Fundamental principles of data assimilation underlying the Verdandi library: applications to biophysical model personalization within euHeart

Dominique Chapelle, Marc Fragu, Vivien Mallet, Philippe Moireau

Fundamental principles of data assimilation underlying the *Verdandi* library: applications to biophysical model personalization within euHeart

D. Chapelle, M. Fragu, V. Mallet, P. Moireau
Inria, Rocquencourt, B.P. 105, 78150 Le Chesnay, France

Published in *Medical & Biological Engineering & Computing*
(DOI: 10.1007/s11517-012-0969-6)

Abstract

We present the fundamental principles of data assimilation underlying the *Verdandi* library, and how they are articulated with the modular architecture of the library. This translates – in particular – into the definition of standardized interfaces through which the data assimilation library interoperates with the model simulation software and the so-called observation manager. We also survey various examples of data assimilation applied to the personalization of biophysical models, in particular for cardiac modeling applications within the euHeart European project. This illustrates the power of data assimilation concepts in such novel applications, with tremendous potential in clinical diagnosis assistance.

1 Introduction

In order to obtain some information – as detailed as possible – on a *natural system*, such as in geophysics, or regarding the important subcategory of *living systems* which constitute the objects of study in biology and medicine, the most straightforward strategy consists in obtaining *measurements* on the system at hand. Note that we deliberately use the term measurement, related but not reduced to *experiments*, to signify that it is not in general possible to design specific experiments allowing to determine all kinds of physical properties of the system – as is done for instance for industrial systems. By contrast, natural systems induce drastic limitations in measurements, in that they generally must be "taken as they are", namely, observed in their current operating conditions, whether this is due to the practical impossibility of apprehending them comprehensively (e.g. in geophysics), or to the undesirable character of any strong perturbation of the system (invasiveness in living systems).

Specializing now our discussion to the human body, abundant measurements are frequently at hand, in the form of clinical images and signals of various origins. Despite the diversity and rich information contents of these data, they are also inevitably limited in many respects. Beside considerations on sampling and noise, this holds in particular as regards their extent. For example, only 2D measurements, or boundary information, may be available for a 3D system, or only part of the whole domain. Limitations also pertain to measurement types, as some quantities are never measured, such as internal stresses in a living tissue, or various physical constitutive parameters (stiffness, contractility, etc.).

Nevertheless, we may want to consider *mathematical models* to describe and predict the behavior of these systems. Clearly, these models may be seen as providing some complementary information on the system. However, their predictivity requires the careful adjustment of many

parameters – in particular regarding the detailed geometry (anatomy), physical properties, boundary conditions and initial conditions needed in the model – most of which being out of reach of the available measurements.

The purpose of *data assimilation* is then to combine the information available from these two sources – measurements on the one hand, and models and the other hand – by seeking an adequate compromise between, on the one hand, the discrepancy computed between simulations of the model and the corresponding measurements and, on the other hand, the *a priori* confidence in the model, since errors are also present in the measurements. The desired output of this procedure is an estimation of the unknown quantities of interest, namely, (1) state variables (the "trajectories" of the system), and in particular their initial values at a given reference time, and (2) physical parameters which must be prescribed in the model equations.

In terms of clinical applications, the expected benefits are to assist and improve both *diagnosis* and *prognosis*. Diagnosis can be enhanced by providing more complete information on the patient (spatially-distributed quantities, and various otherwise unreachable indicators), and with improved accuracy. Benefits also extend to prognosis, since once data assimilation has been performed the model can be more confidently used to predict natural or artificial evolutions of the system, for example to simulate the effect of various possible therapeutic strategies. Note that, in a prognosis perspective, depending on the application at hand, modeling time scales can vary from – typically – microseconds at the sub-cellular level to seconds (e.g. a heart beat), to hours or weeks when longer evolutions are considered, for instance tissue remodeling.

In this paper, we present fundamental concepts of data assimilation, for both variational and sequential families of methods. We also survey some application examples pertaining to the personalization of biophysical models, in particular for cardiac modeling applications within the euHeart European project. This leads us to discussing the architecture of the *Verdandi* library, and how it relates to the fundamentals of data assimilation and addresses applicative needs.

2 Definitions and notation

We now introduce some basic notation necessary to discuss the fundamental principles of data assimilation.

Physical model definition – First of all, we consider physical models in the form of dynamical systems governed by equations of the type

$$\dot{\mathrm{x}} = A(\mathrm{x}, \theta, t). \tag{1}$$

In this equation, x denotes the so-called *state variable*, namely, the physical quantity which the model aims at describing in its time-wise evolution – hence, the time derivative in the left-hand side – and also frequently spatial variations for distributed quantities. In this generic notation, the whole model is essentially summarized in the so-called *dynamical operator* A, which applies on the state variable itself, and may depend on time t as well as on a set of physical parameters denoted by θ. This operator may arise from various types of physical formulations, e.g. in solid and fluid mechanics or electrophysiology. Mathematically speaking, it may take the form of partial differential equations (PDEs) or ordinary differential equations (ODEs, namely, only differentiated with respect to the time variable), or algebraic systems, in particular.

Clearly, in such model formulations we need to prescribe – hence to estimate when unknown via the data assimilation procedure – the initial condition $\mathrm{x}(0)$ and the parameter vector θ. We dissociate the estimation of the state through $\mathrm{x}(0)$ and the estimation of the parameter θ (also called identification), but formally the two types of estimation can be considered together by

defining an augmented state $x = \binom{\mathrm{x}}{\theta}$ verifying

$$\dot{x} = A(x), \quad \text{and } x(0) = \begin{pmatrix} \mathrm{x}(0) \\ \theta \end{pmatrix}, \tag{2}$$

with a slight abuse of notation since the new augmented operator A has two components $\binom{A(\cdot)}{0}$. Then every estimation can be considered in the light of state estimation.

The variables x and θ can represent fields in a space-distributed form, but they are then space-discretized. In that case we will denote by

$$\dot{\vec{X}} = \mathbf{A}(\vec{X}, \vec{\theta}, t), \tag{3}$$

the space discretization of (1). Typically, in the models considered the state variable may contain a large number of scalar coefficients – typically 10^3 to 10^7 degrees of freedom in a continuum mechanics model – whereas the size of the parameter vector is generally much more limited, and in practice we seldom have to estimate more than a few hundreds of parameter values. Once the initial condition and the parameter vector associated with (1) are estimated, the model can be simulated in time, using appropriate numerical techniques.

We also have to consider time discretizations of the model for solving the dynamics in practice. Every time scheme can be summarized in the form

$$\vec{X}_{n+1} = \mathbf{A}_{n+1|n}(\vec{X}_n, \vec{\theta}), \tag{4}$$

where we frequently have $\vec{X}_n = \vec{X}(t_n)$, even though in some specific discretization schemes we may gather more variables.

Finally, let us point out that every effort to model a physical system suffers from modeling errors. In the data assimilation formalism the modeling error can be considered as a time-dependent quantity $\omega(t)$ that also needs to be estimated. Then we have

$$\dot{\mathrm{x}} = A(\mathrm{x}, \theta, \omega, t). \tag{5}$$

with, most of the time, a linear expression of the form

$$\dot{\mathrm{x}} = A(\mathrm{x}, \theta, t) + B\omega(t). \tag{6}$$

This quantity can be considered in a deterministic framework as a perturbation variable in a specific space, or as a probabilistic uncertainty.

Measurements description – Another important notation concerns the measurements, typically represented by an equation of the type

$$z = H(\mathrm{x}, t) + \chi(t), \tag{7}$$

where z denotes the actual data, H is the so-called *observation operator*, and χ accounts for the error inherent to the measurement process, often called the *noise*, and for representativeness errors. Note that the quantity z will frequently correspond to pre-processed – not raw – data, for example medical images processed with segmentation or optical flow techniques in order to extract some position, displacement or velocity information. We further emphasize that this equation also represents a model – in this case of the measurements – where modeling ingredients are embedded both in the expression of H and the characterization of χ, which may be of probabilistic or deterministic nature.

In practice, we have to take into account that the measurements are also discretized in space and time with a specific level of space discretization and time sampling. It is still possible to rewrite (7) to take into account these discretizations and to adapt the observation operator to model discretization choices. The new observation definition reads

$$\vec{Z}_n = \mathbf{H}_n(\vec{X}_n) + \vec{\chi}_{n,\Delta T}, \tag{8}$$

where in $\vec{\chi}_{n,\Delta T}$ we gather the contributions coming from the measurement noise and the discretization errors (possibly deterministic and biased). For example, for data available at a constant time-sampling ΔT – in general different from, and frequently larger than, the computational time step of the model simulation – we can consider that the observations are only available at simulation time steps coinciding with (or closest to) data sampling times, i.e.

$$\vec{Z}_n = \begin{cases} \vec{Z}(k\Delta T) & \text{if } |t_n - k\Delta T| \leq \epsilon_{\text{obs}}\Delta T \\ \text{undefined} & \text{otherwise} \end{cases}$$

Alternatively, we can regenerate an observation at every simulation time step by considering some time interpolation strategy, for instance a linear scheme, viz.

$$\vec{Z}_n = \Big(\big(\tfrac{(k+1)\Delta T - t_n}{\Delta T}\big)\vec{Z}(k\Delta T) + \big(\tfrac{t_n - k\Delta T}{\Delta T}\big)\vec{Z}((k+1)\Delta T) \Big), \quad \text{for } t_n \in \big[k\Delta T, (k+1)\Delta T\big].$$

3 Fundamental principles and methods

We now proceed to present some fundamental concepts of data assimilation. For the sake of simplicity, we mostly restrict our presentation to a deterministic point of view, whereas many concepts can also be interpreted in a probabilistic light, namely, within the so-called Bayesian framework, see e.g. [2] and references therein.

3.1 Least square minimization: the BLUE algorithm

Let us start by considering only one time and one observation, and let us also assume that the observation operator is linear. The most straightforward approach consists in minimizing the following criterion

$$\min_{\mathrm{x}} \left\{ \mathcal{J}(\mathrm{x}) = \frac{1}{2}\|z - H\mathrm{x}\|^2_M + \frac{1}{2}\|\mathrm{x} - \mathrm{x}_\diamond\|^2_{N_\diamond} \right\}, \tag{9}$$

where M and $N_\diamond$ denote the norms associated with the observation space and state space, respectively. In other words, M and $N_\diamond$ characterize our understanding of the underlying quantities z and x – for example their regularities if they correspond to distributed fields – and they also weigh our confidence in each term in the global balance expressed by the criterion. When considering probabilistic fields, the inverse of these norm operators can be associated with the measurement noise covariance W for the observations, and the state covariance $P_\diamond$ for the state when also choosing $\mathrm{x}_\diamond = \mathbb{E}(\mathrm{x})$ as the expected (or mean) value of x.

In practice this minimization is solved after adequate spatial discretization – when x represents a field – leading to a criterion

$$\begin{aligned} J(\vec{X}) &= \frac{1}{2}\|\vec{Z} - \mathbf{H}\vec{X}\|^2_{\mathbf{M}} + \frac{1}{2}\|\vec{X} - \vec{X}_\diamond\|^2_{\mathbf{N}_\diamond} \\ &= \frac{1}{2}(\vec{Z} - \mathbf{H}\vec{X})^\intercal \mathbf{W}^{-1}(\vec{Z} - \mathbf{H}\vec{X}) + (\vec{X} - \vec{X}_\diamond)^\intercal \mathbf{P}_\diamond^{-1}(\vec{X} - \vec{X}_\diamond), \end{aligned} \tag{10}$$

where $\mathbf{M}$ and $\mathbf{N}_\diamond$ denote the matrix counterparts of the above norms, and $\mathbf{W}$ and $\mathbf{P}_\diamond$ the respective inverse matrices, namely, covariance matrices.

It is easy to show that the result of this minimization is

$$\hat{\vec{X}} = \vec{X}_\diamond + (\mathbf{P}_\diamond^{-1} + \mathbf{H}^\intercal \mathbf{W}^{-1} \mathbf{H})^{-1} \mathbf{H}^\intercal \mathbf{W}^{-1} (\vec{Z} - \mathbf{H}\vec{X}_\diamond). \tag{11}$$

By defining

$$\mathbf{P}_\diamond^+ = (\mathbf{P}_\diamond^{-1} + \mathbf{H}^\intercal \mathbf{W}^{-1} \mathbf{H})^{-1}, \tag{12}$$

we see that (11) becomes

$$\hat{\vec{X}} = \vec{X}_\diamond + \mathbf{P}_\diamond^+ \mathbf{H}^\intercal \mathbf{W}^{-1} (\vec{Z} - \mathbf{H}\vec{X}_\diamond). \tag{13}$$

Using the matrix inversion lemma (see e.g. [15]), we obtain that (11) can be rewritten as

$$\hat{\vec{X}} = \vec{X}_\diamond + \mathbf{P}_\diamond \mathbf{H}^\intercal (\mathbf{W} + \mathbf{H}\mathbf{P}_\diamond \mathbf{H}^\intercal)^{-1} (\vec{Z} - \mathbf{H}\vec{X}_\diamond). \tag{14}$$

These formulations provide a linear least square estimator $\hat{\vec{X}}$ of the state $\vec{X}$ given one observation $\vec{Z}$ and an a priori $\vec{X}_\diamond$. Everything can be summarized in the *filtering* form

$$\begin{cases} \hat{\vec{X}} = \vec{X}_\diamond + \mathbf{K}(\vec{Z} - \mathbf{H}\vec{X}_\diamond) \\ \text{with} \\ \mathbf{K} = \mathbf{P}_\diamond^+ \mathbf{H}^\intercal \mathbf{W}^{-1} = (\mathbf{P}_\diamond^{-1} + \mathbf{H}^\intercal \mathbf{W}^{-1} \mathbf{H})^{-1} \mathbf{H}^\intercal \mathbf{W}^{-1} = \mathbf{P}_\diamond \mathbf{H}^\intercal (\mathbf{W} + \mathbf{H}\mathbf{P}_\diamond \mathbf{H}^\intercal)^{-1} \end{cases} \tag{15}$$

This estimator is classically referred to as the BLUE (best linear unbiased estimator) method.

In the case of a non-linear observation operator $\mathbf{H}(\cdot)$, it is still possible to compute the minimum via e.g. a Newton-like algorithm. We obtain

$$\begin{cases} \hat{\vec{X}}^{(k+1)} = \vec{X}^{(k)} + \mathbf{K}_{(k)} (\vec{Z} - \mathbf{H}(\vec{X}^{(k)})) \\ \text{with} \\ \mathbf{K}_{(k)} = \mathbf{P}_{(k)}^+ \frac{\partial \mathbf{H}_{(k)}}{\partial \vec{X}}^\intercal \mathbf{W}^{-1} = \left(\mathbf{P}_{(k)}^{-1} + \frac{\partial \mathbf{H}_{(k)}}{\partial \vec{X}}^\intercal \mathbf{W}^{-1} \frac{\partial \mathbf{H}_{(k)}}{\partial \vec{X}} \right)^{-1} \frac{\partial \mathbf{H}_{(k)}}{\partial \vec{X}}^\intercal \mathbf{W}^{-1} \end{cases}$$

but we can also be satisfied with an approximate version using only the first iteration of the Newton algorithm. In this case, it is sufficient to use (15) with $\mathbf{H}$ substituted with $\frac{\partial \mathbf{H}}{\partial \vec{X}}$ in the definition of the gain $\mathbf{K}$.

3.2 BLUE with multiple observations

Now, let us explain how this can be extended when considering multiple observations $(\vec{Z}_n)$. Using minimization principles we would find that the estimator is given by

$$\hat{\vec{X}}_n = \left(\sum_{k=0}^{n} \mathbf{H}_k^\intercal \mathbf{W}_k^{-1} \mathbf{H}_k \right)^{-1} \sum_{k=0}^{n} \mathbf{H}_k^\intercal \mathbf{W}_k^{-1} \vec{Z}_k. \tag{16}$$

Then, defining the sequence of symmetric matrices $(\mathbf{P}_n)$

$$\mathbf{P}_n^{-1} = \sum_{k=0}^{n} \mathbf{H}_k^\intercal \mathbf{W}_k^{-1} \mathbf{H}_k, \tag{17}$$

we can show as in [15] that $\hat{\vec{X}}_n$ can be computed recursively by

$$\hat{\vec{X}}_n = \hat{\vec{X}}_{n-1} + \mathbf{K}_n (\vec{Z}_n - \mathbf{H}_n \hat{\vec{X}}_{n-1}), \tag{18}$$

with the filter given by

$$\begin{aligned}\mathbf{K}_n &= \mathbf{P}_{n-1}\mathbf{H}_n^\intercal(\mathbf{W}_n + \mathbf{H}_n\mathbf{P}_{n-1}\mathbf{H}_n^\intercal)^{-1} \\ &= (\mathbf{H}_n^\intercal\mathbf{W}_n^{-1}\mathbf{H}_n + \mathbf{P}_{n-1}^{-1})^{-1}\mathbf{H}_n^\intercal\mathbf{W}_n^{-1} \\ &= \mathbf{P}_n\mathbf{H}_n^\intercal\mathbf{W}_n^{-1}. \end{aligned} \tag{19}$$

This recursive form can only be defined in a linear context but an approximate version can then be formulated for nonlinear operators by again substituting $\mathbf{H}_n$ with $\frac{\partial \mathbf{H}_n}{\partial \vec{X}}$ in (19).

3.3 General variational minimization: The 4D-Var

The previous section allowed us to understand in a simple case the equivalence between criterion minimization and recursive estimation formulae, and also the link between the probabilistic point of view – involving mean and covariances computations – and the deterministic point of view. We can now consider the more general configuration with a model given as a dynamical system and a time-sequence of observations. We demonstrate the detailed strategy for the time-continuous case, but similar principles also apply to time-discrete systems that can reflect the time-discretization of an underlying time-continuous model.

In variational procedures, we consider a criterion to be minimized in order to achieve the above-mentioned compromise between simulation-measurements discrepancy and model confidence, see e.g. [2, 5] and references therein. A typical criterion would read

$$\mathcal{J}_T(\xi_\mathrm{x}, \xi_\theta) = \int_0^T \|z - H(\mathrm{x})\|_M^2 \, dt + \|\xi_\mathrm{x}\|^2_{(P_\diamond)^{-1}} + \|\xi_\theta\|^2_{(P_*)^{-1}}, \tag{20}$$

where $\|.\|_M$, $\|.\|_{(P_\diamond)^{-1}}$ and $\|.\|_{(P_*)^{-1}}$ denote suitable norms for each quantity concerned, and associated with the operators appearing as subscripts. In this criterion, $\mathrm{x}(t)$ is constrained to satisfy the model equation (1) starting from the initial condition $\mathrm{x}(0) = \mathrm{x}_0 + \xi_\mathrm{x}$ and with parameter values given by $\theta = \theta_0 + \xi_\theta$. In order to perform this minimization, a classical strategy consists in computing the gradient of the criterion, which requires the simulation of the so-called *adjoint model*. The adjoint model equation is an evolution system closely related to – and inferred from, indeed – the direct model equation (1), viz.

$$\begin{cases} \dot{p}_\mathrm{x} + \frac{\partial A}{\partial \mathrm{x}}^\intercal p_\mathrm{x} = -\frac{\partial H}{\partial \mathrm{x}}^\intercal M(z - H(\mathrm{x})) \\ p_\mathrm{x}(T) = 0 \\ \dot{p}_\theta + \frac{\partial A}{\partial \theta}^\intercal p_\mathrm{x} = 0 \\ p_\theta(T) = 0 \end{cases} \tag{21}$$

These equations must be simulated backwards in time from the final time T to the initial time in order to obtain the gradient value criterion expressed as

$$\begin{cases} \mathrm{d}_{\xi_\mathrm{x}}\mathcal{J} \cdot \delta\xi_\mathrm{x} = \xi_\mathrm{x}^\intercal (P_\diamond)^{-1}\delta\xi_\mathrm{x} - p_\mathrm{x}(0)^\intercal\delta\xi_\mathrm{x} \\ \mathrm{d}_{\xi_\theta}\mathcal{J} \cdot \delta\theta = \xi_\theta^\intercal (P_*)^{-1}\delta\xi_\theta - p_\theta(0)^\intercal\delta\xi_\theta, \end{cases}$$

Hence, each gradient computation requires the forward simulation of the direct model and the backward simulation of the adjoint, and this must be repeated until convergence of the minimization algorithm. This type of variational procedure is also referred to as "4D-Var" in the data assimilation community, while "3D-Var" is used to refer to minimization estimation performed for static models, or for dynamic models at a given time (namely, without time integral).

Concerning the time discretization, it is classical to formulate an optimal discrete time minimization criterion and find its corresponding adjoint rather than discretizing directly (21). Hence, we consider a criterion of the form

$$J_N(\xi_{\mathrm{x}}, \xi_\theta) = \sum_{k=0}^{N} \|\vec{Z}_k - \mathbf{H}(\vec{X}_k)\|^2_{\mathbf{M}_k} + \|\vec{\xi}_{\mathrm{x}}\|^2_{(\mathbf{P}_\diamond)^{-1}} + \|\vec{\xi}_\theta\|^2_{(\mathbf{P}_*)^{-1}}, \tag{22}$$

with for example $\mathbf{M}_k = \Delta t \mathbf{M}$.

3.4 Sequential optimal procedure: The Kalman algorithms family

By contrast, sequential procedures – also often referred to as *filtering* – proceed by simulating equations closely resembling the direct model equations, with an additional correction term taking into account the discrepancy between the simulation and the actual measurements, namely $z - H(x)$, quantity called the *innovation*. For example when only the initial condition is unknown the filtering equation would be of the type

$$\dot{\hat{x}} = A(\hat{x}, t) + K(z - H(\hat{x})), \tag{23}$$

where the operator K – frequently linear – is called the *filter*. The filtering equation simulation is then started from the candidate initial condition $x_\diamond$, and the aim of the correction is to bring the simulated trajectory close to the target system.

This type of strategy was made extremely popular by the Kalman theory, which formulated an optimal setting for deriving the filter operator, initially when the operators A and H are both linear. In this case, the Kalman equations read

$$\begin{cases} \dot{\hat{x}} = A\hat{x} + PH^\intercal M(z - H\hat{x}) \\ \dot{P} - PA^\intercal - AP + PH^\intercal MHP = 0 \\ P(0) = P_\diamond \\ \hat{x}(0) = x_\diamond \end{cases} \tag{24}$$

where $W = M^{-1}$ and $P_\diamond$ denote the so-called covariance operators of the measurement noise and initial condition uncertainty, respectively. We can see that the filter expression is based on the computation of the time-dependent covariance P which satisfies a Riccatti equation. In fact, in the linear case the variational and Kalman procedures can be shown to be exactly equivalent, and the minimizing direct and adjoint states (x_∞ and p_∞, respectively) are related to the Kalman filter equations by the identity [2]

$$x_\infty = \hat{x} + Pp_\infty.$$

When nonlinearities are to be considered, various extensions are available, and in particular the Extended Kalman Filtering (EKF) approach, in which the linearized forms of the operators are used in the filter equation. However, in such a case the approach is no longer equivalent to the variational setting. Nevertheless, some alternative filter equations can be derived from the variational formulation, but the filter computation then requires solving a Hamilton-Jacobi-Bellman equation in a space which has the dimension of the state variable [15], which is in general not practical.

Note that, in practice, the Kalman (or EKF) approach itself is also quite limited as regards the size of the system which can be handled, since the covariance operator P has the size of the state variable, and is "dense", unlike the dynamical operators. In order to circumvent

this limitation some alternative approaches can be proposed, see Section 3.5, or discretizations involving much fewer degrees of freedom must be considered. This is particularly the case in the context of reduced basis or POD discretizations [9].

Concerning time discretization, it is classical to formulate the optimal filter directly from the optimal time-discrete minimization criteria rather than discretizing directly (24). Hence after some quite tedious computations very similar to those presented in Section 3.2, we can formulate a prediction-correction scheme

1. Prediction:

$$\begin{cases} \hat{\vec{X}}^-_{n+1} &= A_{n+1|n}(\vec{X}^+_n) \\ \mathbf{P}^-_{n+1} &= \frac{\partial \mathbf{A}_{n+1|n}}{\partial \vec{X}} \mathbf{P}^+_n \frac{\partial \mathbf{A}_{n+1|n}}{\partial \vec{X}}^\intercal \end{cases} \tag{25a}$$

2. Correction:

$$\begin{cases} \mathbf{P}^+_{n+1} &= \left(\frac{\partial \mathbf{H}_{n+1}}{\partial \vec{X}}^\intercal \mathbf{W}^{-1}_{n+1} \frac{\partial \mathbf{H}_{n+1}}{\partial \vec{X}} + (\mathbf{P}^-_{n+1})^{-1} \right)^{-1} \\ \mathbf{K}_{n+1} &= \mathbf{P}^+_{n+1} \frac{\partial \mathbf{H}_{n+1}}{\partial \vec{X}}^\intercal \mathbf{W}^{-1}_{n+1} \\ \hat{\vec{X}}^+_{n+1} &= \hat{\vec{X}}^-_{n+1} + \mathbf{K}_{n+1}(\vec{Z}_{n+1} - \mathbf{H}_{n+1}(\hat{\vec{X}}^-_{n+1})) \end{cases} \tag{25b}$$

For the sake of simplicity, we have introduced the Kalman filter in a deterministic context, but the probabilistic counterpart exists. In a linear framework the expressions are exactly the same, albeit with the additional interpretation

$$\begin{cases} \text{a priori mean: } \hat{\vec{X}}^-_{n+1} = \mathbb{E}(\vec{X}_{n+1} | \vec{Z}_0, \dots, \vec{Z}_n), \\ \text{a priori covariance: } \mathbf{P}^-_{n+1} = \mathbb{E}\big((\vec{X}_{n+1} - \hat{\vec{X}}^-_{n+1})(\vec{X}_{n+1} - \hat{\vec{X}}^-_{n+1})^\intercal\big), \\ \text{a posteriori mean: } \hat{\vec{X}}^+_{n+1} = \mathbb{E}\big(\vec{X}_{n+1} | \vec{Z}_0, \dots, \vec{Z}_{n+1}\big), \\ \text{a posteriori covariance: } \mathbf{P}^+_{n+1} = \mathbb{E}((\vec{X}_{n+1} - \hat{\vec{X}}^+_{n+1})(\vec{X}_{n+1} - \hat{\vec{X}}^+_{n+1})^\intercal), \end{cases} \tag{26}$$

This results extend to the non-linear context with the EKF (25) but the identities (26) are then only approximate. To improve the quality of this approximation, the Unscented Kalman Filter has then been introduced [12], based on the idea of substituting means and covariances by empirical quantities computed from sample points:

$$\begin{cases} \hat{\vec{X}}^-_{n+1} = \sum_{i=1}^d \alpha_i \vec{X}^{[i]-}_{n+1}, \\ \mathbf{P}^-_{n+1} = \sum_{i=1}^d \alpha_i (\vec{X}^{[i]-}_{n+1} - \hat{\vec{X}}^-_{n+1})(\vec{X}^{[i]-}_{n+1} - \hat{\vec{X}}^-_{n+1})^\intercal, \\ \hat{\vec{X}}^+_{n+1} = \sum_{i=1}^d \alpha_i \vec{X}^{[i]+}_{n+1}, \\ \mathbf{P}^+_{n+1} = \sum_{i=1}^d \alpha_i (\vec{X}^{[i]+}_{n+1} - \hat{\vec{X}}^+_{n+1})(\vec{X}^{[i]+}_{n+1} - \hat{\vec{X}}^+_{n+1})^\intercal, \end{cases} \tag{27}$$

with $\sum_{i=1}^d \alpha_i = 1$.

In practice, the correction particles are sampled around the mean $\hat{\vec{X}}^+_n$ with a covariance $\mathbf{P}^+_n$ and the prediction samples then verify

$$\vec{X}^{[i]-}_{n+1} = \mathbf{A}(\vec{X}^{[i]+}_n). \tag{28a}$$

Then, by computing

$$\vec{Z}^{[i]-}_{n+1} = \mathbf{H}(\vec{X}^{[i]-}_{n+1}), \quad \vec{Z}^-_{n+1} = \sum_{i=1}^d \alpha_i \vec{Z}^{[i]}_{n+1} \tag{28b}$$

The gain is defined by

$$\begin{cases} \mathbf{K}_{n+1} = (\mathbf{P}_{n+1}^{\vec{X},\vec{Z}}) \cdot (\mathbf{P}_{n+1}^{\vec{Z},\vec{Z}})^{-1} \\ \mathbf{P}_{n+1}^{\vec{X},\vec{Z}} = \sum_{i=1}^{d} \alpha_i (\vec{X}_{n+1}^{[i]-} - \hat{\vec{X}}_{n+1}^{-})(\vec{Z}_{n+1}^{[i]-} - \vec{Z}_{n+1}^{-})^\intercal \\ \mathbf{P}_{n+1}^{\vec{Z},\vec{Z}} = \sum_{i=1}^{d} \alpha_i (\vec{Z}_{n+1}^{[i]-} - \vec{Z}_{n+1}^{-})(\vec{Z}_{n+1}^{[i]-} - \vec{Z}_{n+1}^{-})^\intercal + W_{n+1} \end{cases} \tag{28c}$$

so that we keep having

$$\begin{cases} \hat{\vec{X}}_{n+1}^{+} = \hat{\vec{X}}_{n+1}^{-} + \mathbf{K}_{n+1}(\vec{Z}_{n+1} - \vec{Z}_{n+1}^{-}) \\ \mathbf{P}_{n+1}^{+} = \mathbf{P}_{n+1}^{-} - \mathbf{P}_{n+1}^{\vec{X},\vec{Z}} (\mathbf{P}_{n+1}^{\vec{Z},\vec{Z}})^{-1} (\mathbf{P}_{n+1}^{\vec{X},\vec{Z}})^\intercal \end{cases} \tag{28d}$$

and proceed recursively with new correction particles $\vec{X}_{n+1}^{[i]+}$.

The Ensemble Kalman Filter, introduced in [8], follows the same principles of approximating the covariances by sampled particles with, most of the time, an increased number of particles with respect to the UKF filter, and various ways of sampling the particles around the mean value. Finally Monte Carlo strategies exploit a very large number of particles to give a better approximation of the non-linear optimal filter but the practical details of these methods are beyond the scope of this review focused on large dimensional systems coming from the discretization of PDEs.

3.5 Reduced-order sequential strategies

3.5.1 Reduced-Order Extended Kalman Filtering (ROEKF)

In order to deal with the limitations of sequential strategies due to the system size, a classical strategy consists in assuming a specific reduced-order form for the covariance operators. For example, making the ansatz

$$\forall t, \quad P(t) = L(t)U(t)^{-1}L(t)^\intercal \tag{29}$$

with U an invertible matrix of small size r and L an extension operator, we can show that within linear assumptions the solution of the Riccatti equation in (24) reduces to

$$\dot{L} = AL \text{ and } \dot{U} = L^\intercal H^\intercal M H L. \tag{30}$$

which is actually computable in practice.

In a non-linear framework, we can then approximate the covariance dynamics by extending (30) as

$$\dot{L} = \frac{\partial A}{\partial x} L \text{ and } \dot{U} = L^\intercal \frac{\partial H}{\partial x}^\intercal M \frac{\partial H}{\partial x} L. \tag{31}$$

These strategies have relevant applications in the case of parameter identification *per se*. It is common, indeed, to assume more space regularity for the parameters than for the state initial condition. Hence, after discretization the parameters can be represented by a small number of degrees of freedom. Assuming that we can limit $U(0)$ to the parametric space, the extension operator can then be decomposed into two components $L = \begin{pmatrix} L_{\mathrm{x}} \\ L_\theta \end{pmatrix}$ with $L_\theta = \mathbb{1}$ and

$$\dot{L}_{\mathrm{x}} = \frac{\partial A}{\partial \mathrm{x}} L_{\mathrm{x}} + \frac{\partial A}{\partial \theta}. \tag{32}$$

We recognize in this expression the dynamics of the sensitivity operator $\frac{\partial \mathrm{x}}{\partial \theta}$, which provides a nice interpretation for this strategy of uncertainty covariance reduction. Furthermore, in the

linear framework we can prove that this sequential estimator corresponds to the optimal filter associated with the criterion

$$\mathcal{J}(\xi_\theta) = \int_0^T \|z - H(\mathrm{x})\|_M^2 \, dt + \|\xi_\theta\|^2_{(P_*)^{-1}}, \tag{33}$$

where x follows the trajectory associated with ξ_θ and fixed initial condition, which is commonly used in variational identification procedures. All this justifies naming this strategy *Reduced-Order Extended Kalman Filter* (ROEKF), but it is also known as *Singular Evolutive Extended Kalman Filter* following the work of [22].

This concept can be applied directly on time and space discretized versions of the equations, which leads to a *discrete time Reduced-Order Extended Kalman Filter*.

1. Prediction:

$$\begin{cases} \hat{\vec{X}}^-_{n+1} & = \mathbf{A}_{n+1|n}(\vec{X}^+_n) \\ \mathbf{L}_{n+1} & = \frac{\partial \mathbf{A}_{n+1|n}}{\partial \vec{X}} \mathbf{L}_n \end{cases} \tag{34a}$$

2. Correction:

$$\begin{cases} \mathbf{U}_{n+1} & = \mathbf{U}_n + \mathbf{L}^\intercal_{n+1} \frac{\partial \mathbf{H}_{n+1}}{\partial \vec{X}}^\intercal \mathbf{W}^{-1}_{n+1} \frac{\partial \mathbf{H}_{n+1}}{\partial \vec{X}} \mathbf{L}_{n+1} \\ \mathbf{K}_{n+1} & = \mathbf{P}^+_{n+1} \frac{\partial \mathbf{H}_{n+1}}{\partial \vec{X}}^\intercal \mathbf{W}^{-1}_{n+1} \\ \hat{\vec{X}}^+_{n+1} & = \hat{\vec{X}}^-_{n+1} + \mathbf{K}_{n+1}(\vec{Z}_{n+1} - \mathbf{H}_{n+1} \hat{\vec{X}}^-_{n+1}) \end{cases} \tag{34b}$$

3.5.2 Reduced-Order Unscented Kalman Filtering (ROUKF)

Alternatively, this strategy can be coupled with the UKF approach by showing that particles can be generated only in the space of small dimension and the computation made in the UKF filter (28) can be compatible with the time discretized counterpart of (29). This was proven in [16, 17] which also provided a very general version of the *Reduced Order Unscented Kalman Filter* (ROUKF). In particular this algorithm is close to the *Singular Evolutive Interpolated Kalman Filter* [21, 10] for a choice of particles $d = r + 1$ and reads as follows.

Algorithm – Given an adequate sampling rule, we store the corresponding weights in the diagonal matrix $\mathbf{D}_\alpha$ and precompute so-called unitary sigma-points (i.e. with zero mean and unit covariance) denoted by $(\vec{I}^{[i]})_{1 \le i \le r+1}$; we then perform at each time step

1. Sampling:

$$\begin{cases} \mathbf{C}_n & = \sqrt{(\mathbf{U}_n)^{-1}} \\ \hat{\vec{X}}^{[i]+}_n & = \hat{\vec{X}}^+_n + \mathbf{L}_n \cdot \mathbf{C}^\intercal_n \cdot \vec{I}^{[i]}, \quad 1 \le i \le r+1 \end{cases} \tag{35a}$$

2. Prediction:

$$\begin{cases} \hat{\vec{X}}^{[i]-}_{n+1} & = \mathbf{A}_{n+1|n}(\hat{\vec{X}}^{[i]+}_n), \quad 1 \le i \le r+1 \\ \hat{\vec{X}}^-_{n+1} & = \sum_{i=1}^{r+1} \alpha_i \hat{\vec{X}}^{[i]-}_{n+1} \end{cases} \tag{35b}$$

3. Correction:

$$\begin{cases} \mathbf{L}_{n+1} & = [\hat{\vec{X}}^{[*]-}_{n+1}]\mathbf{D}_\alpha[\vec{I}^{[*]}]^\intercal \\ \vec{Z}^{[i]-}_{n+1} & = \mathbf{H}_{n+1}(\hat{\vec{X}}^{[i]-}_{n+1}) \\ \vec{Z}^{-}_{n+1} & = \sum_{i=1}^{r+1} \alpha_i \vec{Z}^{[i]-}_{n+1} \\ \mathbf{\Gamma}_{n+1} & = [\vec{Z}^{[*]-}_{n+1}]\mathbf{D}_\alpha[\vec{I}^{[*]}]^\intercal \\ \mathbf{U}^{n+1} & = \mathbf{1} + \mathbf{\Gamma}^\intercal_{n+1}\mathbf{W}^{-1}_{n+1}\mathbf{\Gamma}_{n+1} \\ \hat{\vec{X}}^{+}_{n+1} & = \hat{\vec{X}}^{-}_{n+1} - \mathbf{L}_{n+1}\mathbf{U}^{n+1}\mathbf{\Gamma}^\intercal_{n+1}\mathbf{W}^{-1}_{n+1}(\vec{Z}_{n+1} - \vec{Z}^{-}_{n+1}) \end{cases} \tag{35c}$$

where we denote by $[\vec{I}^{[*]}]$ the matrix concatenating the $(\vec{I}^{[i]})$ vectors side by side, and similarly for other vectors.

3.6 Luenberger observers

The so-called *observer theory*, initiated by Luenberger [14], is based on the simple realization that, defining the estimation error

$$\tilde{\mathrm{x}} = \mathrm{x} - \hat{\mathrm{x}},$$

we obtain in the linear case, when subtracting the direct model and filtered equations, the dynamics

$$\dot{\tilde{\mathrm{x}}} = (A - KH)\tilde{\mathrm{x}} - K\chi. \tag{36}$$

This type of dynamical equation is well-known in control theory: it is similar to the closed-loop controlled equation of a system of natural dynamics governed by A, and submitted to a feedback control defined by the operator K applied on the quantity observed through H. Hence, obtaining an accurate estimation of the state variable is exactly equivalent to driving the estimation error $\tilde{\mathrm{x}}$ to zero – namely, stabilizing this error – using the feedback control K.

This approach opened new avenues for formulating novel filtering approaches, because for actual dynamical systems control and stabilization motivations have frequently already led to the formulation of effective feedback controls, used in a large variety of industrial systems. Hence, these approaches can be quite directly adapted to obtain adequate filters which – unlike the Kalman filter – are tractable in practice for large systems. Moreover, these filters are often deeply rooted in the physics of the system considered, hence the computational building blocks needed are likely to be at hand in the system simulation software. However, in the Luenberger approach we lose the Kalman optimality, which of course only holds in quite restricted (linear) cases.

For examples of such approaches applicable to beating heart models, with detailed descriptions, theoretical analyses and extensive assessments, we refer in particular to [18, 19]. We also point out that the Luenberger observer approach is also sometimes referred to as "nudging" in the data assimilation community [1].

3.7 Joint state-parameter estimation with Luenberger observers

As apparent in the above discussion, Luenberger observers were originally designed for state estimation. When parameters are to be jointly estimated, it is quite classical in the filtering context to complement the state equation (1) with the artificial parameter dynamics

$$\dot{\theta} = 0, \tag{37}$$

as already discussed when introducing (2). Then the whole estimation objective is to estimate the initial condition of the augmented state (x, θ). Of course, when a Kalman approach is out of reach for the state variable alone, it holds *a fortiori* for the augmented state. On the other hand, devising a Luenberger observer for the augmented state is difficult, because part of the dynamics is non-physical, hence feedback controls are not readily available for the joint system.

Nevertheless, an effective approach for joint state-parameter estimation was proposed in [18], based on a Luenberger observer applied on the state equation alone. In essence, this first-stage state estimation reduces the uncertainty to the parameter space, which allows to consider a ROEKF approach for handling the remaining parameter uncertainty. This algorithm can be summarized as

$$\begin{cases} \dot{\hat{\mathrm{x}}} = A(\hat{\mathrm{x}}, \hat{\theta}) + K_{\mathrm{x}}(z - H\hat{\mathrm{x}}) + L_{\mathrm{x}}\dot{\hat{\theta}} \\ \dot{\hat{\theta}} = U^{-1} L_{\mathrm{x}}{}^{\intercal} H^{\intercal} M (z - H\hat{\mathrm{x}}) \\ \dot{L}_{\mathrm{x}} = \left(\frac{\partial A}{\partial \mathrm{x}} - K_{\mathrm{x}} H\right) L_{\mathrm{x}} + \frac{\partial A}{\partial \theta} \\ \dot{U} = L_{\mathrm{x}}{}^{\intercal} H^{\intercal} M H L_{\mathrm{x}} \\ \hat{\mathrm{x}}(0) = \mathrm{x}_0 \\ \hat{\theta}(0) = \theta_0 \\ L_{\mathrm{x}}(0) = 0 \\ U(0) = U_0 \end{cases} \tag{38}$$

where K_{x} denotes the state filter (Luenberger observer operator), L_{x} represents the sensitivity of the state variable with respect to the parameters, and U the inverse of the parameter estimation error covariance. This methodology derives from the above-discussed *reduced-order filtering* approaches, see also e.g. [22], since only the part of the dynamics concerning parameters is handled using optimal filtering.

This joint estimation approach was later extended in [16] towards strategies inspired from the ROUKF crucially allowing to avoid the computation of differentiated operators required in (38).

4 Application examples

4.1 Within the euHeart consortium

Two recent results have illustrated the great potential of data assimilation in the context of the biophysical personalization of cardiovascular systems as purported in the euHeart project [24].

The first example in [26] consists in estimating a parameter associated with the passive part of a cardiac constitutive law. Here, the tissue passive behavior is assumed to be described by the Guccione law, one of the main material laws used in cardiac mechanics. The overall mechanical model is assumed to be quasi-static and submitted to time-dependent external forces. In this context, the strategy followed by [26] is similar to a BLUE filter with multiple observations, since there is no dynamics associated with the state. In this context, the uncertainty was reduced to the parameter space related to the Guccione law, and an ROUKF algorithm was chosen for its combined robustness and ease of implementation. The results on synthetically generated data demonstrate the potential of the approach for the estimation of such complex constitutive laws.

The second example in [3] also used a ROUKF filter, albeit in its full version, since the model considered is a complete dynamical fluid-structure interaction problem in large strains. Here again, the motivation is to estimate some parameters in the solid constitutive law designed for the arterial walls. The authors show in particular how a complex time-discretization scheme

can be processed in the data assimilation formalism presented in the previous section. In this case also, results on synthetically generated data demonstrate the effectiveness of the approach.

4.2 Other examples in the cardiovascular context

The previous two estimation problems have also been considered within variational procedures, in [27] for the estimation of a cardiac constitutive law and in [6] for the estimation of wall parameters in blood flows with fluid-structure interaction models. In the second case synthetic data experiments are performed but theoretical stability results are provided whereas in [27], the Guccione law parameters are obtained from real data and compared to the literature. Concerning the estimation of the active part of a mechanical model, variational methods have also been employed in [7] to identify some *contractility* parameters. In all these results, the data are assumed to be displacements extracted by adequate processing techniques from the image cine-MRI or tagged-MRI.

Data assimilation is also becoming more popular in electrophysiology, where variational strategies have been employed e.g. in [20, 23] or filtering methods in [25]. The identification of parameters related to the cell electric properties are the main objectives, but state estimation of the propagating wave can also be valuable, in particular when the electrical activity is an input in a mechanical model to initiate the contraction. Therefore, Eikonal approximations of the wave propagation have been combined with probabilistic filtering in [13] to estimate the wave trajectory and some propagation parameters.

4.3 Specific examples with image data

Image data provide a very rich source of information, albeit are frequently difficult to directly associate with a measurement operator H, e.g. providing some displacements corresponding to a biomechanical model in a Lagrangian formulation. In general, it is much easier to extract from the cardiac or cardiovascular images only some surfaces of interest describing the contours of the tissue – or the tagged planes in tagged MRI. This kinematical information is fundamentally Eulerian, as opposed to Lagrangian displacements in the model. Therefore, in [19] and [4] for domain contours or in [11] for tagged data, the authors proposed to extend the classical data assimilation observation operator to a so-called *discrepancy operator*, in order to directly use the information provided by the segmentation processes. The key idea is to note that the data and observation operator are always considered together in data assimilation methods in the form $z - H(\mathrm{x})$. As initiated in [19], this quantity is thus replaced by a non-linear form by considering an operator D,

$$z - H(\mathrm{x}) \to D(\mathrm{x}, t),$$

In the case of segmentations and model comparison we can then consider a discrepancy operator based on distances computations

$$D(\mathrm{x}, t) = \underline{\mathrm{dist}}(\underline{\mathrm{x}}(\underline{\xi}, t), t),$$

with, e.g., interpolation rules between successive images

$$\underline{\mathrm{dist}} : \left| \begin{array}{l} L^2(\Sigma) \times [0,T] \mapsto L^2(\Sigma) \\ \qquad (\underline{\mathrm{x}}(\underline{\xi}), t) \to \underline{\mathrm{dist}}(\underline{\mathrm{x}}(\underline{\xi}), t) = \Big(\big(\frac{t_{k+1}-t}{\Delta T}\big) \mathrm{dist}_{\mathcal{S}_k}(\underline{\mathrm{x}}(\underline{\xi})) + \big(\frac{t-t_k}{\Delta T}\big) \mathrm{dist}_{\mathcal{S}_{k+1}}(\underline{\mathrm{x}}(\underline{\xi})) \Big) \times \\ \qquad\qquad \Big(\big(\frac{t_{k+1}-t}{\Delta T}\big) \underline{n}_{\mathcal{S}_k}(\underline{\mathrm{x}}(\underline{\xi})) + \big(\frac{t-t_k}{\Delta T}\big) \underline{n}_{\mathcal{S}_{k+1}}(\underline{\mathrm{x}}(\underline{\xi})) \Big), \quad \text{if } t \in [t_k, t_{k+1}] \end{array} \right. \tag{39}$$

This methodology was applied in particular in [4] to estimate regional contractility parameters in an infarcted heart based on actual MR images, thereby demonstrating the potential of the complete modeling and estimation chain for diagnosis assistance purposes.

5 Verdandi: a generic data assimilation library

There exists data assimilation software with different scientific contents and technical designs. For reference, we list the main software that is adapted to high-dimensional applications that we are aware of. The OpenDA library (`http://www.openda.org`) focuses on Kalman filters and methods for parameter estimation. It is implemented in Java and relies on XML configuration files. It originates from Delft University of Technology and is distributed under the GNU LGPL. The Parallel Data Assimilation Framework (PDAF, `http://pdaf.awi.de`) is also focused on Kalman filters, with special attention given to parallelism. It is written in Fortran 90 and distributed by the Computing Center of the Alfred Wegener Institute under the GNU GPL. The Data Assimilation Research Testbed (DART, `http://www.image.ucar.edu/DAReS/DART`), from the National Center for Atmospheric Research (NCAR), provides another Fortran implementation of the Kalman ensemble filter. Several major organizations involved in data assimilation have their internal solutions. For instance, the European Centre for Medium-Range Weather Forecasts (ECMWF) develops the Object-Oriented Prediction System (OOPS), written in C++ and Fortran. Different tools are also available to help implementing data assimilation algorithms. An example is the open-source coupler PALM (`http://www.cerfacs.fr/globc/PALM_WEB`) distributed by Cerfacs. It can ease the coupling between a data assimilation algorithm, a numerical model and an observation module.

Realizing that data assimilation principles and practical experiences present a generic character was the starting point of Verdandi. In order to guarantee genericity and performance at the same time, we took different directions than other data assimilation software. The library also provides unique tools to carry out data assimilation experiments. The design of the library is detailed below.

We implemented a wide range of estimation algorithms within a common modularity paradigm, with the major objective to provide biophysical personalization tools for the various above-discussed models, but more generally to be of interest for the modeling community concerned with merging models and data. Despite their diversity, all these models have in common the large dimension of the corresponding numerical systems to be solved, and the *Verdandi* library is targeted at providing generic data assimilation procedures well-adapted to large-dimensional systems. This library gathers until now optimal interpolation, Kalman filter and non-linear extensions, reduced-order versions, ensemble Kalman filter, non-linear filters in probabilistic or deterministic forms but also 4D-Var adjoint-based variational strategies.

The *Verdandi* library is distributed[1] under the GNU LGPL license. It is based on generic C++ programming – a choice guided by performance – but is directly linkable with Python programs in order to ease scripting, debugging, visualization of results, and so on. Full compliance with the C++ standard warrants straightforward portability, and it has been thoroughly tested on Linux, MacOS, and Windows platforms.

5.1 Library overall design

Following the above methodological discussion, we identify three main components in the overall design of *Verdandi*:

[1] `http://verdandi.gforge.inria.fr`

– the numerical model, which provides the state vector $\vec{X}_n$, the dynamics with $\mathbf{A}_{n+1|n}$ (and possibly its tangent linear and adjoint versions), and statistics for the initial state error $\mathbf{P}_\diamond$ and the model error (when applicable);

– the observation manager, which provides the observations $\vec{Z}_n$, their error covariance matrix $\mathbf{W}_n$ and the observation operator $\mathbf{H}_n$;

– the data assimilation algorithm, which drives the model and the observation manager along the assimilation procedure.

Each component is implemented in a separate C++ class. A class is an entity that contains variables, called attributes, and member functions that usually operate on the attributes. For instance, the model class has the state vector $\vec{X}_n$ as attribute, and a member function called `GetState` to access this state vector.

An assimilation algorithm encapsulates the model and the observation manager. It is in charge of all assimilation computations, such as the correction step (25b) in the Kalman filter. It delegates the specific computations to the model and the observation operator, such as the time integration over one time step (4). An assimilation algorithm is implemented with essentially no assumption on the inner implementation of the model and the observation manager. *Verdandi* provides several data assimilation algorithms and expects that the user plugs a model and an observation manager with specified interfaces. Figure 1 illustrates the links between the three components.

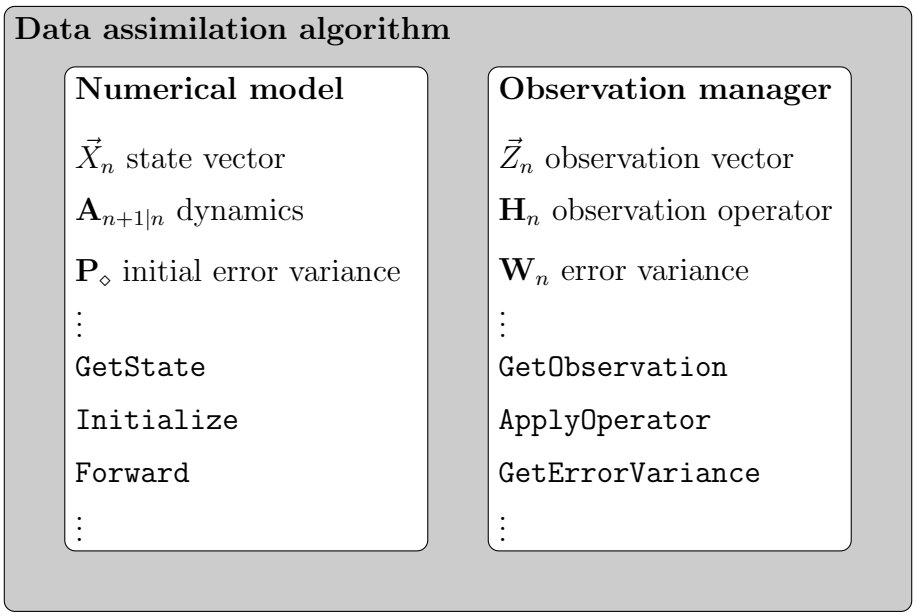

Figure 1: In *Verdandi*, the data assimilation algorithm is implemented in a class that encapsulates a model and an observation manager. The model defines the state vector, the dynamics, the associated error statistics, ... The observation manager provides the observations, the observation operator, the associated error statistics, ... These variables are accessed through member functions like `GetState` or `GetObservation`.

5.2 Standard interfaces

The model and the observation manager are supposed to provide a standard interface so that *Verdandi* algorithms may be applied with them. In the model interface, one may find, among

other member functions: `GetState()` that returns a reference on the state vector; `Forward()` that carries out the time integration for one time step; `ApplyTangentLinearOperator(x)` that applies the tangent linear operator to the vector `x`; `GetStateErrorVarianceRow(i, v)` that returns in `v` the row `i` of the state error covariance matrix. A simulation without assimilation may be carried out with the following C++ lines:

```
model.Initialize();
while (!model.HasFinished())
{
    model.InitializeStep();
    model.Forward();
    model.FinalizeStep();
}
model.Finalize();
```

Each line makes a call to one model member function. The time loop stops when the model declares the simulation over.

Similarly, the observations and associated variables are provided by a few member functions of the observation manager. Notice that the observation operator depends on the model since it maps from the model state space into observation space. As a consequence, several member functions of the observation manager take the model as argument. Among the member functions of the interface, one may find: `SetTime(model, t)` that prepares the observation manager so that subsequent calls return values at time `t`; `GetObservation(y)` that returns the observation vector; `ApplyOperator(x, y)` that applies the observation operator; `GetErrorVariance()` that returns a reference to observation error covariance matrix.

The optimal interpolation, which replaces the state vector with BLUE whenever observations become available, is roughly implemented with these lines:

```
model.Initialize();
observation_manager.Initialize();
while (!model.HasFinished())
{
    model.InitializeStep();
    model.Forward();
    observation_manager.SetTime(model, model.GetTime());
    if (observation_manager.HasObservation())
    {
        state = model.GetState(); // Copy by reference.
        observation_manager.GetInnovation(state, innovation);
        ComputeBLUE(model, observation_manager, innovation, state);
        model.StateUpdated(); // Informs the model that its state
                              // has been updated.
    }
    model.FinalizeStep();
}
model.Finalize();
```

5.3 Implementation strategies

The algorithms are implemented with few assumptions on the data structures. The state vector, the error statistics, the observations, and so on, are provided in data structures defined

by the user. Out of the box, *Verdandi* supports a wide range of data structures, from dense to sparse structures, from contiguous to distributed memory blocks. For example, within the model, the state vector may be composed of a collection of non-contiguous memory blocks – one block per physical variable. In case even more structures were needed, advanced users could define their own data structures, provided they would implement the corresponding basic linear algebra.

The core library is implemented in C++, but the core model and observation manager can be implemented in another language. Users however need to implement the adequate C++ interface to their software. This task is rather straightforward when the software is in Fortran, C or C++. *Verdandi* also provides a C++ interface to a generic Python model, so that writing the interface to a Python model should be an easy task. Note that the high-dimensional data (especially the state vector) can always be provided to C++ by reference (or with pointers) so that no memory duplication is required, even with an underlying Python model.

In order to guarantee the best performance along with the genericity, the type of the data structures must be known at compile time. Hence an assimilation algorithm is always a C++ class template, with the types of the model and observation manager as arguments. The data types are defined by the user, inside the model and the observation manager, with `typedef` declarations. All key variables can have their own data structure – for example, the state vector and the observation vector may be stored in two different formats.

In addition to its C++ core (mainly, the assimilation algorithms available in C++ classes), *Verdandi* supports the automatic generation of Python interfaces. The interfaces are generated by SWIG, for any assimilation algorithm, and also for the numerical model and the observation manager. The C++ interfaces are first compiled, and SWIG generates wrappers in Python, so that all member functions of the model, the observation manager and the assimilation algorithm are exposed in Python.

5.4 Other features

All *Verdandi* objects are configured with Lua scripts. It means that the configuration files are interpreted and can execute non-trivial operations (system calls, numerical computations, ...).

The library includes perturbation schemes which are primarily used in Monte Carlo simulations and Ensemble Kalman filter. The perturbations can be applied to input parameters of the model. Input fields can be perturbed with spatially correlated perturbations.

Different tools are provided to help users implement their interfaces. One tool collects and aggregates observations within given time spans, so that observations can be assimilated even if the model times do not coincide with the observation times. Another tool allows the classes (model, observation manager, assimilation algorithm and possibly others) to send/receive messages to/from one or several other classes. In particular, it provides a direct communication channel between the model and the observation manager, which is otherwise impossible since both objects are entirely driven by the assimilation algorithm. Once a message is received by an object, this can trigger an action that is not part of the assimilation algorithm.

6 Conclusions

We have presented the fundamental principles of data assimilation underlying the *Verdandi* library, and how they are articulated with the modular architecture of the library. This translates – in particular – into the definition of standardized interfaces through which the data assimilation library interoperates with the model simulation software and the observation manager.

We also discussed various examples of data assimilation applied to the personalization of biophysical models, in particular for cardiac modeling applications within the euHeart European project. Whereas such applications are somewhat specific in some respects – for example in the type of data considered – this also shows that they can benefit from data assimilation methodologies valid within a wider scope. This justifies pursuing the development of the library in a long-term perspective, for the benefit of the VPH community – for example *Verdandi* is now also used and further developed in the VPH-Share European project – as well as in other application fields and in the industry, which is why the LGPL open-source license was selected for the software distribution.

Acknowledgments: This work has been partially supported by the European Commission (FP7-ICT-2007-224495: euHeart and FP7-ICT-2009-269978: VPH-Share).

References

[1] Auroux D, Blum J (2008). A nudging-based data assimilation method: the Back and Forth Nudging (BFN) algorithm. Nonlinear Processes In Geophysics, 15(2):305–319.

[2] Bensoussan A (1971). Filtrage Optimal des Systèmes Linéaires. Dunod.

[3] Bertoglio C, Moireau P, Gerbeau JF (2012). Sequential parameter estimation for fluid-structure problems. application to hemodynamics. Int. J. Num. Meth. Biomedical Engng., published online, DOI: 10.1002/cnm.1476.

[4] Chabiniok R, Moireau P, Lesault PF, Rahmouni A, Deux JF, Chapelle D (2011). Estimation of tissue contractility from cardiac cine-mri using a biomechanical heart model. Biomechanics and Modeling in Mechanobiology. Published online.

[5] Chavent G (2010). Nonlinear Least Squares for Inverse Problems. Springer.

[6] D'Elia M, Perego M, Veneziani A (2011). A variational data assimilation procedure for the incompressible Navier-Stokes equations in hemodynamics. Journal of Scientific Computing, 1–20.

[7] Delingette H, Billet F, Wong KCL, Sermesant M, Rhode K, Ginks M, Rinaldi CA, Razavi R, Ayache N (2012) Personalization of Cardiac Motion and Contractility From Images Using Variational Data Assimilation. IEEE Transactions on Biomedical Engineering, 59(1):20–24.

[8] Evensen G (1994). Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. Journal of Geophysical Research, 99:10143–10162.

[9] Chapelle D, Gariah A, Moireau P, Sainte-Marie J (2012). A Galerkin strategy with Proper Orthogonal Decomposition for parameter-dependent problems - Analysis, assessments and applications to parameter estimation. Submitted to M2AN.

[10] Hoteit I, Pham DT, Blum J (2002). A simplified reduced order Kalman filtering and application to altimetric data assimilation in Tropical Pacific. Journal of Marine Systems, 36(1–2):101–127.

[11] Imperiale A, Chabiniok R, Moireau P, Chapelle D (2011). Constitutive parameter estimation methodology using tagged-MRI data. In Proceedings of FIMH'11. Springer.

[12] Julier S, Uhlmann J, Durrant-Whyte H (2000). A new method for the nonlinear transformation of means and covariances in filter and estimators. IEEE Transactions on Automatic Control, 45(3):447–482.

[13] Konukoglu E, Relan J, Cilingir U, Menze BH, Chinchapatnam P, Jadidi A, Cochet H, Hocini M, Delingette H, Jais P, Haïssaguerre M, Ayache N, Sermesant M (2011). Efficient probabilistic model personalization integrating uncertainty on data and parameters: Application to Eikonal-Diffusion models in cardiac electrophysiology. Prog Biophys Mol Bio, 107(1):134–146.

[14] Luenberger DG (1963). Determining the State of a Linear with Observers of Low Dynamic Order. PhD Thesis, Stanford University.

[15] Moireau P (2008). Filtering-based Data Assimilation for Second-Order Hyperbolic PDEs. Applications in Cardiac Mechanics. PhD Thesis, Ecole Polytechnique.

[16] Moireau P, Chapelle D (2010). Reduced-order Unscented Kalman Filtering with application to parameter identification in large-dimensional systems. COCV. Published online, doi:10.1051/cocv/2010006.

[17] Moireau P, Chapelle D (2011). Erratum of article "reduced-order Unscented Kalman Filtering with application to parameter identification in large-dimensional systems". COCV, 17:406–409. doi:10.1051/cocv/2011001.

[18] Moireau P, Chapelle D, Le Tallec P (2008). Joint state and parameter estimation for distributed mechanical systems. Computer Methods in Applied Mechanics and Engineering, 197:659–677.

[19] Moireau P, Chapelle D, Le Tallec P (2009). Filtering for distributed mechanical systems using position measurements: Perspectives in medical imaging. Inverse Problems, 25(3):035010 (25pp). doi:10.1088/0266-5611/25/3/035010.

[20] Moreau-Villeger V, Delingette H, Sermesant M, Ashikaga H, McVeigh ER, Ayache N (2006). Building maps of local apparent conductivity of the epicardium with a 2-D electrophysiological model of the heart. IEEE Transactions on Biomedical Engineering, 53(8):1457–1466.

[21] Pham DT (2001). Stochastic methods for sequential data assimilation in strongly nonlinear systems. Journal of Marine Systems, 129:1,194–1,207.

[22] Pham DT, Verron J, Roubaud MC (1998). A singular evolutive extended Kalman filter for data assimilation in oceanography. Journal of Marine systems, 16(3-4):323–340.

[23] Relan J, Chinchapatnam P, Sermesant M, Rhode K, Ginks M, Delingette H, Rinaldi CA, Razavi R, Ayache N (2011). Coupled Personalization of Cardiac Electrophysiology Models for Prediction of Ischaemic Ventricular Tachycardia. Journal of the Royal Society Interface Focus, 1(3):396-407.

[24] Smith N, de Vecchi A, McCormick M, Nordsletten D, Camara O, Frangi AF, Delingette H, Sermesant M, Relan J, Ayache N, Krueger MW, Schulze WHW, Hose R, Valverde I, Beerbaum P, Staicu C, Siebes M, Spaan J, Hunter P, Weese J, Lehmann H, Chapelle D, Razavi R (2011). euHeart: personalized and integrated cardiac care using patient-specific cardiovascular modelling. Interface Focus, 1(3):349–364.

[25] Wang L, Zhang H, Wong KCL, Shi P (2009). A reduced-rank square root filtering framework for noninvasive functional imaging of volumetric cardiac electrical activity. *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, 533–536.

[26] Xi J, Lamata L, Lee J, Moireau P, Chapelle D, Smith N (2011). Myocardial transversely isotropic material parameter estimation from in-silico measurements based on a reduced-order unscented Kalman filter. Journal of the Mechanical Behavior of Biomedical Materials, 4(7):1090–1102.

[27] Xi J, Lamata P, Shi W, Niederer S, Land S, Rueckert D, Duckett D, Shetty A, Rinaldi CA, Razavi R (2011). An automatic data assimilation framework for patient-specific myocardial mechanical parameter estimation. Functional Imaging and Modeling of the Heart, 392–400.