ESAIM: PROCEEDINGS AND SURVEYS, September 2014, Vol. 45, p. 178-188
J.-S. Dhersin, Editor

STATISTICAL INFERENCE FOR PARTIAL DIFFERENTIAL EQUATIONS *

Emmanuel Grenier[1], Marc Hoffmann[2], Tony Lelièvre[3], Violaine Louvet[4], Clémentine Prieur[5], Nabil Rachdi[6] and Paul Vigneaux[1]

Abstract. Many physical phenomena are modeled by parametrized PDEs. The poor knowledge on the involved parameters is often one of the numerous sources of uncertainties on these models. Some of these parameters can be estimated, with the use of real world data. The aim of this mini-symposium is to introduce some of the various tools from both statistical and numerical communities to deal with this issue. Parametric and non-parametric approaches are developed in this paper. Some of the estimation procedures require many evaluations of the initial model. Some interpolation tools and some greedy algorithms for model reduction are therefore also presented, in order to reduce time needed for running the model.

Introduction

Many physical phenomena are modeled by parametrized PDEs. The involved parameters are often unknown and have to be estimated. This mini-symposium focuses on this challenging issue. The first two sections are based on statistical estimation tools. Section 1 is interested in transport-fragmentation equations, and the aim is a non parametric estimation of the division rate of a given cell. Section 2 deals with an industrial application in thermal regulation of an aircraft cabin, and one is interested in estimating parameters appearing in boundary conditions of Navier-Stokes equations. As parameter estimation often requires many computations of the underlying model, one is also interested in speeding the computation time. It is the aim of Sections 3 and 4. Section 3 couples a SAEM algorithm with an interpolation approach to speed up the estimation procedure of parameters involved in KPP equations used to model the evolution of a tumor extracted from MRI images. Section 4 presents greedy algorithms dedicated to solve high-dimensional PDEs. Parametric (see Sections 2,3,4) and non-parametric (see Section 1) approaches are presented.

---

* *This paper is based on the MS EsPaEDP proposed during the SMAI 2013 workshop by both GdR Calcul and MASCOT-NUM.*

[1] ENS Lyon, UPMA; INRIA, Project-team NUMED
[2] Paris Dauphine, CEREMADE
[3] Université Paris-Est, ENPC, CERMICS; INRIA, Project-team MICMAC
[4] Université de Lyon, ICJ; INRIA, Project-team NUMED
[5] Université Grenoble Alpes, LJK; INRIA, Project-team MOISE
[6] EADS Innovation Works

Article published online by EDP Sciences and available at http://www.esaim-proc.org or http://dx.doi.org/10.1051/proc/201445018

1. Statistical inference in transport-fragmentation equations

1.1. Context

We consider (simple) particle systems that serve as toy models for the evolution of cells or bacteria: Each particle grows by ingesting a common nutrient. After some time, each particle gives rise to two offsprings by cell division. We structure the model by state variables like size, growth rate and so on. Deterministically, the density of structured state variables evolves according to a transport-fragmentation PDE. Stochastically, the particles evolve according to a PDMP (piecewise deterministic Markov process) that evolves along a branching tree. Growth-fragmentation type equations provide a natural framework for the study of size-structured populations: Let $n(t,x)$ denote the density of cells of size x at time t. The parameter of interest is the division rate $B(x)$. At division, a cell of size x gives birth to two cells of size $x/2$. The growth of the cell size by nutrient uptake is given by a growth rate $g(x) = \tau x$ (for simplicity). The temporal evolution of n is governed by the transport-fragmentation equation

$$\partial_t n(t,x) + \partial_x\big(\tau x n(t,x)\big) + B(x)n(t,x) = 4B(2x)n(t,2x)$$

with $n(t, x = 0) = 0$, $t > 0$ and $n(0,x) = n^{(0)}(x)$, $x \geq 0$. It is obtained by mass conservation law: the LHS term is obtained by density evolution plus growth by nutrient plus division of cells of size x, while the RHS is obtained by division of cells of size $2x$.

1.2. Objectives

Our main goal is to estimate non-parametrically $B(x)$ from genealogical data of a cell population of size N living on a binary tree. We also want to avoid solving an inverse problem as it is the case for alternative approaches (see e.g., [11,12]) for estimating $B(x)$, thanks to richer data set provided by genealogical data (*i.e.* observed along a genealogical tree). Finally, we wish to reconcile the deterministic approach with a rigorous statistical analysis (relaxing the steady-state implicit approximation of deterministic approaches).

Our strategy to reach this goal is: **1)** Construct a stochastic model accounting for the stochastic dependence structure on a tree for which the (mean) empirical measure of N particles solves the fragmentation-transport equation (in a weak sense). **2)** Develop appropriate statistical tools to estimate $B(x)$. **3)** Incorporate the additional difficulty of growth variability: each cell has a stochastic growth rate inherited from its parent.

1.3. Results

We construct a Markov process on a binary tree $(X_t, V_t) \in \big(\bigcup_{k\geq 0}[0,\infty)^k\big)^2$, where X_t denotes the size and V_t the growth rate of living cells at time t, inherited from their parent according to a kernel ρ.

Result 1: We prove in [10] that the function $n(t,\cdot) := \mathbb{E}\big[\sum_{i=1}^{\infty}\delta_{X_i(t),V_i(t)}\big]$ is a (weak)-solution of an extension of the transport-fragmentation equation:

$$\partial_t n(t,x,v) + v\,\partial_x\big(x\,n(t,x,v)\big) + B(x)n(t,x,v) = 4B(2x)\int \rho(v',v)n(t,2x,dv').$$

The initial framework $g(x) = \tau x$ is retrieved as soon as $\rho(v', dv) = \delta_\tau(dv)$.

Result 2: We assume that we are given genealogical data of the form $(\xi_u, \tau_u)_{u\in\mathcal{U}_N}$, where $\mathcal{U}_N$ is a (connected) subset of size N of the binary tree $\mathcal{U} = \cup_{k\geq 0}\{0,1\}^k$. This means that we observe the size ξ_u and the variability τ_u of the cell for each node $u \in \mathcal{U}_N$. This means that during its lifetime, at time t, the cell u has size $\xi_u \exp(\tau_u t)$ and the variability τ_u thus denotes the growth rate attached to each cell and that may vary from one individual

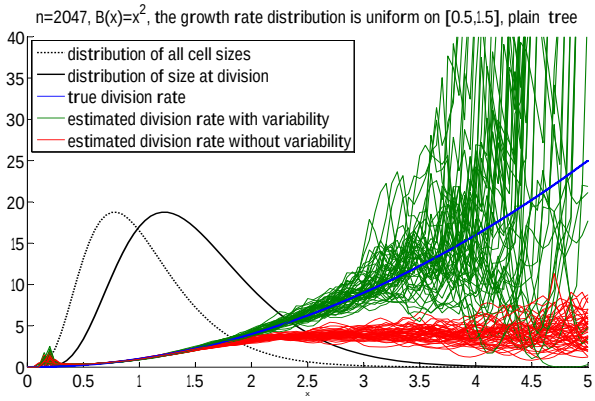


FIGURE 1. Simulated data: for $N = 2047$ we see the quality of $\widehat{B}_n$ over 50 Monte-Carlo simulations. The true B is in solid blue, and the Monte-Carlo reconstructions are in green. The reconstruction is fairly good in the region where the density ν_B (in solid black) is not too small, and it consistently deteriorates beyond $x \approx 3$, where almost no data of size $x \geq 3$ were observed. Note however the dramatic effect of ignoring variability (the Monte-Carlo estimators in red) beyond $x \approx 2.5$.

to another. This is a reasonable assumption that we have been able to implement in practice. We can construct an estimator $(\widehat{B}_N(x), x > 0)$ of the s-regular division rate $B(x)$ s.t.

$$\mathbb{E}\big[\|\widehat{B}_N - B\|^2_{L^2_{\rm loc}}\big]^{1/2} \leq C(\log N)N^{-s/(2s+1)}.$$

Construction of $\widehat{B}$: Let $\nu_B(y)$ denote the (asymptotic or invariant) density distribution of the size of a cell at division. The construction of $\widehat{B}$ is based on the key representation formula proved in [10]

$$B(y) = \frac{y}{2}\frac{\nu_B(y/2)}{\mathbb{E}_{\nu_B}\left[\frac{1}{\tau_{u^-}}\mathbf{1}_{\{\xi_u^- \leq y,\, \xi_u \geq y/2\}}\right]}. \tag{1}$$

Introduce a kernel function $K : [0,\infty) \to \mathbf{R}$, $\int_{[0,\infty)} K(y)dy = 1$. and set $K_h(y) = h^{-1}K\big(h^{-1}y\big)$ for $y \in [0,\infty)$ and $h > 0$. We construct the following estimator based on an empirical regularisation of (1):

$$\widehat{B}_n(y) = \frac{y}{2}\frac{n^{-1}\sum_{u\in\mathcal{U}_n} K_h(\xi_u - y/2)}{n^{-1}\sum_{u\in\mathcal{U}_n}\frac{1}{\tau_{u^-}}\mathbf{1}\{\xi_{u^-} \leq y, \xi_u \geq y/2\} \bigvee \varpi},$$

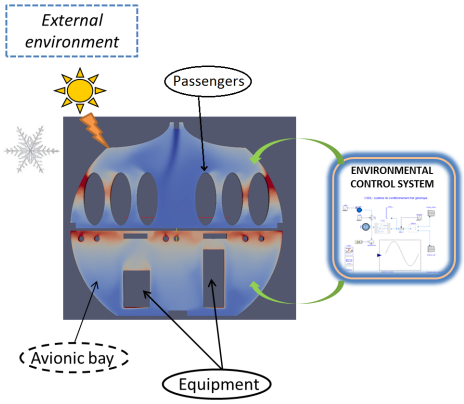
specified by the kernel K, the bandwidth h and the threshold $\varpi > 0$ (that guarantees that $\widehat{B}$ is well defined). This approach avoids the implementation of an inverse problem where only the estimation rate $N^{-s/(2s+3)}$ is achievable ([11], [12]). It also matches both deterministic and stochastic approaches rigorously.

2. Calibration of a PDE system for thermal regulation of an aircraft cabin

2.1. Context

Thermal energy management onboard modern commercial aircraft has become an important challenge for aircraft manufacturers so as to propose competitive solutions to new markets demands. Modern aircraft use more and more new and highly dissipative heat sources (electrical packs, power electrics, etc.) and thus the integration of such equipment needs a careful understanding of the thermal behaviour of this new environment. Therefore, new requirements have to be satisfied in order to improve passenger thermal comfort and to ensure the thermal control of equipment in the avionic bay.

This work aims at presenting the thermal regulation problem inside a commercial aircraft and some scientific challenges inherent to this study.



The thermal exchanges in the cabin and bay of an aircraft are modelled thanks to Navier-Stokes equations that are implemented into a software. The resolution of such equations induces very often parameters with unknown exact value. There are two kinds of unknown parameters: first, the ones subjected to lack of knowledge or variability (e.g turbulence rate, equipment temperature, etc.), and second, the parameters that have to be estimated (e.g thermal contact resistance, thermal conductivity, heat dissipation, etc.). The latter parameters involve additional information which is in practice composed of datasets provided from former aircrafts, real experiments, trials, etc.

2.2. **Problem**

To illustrate our purpose, let us consider a simplified thermal exchange modelling given by Navier-Stokes equations :

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \nabla.(\rho\, u) & = & 0 \\ \dfrac{\partial(\rho C_p T)}{\partial t} + \nabla.(u.\rho C_p T) & = & \nabla.(k\nabla T) \\ \dfrac{\partial(\rho u)}{\partial t} + (u.\nabla)u + \nabla p & = & \mu\Delta u + \rho g \\ + \text{ Set of turbulence model} \end{cases} \tag{2}$$

with boundary conditions : $\begin{cases} u(M) = u_0(M) \quad \text{for} \quad M = \{x_1, \cdots, x_M\} \subset \Gamma \quad \text{(boundary)} \\ \int k\nabla T \cdot \mathbf{n} = \phi \\ \text{turbulence model RANS}(\tau) \ + \text{ specific conditions} \end{cases}$

where ρ= air density, u=air speed, k= air conductivity, T=temperature, μ=viscosity , τ=turb. rate.

In this application, we consider that the turbulence rate τ and the air speed condition $u_0(M)$ are *uncertain* (for simplicity, for $x \in M$ write $u_0(x) = u_0^*(x) + \epsilon(x)$ where $u_0^*(x)$ is the deterministic part of the boudary condition, and $\epsilon(x)$ is a random variable which models the uncertainties). Then, the heat dissipation coefficient ϕ should be estimated. The observable of interest we consider is the temperature around the equipment given from a post-processing after resolution of the system above. Thus, a simulated temperature can be seen as a real-valued function $(\mathbf{x}, \theta) \mapsto h(\mathbf{x}, \theta)$, where $\mathbf{x} = (\tau, \varepsilon)$ with $\varepsilon = (\epsilon(x_1), \cdots, \epsilon(x_M))$, and $\theta = \phi$.
As the variables $\mathbf{x} = (\tau, \varepsilon)$ are uncertain, we can choose to model this uncertainty by a random vector $\mathbf{X} = (\tau, \varepsilon)$ where τ and ε are seen as random variables drawn from some given distributions.
Besides this modelling, in order to characterize the parameter $\theta = \phi$, one needs a database of temperatures of the equipment which can be obtained from sensors placed on it. Let us denote such database by $Y_1, ..., Y_n$.

Finally, the estimation problem consists in estimating $\theta = \phi$ from the (stochastic) simulation model $(\mathbf{X}, \theta) \mapsto h(\mathbf{X}, \theta)$.

2.3. **Estimation method**

The method we present is taken from N. Rachdi et al. [21]. The principle consists in estimating a parameter $\theta \in \Theta \subset \mathbb{R}^k$ which minimizes "a distance" between the *empirical distribution* of the Y_i's (measurements) and the *simulated distribution* of the random variable $h(\mathbf{X}, \theta)$ based on a sample $h(\mathbf{X}_1, \theta), ..., h(\mathbf{X}_m, \theta)$ provided from numerical simulations, where $\mathbf{X}_1, ..., \mathbf{X}_m$ are m simulations of the random variable $\mathbf{X} \in (\mathcal{X}, \mathbb{P}_{\mathbf{X}})$. Indeed, assume that the random simulation outputs $\{h(\mathbf{X}, \theta),\, \theta \in \Theta\}$ induce a (Lebesgue) density family $\{f_\theta,\, \theta \in \Theta\}$ where f_θ is the density of $h(\mathbf{X}, \theta)$. Hence, a maximum-likelihood based method would provide the following estimator

$$\widehat{\theta}_n = \underset{\theta \in \Theta}{\operatorname{argmin}} - \sum_{i=1}^{n} \log\left(f_\theta(Y_i)\right) . \tag{3}$$

But, in our framework we do not know explicitly the density functions f_θ as it is the result of complex simulations. Unlike classical maximum-likelihood methods, we do not form a parametric "density model" for the measurements (Gaussian, Beta, etc.) but this density model is provided from simulations of the random variable $h(\mathbf{X}, \theta)$. In this case, f_θ in (3) would be the density of $h(\mathbf{X}, \theta)$ which does not have necessarily an analytical form. We then propose to replace f_θ by a kernel estimator f_θ^m (among others) $f_\theta^m = \dfrac{1}{m} \displaystyle\sum_{j=1}^{m} K_b(Y_i - h(\mathbf{X}_j, \theta))$ where for instance $K_b(y) = \frac{1}{\sqrt{2\,\pi}\,b} e^{-y^2/2b^2}$. Then, replacing f_θ by f_θ^m in (3) provides the computable estimator

$$\widehat{\theta}_{n,m} = \underset{\theta \in \Theta}{\operatorname{argmin}} - \sum_{i=1}^{n} \log\left(\sum_{j=1}^{m} K_b(Y_i - h(\mathbf{X}_j, \theta))\right) . \tag{4}$$

Under mild conditions, Theorem 3.1 in [21] proves the consistency in a general case when considering other contrast functions than log, and Theorem 6.2 in [20] shows the consistency in the special case of the log-contrast function. In such estimation procedure, we need a lot of system runs providing the desired observable h which can be CPU time expensive. To avoid this difficulty, surrogate model techniques may be considered, which aim at replacing the costly model h by a mathematical approximation $\widetilde{h}$ in (4), very cheap to evaluate.

3. Coupling with SAEM algorithm: population parametrization for reaction diffusion equations

3.1. **Context**

A crucial step in the validation of a model is of course to compare it with real world data. This is usually done by using nonlinear regression techniques in the case of individual data, or by statistical approaches (for instance using a SAEM algorithm [14], [7]), which is a maximum likelihood estimation method, in the case of population data. In both cases, this requires a large number of evaluations of the model, for a large number of different sets of parameters. The evaluation time for a single set of parameters may be very long, in particular when partial differential equations are involved (it may go up to a few minutes, or a few hours, or even days). In this case, nonlinear regression algorithms or population approaches can not be done within a reasonable time. It is therefore crucial to find methods to accelerate them.

The application we have in mind is the study of the evolution of the volume of a tumour extracted from MRI images. In that case, the PDE model can be the classical KPP equation, and the tumour volume is the integral of the tumoral concentration.

3.2. **SAEM coupled with model precomputation**

Our strategy is to speed up the computation associated to the evaluation of the scalar time series associated to the solution of the full PDE. We couple a SAEM algorithm with evaluation of the model through interpolation on a precomputed mesh of the parameters domain.

The idea is the following: to compute quickly a function, we interpolate it from precomputed values, on a grid. The main issue is to construct a grid in an efficient way:
- Interpolation should be easy on the mesh. Here we choose a mesh composed of cubes (tree of cubes) to ensure construction simplicity and high interpolation speed
- Mesh should be refined in areas where the function changes rapidly (speed of variation may be measured in various ways, see below).

Let us describe the algorithm in dimension N. We consider J fixed probabilities $0 < q_j < 1$ with $\sum_{j=1}^{J} q_j = 1$ and J positive functions $\psi_j(x)$ (required precisions, as a simple example, take $\psi_j(x) = 1$ for every x). We start with a cube (or more precisely hyper-rectangle) $C_{init} = \Pi_{i=1}^{N}[x_{min,i}, x_{max,i}]$ to prescribe the area of search. The algorithm is iterative. At step n, we have $1 + 2^N n$ cubes C_i with $1 \leq i \leq 1 + 2^N n$, organized in a tree. To each cube we attach J different weights ω_i^j (where $1 \leq j \leq J$, see below for examples of weights), and the 2^N values on its 2^N summits.
- First we choose j between 1 and J with probability q_j.
- Then we choose, amongst the leaves of the tree, the smallest index i such that $\omega_i^j \ / \ \sup_{x \in C_i} \psi_j(x)$ is maximum.
- We then split the cube C_i in 2^N small cubes of equal sizes, which become 2^N new leaves of our tree, the original C_i becoming a node. To each new cube we attach J weights ω_i^j.

Then we iterate the procedure at convenience. We stop the algorithm when a criterion is satisfied or after a fixed number of iterations. We then have a decomposition of the initial cube in a finite number of cubes, organized in a tree (each node having exactly 2^N leaves), with the values of f on each summit. It is interesting to notice that this approach can be easily parallelized to ensure an optimal use of the processors.

If we want to evaluate f at some point x, as during a SAEM computation, we first look for the cube C_i in which x lies, and then approximate f by the interpolation f_{inter} of the values on the summits of the cube C_i. Note that this procedure is very fast, since, by construction, the cubes form a tree, each node having 2^N nodes. The identification of the cube in which x lies is simply a walk on this tree. At each node we simply have to compare the coordinates of x with the centre of the "node" cube, which immediately gives in which "son" x lies. The interpolation procedure (approximation of $f(x)$ knowing the values of f on the summits of the cube) is also classical and rapid (linear in the dimension N).

3.3. **Application: parametrization of a KPP model**

We want to illustrate the previous methodology in the context of the estimation of the parameters associated to the so called KPP equation:

$$\partial_t u - \nabla.(D\nabla u) = Ru(1-u), \tag{5}$$

where $u(x)$ is the unknown concentration (assumed to be initially a compact supported function, for instance), D the diffusion coefficient and R the reaction rate. These equations are posed in a domain Δ with Neumann boundary conditions. Note that the geometry of the domain Δ can be rather complex (e.g., when u is the density of tumor cells in the brain). Initially the support of u is very small and located at some point $x_0 \in \Delta$. Therefore we may assume that

$$u(T_0, x) = \alpha 1_{|x-x_0| \leq \varepsilon}, \tag{6}$$

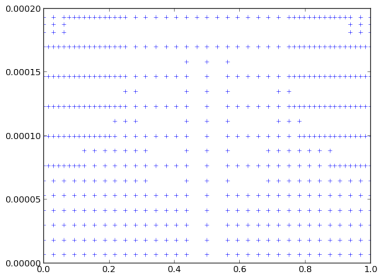


FIGURE 2. An example of an inhomogeneous mesh of the space of parameters (with 500 points). The two parameters are $w = D/R$ and x_0.

for some time T_0 (in the past).

We generate a virtual population of solutions of the KPP equation, assuming Gaussian distributions on its parameters, and adding noise. We then try to recover the distributions of the parameters by a SAEM approach (using Monolix software [22]). For this we first precompute solutions of the KPP equation on a regular or non regular mesh (see Figure 2), and then run SAEM algorithm using interpolations of the precomputed values of KPP equation (instead of the genuine KPP).

As illustrated in table 1, the results are of good quality. Adding some noise deteriorates the accuracy but the results are reasonable for practical applications.

	Theor	E1		E2		E3	
			error		error		error
R	0.0245	0.0237	-3.3%	0.0234	-4.5%	0.0231	-5.7%
D	$8.64e^{-7}$	$8.67e^{-7}$	0.3%	$8.79e^{-7}$	1.7%	$9.62e^{-7}$	11%
x_0	0.415	0.399	-3.9%	0.393	-5.3%	0.37	-11%
ω_R	0.201	0.196	-2.5%	0.263	31%	0.253	26%
ω_D	0.205	0.188	-8.3%	0.247	20%	0.395	93%
ω_{x_0}	0.254	0.244	-3.9%	0.241	-5%	0.616	143%

TABLE 1. Results (from Monolix) and errors for the mean parameters of the population. Column E1 refers to a population without noise (see text). Column E2 (resp. E3) refers to a population with a 5% (resp. 10%) noise. Test with homogeneous grid.

The main interest of this methodology is to tackle problem of parameters identification in complex PDE systems. The computational cost of the whole algorithm that mean generation of the mesh and SAEM computation, can be divided in two distinct parts: an offline time corresponding to the computation of the mesh, which can be done once and for all, and an online time corresponding to the estimation of the parameters for a given population.

In the previous example, the gain is of order 725 for the homogeneous grid and 1200 for the heterogeneous grid compared to a full computation with a whole resolution of the PDE during the SAEM algorithm.

3.4. **Conclusions**

In this work we present a new method combining SAEM algorithm and a precomputation step. This method could be helpful to reduce the overall computation time when the model is very long to compute, for instance when the model is based on partial differential equations.

In a future work we intend to study in details the method which performs simultaneously the precomputation of the parameter space and the SAEM algorithm.

4. Greedy algorithms and model reduction

In this section, we will briefly present a general method to approximate high-dimensional functions. This technique can be used in particular to solve high-dimensional partial differential equations. This section is related to a series of recent works [4–6, 16]. We also refer to the contribution of Virginie Ehrlacher to this volume for an application of this technique to eigenvalue problems.

4.1. **An introductory example**

To fix the idea, let us consider the parametric problem: find $u(\theta, x)$ a real valued function solution to, for all $\theta \in \mathcal{T}$,

$$\begin{cases} -\mathrm{div}_x(a(\theta,x)\nabla_x u(\theta,x)) = f(\theta,x) \quad \forall x \in \mathcal{X}, \\ u(\theta,x) = 0 \quad \forall x \in \partial\mathcal{X}. \end{cases} \tag{7}$$

Here, x varies in subdomain $\mathcal{X}$ of $\mathbb{R}^d$, θ is a parameter which lives in $\mathcal{T}$ a subset of $\mathbb{R}^p$. We assume in the following that $(\theta, x) \mapsto a(\theta, x)$ and $(\theta, x) \mapsto f(\theta, x)$ are two real valued functions such that for almost all values of the parameter $\theta \in \mathcal{T}$, the problem (7) is well posed. For example, f is in $L^2(\mathcal{T} \times \mathcal{X})$, a is in $L^\infty(\mathcal{T} \times \mathcal{X})$ and is bounded from below by a positive constant, so that there exists a unique solution $u \in L^2(\mathcal{T}, H^2(\mathcal{X}) \cap H^1_0(\mathcal{X}))$. This is the setting we will consider in the following. The functional spaces $H^1_0(\mathcal{X})$ and $H^2(\mathcal{X})$ are the classical Sobolev spaces: $H^1_0(\mathcal{X}) = \{v : \mathcal{X} \mapsto \mathbb{R},\ v \in L^2(\mathcal{X}),\ |\nabla_x v| \in L^2(\mathcal{X}) \text{ and } v = 0 \text{ on } \partial\mathcal{X}\}$ and $H^2(\mathcal{X}) = \{v : \mathcal{X} \mapsto \mathbb{R},\ v \in L^2(\mathcal{X}),\ |\nabla_x v| \in L^2(\mathcal{X}) \text{ and } |\nabla^2_{x,x} v| \in L^2(\mathcal{X})\}$.

A parameter estimation problem typically writes as follows: given some observations on the function $u(\theta, x)$, how to estimate the values of $\theta \in \mathcal{T}$? Many approaches have been proposed to solve this inverse problem, and it is not the aim of this section to discuss them. We rather would like to explain a method to approximate the high-dimensional function $(\theta, x) \mapsto u(\theta, x)$. This approximation can then be used to solve the inverse problem, for example to provide a first guess to a deterministic optimization approach or to build a variance reduction technique in a Bayesian technique. Such an approximation is sometimes called a response surface, or a reduced order model.

The difficulty is of course that the functions u depends on the variable (θ, x) with dimension $d+p$. Standard approximation techniques based for example on tensorization of one dimensional grids lead to a huge number of degrees of freedom, since the complexity is exponential in the dimension. This is the so-called curse of dimensionality. Various approaches have been proposed to tackle this difficulty such as sparse grids techniques base [3, 24] or reduced bases methods [17, 19]. We here focus on an algorithm introduced by Ladevèze [15], Ammar [1] and Nouy [18] that we call below the greedy algorithm. This algorithm is also sometimes called the Proper Generalized Decomposition.

4.2. **The greedy algorithm**

Let us now present the greedy algorithm we are interested in. The bottom line is to approximate the function $u(\theta, x)$ as a sum of tensor products:

$$u(\theta, x) = \sum_{k \geq 1} r_k(\theta) s_k(x)$$

and to compute each of the terms in this sum iteratively, as the best next tensor product approximation. Depending on the problem under consideration, this best approximation is defined in various ways. This algorithm is greedy in the sense that the terms are computed iteratively and once one of them is computed, it is not modified in the following iterations. For simplicity, we consider the tensor product of only two functions ($r_k(\theta)$ and $s_k(x)$) but the algorithm equally applies to a tensor product of more than two functions. For example, if the parametric space is high-dimensional (namely if p is large), one could think of using a decomposition of the form $u(\theta, x) = \sum_{k \geq 1} r^1_k(\theta_1) \dots r^p_k(\theta_p) s_k(x)$.

In the specific example (7) above, the algorithm writes as follows: iterate on $K \geq 0$

$$(r_{K+1}, s_{K+1}) \in \underset{r \in L^2(\mathcal{T}), s \in H_0^1(\mathcal{X})}{\operatorname{argmin}} \mathcal{E}\left(\sum_{k=1}^{K} r_k(\theta) s_k(x) + r(\theta) s(x)\right) \tag{8}$$

where

$$\mathcal{E}(v) = \frac{1}{2} \int_{\mathcal{T}\times\mathcal{X}} a(\theta, x) |\nabla_x v|^2 \, d\theta \, dx - \int_{\mathcal{T}\times\mathcal{X}} f(\theta, x) v(\theta, x) \, d\theta \, dx$$

is the energy functional associated to (7), defined for $v \in L^2(\mathcal{T}, H_0^1(\mathcal{X}))$. The energy functional $\mathcal{E}$ has a unique minimum, which is characterized by the Euler equations (7). The idea underlying the algorithm (8) is that at each iteration, the best tensor product minimizing $\mathcal{E}$ is chosen.

In practice, to solve (8), one actually considers the Euler Lagrange equations associated to (8), which are: iterate on $K \geq 0$, find $r_{K+1} \in L^2(\mathcal{T})$ and $s_{K+1} \in H_0^1(\mathcal{X})$ such that, for all $\delta r \in L^2(\mathcal{T})$ and $\delta s \in H_0^1(\mathcal{X})$

$$\begin{aligned} &\int_{\mathcal{T}\times\mathcal{X}} a(\theta, x) \nabla_x (u_K(\theta, x) + r_{K+1}(\theta) s_{K+1}(x)) \cdot \nabla_x (r_{K+1}(\theta) \delta s(x) + \delta r(\theta) s_{K+1}(x)) \, d\theta \, dx \\ &= \int_{\mathcal{T}\times\mathcal{X}} f(\theta, x) (r_{K+1}(\theta) \delta s(x) + \delta r(\theta) s_{K+1}(x)) \, d\theta \, dx \end{aligned} \tag{9}$$

where, for the ease of notation, we introduced

$$u_K(\theta, x) = \sum_{k=1}^{K} r_k(\theta) s_k(x). \tag{10}$$

The problem (9) is the weak form of the problem:

$$\begin{cases} -\operatorname{div}_x \left(\left(\int_{\mathcal{T}} a(\theta, x) (r_{K+1}(\theta))^2 \, d\theta \right) \nabla_x s_{K+1}(x) \right) = \int_{\mathcal{T}} r_{K+1}(\theta) \left(f(\theta, x) + \operatorname{div_x} \left(a(\theta, x) \nabla_x u_K(\theta, x) \right) \right) \, d\theta, \\ r_{K+1}(\theta) \int_{\mathcal{X}} a(\theta, x) |\nabla_x s_{K+1}|^2 \, dx = \int_{\mathcal{X}} \left(f(\theta, x) + \operatorname{div}_x (a(\theta, x) \nabla_x u_K(\theta, x)) \right) s_{K+1}(x) \, dx, \end{cases} \tag{11}$$

where the first equation is an elliptic problem on s_{K+1} (for a fixed function r_{K+1}) with homogeneous Dirichlet boundary conditions, and the second equation gives r_{K+1} (for a fixed function s_{K+1}). Two remarks are in order. First, it is obvious from the formulation (11) that the problem defining the couple (r_{K+1}, s_{K+1}) is *nonlinear*: starting from the linear problem (7), we end up with the nonlinear problem (11). This is because the space of tensor products is not a linear space. Second, if we assume that the data a and f admit a separated representation of the form $a(\theta, x) = \sum_{k\geq 1} r_k^a(\theta) s_k^a(x)$ and $f(\theta, x) = \sum_{k\geq 1} r_k^f(\theta) s_k^f(x)$, then, all the integrals involved in (11) are either integrals over $\mathcal{T}$ or over $\mathcal{X}$, using the Fubini's theorem: there is no integral over the product space $\mathcal{T} \times \mathcal{X}$. In practice, (11) is typically solved by a fixed point algorithm.

Notice that compared to the original problem which was with complexity N^{d+p} (if N denotes the number of degrees of freedom per dimension), the new formulation is a sequence of problems with a much smaller complexity, namely $N^d + N^p$. This comes at a price: the nonlinearity of (11).

4.3. **Convergence**

The algorithm (8) is at the interface between two approximation techniques:

(i) the techniques developed in order to get the best rank n approximations of tensors, using appropriate *formats* and associated approximation algorithms (which are not necessarily greedy algorithms), see [9, 13];

(ii) the greedy algorithms developed in the field of nonlinear approximation, to approximate a function as a sum of elements of a *dictionary* (which is not necessarily the set of tensor products), see [23].

In the algorithm (8), we both use tensor products to approximate the solution (as in item (i) above) and a greedy technique to compute the terms of the sum (as in item (ii) above).

The convergence of the algorithm can actually be deduced from general results on the convergence of greedy algorithms [8, 16].

Theorem 1. *Let us consider the algorithm* (8) *and the function* u_K *defined by* (10)*. The following convergence result holds:* $\lim_{K\to\infty} \|u_K - u\|_{L^2(\mathcal{T},H_0^1(\mathcal{X}))} = 0$*. Moreover, if* $u \in \mathcal{L}^1$ *where the Banach space* $\mathcal{L}^1 \subset L^2(\mathcal{T}, H_0^1(\mathcal{X}))$ *is defined as the set of functions with finite* projective norms

$$\mathcal{L}^1 = \left\{ u(\theta,x) = \sum_{k\geq 0} c_k r_k(\theta) s_k(x),\ \textit{s.t. } r_k \in L^2(\mathcal{T}), s_k \in H_0^1(\mathcal{X})), \|r_k(\theta)s_k(x)\|_{L^2(\mathcal{T},H_0^1(\mathcal{X}))} = 1 \textit{ and } \sum_{k\geq 0} |c_k| < \infty \right\},$$

then there exists $C > 0$ *(which depends on* u*) such that, for all positive* K*,* $\|u_K - u\|_{L^2(\mathcal{T},H_0^1(\mathcal{X}))} \leq CK^{-1/6}$.

The convergence rate can be improved to $-1/2$ using an orthogonal version of the algorithm. The theorem also holds for tensor products of more than two functions.

This result essentially tells us that the algorithm is safe: it is converging. On the other hand, the convergence rate is rather slow. In practice, one typically observes that the convergence is exponential for small values of K, and then slows down.

4.4. **Extensions and open questions**

The prototypical example (7) enjoys two specific properties: it is *linear* and *symmetric*. In [4], we were able to generalize the convergence result to nonlinear problems, which are still defined as the minimum of some functional. In this case however, we have no convergence rates. In [6], we investigated various techniques to generalize the approach to linear but non-symmetric problems, which are thus not simply associated to an energy minimization problem: there is up to now no satisfactory technique to treat non-symmetric problems. We have also on-going works on parametric eigenvalue problems.

Generally speaking, the main question which seems difficult to attack is the following: if the solution to the original problem admits a separated representation (sum of tensor product functions) with a small number of terms, will the greedy algorithms be able to approximate efficiently this function ? There has been very encouraging results in that direction for some greedy algorithms recently [2] but it is unclear if they can be extended to our setting.

References

[1] A. Ammar, B. Mokdad, F. Chinesta, and R. Keunings. A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. *J. Non-Newtonian Fluid Mech.*, 139:153–176, 2006.

[2] P. Binev, A. Cohen, R. Dahmen, W.and DeVore, G. Petrova, and P. Wojtaszczyk. Convergence rates for greedy algorithms in the reduced basis method. *SIAM J. Math. Anal.*, 43:1457–1472, 2011.

[3] H.-J. Bungartz and M. Griebel. Sparse grids. *Acta Numer.*, 13:147–269, 2004.

[4] E. Cancès, V. Ehrlacher, and T. Lelièvre. Convergence of a greedy algorithm for high-dimensional convex nonlinear problems. *Math. Models and Methods in Applied Sciences*, 21(12):2433–2467, 2011.

[5] E. Cancès, V. Ehrlacher, and T. Lelièvre. Greedy algorithms for high-dimensional eigenvalue problems, 2012. `http://hal.archives-ouvertes.fr/hal-00809855`.

[6] E. Cancès, V. Ehrlacher, and T. Lelièvre. Greedy algorithms for high-dimensional non-symmetric linear problems, 2012. `http://hal.archives-ouvertes.fr/hal-00745611`.

[7] B. Delyon, M. Lavielle, and E. Moulines. Convergence of a stochastic approximation version of the EM algorithm. *The Annals of Statistics*, 27(1):94–128, 1999.

[8] R.A. DeVore and V.N. Temlyakov. Some remarks on greedy algorithms. *Adv. Comput. Math.*, 5:173–187, 1996.

[9] S.V. Dolgov, B.N. Khoromskij, and I. Oseledets. Fast solution of multi-dimensional parabolic problems in the TT/QTT formats with initial application to the fokker-planck equation. *SIAM J. Sci. Comp.*, 34(6):3016–3038, 2012.
[10] M. Doumic, M. Hoffmann, N. Krell, and L. Robert. Statistical estimation of a growth-fragmentation model observed on a genealogical tree. *arXiv:1210.3240*, 2013.
[11] M. Doumic, M. Hoffmann, P. Reynaud-Bouret, and V. Rivoirard. Nonparametric estimation of the division rate of a size-structured population. *SIAM Journal on Numerical Analysis*, 50:925–950, 2012.
[12] M. Doumic, B. Perthame, and J. Zubelli. Numerical solution of an inverse problem in size-structured population dynamics. *Inverse Problems*, 25:25pp, 2009.
[13] W. Hackbusch. *Tensor spaces and numerical tensor calculus*. Springer, 2012.
[14] E. Kuhn and M. Lavielle. Maximum likelihood estimation in nonlinear mixed effects models. *Computational Statistics and Data Analysis*, 49(4):1020–1038, 2005.
[15] P. Ladevèze. *Nonlinear computational structural mechanics: new approaches and non-incremental methods of calculation.* Springer, 1999.
[16] C. Le Bris, T. Lelièvre, and Y. Maday. Results and questions on a nonlinear approximation approach for solving high-dimensional partial differential equations. *Constructive Approximation*, 30(3):621–651, 2009.
[17] L. Machiels, Y. Maday, and A.T. Patera. Output bounds for reduced-order approximations of elliptic partial differential equations. *Comput. Methods Appl. Mech. Engrg.*, 190(26-27):3413–3426, 2001.
[18] A. Nouy. A generalized spectral decomposition technique to solve a class of linear stochastic partial differential equations. *Comput. Methods Appl. Mech. Engrg.*, 196:4521–4537, 2007.
[19] C. Prud'homme, D. Rovas, K. Veroy, Y. Maday, A.T. Patera, and G. Turinici. Reliable real-time solution of parametrized partial differential equations: Reduced-basis output bounds methods. *Journal of Fluids Engineering*, 124(1):70–80, 2002.
[20] N. Rachdi, J.-C. Fort, and T. Klein. Stochastic inverse problem with noisy simulator—application to aeronautical model. *Ann. Fac. Sci. Toulouse Math. (6)*, 21(3):593–622, 2012.
[21] N. Rachdi, J.-C. Fort, and T. Klein. Risk bounds for new m-estimation problems. *ESAIM: Probability and Statistics*, eFirst, 8 2013.
[22] Monolix Team. *The Monolix software, Version 4.1.2. Analysis of mixed effects models.* LIXOFT and INRIA, http://www.lixoft.com/, March 2012.
[23] V.N. Temlyakov. Greedy approximation. *Acta Numerica*, 17:235–409, 2008.
[24] T. von Petersdorff and C. Schwab. Numerical solution of parabolic equations in high dimensions. *M2AN Math. Model. Numer. Anal.*, 38(1):93–127, 2004.