Efficient uncertain k eff computations with the Monte Carlo resolution of generalised Polynomial Chaos Based reduced models

Gaël Poëtte, Emeric Brun

Efficient uncertain k_{eff} computations with the Monte Carlo resolution of generalised Polynomial Chaos Based reduced models

Gaël Poëtte[a], Emeric Brun[b]

[a] *CEA DAM CESTA, F-33114 Le Barp, France*
[b] *CEA DEN SERMA, ???, France*

Abstract

In this paper, we are interested in taking into account uncertainties for k_{eff} computations in neutronics. More generally, the material of this paper can be applied to propagate uncertainties in eigenvalue/eigenvector computations for the linear Boltzmann equation. In [1, 2], an intrusive MC solver for the gPC based reduced model of the instationary linear Boltzmann equation has been put forward. The MC-gPC solver presents interesting characteristics (mainly a better efficiency than non-intrusive strategies and spectral convergence): our aim is to recover these characteristics in an eigenvalue/eigenvector estimation context. This is done in practice at the price of few well identified modifications of an existing Monte Carlo implementation.

Keywords: Transport, Monte Carlo, Numerical scheme, Neutronics, MC-gPC, gPC, Uncertainty Quantification, intrusive, k_{eff}

1. Introduction

In this article, we are interested in the Monte Carlo (MC) resolution of the uncertain eigenvalue problem for the transport equation. In particular, we are interested in neutronics but many other physical applications could benefit the results of this paper (biology [3], socio-economics [4, 5, 6], epidemiology [7] etc.). Solving an eigenvalue problem for the transport equation consists in looking for the couple (k_{eff}, u), see [8, 9], satisfying the following partial differential equation (PDE):

$$\begin{aligned}\mathbf{v}\cdot\nabla_{\mathbf{x}}u(\mathbf{x},\mathbf{v})+v\sigma_t(\mathbf{x},\mathbf{v})u(\mathbf{x},\mathbf{v}) =& v\sigma_s(\mathbf{x},\mathbf{v})\int P_s(\mathbf{x},\mathbf{v}\cdot\mathbf{v}')u(\mathbf{x},\mathbf{v}')\,\mathrm{d}\mathbf{v}',\\ &+\frac{v\nu_f(\mathbf{x},\mathbf{v})\sigma_f(\mathbf{x},\mathbf{v})}{k_{\text{eff}}}\int P_f(\mathbf{x},\mathbf{v}\cdot\mathbf{v}')u(\mathbf{x},\mathbf{v}')\,\mathrm{d}\mathbf{v}'.\end{aligned} \tag{1}$$

In the above expression, k_{eff} is the first eigenvalue, and u, its corresponding eigenvector. Variables $\mathbf{x}\in\mathcal{D}\subset\mathbb{R}^3$ and $\mathbf{v}\in\mathcal{V}\subset\mathbb{R}^3$ are respectively the space and velocity variables. The last one may be decomposed into $\mathbf{v}=v\omega$ where $v=|\mathbf{v}|\in\mathbb{R}^+$ and $\omega=\frac{\mathbf{v}}{v}\in\mathbb{S}^2$. The cross-sections $\sigma_t=\sigma_a+\sigma_s+\sigma_f=\sigma_t(\mathbf{x},\mathbf{v})$, $\sigma_s=\sigma_s(\mathbf{x},\mathbf{v})$ and $\sigma_f=\sigma_f(\mathbf{x},\mathbf{v})$ are given functions of $(\mathbf{x},\mathbf{v})$. They stand for the total, absorption, scattering and fission cross-sections. Coefficient $\nu_f=\nu_f(\mathbf{x},\mathbf{v})$ is the multiplicity of the fission reaction. The quantities P_s, P_f define how the velocities and angles are

Email addresses: gael.poette@cea.fr (Gaël Poëtte), emeric.brun@cea.fr (Emeric Brun)

scattered depending on the type of reaction encountered: they (at least) satisfy $\int P_\alpha(\mathbf{x}, \mathbf{v}\cdot\mathbf{v}')\,\mathrm{d}\mathbf{v}' = 1, \forall \alpha \in \{s, f\}, \mathbf{x} \in \mathcal{D}, \mathbf{v} \in \mathcal{V} \in \mathbb{R}^3$. Of course, the above notations are for macroscopic cross-sections, in the sense that many physical reactions are summed-up in the above notations, see [10, 11, 12]. Boundary conditions must be supplemented to system (1):

$$u(\mathbf{x}, \mathbf{v}) = u_b(\mathbf{v}), \quad \mathbf{x} \in \partial\mathcal{D}, \quad \omega \cdot n_s < 0, \tag{2}$$

where n_s is the outward normal to Ω at $\mathbf{x}$. System (1) together with boundary conditions (2) define the well-posed [9, 8] mathematical problem we want to solve. To be more precise, in this paper, we are even interested in *being able to accurately take uncertainties, in a broad sense*[1]*, into account.*

When dealing with an uncertainty quantification problem, it is common to explicit, in a general manner, the dependence of the solution with respect to the uncertain vector of parameters denoted here by $\mathbf{X}$. Note that without loss of generality in the following sections, we consider that $\mathbf{X}$ is a vector $\mathbf{X} = (X_1, ..., X_Q)^t$ of Q independent random variables of probability measure $\mathrm{d}\mathcal{P}_{\mathbf{X}} = \prod_{i=1}^{Q} \mathrm{d}\mathcal{P}_{X_i}$. It is always possible to come back to such framework[2].

As a result, solving the uncertain counterpart of (1) consequently resumes to solving the Stochastic PDE (SPDE) given by

$$\begin{aligned}\mathbf{v}\cdot\nabla_{\mathbf{x}} u(\mathbf{x}, \mathbf{v}, \mathbf{X}) + v\sigma_t(\mathbf{x}, \mathbf{v}, \mathbf{X})u(\mathbf{x}, \mathbf{v}, \mathbf{X}) = v\sigma_s(\mathbf{x}, \mathbf{v}, \mathbf{X}) \int P(\mathbf{x}, \mathbf{v}\cdot\mathbf{v}', \mathbf{X})u(\mathbf{x}, \mathbf{v}', \mathbf{X})\,\mathrm{d}\mathbf{v}', \\ + \frac{v\nu_f(\mathbf{x}, \mathbf{v}, \mathbf{X})\sigma_f(\mathbf{x}, \mathbf{v}, \mathbf{X})}{k_{\mathrm{eff}}(\mathbf{X})} \int P_f(\mathbf{x}, \mathbf{v}\cdot\mathbf{v}', \mathbf{X})u(\mathbf{x}, \mathbf{v}', \mathbf{X})\,\mathrm{d}\mathbf{v}',\end{aligned} \tag{3}$$

together with

$$u(\mathbf{x}, \mathbf{v}, \mathbf{X}) = u_b(\mathbf{v}, \mathbf{X}), \quad \mathbf{x} \in \partial\mathcal{D}(\mathbf{X}), \quad \omega \cdot n_s(\mathbf{X}) < 0. \tag{4}$$

In the above problem, we are mainly interested in the statistics of $\mathbf{X} \to k_{\mathrm{eff}}(\mathbf{X})$ and $\mathbf{X} \to u(\mathbf{x}, \mathbf{v}, \mathbf{X})$ (i.e. mean, variance, histogram, sensitivity indices [18] etc.) at specified locations $\mathbf{x} \in \mathcal{D}$ and velocities $\mathbf{v} \in \mathcal{V}$.

Of course, different values of $\mathbf{X}$ correspond to different fully decoupled deterministic equations: in principle, there is no difficulty in solving such uncertain problems. The main issue comes from the fact that exact propagation of uncertainties is very expensive from the computational point of view: equation (1) is often solved thanks to a Monte Carlo scheme [19, 20, 8, 21, 22, 23]. This resolution method is known to be efficient for high $(3(\mathbf{x})+1(t)+3(\mathbf{v}) = 7)$ dimensional problems but costly. Running several deterministic MC computations for several values of $\mathbf{X}$ can consequently be prohibitive.

In [1, 2], a P-truncated generalised Polynomial Chaos (gPC) based reduced model of the *instationary uncertain linear Boltzmann equation* has been introduced. It is solved thanks to an astute Monte Carlo (MC) scheme [1]: the idea is to make the MC particles solve not only the physical fields $(\mathbf{x}, \mathbf{v})$ but also the uncertain one $\mathbf{X}$, *on-the-fly* during the MC resolution. Similar approaches have been developed for the Fokker-Planck equation [6] and for the quadratic Boltzmann equation [4, 24] and give promising results on other (usually MC solved) physical models. The spectral (i.e. fast) convergence of the built hierachical models has been numerically [1] and theoretically

[1] geometrical, in the cross-sections, in the multiplicity, in the boundary conditions etc.

[2] At the cost of more or less tedious pretreatments leading to a controled approximation [13, 14, 15] and decorrelation [16, 17].

[2] demonstrated. To sum-up, important gains have been observed in low to moderate stochastic dimensions[3] $Q \sim 1-10$ with *simple* code modifications of an existing MC code and *without changing* the HPC strategy[4] of the code. see [1, 2]. Now, the question is: is it possible to generalise the methodology used in these papers to be able to compute uncertain eigenvalues/eigenvectors? This paper presents our efforts in order to answer the latter question. We here design an uncertain eigenvalue solver. It relies on the material of [1] and some additional key ingredients described in this paper (mainly, a stochastic power iteration method). The approach in [1], denoted by MC-gPC in the following, enters the class of intrusive[5] generalised Polynomial Chaos (gPC) methods. It consequently demands some code modifications: in the following sections, care is taken to highlight the modifications needed with respect to an already existing MC eigenvalue/eigenvector solver to be able to reproduce the results of this paper.

The paper is organized as follows: in section 2, we recall one specific deterministic k_{eff} MC algorithm allowing to solve (1) (power iteration). Several algorithms exist, see [25], but we can not go through every of them. Following the descriptions of the next sections should allow the interested reader to perform the relevant modifications to its own eigenvalue solver. In section 3, the gPC based reduced model of (3) is built. Its MC resolution is tackled in section 4. Care is taken to put forward where, with respect to the algorithm described in section 2, modifications of a deterministic MC eigenvalue solver must be made to take into account uncertainties *on-the-fly* during the MC resolution. Section 5 is devoted to benchmarks and numerical test-cases, section 6, to concluding remarks.

2. The Monte Carlo resolution of a deterministic eigenvalue problem

In this section, we recall the main steps of the resolution of the deterministic eigenvalue problem (1). The aim here is to briefly recall them in order to ease the comparisons with the new stochastic solver described in section 4 and especially to identify clearly where a deterministic MC solver could be modified in order to take into account, *on-the-fly* during the MC resolution, the uncertainties.

The deterministic criticity problem can be stated as follows: find $k_{\text{eff}} \in \mathbb{R}^+, u \neq 0$ such that

$$\begin{cases} \mathbf{v}\cdot\nabla_{\mathbf{x}} u(\mathbf{x},\mathbf{v}) + v\sigma_t(\mathbf{x},\mathbf{v})u(\mathbf{x},\mathbf{v}) = v\sigma_s(\mathbf{x},\mathbf{v})\displaystyle\int P_s(\mathbf{x},\mathbf{v}\cdot\mathbf{v}')u(\mathbf{x},\mathbf{v}')\,\mathrm{d}\mathbf{v}', \\ \qquad\qquad + \dfrac{v\nu_f(\mathbf{x},\mathbf{v})\sigma_f(\mathbf{x},\mathbf{v})}{k_{\text{eff}}}\displaystyle\int P_f(\mathbf{x},\mathbf{v}\cdot\mathbf{v}')u(\mathbf{x},\mathbf{v}')\,\mathrm{d}\mathbf{v}', \\ u(\mathbf{x},\mathbf{v}) = u_b(\mathbf{v}), \quad \mathbf{x}\in\partial\mathcal{D}, \quad \frac{\mathbf{v}}{v}\cdot n_s < 0, \text{ with } |\mathbf{v}| = v. \end{cases} \tag{5}$$

In the above equation, we assume that $\mathcal{D}$ is smooth and bounded and that we have $\sigma_a(\mathbf{x},\mathbf{v}) = \sigma_t(\mathbf{x},\mathbf{v}) - \sigma_s(\mathbf{x},\mathbf{v}) \geq 0$, with also $\sigma_s(\mathbf{x},\mathbf{v}) \geq 0, \sigma_f(\mathbf{x},\mathbf{v}) > 0$ holding $\forall \mathbf{x}\in\mathcal{D}, \mathbf{v}\in\mathcal{V}$. Then Krein-Rutman's theorem [9, 26, 27] ensures that (k_{eff}, u) satisfying (5) exists. Besides, k_{eff} is the

[3] MC-gPC being based on gPC which is sensitive to the curse of dimension, the $P-$truncated reduced models remains exponentially sensitive to P and Q, see [1, 2].

[4] The HPC strategy we have in mind is commonly called *replication domain*, see [23]. It consists in replicating the geometry on several processors and tracking several MC particles populations with different initial seeds in every replicated domains. At the end of the time steps, the contribution of every processors are averaged. This parallel strategy is particularly well suited to MC codes, taking advantage of the independence of the MC particles.

[5] It demands some code modifications of the MC solver, see [1], and does not propagate the uncertainties by relying on several runs of a black-box code.

highest eigenvalue, it is simple and its associated eigenvector u is the only positive eigenvector for $\mathbf{x} \in \mathcal{D}, \mathbf{v} \in \mathcal{V}$. As a consequence, problem (5) is well-posed and we can try to solve it numerically.

In order to solve the previously described eigenvalue/eigenvector problem, the power iteration method [9, 25] is usually applied. Its convergence is ensured in the previous aformentionned conditions. Several 'versions' of the power iteration method exist for k_{eff} computations, see [25]. We suggest recalling one of these versions (which can be found in [25]) in the following lines. This version has been chosen because it strongly relates a *power iteration* to a *time step*: this helps reusing the results of [1] for the unstationary linear Boltzmann equation, see section 4.

Data: Δt, N_{MC}
Result: The eigenvalue k_{eff} and eigenvector which can be built from list_of_particles
begin
  #initialisation of a population of particles, see algo. 2
  list_of_particles=sampleParticles(N_{MC})
  #old number of physical particles on the whole geometry
  set $U_{\text{old}} = 1$
  #new number of physical particles on the whole geometry
  set $U_{\text{new}} = 1$
  #estimated eigenvalue
  set $k_{\text{eff}} = 1$
  while *iter < iter_max* **do**
    #tracking of the population of particles, see algo. 8
    U_{new}=trackParticles(list_of_particles, $\Delta t, k_{\text{eff}}$)
    #update eigenvalue
    $k_{\text{eff}} \leftarrow k_{\text{eff}} \times \frac{U_{\text{new}}}{U_{\text{old}}}$
    #update the old number of physical particles
    $U_{\text{old}} \leftarrow U_{\text{new}}$
    #apply a population control algorithm, see algo. 3
    populationControl(list_of_particles, N_{MC})
    iter++
  end
end

Algorithm 1: General canvas of a deterministic k_{eff} calculation Monte Carlo code

Algorithm 1 is a coarse grain description of the power iteration method. It first needs the initialisation of a population of MC particles. An MC particle p is defined as a particular solution of (5) of the form

$$u_p(\mathbf{x}, \mathbf{v}, t) = w_p(t)\delta_{\mathbf{x}}(\mathbf{x}_p(t))\delta_{\mathbf{v}}(\mathbf{v}_p(t)). \tag{6}$$

For u_p to be solution of (5), its fields $w_p(t), \mathbf{x}_p(t), \mathbf{v}_p(t)$ must respect some compatibility conditions [1, 28, 19] as time evolves (function trackParticles, described later on, makes sure those conditions are fulfilled). Once those conditions satisfied, each u_p solves (5) and, formally, by linearity,

$$\sum_{p=1}^{N_{MC}} u_p(\mathbf{x}, t, \mathbf{v}) \overset{\substack{N_{MC}\to\infty \\ t\to\infty}}{\approx} u(\mathbf{x}, \mathbf{v}),$$

solves (5). The sampling phase, *via* sampleParticles in algorithm 1 and detailed in algorithm 2 builds a population of particles and sets their initial fields $(w_p(0), \mathbf{x}_p(0), s_p(0), \mathbf{v}_p(0))_{p\in\{1,...,N_{MC}\}}$.

sampleParticles(N_{MC})
Data: N_{MC}
Result: list_of_particles (a normalised population of N_{MC} MC particles)
begin
  list_of_particles=[ ]
  for $p \in \{1, ..., N_{MC}\}$ **do**
    #build particle p
    set $s_p = \Delta t$ *#remaining life time of particle p (must go down to zero)*
    set $\mathbf{x}_p = \mathcal{U}(\mathcal{D})$ *#spatially uniformly distributed MC particles in* $\mathcal{D}$
    set $\mathbf{v}_p = \mathcal{L}(\mathcal{V})$ *#velocities sampled from a chosen spectrum* $\mathcal{V}$
    set $w_p = \frac{1}{N_{MC}}$ *#normalised population:* $\sum_p w_p = 1$
    list_of_particles.append($\{w_p, \mathbf{x}_p, \mathbf{v}_p, s_p\}$)
  end
end

Algorithm 2: description of sampleParticles: to initialise a population of MC particles for a deterministic k_{eff} computation

At the end of the sampling phase, a list of particles is built. Some quantities are initialised, cf. the lines before the while loop in algorithm 1: they will allow updating the eigenvalue through the iteration. Note that choosing $k_{\text{eff}} = 1$ in algorithm 1 together with uniformly distributed MC particles in algorithm 2 is arbitrary: those represent guess eigenvalue and guess eigenvector of the iterative process and some other choices may be better suited (but this is case dependent).

Then, in algorithm 1 comes the *while* loop: the maximum number of iteration is supposed to be a parameter. Of course, more elaborate MC codes often have more efficient and relevant stopping criterion but going through them is beyond the scope of this paper. The first call in the *while* loop is trackParticles: this function typically makes sure every u_p are MC solutions of (5). Function trackParticles is described in algorithm 8, in the appendix[6]. At the end of trackParticles, the total number of physical particles is updated (in U_{new}) and allows updating the k_{eff} value. A population control algorithm is often called: it allows having about the same number of MC particles through the iterations (as some may be lost[7] going ouside the domain $\mathcal{D}$).

[6]It is described only in the appendix because it does not bear any novelty with respect to the content of [1].

[7]In this paper, we rely on a semi-analog MC scheme, see [28] also called *implicit capture* [11, 8], so that MC particles are not created during the tracking. For supercritical situations, the weights of the MC particles grow.

populationControl(list_of_particles, N_{MC})
Data: list_of_particles, N_{MC}
Result: list_of_particles (with N_{MC} MC particles)
begin
  $N_{MC}^{\text{new}} = 0$
  for $p \in$list_of_particles **do**
    set $s_p = \Delta t$ *#reset the life time of particle p (must go down to zero)*
    $N_{MC}^{\text{new}} + +$
  end
  while $N_{MC}^{new} < N_{MC}$ **do**
    #Choose randomly a particle p in list_of_particles
    p = sampleMCParticleFromList(list_of_particle)
    #splitMCParticle*'s output is a number of particles*
    #splitMCParticle*'s output may depend on* $w_p, \mathbf{x}_p, \mathbf{v}_p$
    #in practice, we choose to split a particle in two if sampled
    split_p = splitMCParticle($\{w_p, \mathbf{x}_p, \mathbf{v}_p\}$)
    $w_p \leftarrow w_p \times \frac{1}{\text{split\_p}}$
    for $p' \in \{1, ..., \text{split\_p} - 1\}$ **do**
      set $s_{p'} = s_p$
      set $\mathbf{x}_{p'} = \mathbf{x}_p$
      set $\mathbf{v}_{p'} = \mathbf{v}_p$
      set $w_{p'} = w_p$
      list_of_particles.append($\{w_{p'}, \mathbf{x}_{p'}, \mathbf{v}_{p'}, s_{p'}\}$)
    end
    $N_{MC}^{\text{new}} +=$ split_p
  end
end
Algorithm 3: description of populationControl: to make sure we always have about N_{MC} particles within domain $\mathcal{D}$

One population control algorithm (many exists, we can not go through every of them in this paper) is presented in algorithm 3. Suppose that after the tracking step, the number of MC particles is below N_{MC}: the question is *how can we add particles without introducing any bias/error*? Algorithm 3 is one way to answer that question: assume that at the end of the power iteration, the number of remaining MC particles is $N_{MC}^{\text{new}} < N_{MC}$. We want to add, in an unbiased manner, $N_{MC} - N_{MC}^{\text{new}}$ particles to our list_of_particles. First, a particle p is randomly chosen in list_of_particles. Based on some criterion on $(w_p, \mathbf{x}_p, \mathbf{v}_p)$, the algorithm chooses to split an MC particle or not into split_p ones. By splitting, we mean that from one MC particle p, we can choose to build split_p> 0 MC particles $p' \in \{1, ..., \text{split\_p} - 1\} \bigcup \{p\}$ having the same fields as p except for their weights $w_{p'} = \frac{w_p}{\text{split\_p}}$, $\forall p' \in \{1, ..., \text{split\_p} - 1\} \bigcup \{p\}$, so that $\sum_{p'} w_{p'} = w_p$. Note that in this paper, we focus on unbiased population control technics (not only unbiased on the first moment of u, the reason will be clarified later on). In practice, in this paper, split_p is chosen to be 2, i.e. particle p can only be split in 2, independently of its fields $w_p, \mathbf{x}_p, \mathbf{v}_p$. Better splitting strategies may be at hand but they are beyond the scope of this paper.

With this brief description, we highlighted the main steps of an eigenvalue/eigenvector resolution for the transport equation. In the following section 3, we build gPC based hierachical reduced models. They allow efficiently taking uncertainties into account (i.e. accurate results can be recovered with low truncation orders, see [2]). In section 4, we explain how we can even solve those reduced models thanks to an MC solver. In particular, we explain how the (deterministic) material of this section 2 can be modified to solve, *on-the-fly* during the MC resolution, the gPC based reduced model of section 3.

3. The gPC based hierachical reduced models

We now would like to take uncertainties into account for the problem described in the previous section 2. Let us introduce $\mathbf{X} \sim \mathrm{d}\mathcal{P}_{\mathbf{X}}$, a vector of independent random variables of probability measure $\mathrm{d}\mathcal{P}_{\mathbf{X}}$ modeling the uncertainties. Then, by expliciting the dependence of u, k_{eff} with respect to $\mathbf{X}$, the uncertain version of (5) is given by

$$\begin{cases} \mathbf{v}\cdot\nabla_{\mathbf{x}}u(\mathbf{x},\mathbf{v},\mathbf{X}) + v\sigma_t(\mathbf{x},\mathbf{v},\mathbf{X})u(\mathbf{x},\mathbf{v},\mathbf{X}) = v\sigma_s(\mathbf{x},\mathbf{v},\mathbf{X})\displaystyle\int P_s(\mathbf{x},\mathbf{v}\cdot\mathbf{v}',\mathbf{X})u(\mathbf{x},\mathbf{v}',\mathbf{X})\,\mathrm{d}\mathbf{v}', \\ \qquad\qquad + \dfrac{v\nu_f(\mathbf{x},\mathbf{v},\mathbf{X})\sigma_f(\mathbf{x},\mathbf{v},\mathbf{X})}{k_{\text{eff}}(\mathbf{X})}\displaystyle\int P_f(\mathbf{x},\mathbf{v}\cdot\mathbf{v}',\mathbf{X})u(\mathbf{x},\mathbf{v}',\mathbf{X})\,\mathrm{d}\mathbf{v}', \\ u(\mathbf{x},\mathbf{v},\mathbf{X}) = u_b(\mathbf{v},\mathbf{X}), \quad \mathbf{x}\in\partial\mathcal{D}(\mathbf{X}), \quad \frac{\mathbf{v}}{v}\cdot n_s(\mathbf{X}) < 0, \text{ with } |\mathbf{v}| = v \text{ and } \mathbf{X}\sim \mathrm{d}\mathcal{P}_{\mathbf{X}}. \end{cases} \tag{7}$$

Of course, we assume that the probable values of $\mathbf{X}$ never question the existence and unicity conditions for $(k_{\text{eff}}(\mathbf{X}), u(\mathbf{x},\mathbf{v},\mathbf{X}))$, put forward in section 2 (in other words, $\forall\mathbf{X}$, (7) is well-posed). In principle, there is no difficulty solving the above problem as several values of $\mathbf{X}$ corresponds to several fully decoupled problems. The whole problem comes from the fact that exact propagation of uncertainties is very expensive from the computational point of view: MC codes, to solve (7) (i.e. several times (5) for several $\mathbf{X}$), are efficient but costly. In this section, in order to avoid the multiplication of runs of a costly code, we suggest building and solving a reduced model of (7) allowing to accurately capture the uncertainties.

In this paper, we are interested in the construction of gPC based reduced models in order to take into account uncertainties. Let us introduce the polynomials basis $(\phi_k^{\mathbf{X}})_{k\in\mathbb{N}}$ orthonormal with respect to the scalar product defined by $\mathrm{d}\mathcal{P}_{\mathbf{X}}$, i.e such that

$$\int \phi_k^{\mathbf{X}}(\mathbf{X})\phi_l^{\mathbf{X}}(\mathbf{X})\,\mathrm{d}\mathcal{P}_{\mathbf{X}} = \delta_{k,l}, \forall(k,l)\in\mathbb{N}^2.$$

In practice, this basis is built once and for all once $\mathrm{d}\mathcal{P}_{\mathbf{X}}$ known. In the above expression, the basis must be truncated up to certain orders $(p_i)_{i\in\{1,\dots,Q\}}$ which may depend on the directions $(X_i)_{i\in\{1,\dots,Q\}}$. Assume that $\forall i \in \{1,..,Q\}$, $p_i = p_{1\text{D}}$, then the total number of polynomial coefficients, abusively called the polynomial order later on, is[8] $P = P(p_{1\text{D}}, Q) = (p_{1\text{D}}+1)^Q$. It exhibits an exponential growth with both $p_{1\text{D}}$ and Q. This is commonly called *the curse of dimensionality* [30, 31]. As a consequence, the reduced models described in this paper, in practice, can only be applied to a moderate number of uncertain parameters ($Q \sim 10$). The multivariate polynomial basis is built by tensorization of one-dimensional polynomial basis in every stochastic

[8] Of course, simplexes such as the ones presented in [29] may be used and have less coefficients but studying their effects is beyond the scope of this paper.

direction $(X_i)_{i\in\{1,...,Q\}}$. In the following sections, for conciseness in the notations, we map[9] the set of polynomial indices $\mathcal{A}^{p_{1D},Q} = \{(k_1,...k_Q)|\forall i \in \{1,...,Q\}, k_i \leq p_{1D}\}$ into $\{0,...,P\}$ to build the tensorized basis $(\phi_k^{\mathbf{X}}(\mathbf{X}) = \prod_{i=1}^{Q} \phi_{k_i}^{X_i}(X_i))_{k\in\{0,...,P\}}$. In the previous expression, $\forall i \in \{1,...,Q\}$, the basis $(\phi_k^{X_i})_{k\in\{0,...,p_{1D}\}}$ is a one-dimensional polynomial basis orthonormal with respect to $\mathrm{d}\mathcal{P}_{X_i}$. When P grows, we assume it grows because the one-dimensional polynomial orders p_{1D} grow.

Let us assume we want to approximate a function $\mathbf{X} \to F(\mathbf{X})$ such that $\int F^2(\mathbf{X})\,\mathrm{d}\mathcal{P}_{\mathbf{X}} < \infty$. Then the P-truncated gPC expansion defined by the polynomial approximation

$$F^P(\mathbf{X}) = \sum_{k=0}^{P} F_k \phi_k^{\mathbf{X}}(\mathbf{X}) \xrightarrow[P\to\infty]{L^2} F(\mathbf{X}), \tag{8}$$

bears some interesting convergence properties [32, 33, 34]. Spectral convergence for F solution of the unstationary linear Boltzmann equation has even been proved in [2].

Independently of the dimension or of the polynomial basis, the gPC coefficients $(F_k)_{k\in\{0,..,P\}}$ are defined by integration: they correspond to the projection of F on the components of the gPC basis with respect to the scalar product defined by $\mathrm{d}\mathcal{P}_{\mathbf{X}}$:

$$F_k = \int F(\mathbf{X})\phi_k^{\mathbf{X}}(\mathbf{X})\,\mathrm{d}\mathcal{P}_{\mathbf{X}}, \forall k \in \mathbb{N}. \tag{9}$$

We naturally want to apply the above material to $\mathbf{X} \to u(\mathbf{x},\mathbf{v},\mathbf{X})\ \forall \mathbf{x} \in \mathcal{D}, \mathbf{v} \in \mathcal{V}$ and $\mathbf{X} \to k_{\text{eff}}(\mathbf{X})$ instead of F. As a consequence, our aim is to compute the gPC coefficients of the couple (k_{eff}, u) by numerical integration. We would like to integrate those coefficients during the MC resolution of the transport equation. In the next section, we explain how we can estimate them *on-the-fly* during the MC resolution thanks to simple modifications of an already existing MC solver.

4. The Monte Carlo resolution of an uncertain eigenvalue problem

In this section, we describe how a classical eigenvalue/eigenvector MC solver can be easily modified in order to take uncertainties into account, relying on a gPC based reduced model. As explained in the previous section 3, the main idea consists in being able to compute, during the MC resolution, the gPC coefficients of the eigenvalue/eigenvector couple (k_{eff}, u).

First, the deterministic MC eigenvalue/eigenvector algorithm 1 must be compared to algorithm 4: the algorithm needs one additional numerical input parameter P with respect to algorithm 1. Second, the population of MC particles is replaced by a population of *uncertain* MC particles, exactly as described in [1]. In a nutshell, it implies sampling the uncertain dimensions $\mathbf{X}$ together with the physical variables $(\mathbf{x},\mathbf{v})$, see algorithm 5. This point is important in practice for efficiency because it avoids the tensorisation of the physical variables with the uncertain ones: any non-intrusive method implying the independent N runs of an MC code needs to track $N \times N_{MC}$ particles whereas the solver described in algorithm 4 only tracks a population of N_{MC} particles having fields in the whole space $(\mathbf{x},\mathbf{v},\mathbf{X}) \in \mathcal{D} \times \mathcal{V} \times \mathrm{d}\mathcal{P}_{\mathbf{X}}$. In algorithm 5, each uncertain MC particle has an additional field $\mathbf{X}_p = (X_p^1,...,X_p^Q)^t$ sampled according to $\mathrm{d}\mathcal{P}_{\mathbf{X}}$. In other words, we are looking for particular solutions

$$u_p(\mathbf{x},t,\mathbf{v},\mathbf{X}) = w_p(t)\delta_{\mathbf{x}}(\mathbf{x}_p(t))\delta_{\mathbf{v}}(\mathbf{v}_p(t))\delta_{\mathbf{X}}(\mathbf{X}_p(t)).$$

[9] It is only a renumerotation.

Now, the consistent operations for the above *uncertain MC particle* u_p to be solution of the uncertain transport equations have been put forward in [1] and are recalled briefly[10] in Appendix A.

The next step in algorithm 4 consists in initialisations of $(U^k_{\text{new}}, U^k_{\text{old}}, k^k_{\text{eff}})_{k\in\{0,\dots,P\}}$. In the deterministic algorithm 1, those quantities were scalar. In the stochastic counterpart, those are vectors of size $P+1$.

Then comes the *while* loop and a stopping criterion for the iterative algorithm (which is here relatively simple with iter_max). The first function of this loop is trackUncertainParticles, described in Appendix A.2. Its main output is a vector of updated gPC coefficients of the total number of physical particles of the geometry $(U^k_{\text{new}})_{k\in\{0,\dots,P\}}$ at current time/iteration n, defined by

$$\begin{aligned} U^k_{\text{new}}(t^n) &= \int_{\mathcal{D}} \int_{N_{MC}} \int u(\mathbf{x},\mathbf{v},\mathbf{X})\phi^{\mathbf{X}}_k(\mathbf{X})\,\mathrm{d}\mathcal{P}_{\mathbf{X}}\,\mathrm{d}\mathbf{x}\,\mathrm{d}\mathbf{v}, \forall k\in\{0,...,P\}, \\ &\approx \sum_{p=1} w_p(t^n)\phi^{\mathbf{X}}_k(\mathbf{X}_p), \forall k\in\{0,...,P\}. \end{aligned}$$

Once this vector updated, the gPC coefficients of $k_{\text{eff}}(\mathbf{X})$ must be updated. This part of the algorithm is typical of eigenvalue/eigenvector computation and is original with respect to the material of [1]. The equivalent of operation $k_{\text{eff}} \leftarrow k_{\text{eff}}\frac{U_{\text{new}}}{U_{\text{old}}}$ for the deterministic algorithm 1 must be performed: this corresponds to the blue/cyan lines of algorithm 4. A change of color has been used with the cyan line because this operation needs a discretisation hypothesis. The cyan line in algorithm 4 is a particular discretisation of operation

$$k^{k,\text{new}}_{\text{eff}} = \int k^P_{\text{eff}}(\mathbf{X}) \times \frac{U^P_{\text{new}}(\mathbf{X})}{U^P_{\text{old}}(\mathbf{X})}\phi^{\mathbf{X}}_k(\mathbf{X})\,\mathrm{d}\mathcal{P}_{\mathbf{X}}, \forall k\in\{0,..,P\}. \tag{10}$$

The above update (10) of the gPC coefficients of k_{eff} can not, in general, be analytical. It must be discretised. Several algorithms are available in the literature, see [35, 36] for example, to take into account nonlinearities in gPC computations. In the numerical section 5 of this paper, we rely on numerical integration of (10) thanks to Gauss quadrature rules [37]. Basically, they allow approximating $(\mathbf{X}, \mathrm{d}\mathcal{P}_{\mathbf{X}})$ by N_G points/weights $(\mathbf{X}_g, w_g)_{g\in\{1,\dots N_G\}}$ and are very efficient in low to moderate stochastic dimensions (from $Q=1$ to $Q=10$, see [37, 38, 39, 40]). Hence, (10) is in practice replaced by

$$\begin{aligned} k^{k,\text{new}}_{\text{eff}} &= \int k^P_{\text{eff}}(\mathbf{X}) \times \frac{U^P_{\text{new}}(\mathbf{X})}{U^P_{\text{old}}(\mathbf{X})}\phi^{\mathbf{X}}_k(\mathbf{X})\,\mathrm{d}\mathcal{P}_{\mathbf{X}}, \\ &\approx \sum_{g=1}^{N_G} k^P_{\text{eff}}(\mathbf{X}_g) \times \frac{U^P_{\text{new}}(\mathbf{X}_g)}{U^P_{\text{old}}(\mathbf{X}_g)}\phi^{\mathbf{X}}_k(\mathbf{X}_g)w_g, \end{aligned} \tag{11}$$

where $k^P_{\text{eff}}(\mathbf{X})$, $U^P_{\text{old}}(\mathbf{X})$ and $U^P_{\text{new}}(\mathbf{X})$ are (nonlinear[11] functions of) polynomials of order P: those are built in function buildPunctualValues, see the blue lines in algorithm 4 and algorithm 6.

[10] In particular, we must have $\mathbf{X}_p(t) = \mathbf{X}_p$, this fields must not change during the tracking.

[11] See the description of buildPunctualValues below.

Data: Δt, N_{MC}, P
Result: The gPC coefficients $(k_{\text{eff}}^k)_{k\in\{0,...,P\}}$ of k_{eff} and eigenvector which can be built from list_of_particles
begin
#initialisation of a population of particles, see algo. 5
list_of_particles=sampleUncertainParticles(N_{MC})
#old gPC coefficients of the number of physical particles on the whole geometry
set $U_{\text{old}}^0 = 1$
#new gPC coefficients of the number of physical particles on the whole geometry
set $U_{\text{new}}^0 = 1$
#gPC coefficients of the estimated eigenvalue
set $k_{\text{eff}}^0 = 1$
for $k \in \{1, ..., P\}$ **do**
$U_{\text{old}}^k = 0$
$U_{\text{new}}^k = 0$
$k_{\text{eff}}^k = 1$
end
while *iter* $<$ *iter_max* **do**
#tracking of the population of uncertain particles, see algo. 10
$(U_{\text{new}}^k)_{k\in\{0,..,P\}}$=trackUncertainParticles(list_of_particles, $\Delta t, k_{\text{eff}}^0, ..., k_{\text{eff}}^P$)
#build punctual uncertain values
$(U_{\text{new}}^P(\mathbf{X}_g))_{g\in\{1,..,N_G\}} = buildPunctualValues((\mathbf{X}_g)_{g\in\{1,..,N_G\}}, (U_{\text{new}}^k)_{k\in\{0,...P\}})$
$(U_{\text{old}}^P(\mathbf{X}_g))_{g\in\{1,..,N_G\}} = buildPunctualValues((\mathbf{X}_g)_{g\in\{1,..,N_G\}}, (U_{\text{old}}^k)_{k\in\{0,...P\}})$
$(k_{\text{eff}}^P(\mathbf{X}_g))_{g\in\{1,..,N_G\}} = buildPunctualValues((\mathbf{X}_g)_{g\in\{1,..,N_G\}}, (k_{\text{eff}}^k)_{k\in\{0,...P\}})$
#update the gPC coefficients of the eigenvalue
for $k \in \{1, ..., P\}$ **do**
$$k_{\text{eff}}^k \leftarrow \sum_{g=1}^{N_G} k_{\text{eff}}^P(\mathbf{X}_g) \times \frac{U_{\text{new}}^P(\mathbf{X}_g)}{U_{\text{old}}^P(\mathbf{X}_g)} \phi_k^{\mathbf{X}}(\mathbf{X}_g) w_g$$
end
#update the old number of physical particles
for $k \in \{1, ..., P\}$ **do**
$U_{\text{old}}^k \leftarrow U_{\text{new}}^k$
end
#apply a population control algorithm, see algo. 7
uncertainPopulationControl(list_of_particles, N_{MC})
iter++
end
end

Algorithm 4: General canvas of a stochastic/uncertain k_{eff} calculation Monte Carlo code

Function buildPunctualValues builds punctual approximations of $U_{\text{new}}^P(\mathbf{X}), U_{\text{old}}^P(\mathbf{X}), k_{\text{eff}}^P(\mathbf{X})$ at the quadrature points $(\mathbf{X}_g)_{g\in\{1,..,N_G\}}$ based on their respective gPC coefficients $(U_{\text{new}}^k)_{k\in\{0,...,P\}}$, $(U_{\text{old}}^k)_{k\in\{0,...,P\}}$, $(k_{\text{eff}}^k)_{k\in\{0,...,P\}}$. In its simplest form, buildPunctualValues resumes to building the

following polynomial approximations

$$
\begin{aligned}
k_{\text{eff}}^{P}(\mathbf{X}) &= \sum_{k=0}^{P} k_{\text{eff}}^{k} \phi_k^{\mathbf{X}}(\mathbf{X}), \\
U_{\text{old}}^{P}(\mathbf{X}) &= \sum_{k=0}^{P} U_{\text{old}}^{k} \phi_k^{\mathbf{X}}(\mathbf{X}), \\
U_{\text{new}}^{P}(\mathbf{X}) &= \sum_{k=0}^{P} U_{\text{new}}^{k} \phi_k^{\mathbf{X}}(\mathbf{X}),
\end{aligned}
\tag{12}
$$

at the quadrature points $(\mathbf{X}_g)_{g\in\{1,...,N_G\}}$. Of course, function buildPunctualValues may encapsulate more elaborated gPC based approximations. Having resort to more elaborated approximations may be motivated by a will to preserve positivity, to respect a maximum principle or avoid oscillating reconstructions, see [41, 42, 43, 44, 45, 46, 47, 48, 49, 50]. After all, for the polynomial approximations (12), we have no guaranty of positivity[12]. Ensuring the positiveness of those quantities can be important in practice as negative values may trigger numerical instabilities of the stochastic power iteration algorithm[13]. To anticipate on this potential problem, we embed in buildPunctualValues the possibility to use, for example, a *positivity preserving procedure* based on the material of the furnished literature [41, 42, 28, 43, 44, 45, 47, 48, 46, 49]. Of course, we will not use it if not necessary. But care will be taken to monitor and analyse any loss of positiveness in the numerical examples of section 5.

sampleParticles(N_{MC})
Data: N_{MC}
Result: list_of_particles (a normalised population of N_{MC} uncertain MC particles)
begin
  list_of_particles=[ ]
  for $p \in \{1, ..., N_{MC}\}$ **do**
    #build particle p
    set $s_p = \Delta t$ *#remaining life time of particle p (must go down to zero)*
    set $\mathbf{x}_p \sim \mathcal{U}(\mathcal{D})$ *#spatially uniformly distributed MC particles in* $\mathcal{D}$
    set $\mathbf{v}_p \sim \mathcal{L}(\mathcal{V})$ *#velocities sampled from a chosen spectrum* $\mathcal{V}$
    set $w_p = \frac{1}{N_{MC}}$ *#normalised population:* $\sum_p w_p = 1$
    set $\mathbf{X}_p \sim d\mathcal{P}_{\mathbf{X}}$ *#uncertain MC particles take into account the uncertainty*
    list_of_particles.append($\{w_p, \mathbf{x}_p, \mathbf{v}_p, s_p, \mathbf{X}_p\}$)
  end
end

Algorithm 5: description of sampleUncertainParticles: to initialise a population of uncertain MC particles for a stochastic/uncertain k_{eff} computation

[12] The polynomial approximation does lose positiveness in certain situations, see [2].

[13] Remember for example that $k_{\text{eff}}^{P}(\mathbf{X}_p)$ is used within trackUncertainParticles for MC particle p. If negative, MC particle p sees a negative fission cross-section which is non physical and may be numerically problematic.

buildPunctualValues($(\mathbf{X}_g)_{g\in\{1,..,N_G\}}$,$(U^k)_{k\in\{0,..,P\}}$)
Data: $(\mathbf{X}_g)_{g\in\{1,..,N_G\}}$,$(U^k)_{k\in\{0,..,P\}}$
Result: $(U^P(\mathbf{X}_g))_{g\in\{1,..,N_G\}}$
begin
  #Polynomial reconstruction (i.e. $s(u)=\frac{u^2}{2}$)
  positive_reconstrunction_needed=false
  for $g\in\{1,...,N_G\}$ **do**
    $U^P(\mathbf{X}_g)=\sum_{k=0}^{P} U^k\phi_k^{\mathbf{X}}(\mathbf{X}_g)$
    if $U^P(\mathbf{X}_g)<0$ **then**
      positive_reconstrunction_needed=true
      break
    end
    if positive_reconstrunction_needed **then**
      #Apply your favorite positivity preserving reconstruction
      $U^P(\mathbf{X}_g)$=compute_positive_preserving_reconstruction($(U^k)_{k\in\{0,...,P\}}$)
    end
  end
end

Algorithm 6: description of buildPunctualValues to build punctual values at the quadrature points $(\mathbf{X}_g)_{g\in\{1,..,N_G\}}$ from the gPC coefficients $(U^k)_{k\in\{0,..,P\}}$.

The loop finally ends with a population control algorithm, in case the number of MC particles has diminished due to leaks from the boundary conditions. Algorithm 7 presents one possibility for the uncertain population control. The algorithm is simple and is only based on the possibility to split an MC particle. Splittings are unbiased for every moments of u. This ensures consistency: this may not be the case for combing [51] for example (which is only unbiased for the first moment, i.e. the mean). For more elaborated population control algorithm (such as combing, russian roulette etc.), the algorithm must probably be adapted to this uncertain framework and should be studied case-by-case. Algorithms for an efficient uncertain population control shall probably be designed (merging algorithms for example could be unbiased for the gPC coefficients estimation). We consider this is beyond the scope of this paper.

uncertainPopulationControl(list_of_particles, N_{MC})
Data: list_of_particles, N_{MC}
Result: list_of_particles (with N_{MC} MC particles)
begin
  $N_{MC}^{\text{new}} = 0$
  for $p \in$list_of_particles **do**
    set $s_p = \Delta t$ *#reset the life time of particle p (must go down to zero)*
    $N_{MC}^{\text{new}} + +$
  end
  while $N_{MC}^{new} < N_{MC}$ **do**
    #Choose randomly a particle p in list_of_particles
    p = sampleMCParticleFromList(list_of_particle)
    #splitMCParticle*'s output is a strictly positive*[a] *number of particles*
    #splitMCParticle*'s output may depend on* $w_p, \mathbf{x}_p, \mathbf{v}_p, \mathbf{X}_p$
    #in practice, we choose to split a particle in two if sampled
    split_p = splitMCParticle($\{w_p, \mathbf{x}_p, \mathbf{v}_p, \mathbf{X}_p\}$)
    $w_p \leftarrow w_p \times \frac{1}{\text{split\_p}}$
    for $p' \in \{1, ..., \text{split\_p} - 1\}$ **do**
      set $s_{p'} = s_p$
      set $\mathbf{x}_{p'} = \mathbf{x}_p$
      set $\mathbf{v}_{p'} = \mathbf{v}_p$
      set $w_{p'} = w_p$
      set $\mathbf{X}_{p'} = \mathbf{X}_p$
      list_of_particles.append($\{w_{p'}, \mathbf{x}_{p'}, \mathbf{v}_{p'}, s_{p'}, \mathbf{X}_p\}$)
    end
    $N_{MC}^{\text{new}} +=$ split_p
  end
end

Algorithm 7: description of uncertainPopulationControl: to make sure we always have about N_{MC} uncertain MC particles within domain $\mathcal{D}$

[a] Combing [51], leading to split_p= 0, are only unbiased for the first moment (i.e. the mean u^0, k_{eff}^0): here, we need to be unbiased for every orders $\{0, ..., P\}$, i.e. every gPC coefficients.

With the previous lines, we described the uncertain eigenvalue/eigenvector MC solver we apply in the next section.

5. Numerical results

In this last section, we present some numerical results obtained from the MC resolution of $P-$truncated gPC reduced models (i.e. MC-gPC$_P$) described through sections 3–4. We first begin, in section 5.1, with simple numerical test-cases for which uncertain analytical solutions can be built (in infinite medium). This section has several aims:

– first, it allows giving a hint at what kind of studies (uncertainty propagation, sensitivity analysis etc.) can be efficiently tackled for k_{eff} computations.

– Second, it allows building few uncertain analytical solutions based on already intensively used deterministic ones [52]: this can considerably ease the verification step (cf. V&V, see [53]) for anyone willing to develop the material of this paper in its own MC implementation.

– Third, it allows recovering the spectral convergence of gPC[14], which has already been numerically [1] and theoretically [2] recovered for instationary problems, but in an eigenvalue/eigenvector computation context.

– Finally, it allows showing that perturbation methods, intensively applied in MC computations, are implicitly taken into account by our uncertain framework. In other words, the MC-gPC solver is more general than a perturbation one:

  – it ensures recovering the results of a perturbation analysis if the input random variables present small fluctuations,

  – it allows capturing solutions outside the perturbative regime[15] by numerical integration[16].

In sections 5.3–5.2, we consider test-cases for which, to our knowledge, no analytical solutions are available. The results obtained from MC-gPC are compared to the ones obtained with non-intrusive uncertainty propagation methods (reference). Non-intrusive methods use a simulation code as a black-box and perform the uncertainty analysis of interest by running N times the MC code at some prescribed points[17] $(\mathbf{X}_i, w_i)_{i\in\{1,\dots,N\}}$. MC-gPC presents equivalent accuracies as the most efficient non-intrusive methods, obtained in competitive restitution times. Section 5.2 focuses on a sensitivity analysis study and section 5.3 takes into account geometrical uncertainties. In particular, the test-cases of these sections are not anymore in an infinite medium and they allow checking the population control strategy is efficient in an uncertain context.

Before tackling the numerical results, we would like to make two remarks which the reader must keep in mind while reading the next section, in order to ease the descriptions and interpretations of the test-cases.

Remark 5.1. *From [1], we already have some hints at what can be expected in term of efficiency for the MC-gPC solver, mainly on function trackUncertainParticles. Let us recall few points here:*

– *first, due to the fact that the tallies must be performed on a vector of polynomial coefficients which can be of important size, especially in high stochastic dimensions, the cost of an uncertain MC particle is superior to the cost of a classical MC particle.*

– *As a consequence, the less frequent the tallies, the more efficient the method.*

Now, for an eigenvalue/eigenvector computation, the material of this paper needs at least one additional step with respect to the material of [1] (the k_{eff} update at the end of one iteration). In the

[14] This is possible because we have access to uncertain analytical solutions.

[15] i.e. when the fluctuations of the inputs are not small.

[16] In opposition to numerical differentiation for perturbative methods.

[17] The points $(\mathbf{X}_i, w_i)_{i\in\{1,\dots,N\}}$ allow a consistent discretisation of $(\mathbf{X}, \mathrm{d}\mathcal{P}_{\mathbf{X}})$. Several choices are possible for the points/weights: MC [19], LHS [54, 55], sparse-grids [31], adaptive grids [29], Gauss points [38] etc. The choice of the points is crucial in practice as the efficiency of the non-intrusive method strongly depends on the efficiency of the integration. In the following, Gauss quadrature rules are used for their efficiency in the low to moderate stochastic dimensions [38, 28] tackled in this paper.

following sections, care will be taken to focus on points complementary to the ones already discussed in [1].

Remark 5.2. *The second remark concerns the potential loss of positiveness of the MC-gPC approximations of* $\mathbf{x}, \mathbf{X} \to \int u(\mathbf{x}, \mathbf{v}, \mathbf{X})\, \mathrm{d}\mathbf{v}$ *or of* $\mathbf{X} \to U(\mathbf{X}) = \iint u(\mathbf{x}, \mathbf{v}, \mathbf{X})\, \mathrm{d}\mathbf{v}\, \mathrm{d}\mathbf{x}$ *or of* $\mathbf{X} \to k_{\text{eff}}(\mathbf{X})$. *As explained earlier, function buildPunctualValues can natively embed some positivity preserving reconstruction. Of course, those may be more computationally intensive. For this reason, care is taken in the next paragraph to monitor whether such more elaborated reconstructions are needed in practice or not. In the next section, for the different test-cases, we document each time a positivity preserving reconstructions is needed and whether not using it triggers or not important robustness problems which may compromise or not the computations. A complementary discussion on the topic is also provided after the numerical results in section 5.4.*

5.1. Analytical solutions of uncertain k_{eff} *in an infinite medium*

In this section, we build analytical uncertain solutions of (3) in some particular configurations. If we assume the particles we consider are monokinetic, i.e. $\sigma_\alpha(\mathbf{x}, \mathbf{v}, \mathbf{X}) = \sigma_\alpha(\mathbf{x}, \mathbf{X})$, $\forall \alpha \in \{s, t, f\}, \forall \mathbf{X} \sim \mathrm{d}\mathcal{P}_{\mathbf{X}}, \forall \mathbf{x} \in \mathcal{D}$, with $v = 1$, and that the scattering and fission reactions are deterministic, homogeneous and isotropic, i.e. $P_\alpha(\mathbf{x}, \mathbf{v}, \mathbf{X}))\, \mathrm{d}\mathbf{v} = \mathbf{1}_{\mathbb{S}^2}(\omega)\, \mathrm{d}\omega\ \forall \mathbf{x} \in \mathcal{D}$, $\forall \mathbf{v} = v\omega = \omega \in \mathcal{V} = \mathbb{S}^2$, $\forall \alpha \in \{s, f\}$, equation (3) resumes to

$$\begin{aligned} \omega \cdot \nabla_{\mathbf{x}} u(\mathbf{x}, \omega, \mathbf{X}) + \sigma_t(\mathbf{x}, \mathbf{X}) u(\mathbf{x}, \omega, \mathbf{X}) = \sigma_s(\mathbf{x}, \mathbf{X}) \int u(\mathbf{x}, \omega', \mathbf{X})\, \mathrm{d}\omega', \\ + \frac{\nu_f(\mathbf{x}, \mathbf{X}) \sigma_f(\mathbf{x}, \mathbf{X})}{k_{\text{eff}}(\mathbf{X})} \int u(\mathbf{x}, \omega', \mathbf{X})\, \mathrm{d}\omega'. \end{aligned} \tag{13}$$

Let us integrate the above equation with respect to $\mathbf{x} \in \mathcal{D}$ to get

$$\begin{aligned} \omega \cdot \int_{\partial\mathcal{D}} u(\mathbf{x}, \omega, \mathbf{X})\, \mathrm{d}\mathbf{x} + \int_{\mathcal{D}} \sigma_t(\mathbf{x}, \mathbf{X}) u(\mathbf{x}, \omega, \mathbf{X})\, \mathrm{d}\mathbf{x} = \int_{\mathcal{D}} \sigma_s(\mathbf{x}, \mathbf{X}) \int u(\mathbf{x}, \omega', \mathbf{X})\, \mathrm{d}\omega'\, \mathrm{d}\mathbf{x}, \\ + \int_{\mathcal{D}} \frac{\nu_f(\mathbf{x}, \mathbf{X}) \sigma_f(\mathbf{x}, \mathbf{X})}{k_{\text{eff}}(\mathbf{X})} \int u(\mathbf{x}, \omega', \mathbf{X})\, \mathrm{d}\omega'\, \mathrm{d}\mathbf{x}. \end{aligned} \tag{14}$$

Assume periodic boundary conditions (i.e. such that $\int_{\partial\mathcal{D}} u(\mathbf{x}, \mathbf{v}, \mathbf{X})\, \mathrm{d}\mathbf{x} = 0$) and homogeneous cross-sections (i.e. $\sigma_\alpha(\mathbf{x}, \mathbf{X}) = \sigma_\alpha(\mathbf{X})$), we obtain

$$\sigma_t(\mathbf{X}) = \sigma_s(\mathbf{X}) + \frac{\nu_f(\mathbf{X}) \sigma_f(\mathbf{X})}{k_{\text{eff}}(\mathbf{X})}, \tag{15}$$

hence,

$$k_{\text{eff}}(\mathbf{X}) = \frac{\nu_f(\mathbf{X}) \sigma_f(\mathbf{X})}{\sigma_t(\mathbf{X}) - \sigma_s(\mathbf{X})} = \frac{\nu_f(\mathbf{X}) \sigma_f(\mathbf{X})}{\sigma_a(\mathbf{X}) + \sigma_f(\mathbf{X})}. \tag{16}$$

Note that if the cross-sections and the multiplicity are deterministic, we recover the classical expression of a k_{eff} in an infinite medium. In the following, we intensively have resort to the conditions of test-case UD2O-1-0-IN[18] of [52]. MC-gPC results are obtained with $N_{MC} = 10^5$ and this number remains the same all along the iterations[19]. Depending on where the uncertainty comes from, the probability measure of $k_{\text{eff}}(\mathbf{X})$ may be different. Let us consider four different sources of uncertain-

[18] Infinite medium, $\overline{\nu}_f = 1.7$, $\overline{\sigma}_a = 0.027314$, $\overline{\sigma}_s = 0.464338$, $\overline{\sigma}_f = 0.054628$ where the $\overline{\ }$ notation stands for averaging.

[19] Due to the periodic boundary conditions, no particle is lost and the population control algorithm is not active.

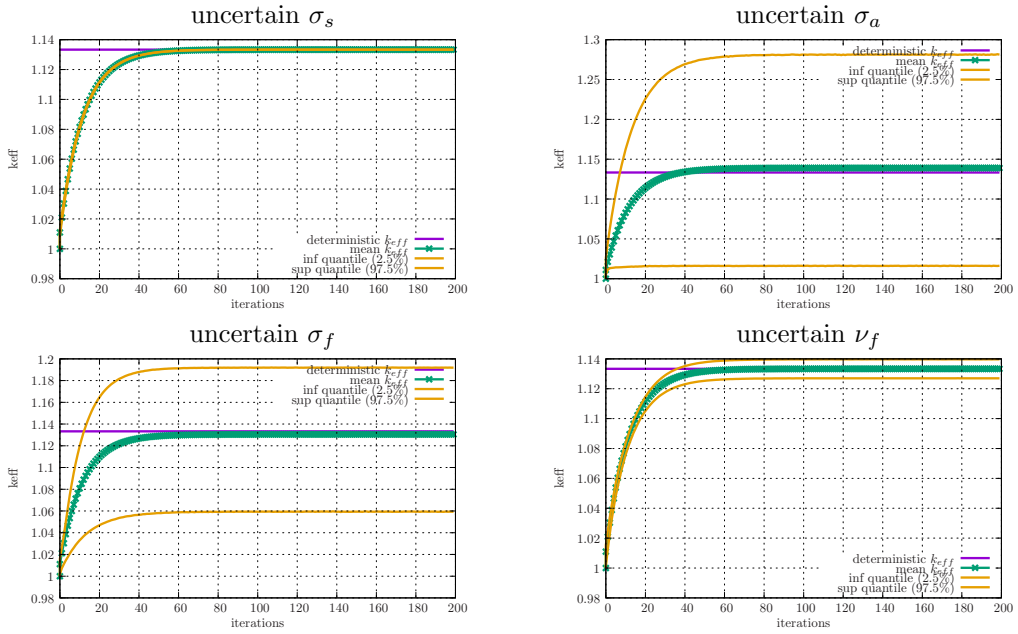

Figure 1: Evolution of $(k_{\text{eff}}^n)_{n\in\{1,\dots,200\}}$ where n is the number of the (stochastic) power iteration for MC-gPC$_{P=3}$ in the four situations depicted in section 5.1. Care has been taken to have $k_{\text{eff}}^n \approx k_{\text{eff}}^\infty$ for n close to 200 in the four situations.

ties[20]:

– UD2O-1-0-IN-σ_s: if $\nu_f, \sigma_f, \sigma_a$ are deterministic and $\sigma_s(X) = \overline{\sigma}_s + \hat{\sigma}_s X$, then, independently of the distribution of X, k_{eff} is deterministic[21] and is given by $k_{\text{eff}}(X) = k_{\text{eff}} = 1.1333333$ (as in [52]). Figure 1 (top left) presents the results $(k_{\text{eff}}^n)_{n\in\{1,\dots,200\}}$ obtained with MC-gPC$_{P=3}$ with respect to the number of (stochastic) power iterations n. Each time, care has been taken to make sure n is large enough to have $k_{\text{eff}}^n \approx k_{\text{eff}}^\infty$ as $n \approx 200$. Figure 1 displays the reference solution $k_{\text{eff}}(X) = k_{\text{eff}} = 1.1333333$ together with the mean solution and the 2.5% and 97.5% quantiles[22]. Here, the solution being deterministic, the quantiles are close to the mean and the confidence interval is only an *error bound*, not an uncertainty bound. In a sense, this shows that MC-gPC accurately captures deterministic solutions. This is also emphasized in figure 2 displaying the reference histogram obtained by sampling (10^6 samples) X through (16) and the MC-gPC$_{P=3}$ one obtained by sampling X through the gPC approximation $k_{\text{eff}}^P(X) = \sum_{k=0}^P k_{\text{eff}}^k \phi_k^X(X)$ for $P = 3$: we recover a Dirac at $k_{\text{eff}} = 1.133333$.
From an efficiency point of view, it is interesting noting that a non-intrusive method would have needed at least[23] $N = 2$ runs of an MC black box code to conclude that the solution is not affected by the uncertainty on σ_s. For this test-problem, one run of an MC black-box

[20] In the next paragraph, if $Q = 1$, then we denote $\mathbf{X} = X$ to insist on the fact that $\mathbf{X}$ is scalar.
[21] Indeed, (16) is independent of σ_s.
[22] The quantiles have been evaluated with 10^6 samples of the MC-gPC$_{P=3}$ approximation.
[23] But in practice, much more are needed.

code has the *same computational cost* as one run of the MC-gPC one. As a consequence, we can consider we, at least, gain a factor $\times 2$ here.
No positivity preserving procedure (see remark 5.2) is needed all along the calculation.

– UD2O-1-0-IN-σ_a: if $\nu_f, \sigma_s, \sigma_f$ are deterministic and $\sigma_a(X) = \overline{\sigma}_a + \widehat{\sigma}_a X$ is uncertain with $X \sim \mathcal{U}([-1,1])$ and $\hat{\sigma}_a = 10^{-2}$, we recover the results of figures 1–2 (top right for both). On figure 1 (top right), we first realise that the mean k_{eff} is slightly higher than 1.133333[24]: the mean $k^0_{\text{eff}} = \int k_{\text{eff}}(X)\, \mathrm{d}\mathcal{P}_{\mathbf{X}}$ in this case is different from the k_{eff} at the design point $\overline{X}$ (mean of X), i.e. $k^0_{\text{eff}} \neq k_{\text{eff}}(\overline{X}) = 1.133333$. This implies some nonlinearities are at play and must be taken into account to accurately recover the behavior of the uncertain solution. We will come back on this point few lines below, when comparing gPC methods to perturbative ones. The area between the curves of the quantiles for 2.5% and 97.5% ensures a 95% probability of having the k_{eff} between those bounds. Note that the quantile curves are not exactly equidistant from the mean: this is because the k_{eff} distribution is slightly skewed as testifies figure 2 (top right). As a consequence, k_{eff} closer to 1 are more probable than higher ones.
The quantiles have been estimated sampling 10^6 points $(X_i)_{i\in\{1,\dots,10^6\}}$ according to $\mathrm{d}\mathcal{P}_{\mathbf{X}}$ and estimating them classically through the MC-gPC$_{P=3}$ approximation of $(k^P_{\text{eff}}(X_i))_{i\in\{1,\dots,10^6\}}$. From an efficiency point of view, the gain is not $\times 10^6$ as there exists non-intrusive methods relying on much less points: the most efficient non-intrusive method we found for this test-case is non-intrusive gPC with $P = 3$ and $N = 4$ Gauss-Legendre points, see [38, 39]. As a consequence, in this case, the gain is a factor $\times 4$ between an efficient non-intrusive method and MC-gPC.
No positivity preserving procedure (see remark 5.2) is needed all along the calculation.

– UD2O-1-0-IN-σ_f: if $\nu_f, \sigma_a, \sigma_s$ are deterministic and $\sigma_f(X) = \overline{\sigma}_f + \hat{\sigma}_f X$ is uncertain with $X \sim \mathcal{U}([-1,1])$ and $\hat{\sigma}_f = 10^{-2}$, we obtain the figures 1–2 (bottom left for both). This time, the mean k_{eff} is slightly lower than the deterministic one (i.e. once again, nonlinearities are at play). The quantile curves show that 95% of the probable quantiles are between 1.06 and 1.19. This is emphasized by figure 2 (bottom left) where we can see that the k_{eff} distribution is, in this case, skewed toward the important k_{eff} values.
On this problem, the gain is $\times 4$ in restitution time with respect to the most efficient non-intrusive method available ($N = 4$ Gauss-Legendre points of non-intrusive gPC with $P = 3$).
No positivity preserving procedure (see remark 5.2) is needed all along the calculation.

– UD2O-1-0-IN-ν_f: if $\sigma_f, \sigma_a, \sigma_s$ are deterministic and $\nu_f(X) = \overline{\nu}_f + \hat{\nu}_f X$ with $X \sim \mathcal{U}([-1,1])$ and $\hat{\nu}_f = 10^{-2}$, then it is, in this case, easy prooving that the k_{eff} distribution is uniform in interval $[1.126666, 1.139999]$. Indeed, see (16), in this case, the transformation $X \to k_{\text{eff}}(X)$ is linear. In this case, the mean k_{eff} is equal to $k_{\text{eff}}(\overline{X}) = 1.133333$ and is accurately recovered by the MC-gPC numerical method (see figure 1 bottom right). Besides here, the 95% confidence interval is narrower than in the previous cases, see figure 1 (bottom right). MC-gPC also allows recovering the uniform distribution of k_{eff} as attests figure 2 (bottom right).
On this problem, the gain is $\times 4$ in restitution time with respect to the most efficient non-intrusive method available ($N = 4$ Gauss-Legendre points of non-intrusive gPC with $P = 3$).
No positivity preserving procedure (see remark 5.2) is needed all along the calculation.

[24] given by the deterministic solution.

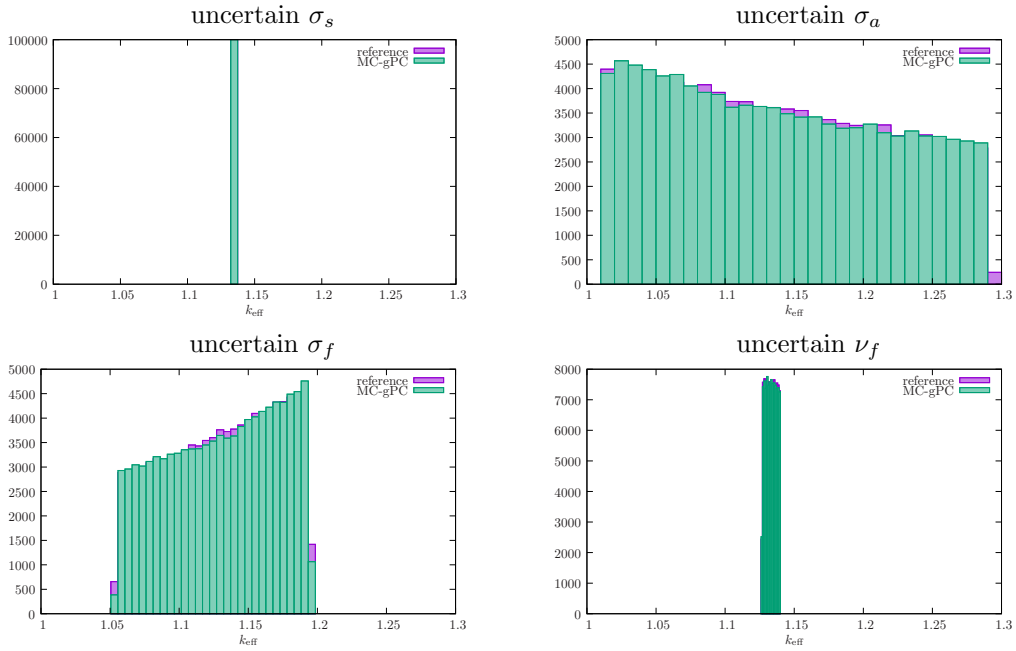

Figure 2: Comparisons of the histograms from the uncertain analytical solution (16) and the MC-gPC approximation with $P = 3$ in the four situations depicted in section 5.1.

In the above lines, care has been taken to focus and comment on the uncertainty quantification results rather than on reference/MC-gPC comparisons. In every cases, MC-gPC with only $P = 3$ allows recovering accurate results with only one run of a code[25]. The good agreement with small polynomial order ($P = 3$) is in accordance with the fast (spectral) convergence numerically recovered in [1] and theoretically recovered in [2] (but for the unstationary linear Boltzmann equation, not for an eigenvalue problem). For the histograms of figure 2 for example, the differences between the reference and the MC-gPC histograms only come from the different initial samplings (10^6 samples) of X. If we take exactly the same samples for X as input of (16) and as input of $k_{\text{eff}}^P(X) = \sum_{k=0}^P k_{\text{eff}}^k \phi_k^X(X)$ with $P = 3$, both histograms are not discernable.
We also here insist on the fact that every gPC results were obtained by only running once each uncertain computations: this means that we only have to take care once that the (stochastic) power iteration method is in a stationary regime, i.e. that $k_{\text{eff}}^n(X) \approx k_{\text{eff}}^\infty(X)$ where n is an iteration of the while loop of algorithm 4. With a non-intrusive application, care must be taken to have $k_{\text{eff}}^n(X_i) \approx k_{\text{eff}}^\infty(X_i)$ for every realisations $(X_i)_{i\in\{1,\dots,N\}}$ of the uncertain inputs. This can be tedious to check in practice $\forall i \in \{1, ..., N\}$.

Let us now perform a convergence study on k_{eff} with respect to the polynomial order P. For

[25] With the non-intrusive gPC, comparable accuracies are obtained with $P = 3$ and $N = 4$ Gauss-Legendre points.

this, let us consider the case UD2O-1-0-IN-σ_a, i.e. σ_a is uncertain[26], i.e. we have

$$k_{\text{eff}}(X) = \frac{\nu_f \sigma_f}{\overline{\sigma}_a + \hat{\sigma}_a X + \sigma_f}.$$

This corresponds to the figures 1–2 (both top left). Figure 3 (left) presents the results of a convergence study, the logarithm of the error with respect to P, performed with MC-gPC$_P$. The fluctuations of the absorption cross-section is $\hat{\sigma}_a = 10^{-2}$. Very accurate results can be reached

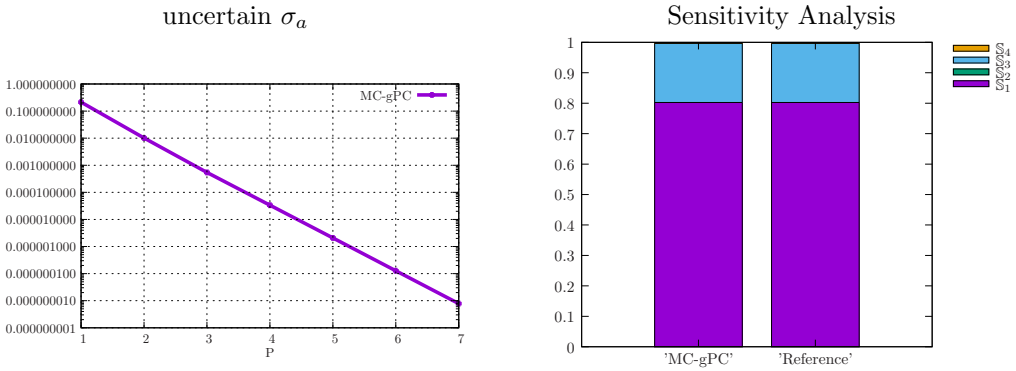

Figure 3: Convergence study on the uncertain σ_a case of section 5.1. The plot displays the logarithm of the L^2-norm of the error with respect to truncated order P.

with low polynomial orders P. The linear curve obtained on figure 3 (left) tends to show that spectral convergence is reached for this test-problem. The theoretical spectral convergence on an eigenvalue/eigenvector problem remains beyond the scope of this paper. Still, the numerical results are promising.

Let us tackle a third point now: pertubative methods are commonly used in MC codes, see [12, 21, 22]. The interested reader may wonder what are the differences between them and the gPC developments we use in this paper[27]. Perturbative methods are based on a Taylor development around the mean $\overline{X}$ of X: we can always rewrite $X = \overline{X} + \epsilon$ (if X has finite variance) with ϵ a centered fluctuation. Then we can perform the Taylor development of $k_{\text{eff}}(X)$ when $\epsilon \sim 0$:

$$k_{\text{eff}}(X) \underset{\epsilon \sim 0}{\approx} k_{\text{eff}}(\overline{X}) + \epsilon \partial_X k_{\text{eff}}(\overline{X}) + \sum_{k=2}^{P} \epsilon^k \partial_X^k k_{\text{eff}}(\overline{X}). \tag{17}$$

The above polynomial approximation relies on hypothesis $\epsilon \sim 0$ and uses the derivatives of different orders with respect to X. On another hand, we recall that for gPC, we have

$$k_{\text{eff}}(X) \approx \sum_{k=0}^{P} k_{\text{eff}}^k \phi_k^X(X) \text{ with } \forall k \in \{0, .., P\},\ k_{\text{eff}}^k = \int k_{\text{eff}}(X) \phi_k^X(X)\, \mathrm{d}\mathcal{P}_{\mathbf{X}}. \tag{18}$$

[26] The same could be done for every other situations but the material would be redundant.
[27] After all, both are polynomial based approximations.

Let us consider a simple test-case on which few differences/characteristics can be illustrated. We consider a modification of UD2O-1-0-IN-σ_a for which[28] $\overline{\sigma}_a = 0.27314$, and two different values of $\hat{\sigma}_a = 0.2$, and $\hat{\sigma}_a = 0.02$. We suggest comparing the asymptotical results (i.e. when the polynomial coefficients of the Taylor and gPC expansions are analytically computed) obtained from perturbative methods and gPC. Figure 4 presents the results obtained by perturbative/Taylor methods and

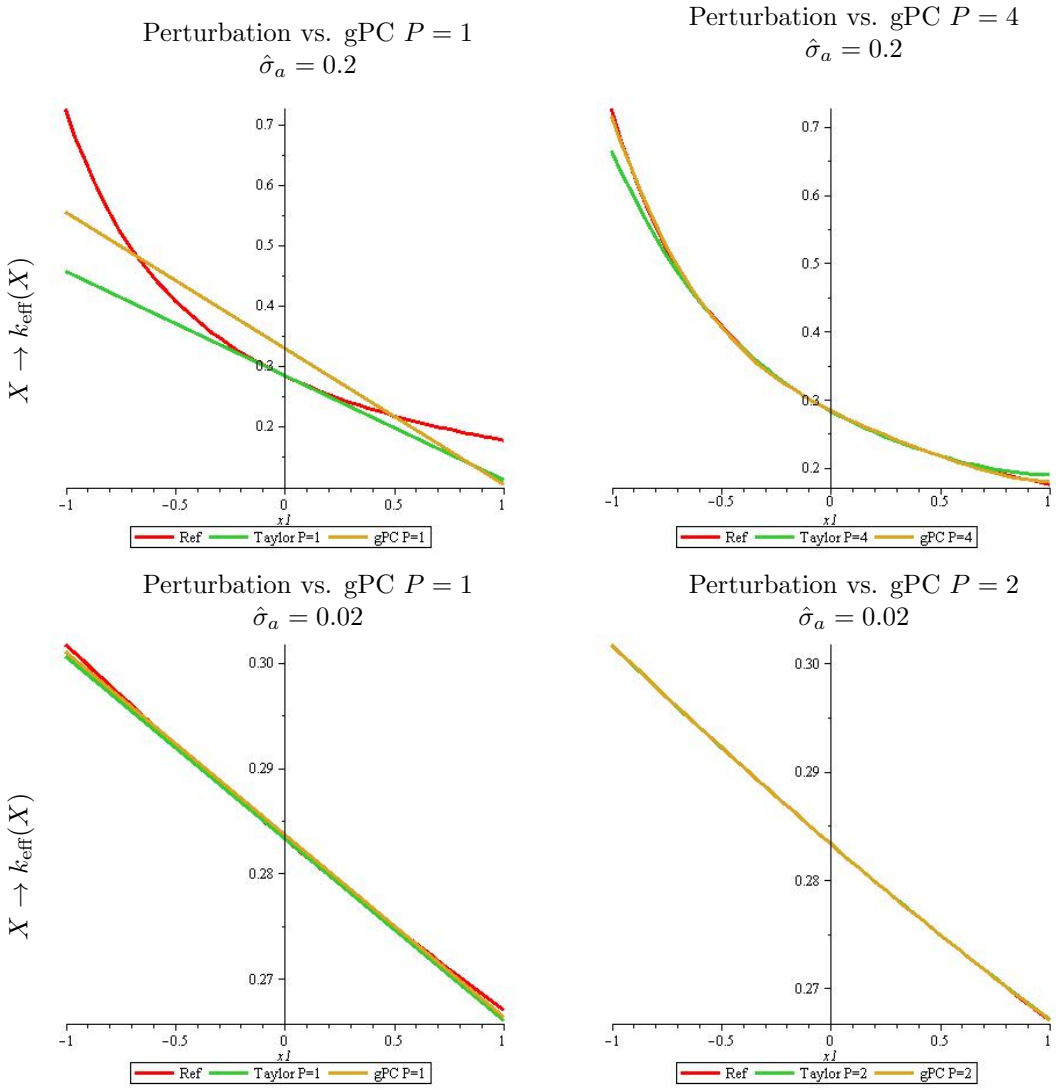

Figure 4: Comparisons of pertubative methods and gPC ones on the UD2O-1-0-IN-σ_a problem.

[28]The mean absorption cross-section has been multiplied by 10 with respect to configuration UD2O-1-0-IN-σ_a.

gPC for several polynomial order $P \in \{1, 2, 4\}$ in the same conditions. Figure 4 (top left) for $P = 1$ presents the reference curve (red) $X \to k_{\text{eff}}(X)$ and the $\text{Taylor}_{P=1}$ and $\text{gPC}_{P=1}$ ones. The $\text{Taylor}_{P=1}$ curve is tangent to the point $k_{\text{eff}}(\overline{X}) = k_{\text{eff}}(0)$ whereas the $\text{gPC}_{P=1}$ focuses on the trend of $X \to k_{\text{eff}}(X)$. For this reason, perturbative methods are often called *local* approximation methods whereas gPC ones are commonly called *global* ones. Of course, if P grows, both approximation methods coincide, see figure 4 (top right) for $P = 4$, even if $\text{gPC}_{P=4}$ gives slightly better results than $\text{Taylor}_{P=4}$. Now, evaluating fourth order *derivatives*, see (17), with an MC solver can be quite cumbersome due to the numerical noise it often implies. On another hand, gPC only relies on integral evaluations (see (18)), fitted to an MC solver framework.

The bottom pictures of figure 4 compares perturbative and gPC methods for $P = 1, 2$ for a smaller variance of σ_a, i.e. a smaller perturbation: in the regime $\hat{\sigma}_a X = \epsilon \ll 1$, both methods give equivalent results. For $P = 2$, the Taylor and gPC curves are not discernable. In this sense, gPC captures the perturbative regime: perturbative results can be recovered from gPC approximations if the input variances are small (or the transformation close to linear).

Finally, we consider a 4D uncertain UD2O-1-0-IN with $\mathbf{X} = (X_1, X_2, X_3, X_4)^t$, with $(X_i)_{i\in\{1,\dots,4\}}$ independent uniformly distributed in $[-1, 1]$. In this case, the analytical solution is given by

$$k_{\text{eff}}(\mathbf{X}) = k_{\text{eff}}(X_1, X_2, X_3, X_4) = \frac{(\overline{\nu}_f + \hat{\nu}_f X_4)(\overline{\sigma}_f + \hat{\sigma}_f X_3)}{\overline{\sigma}_a + \hat{\sigma}_a X_1 + \overline{\sigma}_f + \hat{\sigma}_f X_3}, \tag{19}$$

where the overlined quantities are the same as in UD2O-1-0-IN and $\hat{\sigma}_\alpha = \hat{\nu}_f = 10^{-2}$, $\forall \alpha \in \{a, s, t\}$. Now, we would not only like to perform an uncertainty propagation but also a sensitivity analysis:

	$\mathbb{S}_1$ (σ_a)	$\mathbb{S}_2$ (σ_s)	$\mathbb{S}_3$ (σ_f)	$\mathbb{S}_4$ (ν_f)
MC-$\text{gPC}_{P=3}$	0.8025142731	$2.32595730012\times10^{-10}$	0.1945516072	0.001831871428
Analytical	0.8025421202	0.0000000000	0.1945282914	0.001832695030

Table 1: Results of the (Sobol) sensitivity analysis on the 4D uncertain version of UD2O-1-0-IN. Note that interactions between the uncertain inputs are negligible for this test-case.

we aim at identifying which of the uncertain input parameters explain the most the variance of the output (k_{eff} here). Many sensitivity indices are available in the literature, see [18] for an exhaustive review, but not every of them are efficient. The most reliable ones, Sobol indices see [18] denoted by $(\mathbb{S}_i)_{i\in\{1,\dots,Q\}}$ in this document, are usually very costly and sometimes computationally out of reach (need for $(Q+2)\times N$ runs of a black box code with $N \sim 100-1000$, see [56]). We can evaluate those Sobol indices on the previously described test-case from the computed gPC coefficients, see [57, 58]. The analytical values of the Sobol indices are given in table 1 together with numerical results obtained with MC-$\text{gPC}_{P=3}$ $\forall X_i, i \in \{1, .., 4\}$. Figure 3 (right) presents a stacked histogram of the same results. Qualitatively, with figure 3 (right), we can see that the MC-$\text{gPC}_{P=3}$ approximation gives satisfactory results. Parameter X_1 related to the uncertainty of σ_a explains 80.25% of the k_{eff} variance. Parameter X_3, related to the uncertainty of σ_f, explains 19.45% of the same quantities. The remaining (0.30%) percents are explained by the other variables and their interactions. With table 1, we recover the fact that X_2, controling the uncertainty on σ_s, is not influent: of course, numerically with MC-gPC, we do not obtain exactly zero, there remains a numerical MC error. Otherwise, MC-$\text{gPC}_{P=3}$ (hence $(P + 1)^4 = (3 + 1)^4 = 256$ coefficients) accurately captures the Sobol indices with only 1 run of the MC-gPC code.

From an efficiency point of view, once again, the same study can be carried out non-intrusively. The best results are obtained with non-intrusive gPC with $P = (P_{1D}+1)^4 = (3+1)^4 = 256$ and $N = N_{1D}^4 = 5^4 = 625$. Table 2 presents the computational times obtained with MC-gPC$_{P=3}$ and

	$N_{MC} = 10^4$	$N_{MC} = 10^5$	$N_{MC} = 10^6$
MC-gPC$_{P=3}$	43.2 s.	412.1 s.	4021.1 s.
	$N_{MC} = 10^4$, $N^4 = 5^4 = 625$	$N_{MC} = 10^5$, $N^4 = 625$	$N_{MC} = 10^5$, $N^4 = 625$
ni-gPC$_{P=3}$	$0.77 \times 625 = 481.2$ s.	$7.17 \times 625 = 4481.2$ s.	$73.1 \times 625 = 45687.5$ s.
sequential gain	$\times 11.13$	$\times 10.87$	$\times 11.36$

Table 2: Results of the performance analysis on the 4D uncertain version of UD2O-1-0-IN. MC-gPC is compared to non-intrusive gPC (ni-gPC) for equivalent accuracies.

non-intrusive gPC$_{P=3}$ (ni-gPC$_{P=3}$) leading to equivalent accuracies on the Sobol indices previously commented on. In table 2, for MC-gPC, are displayed the computational times of one run with different number of N_{MC} of MC particles. For ni-gPC are displayed the average computational time over the $N^4 = 625$ runs times $N^4 = 625$. The sequential gain with MC-gPC is displayed in the last line of table 2: globally, a factor $\times 10$ is gained if every non-intrusive runs are performed sequentially. The gain here is not of a factor $\times N = 625$ mainly because when MC-gPC needs to compute as many coefficients, its computational time is affected, see remark 5.1. No positivity preserving procedure (see remark 5.2) is needed all along the calculation.

In the next sections, we consider test-cases for which, to our knowledge, no analytical solution is available. The reference solutions are computed using non-intrusive gPC [38, 40, 58].

5.2. Sensitivity Analysis: an efficient computation of Sobol's indices

In this section, we revisit test-case UD2O-1-0-SL[29] of [52] by considering uncertain absorption ($\sigma_a(\mathbf{X}) = \sigma_a(X_1)$), scattering ($\sigma_s(\mathbf{X}) = \sigma_s(X_2)$), fission ($\sigma_f(\mathbf{X}) = \sigma_f(X_3)$) cross-sections and multiplicity ($\nu_f(\mathbf{X}) = \nu_f(X_4)$). For this test-case, we choose $\hat{\sigma}_a = 10^{-2} = \hat{\sigma}_s = \hat{\sigma}_f = \hat{\nu}_f$ just as in the sensitivity analysis performed in the previous section 5.1. In other words, this test-case is in exactly the same conditions as the 4D study of section 5.1 but with outward boundary conditions on the right hand side of the spatial domain. The MC-gPC$_{P=4}$ computations are performed with $N_{MC} = 10^7$ MC particles. Figure 5 presents the mean and the 2.5% and 97.5% quantile curves of the uncertain eigenvalue k_{eff} (figure 5 right) and of the uncertain eigenvector $x \to \int u(x, \omega, \mathbf{X})\, \mathrm{d}\omega$ (figure 5 left). On figure 5 right, we can see that *in mean*, the benchmark is critical. But the k_{eff} have 95% probability of being between $[0.94, 1.06]$. On the left picture of figure 5, we can see, *via* the 95% quantile curves, that the uncertainty is more pronounced in the left hand side of the domain and less important in the vicinity of the right boundary. The computation has been carried out with MC-gPC$_{P=4}$ implying the estimation of $(P+1)^Q = (4+1)^4 = 625$ gPC coefficients during the MC resolution.

Now, one may wonder which one(s) of the uncertain parameter(s) is the most influent in this configuration. Figure 6 presents the results of a sensitivity analysis based on Sobol's indices[30] [59, 18] on the uncertain eigenvalue k_{eff} : figure 6 (left) presents the percentage of variance explained

[29] The conditions are the same as test-case UD2O-1-0-IN of section 5.1 but with outward boundary condition on the right hand side of $\mathcal{D} = [0, 10.371065]$, i.e. at $x = 10.371065$.

[30] amongst the most reliable ones, see [18].

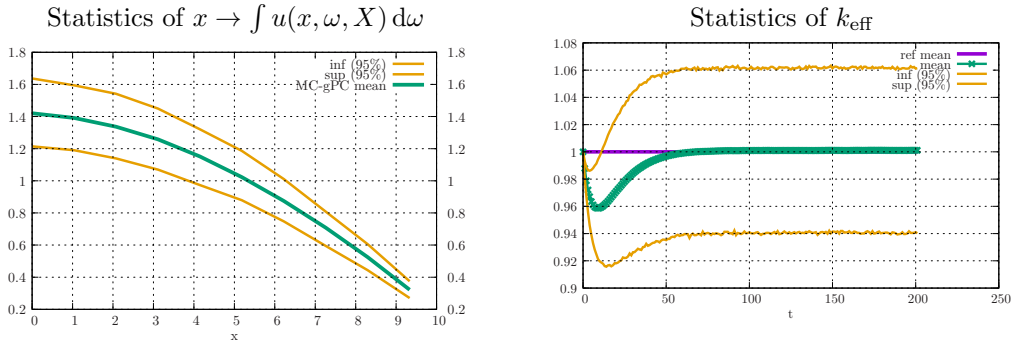

Figure 5: UD2O-1-0-SL problem [52] with uncertain absorption X_1, scattering X_2, fission X_3 cross-sections and multiplicity X_4. Mean and 95% quantiles for the eigenvalue k_{eff} and its corresponding eigenvector $x \to \int u(x,\omega,\mathbf{X})\,\mathrm{d}\omega$.

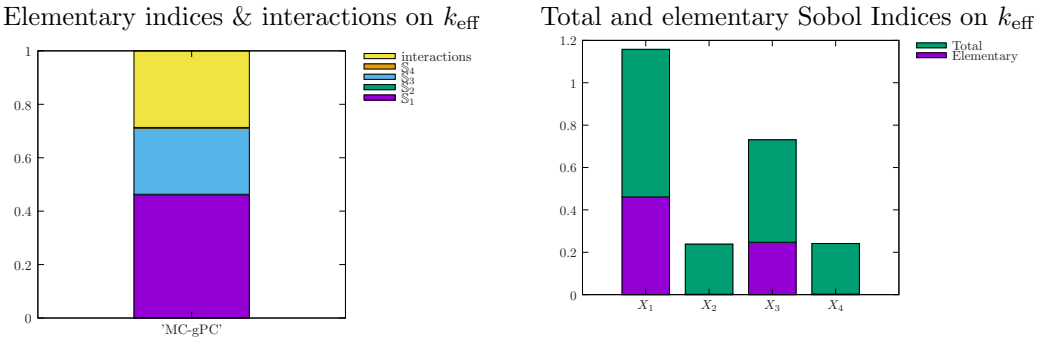

Figure 6: UD2O-1-0-SL problem [52] with uncertain absorption X_1, scattering X_2, fission X_3 cross-sections and multiplicity X_4. Sobol indices of the uncertain eigenvalue k_{eff} .

by each parameters independently (they are commonly called *elementary effects*) and the part explained by interactions between those parameters. About 50% of the variance on k_{eff} is explained by the uncertainty on $\sigma_a(X_1)$, 20% comes from the uncertainty on $\sigma_f(X_3)$ and the remaining 30% comes from interactions between the different sources of uncertainties. It is interesting noticing that once the boundary condition changed with respect to the 4D study of section 5.1, the results of the sensitivity analysis are considerably different: in section 5.1-figure 3 (right), 80% of the fluctuations were explained by the uncertainty on σ_a. None of the four random variables were interacting. Figure 6 (right) compares the elementary and total effects for every of the four random inputs: we recover the fact that all random variables strongly interact with each other (due to their important total effects). Variables σ_s and ν_f only play a role in interaction with the other ones. The comparison between the 4D test-cases of section 5.1 and of section 5.2 allows insisting on the importance of being able to carry out systematic reliable sensitivity analysis in every configurations of interest: a small change in the conditions of the test-problem can lead to completely different results and interpretations.

Figure 7 presents a sensitivity analysis on the uncertain eigenvector, i.e. on the profile $x \to \int u(x,\omega,\mathbf{X})\,\mathrm{d}\omega$. Each picture compares the elementary and total effects of each input random

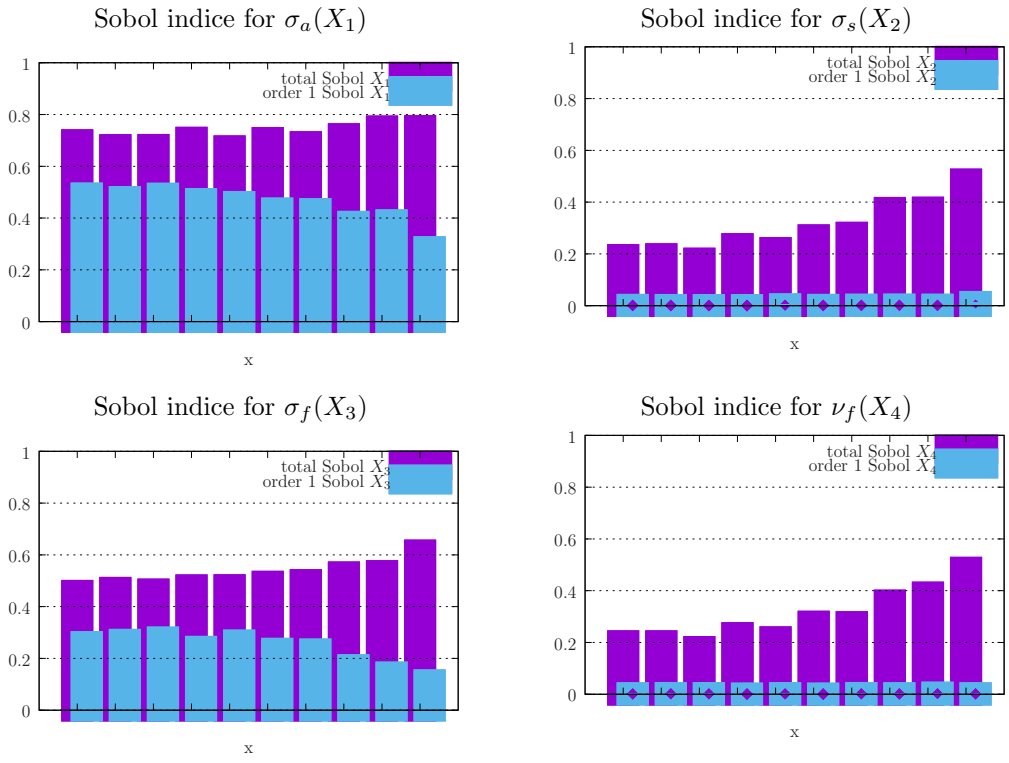

Figure 7: UD2O-1-0-SL problem [52] with uncertain absorption X_1, scattering X_2, fission X_3 cross-sections and multiplicity X_4. Elementary and Total Sobol spatial profiles of the uncertain eigenvector $x \to \int u(x, \omega, \mathbf{X})\, \mathrm{d}\omega$.

variables on the eigenvector. From figure 7 top-right, we can see that the elementary effects of $\sigma_s(X_2)$ and $\nu_f(X_4)$ have an homogeneous role on the spatial profile. But their total effects are stronger in the vicinity of the right hand side boundary. On another hand, the uncertainties from $\sigma_a(X_1), \sigma_f(X_3)$ (left column of figure 7) explain the increase of variance in the vicinity of $x \sim 0$ as their elementary Sobol indices are slightly more important in this spatial region. Of course, interactions remain the preponderant effect, even on the eigenvector, especially in the vicinity of the right boundary.
Finally, no positivity preserving procedure (see remark 5.2) is needed all along the calculation.

5.3. Geometrical uncertainty: UD2O-H2O(1)-1-0-SL with uncertain interface position

In this section, we consider a configuration based on test-problem UD2O-H2O(1)-1-0-SL from [52]: originally, the test-case is a critical benchmark with a slab of UD2O[31] of dimension $\overline{x}_c = 9.214139cm$ next to an H2O[32] reflector of thickness $1.830563cm$. The whole geometry consists of interval $\mathcal{D} = [0, 11.044702]$. Now, the test-case is made uncertain by considering that $x_c(X) =$

[31] $\overline{\nu}_f = 1.7$, $\overline{\sigma}_a = 0.027314$, $\overline{\sigma}_s = 0.464338$, $\overline{\sigma}_f = 0.054628$.
[32] $\overline{\nu}_f = 0.0$, $\overline{\sigma}_a = 0.054628$, $\overline{\sigma}_s = 0.491652$, $\overline{\sigma}_f = 0.0$.

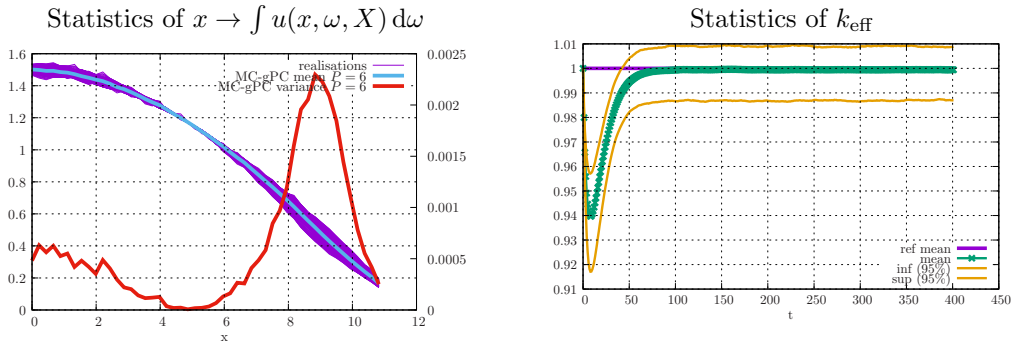

Figure 8: UD2O-H2O(1)-1-0-SL problem [52] with uncertain interface between UD2O and H2O.

$\overline{x}_c + \hat{x}_c X$ with $X \sim [-1,1]$. In practice, we keep $\overline{x}_c = 9.214139$ and set $\hat{x}_c = 0.99$. The MC-gPC results are obtained with $N_{MC} = 3 \times 10^7$ MC particles and $P = 6$.
Figure 8 presents the results (k_{eff}, u) obtained with MC-gPC$_{P=6}$ for this test problem. Figure 8 (left) displays spatial profiles of the mean $x \to \mathbb{E}[U^P](x) = U_0(x)$ and of realisations $x \to U^P(x,X) = \sum_{k=0}^{P} U_k(x)\phi_k^X(X)$, with $U_k(x) = \int u_k(x,\omega)\,\mathrm{d}\omega, \forall k \in \{0,..,P\}$ (left scale) and the variance $x \to \mathbb{V}[U^P](x) = \sum_{k=1}^{P} U_k^2(x)$ (right scale). Note that the $(U_k)_{k\in\{1,...,P\}}$ are taken at the last iteration (no averaging through the iterations). The uncertainty is mainly localized in the vicinity of $\overline{x}_c \pm 1$. But it is also non negligible in the vicinity $x \sim 0$. Figure 8 (right) displayes the mean k_{eff} and the 95% confidence intervals through the iterations of the (stochastic) power iteration method. The distribution is skewed toward sub-critical k_{eff} as testifies the non-symmetric 2.5% and 97.5% quantile curves. Still, the test-case remains, in mean, critical.
Finally, this test-case with geometrical uncertainties presents a particular advantage of using an intrusive framework: the mesh is unique and simple. Let us describe the situation in a non-intrusive framework. Taking into account geometrical uncertainties for the test-case of this section would impose:

- either relying on as many meshes as points $(X_i)_{i\in\{1,..,N\}}$ at which we have to run the code,
- or relying on some averaging procedures within the cells affected by the uncertainties.

In both cases, the study is either tedious and cumbersome or needs some modeling hypothesis and code modifications (intrusive) to define the uncertain average cross-sections seen by the deterministic MC particles within the cell concerned by the uncertainty. In the present test-case, having resort to several meshes is not especially hard but more complex geometries can be extremely hard to tesselate and each mesh may demand a considerable amount of work. Finally, even if the meshes are easy to generate, it remains the question of the visualisation of the statistical quantities (mean, variance, realisations etc.) with respect to $\mathbf{x}$: on which mesh should we plot them? How do we compute those quantities when cells are overlapping from one realisation to the other? In this context, MC-gPC considerably eases the pre/post-treatments in addition to ensuring an important gain.
From a computational point of view, the MC-gPC$_{P=6}$ which produced the results of figure 8 takes about 527.6 s. on one computational unit. The average computational cost of a deterministic black-box MC criticality code in order to obtain the same accuracy take $= 427.8s$. and needs $N = 8$

Gauss-Legendre points, hence a total sequential computational time of $427.8s. \times 8 = 3422.4s.$. On this benchmark, MC-gPC ensures a gain of a factor $\times \frac{3422.4}{527.6} \approx 6.48$. Finally, a positivity preserving procedure (see remark 5.2) is needed all along the calculation.

5.4. A final test-case challenging the positiveness of the gPC approximation

All along the paper, care as been taken to anticipate on one well-known flaw of the gPC approximations, namely the potential loss of positiveness of the $P-$truncated quantities approximated by a gPC expansions (here, eigenvalue k_{eff} and the eigenvector u). In all the previous test-cases, this *loss of positiveness was not encountered*. In this section, we are going to show that this is because all the previous benchmarks were converged (N_{MC} are important enough). We here discuss on the consequences of having to small polynomial orders P or number of MC particles N_{MC}. We know that

- theoretically, due to the different hypothesis made in section 2, $\forall X \sim \mathrm{d}\mathcal{P}_{\mathbf{X}}$, $u(x,\omega,X) \geq 0$,
- numerically, we can encounter some situations for which $\exists \Omega_- \subset \mathrm{d}\mathcal{P}_{\mathbf{X}}$ such that[33] $|\Omega_-| > 0$ and $\forall X \in \Omega_-$, $u^{P,N_{MC}}(x,\omega,X) < 0$.

The first situation occurs all along the previous benchmarks. The second one

- occurs, for example (see figure 9), when considering the benchmark of the previous section 5.3 with a coarser MC discretisation ($1000\times$ less MC particles),
- *does not* occur when considering the benchmark of the previous sections with coarser polynomial approximations (not even for $P = 1$).

This tends to show that the MC error is likely to be responsible for the loss of positiveness[34]. Let us investigate a little bit more on this behavior: figure 9 presents the same results as figure 8 ($P = 6$ and $N_{MC} = 3 \times 10^7$) but with $P = 6$ and only $N_{MC} = 3 \times 10^3$ MC particles, i.e. $1000\times$ less MC particles. With such low number of MC particles, the results are much noisier in figure 9 than in figure 8, independently of the quantity of interest (mean, variance and realisations). But in the vicinity of $x = 10$, some realisations of $x, X \to \int u(x,\omega,X)\,\mathrm{d}\omega$ are below zero.

Despite this local loss of positiveness of $\int u^P(x,\omega,X)\,\mathrm{d}\omega$, quantity $U^P_{\text{new}}(X) = \iint u^P(x,\omega,X)\,\mathrm{d}\omega\,\mathrm{d}x$ remains positive (and so remains $k^P_{\text{eff}}(X)$) so that the code does not crash or encounter any instabilities, any numerical difficulties. It did not trigger any robustness difficulties as testifies figure 9 (top-right) where the same number of iterations have been performed. Of course, we have no guaranty that this behaviour (no robustness problems) will be maintained if more spatial locations and realisations lose positiveness. But, once again, we do not encounter any problem in the several benchmarks of this paper. Figure 9 (top-right) compares the mean and the variance of a fine ($N_{MC} = 3 \times 10^7$) and a coarse ($N_{MC} = 3 \times 10^3$) k_{eff} approximation through 200 power iterations. Figure 9 (top-right and bottom) also allows highlighting one important property of the MC-gPC solver: a coarse MC approximation induces a higher variance in term of both k_{eff} and u. For the coarse approximation, the numerical noise is more important than the fluctuations induced by the uncertainties. This is only once the numerical resolution is fine enough that MC-gPC is able to

[33] The notation $|\Omega|$ denotes the volume of Ω.

[34] We insist we here have a discussion based on the benchmark of section 5.3 but the same kind of behavior can be observed on all the other test-cases of this paper.

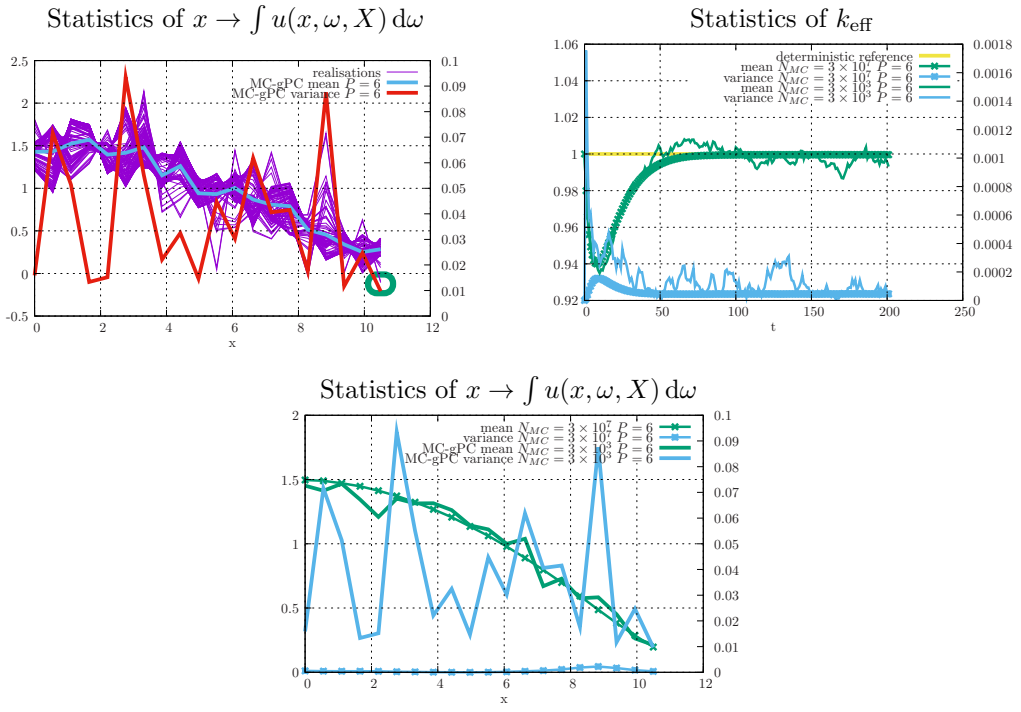

Figure 9: UD2O-H2O(1)-1-0-SL problem [52] with uncertain interface between UD2O and H2O with a coarse MC discretisation $N_{MC} = 3 \times 10^3$ (top-left). Figure (top-right and bottom) compares the fine resolution of figure 8 to the coarse one of this section.

extract the 'physical' variance from the one related to the numerical error. In a sense, this loss of positiveness is a good indicator of a coarse approximation and, as tackled earlier, relevant population control algorithm may be designed specifically for MC-gPC in order to improve the accuracy of the gPC expansion of u (for example where it goes below zero).

6. Conclusion

In this paper, we generalise the MC-gPC solver (presented in [1] for the resolution of the uncertain linear Boltzmann equation) to the resolution of uncertain eigenvalue/eigenvector problems (k_{eff} computations). We build hierachical generalised Polynomial Chaos based reduced model for eigenvalue/eigenvector problems, use the MC-gPC strategy described in [1] and complete the solver by developping a gPC-adapted *power iteration* method. Care is taken to describe precisely how an existing MC eigenvalue/eigenvector solver can be modified to apply MC-gPC. The modifications are quite simple and do not induce any change in the HPC parallel strategy developed for the native MC code.

Computational gains are put forward on well-known criticality benchmarks made uncertain in low to moderate stochastic dimension (i.e. for a low to moderate number of uncertain parameters). The new solver also presents some interesting properties with respect to the iterating process needed

in order to compute the eigenvalue/eigenvector: in a non-intrusive context, using several runs of an MC black box code, propagating uncertainties implies checking the convergence of the power iteration for every runs. The strategy suggested in this paper implies only one run of an MC-gPC eigenvalue/eigenvector solver and the user only has to check the convergence of one run. Furthermore, it presents some advantages when considering geometrical uncertainties: in a non-intrusive context, as many meshes as runs must be made (which can be tedious in complex configurations). With MC-gPC, only one mesh is needed. To sum-up, the MC-gPC eigenvalue/eigenvector solver can efficiently tackle sensitivity analysis, uncertainty propagation in uncertain geometries and generalises perturbative methods. In such eigenvalue/eigenvector context, MC-gPC does present an inconvenience: positivity of the gPC approximation of the k_{eff} eigenvalue or of the eigenvector can be lost for coarse MC discretisations.

Amongst the perspectives of this work we can count on improvements of the positiveness-preserving strategy hinted at beforehand: gPC is known to have this flaw and many other numerical strategies are at hand in the furnished literature [50, 43, 42, 45, 48]. Furthermore, this paper mainly focuses on the description of the gPC adapted power iteration method and on some benchmarks. The fast (spectral) convergence with respect to P of the $P-$truncated gPC based reduced models have been numerically recovered on the k_{eff} benchmarks of section 5. A theoretical proof such as the one of [2] for the linear Boltzmann equation could be at hand and help understanding and anticipating in which cases the reduced models are efficient or not for eigenvalue/eigenvector computations. In this paper, we presented the MC-gPC modifications one has to perform for *one particular power iteration method*: some others may be better suited. The same applies regarding the population control algorithm or the choice of the time step or the stopping criterion of the uncertain power iteration method. Simple strategies were considered in this paper in order to focus on MC-gPC, the modifications needed for the power iteration method and for the sake of reproducibility of the numerical results. More elaborate ones (MC-gPC combined to a better power iteration strategy or a better population control algorithm adapted to the uncertain context) could strongly benefit and improve the MC-gPC eigenvalue/eigenvector solver.

Acknowledgments

References

[1] G. Poëtte, A gPC-intrusive Monte Carlo scheme for the resolution of the uncertain linear Boltzmann equation, Journal of Computational Physics 385 (2019) 135 – 162. doi:https://doi.org/10.1016/j.jcp.2019.01.052.
URL http://www.sciencedirect.com/science/article/pii/S002199911930110X

[2] G. Poëtte, Spectral convergence of the generalized Polynomial Chaos reduced model obtained from the uncertain linear Boltzmann equation, Preprint submitted to Mathematics and Computers in Simulation (2019).

[3] B. Perthame, Transport Equations in Biology, Birkhauser Verlag, Basel Boston Berlin, 2000.

[4] L. Pareschi, M. Zanella, Monte Carlo stochastic Galerkin methods for the Boltzmann equation with uncertainties: space-homogeneous case (03 2020). doi:10.13140/RG.2.2.28177.17760.

[5] L. Pareschi, P. Vellucci, M. Zanella, Kinetic models of collective decision-making in the presence of equality bias, Physica A: Statistical Mechanics and its Applications 467 (2017) 201 – 217.

doi:https://doi.org/10.1016/j.physa.2016.10.003.
URL http://www.sciencedirect.com/science/article/pii/S0378437116306902

[6] J. A. Carrillo, M. Zanella, Monte Carlo gPC methods for diffusive kinetic flocking models with uncertainties Preprint (2019).

[7] B. M. Althouse, E. A. Wenger, J. C. Miller, S. V. Scarpino, A. Allard, L. Hébert-Dufresne, H. Hu, Stochasticity and heterogeneity in the transmission dynamics of sars-cov-2, 2020.

[8] M. Coste-Delclaux, C. DIOP, A. NICOLAS, B. BONIN, Neutronique, E-den, Une monographie de la Direction de l'énergie nucléaire, CEA Saclay; Groupe Moniteur, 2013.
URL https://hal-cea.archives-ouvertes.fr/cea-01152822

[9] F. Golse, G. Allaire, Transport et Diffusion, 2015, polycopié de cours.

[10] J. Spanier, E. M. Gelbard, Monte Carlo Principles and Neutron Transport Problems, Addison-Wesley, 1969.

[11] E. E. Lewis and W. F. Miller Jr., Computational Methods of Neutron Transport, John Wiley and Son New York, 1984.

[12] G. I. Bell, S. Glasstone, Nuclear Reactor Theory, 1970.

[13] R. A. Todor, C. Schwab, Karhunen-Loève approximation of random fields by generalized fast multipole methods, J. Comp. Phys. 217 (1) (2006) 100–122.

[14] M. Meyer, H. Matthies, Efficient model reduction in non-linear dynamics using the Karhunen-Loève expansion and dual-weighted-residual methods, Comp. Meth. Appl. Mech. Eng. Informatikbericht 2003-08, TU Braunschweig, Germany (2004).

[15] J. Mercer, Functions of Positive and Negative Type and their Connection with the Theory of Integral Equations, Philos. Trans. Roy. Soc. 209 (1909).

[16] R. Lebrun, A. Dutfoy, A Generalization of the Nataf Transformation to Distributions with Elliptical Copula, Prob. Eng. Mech. 24,2 (2009) 172–178.

[17] R. Lebrun, A. Dutfoy, An Innovating Analysis of the Nataf Transformation from the Copula viewpoint, Prob. Eng. Mech. 24,3 (2009) 312–320.

[18] B. Iooss, P. Lemaître, A Review on Global Sensitivity Analysis Methods, Dellino, Gabriella and Meloni, Carlo, Springer US, Boston, MA, 2015, pp. 101–122. doi:10.1007/978-1-4899-7547-8_5.
URL http://dx.doi.org/10.1007/978-1-4899-7547-8_5

[19] B. Lapeyre, E. Pardoux, R. Sentis, Méthodes de Monte Carlo pour les équations de transport et de diffusion, no. 29 in Mathématiques & Applications, Springer-Verlag, 1998.

[20] E. Brun, S. Chauveau, F. Malvagi, Patmos: A prototype Monte Carlo transport code to test high performance architectures, In Proceedings of International Conference on Mathematics & Computational Methods Applied to Nuclear Science & Engineering, Jeju, Korea, 2017.

[21] E. Brun, F. Damian, C. Diop, E. Dumonteil, F. Hugot, C. Jouanne, Y. Lee, F. Malvagi, A. Mazzolo, O. Petit, J. Trama, T. Visonneau, A. Zoia, Tripoli-4®, cea, edf and areva reference monte carlo code, Annals of Nuclear Energy 82 (2015) 151 – 160, joint International Conference on Supercomputing in Nuclear Applications and Monte Carlo 2013, SNA + MC 2013. Pluri- and Trans-disciplinarity, Towards New Modeling and Numerical Simulation Paradigms. doi:https://doi.org/10.1016/j.anucene.2014.07.053.
URL http://www.sciencedirect.com/science/article/pii/S0306454914003843

[22] T. Goorley, MCNP6.1.1-Beta Release Notes LA-UR-14-24680 (2014).

[23] D. Dureau, G. Poëtte, Hybrid Parallel Programming Models for AMR Neutron Monte Carlo Transport, in: Joint International Conference on Supercomputing in Nuclear Applications + Monte Carlo, no. 04202 in Parallelism and HPC, Monte Carlo, 2013.

[24] L. Pareschi, An introduction to uncertainty quantification for kinetic equations and related problems (2020). arXiv:2004.05072.

[25] R. N. Blomquist, E. M. Gelbard, Alternative Implementations of the Monte Carlo Power Method, Nuclear Science and Engineering 141 (2) (2002) 85–100. arXiv:https://doi.org/10.13182/NSE01-30, doi:10.13182/NSE01-30.
URL https://doi.org/10.13182/NSE01-30

[26] Brezis H., Analyse fonctionnelle, Paris, (1983).

[27] Planchard J., Méthodes mathématiques en neutronique, Vol. Collection de la Direction des Études et Recherches d'EDF, (1995).

[28] G. Poëtte, Contribution to the mathematical and numerical analysis of uncertain systems of conservation laws and of the linear and nonlinear boltzmann equation, Habilitation à diriger des recherches, Université de Bordeaux 1 (Sep. 2019).
URL https://hal.archives-ouvertes.fr/tel-02288678

[29] G. Blatman, Adaptive sparse polynomial chaos expansions for uncertainty propagation and sensitivity analysis, Thèse de doctorat, Université Blaise Pascal - Clermont II (2009).

[30] G. Blatman, B. Sudret, Sparse Polynomial Chaos Expansions and Adaptive Stochastic Finite Elements using a Regression Approach, C. R. Méc. 336 (2008) 518–523. doi:10.1016/j.crme.2008.02.013.

[31] T. Crestaux, Polynômes de Chaos pour la Propagation et la Quantification d'Incertitudes, Tech. rep., CEA (2006).

[32] N. Wiener, The Homogeneous Chaos, Amer. J. Math. 60 (1938) 897–936.

[33] R. Cameron, W. Martin, The Orthogonal Development of Non-Linear Functionals in Series of Fourier-Hermite Functionals, Annals of Math. 48 (1947) 385–392.

[34] Ernst, Oliver G., Mugler, Antje, Starkloff, Hans-Jörg, Ullmann, Elisabeth, On the convergence of generalized polynomial chaos expansions, ESAIM: M2AN 46 (2) (2012) 317–339. doi:10.1051/m2an/2011045.
URL https://doi.org/10.1051/m2an/2011045

[35] B. J. Debusshere, H. N. Najm, P. P. Pébay, O. M. Knio, R. G. Ghanem, O. P. L. Maître, Numerical Challenges in the Use of Polynomial Chaos Representations for Stochastic Processes, J. Sci. Comp. 26 (2004) 698–719.

[36] J. A. S. Witteveen, H. Bijl, Using Polynomial Chaos for Uncertainty Quantification in Problems with Non Linearities, 47th AIAA Aerospace Sciences Meeting and Exhibit AIAA 2006-2066 (2006).

[37] W. Gautschi, Orthogonal polynomials: applications and computation, Vol. 5, Oxford University Press, 1996.

[38] J.-M. Martinez, J. Cahen, A. Millard, D. Lucor, F. Huvelin, J. Ko, N. Poussineau, Modélisation des Incertitudes par Polynômes de Chaos – Étude d'un Écoulement en Milieux Poreux, Tech. Rep. Rapport DM2S/DIR/RT/06-006/A, CEA-CEMRACS (2006).

[39] F. Simon, P. Guillen, P. Sagaut, D. Lucor, A gPC based approach to uncertain transonic aerodynamics, CMAME 199 (2010) 1091–1099.

[40] D. Lucor, J. Meyers, P. Sagaut, Sensitivity Analysis of LES to Subgrid-Scale-Model Parametric Uncertainty using Polynomial Chaos, J. Fluid Mech. 585 (2007) 255–279.

[41] G. Poëtte, B. Després, D. Lucor, Uncertainty Quantification for Systems of Conservation Laws, J. Comp. Phys. 228 (7) (2009) 2443–2467. doi:http://dx.doi.org/10.1016/j.jcp.2008.12.018.

[42] B. Després, G. Poëtte, D. Lucor, Robust Uncertainty Propagation in Systems of Conservation Laws with the Entropy Closure Method, Vol. 92 of Lecture Notes in Computational Science and Engineering, Uncertainty Quantification in Computational Fluid Dynamics, 2013.

[43] J. Kusch, G. W. Alldredge, M. Frank, Maximum-principle-satisfying second-order intrusive poly- nomial moment scheme, arXiv preprint arXiv:1712.06966 (2017).

[44] J. Kusch, M. Frank, Intrusive methods in uncertainty quantification and their connection to kinetic theory, International Journal of Advances in Engineering Sciences and Applied Mathematics (2018) 1–16.

[45] J. Kusch, R. G. McClarren, M. Frank, Filtered stochastic galerkin methods for hyperbolic equations, J. Comput. Phys. (2018).

[46] L. Schlachter, F. Schneider, O. Kolb, Weighted essentially non-oscillatory stochastic galerkin approximation for hyperbolic conservation laws (2019). arXiv:1912.09171.

[47] L. Schlachter, F. Schneider, A hyperbolicity-preserving stochastic galerkin approximation for uncertain hyperbolic systems of equations, arXiv preprint arXiv:1710.03587 (2017).

[48] J. Dürrwächter, T. Kuhn, F. Meyer, L. Schlachter, F. Schneider, A hyperbolicity-preserving discontinuous stochastic galerkin scheme for uncertain hyperbolic systems of equations, Journal of Computational and Applied Mathematics 370 (2020) 112602. doi:10.1016/j.cam.2019.112602. URL `http://dx.doi.org/10.1016/j.cam.2019.112602`

[49] O. P. L. Maître, O. M. Knio, Uncertainty Propagation using Wiener-Haar Expansions, J. Comp. Phys. 197 (2004) 28–57.

[50] X. Wan, G. Karniadakis, Multi-Element generalized Polynomial Chaos for Arbitrary Probability Measures, SIAM J. Sci. Comp. 28(3) (2006) 901–928.

[51] M. Faucher, D. Mancusi, A. Zoia, New kinetic simulation capabilities for tripoli-4®: Methods and applications, Annals of Nuclear Energy 120 (2018) 74 – 88. doi:https://doi.org/10.1016/j.anucene.2018.05.030.
URL http://www.sciencedirect.com/science/article/pii/S0306454918302676

[52] Sood, Avneet, Forster, Arthur, Parsons, Kent, Analytical benchmark test set for criticality code verification, Progress in Nuclear Energy 42 (1) (2003) 55–106.

[53] A. S. o. M. E. ASME V&V 20-2009, Standard for Verification and Validation in Computational Fluid Dynamics and Heat Transfer, ASME (2009).

[54] R. E. Caflisch, Monte carlo and quasi-monte carlo methods, Acta numerica 7 (1998) 1–49.

[55] R. E. Caflisch, W. J. Morokoff, A. B. Owen, Valuation of mortgage backed securities using brownian bridges to reduce effective dimension (1997).

[56] A. Saltelli, Making best use of model evaluations to compute sensitivity indices, Computer Physics Communications 145 (2) (2002) 280 – 297. doi:https://doi.org/10.1016/S0010-4655(02)00280-1.
URL http://www.sciencedirect.com/science/article/pii/S0010465502002801

[57] B. Sudret, Global Sensitivity Analysis using Polynomial Chaos Expansion, Rel. Engrg. Syst. Saf. 93 (2007) 964–979. doi:10.1016/j.ress.2007.04.002.

[58] G. Blatman, B. Sudret, Efficient computation of global sensitivity indices using sparse polynomial chaos expansions, Rel. Eng. Syst. Saf. 95 (2010) 1216–1229.

[59] I. Sobol, Sensitivity estimates for nonlinear mathematical models, Matematicheskoe Modelirovanie 2 (1990) 112–118, in Russian, translated in English in Sobol, I. (1993). Sensitivity analysis for non-linear mathematical models. Mathematical Modeling and Computational Experiment (Engl. Transl.), 1993, 1, 407–414.

Appendix A. Reminder of the material of [1]

The material of this paper strongly relies on some algorithmic results from [1]. In order to focus, in the corpse of the paper, on the originality and the novelty, i.e. the uncertain eigenvalue/eigenvector computations, we recall briefly what is already detailed in [1] in this appendix.

Appendix A.1. Reminder for the deterministic particle tracking algorithm [1]

In this section, we recall how the tracking of the MC particles is made in practice, i.e. how we make sure that an MC particle u_p is a particular solution of (5) (see section 2). This corresponds to an algorithmic description of the semi-analog MC scheme also called *implicit capture* in the literature, see [19, 28, 10, 11]. It is summed up in algorithm 8. The algorithm begins by initialising variables U_{new} for the eigenvalue and $(U_i)_{i\in\{1,\dots,N_{\mathbf{x}}\}}$ for the eigenvector with $N_{\mathbf{x}}$ the number of cells tesselating $\bigcup_{i=1}^{N_{\mathbf{x}}} \mathcal{D}_i = \mathcal{D}$. Those variables will be used (cf. when $\tau > s_p$ in algorithm 8) to tally the MC particles contributions. Now, if the life time of a particle p is non-zero or if p did not go out

of $\mathcal{D}$, several random variables are sampled and are used to modify its fields $(w_p, \mathbf{x}_p, \mathbf{v}_p, s_p)$. Those samplings are quite classical and their relevance has been put forward in several books or papers [19, 28]. In particular, at a collision (i.e. if $\tau < s_p$), the weight w_p is multiplied by the ratio $\frac{\sigma_S}{\sigma_t}$ in which σ_S uses the current k_{eff} value. The expression of σ_S is described in algorithm 9.

trackParticles(list_of_particles, Δt, k_{eff})
Data: list_of_particles, Δt, k_{eff}
Result: an updated number of physical particles U_{new} and list_of_particles
begin
  $U_{\text{new}} \leftarrow 0$
  for $i \in \{1, ..., N_{\mathbf{x}}\}$ **do**
    $U_i \leftarrow 0$
  end
  for $p \in \{1, ..., N_{MC}\}$ **do**
    #MC particle p has fields $s_p, \mathbf{x}_p, \mathbf{v}_p$
    while $s_p > 0$ *and* $w_p > 0$ **do**
      if $\mathbf{x}_p \notin \mathcal{D}$ **then**
        #application of arbitrary boundary conditions
        apply_boundary_conditions$(\mathbf{x}_p, s_p, \mathbf{v}_p)$
      end
      $\tau = -\frac{\ln(\mathcal{U})}{v\sigma_t(\mathbf{x}_p, s_p, \mathbf{v}_p)}$ where $\mathcal{U} \sim \mathcal{U}([0, 1])$.
      if $\tau > s_p$ **then**
        #move the particle p
        $\mathbf{x}_p \mathrel{-}= \mathbf{v}_p s_p$,
        #set the life time of particle p to zero:
        $s_p = 0$
        #tally the contribution of particle p
        $U_i \leftarrow U_i + w_p \mathbf{1}_{\mathcal{D}_i}(\mathbf{x}_p)$
        $U_{\text{new}} \leftarrow U_{\text{new}} + w_p$
      end
      else
        #move the particle p
        $\mathbf{x}_p \mathrel{-}= \mathbf{v}_p \tau$,
        #set the life time of particle p to:
        $s_p \mathrel{-}= \tau$
        #change its weight
        $w_p \times = \dfrac{\sigma_S(\mathbf{x}_p, s_p, \mathbf{v}_p, k_{\text{eff}})}{\sigma_t(\mathbf{x}_p, s_p, \mathbf{v}_p)}$
        Sample the velocity of particle p from $P(\mathbf{x}_p, s_p, \mathbf{v}_p, \mathbf{v}')\,\mathrm{d}\mathbf{v}'$
        $\mathbf{v}_p \sim P(\mathbf{x}_p, s_p, \mathbf{v}_p, \mathbf{v}')\,\mathrm{d}\mathbf{v}'$
      end
    end
  end
end

Algorithm 8: The deterministic *tracking* of MC particles.

$\sigma_S(\mathbf{x}_p, s_p, \mathbf{v}_p, k_{\text{eff}})$
Data: $\mathbf{x}_p, s_p, \mathbf{v}_p, k_{\text{eff}}$
Result: σ_S
begin
$\quad \sigma_S = \sigma_s(\mathbf{x}_p, \mathbf{v}_p) + \nu_f(\mathbf{x}_p, \mathbf{v}_p)\dfrac{\sigma_f(\mathbf{x}_p, \mathbf{v}_p)}{k_{\text{eff}}}$
end

Algorithm 9: Expression of σ_S for the deterministic eigenvalue/eigenvector computations

Appendix A.2. Reminder for the stochastic particle tracking algorithm [1]

In this section, we describe the stochastic counterpart of section Appendix A.1, i.e. the stochastic counterpart of the semi-analog MC scheme (or implicit capture) of algorithm 8. The modifications of algorithm 8 are highlighted in blue in algorithm 10. Algorithm trackUncertainParticles has a vector of gPC coefficients of k_{eff} as input, instead of only a scalar k_{eff} (as in trackParticles). During the initialisation phase, vectors must be set to zero. The first important modification comes from the fact that each uncertain MC particle p must compute its own $k^P_{\text{eff}}(\mathbf{X}_p)$ depending on its own field $\mathbf{X}_p$. In particular, $k^P_{\text{eff}}(\mathbf{X}_p)$ is used in the evaluation of σ_S, see algorithm 11, computed when particle p encounters a collision. Note that each sampling (the interaction time τ, the outer velocity at a collision) depends on the $\mathbf{X}_p$ field of particle p. This is already intensively explained and justified in [1]. Finally, the uncertain MC particle contributions are tallied in the vectors $(U^k_{\text{new}}, (U^k_i)_{i\in\{1,\dots,N_{\mathbf{x}}\}})_{k\in\{0,\dots,P\}}$ conditionnally to having $\tau > s_p$.

trackUncertainParticles(list_of_particles, Δt, $k_{\text{eff}}^0, ..., k_{\text{eff}}^P$)
Data: list_of_particles, Δt, $k_{\text{eff}}^0, ..., k_{\text{eff}}^P$
Result: an updated number of gPC coefficients $(U_{\text{new}}^k)_{k\in\{0,...,P\}}$ and list_of_particles
begin
  for $k \in \{0, ..., P\}$ **do**
    set $U_{\text{new}}^k = 0$
    for $i \in \{1, ..., N_{\mathbf{x}}\}$ **do**
      $U_i^k \leftarrow 0$
    end
  end
  for $p \in \{1, ..., N_{MC}\}$ **do**
    #compute $k_{\text{eff}}(\mathbf{X}_p)$ *seen by MC particle* p *having fields* $s_p, \mathbf{x}_p, \mathbf{v}_p, \mathbf{X}_p$
    $k_{\text{eff}}^P(\mathbf{X}_p) = buildPonctualUncertainValue(\mathbf{X}_p, k_{\text{eff}}^0, ..., k_{\text{eff}}^P)$
    while $s_p > 0$ *and* $w_p > 0$ **do**
      if $\mathbf{x}_p \notin \mathcal{D}$ **then**
        apply_boundary_conditions($\mathbf{x}_p, s_p, \mathbf{v}_p, \mathbf{X}_p$)
      end
      $\tau = -\frac{\ln(\mathcal{U})}{v\sigma_t(\mathbf{x}_p, s_p, \mathbf{v}_p, \mathbf{X}_p)}$ where $\mathcal{U} \sim \mathcal{U}([0, 1])$.
      if $\tau > s_p$ **then**
        $\mathbf{x}_p \mathrel{-}= \mathbf{v}_p s_p$,
        $s_p = 0$
        for $k \in \{0, ..., P\}$ **do**
          $U_{\text{new}}^k \mathrel{+}= w_p \times \phi_k^{\mathbf{X}}(\mathbf{X}_p)$
          $U_i^k \leftarrow U_i^k + w_p \mathbf{1}_{\mathcal{D}_i}(\mathbf{x}_p)\phi_k^{\mathbf{X}}(\mathbf{X}_p)$
        end
      end
      else
        $\mathbf{x}_p \mathrel{-}= \mathbf{v}_p \tau$,
        $s_p \mathrel{-}= \tau$
        $w_p \times = \frac{\sigma_S(\mathbf{x}_p, s_p, \mathbf{v}_p, \mathbf{X}_p, k_{\text{eff}}^P(\mathbf{X}_p))}{\sigma_t(\mathbf{x}_p, s_p, \mathbf{v}_p, \mathbf{X}_p)}$
        Sample the velocity of particle p from $P(\mathbf{x}_p, s_p, \mathbf{v}_p, \mathbf{v}', \mathbf{X}_p)\,\mathrm{d}\mathbf{v}'$
        $\mathbf{v}_p \sim P(\mathbf{x}_p, s_p, \mathbf{v}_p, \mathbf{v}', \mathbf{X}_p)\,\mathrm{d}\mathbf{v}'$
      end
    end
  end
end

Algorithm 10: The stochastic/uncertain *tracking* of MC particles.

$\sigma_S(\mathbf{x}_p, s_p, \mathbf{v}_p, \mathbf{X}_p, k_{\text{eff}})$
Data: $\mathbf{x}_p, s_p, \mathbf{v}_p, \mathbf{X}_p, k_{\text{eff}}$
Result: σ_S
begin
$\quad \sigma_S = \sigma_s(\mathbf{x}_p, \mathbf{v}_p, \mathbf{X}_p) + \nu_f(\mathbf{x}_p, \mathbf{v}_p, \mathbf{X}_p)\dfrac{\sigma_f(\mathbf{x}_p, \mathbf{v}_p, \mathbf{X}_p)}{k_{\text{eff}}}$
end

Algorithm 11: Expression of σ_S for the stochastic eigenvalue/eigenvector computations