

# Formulation and Stochastic Galerkin Methods for Stochastic Partial Differential Equations III

**Hermann G. Matthies**

with Andreas Keese

**Institute of Scientific Computing  
Technische Universität Braunschweig, Germany**

wire@tu-bs.de

<http://www.wire.tu-bs.de>



# Repetition of Second Summary

Stochastic Galerkin methods work.

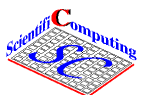
Galerkin procedure is **numerically stable**  $\Rightarrow$  **convergence**.

Convergence **rates** seemingly only with **regularity**.

**Stochastic calculations** produce **huge** amounts of data, which is **expensive** to **operate on** and to **store**.

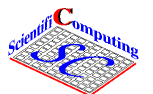
Results a priori live in **very high dimensional** spaces.

They have a **natural tensor product** structure.



# Overview III

1. Approximation Theory
2. Computations for Stochastic Galerkin (linear)
3. Computations for Stochastic Galerkin (non-linear)
4. Stochastic Collocation for PCE
5. Time Evolution Problems
6. Sparse Tensor Product Approximation



# Approximation Theory

- Stability of discrete approximation under truncated KLE and PCE. Matrix stays uniformly positive definite.
- Convergence follows from C ea's Lemma.
- Convergence rates under stochastic regularity in stochastic Hilbert spaces—stochastic regularity theory?.
- Error estimation via dual weighted residuals possible.

**Theorem:** Let  $p > 0$ ,  $r > 1$ ,  $|\rho| \leq 1$ . Let  $P_{k,m}$  be projection onto  $\mathcal{S}_{k,m}$ .

$$\forall R \in (\mathcal{S})^{\rho,-p} \quad \|R - P_{k,m}(R)\|_{\rho,-p}^2 \leq \|R\|_{\rho,-p+r}^2 c(m, k, r)^2,$$

where  $c(m, k, r)^2 = c_1(r)m^{1-r} + c_2(r)2^{-kr}$ . But  $\dim \mathcal{S}_{k,m}$  grows too quickly with  $k$  and  $m$ . Sparser spaces and error estimates needed.

# Equations for Linear Stochastic Galerkin

In what follows we use orthonormal  $\hat{H}_\alpha(\boldsymbol{\theta}) = H_\alpha(\boldsymbol{\theta})/\sqrt{\alpha!}$  instead of  $H_\alpha(\boldsymbol{\theta})$ . With  $\mathbf{f}^{(\gamma)} := \mathbb{E} \left( \hat{H}_\gamma f(\boldsymbol{\theta}) \right)$  and  $\mathbf{f} = [\mathbf{f}^{(0)}, \dots, \mathbf{f}^{(\gamma)}, \dots, \mathbf{f}^{(\lambda)}]$ :

$$\sum_{\alpha} \sum_{\beta} \sum_j \xi_j^{(\alpha)} \underbrace{\mathbb{E} \left( \hat{H}_\alpha \hat{H}_\beta \hat{H}_\gamma \right)}_{=:\Delta_{\beta,\gamma}^{(\alpha)}} \underbrace{\int \nabla \mathbf{N}(x) \boldsymbol{\kappa}_j g_j(x) \nabla \mathbf{N}(x)^T dx}_{\mathbf{K}_j} \mathbf{u}^{(\beta)} = \mathbf{f}^{(\gamma)}$$

$$\mathbb{E} \left( \hat{H}_\alpha \hat{H}_\beta \hat{H}_\gamma \right) = \Delta_{\beta,\gamma}^{(\alpha)} = C_{\beta\gamma}^{(\alpha)} \sqrt{\frac{\alpha!}{\beta!\gamma!}}$$

Equations and  $\mathbf{u}, \mathbf{f}$  have **tensor product** structure (**storage** and **use**).

$$\mathbf{K} \mathbf{u} = \sum_j \sum_{\alpha} \xi_j^{(\alpha)} \Delta^{(\alpha)} \otimes \mathbf{K}_j \mathbf{u} = \mathbf{f}$$

# Computations for Linear Iterative Solution

$$\mathbf{K} \mathbf{u} = \left[ \sum_j \sum_{\alpha} \xi_j^{(\alpha)} \Delta^{(\alpha)} \otimes \mathbf{K}_j \right] \mathbf{u} = \mathbf{f}$$

With preconditioner  $\mathbf{P} = \mathbf{I} \otimes \mathbf{K}_0$  a simple iteration is

$${}^{k+1}\mathbf{u} = {}^k\mathbf{u} + \mathbf{P}^{-1}(\mathbf{f} - \mathbf{K} {}^k\mathbf{u}).$$

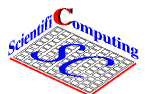
Of course  $\mathbf{P}^{-1}$  is only a **symbolic notation**.

Which means for 'material'  $\kappa_j g_j(x)$  and each  $\beta$  compute the 'residuum'

$${}^k\mathbf{w}_j^{(\beta)} := \mathbf{K}_j {}^k\mathbf{u}^{(\beta)} \text{ and then } {}^{k+1}\mathbf{u} = {}^k\mathbf{u} + {}^k\mathbf{v} \text{ with}$$

$${}^k\mathbf{v}^{(\gamma)} = \mathbf{K}_0^{-1}(\mathbf{f}^{(\gamma)} - \sum_{\alpha, \beta, j} {}^k\mathbf{w}_j^{(\beta)} \xi_j^{(\alpha)} \Delta_{\beta, \gamma}^{(\alpha)})$$

This is one 'preconditioner solve' of the deterministic code.



# Interlude: Computation of Moments from PCE

Discrete version of PCE of  $u(\boldsymbol{\theta})$ :

$$u(\boldsymbol{\theta}) = \bar{u} + \tilde{u}(\boldsymbol{\theta}) = \bar{u} + \sum_{\gamma \neq 0} u^{(\gamma)} \hat{H}_{\gamma}(\boldsymbol{\theta})$$

Let  $M_u^{(k)} = \mathbb{E} \left( \overbrace{\tilde{u}(\boldsymbol{\theta}) \otimes \dots \otimes \tilde{u}(\boldsymbol{\theta})}^{k \text{ times}} \right) = \mathbb{E} \left( \tilde{u}^{\otimes k} \right)$ , totally **symmetric**,

especially  $M_u^{(1)} = \bar{u}$ ,  $M_u^{(2)} = C_u = \sum_{\gamma \neq 0} u^{(\gamma)} \otimes u^{(\gamma)}$ .

It results that **multi-point correlation**

$$M_u^{(k)} = \mathbb{E} \left( \tilde{u}^{\otimes k} \right) = \sum_{\gamma_{(1)}, \dots, \gamma_{(k)} \neq 0} \mathbb{E} \left( \prod_{j=1}^k \hat{H}_{\gamma_{(j)}}(\boldsymbol{\theta}) \right) u^{(\gamma_{(1)})} \otimes \dots \otimes u^{(\gamma_{(k)})}.$$

# Non-Linear Equations

Example: Use  $\kappa(x, u, \omega) = a(x, \omega) + b(x, \omega)u^2$ , and  $a, b$  random.

Space discretisation generates a **non-linear** equation

$\mathbf{A}(\boldsymbol{\theta}, \mathbf{u}(\boldsymbol{\theta})) = \mathbf{A}(\boldsymbol{\theta}, \sum_{\beta} \mathbf{u}^{(\beta)} \hat{H}_{\beta}(\boldsymbol{\theta})) = \mathbf{f}(\boldsymbol{\theta})$ . Projection onto PCE:

$$\mathbf{a}[\mathbf{u}] = [\dots, \mathbb{E} \left( \hat{H}_{\alpha}(\boldsymbol{\theta}) \mathbf{A}(\boldsymbol{\theta}, \sum_{\beta} \mathbf{u}^{(\beta)} \hat{H}_{\beta}(\boldsymbol{\theta})) \right), \dots] = \mathbf{f} = \mathbf{F} \boldsymbol{\varphi} \boldsymbol{\Phi}^T$$

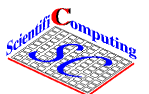
Expressions in  $\mathbf{a}$  need **high-dimensional** integration (in each iteration),

e.g. **Monte Carlo** or **Smolyak** (sparse grid) quadrature:

$$\mathbf{a}^{(\alpha)} = \mathbb{E} \left( \hat{H}_{\alpha}(\boldsymbol{\theta}) \mathbf{A}(\boldsymbol{\theta}, \mathbf{u}(\boldsymbol{\theta})) \right) \approx \sum_{z=1}^Z w_z \hat{H}_{\alpha}(\boldsymbol{\theta}_z) \mathbf{A}(\boldsymbol{\theta}_z, \mathbf{u}(\boldsymbol{\theta}_z))$$

The residual equation to be solved is

$$\mathbf{r}(\mathbf{u}) := \mathbf{f} - \mathbf{a}[\mathbf{u}] = 0.$$



# Solution of Non-Linear Equations

Assume **solver** for deterministic problem:  $\mathbf{r}(\mathbf{u}) = 0$ :

$${}^{k+1}\mathbf{u} = {}^k\mathbf{u} + {}^k\mathbf{w} = {}^k\mathbf{u} + S({}^k\mathbf{u}, \mathbf{r}({}^k\mathbf{u})) =: T({}^k\mathbf{u}, \mathbf{r}({}^k\mathbf{u}))$$

Then stochastic Galerkin non-linear iteration may be

$${}^{k+1}\mathbf{u} = {}^k\mathbf{u} + {}^k\mathbf{w} = {}^k\mathbf{u} + \mathbf{S}({}^k\mathbf{u}, \mathbf{r}({}^k\mathbf{u})) =: \mathbf{T}({}^k\mathbf{u}, \mathbf{r}({}^k\mathbf{u}))$$

with

$$\mathbf{S}({}^k\mathbf{u}, \mathbf{r}({}^k\mathbf{u})) = [S({}^k\mathbf{u}^{(0)}, \mathbf{r}^{(0)}({}^k\mathbf{u})) \dots, S({}^k\mathbf{u}^{(\beta)}, \mathbf{r}^{(\beta)}({}^k\mathbf{u})), \dots]$$

The only **interaction** with the deterministic solver is

- **computing residua** for realisations  $\theta_z$
- using **iteration** for those residua.

# Solution of Non-Linear Equations II

A quasi-Newton method may accelerate convergence

$${}^{k+1}\mathbf{u} = {}^k\mathbf{u} + {}^k\mathbf{w}, \quad {}^k\mathbf{w} = \mathbf{H}_k(\mathbf{r}({}^k\mathbf{u}))$$

$$\mathbf{H}_k(\mathbf{r}) = \mathbf{S}({}^k\mathbf{u}, \mathbf{r}) + \sum_{j=1}^k (a_j \mathbf{p}_j \mathbf{p}_j^T \mathbf{r} + b_j \mathbf{q}_j \mathbf{q}_j^T \mathbf{r})$$

Tensors  $\mathbf{p}$  and  $\mathbf{q}$  computed from residuum and last increment. Notice tensor products of (hopefully sparse) tensors.

Needs pre-conditioner  $\mathbf{S}$  for good convergence: May use linear solver as described before, i.e.

$$\mathbf{S} = \mathbf{I} \otimes \mathbf{K}_0.$$

# Stochastic Collocation

As the  $\{\hat{H}_\alpha(\boldsymbol{\theta})\}$  are an **orthonormal** basis, the PCE of

$$\mathbf{u}(\boldsymbol{\theta}) = \sum_{\beta} \mathbf{u}^{(\beta)} \hat{H}_{\beta}(\boldsymbol{\theta})$$

may be formally computed from **simple projections**:

$$\begin{aligned} \mathbf{u}^{(\beta)} &= \langle \hat{H}_{\beta}(\boldsymbol{\theta}), \mathbf{u}(\boldsymbol{\theta}) \rangle = \mathbb{E} \left( \hat{H}_{\beta}(\boldsymbol{\theta}) \mathbf{u}(\boldsymbol{\theta}) \right) = \\ &= \mathbb{E} \left( \hat{H}_{\beta}(\boldsymbol{\theta}) \sum_{\alpha} \mathbf{u}^{(\alpha)} \hat{H}_{\alpha}(\boldsymbol{\theta}) \right) = \sum_{\alpha} \mathbf{u}^{(\alpha)} \mathbb{E} \left( \hat{H}_{\beta}(\boldsymbol{\theta}) \hat{H}_{\alpha}(\boldsymbol{\theta}) \right) = \sum_{\alpha} \mathbf{u}^{(\alpha)} \delta_{\alpha, \beta}. \end{aligned}$$

$$\text{Hence } \mathbf{u}^{(\beta)} = \mathbb{E} \left( \hat{H}_{\beta}(\boldsymbol{\theta}) \mathbf{u}(\boldsymbol{\theta}) \right) \approx \sum_{z=1}^Z w_z \hat{H}_{\beta}(\boldsymbol{\theta}_z) \mathbf{u}(\boldsymbol{\theta}_z).$$

This only needs **sample solutions**  $\mathbf{u}(\boldsymbol{\theta}_z)$  and works both for **linear** and **non-linear** problems.

# Work Count

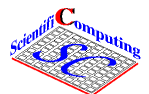
Stochastic **collocation** faces same **stability problems** as was previously explained for **Monte Carlo**.

Assume  $N$  deterministic and  $M$  stochastic variables,  
i.e.  $\mathbf{u}$  is  $N \times M$ .

Assume that for the **stochastic Galerkin** method we need  $J$  iterations,  
then the total work is  $Z \cdot J$  **residua** +  $M \cdot J$  **iterations**.

Assume that for the **stochastic collocation** method we need  $I$  iterations  
on average, then the total work is  $Z \cdot I$  **residua** +  $Z \cdot I$  **iterations**.

Often  $J$  slightly larger than  $I$ , but  $Z$  is much larger than  $M$ .  
Often **iterations** cost more than **residua**.



# Time Dependent Problems

For a **time dependent** problem with  $u \in \mathcal{U}$  and  $f \in \mathcal{F}$  (usually  $= \mathcal{U}^*$  )

$$\dot{u} + A[u] = f(t),$$

if now either the (possibly non-linear) operator or the rhs  $f$  are stochastic, the **stochastic solution** may be sought in a space  $\mathcal{U} \otimes \mathcal{S}$

If this evolution problem is normally **space-discretised** via  $u(x, t) = \sum_k u_k(t) N_k(x)$  to give **ODEs** for the  $[\dots, u_k(t), \dots]^T = \mathbf{u}(t)$ :

$$\dot{\mathbf{u}} + \mathbf{A}[\mathbf{u}] = \mathbf{f}(t),$$

the stochastic ansatz  $\mathbf{u}(t, \boldsymbol{\theta}) = \sum_{\alpha} \mathbf{u}^{(\alpha)}(t) \hat{H}_{\alpha}(\boldsymbol{\theta})$  gives for  $\mathbf{u}(t) = [\dots, \mathbf{u}^{(\alpha)}(t), \dots]$  many more ODEs:

$$\dot{\mathbf{u}} + \mathbf{a}[\mathbf{u}] = \mathbf{f}(t),$$

# What is Wrong with the pure PCE?

Remember already for data field  $\kappa(x, \omega)$   
a **KLE** was additionally used to have fewer terms.

For a solution  $\mathbf{u} = [\dots, \mathbf{u}^{(\alpha)}, \dots]$  there are **too many** PCE terms,  
all stochastically relevant information is encoded **optimally** in  
**Karhunen-Loève expansion (KLE)**.

When only a few vectors  $\mathbf{u}_\ell$  are needed in the end,  
why compute all the  $\mathbf{u}^{(\alpha)}$  in between?

Stochastic Galerkin may be adapted (this is a kind of **model reduction**)  
to work on much less information.



# Simpler Case of Additive Noise

Assume first that  $\mathbf{K}$  is **not random**:  $\forall \gamma \in \mathcal{J}_{k,m}$  satisfy:

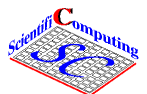
$$\sum_{\beta \in \mathcal{J}_{k,m}} \mathbb{E} \left( \hat{H}_\gamma(\boldsymbol{\theta}) \hat{H}_\beta(\boldsymbol{\theta}) \right) \mathbf{K} \mathbf{u}^{(\beta)} = \mathbf{K} \mathbf{u}^{(\gamma)} = \mathbb{E} \left( \mathbf{f}(\boldsymbol{\theta}) H_\gamma(\boldsymbol{\theta}) \right) =: \mathbf{f}^{(\gamma)}$$

There are **too many**  $\mathbf{f}^{(\gamma)}$ . Use discrete version of **KLE** of  $\mathbf{f}(\boldsymbol{\theta})$ :

$$\mathbf{f}(\boldsymbol{\theta}) = \bar{\mathbf{f}} + \sum_{\ell} \varphi_{\ell} \phi_{\ell}(\boldsymbol{\theta}) \mathbf{f}_{\ell} = \bar{\mathbf{f}} + \sum_{\ell} \sum_{\gamma} \varphi_{\ell} \phi_{\ell}^{(\gamma)} \hat{H}_{\gamma}(\boldsymbol{\theta}) \mathbf{f}_{\ell}.$$

In particular  $\mathbf{f}^{(\gamma)} = \sum_{\ell} \varphi_{\ell} \phi_{\ell}^{(\gamma)} \mathbf{f}_{\ell}$ . The **SVD** of  $\mathbf{f}(\boldsymbol{\theta})$  is with  $\boldsymbol{\Phi} = (\phi_{\ell}^{(\gamma)})$ ,  $\boldsymbol{\varphi} = \text{diag}(\varphi_{\ell})$ , and  $\mathbf{F} = [\dots, \mathbf{f}_{\ell}, \dots]$ ;

$$\mathbf{f} = [\dots, \mathbf{f}^{(\gamma)}, \dots] = \mathbf{F} \boldsymbol{\varphi} \boldsymbol{\Phi}^T = \sum_{\ell} \varphi_{\ell} \mathbf{v}_{\ell} \phi_{\ell}^T.$$



## Solution for Additive Noise

Observe  $\mathbf{K}\bar{\mathbf{u}} = \bar{\mathbf{f}}$ , and set  $\mathbf{V} := \mathbf{K}^{-1}\mathbf{F}$  (i.e.  $\forall \ell$  solve  $\mathbf{K}\mathbf{v}_\ell = \mathbf{f}_\ell$ ),  
 then  $\mathbf{u}(\boldsymbol{\theta}) = \bar{\mathbf{u}} + \mathbf{V}\boldsymbol{\varphi}\boldsymbol{\Phi}^T\mathbf{H} = \bar{\mathbf{u}} + \sum_\ell \varphi_\ell \mathbf{v}_\ell \boldsymbol{\phi}_\ell^T \mathbf{H}$

But this is **not** the SVD of  $\mathbf{u}(\boldsymbol{\theta})$ ! This via **eigenproblem**:

**Covariance**  $\mathbf{C}_f := \mathbb{E} \left( \tilde{\mathbf{f}}(\boldsymbol{\theta}) \otimes \tilde{\mathbf{f}}(\boldsymbol{\theta}) \right) = \mathbf{F}\boldsymbol{\varphi}^2\mathbf{F}^T$ . Hence  
 $\mathbf{C}_u := \mathbb{E} \left( \tilde{\mathbf{u}}(\boldsymbol{\theta}) \otimes \tilde{\mathbf{u}}(\boldsymbol{\theta}) \right) = \mathbf{K}^{-1}\mathbf{F}\boldsymbol{\varphi}^2(\mathbf{K}^{-1}\mathbf{F})^T = \mathbf{V}\boldsymbol{\varphi}^2\mathbf{V}^T$

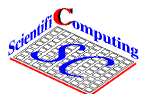
Even **fewer** terms needed with SVD of  $\mathbf{u}(\boldsymbol{\theta})$  from

$$\mathbf{C}_u\mathbf{U} = (\mathbf{V}\boldsymbol{\varphi}^2\mathbf{V}^T)\mathbf{U} = \mathbf{U}\mathbf{v}^2$$

**Sparsification** achieved for  $\mathbf{u}$  via **SVD** with small  $m$ :

$$\mathbf{u} = \bar{\mathbf{u}} + [\dots, \mathbf{u}_m] \text{diag}(v_m) (y_m^{(\beta)})^T \mathbf{H} = \bar{\mathbf{u}} + \mathbf{U}\mathbf{v}\mathbf{Y}^T\mathbf{H},$$

with  $\mathbf{Y}^T := \text{truncate} (\mathbf{v}\mathbf{U}^T\mathbf{V}\boldsymbol{\varphi}^{-1}\boldsymbol{\Phi}^T)$ .



# Example: Computation of Moments

$$\text{Let } M_f^{(k)} = \mathbb{E} \left( \overbrace{\tilde{\mathbf{f}}(\omega) \otimes \dots \otimes \tilde{\mathbf{f}}(\omega)}^{k \text{ times}} \right) = \mathbb{E} \left( \tilde{\mathbf{f}}^{\otimes k} \right), \text{ totally symmetric,}$$

$$\text{and } M_f^{(1)} = \bar{\mathbf{f}}, \quad M_f^{(2)} = \mathbf{C}_f.$$

$$\text{KLE of } \mathbf{C}_f \text{ is } \mathbf{C}_f = \sum_l \varphi_l \mathbf{f}_l \mathbf{f}_l^T = \sum_l \varphi_l \mathbf{f}_l \otimes \mathbf{f}_l.$$

For deterministic operator  $\mathbf{K}$ , just compute  $\mathbf{K} \mathbf{v}_l = \mathbf{f}_l$ , and then

$$\mathbf{K} \bar{\mathbf{u}} = \bar{\mathbf{f}}, \text{ and } M_u^{(2)} = \mathbf{C}_u = \mathbb{E} \left( \tilde{\mathbf{u}}^{\otimes k} \right) = \sum_l \varphi_l \mathbf{v}_l \otimes \mathbf{v}_l.$$

As  $\tilde{\mathbf{u}}(\boldsymbol{\theta}) = \sum_l \sum_{\alpha} \varphi_l \phi_l^{(\alpha)} H_{\alpha}(\boldsymbol{\theta}) \mathbf{v}_l$ , it results that **multi-point correlation**

$$M_u^{(k)} = \sum_{l_1 \leq \dots \leq l_k} \prod_{m=1}^k \varphi_{l_m} \sum_{\alpha_{(1)}, \dots, \alpha_{(k)}} \prod_{n=1}^k \phi_{l_m}^{(\alpha_{(n)})} \mathbb{E} \left( H_{\alpha_{(1)}} \cdots H_{\alpha_{(k)}} \right) \mathbf{v}_{l_1} \otimes \dots \otimes \mathbf{v}_{l_k}.$$

# Sparsification

**Goal:** As with additive noise, compute only with  $\mathbf{f}_\ell, \mathbf{u}_\ell$  from **SVD**.

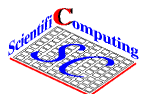
Start **iteration** with tensor product of **low rank**  $L$ .

$$\mathbf{f} = [\dots, \mathbf{f}^{(\gamma)}, \dots] = \mathbf{F}\boldsymbol{\varphi}\boldsymbol{\Phi}^T = \sum_{\ell} \varphi_{\ell} \mathbf{v}_{\ell} \boldsymbol{\phi}_{\ell}^T.$$

At each iteration  $k$ , rank of **iterative approximation**  $\mathbf{u}_k$  will increase by number of terms in **matrix sum**.

In each iteration, perform a **SVD** of  $\mathbf{u}_k$  and reduce rank again to  $L$ .

Resulting iteration **converges** to **SVD** = **discrete KLE** of  $\mathbf{u}$ .



# Operation Count

$$\mathbf{K} \mathbf{u} = \sum_j \sum_{\alpha} \xi_j^{(\alpha)} \Delta^{(\alpha)} \otimes \mathbf{K}_j \mathbf{u} = \mathbf{f}$$

Assume sum in  $\mathbf{K}$  has  $K$  terms,  $\mathbf{u}$  and  $\mathbf{f}$  have size  $N \times M$ ,  
 and  $\mathbf{f} = [\dots, \mathbf{f}^{(\gamma)}, \dots] = \mathbf{F} \boldsymbol{\varphi} \boldsymbol{\phi}^T = \sum_{\ell} \varphi_{\ell} \mathbf{v}_{\ell} \boldsymbol{\phi}_{\ell}^T$   
 has  $L \ll N, L \ll M$  terms.

Each application of  $\mathbf{K}$  on **full**  $\mathbf{u}_k$  needs  $K \times M$   $\mathbf{K}$ -multiplications plus  
 $K \times N$   $\Delta$ -multiplications.

Each application of  $\mathbf{K}$  on **low rank tensor product**  $\mathbf{u}_k$  needs  $K \times L$   
 $\mathbf{K}$ -multiplications plus  $K \times L$   $\Delta$ -multiplications, which is **much less**.

**Storage** is **reduced** from  $N \times M$  to  $L \times (N + M)$ .

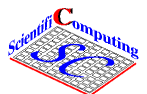
# Integration Rules

For **Monte Carlo (MC)**, the points  $\theta_z$  are random according to measure  $\Gamma$ , and weights are  $w_z = 1/Z$ .

For **Quasi Monte Carlo (QMC)**, the points  $\theta_z$  are non-random according to **number-theoretic low discrepancy** series, and weights are still  $w_z = 1/Z$ .

For **Product Gauss (full tensor product)** rules the points  $\theta_z$  and weights  $w_z$  come from one dimensional rules.

For **sparse grid Smolyak** rules, the points  $\theta_z$  and weights  $w_z$  are known, they come from combination of **different** rules in different dimensions.



# High-Dimensional Integration

$$\text{We want } \mathbb{E}(\psi(\boldsymbol{\theta})) = \int_{\Theta_m} \psi(\boldsymbol{\theta}_m) d\Gamma_m(\boldsymbol{\theta}_m) =$$

$$\int_{\theta_1} \cdots \int_{\theta_m} \psi(\theta_1, \dots, \theta_m) d\Gamma_1(\theta_1) \cdots d\Gamma_1(\theta_m)$$

Expected error  $\epsilon$ :

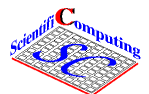
Pure Monte Carlo (MC) has  $\epsilon = O(\|\psi\|_2 Z^{-1/2})$

Quasi Monte Carlo (QMC) has  $\epsilon = O(\|\psi\|_{BV} Z^{-1} (\log Z)^m)$

Quadrature-formulas (integrand in  $C^r(\Theta_m)$ ):

Full Product  $k$ -point Gauss  $\epsilon = O(Z^{-(2r-1)/m})$  with  $Z = O(k^m)$

Sparse Grid Smolyak  $\epsilon = O(Z^{-r} (\log Z)^{(m-1)(r+1)})$  with  $Z = O\left(\frac{2^k}{k!} m^k\right)$



# Evaluation of Residuum through Integration

Model problem, evaluation of  $\mathbb{E} ((\mathbf{f}(\boldsymbol{\theta}, \mathbf{H}(\boldsymbol{\theta})\mathbf{u}) - \mathbf{A}(\boldsymbol{\theta})[\mathbf{H}(\boldsymbol{\theta})\mathbf{u}])H_\gamma(\boldsymbol{\theta}))$

Polynomial- degree	$\sigma$ of component	Monte Carlo	Quadrature $k = 10$
		$Z = 10^6$ abs. error $\cdot 10^3$	$Z = 36$ abs. error $\cdot 10^3$
0	0.26	0.5	$\approx 0$
1	0.27	0.2	0.008
2	0.61	1.2	$\approx 0$
3	0.77	1.5	0.07
4	2.29	4.5	$\approx 0$

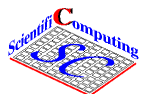
For error  $1 \cdot 10^{-3}$  in degree 4 ca.  $\approx 20$  million MC evaluations required.  
Variance grows with degree.

# Third Summary

- Stochastic Galerkin methods work.
- They are computationally possible on today's hardware.
- They are numerically stable, and have variational convergence theory behind them.
- They can use existing software efficiently.
- They can be **sparsified** via sparse **tensor products**.
- Software framework is being built for easy integration of existing software.

# Important Features of Stochastic Galerkin

- For efficiency try and use **sparse** representation throughout: ansatz in tensor products, as well as storage of solution and residuum—and matrix in tensor products, sparse grids for integration.
- In contrast to MCS, they are stable and have only **cheap** integrands.
- Can be coupled to **existing software**, only marginally more complicated than with MCS.



# Outlook

- Stochastic problems at very beginning (like FEM in the 1960's), when to choose which stochastic discretisation?
- Nonlinear (and instationary) problems possible (but much more work).
- Development of framework for stochastic coupling and parallelisation.
- Computational algorithms have to be further developed.
- Hierarchical parallelisation well possible.