Lecture 2: Experiment Design

PAMS’18

Zsolt István

zsolt.istvan@imdea.org

Slide contents heavily influenced by G. Alonso’s Advanced Systems Lab lecture slides.

Recap

Last week we talked about

- Open/closed systems
- Response time and throughput
- IRTL
- Warm up/cool down phases

Performance Numbers for your System

- Modeling the system under test (SUT)
  - Cheap and quick
  - Applies to wide range of systems
  - Results might not be very accurate, but trends are important
  - No guarantee that the system behaves the same!
- Experimental measurement of the SUT
  - Can be expensive and slow
  - Specific to instance (and even underlying HW)
  - Accurate
  - Repetitions to increase credibility

Steps of Experimentation

- 1) **Preparation**
  - What is the SUT?
  - What is your hypothesis? (the reason for running the experiment!)
  - What workload can best address that hypothesis?
  - How to reduce the effect of unimportant factors?
- 2) **Experimentation**
  - Run workloads, collect ample statistics
  - Make sure that statistics gathering is not altering the SUT behavior significantly
- 3) **Analysis**
  - Report the results – context of the hypothesis
  - If necessary, repeat with refinements

Most important: Hypothesis

- Experimenting without a hypothesis is pointless
  - There is always a lot of “noise” and “interesting trends”, but most of them are probably irrelevant…
- Formulate what is your expectation (make it explicit)
  - Create “mock” graphs, numbers, tables
  - Write down your expected reason for the behavior
- Compare reality with expectation
  - Once the experiment finished, see if it matches or not the expectation
  - If yes: verify that the reasons you suspected make sense
  - If no: Perform more experiments to understand, revise hypothesis

Examples of Hypothesis

- Scenario: We must increase the throughput of our image processing system by 25% to match client demand. We upgraded our servers with 2x faster CPUs (kept RAM and GPU unchanged), but the system is only 5% faster.

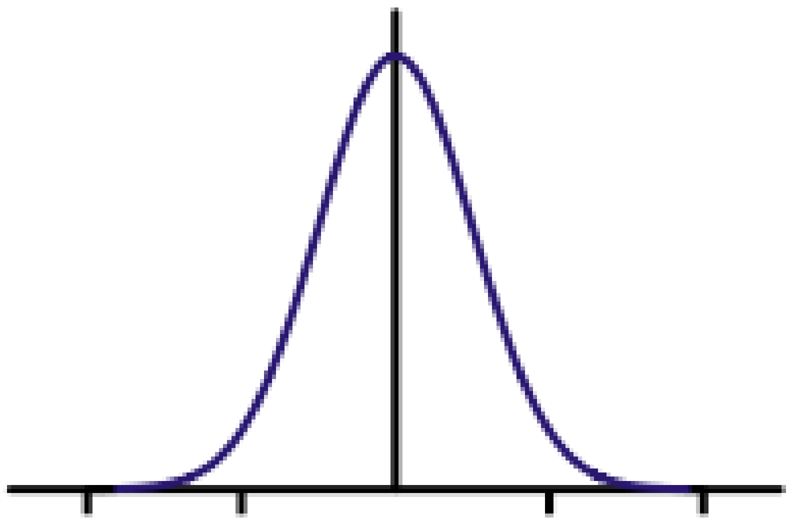
- Hypothesis: …

- Possible experiments: …

- Possible solution: …

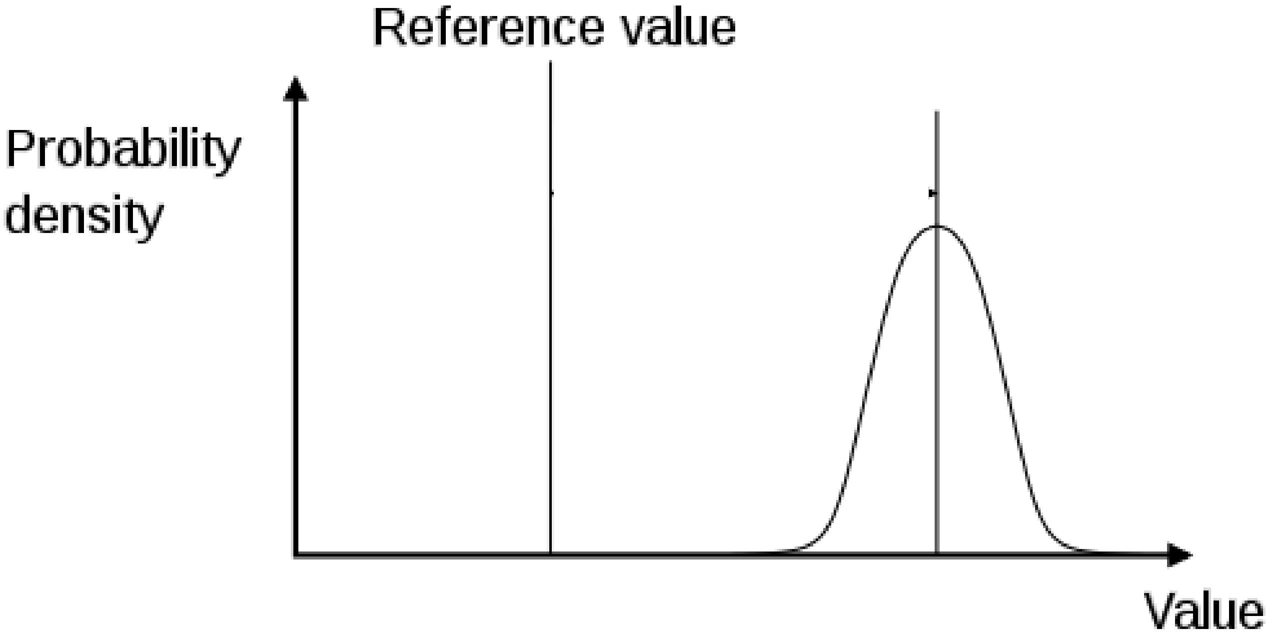
When taking measurements…

- We are repeatedly taking samples of a random variable
- Often no idea of its distribution, but assume “bell curve”
  - (Can be misleading, more on it later)



When taking measurements...

- What is the difference between accuracy and precision?



How to ensure that measurements are **accurate**?

- Accuracy can be affected by noise, outside interference
- Can also depend on what we consider as the SUT

- E.g. We are measuring the RT of our server, once in our office and once deployed in the cloud – we see very different numbers, but the CPUs are quite similar. Why?

How to ensure that measurements are **precise**?

- This is trickier, because the value we are measuring is a random variable with unknown distribution
- Try and take measurements always the same way (same environment, parameters, etc.).
- Include ample information with the results, to separate, for instance, different operations of a workload or different phases of an operation
  - Could help with precision of some of the values

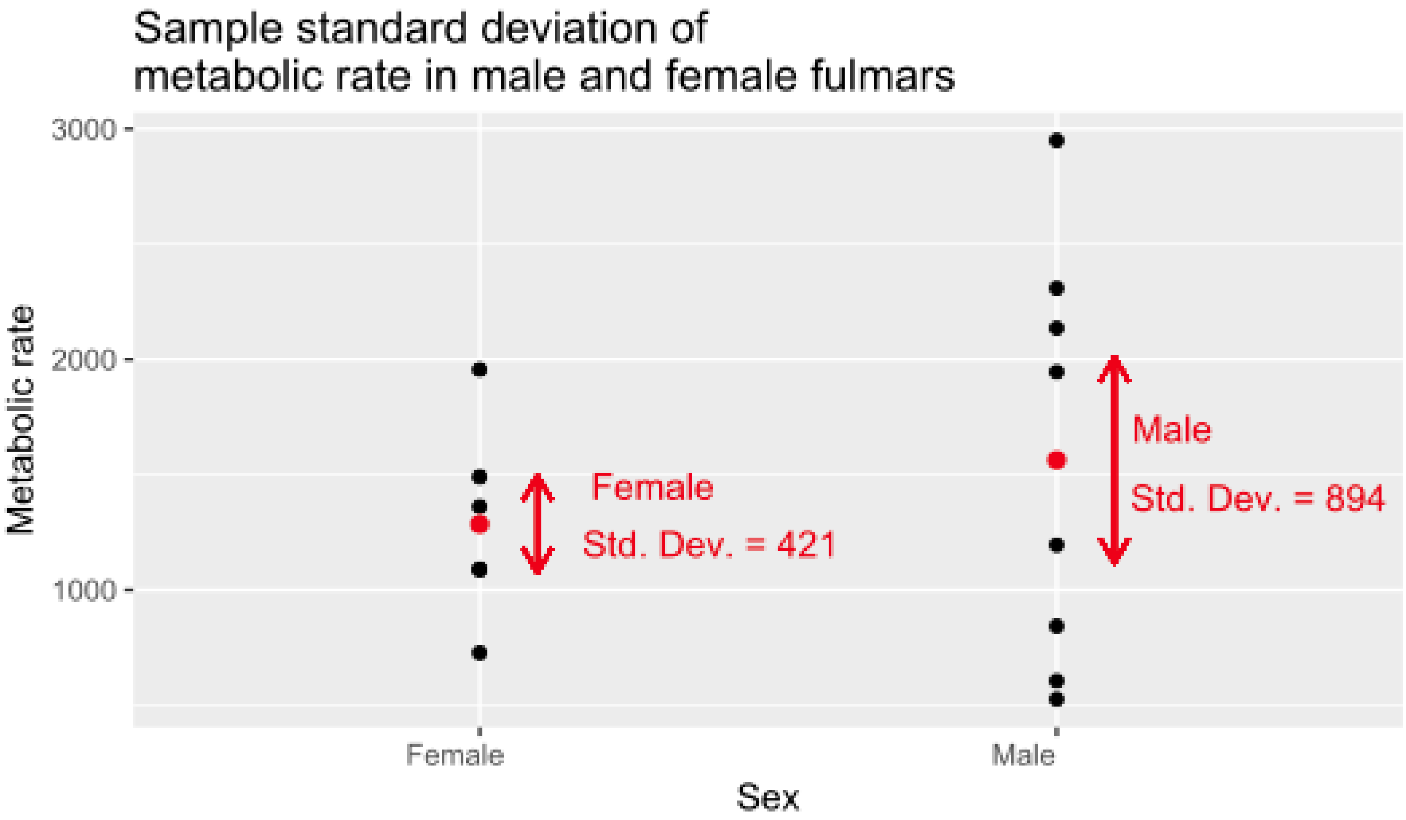
Averages can be useful, but...

- The average is a single number, easy to keep in mind, easy to compute with
- BUT can also hide a lot of details…

- The average number of limbs per human in the US is 3.98
  - Wait, what?

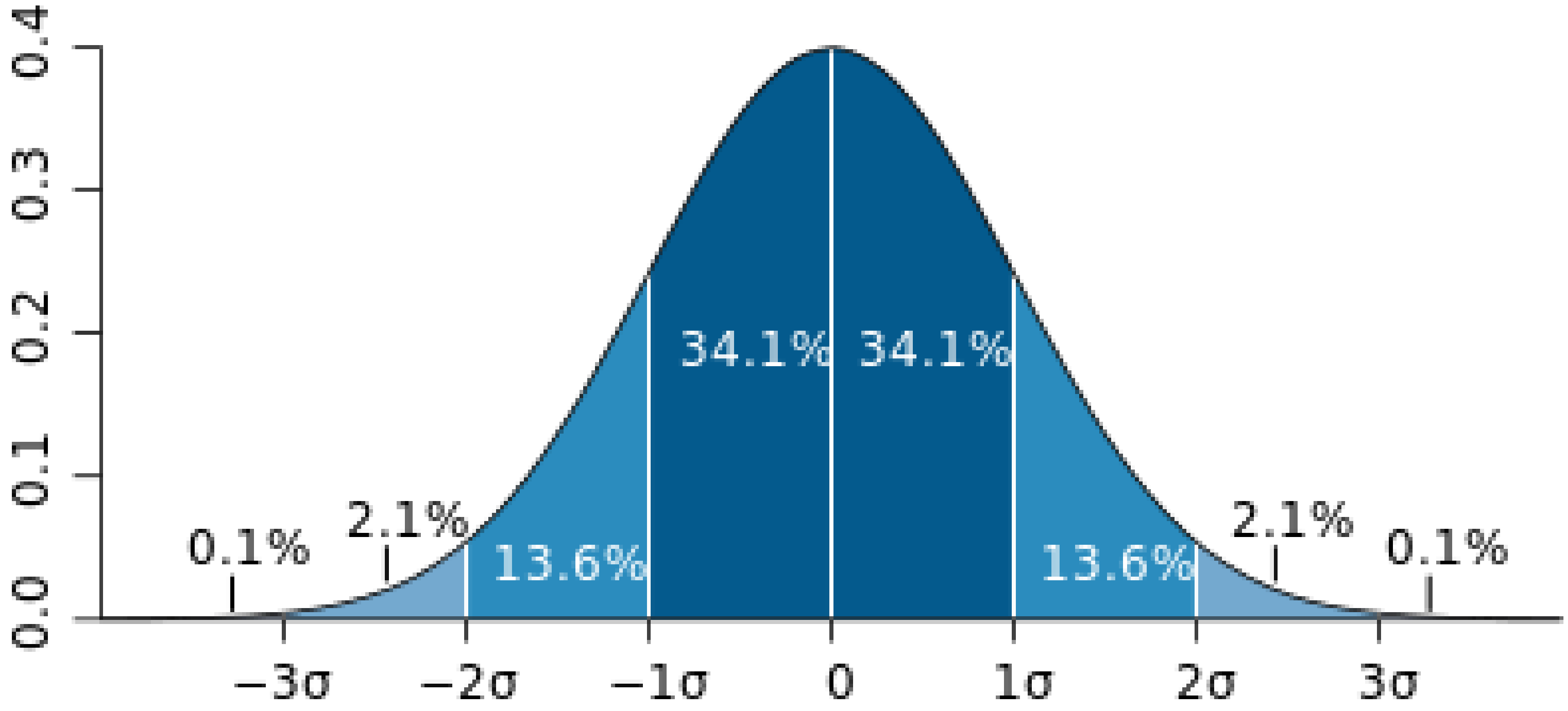
- How to make an average more meaningful?

Average & Standard Deviation



$$s = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}}.$$

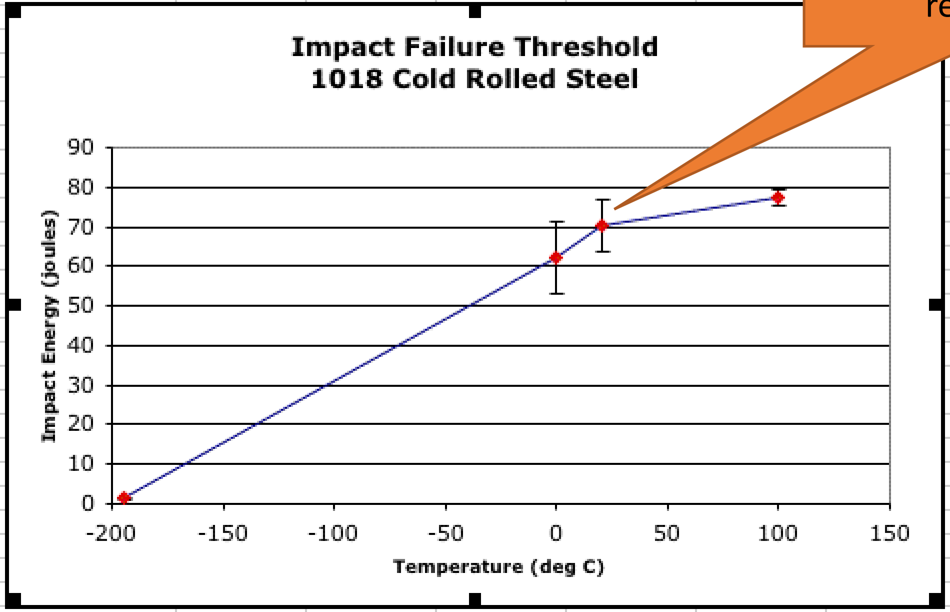
Standard Deviation of Normal Distribution



Standard Error

- Standard Deviation – Variability within a group
- Standard Error – Variability of the means of groups
  - SE = $\frac{SD}{\sqrt{N}}$

	Temperature (C)			
	-195	0	20	100
Impact Energy (joules)	1	52	48	74
	1	58	66	82
	2	82	74	72
	1	35	86	80
	2	84	78	79
Mean	1.4	62.2	70.4	77.4
Standard Deviation	0.5	20.8	14.4	4.2
Standard Error	0.2	9.3	6.5	1.9



Here the error bar represents SE. Can be ambiguous, make sure to tell your readers what the bars represent!

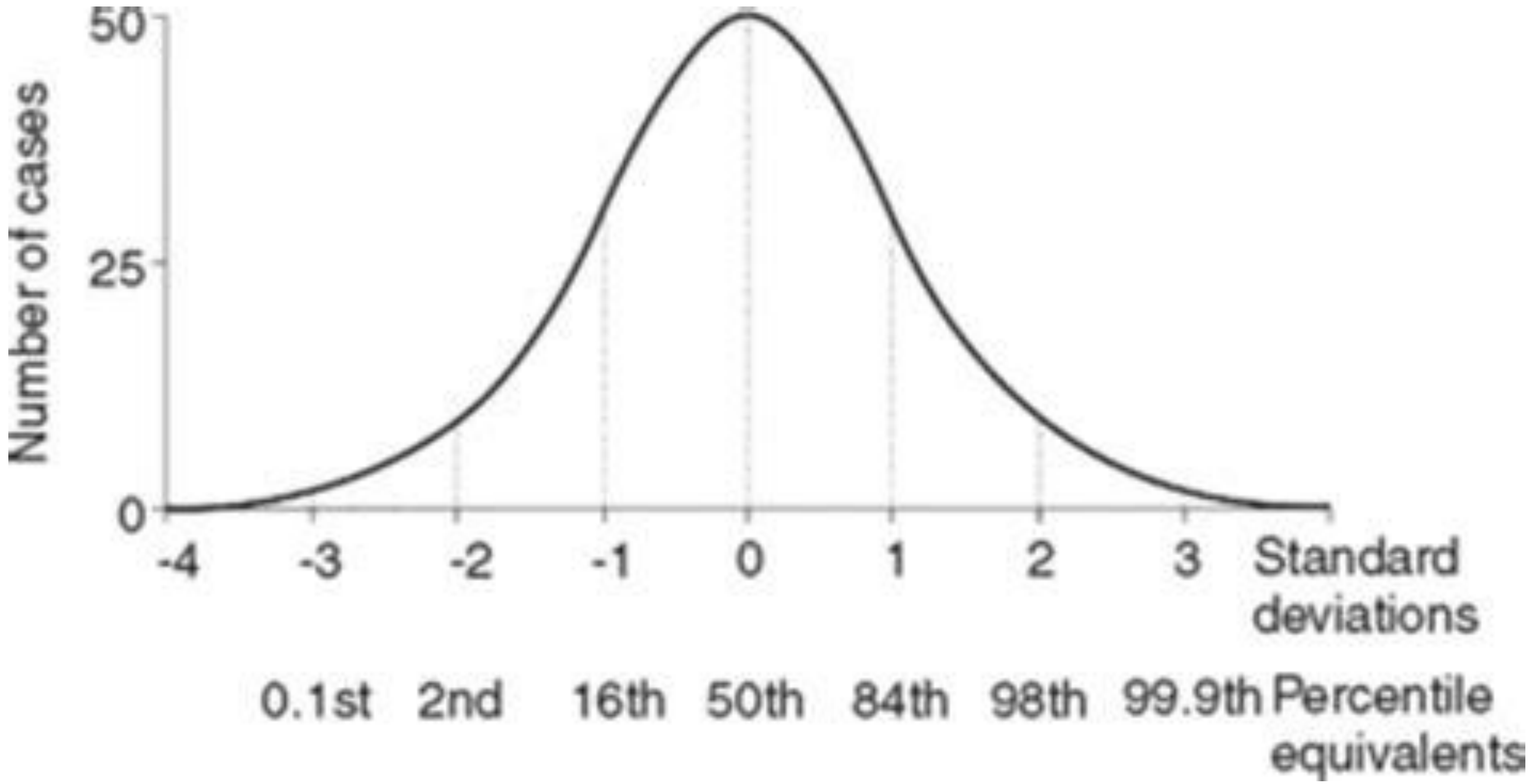
Not everything is a normal distribution!

- Examples at blackboard:

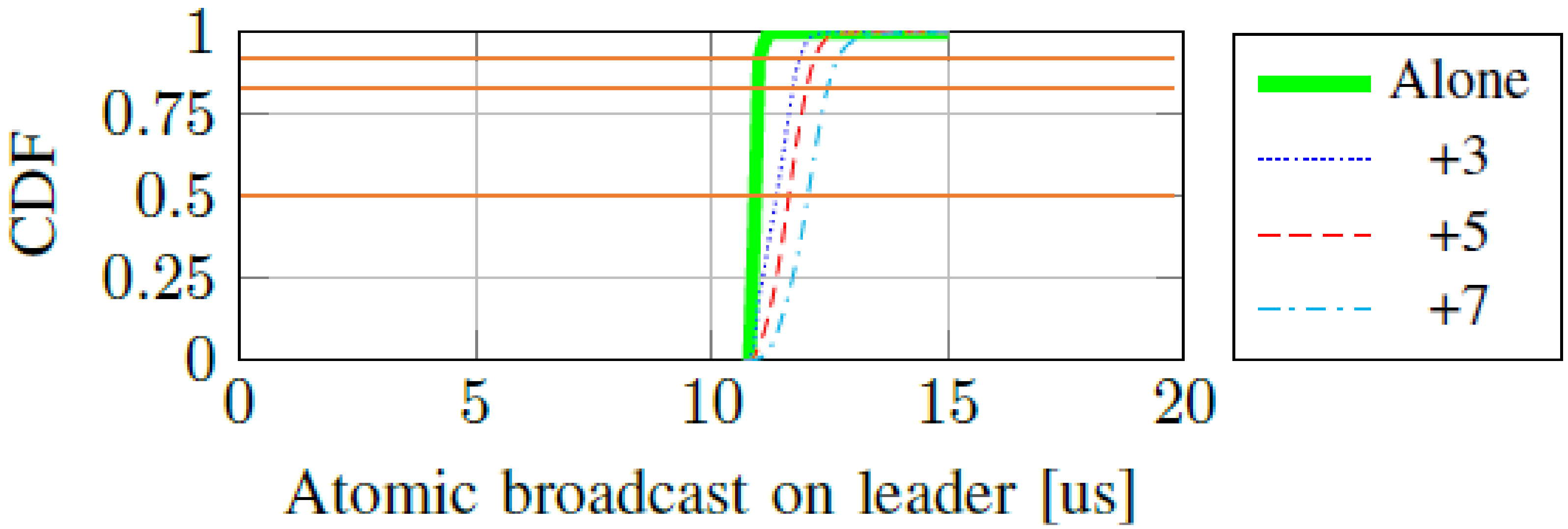
Percentiles

- Need a way of describing the data – especially if not normal distributed
- Xth percentile = X% of the data points are less than the value

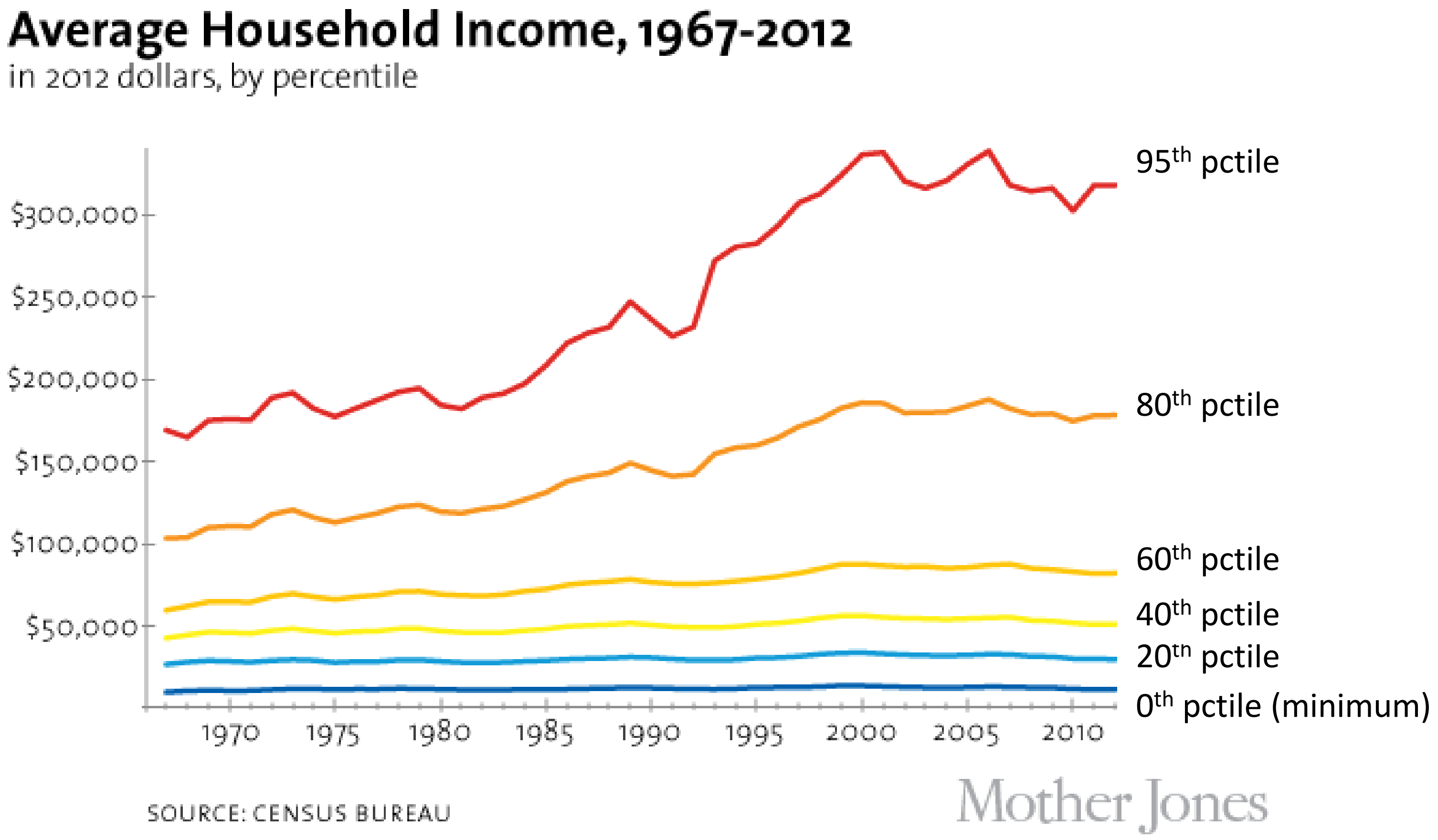
Percentiles of a normal distribution



Percentile example (CDF)

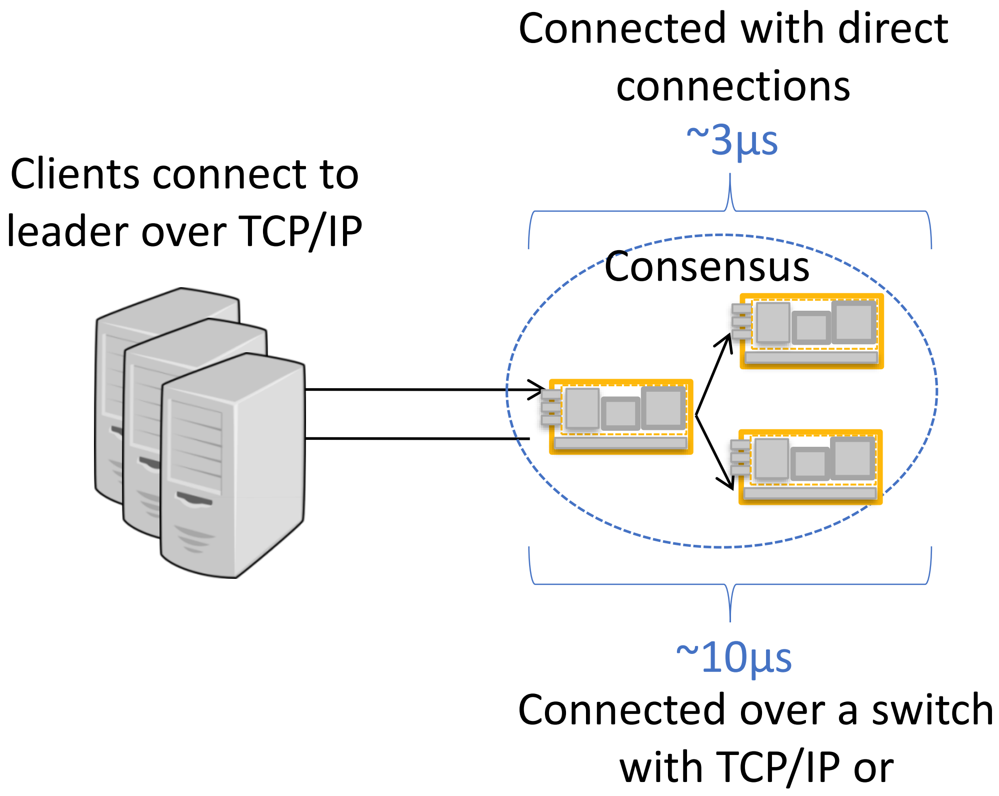


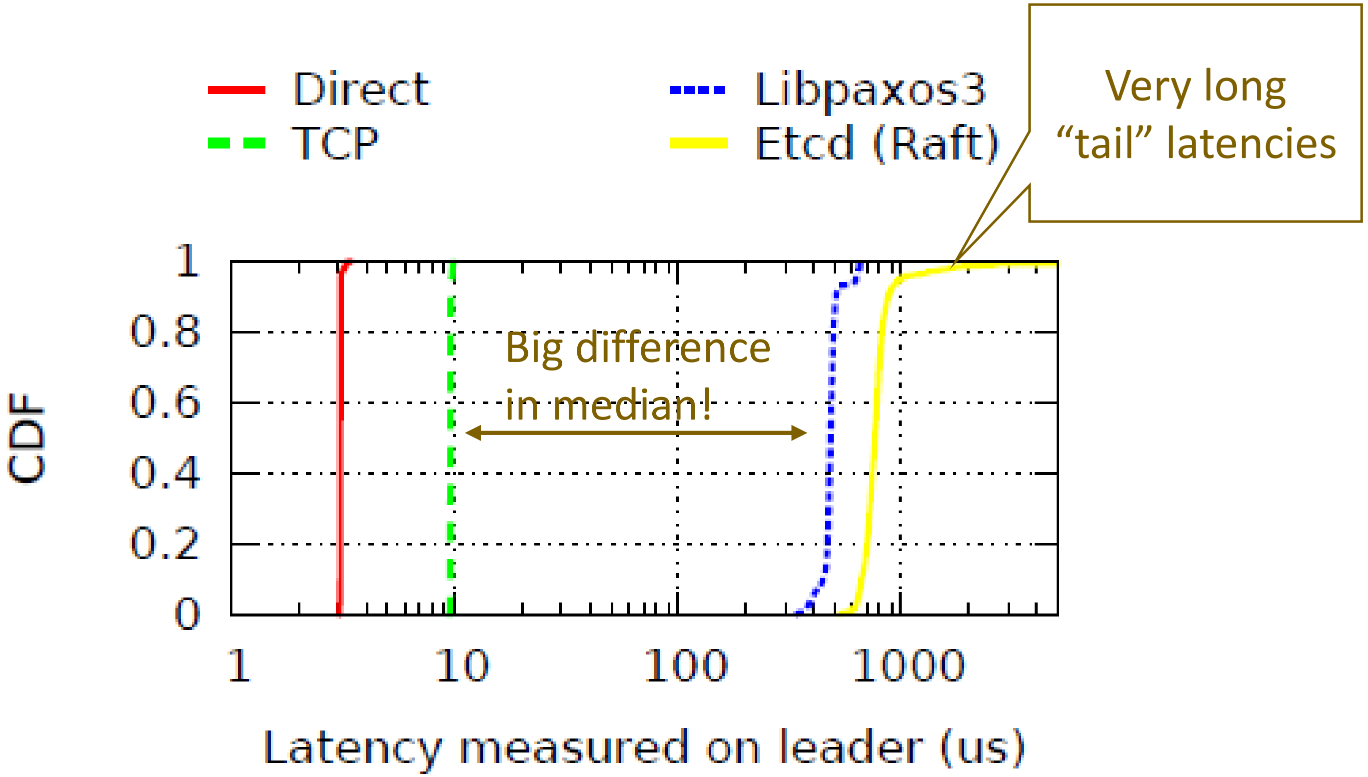
Percentile example (2)



Example from Research

- Our work on Distributed Consensus using specialized hardware (NSDI'16)





Percentiles summary

- Need a way of describing the data – especially if not normal distributed
- Xth percentile = X% of the data points are less than the value

- Average of values == the mean value
- 50th percentile of values == the median value

- **Why do we care in SW about the >99th percentiles?**

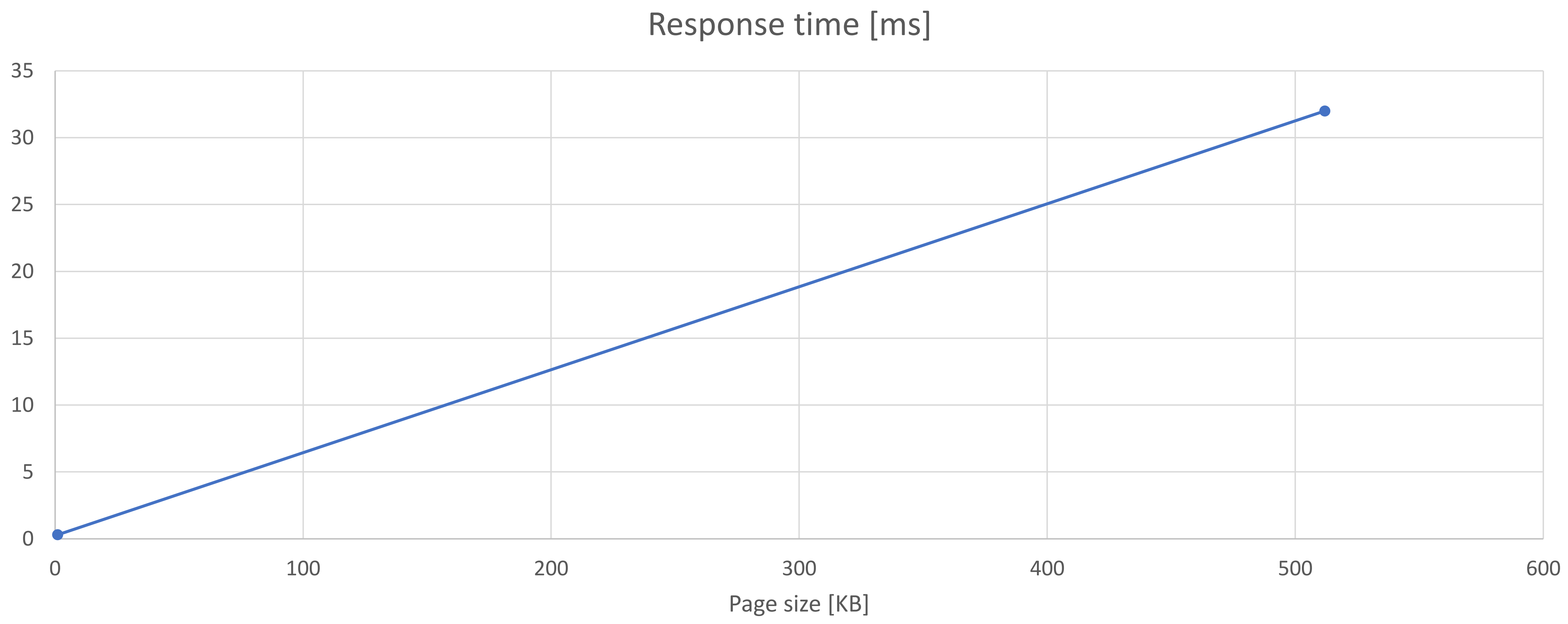
How to explore the effect of a parameter?

- “We are building a web server, how long does it take to serve a webpage?”
  - What could have an effect on the time? E.g. Page size in KB

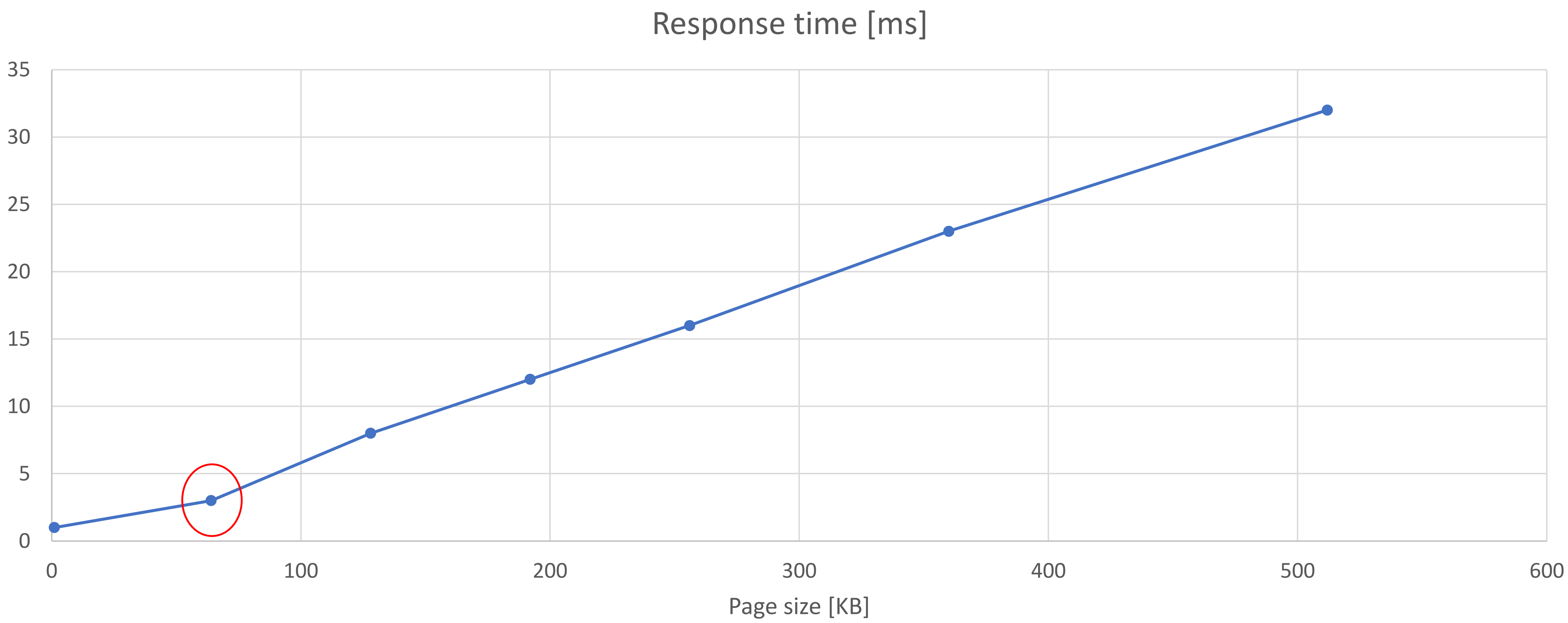
- Experiment with all sizes? 100s, 1000s of points?
  - Experiments are expensive (take minutes, use infrastructure, etc.)
  - Unless behavior is erratic, not all points bring new information...

- *Hypothesis: RT is linear in page size because the server should be network bound.*

Taking extremes

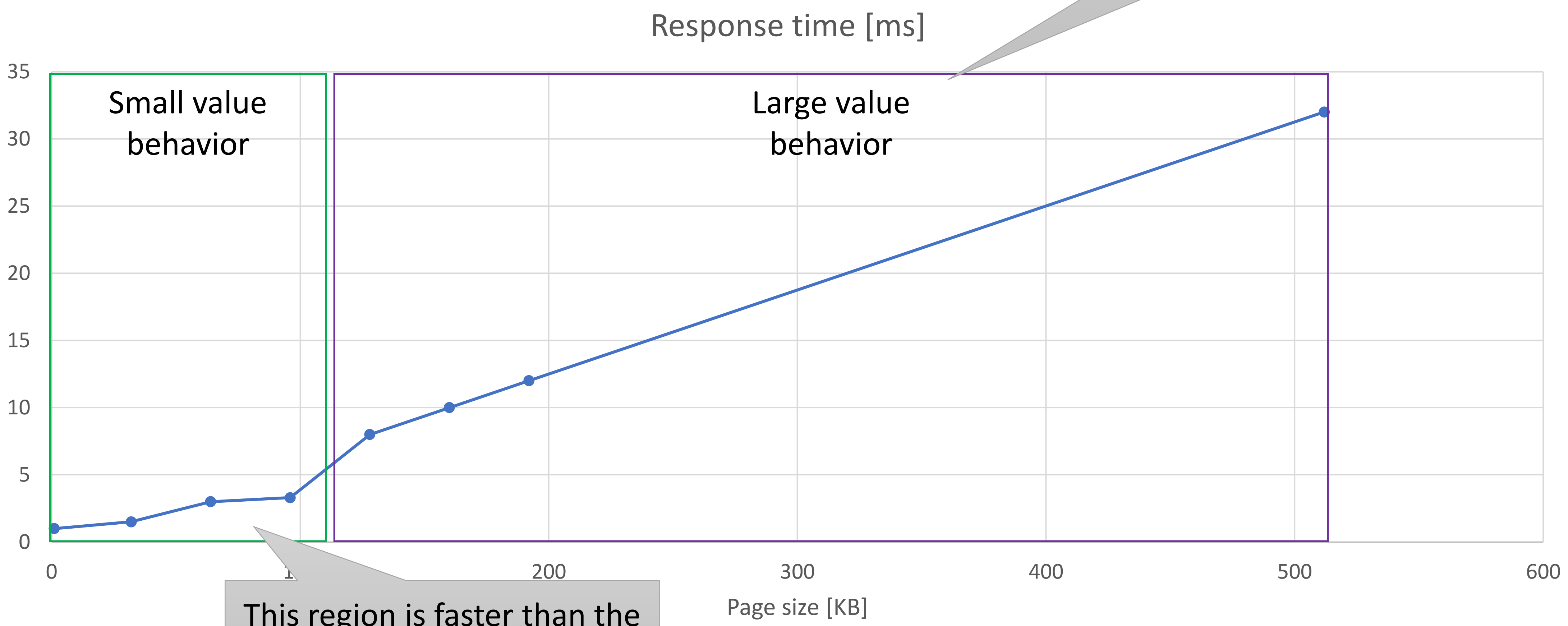


Use several steps



Zoom in

Hypothesis holds in this region, but…

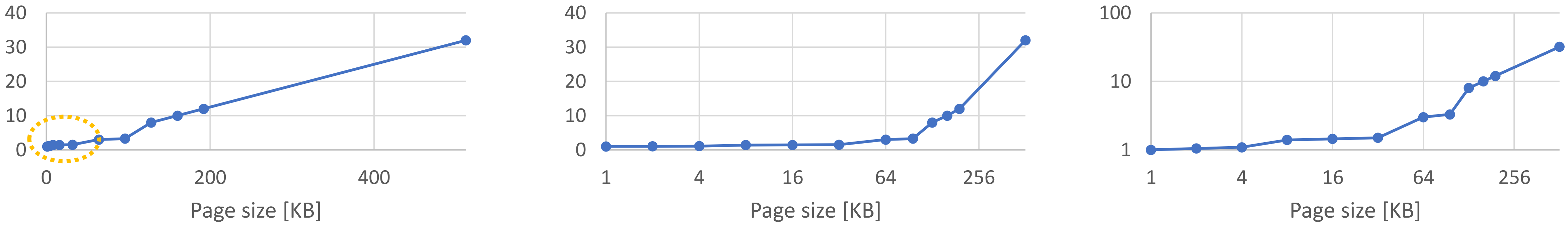


This region is faster than the other (the slope changes). Have to investigate.

Logarithmic steps

- What if we wanted to test up to 10MB page size? Equal steps from 1KB?
  - Can be useful to double/triple the value: 1,2,4,8,16,32,64,… etc.
  - Reach high value sizes in relatively few experiments
  - “Zoom in” as necessary

- If steps are logarithmic, change plotting as well (*caution!*)



Experiment design

- Often more than one factor has effect
  - E.g., request size, number of CPU cores, network connection speed, etc.
- How to determine which one has biggest impact?

- **2^k Factorial experiment**
  - For each factor (can be anything that affects our response variable), consider a low and high level.
  - Measure the system with all combinations (hence the 2^k)
  - Should be combined with repetitions (not covered in this lecture)

Interacting Factors

- Ideally no factor's effect should depend on the level of an other
- In practice some factors can be interacting

	Factor1-low	Factor1-high
Factor2-low	13	23
Factor2-high	16	32

Example of 2^2 Factorial Design

- Running our processing system on different HW platforms

Throughput	1GB Memory	16GB Memory
2MB Cache	32	68
8MB Cache	52	155

- Two factors: Memory (x_A) and Cache (x_B)
  - Low level: x_A =-1
  - High level: x_A =1

The model

- Non linear regression for performance (just a model!)
  - $y = q_0 + q_A x_A + q_B x_B + q_{AB} x_A x_B$

- In our example:

  - $32 = q_0 - q_A - q_B + q_{AB}$
  - $68 = q_0 + q_A - q_B - q_{AB}$
  - $52 = q_0 - q_A + q_B - q_{AB}$
  - $155 = q_0 + q_A + q_B + q_{AB}$

The model (II)

- Computation in a table

q0	qA	qB	qAB	y
1	-1	-1	1	32
1	1	-1	-1	68
1	-1	1	-1	52
1	1	1	1	155
307	139	107	67	Total
76.75	34.75	26.75	16.75	Total/4

- After solving the system:
  - q_0 = 76.75 (average of experiments)
  - q_A = 34.75 (effect of Memory)
  - q_B = 26.75 (effect of Cache)
  - q_{AB} = 16.75 (effect of the interaction between the two)