is not simulated due to insufficient time

Other Assumptions - 2

Assumption/Parameter	Description/Value
Hypothetical Reference Model (HRM), see note following the tables	101 PTP Instances (100 hops; GM, followed by 99 PTP Relay Instances, followed by PTP End Instance
Computed performance results	(a) max$\lvert dTE_{R(k,\ 0)}\rvert$ (i.e., maximum absolute relative time error between node k (k > 0) and GM, both filtered (PLL filter output at each node) and unfiltered (input to PLL filter at each node)
Use syncLocked mode for PTP Instances downstream of GM	Yes
Endpoint filter parameters	K_pK_o = 11, K_iK_o = 65 (f_{3dB} = 2.5998 Hz, 1.288 dB gain peaking, ζ = 0.68219)
Simulation time	3150 s; discard first 50 s to eliminate any startup transient before computing max$\lvert dTE_{R(k,\ 0)}\rvert$ (i.e., 10 cycles of frequency variation after discard)

Other Assumptions - 3

Assumption/Parameter	Description/Value
Number of independent replications, for each simulation case	300
GM rateRatio and neighborRateRatio computation granularity	0 (i.e., we do not truncate when computing timestamp differences and ratios of differences, but use floating point arithmetic)
Mean link delay	500 ns
Link asymmetry	0
Any variable PHY delay in addition to the dynamic timestamp error described above is assumed to be zero	0

Other Assumptions - 4

- neighborRateRatio is computed using windows of size of 1
  - The difference is taken between respective timestamps of current Pdelay exchange and the previous Pdelay exchange

max|dTE$_R$| Results - 1

- max|TE$_R$| results are for 300 independent replications of simulations for each case
  - max|TE$_R$| is computed for each replication, but after discarding the first 50 s of each replication to remove any startup transient
  - The 300 values are sorted in ascending order
  - A 99% confidence interval for the 0.95 quantile of max|TE$_R$| is given by the interval between the 275th and 294th smallest sample (and the midpoint, i.e., the 285th smallest sample, is taken as a point estimate of the 0.95 quantile)
    - This is because the quantiles of independent samples of a population have a binomial distribution (this result has been used in previous presentations of multiple replication simulation results)
  - The maximum of each set of 300 independent results for each case is also obtained

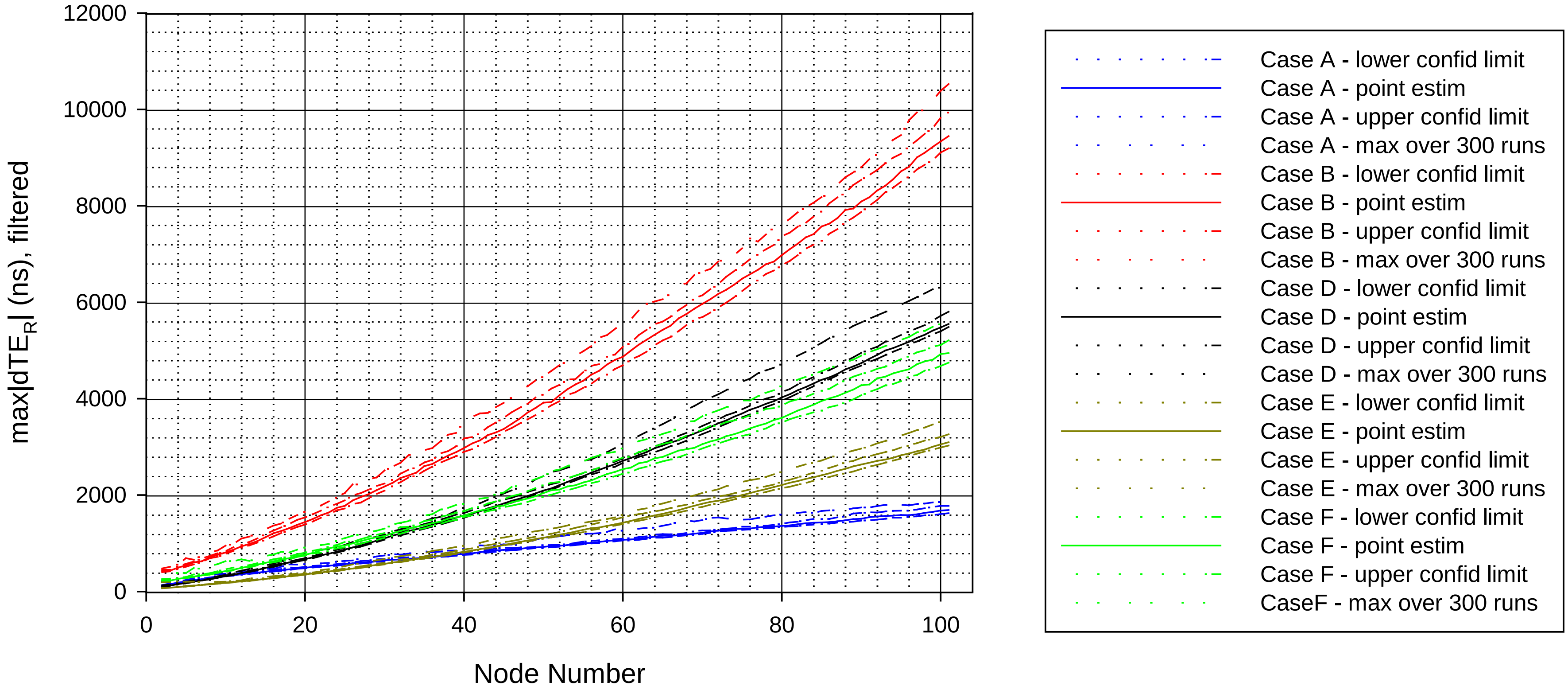
max|dTE_R| Results - 2

- Results for max|dTE_R|, relative to the GM, versus node number are summarized on the next four slides
  - Slide 18: Filtered max|dTE_R|, 99% confidence intervals and maxima over 300 runs, for all cases
  - Slide 19: Filtered max|dTE_R|, only maxima over 300 runs (to reduce clutter)
  - Slide 20: Unfiltered max|dTE_R|, 99% confidence intervals and maxima over 300 runs, for all cases
  - Slide 21: Unfiltered max|dTE_R|, only maxima over 300 runs and only cases A, D, and E (to reduce clutter)

max|dTE$_R$| Results - 3

Cases A, B, D, E, F - mult replic results - filt
GM time error modeled; dTE$_R$ is relative to GM

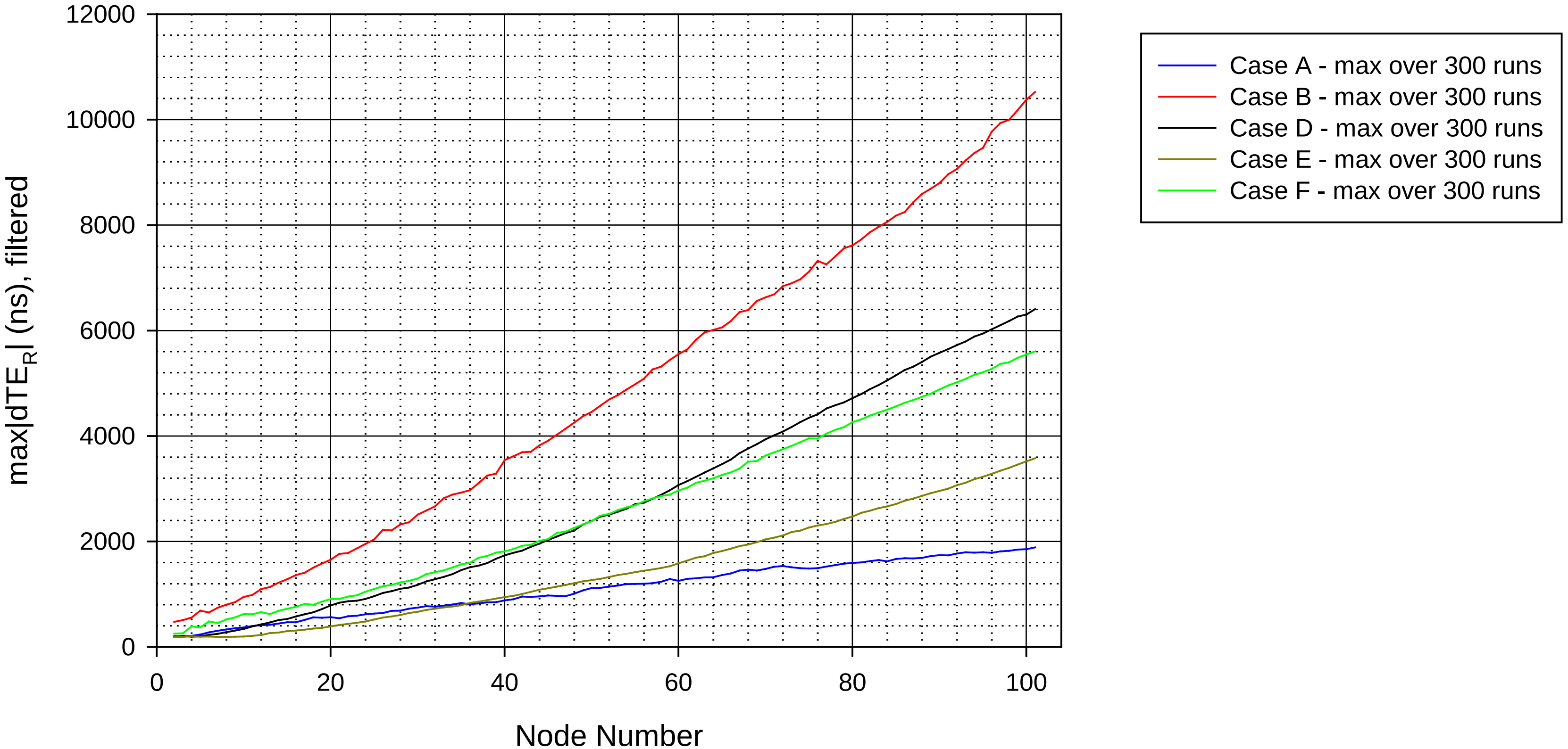
GM labeled node 1
neighborRateRatio measured with window of size 1 (N = 1) and no median calculation
KpKo = 11, KiKo = 65 (f3dB = 2.6 Hz, gain pk = 1.288 dB, zeta = 0.68219)
Sync interval variation: +/-10% with 90% probability (Gamma distribution)
Pdelay interval variation: 1.0 to 1.3 of input Pdelay interval (uniform distribution)
Timestamp granularity and dynamic timestamp error have uniform distributions



max|dTE$_R$| Results - 4

Cases A, B, D, E, F - mult replic results - filt
GM time error modeled; dTE$_R$ is relative to GM

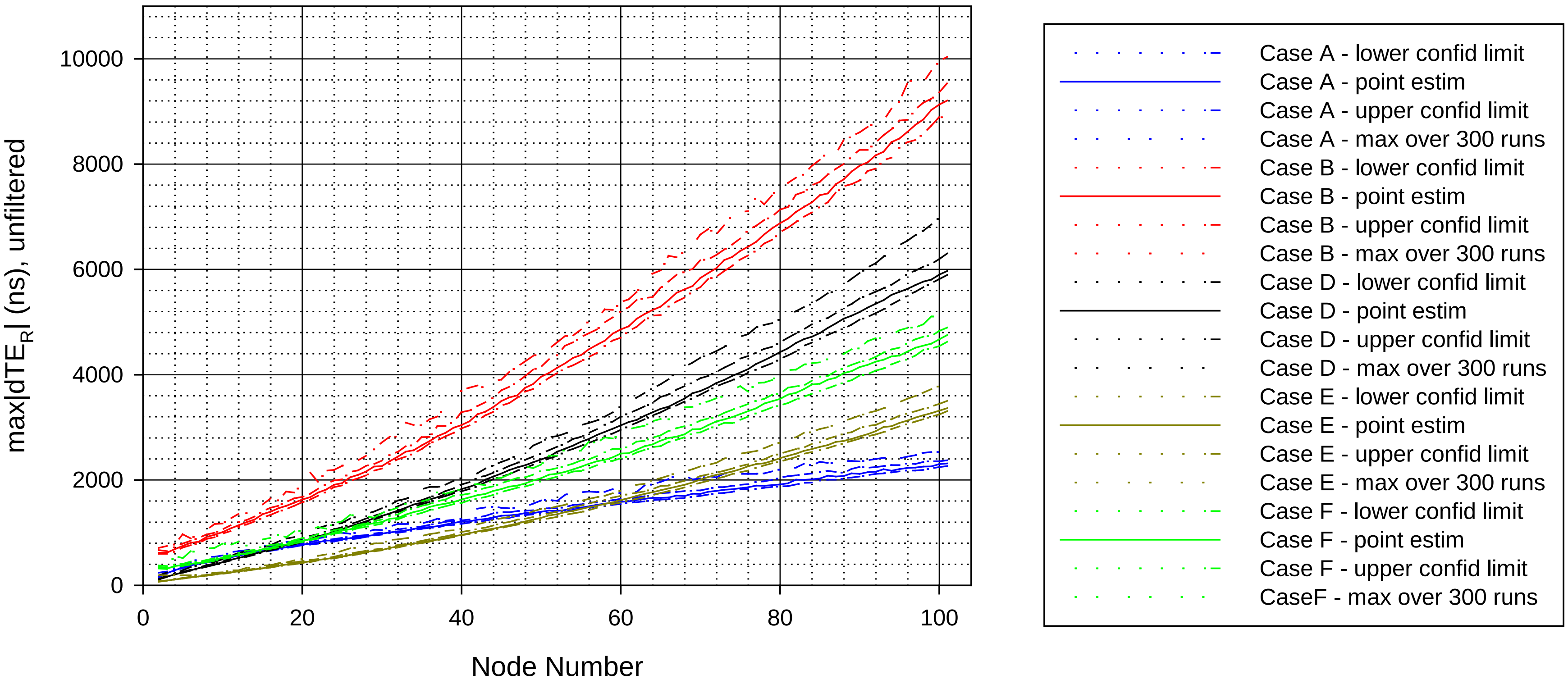
GM labeled node 1
neighborRateRatio measured with window of size 1 (N = 1) and no median calculation
KpKo = 11, KiKo = 65 (f3dB = 2.6 Hz, gain pk = 1.288 dB, zeta = 0.68219)
Sync interval variation: +/-10% with 90% probability (Gamma distribution)
Pdelay interval variation: 1.0 to 1.3 of input Pdelay interval (uniform distribution)
Timestamp granularity and dynamic timestamp error have uniform distributions



max|dTE$_R$| Results - 5

Cases A, B, D, E, F - mult replication results - unfil
GM time error modeled; dTE$_R$ is relative to GM

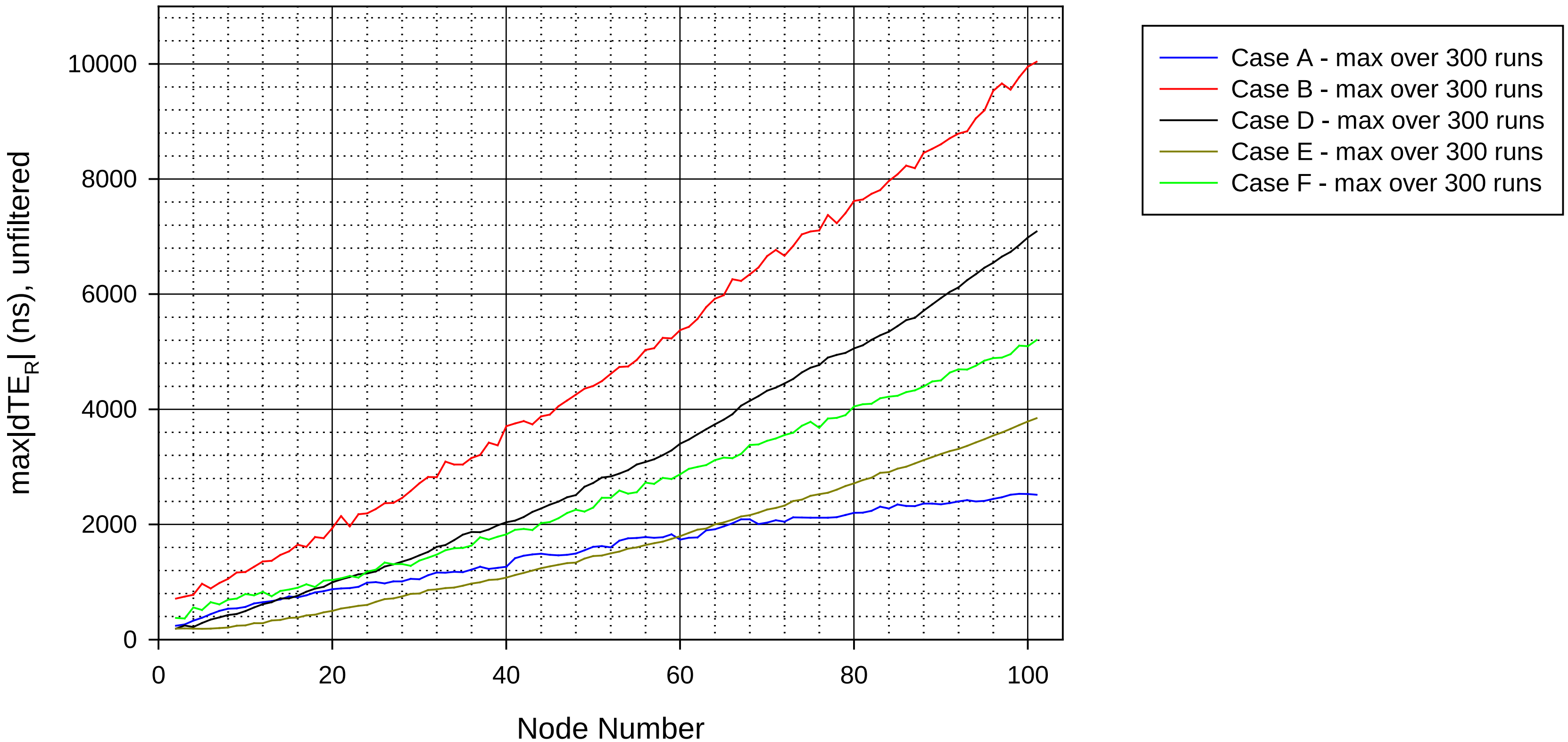
GM labeled node 1
neighborRateRatio measured with window of size 1 (N = 1) and no median calculation
KpKo = 11, KiKo = 65 (f3dB = 2.6 Hz, gain pk = 1.288 dB, zeta = 0.68219)
Sync interval variation: +/-10% with 90% probability (Gamma distribution)
Pdelay interval variation: 1.0 to 1.3 of input Pdelay interval (uniform distribution)
Timestamp granularity and dynamic timestamp error have uniform distributions



max|dTE$_R$| Results - 6

Cases A, B, D, E, F - mult replication results - unfil
GM time error modeled; dTE$_R$ is relative to GM

GM labeled node 1
neighborRateRatio measured with window of size 1 (N = 1) and no median calculation
KpKo = 11, KiKo = 65 (f3dB = 2.6 Hz, gain pk = 1.288 dB, zeta = 0.68219)
Sync interval variation: +/-10% with 90% probability (Gamma distribution)
Pdelay interval variation: 1.0 to 1.3 of input Pdelay interval (uniform distribution)
Timestamp granularity and dynamic timestamp error have uniform distributions

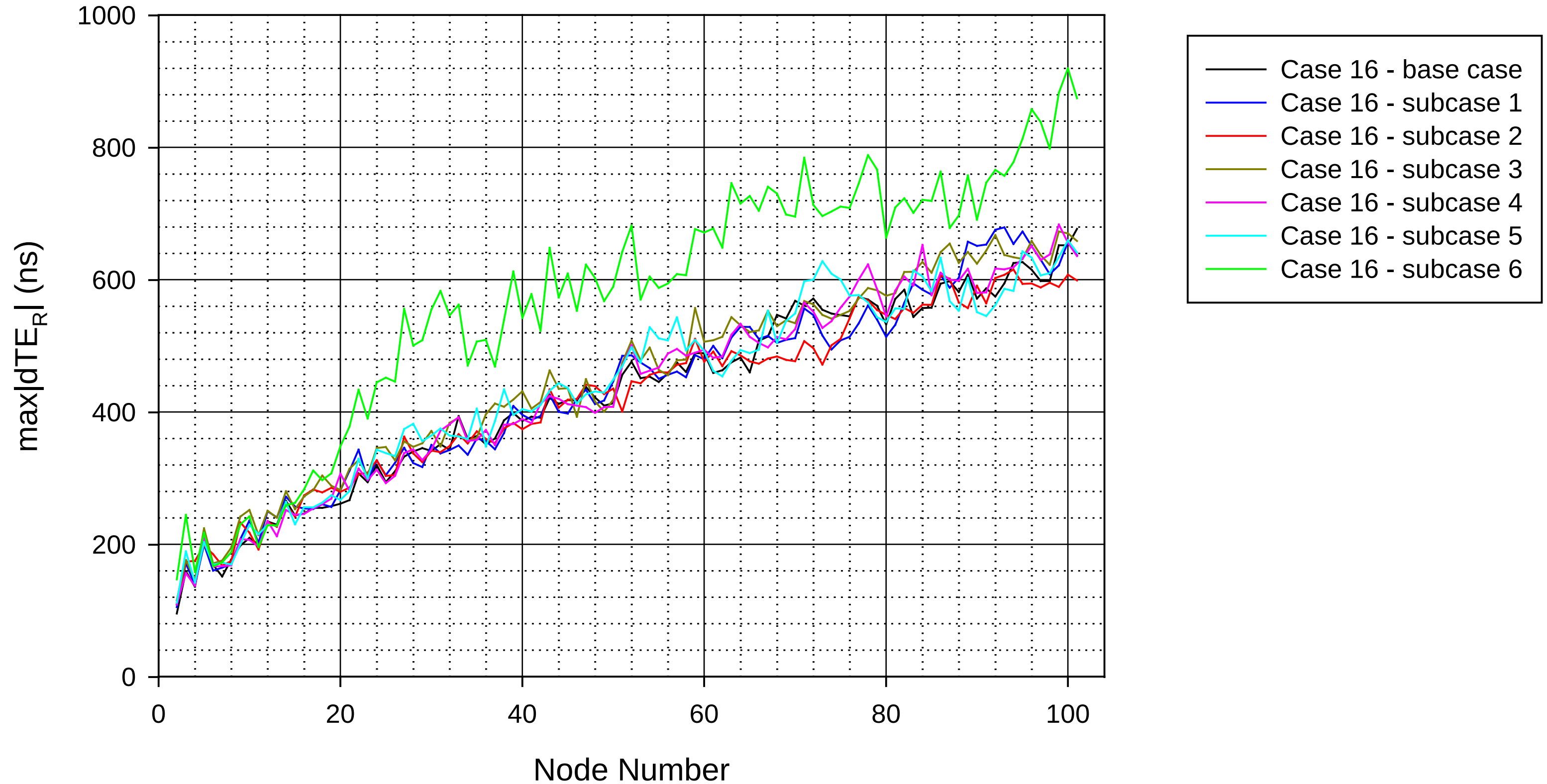


Summary of Results at Last Node

Case	Filtered/ Unfiltered	Lower (ns)	Point Est (ns)	Upper (ns)	Max (ns)
A	Filtered	1624	1688	1772	1888
	Unfiltered	2265	2315	2375	2515
B	Filtered	9190	9443	9945	10524
	Unfiltered	8870	9207	9535	10037
D	Filtered	5483	5546	5800	6407
	Unfiltered	5894	5969	6304	7089
E	Filtered	3024	3090	3256	3578
	Unfiltered	3307	3366	3503	3845
F	Filtered	4739	4940	5204	5605
	Unfiltered	4623	4754	4896	5204

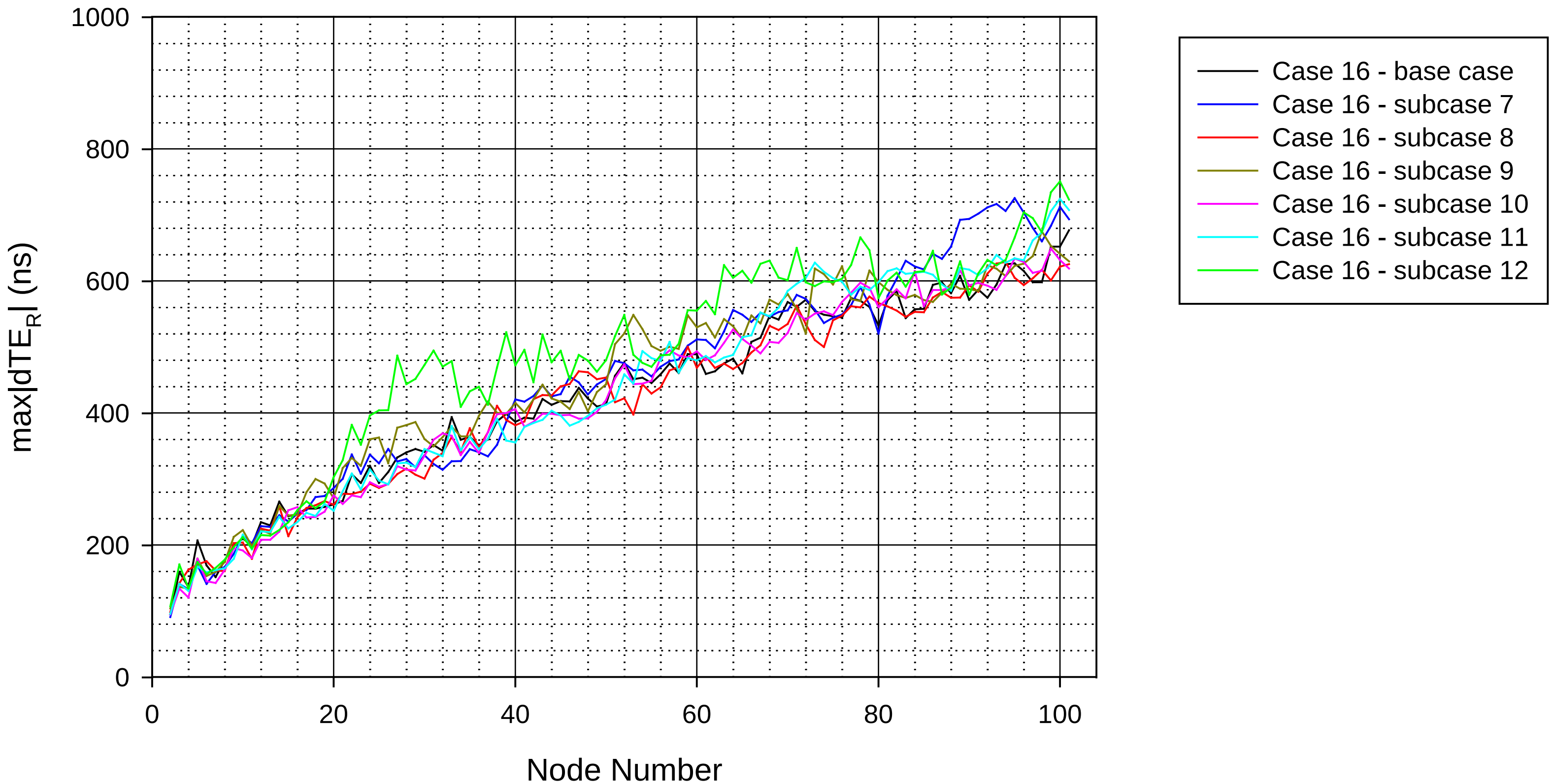
max|dTE_R| Results from[4] for comparison – 1

Case 16 - single replication results
Base case: no Sync or Pdelay interval variation
Subcases 1-3: Sync var (+/- 10, 20, 30%)
Subcases 4-6: Sync (+/- 10, 20, 30%) and Pdelay var (0-30%)
GM time error modeled
GM labeled node 1
Clock Model (all clocks): Frequency vs temperature stability and temperature vs time profile from [2]
Accumulate neighborRateRatio, which is measured with window of size 11 and median
KpKo = 11, KiKo = 65 (f3dB = 2.6 Hz, gain pk = 1.288 dB, zeta = 0.68219)



max|dTE$_R$| Results from[4] for comparison – 2

Case 16 - single replication results
Base case: no Sync or Pdelay interval variation
Subcases 7-9: Sync var (+/- 10, 20, 30%)
Subcases 10-12: Sync (+/- 10, 20, 30%) and Pdelay var (0-30%)
GM time error modeled
GM labeled node 1
Clock Model (all clocks): Frequency vs temperature stability and temperature vs time profile from [2]
Accumulate neighborRateRatio, which is measured with window of size 7 (11 for base case) and median
KpKo = 11, KiKo = 65 (f3dB = 2.6 Hz, gain pk = 1.288 dB, zeta = 0.68219)



max|dTE$_R$| Results from[4] for comparison – 3

Subcase	max|dTE$_R$|, 64 hops (ns)	max|dTE$_R$|, 100 hops (ns)
Base case	460	677
1	529	637
2	477	599
3	521	659
4	514	636
5	490	642
6	727	875
7	549	694
8	476	626
9	513	630
10	513	619
11	515	708
12	616	724

64 hops results are for node 65

100 hops results are for node 101

Base case is case 16 of [1], replication 1

Comparison of Monte Carlo and Time Series Simulation Results

Taken from [5], with corrected time series simulation results for cases B and F (and other minor corrections, e.g., mis-copied digits)

Compare Monte Carlo results with unfiltered Time Series Results

		Confidence Intervals & MAX →	Monte Carlo				Time Series – Unfiltered				Time Series – Filtered			
Case	Reason	Key Factor	Lower	Point	Upper	MAX	Lower	Point	Upper	MAX	Lower	Point	Upper	MAX
A	Baseline with previous assumptions	pDelayInterval 31.25ms; 1ms Residence Time & pDelay Turnaround; 8ns Dynamic Timestamp Error	2,543	2,657	2,774	2,941	2,265	2,315	2,375	2,515	1,624	1,688	1,772	1,887
							-10.9%	-12.9%	-14.4%	-14.5%				
B	Verify optimised pDelayInterval	pDelay Interval 1000ms	13,621	13,927	14,505	15,566	8,870	9,207	9,535	10,037	9,190	9,443	9,945	10,,524
							-34.9%	-33.9%	-34.3%	-35.5%				
C		pDelay Interval 250ms	4,175	4,285	4,498	4,609	**Not Run**							
D		pDelay Interval 31.25ms	6,326	6,469	6,710	6,915	5,894	5,969	6,304	7,089	5,483	5,546	5,800	6,407
							-6.8%	-7.7%	-6.1%	2.5%				
E	Verify effect of reduced Timestamp Error	Timestamp Errors halved pDelay Interval 31.25ms	3,623	3,684	3,915	3,996	3,307	3,366	3,503	3,845	3,024	3,090	3,256	3,578
							-8.7%	-8.6%	-10.5%	-3.8%				
F	Verify effect of reduced Clock Drift	Clock Drift halved pDelay Interval 1000ms	6,816	6,961	7,224	7,775	4,623	4,754	4,896	5,204	4,739	4,940	5,2045	13,087
							-32.2%	-31.7%	-32.2%	-33.1%				

Discussion of max|dTE_R| Results – 1

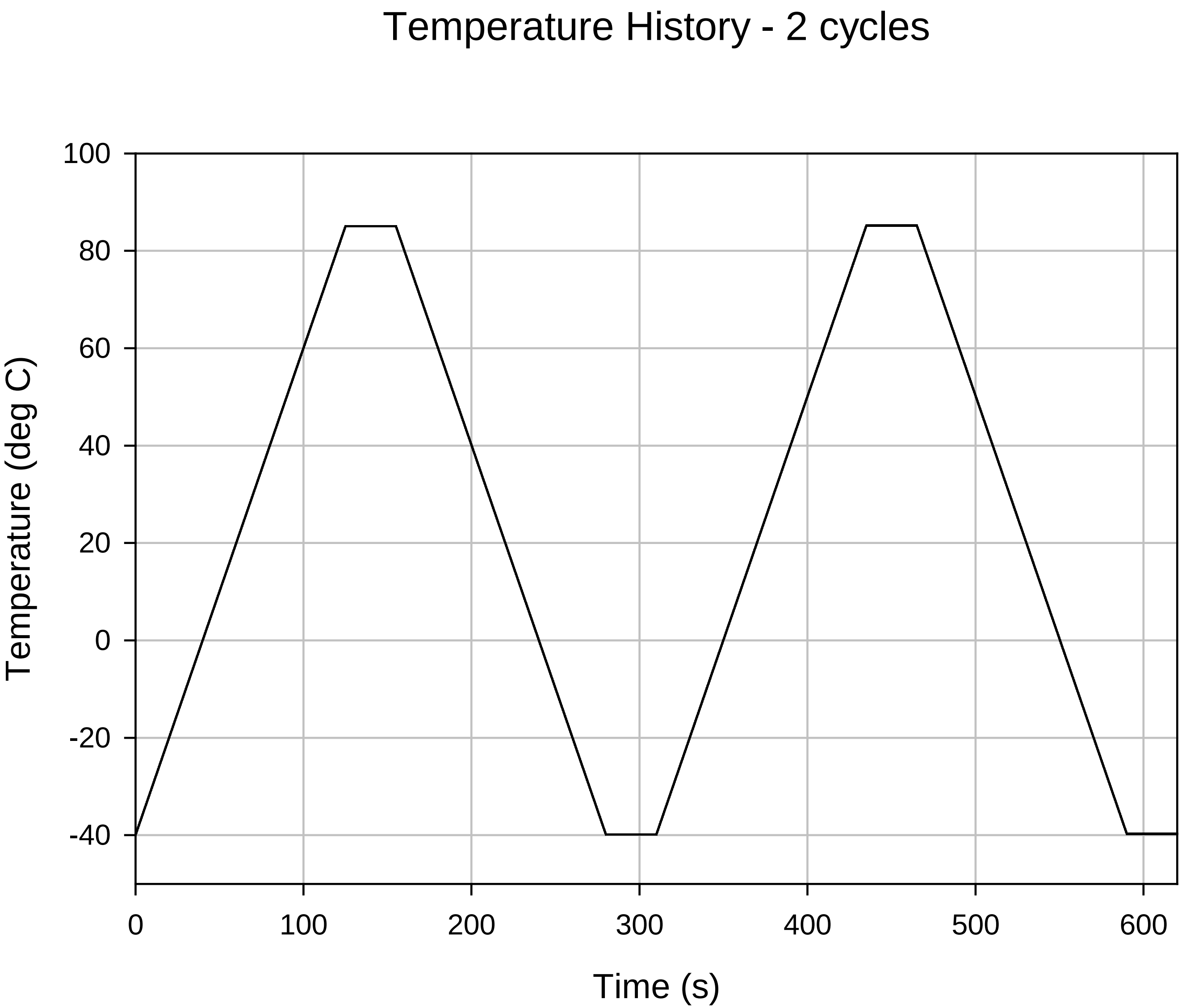
- Case A is the best case; the assumptions for this case are similar to previous simulation cases [4], except that
  - the mRR smoothing factor N is 1 instead of 7 with median calculation
  - The dynamic timestamp error has a uniform distribution over [-8 ns, +8 ns] instead of ±8 ns each with 0.5 probability
- The case A results are on the order of 3 – 4 times the previous results [4]; this is likely due to the smaller value of N
- Cases D and E are larger still
  - The case E results are on the order of 4 – 6.5 times the previous results [4]
  - The case D results are on the order of 8 – 12 times the previous results [4]
  - This is due to the much larger residence time and Pdelay turnaround time compared to the previous simulations (10 ms here versus 1 ms in [4])
    - Note that some of the previous simulation cases of [2] did use 10 ms residence time and had much larger results (12000 – 14000 ns over 100 hops; note that the dynamic timestamp error was ±8 ns each with 0.5 probability, which likely explains the larger results compared to case D here)

Discussion of max|dTE_R| Results – 2

- Max results for cases B and F are now smaller than in Revision 0 of this presentation, due to correcting an error in the post-processing code that performed interpolation (needed so that dTE relative to the GM could be computed)
- Unfiltered Time Series Simulation results are now smaller than corresponding Monte Carlo simulation results in almost all cases
  - The former are 30 – 35% smaller for cases B and F, 5 – 15% smaller for cases A, D, and E (except the maximum values for cases D and E are much closer for the Monte Carlo and Time Series simulations.
- One possible reason that the Monte Carlo simulation results are larger might be that different assumptions are used for the GM and Local Clock frequency offset and frequency drift rates
  - The Monte Carlo simulations choose the frequency drift rates randomly, and independently for each node, in the range ±1.5 ppm/s with 80% probability, and set the drift rate to 0 with 20% probability
  - The Time Series simulations use a temperature profile and frequency offset versus temperature stability that is a cubic polynomial
- We now consider this more carefully; some of the slides that follow are adapted from [6]

Model for Temperature Variation

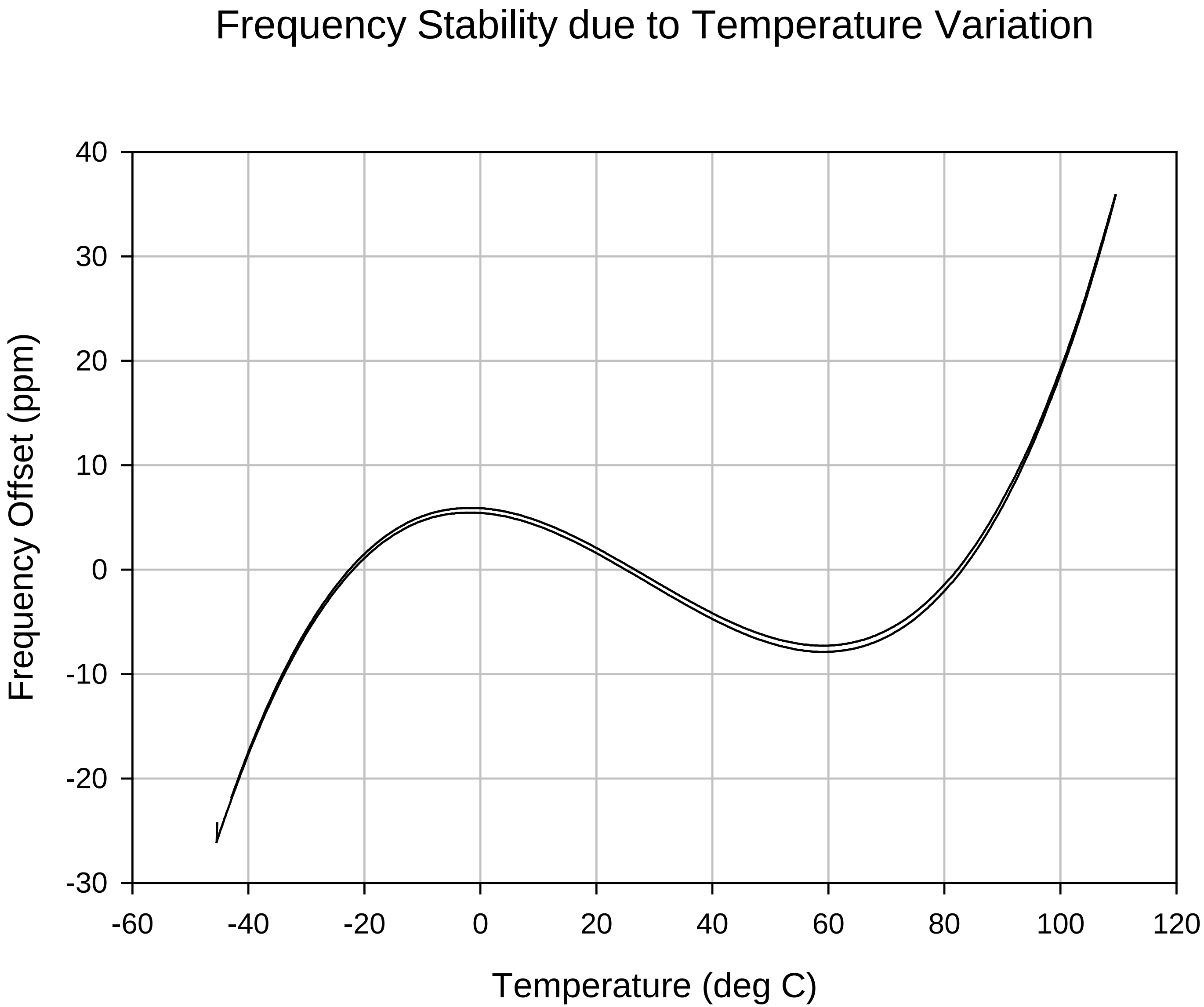
- **We focus on cases A – E (case F has a smaller rate of temperature change)**
- The temperature history of [2] – [4] is assumed to vary between – 40°C and +85°C, at a rate of 60°C /minute, or 1°C /s
- When the temperature is increasing and reaches +85°C, it remains at +85°C for 30 s
- The temperature then decreases from +85°C to – 40°C at a rate of 1°C /s; this takes 125 s
- The temperature then remains at – 40°C for 30 s
- The temperature then increases to +85°C at a rate of 1°C /s; this takes 30 s
- The duration of the entire cycle (i.e., the period) is therefore 310 s
- The temperature dependence is plotted over 620 s (2 cycles (periods)) on the next slide


Temperature History - 2 cycles
Temperature (deg C)
100
80
60
40
20
0
-20
-40
0
100
200
300
400
500
600
Time (s)

Temperature Dependence of Frequency – 1

- This slide and the next 6 slides are taken from [6] (and can be considered a review)
- The temperature dependence of frequency is based on data provided by Reference [3] of [6]
- Note that the dependence shows a small amount of hysteresis
- The plot has the characteristic of a third-order polynomial
- Note that temperature dependence of frequency is often referred to as frequency stability due to temperature variation

Temperature Dependence of Frequency - 2



Temperature Dependence of Frequency - 3

- To facilitate the use of this data in the subsequent calculations of this presentation and in future simulations, the LINEST function of Excel was used to obtain a third-order polynomial least-squares fit
- The polynomial is of the form

$$y(T) = a_3 T^3 + a_2 T^2 + a_1 T + a_0$$

where

T = temperature in °C

y = frequency offset in ppm

$a_3, a_2, a_1,$ and a_0 are coefficients to be determined from a least-squares fit

- The least-squares fit of LINEST produces:

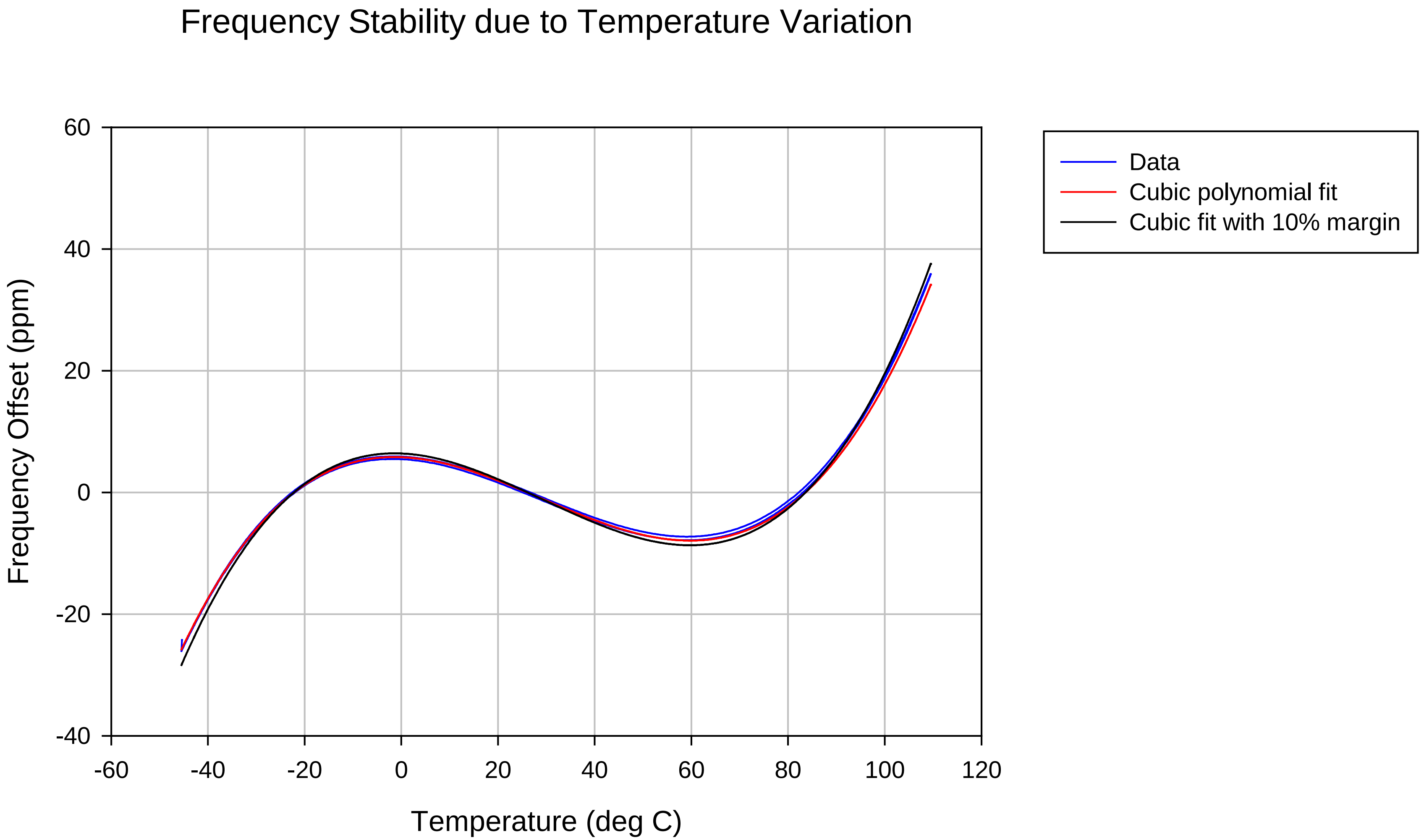
$$a_3 = 0.00012$$
$$a_2 = -0.0105$$
$$a_1 = -0.0305$$
$$a_0 = 5.73845$$

Temperature Dependence of Frequency - 4

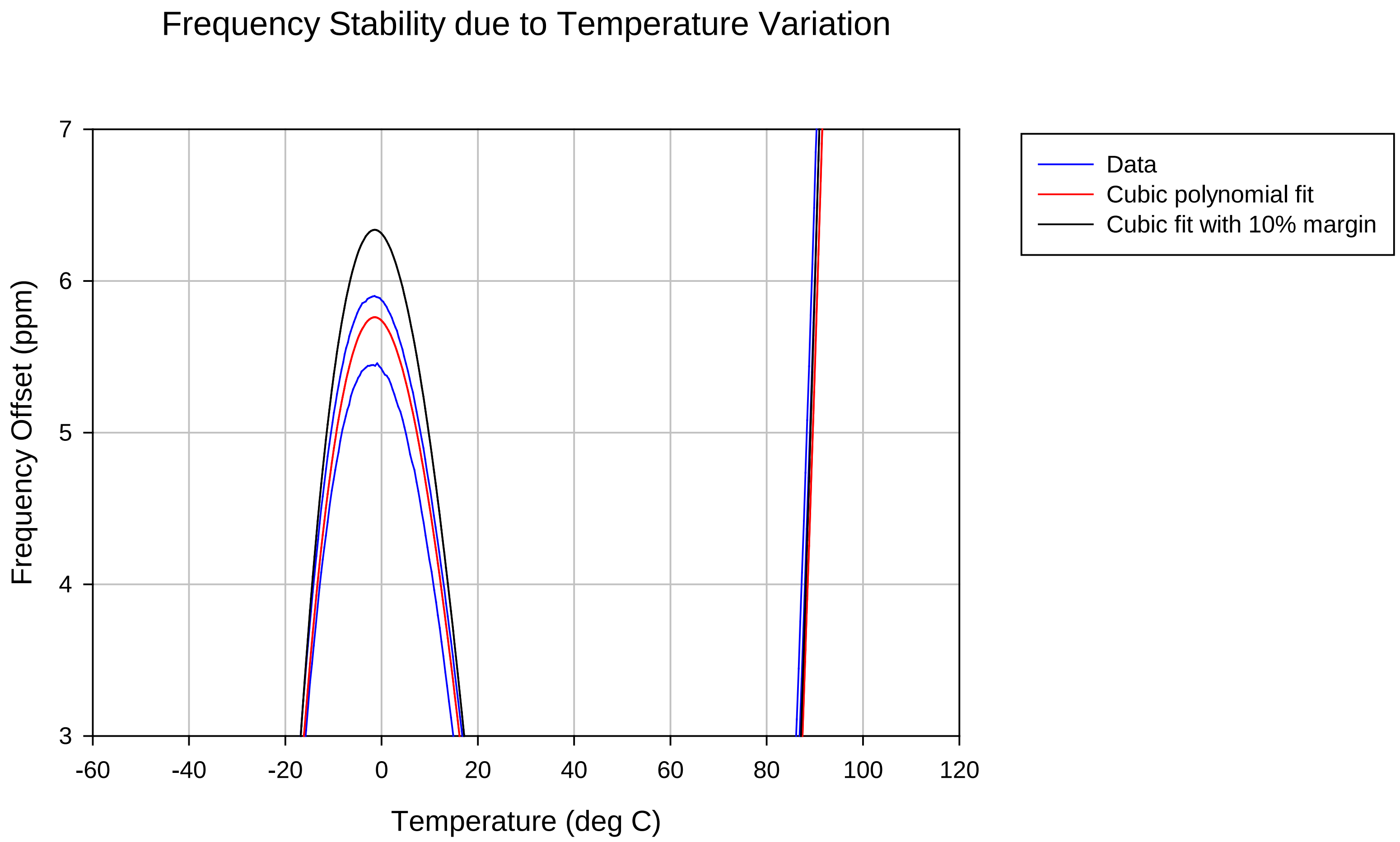
- The 3rd-order polynomial with these coefficients, along with the Excel file data, is shown plotted on the next slide
- At least visually, the results confirm that the cubic polynomial is a good fit
- In addition, a second polynomial, with 10% margin, is obtained by multiplying each of the coefficients a_i on the previous slide by 1.1
  - This second polynomial, with 10% margin, also is plotted
- The differences between the plots is not easily visible on the scale of the plots
- To see the differences more easily, a second plot is presented (on the slide after the next one) showing the detail of -18°C to +18 -18°C, and 87°C to 92 °C
  - On this scale, the factor of 1.1 that relates the curve with margin (black curve) and curve without margin (red curve) is easily seen
- As indicated earlier, the results for the time dependence of frequency drift rate, frequency offset, and phase offset will be shown for cases both with and without the 10% margin
- Finally, the rate of change of frequency with respect to temperature is plotted

$$\frac{dy}{dt} = 3a_3[T(t)]^2 + 2a_2[T(t)] + a_1$$

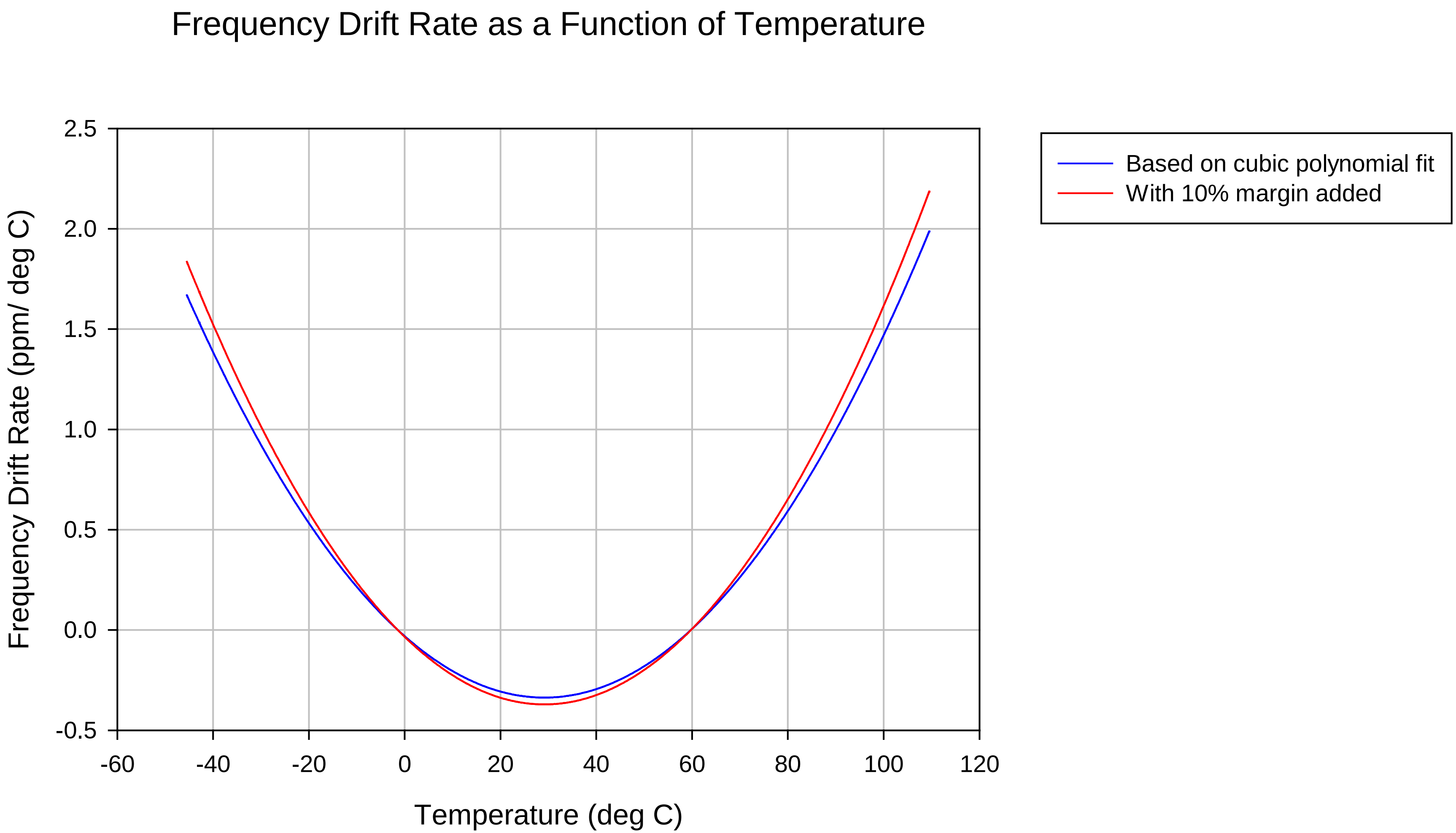
Temperature Dependence of Frequency - 5



Temperature Dependence of Frequency - 5



Temperature Dependence of Frequency - 5



Resulting Dependence of Frequency Offset on Time - 1

- As indicated previously, the frequency offset, y, as a function of temperature, T, is modeled as a cubic polynomial

$$y(T) = A[a_3T^3 + a_2T^2 + a_1T + a_0]$$

where

T = temperature in °C

y = frequency offset in ppm

$a_3, a_2, a_1,$ and a_0 are coefficients to be determined from a least-squares fit

A = factor applied corresponding to desired margin (e.g., $A = 1.1$ for 10% margin)

- Let the time dependence of temperature (on the previous three slides) be represented by $T(t)$; then the time-dependence of frequency offset is given by

$$y(t) = A\{a_3[T(t)]^3 + a_2[T(t)]^2 + a_1T(t) + a_0\}$$

- The time history of frequency offset, with and without 10% margin for the frequency stability data, is plotted on the next two slides for 12000 s and 2000 s, respectively

Resulting Dependence of Frequency Offset on Time - 1

- As indicated previously, the frequency offset, y, as a function of temperature, T, is modeled as a cubic polynomial

$$y(T) = A[a_3 T^3 + a_2 T^2 + a_1 T + a_0]$$

where

T = temperature in °C

y = frequency offset in ppm

$a_3, a_2, a_1,$ and a_0 are coefficients to be determined from a least-squares fit

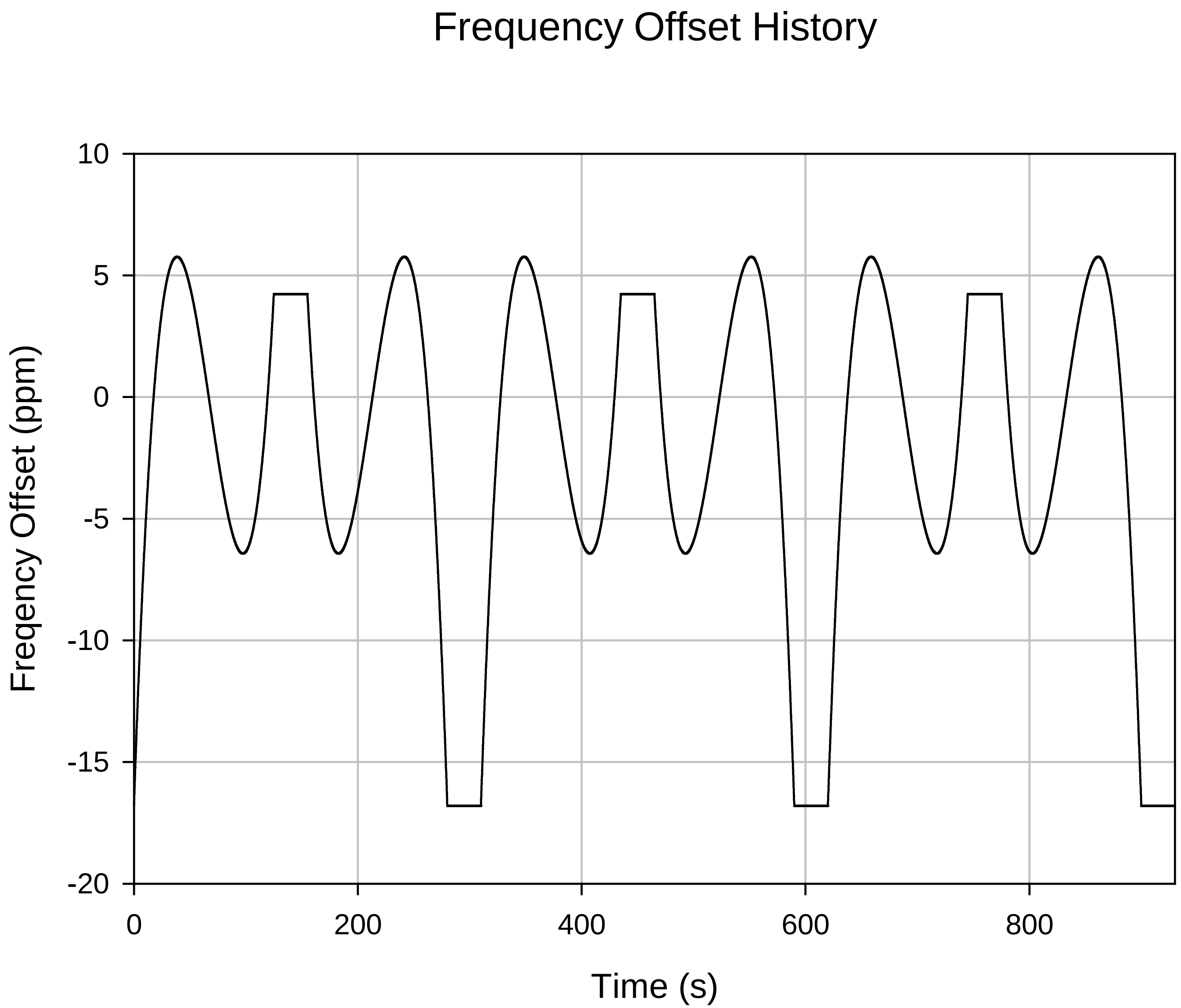
A = factor applied corresponding to desired margin (e.g., $A = 1.1$ for 10% margin)

- Let the time dependence of temperature (on the previous three slides) be represented by $T(t)$; then the time-dependence of frequency offset is given by

$$y(t) = A\{a_3[T(t)]^3 + a_2[T(t)]^2 + a_1 T(t) + a_0\}$$

- The time history of frequency offset is plotted on the next two slides for 3 cycles (930 s)
  - The 10% margin considered in [6] is small, and is ignored

Resulting Dependence of Frequency Offset on Time - 2



Resulting Dependence of Frequency Drift Rate on Time - 1

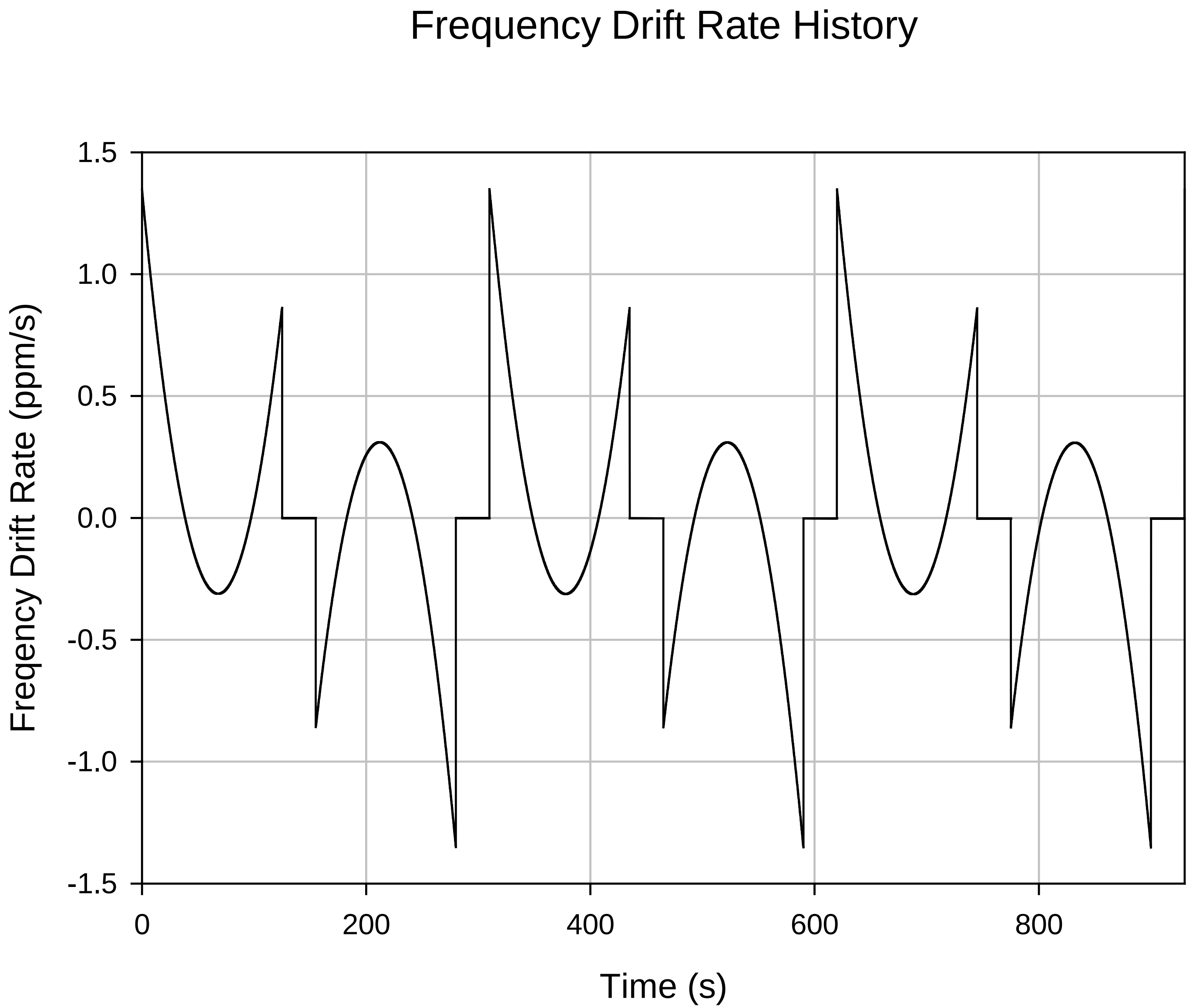
- The frequency drift rate is obtained by differentiating $y(t)$ with respect to time

$$y(t) = A\{a_3[T(t)]^3 + a_2[T(t)]^2 + a_1 T(t) + a_0\}$$

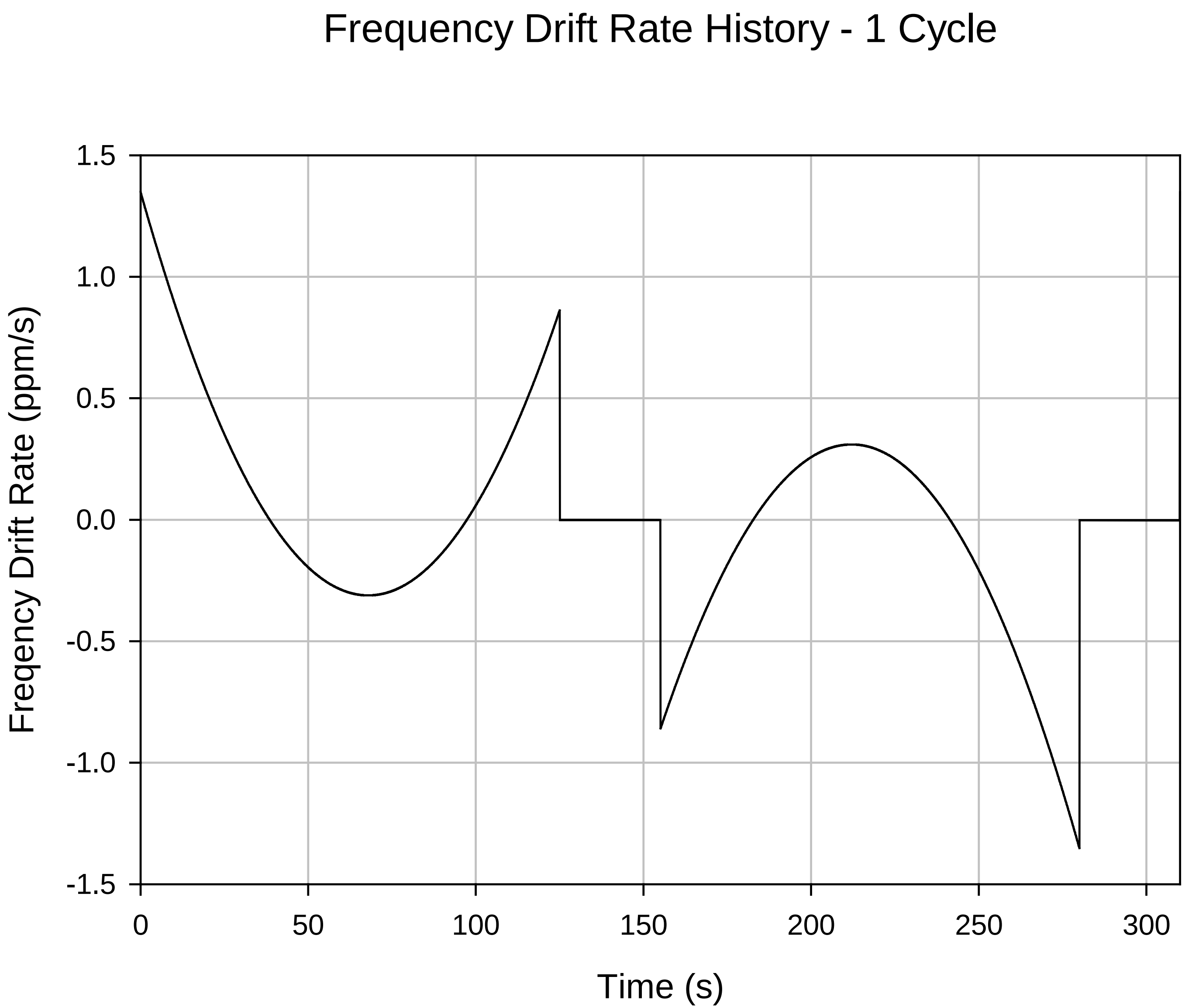
$$\frac{dy}{dt} = \frac{dy}{dT}\cdot\frac{dT}{dt} = A\{3a_3[T(t)]^2 + 2a_2[T(t)] + a_1\}\cdot\frac{dT}{dt}$$

- For the temperature profile above, dT/dt is 0 during the periods when temperature is constant at -40°C or +85°C, +1°C /s during the periods when temperature is increasing, and -1°C /s during the periods when temperature is decreasing
- The time history of frequency drift rate, is plotted on the next two slides for 3 cycles (930 s) and 1 cycle (310 s), respectively

Resulting Dependence of Frequency Drift Rate on Time - 2



Resulting Dependence of Frequency Drift Rate on Time - 3



Resulting Dependence of Frequency Drift Rate on Time - 4

- The maximum absolute value of frequency drift rate is approximately 1.35 ppm/s (i.e., the range is [-1.35 ppm, +1.35 ppm])
- However, for 19.35% of the cycle (60 s/310 s) the frequency drift rate is 0 because the temperature is constant for 60 s of the 310 s period
- In addition, the absolute value of the frequency drift rate exceeds 1.0 ppm/s for approximately 4.84% of each cycle (approximately 15 s out of 310 s)
- See backup for results for free-run phase history versus time (not important for the current discussion)

Equivalent Drift Rate Probability for Frequency Stability Model - 1

- The Monte Carlo simulations use the following probabilistic model for Cases A – E
  - Frequency drift rate is 0 ppm/s with 20% probability
  - Frequency drift rate is chosen randomly from a uniform distribution in the range [-1.5 ppm/s, +1.5 ppm/s], with 20% probability
  - Then, if R is a random variable representing the drift rate in ppm/s, the drift rate probability density function (pdf) can be written

$$p_R(x) = 0.2\delta(x) + \frac{0.8}{3} \qquad \text{for} \quad -1.5 \text{ ppm/s} \le x \le +1.5 \text{ ppm/s}$$

- One approach to deriving an equivalent probability density function for the frequency versus temperature stability model described above is to assume that a value of time is chosen randomly from a uniform distribution over the 310 second period
  - With this approach, the plot on slide 43 can be considered to represent the random variable R (frequency drift rate) as a function of the random variable t (time)

Equivalent Drift Rate Probability for Frequency Stability Model - 2

- The function given by the curve $R(t)$ on slide 43 can be used to derive $p_R(x)$ from $p_t(x)$
- However, the function $R(t)$ on slide 43 is not one-to-one, and the analytical derivation of $p_R(x)$ from the unform distribution $p_t(x)$ will be complicated
  - For each value of frequency drift rate in the range [-1.5 ppm, +1.5 ppm], the corresponding values of time must be found, and dR/dt must be computed at each of these points
  - This is complicated by the fact that the number of values of *t* corresponding to a given value of *R* is different depending on where *R* is in the [-1.5 ppm, +1.5 ppm] range
  - In addition, the period between 125 s and 155 s, and between 280 s and 310 s, where the frequency drift rate is zero, give rise to a delta function at zero (in the pdf) of strength (i.e., amplitude) 60/310 = 0.19355

Equivalent Drift Rate Probability for Frequency Stability Model - 3

- Fortunately, a Monte Carlo approach can be used, which is approximate but much simpler
  - Generate a random sample of time in the range [0, 310] s
  - Compute the corresponding frequency drift rate
  - Repeat this a large number of times; the resulting values can be considered samples of the random variable R
  - Use these samples to construct estimates of the pdf, probability function (histogram), or any other statistics of interest
- Note that this is the technique that would be used if this equivalent pdf were used to generate random samples of frequency drift rate in the Monte Carlo simulations for 60802 network dTE
- This approach was used to construct the pdf and probability function (histogram) on the following slides
  - 10^8 samples of frequency drift rate were generated

Equivalent Drift Rate Probability for Frequency Stability Model - 4

- The next slide shows the simulated (Monte Carlo approach) probability histogram for the frequency drift rate based on 10^8 samples, assuming the periodic time dependence described above and time chosen randomly from a uniform distribution over one period
- As indicated previously, the range of frequency drift rate is [-1.35 ppm/s, +1.35 ppm/s]
- The total range of 2.7 ppm/s is divided into 27 bins of size 0.1 ppm/s each
- Each of the 10^8 samples is placed on one of the bins, and on completion the number of samples in each bin is divided by 10^8
- The resulting histogram is shown on the next slide, and the height of each bar is the simulated probability that a frequency drift rate sample will be in the respective bin (range)
- The large peak at 0 ppm/s represents the probability that the frequency drift rate is zero; ideally, this would have height 0.2, but since the bin of nonzero width, it include the probability of being in a range of width 0.1 ppm/s about zero
  - With this added probability, the height is 0.26

Equivalent Drift Rate Probability for Frequency Stability Model - 5

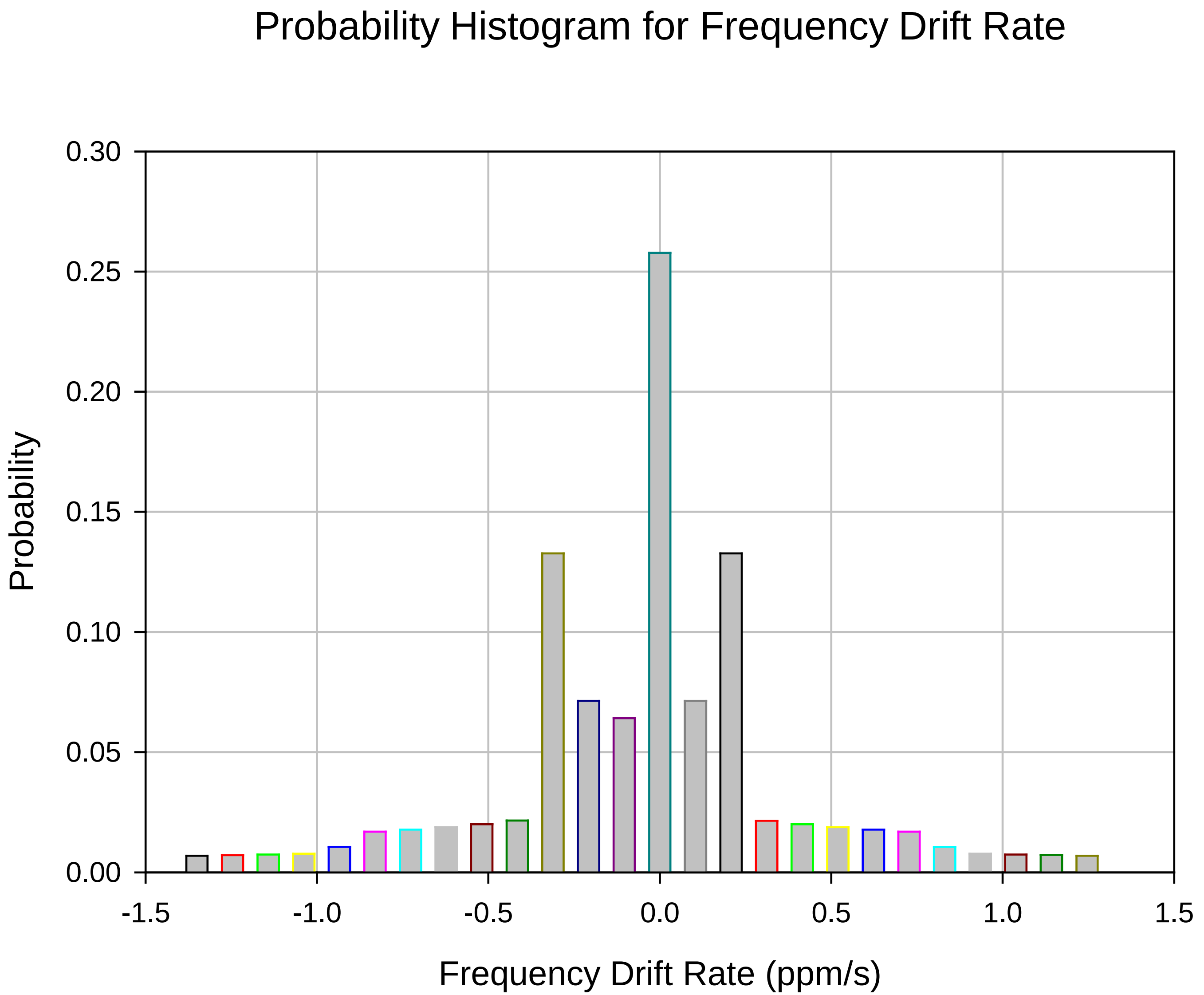
- The large side-peaks around 0.3 ppm/s and -0.3 ppm/s represent increased probability of these (and smaller absolute value) frequency drift rates
  - Examining the frequency drift rate time history for one period (slide 43), the two parabolas represent non-zero frequency drift rate when the temperature is either increasing or decreasing
  - For frequency drift rates in the range [-0.3 ppm/s, +0.3 ppm/s], both the periods of increasing and decreasing temperature (i.e., both parabolas) contribute to the probability
  - For frequency drift rates outside this range, only one parabola contributes to the probability
- Slide 51 shows a probability histogram, but with 10,0000 bins (and 10^8 samples
  - Now, the height (probability) for each bin is smaller than on slide 50, because the bins are narrower
  - However, the probabilities do still sum to 1

Equivalent Drift Rate Probability for Frequency Stability Model - 6

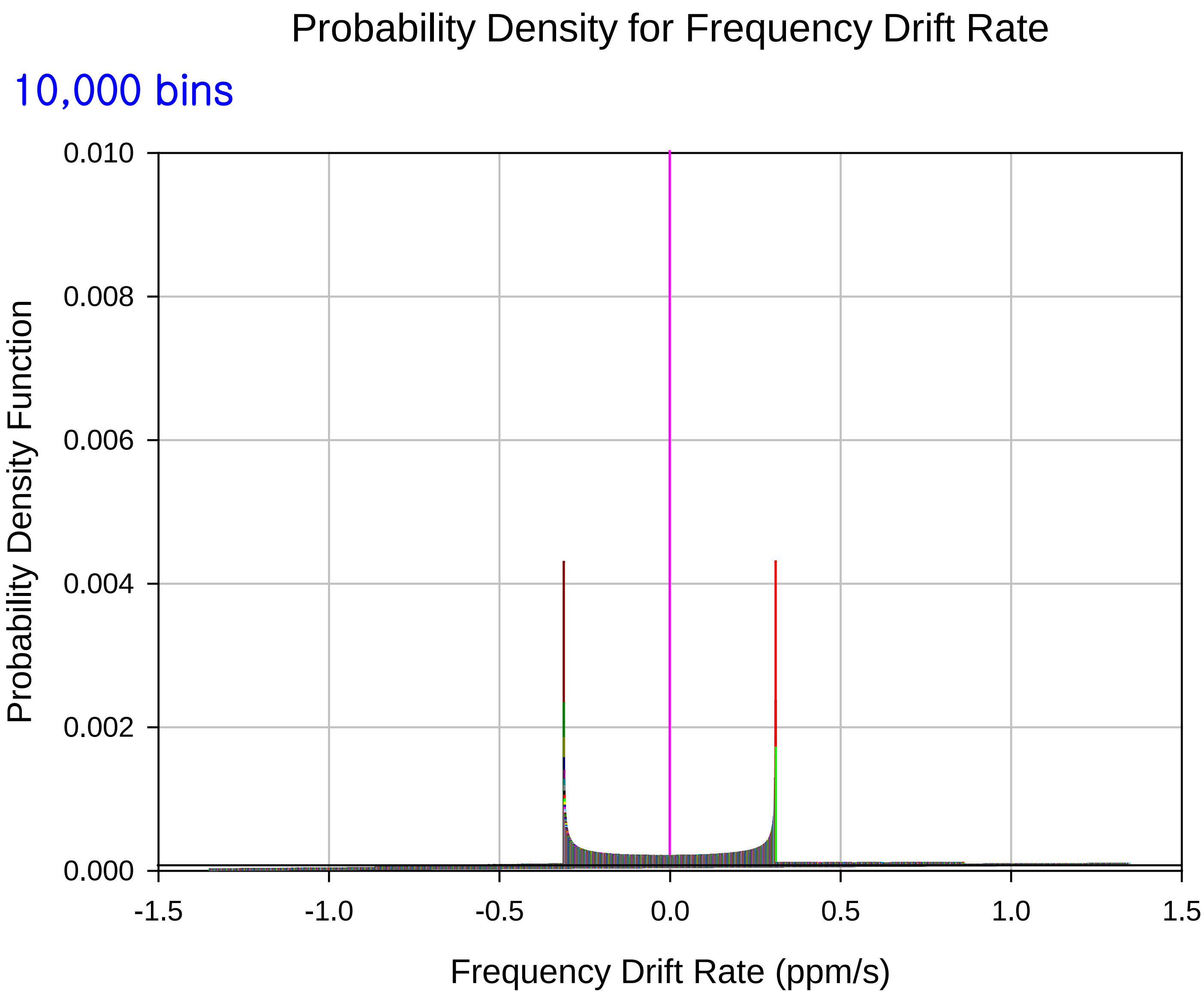
- Slide 52 shows an estimate of the probability density function, based on 10,000 bins and 10^8 samples
  - Each point is obtained by dividing the probability for each bin in slide 51 by the width of the bin
  - This results in the area under the curve summing to 1 (i.e., it is equal to the summation over bins of the height of each bin multiplied by the bin width, which is equal to the sum over the bins of the probabilities)

Equivalent Drift Rate Probability for Frequency Stability Model - 7

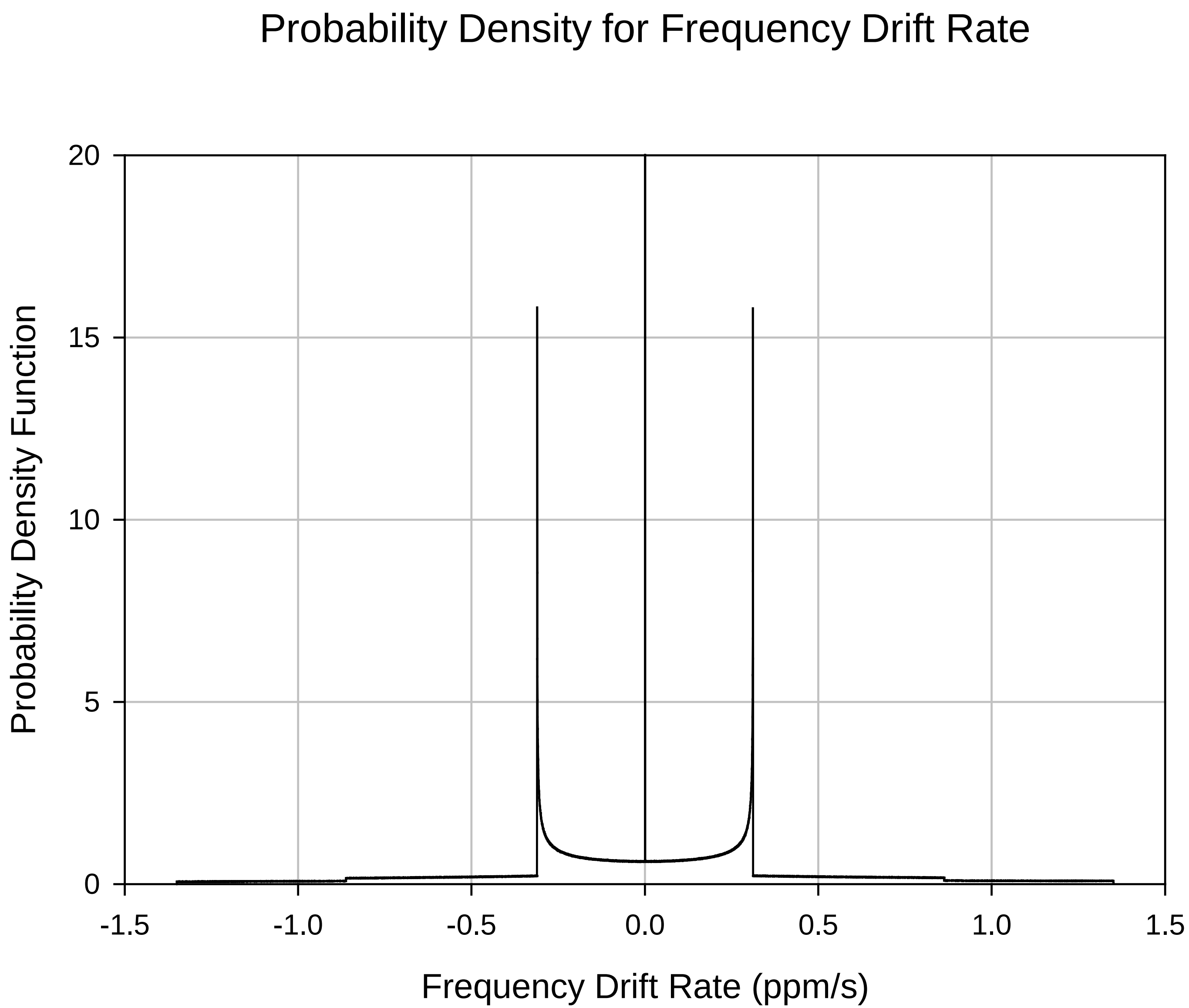
26 bins



Equivalent Drift Rate Probability for Frequency Stability Model - 8



Equivalent Drift Rate Probability for Frequency Stability Model - 9

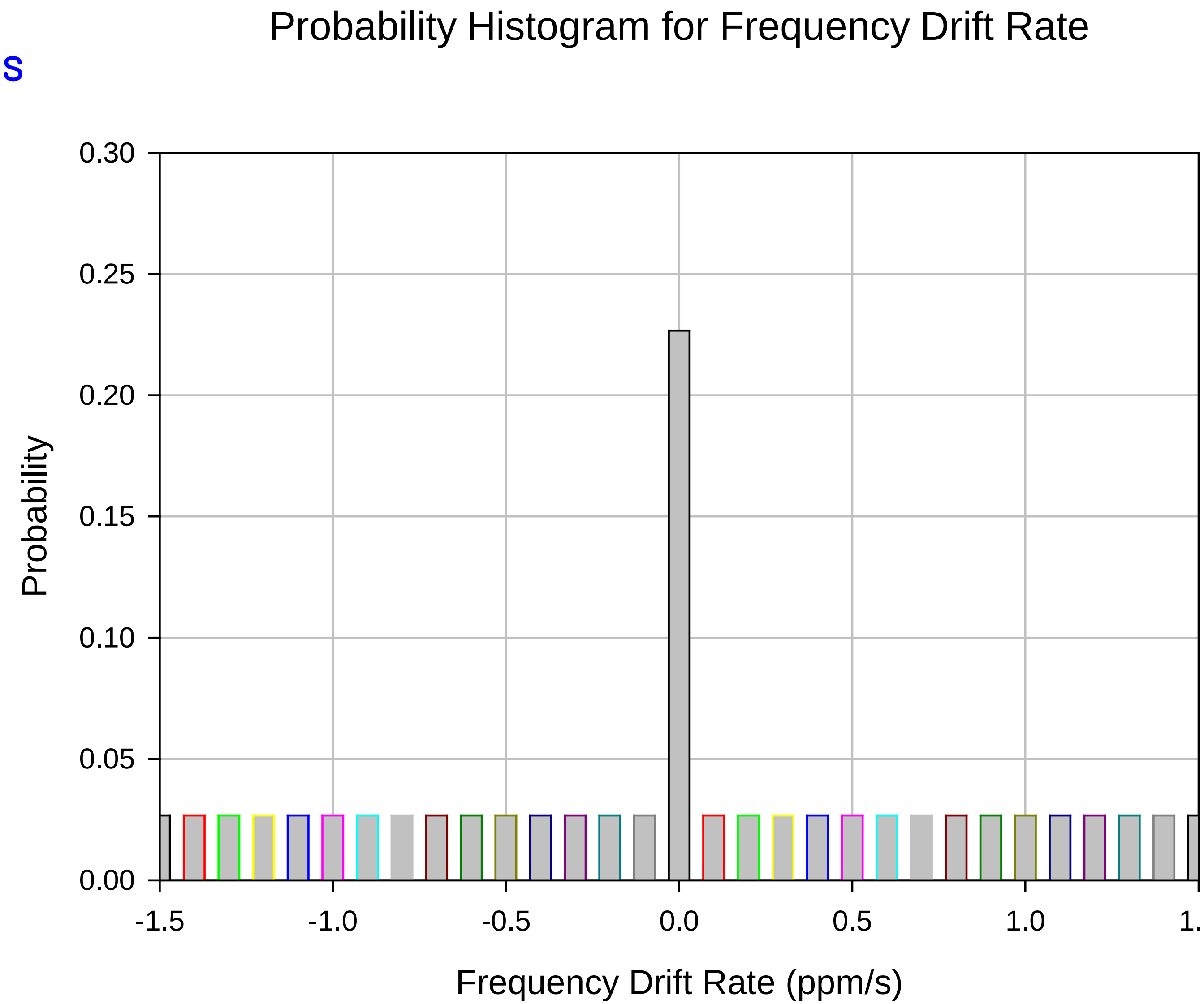


Equivalent Drift Rate Probability for Frequency Stability Model - 10

- Slide 54 shows the probability histogram for the frequency drift rate probability model used in the dTE Monte Carlo simulations
  - 20% probability that the frequency drift rate is 0
  - 80% probability that the frequency drift rate is uniformly distributed in the range [-1.5 ppm/s, +1.5 ppm/s]
  - The number of bins is 30 (so that the bin width is approximately the same as on slide 51)

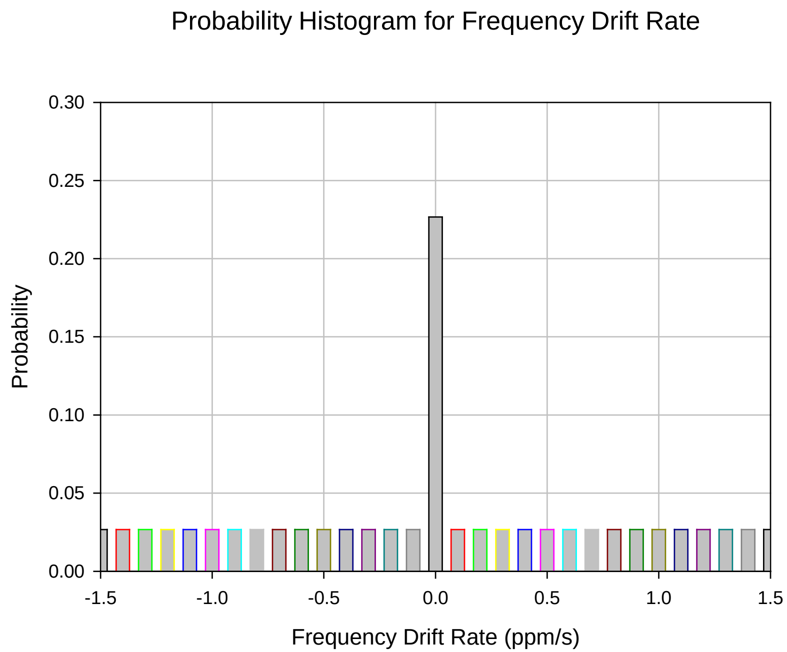
Equivalent Drift Rate Probability for Frequency Stability Model - 11

30 bins

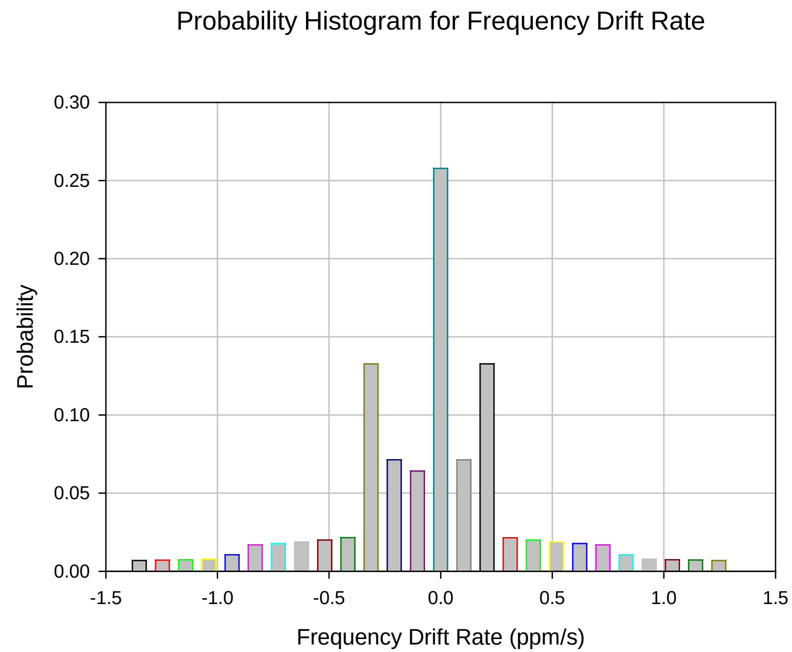


Equivalent Drift Rate Probability for Frequency Stability Model - 12

Monte Carlo Simulations



Derived from Time Series Simulations Frequency Stability Model



- The probability of frequency draft rates with absolute value exceeding 0.5 ppm/s is considerably larger for the Monte Carlo simulation model
- The largest magnitude frequency drift rate is 1.5 ppm/s for the Monte Carlo simulation model, and 1.35 ppm/s for the time series simulation model
- This could partially explain why the max|dTE| results are larger for the Monte Carlo simulations

Conclusions and Next Steps

- The Monte Carlo simulation results for max|dTE| are larger than corresponding time series simulation results (except for 2 instances where the absolute maxima are similar)
- The reason for this might be explained by the fact that in the Monte Carlo simulations the probability of larger frequency drift rates (e.g., larger than 0.5 ppm/s in magnitude) is larger
- It would be useful to run the Monte Carlo simulations again for cases A, B, D, E, and F, but with the frequency drift rate probability model based on the frequency drift rate time history (slides 46 and 47)
- In addition, assumptions for the next set of simulation cases to be run with the time domain simulator are described in [7]; these are summarized on the following slides

Next Time Series Simulations – 1

- Common assumptions for all cases (taken from [7])
  - Clock stability and temperature profile modeled as described in slides 9 – 11
  - Variation in Sync and Pdelay intervals modeled as described in slides 5 – 8
  - Other assumptions in the table on slides 13 and 14 used
  - Mean Sync interval: 125 ms
  - Mean Pdelay interval: 125 ms
  - Timestamp granularity: 8 ns (modeled by truncating to next lower multiple of 8 ns)
  - Dynamic timestamp error is taken to have a uniform distribution over $\pm$ 4 ns
  - Residence time: 10 ms
  - Pdelay turnaround time: 10 ms
  - Window size (N) for mean Neighbor Rate Ratio (mNRR) smoothing: 3
  - Window size (M) for median computation in mNRR smoothing: 1 (median is not taken)

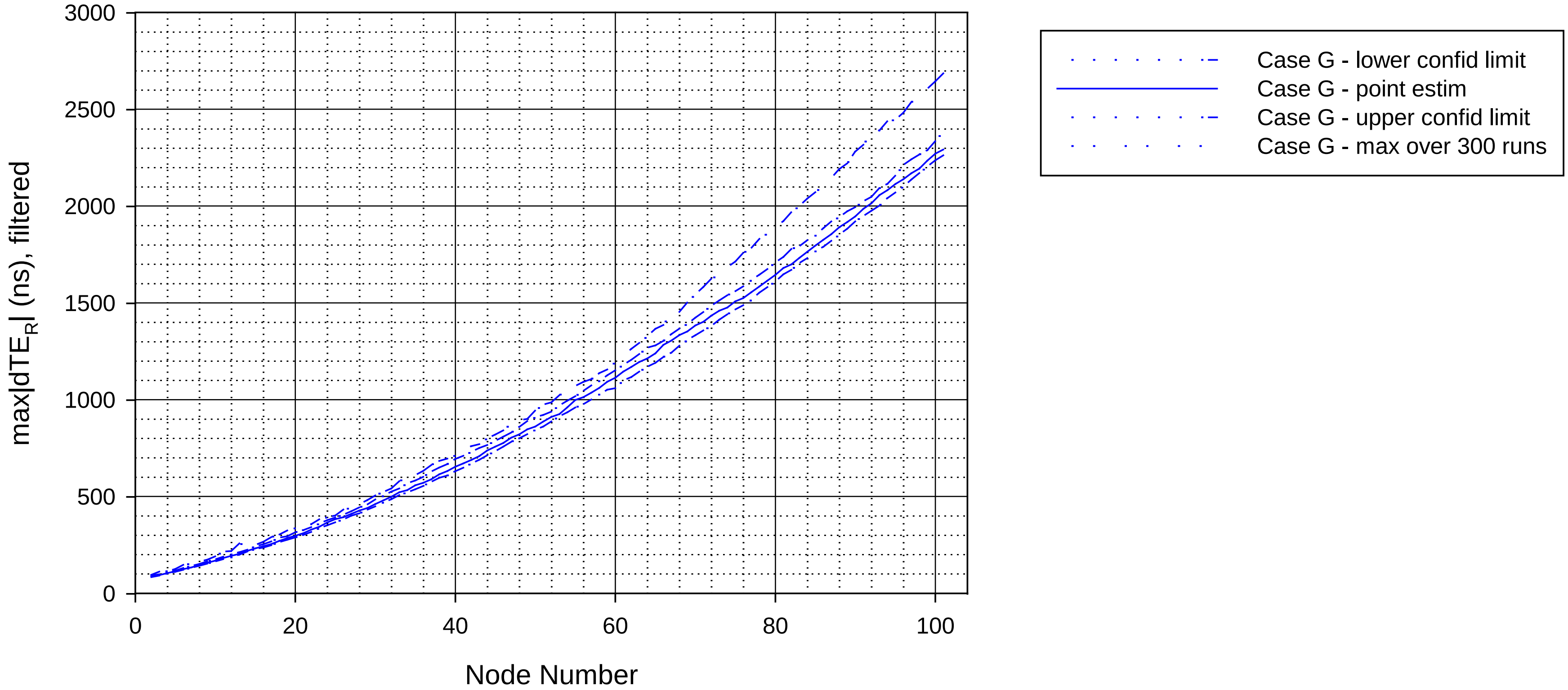
Next Time Series Simulations - 2

- Consider four simulation cases (taken from [7])
  1. Base case, using above assumptions and no compensation or link delay averaging
  2. Base case plus neighborRateRatio drift correction
  3. Base case, plus neighborRateRatio drift correction, plus rateRatio drift correction
  4. Base case, plus neighborRateRatio drift correction, plus rateRatio drift correction, plus mean link delay averaging
- Algorithms for neighborRateRatio drift correction, rateRatio drift correction, and mean link delay averaging are described in slides 29 – 44 of [7]
  - **These algorithms (items 2 – 4) must be added to time domain simulator**
  - The next slides show results for the base case, designated case G for convenience

max|dTE$_R$| Results – Base Case (Case G) - 1

Base case (case G) - mult replication results - filt
GM time error modeled; dTE$_R$ is relative to GM

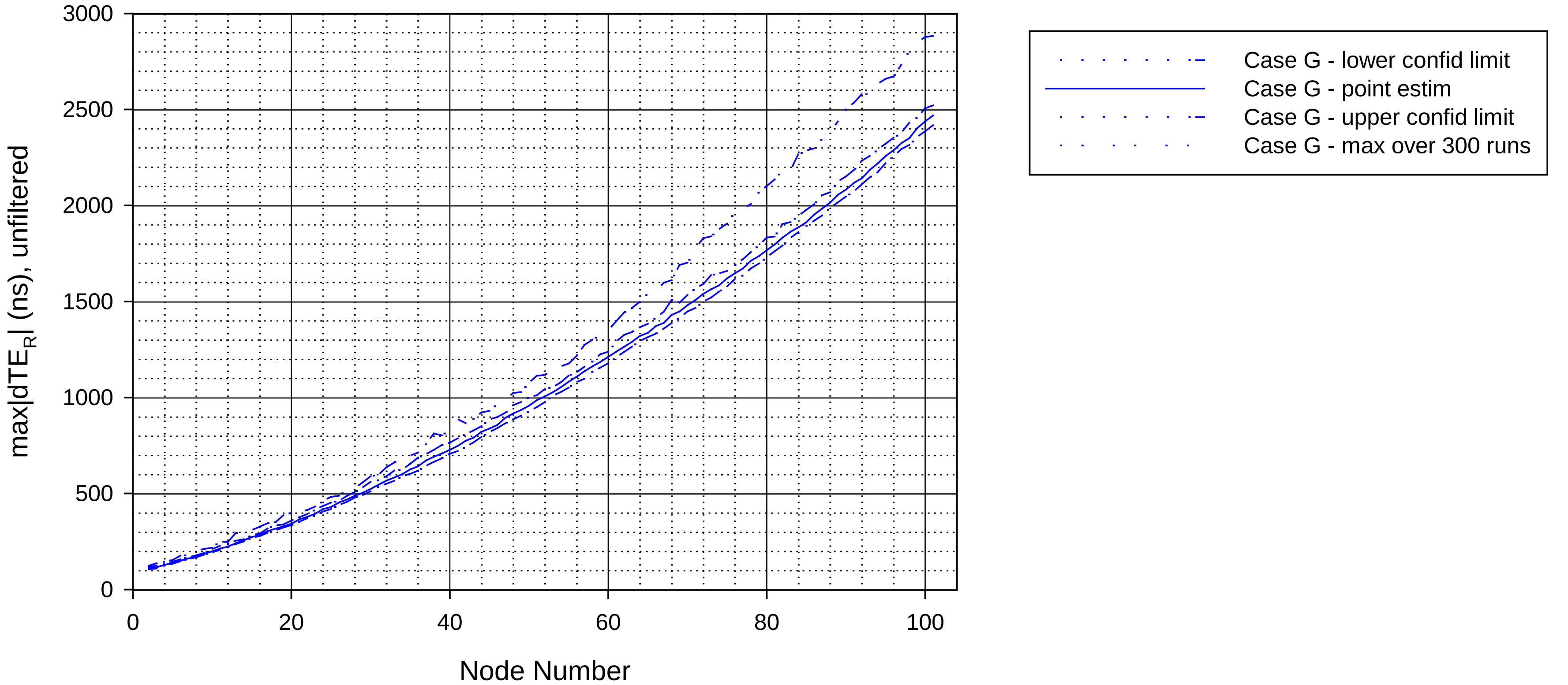
GM labeled node 1
neighborRateRatio measured with window of size 1 (N = 1) and no median calculation
KpKo = 11, KiKo = 65 (f3dB = 2.6 Hz, gain pk = 1.288 dB, zeta = 0.68219)
Sync interval variation: +/-10% with 90% probability (Gamma distribution)
Pdelay interval variation: 1.0 to 1.3 of input Pdelay interval (uniform distribution)
Timestamp granularity and dynamic timestamp error have uniform distributions



max|dTE$_R$| Results – Base Case (Case G) - 2

Base case (case G) - mult replication results - unfil
GM time error modeled; dTE$_R$ is relative to GM

GM labeled node 1
neighborRateRatio measured with window of size 1 (N = 1) and no median calculation
KpKo = 11, KiKo = 65 (f3dB = 2.6 Hz, gain pk = 1.288 dB, zeta = 0.68219)
Sync interval variation: +/-10% with 90% probability (Gamma distribution)
Pdelay interval variation: 1.0 to 1.3 of input Pdelay interval (uniform distribution)
Timestamp granularity and dynamic timestamp error have uniform distributions



max|dTE_R| Results - Base Case (Case G) - 3

Summary of Results at Last Node

Case	Filtered/ Unfiltered	Lower (ns)	Point Est (ns)	Upper (ns)	Max (ns)
G	Filtered	2294	1688	2387	2688
	Unfiltered	2419	2469	2521	2883

Compare with Monte Carlo Simulation result from[7] (slide 16) for case of no algorithms:

3938 ns

The time domain simulation result max (unfiltered) of2883 ns is 26.8% smaller

As indicated above (slide 57), it would be useful to run the Monte Carlo simulations again for the base case, but with the frequency drift rate probability model based on the frequency drift rate time history (slides 46 and 47)

Thank you

References – 1

[1] David McCall, *60802 Dynamic Time Error – Additions – Error due to drift during Sync messaging – Potential Contribution*, IEC/IEEE 60802 Presentation, March 2022 (available at https://www.ieee802.org/1/files/public/docs2022/60802-McCall-Stanton-Time-Sync-Error-Model-and-Analysis-0322-v01.pdf)

[2] Geoffrey M. Garner, *New Simulation Results for dTE for an IEC/IEEE 60802 , Based on New Frequency Stability Model, Revision 1*, IEC/IEEE 60802 presentation, April 9, 2021 (available at https://www.ieee802.org/1/files/public/docs2021/60802-garner-new-simulation-results-new-freq-stab-model-0421-v01.pdf)

[3] Geoffrey M. Garner, *New Simulation Results for dTE for an IEC/IEEE 60802 , Based on New Frequency Stability Model, Version (Revision) 1*, IEC/IEEE 60802 presentation, May 3, 2021 (available at https://www.ieee802.org/1/files/public/docs2021/60802-garner-multipe-replic-simulation-results-new-freq-stab-model-0421-v01.pdf)

References – 2

[4] Geoffrey M. Garner, *New Simulation Results for dTE for an IEC/IEEE 60802 Network, with Variable Inter-Message Intervals, Revision 2*, IEC/IEEE 60802 presentation, July 1, 2021 (available at https://www.ieee802.org/1/files/public/docs2021/60802-garner-single-replic-simul-results-variable-intermsg-intervals-0621-v02.pdf)

[5] David McCall, *60802 Dynamic Time Sync Error – Monte Carlo Analysis Results for Comparison with Time Series Simulations*, Revision v02, IEC/IEEE 60802 presentation, March 2022 (available at https://www.ieee802.org/1/files/public/docs2022/60802-McCall-Time-Sync-Monte-Carlo-Results-for-Time-Series-Comparison-0322-v02.pdf)

[6] Geoffrey M. Garner, *Phase and Frequency Offset, and Frequency Drift Rate Time History Plots Based on New Frequency Stability Data*, IEC/IEEE 60802 presentation, March 8, 2021 (available at https://www.ieee802.org/1/files/public/docs2021/60802-garner-temp-freqoffset-plots-based-on-new-freq-stabil-data-0321-v00.pdf)

References - 3

[7] David McCall, *60802 Dynamic Time Sync Error – Recommended Parameters & Correction Factors, IEC/IEEE 60802 presentation, Revision v04*, March 2022 (available at https://www.ieee802.org/1/files/public/docs2022/60802-McCall-Time-Sync-Recommended-Parameters-Correction-Factors-0322-v04.pdf)

BACKUP

Resulting Dependence of Phase Offset on Time - 1

- The phase offset as a function of time, $x(t)$, is obtained by integrating the frequency offset time history, $y(t)$, and expressing $x(t)$ in appropriate units
  - Assuming phase offset is zero at time zero:

$$x(t) = \int_0^t y(u)\, du$$

  - If $y(t)$ is in ppm, then $x(t)$ will be in μs
- The integration is done numerically, using the trapezoidal rule
  - A higher-order scheme (e.g., Simpson's rule) was not used because this would require values of in between time steps (or, alternatively, would not produce a result at every time step)
  - The iteration is:

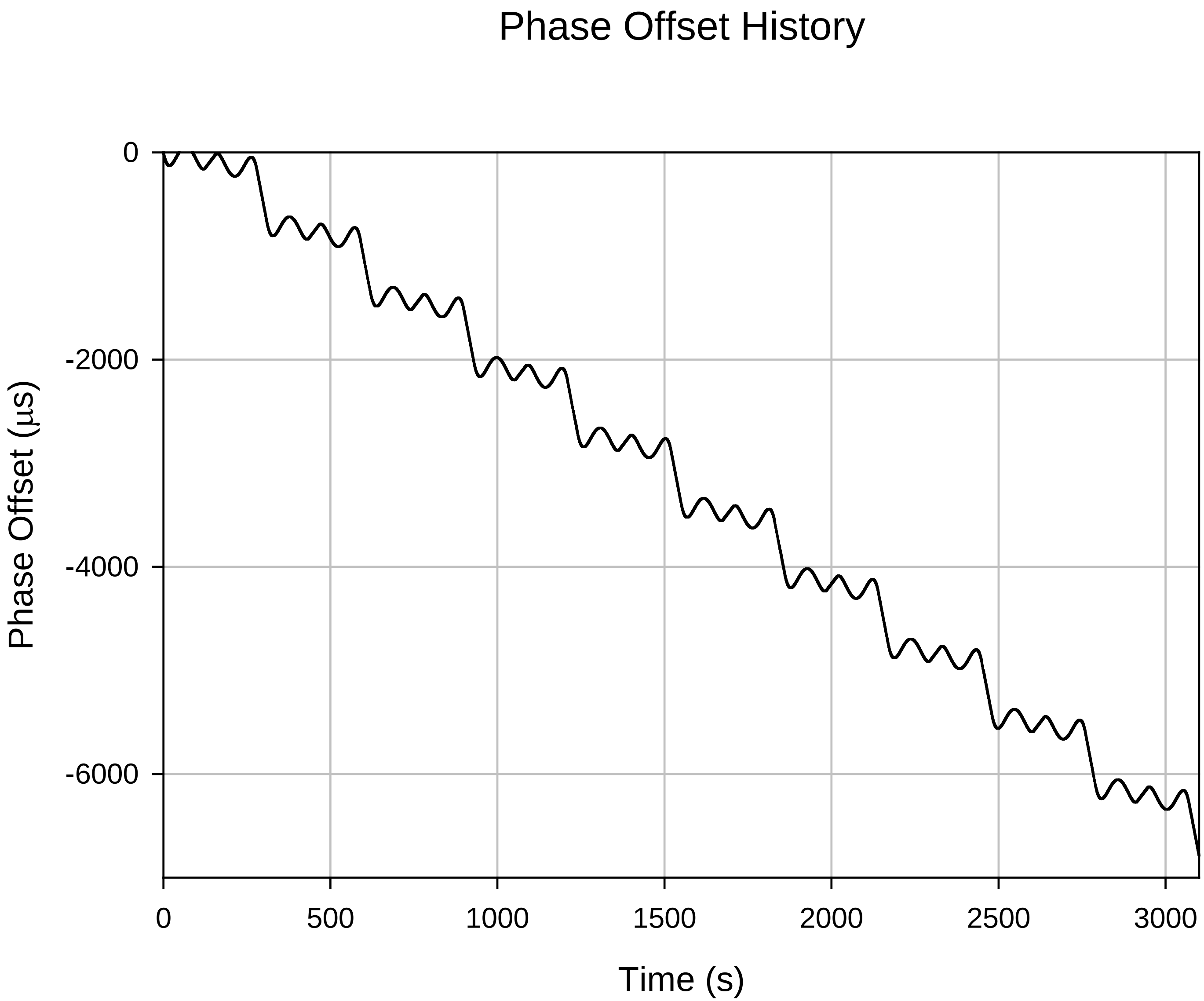
$$x_{k+1} = x_k + \frac{\Delta t}{2}\left(y_{k+1} + y_k\right)$$

where

$k =$ timestep index (i.e., $t = k\Delta t$)

$\Delta t =$ timestep size in s

Resulting Dependence of Phase Offset on Time - 2



Resulting Dependence of Phase Offset on Time - 4

- The phase time history shows a phase drift of approximately – 6.7 ms over 3100 s
- This is equivalent to a frequency offset of 6700/3100 μs/s = – 2.2 ppm
- This frequency offset arises because, with the frequency offset as a function of temperature (slide 8) and temperature profile (slide 15) assumptions, the frequency offset averaged over one cycle of temperature variation is not zero (i.e., it is approximately – 4.2 ppm)
- Actually, the average frequency offset of the crystal unit does not matter, because this is the frequency offset of the free-running LocalClock entity in 802.1AS
  - The neighborRateRatio is measured using the Pdelay messages and the overall rateRatio relative to the GM is accumulated in a TLV
  - What matters is how much the frequency of the free-running LocalClock entity changes between Pdelay messages
- Finally, as described in Reference [3] of [6], the actual long-term frequency offset can be different from this value because the frequency versus temperature curve of slide 8 can be shifted vertically due to the frequency tolerance of the crystal unit