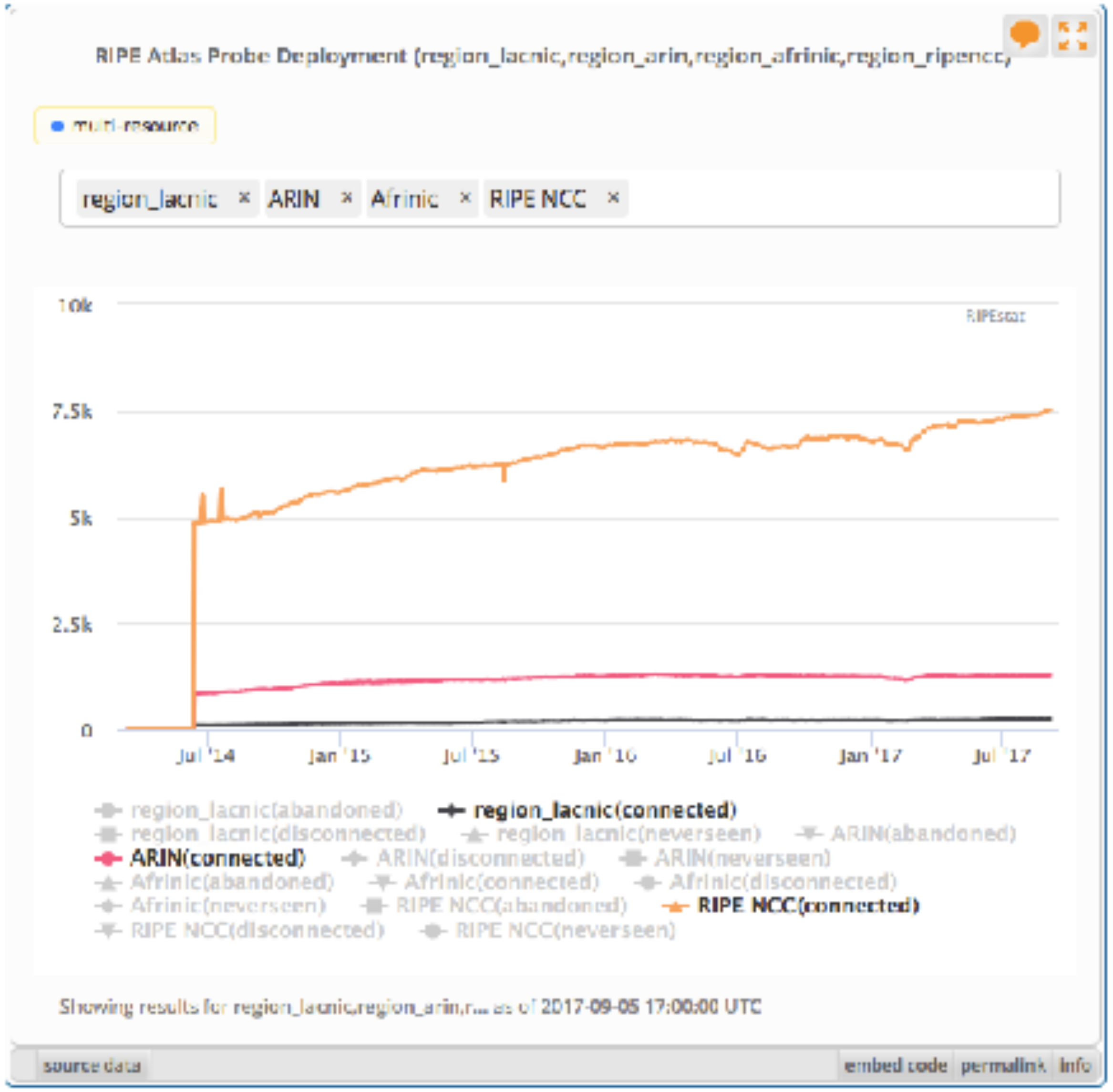
Comparing a set of latency measurements

RIPE MAT WG
October 2017
Agustín Formoso
LACNIC Labs
@aguformoso @ProyectoSimon

Motivation: coverage

- Lack of coverage of Atlas probes in the LAC region



RIPE Atlas Probe Deployment (region_lacnic,region_arin,region_afrinic,region_ripencc)

Showing results for region_lacnic,region_arin,r... as of 2017-09-05 17:00:00 UTC

RIPE NCC

7517 conn. Atlas probes

24k active ASes

30% coverage

ARIN

1260 conn. Atlas probes

LACNIC

242 conn. Atlas probes

6k active ASes

3.8% coverage

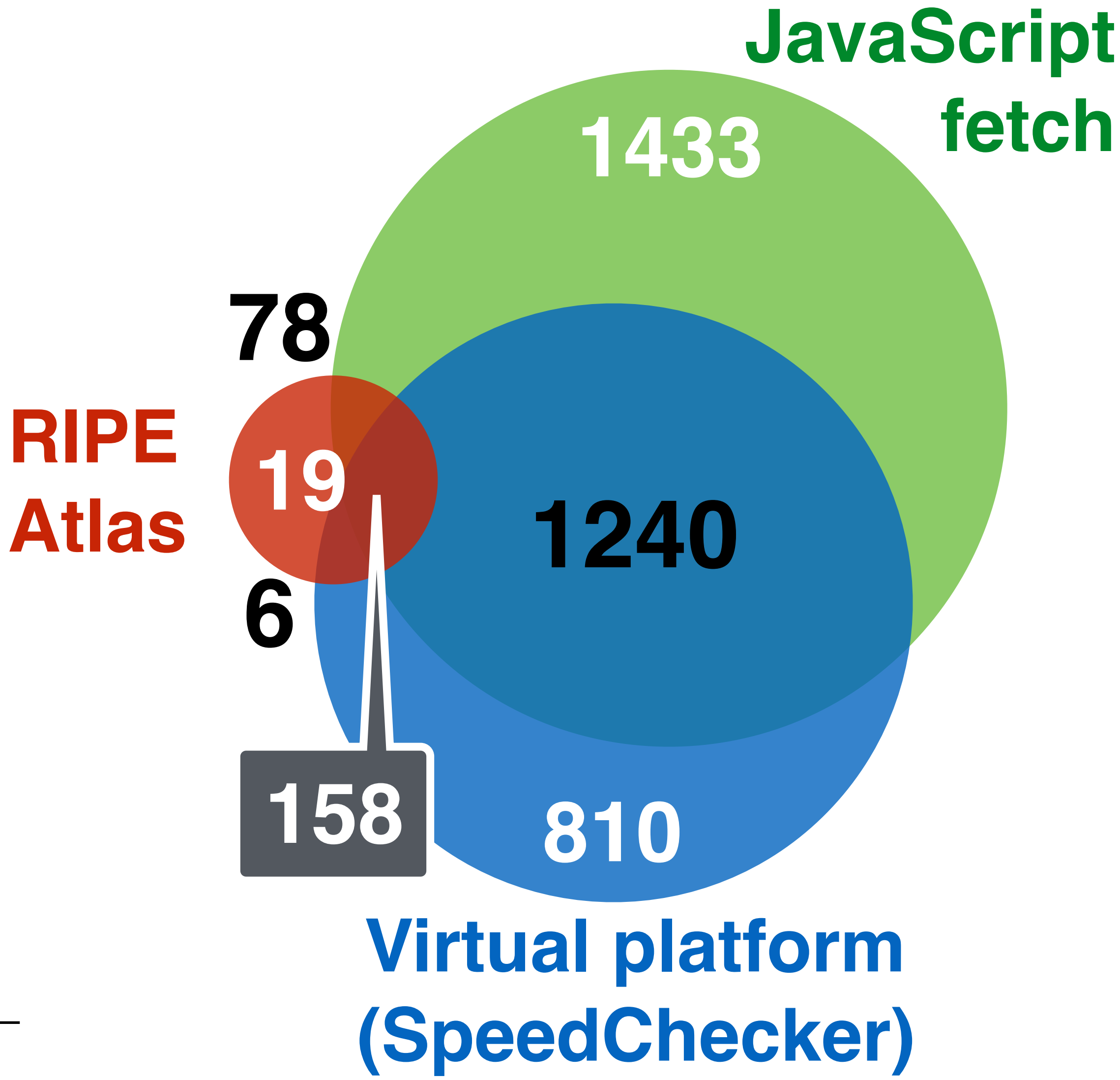
Motivation: coverage

- What if we could perform comparable measurements using other platform? (RTT, or equivalent)
    - Cristian Varas @ RIPE 73
    - Randy Bush @ RIPE 69
- Let’s run the alternatives in parallel and see what results we get…

Motivation: coverage

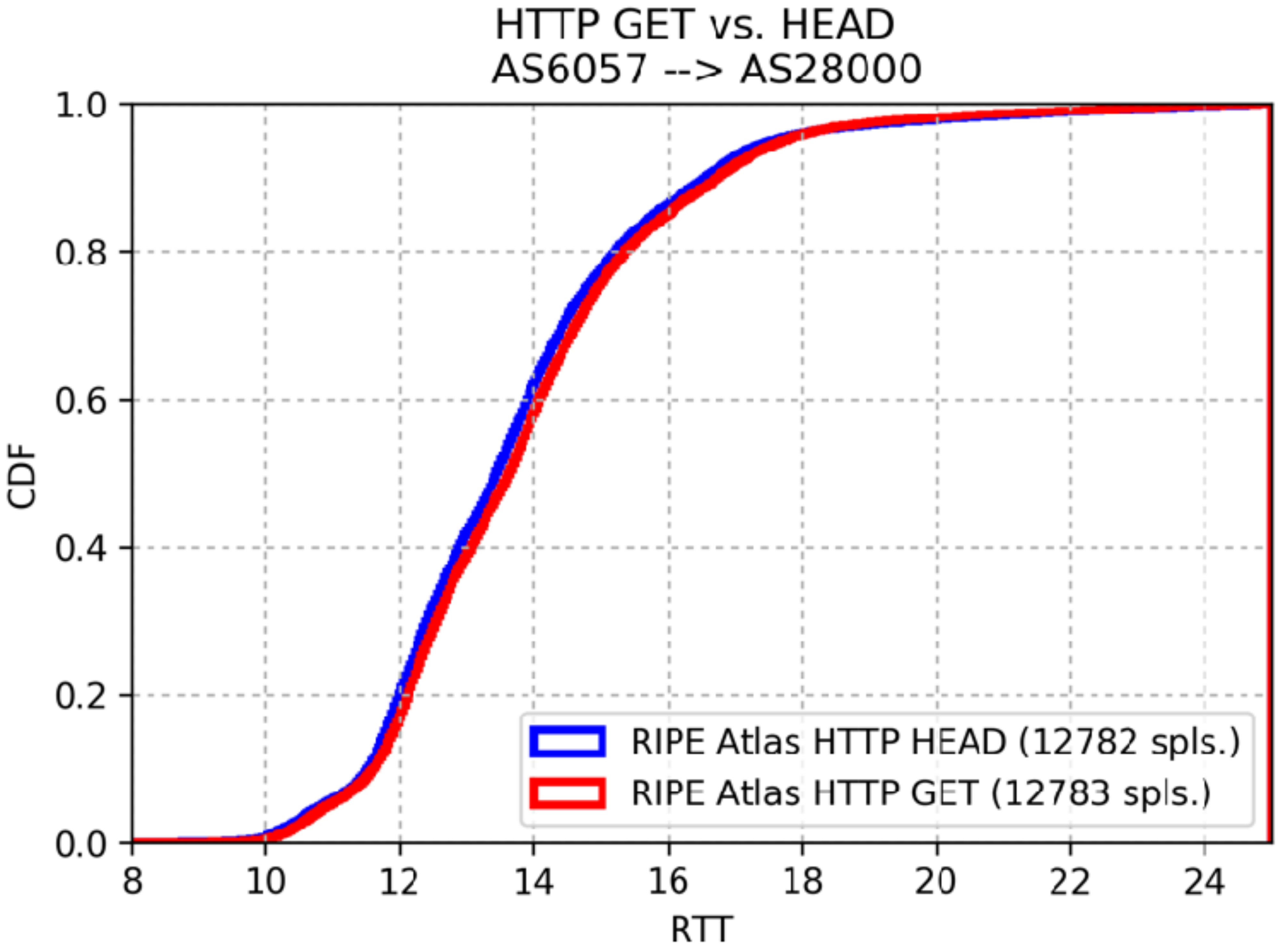
- Going virtual: huge coverage
- Measure differences in the intersections
- How can one help the other?

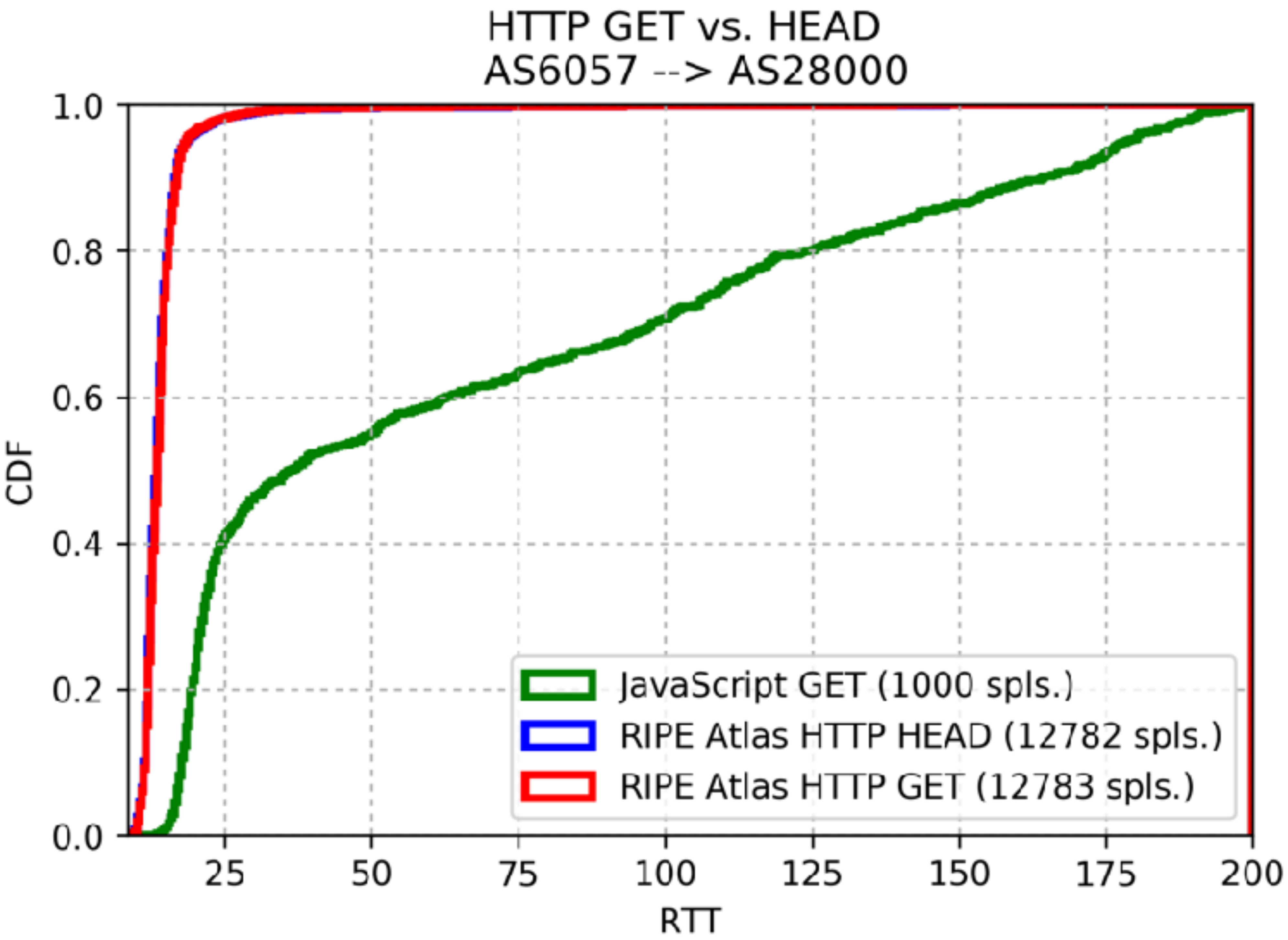
	LACNIC AS count	% active ASes
Atlas	261	4%
Virtual platform	2214	34%
Javascript	2909	**44%**



ASes covered by platforms

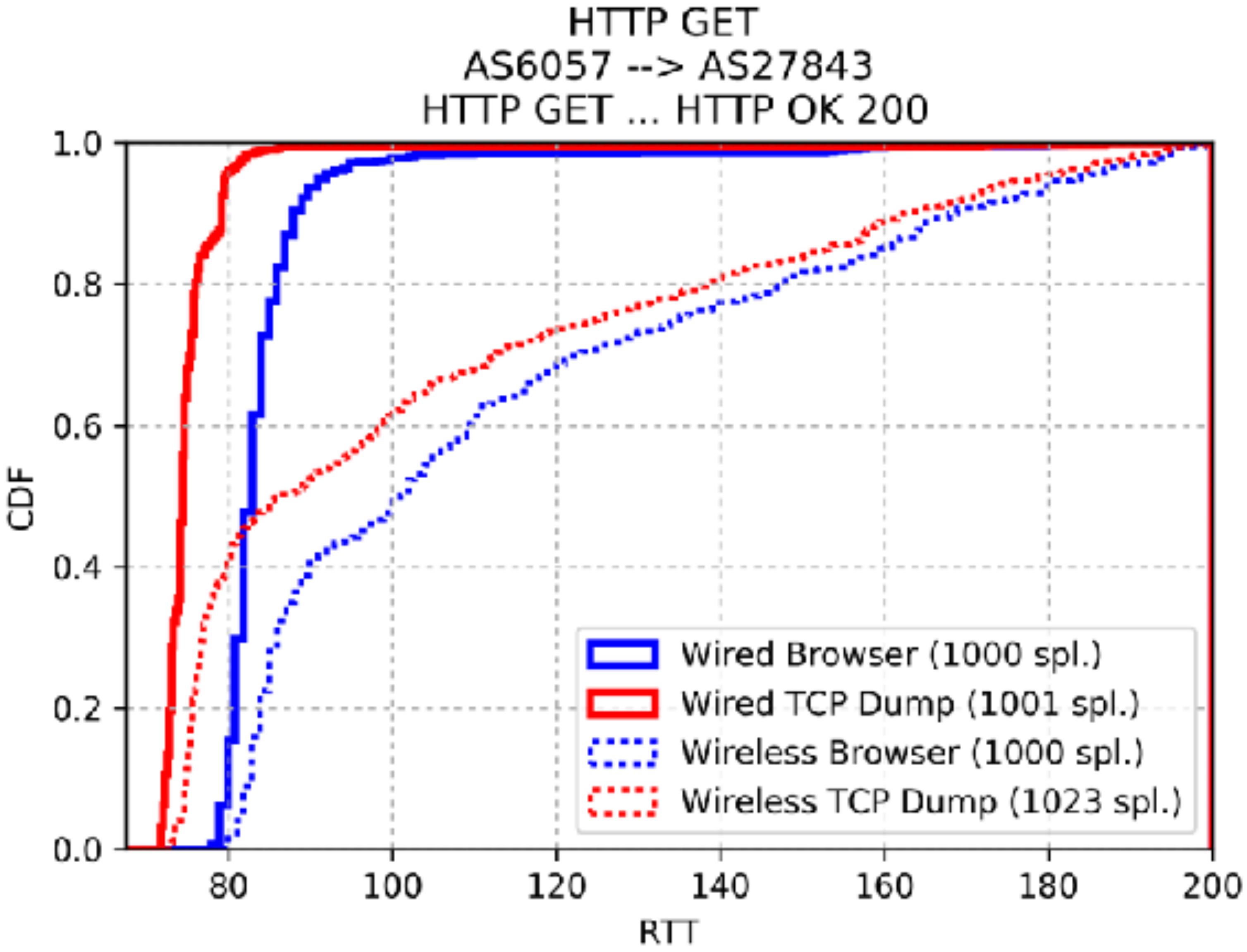
Lab: one week

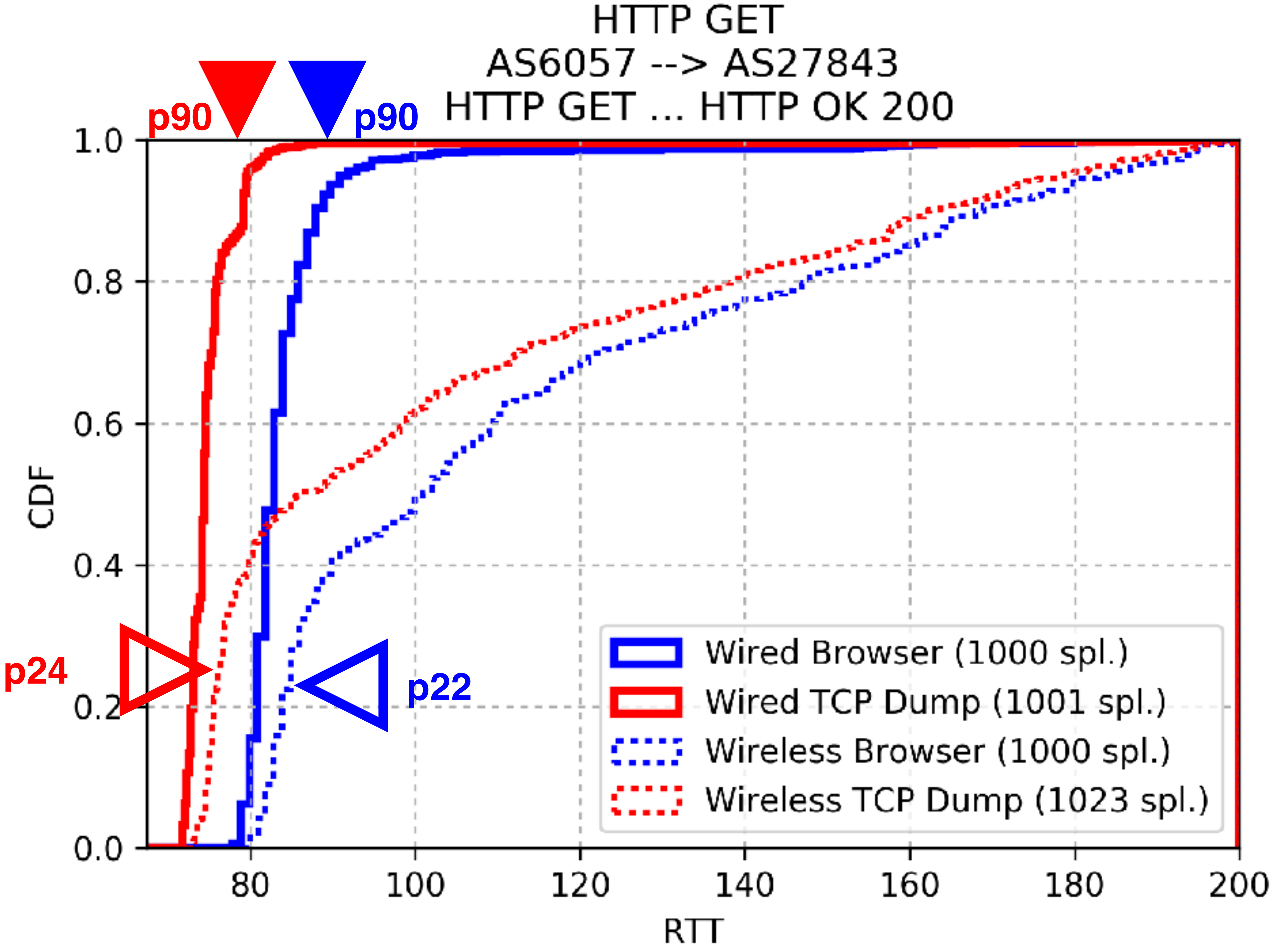

HTTP GET vs. HEAD
AS6057 --> AS28000
CDF
1.0
0.8
0.6
0.4
0.2
0.0
8
10
12
14
16
18
20
22
24
RTT
RIPE Atlas HTTP HEAD (12782 spls.)
RIPE Atlas HTTP GET (12783 spls.)


HTTP GET vs. HEAD
AS6057 --> AS28000
CDF
1.0
0.8
0.6
0.4
0.2
0.0
25
50
75
100
125
150
175
200
RTT
🟩 JavaScript GET (1000 spls.)
🟦 RIPE Atlas HTTP HEAD (12782 spls.)
🟥 RIPE Atlas HTTP GET (12783 spls.)

Wired vs. Wireless

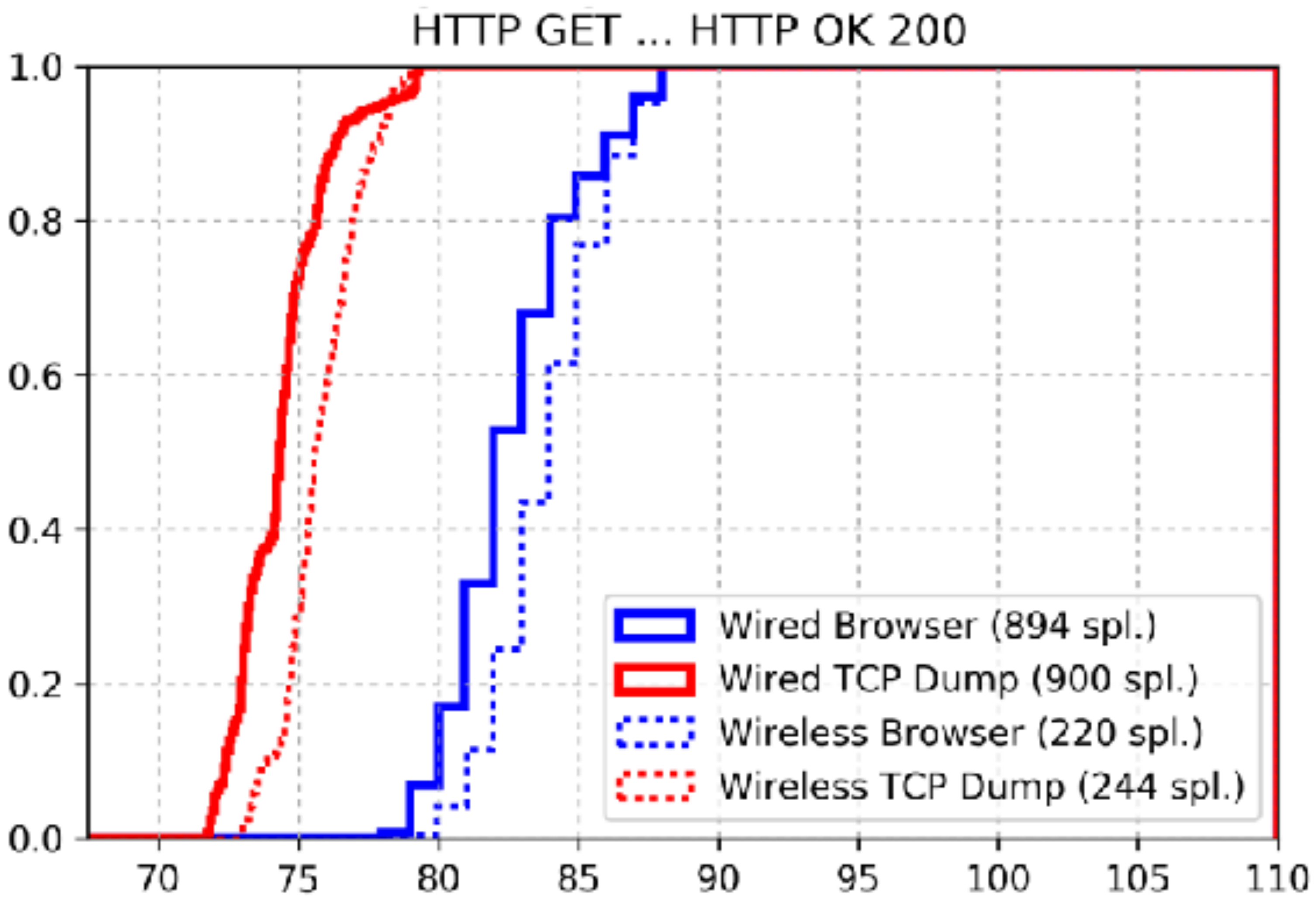
- Launch the JavaScript tester for my laptop
- Get RTTs as seen from the browser
- …and as seen from tcpdump utility
- Repeat for wired connection immediately after




HTTP GET
AS6057 --> AS27843
HTTP GET ... HTTP OK 200
p90
p90
p24
p22
CDF
RTT
0.0
0.2
0.4
0.6
0.8
1.0
80
100
120
140
160
180
200
Wired Browser (1000 spl.)
Wired TCP Dump (1001 spl.)
Wireless Browser (1000 spl.)
Wireless TCP Dump (1023 spl.)

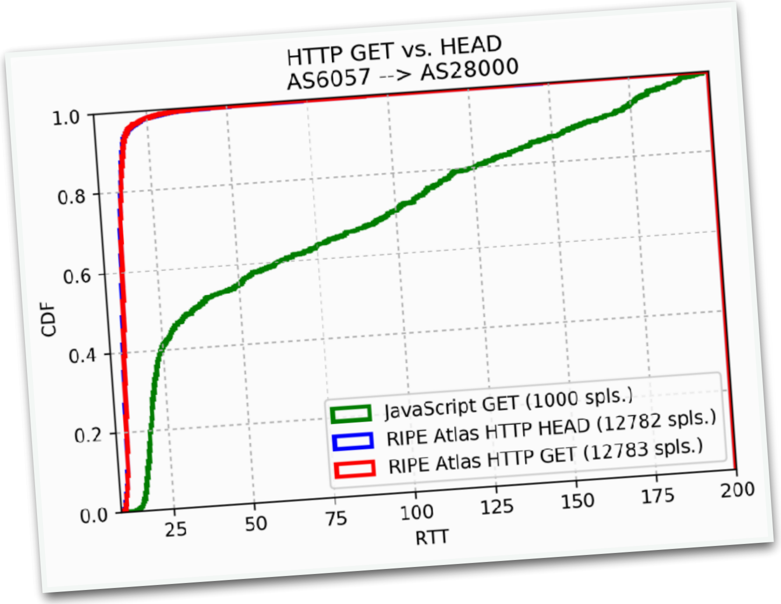
Wired vs. Wireless

- Set wired p90 as the cutoff
- Wireless best cases remain
  - Results in somewhere between p22 / p24
  - That's ~78 / 76% samples that were slowed down in the wireless process
  - Keeping p90 is unhealthy in this case.
- **p22 rule-of-thumb** for JavaScript measurements
- **Remaining constant across percentiles**. Depends on…? Stack config?



TCP Dump	❘ ⋮	+1.21 ms
Browser	❘ ⋮	+2.0 ms
Browser overhead	❘ ❘	+7.6 ms

Wired vs. Wireless

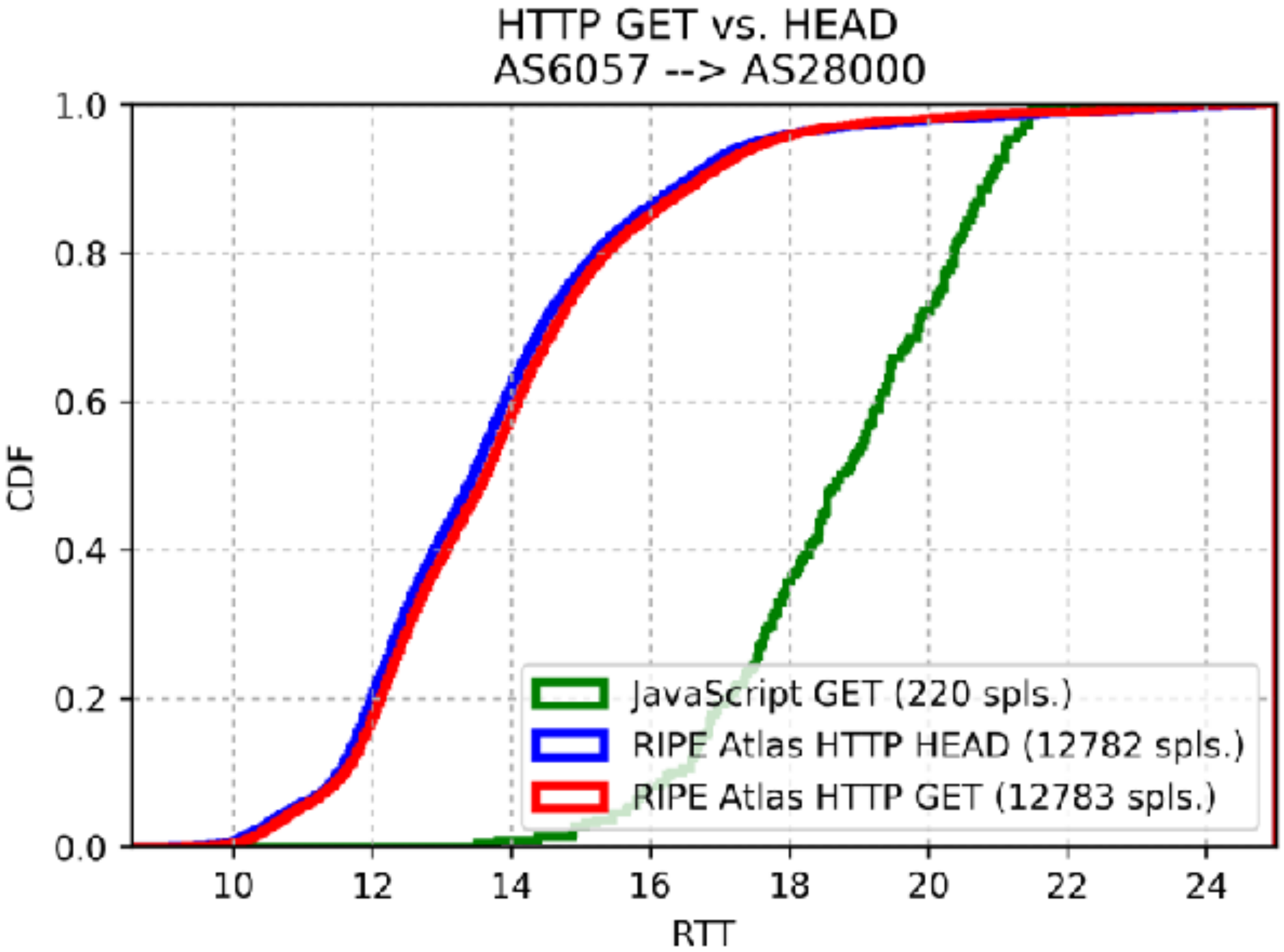


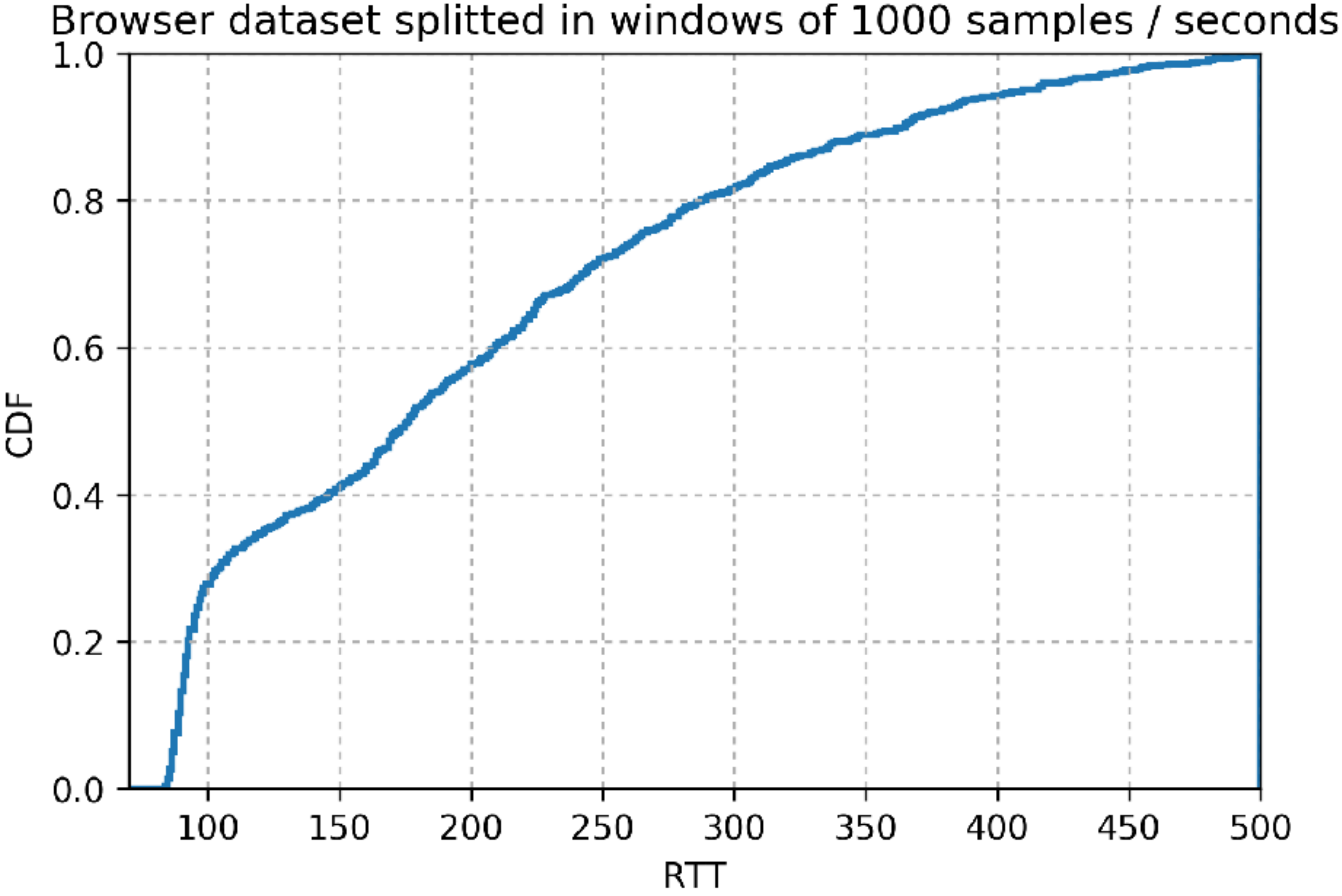
- Stripping out everything over p22

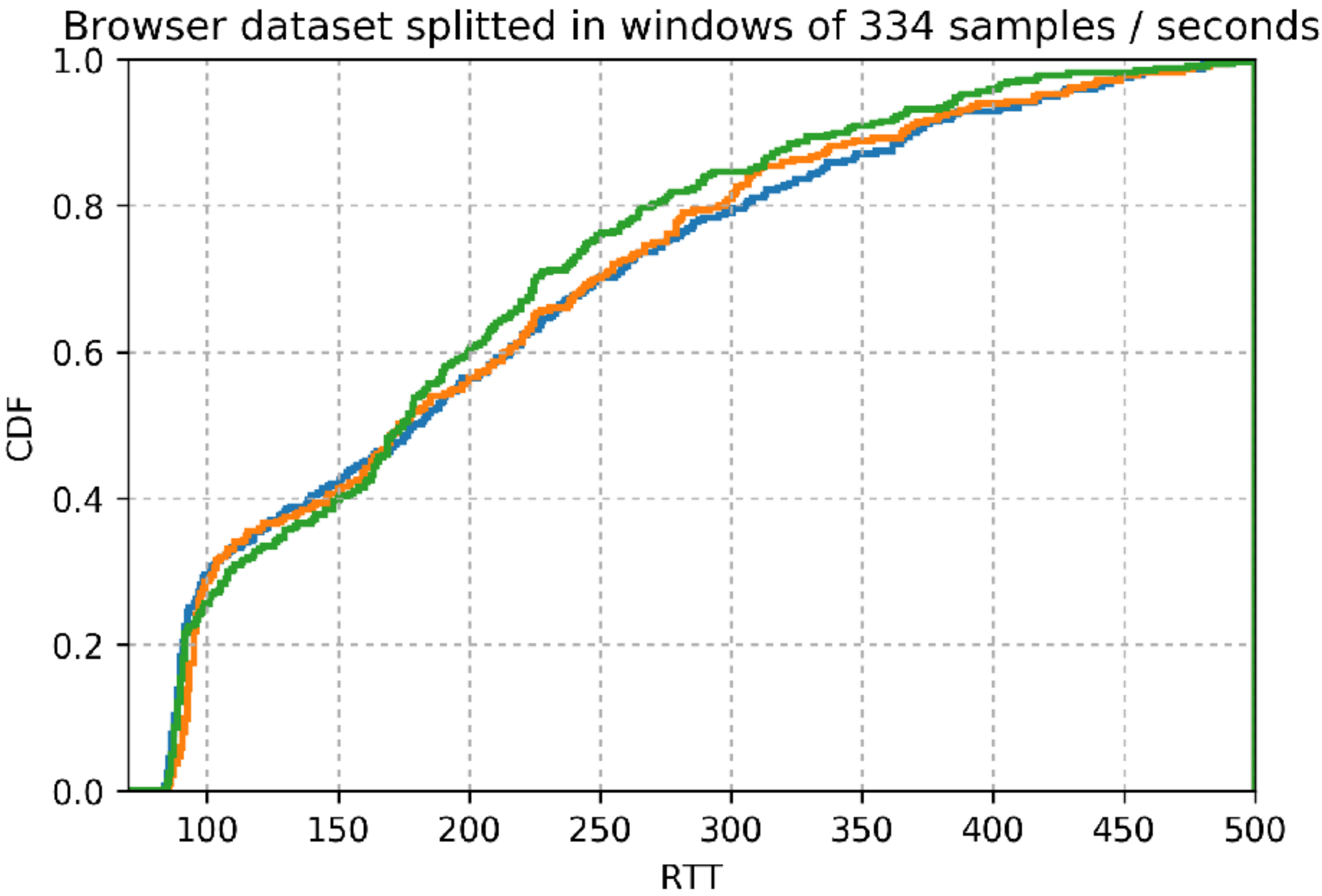
- Similar CDF profile

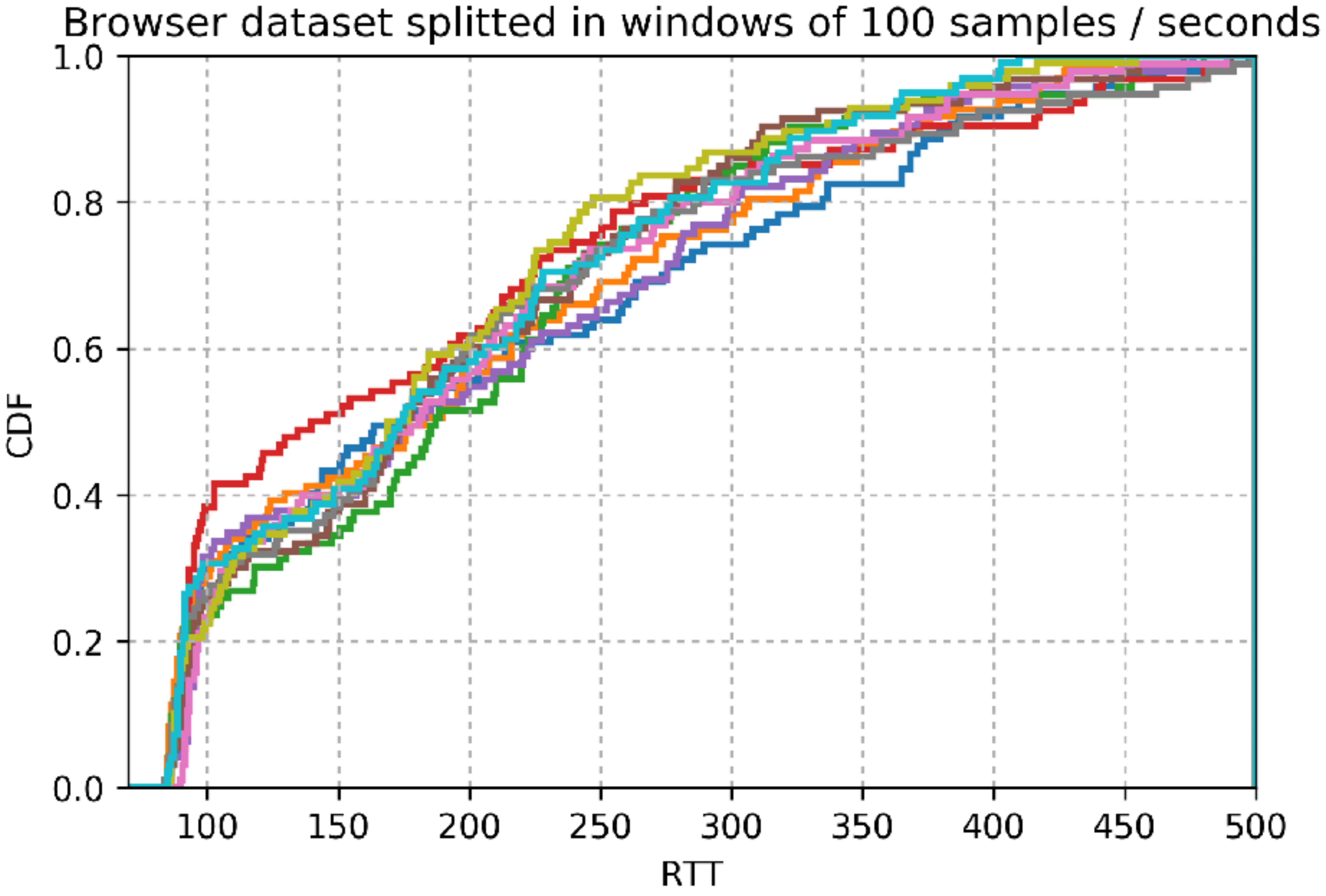
- Hypothetical constant would push green RTTs down

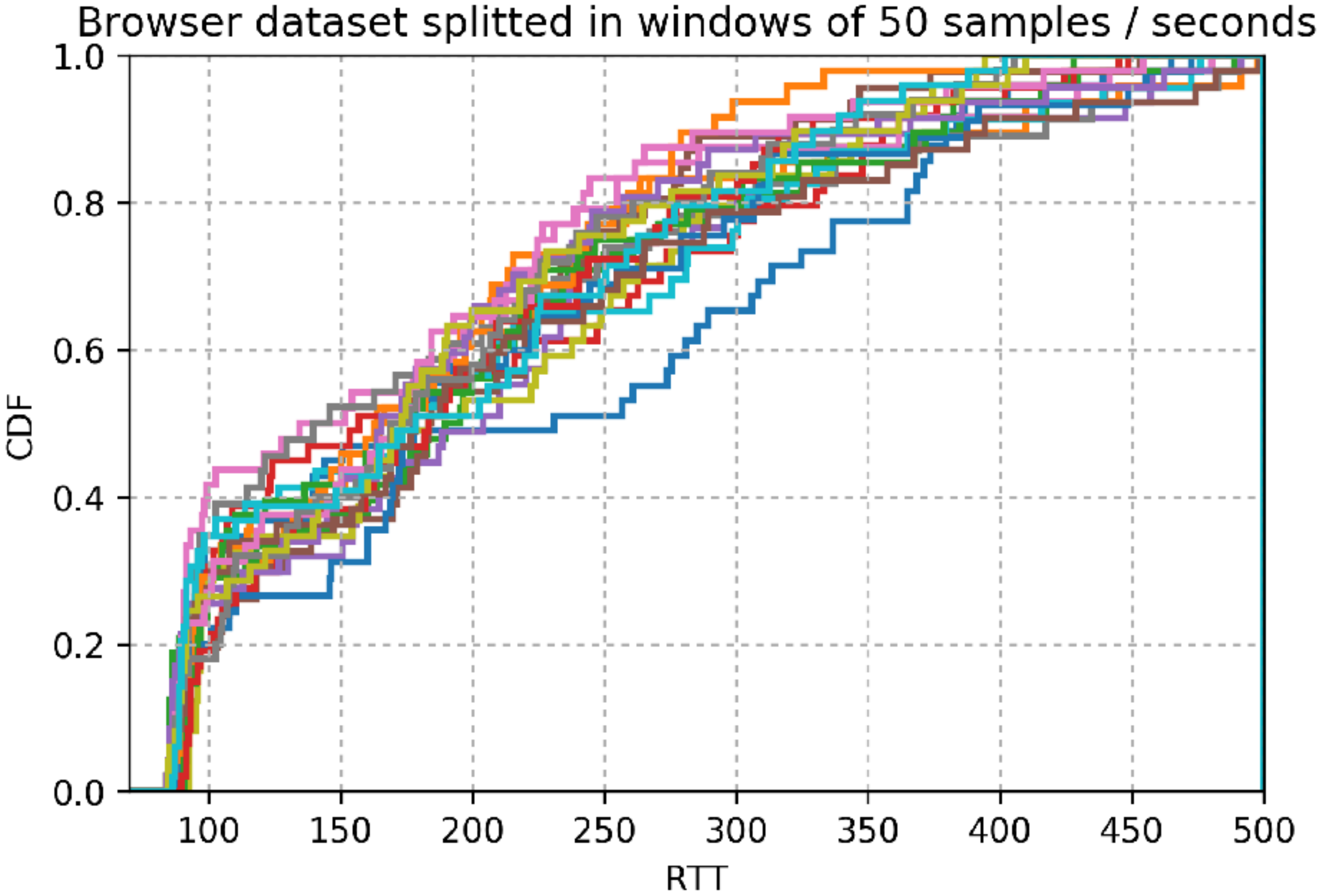
- JavaScript GETs behave well



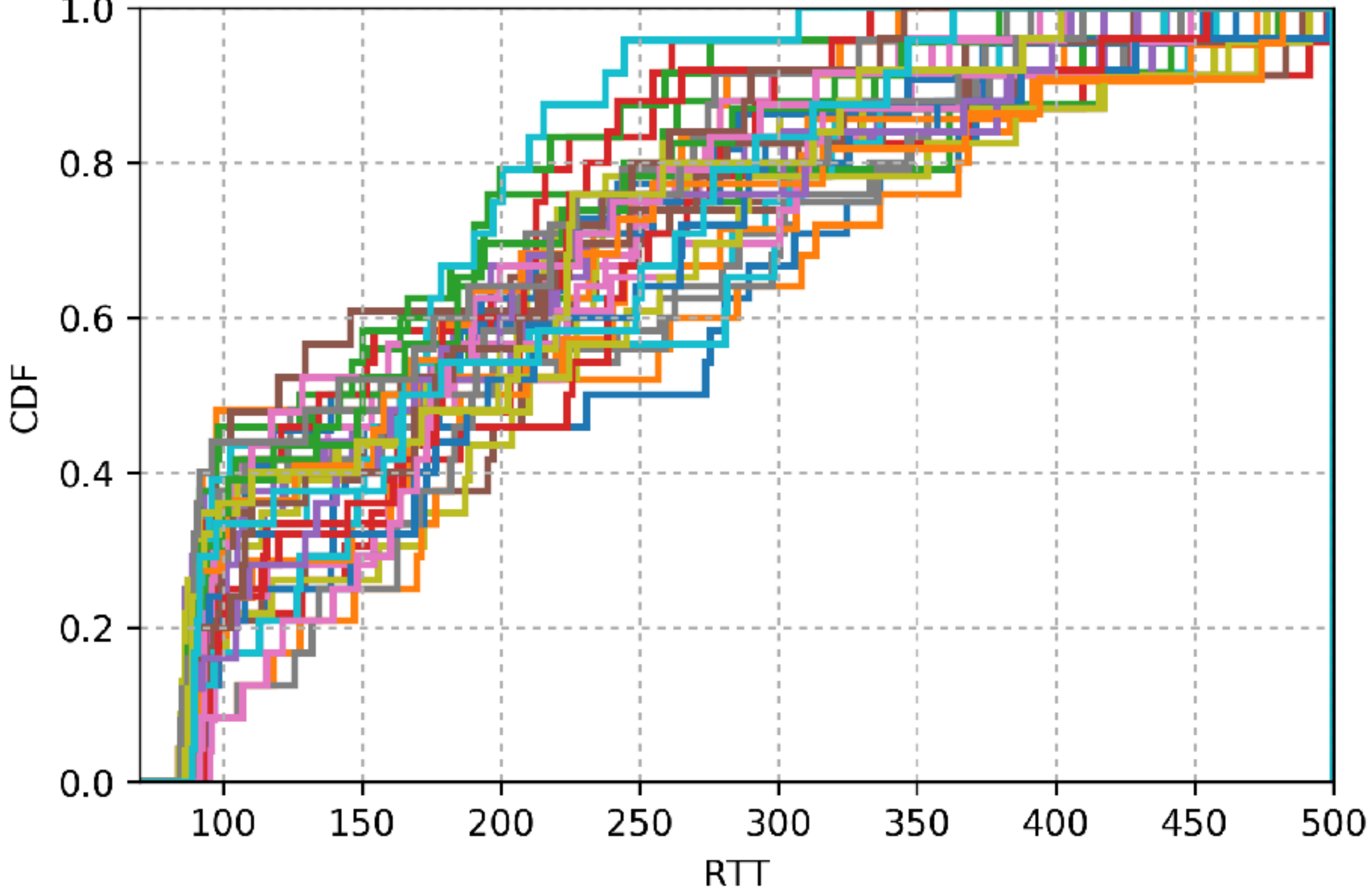

Browser dataset splitted in windows of 1000 samples / seconds
CDF
1.0
0.8
0.6
0.4
0.2
0.0
100
150
200
250
300
350
400
450
500
RTT


Browser dataset splitted in windows of 334 samples / seconds

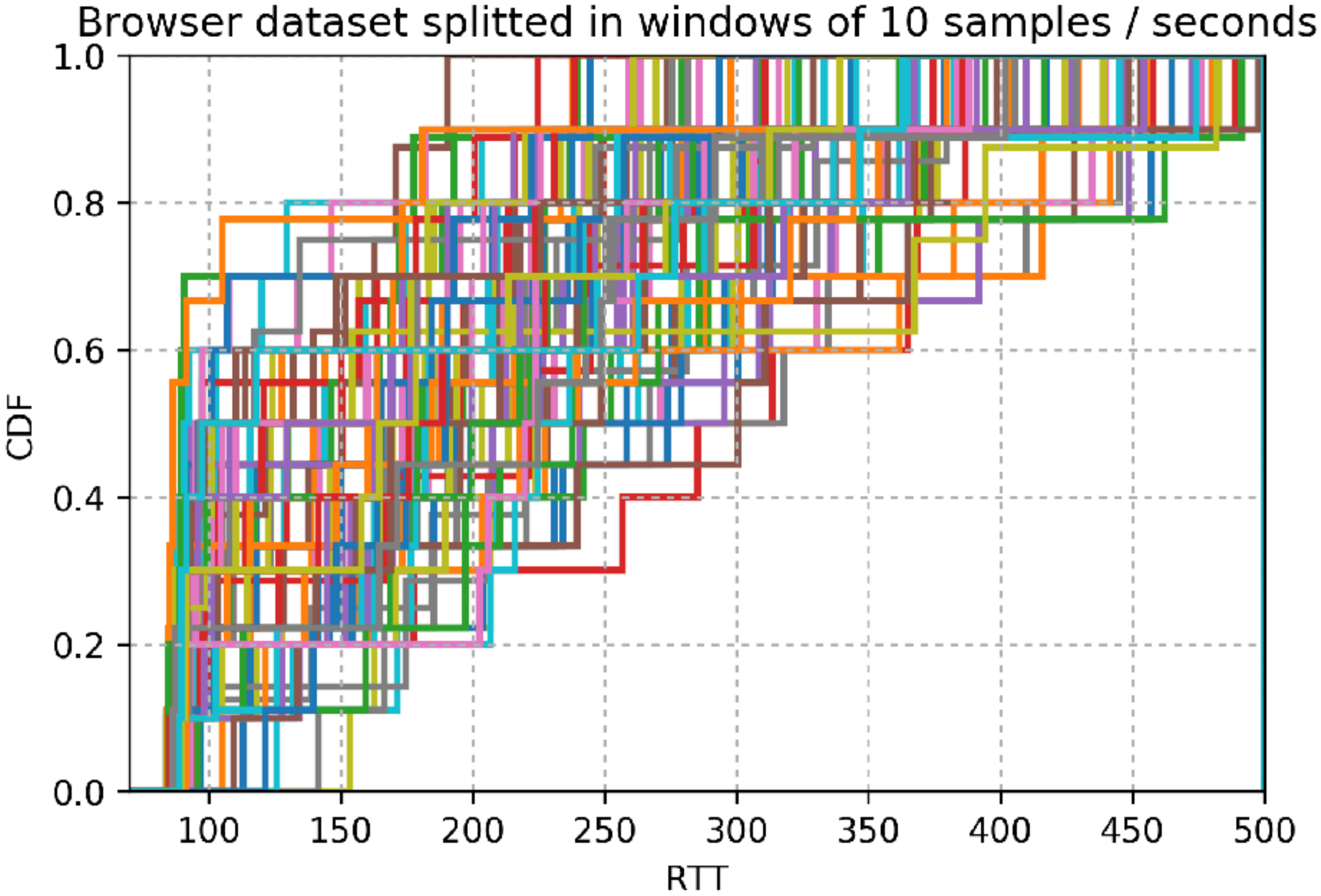

Browser dataset splitted in windows of 100 samples / seconds
CDF
RTT
0.0
0.2
0.4
0.6
0.8
1.0
100
150
200
250
300
350
400
450
500



Browser dataset splitted in windows of 50 samples / seconds
CDF
1.0
0.8
0.6
0.4
0.2
0.0
100
150
200
250
300
350
400
450
500
RTT

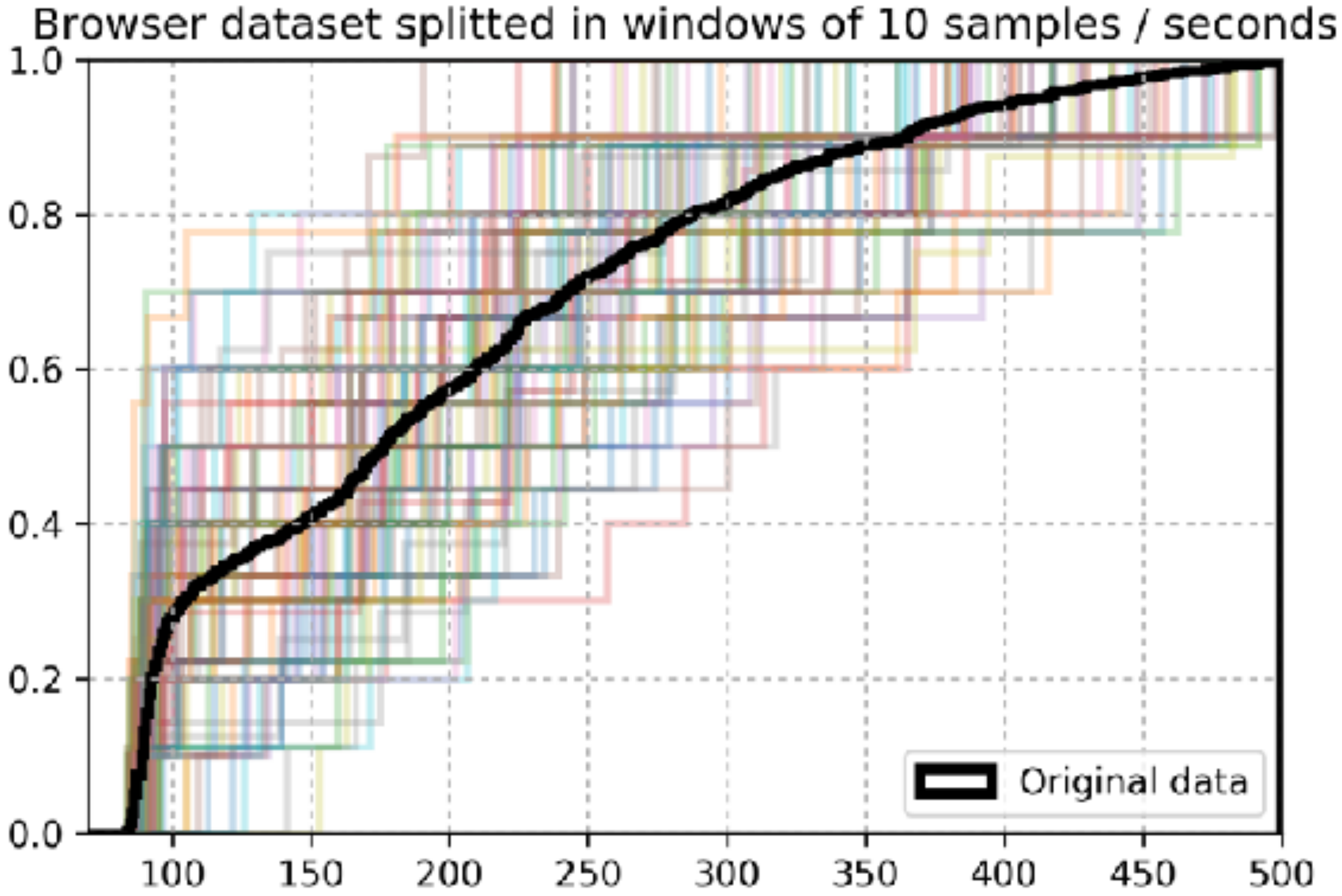


Browser dataset splitted in windows of 25 samples / seconds


Browser dataset splitted in windows of 10 samples / seconds
CDF
RTT
0.0
0.2
0.4
0.6
0.8
1.0
100
150
200
250
300
350
400
450
500

Emulating the user

- This is how the experiment's reality looks like
- How do short-lived experiments represent the **reality**?
- Split dataset in windows of 10 random samples

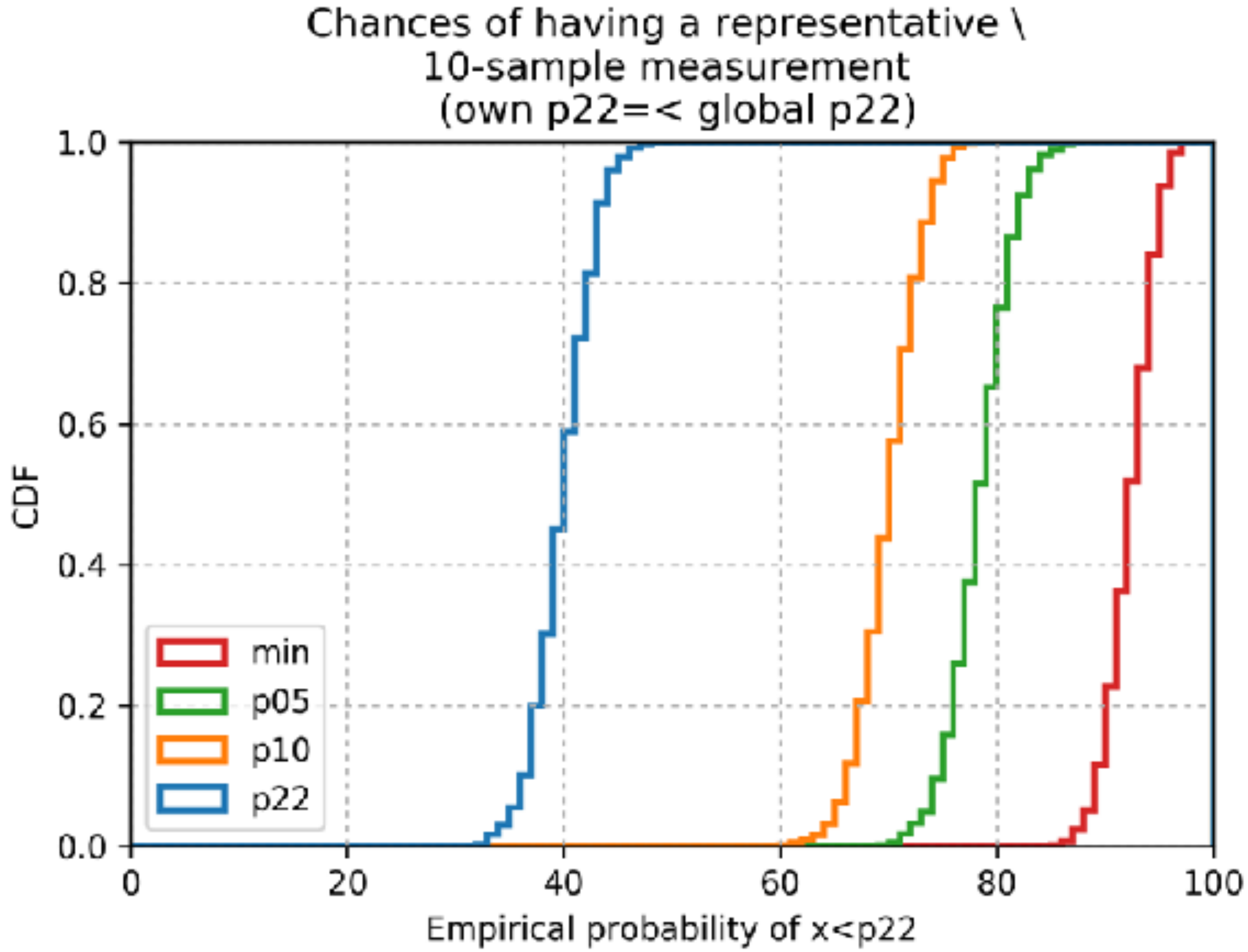


Emulating the user

- Random iterations
  run 100 experiments
  check probabilities
  do it 1000 times

- The **min** appears to be the only reliable metric

	x
p22	41.9
p10	65.8
p05	71.7
min	**90.0**



Partial Conclusions

- HTTP HEAD is *slightly* faster than GET towards RIPE Atlas Anchors…still don't know *where* (anchor/network).

- JavaScript tester appears to have a constant delay over the percentiles compared to Atlas HTTP methods, after filter.

- p22 rule of thumb for our browser-based wireless measurements

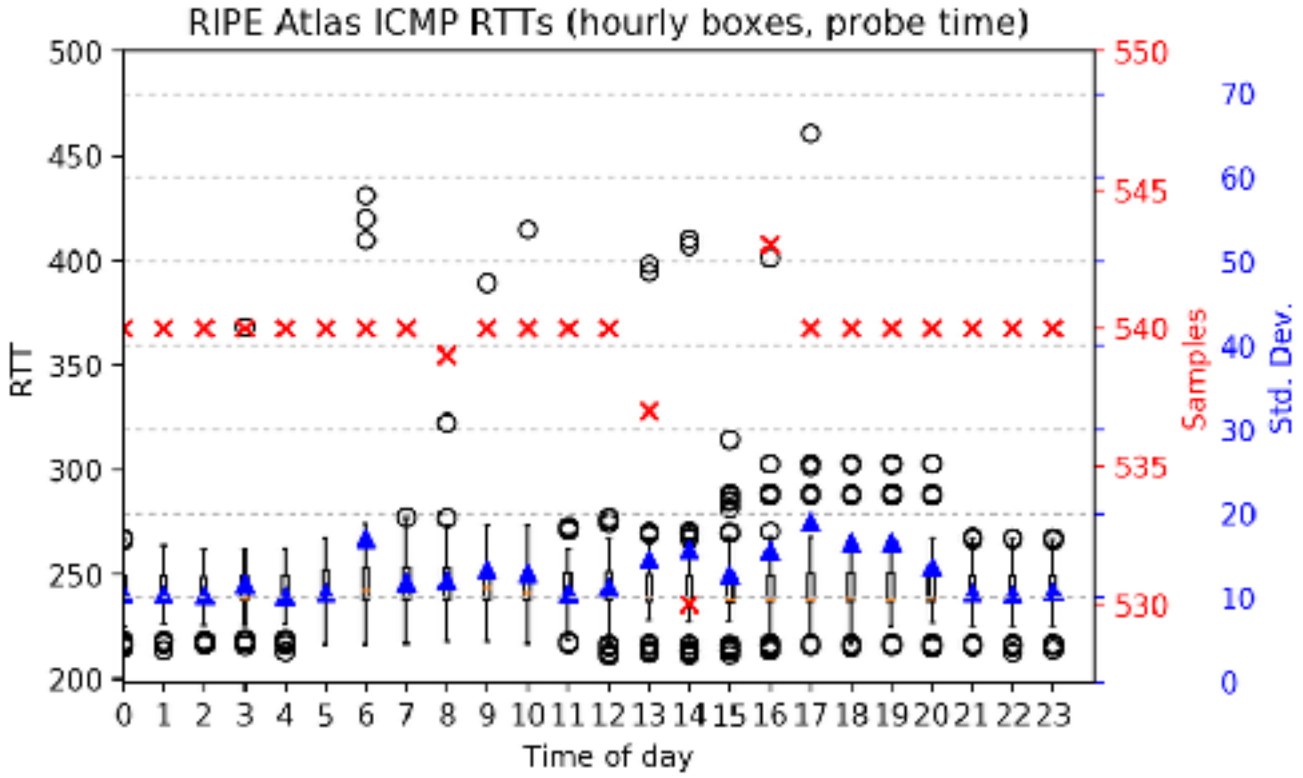
  - Keeping the usual p90 is not an option on wireless!
    …neither is IQR filtering

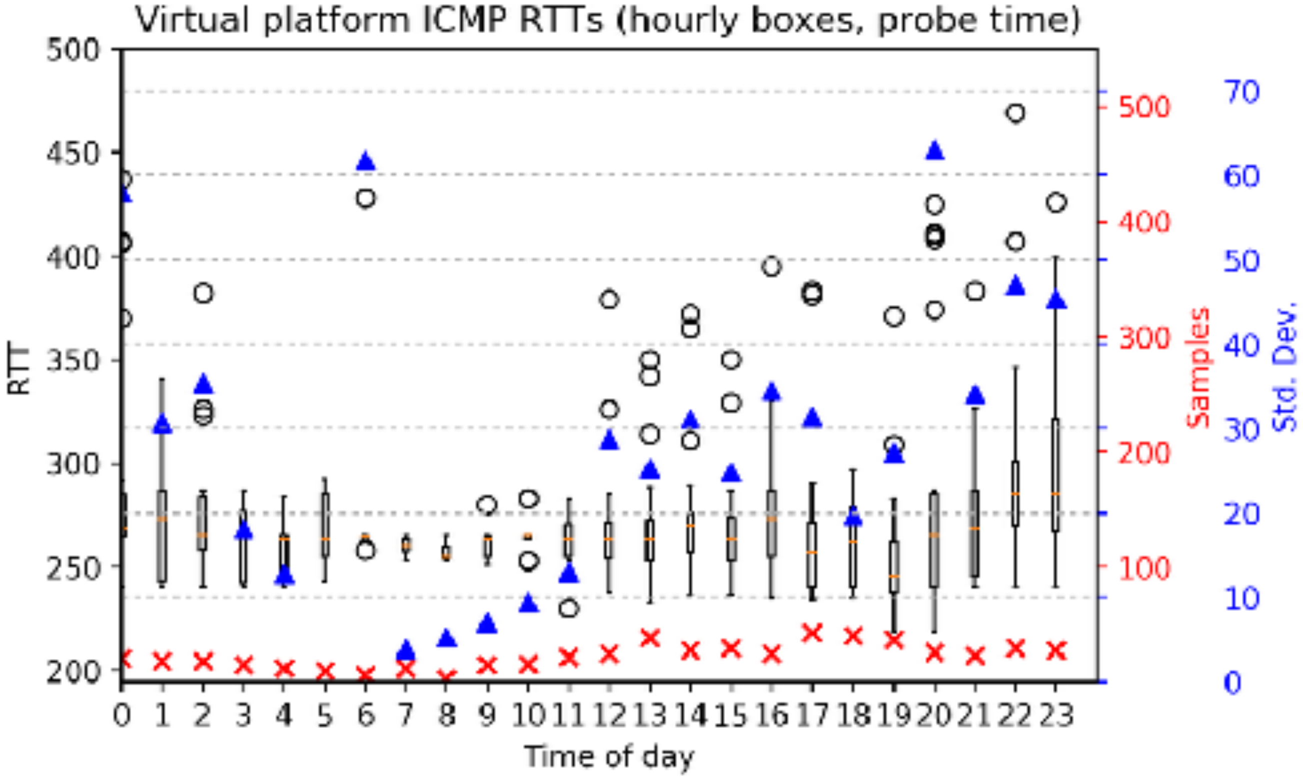
  - p22 might vary from probe to probe 😬

- Using **min** is accurate for about 90% of the time

ICMP Ping comparison

- Atlas ICMP Ping
- Virtual platform ICMP Ping
  - Probe selection, same AS origin
- 10 packets, 1 sec. apart
- Every
  - 4 minutes on Atlas
  - 10 minutes on virtual platform
- Same target IP address





ICMP Ping comparison

- Some modes found in common

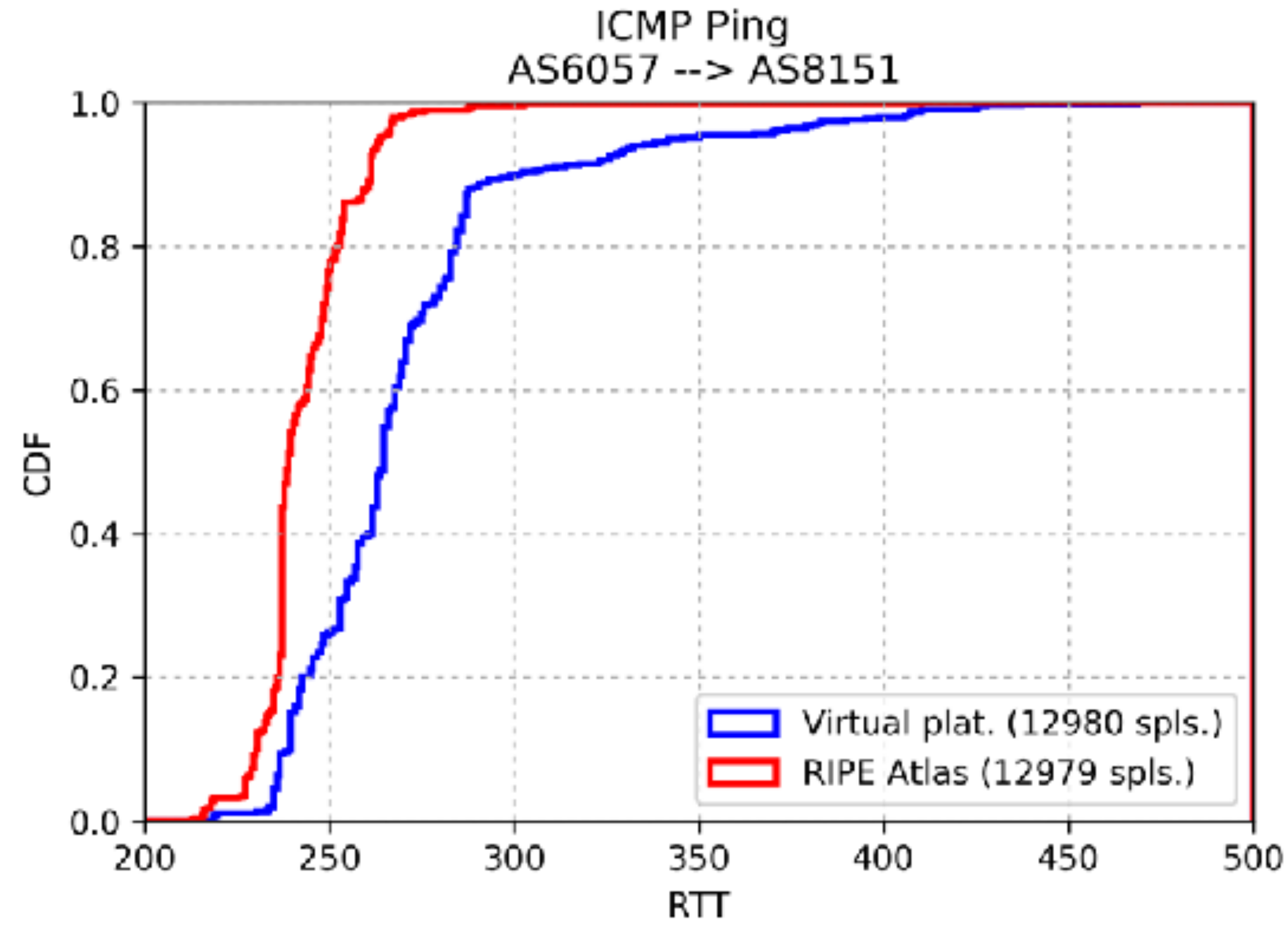
- Virtual platform has wireless probes

- Similarity with previous wireless measurements?

  - Strong mode + long tail pattern

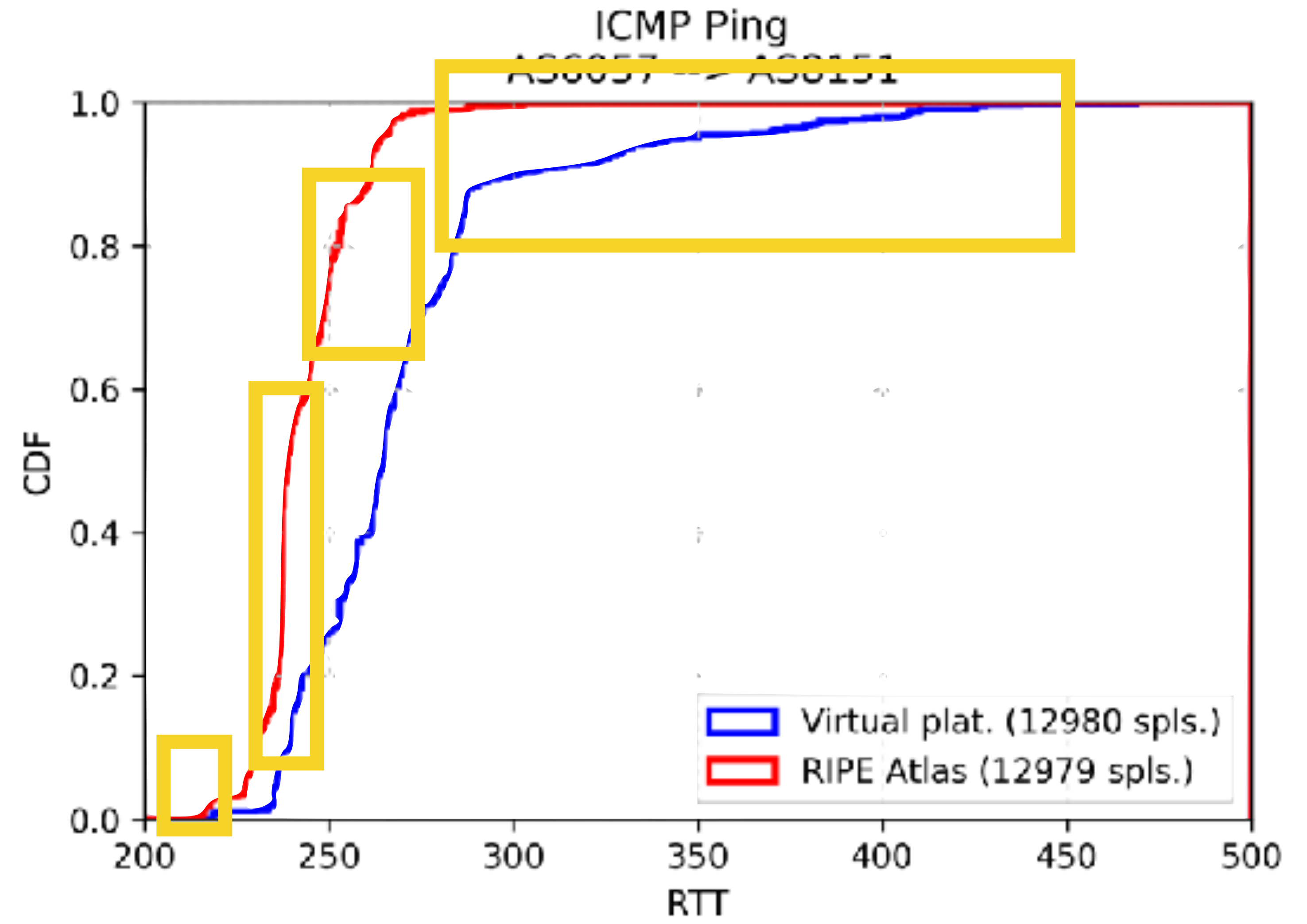
  - Mode detection with Python *peakutils* library

  - We can apply our p22 filter…



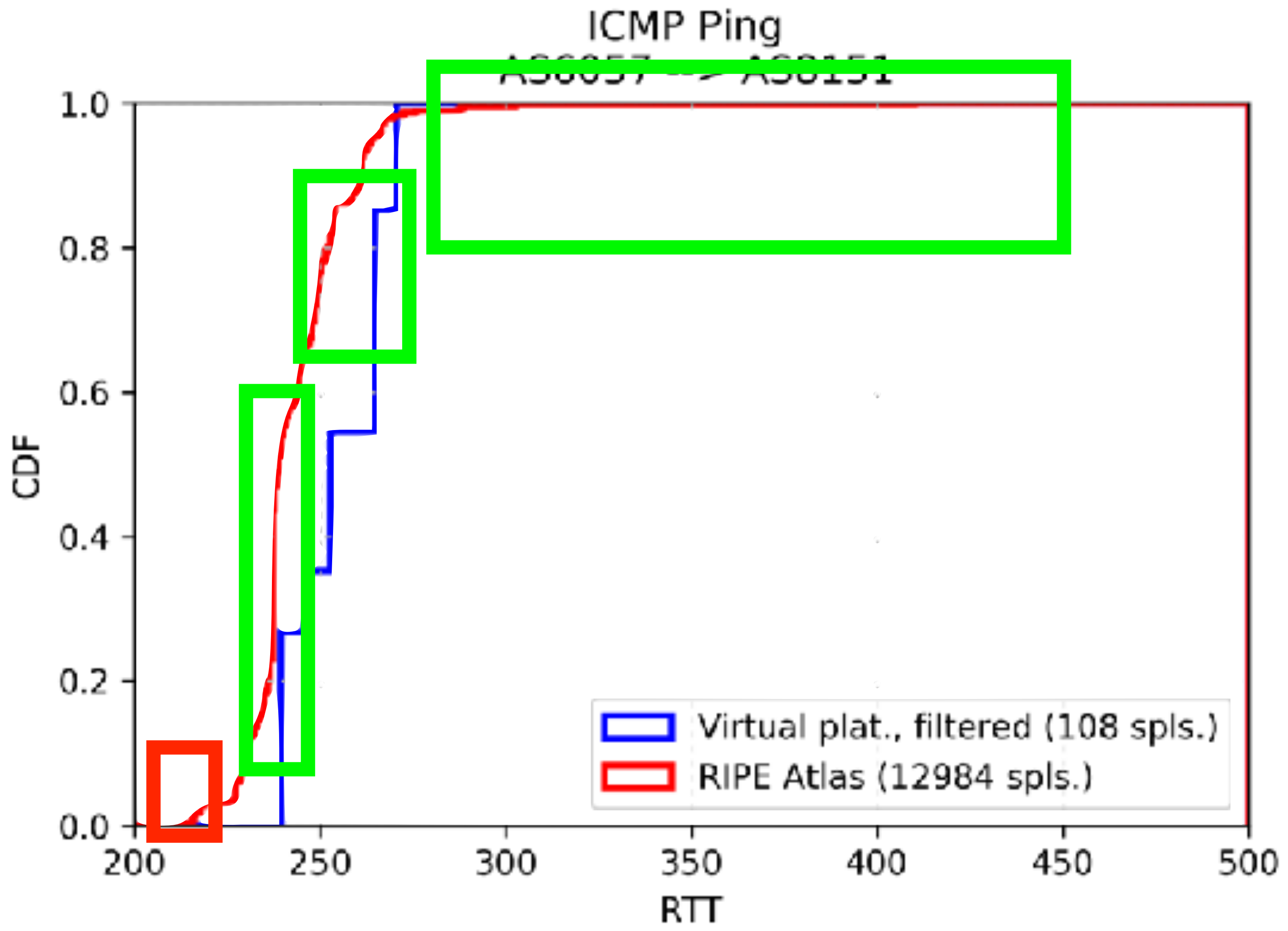
ICMP Ping comparison

- Some modes found in common
- Virtual platform has wireless probes
- Similarity with previous wireless measurements?
  - Strong mode + long tail pattern
  - Mode detection with Python *peakutils* library
  - We can apply our p22 filter…



ICMP Ping comparison

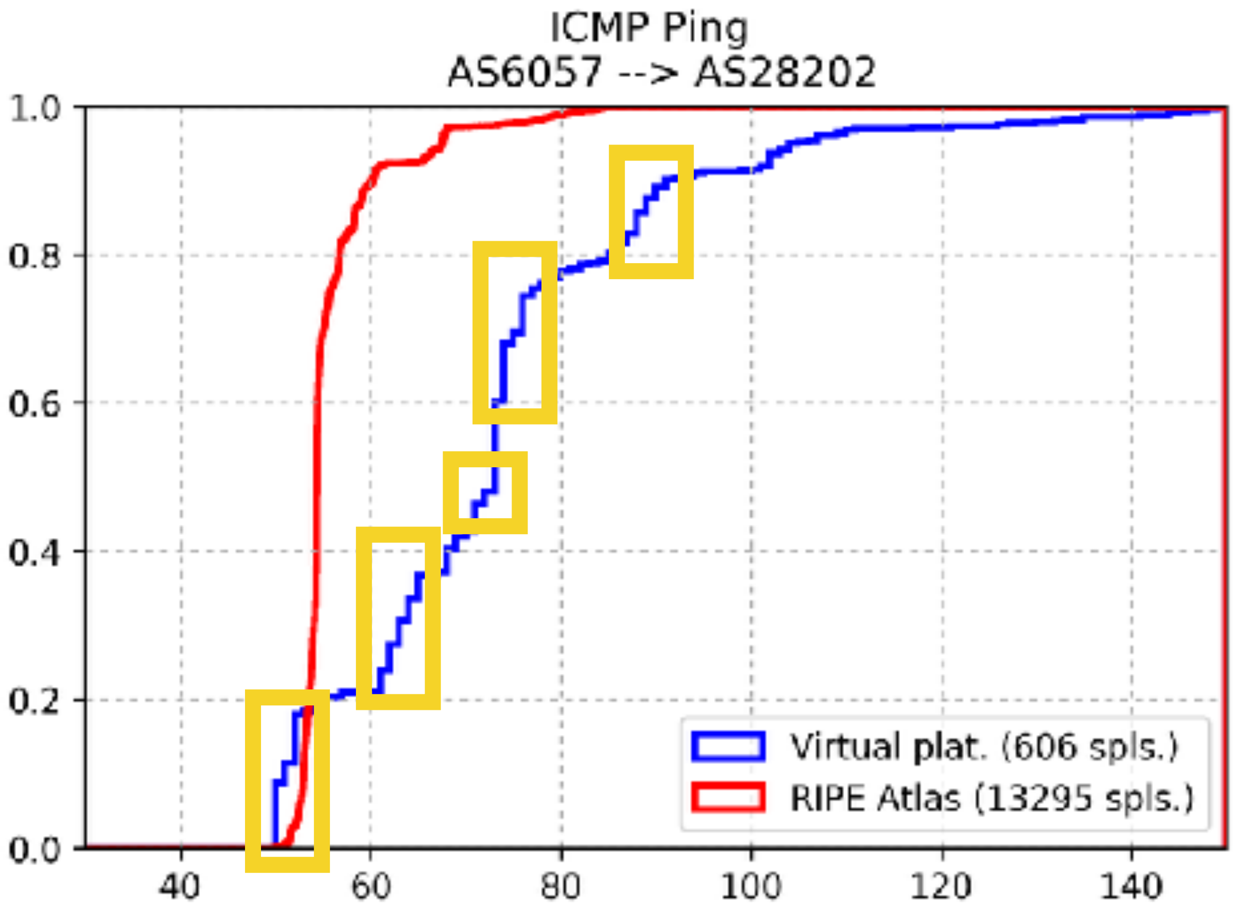
- After applying our rule-of-thumb filter
  - Aggressive
  - Keeps *local* p22 samples
  - Unfortunately the most important mode was lost
  - Curve softens over time. “Mode erosion”

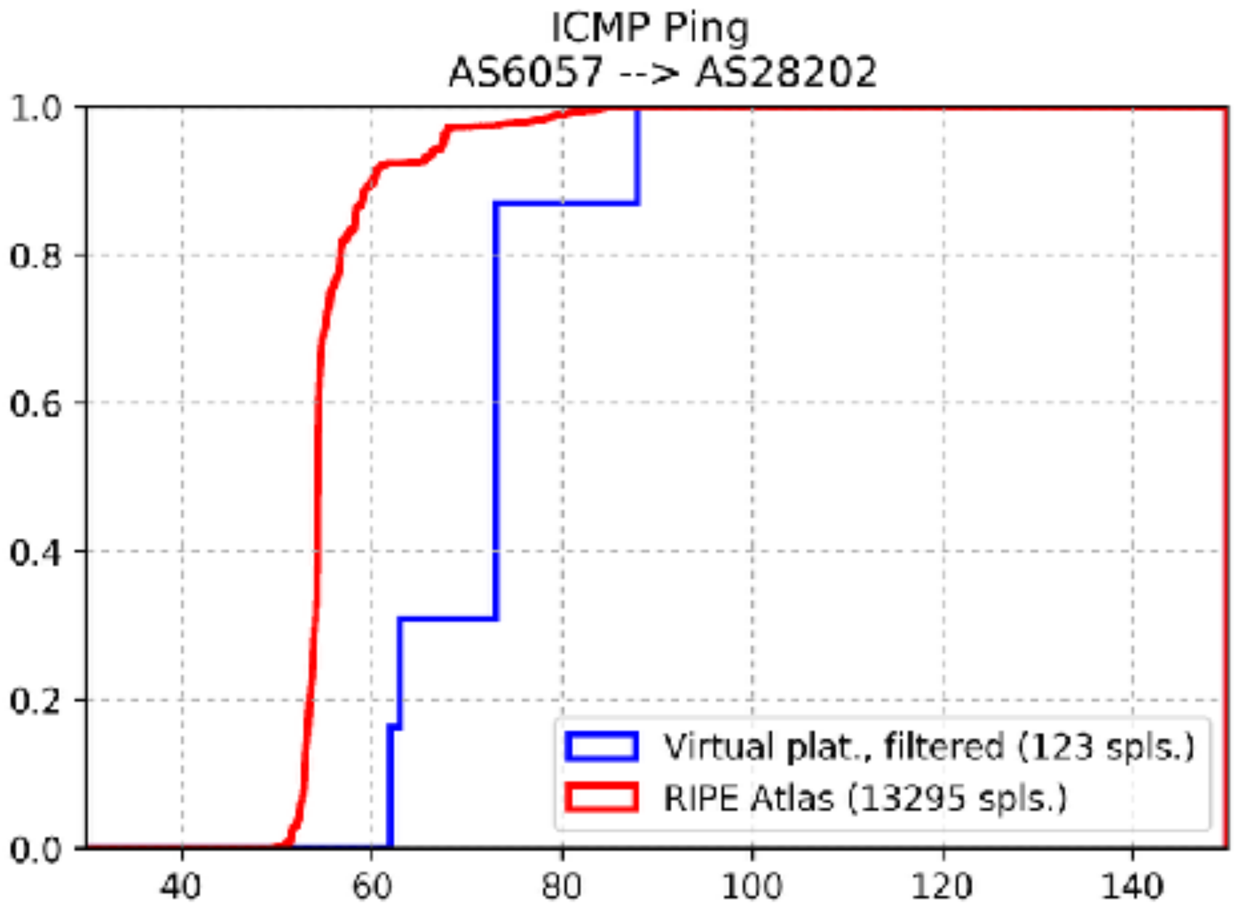


Virtual platform resolution!

ICMP Ping comparison

- Another case: before and after





Conclusions

- High noise introduced into browser-based wireless probes. If not using min:
  - Look for cutoff point. Might be as low as p22.
  - Need to do aggressive filtering
  - Still, they are comparable to RIPE Atlas
- High-level correspondence on latency modes between virtual platform and RIPE Atlas.

Future work

- Formal modeling of delays. How should they behave?

- Suggested approach

  - p22 cutoff holds true for our lab scenario: calibration for cutoff discovery, on a per-probe basis (per-measurement basis?)

- A practice we'll have to drop: IQR filtering. The useful appears to be q22 and below

Final notes

- Local browser test
  - Chrome version 61.0 on macOS 10.12.6
- Virtual platform: no v6!!

Thank you!

Questions / Comments?

lacnic

Proyecto
SIMON