用于随机线弹性不确定性传播的随机虚拟单元方法

郑志宝、乌多·纳肯霍斯特

汉诺威莱布尼茨大学力学与计算力学研究所 &国际研究培训小组 *2657，Appelstraße 9a，30167*，德国汉诺威

arXiv:2305.04253v1 [math.NA] 7 May 2023

摘要

本文提出了用于传播线性弹性随机问题中不确定性的随机虚单元方法。我们首先推导了二维和三维线性弹性问题的随机虚单元方程，这些问题可能涉及材料属性、外力、边界条件等方面的不确定性。为此，我们构造了一个耦合确定性虚拟元空间与随机空间的随机虚拟元空间，并用其近似未知随机解。随后，我们开发了两种数值框架来求解所推导的随机虚单元方程，包括基于多项式混沌近似的方法和基于弱侵入式近似的方法。在基于PC的方法框架中，利用多项式混沌基近似随机解，并通过增广的确定性虚单元方程求解；该方程由对原始随机虚单元方程应用随机伽辽金方法生成。在基于弱侵入式近似的方法框架中，随机解由一组随机变量与确定性向量乘积之和近似，其中，确定性向量通过采用随机Galerkin方法将原始随机问题转化为确定性虚单元方程来求解，而随机变量则通过采用经典Galerkin过程将原始随机问题转化为一维随机代数方程来求解。该方法成功避免了高维随机问题的维数灾难，因为所有随机输入都嵌入到一维随机代数方程中，其计算量对随机维数的依赖较弱。针对具有低维和高维随机输入的二维和三维问题的数值结果，展示了该方法的良好性能。

所提出的方法。关键词：随机虚单元方法；多项式混沌展开；弱侵入式近似；维数灾难；不确定性量化；

---

1. 引言

求解复杂偏微分方程的数值技术正以惊人的速度不断发展，例如有限差分法、有限体积法、有限元法和谱元法等 [1]。作为有限元方法的一种推广，虚单元方法（VEM）于过去十年间被提出并受到广泛关注 [2, 3, 4, 5]。与经典有限元法相比，VEM 可以利用任意多边形或多面体对二维或三维几何域进行离散，不受有限元方法所用规则单元的限制，因此具有很高的灵活性，且对网格不敏感。在 VEM 中，形函数可以是非多项式的。它无需在单元上构造显式形函数，因为所有数值积分都转移到边上，而非在单元上进行。有限元法的若干局限也得以避免，例如凸单元要求，以及小边和小内角导致的单元退化。VEM 在各种问题中的应用已得到广泛研究，例如线性弹性问题 [6, 7, 8],大变形问题 [9, 10], 接触问题 [11, 12, 13], 断裂和裂纹扩展问题 [14, 15, 16], 拓扑优化 [17, 18] 等。尽管已有大量研究，但将 VEM 用于处理不确定性问题仍存在空白。在许多实际工程问题中，系统固有的不确定性或认知不确定性不可避免。预测物理模型中的不确定性传播已成为系统分析的重要组成部分 [19], ，并由此推动了专用于不确定性分析的数值方法的发展。

本文重点研究将确定性虚单元法扩展为随机虚拟单元法（SVEMs），以对可能涉及随机材料性质和随机外力等因素的二维和三维线弹性随机问题进行不确定性分析，而这一领域目前仍存在研究空白。本文的主要贡献包括两个方面：第一项贡献是将确定性虚单元离散方法扩展到随机情形，并推导相应的随机虚拟单元——预印本提交至 *arXiv*，*2023*年*5*月*9*日

方程（SVEE），第二项贡献是提出两种数值方法，以高效且准确地求解推导出的 SVEE。对于第一项贡献，我们将确定性虚拟元空间扩展为随机虚拟元空间，使经典虚拟元空间与随机空间相结合，从而能够为该空间中的确定性函数和随机函数提供（随机）虚拟元近似。所构造的随机虚拟元空间可视为由随机输入参数化的确定性虚拟元空间。对于随机输入的每个样本实现，它都会退化为确定性虚拟元空间，并继承经典虚拟元空间的所有性质。因此，我们可以直接使用具有随机系数的确定性虚拟基函数的线性组合来近似随机解（即未知随机解）。此外，在确定性 VEM [3, 7, 8, 18] 中用于计算虚拟基函数梯度和稳定化项的数值技术，只需稍作修改，仍可用于随机离散。通过这种方式，我们可以组装随机刚度矩阵和随机力向量，得到复杂度与确定性 VEM 相近的 SVEE。

与 VEM 是有限元方法的一种推广类似，SVEM 也被视为随机有限元方法的一种推广 [20]。因此，用于求解所推导出的随机变分不等式的数值算法可以借鉴随机有限元求解算法，例如蒙特卡罗模拟（MCS）及其改进方法 [21, 22], 谱随机有限元法 [23, 24], 随机配点法 [25, 26], 响应面法和克里金法[27, 28], 等。作为第二项贡献，我们开发了两种数值方法来求解所推导出的 SVEE，包括基于多项式混沌展开的 SVEM（PC-SVEM）和基于弱侵入式近似的 SVEM（WIN-SVEM）。本文提及的其他方法也可以类似地推广到 SVEMs。PC-SVEM 是谱随机有限元法 [23]的一种自然扩展。在该方法中，随机解被分解为一组多项式混沌基与确定性向量乘积之和。通过采用随机伽辽金方法，原始 SVEE 被转化为一个增广确定性方程，其规模远大于原始 SVEE。此外，随着物理模型的自由度、随机维数以及多项式混沌基的展开阶数增加，该方程的规模会急剧增大；因此，在处理

大规模和/或高维随机问题时会遭遇维数灾难。为解决这一问题，我们进一步提出了WIN-SVEM，这是我们先前用于求解随机有限元方程 [29, 30]的工作的扩展。在该方法中，随机解近似表示为一组随机变量与确定性向量乘积之和。与 PC-SVEM 不同，随机变量和确定性向量事先均未知。为此，我们通过专门的迭代过程求解它们。确定性向量通过若干个确定性方程求得，这些方程是利用与 PC-SVEM 类似的随机Galerkin过程得到的。随机变量则通过一维随机代数方程求得，这些方程是将经典Galerkin过程应用于原始 SVEE 后得到的。这样，所有随机输入都被嵌入这些一维随机代数方程中，并通过一种对维数依赖较弱的非侵入式采样方法高效求解。因此，所提出的 WIN-SVEM 成功避免了高维随机问题中的维数灾难。

本文的组织结构如下：第 2 节介绍线弹性随机问题的随机虚拟元离散，并推导随机虚单元方程。第 3 节提出 PC-SVEM，用于求解所推导的随机系统。随后，第 4 节提出 WIN-SVEM，以高效求解所推导的随机系统，尤其着眼于高维随机问题。第 5 节给出涉及低维和高维随机输入的二维和三维数值算例，以展示所提方法的性能。第 6 节总结全文并展望未来工作。

2. 随机虚单元方程

2.1. 线弹性随机方程

设 $(\Theta, \Xi, \mathcal{P})$ 为一个适当的概率空间，其中 Θ 表示基本事件空间，Ξ 是定义在 Θ 上的 σ-代数，$\mathcal{P}$ 为概率测度。本文考虑以下弹性随机方程

$$\begin{cases} -\nabla \cdot \boldsymbol{\sigma}(\mathbf{x},\theta) = \boldsymbol{f}(\mathbf{x},\theta) & \text{in } \Omega \\ \boldsymbol{\sigma}(\mathbf{x},\theta)\cdot\mathbf{n} = \boldsymbol{g}(\mathbf{x},\theta) & \text{on } \Gamma_N, \\ \boldsymbol{u}(\mathbf{x},\theta) = \boldsymbol{u}_D(\mathbf{x},\theta) & \text{on } \Gamma_D \end{cases} \tag{1}$$

其中，确定性区域 $\Omega \subset \mathbb{R}^d$ 的边界为 $\partial\Omega$，空间维数 d 可取 $= 2, 3$，d 维空间坐标表示为 $\mathrm{x} = (\mathrm{x1}, \cdots, \mathrm{x}_d) \in \Omega$，$\nabla \cdot (\cdot)$ 表示散度算子，$\boldsymbol{\sigma}(\mathrm{x}, \theta)$ 为随机应力张量，向量值位移场 $\mathrm{u}(\mathrm{x}, \theta) = [\mathrm{u}_1(\mathrm{x}, \theta), \cdots, \mathrm{u}_d(\mathrm{x}, \theta)]^{\mathrm{T}} \in \mathbb{R}^d$ 是待求的未知随机解，向量值场 $\mathrm{f}(\mathrm{x}, \theta) = [\mathrm{f}_1(\mathrm{x}, \theta), \cdots, \mathrm{f}_d(\mathrm{x}, \theta)]^{\mathrm{T}} \in \mathbb{R}^d$ 与随机外力有关，而 Γ_N 和 Γ_D 是边界段，分别对应诺伊曼边界条件 $\mathrm{g}(\mathrm{x}, \theta) = [\mathrm{g}_1(\mathrm{x}, \theta), \cdots, \mathrm{g}_d(\mathrm{x}, \theta)]^{\mathrm{T}} \in \mathbb{R}^d$ 和狄利克雷边界条件 $\mathrm{u}_D(\mathrm{x}, \theta) = [\mathrm{u}_{D,1}(\mathrm{x}, \theta), \cdots, \mathrm{u}_{D,d}(\mathrm{x}, \theta)]^{\mathrm{T}} \in \mathbb{R}^d$。本文仅考虑线弹性随机问题。采用以下线性随机应变张量和线弹性本构关系

$$\boldsymbol{\varepsilon}(\boldsymbol{u}(\mathbf{x},\theta)) = \frac{1}{2}\left(\nabla\boldsymbol{u}(\mathbf{x},\theta) + (\nabla\boldsymbol{u}(\mathbf{x},\theta))^{\mathrm{T}}\right) \in \mathbb{R}^{d\times d}, \quad \boldsymbol{\sigma}(\boldsymbol{u}(\mathbf{x},\theta),\theta) = \boldsymbol{C}(\mathbf{x},\theta)\,\boldsymbol{\varepsilon}(\boldsymbol{u}(\mathbf{x},\theta)) \in \mathbb{R}^{d\times d}, \tag{2}$$

其中，$\boldsymbol{C}(\mathbf{x},\theta)$ 是一个四阶弹性张量，可能与随机材料属性相关，例如随机杨氏模量和随机泊松比。

为求解方程 (1)，考虑其弱形式如下：求随机函数 $\boldsymbol{u}(\mathbf{x}, \theta) \in \mathscr{V}: \Omega \times \Theta \to \mathbb{R}^d$ 使得下列等式对 $\mathcal{P}$-几乎必然成立 $\theta \in \Theta$,

$$\mathscr{W}(\boldsymbol{u}(\mathbf{x},\theta),\boldsymbol{v}(\mathbf{x}),\theta) = \mathscr{F}(\boldsymbol{v}(\mathbf{x}),\theta), \quad \forall \boldsymbol{v}(\mathbf{x}) \in \mathscr{V}, \tag{3}$$

其中，函数空间定义为 $\mathscr{V} = \left\{\boldsymbol{v} \in \left[\mathscr{H}^1(\Omega)\right]^d : \boldsymbol{v} = \boldsymbol{0} \text{ 在 } \Gamma_D\right\}$ 上，且 $\mathscr{H}^1(\Omega)$ 是 Ω 上平方可积标量函数空间的子空间，其中包含函数及其弱导数 [31]。项 $\mathscr{W}(\boldsymbol{u}(\mathbf{x}, \theta), \boldsymbol{v}(\mathbf{x}), \theta)$ 和 $\mathscr{F}(\boldsymbol{v}(\mathbf{x}), \theta)$ 定义为

$$\mathscr{W}(\boldsymbol{u}(\mathbf{x},\theta),\boldsymbol{v}(\mathbf{x}),\theta) = \int_\Omega \boldsymbol{\sigma}(\boldsymbol{u}(\mathbf{x},\theta),\theta) : \boldsymbol{\varepsilon}(\boldsymbol{v}(\mathbf{x}))\,\mathrm{d}\mathbf{x} \tag{4}$$

$$= \int_\Omega [\boldsymbol{C}(\mathbf{x},\theta)\,\boldsymbol{\varepsilon}(\boldsymbol{u}(\mathbf{x},\theta))] : \boldsymbol{\varepsilon}(\boldsymbol{v}(\mathbf{x}))\,\mathrm{d}\mathbf{x}, \tag{5}$$

$$\mathscr{F}(\boldsymbol{v}(\mathbf{x}),\theta)=\int_{\Omega}\boldsymbol{f}(\mathbf{x},\theta)\cdot\boldsymbol{v}(\mathbf{x})\,\mathrm{d}\mathbf{x}+\int_{\Gamma_N}\boldsymbol{g}(\mathbf{x},\theta)\cdot\boldsymbol{v}(\mathbf{x})\,\mathrm{d}\mathbf{s}. \tag{6}$$

2.2. 随机虚拟元离散化

我们对弱形式 Eq. (3) 采用随机虚单元离散。具体而言，将区域 Ω 划分为 n_e 个互不重叠的多边形单元 $\Omega=\bigcup_{\mathrm{e}=1}^{n_e}\Omega^{(\mathrm{e})}$，且每个单元 $\Omega^{(\mathrm{e})}$，$\mathrm{e}=1,\ldots,n_e$ 包含 $n^{(\mathrm{e})}$ 个顶点和 $m^{(\mathrm{e})}$ 条边。本文仅考虑低阶虚单元，但所提出的方法可推广至高阶虚单元 [32]。我们给出单元 $\Omega^{(\mathrm{e})}$的如下近似离散虚空间 $\mathscr{V}_h\left(\Omega^{(\mathrm{e})}\right)\subset\mathscr{V}$

$$\mathscr{V}_h\left(\Omega^{(\mathrm{e})}\right)=\left\{\boldsymbol{v}_h\in\left[\mathscr{H}^1\left(\Omega^{(\mathrm{e})}\right)\bigcap\mathscr{C}^0\left(\Omega^{(\mathrm{e})}\right)\right]^d:\boldsymbol{v}_{h,\partial\Omega_i^{(\mathrm{e})}}\in\left[\mathscr{P}_1\left(\partial\Omega_i^{(\mathrm{e})}\right)\right]^d,\forall\partial\Omega_i^{(\mathrm{e})}\in\partial\Omega^{(\mathrm{e})},\right.$$
$$\left.\nabla\cdot[\boldsymbol{C}(\theta)\boldsymbol{\varepsilon}(\boldsymbol{v}_h)]=\mathbf{0},\ \forall\theta\in\Theta\ \text{ on }\ \Omega^{(\mathrm{e})}\right\}, \tag{7}$$

其中 $\mathscr{P}_1$ 表示次数不超过 1 的多项式空间，$\boldsymbol{v}_h$ 在每条边 $\partial\Omega^{(\mathrm{e})}{}_i$ 上都是多项式，$i=1,\cdots,m^{(\mathrm{e})}$ 在 $\Omega^{(\mathrm{e})}$ 上具有 $\mathscr{C}^0$- 连续性，并且对于所有 $\theta\in\Theta$，$\nabla\cdot[\boldsymbol{C}(\theta)\ \boldsymbol{\varepsilon}(\boldsymbol{v}_h)]$ 在单元 $\Omega^{(\mathrm{e})}$ 上为零。通过这种方式，我们将经典虚拟元空间与随机输入 θ耦合起来。函数 $\boldsymbol{v}_h$ 在单元 $\Omega^{(\mathrm{e})}$ 上未知，但在边 $\partial\Omega^{(\mathrm{e})}$上是显式已知的。现在我们在离散空间上考虑弱形式方程 (3)：求 $\boldsymbol{u}_h(\mathbf{x},\theta)\in\mathscr{V}_h:\Omega\times\Theta\to\mathbb{R}^d$，使得下式对 $\mathcal{P}$-几乎处处的 $\theta\in\Theta$成立，

$$\mathscr{W}_h(\boldsymbol{u}_h(\mathbf{x},\theta),\boldsymbol{v}_h(\mathbf{x}),\theta)=\mathscr{F}_h(\boldsymbol{v}_h(\mathbf{x}),\theta),\quad\forall\boldsymbol{v}_h(\mathbf{x})\in\mathscr{V}_h, \tag{8}$$

其中 $\mathscr{W}_h$ 和 $\mathscr{F}_h$ 通过遍历所有单元 $\left\{\Omega^{(\mathrm{e})}\right\}_{\mathrm{e}=1}^{n_e}$进行组装

$$\mathscr{W}_h(\boldsymbol{u}_h(\mathbf{x},\theta),\boldsymbol{v}_h(\mathbf{x}),\theta)=\sum_{\mathrm{e}=1}^{n_e}\mathscr{W}_h^{(\mathrm{e})}(\boldsymbol{u}_h(\mathbf{x},\theta),\boldsymbol{v}_h(\mathbf{x}),\theta),\quad\mathscr{F}_h(\boldsymbol{v}_h(\mathbf{x}),\theta)=\sum_{\mathrm{e}=1}^{n_e}\mathscr{F}_h^{(\mathrm{e})}(\boldsymbol{v}_h(\mathbf{x}),\theta). \tag{9}$$

需要注意的是，$\mathscr{W}_{(\mathrm{e})\,h}(\boldsymbol{u}_h(\mathbf{x},\ \theta),\boldsymbol{v}_h(\mathbf{x}),\ \theta)$ 和 $\mathscr{F}^{(\mathrm{e})}{}_h(\boldsymbol{v}_h(\mathbf{x}),\ \theta)$ 无法采用与经典有限元法类似的方式进行计算，因为函数 $\boldsymbol{v}_h$ 在单元 $\Omega^{(\mathrm{e})}$ 上是未知的。为此，类似于确定性 VEM，定义一个投影算子 $\boldsymbol{\Pi}^{(\mathrm{e})}:\ \mathscr{V}_h\left(\Omega^{(\mathrm{e})}\right)\to\left[\mathscr{P}_1\left(\Omega^{(\mathrm{e})}\right)\right]^d$，使得对于 $\forall\boldsymbol{v}_h\in\mathscr{V}_h\left(\Omega^{(\mathrm{e})}\right),\ \boldsymbol{p}_1\in\left[\mathscr{P}_1\left(\Omega^{(\mathrm{e})}\right)\right]^d,\theta\in\Theta,$

$$\mathscr{W}_h^{(\mathrm{e})}\left(\boldsymbol{v}_h(\mathbf{x})-\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x}),\boldsymbol{p}_1(\mathbf{x}),\theta\right)=0 \tag{10}$$

成立。因此，我们可以将项 $\mathscr{W}_{(\mathrm{e})h}(\boldsymbol{u}_h(\mathbf{x},\theta),\boldsymbol{v}_h(\mathbf{x}),\theta)$ 重新表述为

$$\begin{aligned}
&\mathscr{W}_h^{(\mathrm{e})}(\boldsymbol{u}_h(\mathbf{x},\theta),\boldsymbol{v}_h(\mathbf{x}),\theta)\\
=&\mathscr{W}_h^{(\mathrm{e})}\left(\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta)+\boldsymbol{u}_h(\mathbf{x},\theta)-\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta),\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x})+\boldsymbol{v}_h(\mathbf{x})-\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x}),\theta\right), \qquad (11)\\
=&\underbrace{\mathscr{W}_h^{(\mathrm{e})}\left(\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta),\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x}),\theta\right)}_{=\mathscr{W}_C^{(\mathrm{e})}(\theta)}+\underbrace{\mathscr{W}_h^{(\mathrm{e})}\left(\boldsymbol{u}_h(\mathbf{x},\theta)-\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta),\boldsymbol{v}_h(\mathbf{x})-\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x}),\theta\right)}_{=\mathscr{W}_S^{(\mathrm{e})}(\theta)}\\
&+\underbrace{\mathscr{W}_h^{(\mathrm{e})}\left(\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta),\boldsymbol{v}_h(\mathbf{x})-\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x}),\theta\right)}_{=0}+\underbrace{\mathscr{W}_h^{(\mathrm{e})}\left(\boldsymbol{u}_h(\mathbf{x},\theta)-\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta),\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x}),\theta\right)}_{=0}, \qquad (12)
\end{aligned}$$

其中，$\mathscr{W}^{(\mathrm{e})}{}_C(\theta)$ 和 $\mathscr{W}_{(\mathrm{e})S}(\theta)$ 分别称为随机一致项和随机稳定化项。最后两项根据公式（10）得到，因为 $\boldsymbol{\Pi}^{(\mathrm{e})}v_h(\mathbf{x}),\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta)\in\left[\mathscr{P}_1\left(\Omega^{(\mathrm{e})}\right)\right]^d$。此外，$\boldsymbol{\varepsilon}^{(\mathrm{e})}\left(\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta)\right)$ 和 $\boldsymbol{\varepsilon}^{(\mathrm{e})}\left(\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x})\right)$ 均为（随机）常应变张量。因此，随机一致项 $\mathscr{W}_{(\mathrm{e})C}(\theta)$ 通过以下方式计算。

$$\begin{aligned}
\mathscr{W}_C^{(\mathrm{e})}(\theta)&=\int_{\Omega^{(\mathrm{e})}}\left[\boldsymbol{C}^{(\mathrm{e})}(\mathbf{x},\theta)\boldsymbol{\varepsilon}^{(\mathrm{e})}\left(\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta)\right)\right]:\boldsymbol{\varepsilon}^{(\mathrm{e})}\left(\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x})\right)\mathrm{d}\mathbf{x}\\
&=a^{(\mathrm{e})}\left[\boldsymbol{C}^{(\mathrm{e})}(\mathbf{x},\theta)\boldsymbol{\varepsilon}^{(\mathrm{e})}\left(\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta)\right)\right]:\boldsymbol{\varepsilon}^{(\mathrm{e})}\left(\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x})\right), \qquad (13)
\end{aligned}$$

其中，$a^{(\mathrm{e})}$ 表示单元 $\Omega^{(\mathrm{e})}$ 的面积（对于二维多边形单元）或体积（对于三维多边形单元）。

此外，我们令单元 $\Omega^{(\mathrm{e})}$ 上的随机解 $\boldsymbol{u}_h(\mathbf{x},\theta)$ 和函数 $v_h(\mathbf{x})$ 采用一组虚拟基函数 $\left\{\varphi_i(\mathbf{x})\in\mathscr{V}_h\left(\Omega^{(\mathrm{e})}\right)\right\}_{i=1}^{n^{(\mathrm{e})}}$进行近似

$$\boldsymbol{u}_h(\mathbf{x},\theta)=\sum_{i=1}^{n^{(\mathrm{e})}}\varphi_i(\mathbf{x})\,\mathbf{u}_i^{(\mathrm{e})}(\theta),\quad \boldsymbol{v}_h(\mathbf{x})=\sum_{i=1}^{n^{(\mathrm{e})}}\varphi_i(\mathbf{x})\,\mathbf{v}_i^{(\mathrm{e})}, \qquad (14)$$

其中，单元 $\Omega^{(\mathrm{e})}$ 的解向量 $\mathrm{u}^{(\mathrm{e})}\mathrm{i}(\theta)$ 为 $\mathrm{u}^{(\mathrm{e})}(\theta)=\left[\mathrm{u}_1^{(\mathrm{e})\mathrm{T}}(\theta),\ \cdots,\ \mathrm{u}^{(\mathrm{e})\mathrm{T}}\mathrm{n}^{(\mathrm{e})}(\theta)\right]^{\mathrm{T}}\in\mathbb{R}^{n^{(\mathrm{e})}\mathrm{d}}$，而 $\mathrm{u}^{(\mathrm{e})}\mathrm{i}(\theta)=\left[\mathrm{u}^{(\mathrm{e})}\mathrm{i},1(\theta),\ \cdots,\ \mathrm{u}^{(\mathrm{e})}\mathrm{i},\mathrm{d}(\theta)\right]^{\mathrm{T}}\in\mathbb{R}^d$ 是第 i 个顶点的解向量。向量 $\mathrm{v}^{(\mathrm{e})}\mathrm{i}\in\mathbb{R}^{n^{(\mathrm{e})}\mathrm{d}}$ 具有类似的表达式，但不涉及随机输入 θ。将投影算子 $\boldsymbol{\Pi}^{(\mathrm{e})}$ 应用于 $\mathrm{u}_h(\mathbf{x},\theta)$ 和 $\mathrm{v}_h(\mathbf{x})$，可得

$$\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta)=\sum_{i=1}^{n^{(\mathrm{e})}}\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})\,\mathbf{u}_i^{(\mathrm{e})}(\theta),\quad \boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x})=\sum_{i=1}^{n^{(\mathrm{e})}}\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})\,\mathbf{v}_i^{(\mathrm{e})}. \qquad (15)$$

在此基础上，我们将应变张量 $\boldsymbol{\varepsilon}^{(\mathrm{e})}\left(\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x},\theta)\right)$ 和 $\boldsymbol{\varepsilon}^{(\mathrm{e})}\left(\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x})\right)$ 改写为以下向量形式

$$\boldsymbol{\varepsilon}^{(\mathrm{e})}_{\boldsymbol{\Pi},\mathrm{vec}}\left(\mathbf{u}^{(\mathrm{e})}(\theta)\right)=\mathbf{B}^{(\mathrm{e})}\mathbf{u}^{(\mathrm{e})}(\theta)=\left[\mathbf{B}^{(\mathrm{e})}_1,\cdots,\mathbf{B}^{(\mathrm{e})}_{n^{(\mathrm{e})}}\right]\mathbf{u}^{(\mathrm{e})}(\theta)\in\mathbb{R}^{\frac{d(d+1)}{2}},\quad \boldsymbol{\varepsilon}^{(\mathrm{e})}_{\boldsymbol{\Pi},\mathrm{vec}}\left(\mathbf{v}^{(\mathrm{e})}\right)=\mathbf{B}^{(\mathrm{e})}\mathbf{v}^{(\mathrm{e})}\in\mathbb{R}^{\frac{d(d+1)}{2}},\tag{16}$$

其中矩阵 $\mathbf{B}^{(\mathrm{e})}{}_i\in\mathbb{R}^{\frac{d(d+1)}{2}\times d}$, $i=1,\cdots,n^{(\mathrm{e})}$ 由下式给出。

$$\mathbf{B}^{(\mathrm{e})}_i=\begin{bmatrix}\frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_1} & 0\\ 0 & \frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_2}\\ \frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_2} & \frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_1}\end{bmatrix}\ (d=2)\ \text{ or }\ \begin{bmatrix}\frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_1} & 0 & 0\\ 0 & \frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_2} & 0\\ 0 & 0 & \frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_3}\\ \frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_2} & \frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_1} & 0\\ 0 & \frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_3} & \frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_2}\\ \frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_3} & 0 & \frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_1}\end{bmatrix}\ (d=3),\tag{17}$$

其中，分量 $\frac{\partial\boldsymbol{\Pi}^{(\mathrm{e})}\varphi_i(\mathbf{x})}{\partial x_j}$，$i=1$、$\cdots$、$n^{(\mathrm{e})}$、$j=1$、$\cdots$、$d$ 通过根据公式 (10) 将计算转移到单元 $\Omega^{(\mathrm{e})}$ 的边和面上进行求值，这与确定性虚元法相同，详情请参见 [2, 3, 7, 8]。这样一来，无需知道函数 $\{\varphi_i(\mathbf{x})\}^{n^{(\mathrm{e})}}{}_{i=1}$ 的显式表示，只需知道它们在边上的迹即可。可以采用类似于经典有限元法中所使用的拉格朗日线性基函数来实现这一目的。将公式 (16) 代入公式 (13)，可得

$$\mathscr{W}^{(\mathrm{e})}_C(\theta)=a^{(\mathrm{e})}\mathbf{v}^{(\mathrm{e})\mathrm{T}}\mathbf{B}^{(\mathrm{e})\mathrm{T}}\mathbf{G}^{(\mathrm{e})}(\theta)\,\mathbf{B}^{(\mathrm{e})}\mathbf{u}^{(\mathrm{e})}(\theta),\tag{18}$$

其中$\mathbf{G}^{(\mathrm{e})}(\theta)\in\mathbb{R}^{\frac{d(d+1)}{2}\times d(d+1)^2}$是单元 $\Omega^{(\mathrm{e})}$ 的张量 $\boldsymbol{C}(\mathbf{x},\theta)$ 的矩阵形式。二维和三维问题中$\mathbf{G}(\theta)$的两种详细表示可见数值算例部分。因此，与随机一致项 $\mathscr{W}_{(\mathrm{e})\,C}(\theta)$ 对应的随机单元刚度矩阵为

$$\mathbf{k}^{(\mathrm{e})}_C(\theta)=a^{(\mathrm{e})}\mathbf{B}^{(\mathrm{e})\mathrm{T}}\mathbf{G}^{(\mathrm{e})}(\theta)\,\mathbf{B}^{(\mathrm{e})}\in\mathbb{R}^{n^{(\mathrm{e})}d\times n^{(\mathrm{e})}d}.\tag{19}$$

此外，我们来考虑式（12）中随机稳定化项 $\mathscr{W}_{(\mathrm{e})\,S}(\theta)$ 的计算，该计算可以借助多种数值策略 [2, 7, 8]来实现。这里我们采用 [3, 18], 中提出的方法，该方法对应于

$$\mathscr{W}_S^{(\mathrm{e})}(\theta)=\gamma_S^{(\mathrm{e})}(\theta)\sum_{i=1}^{n^{(\mathrm{e})}}\left[\boldsymbol{u}_h(\mathbf{x}_i,\theta)-\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{u}_h(\mathbf{x}_i,\theta)\right]\cdot\left[\boldsymbol{v}_h(\mathbf{x}_i)-\boldsymbol{\Pi}^{(\mathrm{e})}\boldsymbol{v}_h(\mathbf{x}_i)\right],\tag{20}$$

其中，$\mathbf{x}_i=(x_{i,1},\cdots,x_{i,d})$，$x_{i,j}$表示单元$\Omega^{(\mathrm{e})}$中第$j$个顶点的第$i$个坐标值。在实际实现中，其计算方式为 [18]

$$\mathscr{W}_S^{(\mathrm{e})}(\theta)=\gamma_S^{(\mathrm{e})}(\theta)\,\mathbf{v}^{(\mathrm{e})\mathrm{T}}\left(\mathbf{I}_{n^{(\mathrm{e})}d}-\mathbf{S}^{(\mathrm{e})}\right)^{\mathrm{T}}\left(\mathbf{I}_{n^{(\mathrm{e})}d}-\mathbf{S}^{(\mathrm{e})}\right)\mathbf{u}^{(\mathrm{e})}(\theta),\tag{21}$$

其中系数$\gamma^{(\mathrm{e})}{}_S(\theta)=\frac{1}{2}\mathrm{Tr}\left(\mathbf{G}^{(\mathrm{e})}(\theta)\right)$，$\mathrm{Tr}(\cdot)$表示矩阵的迹算子，$\mathbf{I}_{n^{(\mathrm{e})}d}\in\mathbb{R}^{n^{(\mathrm{e})}d\times n^{(\mathrm{e})}d}$为单位矩阵，而确定性矩阵$\mathbf{S}^{(\mathrm{e})}\in\mathbb{R}^{n^{(\mathrm{e})}d\times n^{(\mathrm{e})}d}$定义为

$$\mathbf{S}^{(\mathrm{e})}=\begin{bmatrix}h_{11}^{(\mathrm{e})}\mathbf{I}_d & \cdots & h_{1n^{(\mathrm{e})}}^{(\mathrm{e})}\mathbf{I}_d\\ \vdots & \ddots & \vdots\\ h_{n^{(\mathrm{e})}1}^{(\mathrm{e})}\mathbf{I}_d & \cdots & h_{n^{(\mathrm{e})}n^{(\mathrm{e})}}^{(\mathrm{e})}\mathbf{I}_d\end{bmatrix},\quad h_{ij}^{(\mathrm{e})}=\mathbf{X}_i^{(\mathrm{e})\mathrm{T}}\mathbf{A}_j^{(\mathrm{e})}+\frac{1}{n^{(\mathrm{e})}},\tag{22}$$

其中，向量$\mathbf{X}^{(\mathrm{e})}{}_i$和$\mathbf{A}^{(\mathrm{e})}{}_i$由下式给出。

$$\mathbf{X}_i^{(\mathrm{e})}=\left[x_{1,i}-\frac{1}{n^{(\mathrm{e})}}\sum_{j=1}^{n^{(\mathrm{e})}}x_{1,j},\cdots,x_{d,i}-\frac{1}{n^{(\mathrm{e})}}\sum_{j=1}^{n^{(\mathrm{e})}}x_{d,j}\right]^{\mathrm{T}}\in\mathbb{R}^d,\quad\mathbf{A}_i^{(\mathrm{e})}=\left[\frac{\partial\boldsymbol{\Pi}\varphi_i(\mathbf{x})}{\partial x_1},\cdots,\frac{\partial\boldsymbol{\Pi}\varphi_i(\mathbf{x})}{\partial x_d}\right]^{\mathrm{T}}\in\mathbb{R}^d.\tag{23}$$

因此，与随机稳定化项$\mathscr{W}_{(\mathrm{e})S}(\theta)$相对应的随机单元刚度矩阵为

$$\mathbf{k}_S^{(\mathrm{e})}(\theta)=\gamma_S^{(\mathrm{e})}(\theta)\left(\mathbf{I}_{n^{(\mathrm{e})}d}-\mathbf{S}^{(\mathrm{e})}\right)^{\mathrm{T}}\left(\mathbf{I}_{n^{(\mathrm{e})}d}-\mathbf{S}^{(\mathrm{e})}\right)\in\mathbb{R}^{n^{(\mathrm{e})}d\times n^{(\mathrm{e})}d},\tag{24}$$

其与确定性虚单元法中的情况非常接近，但系数$\gamma^{(\mathrm{e})}{}_S(\theta)$涉及随机输入$\theta$。因此，可以继承确定性虚单元矩阵的计算过程，用于$\mathbf{k}^{(\mathrm{e})}{}_S(\theta)$。我们只需稍加注意随机系数$\gamma^{(\mathrm{e})}{}_S(\theta)$的计算。

在最后一步，我们考虑随机项$\mathscr{F}^{(\mathrm{e})}{}_h(\boldsymbol{v}_h(\mathbf{x}),\theta)$，它使用边和面（二维情形）或面和单元（三维情形）上的一点积分规则精确计算得到。

$$\mathscr{F}_h^{(\mathrm{e})}(\boldsymbol{v}_h(\mathbf{x}),\theta)=\int_{\Omega^{(\mathrm{e})}} f^{(\mathrm{e})}(\mathbf{x},\theta)\cdot\boldsymbol{v}_h(\mathbf{x})\,\mathrm{d}\mathbf{x}+\int_{\Gamma_N^{(\mathrm{e})}} g^{(\mathrm{e})}(\mathbf{x},\theta)\cdot\boldsymbol{v}_h(\mathbf{x})\,\mathrm{d}\mathbf{s}$$

$$=\frac{a^{(\mathrm{e})}}{\left(n^{(\mathrm{e})}\right)^2}\mathbf{v}^{(\mathrm{e})\mathrm{T}}\mathbf{Z}_1^{\mathrm{T}}\mathbf{Z}_1\boldsymbol{f}^{(\mathrm{e})}(\theta)+\mathbf{v}^{(\mathrm{e})\mathrm{T}}\sum_{j=1}^{n_{\Gamma_N}^{(\mathrm{e})}}\frac{b_j^{(\mathrm{e})}}{\left(n_{\Gamma_{N,j}}^{(\mathrm{e})}\right)^2}\mathbf{Z}_{2,j}^{\mathrm{T}}\mathbf{Z}_{2,j}\boldsymbol{g}^{(\mathrm{e})}(\theta),\tag{25}$$

其中，$\mathrm{n}_{\Gamma_N}^{(\mathrm{e})}$ 表示单元 $\Omega^{(\mathrm{e})},\Gamma^{(\mathrm{e})}$ N 的边数（二维情形）或面数（三维情形），j 表示第 j 条边或第 j 个面，且包含 $\mathrm{n}_{\Gamma_N}^{(\mathrm{e})},_j$ 个顶点；$\mathrm{b}^{(\mathrm{e})}$ j 表示 $\Gamma^{(\mathrm{e})}$ N, j 的长度（二维单元）或面积（三维单元）。由各顶点处的值组成的向量为 $\mathrm{f}\,(\mathrm{e})\,(\theta)=\left[\mathrm{f}\,(\mathrm{e})\mathrm{T}\ 1\,(\theta),\cdots,\mathrm{f}^{(\mathrm{e})\mathrm{T}}\mathrm{n}^{(\mathrm{e})}(\theta)\right]^{\mathrm{T}}\in\mathbb{R}^{n^{(\mathrm{e})}}\mathrm{d}$, $\mathrm{f}\,(\mathrm{e})\,\mathrm{i}\,(\theta)=\left[\mathrm{f}\,(\mathrm{e})\,\mathrm{i},1\,(\theta),\cdots,\mathrm{f}\,(\mathrm{e})\,\mathrm{i},\mathrm{d}\,(\theta)\right]^{\mathrm{T}}\in\mathbb{R}^{d}$ 以及 $\mathrm{g}^{(\mathrm{e})}(\theta)=\left[\mathrm{g}_1^{(\mathrm{e})\mathrm{T}}(\theta),\cdots,\mathrm{g}^{(\mathrm{e})\mathrm{T}}\mathrm{n}^{(\mathrm{e})}(\theta)\right]^{\mathrm{T}}\in\mathbb{R}^{n^{(\mathrm{e})}}\mathrm{d}$, $\mathrm{g}^{(\mathrm{e})}\,\mathrm{i}\,(\theta)=\left[\mathrm{g}^{(\mathrm{e})}\,\mathrm{i},1\,(\theta),\cdots,\mathrm{g}^{(\mathrm{e})}\,\mathrm{i},\mathrm{d}\,(\theta)\right]^{\mathrm{T}}\in\mathbb{R}^{d}$。确定性矩阵定义为 $\mathrm{Z}_1=\Big[\underbrace{\mathrm{I}_d,\cdots,\mathrm{I}_d}_{\mathrm{(e)}}\Big]\in\mathbb{R}^{d\times n^{(\mathrm{e})}}\mathrm{I}\,\mathrm{n}\,\mathrm{d}$ 和 $\mathrm{Z}2,_j=\left[\delta_1^*,\Gamma^{(\mathrm{e})}\,\mathrm{N},\mathrm{j}\mathrm{I}_d,\cdots,\delta^*\,\mathrm{n}^{(\mathrm{e})},\Gamma^{(\mathrm{e})}\,\mathrm{N},\mathrm{j}\mathrm{I}_d\right]\in\mathbb{R}^{d\times n^{(\mathrm{e})}}\mathrm{d}$，其中 $\delta^*\,\mathrm{i},\Gamma^{(\mathrm{e})}$ N,j = 1 当顶点 i $\in\Gamma^{(\mathrm{e})}$ N, j 时取值为 1，否则取值为 0。因此，随机单元力向量计算如下：

$$\mathbf{f}^{(\mathrm{e})}(\theta)=\frac{a^{(\mathrm{e})}}{\left(n^{(\mathrm{e})}\right)^2}\mathbf{Z}_1^{\mathrm{T}}\mathbf{Z}_1\boldsymbol{f}^{(\mathrm{e})}(\theta)+\sum_{j=1}^{n_{\Gamma_N}^{(\mathrm{e})}}\frac{b^{(\mathrm{e})}}{\left(n_{\Gamma_{N,j}}^{(\mathrm{e})}\right)^2}\mathbf{Z}_{2,j}^{\mathrm{T}}\mathbf{Z}_{2,j}\boldsymbol{g}^{(\mathrm{e})}(\theta)\in\mathbb{R}^{n^{(\mathrm{e})}d}.\tag{26}$$

组装上述随机单元刚度矩阵和随机单元力向量后，可得到以下 SVEE

$$\mathbf{K}(\theta)\,\mathbf{u}(\theta)=\mathbf{F}(\theta),\tag{27}$$

其中，全局随机刚度矩阵$\mathbf{K}(\theta)$和全局随机力向量$\mathbf{F}(\theta)$由下式组装得到

$$\mathbf{K}(\theta)=\bigcup_{\mathrm{e}=1}^{n_e}\left(\mathbf{k}_C^{(\mathrm{e})}(\theta)+\mathbf{k}_S^{(\mathrm{e})}(\theta)\right)\in\mathbb{R}^{n\times n},\quad\mathbf{F}(\theta)=\bigcup_{\mathrm{e}=1}^{n_e}\mathbf{f}^{(\mathrm{e})}(\theta)\in\mathbb{R}^{n},\tag{28}$$

其中，$\bigcup(\cdot)$表示对所有随机单元矩阵和向量进行组装的组装算子，总自由度（DoF）为$n=n_e d$。

此外，在许多情况下，随机矩阵 $\mathbf{G}(\mathbf{x},\theta)$（或张量 $\boldsymbol{C}(\mathbf{x},\theta)$）可以用以下分离形式进行近似

$$\mathbf{G}(\mathbf{x},\theta)=\sum_{i=0}^{m}\xi_i(\theta)\,\mathbf{G}_i, \tag{29}$$

其中 $\xi_0(\theta)\equiv 1$, $\{\xi_i(\theta)\}_{i=1}^{m}$ 是标量随机变量，$\left\{\mathbf{G}_i\in\mathbb{R}^{\frac{d(d+1)}{2}\times\frac{d(d+1)}{2}}\right\}_{i=0}^{m}$ 是确定性矩阵。对于非分离随机矩阵 $\mathbf{G}(\mathbf{x},\theta)$，可以采用模拟随机场的方法，对高斯随机输入和非高斯随机输入均获得类似于式（29）的近似，例如 Karhunen-Lo`eve 展开和多项式混沌展开 [33, 34, 35]。这样，我们可以将 SVEE (27) 重新表述为

$$\left(\sum_{i=0}^{m}\xi_i(\theta)\,\mathbf{K}_i\right)\mathbf{u}(\theta)=\mathbf{F}(\theta), \tag{30}$$

其中，确定性矩阵 $\{\mathbf{K}_i\}_{i=0}^{m}$ 通过以下方式组装：

$$\mathbf{K}_i=\bigcup_{\mathrm{e}=1}^{n_e}\left(\mathbf{k}_C^{(\mathrm{e})}(\mathbf{G}_i)+\mathbf{k}_S^{(\mathrm{e})}(\mathbf{G}_i)\right)\in\mathbb{R}^{n\times n} \tag{31}$$

由于式 (19) 中的随机单元矩阵 $\mathbf{k}^{(\mathrm{e})}{}_C(\theta)$ 和式 (24) 中的 $\mathbf{k}^{(\mathrm{e})}{}_S(\theta)$ 线性依赖于矩阵分量 $\{\mathbf{G}_i\}_{i=0}^{m}$。

3. PC-SVEM：基于多项式混沌的谱随机虚拟元方法 d

基于多项式混沌的方法已得到充分发展，并广泛应用于求解各种随机问题。本节介绍一种 PC-SVEM，用于求解 SVEE (27)（或分离形式的式 (30)）。在该方法中，随机解 $\mathbf{u}(\theta)$ 使用（广义）多项式混沌基展开如下

$$\mathbf{u}_{\mathrm{PC},k}(\theta)=\sum_{i=1}^{k}\Gamma_i(\theta)\,\mathbf{d}_{\mathrm{PC},i}, \tag{32}$$

其中，$\{\mathbf{d}_{\mathrm{PC},i}\}_{i=1}^{k}$ 为待求解的相应确定性向量，$\{\Gamma_i(\theta)\}_{i=1}^{k}$ 为多项式混沌基。实际上，我们可以针对不同类型的随机输入选择不同的多项式混沌基，例如，针对高斯随机变量使用 Hermite 多项式混沌基，针对均匀随机变量使用勒让德多项式混沌基 [23, 24]。随后采用随机Galerkin方法将 SVEE (27) 转化为以下确定性方程 [23]

$$\int_{\Theta}\left[\mathbf{K}(\theta)\sum_{i=1}^{k}\Gamma_i(\theta)\,\mathbf{d}_{\mathrm{PC},i}-\mathbf{F}(\theta)\right]\Gamma_j(\theta)\,\mathrm{d}\mathcal{P}(\theta)=0,\quad j=1,\cdots,k, \tag{33}$$

其中，$\mathcal{P}(\theta)$ 是随机输入 θ 的概率测度。此外，上述方程可以改写为紧凑形式。

$$\mathbf{K}_{\mathrm{PC}}\mathbf{d}_{\mathrm{PC}} = \mathbf{F}_{\mathrm{PC}}, \tag{34}$$

其中，增广确定性矩阵 $\mathbf{K}_{\mathrm{PC}} \in \mathbb{R}^{nk\times nk}$ 和增广确定性向量 $\mathbf{d}_{\mathrm{PC}}$、$\mathbf{F}_{\mathrm{PC}} \in \mathbb{R}^{nk}$ 由.....组装而成。

$$\mathbf{K}_{\mathrm{PC}} = \begin{bmatrix} \mathbf{K}_{\mathrm{PC},11} & \cdots & \mathbf{K}_{\mathrm{PC},1k} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{\mathrm{PC},k1} & \cdots & \mathbf{K}_{\mathrm{PC},kk} \end{bmatrix}, \quad \mathbf{d}_{\mathrm{PC}} = \begin{bmatrix} \mathbf{d}_{\mathrm{PC},1} \\ \vdots \\ \mathbf{d}_{\mathrm{PC},k} \end{bmatrix}, \quad \mathbf{F}_{\mathrm{PC}} = \begin{bmatrix} \mathbf{F}_{\mathrm{PC},1} \\ \vdots \\ \mathbf{F}_{\mathrm{PC},k} \end{bmatrix}, \tag{35}$$

其中矩阵和向量分量 $\mathbf{K}_{\mathrm{PC},ji} \in \mathbb{R}^{n\times n}$ 和 $\mathbf{F}_{\mathrm{PC},j} \in \mathbb{R}^{n}$, $i, j = 1, \cdots, k$ 是

$$\mathbf{K}_{\mathrm{PC},ji} = \int_{\Theta} \mathbf{K}(\theta)\,\Gamma_i(\theta)\,\Gamma_j(\theta)\,\mathrm{d}\mathcal{P}(\theta), \quad \mathbf{F}_{\mathrm{PC},j} = \int_{\Theta} \mathbf{F}(\theta)\,\Gamma_j(\theta)\,\mathrm{d}\mathcal{P}(\theta). \tag{36}$$

此外，若考虑分离形式方程（30），上述矩阵 $\mathbf{K}_{\mathrm{PC},ji}$ 的计算可简化为

$$\mathbf{K}_{\mathrm{PC},ji} = \sum_{l=0}^{m} \left[\int_{\Theta} \xi_l(\theta)\,\Gamma_i(\theta)\,\Gamma_j(\theta)\,\mathrm{d}\mathcal{P}(\theta) \right] \mathbf{K}_i, \tag{37}$$

其中仅涉及数值积分 $\int_{\Theta} \xi_l(\theta)\ \Gamma_i(\theta)\ \Gamma_j(\theta)\ \mathrm{d}\mathcal{P}(\theta)$。对于低维随机问题，得益于随机空间[26]上高效的数值积分策略，计算成本足够低。

值得注意的是，多项式混沌基的总数为$k = \frac{(m+r)!}{m!r!}$，其中（·）！表示阶乘运算符，r为多项式混沌基的展开阶数。因此，与经典的基于多项式混沌的方法类似，所提出的 PC-SVEM 仍受维数灾难的影响，因为式（34）中的矩阵/向量规模 nk 会随着物理模型的空间自由度n 、随机输入的随机维数m 以及多项式混沌基的展开阶数r 的增加而急剧增大。例如，当$n = 1\times 10^3$、$m = 10$ 且$r = 4$时，其规模约为$nk = 1\times 10^6$，这需要付出极其高昂的计算代价。尽管已有若干方法用于减轻计算负担，例如专用迭代算法和稀疏多项式混沌近似 [36, 37, 38]，但使用 PC-SVEM 求解超高维随机问题仍然具有挑战性。

4. WIN-SVEM：弱侵入式随机虚单元方法

4.1. 弱侵入式随机虚拟单元方法

为避免上述 PC-SVEM 中出现的维数灾难，本节提出 WIN-SVEM，可将其视为我们此前关于随机有限元方法的研究 [29, 30] 向 SVEM 的扩展。为此，我们考虑由以下级数展开近似随机解$\mathbf{u}(\theta)$

$$\mathbf{u}_{\mathrm{WIN},k}(\theta) = \sum_{i=1}^{k} \lambda_i(\theta)\, \mathbf{d}_{\mathrm{WIN},i}, \tag{38}$$

其中，$\{\lambda_i(\theta) \in \mathbb{R}\}_{i=1}^{k}$ 为标量随机变量，$\{\mathbf{d}_{\mathrm{WIN},i} \in \mathbb{R}^n\}_{i=1}^{k}$ 为确定性向量，k 为保留项的数量。需要指出的是，数量 k 以及所有数对 $\{\lambda_i(\theta), \mathbf{d}_{\mathrm{WIN},i}\}_{i=1}^{k}$ 均无法预先确定。本文提出了一种迭代算法，逐一求解这些数对 $\{\lambda_i(\theta), \mathbf{d}_{\mathrm{WIN},i}\}$。具体而言，假设已知第 $(k-1)$ 次近似解 $\mathbf{u}_{\mathrm{WIN},k-1}(\theta) = \sum_{i=1}^{k-1} \lambda_i(\theta)\, \mathbf{d}_{\mathrm{WIN},i}$，目标是求解第 k 个数对 $\{\lambda_k(\theta), \mathbf{d}_{\mathrm{WIN},k}\}$。原始 SVEE（27）可重写为

$$\mathbf{K}(\theta)\, \lambda_k(\theta)\, \mathbf{d}_{\mathrm{WIN},k} = \mathbf{F}_k(\theta), \tag{39}$$

其中，随机向量 $\mathbf{F}_k(\theta) = \mathbf{F}(\theta) - \mathbf{K}(\theta) \sum_{i=1}^{k-1} \lambda_i(\theta)\, \mathbf{d}_{\mathrm{WIN},i}$。这样，式（39）只涉及一对未知量 $\{\lambda_k(\theta), \mathbf{d}_{\mathrm{WIN},k}\}$。然而，与 PC-SVEM 中随机基（即多项式混沌基）已知的情况不同，在这里，随机变量 $\lambda_k(\theta)$ 和确定性向量$\mathbf{d}_{\mathrm{WIN},k}$ 均为未知量。为避免这一问题，采用交替迭代求解。具体而言，若随机变量 $\lambda_k(\theta)$ 已知（或给定一个初始值），则利用随机伽辽金方法 [23]（与式（33）中的方法类似），式（39）可转化为以下确定性虚拟元方程

$$\int_{\Theta} \left[\mathbf{K}(\theta)\, \lambda_k(\theta)\, \mathbf{d}_{\mathrm{WIN},k} - \mathbf{F}_k(\theta)\right] \lambda_k(\theta)\, \mathrm{d}\mathcal{P}(\theta) = 0, \tag{40}$$

这等价于

$$\mathbf{K}_{\mathrm{WIN},k}\mathbf{d}_{\mathrm{WIN},k} = \mathbf{F}_{\mathrm{WIN},k}, \tag{41}$$

其中，确定性矩阵 $\mathbf{K}_{\mathrm{WIN},k} = \int_{\Theta} \mathbf{K}(\theta)\, \lambda^2{}_k(\theta)\, \mathrm{d}\mathcal{P}(\theta) \in \mathbb{R}^{n\times n}$ 和确定性向量$\mathbf{F}_{\mathrm{WIN},k} = \int_{\Theta} \mathbf{F}_k(\theta)\, \lambda_k(\theta)\, \mathrm{d}\mathcal{P}(\theta) \in \mathbb{R}^n$。现有的数值求解器可用于高效地求解该问题。

高效且准确地 [39]。请注意，方程（41）的规模与原始 SVEE（27）相同，不同于基于多项式混沌的导出方程（34）的增广规模，因此能够节省大量计算工作。在实际实现中，我们令向量 $\mathbf{d}_{\mathrm{WIN},k}$ 与已获得的向量 $\{\mathbf{d}_{\mathrm{WIN},i}\}_{=1}^{k-1}{}_{i}$ 正交，以加快收敛速度，这通过格拉姆-施密特正交归一化实现。

$$\mathbf{d}_{\mathrm{WIN},k} = \mathbf{d}_{\mathrm{WIN},k} - \sum_{i=1}^{k-1} \left(\mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \mathbf{d}_{\mathrm{WIN},i}\right) \mathbf{d}_{\mathrm{WIN},i}, \quad \mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \mathbf{d}_{\mathrm{WIN},k} = 1, \tag{42}$$

其中，$\{\mathbf{d}_{\mathrm{WIN},i}\}_{=1}^{k-1}{}_{i}$是满足$\mathbf{d}^{\mathrm{T}}{}_{\mathrm{WIN},i}\mathbf{d}_{\mathrm{WIN},j} = \delta_{ij}$的归一化正交向量，其中 δ_{ij}是克罗内克δ。

利用确定性向量$\mathbf{d}_{\mathrm{WIN},k}$ 通过方程（41）求解后，随机变量 $\lambda_k(\theta)$ 随即利用以下经典 Galerkin过程重新计算

$$\mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \left[\mathbf{K}(\theta)\, \lambda_k(\theta)\, \mathbf{d}_{\mathrm{WIN},k} - \mathbf{F}_k(\theta)\right] = 0. \tag{43}$$

由于随机矩阵$\mathbf{K}(\theta)$是正定的，且$\mathbf{z}^{\mathrm{T}}\mathbf{K}(\theta)\,\mathbf{z} > 0, \forall \mathbf{z} \neq \mathbf{0} \in \mathbb{R}^n, \forall \theta \in \Theta$ 成立，上式可改写为

$$\lambda_k(\theta) = \frac{\mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \mathbf{F}_k(\theta)}{\mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \mathbf{K}(\theta)\, \mathbf{d}_{\mathrm{WIN},k}}. \tag{44}$$

为避免高维问题中产生的维数灾难，我们采用非侵入式采样方法 [30] 求解方程（44），而不是采用基于多项式混沌的近似，即

$$\lambda_k\left(\widehat{\boldsymbol{\theta}}\right) = \left[\mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \mathbf{F}_k\left(\widehat{\boldsymbol{\theta}}\right)\right] \oslash \left[\mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \mathbf{K}\left(\widehat{\boldsymbol{\theta}}\right) \mathbf{d}_{\mathrm{WIN},k}\right] \in \mathbb{R}^{n_s}, \tag{45}$$

其中，$\square\left(\widehat{\boldsymbol{\theta}}\right)$ 表示 $\square(\theta)$的 n_s 个样本实现，$\mathrm{d}^{\mathrm{T}}\,\mathrm{WIN,kF}_k\left(\widehat{\boldsymbol{\theta}}\right) \in \mathbb{R}^{n_s}$和 $\mathrm{d}^{\mathrm{T}}\,\mathrm{WIN,kK}\left(\widehat{\boldsymbol{\theta}}\right)\mathrm{d}_{\mathrm{WIN},k} \in \mathbb{R}^{n_s}$ 分别是随机变量 $\mathrm{d}^{\mathrm{T}}\,\mathrm{WIN,kF}_k(\theta)$ 和 $\mathrm{d}^{\mathrm{T}}\,\mathrm{WIN,kK}(\theta)\,\mathrm{d}_{\mathrm{WIN},k}$的样本向量，而 $\oslash$ 表示两个样本向量的逐元素除法，也称为 Hadamard 除法算子。这样，所有随机输入都嵌入到样本实现向量 $\mathrm{d}^{\mathrm{T}}\,\mathrm{WIN,kF}_k\left(\widehat{\boldsymbol{\theta}}\right)$和 $\mathrm{d}^{\mathrm{T}}\,\mathrm{WIN,kK}\left(\widehat{\boldsymbol{\theta}}\right)\mathrm{d}_{\mathrm{WIN},k}$中，因此不受随机输入随机维数的影响。由此可成功避免维数灾难，下一节将对此进行详细讨论。

通过执行方程（41）和方程（45）的迭代过程，直至达到指定精度，我们可以计算第k对 $\{\lambda_k(\theta), \mathbf{d}_{\mathrm{WIN},k}\}$。类似的迭代也用于计算其他配对 $\{\lambda_{k+1}(\theta), \mathbf{d}_{\mathrm{WIN},k+1}\}, \cdots$，直至获得随机解的良好近似。不过需要注意的是，方程（38）中的随机解 $\mathbf{u}_{\mathrm{WIN},k}(\theta)$ 是按顺序进行近似的，因此并未严格满足原始 SVEE（27）。在某些情况下，这种近似的精度较低 [29]。为避免这一问题，我们引入一个重新计算过程。为此，将$\mathbf{D}_{\mathrm{WIN}} = [\mathbf{d}_{\mathrm{WIN},1}, \cdots, \mathbf{d}_{\mathrm{WIN},k}] \in \mathbb{R}^{n\times k}$ 视为一组降阶基函数，并通过以下降阶随机方程重新计算随机变量向量 $\mathbf{\Lambda}(\theta) = [\lambda_1(\theta), \cdots, \lambda_k(\theta)]^{\mathrm{T}} \in \mathbb{R}^k$

$$\left[\mathbf{D}_{\mathrm{WIN}}^{\mathrm{T}}\mathbf{K}(\theta)\mathbf{D}_{\mathrm{WIN}}\right]\mathbf{\Lambda}(\theta) = \mathbf{D}_{\mathrm{WIN}}\mathbf{F}(\theta), \tag{46}$$

需要针对n_s 个样本实现重复求解，才能得到最终解 $\mathbf{\Lambda}\left(\theta^{(i)}\right), i = 1, \cdots, n_s$，但由于降阶随机矩阵 $\mathbf{D}^{\mathrm{T}}{}_{\mathrm{WIN}}\mathbf{K}(\theta)\mathbf{D}_{\mathrm{WIN}} \in \mathbb{R}^{k\times k}$ 和降阶随机向量 $\mathbf{D}^{\mathrm{T}}{}_{\mathrm{WIN}}\mathbf{F}(\theta) \in \mathbb{R}^k$ 的规模在大多数情况下都远小于原始 SVEE（27），因此所需的计算量很小。

我们着重说明所提方法的弱侵入性。一方面，方程（38）可视为一种随机解的侵入式近似，与基于 PC 的侵入式近似（32）非常相似。另一方面，在方程（41）中求解$\mathbf{d}_{\mathrm{WIN},k}$ 时，具体实现仅涉及确定性计算，且矩阵$\mathbf{K}_{\mathrm{WIN},k}$ 与原始随机矩阵 $\mathbf{K}(\theta)$保持相同的规模和矩阵属性，因此具有弱侵入性。此外，用于计算随机变量 $\lambda_k(\theta)$ 的方程（45）完全是非侵入式的。从这些意义上说，我们仅以弱侵入和完全非侵入的方式实现侵入式随机解近似。该方法结合了侵入式方法的高效率和非侵入式方法对维数的弱依赖性，能够高效、准确地求解高维随机问题。

4.2. 高维随机问题

在本节中，我们将说明，所提出的 WIN-SVEM 无需任何修改即可应用于高维随机问题。我们从高随机维数对求解方程（41）和方程（45）的影响这一角度来阐述这一点。我们仅考虑方程（30）

在本节中，较大的数量 m在方程（30）中被截断，以生成一个高维随机问题。通过这种方式，方程（41）中的确定性矩阵 $\mathbf{K}_{\mathrm{WIN},k}$ 和确定性向量 $\mathbf{F}_{\mathrm{WIN},k}$通过以下方式计算。

$$\mathbf{K}_{\mathrm{WIN},k} = \sum_{i=0}^{m} \left[\int_{\Theta} \xi_i(\theta) \lambda_k^2(\theta) \mathrm{d}\mathcal{P}(\theta) \right] \mathbf{K}_i, \tag{47}$$

$$\mathbf{F}_{\mathrm{WIN},k} = \int_{\Theta} \mathbf{F}(\theta) \lambda_k(\theta) \mathrm{d}\mathcal{P}(\theta) - \sum_{i=0}^{m} \sum_{j=1}^{k-1} \left[\int_{\Theta} \xi_i(\theta) \lambda_j(\theta) \lambda_k(\theta) \mathrm{d}\mathcal{P}(\theta) \right] \mathbf{K}_i \mathbf{d}_{\mathrm{WIN},j}, \tag{48}$$

其中，概率积分通过以下非侵入式采样方法进行近似

$$\int_{\Theta} \xi_i(\theta) \lambda_j(\theta) \lambda_k(\theta) \mathrm{d}\mathcal{P}(\theta) = \widehat{\mathbb{E}} \left\{ \xi_i\left(\widehat{\boldsymbol{\theta}}\right) \odot \lambda_j\left(\widehat{\boldsymbol{\theta}}\right) \odot \lambda_k\left(\widehat{\boldsymbol{\theta}}\right) \right\}, \quad i = 0, \cdots, m, j = 1, \cdots, k, \tag{49}$$

其中，$\left\{ \xi_i\left(\widehat{\boldsymbol{\theta}}\right) \in \mathbb{R}^{n_s} \right\}_{i=0}^{m}$ 是随机变量 $\{\xi_i(\theta)\}_{i=0}^{m}$的样本向量，算子 $\odot$ rep-$\widehat{\ }$表示样本向量的逐元素乘法，$\mathbb{E}\{\cdot\}$ 是样本向量的期望算子。式（49）总共需要 $k(m+1)$ 次期望运算，其对随机维数不敏感，即使随机维数非常高，计算量也很低。请注意，尽管这里只展示了随机矩阵$\mathbf{K}(\theta)$中的高维输入，但如果随机向量$\mathbf{F}(\theta)$包含高维随机输入，上述计算对于 $\int_{\Theta} \mathbf{F}(\theta) \lambda_k(\theta) \mathrm{d}\mathcal{P}(\theta)$同样高效。

此外，式（45）右侧的样本向量通过以下方式计算：

$$\mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \mathbf{K}\left(\widehat{\boldsymbol{\theta}}\right) \mathbf{d}_{\mathrm{WIN},k} = \sum_{i=0}^{m} \xi_i\left(\widehat{\boldsymbol{\theta}}\right) \left(\mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \mathbf{K}_i \mathbf{d}_{\mathrm{WIN},k} \right) \in \mathbb{R}^{n_s}, \tag{50}$$

$$\mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \mathbf{F}_k\left(\widehat{\boldsymbol{\theta}}\right) = \mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \mathbf{F}(\theta) - \sum_{i=0}^{m} \sum_{j=1}^{k-1} \left[\xi_i\left(\widehat{\boldsymbol{\theta}}\right) \odot \lambda_j\left(\widehat{\boldsymbol{\theta}}\right) \right] \left(\mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}} \mathbf{K}_i \mathbf{d}_{\mathrm{WIN},j} \right) \in \mathbb{R}^{n_s}, \tag{51}$$

这需要为 $k(m+1)$ 次操作，总计适用于 $\mathbf{d}^{\mathrm{T}}{}_{\mathrm{WIN},\ k}\mathbf{K}_i\mathbf{d}_{\mathrm{WIN},\ j}$、$i = 0, \cdots$、$m$、$j = 1, \cdots$、$k$，对于高维随机问题，其计算成本也很低。因此，方程（41）和方程（45）均不受随机维数的影响。所提出的方法能够成功避免维数灾难。

4.3. 算法实现

上述用于求解随机变分不等式的 WIN-SVEM 方法总结于算法1中，该算法包含两个循环过程。内循环从步骤4到步骤9，用于求解第k个向量对 $\{\lambda_k(\theta)\,,\mathbf{d}_{\mathrm{WIN},k}\}$。从步骤2到步骤13的外循环则利用一组向量对 $\{\lambda_i(\theta)\,,\mathbf{d}_{\mathrm{WIN},i}\}_{i=1}^{k}$来近似随机解。为执行内循环，需要在步骤3中初始化一个随机样本向量$\lambda^{(0)}{}_k\ \left(\widehat{\boldsymbol{\theta}}\right)\in\mathbb{R}^{n_s}$。在数值实现中，由于初始样本对所提出方法的计算精度和效率影响较小，因此可以选择任意非零的n_s 维向量作为初始值。每次内循环结束后，我们只需在外循环的步骤10中更新随机力向量$\mathbf{F}_{k+1}(\theta)$，并在步骤11中存储降阶矩阵$\mathbf{D}_{\mathrm{WIN}}$。

算法1用于求解随机变分不等式的 WIN-SVEM

1: $k\leftarrow 1$

2: **当** $\epsilon_{\mathbf{u},k}\geq\epsilon_{\mathbf{u}}$ **执行**

3: 初始化随机样本 $\lambda^{(0)}{}_k\ \left(\widehat{\boldsymbol{\theta}}\right)=\left\{\lambda^{(0)}{}_k\ \left(\theta^{(i)}\right)\right\}_{i=1}^{n_s}\in\mathbb{R}^{n_s}$

4: **当** $\epsilon_{\mathbf{d},j}>\epsilon_{\mathbf{d}}$ **时**

5: 计算确定性向量 $\mathbf{d}^{(j)}{}_{\mathrm{WIN},k}$，方法是求解方程（41）

6: 正交归一化 $\mathbf{d}^{(j)}{}_{\mathrm{WIN},k}$，使用方程（42）

7: 通过方程（45）更新随机样本向量 $\lambda^{(j)}{}_k\ \left(\widehat{\boldsymbol{\theta}}\right)\in\mathbb{R}^{n_s}$

8: 计算局部迭代误差 $\epsilon_{\mathbf{d},j}$, $j\leftarrow j+1$

9: **end**

10: 更新随机力向量 $\mathbf{F}_{k+1}(\theta)=\mathbf{F}_k(\theta)-\mathbf{K}(\theta)\ \lambda_k(\theta)\ \mathbf{d}_{\mathrm{WIN},k}$

11: 更新确定性矩阵 $\mathbf{D}_{\mathrm{WIN}}=[\mathbf{D}_{\mathrm{WIN}},\mathbf{d}_{\mathrm{WIN},k}]\in\mathbb{R}^{n\times k}$

12: 计算局部迭代误差 $\epsilon_{\mathbf{u},k}$, $k\leftarrow k+1$

13: **结束**

14: 通过公式（46）重新计算随机变量向量 $\boldsymbol{\Lambda}(\theta)\in\mathbb{R}^k$

上述算法包含两个用于检查收敛的迭代准则，即内循环第 8 步中的 $\epsilon_{\mathbf{d},j}$，以及外循环第 12 步中的 $\epsilon_{\mathbf{u},k}$。迭代误差 $\epsilon_{\mathbf{d},j}$ 定义为

$$\epsilon_{\mathbf{d},j}=\frac{\left(\mathbf{d}_{\mathrm{WIN},k}^{(j)}-\mathbf{d}_{\mathrm{WIN},k}^{(j-1)}\right)^{\mathrm{T}}\left(\mathbf{d}_{\mathrm{WIN},k}^{(j)}-\mathbf{d}_{\mathrm{WIN},k}^{(j-1)}\right)}{\mathbf{d}_{\mathrm{WIN},k}^{(j)\mathrm{T}}\mathbf{d}_{\mathrm{WIN},k}^{(j)}}=2-2\mathbf{d}_{\mathrm{WIN},k}^{(j)\mathrm{T}}\mathbf{d}_{\mathrm{WIN},k}^{(j-1)},\tag{52}$$

它衡量向量 $\mathbf{d}^{(j)}{}_{\mathrm{WIN},k}$ 与 $\mathbf{d}^{(j-1)}{}_{\mathrm{WIN},k}$ 之间的差异，当满足 $\epsilon_{\mathbf{d},j}<\epsilon_{\mathbf{d}}$ 时，计算即停止。类似地，迭代误差 $\epsilon_{\mathbf{u},k}$ 定义为

$$\begin{aligned}\epsilon_{\mathbf{u},k}&=\frac{\int_{\Theta}\left[\mathbf{u}_{\mathrm{WIN},k}(\theta)-\mathbf{u}_{\mathrm{WIN},k-1}(\theta)\right]^{\mathrm{T}}\left[\mathbf{u}_{\mathrm{WIN},k}(\theta)-\mathbf{u}_{\mathrm{WIN},k-1}(\theta)\right]\mathrm{d}\mathcal{P}(\theta)}{\int_{\Theta}\mathbf{u}_{\mathrm{WIN},k}^{\mathrm{T}}(\theta)\,\mathbf{u}_{\mathrm{WIN},k}(\theta)\,\mathrm{d}\mathcal{P}(\theta)}\\&=\frac{\int_{\Theta}\lambda_k^2(\theta)\,\mathrm{d}\mathcal{P}(\theta)\,\mathbf{d}_{\mathrm{WIN},k}^{\mathrm{T}}\mathbf{d}_{\mathrm{WIN},k}}{\sum_{i,j=1}^{k}\int_{\Theta}\lambda_i(\theta)\,\lambda_j(\theta)\,\mathrm{d}\mathcal{P}(\theta)\,\mathbf{d}_{\mathrm{WIN},i}^{\mathrm{T}}\mathbf{d}_{\mathrm{WIN},j}}=\frac{\int_{\Theta}\lambda_k^2(\theta)\,\mathrm{d}\mathcal{P}(\theta)}{\sum_{i=1}^{k}\int_{\Theta}\lambda_i^2(\theta)\,\mathrm{d}\mathcal{P}(\theta)},\end{aligned}\tag{53}$$

其衡量了第 k 对 $\{\lambda_k(\theta)$、$\mathrm{d}_{\mathrm{WIN},k}\}$ 对随机解 $\mathrm{u}_{\mathrm{WIN},k}(\theta)$ 的贡献。然而，在许多情况下，[29] 由于随机变量 $\{\lambda_i(\theta)\}^k{}_{i=1}$ 是按顺序计算的，$\int_{\Theta}\lambda^2\,\mathrm{k}(\theta)\mathrm{d}\mathcal{P}(\theta)$ 可能不会持续减小，因此公式（53）可能不是一个良好的误差指标。我们通过将公式（53）中的 $\{\lambda_i(\theta)\}^k{}_{i=1}$ 替换为等价的随机变量$\left\{\widetilde{\lambda}_i(\theta)\right\}^k{}_{i=1}$来避免这一问题。为此，我们计算随机变量向量$\mathbf{\Lambda}(\theta)=[\lambda_1(\theta)$、$\cdots,\lambda_k(\theta)]^{\mathrm{T}}$ 的自相关函数，得到

$$\mathbf{C}_{\mathbf{\Lambda\Lambda}}=\widehat{\mathbb{E}}\left\{\mathbf{\Lambda}\left(\widehat{\boldsymbol{\theta}}\right)\mathbf{\Lambda}^{\mathrm{T}}\left(\widehat{\boldsymbol{\theta}}\right)\right\}\in\mathbb{R}^{k\times k},\tag{54}$$

其通过特征分解可分解为

$$\mathbf{C}_{\mathbf{\Lambda\Lambda}}=\mathbf{QZQ}^{\mathrm{T}},\tag{55}$$

其中$\mathbf{Q}\in\mathbb{R}^{k\times k}$ 是正交矩阵，$\mathbf{Z}$ 是由矩阵$\mathbf{C}_{\mathbf{\Lambda\Lambda}}$的特征值按降序组成的对角矩阵。我们构造一个等效的随机变量向量 $\mathbf{\Lambda}(\theta)=\mathbf{Q}^{\mathrm{T}}\mathbf{\Lambda}(\theta)=\left[\widetilde{\lambda}_1(\theta),\cdots,\widetilde{\lambda}_k(\theta)\right]^{\mathrm{T}}\in\mathbb{R}^k$，其自相关函数恰好为$\mathbf{C}_{\widetilde{\mathbf{\Lambda}}\widetilde{\mathbf{\Lambda}}}=\mathbf{Z}$。将等效随机变量 $\left\{\widetilde{\lambda}_i(\theta)\right\}^k{}_{i=1}$ 代入式（53），我们重新计算迭代误差$\epsilon_{\mathbf{u},k}$，得到

$$\epsilon_{\mathbf{u},k}=\frac{\int_{\Theta}\widetilde{\lambda}_k^2(\theta)\,\mathrm{d}\mathcal{P}(\theta)}{\sum_{i=1}^{k}\int_{\Theta}\widetilde{\lambda}_i^2(\theta)\,\mathrm{d}\mathcal{P}(\theta)}=\frac{\mathbf{Z}_k}{\mathrm{Tr}(\mathbf{Z})},\tag{56}$$

其中，$\mathbf{Z}_k$ 是矩阵$\mathbf{Z}$的第k个对角元素。需要指出的是，上述重新表述并未提高随机解的近似精度，仅提供了一种等价的】【。

表示。这样，随着保留项k的增加，迭代误差$\epsilon_{\mathbf{u},k}$会持续减小。基本实现的更多细节以及式 (53) 与式 (56) 的比较可见于 [29]。

5. 数值算例

本节通过二维和三维数值算例检验所提出的两种方法。对于算法1，步骤2中内循环的收敛误差 $\epsilon_{\mathbf{d}}$ 和步骤4中外循环的收敛误差 $\epsilon_{\mathbf{u}}$ 分别设为 1×10^{-3} 和 1×10^{-6}。对于这两个算例，使用$n_s = 1\times10^4$个随机样本进行蒙特卡罗模拟并生成参考解。在我们的算例中，1×10^4个样本足以获得收敛的概率解。所提出的 WIN-SFEM 也使用相同的 1×10^4 个随机样本，以消除抽样过程造成的影响。此外，这些算例均在一台台式计算机的单个核心上运行（该计算机配备十六个核心，处理器为英特尔酷睿 i7，主频 2.50GHz）。

5.1. 算例*1*：随机虚拟元方法对二维随机问题的分析

5.1.1. 随机输入的模拟

在本例中，我们考虑图 1 [40] 所示二维模型基于随机虚拟元方法的平面应力分析。空间离散采用 Voronoi网格，共包含 2018 个顶点、1000 个沃罗诺伊单元和 4036 个自由度。如图1所示，模型在两个红点处固定。沿着

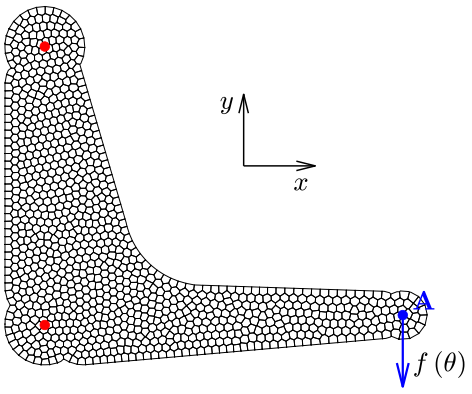


图 1：二维模型的几何形状和 Voronoi网格。

y 方向，在蓝点处施加随机力 f (θ) ()（单位：N），其中 $\xi_f(\theta)$ 是标准高斯随机变量。此外，材料属性矩阵 **G**(x, y, θ) 定义为

$$\mathbf{G}(x,y,\theta)=\frac{E(x,y,\theta)}{1-\nu^2}\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1}{2}(1-\nu)\end{bmatrix}\in\mathbb{R}^{3\times3}, \tag{57}$$

其中，泊松比为 $\nu=0.3$，杨氏模量 E(x,y, θ) 被建模为二维随机场，其均值为 E_0 (x,y) = 100 MPa，协方差函数为 [41]

$$\mathrm{Cov}_{EE}(x_1,y_1;x_2,y_2)=\sigma_E^2\left(1+\frac{|x_1-x_2|}{l_x}\right)\left(1+\frac{|y_1-y_2|}{l_y}\right)\exp\left(-\frac{|x_1-x_2|}{l_x}-\frac{|y_1-y_2|}{l_y}\right), \tag{58}$$

其中，标准差为 σ_E = 10 MPa，l_x 和 l_y 分别为 x 和 y方向上的相关长度。随机场 E(x,y, θ) 通过以下 Karhunen-Lo`eve 展开进行近似 [23, 34]

$$E(x,y,\theta)=E_0(x,y)+\sum_{i=1}^{m}\xi_i(\theta)\sqrt{\kappa_i}E_i(x,y), \tag{59}$$

其中，$\{\xi_i(\theta)\}^m{}_{i=1}$ 是相互独立的标准高斯随机变量，且它们也独立于随机变量 $\xi_f(\theta)$。 $\{\kappa_i, E_i(x,y)\}^m{}_{i=1}$ 是协方差函数 $\mathrm{Cov}_{EE}(x_1,y_1;x_2,y_2)$ 的特征值和特征向量。它们通过以下第二类 Fredholm积分方程求解

$$\int_\Omega \mathrm{Cov}_{EE}(x_1,y_1;x_2,y_2)\,E_i(x_1,y_1)\,\mathrm{d}x_1\mathrm{d}y_1=\kappa_i E_i(x_2,y_2), \tag{60}$$

利用现有的特征值求解器 [42] 可以高效求解。为确保 E(x,y, θ) > 0，在数值实现中会剔除满足 $\min_{x,y\in\Omega}E\left(x,y,\theta^{(i)}\right)\le 1\times10^{-3}$ 的样本 $\theta^{(i)}$。这样，式（29）中的确定性矩阵 $\{\mathbf{G}_i(x,y)\}^m{}_{i=0}$ 可表示为

$$\mathbf{G}_i(x,y)=\frac{\sqrt{\kappa_i}E_i(x,y,\theta)}{1-\nu^2}\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1}{2}(1-\nu)\end{bmatrix}\in\mathbb{R}^{3\times3},\ \text{where } \kappa_0\equiv1. \tag{61}$$

在此示例中，我们通过在方程 (59) 中引入不同的截断项 m ，考虑低维情形和高维情形。对于低维情形，我们令相关长度为 l_x = max(x) – min(x) 和 l_y = max(y) –min(y。为实现截断

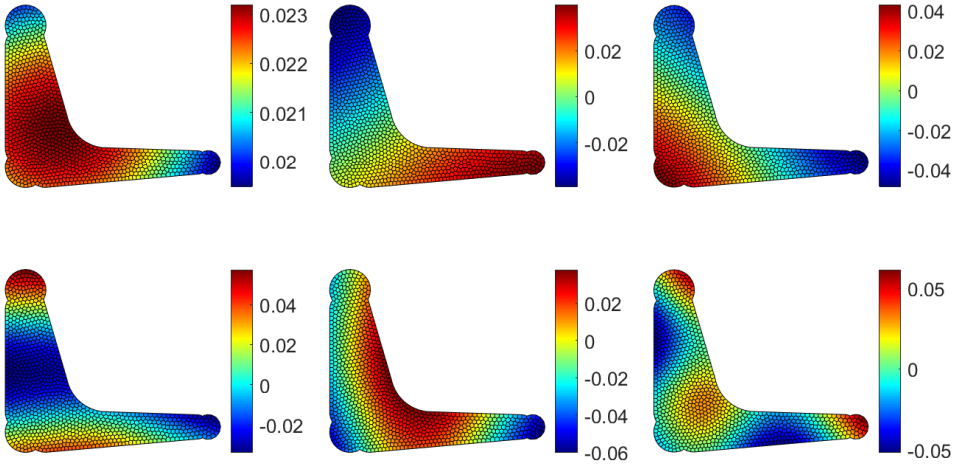


图2：E igenvectors $\{E_i(x,y)\}^6{}_{i=1}$ 的低维 cas 中协方差函数 $\mathrm{Cov}_{EE}(x_1,y_1;x_2,y_2)$ 的特征向量 e.

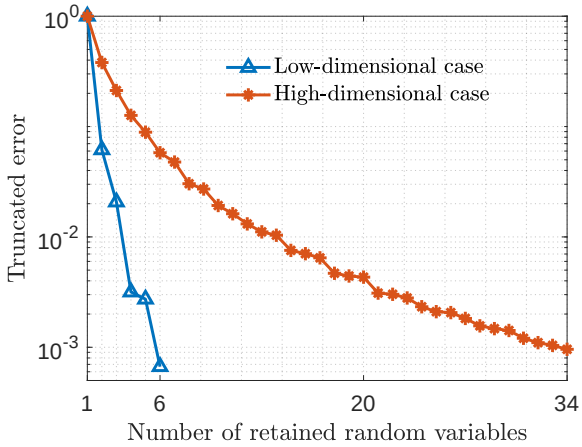


图3：保留不同数量随机变量时的截断误差。

误差 $\kappa_m/\sum^m \mathrm{i}_{=1}\kappa_i < 1\times10^{-3}$，截断项的数量为 m = 6。协方差函数 $\mathrm{Cov}_{EE}(\mathrm{x}_1,\mathrm{y}_1;\mathrm{x}_2,\mathrm{y}_2)$ 对应的特征向量 $\{\mathrm{E}_i(\mathrm{x},\mathrm{y})\}^6\,\mathrm{i}_{=1}$ 如图 2 所示。值得注意的是，我们可以使用 $\mathrm{Cov}_{EE}(\mathrm{x}_1,\mathrm{y}_1;\mathrm{x}_2,\mathrm{y}_2)$ 的离散化协方差矩阵 $\mathrm{Cov}_{EE}\in\mathbb{R}^{2018\times2018}$ 求解公式（60），该矩阵仅取决于顶点，与单元无关。这样，图 2 中的特征向量便绘制在网格的每个顶点上。对于高维情形，相关长度设为 $\mathrm{l}_x=\frac{1}{8}[\max(\mathrm{x})-\min(\mathrm{x})]$ 和 $\mathrm{l}_y=\frac{1}{8}[\max(\mathrm{y})-\min(\mathrm{y})]$，并保留 m = 34 个截断项以达到 $\kappa_{34}/\sum^{34}\mathrm{i}_{=1}\kappa_i < 1\times10^{-3}$。低维和高维情形的截断误差见图 3。由于相关长度较小，在彩票论坛 - -

高维情形下，收敛到指定截断误差的速度较慢，并且需要更多截断项来捕捉局部相关性。

5.1.2. 低维情形

在本节中，我们使用所提出的 PC-SVEM 和 WIN-SVEM 求解低维情形。对于 PC-SVEM，采用由七个标准高斯随机变量 $(\{\xi_i(\theta)\}_{i=1}^{6}$ 和 $\xi_f(\theta))$ 构成的二阶 Hermite 多项式混沌基，用于 $\{\Gamma_i(\theta)\}_{i=1}^{36}$。增广确定性方程 (34) 的规模为 145296，远大于原随机问题的 4036 个自由度。在本例中，我们不对更高阶多项式混沌基进行数值实现。一方面，二阶多项式混沌基足以实现对随机解的良好近似。另一方面，如果采用三阶多项式混沌基，增广方程 (34) 的规模将达到 484320，对于空间尺度如此小的问题而言，这会在存储和求解方面带来过高的计算负担。

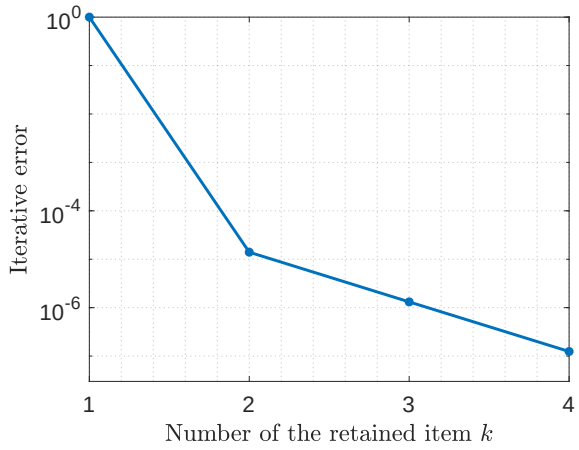


图 4：低维情形下不同保留项的迭代误差。

对于 WIN-SVEM，执行算法1，不同保留项的迭代误差 $\epsilon_{\mathbf{u},\,k}$ 如图4所示。随机解近似公式（38）仅保留四项 $k = 4$，这表明所提出的 WIN-SVEM 具有良好的收敛性。此外，确定性向量 $\{\mathbf{d}_{\mathrm{WIN},\,i}\}_{i=1}^{4}$ 在 x 和 y 方向上的分量 $\left\{\mathbf{d}_{\mathbf{u}_x,\,i} \in \mathbb{R}^{2018}\right\}_{i=1}^{4}$ 和 $\left\{\mathbf{d}_{\mathbf{u}_y,\,i} \in \mathbb{R}^{2018}\right\}_{i=1}^{4}$ 分别展示在图5的第一行和第二行中，而通过公式（46）求得的相应随机变量 $\{\lambda_i(\theta)\}_{i=1}^{4}$ 的概率密度函数（PDF）见

图5的第三行。随着保留项 k 增加，随机变量$\{\lambda_i(\theta)\}^4{}_{i=1}$ 的取值范围变得更小，并且更加集中于零附近，这表明后续保留项对随机解的贡献越来越小。

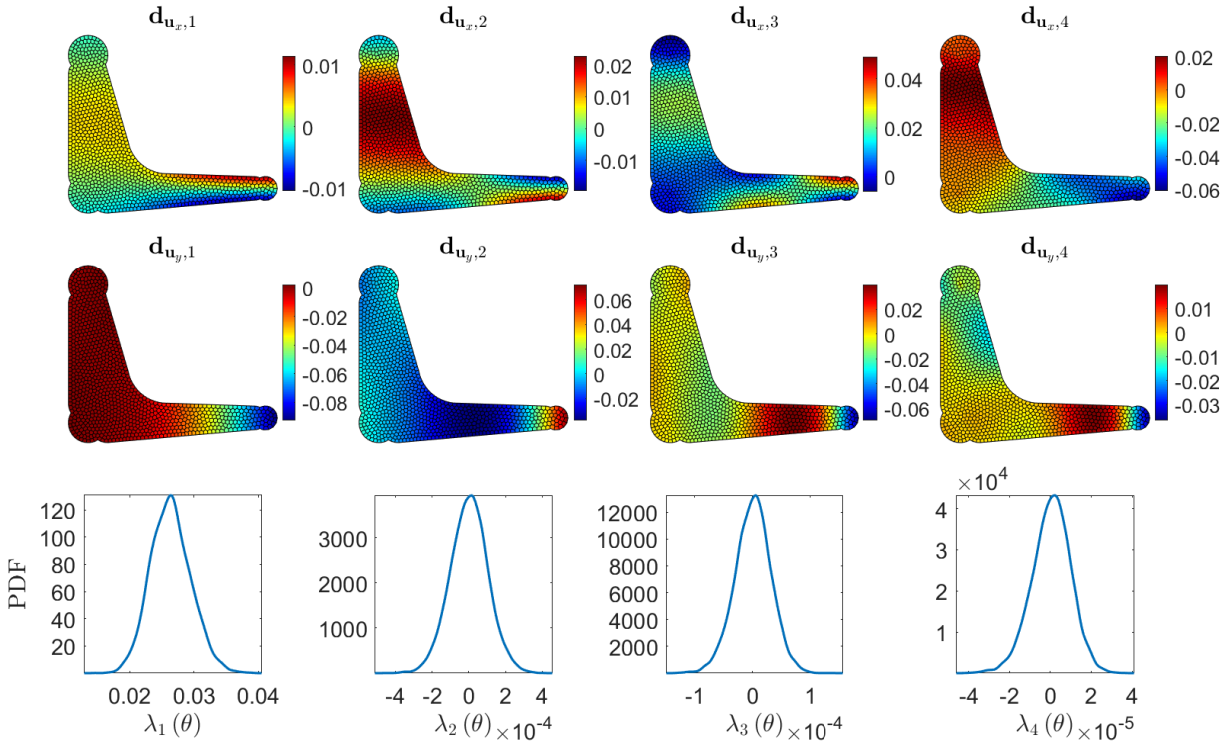
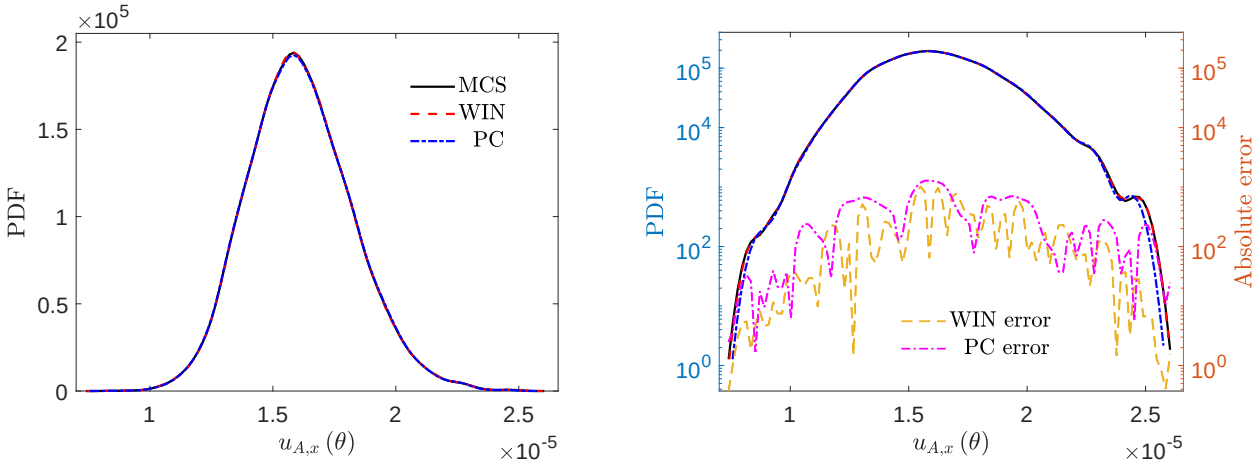


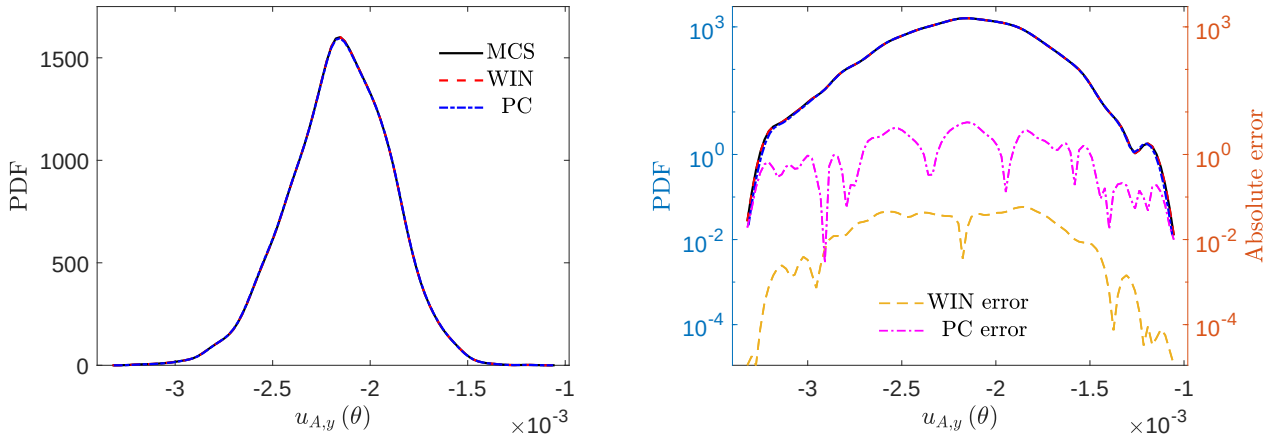
图5：随机解的分量：沿x 方向的确定性向量 $\{\mathbf{d}_{\mathbf{u}_x,i}\}^4{}_{i=1}$ （第一行），沿y 方向的确定性向量 $\{\mathbf{d}_{\mathbf{u}_y,i}\}^4{}_{i=1}$ （第二行），以及随机变量 $\{\lambda_i(\theta)\}_{i=1}^4$ 的概率密度函数（第三行）。

为展示所提出的 PC-SVEM 和 WIN-SVEM 的计算精度，我们在图6中比较了采用基于 PC-SVEM、WIN-SVEM 和 MCS 的随机虚拟元方法得到的点 A（即图1中力 $f\theta$ 作用处的蓝色点）x 和y 方向上的随机位移概率密度函数 $u_{A,x}(\theta)$ 和 $u_{A,y}(\theta)$，以及它们的绝对误差。从图6a和图6c可以看出，PC-SVEM 和 WIN-SVEM 得到的随机位移 $u_{A,x}(\theta)$ 和 $u_{A,y}(\theta)$ 的概率密度函数与 MCS 的结果吻合良好，说明这两种所提出的方法具有较高的精度。图6b和图6d所示的对数刻度比较表明，WIN-SVEM 的绝对误差小于 PC-SVEM，尤其是对于$u_{A,y}(\theta)$ 这一随机位移的y 方向分量。此外，WIN-SVEM 能比 PC-SVEM 更准确地捕捉概率密度函数的尾部

，这对于许多不确定性量化问题非常有用，例如模拟具有长尾概率分布的物理现象，以及在结构可靠性分析中估计较小的失效概率。因此，对于有此类需求的问题，推荐使用WIN-SVEM。



（a）PC-方法获得的点 A 在x 方向上的随机位移概率密度函数对比；（b）PC-和 WIN-SVEMs 方法分别获得的点 A 在x 方向上的$u_{A,x}$ (θ) 概率密度函数的绝对误差。WIN-SVEMs 相对于蒙特卡罗模拟参考概率密度函数。



（c）PC-方法获得的点 A 在y 方向上的随机位移概率密度函数对比；（d）PC-和 WIN-SVEMs 方法分别获得的点 A 在y 方向上的$u_{A,y}$ (θ) 概率密度函数的绝对误差。WIN-SVEMs 相对于蒙特卡罗模拟参考概率密度函数。

图 6：通过 PC-SVEM、WIN-SVEMs 和蒙特卡罗模拟得到的点 A 在x 和y 方向上的随机位移$u_{A,x}$ (θ) 和$u_{A,y}$(θ) 的概率密度函数及其绝对误差。

此外，我们来关注所提方法的计算效率。表1第二至第四列列出了 PC-SVEM、WIN-SVEM 和蒙特卡罗模拟数值执行所需的计算时间（单位：秒），其中 WIN-SVEM 的求解时间和重新计算时间分别是算法1步骤 2 至步骤 13 以及重新计算步骤 14 的计算时间。WIN-SVEM 的重新计算过程成本几乎可以忽略，因为对于不同的样本实现，只需求解规模为 4 的确定性方程。结果发现，PC-SVEM 和 WIN-SVEM 的计算成本都远低于蒙特卡罗模拟。然而，由于 PC-SVEM 需要求解一个规模非常大的确定性方程，其计算量高于 WIN-SVEM。

表1：随机维数为7和35时的计算成本。

方法	WIN	PC	MCS	WIN	MCS
随机维数	7			35	
求解时间	0.51			12.33	
重新计算时间	0.04			0.35	
总时间（秒）	0.55	26.09	159.13	12.68	221.52

5.1.3. 高维情形

在本节中，我们求解总共包含35个随机维数的高维情形（$\{\xi_i(\theta)\}^{34}_{i=1}$ 和 $\xi_f(\theta)$）。本节仅采用 WIN-SVEM。对于 PC-SVEM，即使仅使用二阶多项式混沌基，所得确定性方程（34）的规模也约为 2.69×10^6，这会遭遇维数灾难。采用算法1后，高维情形下不同保留项对应的迭代误差 $\epsilon_{\mathbf{u},k}$ 如图7所示。对于高维随机问题，所提出的 WIN-SVEM 仍具有良好的收敛性。在此情形下保留了 $k=6$ 项，与低维情形相比略有增加。该情形下的计算时间见表1第五列和第六列。即使在高维情形下，WIN-SVEM 的计算成本也很低，远低于蒙特卡罗模拟。与低维情形相比，随着随机维数增加，高维情形的计算成本并未大幅上升。从这些方面来看，所提出的 WIN-SVEM 避免了

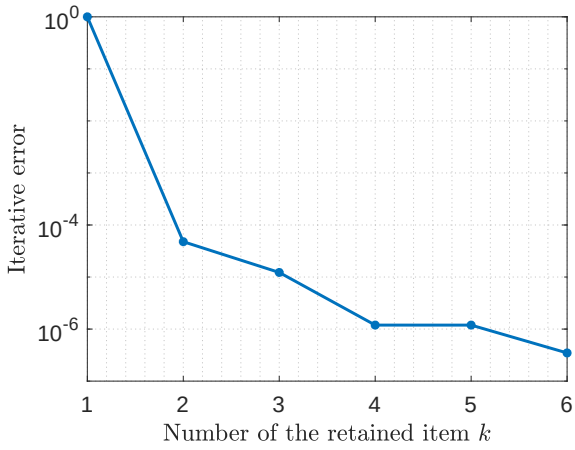
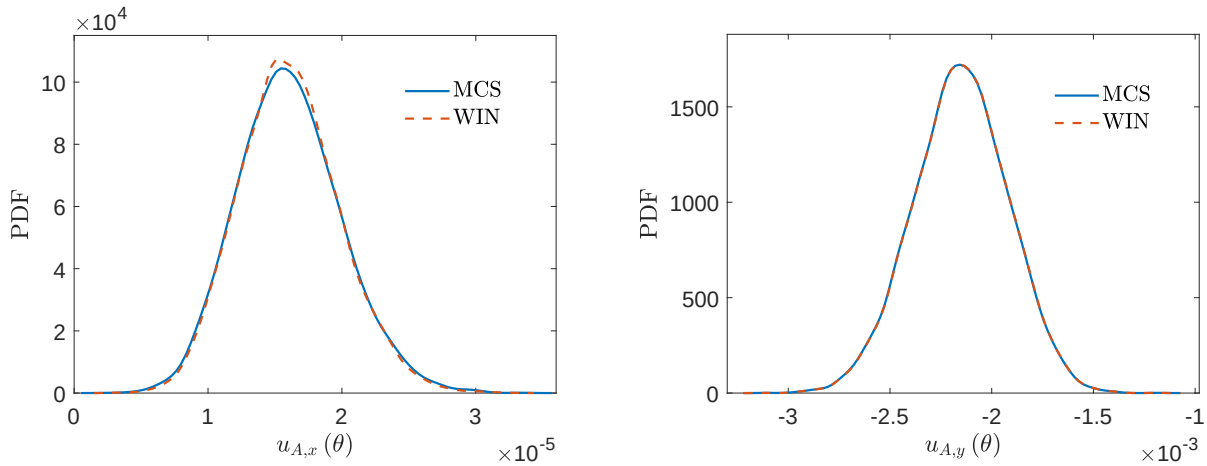


图 7：高维情形下不同保留项的迭代误差。

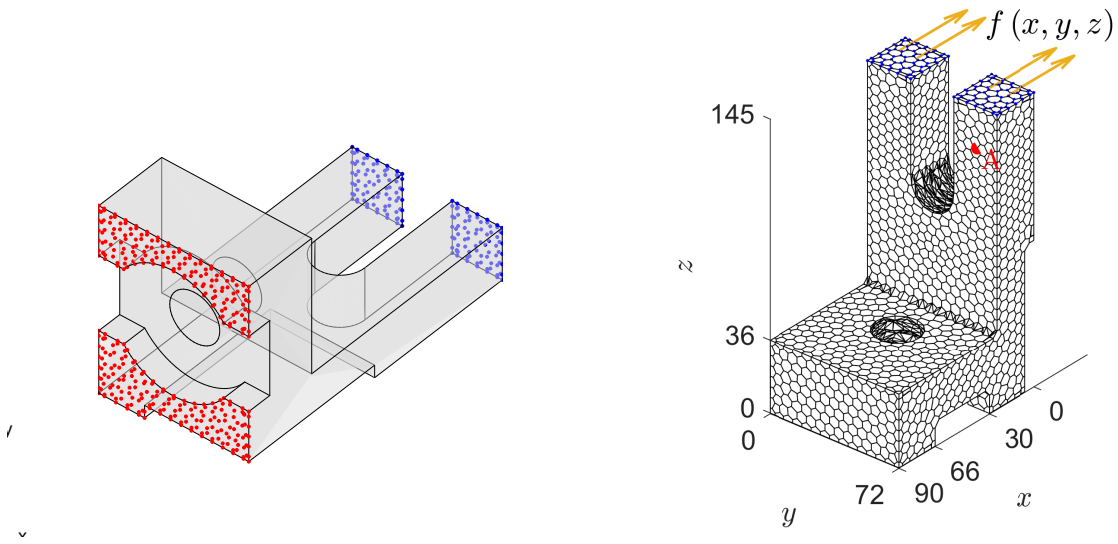
成功克服维数灾难。此外，图 8 对比了通过 WIN-SVEM 和 MCS 得到的点 A 在 x 和 y 方向上的随机位移概率密度函数$u_{A,x}(\theta)$ 和$u_{A,y}(\theta)$。对于这两种随机位移，WIN-SVEM 的计算精度仍与 MCS 相当。



（a）通过 WIN-$u_{A,\ x}(\theta)$ 获得的点 A 在x 方向上的随机位移的概率密度函数与通过 WIN- SVEM 和蒙特卡罗模拟获得的结果的比较。（b）通过 WIN-$u_{A,\ y}(\theta)$ 获得的点 A 在y 方向上的随机位移的概率密度函数与通过 SVEM 和蒙特卡罗模拟获得的结果的比较。

图 8：分别由 WIN-SVEM 和蒙特卡罗模拟（MCS）得到的点 A 沿x 和y 方向的随机位移 $u_{A,x}(\theta)$（左）和 $u_{A,\ y}(\theta)$（右）的概率密度函数。

5.2. 示例*2*：三维机械零件的随机虚拟元方法分析



(a) 几何模型。　　(b) Voronoi网格。

图9：三维机械零件的几何形状（左）及其Voronoi网格（右）。

本例中，我们对图 9a 所示的三维机械零件进行 SVEM 分析，在红色表面施加狄利克雷边界条件$\mathbf{u}_x(\theta) = \mathbf{u}_y(\theta) = \mathbf{u}_z(\theta) = \mathbf{0}$，并在蓝色表面沿$x$方向施加外力$f(x, y, z) = -500$ N/mm^2。模型采用图 9b 所示的 Voronoi网格进行离散，共包含 28232 个顶点、4389 个单元和 84696 个自由度。本例中的随机材料矩阵$\mathbf{G}(x, y, z, \theta)$定义为

$$\mathbf{G}(x,y,z,\theta) = \frac{E(x,y,z,\theta)}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2}-\nu & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2}-\nu & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2}-\nu \end{bmatrix} \in \mathbb{R}^{6\times 6}, \tag{62}$$

其中，泊松比为 $\nu = 0.3$，杨氏模量$E(x, y, z, \theta)$是一个三维随机场，其协方差函数为

$$\mathrm{Cov}_{EE}\left(x_1,y_1,z_1;x_2,y_2,z_2\right)=\sigma_E^2\exp\left(-\frac{|x_1-x_2|}{l_x}-\frac{|y_1-y_2|}{l_y}-\frac{|z_1-z_2|}{l_z}\right),\tag{63}$$

其中，标准差为 $\sigma_E=41.8$ GPa，相关长度由$l_x=\max(x)-\min(x)$、$l_y=\max(y)-\min(y)$和 $l_z=\max(z)-\min(z)$ 给出。随机场$E(x,y,z,\theta)$ 具有类似于式 (59) 的级数展开

$$E\left(x,y,z,\theta\right)=E_0\left(x,y,z\right)+\sum_{i=1}^{m}\xi_i\left(\theta\right)\sqrt{\kappa_i}E_i\left(x,y,z\right),\tag{64}$$

其中，函数 $E_0(x,y,z)=208$ GPa，$\{\xi_i(\theta)\}_{i=1}^{m}$ 是定义在 $[0,1]$ 上的相互独立的均匀随机变量。需要指出的是，随机场 $E(x,y,z,\theta)$ 的均值为 $E_0(x,y,z)+0.5\sum_{i=1}^{m}\sqrt{\kappa_i}E_i(x,y,z)$，而非 $E_0(x,y,z)$。类似地，$\{\kappa_i,E_i(x,y)\}_{i=1}^{m}$ 是协方差函数 $\mathrm{Cov}_{EE}(x_1,y_1,z_1;x_2,y_2,z_2)$ 的特征值和特征向量，可通过类似公式（60）的积分方程求解。本例中截断项数设为 $m=13$，以达到截断误差$\kappa_{13}\Big/\sum_{i=1}^{13}\kappa_i<1\times10^{-3}$。

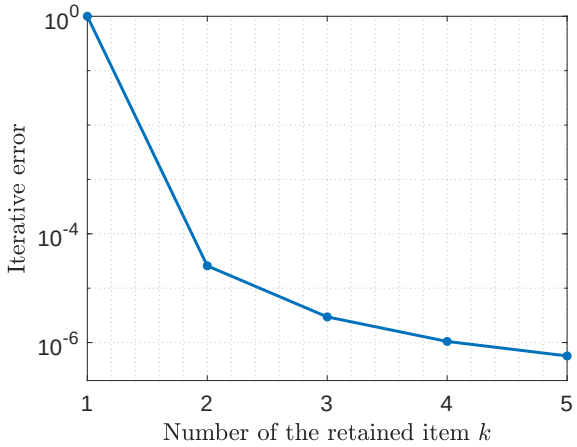
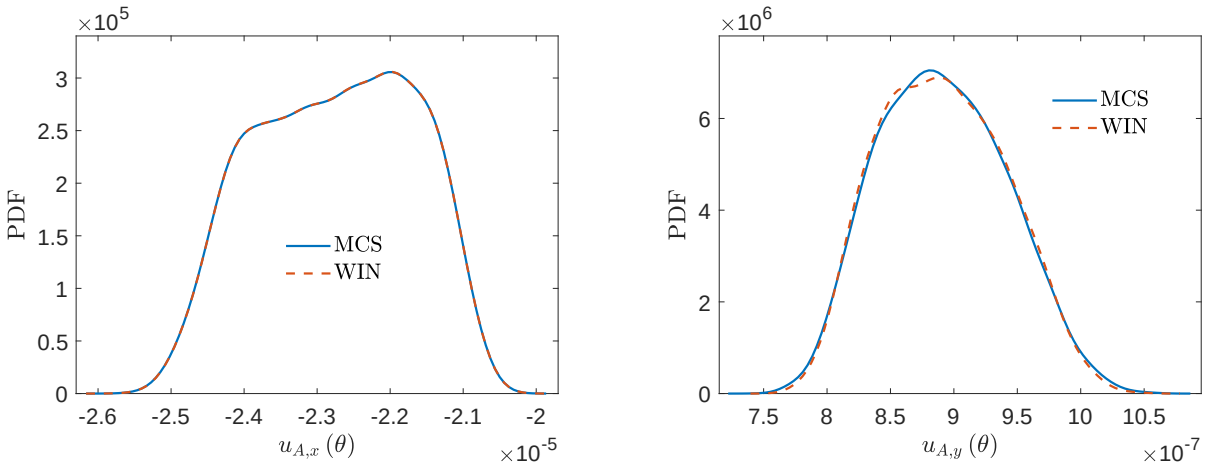
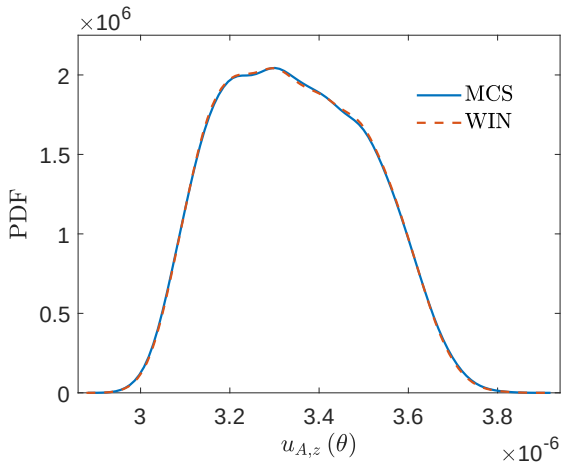


图10：不同保留项的迭代误差。

与上一节中的示例类似，在本例中，PC-SVEM 也会遭遇维数灾难，因为导出的确定性方程 (34) 的规模约为 8.89×10^6，即使只使用二阶多项式混沌基也是如此。因此，本问题仅采用 WIN-SVEM 求解。图10展示了不同保留项数对应的迭代误差。保留五项即可满足指定的收敛误差，这验证了 WIN-SVEM 在三维问题中的良好收敛性。



(a) 随机位移概率密度函数的比较 (b) 分别由 WIN-$u_{A,x}(\theta)$ 和 WIN-SVEM 与蒙特卡罗模拟得到的点 A 的x 方向随机位移概率密度函数的比较。$u_{A,y}(\theta)$ 在点 A 的y 方向上。分别为 SVEM 和蒙特卡罗模拟。



(c) 分别由 WIN-SVEM 和蒙特卡罗模拟得到的点 A 的z 方向随机位移$u_{A,z}(\theta)$概率密度函数的比较。

图 11：分别由 WIN-SVEM 和蒙特卡罗模拟得到的点 A 的x、y和z方向随机位移$u_{A,x}(\theta)$（左上）、$u_{A,y}(\theta)$（右上）和$u_{A,z}(\theta)$（下）的概率密度函数。

随机问题。关于计算精度，图11比较了通过 WIN-SVEM 和蒙特卡罗模拟（MCS）得到的点 A（如图9b所示）在x、y 和z 方向上的随机位移$u_{A,x}(\theta)$、$u_{A,y}(\theta)$ 和$u_{A,z}(\theta)$ 的概率密度函数。对于随机位移$u_{A,x}(\theta)$ 和$u_{A,z}(\theta)$ 的概率密度函数，WIN-SVEM 与蒙特卡罗模拟（MCS）的结果非常吻合。随机位移的概率密度函数

位移 $u_{A,y}(\theta)$ 的精度略低于 $u_{A,x}(\theta)$ 和 $u_{A,z}(\theta)$，但仍达到了可接受的精度。如果实际应用中需要更高精度，我们只需在随机解近似中保留更多项即可进一步提高精度。此外，本例中 WIN-SVEM 和蒙特卡罗模拟的计算时间分别为 213.45s（包括求解时间 212.89s 和重新计算时间 0.56s）和 2.74×10^4s。所提出的 WIN-SVEM 仍比蒙特卡罗模拟便宜得多，提速超过百倍。

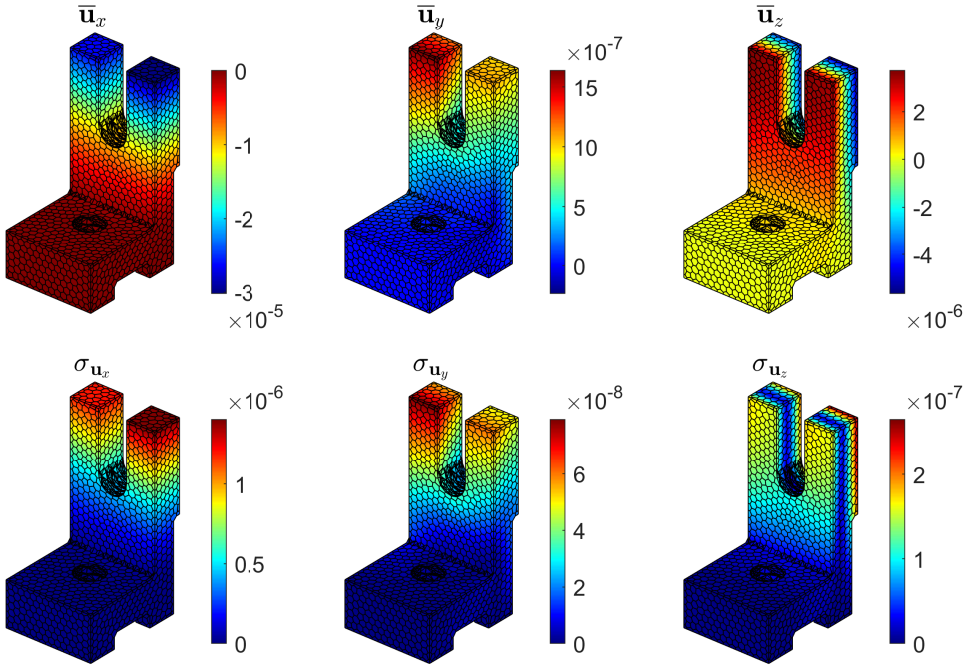


图 12：均值函数 $\mathbf{u}_x$、$\mathbf{u}_y$ 和 $\mathbf{u}_z$（第一行）在 x、y 和 z 方向上的分布，以及标准差函数 $\boldsymbol{\sigma}_{\mathbf{u}_x}$、$\boldsymbol{\sigma}_{\mathbf{u}_y}$ 和 $\boldsymbol{\sigma}_{\mathbf{u}_z}$（第二行）在 x、y 和 z 方向上的分布。

我们还要强调，基于 WIN-SVEM 可以轻松计算随机解的统计性质。此处我们关注一阶和二阶全局统计矩，即随空间位置变化的均值和标准差。根据式（38），均值向量$\mathbf{u}$ 计算如下#+#+#+#+

$$\bar{\mathbf{u}} = \sum_{i=1}^{k} \widehat{\mathbb{E}}\left\{\lambda_i\left(\widehat{\boldsymbol{\theta}}\right)\right\} \mathbf{d}_{\mathrm{WIN},i}, \tag{65}$$

它只涉及样本向量 $\left\{\lambda_i\left(\widehat{\boldsymbol{\theta}}\right) \in \mathbb{R}^{10^4}\right\}_{i=1}^{k}$ 的期望，计算量非常小。$\mathbf{u}$ 在x、y 和z 方向上的均值分量$\mathbf{u}_x$、$\mathbf{u}_y$ 和$\mathbf{u}_z$ 如图所示

见图 12 第一行。此外，标准差向量 $\boldsymbol{\sigma}_{\mathbf{u}}$ 通过以下方式计算：

$$\begin{aligned}\boldsymbol{\sigma}_{\mathbf{u}} &= \sqrt{\sum_{i,j=1}^{k} \widehat{\mathbb{E}}\left\{\left[\lambda_i\left(\widehat{\boldsymbol{\theta}}\right) - \widehat{\mathbb{E}}\left\{\lambda_i\left(\widehat{\boldsymbol{\theta}}\right)\right\}\right] \odot \left[\lambda_j\left(\widehat{\boldsymbol{\theta}}\right) - \widehat{\mathbb{E}}\left\{\lambda_j\left(\widehat{\boldsymbol{\theta}}\right)\right\}\right]\right\} \mathbf{d}_{\mathrm{WIN},i} \odot \mathbf{d}_{\mathrm{WIN},j}} \\ &= \sqrt{\sum_{i,j=1}^{k} \left[\widehat{\mathbb{E}}\left\{\lambda_i\left(\widehat{\boldsymbol{\theta}}\right) \odot \lambda_j\left(\widehat{\boldsymbol{\theta}}\right)\right\} - \widehat{\mathbb{E}}\left\{\lambda_i\left(\widehat{\boldsymbol{\theta}}\right)\right\} \widehat{\mathbb{E}}\left\{\lambda_j\left(\widehat{\boldsymbol{\theta}}\right)\right\}\right] \mathbf{d}_{\mathrm{WIN},i} \odot \mathbf{d}_{\mathrm{WIN},j}},\end{aligned} \tag{66}$$

其在x、y 和z 方向上的分量 $\boldsymbol{\sigma}_{\mathbf{u}_x}$、 $\boldsymbol{\sigma}_{\mathbf{u}_y}$ 和 $\boldsymbol{\sigma}_{\mathbf{u}_z}$ 可在图 12 第二行中找到。

6. 结论

本文提出了两种数值方法，即 PC-SVEM 和 WIN-SVEM，用于求解由二维和三维线弹性随机问题的随机虚单元离散导出的随机系统。首先，将确定性虚元法扩展为随机虚拟元方法（SVEM），以处理随机材料属性和随机外力等。SVEM 的若干关键计算可沿用确定性虚元法，例如式 (17) 中虚拟基函数的梯度计算，以及式 (24) 中稳定化单元刚度矩阵的计算（随机系数 $\gamma^{(\mathrm{e})}s(\theta)$ 除外）。数值结果表明，PC-SVEM 和 WIN-SVEM 的精度均与蒙特卡罗模拟相当。然而，PC-SVEM 受到维数灾难的影响，无法用于高维随机问题。相比之下，WIN-SVEM 无需任何修改即可高效求解低维和高维随机问题，这一点已通过一个随机维数高达 35 的数值算例得到验证。此外，尽管本文仅研究了线弹性随机问题，但所提出的 PC-SVEM 和 WIN-SVEM 均可应用于更一般的情况，后续研究将对此作进一步探讨。

致谢

作者感谢亚历山大·冯·洪堡基金会，以及由德国研究基金会（DFG）资助的国际研究培训小组 2657（IRTG 2657）（资助编号 433082294）。

参考文献

[1] A. Quarteroni, A. Valli, 《偏微分方程数值近似》, Springer Science & Business Media, 2008 年。[2] L. Beir˜ao da Veiga, F. Brezzi, A. Cangiani, G. Manzini, L. D. Marini, A. Russo, 《虚单元方法的基本原理》, 《应用科学中的数学模型与方法》23 (2013) 199–214。[3] L. Beir˜ao da Veiga, F. Brezzi, L. D. Marini, A. Russo, 《虚单元方法指南》, 《应用科学中的数学模型与方法》24 (2014) 1541–1573。[4] M. Mengolini, M. F. Benedetto, A. M. Arag´on, 《虚单元方法的工程视角及其与标准有限元方法的相互作用》, 《计算机方法：应用力学与工程》350 (2019) 995–1023。[5] P. F. Antonietti, L. B. da Veiga, G. Manzini, 《虚单元方法及其应用》, Springer Science & Business Media, 2022 年。[6] L. B. Da Veiga, F. Brezzi, L. D. Marini, 《用于线弹性问题的虚单元》, 《SIAM 数值分析杂志》51 (2013) 794–812。[7] A. L. Gain, C. Talischi, G. H. Paulino, 《任意多面体网格上三维线弹性问题的虚单元方法》, 《计算机方法：应用力学与工程》282 (2014) 132–160。[8] E. Artioli, L. Beir˜ao da Veiga, C. Lovadina, E. Sacco, 《多边形网格上的任意阶二维虚单元：第一部分，弹性问题》, Co mputational Mechanics 60 (2017) 355–377。[9] H. Chi, L. B. Da Veiga, G. Paulino, 《有限变形下虚单元方法（VEM）的一些基本公式》, 《计算机方法：应用力学与工程》318 (2017) 148–192。[10] P. Wriggers, M. De Bellis, B. Hudobivnik, 《适用于大不可压缩应变的 Taylor–Hood 型虚单元公式》, 《计算机方法：应用力学与工程》385 (2021) 114021。[11] P. Wriggers, W. T. Rust, B. Reddy, 《接触虚单元方法》, Computational Mechanics 58 (2016) 1039–1050。[12] P. Wriggers, W. T. Rust, 《一种包含大变形的摩擦接触虚单元方法》, Engineering Computations 36 (2019) 2133–2161。[13] F. Aldakheel, B. Hudobivnik, E. Artioli, L. B. da Veiga, P. Wriggers, 《接触力学的曲线虚单元》, 《计算机方法：应用力学与工程》372 (2020) 113394。[14] M. F. Benedetto, S. Berrone, S. Pieraccini, S. Scial`o, 《用于离散裂缝网络模拟的虚单元方法》, 《计算机方法：应用力学与工程》280 (2014) 135–156。[15] F. Aldakheel, B. Hudobivnik, A. Hussein, P. Wriggers, 《利用高效虚单元格式进行脆性断裂相场建模》, 《计算机方法：应用力学与工程》341 (2018) 443–466。[16] A. Hussein, B. Hudobivnik, P. Wriggers, 《结合自适应相场与离散切割方法的

裂纹路径预测，《计算机方法：应用力学与工程》372 (2020) 113329。[17] P. F. Antonietti, M. Bruggi, S. Scacchi, M. Verani，关于用于多边形网格拓扑优化的虚单元方法：一项数值研究，《计算机与应用数学》74 (2017) 1091–1109。[18] H. Chi, A. Pereira, I. F. Menezes, G. H. Paulino，基于虚单元方法（VEM）的拓扑优化：一个集成框架，《结构与多学科优化》62 (2020) 1089–1114。[19] R. C. Smith，不确定性量化：理论、实现与应用，第12卷，SIAM，2013。[20] G. Stefanou，随机有限元法：过去、现在与未来，《计算机方法：应用力学与工程》198 (2009) 1031–1051。[21] M. Papadrakakis, V. Papadopoulos，采用蒙特卡罗模拟进行随机有限元分析的稳健高效方法，《计算机方法：应用力学与工程》134 (1996) 325–340。[22] I. G. Graham, F. Y. Kuo, D. Nuyens, R. Scheichl, I. H. Sloan，具有随机系数的椭圆型偏微分方程的准蒙特卡罗方法及其应用，《计算物理学杂志》230 (2011) 3668–3694。[23] R. G. Ghanem, P. D. Spanos，随机有限元：谱方法，Courier Corporation，2003。[24] D. Xiu, G. E. Karniadakis，用于随机微分方程的 Wiener–Askey多项式混沌，《SIAM科学计算杂志》24 (2002) 619–644。[25] I. Babuˇska, F. Nobile, R. Tempone，一种用于具有随机输入数据的椭圆型偏微分方程的随机配置法，SIAM 数值分析杂志45 (2007) 1005–1034。[26] D. Xiu，随机计算数值方法：谱方法，普林斯顿大学出版社，2010。[27] A. I. Khuri, S. Mukhopadhyay，响应面方法，《Wiley跨学科评论：计算统计学》2 (2010) 128–149。[28] J. N. Fuhg, A. Fau, U. Nackenhorst，克里金法自适应采样方法的最新进展与比较综述，《计算工程方法档案》28 (2021) 2689–2747。[29] Z. Zheng, M. Beer, H. Dai, U. Nackenhorst，一种用于随机结构动力学分析的弱侵入式随机有限元法，《计算机方法：应用力学与工程》399 (2022) 115360。[30] Z. Zheng, M. Valdebenito, M. Beer, U. Nackenhorst，一种求解定义在随机域上的偏微分方程的随机有限元方案，《计算机方法：应用力学与工程》405 (2023) 115860。[31] K. Yosida，泛函分析，Springer Science & Business Media，2012。[32] L. B. Da Veiga, F. Dassi, A. Russo，多面体网格上的高阶虚单元法，《计算机与应用数学》74 (2017) 1110–1122。[33] S. Sakamoto, R. Ghanem，用于模拟非高斯非平稳随机过程的多项式混沌分解，《工程力学杂志》128 (2002) 190–201。[34] Z. Zheng, H. Dai，通过Karhunen–Lo`eve展开模拟多维随机场，《计算机

《计算机方法：应用力学与工程》324 (2017) 221–247。[35] Z. Zheng, H. Dai, Y. Wang, W. Wang, 一种模拟非平稳非高斯随机过程的基于样本的迭代方案，《机械系统与信号处理》151 (2021) 107420。[36] M. F. Pellissetti, R. G. Ghanem，随机有限元背景下线性方程组的迭代求解，《工程软件进展》31 (2000) 607–616。[37] A. Keese, H. G. Matthies，随机有限元方程求解的分层并行化，《计算机与结构》83 (2005) 1033–1047。[38] G. Blatman, B. Sudret，一种为随机有限元分析构建稀疏多项式混沌展开的自适应算法，《概率工程力学》25 (2010) 183–197。[39] D. M. Young，大型线性系统的迭代求解，爱思唯尔，2014。[40] C. Talischi, G. H. Paulino, A. Pereira, I. F. Menezes，Polymesher：一个用于多边形单元的通用网格生成器，以MATLAB编写，《结构与多学科优化》45 (2012) 309–328。[41] P. D. Spanos, M. Beer, J. Red-Horse，采用修正指数协方差核的随机过程Karhunen–Lo`eve展开，《工程力学杂志》133 (2007) 773–779。[42] Y. Saad，大型特征值问题的数值方法，SIAM，2011。