Adaptive estimation and prediction of power and performance in high performance computing

Reza Zamani and Ahmad Afsahi

Department of Electrical and Computer Engineering

Queen's University

Kingston, Ontario

Canada

Presentation Outline

- Introduction and Motivation
- Models and Algorithms
- Experimental Framework
- Power Estimation
- Power and Performance Prediction
- Further Investigation
- Conclusion and Future Work

Introduction and Motivation

- Power constraints on HPC
  - Electricity bill
  - Environmental impacts
  - Cooling issues

- Jaguar, the current leading system on the Top500 list
  - Performance of 1.75 petaflops,
  - 224162 cores,
  - 6.9 megawatts of power

Introduction and Motivation

- Power Management (PM) objectives:
  - Power capping
  - Energy saving
  - Thermal management
- How do we measure the fine grain impact of PM decisions on power consumption? (Action/Reaction)
  - Feedback update delay
  - Feedback update granularity
  - Embedded power measurement features (Available ?)
  - Accurate power models (Available?)

Introduction and Motivation

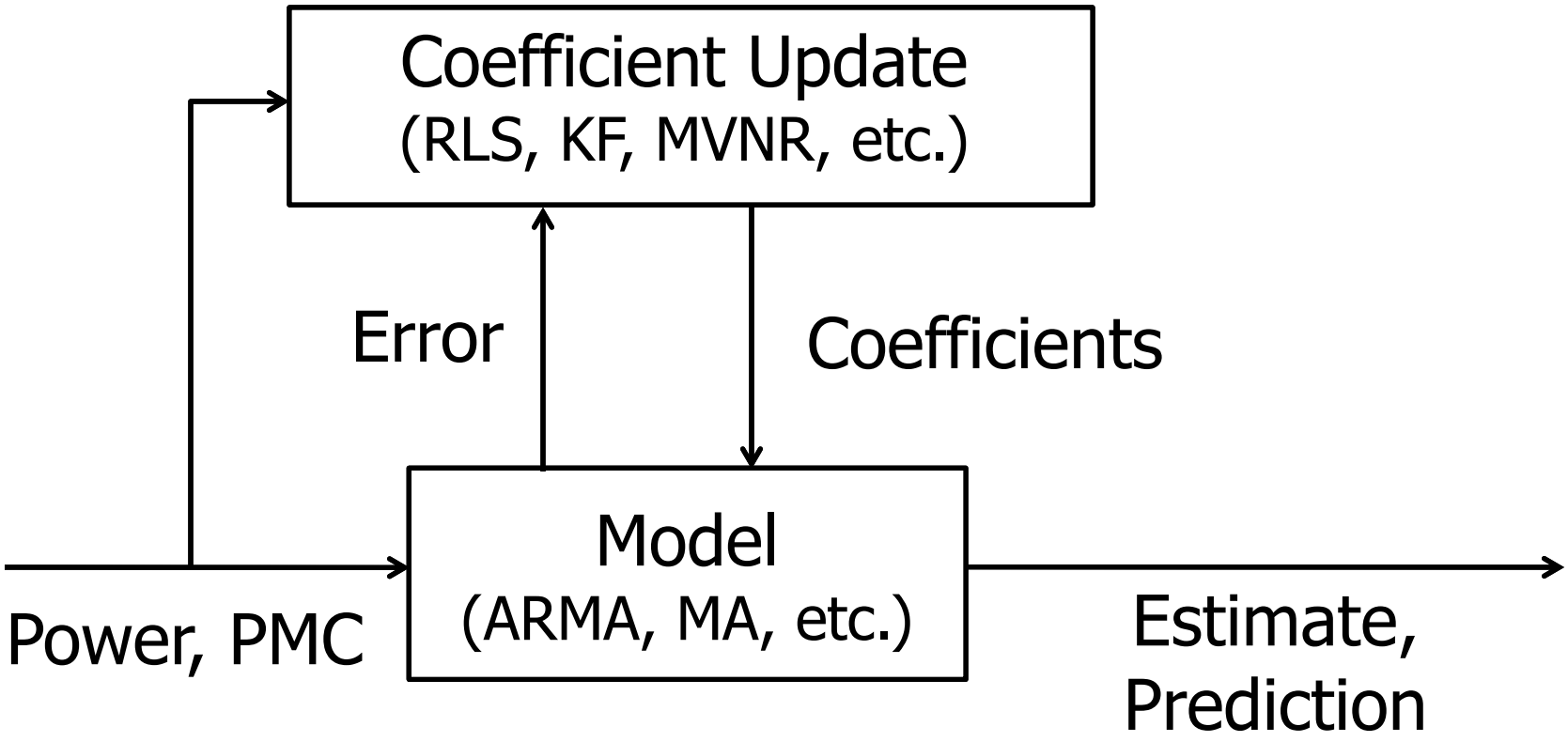
- Benefits
- Power Estimation: Enabling the PM to understand the consequences of its decisions
  - Almost immediately (Not after the total execution of the program)
- Power Prediction: Reduce the future penalty of present PM decisions
- Is this something new?

Introduction and Motivation

- Model Power/Performance Relationship:
  - Platform-independent
  - Application-independent
  - No internal power tapping
  - Too complex?
    - Interdisciplinary approach vs. Architectural approach
  - Stochastic approach
  - Utilizing “both” the current and the past PMC values
  - Integrating feedback power measurements for more accuracy
  - ARMA + update algorithms (e.g. RLS, MVNR, KF, BMVNR, etc)
  - Tests and results on a real system with HPC benchmarks

Models and Algorithms

- Black Box
- Models: ARMA, MA, etc.
- Coefficient Update Algorithms: RLS, KF, MVNR, etc.



Models and Algorithms

- Moving Average (MA)
- Autoregressive Moving Average (ARMA)
  - ARMA(p,q): AR(p) + MA(q)

$$X_t = c + \varepsilon_t + \sum_{i=1}^{p} \psi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$$

Models and Algorithms

- The discrete-time Kalman filter (KF)
- Cycles of:
  - Prediction
  - Correction
- Signal model:
  - Process equation $x_{k+1} = F_k x_k + w_k$
  - Measurement equation: $z_k = H_k' x_k + v_k$

Models and Algorithms

- System identification using KF

$$y_k + \sum_{j=1}^{n} a^{(j)} y_{k-j} = \sum_{j=1}^{m} a^{(n+j)} u_{k-j}$$

- Time varying coefficients
- The (*m* + *n*) coefficients of the ARMA are assumed to be the state of a process

$$y_k + \sum_{j=1}^{n} a_k^{(j)} y_{k-j} = \sum_{j=1}^{m} a_k^{(n+j)} u_{k-j} + v_k$$

$$x_k^{(1)} = a_k^{(1)}, x_k^{(2)} = a_k^{(2)}, \ldots, x_k^{(n+m)} = a_k^{(n+m)}$$

$$H_k' = [-y_{k-1} \ \ldots \ -y_{k-n} \ u_{k-1} \ldots \ u_{k-m}]$$

Models and Algorithms

- Recursive least-squares filter
  - Recursively produces the least squares of the error signal
  - Does not require statistical information about the input signal

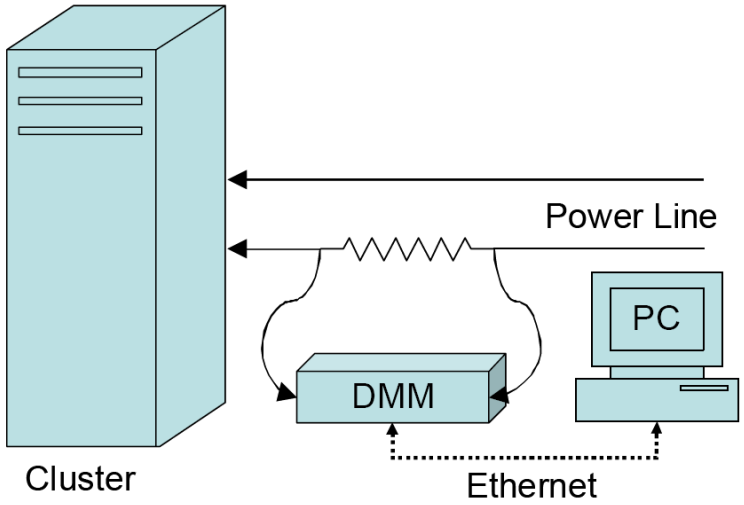
- Computationally less intensive than the KF
  - No matrix inversion.

- RLS filter can be reformulated as a KF

$$x_{k+1} = \lambda^{-1/2} x_k, \qquad z_k = H_k' x_k + v_k$$

Experimental Framework

- Dell PowerEdge R805 SMP server:
  - Two quad-core 2.0 GHz AMD Opteron
- Power measurement
  - Keithley 2701/7710 digital multi-meter (DMM),
  - a shunt resistor
- Ubuntu Linux (kernel 2.6.28.9)
  - patched with the perfctr library (2.6.39)



Experimental Framework

- Performance Monitoring Counters (PMCs) used:
  - Dispatch stalls
  - Memory controller page access event
  - Retired ×86 instructions
  - Cycles with no FPU ops retired
- Benchmark Applications:
  - Serial and OpenMP applications from the NAS parallel benchmarks (NPB)
  - NPB-3.3-SER benchmark suite
    - BT.A, BT.B, CG.B, EP.B, FT.B, LU.A, LU.B, SP.A, SP.B, UA.A, UA.B
  - NPB-3.3-OMP (with 8-threads)
    - BT.C, CG.C, LU.B, SP.C, UA.B, BT.B

Power Estimation

- How power measurement samples and performance monitoring counter samples are related?
- Not only looking at the current samples
  - But also considering previous power and PMC samples

- Moving Average – MA(0)

$$P[t] = \sum_{j=1}^{j_{max}} \alpha_{0,j} c_j[t] = A_0 C'[t]$$

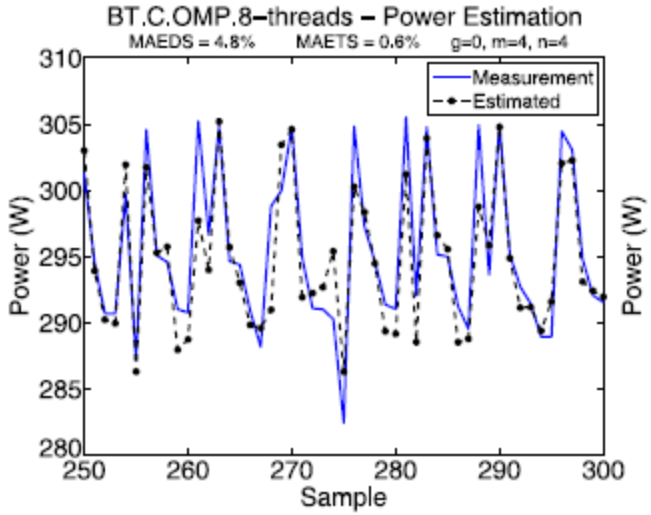
- ARMA (n,m)
  - ARMA(4,4)

$$P[t] + \sum_{i=1}^{n} \beta_i P[t-i] = \sum_{i=0}^{m} A_i C'[t-i]$$

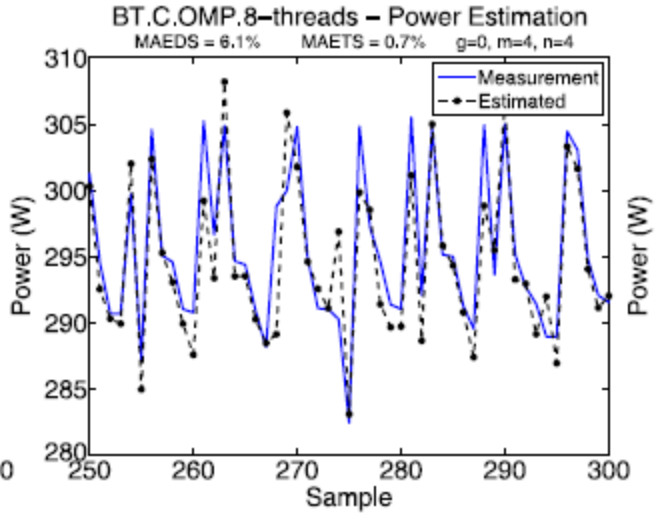
Power Estimation

- Dynamic Power vs. Total Power
  - P = P(dynamic)+ P(static)
- Reporting Errors: Mean Absolute Error of dynamic
  - Mean Absolute Error of Dynamic Signal (MAEDS)
  - Mean Absolute Error of Total Signal (MAETS)
- Power measurements 250W - 300 W
  - Dynamic signal range = 50 W, Static part = 250 W
- Mean absolute error of estimation of 10 W
  - MAETS of 3.3% (10/300)
  - MAEDS of 20% (10/50)

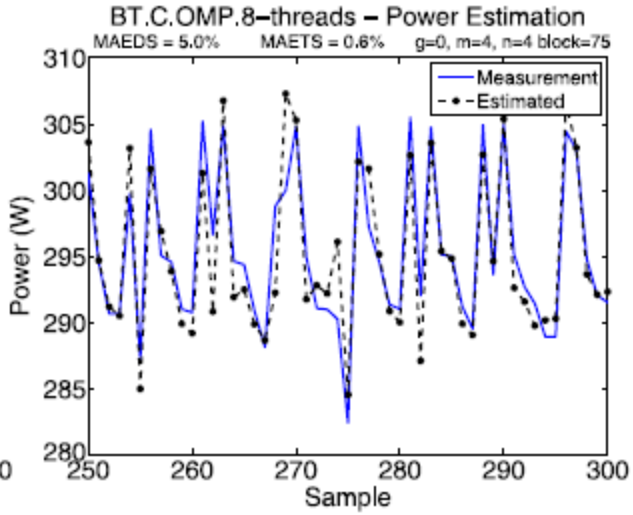
Power Estimation



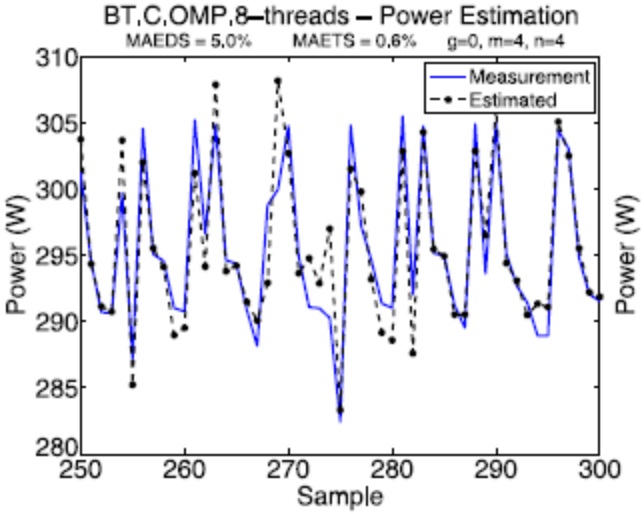
(a) RLS



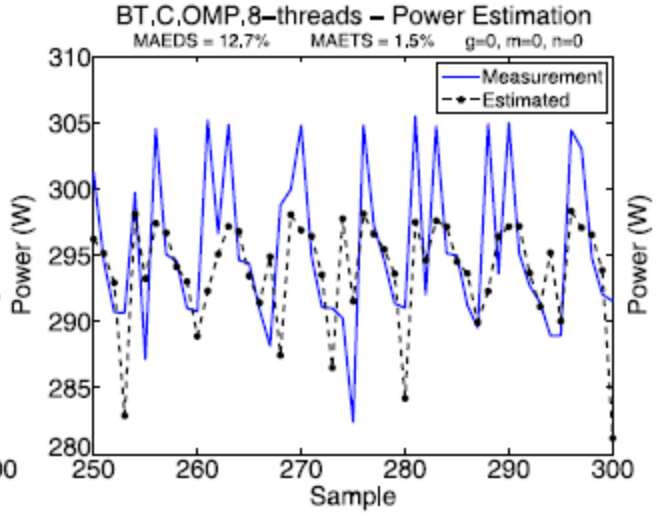
(b) MVNR



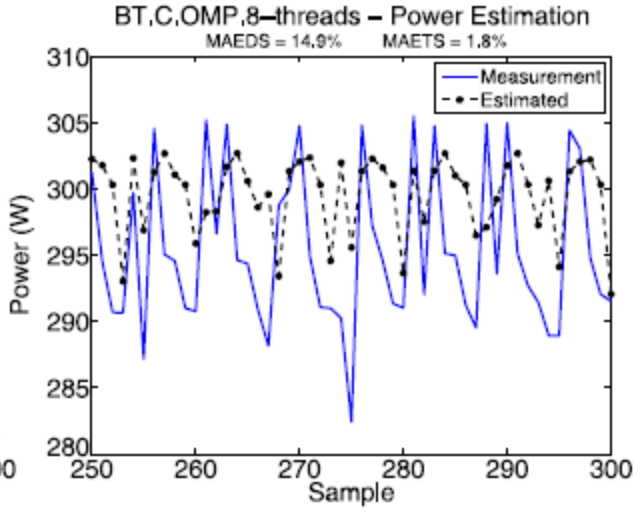
(c) BMVNR



(d) KF



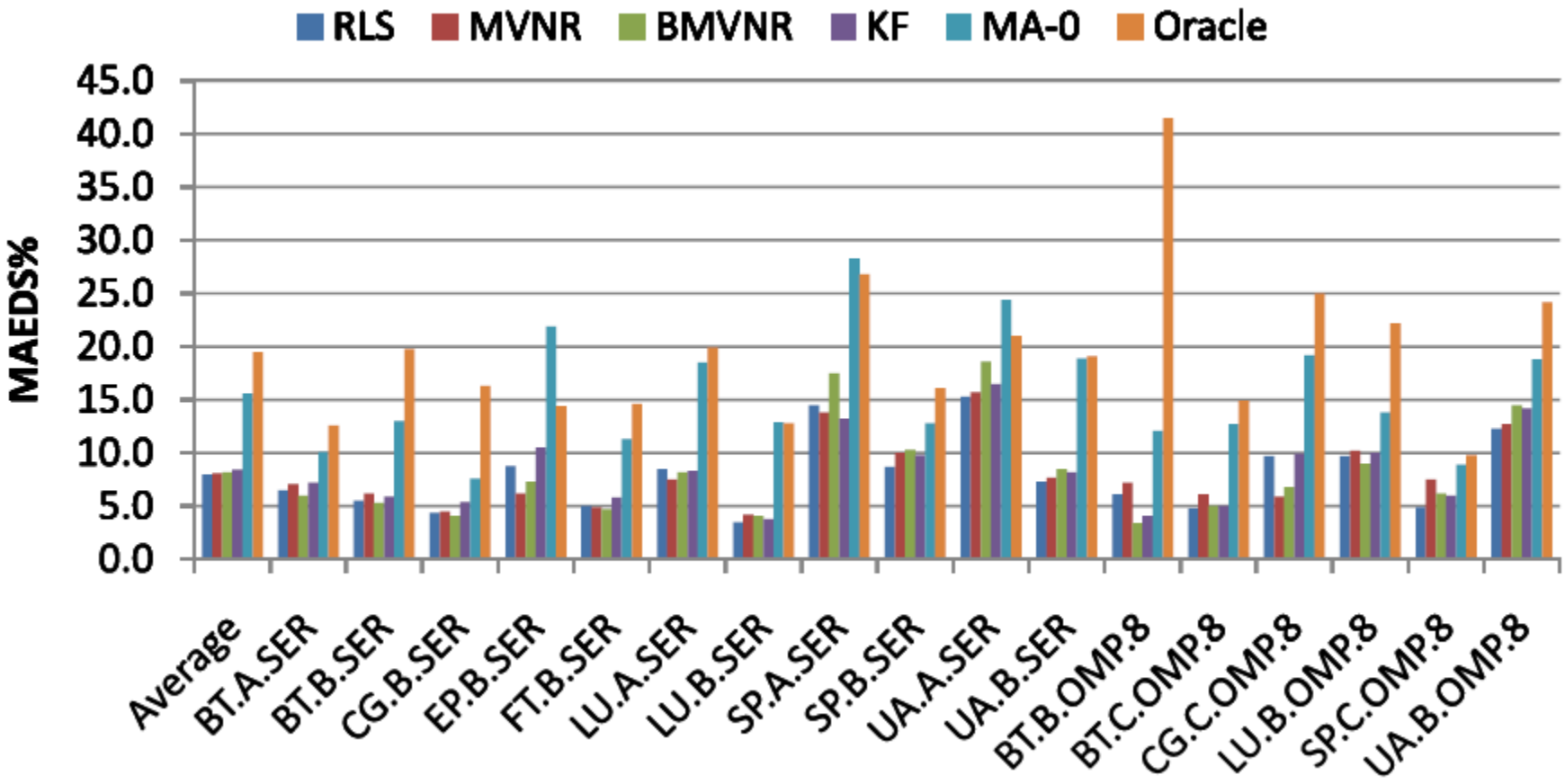
(e) MA-0



(f) Oracle

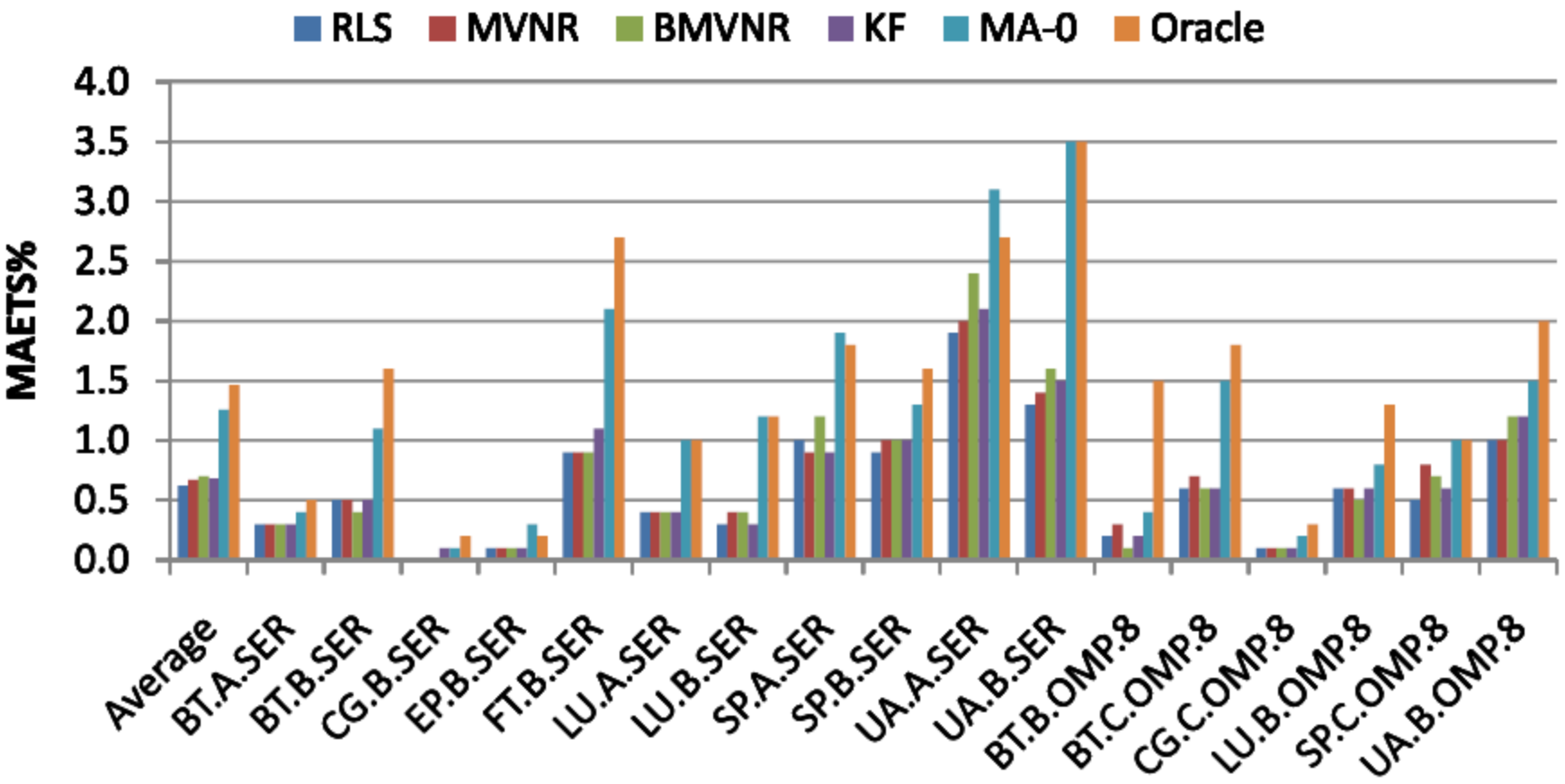
Power Estimation

- Mean Absolute Error of Dynamic Signal



Power Estimation

- Mean Absolute Error of Total Signal



Power Estimation

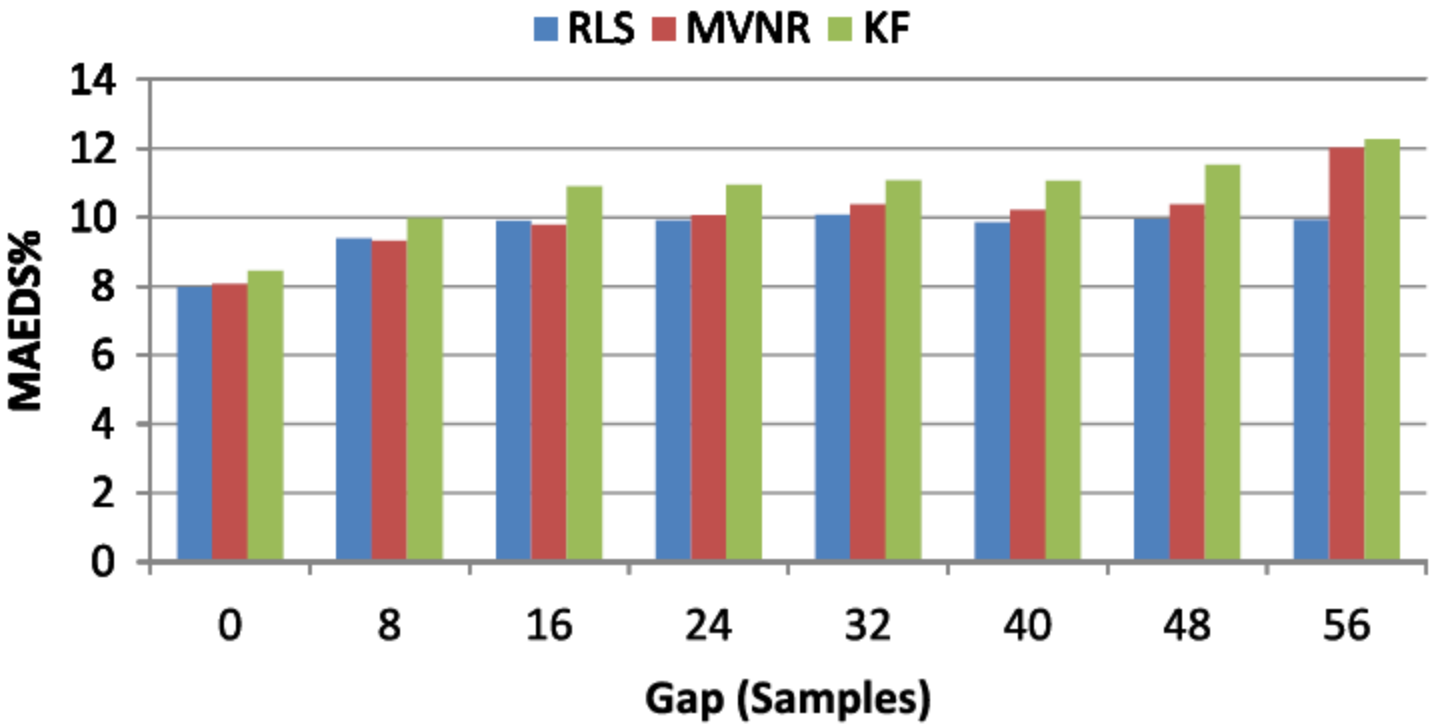
- Computation-time overhead
  - Measurement sampling rate
  - Estimation method computation time
    - Much smaller

- Actual implementation of RLS : approximately 710 usec/sample
  - MVNR : 321 x RLS
  - BMVNR: 37 x RLS
  - KF: 117 x RLS

Power Estimation

- Sensitivity to measurement update delay

$$\mathrm{P}[t] + \sum_{i=1}^{n} \beta_i \mathrm{P}[t-i-g] = \sum_{i=0}^{m} A_i \mathrm{C}'[t-i-g\Delta_{i0}]$$

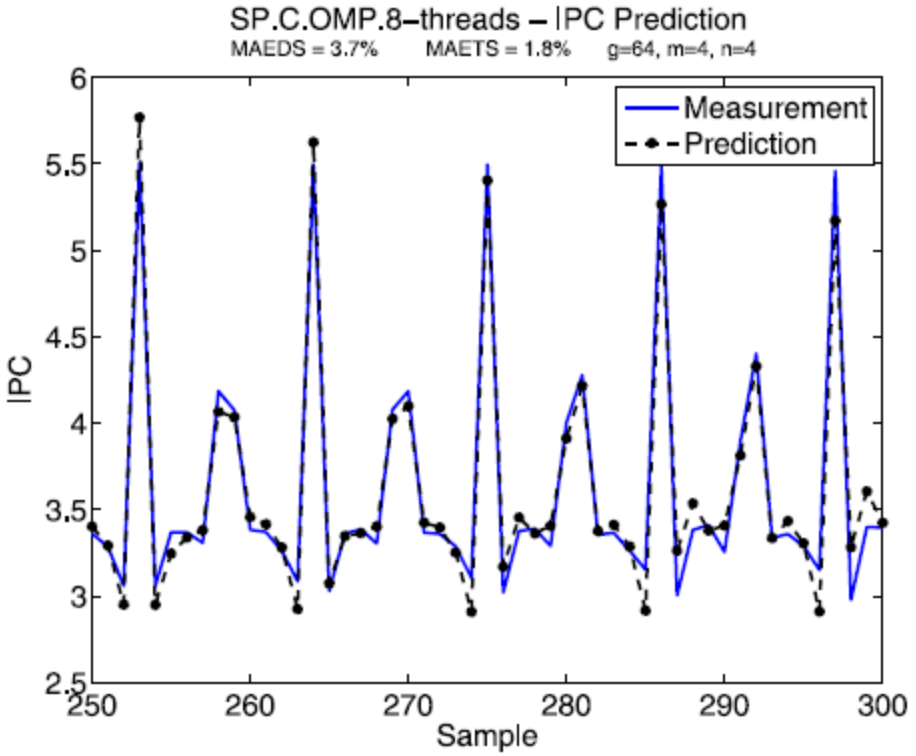


Power and Performance Prediction

- **Performance and power prediction using ARMA**

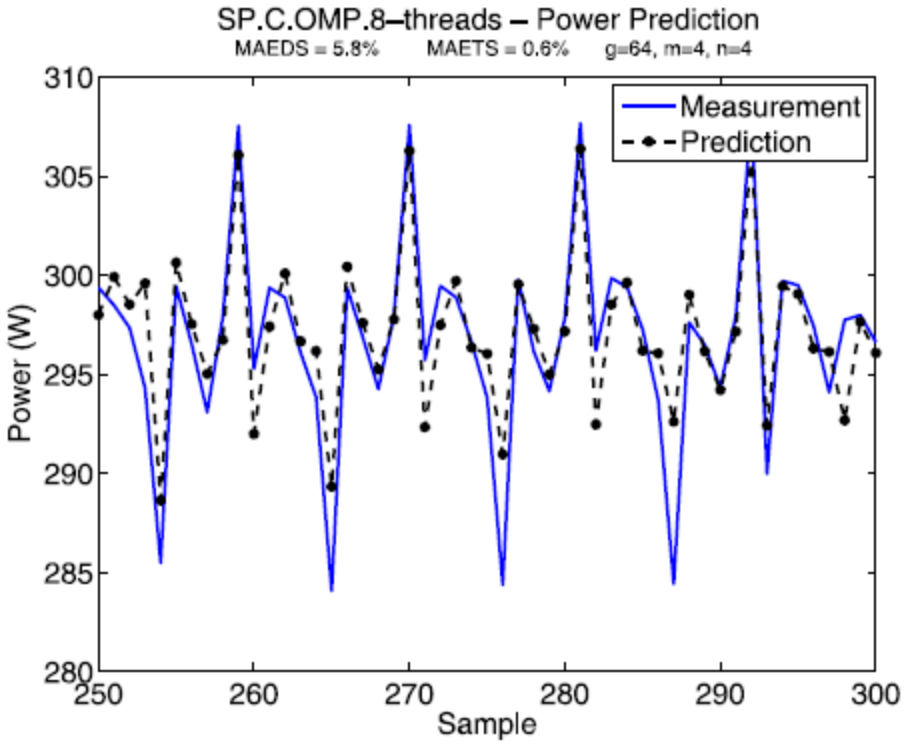
$$\mathrm{M}[t] + \sum_{i=1}^{n} \beta_i \mathrm{M}[t-i-g] = \sum_{i=1}^{m} A_i \mathrm{C}'[t-i-g]$$

- The IPC prediction MAEDS over all applications (one time step ahead)
  - RLS: 5.9%
  - KF: 6.5%
  - MVNR: 5.6%



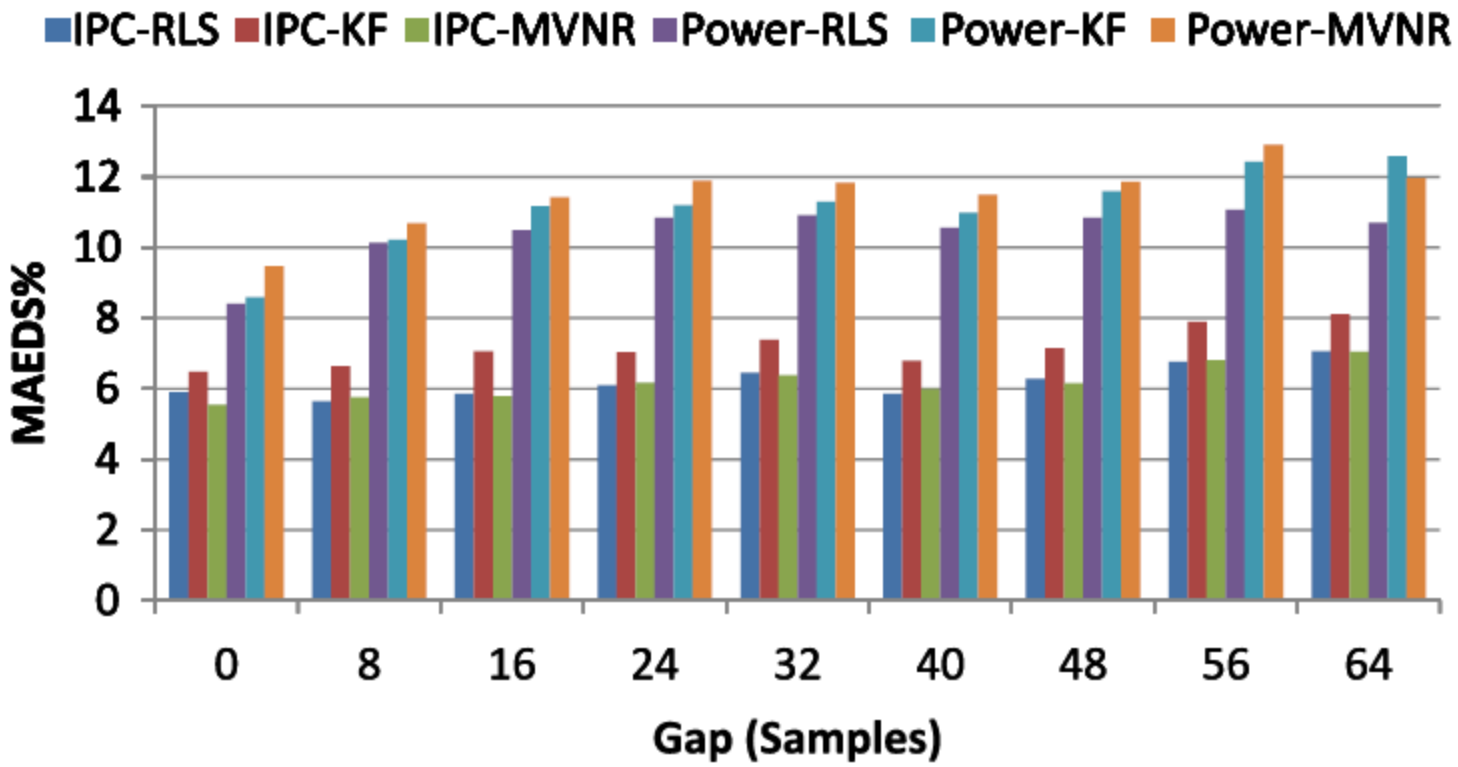
Power and Performance Prediction

- The power prediction MAEDS over all applications (one time step ahead)
  - RLS: 8.4%
  - KF: 8.6%
  - MVNR: 9.5%



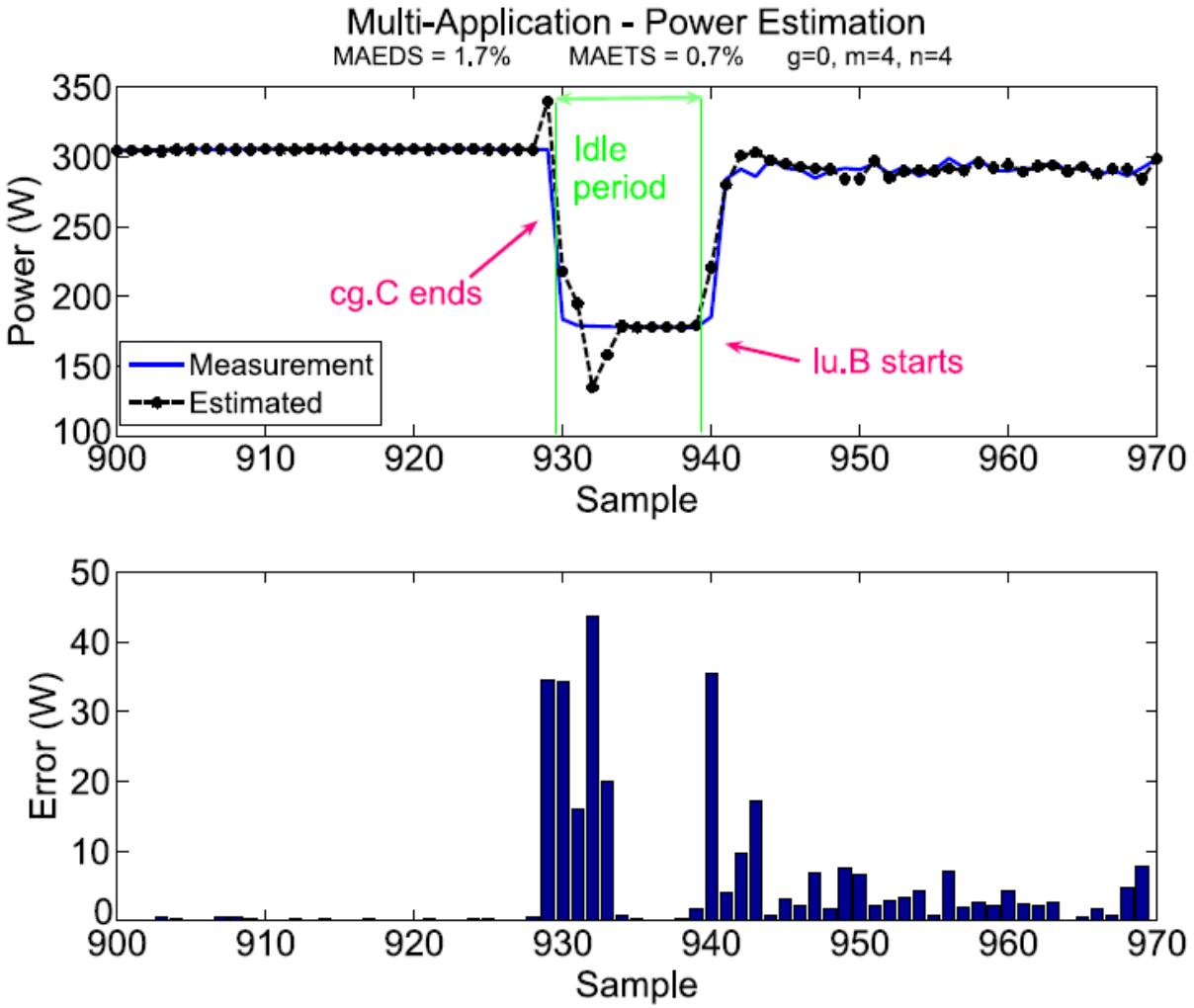
Power and Performance Prediction

- Sensitivity to measurement update delay



Further Investigation

- Runtime power estimation of multiple applications
- Extreme cases:
  - Idle period
  - Start/end of a benchmark



Conclusion and Future Work

- Using ARMA models for modeling power and PMC relationships
  - RLS, MVNR, BMVNR, KF, MA-0, and Oracle
- A good model candidate
- Not significantly sensitive to feedback delay
- Extendable idea for predicting PMC/Power

- Winning model/algorithm: ARMA-RLS
  - Real-time integration
  - Including DVFS effects

Questions?