Polynomial Chaos for Variability Assessment of Electronic and Microwave Designs

Flavio CANAVERO & Paolo MANFREDI

Politecnico di Torino, Italy

flavio.canavero@polito.it

Ack

Paolo Manfredi, Flavio G. Canavero

Politecnico di Torino, Italy

&

Dries Vande Ginste, **Daniël De Zutter**

[2]Universiteit Gent, Belgium

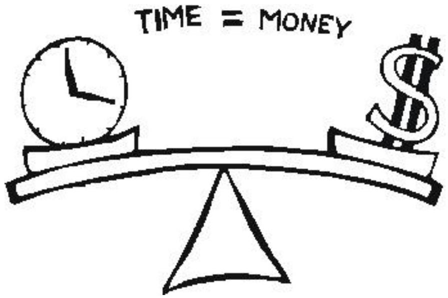
dries.vande.ginste@intec.ugent.be

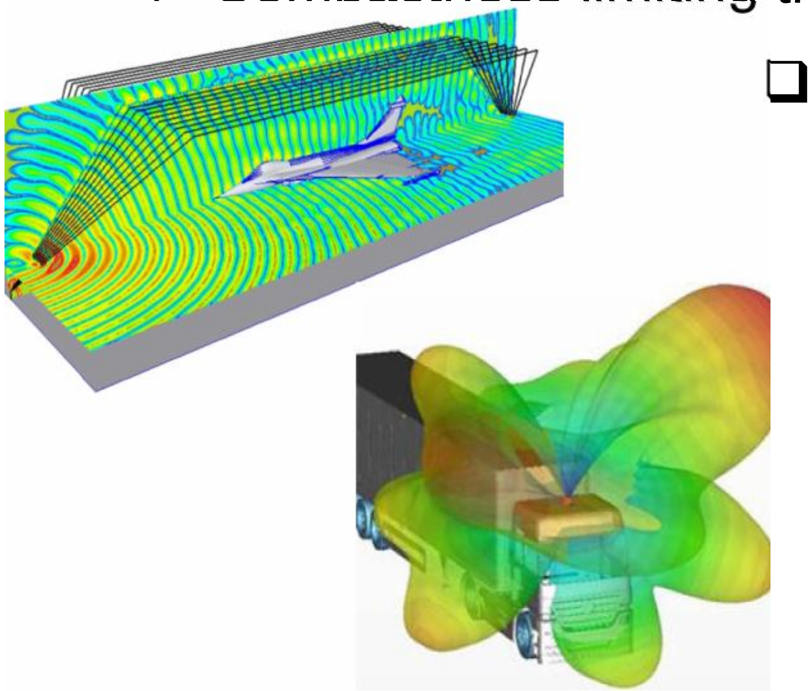
Outline

- Motivation and State of the art
- Polynomial Chaos approach
    - Differential Equations
    - Linear Algebraic Equations
    - Nonlinear Equations
- The challenge related to the large number of varying parameters
- Conclusions

General motivation

- ❑ The design of electronic devices undergoes three **major constraints**:
  - ❖ Budget
  - ❖ Time-to-market
  - ❖ Compactness limiting the design tuning





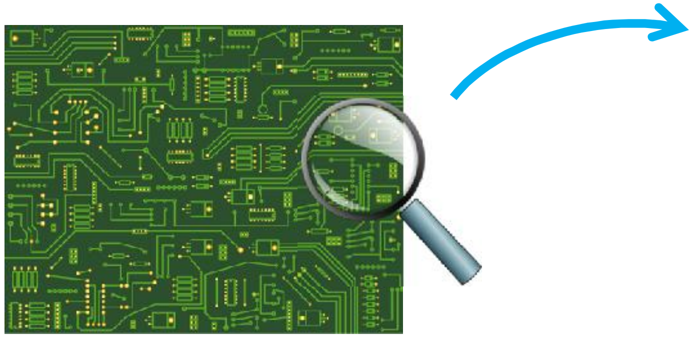
- ❑ Simulation tools help engineers to perform **right-the-first-time designs**:
  - ❖ avoidance of re-fabrication
  - ❖ minimization of measurements

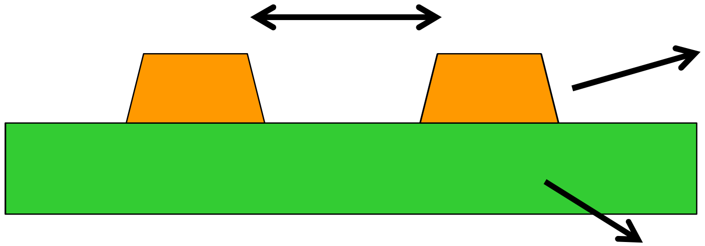
⇩

Time and money saving

Interconnect variability

- ❑ The manufacturing process introduces **variability** in the geometrical and material properties of interconnects



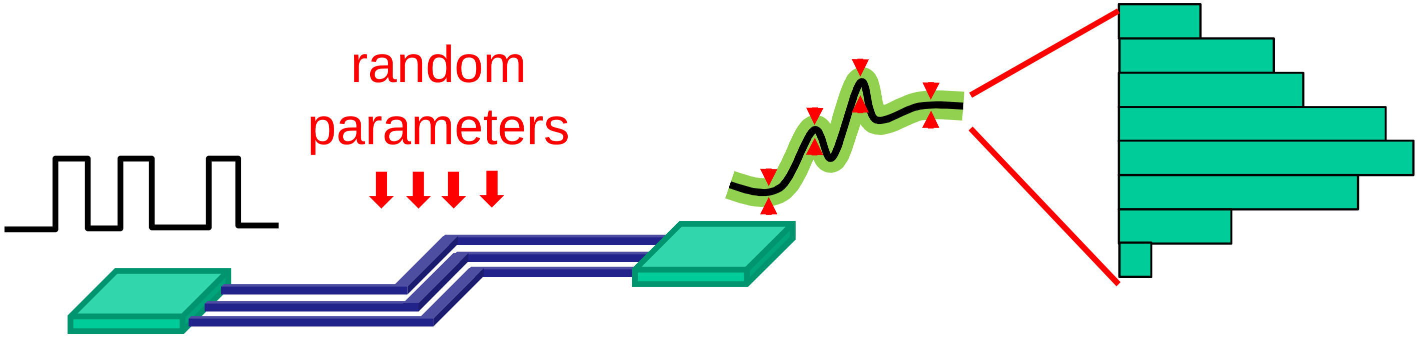


Photolitograpy: trace separation

Etching: edge profile

Impurities: permittivity, loss tangent, etc.

- ❑ Deterministic excitations produce **stochastic responses**



The Monte Carlo method

- ❑ Interconnect designers need to perform statistical simulations for **variation-aware verifications**

- ❑ Virtually all available **commercial design software** relies on the Monte Carlo method

  - ❖ Robust and easy to implement ☺

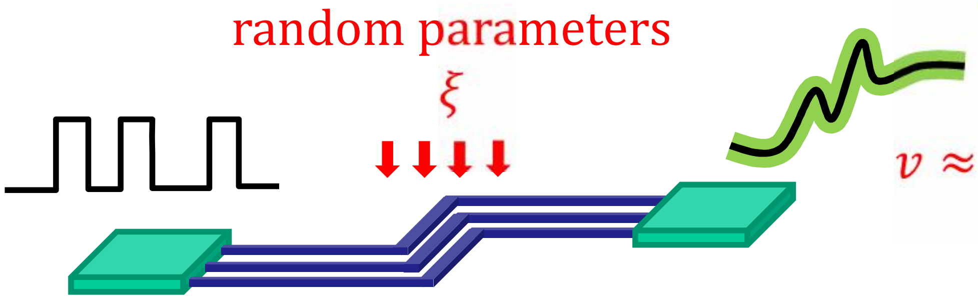
  - ❖ Time consuming (slow convergence) ☹

Outline

- Motivation and State of the art
- **Polynomial Chaos approach**
  - **Differential Equations**
  - **Linear Algebraic Equations**
  - **Nonlinear Equations**
- The challenge related to the large number of varying parameters
- Conclusions

The Polynomial Chaos (PC) approach

- ❑ Expand line voltages and currents in terms of polynomial chaos expansions



deterministic coefficients
(to be determined)

$$v \approx v_0\varphi_0(\xi) + v_1\varphi_1(\xi) + v_2\varphi_2(\xi) + \cdots$$

orthonormal polynomials

- ❑ Polynomials are **orthonormal** w.r.t. probability distribution of parameters

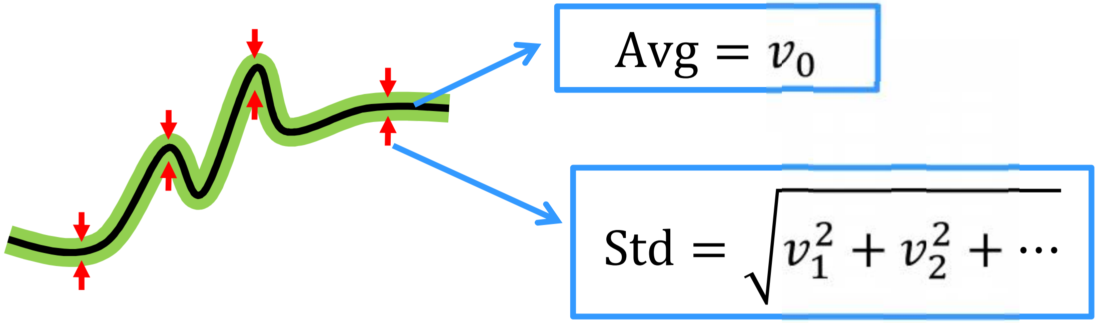
$$\langle \varphi_k, \varphi_j \rangle = \int \varphi_k(\xi)\varphi_j(\xi)w(\xi)d\xi = \begin{cases} = 1 & \text{if } \; k = j \\ = 0 & \text{otherwise} \end{cases}$$

E.g, Gaussian: $w(\xi) = \frac{1}{\sqrt{2}} e^{-\xi^2/2}$ ⇨ $\{\varphi_k\}$: Hermite polynomials

$$\varphi_0 = 1$$
$$\varphi_1 = \xi$$
$$\varphi_2 = (\xi^2 - 1)/\sqrt{2}$$

PC: Statistical information

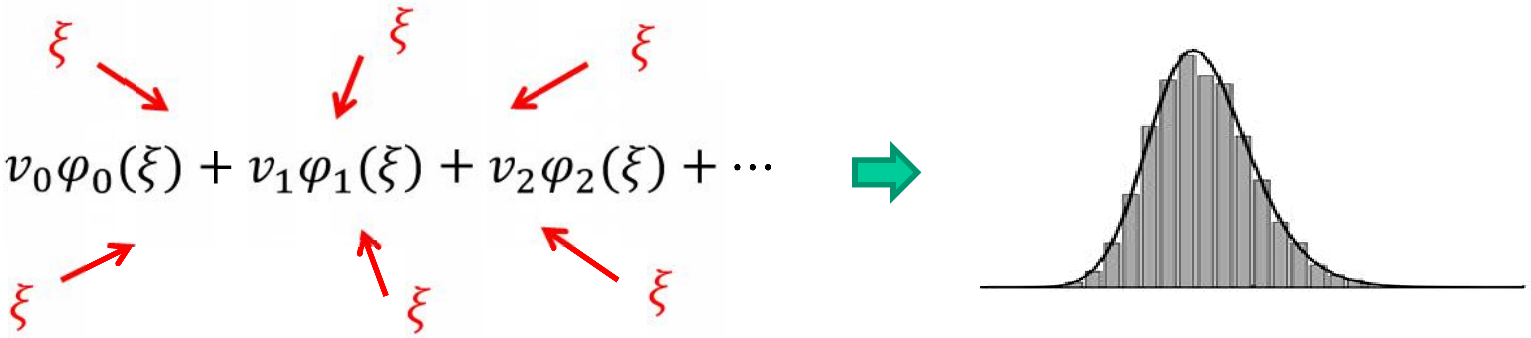
- ❑ **Statistical information** is retrieved from the expansion
- ❑ Average and standard deviation are readily given:



$$\text{Avg} = v_0$$

$$\text{Std} = \sqrt{v_1^2 + v_2^2 + \cdots}$$

- ❑ Other moments, distribution functions, quantiles, etc. can be obtained by randomly sampling the expansion



$$v_0\varphi_0(\xi) + v_1\varphi_1(\xi) + v_2\varphi_2(\xi) + \cdots$$

PC: Multiple random variables and multivariate bases

- ❑ Orthogonal **multivariate basis** built using products of univariate functions

Number of variables

$$¤_k(¶_1,¶_2,\dots,¶_n)=\prod_{i=1}^{n}\underbrace{¤_{k_i}(¶_i)}_{\text{univariate basis}},\qquad \sum_{i=1}^{n}k_i\le p$$

order of expansion

Inner product readily extends

- ❑ For example bivariate Hermite basis:

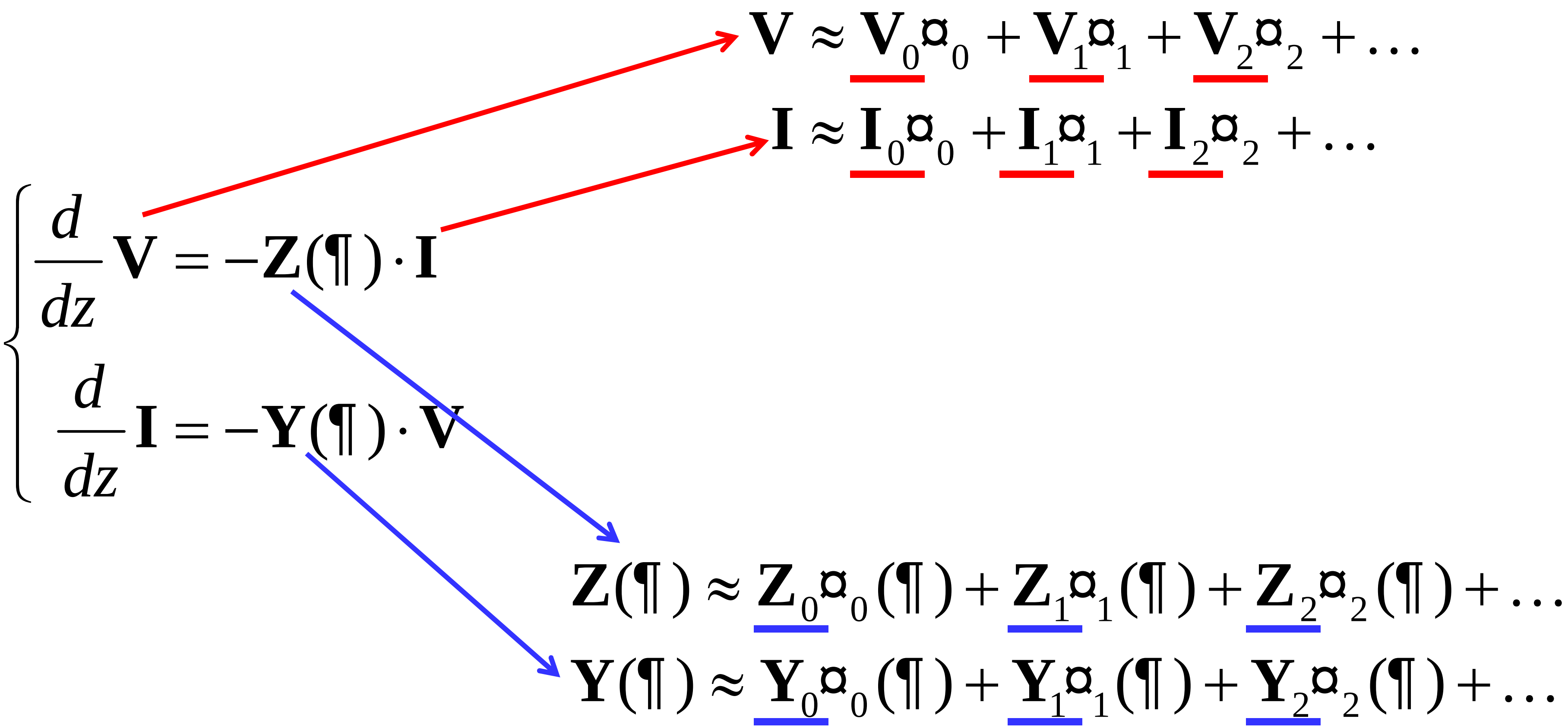
index k	order p	Basis function $¤_k(¶_1,¶_2)$
0	0	$¤_0=1$
1	1	$¤_1=¶_1$
2		$¤_2=¶_2$
3	2	$¤_3=¶_1^2-1$
4		$¤_4=¶_1¶_2$
5		$¤_5=¶_2^2-1$

- ❑ Number of terms:

$$\frac{(p+n)!}{p!n!}$$

PC for systems governed by Differential Equations

- **<u>Step 1</u>**: represent the random per-unit-length parameters inside the governing equations in terms polynomial chaos expansions



$$\mathbf{V} \approx \underline{\mathbf{V}_0}\text{¤}_0 + \underline{\mathbf{V}_1}\text{¤}_1 + \underline{\mathbf{V}_2}\text{¤}_2 + \ldots$$

$$\mathbf{I} \approx \underline{\mathbf{I}_0}\text{¤}_0 + \underline{\mathbf{I}_1}\text{¤}_1 + \underline{\mathbf{I}_2}\text{¤}_2 + \ldots$$

$$\begin{cases} \dfrac{d}{dz}\mathbf{V} = -\mathbf{Z}(\text{¶})\cdot\mathbf{I} \\ \dfrac{d}{dz}\mathbf{I} = -\mathbf{Y}(\text{¶})\cdot\mathbf{V} \end{cases}$$

$$\mathbf{Z}(\text{¶}) \approx \underline{\mathbf{Z}_0}\text{¤}_0(\text{¶}) + \underline{\mathbf{Z}_1}\text{¤}_1(\text{¶}) + \underline{\mathbf{Z}_2}\text{¤}_2(\text{¶}) + \ldots$$

$$\mathbf{Y}(\text{¶}) \approx \underline{\mathbf{Y}_0}\text{¤}_0(\text{¶}) + \underline{\mathbf{Y}_1}\text{¤}_1(\text{¶}) + \underline{\mathbf{Y}_2}\text{¤}_2(\text{¶}) + \ldots$$

PC for Differential Equations: Galerkin weighting

- **Step 2**: project equations onto each basis function

$$\langle \cdot, \text{¤}_0 \rangle \quad \frac{d}{dz}\mathbf{V}_0 \underbrace{\langle \varphi_0, \varphi_0 \rangle}_{=1} + \frac{d}{dz}\mathbf{V}_1 \cancel{\langle \varphi_1, \varphi_0 \rangle} + \ldots = -(\mathbf{Z}_0\mathbf{I}_0 \underbrace{\langle \varphi_0 \varphi_0 \varphi_0 \rangle}_{=1}$$

$$+\mathbf{Z}_0\mathbf{I}_1 \underbrace{\cancel{\langle \varphi_0, \varphi_1, \varphi_0 \rangle}}_{=0} + \mathbf{Z}_1\mathbf{I}_0 \underbrace{\cancel{\langle \varphi_1, \varphi_0, \varphi_0 \rangle}}_{=0} + \mathbf{Z}_1\mathbf{I}_1 \underbrace{\langle \varphi_1 \varphi_1 \varphi_0 \rangle}_{=1} + \ldots$$

$$\frac{d}{dz}\mathbf{V}_0 = -(\mathbf{Z}_0\mathbf{I}_0 + \mathbf{Z}_1\mathbf{I}_1 + \ldots$$

$$\langle \cdot, \text{¤}_1 \rangle \quad \frac{d}{dz}\mathbf{V}_0 \cancel{\langle \varphi_0, \varphi_1 \rangle} + \frac{d}{dz}\mathbf{V}_1 \underbrace{\langle \varphi_1, \varphi_1 \rangle}_{=1} + \ldots = -(\mathbf{Z}_0\mathbf{I}_0 \overbrace{\cancel{\langle \varphi_0, \varphi_0, \varphi_1 \rangle}}^{=0}$$

$$+\mathbf{Z}_0\mathbf{I}_1 \underbrace{\langle \varphi_0, \varphi_1, \varphi_1 \rangle}_{=1} + \mathbf{Z}_1\mathbf{I}_0 \underbrace{\langle \varphi_1, \varphi_0, \varphi_1 \rangle}_{=1} + \mathbf{Z}_1\mathbf{I}_1 \underbrace{\cancel{\langle \varphi_1, \varphi_1, \varphi_1 \rangle}}_{=0} + \ldots$$

$$\frac{d}{dz}\mathbf{V}_1 = -(\mathbf{Z}_0\mathbf{I}_1 + \mathbf{Z}_1\mathbf{I}_0 + \ldots$$

PC for Differential Equations: deterministic eq’s

- Repeating the procedure for all the basis functions…

$$\begin{cases} \frac{d}{dz}\mathbf{V}_0 = -(\mathbf{Z}_0\mathbf{I}_0 + \mathbf{Z}_1\mathbf{I}_1 + \dots \\ \frac{d}{dz}\mathbf{V}_1 = -(\mathbf{Z}_0\mathbf{I}_1 + \mathbf{Z}_1\mathbf{I}_0 + \dots \\ \vdots \\ \frac{d}{dz}\mathbf{V}_P = \dots \end{cases}$$

P+1 eqs. for P+1 unknowns

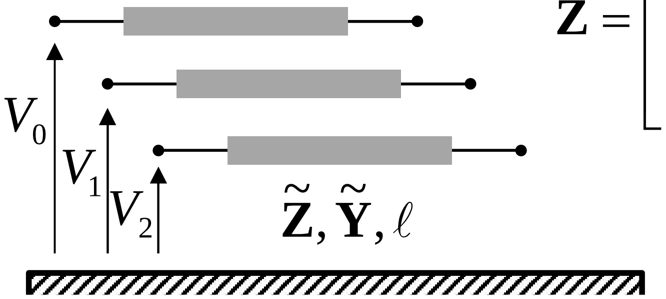
$$\frac{d}{dz}\tilde{\mathbf{V}} = -\tilde{\mathbf{Z}}\cdot\tilde{\mathbf{I}}$$

new unknowns (coefficients)

- Applying also to the second transmission-line equation

Deterministic system!!

$$\begin{cases} \frac{d}{dz}\tilde{\mathbf{V}} = -\tilde{\mathbf{Z}}\cdot\tilde{\mathbf{I}} \\ \frac{d}{dz}\tilde{\mathbf{I}} = -\tilde{\mathbf{Y}}\cdot\tilde{\mathbf{V}} \end{cases}$$



$$\tilde{\mathbf{Z}} = \begin{bmatrix} \mathbf{Z}_0 & \mathbf{Z}_1 & 2\mathbf{Z}_2 \\ \mathbf{Z}_1 & (\mathbf{Z}_0 + 2\mathbf{Z}_2) & 2\mathbf{Z}_1 \\ \mathbf{Z}_2 & \mathbf{Z}_1 & (\mathbf{Z}_0 + 4\mathbf{Z}_2) \end{bmatrix}$$

non-symmetric!

PC for Differential Equations: orthonormalization

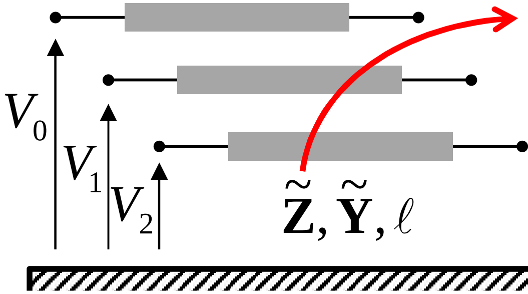
- ❑ In practice, commercial solvers only support **reciprocal lines**, i.e. with **symmetric** augmented matrices
- ❑ **Re-normalize** the basis functions so that $\boxed{\left\langle ¤_k , ¤_k \right\rangle \equiv 1}$

PC for Differential Equations: orthonormalization

- ❑ In practice, commercial solvers only support **reciprocal lines**, i.e. with **symmetric** augmented matrices
- ❑ **Re-normalize** the basis functions so that $\boxed{\langle \text{¤}_k , \text{¤}_k \rangle \equiv 1}$
- ❑ E.g., **orthonormal** Hermite polynomials for Gaussian variability:

$$\text{¤}_k(\text{¶}) = \frac{(-1)^k}{\sqrt{k!}} e^{\text{¶}^2/2} \frac{d^k}{d\text{¶}^k} e^{-\text{¶}^2/2}$$

Galerkin projection with $<\phi_k,\phi_k>\equiv 1$



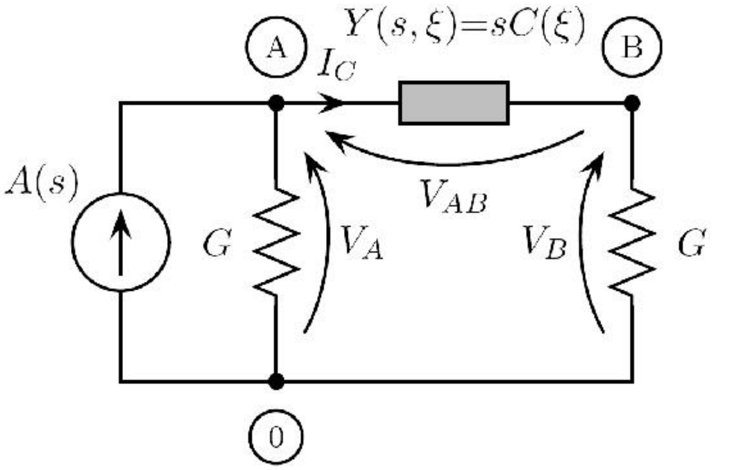
$$\tilde{\mathbf{Z}} = \begin{bmatrix} \mathbf{Z}_0 & \mathbf{Z}_1 & \mathbf{Z}_2 \\ \mathbf{Z}_1 & (\mathbf{Z}_0 + \sqrt{2}\mathbf{Z}_2) & \sqrt{2}\mathbf{Z}_1 \\ \mathbf{Z}_2 & \sqrt{2}\mathbf{Z}_1 & (\mathbf{Z}_0 + 2\sqrt{2}\mathbf{Z}_2) \end{bmatrix}$$

symmetric!!!

A small step for polynomial chaos, a giant leap for SPICE implementation

PC for systems governed by Linear Algebraic Equations

- Polynomial chaos approach can be applied also to lumped elements. For example, RC circuit with random capacitance in Laplace domain:



$$I_C(s,\P) = sC(\P)V_{AB}(s,\P)$$

- **Step 1**: expand governing equations in terms of **orthogonal polynomials** $\text{¤}_k(\P)$ (optimal choice depends on distribution). E.g., 1st order expansion:

known coefficients (related to the statistics of *C*)

unknown coefficients (to be determined)

$$\underline{I_{C0}(s)}\,\underline{\text{¤}_0(\P)} + \underline{I_{C1}(s)}\,\underline{\text{¤}_1(\P)} =$$

$$= s[\underline{C_0}\,\underline{\text{¤}_0(\P)} + \underline{C_1}\,\underline{\text{¤}_1(\P)}][\underline{V_{AB0}(s)}\,\underline{\text{¤}_0(\P)} + \underline{V_{AB1}(s)}\,\underline{\text{¤}_1(\P)}]$$

PC for Linear Equations: Galerkin weighting

- **Step 2**: **Galerkin projection** on ¤_0 and ¤_1 leads to two new equations:

$$<\text{¤}_j,\text{¤}_k>=\begin{cases}\sim_k & \text{if} \quad j=k\\ 0 & \text{if} \quad j\neq k\end{cases} \quad \Rightarrow \quad \begin{cases} I_{C0}(s)=sC_0V_{AB0}(s)+sC_1V_{AB1}(s)\\ I_{C1}(s)=sC_1V_{AB0}(s)+sC_0V_{AB1}(s)\end{cases}$$

- In matrix form:

$$\underbrace{\begin{bmatrix} I_{C0}(s)\\ I_{C1}(s)\end{bmatrix}}_{\tilde{\mathbf{I}}_C(s)} = s\underbrace{\begin{bmatrix} C_0 & C_1\\ C_1 & C_0\end{bmatrix}}_{\tilde{\mathbf{C}}}\underbrace{\begin{bmatrix} V_{AB0}(s)\\ V_{AB1}(s)\end{bmatrix}}_{\tilde{\mathbf{V}}_{AB}(s)}$$

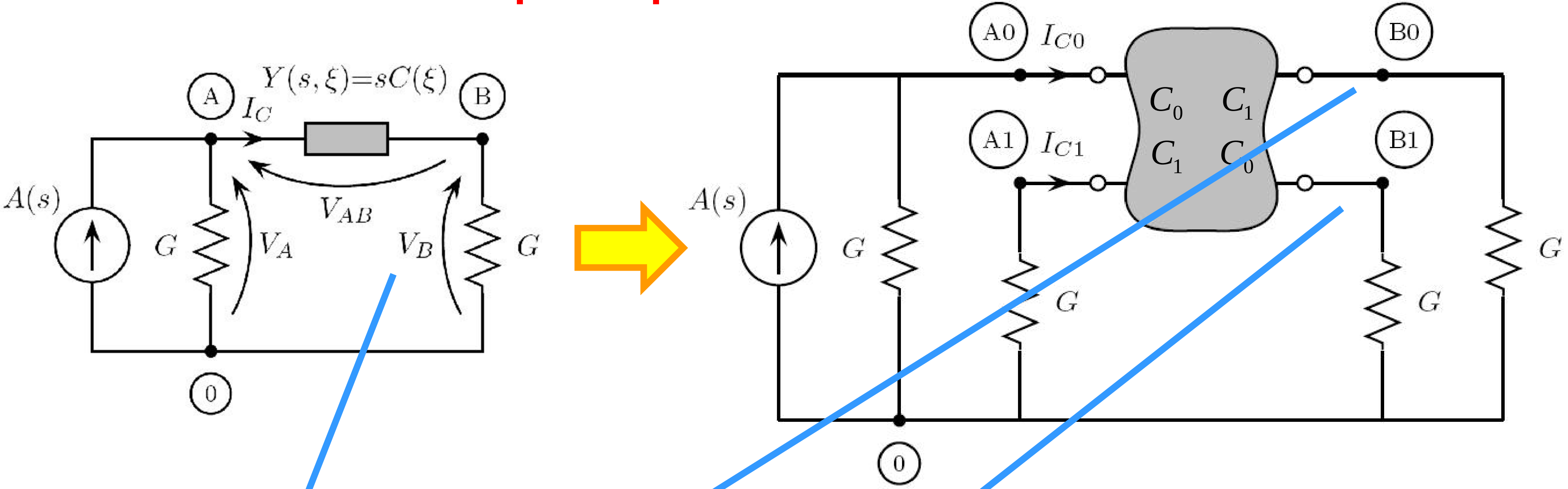
Dependence on ¶ removed due to integration procedure

↓

new equations are deterministic!!

PC for Linear Equations: augmented eq’s

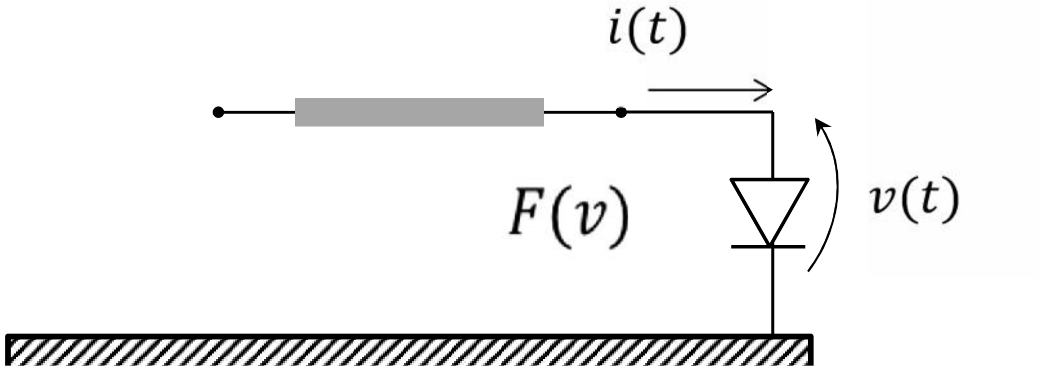
- Repeating the procedure also for non-stochastic elements leads to a set of **deterministic multiport equations**:



- **Solution** of the augmented circuit gives polynomial chaos **coefficients** for the unknown variables. E.g.,

$$V_B(s,¶) \cong V_{B0}(s)¤_0(¶)+V_{B1}(s)¤_1(¶)$$

PC for systems governed by NonLinear Equations



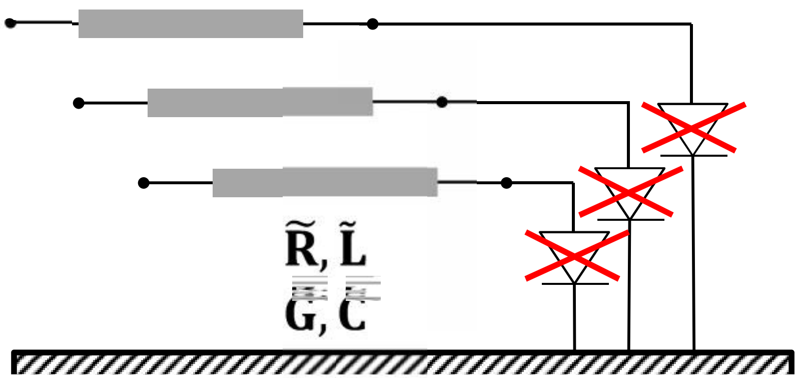
$$\boxed{i(t) = F(v(t))}$$

- ❑ **<u>Step 1</u>**: Expansion

$$i_0\varphi_0 + i_1\varphi_1 + \cdots = F(v_0\varphi_0 + v_1\varphi_1 + \cdots)$$

- ❑ **<u>Step 2</u>**: (Galerkin projection $< \cdot, \varphi_m >$

$$i_m(t) = \int F\,F(v_0\varphi_0(\xi) + v_1\varphi_1(\xi) + \cdots)\,\varphi_m(\xi)w(\xi)d\xi$$


$\tilde{\mathbf{R}}, \tilde{\mathbf{L}}$
$\tilde{\mathbf{G}}, \tilde{\mathbf{C}}$

No closed form!!

PC for NonLinear Equations: Quadrature

- Idea ☼: discretize the above integral using a **quadrature rule** [*]

$$a_{mq} \triangleq \varphi_m(\xi_q)$$

quadrature points

$$i_m(t) = \int F\big(v_0\varphi_0(\xi) + v_1\varphi_1(\xi) + \cdots\big)\,\varphi_m(\xi)w(\xi)d\xi \;\approx\; \sum_{q=1}^{Q} F\big(v_0 a_{0q} + v_1 a_{1q} + \cdots\big)a_{mq}w_q$$

quadrature weights

Deterministic close-form equation

- The approach is approximate but high accuracy with **low number of points** can be achieved by using **Gaussian quadratures, e.g.**

Variability	Basis functions	Quadrature rule
Gaussian	Hermite polynomials	Gauss-Hermite
uniform	Legendre polynomials	Gauss-Legendre

[*] A. Biondi, D. Vande Ginste, D. De Zutter, P. Manfredi, and F.G. Canavero, *IEEE Trans. CPMT*, Jul. 2013

PC Summary

pre-proc.

- **Project** the system matrices onto a basis of orthogonal polynomials
- Use the expansion coefficients to **build** the “augmented” MTL system

solution

- **Solve** (once → faster) the obtained deterministic system, thus finding the expansion coefficients $\mathbf{V}_k$ and $\mathbf{I}_k$

post-proc.

- Use the polynomial chaos expansion to **extract** statistical information. E.g., for one Gaussian random variable:

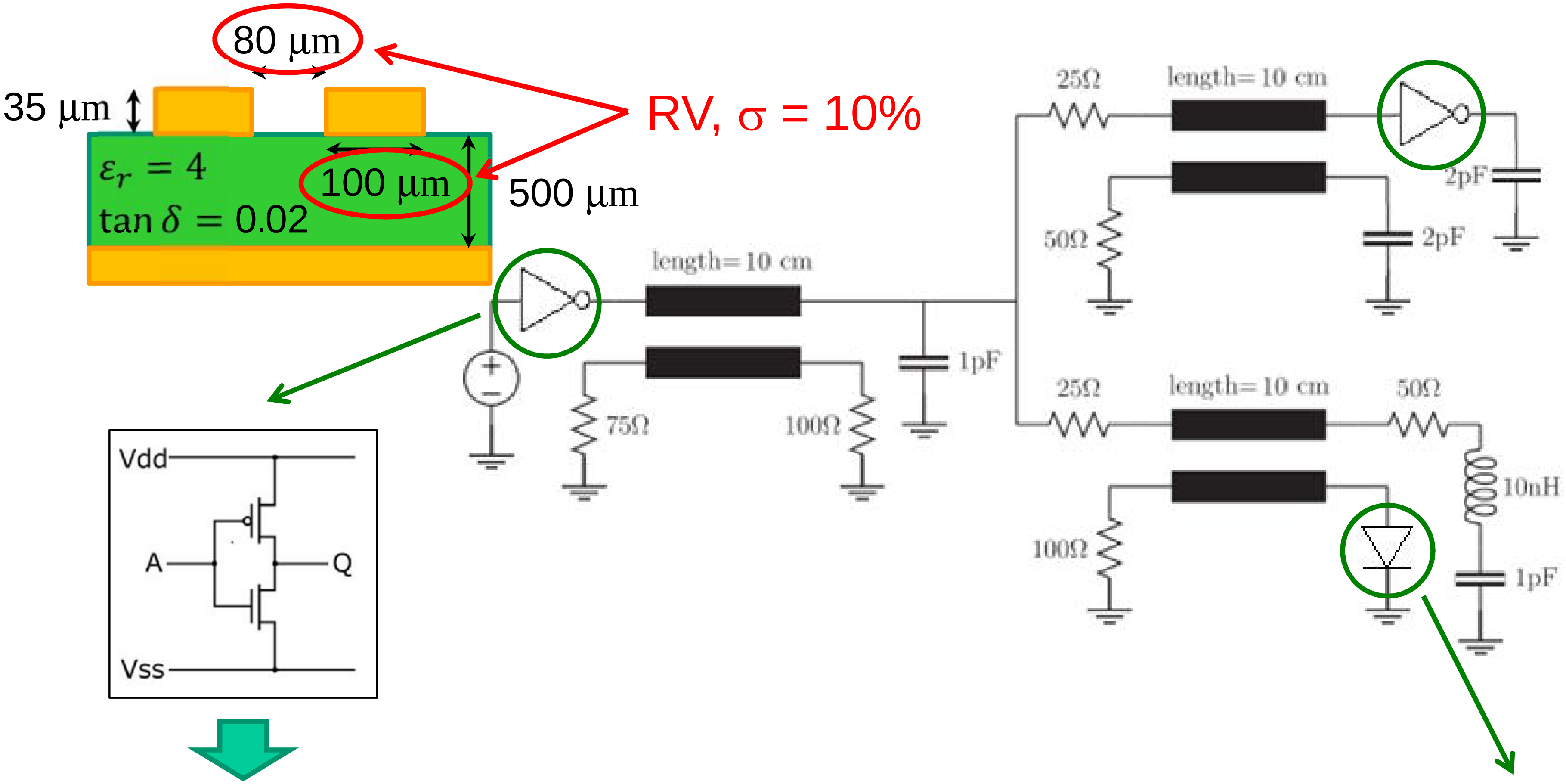
$$\mathbf{V}(\mu,\P) \approx \sum_{k=0}^{P} \mathbf{V}_k(\mu)\text{¤}_k(\P) = \underline{\mathbf{V}_0(\mu)}\cdot\underline{1} + \underline{\mathbf{V}_1(\mu)}\cdot\underline{\P} + \underline{\mathbf{V}_2(\mu)}\cdot\underline{(\P^2-1)} + \ldots$$

deterministic solution

Hermite polynomials

This process can be easily automated!!

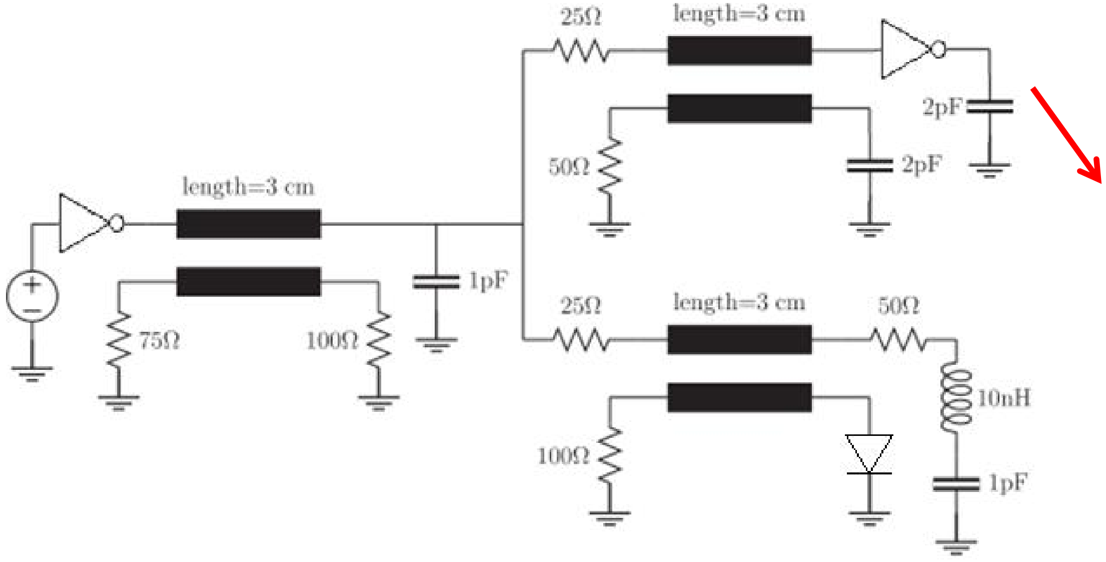
Example: lines+drivers+diode

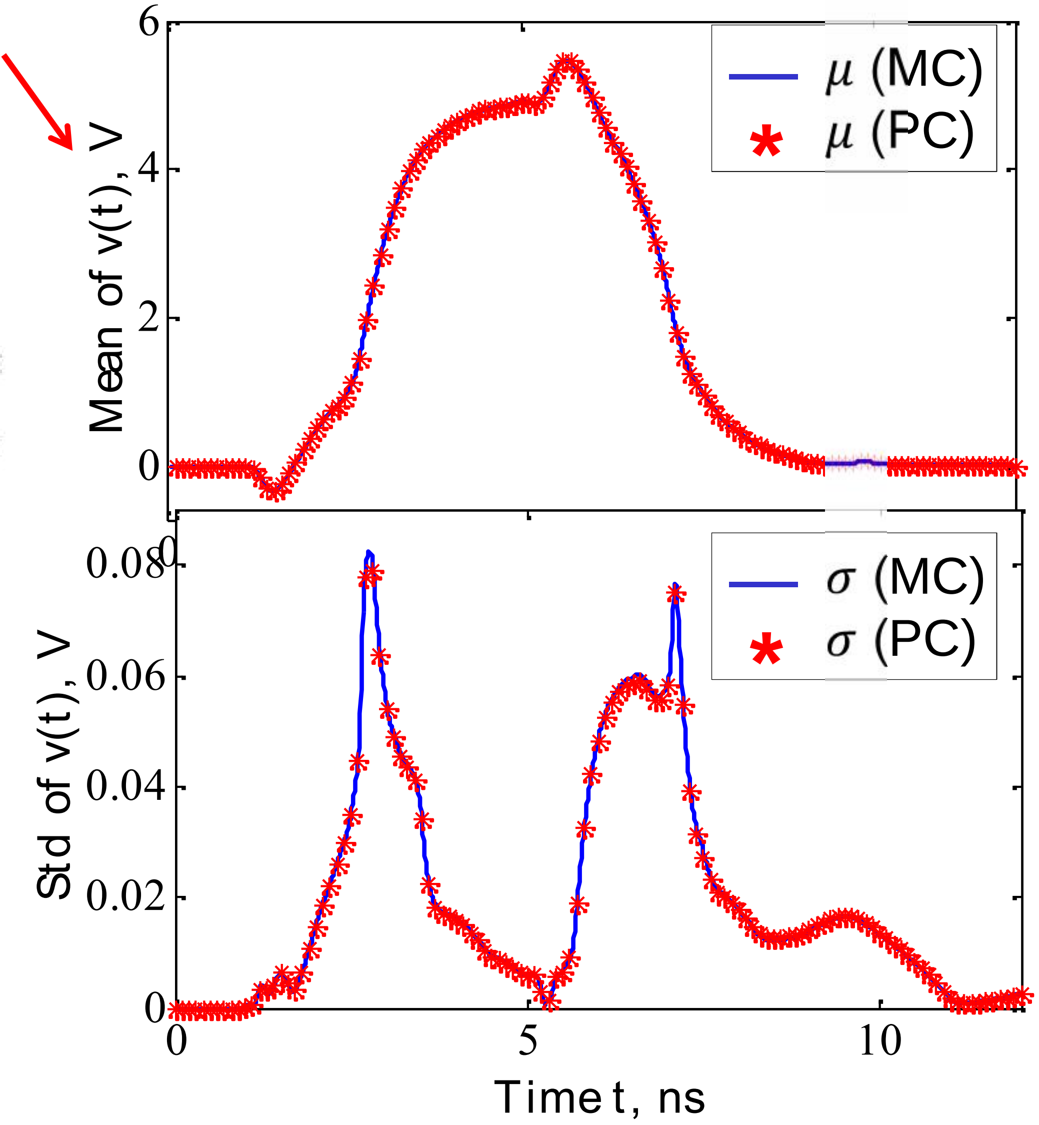


```
.MODEL nch NMOS LEVEL=2
+ CGDO=4.3e-10 CGSO=4.3e-10 CJ=0.3e-3 CJSW=8e-10 MJ=0.66
+ MJSW=0.24 PB=0.58 TOX= 200e-10 PHI=0.6 DELTA=2.34
+ LAMBDA=0.037 GAMMA=0.54 LD=0.15U NSUB=5.37e15 VTO=0.74
+ KP=8e-5 UO=656 UEXP=0.157 UCRIT=31444 VMAX=55261
+ XJ=0.25U NFS=1e12 NEFF=1.001 NSS=1e11 TPG=1 RSH=70
```

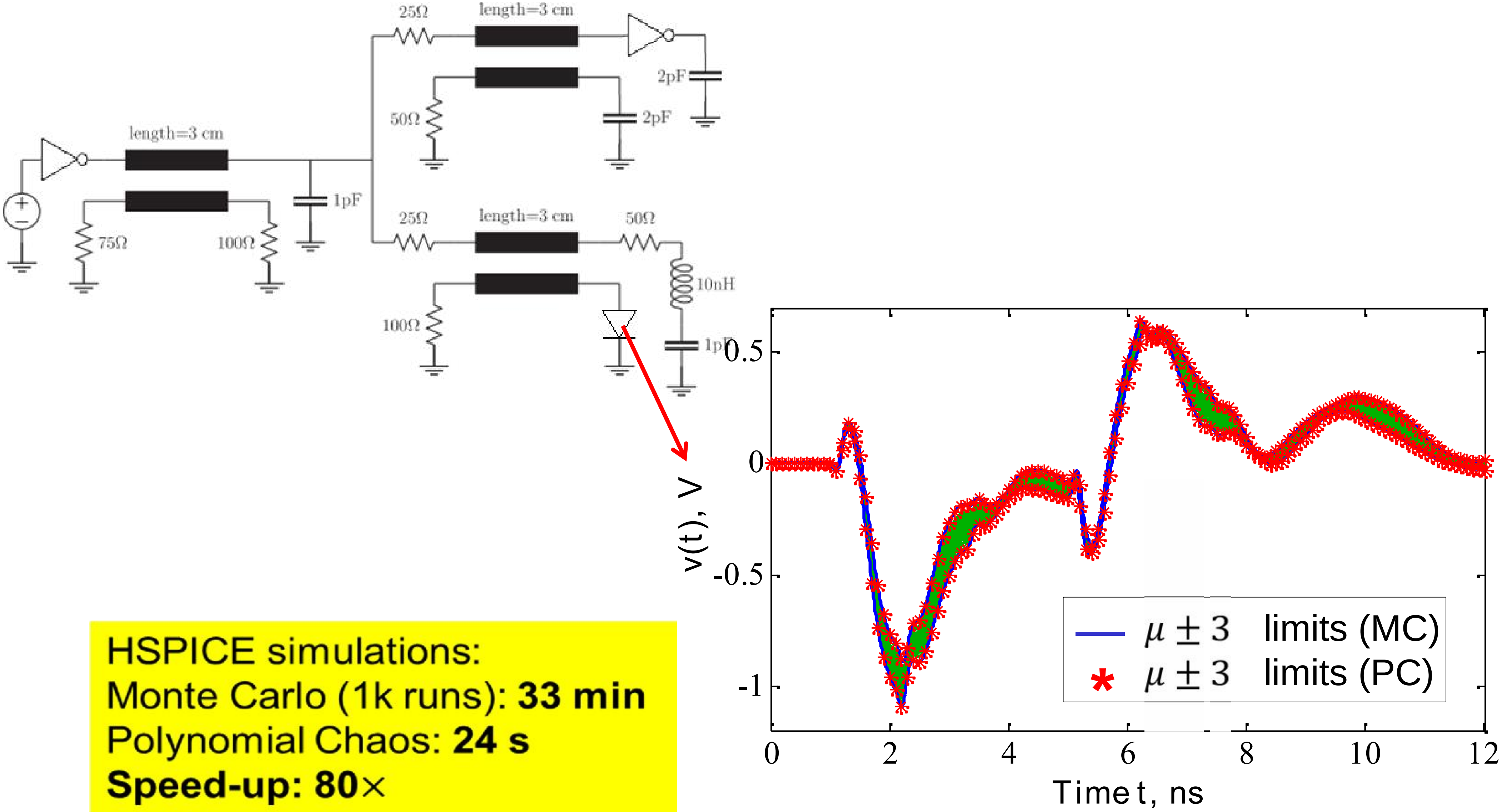
```
.MODEL dmod D
+ CJO=2p
+ RS=1
+ IS=0.05p
```

Example: results *(i)*





Example: results *(ii)*



HSPICE simulations:
Monte Carlo (1k runs): **33 min**
Polynomial Chaos: **24 s**
Speed-up: 80×

Outline

- Motivation and State of the art
- Polynomial Chaos approach
  - Differential Equations
  - Linear Algebraic Equations
  - Nonlinear Equations
- **The challenge related to the large number of varying parameters**
- Conclusions

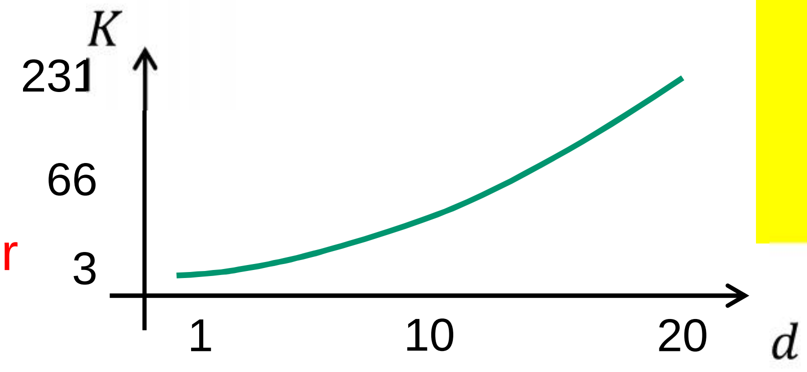
PC limitation: size of the augmented system

- ❑ The augmented deterministic equations are **coupled** and their size K increases in particular with the number of random parameters

$$K = \frac{(p+d)!}{p!\,d!}$$

# of random parameters

expansion order (typ. p = 2)



Efficiency reduced for many random parameters

Is it possible to ***decouple*** *such equations?*

Alternative solution: a) “point matching”

- ❑ **Step a1**: Take the equations with expanded voltages and currents. E.g., the voltage equation (with $K = 2$ for simplicity)

$$\frac{d}{dz}V_0\varphi_0(\xi) + \frac{d}{dz}V_1\varphi_1(\xi) = -Z(\xi)(I_0\varphi_0(\xi) + I_1\varphi_1(\xi))$$

- ❑ **Step a2**: Suppose a set of *K* **match points** (here ξ_0 and ξ_1) be available **for the random parameters** and force the equations to be satisfied at these match points

$$\frac{d}{dz}V_0 a_{00} + \frac{d}{dz}V_1 a_{01} = -Z_0(I_0 a_{00} + I_1 a_{01})$$

$$\frac{d}{dz}V_0 a_{10} + \frac{d}{dz}V_1 a_{11} = -Z_1(I_0 a_{10} + I_1 a_{11})$$

$a_{mk} = \varphi_k(\xi_m)$: polynomial k evaluated at match point m

$Z_m = Z(\xi_m)$: per-unit-length impedance at match point m

deterministic equations!

Alternative solution: b) “decoupling”

- Define the following **transformation** for the voltages and currents

$$\begin{cases} U_0 = V_0 a_{00} + V_1 a_{01} \\ U_1 = V_0 a_{10} + V_1 a_{11} \end{cases} \Rightarrow \begin{bmatrix} U_0 \\ U_1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} V_0 \\ V_1 \end{bmatrix}$$

$$\begin{cases} J_0 = I_0 a_{00} + I_1 a_{01} \\ J_1 = I_0 a_{10} + I_1 a_{11} \end{cases} \Rightarrow \begin{bmatrix} J_0 \\ J_1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} I_0 \\ I_1 \end{bmatrix}$$

$$\boxed{\mathbf{A} = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}}$$

- The matched equations are **uncoupled** with respect to the new variables…

$$\boxed{\begin{cases} \dfrac{d}{dz} U_0 = -Z_0 J_0 \\ \dfrac{d}{dz} J_0 = -Y_0 V_0 \end{cases}} \qquad \boxed{\begin{cases} \dfrac{d}{dz} U_1 = -Z_1 J_1 \\ \dfrac{d}{dz} J_1 = -Y_1 V_1 \end{cases}}$$

… and can therefore be **solved independently**!!!

An iterative and non-intrusive procedure

- ❑ Once the uncoupled responses have been obtained, the **classical** voltage and current coefficients are retrieved via **inverse transformation**

$$\begin{bmatrix} V_0 \\ V_1 \end{bmatrix} = \mathbf{A}^{-1} \begin{bmatrix} U_0 \\ U_1 \end{bmatrix} \qquad \begin{bmatrix} I_0 \\ I_1 \end{bmatrix} = \mathbf{A}^{-1} \begin{bmatrix} J_0 \\ J_1 \end{bmatrix}$$

- ❑ The approach applies to circuits (also **nonlinear**) with an arbitrary number of expansion terms and to **any distribution type**

- ❑ The procedure is fully **iterative** and **non-intrusive**:

  - **analyze the stochastic interconnect problem for each match point, thus obtaining the uncoupled coefficients**
  - **retrieve the classical polynomial chaos coefficients by applying the inverse transformation**

- ❑ **Any standard circuit simulator** can be called by this procedure

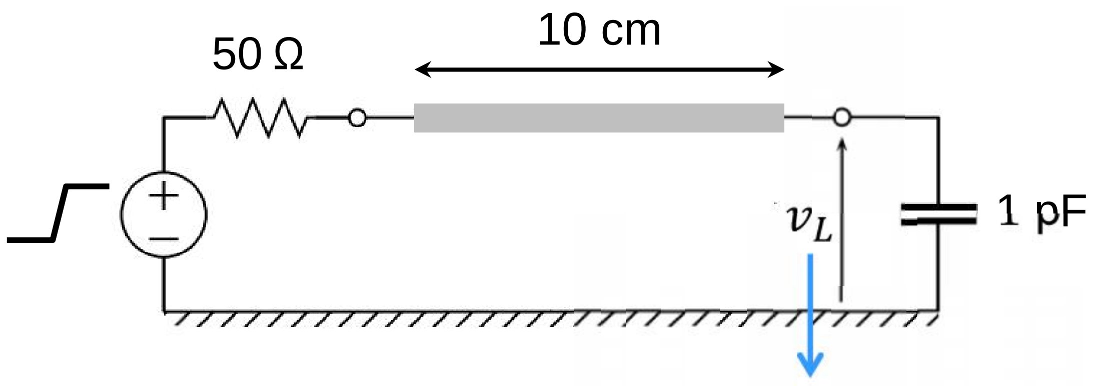
Generation of the match points

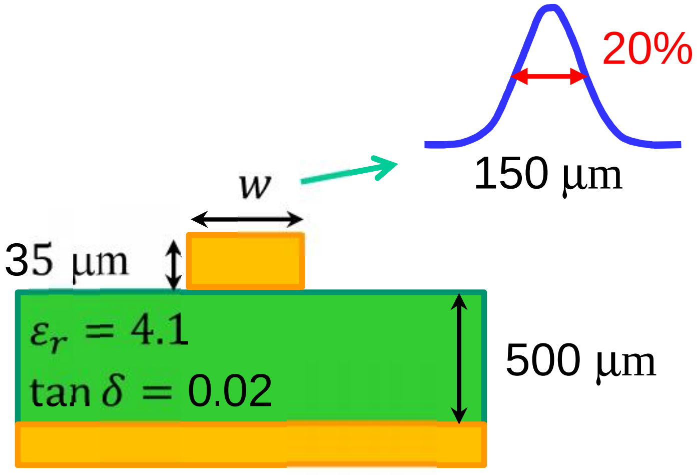
- ❑ The match (or sampling) points for the random parameters are selected according to the ***stochastic testing*** algorithm [2]
- ❑ These are a **subset** of the nodes of a tensor product **Gaussian quadrature rule** (Gauss-Hermite, Gauss-Legendre, depending on the distribution of the parameters…)
- ❑ The first match point is usually $\xi = 0$ (**nominal** value for each parameter)
- ❑ Another quadrature node is included as an additional match point if the corresponding new row of matrix A "has a **large enough component** orthogonal" to the span of the previous rows, until K points are selected
- ❑ Better algorithms might be possibly devised (open issue)…

[2] Z. Zhang, T.A. El-Moselhy, I.M. Elfadel, and L. Daniel, "Stochastic testing method for transistor-level uncertainty quantification based on generalized polynomial chaos," *IEEE Trans. Circuits and Syst.*, Oct. 2013

Decouplig: Tutorial example

- ❑ Microstrip line with **random trace width**





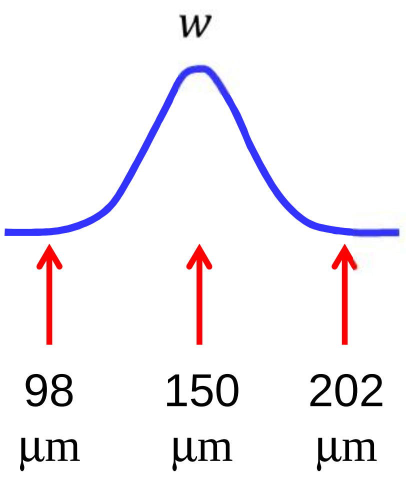
$v_0\varphi_0 + v_1\varphi_1 + v_2\varphi_2$ (expansion with $K = 3$)

- ❑ $K = 3$ sampling points are generated with the stochastic testing algorithm and correspond to **nominal value** and **± 34.6% variations**

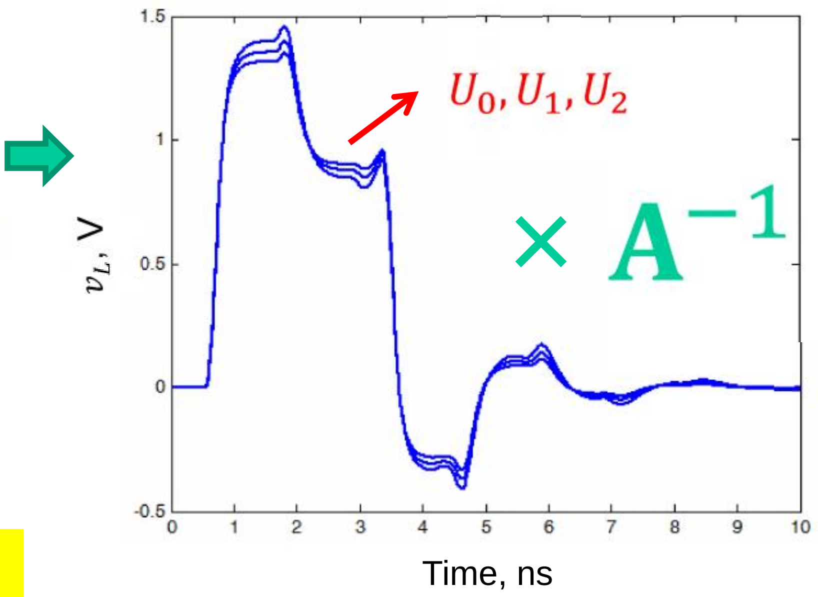
- ❑ The corresponding **transformation matrix** is

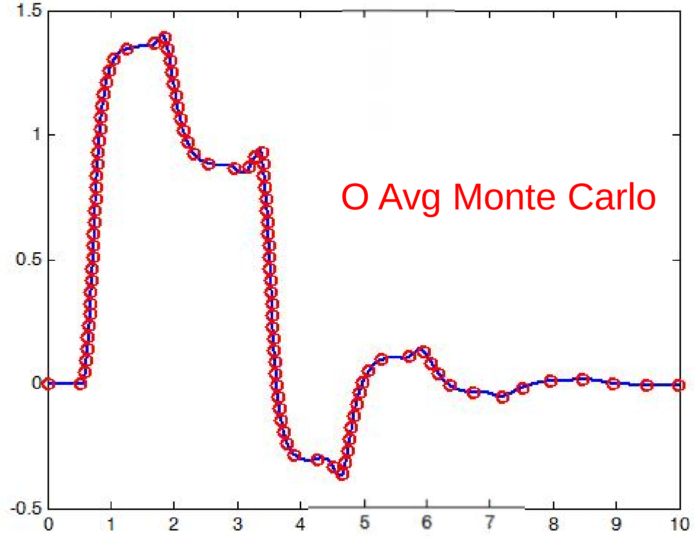
$$\mathbf{A} = \begin{bmatrix} 1 & 0 & -1/\sqrt{2} \\ 1 & -\sqrt{3} & \sqrt{2} \\ 1 & +\sqrt{3} & \sqrt{2} \end{bmatrix}$$

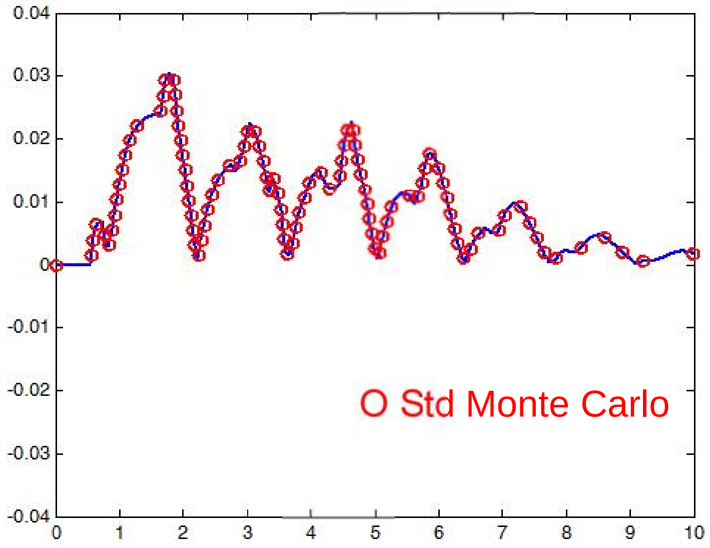
Stochastic testing simulation



simulation at the match points



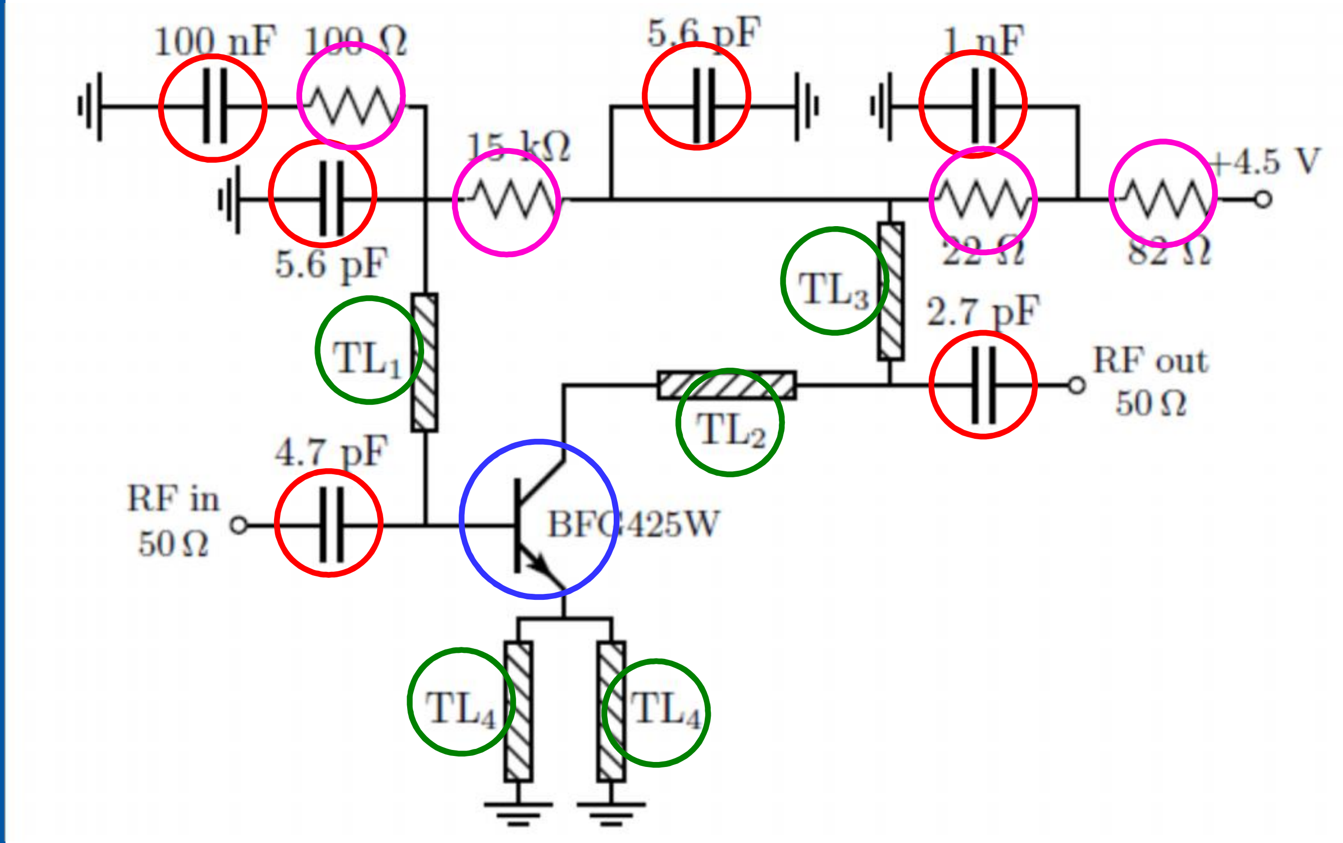




Monte Carlo analysis: …………….. 2 hours
Stochastic testing simulation: …. ……3.6 s

Application example (i)

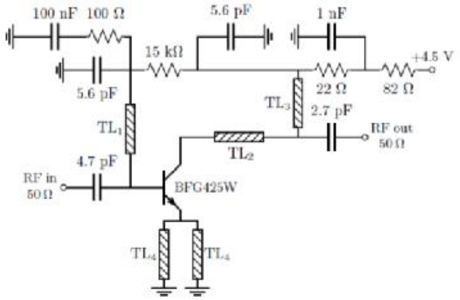
2-GHz BJT low-noise amplifier (LNA)



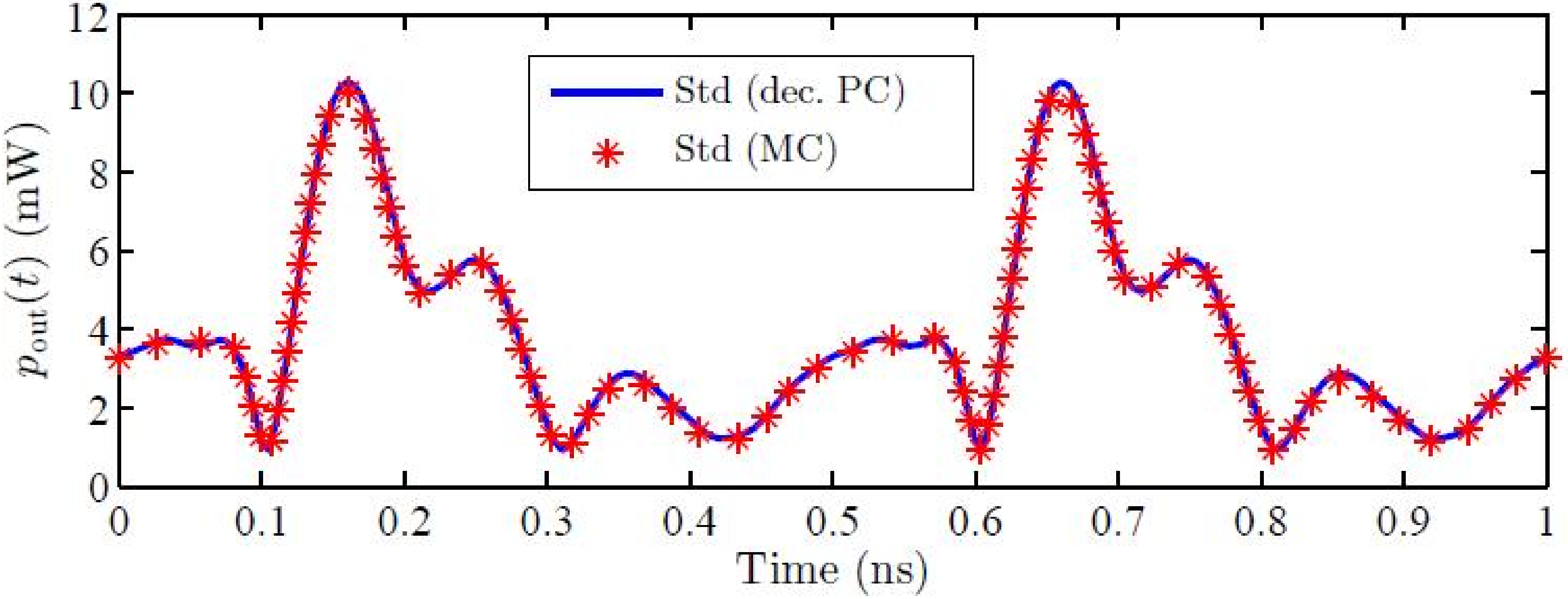
d = 25 Gaussian RVs

- all capacitances with 10% rel. st.dev
- all resistances with 10% rel. st.dev
- parasitics and forward current gain of the BJT with 10% rel. st.dev
- widths of the transmission lines with 5% rel. st.dev

Application example (ii)

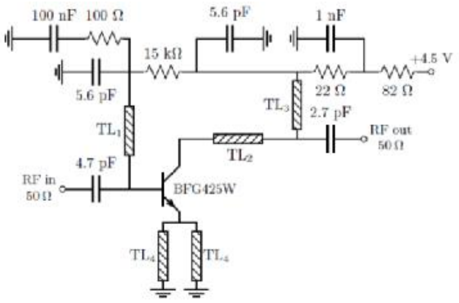


St.dev of steady-state instantaneous output power of LNA (10 dBm input pwr) - harmonic balance simulation in HSPICE

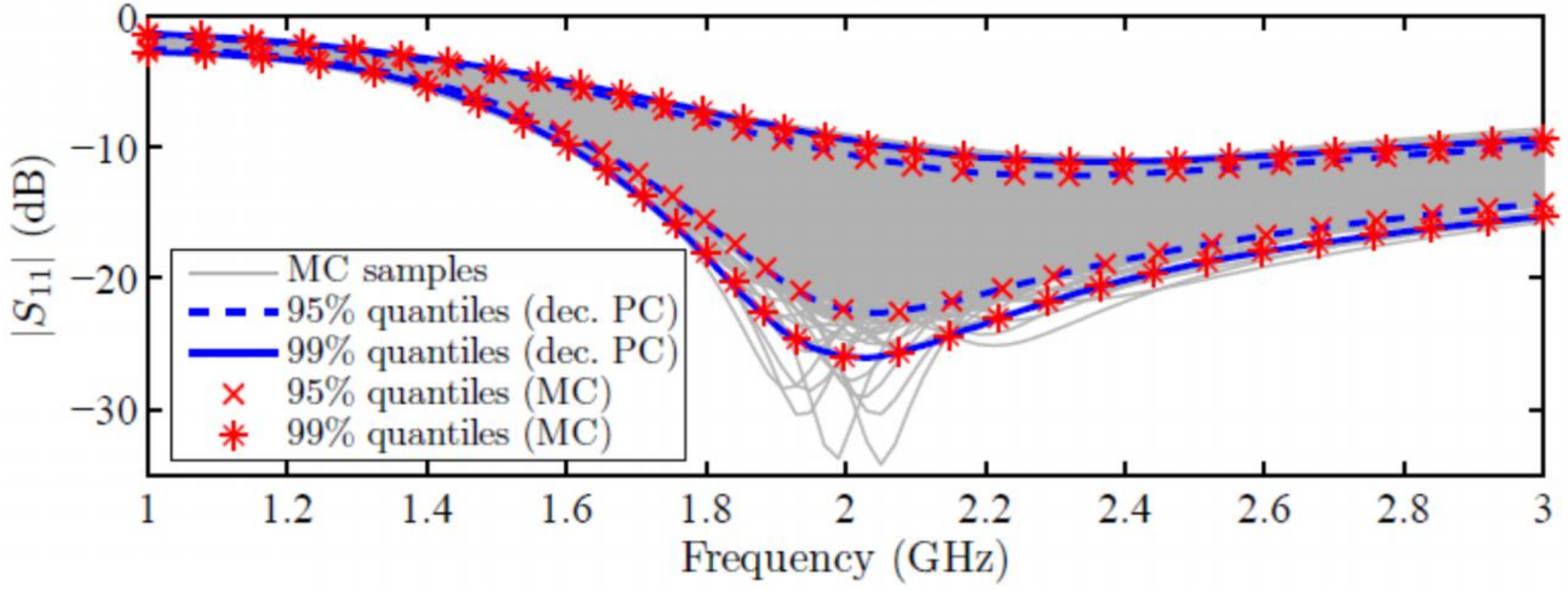


10^5 MC and 171 PC simulations (to have same accuracy)

Application example (ii)



|S11| vs frequency



10^5 MC and 351 PC simulations (to have same accuracy)

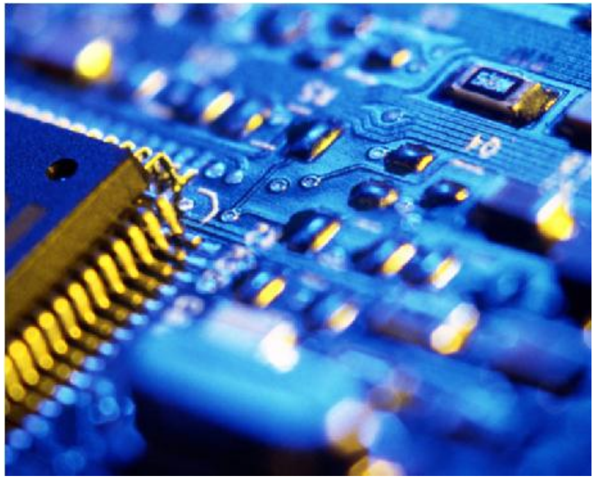
Conclusion *(i)*

❑ Numerical simulation is fundamental in the early design phase of electronic products to **assess system performance**

❑ Besides efficient and accurate simulation tools, there is an increasing demand for the **inclusion of variability** in the analysis

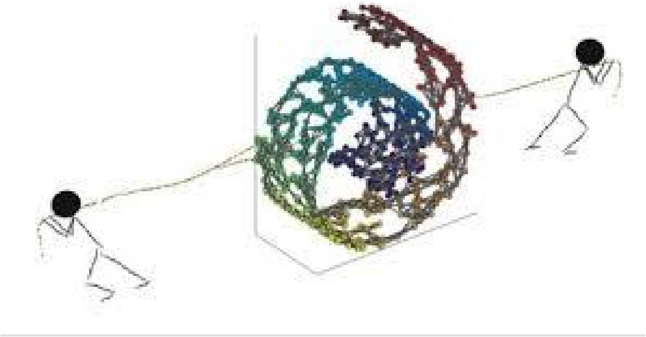
❑ PC methodology handles:

- ❖ Interconnects for high-speed signal transmission
- ❖ Arbitrary microwave passive elements
- ❖ Mixed-signal circuits (thanks to SPICE implementation)



Conclusion *(ii)*

- ❑ PC shows limitations for **high-dimensionality** problems



- ❑ Limitations can be overcome thanks to a simple yet effective **decoupling technique**:
  - ▪ The procedure renders the polynomial chaos technique fully **decoupled** and **non-intrusive**
  - ▪ Compared to collocation techniques, it requires **fewer sampling points** and applies to **any system and distribution type**

Q&A