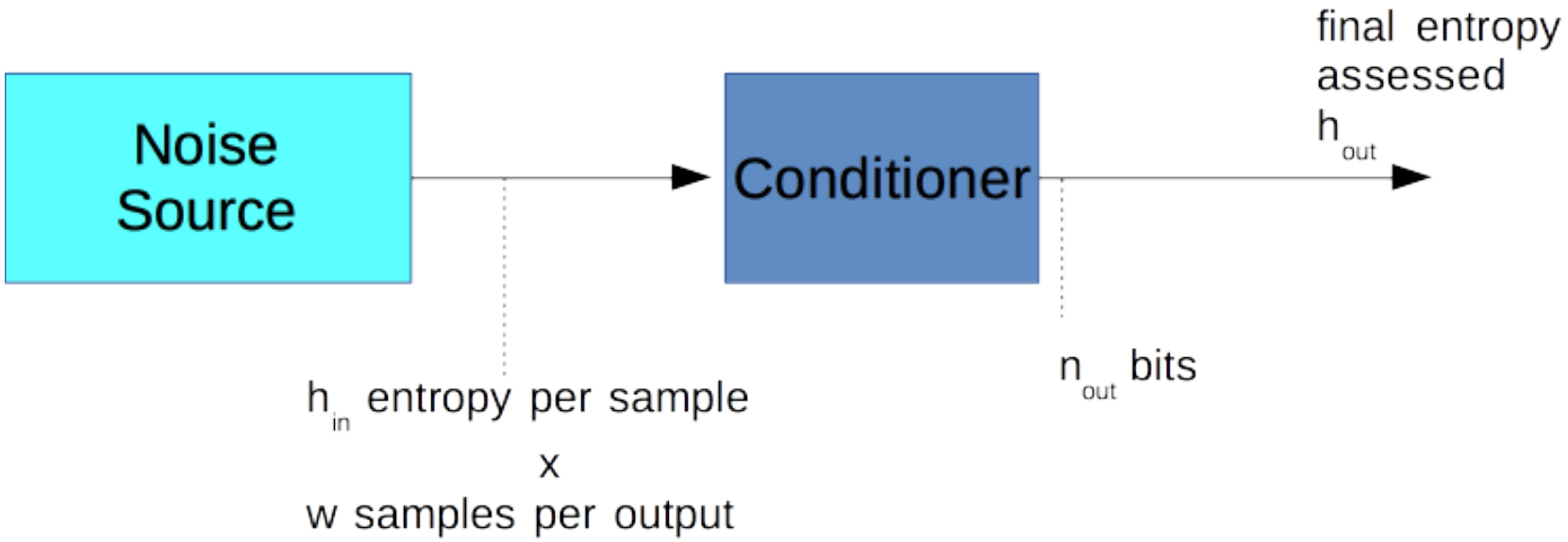
# Conditioning in 90B

John Kelsey, NIST, May 2016

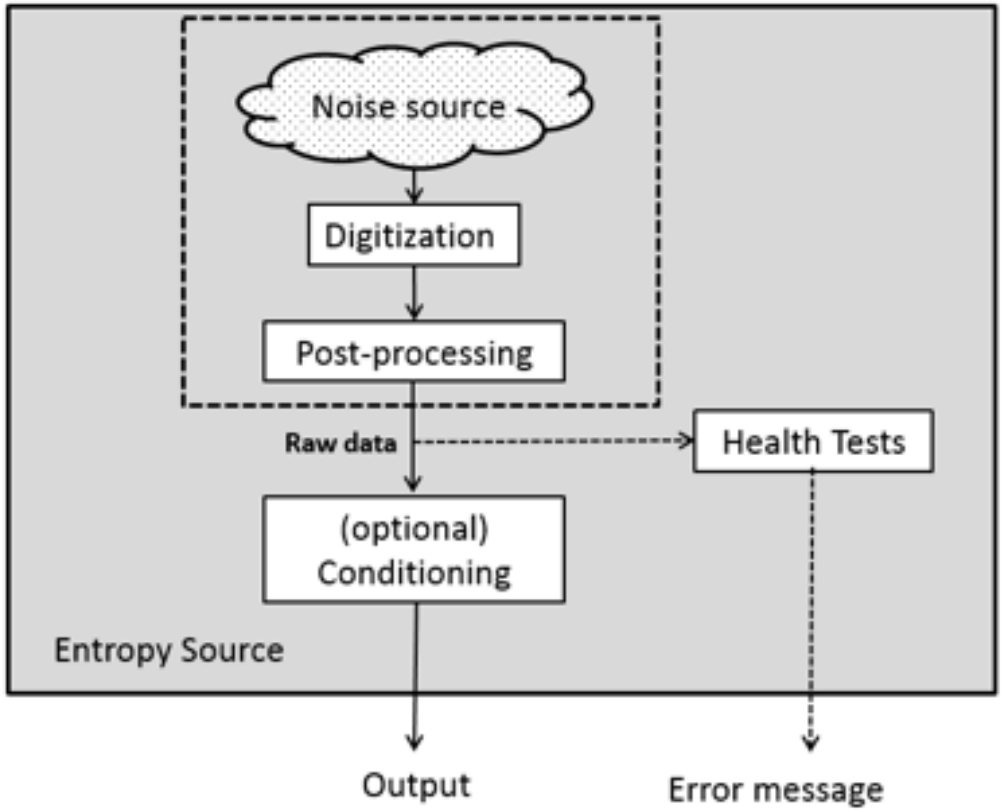


# Overview

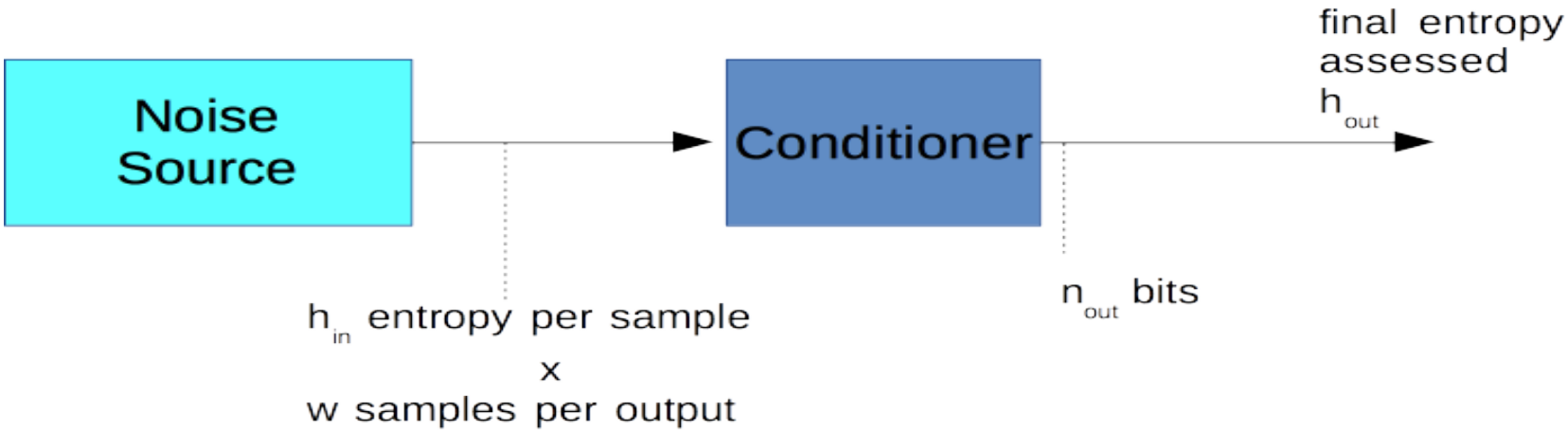
- What is Conditioning?
- Vetted and Non-Vetted Functions
- Entropy Arithmetic
- Open Issues

# Conditioning

- Optional—not all entropy sources have it.
- Improve statistics of outputs
- Typically increase entropy/output.
- Some conditioners can allow the source to produce full-entropy outputs.



# The Big Picture




- The noise source provides samples with $h_{in}$ bits of entropy/sample
- We use w samples for each conditioner output
- How much entropy do we get per output (that is what's $h_{out}$)?

*Figuring out $h_{out}$ is the whole point of this presentation.*

# How Do You Choose a Conditioning Function?

Designers can choose their own conditioning function.

- This can go wrong...
- ...so we estimate entropy of conditioned outputs.
- NEVER allowed to claim full entropy

90B specifies six “vetted” conditioning functions.

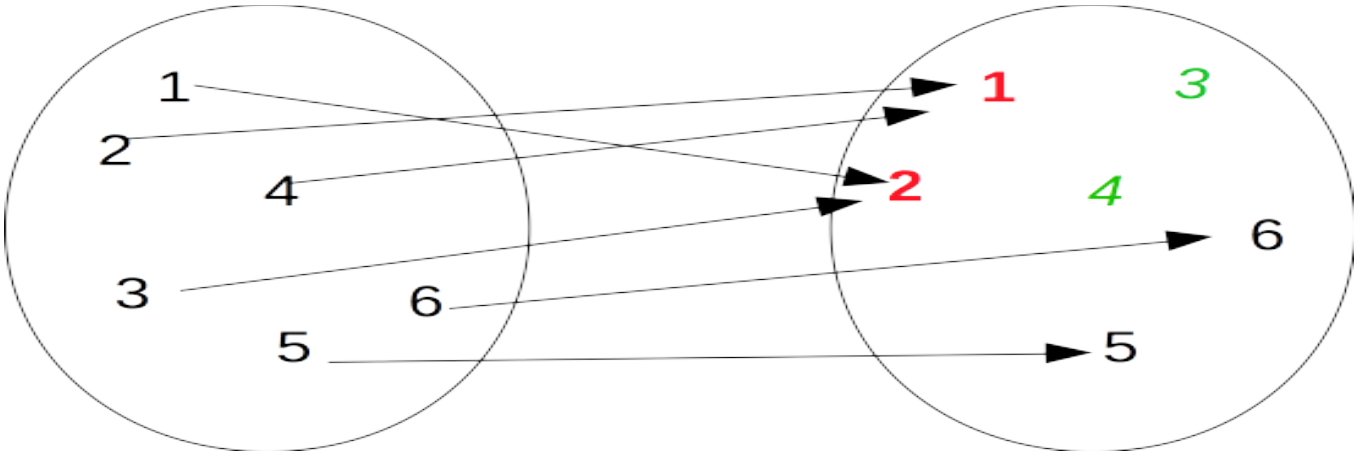
- Cryptographic mechanisms based on well-understood primitives
- Large input and output size, large internal width
- CAN claim full entropy under some circumstances

# The Vetted Functions

- HMAC using any approved hash function.
- CMAC using AES.
- CBC-MAC using AES.
- Any approved hash function.
- Hash_df as described in 90A.
- Block_cipher_df as described in 90A.

- Note: These are all wide (128 or more bits wide), cryptographically strong functions.

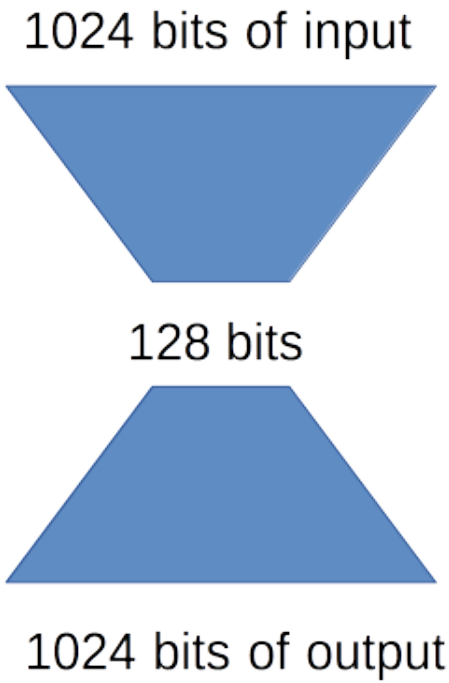
# Internal Collisions




- Suppose we have a random function F() over n bits.
- If we feed it $2^n$ different inputs, do we expect n bits of entropy out?
- NO! Because of *internal collisions*.
  - Some pairs of inputs map to the same output
  - Some outputs have no inputs mapping to them
- Internal collisions have a big impact on how we do entropy accounting!

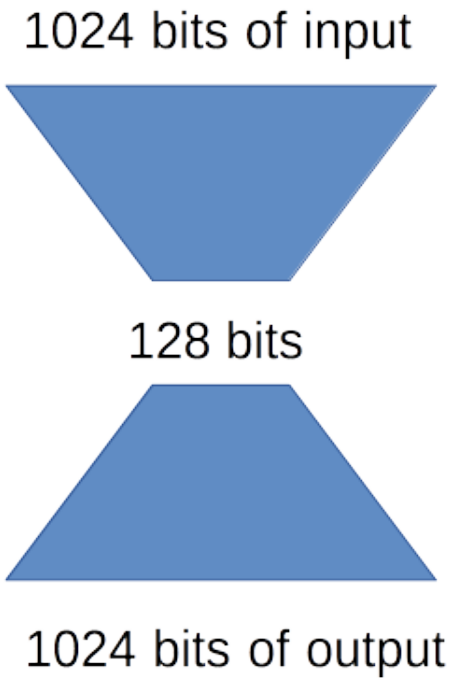
# Internal Width of a Function

- Imagine  function that:
  - Takes a 1024 bit input
  - Maps it down to a 128-bit internal state
  - Generates a new 1024-bit output from that state
- It's obvious that this function can't get more than 128 bits of entropy into its output.
- This is the idea behind *internal width* (q) of a function
- I  this case,  = 128
-  is the "narrow pipe" through which all entropy must pass.

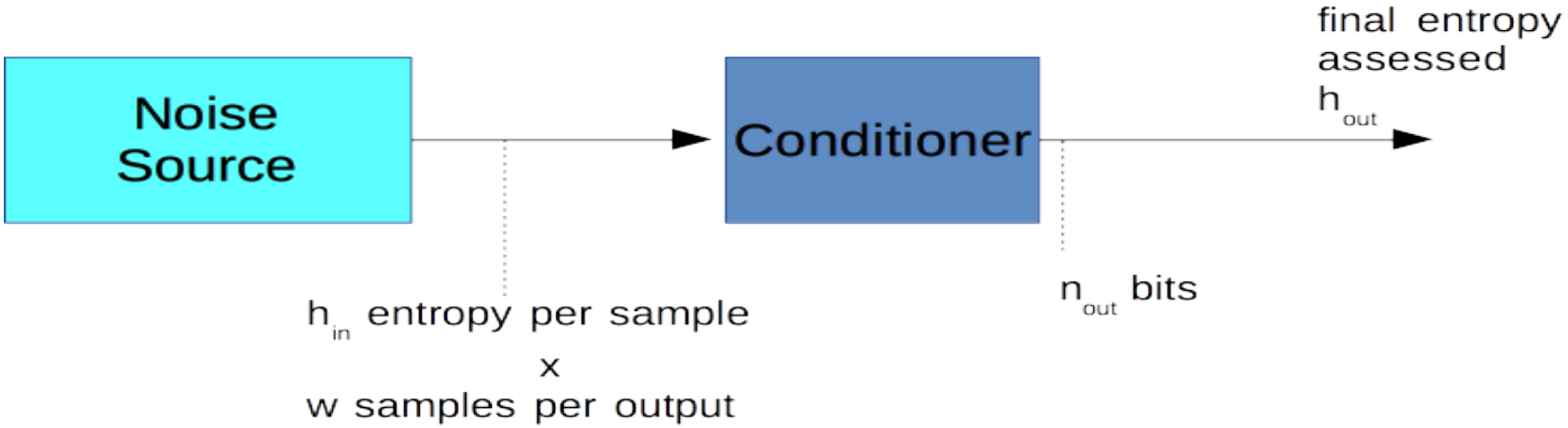


# Relevance of Internal Width

- No matter how much entropy goes into the input of this function, no more than 128 bits can ever come out...
- ...because the output is entirely function of those 128 bits of internal state.
- Our formulas for entropy accounting consider the **minimum** of output and internal width.
- Internal collisions apply just as much to internal width as to output size.

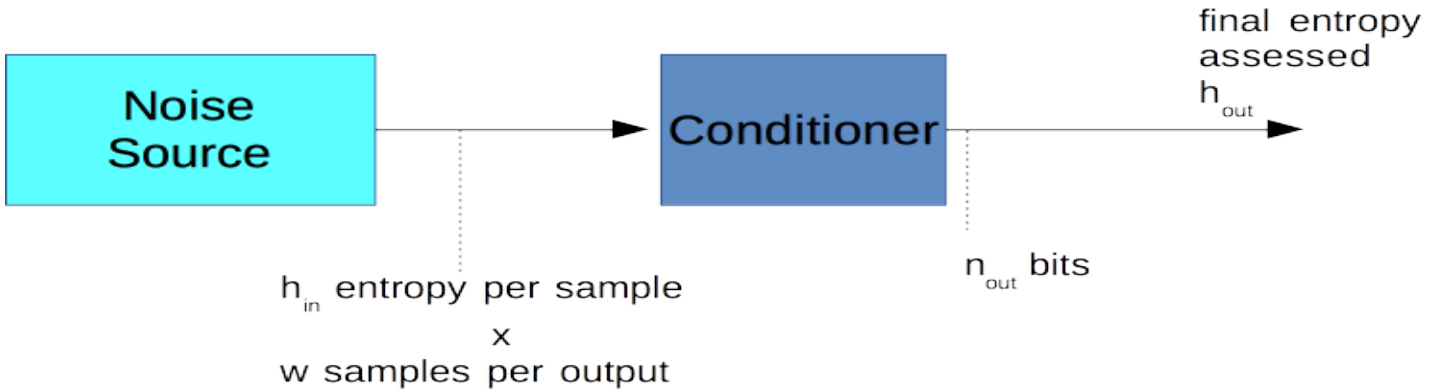


# Entropy Accounting



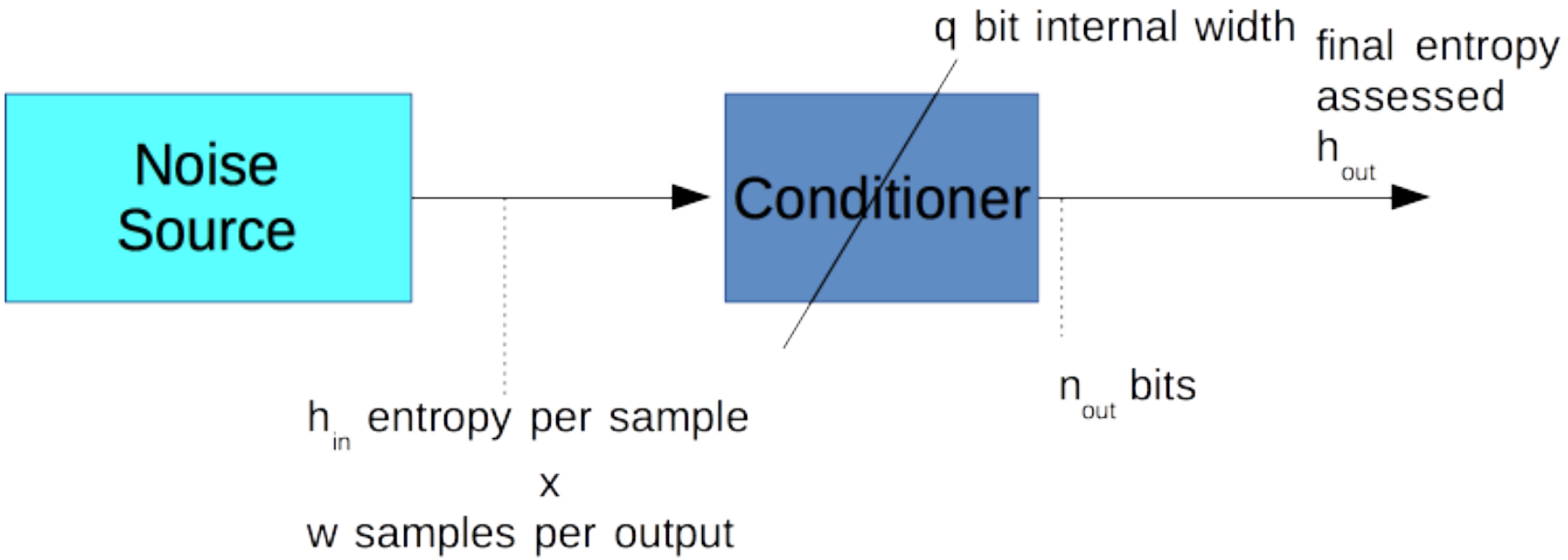

- How do we determine how much entropy we should assess for the output of the conditioner?
  - That is, how do we compute $h_{out}$?
- That's what entropy accounting is all about!

# Entropy Accounting (2)




- Conditioned outputs can't possibly have *more* entropy than their inputs.
  - That is, $h_{out} < h_{in} * w$
- They *can* have less:
  - Internal collisions, bad choice of conditioning function
- We use a couple of fairly simple equations to more-or-less capture this

# Entropy Accounting with Vetted Functions




$$h_{out} = \begin{cases} \min(w \times h_{in}, \mathbf{0.85} n_{out}, \mathbf{0.85} q), & \text{if } w \times h_{in} < 2\min(n_{out}, q) \\ \min(n_{out}, q), & \text{if } w \times h_{in} \geq 2\min(n_{out}, q) \end{cases}$$

# Entropy Accounting with Vetted Functions (2)

- Variables:
  - $h_{in}$ = entropy/sample from noise source
  - w = noise source samples per conditioned output
  - q = internal width of conditioning function
  - $n_{out}$ = output size of conditioning function in bits
  - **$h_{out}$ = entropy per conditioned output (what we are trying to find)**

$$h_{out} = \begin{cases} \min(w \times h_{in}, \mathbf{0.85} n_{out}, \mathbf{0.85} q), & \text{if } w \times h_{in} < 2\min(n_{out}, q) \\ \min(n_{out}, q), & \text{if } w \times h_{in} \geq 2\min(n_{out}, q) \end{cases}$$

# Why Does This Make Sense?

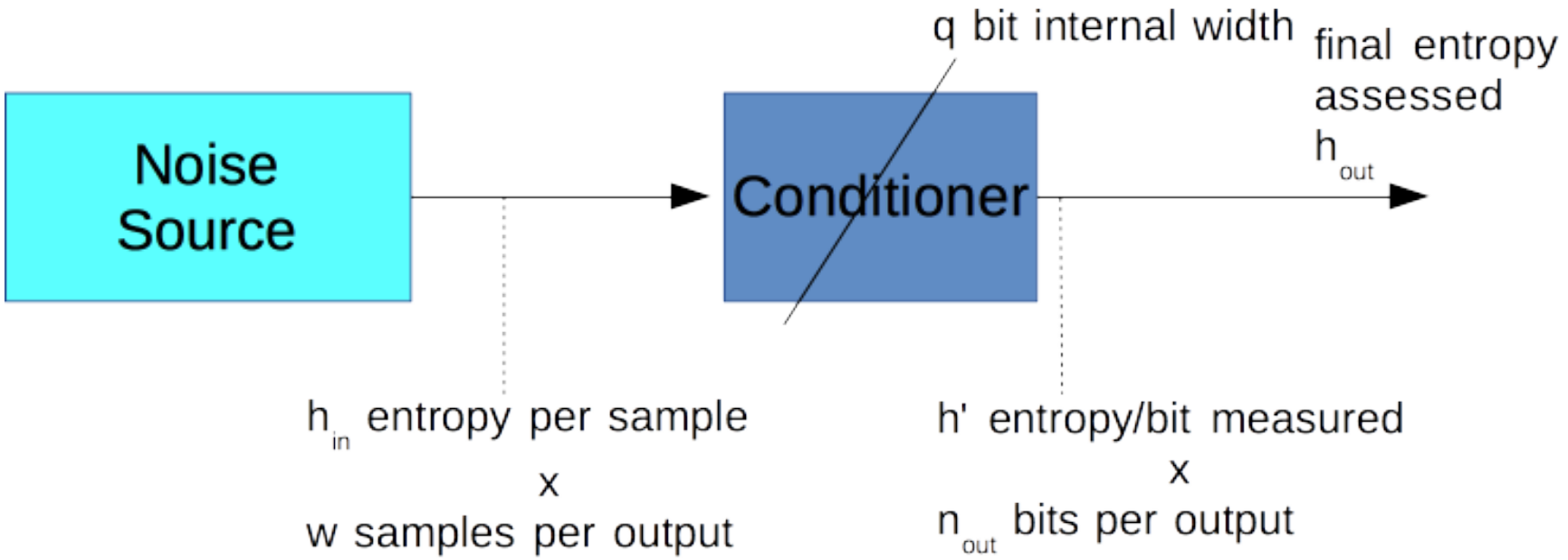
$$h_{out} = \begin{cases} \min(\boldsymbol{w \times h_{in}}, 0.85 n_{out}, 0.85 q), & \text{if } w \times h_{in} < 2\min(n_{out}, q) \\ \min(n_{out}, q), & \text{if } w \times h_{in} \geq 2\min(n_{out}, q) \end{cases}$$

- **We never get more entropy out that was put in**:
  - $h_{out}$ can never be greater than $w \times h_{in}$
- *As we get closer to full entropy, we lose a little to internal collisions*:
  - Until $w \times h_{in} \geq 2\min(n_{out}, q)$ we get only $\mathbf{0.85} n_{out}$ entropy assessed.
- Put twice as much entropy in as we take out in bits to get full entropy:
  - $h_{out} = \min(n_{out}, q)$, if $w \times h_{in} \geq 2\min(n_{out}, q)$

# Non-Vetted Conditioning Functions

- The designer can choose any conditioning function he likes.
- I this case, we must also test the conditioned outputs to make sure the function hasn't catastrophically thrown away entropy.
- Collect 1,000,000 sequential conditioned outputs.
- Use the entropy estimation methods (without restart tests) used for the noise source on the conditioned outputs.
- **Let h' = the estimate from the conditioned outputs per bit.**
- Note: designer must specify q in documentation; labs will verify that by inspection.

# Entropy Accounting with Non-Vetted Functions




$$h_{out} = \min(w \times h_{in}, 0.85 n_{out}, 0.85q, h' \times n_{out}).$$

# Entropy Accounting with Non-Vetted Functions

- Variables:
  - $h_{in}$ = entropy/sample from noise source
  - w = noise source samples per conditioned output
  - q = internal width of conditioning function
  - $n_{out}$ = output size of conditioning function in bits
  - h' = measured entropy/bit of conditioned outputs
  - **$h_{out}$ = entropy per conditioned output (what we are trying to find)**

$$h_{out} = \min(w \times h_{in}, 0.85 n_{out}, 0.85 q, h' \times n_{out}).$$

# Why Does This Make Sense?

$$h_{out} = \min(\boldsymbol{w \times h_{in}}, 0.85 n_{out}, 0.85 q, h' \times n_{out}).$$

- **We never get more entropy out that was put in**:
  - $h_{out}$ can never be greater than $w \times h_{in}$
- *As we get closer to full entropy, we lose a little to internal collisions*:
  - Until $w \times h_{in} \geq 2 \min(n_{out}, q)$ we get only $\mathbf{0.85} n_{out}$ entropy assessed.
- We can't claim more entropy than we saw when evaluating the conditioned outputs!

*Note: There is no way to claim full entropy when using a non-vetted function.*

# What's With the 0.85?

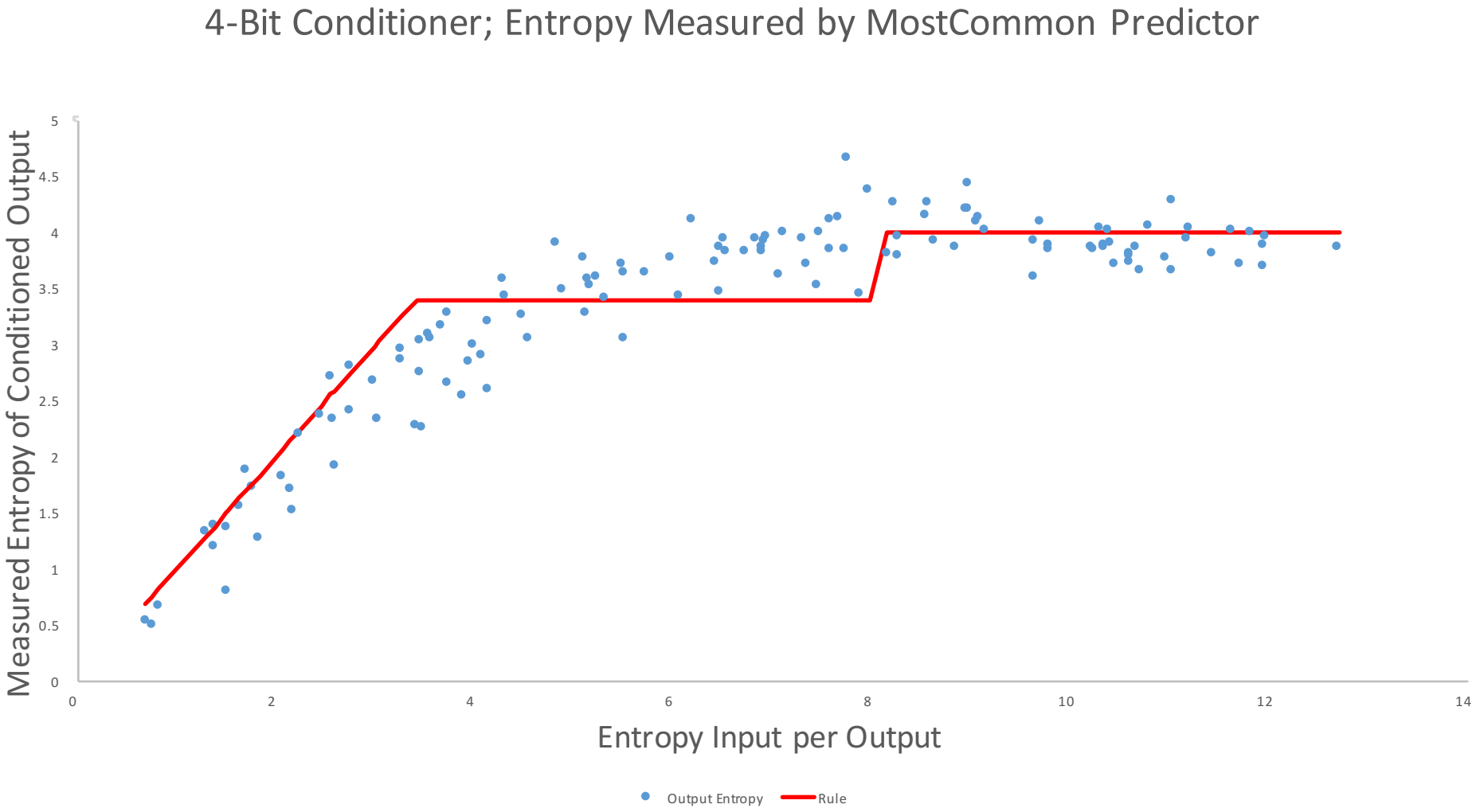
- Internal collisions mean that when $h_{in} = n_{out}$ we do not get full entropy out.
- For smaller functions, this effect is more important (and more variable!)
- Choosing a single constant gives a pretty reasonable, conservative approximation to the reality

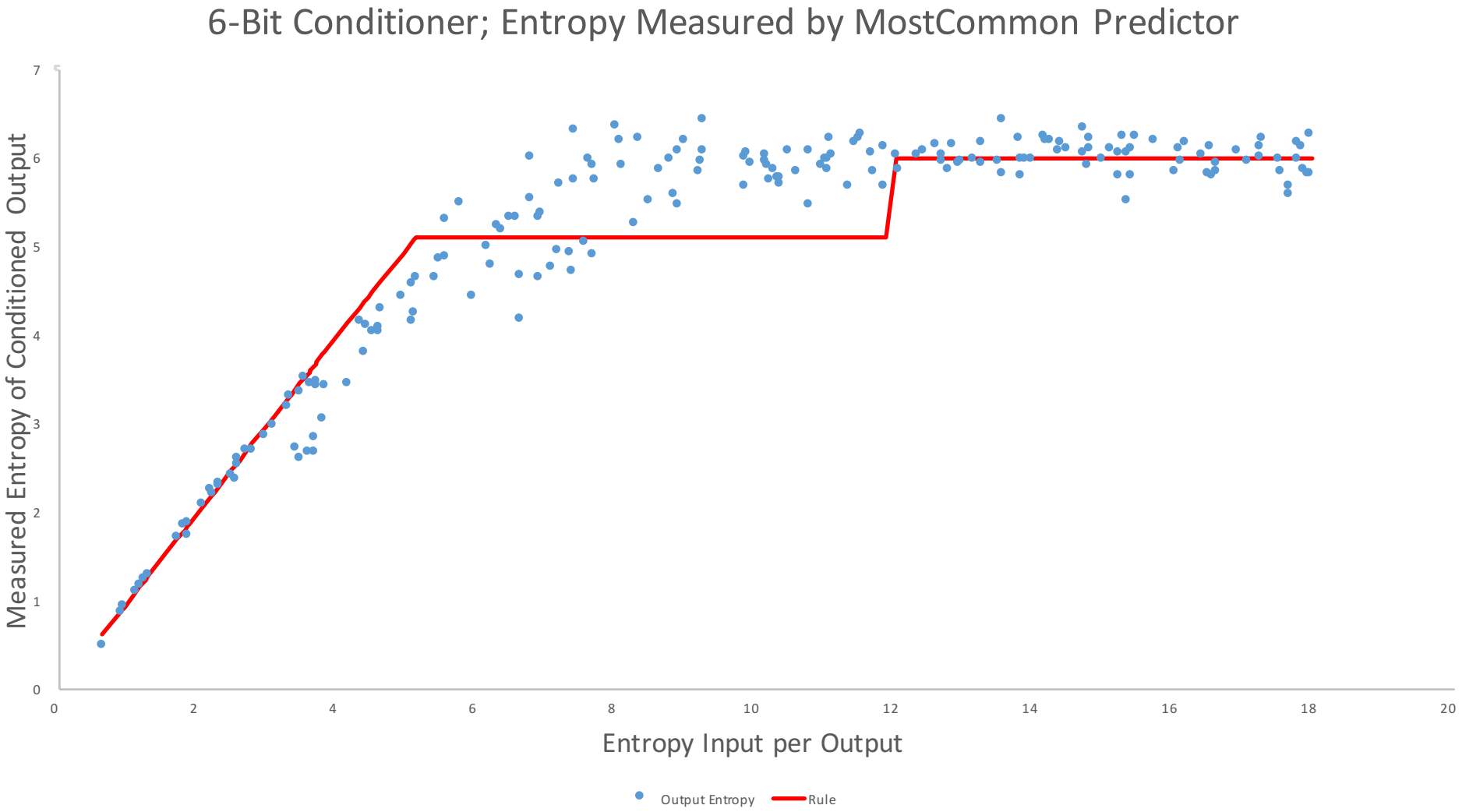
# So, How Well Does This Describe Reality?

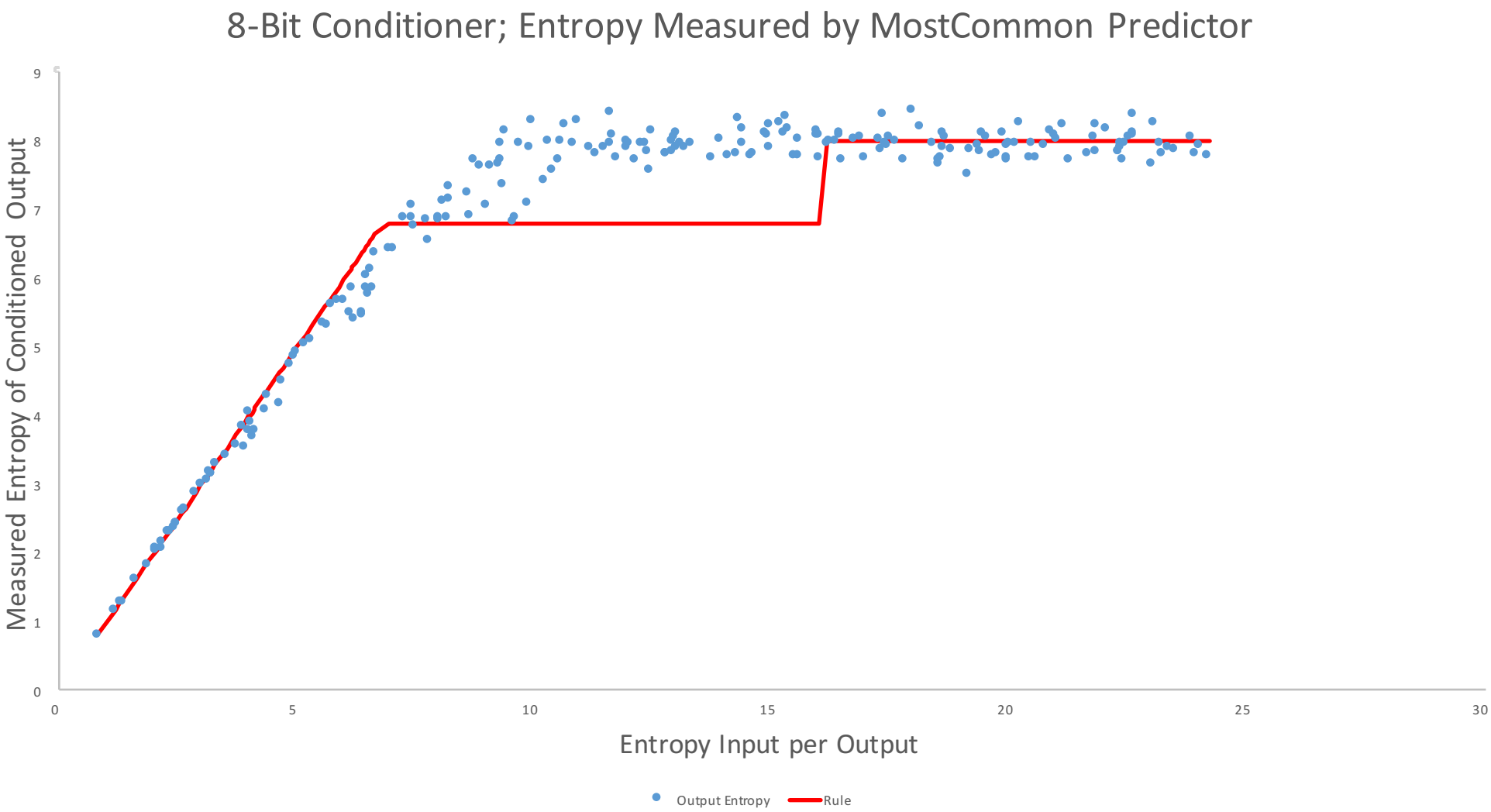
- I ran several large simulations to test how well the formulas worked in practice, using small enough cases to be manageable.
- Conditioning function = SHA1-based MAC.
- Sources: simulated iid sources: near-uniform, uniform, and normal
- Entropy/output was measured using MostCommon predictor
  - Note: this can get overestimates and underestimates by chance
  - Experimental values are expected to cluster around correct values

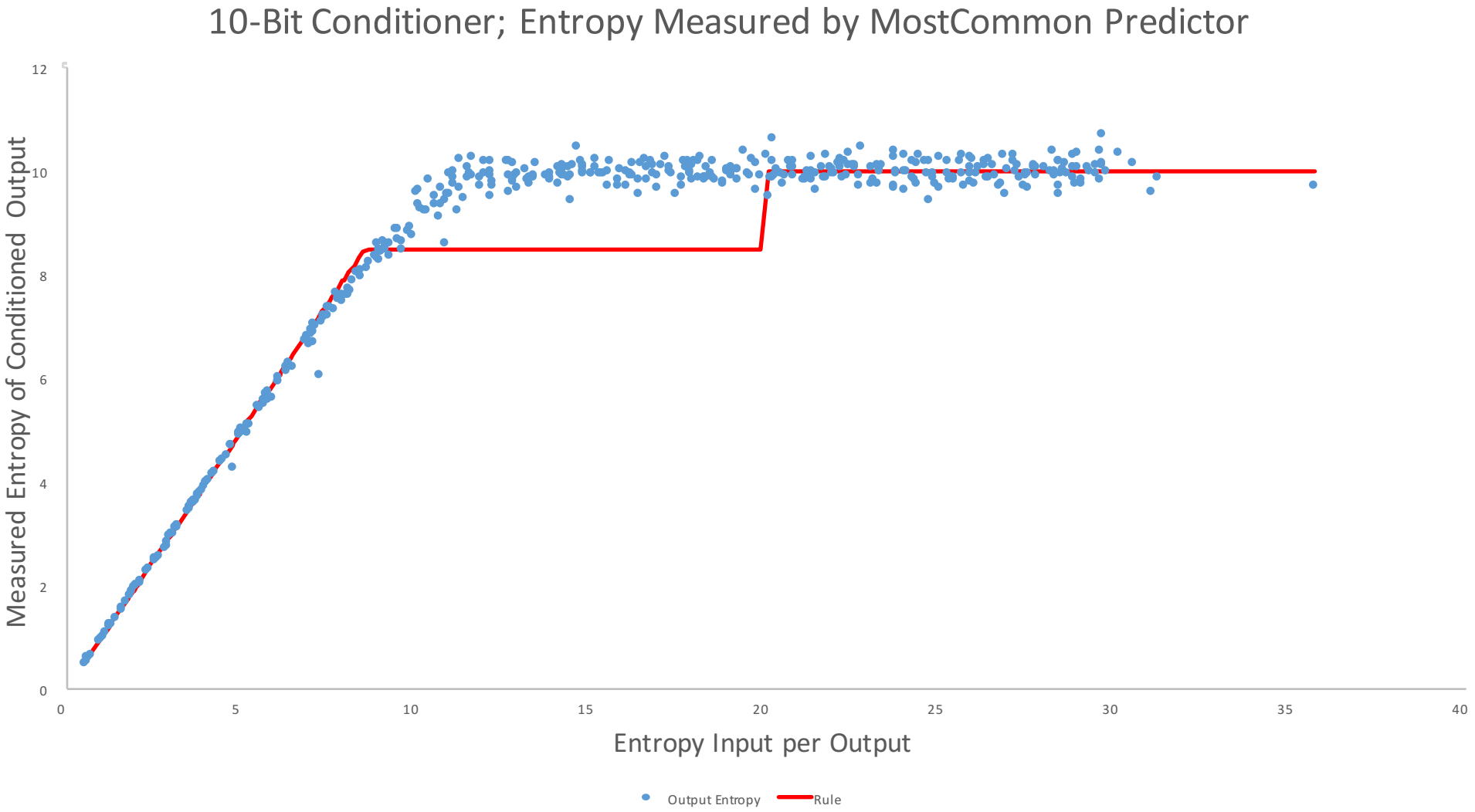
# Reading the Charts

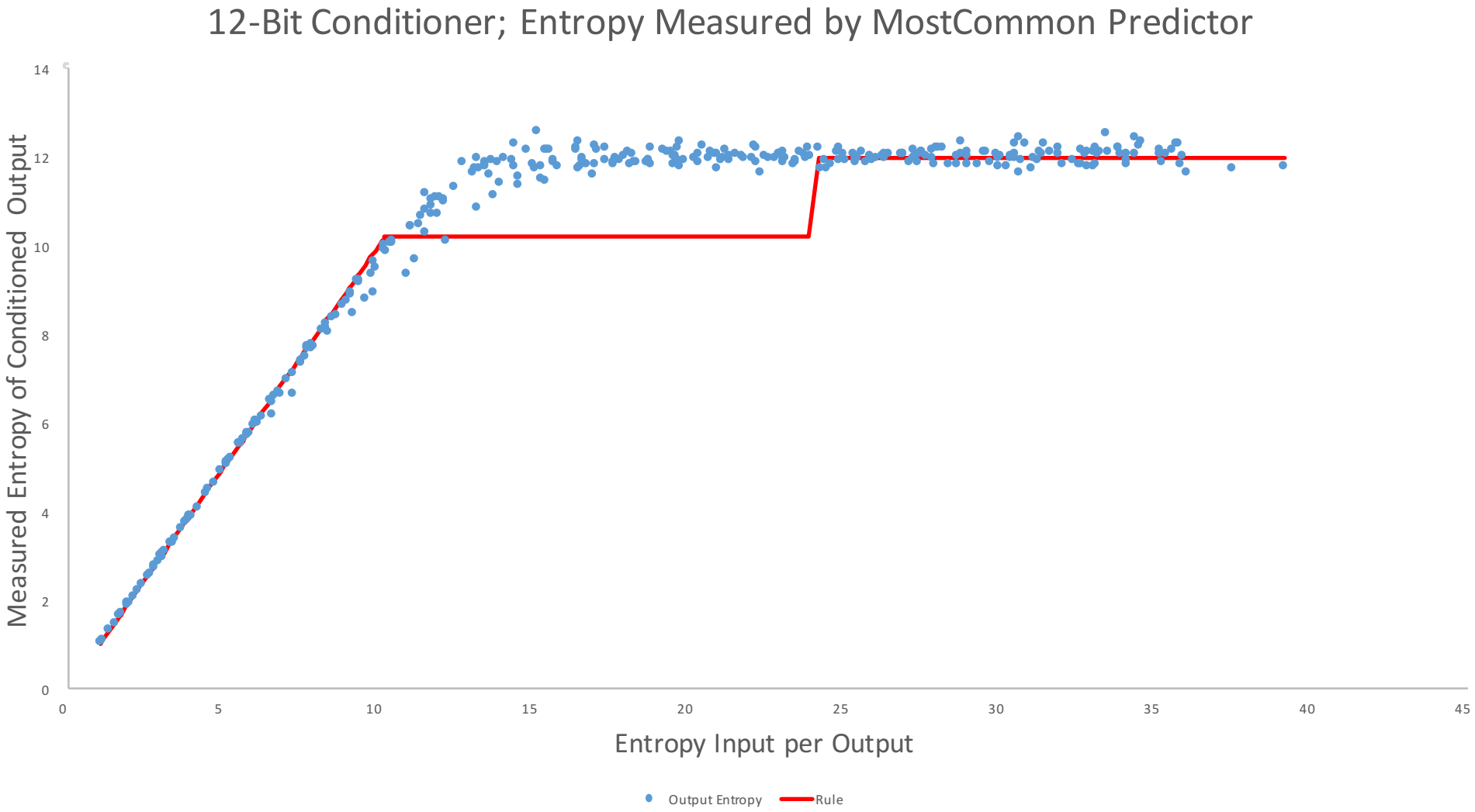
- The entropy accounting rule for vetted functions appears as a red line on all these charts.
- **Each dot is the result of one experiment:**
  - New conditioning function
  - New simulated source (near-uniform, uniform, normal)
  - Generate 100,000 conditioned outputs
  - Measure entropy/output with the MostCommon predictor
- AXES:
  - Horizontal axis is entropy input per conditioned output
  - Vertical axis is measured entropy per conditioned output

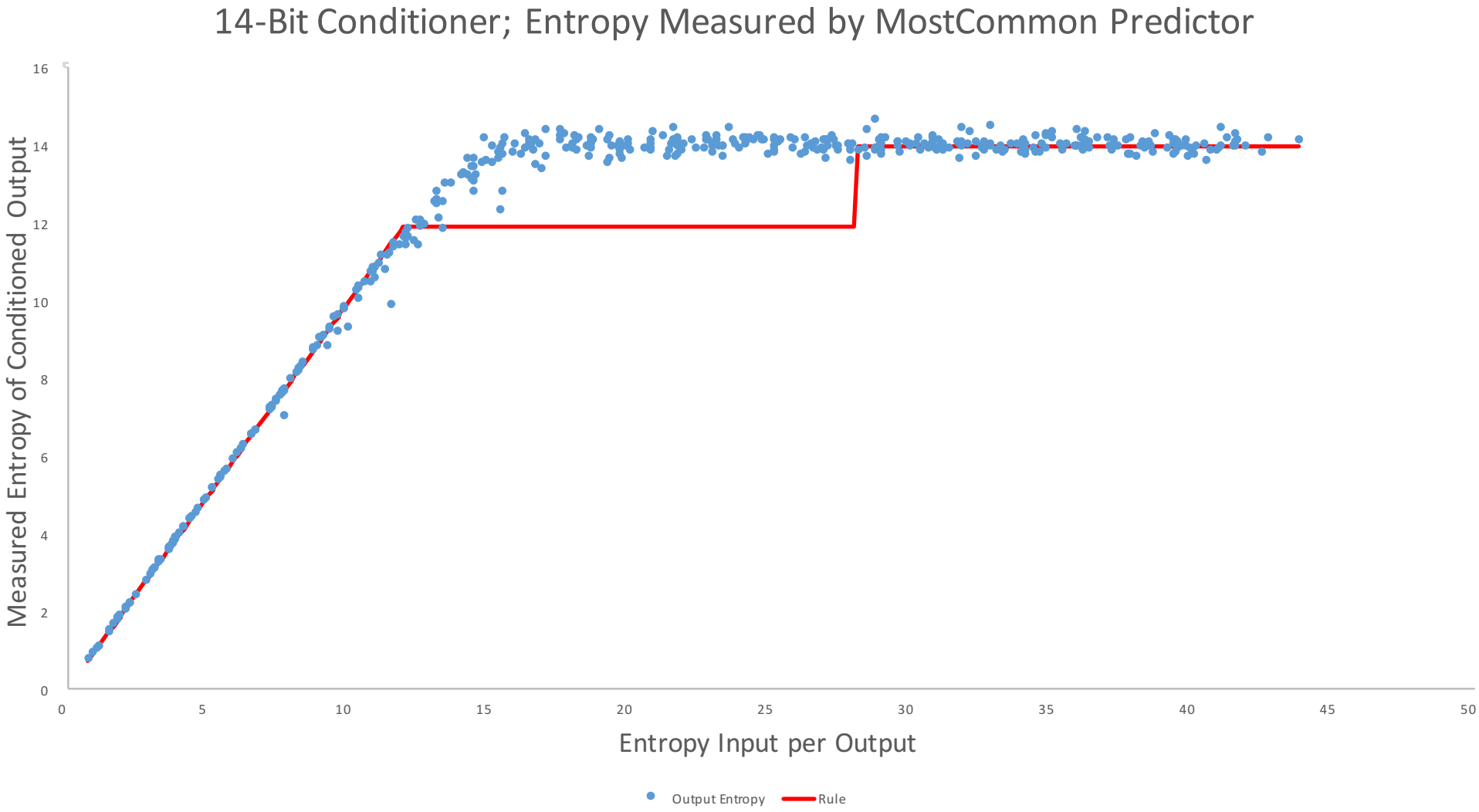

4-Bit Conditioner; Entropy Measured by MostCommon Predictor
Measured Entropy of Conditioned Output
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0
2
4
6
8
10
12
14
Entropy Input per Output
Output Entropy
Rule

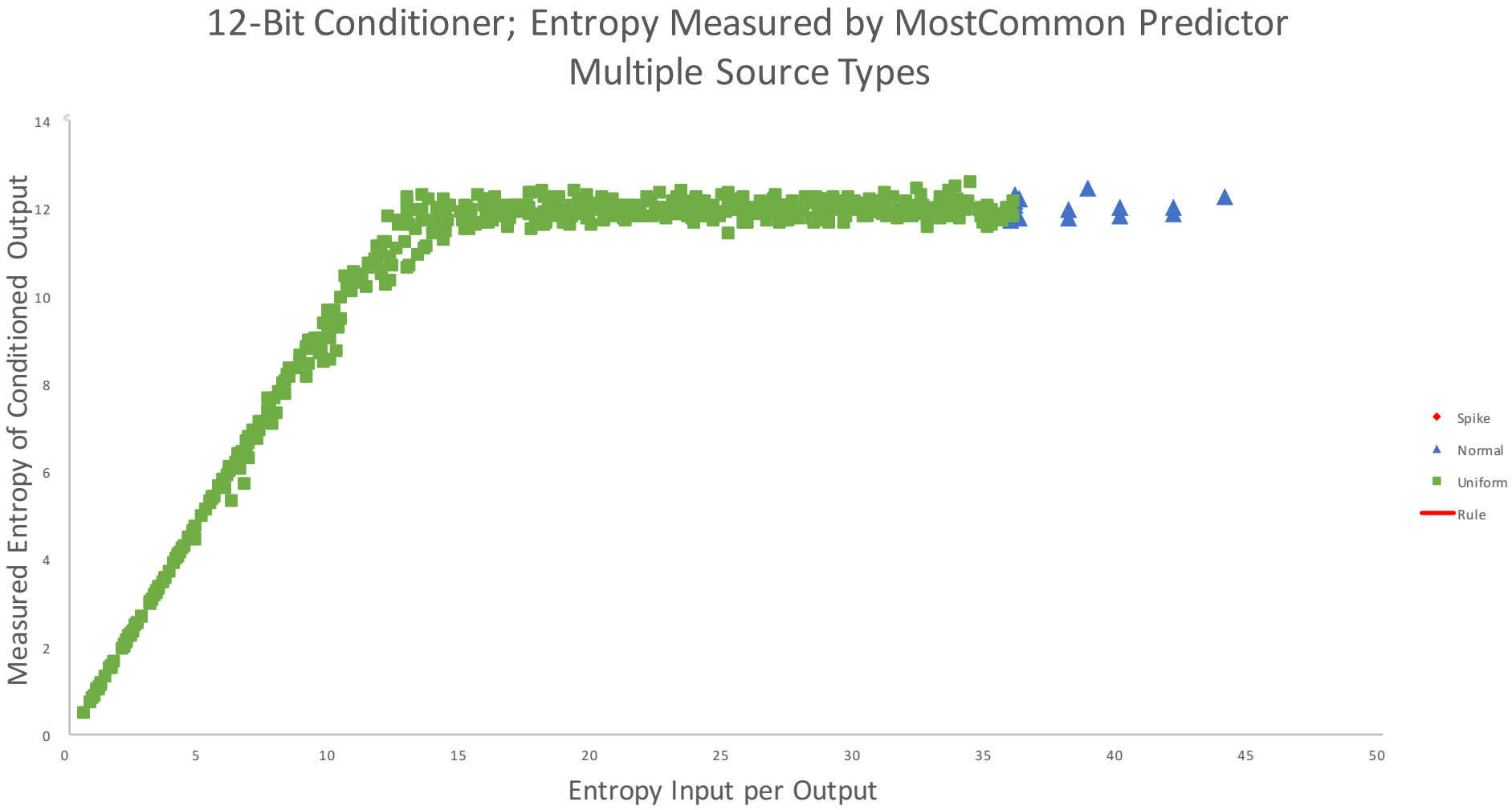

6-Bit Conditioner; Entropy Measured by MostCommon Predictor
Measured Entropy of Conditioned Output
0
1
2
3
4
5
6
7
0
2
4
6
8
10
12
14
16
18
20
Entropy Input per Output
Output Entropy
Rule

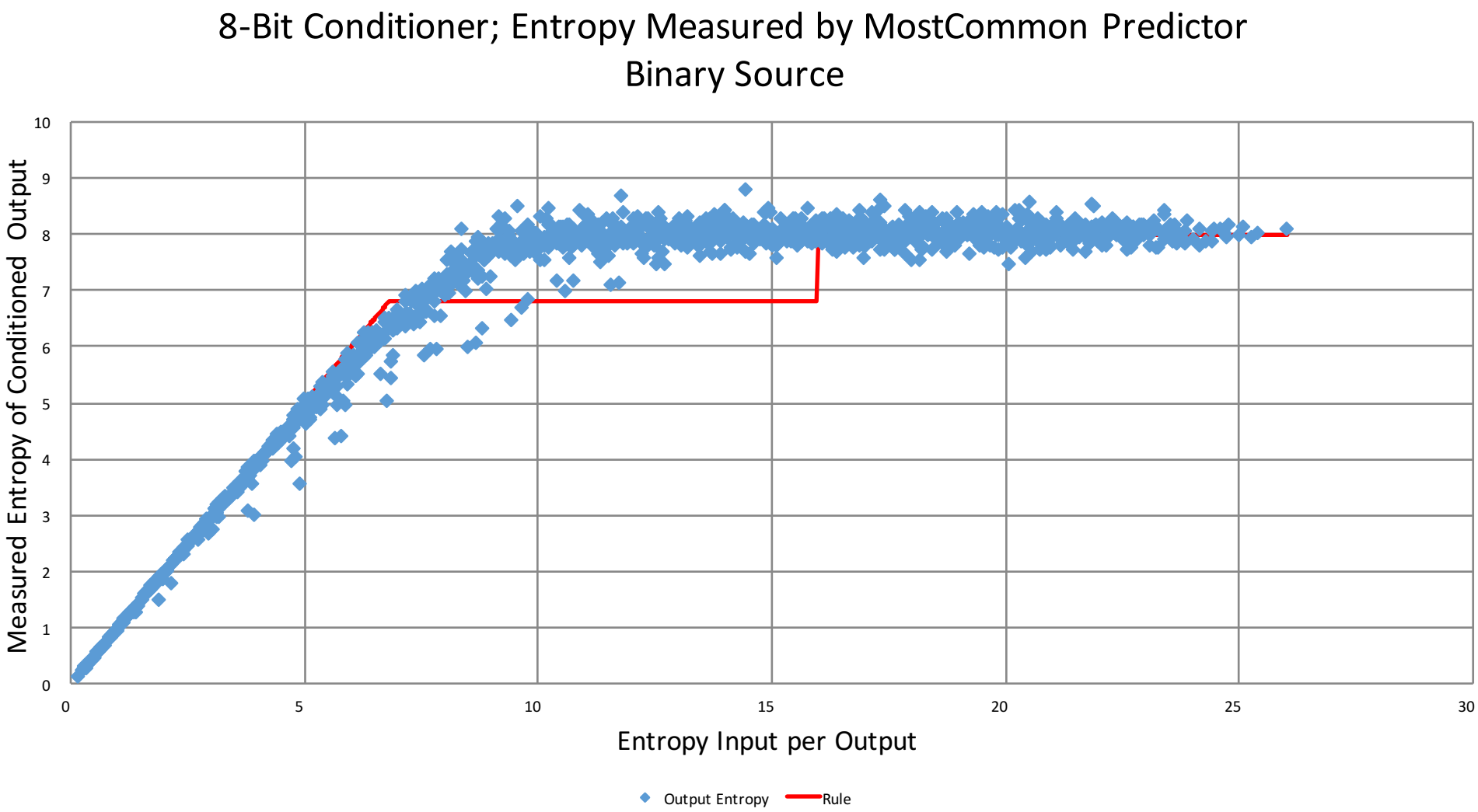

8-Bit Conditioner; Entropy Measured by MostCommon Predictor
Measured Entropy of Conditioned Output
0
1
2
3
4
5
6
7
8
9
0
5
10
15
20
25
30
Entropy Input per Output
Output Entropy
Rule

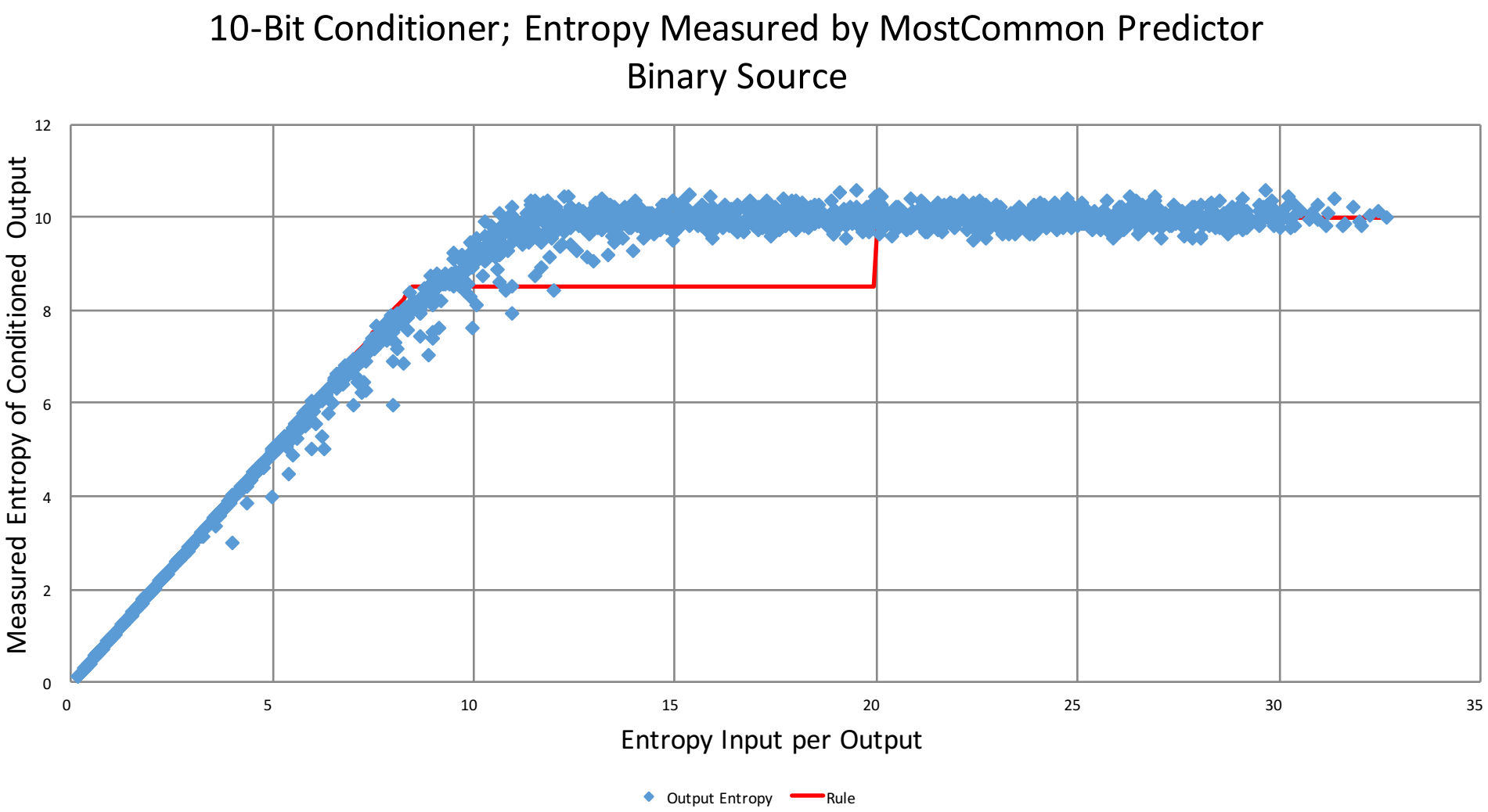

10-Bit Conditioner; Entropy Measured by MostCommon Predictor
Measured Entropy of Conditioned Output
Entropy Input per Output
0
2
4
6
8
10
12
0
5
10
15
20
25
30
35
40
Output Entropy
Rule


12-Bit Conditioner; Entropy Measured by MostCommon Predictor
Measured Entropy of Conditioned Output
Entropy Input per Output
0
2
4
6
8
10
12
14
0
5
10
15
20
25
30
35
40
45
Output Entropy
Rule


14-Bit Conditioner; Entropy Measured by MostCommon Predictor
Measured Entropy of Conditioned Output
Entropy Input per Output
0
2
4
6
8
10
12
14
16
0
5
10
15
20
25
30
35
40
45
50
Output Entropy
Rule


12-Bit Conditioner; Entropy Measured by MostCommon Predictor
Multiple Source Types
Measured Entropy of Conditioned Output
Entropy Input per Output
0
2
4
6
8
10
12
14
0
5
10
15
20
25
30
35
40
45
50
Spike
Normal
Uniform
Rule


8-Bit Conditioner; Entropy Measured by MostCommon Predictor
Binary Source
Measured Entropy of Conditioned Output
Entropy Input per Output
0
1
2
3
4
5
6
7
8
9
10
0
5
10
15
20
25
30
Output Entropy
Rule


10-Bit Conditioner; Entropy Measured by MostCommon Predictor
Binary Source
Measured Entropy of Conditioned Output
0
2
4
6
8
10
12
0
5
10
15
20
25
30
35
Entropy Input per Output
Output Entropy
Rule

# Summary of Empirical Data

- We tested the entropy accounting formulas in practice for small cases
- General result: the formulas work pretty well at giving a reasonable, if somewhat conservative, estimate of entropy from conditioner
- Results are noisier for smaller conditioner sizes, but seem to become smoother and better behaved even as we get to 12- and 14-bit conditioner sizes.

# Wrapup

- Conditioners are an optional component of an entropy source, intended to increase entropy/output.
- We allow vetted and non-vetted conditioners
- Entropy accounting is a little tricky for conditioners.
- We have run some simulations to verify that our entropy accounting gives reasonable answers for small (tractable) cases.

# Open Questions

- The choice of 0.85 as a constant was pretty arbitrary. Should we choose another value?
- Should we make the entropy accounting equation more complicated (and thus more accurate?)
- Should we allow full-entropy from non-vetted functions?
- If so, how should we test the outputs?
  - Maybe iid tests and require a result that's "close enough" to full entropy?
  - Note that iid estimate won't estimate full entropy--it makes a conservative (99% confidence interval) estimate.