Towards a Generic Power Estimator

Leandro Cupertino Georges Da Costa Jean-Marc Pierson

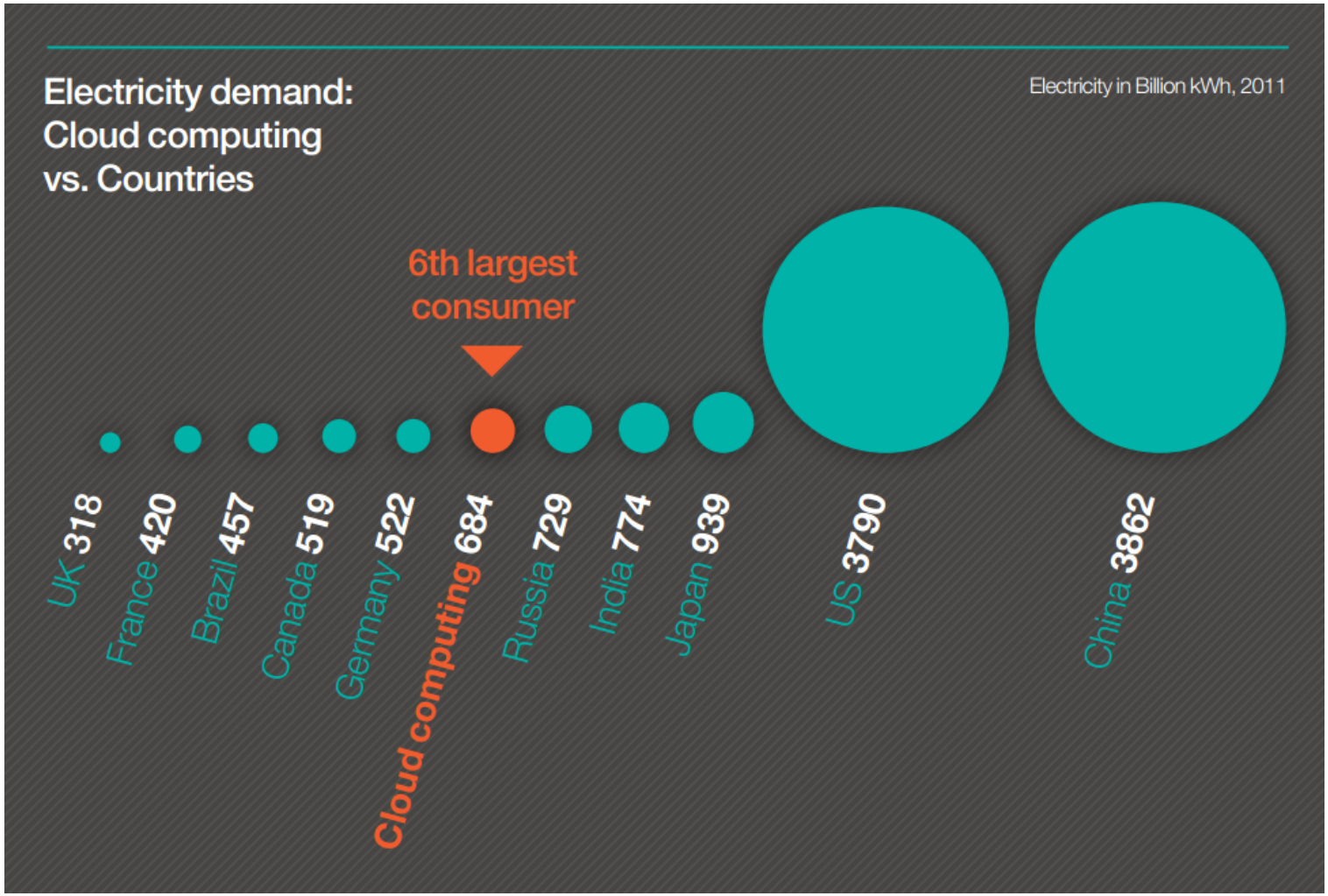
Toulouse Institute of Computer Science Research
University of Toulouse III, France

EnA-HPC – September 1, 2014

Outline

Outline

Introduction

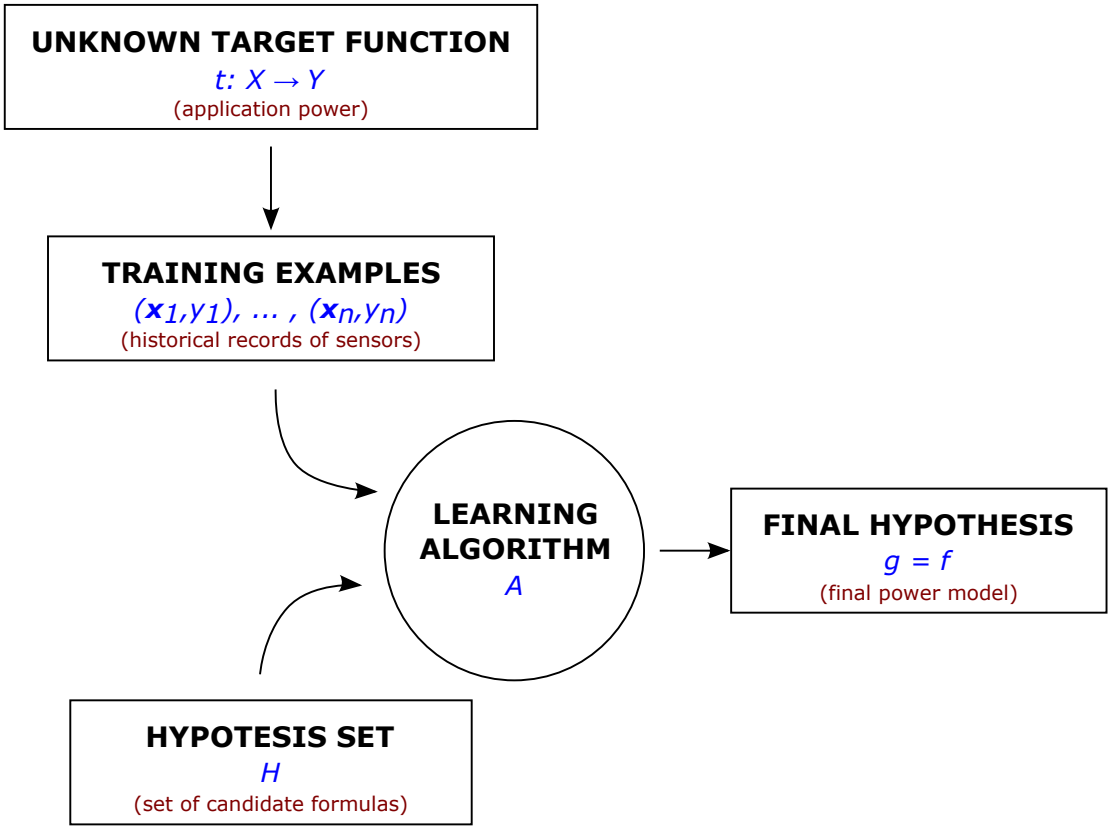


"Clicking Clean: How Companies are Creating the Green Internet," Tech. Report, Greenpeace, April 2014.

Motivation

- Power consumption varies according to servers' usage
  - Energy efficient scheduling
  - Power estimation of applications
- Current models limitations
  - Application specific or lack accuracy
  - Hardware dependent
- External power meters
  - Easy to deploy
  - Energy providers charge power in AC mode

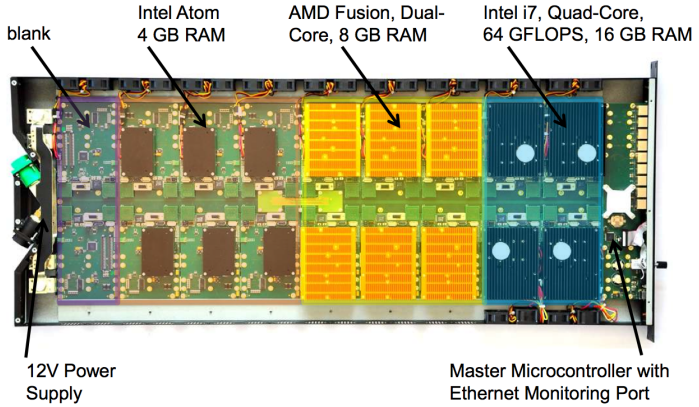
Estimating power with Artificial Neural Networks



1. Definition of the Training Set
2. Performance indicators
3. Accurate power measurements

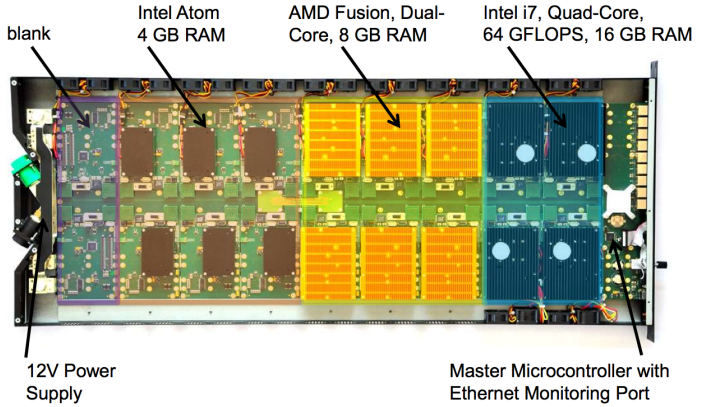
Environment setup

- RECS compute box
  - Processor: Intel Core i7-3615QE
  - Memory: 16GB of RAM
  - NIC: Intel 82579LM Gigabit Ethernet
  - Disk: diskless environment



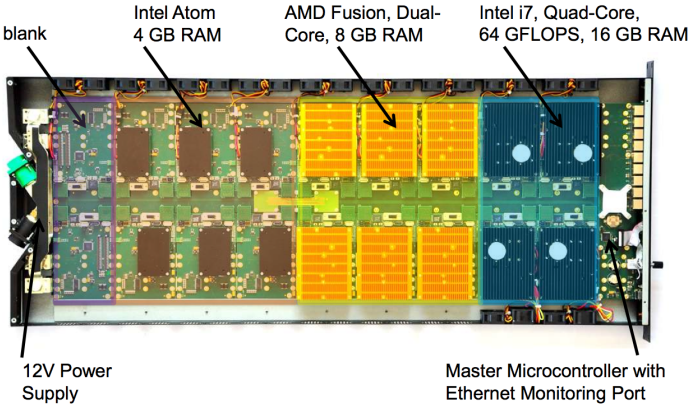
Environment setup

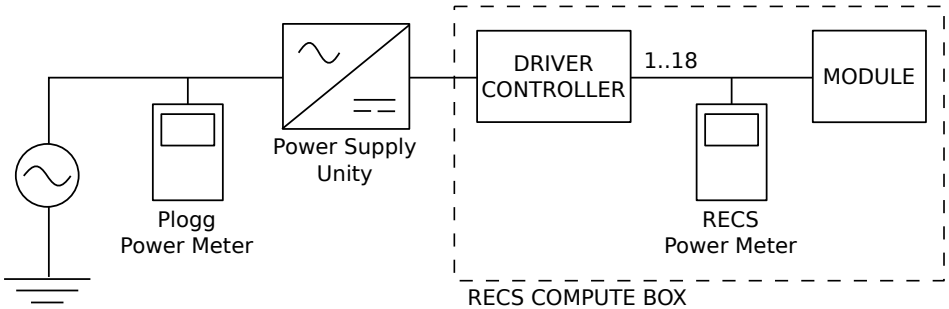
- RECS compute box
  - Processor: Intel Core i7-3615QE
  - Memory: 16GB of RAM
  - NIC: Intel 82579LM Gigabit Ethernet
  - Disk: diskless environment
- OS: Scientific Linux release 6.4 (kernel v2.6.32)



Environment setup

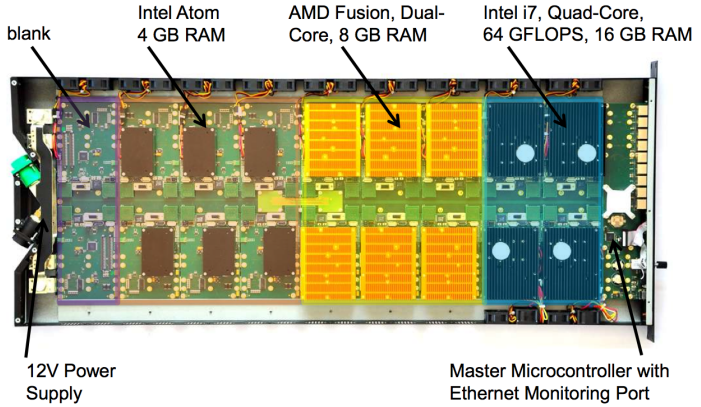
- RECS compute box
  - Processor: Intel Core i7-3615QE
  - Memory: 16GB of RAM
  - NIC: Intel 82579LM Gigabit Ethernet
  - Disk: diskless environment
- OS: Scientific Linux release 6.4 (kernel v2.6.32)
- Power meter: Plogg



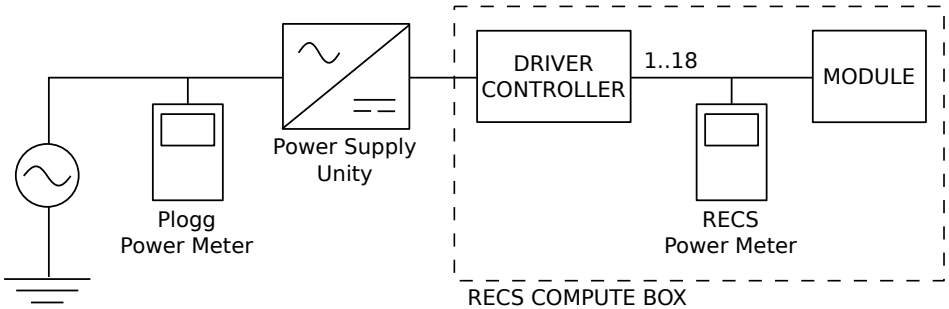


Environment setup

- RECS compute box
  - Processor: Intel Core i7-3615QE
  - Memory: 16GB of RAM
  - NIC: Intel 82579LM Gigabit Ethernet
  - Disk: diskless environment



- OS: Scientific Linux release 6.4 (kernel v2.6.32)
- Power meter: Plogg

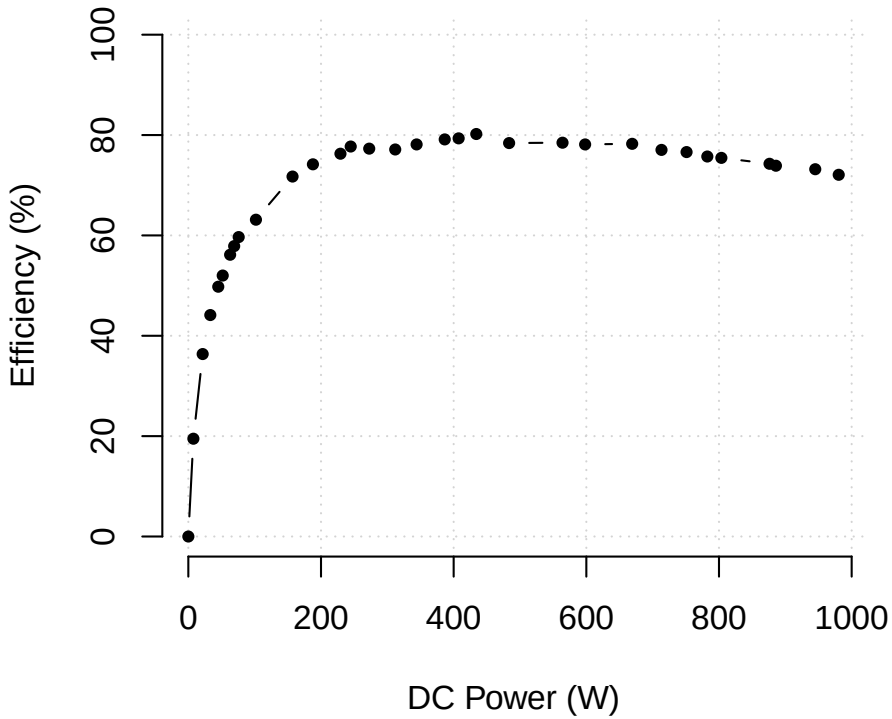


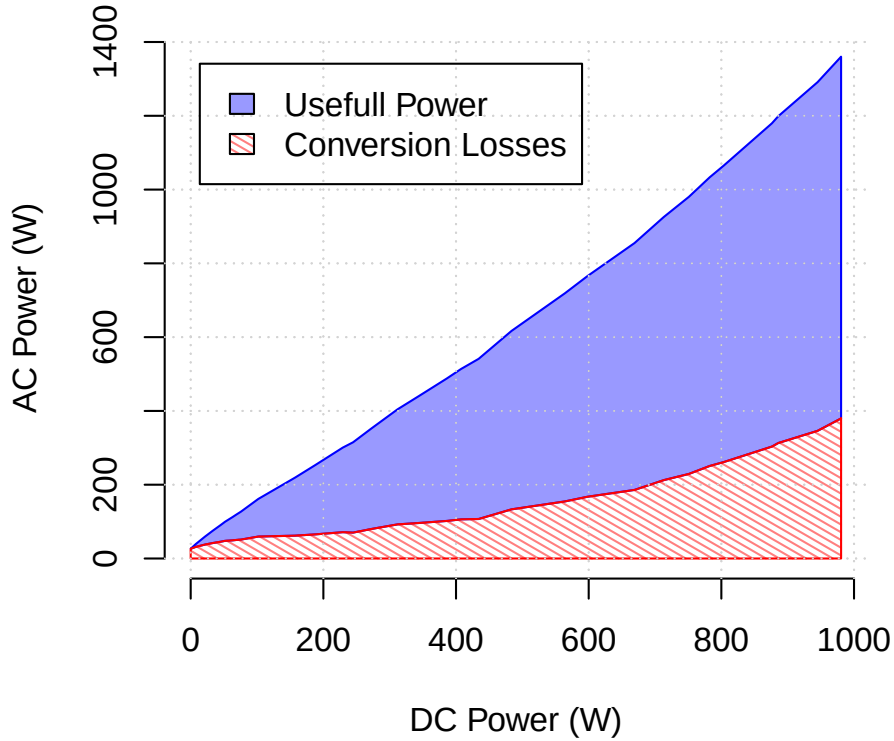
- Modules management and power monitoring is done by an external server to not impact the measurements.

Outline

Modeling PSU's conversion losses

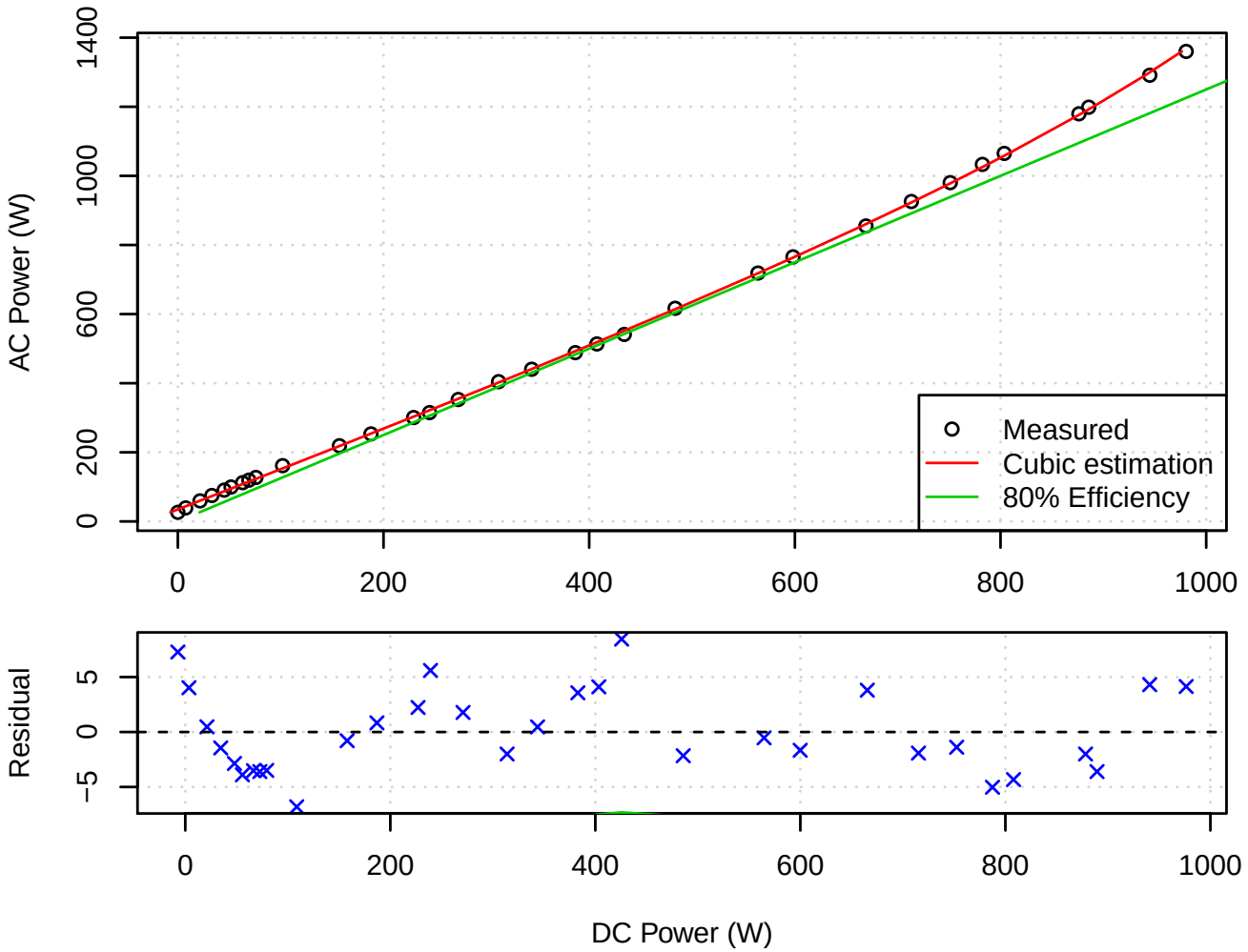
- Conversion losses are sometimes neglected
- PSU Input/Output power data provided by PSU's vendors





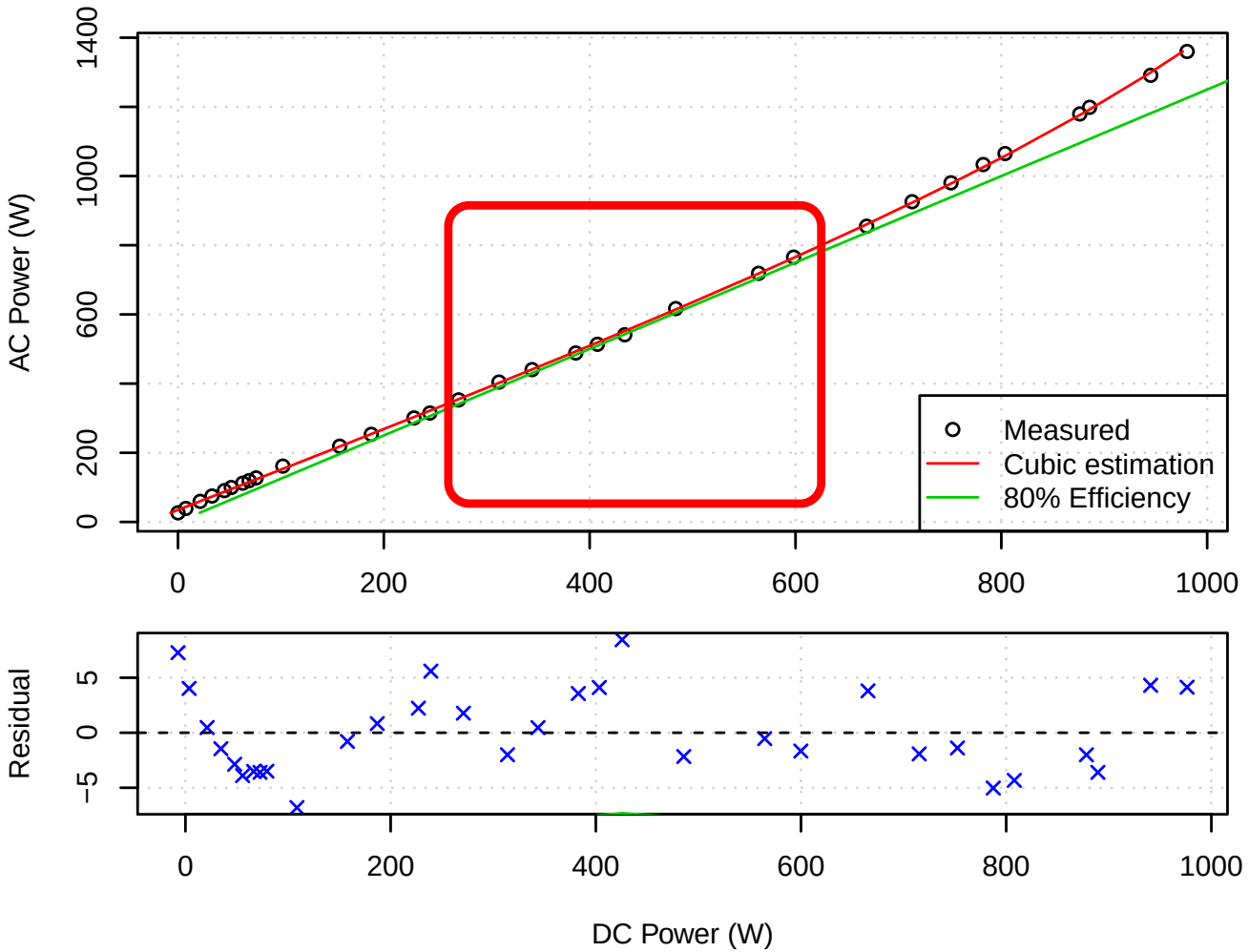
Modeling PSU's conversion losses

- Cubic model: $DC \sim w_0 * AC + w_1 * AC^3$



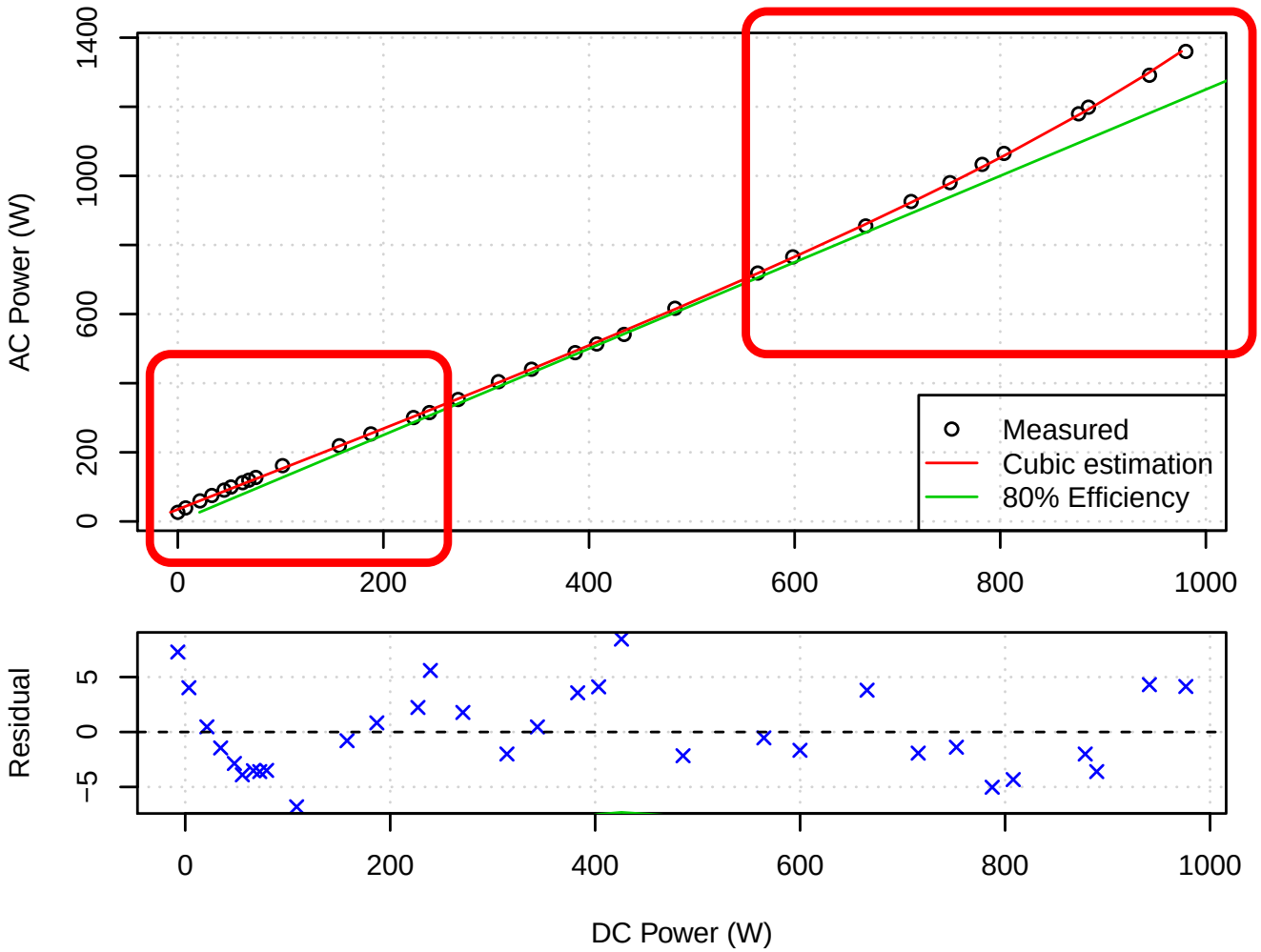
Modeling PSU's conversion losses

- Cubic model: $DC \sim w_0 * AC + w_1 * AC^3$



Modeling PSU's conversion losses

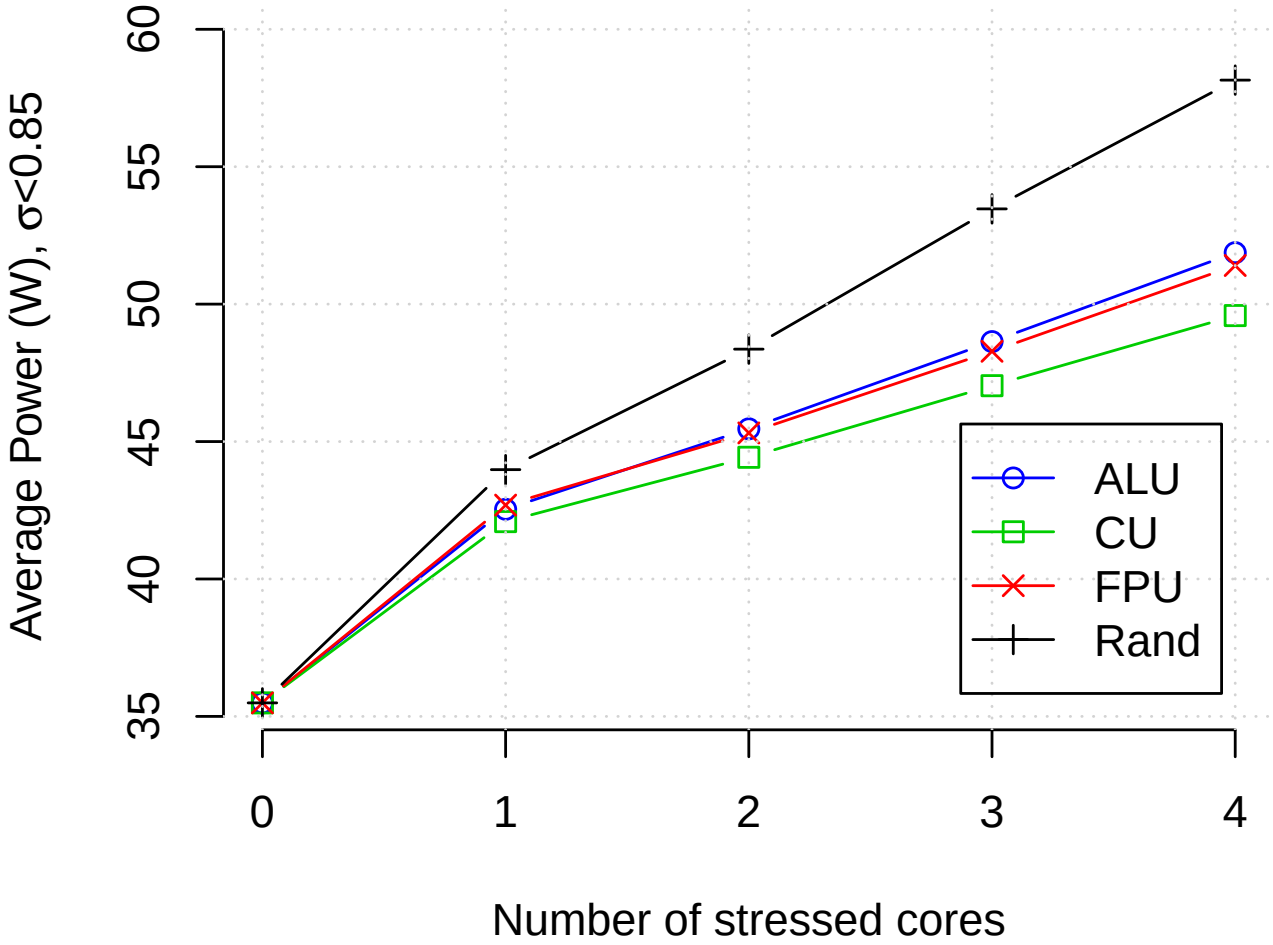
- Cubic model: $DC \sim w_0 * AC + w_1 * AC^3$



Outline

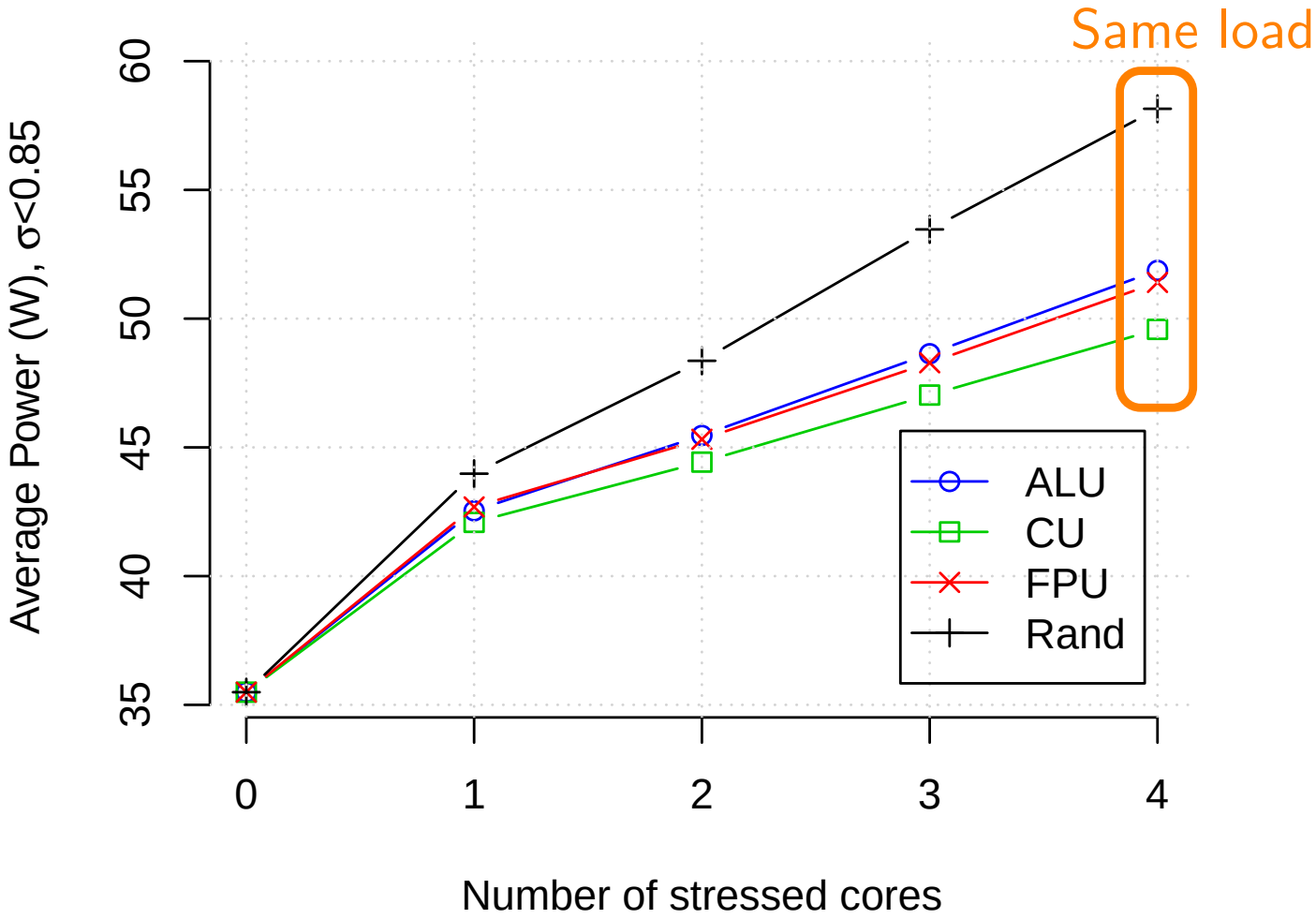
Limitation of CPU proportional models

- Computation intensive benchmarks:
  CU (NOP), ALU (ADD), FPU (FADD), Rand (C rand())



Limitation of CPU proportional models

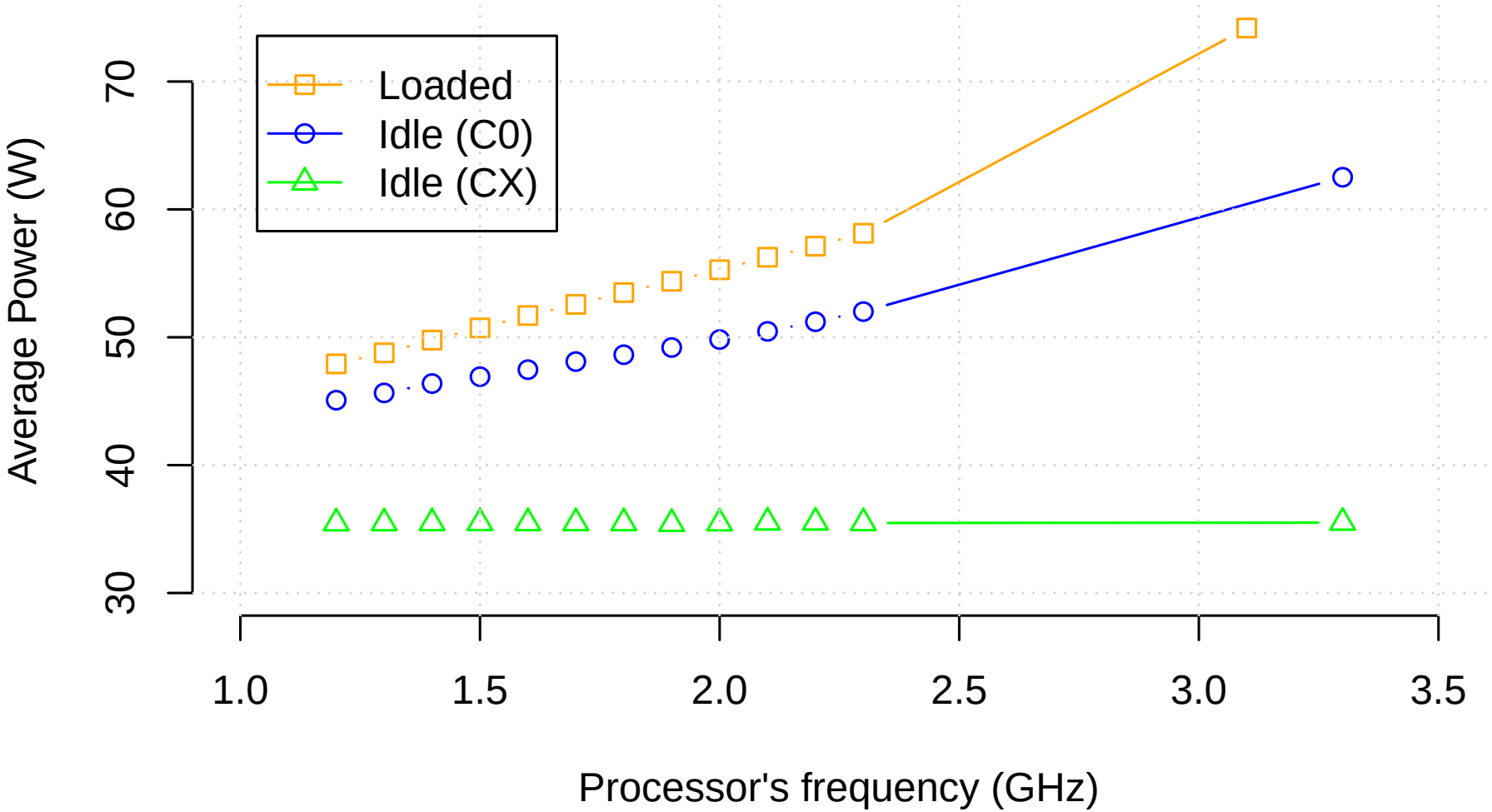
- Computation intensive benchmarks:
  CU (NOP), ALU (ADD), FPU (FADD), Rand (C rand())



Outline

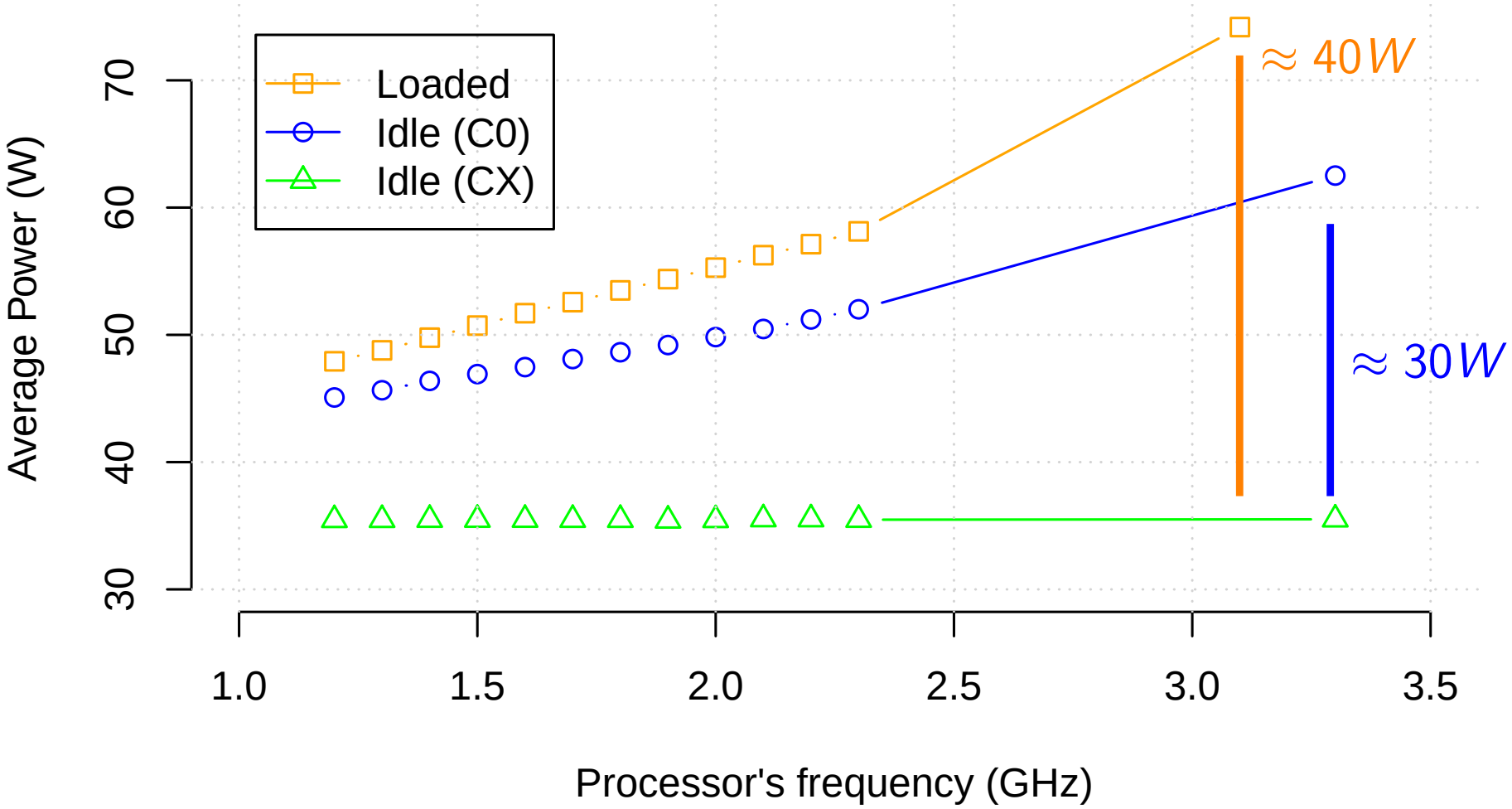
Dynamic Voltage and Frequency Scaling

- Benchmarks:
  Loaded (Rand), C0 (active Cstate), CX (deepest Cstate)

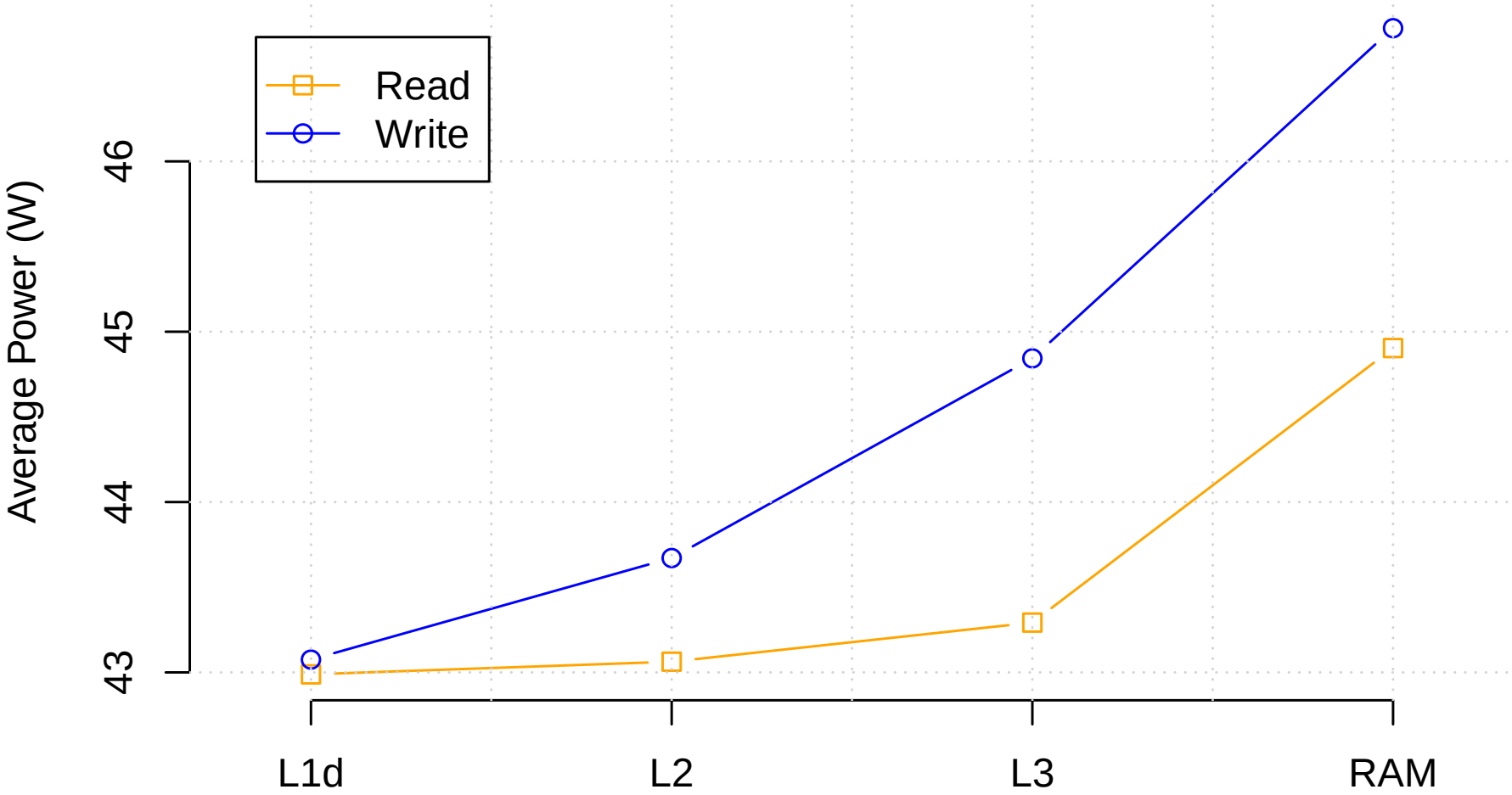


Dynamic Voltage and Frequency Scaling

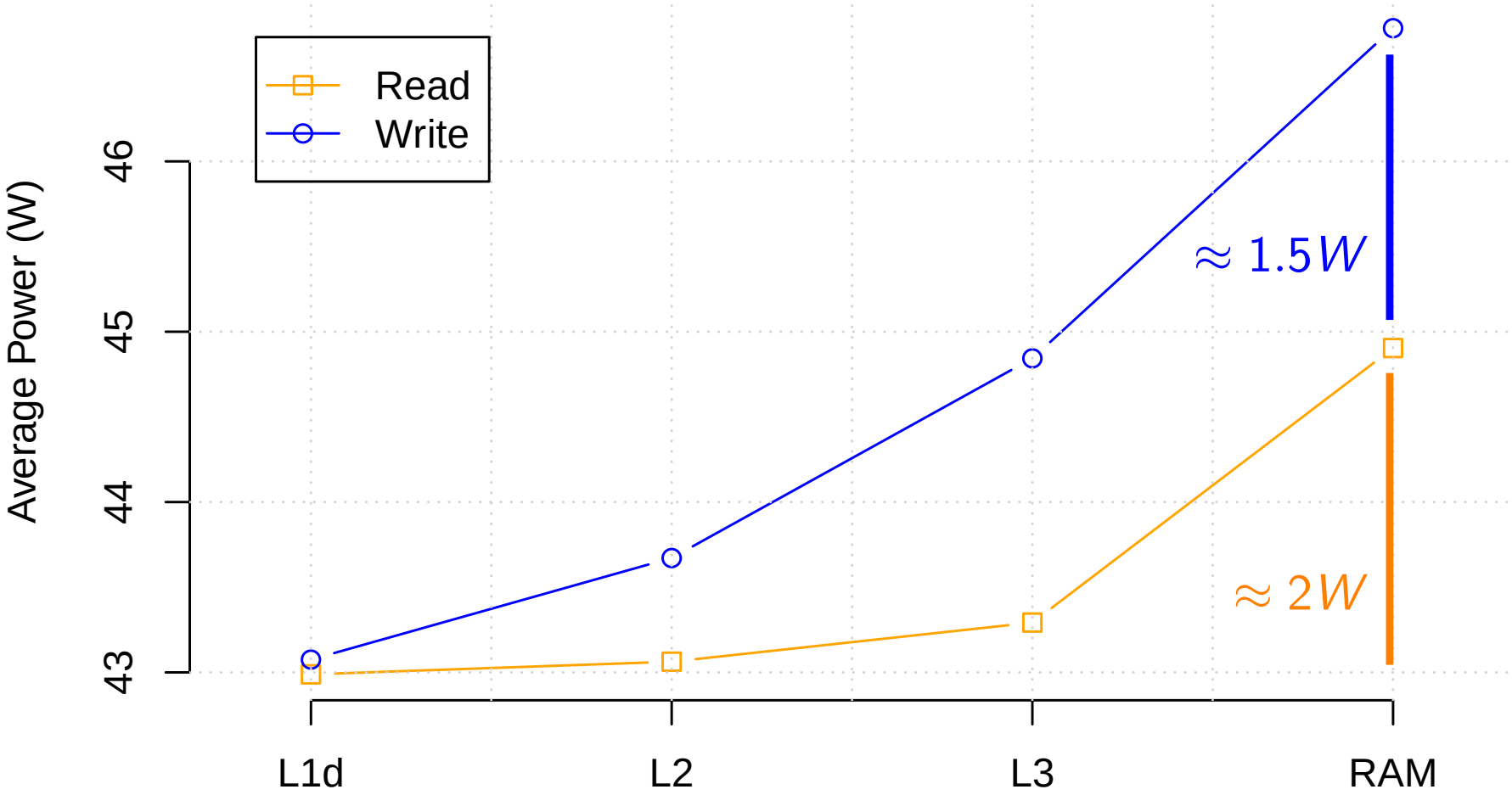
- Benchmarks:
  Loaded (Rand), C0 (active Cstate), CX (deepest Cstate)



Memory power profile

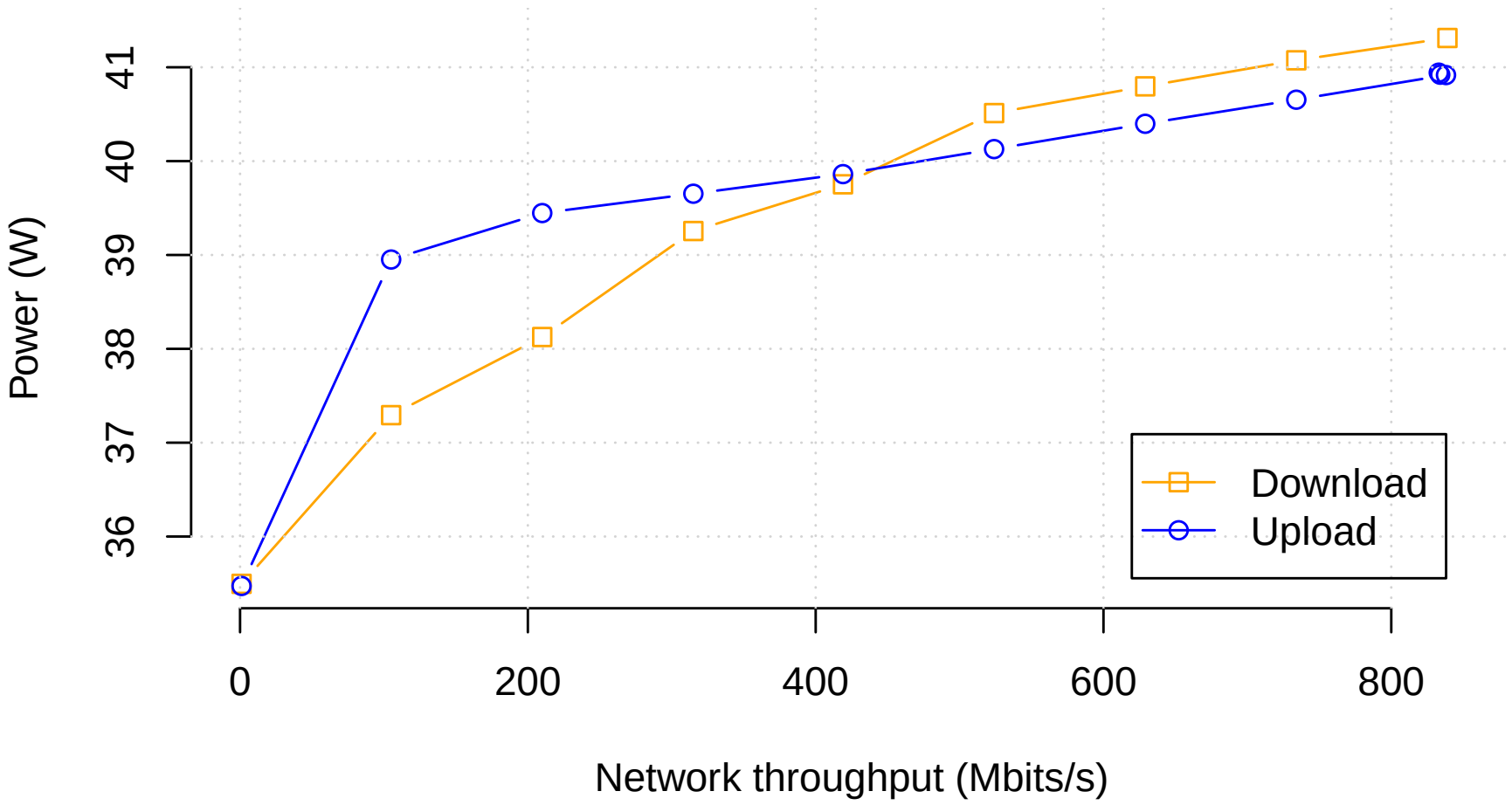


Memory power profile



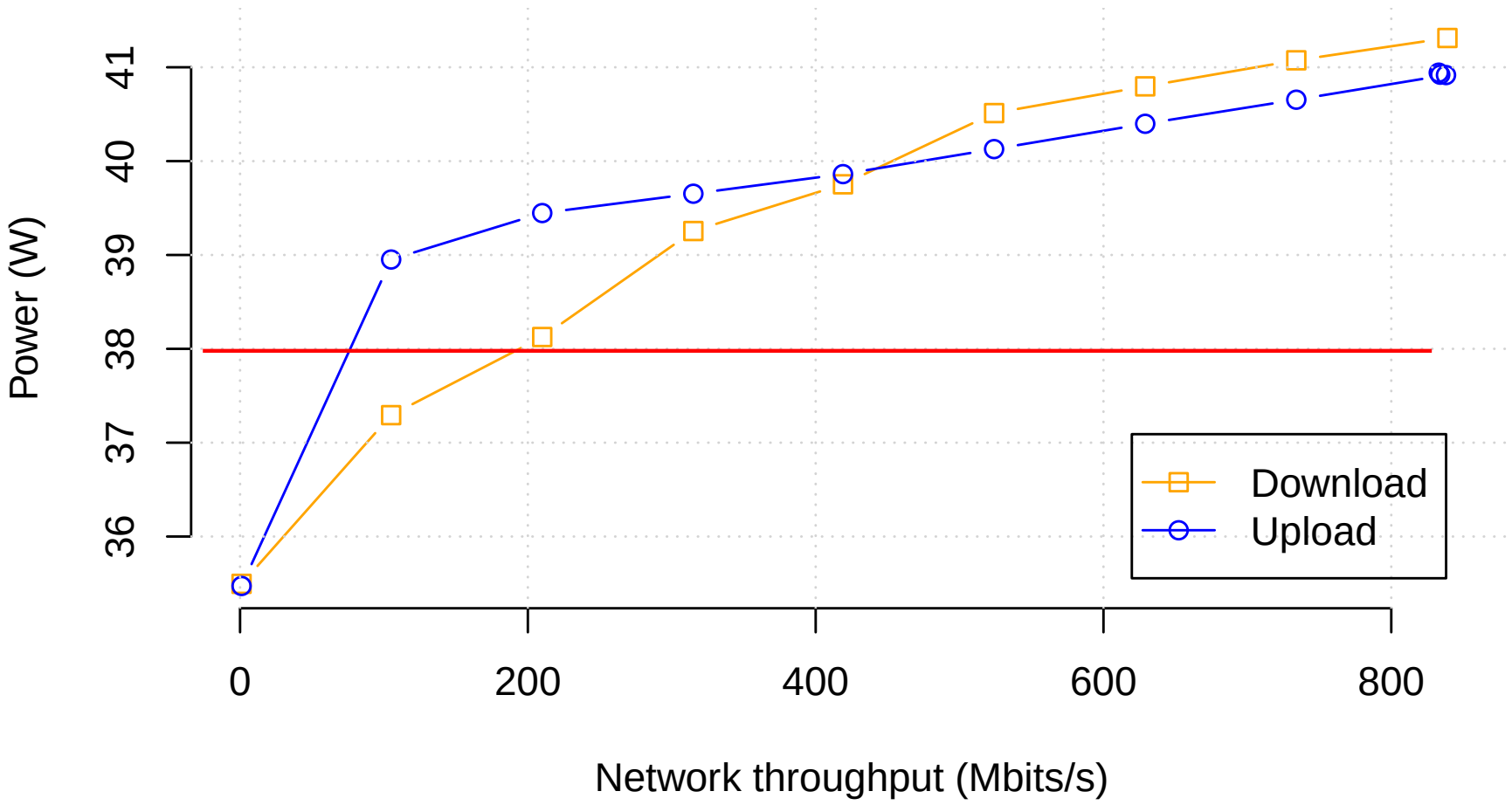
Network usage

- Benchmark: iperf3
- CPU usage $< 5\%$ (Power at $5\% < 38$W)



Network usage

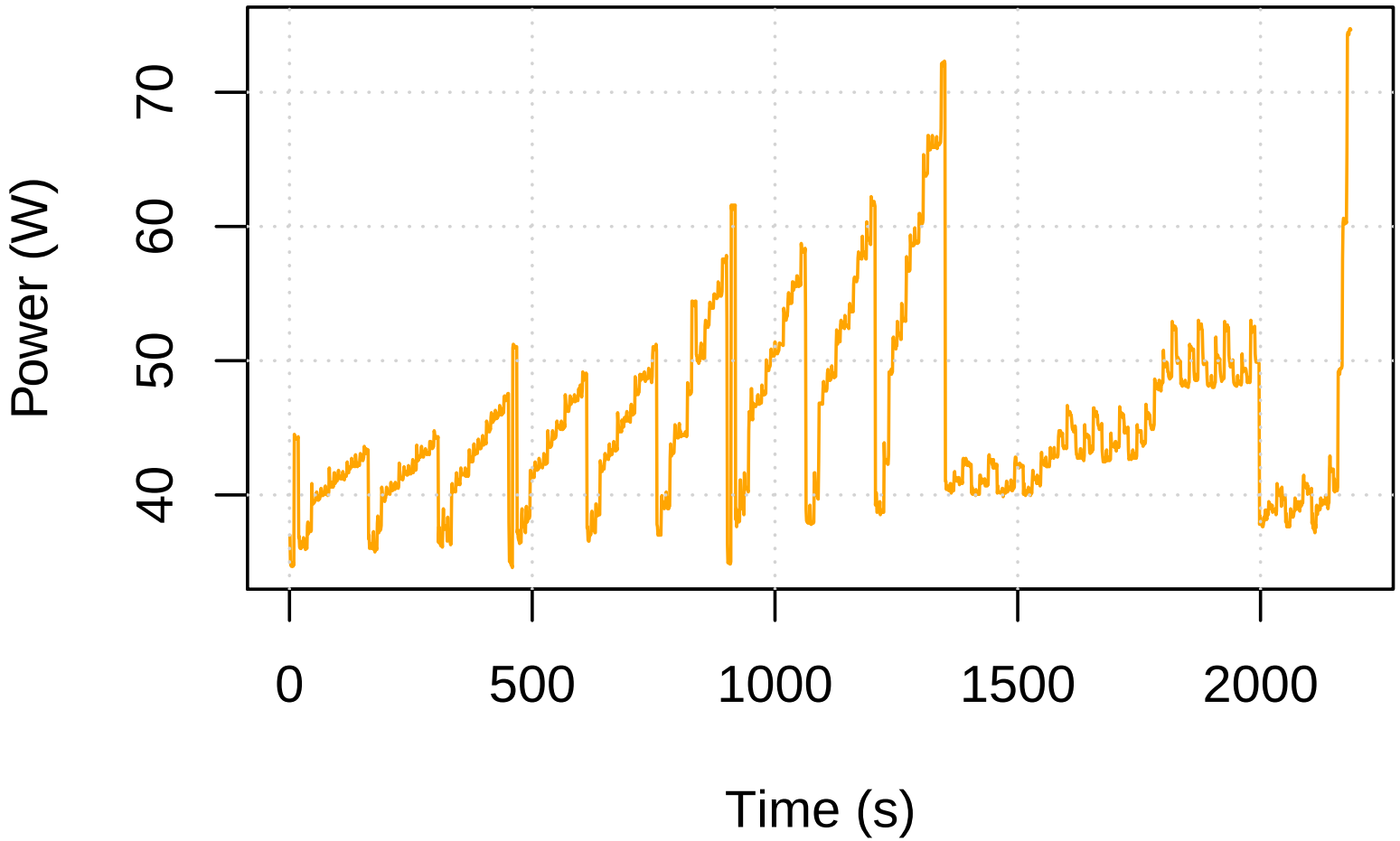
- Benchmark: iperf3
- CPU usage $< 5\%$ (Power at $5\% < 38$W)



Outline

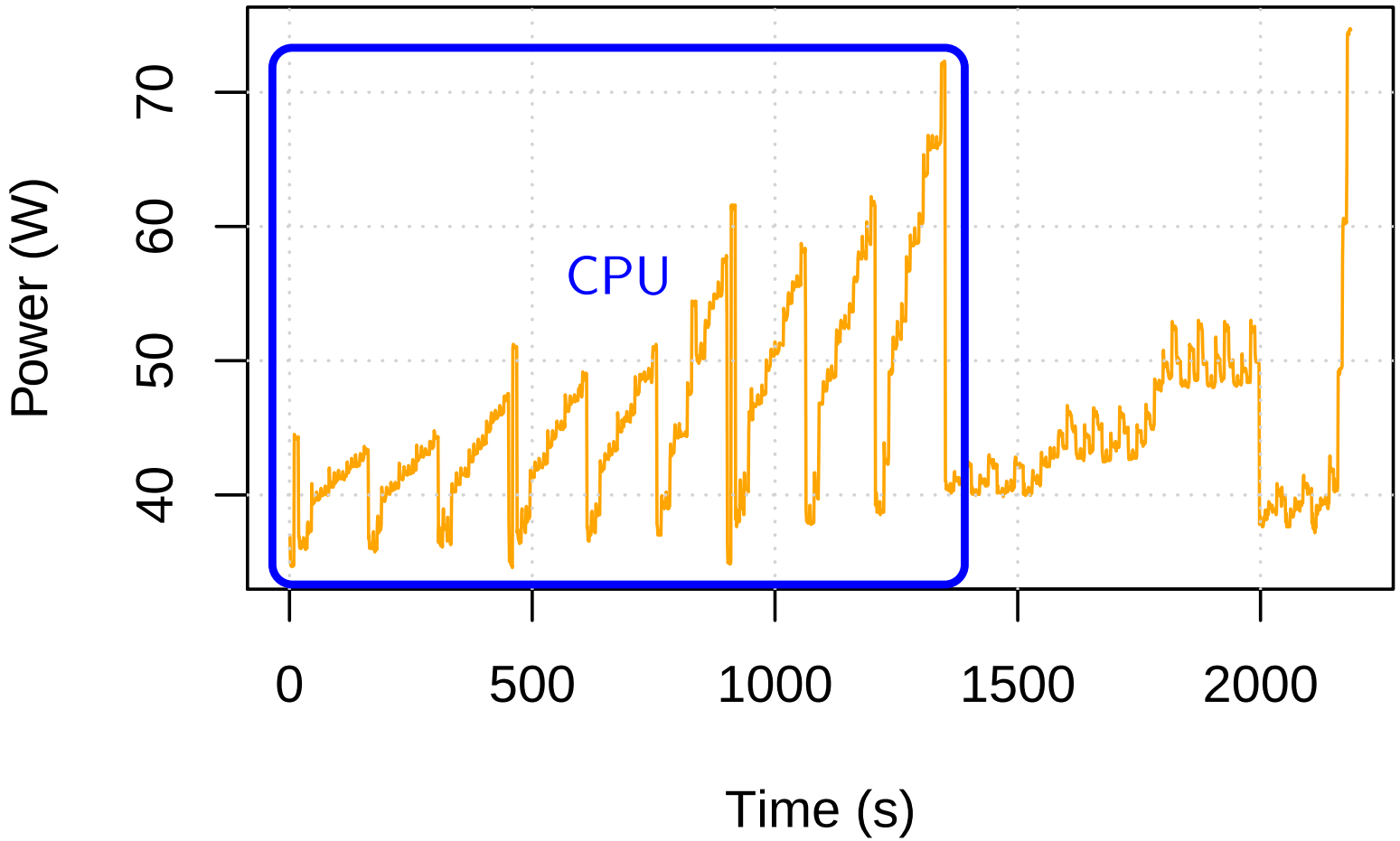
Training set's power profile

- Training set: CPU, memory and network stressed
- 3 frequencies: min (1.2GHz), max (2.3GHz) and boost (3.1GHz)
- Dynamic power: up to 40W of variation



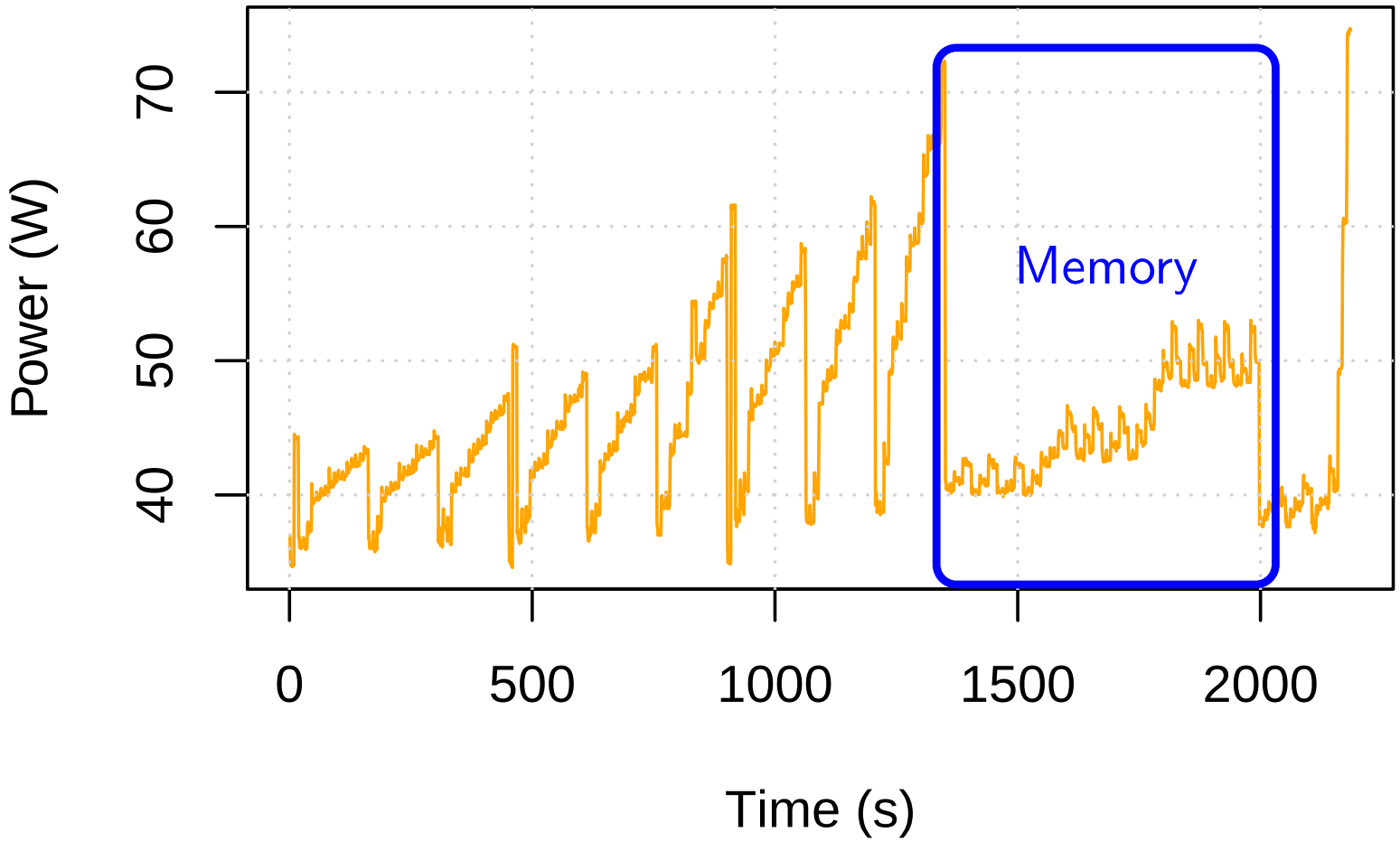
Training set's power profile

- Training set: CPU, memory and network stressed
- 3 frequencies: min (1.2GHz), max (2.3GHz) and boost (3.1GHz)
- Dynamic power: up to 40W of variation



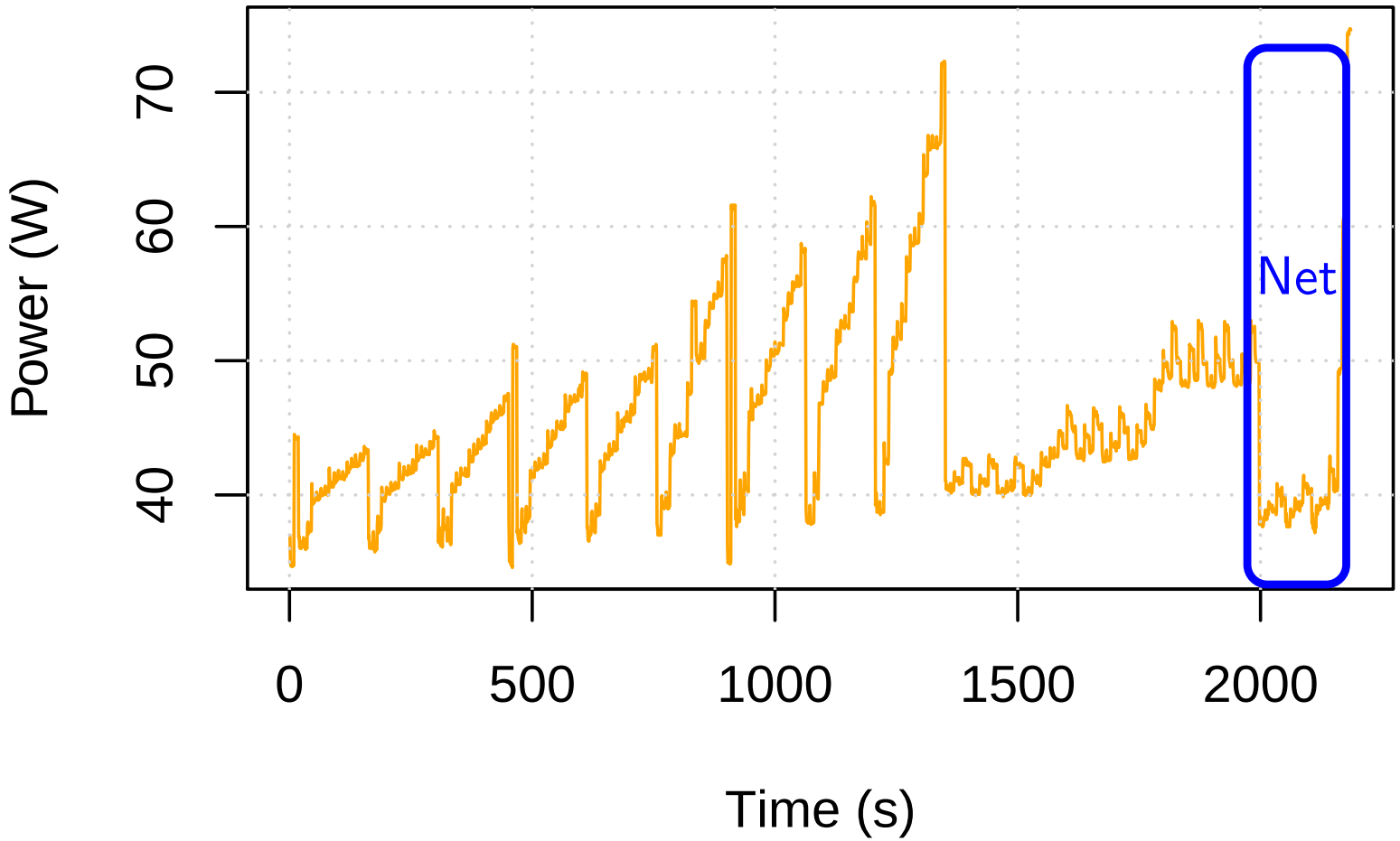
Training set's power profile

- Training set: CPU, memory and network stressed
- 3 frequencies: min (1.2GHz), max (2.3GHz) and boost (3.1GHz)
- Dynamic power: up to 40W of variation



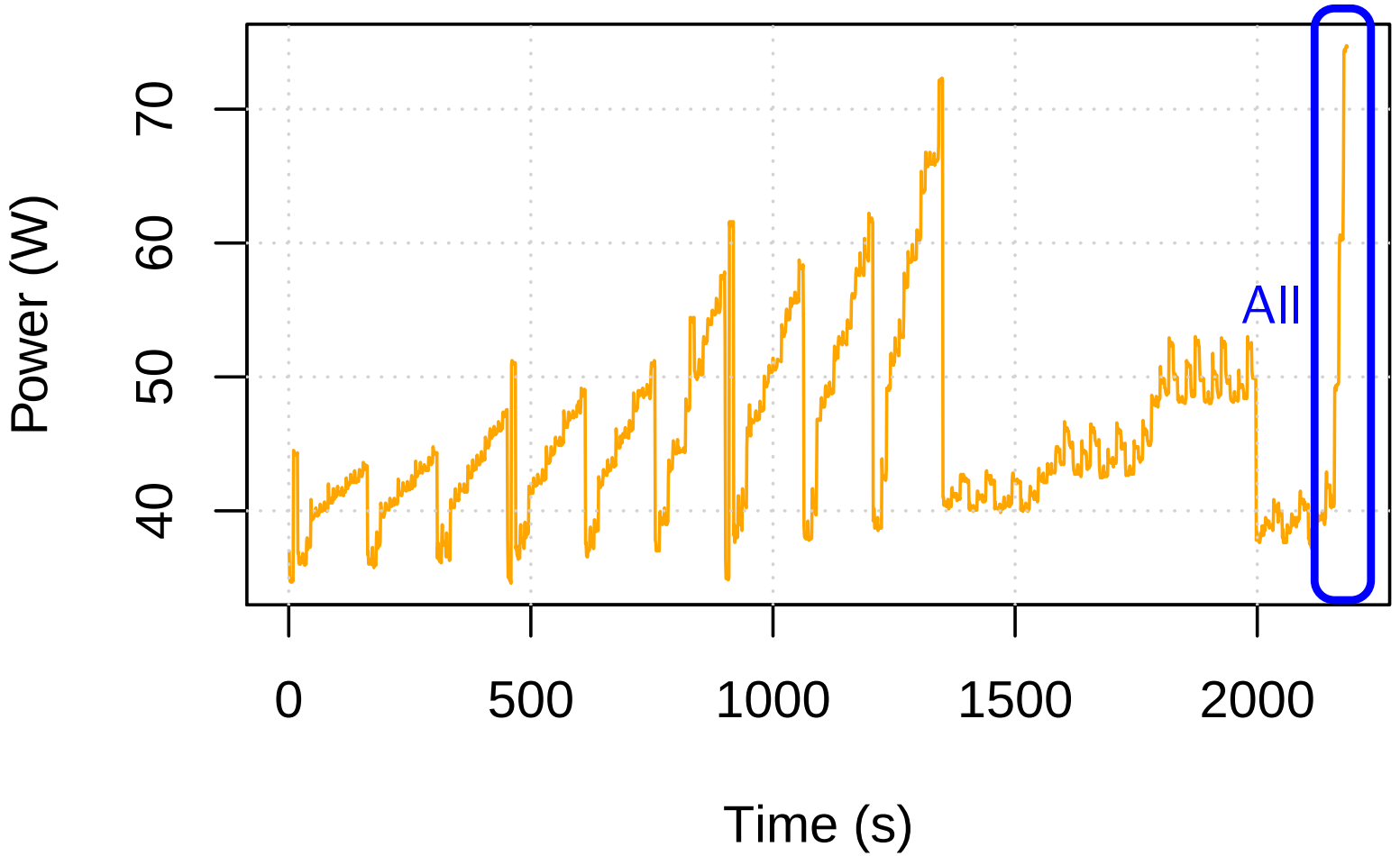
Training set's power profile

- Training set: CPU, memory and network stressed
- 3 frequencies: min (1.2GHz), max (2.3GHz) and boost (3.1GHz)
- Dynamic power: up to 40W of variation



Training set's power profile

- Training set: CPU, memory and network stressed
- 3 frequencies: min (1.2GHz), max (2.3GHz) and boost (3.1GHz)
- Dynamic power: up to 40W of variation



Reference model

- Capacitive model:

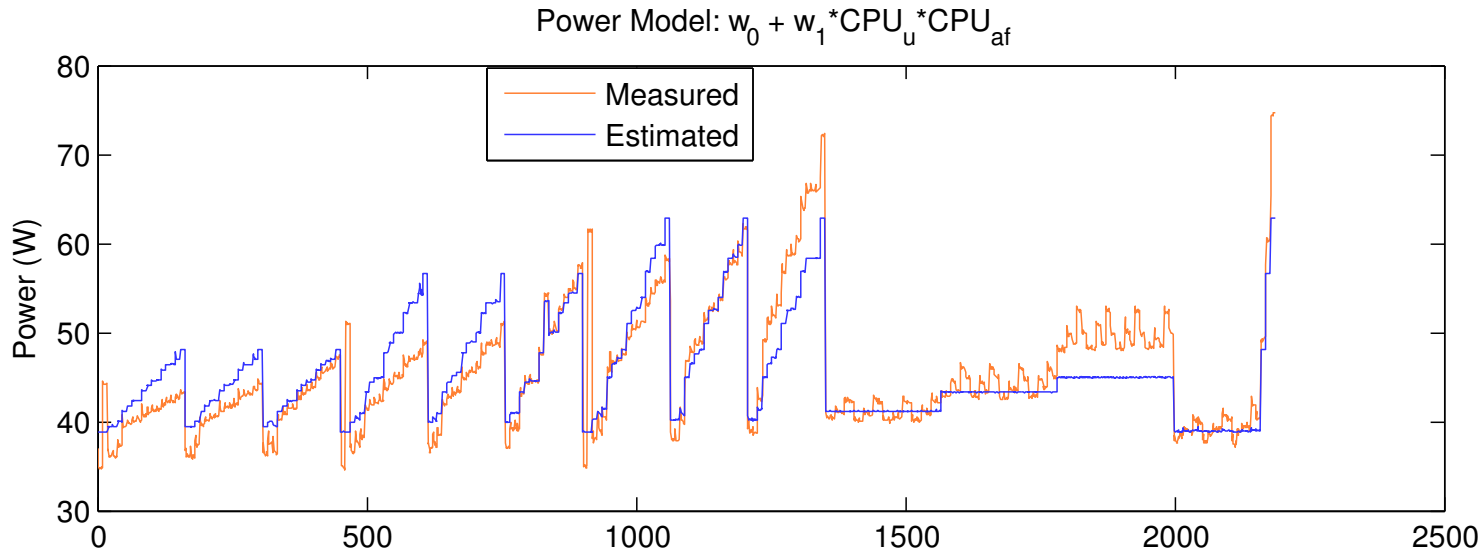
$$P = cv^2 f * u \tag{1}$$

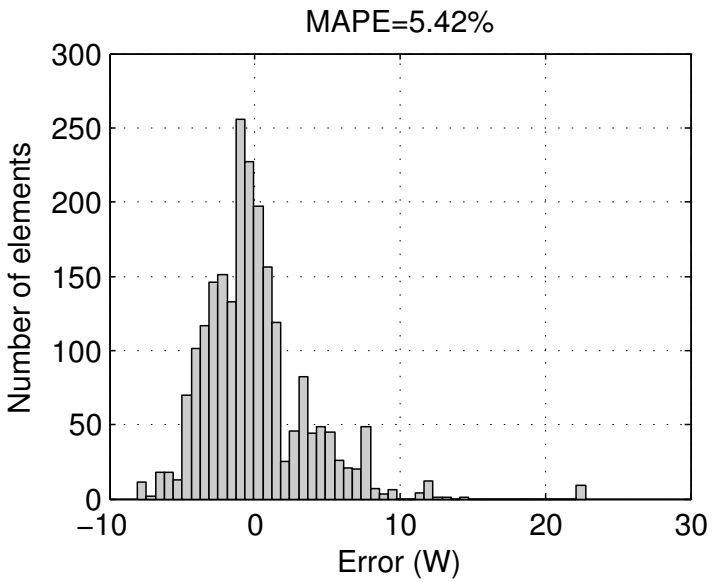
- Simplified version:

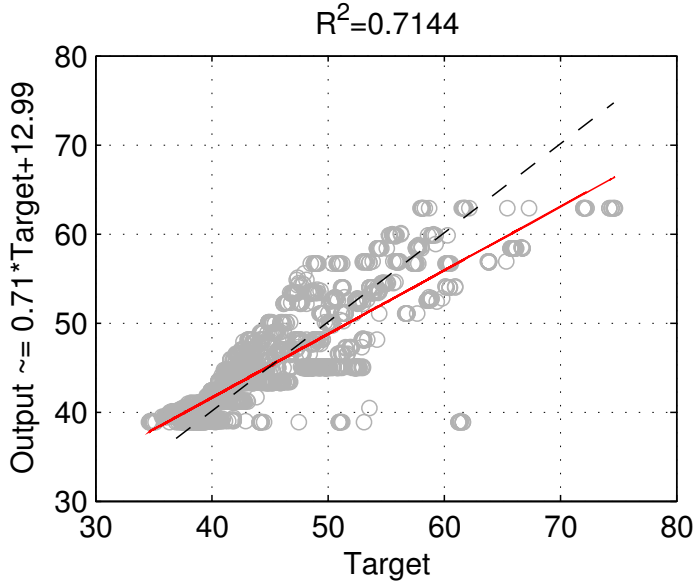
$$P = w_0 + w_1 * f * u \tag{2}$$

- Calibration: Linear Regression

Reference model



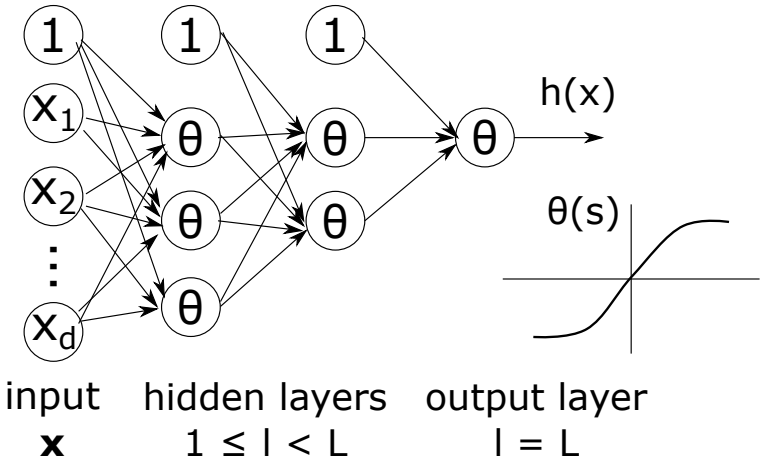




Proposed model

Artificial Neural Networks

- Feedforward Multilayer Perceptron Network topology



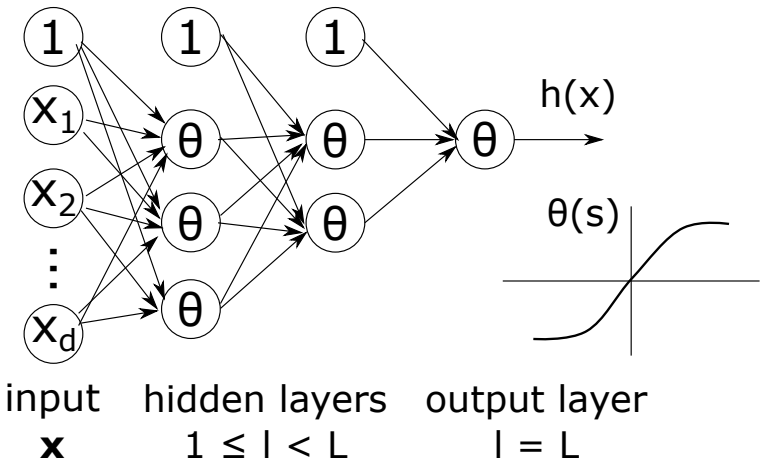
$$x_j^{(l)} = \theta\left(\sum_{i=0}^{d^{l-1}} w_{ij}^{(l)} x_i^{(l-1)}\right) \tag{3}$$

where $\theta(s) = \tanh(s)$

Proposed model

Artificial Neural Networks

- Feedforward Multilayer Perceptron Network topology



$$x_j^{(l)} = \theta\left(\sum_{i=0}^{d^{l-1}} w_{ij}^{(l)} x_i^{(l-1)}\right) \tag{3}$$

where $\theta(s) = \tanh(s)$

- Backpropagation

$$\Delta w_{ij}^{(l)} = -\eta x_i^{(l-1)} \delta_j^{(l)} \tag{4}$$

$$\delta_i^{(l-1)} = \left(1 - \left(x_i^{(l-1)}\right)^2\right) \sum_{j=1}^{d^{(l)}} w_{ij}^{(l)} \delta_j^{(l)} \tag{5}$$

Proposed model

Artificial Neural Networks

ANN setup

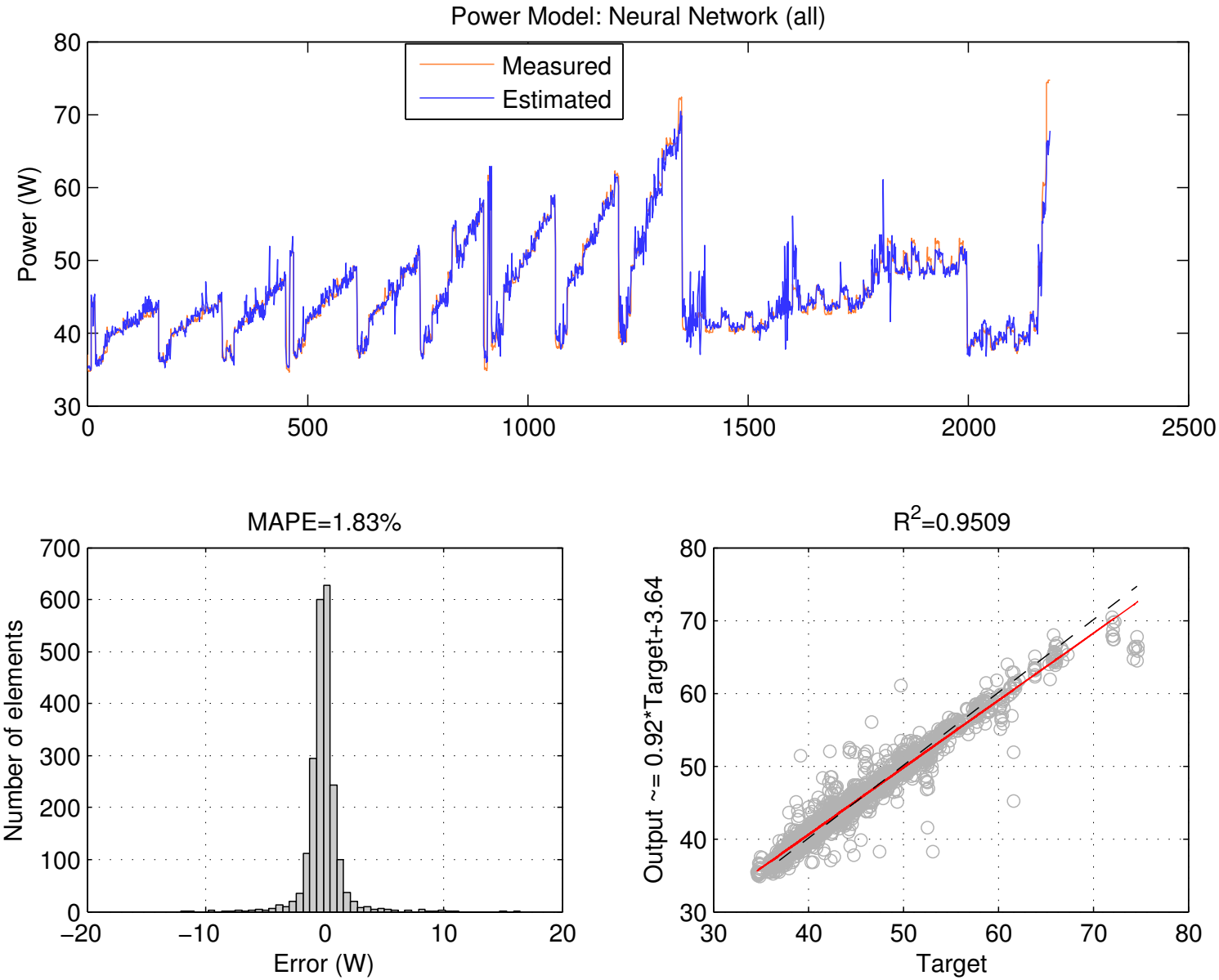
- Topology:
  Feedforward Multilayer Perceptron
  2 hidden layers: 20 5
- Training set:
  70, 15, 15%
- Training algorithm:
  Levenberg-Marquardt

Proposed model

- Variables:

Type	Name	Name
PC	cycles	instructions
	cache references	cache misses
	branch instructions	branch misses
	bus cycles	idle cycles frontend
	idle cycles frontend	cpu clock
	task clock	page faults
	context switches	cpu migrations
	minor faults	major faults
	alignment faults	emulation faults
	L1d loads	L1d load misses
	L1d stores	L1d store misses
	L1d prefetch misses	L1i load misses
	LLC loads	LLC load misses
	LLC stores	LLC stores misses
	L1d prefetches	LLC prefetch misses
	dTLB loads	dTLB load misses
	dTLB stores	dTLB store misses
	iTLB loads	iTLB load misses
	branch loads	branch load misses
	node loads	node load misses
	node stores	node store misses
	node prefetches	node prefetch misses
SYS	cpu usage	resident set size
	received bytes	sent bytes
MSR	cpu frequency	cpu time in C0
	cpu temperature	

Artificial neural network model



Validation

- Use cases:
  HPC Challenge (HPCC), NAS Parallel (NPB), Gromacs, C-Ray, Pybench

Use Case	# Samples		Energy Percentage Error		
	Total	OoB	`const`	capac	ANN
HPCC	87	87	38.41	**15.57**	26.60
NPBA4	75	75	36.37	**16.26**	24.61
NPBB4	329	329	41.11	**19.53**	29.88
Gromacs	964	964	40.03	**17.19**	28.70
C-Ray	17	17	36.16	12.66	**8.13**
PyBench	34	1	8.07	9.13	**0.61**

Validation

- Use cases:
  HPC Challenge (HPCC), NAS Parallel (NPB), Gromacs, C-Ray, Pybench

Use Case	# Samples		Energy Percentage Error		
	Total	OoB	`const`	`capac`	ANN
HPCC	87	87	38.41	**15.57**	26.60
NPBA4	75	75	36.37	**16.26**	24.61
NPBB4	329	329	41.11	**19.53**	29.88
Gromacs	964	964	40.03	**17.19**	28.70
C-Ray	17	17	36.16	12.66	**8.13**
PyBench	34	1	8.07	9.13	**0.61**

Outline

Conclusions and perspectives

Conclusions:

- When using external power meter, energy conversion losses need to be modeled
- CPU proportional models: calibration and accuracy
- Artificial Neural Networks seems promising for generic power estimators

Conclusions and perspectives

Conclusions:

- When using external power meter, energy conversion losses need to be modeled
- CPU proportional models: calibration and accuracy
- Artificial Neural Networks seems promising for generic power estimators

Future work:

- Reduce the number of variables
- Generate a broader training workload to handle HPC use cases
- Validation on distributed applications

Conclusions and perspectives

Conclusions:

- When using external power meter, energy conversion losses need to be modeled
- CPU proportional models: calibration and accuracy
- Artificial Neural Networks seems promising for generic power estimators

Future work:

- Reduce the number of variables
- Generate a broader training workload to handle HPC use cases
- Validation on distributed applications

Thank you!

fontoura@irit.fr