

Software Systems Research – Portfolio Review

Dr. Nelson R. Manohar Alers

nelsonmanohar@yahoo.com

nelsonmanohar.sphosting.com/research



Outline of the Talk – Revisited

- Background [10%]
- Computer-Supported Collaboration [25%]
- Dynamically Customized Web Touring [25%]
- **Multimedia Computing Networking [35%]**
 - Building Robust Network Performance Indicator
 - Distributed Resource Management for Multimedia
- Wrap-Up [5%]



Network Superheterodyne

“Building Robust Network Performance Indicators”

Work at IBM Thomas J. Watson Research Center

“Applying Statistical Process Control to the Adaptive Rate Control Problem”, by Manohar, Nelson R.; Willebeek-Lemair, Marc H.; Prakash, Atul, in Proceedings of Multimedia Computing and Networking Conference, pp. 45-60, San Jose, CA, January 1998.



Motivation

- **Multimedia Computing Networking**
 - **handling the impact of multimedia over the network**
 - e.g., session-oriented, multimedia flows, end-to-end QoS reqs., resource management, value-asset model, etc.
 - **handling of the “impedance mismatch” that exists**
 - between multimedia applications
 - and (to-be provisioning, present or future) networks
 - **we would like the resulting mechanisms or building blocks**
 - to be robust
 - easy to implement
 - low (signaling and tracking) overheads



Related Work

- mechanisms to handle “impedance mismatch”
 - (traditionally) from applications to the network:
 - inducing multimedia application reqs. into networking middleware
 - for example, diffserv (TCP), multicast (routing), etc.
 - (but also) from network to multimedia infrastructure:
 - enhancing network capacity (internet2, vBNS, etc.)
 - **enhancing the intelligence of multimedia infrastructure to adapt to the network state (RSVP, QoS, etc.)**
- network state measurements
 - (active) network probing
 - probe-and-adapt: short-term fluctuations, unnecessary adjustments
 - (passive) network and web traffic characterization
 - (web) spatial/temporal stability: different time-scale components
 - ethernet/web traffic fractal: similar shape regardless of timescale



Factoring Process Variability on Sessions

■ From:

■ probe-and-adapt

- fast indicators
- very low setup cost

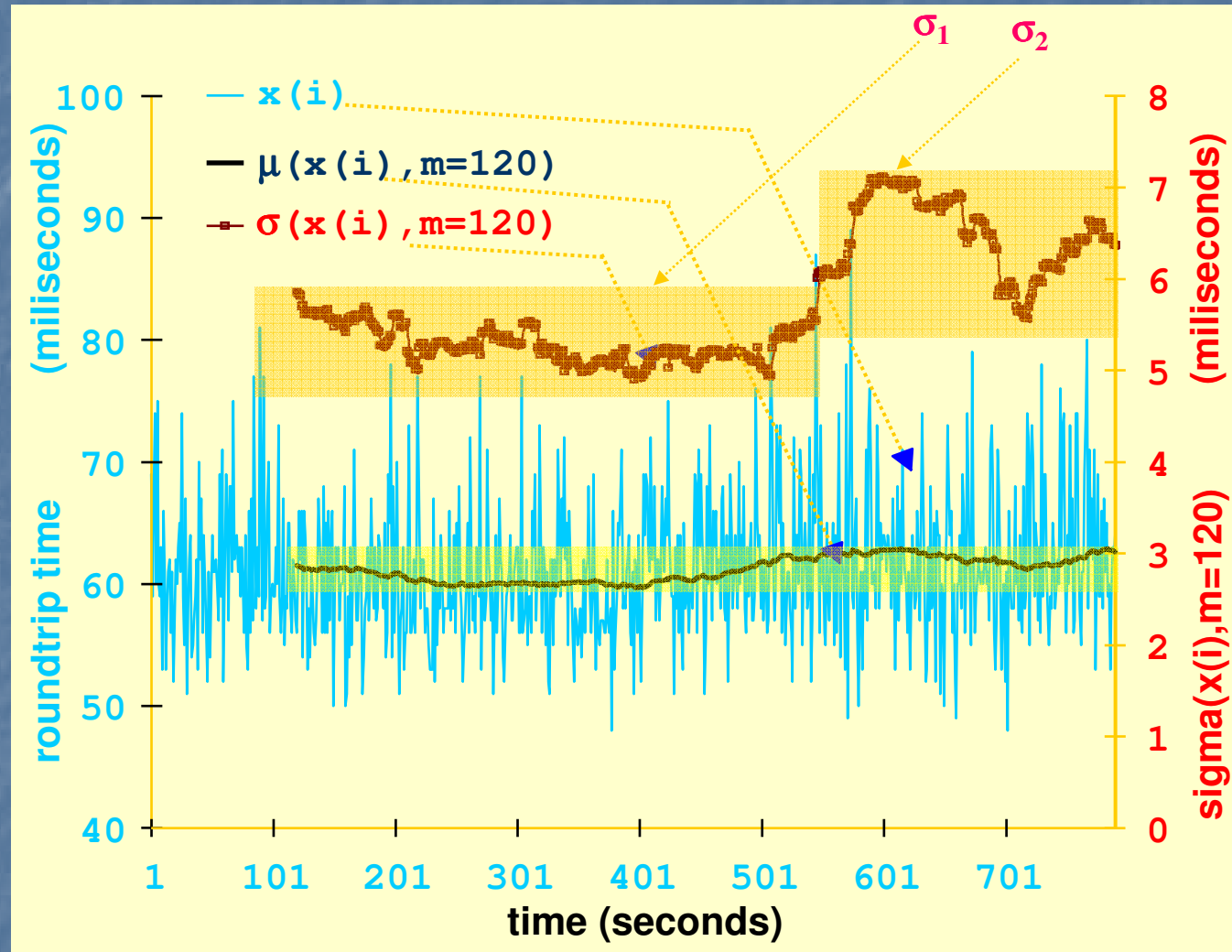
■ no confidence

■ To:

■ process state

- forecast-strength
- some setup cost

■ confidence analysis



RTT network probe: one apparently stable process mean, but upon examination variability states (i.e., a process shifts).



Network Performance Indicator

- **Reliable process performance indicators (PPI)?**

- **robust forecasts**

- strength of sampling and smoothing
- capable of associating confidence to forecasts

- **assimilation of process variability**

- to recognize significant changes

- **generation/tracking of process state**

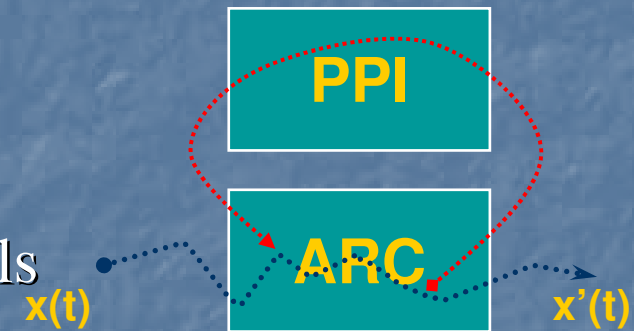
- to understand where we stand

- **Important to adaptive applications**

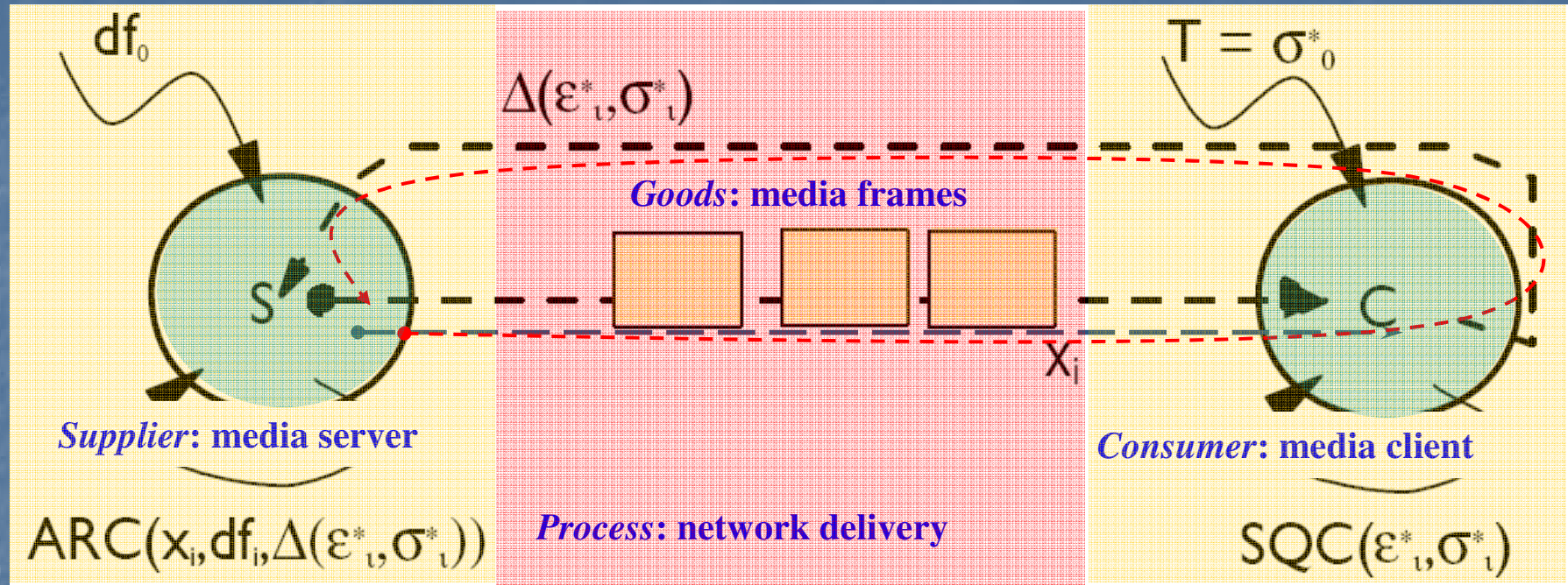
- atop (i.e., guiding) prove-and-adapt protocols
- such as Adaptive Rate Control (ARC)

- **Statistical Process Control (SPC) as PPI-kernel**

- typically long-term horizon, industrial processes
 - run-to-run feedback control vs. online-SPC
 - centralized online-SPC vs. distributed online-SPC
- adapt online-SPC for distributed ARC problem



Statistical Process Control Formulation

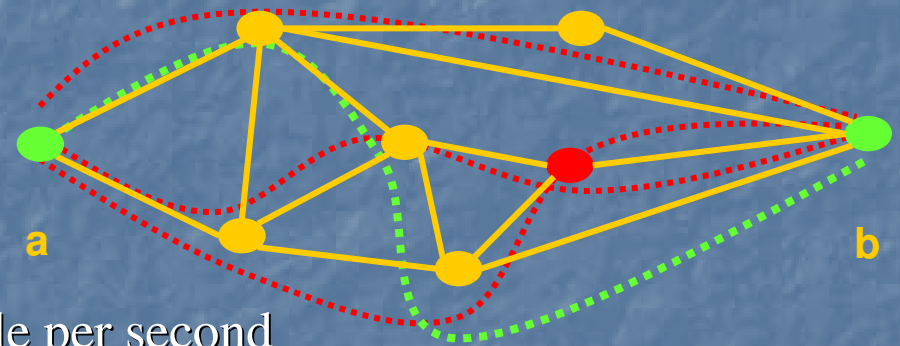


- **Supplier: adaptive rate control**
 - to match production to delivery
- **Process: network delivery**
 - *goods*: media packets, frames, buffers
- **Consumer: smoothing problem**
 - adapts delivery to presentation
- **SPC: end-to-end process performance**
 - quantizes performance measurements
 - into statistical process state – a quality indicator
 - then drives $ARC()$ with such quality indicator

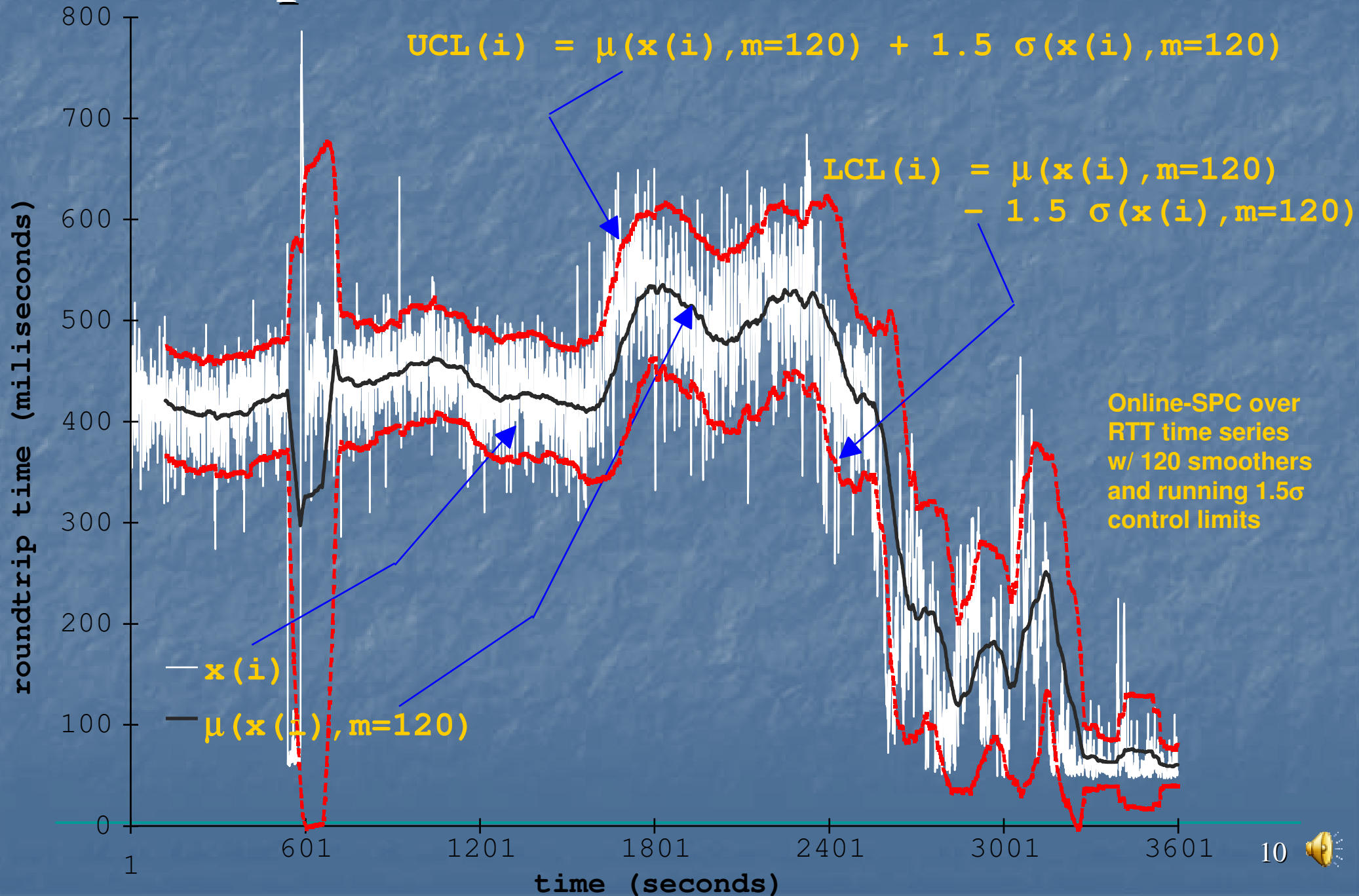


Network Measurements – Setup

- (active) network probing and monitoring
 - network indicator: RTT roundtrip delay (measure of congestion along path)
 - e.g., available bandwidth, **network congestion, stability, jitter**
- probing methodology (time series acquisition)
 - end nodes: (Univ. Michigan at A2) - (Univ. North Carolina – CH)
 - network probe: ping (ICMP)
 - test duration: 3600 seconds
 - timescale: seconds to minutes
 - smoother: 120/30 UWMA smoother
 - sampling frequency: 1 random sample per second
 - sampling load: negligible (40ms to 400ms) with respect to sampling frequency (1 second) with respect to available network bandwidth
 - sampled distribution: no constraining assumption due to law of large numbers, random sampling, temporal/spatial stability, and network fractality
- **bottleneck spots, router instability, underlying distr.**
 - timescale leads to aggregation effect on multiple path routing and corresponding random sampling of underlying distribution(s) across i -th samples

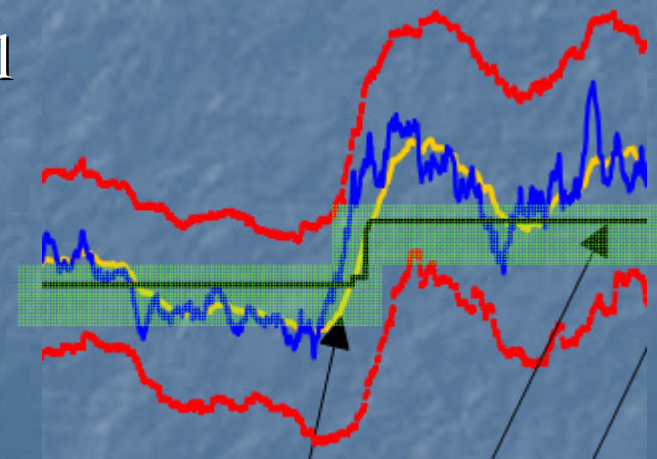
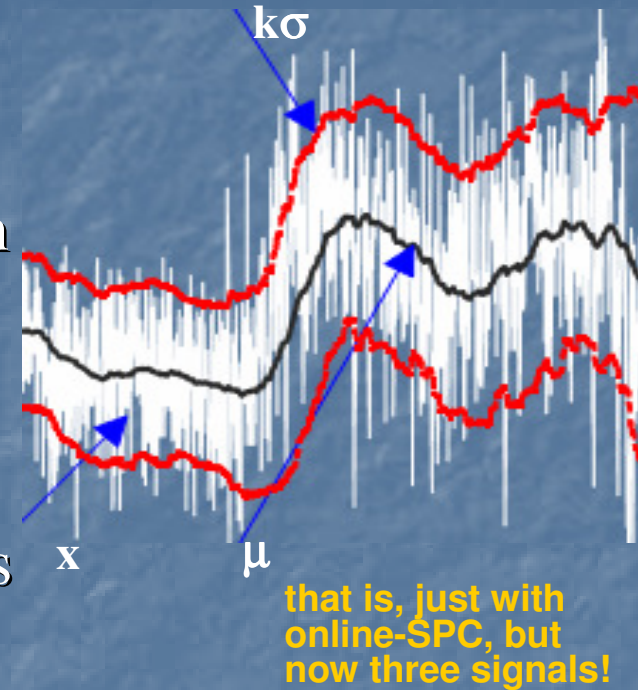


Online-SPC Floating Performance Envelope



Process State – Basic Idea

- **floating performance envelope**
 - tracking window over process state (μ , σ)
 - under associated statistical confidence region ($k\sigma$)
 - reacts to statistically significant changes
- **process stability indicator**
 - quantizes performance into statistical process state
- **providing context for state changes**
 - between **fast** (fractional process state) signal
 - *wrt* **slow** (full process state) signal
- **compare fast against stable signal**
 - approx. same versus significantly different
 - two types of **piece-wise linear** segments
 - **horizontal** segments (process state)
 - **linear slopes** (process changes)
 - building block for adaptive infrastructures



Online-SPC Process State Tracking Kernel

Running Window Indicators

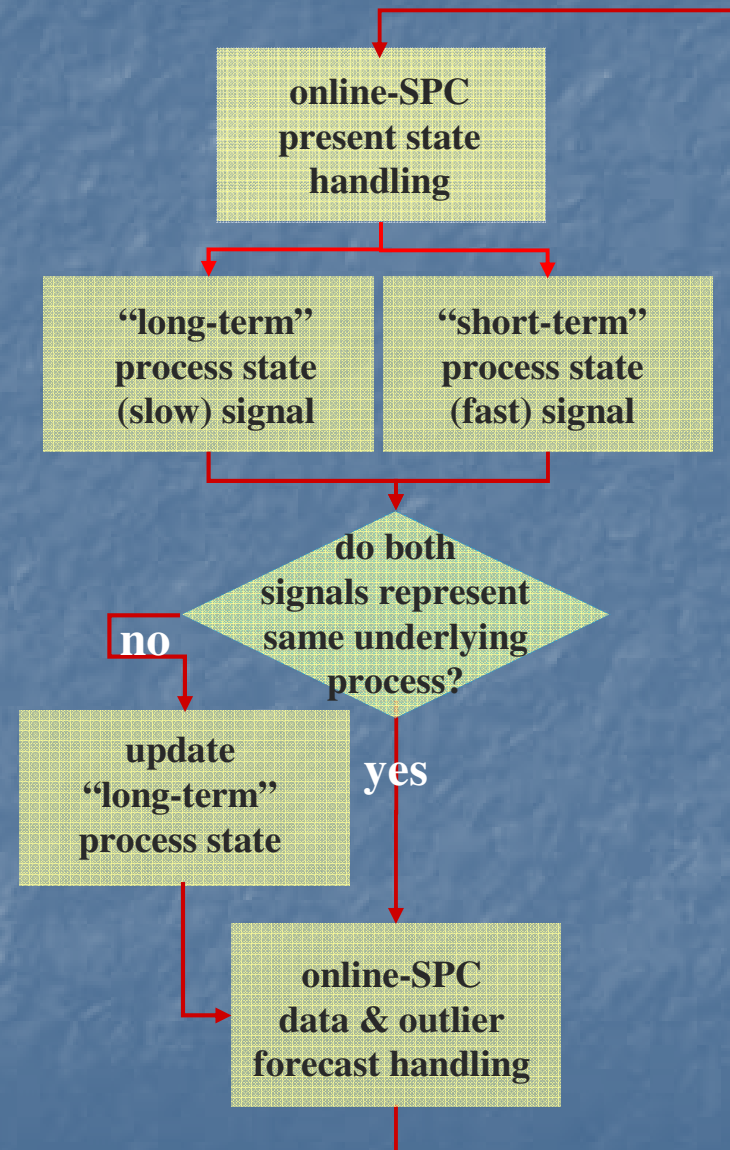
- UWMA(x_i, m) smoothers
- $\mu(i, m') = \mu(x_i \dots x_{m'-i})$
- $\mu(i, m) = \mu(x_i \dots x_{m-i})$
- $\sigma(i, m) = \sigma(x_i \dots x_{m-i})$

Hypothesis Testing

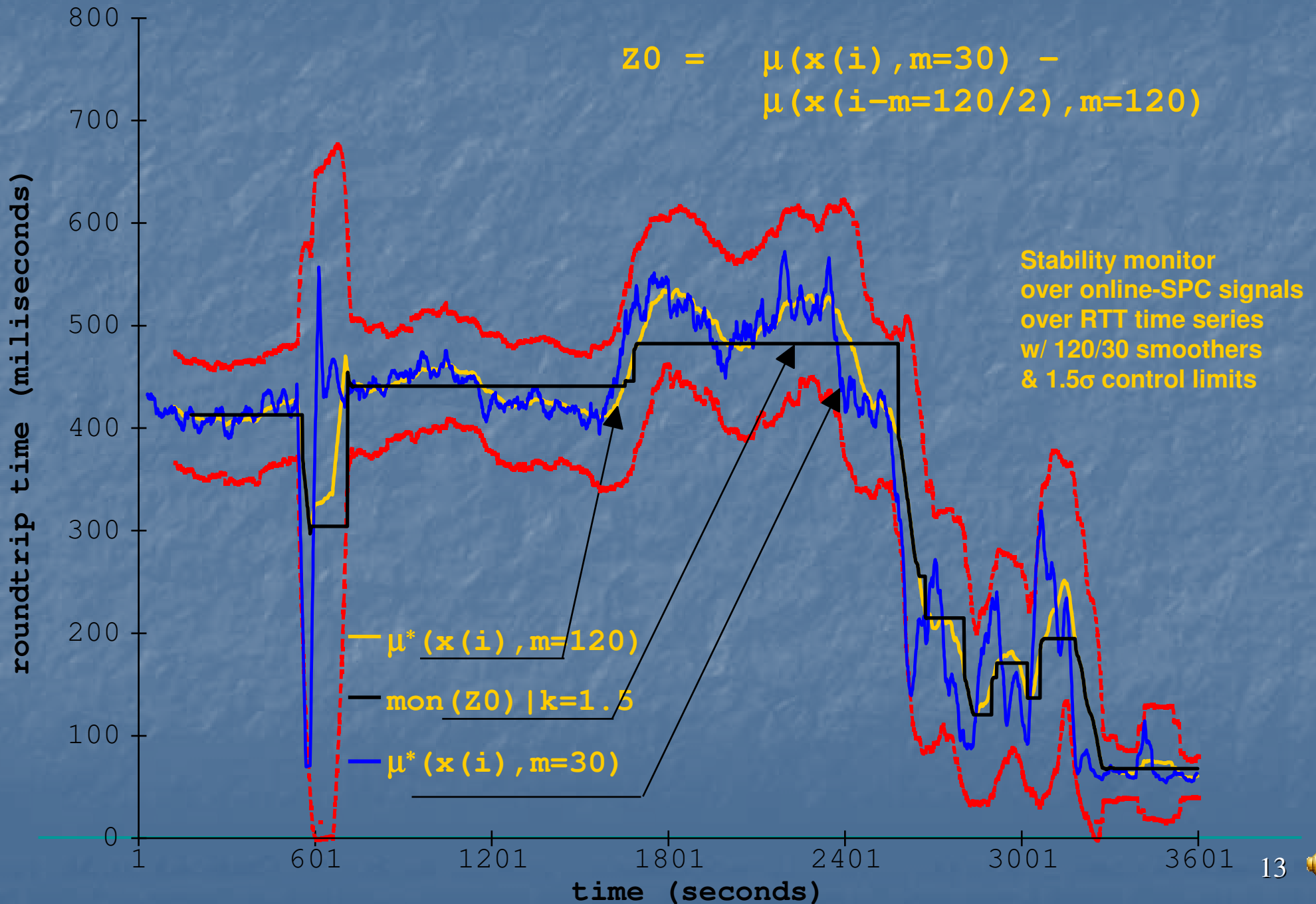
- $H_0 : \mu(i, m') = \mu(i, m)$
- $Z_0 = \mu(i, m') - \mu(i - m/2, m)$

Process State Generation

- if $|Z_0| < k * \sigma(i, m)$
- then $mon_i = mon_{i-1}$
- else $mon_i = \mu(i, m)$
- $mon_{i+1}^* = mon_i$
- $error = mon_i^* - mon_i$



Network Stability Monitor



Stationarity Test: Comparison of Sampled Means

$$Z_0 = \left\{ \begin{array}{l} \mu(i, m' = 30) \\ \text{fast signal (front - end)} \end{array} \right\} - \left\{ \begin{array}{l} \mu(i - \frac{m}{2}, m = 120) \\ \text{slow signal (\sim uncorrelated past)} \end{array} \right\}$$

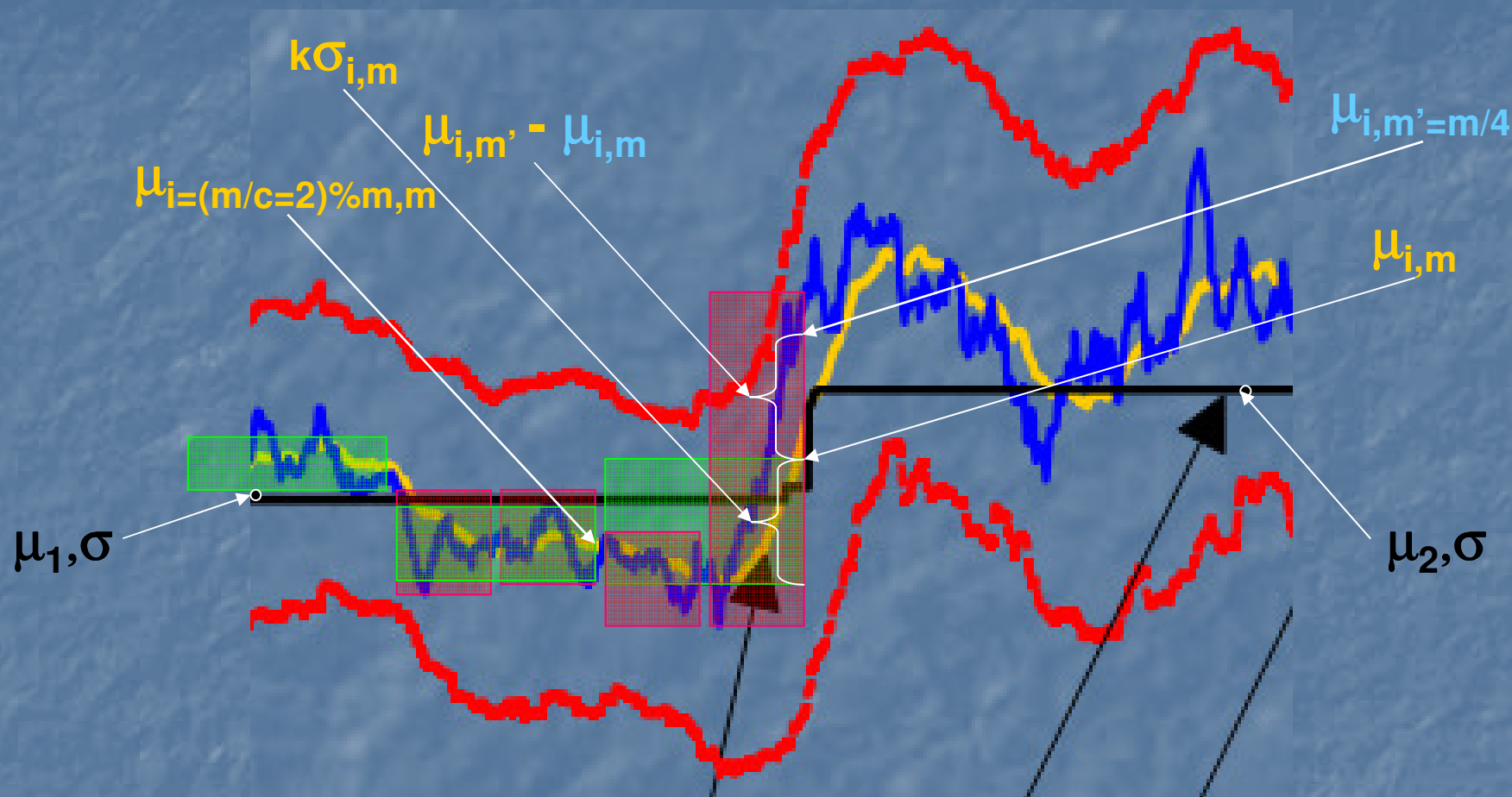
if ($|Z_0| < k \sigma(i, m=120)$)

- Hypothesis tested by comparing two indicators
 - (a) slow signal and
 - (b) fast signal
- This resulting in two process states
 - (a) stationary state or
 - (b) process change
- Sensitization parameters
 - m' (fast signal), m (slow signal)
 - k (confidence), m/c (decorrelator)



Network Performance Envelope:

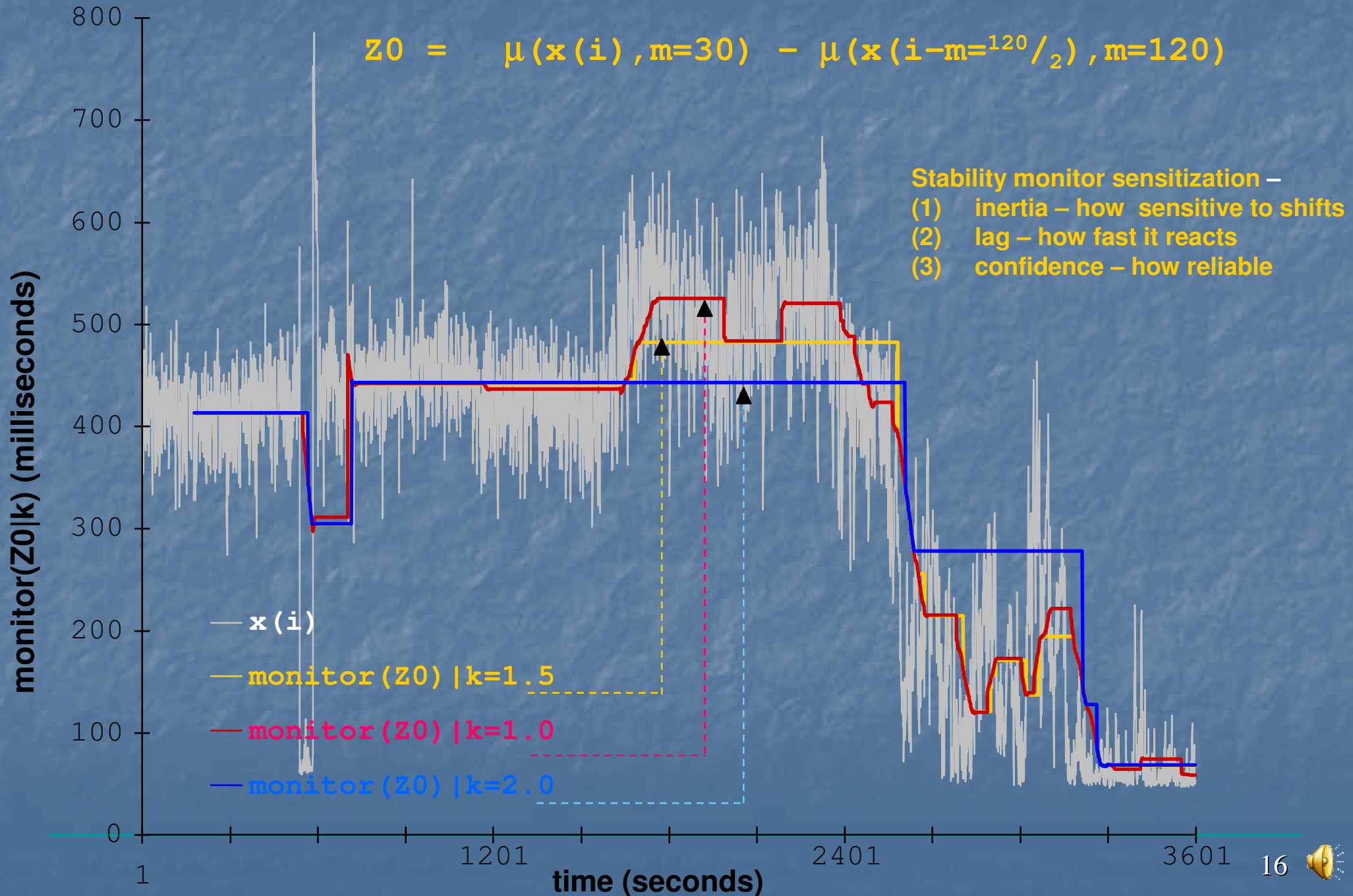
Equality Between Two Population Sampled Means



- *t*-test on unknown mean and unknown variance for a \sim normal population with test statistic
- at $(\alpha=0.5, m=120) \rightarrow t \approx 3.3$

$$t_0 = \frac{\mu(i, m) - \mu(i - m/2, m)}{\sqrt{\frac{\sigma(i, m)^2 + \sigma(i - m/2, m)^2}{m}}}$$

Sensitivity Parametrization



Comparative Analysis Test Space

■ Parameters

- m – size of the moving window outlook for the slow signal
- m' – size of the moving window outlook for the fast signal
- K – number of sigma levels used to clip outliers from the input time series
- k – approx. number of sigma levels used to recognize as statistical significant the current variability on the state tracking signal (i.e., how sensitive to process state shifts)
- c – point ($t - m/c$) into the recent's past of the slow moving used to decorrelate the windows of the slow signal from the fast signal

■ Constraints

- $1 < K \leq 3$
- $1 < k \leq 3$
- $m' < m$
- $c < m$

Test Case Structure

■ input time series

- approximately normal distributed random variable signal
- 3600 samples – representing random 1-second-spaced samples

■ three process state shifts (μ, σ)

- RTT ($\mu=80, \sigma=10$) from t=2 to 1201
- RTT ($\mu=160, \sigma=20$) from t=1202 to 2401
- RTT ($\mu=40, \sigma=5$) from t=2402 to 3601

■ signal recognition setup

- state recognition lag test – square wave signal
- state recognition accuracy test – process state shifts
- tracking error test – random generated signal

Comparison Metrics

■ state monitor's overall accuracy

- monitor's sum of error squares
- $[\text{mon}(\text{tlm}') - \text{RTT}(t)]^2 + [\text{mon}(\text{tlm}') - \text{mon}(\text{tlm})]^2$

■ state monitor's fractality

- monitor's number of states
- monitor's standard deviation

$$\sum_{i=1}^3 \left[\left| \hat{\mu} \left[\text{mon}(t, m') \right] - \mu \left[\text{RTT}(t) \right] \right|^2 \right]_{t \in R(i)}$$

■ monitor's state tracking accuracy

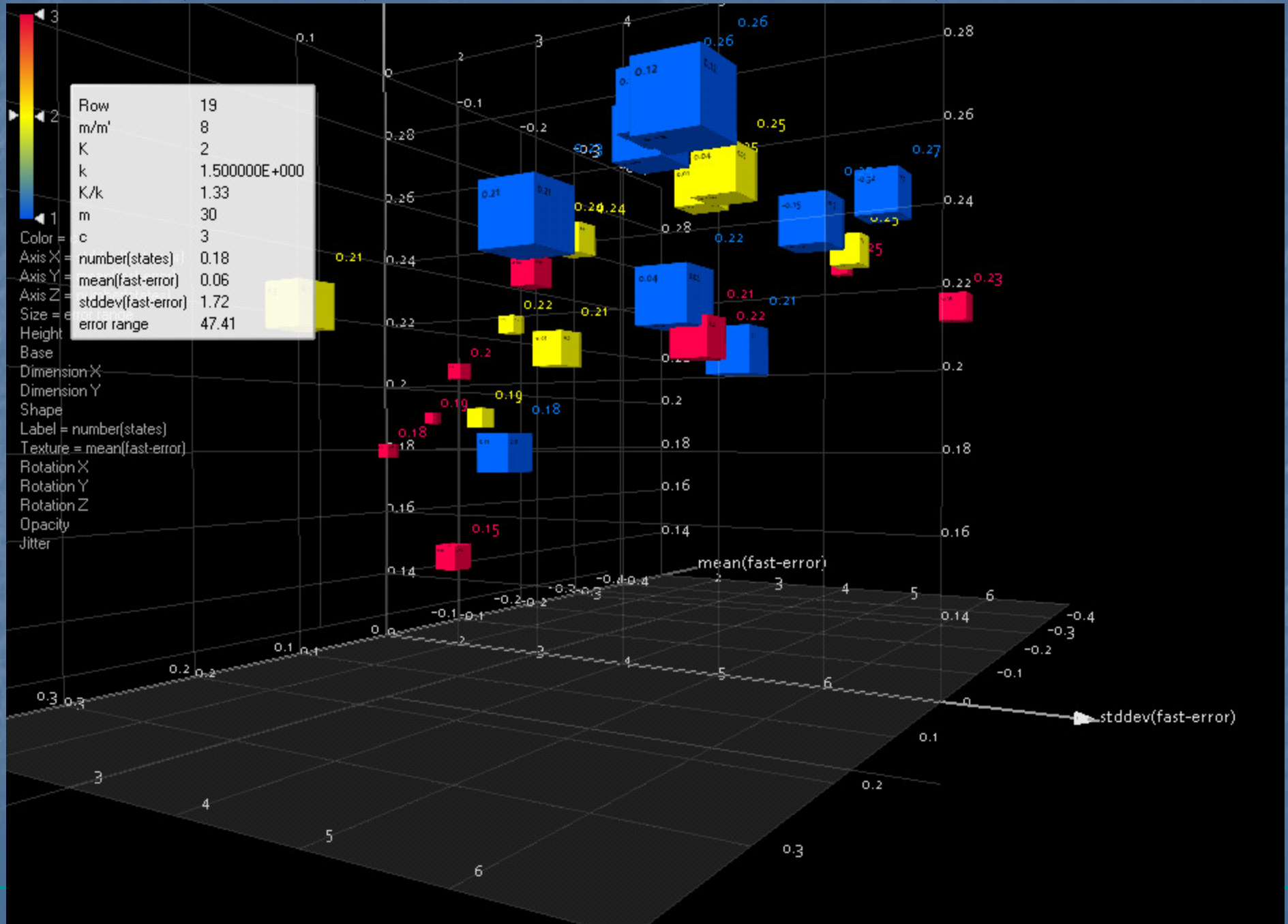
- R1: RTT ($\mu=80, \sigma=10$) from t=2 to 1201
- R2: RTT ($\mu=160, \sigma=20$) from t=1202 to 2401
- R3: RTT ($\mu=40, \sigma=5$) from t=2402 to 3601

Findings (to be updated)

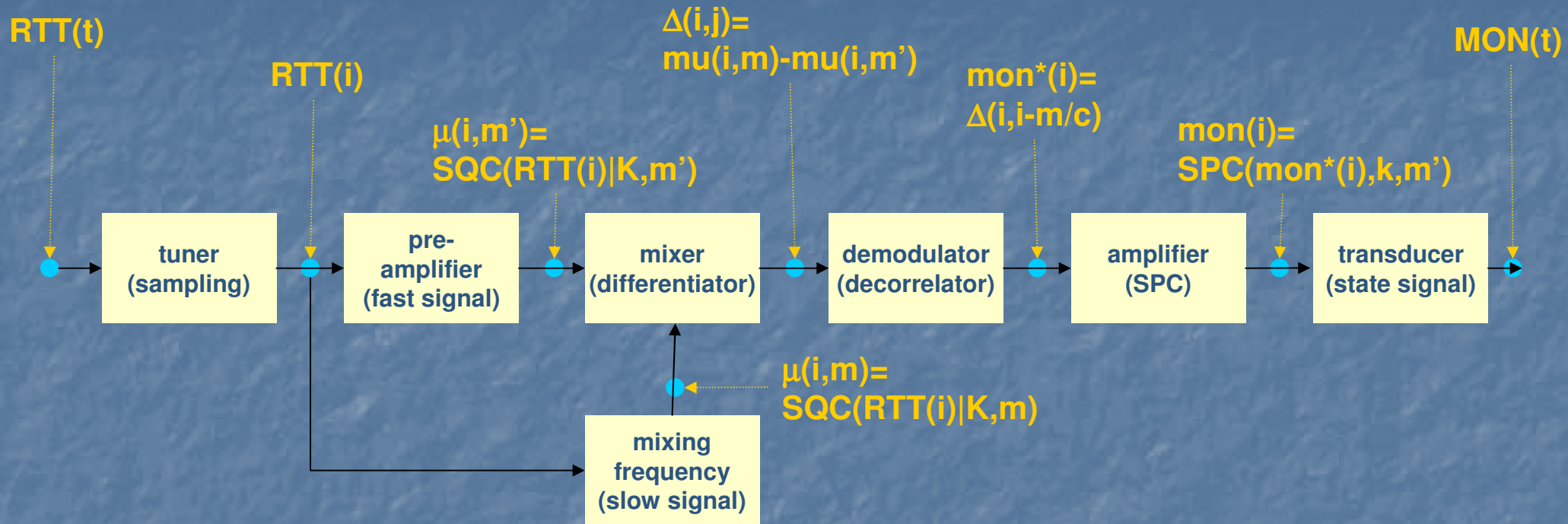
- **Fast vs. Slow Signal Window Size**
 - m, m'
- **Outlier Detection**
 - K, m
- **Decorrelator Point**
 - c, m
- **Error**
 - $k, m/m'$
- **State Fractality**
 - k

Network Performance Envelope Comparative Analysis

mean, stdev, sum \rightarrow numstates,



Network Superheterodyne (Armstrong)

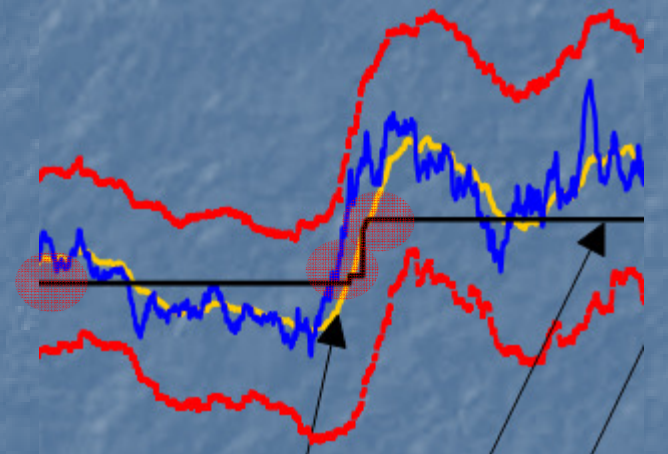
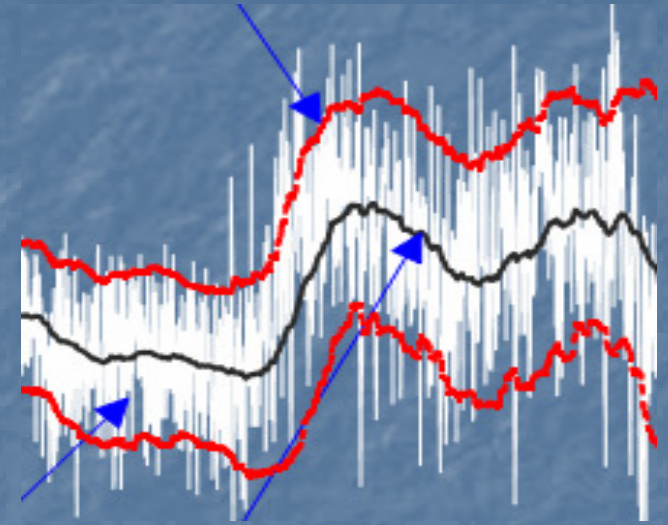


■ www.answers.com, www.fas.org/man/dod-101/navy/docs/es310/

- Of, relating to, or being a form of radio reception
- in which the frequency of an incoming radio signal
- is mixed with a locally generated signal
- and converted to an intermediate frequency
- in order to facilitate amplification and the rejection of unwanted signals
- superheterodyne receivers have better performance because the components can be optimized to work a single intermediate frequency

Network Stability Monitor Requirements

- **robust process state tracking**
 - quantifiable confidence interval
 - forecast confidence
- **low tracking overhead**
 - moving window kernels have straightforward $O(1)$ complexity
- **easy to implement**
 - moving window process state and control rules are simple (see flowchart)
- **low signaling overhead**
 - process state communicated only when necessary, that is, when it changes (i.e., only when statistically significant)



Research Contributions

- **online generation of a robust floating envelope**
 - showed the particular relevance of online-SPC - despite its simplicity
 - used to track the process performance associated with an indicator
 - shown for RTT – indicator for network congestion and flow management
- **formulated online SPC-based monitor of network state**
 - detection of stationary conditions (temporal stability, **process state**)
 - detection of piece-wise linear (**process changes**)
 - fast, robust, reliable , and low overhead
 - some setup vs. parametrizable statistical performance
- **robust state tracking building block**
 - near-stationarity conditions used to distinguish between process state and process changes
 - timescale and robustness targeted for process-performance



Future Work

■ simulations of SPC+ARC()

- further sensitivity analysis of SPC+ARC kernels
 - lag (m, m'), confidence ($k\sigma$), smoothing (m), smoother/sampling (UWMA, EWMA, etc.), adaptation (ω) and adaptation strategy (linear, multiplicative, constant), quantization process (mon), outlier recognition ($k\sigma$), fitness/residual analysis (err^*), etc.
- further performance g and optimality
 - comparative performance of SPC+ARC()
 - statistical performance (alpha errors, beta errors, etc.)

■ implementation of applications of SPC+ARC()

- multimedia networking performance
- other distributed online process control (see next segment)

