Method for Improved IoT Prognostics and Improved Prognostic Cyber Security for Enterprise Computing Systems

Kenny C. Gross, Mengying Li
Oracle Physical Sciences Research Center, Oracle Corporation, San Diego, CA 92121, USA
Kenny.gross@oracle.com mengying.li@oracle.com

Abstract - *Telemetry sampling rates from enterprise servers and for IoT asset prognostic monitoring are often constrained by hardware limitations in physical transducers and A/D digitizing firmware, and by hard-coded firmware in data acquisition instrumentation. The sampling restrictions impose challenges in terms of training advanced pattern recognition for prognostic applications such as Prognostic Cyber Security in enterprise and cloud data centers, and prognostic health management for end-customer IoT critical assets. No matter how slow the inherent sampling rate capabilities are for monitored assets, this paper introduces a novel empirical systematic and rigorous process to produce arbitrarily high telemetry sampling densities from assets for which such high telemetry sampling densities are physically and electronically impossible. This capability is achieved with no hardware or firmware modifications in any of the critical assets being monitored, and hence is backward compatible with legacy assets already in operation.*

Keywords: prognostic cyber security, internet-of-things, IoT, real time prognostics, anomaly detection, AI.

1 Introduction

For enterprise computer servers in cloud data centers, the maximum achievable telemetry sampling rates are constrained by standards that have evolved over the last two decades and will be very difficult to change. Sampling rates are constrained by the ILOM firmware that runs on the server's Service Processor (SP), by the IPMI interface standard, and by the I2C system bus. Today, for typical enterprise servers, the fastest possible sampling interval for physical telemetry readings that have been found to have prognostic significance (e.g. temperatures, voltages, currents, component power levels, fan speeds) is greater than 10 secs for some systems and as slow as once per minute on large servers containing up to a few thousand sensors for the "heavy iron" refrigerator-sized servers. As the number of sensors goes up exponentially with time (in fact, faster than Moore's Law for the last decade), the attainable sampling rates have gone down, thanks to only incremental improvements in bandwidth.

Similarly, the present sampling intervals achievable for "soft" telemetry metrics that have prognostic relevance for Quality-of-Service (QoS) assurance and for prognostic cyber security applications are as long as about 10 min for many important metrics (e.g. loads, utilizations, throughput metrics, queue lengths, transaction latencies, etc). These sampling rates were adequate in the past when only crude performance diagnostic aids were provided with servers to signify severe performance issues (e.g. thresholds to alert human Sysadmins or Service Engineers about exhaustion-of-resource problems).

Unfortunately, threshold-based warnings and diagnostics are "reactive" in nature...i.e. by the time a threshold limit has been exceeded, the problem is already severely underway (or the system is crashed). Because of the increasingly business-critical nature of enterprise and cloud computing, this endemic limitation of threshold-based diagnostics has motivated significant R&D inside Oracle and other systems vendors on machine-learning (ML) based prognostics, to proactively alert human Sysadmins and Services personnel of incipient anomalies, hopefully with enough lead time so that issues can be avoided or proactively remediated, well before end customers become aware of QoS issues or other customer dissatisfiers, and at the earliest possible time for proactive "indicators of compromise" (IOCs) for cyber security applications in business-critical systems.

Slow telemetry sampling rates introduce an even greater problem for ML based prognostic algorithms than for older crude threshold-based prognostics. For example, if an important class of QoS telemetry metrics can only come in at a rate of once every 10 min, this means an Alert to a QoS problem can appear as long as 10 min later than the system is experiencing problems. Although Alerts for threshold-based diagnostics are of only limited usefulness for avoiding customer dissatisfiers (again, because threshold based diagnostics are reactive in nature), getting the alert 10 min earlier is only marginally more useful than 10 min later, as the system is likely already in serious trouble anyway when a threshold is tripped. In other words, slow sampling rates do not make threshold-based diagnostics much worse.

By contrast, machine-learning prognostics have the already proven capability to alert human Sysadmins and Service Engineers hours and sometimes days in advance for slow degradation mechanisms, which is the problem realm where ML-prognostics holds its greatest promise. However, the effectiveness of Oracle's ML prognostics (in terms of minimizing false-alarm and missed-alarm probabilities, FAPs and MAPs) is very dependent upon sampling rates for the monitored metrics.

It is well known from prognostics research at Oracle to date that if a ML algorithm is used for a system with a high sampling rate for all monitored performance metrics, this ML algorithm will perform significantly better than if the same ML monitoring algorithm were used for an identical server configuration with a much slower sampling rate. Similarly, if a ML algorithm is trained with high-sampling-rate telemetry metrics, and then that trained algorithm is use to monitor the identical system but with a slower sampling rate for a real asset in a production environment, that ML algorithm will perform better because it is trained on high-sampling-rate telemetry data, versus if the identical ML algorithm were trained on slow sampling rate telemetry metrics. [The reason for improved prognostic performance with higher-sampling-rate training data is that the ML algorithm, especially those algorithms we are using from the class of mathematics called NonLinear NonParametric (NLNP) Regression, can much better "learn" the patterns of dynamical correlation between/among all the monitored metrics. See technical details in section below: "Why High Sampling Densities are Important".]

Because ML algorithms can perform much better if trained on high sampling rate telemetry metrics, even when the trained algorithms will be monitoring customer assets with slower sampling rates, it would be very desirable if one could "crank up" the sampling rate to arbitrarily high values for training of the ML algorithms.

For almost all existing enterprise servers, storage, and engineered systems, it is either impossible or impractical to "crank up" the sampling rates for internal digitized telemetry time-series metrics for the purposes of getting better training data sets for sensitivity-tuning and prognostic optimization of ML surveillance algorithmics for predictive anomaly detection and Prognostic Cyber Security goals. In almost all cases the digitized sampling rates are hard-coded into the low-level hardware registers and system firmware, in other cases there are no "knobs" accessible to end customers or to internal system-vendor engineers, only because system-bus bandwidths and IO bandwidths could be saturated if either humans or automated agentry were to "turn up" the sampling rates on telemetry time series.

In the future, industry standards may evolve to allow much higher sampling rates for telemetry variables. This will be a slow process. Moreover, it will be impossible to back-fit the $Bs in legacy systems with new internal system bus architectures. What is needed is an innovation that allows extraction of very high sampling rate thermal dynamics from standard architectures meeting present-day standards (I2C, IPMI, ILOM) and with no hardware modifications to the enterprise servers.

We describe in this paper a novel analytical innovation that allows "Telemetric Sampling Densification" that provides very accurate, fine-grained thermal dynamics for any standard enterprise computing servers that Oracle (or other systems vendors) make, even when the servers are constrained to slow sampling rates by industry-standard I2C, IPMI, or ILOM architectures.

The new innovation for boosting telemetry sampling rates presented herein is empirically based and is superior for any prognostic monitoring and cyber security applications where "reference testbed configurations," or prototype systems for IoT assets in the fields of utilities, oil-and-gas, manufacturing, or transportation, can be set up in an internal test/dev laboratory. This paper teaches a sophisticated experimental procedure, described in detail below, that enables "sampling densification" on enterprise servers and all types of IoT critical assets, even for systems for which faster telemetry sampling rates are physically/electronically impossible, and even when all available bandwidth pathways are near saturation and cannot be increased.

In this paper we introduce a new technique that provides high-accuracy, high-sampling-density real-time empirical telemetry for enterprise servers and IoT assets. These fine-grained high-sampling-density telemetry time-series signatures are created with a novel empirical technique we call "sampling densification" that enables extremely high resolution digitized time series telemetry from all types of internal physical transducers, without requiring any hardware modifications for standard architectures meeting present-day sampling and bandwidth protocol standards for data acquisition (DAQ) instrumentation. We are in effect getting high resolution, high sampling rate physical telemetry in systems with low-bandwidth industry-standard system bus and network-interface architectures through the new "sampling densification" innovation introduced below.

The primary use case for this innovation is for enhanced prognostics of IT systems and IoT critical assets, and enhanced cyber security of data center and cloud IT systems and network. However, this technique has important use cases in other industries as well because the technique enhances the prognostic performance of prognostic algorithms, including all classes of machine-learning (ML) and deep-learning (DL) algorithmics presently being evaluated for IoT anomaly detection and Prognostic Cyber Security. This new "Telemetric Sampling Densification" approach provides very accurate, fine-grained telemetry for business-critical and mission-critical assets for which prognostic algorithmics are being applied for early detection of the incipience or onset of degradation modes in physical assets, or malicious intrusion events in networked IT systems.

2 Why Sampling Density is Important

If idealized computing systems existed wherein all the interactions between/among the myriad telemetry signatures were linear, then sampling density would not be a big issue (in fact, we could slow down the telemetry sampling and enjoy a reduction in compute cost for prognostics). However, in today's enterprise computing servers there exist highly nonlinear relationships between/among the telemetry metrics. Just a few examples of reasons for the high nonlinearities in today's servers:

CPU temperatures vs CPU core current and voltage vs CPU operating frequencies:

In the "old days" (i.e. prior to about 5 yrs ago) all CPU chips dissipated heat in direct proportion to the "switching activity" going on inside the CPU. Now that CPU technology is so small, there is significant "leakage power" inside the CPUs. Leakage power is exponentially dependent upon CPU temperature. So now days, the current, voltage, and frequency for the CPUs is a very complex nonlinear relationship between compute load, fan speed (which affects CPU temperature and hence leakage power), external ambient temperature, and even the altitude of the data center (because air at sea level has significantly greater cooling capability than thinner air for example in mile-high Denver). This creates highly nonlinear relationships between the hundreds (to thousands) of physical telemetry time series, which in-turn are correlated with the various load and throughput "soft" telemetry metrics.

Similarly, QoS telemetry metrics have reasonably linear interrelationships when there exists lots of free memory in systems. However, when memory-intensive applications start to get close to the limit of available free memory, applications automatically start swapping to slower SSDs or to much slower spinning HDDs. This introduces highly nonlinear relationships between/among many "soft" telemetry time series.

As a final example of nonlinearities between/among the various classes of telemetry: when IO pathways are "wide open" inside a server and at the interfaces between the IT systems and the external networks, then there are nice linear relationships between "flow" related telemetry and interarrival times for packets (IAT signatures). However, as available bandwidth channels become saturated, there is now a complex nonlinear relationship between "flow" related metrics and latency (or IAT) metrics.

Because of complex nonlinear relationships between/among the thousands of telemetry time series monitored by advanced prognostic algorithms, sampling density now matters a lot in terms of prognostic effectiveness for detecting anomalies either in QoS metrics (for QoS prognostics) or for Prognostic Cyber Security applications. If we had simple, idealized systems where all relationships were always linear among monitored time series, then we could sample sparsely and just linearly interpolate between signals. However, because of the above (and other) sources of highly nonlinear phenomena in today's enterprise servers and networks, we need the highest sampling density we can get so that the pattern recognition algorithms can robustly and accurately "learn" the patterns of interaction across thousands of monitored time series metrics. Moreover, because of the nonlinearities, simple interpolation will do no good for "filling in the blind spots".

What the new empirical procedure brings is the capability to "densify" the sampling rates for all telemetry time series in IT systems and associated networks. Not "analytical" densification (i.e. interpolation), but "empirical" densification. This technique allows extremely fine granularity, in effect attaining arbitrarily high sampling rates even for systems for which it is physically and electronically impossible to increase the sampling rates. This “sampling densification” innovation allows robust, accurate training of advanced pattern recognition algorithmics for high-sensitivity anomaly detection with ultra-low false-alarm and missed-alarm probabilities (FAPs and MAPs), for business critical applications such as QoS Prognostics and Prognostic Cyber Security applications for enterprise servers, and for IoT critical asset prognostic health monitoring.

2.1 Technique Description

See Fig 1. which shows real thermal telemetry from a presently shipping thermal-mechanical asset for which high-sensitivity prognostic anomaly detection surveillance is being designed. The fastest telemetry sampling rate possible for this class of assets is on the order of ~100 second intervals between observations. During this telemetry period shown in Fig 1, the sampling rate is far too coarse to enable accurate characterization of the rapid dynamics for training of advanced Predictive Analytics either for prognostic QoS anomaly detection or for prognostic security applications for the server.

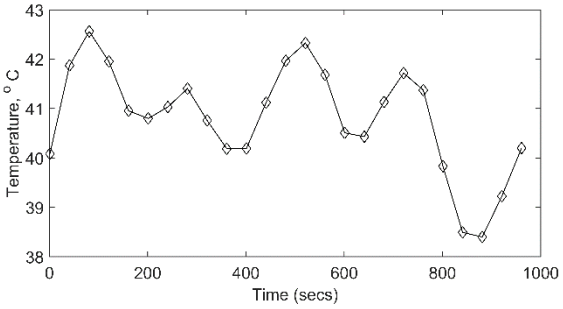
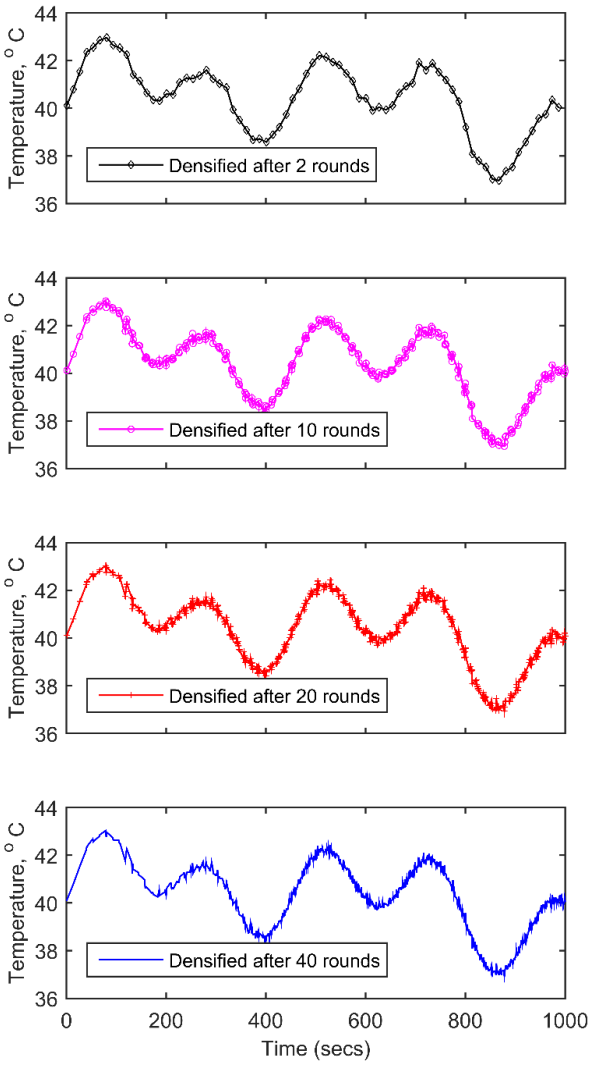


Figure 1. Sparsely sampled thermal signal

We introduce here a systematic empirical procedure that enables high-resolution, fine-grained definition of the telemetry dynamics for any server product even when the telemetry sampling rate on that product is constrained to very coarse sampling. The high-accuracy fine-grained telemetry signatures can then be used for optimized training of ML and DL based Predictive Analytics algorithms. Even though the optimized PA algorithms will then later be used on systems in the data center with lower sampling rates, the prognostic performance is tremendously enhanced by the fact that the prognostic algorithmics are trained, tuned, and optimized with high-sampling-rate telemetry metrics.

We begin by generating a reproducible deterministic dynamic load profile that exercises the CPU, memory, and IO workloads through as wide a range as possible. Note: exercising compute, memory, and IO dynamics through as wide a range as possible is not a requirement nor an enabler for this telemetry sampling densification technique. This technique will work equally well, even for a lightly-loaded system testbed configuration. Instead, the desire to exercise test systems through the widest range possible yields the most robust algorithmics for prognostics. We typically stress test systems dynamically between the maximum possible range, i.e. from completely idle, to totally maxed out on CPU, memory utilization, and saturated IO channels, with lots of dynamic variations between those min/max ranges (to best characterize patterns between/among all classes of monitored telemetry signals). For whatever dynamic stress exercisors are available to run on the testbed configuration, the ROI from this technique presented herein lies in the fact that the signal dynamics are far more accurately defined through dense, fine-grained empirical observations. For this technique, we establish a fixed time window, W, during which the dynamic exerciser scripts will generate a deterministic (and hence exactly replicateable) load profile that exhibits rich dynamics in CPU utilization, memory utilization, and IO metrics. It is desirable to set the width of W to a prime number of seconds (for example 631 secs).

Figure 2. Sparse signal and densified signal after different rounds of densification



At the end of time window W, the exact same deterministic load profile is going to be run again. The reason to set the window width time of the replicatable dynamic load profile time to a prime number will be described in the following subsection.

2.2 Purpose for prime-number profile-window-width (W) time

The telemetry sampling rates that are hard-coded into systems and networks are almost always some fixed number of time units with a uniform sampling rate, such as once every 30 secs or 60 secs. When this is the case, if the window-width W were to be some integer multiple of the sampling rate, e.g. exactly 10 minutes, then the samples in the technique would unintentionally overlay one another. By picking a window-width W that is a prime number of seconds, we minimize the likelihood that when we run a reasonable number of experimental replications (say several dozen replications), that any samples will accidentally overlay one another. Note that if the sampling rate for the telemetry is not fixed, but can be independently specified, then we would make the dynamic-stress-profile window-width W one prime number (e.g. 631 secs) and the sampling interval a second but different prime number (e.g. 79 secs). In this case, samples would not accidentally overlay one another until 79*631 secs or about 14 hrs, which is far longer than needed for the technique below.

Now we generate numerous replications of the deterministic load profile that are long enough to span a maximal range in CPU, memory, and IO "stress" levels and that span a prime-number of seconds. By "numerous replications", ten successive window replications would be sufficient, but the more dynamic profile replications one generates, the higher will be the resolution and accuracy for the "densified" telemetry signature characterizations. We recommend several dozen replications and we have used 40 replications in our demonstration (see example in Fig. 2 for raw data and Fig. 3 for telemetry data that has been normalized to remove any variations from external ambient temperature).

However many replicated profile windows are generated (N), we now analytically "cut up" the telemetry time series into N "chunks", each of which is slightly longer than the window-width time W. Extra time is needed at the beginning and end of each chunk for the "coherence-optimization" step, wherein each chunk gets analytically slid forward and backward to optimize its coherence with respect to an iteratively defined "Reference Curve", as defined in the densification-procedure flowchart below.

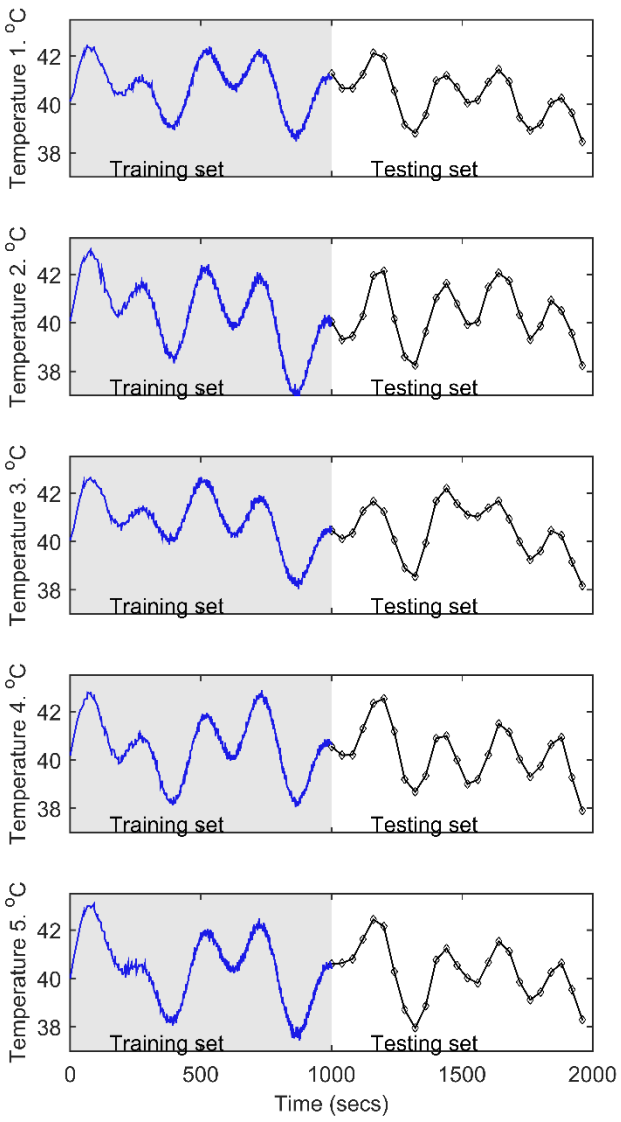


Figure 3 Training set and testing set of MSET Prognostics

It will become apparent that the Reference Curve starts out with very poor resolution and coarse granularity, but with each iteration of the procedure below, the Reference Curve attains increasingly higher resolution and increasingly finer grained definition, which we call "telemetry sample empirical densification."

It is noteworthy to point out the present technique is fundamentally different from numerical interpolation. Numerical interpolation is an analytical technique that "fills in" samples between measured observations. No matter how sophisticated nor how "intelligent" the interpolation technique is, the "blind spot" in between measured observations is not made more accurate by filling in values that have no foundation in measurements. The "sampling densification" procedure

is an empirical (vs analytical) technique and the sampling densification is based on real measurements that yield in the end a highly accurate fine-grained time-series with an arbitrarily dense sampling rate even for systems for which a high sampling rate is physically/electronically impossible. [By "arbitrarily high density"...the prognostics team can pick N to be as high as desired, for as long as the experimental testbed configuration is available.]

2.3 Detailed Algorithmic Description

First iteration:

Step 1: Generate dynamic load profile through a wide range of stress intensity (ideally spanning the envelope from completely idle to totally maxed out in CPU utilization, memory utilization, and IO intensity) on the target system such that the time window of the dynamic load profile is a prime number of seconds and start system monitoring with the Continuous System Telemetry Harness (CSTH) telemetry [Refs 1-3]. A prime number is selected for the load profile-window so that if the system sensor data is uniformly sampled, the CSTH samples from the various load profiles are obtained at different times relative to the start of the load profile window W. [Implementation note: if the system allows specification of the sampling time, then it is further desirable to set the sampling intervals to also be a prime number of clock units, but a different prime number than the profile window width time W.]

Step 2: Pick the first complete dynamic load profile as the Reference Exerciser Profile.

Step 3: Pick the second chunk and slide the data forward/backward to optimize fit (maximum cross correlation) with the Reference Exerciser Profile. Merge this second chunk with the Reference Exerciser Profile to generate an improved Reference Exerciser Profile.

Step 4: Repeat Steps 2 & 3 above for the subsequent chunks, each time merging the resulting chunk with the Reference Exerciser Profile and updating it.

Step 5: Once all the N available chunks are coherence-optimized and merged, perform a second pass through all the chunks starting with the first chunk. In this second pass, before optimizing for a selected chunk, the data generated from the selected chunk is removed from the Reference Exerciser Profile. After optimizing and realigning, the chunk is merged back into the reference Exerciser Profile. This second pass is done so that the effects of any abnormalities or artifacts that may have been present in the earliest chunks during the first iteration are minimized.

Step 6: Take the final Reference Exerciser Profile and convert the timestamps in all chunks to times (in seconds) relative to the beginning of the chunk.

Step 7: To smooth out this data, we now apply in the preferred embodiment a moving-window ensemble average function with a width of 20 samples.

Step 8: Take the data from Step 7 and perform iterative upsampling of the data to make the time intervals exactly uniform. (Note: Step 7 produces densified sampling, but the sampling intervals are not necessarily uniform. Step 8 maintains the high accuracy from Step 7 but transforms the sampling intervals to be exactly equal.)

Step 9: The data from Step 8 is the final densified data.

See Fig 4 showing the accuracy of prediction of a cluster of correlated signals as the sampling densification increases. The pattern recognition tool used in this investigation is the Multivariate State Estimation Technique (MSET) [Refs 4-7]. Plotted in this figure is the root-mean-square-error (RMSE) for MSET estimates, which is a measure of the uncertainty if the trained MSET model, as a function of the number of empirical densification rounds.

Figure 5 shows a pair of illustrative results of the "Telemetric Sampling Densification" procedure. This new procedure effectively produces 1-second sampling resolution from data that, because of industry standard conventions, had a minimum sampling interval of 50 seconds. Figure 5 contrasts the raw telemetry data available from industry-standard system bus and ILOM sampling capabilities (in red), superimposed on the very high resolution, fine-grained dynamic telemetry signature attainable with the new "Telemetric Sampling Densification" innovation introduced in this disclosure.

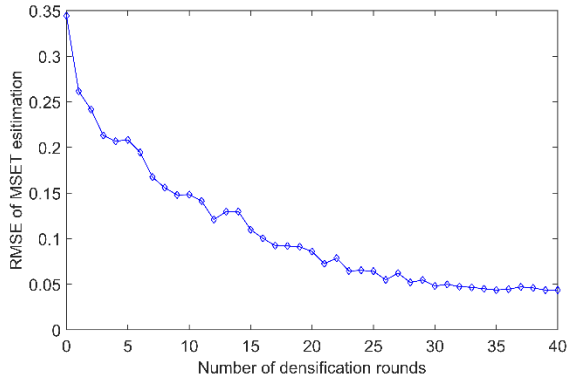


Figure 4 RMSE of MSET estimation of all 5 signals

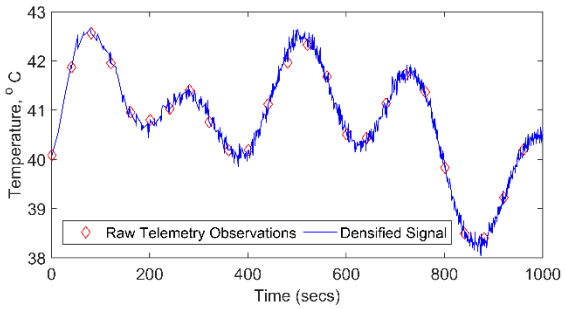


Figure 5. Raw telemetry data and densified signal.

The reader is invited to contrast Figure 1, showing the sampling rate attainable on present generation enterprise servers that have present industry-standard constraints, with the "Telemetric Sampling Densification" fine-grain high-accuracy thermal-dynamic profiles attainable by the new Oracle innovation introduced herein (Fig. 5). This technique allows fine-granularity high-resolution telemetry dynamic characterization for training, tuning, and optimizing prognostic algorithmics, in effect attaining arbitrarily high sampling rates even for IT systems and networks for which it is physically and electronically impossible to increase the sampling rates.

Conclusion

The Telemetry Sampling Densification technique presented in this paper provides a capability to obtain rich, high-sampling-density dynamic signatures for any digitized time series telemetry from testbed and/or asset prototype configurations. The technique allows arbitrarily high telemetric sampling densities for training of advanced Machine Learning and Deep Learning prognostic algorithms, which results in higher prognostic sensitivity for detecting incipient anomalies earlier, and with lower false-alarm and missed-alarm probabilities (FAPs and MAPs).

References

[1] "Electronic Prognostics Through Continuous System Telemetry," K. C. Gross, K. W. Whisnant and A. Urmanov, ***Proc. 60th Meeting of the Society for Machinery Failure Prevention Technology,*** Virginia Beach, VA (April 2006).

[2] "Prognostics of Electronic Components: Health Monitoring, Failure Prediction, Time To Failure," K. G. Gross, K. W. Whisnant and A. M. Urmanov, ***Proc. New Challenges in Aerospace Technology and Maintenance Conf. 2006***, Suntec City, Singapore (Feb 2006).

[3] "Electronic Prognostics Techniques for Mission Critical Electronic Components and Subsystems," K. C. Gross, K. W. Whisnant and A. M. Urmanov, ***Proc. 2006 Components for Military and Space Electronics Symposium,*** Los Angeles, CA, (Feb 2006).

[4] R. Singer, K. C. Gross, J. Herzog, R. King, S. Wegerich, "Model-based nuclear power plant monitoring and fault detection: Theoretical foundations," in ***Proceedings from the Intelligent System Application to Power Systems Conference,*** pp. 60-65, July 1997.

[5] K. C. Gross, S. Wegerich, and R. M. Singer, "New artificial intelligence technique detects instrument faults early," ***Power Magazine***, vol. 42, no. 6, pp. 89-95, 1998.

[6] "Failure Avoidance in Computer Systems," A. Urmanov and K. C. Gross, ***Proc. 59th Meeting of the Society for Machinery Failure Prevention Technology***, Virginia Beach, VA (Apr 18-21, 2005).

[7] "MSET Performance Optimization for Proactive Detection of Software Aging," K. Vaidyanathan and K. C. Gross, Proc. 14th IEEE Intn'l. Symp. on Software Reliability Eng. (ISSRE'03), Denver, CO (Nov. 2003).

[8] K. J. Cassidy, K. C. Gross, and A. Malekpour, "Advanced pattern recognition for detection of complex software aging phenomena in online transaction processing servers," in ***Proc. IEEE Intnl. Performance and Dependability Symposium,*** Washington, D.C., June 23 - 26, 2002.

[9] K. Vaidyanathan and K. C. Gross, "Proactive detection of software anomalies through MSET," *Proc. IEEE Workshop on Predictive Software Models (PSM-2004)*, Chicago, Sept 17-19, 2004.

[10] K. C. Gross, R. M. Singer, S. W. Wegerich, J. P. Herzog, R. VanAlstine, and F. Bockhorst, "Application of a Model-based Fault Detection System to Nuclear Plant Signals," Proc. 9th Intnl. Conf. On Intelligent Systems Applications to Power Systems, pp. 66-70, Seoul, Korea (July 6-10, 1997).

[11] "Multivariate State Estimation Technique (MSET) Surveillance System," J. P. Herzog, K. C. Gross, S. W. Wegerich, and R. M. Singer, Appendix H of ***On-Line Monitoring of Instrument Channel Performance***, **TR-104965**, EPRI (Oct 1998).