Composing Component Tests

Dick Hamlet

Department of Computer Science

Portland, OR 97207 USA

hamlet@cs.pdx.edu

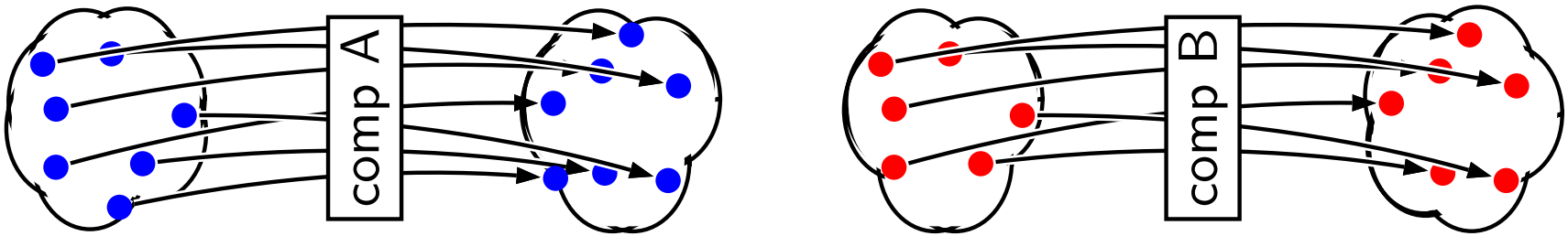
Supported by NSF CCR-0112654 and SFI E.T.S. Walton Fellowship

The Setting (== Unsupported Biases?)

- Testing is at best a craft
  - It often works on small units
  - It often fails on larger systems
- Components (==‘executables’) are ideal ‘units’
- Subdomain-testing theory explains all
- Tools support revealing experiments
- Series composition is fundamental

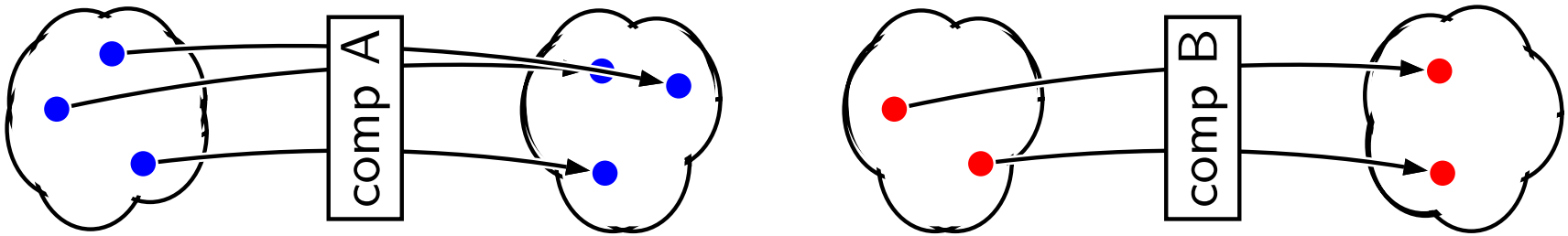
Unit Tests Don't Compose

Unit testing two components A and B



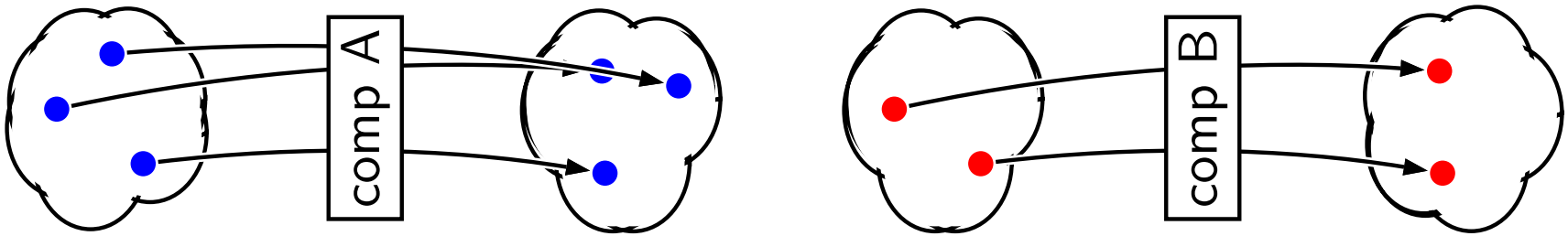
Unit Tests Don't Compose

Unit testing two components A and B

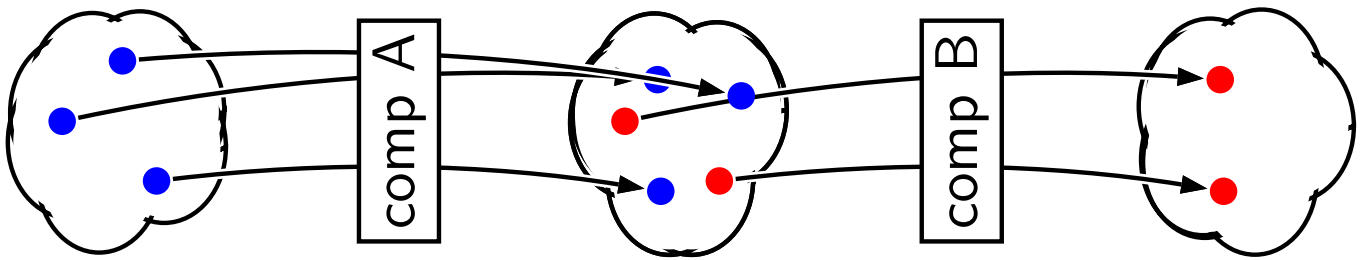


Unit Tests Don't Compose

Unit testing two components A and B

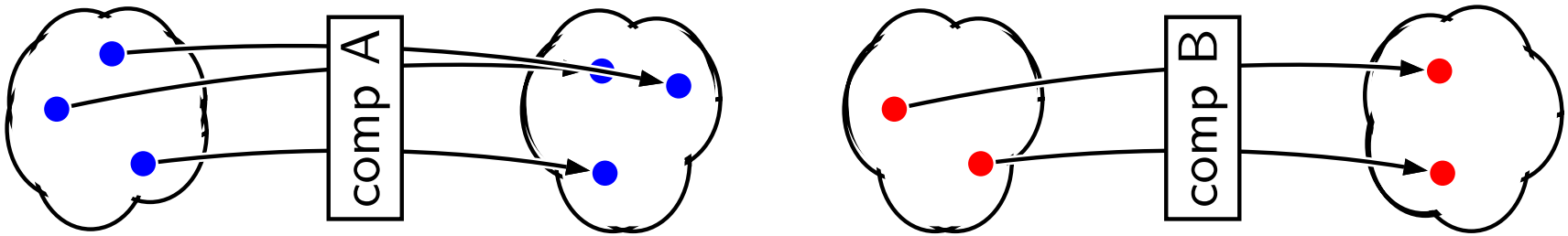


Series combination (interface mismatch)

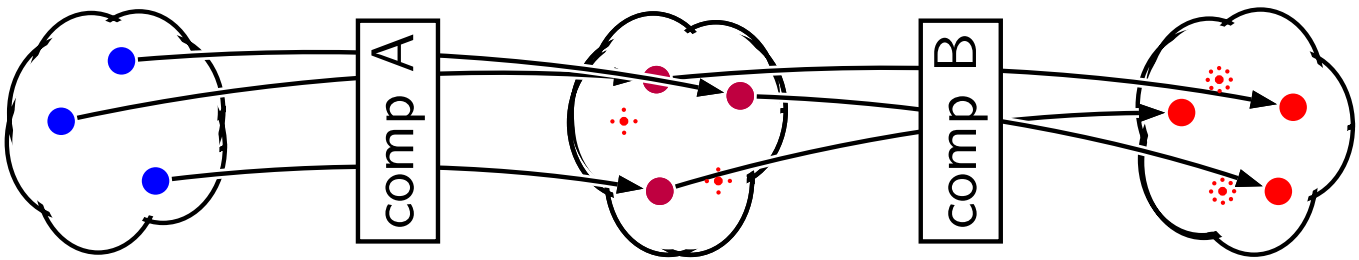


Unit Tests Don't Compose

Unit testing two components A and B

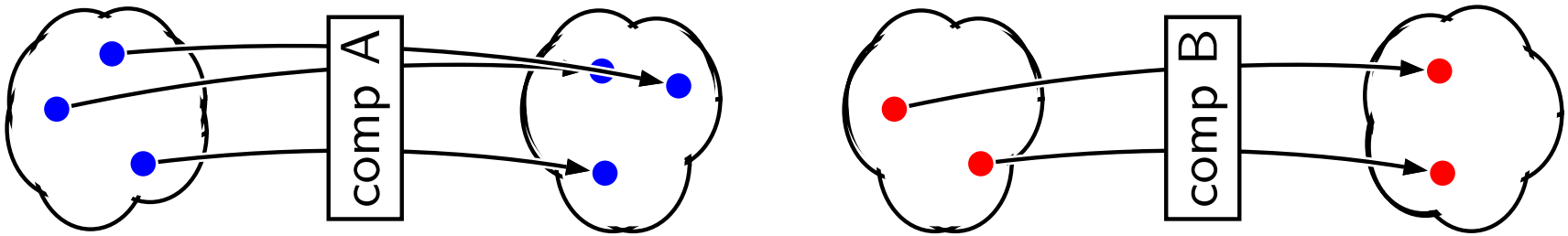


Series combination (forced interface match)

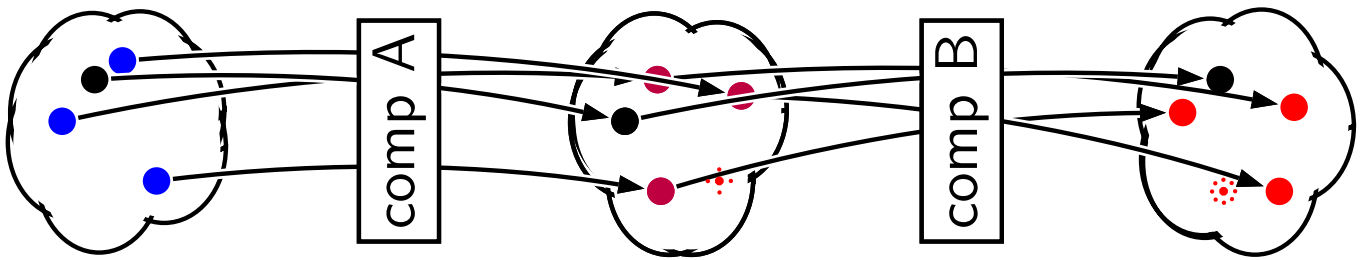


Unit Tests Don't Compose

Unit testing two components A and B



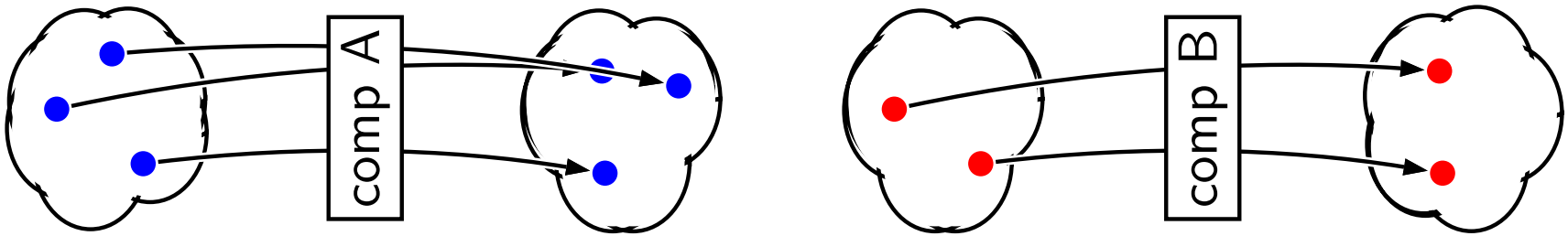
Series combination (forced interface match)



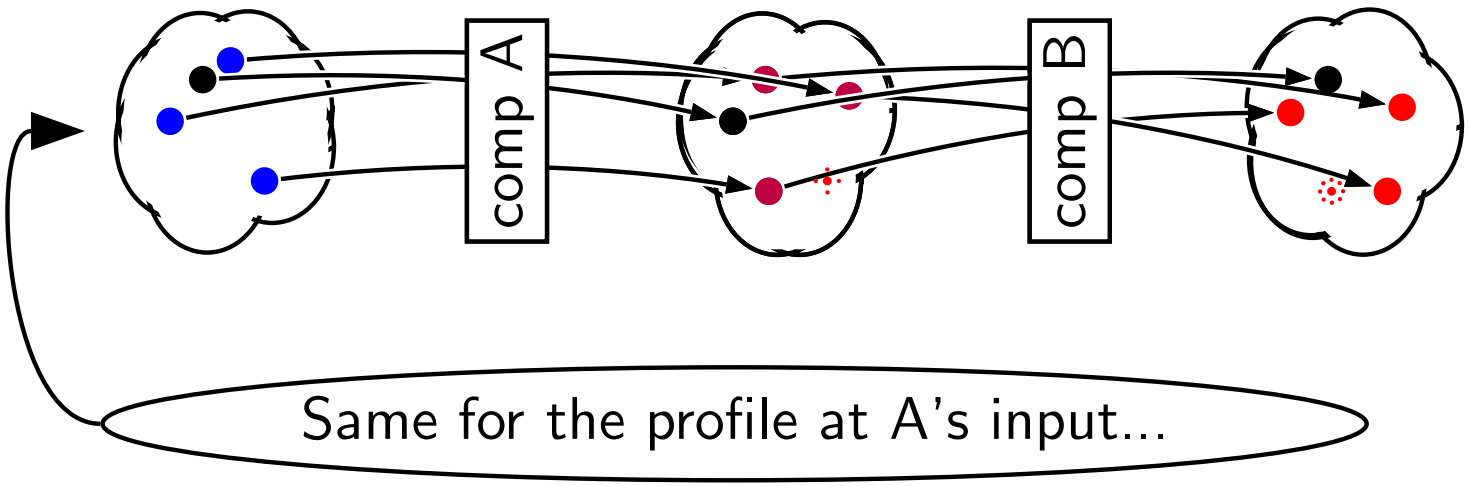
System fails because B test is poor

Unit Tests Don't Compose

Unit testing two components A and B

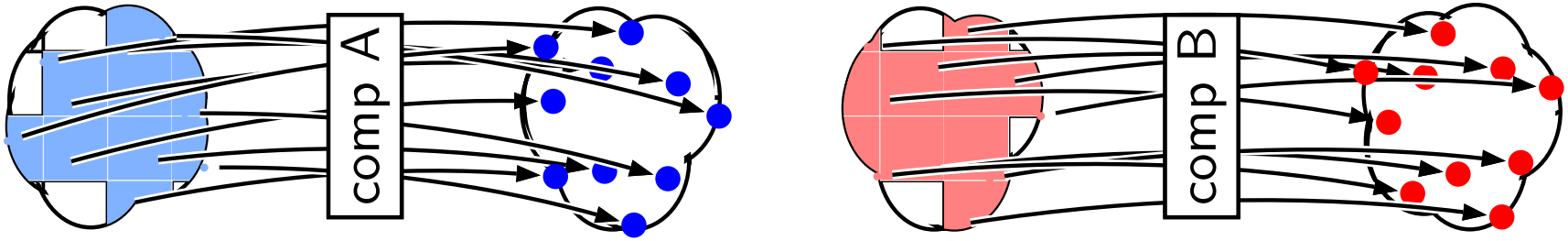


Series combination (forced interface match)



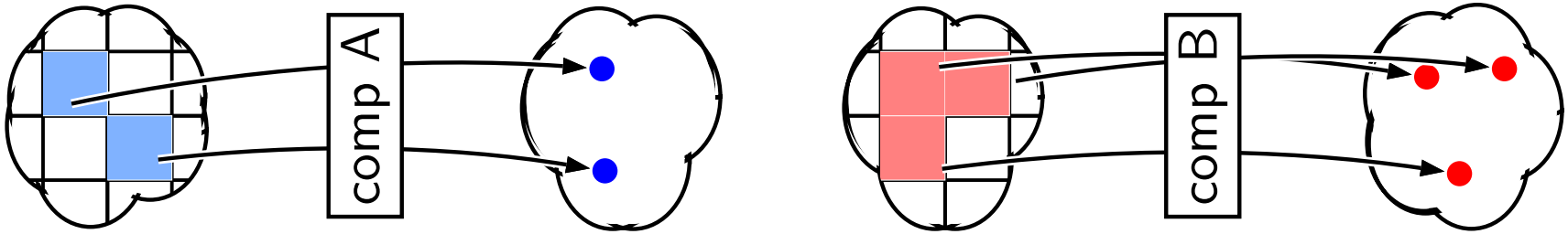
Subdomains to the Rescue!

Subdomain testing two components A and B



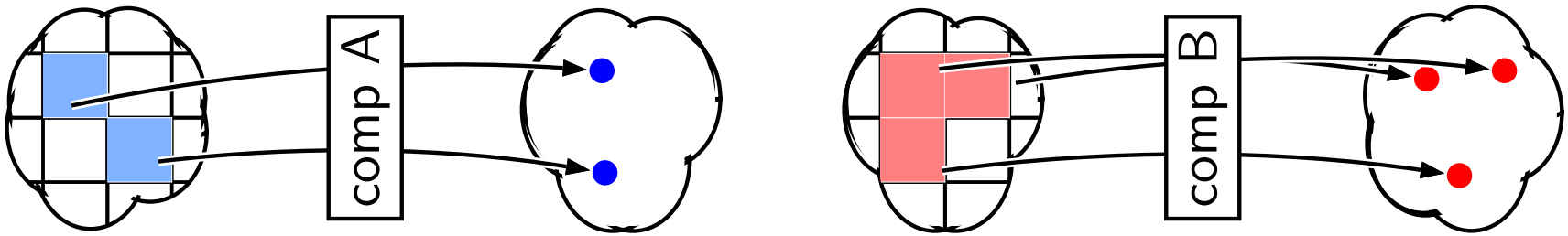
Subdomains to the Rescue!

Subdomain testing two components A and B

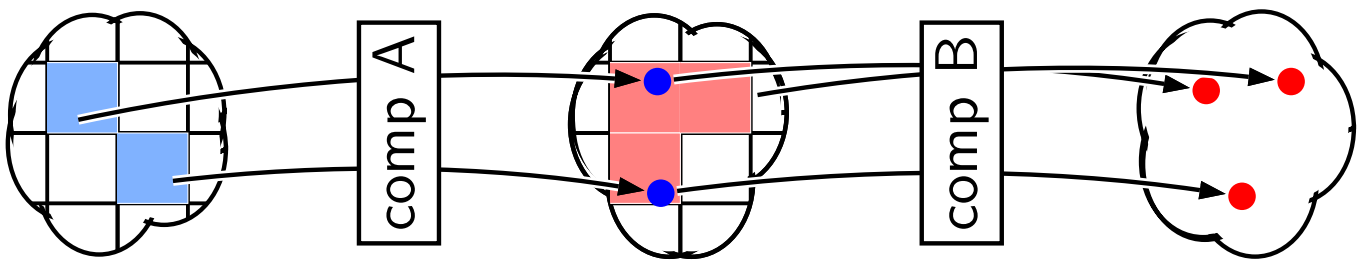


Subdomains to the Rescue!

Subdomain testing two components A and B

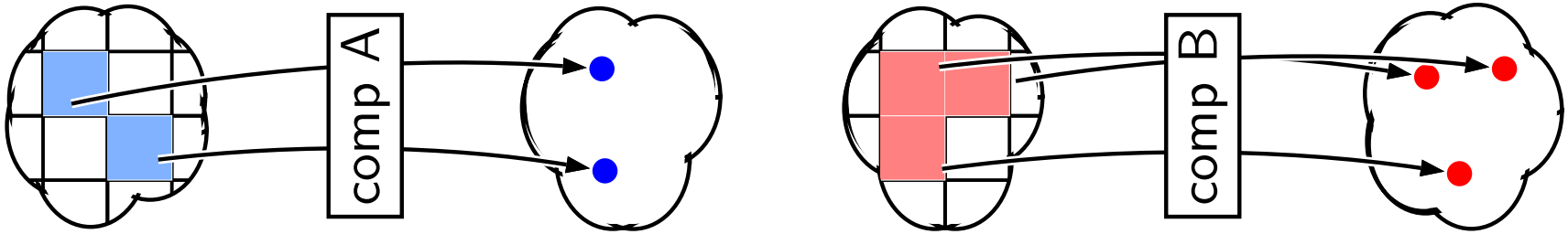


Never an interface mismatch

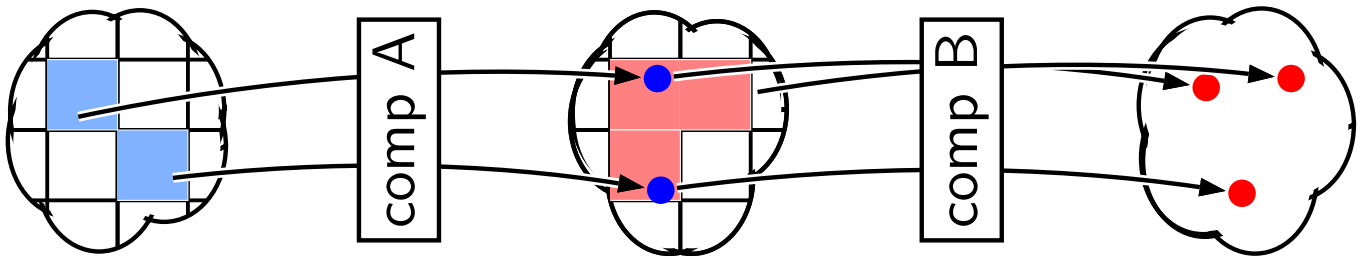


Subdomains to the Rescue!

Subdomain testing two components A and B



Never an interface mismatch



Testing in isolation:

- Component B need not know what Component A will send
- Component A need not know what profile it will see

Tool-supported Component-based Development

1. Develop components → Repository
   - Write component code
   - *Test code to specification*
   - Choose good subdomains
   - *Approximate with subdomain test*

Key:
(Blue) Human, by-hand *(Red slant) Tools, automatic*

Tool-supported Component-based Development

1. Develop components → Repository
   - Write component code
   - *Test code to specification*
   - Choose good subdomains
   - *Approximate with subdomain test*
2. Design system
   - Select components from repository (bottom-up design)
   - *Synthesize system approximation*
   - *Check against system specification*

Key:
(Blue) Human, by-hand *(Red slant) Tools, automatic*

Tool-supported Component-based Development

1. Develop components → Repository
   - Write component code
   - *Test code to specification*
   - Choose good subdomains
   - *Approximate with subdomain test*
2. Design system
   - Decompose system into components (top-down design)
   - *Synthesize system approximation*
   - *Check against system specification*

Key:
(Blue) Human, by-hand *(Red slant) Tools, automatic*

Tool-supported Component-based Development

1. Develop components → Repository
   - Write component code
   - *Test code to specification*
   - Choose good subdomains
   - *Approximate with subdomain test*
2. Design system
   - Describe system and get component approximations
   - *Synthesize system approximation*
   - *Check against system specification*

Key:
(Blue) Human, by-hand *(Red slant) Tools, automatic*

Tool-supported Component-based Development

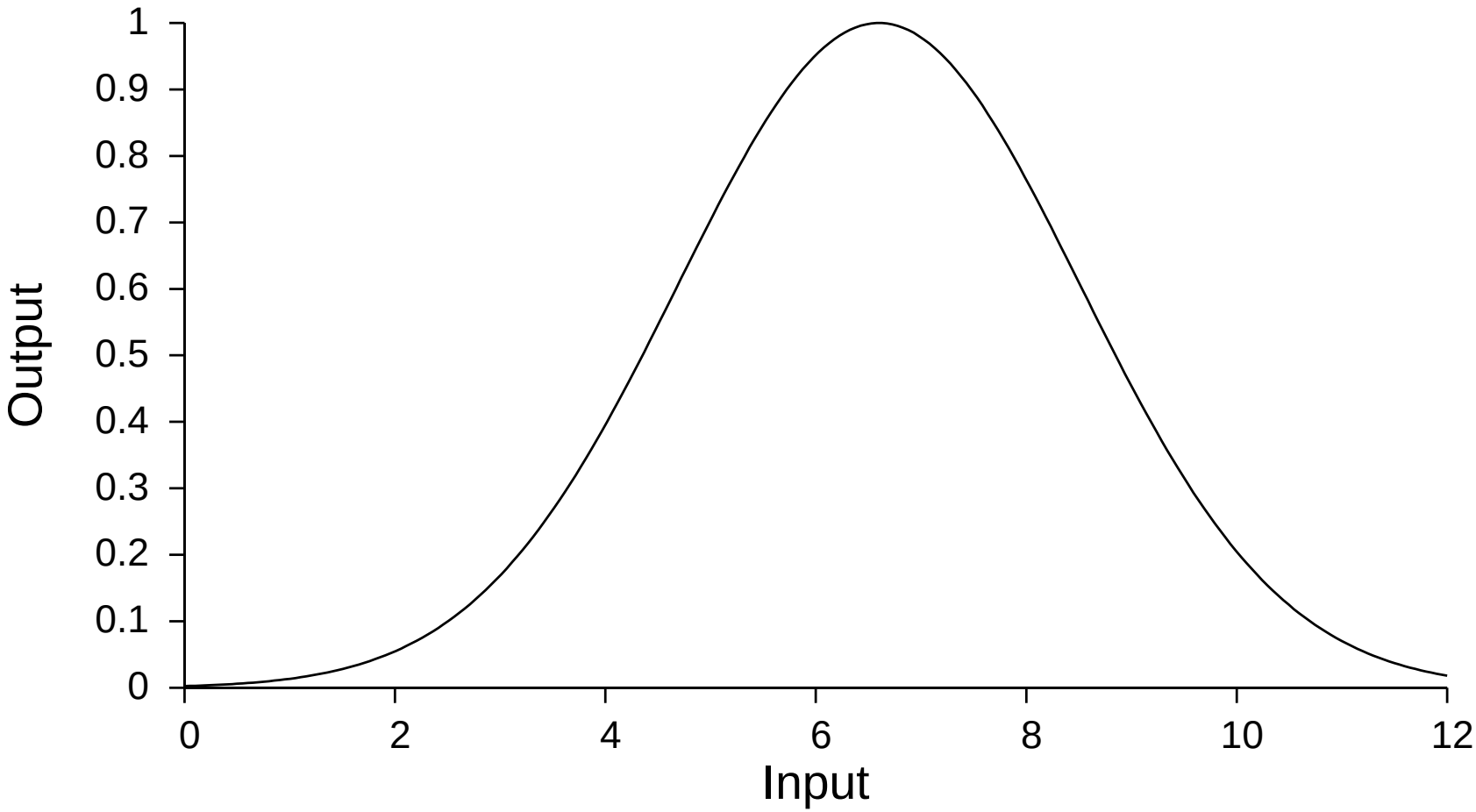
1. Develop components → Repository
   - Write component code
   - *Test code to specification*
   - Choose good subdomains
   - *Approximate with subdomain test*
2. Design system
   - Describe system and get component approximations
   - *Synthesize system approximation*
   - *Check against system specification*
3. Buy components, build system
   - ~~Test system against specification~~

Key:

(Blue) Human, by-hand *(Red slant) Tools, automatic*

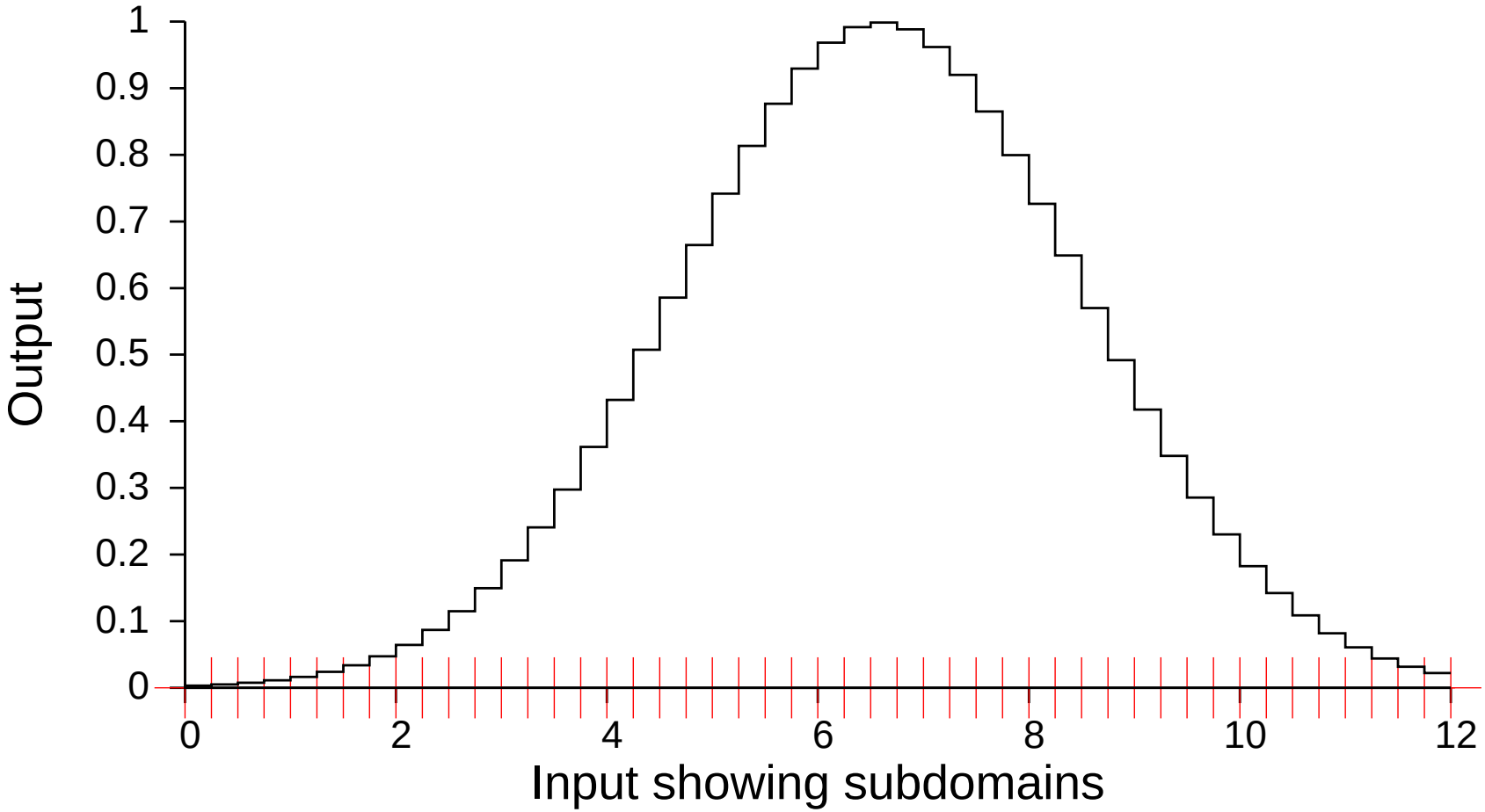
Develop and Approximate Components

Component '`Bell`' measurement:
Gaussian, $\mu = 6.6, \sigma = 2.7$, 142 equispaced tests:



Develop and Approximate Components

Approximation measurements
(3 samples in each of 48 subdomains):

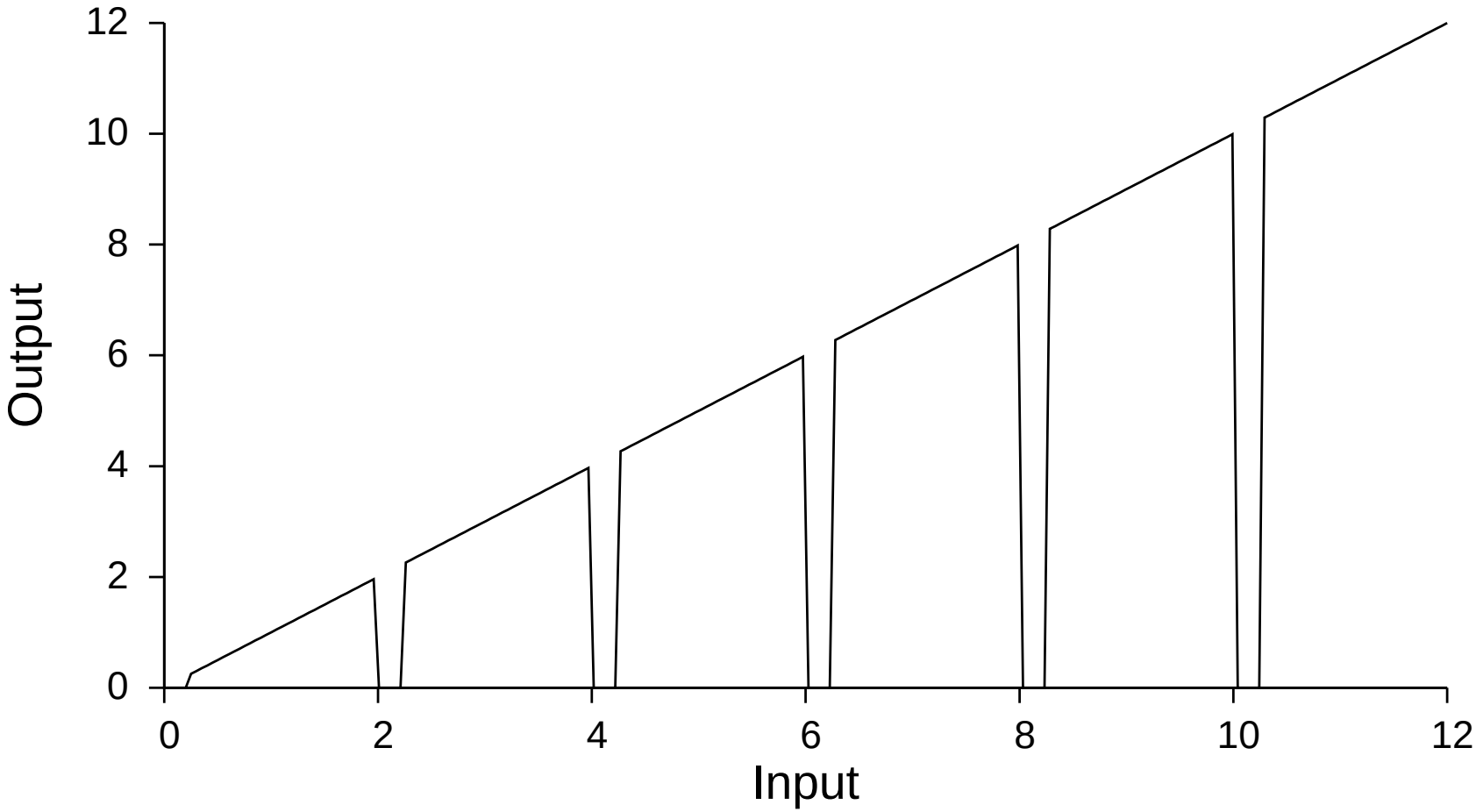


R-M-S error ranges from 0.1% to 4.3% by subdomain

Develop and Approximate Components

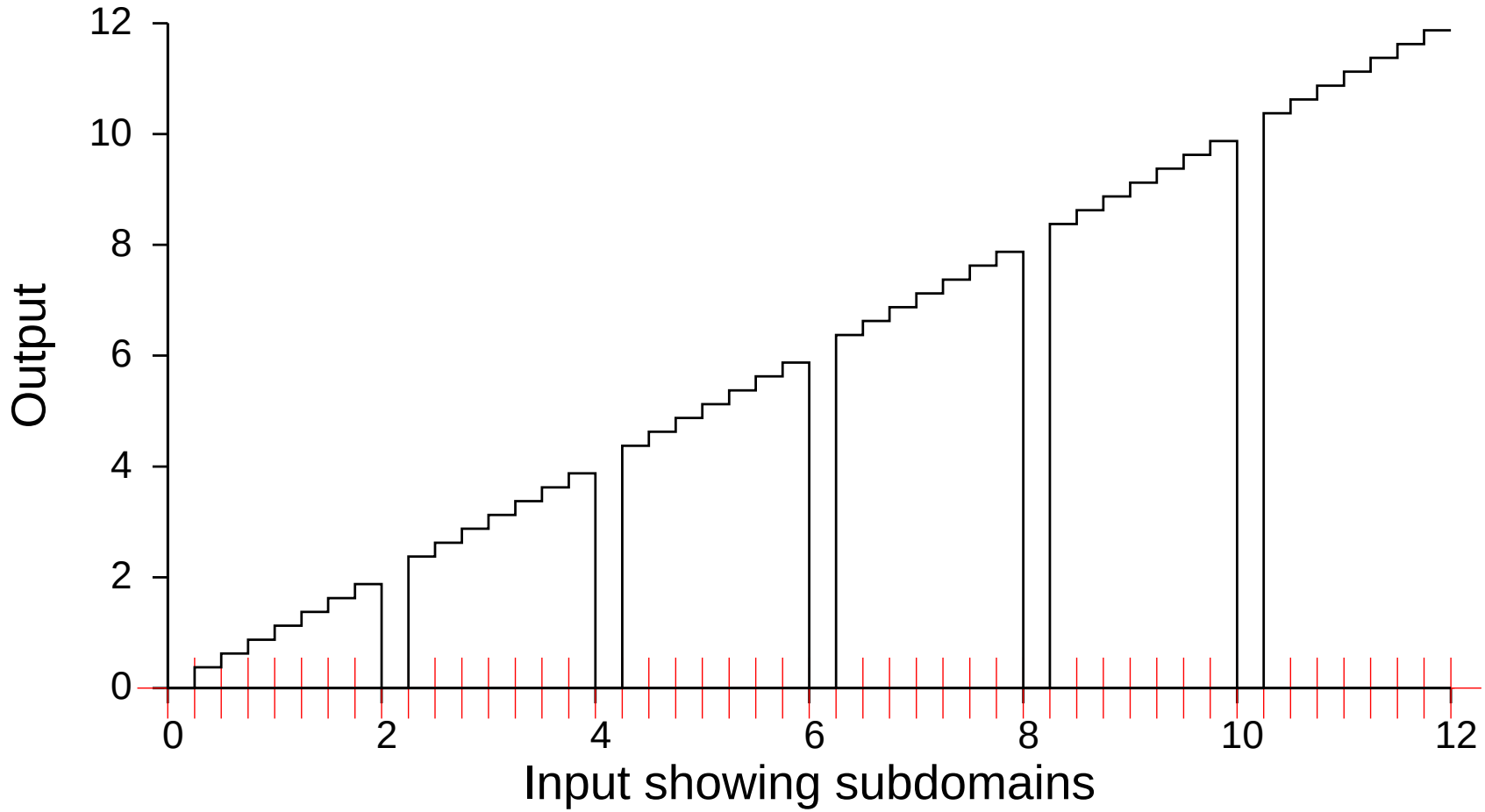
Component '`Chop`' measurement:

Chopping function, 142 equispaced tests:



Develop and Approximate Components

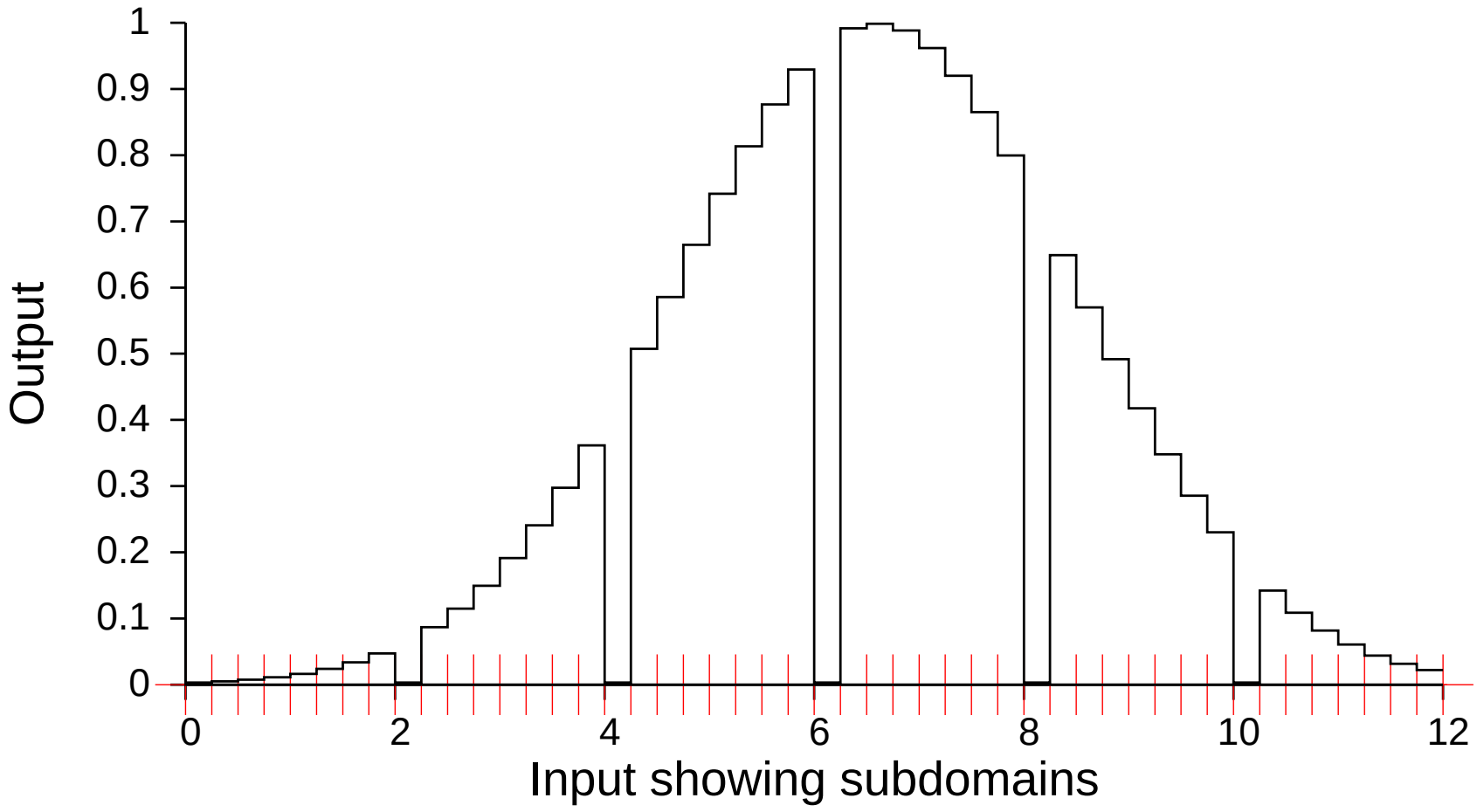
Approximation measurements
(3 samples in each of 48 subdomains):



R-M-S error ranges from 0 to 1.1% by subdomain

Series System Synthesis Prediction

Prediction for system `Chop; Bell`:



Synthesis is fast (5X execution)

R-M-S prediction error 0.1% – 4.5%

Now for Some Content: Research Questions

1. Can non-functional properties (run time, reliability, ...) be synthesized?
2. Can complicated systems be synthesized?
3. What happens as subdomains shrink and move?
4. Suppose the components have persistent state?
5. Suppose the components execute concurrently?
6. How about non-trivial examples?
7. How do the tools work?

Now for Some Content: Research Questions

1. Can non-functional properties (run time, reliability, ...) be synthesized?
2. Can complicated systems be synthesized?
3. What happens as subdomains shrink and move?
4. Suppose the components have persistent state?
5. Suppose the components execute concurrently?
6. How about non-trivial examples?
7. How do the tools work?

TIME'S UP!

Try a Journal

- Most research doesn't break into 10-page papers
  at yearly intervals
- Conference referees are volunteers in a thankless job, but...
  - Skimming the abstract to assign a rating isn't refereeing
  - No feedback to the author
  - A poor paper describing important work is rejected, not fixed
  - Misunderstanding or not understanding $\longrightarrow$ reject

Try a Journal

- Most research doesn't break into 10-page papers
  at yearly intervals
- Conference referees are volunteers in a thankless job, but...
  - Skimming the abstract to assign a rating isn't refereeing
  - No feedback to the author
  - A poor paper describing important work is rejected, not fixed
  - Misunderstanding or not understanding $\longrightarrow$ reject
- Journal referees are volunteers in a thankless job...

For those Research Questions...

- Read the paper in the forthcoming issue of TOSEM
- Or do it yourself – free tools and experiment data at `www.cs.pdx.edu/~hamlet/components.html`