SYNCHRONIZING COMPUTER CLOCKS BY THE USE OF KALMAN FILTERS

Judah Levine
Time and Frequency Division and JILA
NIST and the University of Colorado
Boulder, Colorado 80305
(303) 497 3903
jlevine@boulder.nist.gov

Abstract

The National Institute of Standards and Technology (NIST) currently operates 35 public network time servers that are located at many different sites in the United States. The servers provide time messages over the public Internet in a number of different formats. The times are derived from the NIST atomic clock ensemble in Boulder, Colorado, and are directly traceable to UTC(NIST). The time servers currently receive about $7{\times}10^9$ requests per day, and the number of requests is increasing by a compounded rate of about 5% per month.

I have continued to work on methods of improving the performance of the ensemble of servers so as to be able to support this increasing demand without a corresponding increase in the number of servers. My current efforts involve studies of replacing the phase lock and frequency lock synchronization algorithms with methods based on Kalman filters, and I will report on my latest results.

A method based on Kalman filters has an advantage in principle, since these kinds of algorithms support more sophisticated noise models than can be used by the existing synchronization procedures. However, it is not easy to realize this advantage, because the real noise is generally not stationary and is not well characterized in statistical terms. I have experimented with addressing this problem by adding periodic or quasi-periodic terms to the covariance matrix of the process, and I will show the improvement that I have realized using these methods.

The algorithms have two advantages over many other implementations. The more sophisticated noise models mean that the same performance can be realized by the use of a longer interval between requests. This is a significant advantage, since it directly improves the level of service that can be provided with a given number of systems. The ensemble is independent of the GPS satellites and requires no outside antennas. It is therefore more robust and more difficult to jam.

INTRODUCTION

The National Institute of Standards and Technology (NIST) currently operates 35 public network time servers that are located at many different sites in the United States. The servers provide time messages over the public Internet in a number of different formats. The times are derived from the NIST atomic clock ensemble in Boulder and are directly traceable to UTC(NIST). The exchange of timing information between the remote time servers and the NIST clock ensemble uses messages in the ACTS format **[1]** that are transmitted over voice-grade dial-up telephone lines. The method that is used to synchronize the clock of the server to UTC(NIST) is a modified frequency-lock loop and has been described previously **[2]**. The goal of the algorithm is to combine the calibration information received by the use of the ACTS time messages with the free-running stability of the local clock oscillator to realize a statistical performance that is better than either component by itself. The time of the internal clock is then used as the reference for all of the time messages that are transmitted by the server.

In order to understand the design of the new Kalman algorithm that I will describe in the next sections, it is helpful to describe the LOCKCLOCK algorithm, which is currently used to control the time servers, with particular emphasis on its implied requirements and associated limitations.

THE ACTS PROTOCOL

The ACTS servers are synchronized to UTC(NIST) by means of a hard-wired connection between the servers and the 1 Hz output of the clock ensemble. The time of the ACTS servers is kept within 1-2 µs peak-to-peak of UTC(NIST) by the use of a tight phase-lock loop. This timing jitter is negligible compared to the other contributions to the variance of the data received by the time servers. Therefore, we will assume in the following discussion that the time at the ACTS server is exactly UTC(NIST), and that any variation in the received time is due to down-stream effects such as jitter in the transmission delay through the network or noise in the measurement process at the remote end.

The ACTS protocol is typical of all two-way time distribution systems: the ACTS server transmits a time message every second, and the messages are corrected for the transmission delay from the ACTS server to the remote system (which is the network time server in this discussion) through the telephone network. The delay is estimated by the ACTS server as one-half of the measured round-trip value. The round-trip delay can be measured by either the transmitter or the receiver in principle, although there is often a subtle advantage in having the receiver calculate the estimate. In the current implementation, the ACTS server implements an initial estimate of the network delay, and the receiver makes small corrections to this estimate by averaging consecutive time-difference data **[3]**.

The performance of the synchronization process depends on the fact that the variance of the received time messages (after the delay corrections have been applied) is well characterized as a zero-mean white-noise process. This will be true if the network delay is symmetric on the average and if any asymmetry in the network delay has a white noise spectrum about a mean of 0. The validity of this characterization of the ACTS system is an important implied requirement of the LOCKCLOCK synchronization algorithm. This assumption about the statistical characteristics of the telephone connection to the ACTS servers is becoming less valid as the telephone system is upgraded from physical circuits with well-defined and stable transmission characteristics to logical circuits using packet-switched technology, whose delay characteristics are less stable and less predictable. One of the goals of the work described here is to mitigate the degraded performance of the telephone links, since the stability of the transmission delay through these links plays a central role in the accuracy of the NIST Internet time service.

THE LOCKCLOCK ALGORITHM

The algorithm periodically measures the time difference between the local clock oscillator and the ACTS system by the use of voice-grade telephone lines and the two-way message protocol as described in the previous paragraphs. The measurement of the time difference between the local clock and the received message is performed at interrupt priority within the serial-line driver that receives the ACTS message, and the jitter in this measurement is less than 3 μs peak-to-peak. Therefore, the variance of the measurements is predominantly due to two contributions: the residual un-modeled jitter in the delay through the telephone link back to the ACTS server, and the time dispersion due to the frequency noise of the local clock oscillator. These two-sample Allan deviations of the two contributions are shown in Figure 1 for a typical system that we use for the time service.

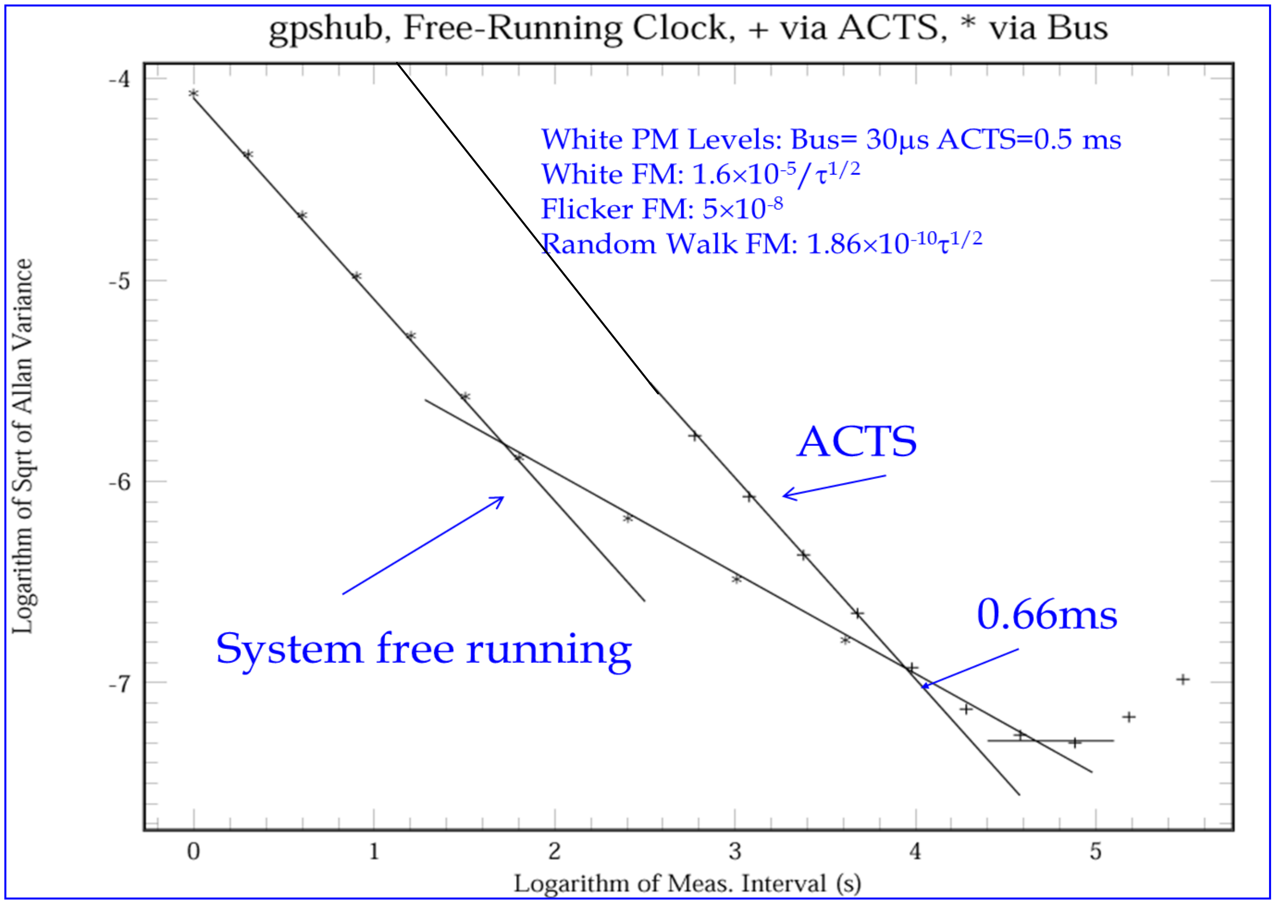

Figure 1. The Allan deviations of the ACTS time messages as received over a standard dial-up telephone line and the time of the clock in a work station. Both of these data are acquired using special-purpose hardware that is used during the calibration procedure but is not part of an operating time server. The parameters in the figure characterize the stability of the system clock using the usual parameters of white, flicker, and random-walk processes. The ACTS data are characterized by white phase noise over the entire measurement domain.

The physical oscillator that drives the system clock is not directly observable, and the Allan deviation of the free-running system clock was measured as the clock oscillator is seen through the operating system software. Therefore, the measurements include the jitter in the operating system handling of the request for the time and the response to that request. The Allan deviation of the ACTS link was measured in a separate configuration using a hard-wired loop-back cable to implement the two-way measurement protocol. In both cases, the reference for the measurement was a local rubidium frequency standard whose frequency stability is several orders of magnitude better than the values shown in Figure 1.

As we can see from the figure, the ACTS data are characterized by white phase noise over the range of averaging times that were observed. The local oscillator is characterized by white phase noise at short averaging times (< 100 s), presumably due to a combination of the noise in the oscillator itself and the jitter in the operating system code that reads the physical clock and returns the value to the application program. The averaging times shown in the figure are much longer than the cycle time of the hardware, so that the jitter is independent of the averaging time in first order, and the white frequency noise of the local clock oscillator becomes important at longer averaging times. The statistical properties of the local oscillator become more divergent at averaging times greater than about 10 hours.

The stability of the ACTS system as seen through the telephone connection and the stability of the local clock oscillator as seen through the operating system software become equal at an averaging time of about 10,000 s, and this determines the minimum averaging time at which the ACTS data can begin to improve the stability of the local clock. (Note that this is still well in the domain of averaging times where the statistics of the local oscillator are characterized by white frequency noise.) If the averaging time is shortened, the white phase noise of the ACTS connection degrades the stability of the local oscillator; if the averaging time is increased the time dispersion due to the fluctuations in the frequency of the local clock oscillator are not attenuated as well as they could be and the time stability of the system is degraded. The detailed results vary somewhat from one time server to another, although the variation is not very great, and the results of Figure 1 were typical of all of our time servers up until recently. (Telephone connections within a single local exchange are generally significantly quieter than the line whose performance is shown in Figure 1. For example, lines within the Boulder local calling area have a time dispersion that is often less than 100 μs for all averaging times. This is almost a factor of 10 more stable than the data presented here. These connections are not typical of the time server network.)

With the optimum averaging time derived in the previous paragraph, the stability of the time server is somewhat better than 0.001 s RMS for averaging times greater than about 10 000 s. The time stability at shorter averaging times is determined by the free-running stability of the system clock oscillator or by noise in the measurement process at the very shortest averaging times. At very short averaging times on the order of a few seconds, the combination of these effects results in a time dispersion of about 30 µs (assuming pure white phase noise), and the time dispersion increases proportional to the square root of the averaging time for averaging times greater than about 100 s. It would be possible to improve the stability in the white phase noise domain by averaging closely spaced time-difference data, although the jitter would improve only as the square root of the number of values used in the average, so that it would not be practical to reduce the jitter by more than a factor of two or three. In any case, this improvement in stability is not of much practical utility, since the stability of the time data received by most network-connected users will be determined by the jitter in the network connection (rather than by the internal accuracy of the time server itself), and this almost always makes the largest contribution to the variance of the data received by a network-based client. (Using these data, it is clear why the pure phase-lock loop that is used in many implementations of the network time protocol must use a short polling interval, since it must operate in the domain where the fluctuations in the measured time differences are characterized by white phase noise. The default polling interval is typically 64 s, which is at the upper end of the white phase noise regime.)

THE IMPACT OF TELEPHONE-LINE STATISTICS

As you can see from the previous discussion, the amplitude of the white phase noise in the telephone line plays a crucial role in the performance of the time servers. If the level of the white phase noise in the telephone line increases, then the cross-over in the Allan deviation curves shown in Figure 1 is pushed towards longer averaging times. Assuming that the crossover is still in the region where the local clock is characterized by white frequency noise, the RMS time dispersion at the cross-over point increases as the square root of the averaging time and the performance of the time server is degraded accordingly. If the increase in the averaging time becomes large enough, then in addition to the degradation in the timing accuracy discussed in the previous sentence, the cross-over point moves outside of the white frequency noise domain of the clock oscillator, the performance degrades more rapidly than as the square root of the averaging time, and a frequency-lock loop is no longer an optimum way of controlling the clock.

USING A BETTER LOCAL OSCILLATOR FOR THE SYSTEM TIME

It is very desirable to operate the time server in a domain where white processes dominate the variance of the data used to synchronize the system, since these processes are amenable to robust estimation techniques by averaging either the phase or the frequency, when white phase noise or white frequency noise, respectively, is the dominant noise type. If the noise in the telephone connection increases, this consideration can still be realized by replacing the oscillator that drives the system clock with a more stable one whose performance can be characterized as no worse than white frequency noise over a greater range of averaging times. However, when the averaging time is such that the local system oscillator is operating in its white frequency noise domain, the time dispersion at the cross-over point increases as the square root of the averaging time, so that the free-running frequency stability of the replacement oscillator must be correspondingly better as well.

The more stable oscillator improves the performance of the time service at shorter averaging times, but the time service cannot be more accurate than the ACTS stability at sufficiently long averaging times. Other effects, such as a nearly-diurnal or long-period wander in the characteristics of the communications channel may also limit the time performance at longer periods. (The amplitude of these long-period effects is bounded, so that the Allan deviation must decrease for sufficiently long averaging times, but this improvement can be exploited only for estimating the *frequency* of very stable clocks, which are not commonly used in these configurations. Furthermore, network-based time services are generally not used for frequency distribution.)

CHOOSING THE AVERAGING TIME

When the stability of the ACTS channel is relatively poor, the long averaging times that are required by the previous considerations are too long to be useful in practice. The long-term accuracy of the time server (as distinct from its stability) is derived from the ACTS messages, since these data provide the only link to UTC(NIST). Increasing the polling interval means that it will take longer for the system to provide accurate time data following a cold start, since the white phase noise of the individual ACTS measurements must be attenuated by the increased averaging. The increased polling interval also means that a glitch in the time of the local clock will persist for a longer time before it is detected and removed. These considerations favor the shortest possible polling interval consistent with the statistics. On the other hand, the cost of the ACTS connections is inversely proportional to the interval between connections, and this consideration would drive the interval between calibrations towards larger values. (A cost-benefit analysis is less important for a time server synchronized by the use of a hard-wired connection to the NIST atomic clock ensemble or to a GPS receiver, since the incremental cost of a calibration is negligible.)

These considerations do not determine the interval precisely, but they suggest that an interval on the order of one hour provides a reasonable balance between these conflicting requirements. This was a good choice originally, since it was close to the cross-over point between the free-running stability of the computer clock and the jitter in the ACTS messages as shown in Figure 1. However, it is too short for the increased jitter in the ACTS transmissions that is more common at present. Therefore, we are led to look for an algorithm that provides an objective method for decoupling the polling interval from the averaging time, keeping the former relatively short while increasing the latter to take advantage of the improved stability of a good local oscillator. The Kalman formalism provides a natural framework for realizing this type of algorithm, and we will discuss the details of the method in the next section.

THE KALMAN ALGORITHM

The goal of any synchronization algorithm is to estimate the time difference between the local clock oscillator and the remote reference at epoch t_k by the use of the measured time difference at that epoch, X_k. Using the language common in Kalman processes, I estimate the time state of the system immediately before the measurement in terms of the previous values of the time and frequency states by the use of

$$x_k^- = x_{k-1} + y_{k-1}\tau + \xi \tag{1}$$

where x and y are the time and frequency states of the system clock, respectively, ξ is the stochastic contribution to the time state due to the white phase noise of the oscillator and the jitter in the processing of time requests by the operating system. These effects are largely independent of averaging time, although they might depend to some extent on the absolute time when the measurement was made because of some diurnal or other long-period effect. I don't consider this dependence here, but it will probably become important in a more comprehensive Kalman analysis that is a future project. We will estimate the time state immediately after the measurement by the use of a linear combination of the state before the measurement and the measurement data, at time t_k, denoted by X_k. Thus,

$$x_k^+ = x_k^- + I(X_k - x_k^-) = (1-I)x_k^- + IX_k \tag{2}$$

where I is the gain factor that is applied to the innovation – the impact of the difference between the time difference that we measured and the value that we expected based on the previous state parameters.

A full vector Kalman algorithm would include parameters to estimate x and y, the time and frequency states of the system clock oscillator. Since we are going to operate in the domain where the clock oscillator is dominated by white frequency noise, the evolution of its state component would be given by

$$y_k = y_{k-1} + \eta \tag{3}$$

where η is the amplitude of the white frequency noise. I am going to confine the discussion to the region in which the frequency noise of the oscillator can be characterized as a single scalar parameter that is independent of epoch or averaging time. A more exact analysis would include long-period contributions to the variation of the frequency. These contributions are generally included by adding a frequency drift term to Equations 1 and 3.

My initial algorithm was a simpler scalar version, and I used the algorithm to estimate only the time state of the system clock. The simpler estimate of the innovation was based on two parameters: the variance of the telephone connection, ε_t^2, and the composite variance of the local clock oscillator, ε_o^2. This composite variance was assumed to be a characteristic of the system hardware as seen through the operating system. I estimated it by

$$\varepsilon_o = \sqrt{\left\langle (x_k^+ - x_k^-)^2 \right\rangle} \tag{4}$$

by the use of a measurement technique with negligible measurement noise. The time dispersion of Equation 4 is due primarily to the white frequency noise of the local oscillator, η, since the measurement noise and the jitter in the processing of the measurements by the operating system are negligible by comparison. In effect, this is equivalent to assuming that the effect of ξ in Equation 1 is negligibly small compared to the effect of η acting through the frequency term in that equation. The goal of the Kalman algorithm was then to partition the measured time differences into a deterministic component due to the time and frequency offset of the system oscillator and stochastic contributions due to the telephone line and the white frequency noise of the clock oscillator.

The Kalman solution is particularly simple in this case, and also can be deduced from simpler considerations. After we have removed the deterministic component of the time difference, the algorithm must combine the remaining stochastic contributions. Since the two contributions arise from completely independent sources, we assume that there are no correlated effects between the telephone line statistics and the fluctuations of the computer clock oscillator. Therefore, the estimate of the state is simply the weighted sum of the two statistically independent estimates normalized by the sum of the weights **[4]**:

$$\hat{x}_k = \left[\frac{1}{\varepsilon_o^2} + \frac{1}{\varepsilon_t^2}\right]^{-1} \left[\frac{x_k^-}{\varepsilon_o^2} + \frac{X_k}{\varepsilon_t^2}\right] = \frac{\varepsilon_t^2}{\varepsilon_o^2 + \varepsilon_t^2} x_k^- + \frac{\varepsilon_o^2}{\varepsilon_o^2 + \varepsilon_t^2} X_k \tag{5}$$

This algorithm is an extension of the previous implementation of LOCKCLOCK, which modeled all of the variance of the time difference measurements as due to the frequency noise of the local clock oscillator. That is, it assumed that $\varepsilon_o >> \varepsilon_t$ so that the estimate of the time difference was essentially equal to the measured ACTS value. This more general result relaxes this requirement, so that poorer quality telephone lines for which the two contributions are more nearly equal can also be handled in a statistically robust manner.

COMPARING THE NEW ALGORITHM WITH THE PREVIOUS ONE

In order to test the usefulness of this model, we have compared the performance of a time server that had a stable telephone connection with another identical server that had a poor connection and was synchronized first with the old algorithm and then with the new one. The results of this comparison are shown in Figure 2.

The lowest trace, which is identified by “A” in the figure, is the time deviation of the time server when it was synchronized to ACTS using a standard voice-grade telephone line. This level of performance is characteristic of all of the servers that use long-distance telephone lines that were installed more than about one or two years ago. The interval between ACTS calibrations is nominally 4000 s, and the performance is approximately characterized by a TDEV of 8×10^{-4} s at the nominal calibration interval of 4000 s, decreasing as $1/\sqrt{\tau}$ s over most of the range of averaging times greater than that value. (This result is roughly comparable to the ACTS performance shown in Figure 1, which shows the result as a two-sample Allan deviation rather than as TDEV.) This dependence is characteristic of white phase noise except for averaging times of a few days, where nearly-diurnal and longer-period temperature variations are probably important.

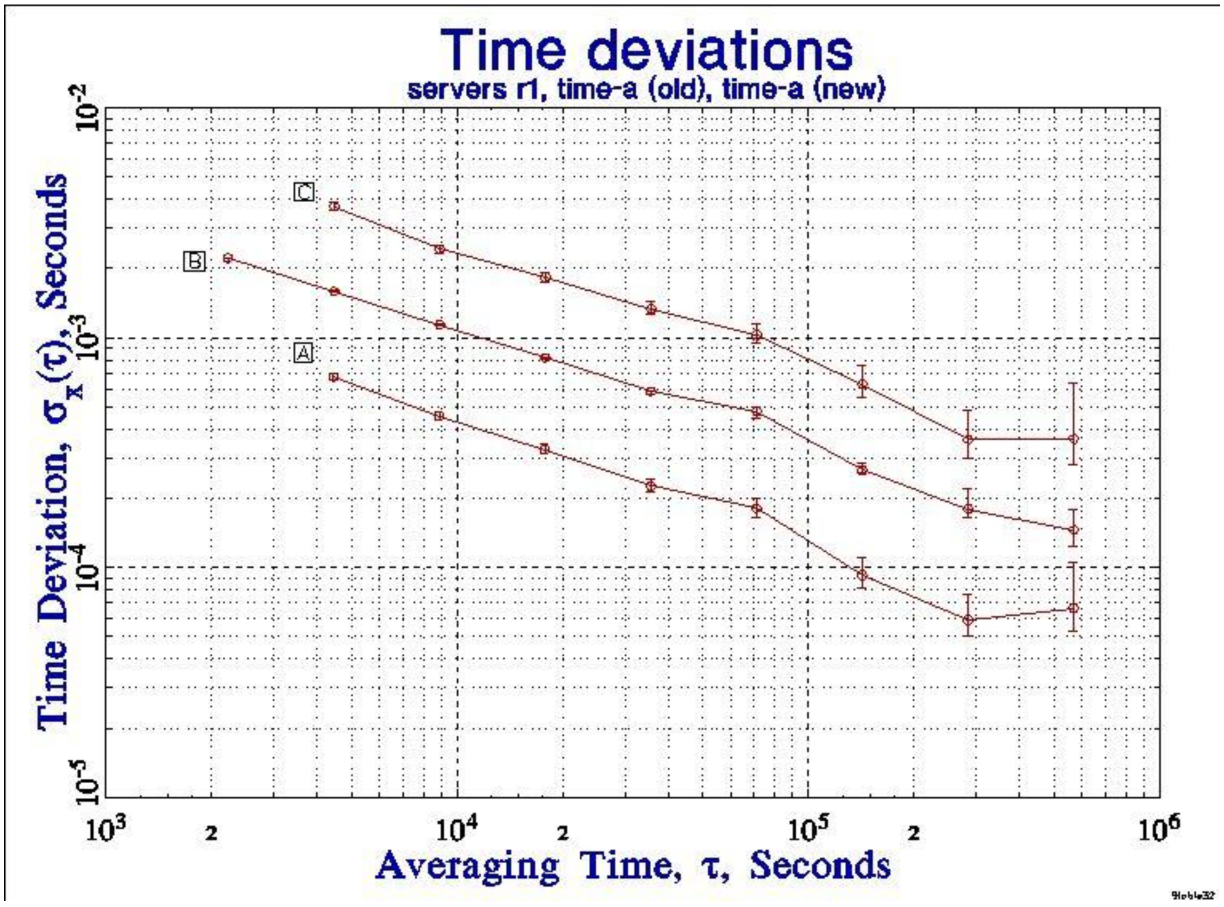

Figure 2. The time deviations (TDEV) for different configurations. The “A” trace shows the TDEV for a typical time server that used one of the original telephone connections. The “C” curve shows the TDEV for one of the newer connections that is less stable, and the “B” curve shows the TDEV for the same server as in “C” synchronized by the user of the new Kalman algorithm described in the text.

The upper trace, which is identified by “C” in the figure, is the time deviation of an identical server when it is synchronized to ACTS by the use of a much poorer telephone line. The dependence is still characterized by white phase noise, but the amplitude is about five times larger than the system discussed in the previous paragraph. In fact, the characterization of white phase noise is only approximately valid, since the actual time series of the measurements are only approximately statistically stationary. This point is illustrated in Figure 3, which shows the first section of the time difference data used to compute the statistic “C” in Figure 2. This time series illustrates that statistical estimators, which assume that the data being analyzed are stationary, can provide very misleading information when this assumption is not valid.

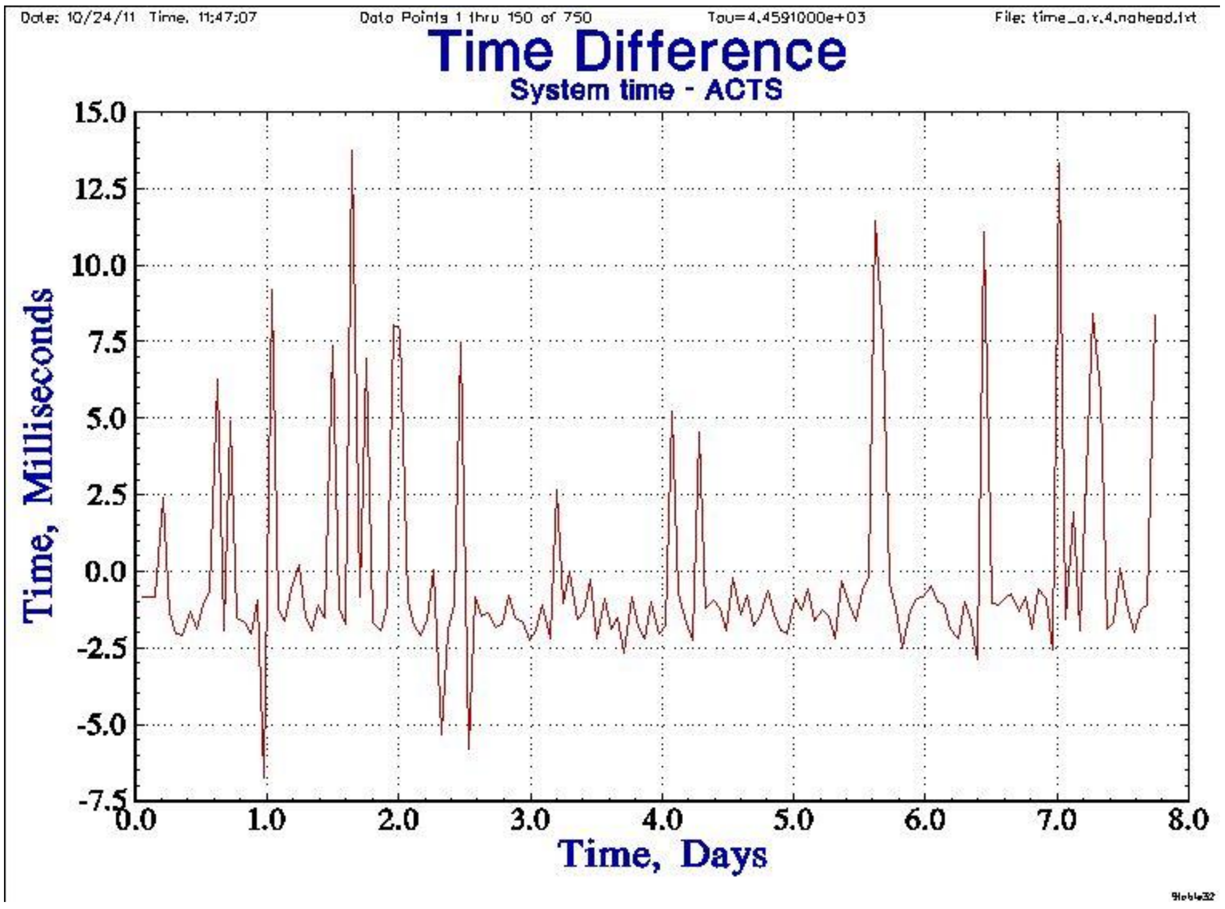

Figure 3. The first portion of the time series used to construct the time deviation shown in statistic "C" of Figure 2.

The middle trace, identified by "B" in the figure, shows the application of the initial Kalman algorithm, which I discussed above, to this server and telephone-line configuration. The interval between ACTS queries was decreased and was de-coupled from the averaging time. The resulting performance is considerably better than the performance of the original algorithm (curve "C"), but the algorithm cannot fully compensate for the poor telephone line, and the performance is not as good as what can be realized with a good connection as shown in curve A. Nevertheless, the results are very encouraging.

The spikes and transients in Figure 3 are superimposed on a much smaller baseline of time differences, which suggests that one way of dealing with them is to identify them as glitches and ignore them rather than trying to design an algorithm for averaging them. In order to make this identification unambiguous, the free-running stability of the system clock must be good enough so that the spike can be assigned to the telephone connection and not to a glitch in the system time. In other words, if we are going to ignore these data as glitches, then the free-running stability of the system clock must be good enough to maintain a time accuracy on the order of 1 ms without any adjustments (in "hold-over" mode). If we consider the data in Figure 3 as "typical," the hold-over period might have to extend for several days, so that an oscillator with a free-running fractional stability of $\leq 10^{-8}$ at an averaging time of a few days would be adequate. This level of performance is significantly better than the stability of oscillators normally found in standard "off the shelf" computer systems, but it is not difficult to use an external oscillator that meets this requirement.

SUMMARY AND FUTURE WORK

The preliminary simplified Kalman algorithm that we have discussed has improved the performance of those NIST time servers that are synchronized using noisy telephone lines by about a factor of two. The result is not as good as the best connections, but the improvement suggests that it might be possible to realize additional improvements with a more comprehensive Kalman algorithm that explicitly modeled the frequency of the clock oscillator in addition to the statistics of the telephone connection. In particular, an algorithm that included approximately diurnal frequency variations might be a better design, because such variations, driven by changes in the ambient temperature of the time server, are almost certainly present. A method for including quasi-periodic contributions to the variance when the amplitude and phase of the admittance are not known and may not be statistically stationary has been described by Gelb **[5]**.

The network link between a server and a client is less stable and more difficult to characterize than the telephone-based ACTS links that are used to synchronize the NIST time servers to UTC(NIST), and the accuracy that can be realized is correspondingly poorer. Nevertheless, the same statistical considerations apply to that process. The measured time differences in this environment often have a character that is similar to the data in Figure 3, so that a combination of a Kalman algorithm with a more stable local oscillator may prove to be useful for these systems as well.

REFERENCES

[1] J. Levine, M. Weiss, D. D. Davis, D. W. Allan and D. B. Sullivan, 1989, *"The NIST Automated Computer Time Service,"* **J. Res. NIST, Vol. 94,** 311-321.

[2] J. Levine, 1995, "*An Algorithm to synchronize the time of a computer to Universal Time,"* **IEEE/ACM Trans. on Networking, Vol. 3,** 42-50.

[3] J. Levine, 2011, *"Timing in Telecommunications Networks,"* **Metrologia, Vol. 48,** pp. S203-S212.

[4] J. Levine, 2011, *"The Statistical Modeling of Atomic Clocks and the Design of Time Scales,"* **Rev. Sci. Instr.**, in press. See the discussion of equation 56.

[5] A. Gelb, 1974, **Applied Optimal Estimation,** The MIT Press, Cambridge, Massachusetts. See especially Chapter 4.