Efficient Uncertainty Modeling for System Design via Mixed Integer Programming

Zichang He, Weilong Cui, Chunfeng Cui,
Timothy Sherwood and Zheng Zhang

UCSB
UNIVERSITY OF CALIFORNIA
SANTA BARBARA

NSF

Uncertainty is Unavoidable!

Uncertainty in real architecture

>50% of fabricated chips are **30%** slower or consume **10x** more standby power [Miranda 2012]

Uncertainty in real architecture

Average project time spent in verification:

57% in 2014 and growing [Foster 2015]

Works on Architecture Uncertainty

- [Cui and Sherwood, MICRO’17] Estimating and understanding architecture risk

  Projection Uncertainty: Unknown from application space

  Design Uncertainty: Unknown from design space

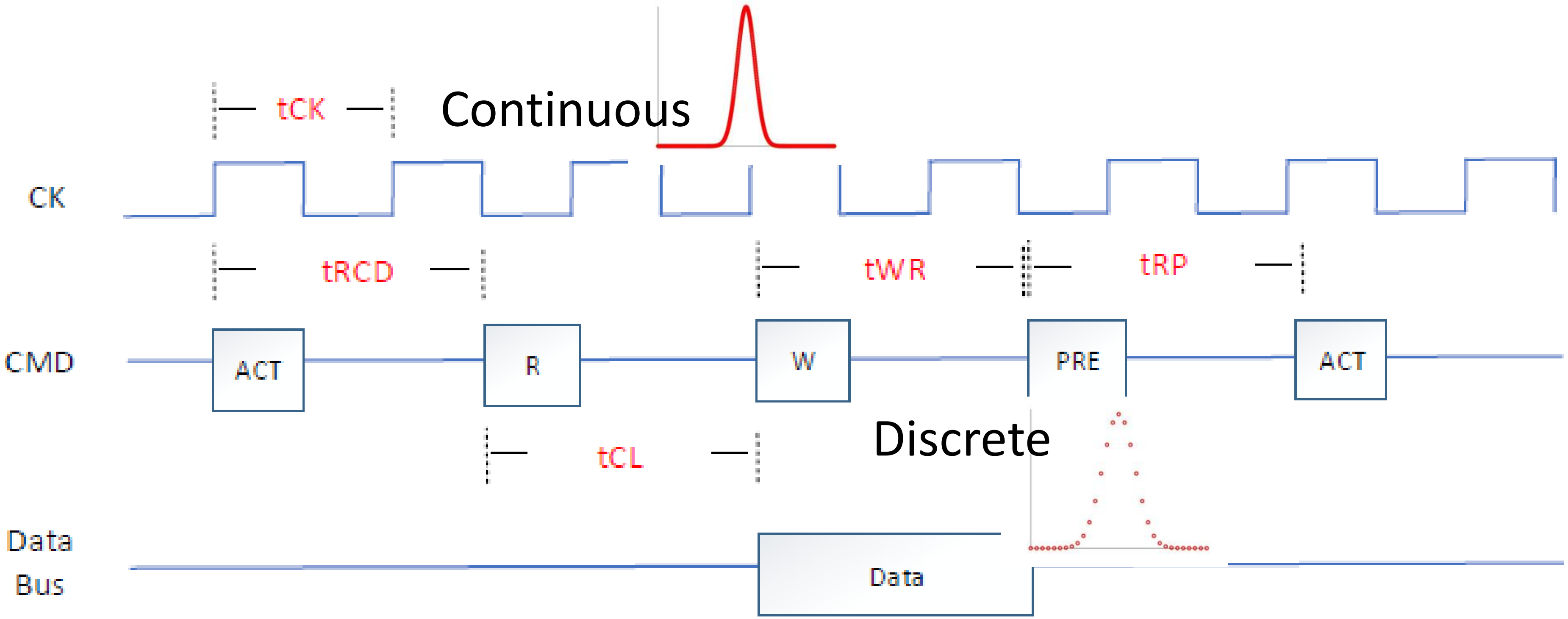
  Process Uncertainty: Unknown from manufacture process

- [Cui et al., ISCA’18] Charm: A Language for Closed-form High-level Architecture Modeling (open source package)

  ......

- [He et al., ICCAD’19 (this work)] Mathematical methodology to analyze and quantify the architecture uncertainty.

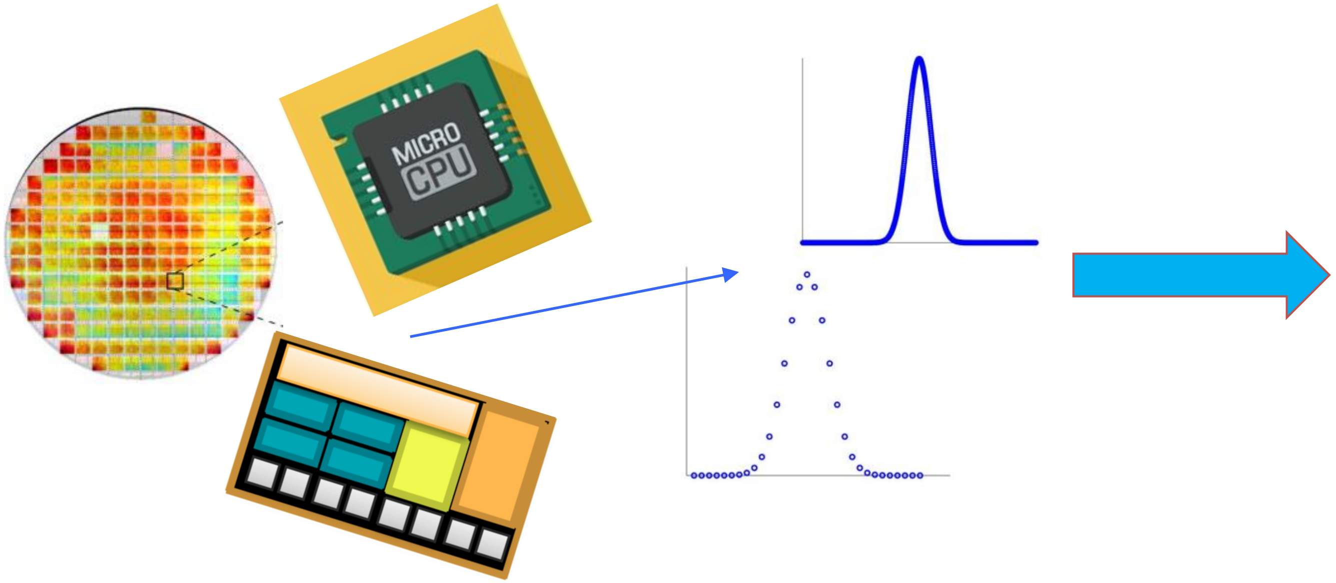
Example of Uncertain Architecture



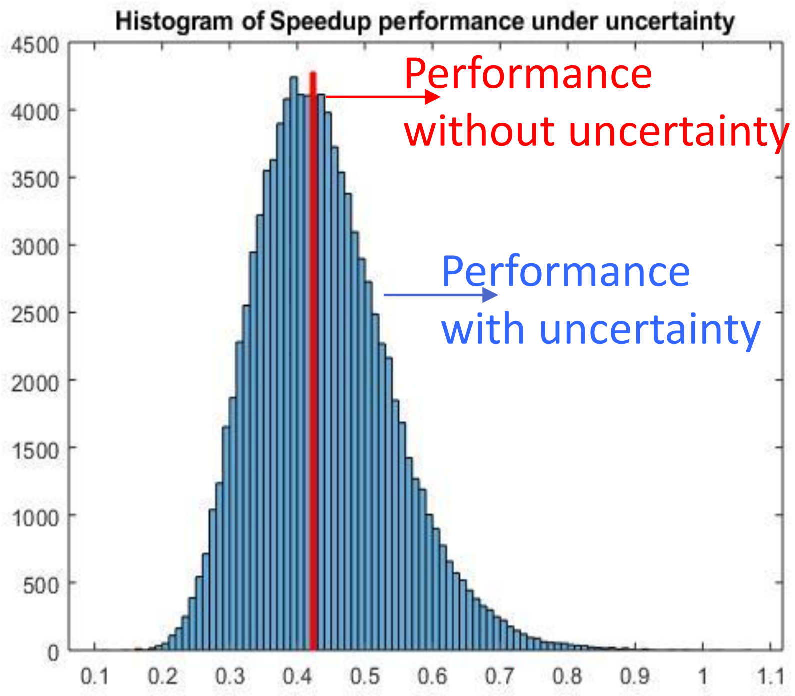
In a DRAM system, timing parameters may be inaccurate as designed. To represent the uncertainty, mixed-type random variables: tCK is continuous, others are discrete.

Uncertainty Quantification: Motivation

Input $\boldsymbol{\xi} \in \mathbb{R}^d$



Output $y(\boldsymbol{\xi})$



Not to eliminate uncertainty, but to know how uncertainty will influence the system: output statistical moments and the shape of distribution

Bottlenecks in Architecture UQ

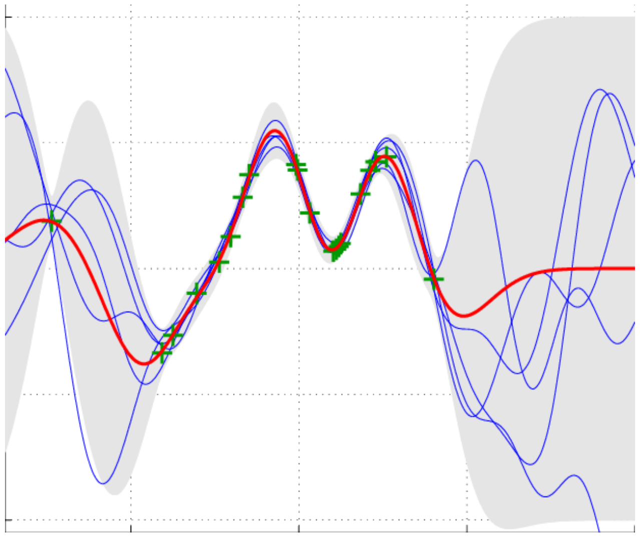
1. Cycle-level simulations are usually very expensive: Monte Carlo is unacceptable.

$$Total\ Time\ Cost = \frac{\#\ of\ Runs \times Time\ Cost\ per\ Run}{Parallisim}$$

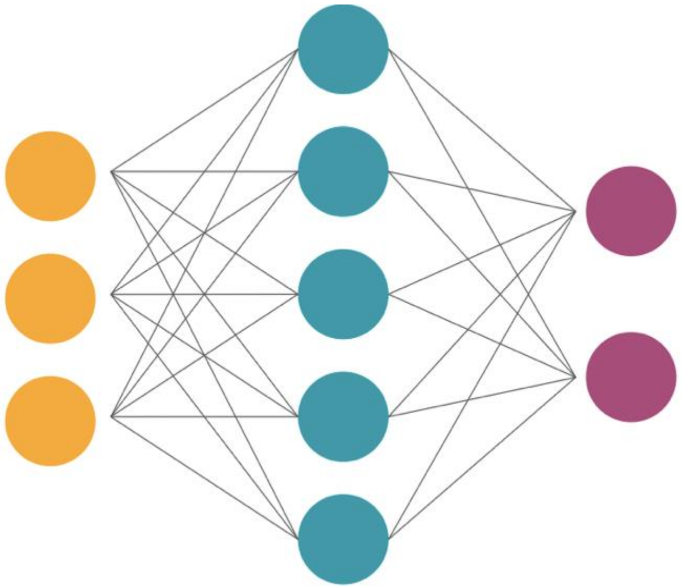
Several Mins, Hours or even Days!

2. The mixed-integer constraint is hard for UQ solver.

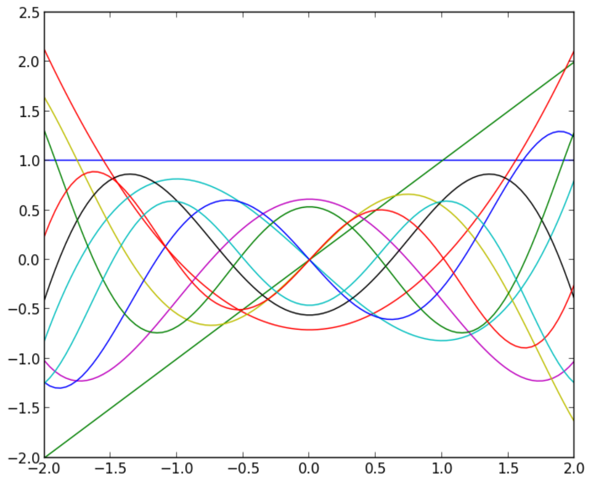
Solution: surrogate modelling



Gaussian Process



Neural Network



Generalized Polynomial Chaos (gPC)

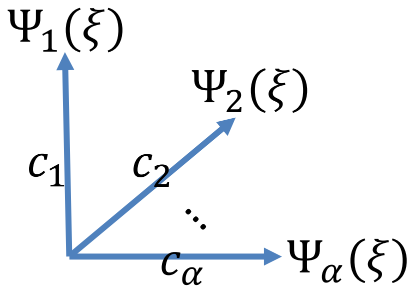
……

Many ML methods do NOT work due to limited samples & integer samples!

Solution: gPC expansion

gPC expansion: Orthogonal basis functions + Coefficients,

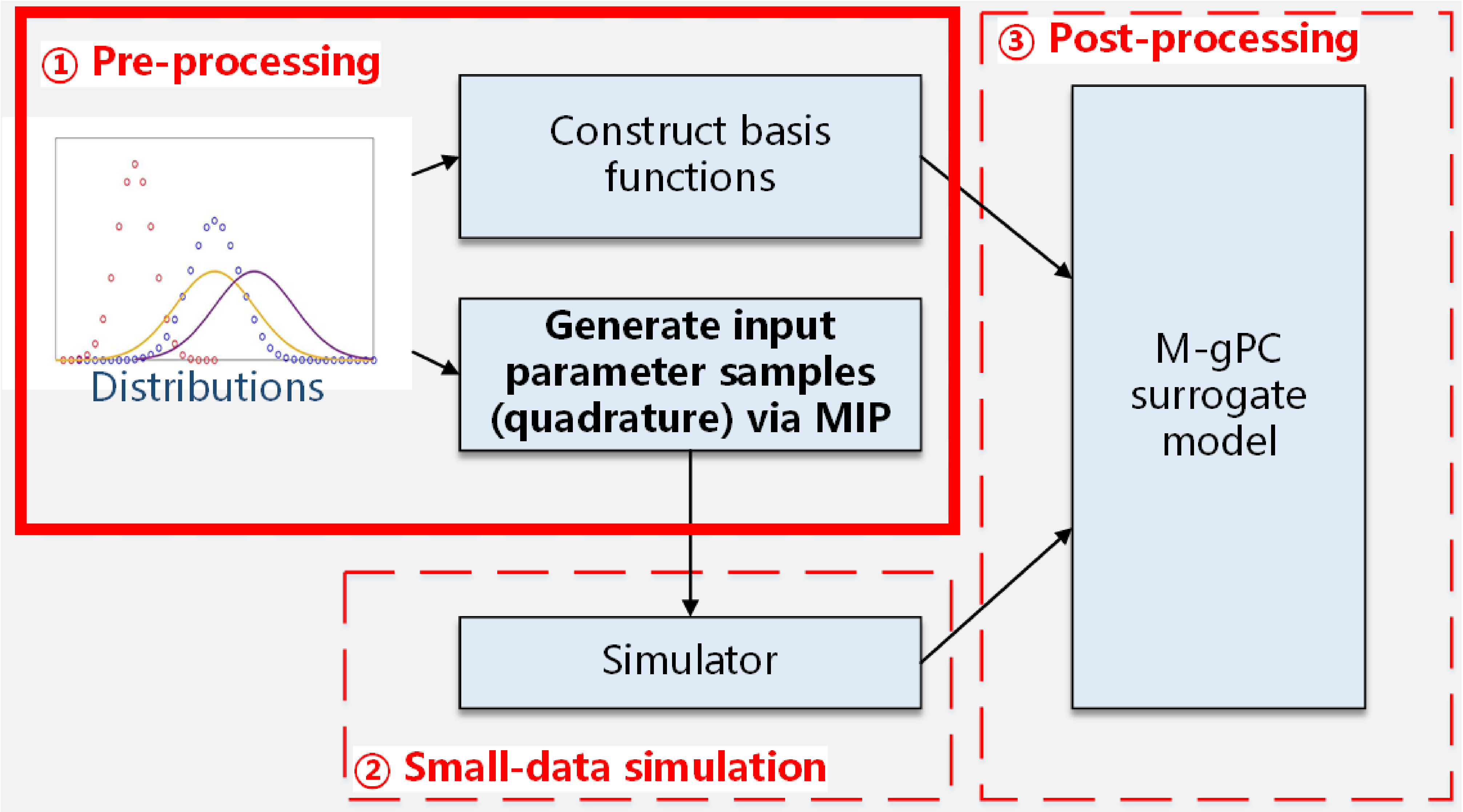
$$y(\xi) \approx \sum_{|\alpha|=0}^{P} c_\alpha \Psi_\alpha(\xi)$$



Classical gPC is NOT enough to solve previous two bottlenecks:

1. Curse of dimensionality ($(P+1)^d$)
2. No integer sampling rule

Proposed M-gPC framework



Pre-processing: Basis construction

Model uncertainty as random variables

$$[\xi_1, \xi_2, \xi_3, \dots \xi_d]$$

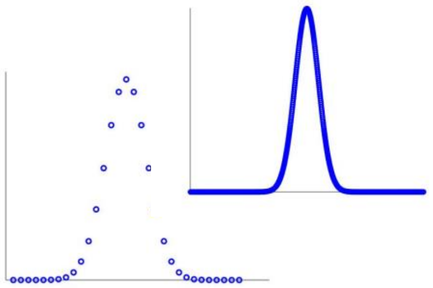
Construct basis function via three term recurrence [Gautschi, 1982]

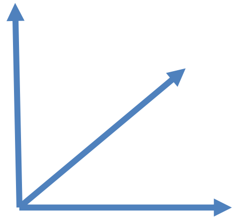
Random variable ξ → TTR relation → Corresponding orthogonal basis $\Psi_\alpha(\xi)$

$$\pi_{i+1}(x) = (x - \alpha_i)\,\pi_i(x) - \beta_i(x)\,\pi_{i-1}(x)$$

$$\pi_{-1}(x) = 0, \quad \pi_0(x) = 1, \quad i = 0, 1, \dots, n$$

$$\alpha_i = \frac{E\left[x\pi_i^2(x)\right]}{E\left[\pi_i^2(x)\right]}, \quad \beta_{i+1} = \frac{E\left[\pi_{i+1}^2(x)\right]}{E\left[\pi_i^2(x)\right]}, \quad i = 0, 1, \dots, n$$



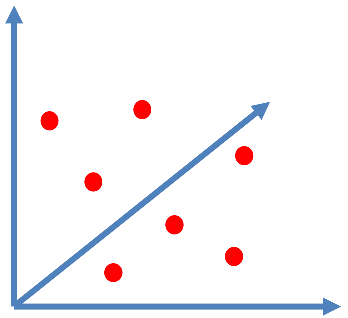


Pre-processing: How to get coefficients

$$y(\xi) \approx \sum_{|\alpha|=0}^{P} c_\alpha \Psi_\alpha(\xi)$$

To estimate coefficients, some testing samples are needed to calculate numerical integration.

$$c_\alpha = \mathrm{E}\left[y(\xi)\,\Psi_\alpha(\xi)\right] \approx \sum_{i=1}^{M} y\left(\boxed{\xi_i}\right)\Psi_\alpha\left(\boxed{\xi_i}\right)\boxed{w_i}$$

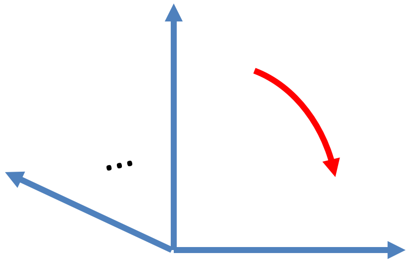


How to determine efficient & mixed-type samples?

Pre-processing: MIP-based Quadrature

Orthogonality → Formulate an optimization problem:

$$\min_{\bar{\xi},\mathbf{w}} \quad \|\mathbf{\Phi}(\bar{\xi})\mathbf{w} - \boldsymbol{e}\|_2^2, \quad \text{s.t.} \quad \mathbf{w} \geq 0,\ \bar{\xi}\mathcal{I} \in \mathbb{Z}^{M|\mathcal{I}|}.$$



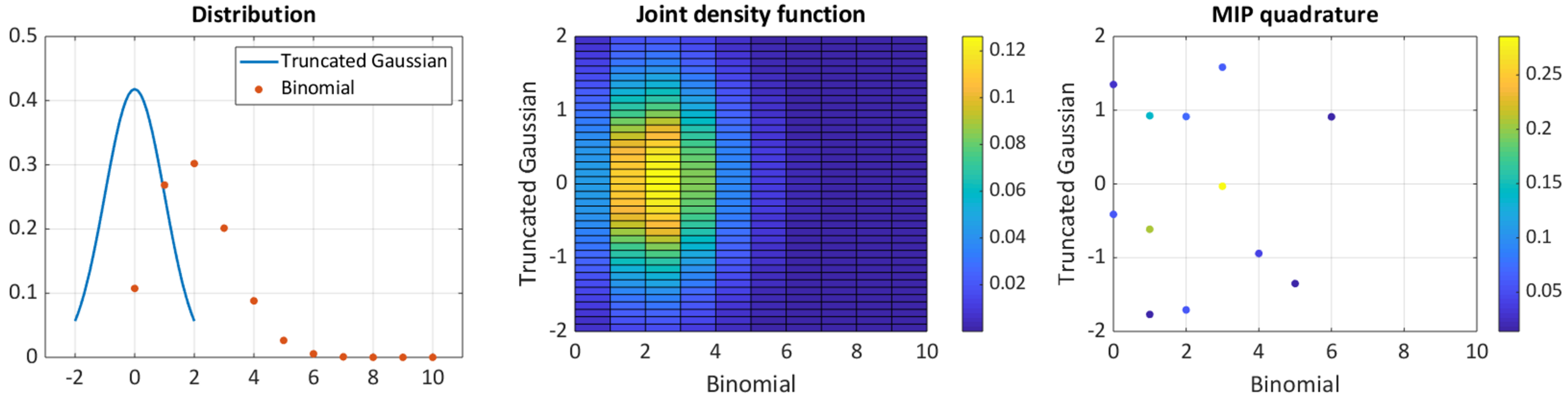
$$= \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Challenges:

1. Large-scale: M x (d+1) unknown
2. Nonlinear: High order polynomial
3. Mixed-Integer constraints (unavoidable obviously)

We proposed a MIP-based solver to handle the former two!

Pre-processing: MIP-based Quadrature



E.g. Two uncertain inputs (one truncated Gaussian & one Binomial), 2th M-gPC order → 12 samples are needed

Pre-processing: MIP-based Quadrature

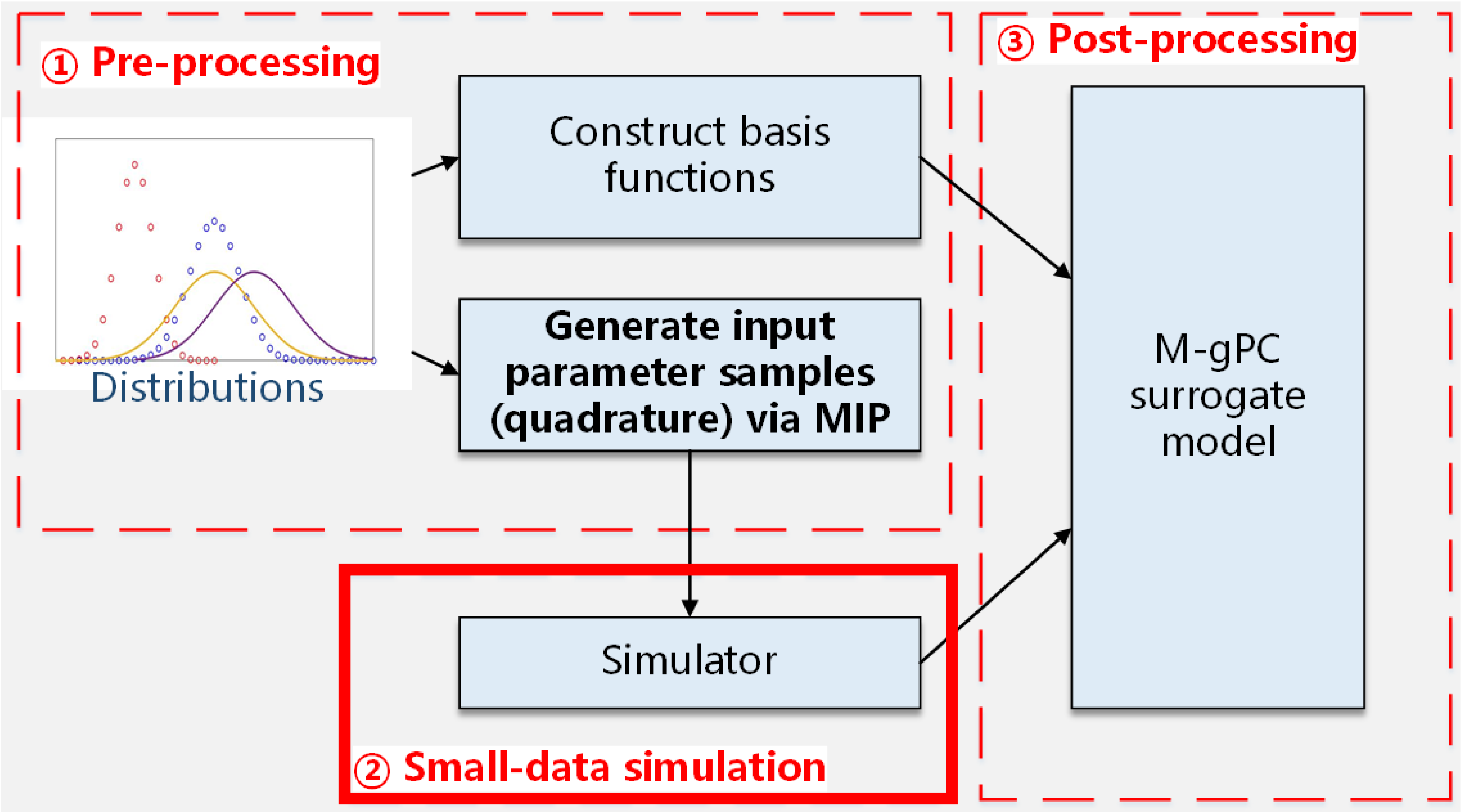
Theoretical guarantee on the # of needed samples:

$$\frac{(d+P)!}{d!P!} \leq \# \leq \frac{(d+2P)!}{d!2P!}$$

Much smaller than $(P+1)^d$ when *d* is large

We also have theoretical guarantee on surrogate approximation, see in [Cui and Zhang, 2018]

Proposed M-gPC framework



M-gPC: Simulation

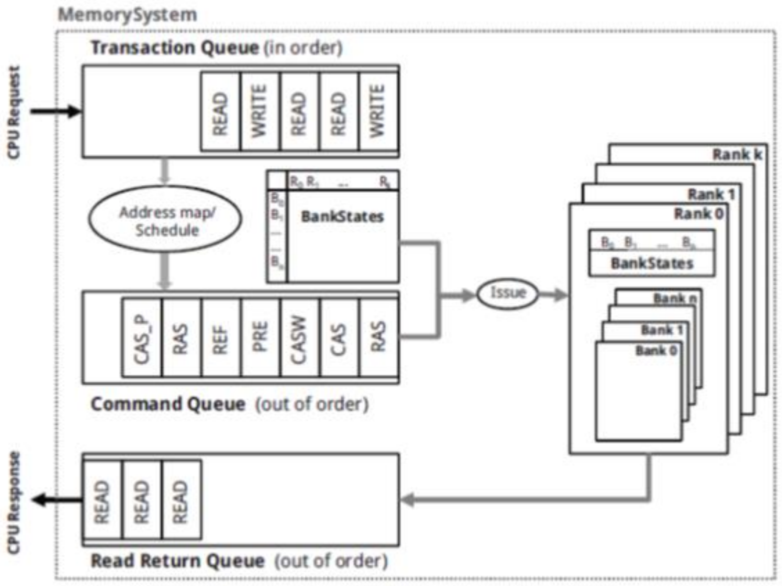
$$c_\alpha = \mathrm{E}\left[y\left(\xi\right)\Psi_\alpha\left(\xi\right)\right] \approx \sum_{i=1}^{M} y\left(\xi_i\right)\Psi_\alpha\left(\xi_i\right) w_i$$

Small-data simulation (M samples)

Selected Samples

$\{\xi_i\}_{i=1}^{M}$

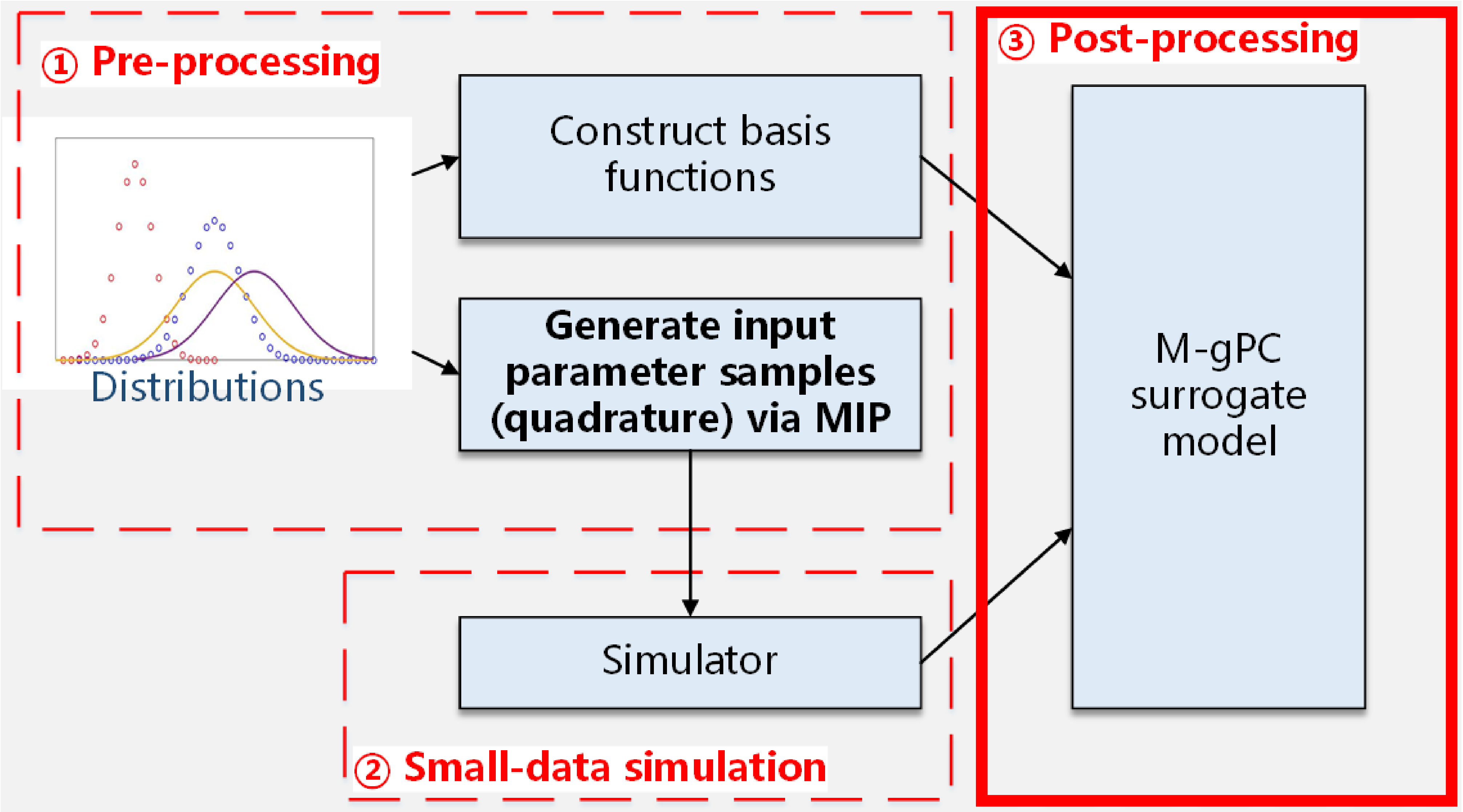
Simulator



Output performance

$\{y\left(\xi_i\right)\}_{i=1}^{M}$

Proposed M-gPC framework



M-gPC: Post-processing

Once M-gPC model is built, output moments are calculated for FREE from the M-gPC coefficients:

$$\mathbb{E}[\mathbf{y}(\xi)] \approx \mathbf{c_0}, \quad \sigma[\mathbf{y}(\xi)] \approx \sqrt{\sum_{|\alpha|=1}^{p} \mathbf{c}_{\alpha}^{2}},$$

Bonus: FREE Sobol global indices based on coefficients

M-gPC: Post-processing

Cheap Monte Carlo simulations → output distribution shape

MC Samples

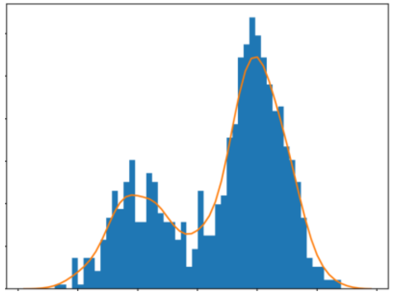
$\{\xi_i\}_{i=1}^{N}$

M-gPC

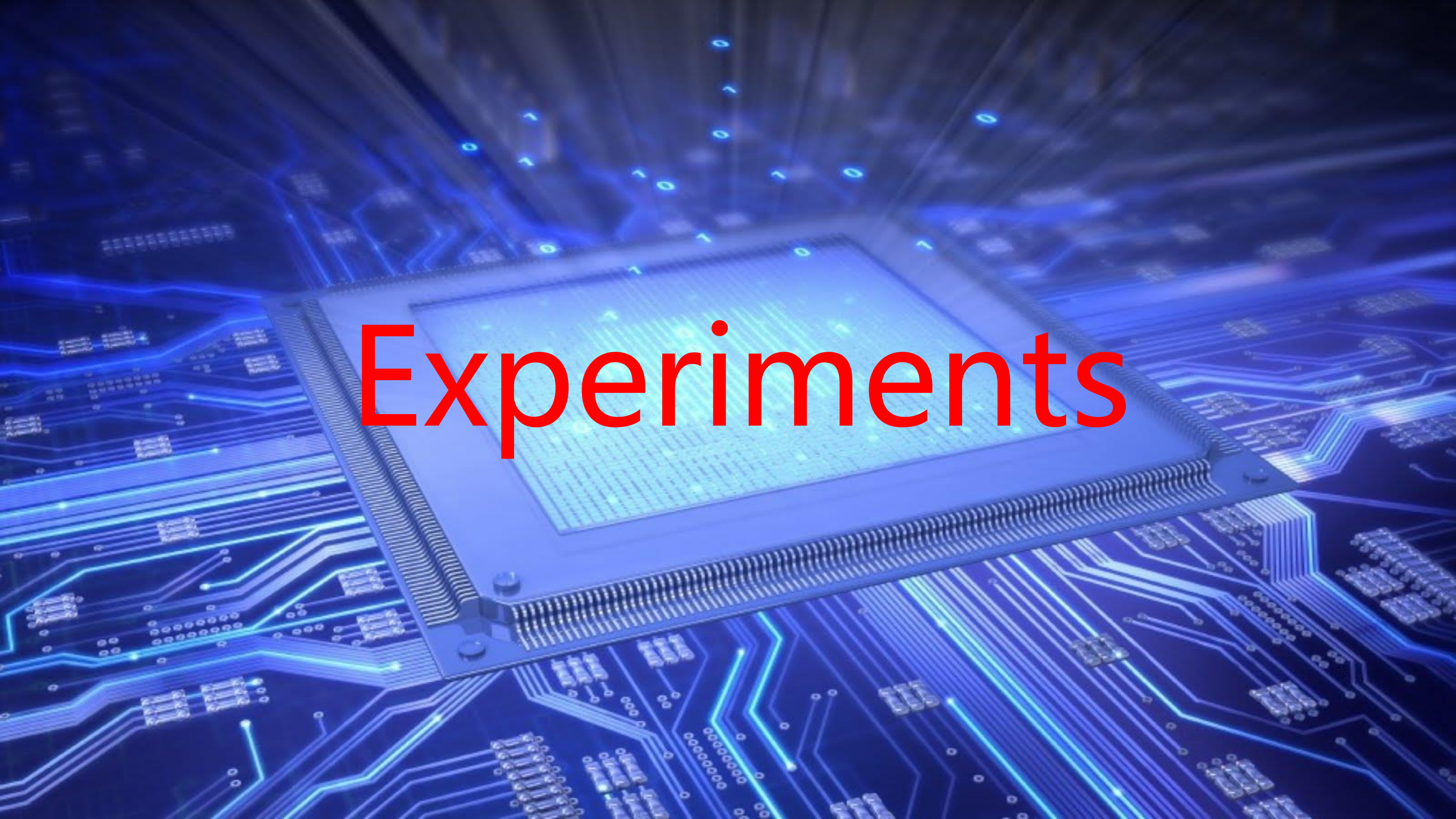
$$y(\xi) \approx \sum_{|\alpha|=0}^{P} c_\alpha \Psi_\alpha$$

Output Histogram

$\{y(\xi_i)\}_{i=1}^{N}$



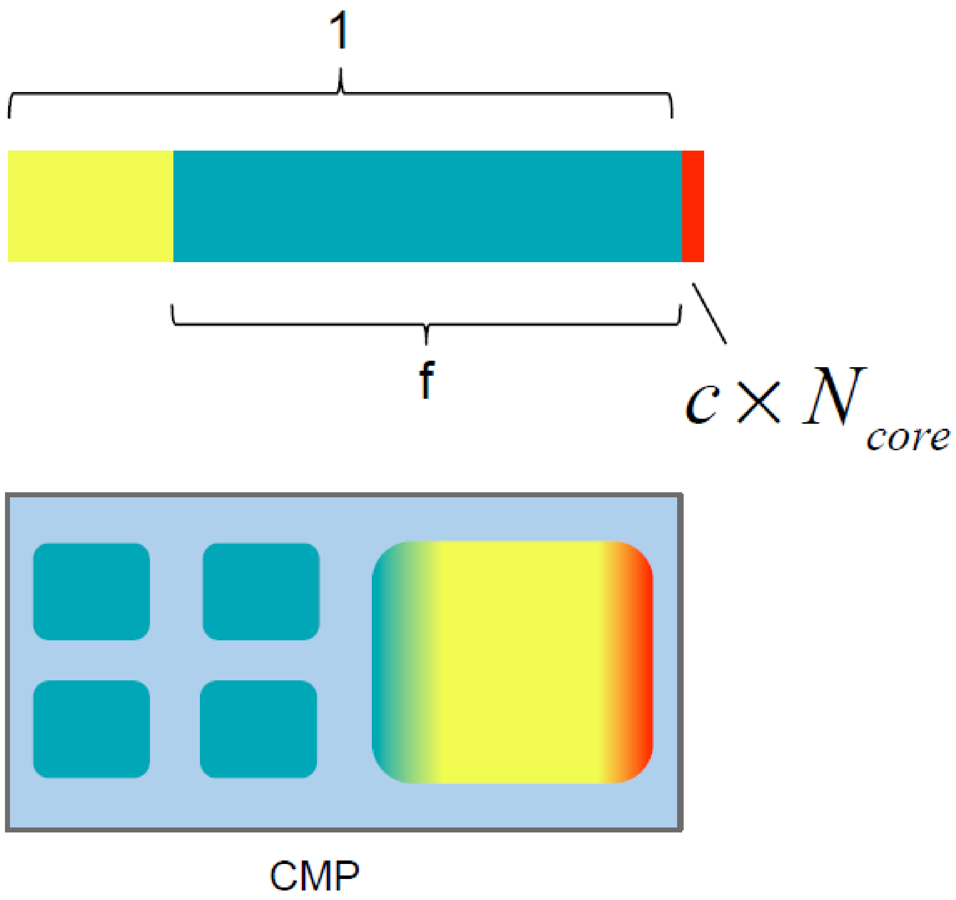
With M-gPC surrogate, no cycle-level simulation any more:
Mins/Hours/Days → Much Less than Seconds!



Experiments

Experiment: Analytical CMP model

Modeling from existing works [Hill and Marty, 2008; Cui and Sherwood, 2017]:



$$\text{Speedup} = 1/\left(T_{\text{sequential}} + T_{\text{parallel}}\right)$$

$$\text{T}_{\text{sequential}} = \left(1 - f + c \times \sum_{i \in \text{core types}} N_{\text{core}_i}\right) / P_{\text{serial}}$$

$$\text{T}_{\text{parallel}} = f / P_{\text{parallel}}$$

$$\text{P}_{\text{serial}} = \max\left\{P_{\text{core}_i} \middle| N_{\text{core}_i} > 0\right\}$$

$$\text{P}_{\text{parallel}} = \sum_{i \in \text{core types}} N_{\text{core}_i} \times P_{core_i}$$

$$\text{P}_{\text{core}_i} = \sqrt{A_{\text{core}_i}}$$

$$\text{A}_{\text{total}} = \sum_{i \in \text{core types}} N_{\text{core}_i} \times A_{\text{core}_i}$$

Experiment: Analytical CMP model

Uncertain inputs	Meaning
$f \sim \frac{\text{Binomial}(M,p)}{M}$	Inputs parallelism of the application
$c \sim \frac{\text{Binomial}(M,p)}{M}$	Communication overhead among cores
$\text{N}_{\text{core}_i} \sim \text{Binomial}\left(M, yield_{\text{core}_i}\right)$	Designed number of each chip
$\text{P}_{\text{core}_i} \sim \text{Truncated Gaussian}\left(\mu, \sigma, 0\right)$	Performance of each core

$$\text{Speedup} = 1/\left(T_{\text{sequential}} + T_{\text{parallel}}\right)$$

$$\text{T}_{\text{sequential}} = \left(1 - f + c \times \sum_{i \in \text{core types}} N_{\text{core}_i}\right)$$

$$\text{T}_{\text{parallel}} = f / P_{\text{parallel}}$$

$$\text{P}_{\text{serial}} = \max\left\{P_{\text{core}_i} \middle| N_{\text{core}_i} > 0\right\}$$

$$\text{P}_{\text{parallel}} = \sum_{i \in \text{core types}} N_{\text{core}_i} \times P_{core_i}$$

$$\text{P}_{\text{core}_i} = \sqrt{A_{\text{core}_i}}$$

$$\text{A}_{\text{total}} = \sum_{i \in \text{core types}} N_{\text{core}_i} \times A_{\text{core}_i}$$

Results: Analytical CMP model

	Sample	Mean	Std	RMSE	MAE	ε
M-gPC	84	0.4353	0.1021	0.0418	0.0311	1e-2
	85	0.4382	0.0974	0.0338	0.0231	1e-3
	87	0.4380	0.0992	0.0306	0.0208	1e-4
	95	0.4376	0.0986	0.0306	0.0228	1e-5
	123	0.4376	0.0987	0.0314	0.0233	1e-6
	179	0.4386	0.0982	0.0289	0.0205	1e-7
	182	0.4387	0.0975	0.0294	0.0214	1e-8
MC	1e3	0.4369	0.1011	N/A	N/A	N/A
	5e3	0.4370	0.1002			
	1e4	0.4383	0.0995			
	5e4	0.4375	0.099			
	1e5	0.4377	0.0987			

More than 800 times speedup

Experiment: DRAM subsystem

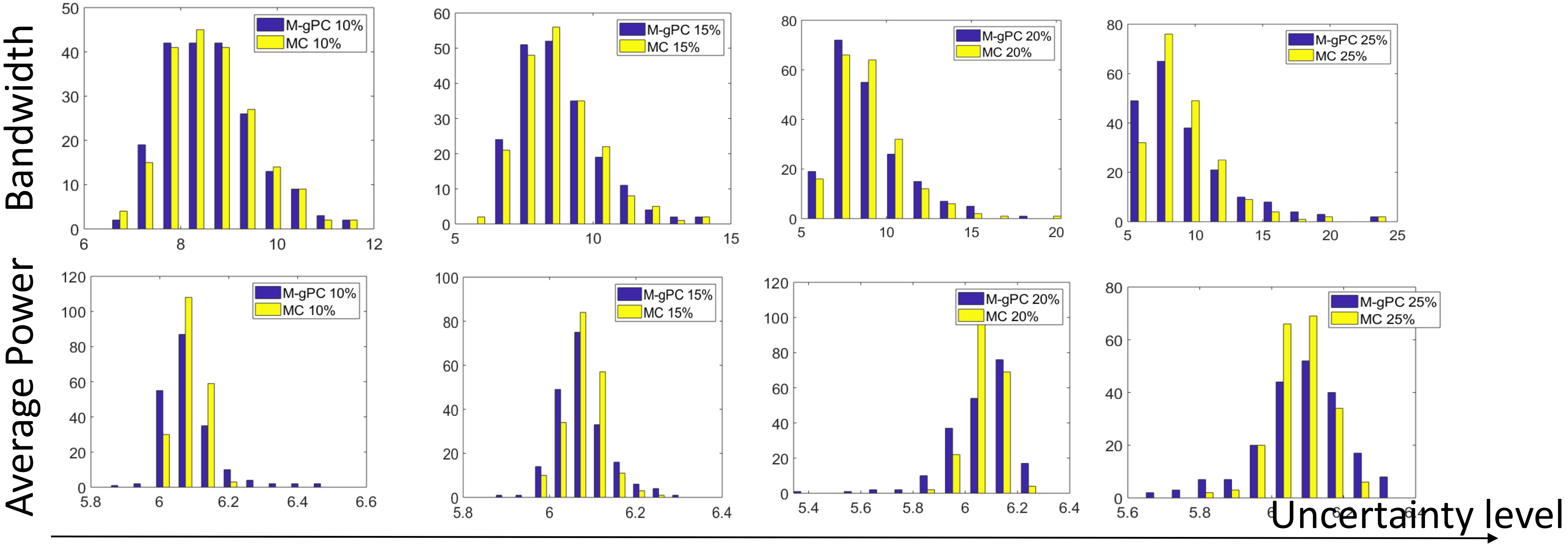
Uncertain inputs	Meaning
$\text{tCK} \sim \text{Truncated Gaussian}\,(\mu, \sigma, 0)$	One tick of Clock
$\text{tRCD} \sim \text{Binomial}\,(M, p)$	Clock cycles between active and read/write
$\text{tCL} \sim \text{Binomial}\,(M, p)$	Clock cycles of read delay
$\text{tRP} \sim \text{Binomial}\,(M, p)$	Clock cycles between pre-charge and active
$\text{tWR} \sim \text{Binomial}\,(M, p)$	Clock cycles between write and pre-charge

Setup: DRAMSim2 Simulator; Output: Bandwidth & Average Power

Experiments different uncertainty levels, configurations & workloads

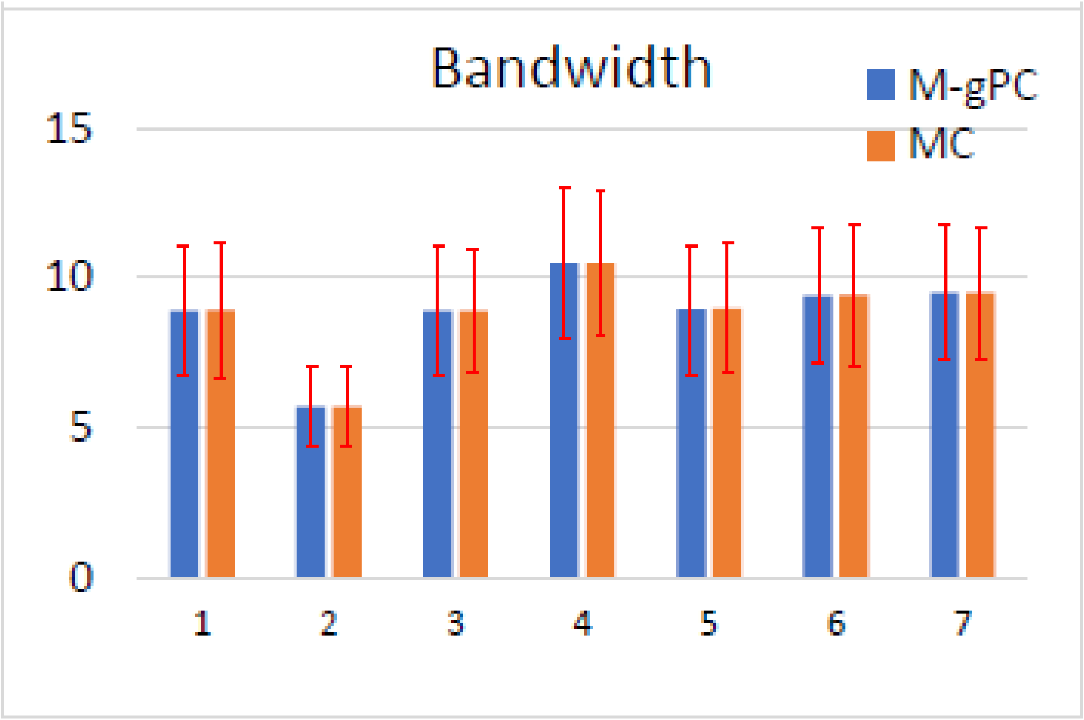
DRAM Results: different uncertainty levels

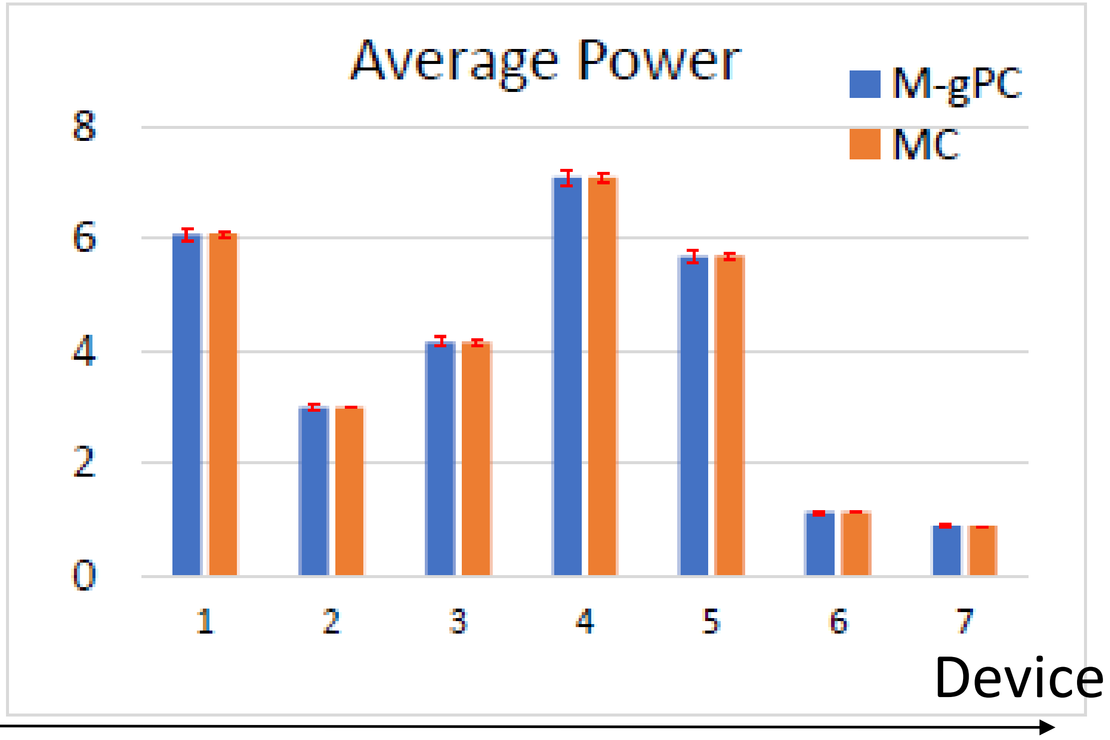
Higher uncertainty levels α, higher standard deviation σ : $\sigma = \alpha \times \mu$



Approximation is more accurate under less uncertainty. For larger uncertainty, we can increase M-gPC order.

DRAM Results: different configs





Moments are estimated accurately

RMSE varies in 1%-4%, MAE varies in 0.8%-2.4%

DRAM Results: different workloads

	Workloads	peribench	gcc	mcf	xalancbmk	x264	deepsjeng	leela	specrand
Bandwidth (GB/s)	Mean (M-gPC)	7.2847	8.9289	6.4874	7.8169	7.3144	7.1424	7.1413	7.2011
	Mean (MC)	7.2834	8.9332	6.4868	7.8201	7.3146	7.1434	7.1422	7.2047
	Std (M-gPC)	1.7545	2.0969	1.5718	1.8619	1.747	1.7194	1.7142	1.7318
	Std (MC)	1.7875	2.1479	1.6026	1.9002	1.7794	1.7538	1.7503	1.772
	RMSE	0.0318	0.0322	0.0319	0.0321	0.032	0.0319	0.0319	0.0319
	MAE	0.0186	0.0185	0.0185	0.0185	0.0187	0.0185	0.0185	0.0187
Average Power (watts)	Mean (M-gPC)	5.6782	6.0804	5.3127	5.7486	5.3717	5.5257	5.4308	5.5089
	Mean (MC)	5.6796	6.0831	5.3149	5.7517	5.3731	5.5281	5.4333	5.5112
	Std (M-gPC)	0.2159	0.1362	0.2168	0.1832	0.1871	0.2073	0.1934	0.2113
	Std (MC)	0.18	0.0817	0.1858	0.1428	0.1486	0.1732	0.1605	0.1773
	RMSE	0.0123	0.0124	0.0121	0.0122	0.0123	0.0121	0.0121	0.0125
	MAE	0.0094	0.0096	0.0093	0.0093	0.0094	0.0093	0.0093	0.0096

Moments are all well captured with small RMSE & MAE

DRAM Results: time on different workloads

Workload	Length of trace	MC time	M-gPC time
600.peribench	46.8M	∼15.1h	∼4.5h + 12.5m
602.gcc	35.7M	∼8.8h	∼2.6h + 12.5m
605.mcf	43.5M	∼14.7h	∼4.4h + 12.5m
623.xalancbmk	42.9M	∼12.6h	∼3.8h + 12.5m
625.x264	30.5M	∼9h	∼2.7h + 12.5m
631.deepsjeng	37.6M	∼12.3h	∼3.7h + 12.5m
641.leela	36.1M	∼11.6h	∼3.5h + 12.5m
998.specrand	32.9M	∼10.6h	∼3.2h + 12.5m

3-4 times speedup, will be larger for more accurate MC simulation

High accuracy MC is too expensive

Low accuracy MC simulations need much more samples to achieve the similar M-gPC accuracy

Take-home message

Uncertainty in architecture design is important

M-gPC surrogate model for expensive cycle-level simulator: much less samples and mixed-type sampling:

- Model speedup: 800x in an analytical example.
- In DRAM, a few samples to get accurate statistical information, while MC is impossible

Thank you!
Questions?