
Politecnico di Milano

Facoltà di Ingegneria dei Sistemi
Corso di Laurea in Ingegneria Matematica

# Estimation of the compliance of the arterial walls through Kalman filtering

*Relatori:*
prof. L. Formaggia
prof. F. Nobile

Annica Elisabeth Andersson (mat. 735603)
Anno Accademico 2010/2011

Two is not equal to three, not even for large values of two.

*Grabel's law*

# Contents

# Abstract

Very powerful tools to model bloodflow in the arteries have been developed in recent years, giving an accurate description of how important variables, like pressure, section area and velocity change during a heartbeat. However, the physical parameters that intervene can vary considerably between patients, making predictions difficult in specific cases.

In order to adapt the simulation to each patient, a Kalman filter has been implemented, first in its classical version, then generalised into an extended Kalman filter (EKF). This method uses the knowledge of how a state vector evolves in time along with *in vivo* measurements to filter the measurement error and the inaccuracy we insert by making a guess on the parameters. If we apply it to a state vector made up of the section area, the mean velocity and the parameter $\beta$, which is related to the compliance of the vessel wall, we arrive to an estimation of the parameter in the specific patient. The procedure, especially the EKF, attains good accuracy in most of the tested cases, and shows robustness towards measurement errors. In addition it can be applied to cases where we only have measurements on one state variable and where we only have a low frequency of measurements.

Having an estimate of the parameter can help choosing the treatment in case of need. For instance it could help dimensioning the stent that has to be inserted, since it gives us the possibility to simulate the result of a local increase of stiffness of the wall.

# Abstract

In anni recenti sono stati sviluppati degli strumenti molto potenti per simulare il flusso sanguigno, che descrivono accuratamente come cambiano durante un battito cardiaco delle variabili importanti, come l'area, la pressione e la velocità. Tuttavia, i parametri fisici in gioco possono differire considerevolmente tra un paziente ed l'altro, rendendo la predizione difficile nei casi specifici.

Nell'ottica di adattare la simulazione a ciascun paziente, abbiamo quindi implementato un filtro di Kalman, prima nella sua versione classica, e poi nella generalizzazione EKF (Extended Kalman Filter). Questo metodo sfrutta la conoscenza del modo in cui uno stato evolve nel tempo e delle misurazioni *in vivo* del paziente per filtrare gli errori di misura e l'errore compiuto nello stabilire la stima iniziale dei parametri sconosciuti. Applicando il metodo ad un vettore di stato composto da area della sezione, velocità media ed il parametro $\beta$, che è legato alla rigidezza della parete dell'arteria, si arriva alla stima del parametro sullo specifico paziente. La procedura, ed in particolare il metodo EKF, ottiene una buona accuratezza nella maggior parte dei casi studiati, e mostra robustezza rispetto ad errori di misura. Inoltre può essere applicato a casi in cui si misura una sola delle variabili di stato, e in cui il numero di misure temporali è ridotto.

Avere una stima del parametro può essere d'aiuto nel definire la terapia in alcuni casi. Per esempio potrebbe servire a dimensionare lo stent da inserire, poiché permette di simulare gli effetti di un irrigidimento locale della parete.

# Introduction

In recent years, mathematical modeling has gained attention in a vast and evergrowing number of fields of science and engineering. Its role in simplifying prototyping and reducing costs cannot be questioned, and has partially determined its success. However, there are fields where simulations can have an even more important outcome: that of reducing the risk of certain types of surgery, by using more effective methods.

This is the domain on which this thesis focuses. In particular it aims at estimating an important parameter of the arterial wall, its compliance, in an *in vivo* scenario. The compliance can then be used to simulate the bloodflow if a stent is inserted inside the vessel, and foresee if there are areas which will be under excessive stress. This could lead to the choice of a certain type of stent instead of another, to a more realistic evaluation of the risks, and even to the definition of the ideal stent for a certain patient.

In order to obtain an estimation of the compliance, we have chosen to use a Kalman filter. This method consists in having knowledge about how a certain state evolves in time, and having some measurements of values related to the state vector. The power of the method lies in the fact that at each measuring-step we can have much fewer measurements than degrees of freedom of the state, meaning that it is a hidden state estimator. For instance, in one of the tests discussed further on, we had 9 measurements at each time step, while the state was a vector of 63 elements. Starting from an initial guess, at each time-step the values we predict from the state evolution and the measurements are confronted, and the state is corrected accordingly, through an optimal (in the linear case) bayesian estimation.

In particular, we have applied this strategy on an augmented state vector, made up of the physical variables (area and mean velocity) and the compliances of the different vessels of our domain. In this case the method tends to filter the measurement error (which is inevitable) and then adapt the compliances so that they yield values similar to the measurements, thus converging towards the "real" values of the parameters. It is very important not to have ambiguities in the system, as to say different sets of parameters that yield the same output, but such problems have not been observed in our test cases.

The measurements can come from different instruments, with the related measurement error, frequency of measure and number of measurements along the vessel. For instance we can measure the area through tomography, or the velocity through Doppler ultrasound. Any of these strategies, or all

of them together if possible, can be applied; we only need the measurements to be related to the state vector.

This work has its foundations in a previous project, completely implemented in MATLAB by the author and a colleague (Federico Bonelli) during a course of computational fluid dynamics at Politecnico di Milano. In turn, that project consisted in reproducing the results found in [SFP03], regarding non-linear blood flow simulations. The methods implemented during that project have then been being slightly modified and embedded into a surrounding structure, then used in this thesis in the extended Kalman filter and as a "real case" to produce measurements.

The linearised models we use are generated from those, but have been heavily manipulated in this thesis in order to reach a form that is compatible with Kalman filtering. The choice of using a Kalman filter to estimate the parameters is inspired by [MCT08], and has yielded three different Kalman filtering methods. These have entirely been coded from scratch in MATLAB by the author for this thesis, or in a small part for the previous project. The measurements are all generated synthetically, by inserting the real value of the parameters into the models.

As for the structure of this document, the first chapter will introduce the model that simulates bloodflow in a bifurcation and its numerical approximation through a Discontinous Galerkin method. The second chapter describes the Kalman filter, from its Bayesian foundations to how the algorithm is structured and derived. Three different cases have been developed: the classical Kalman filter, the time-varying matrix Kalman filter and finally the extended Kalman filter.

Chapter three merges the previous two, defining a state-parameter estimation procedure, based on our bloodflow case. Additionally, it details the linearisation of the fluxes. Finally, the fourth chapter presents the results of the study, in the different cases. After having tackled some specific problems that are related to the linearised and time-varying matrix models, good stability and convergence properties are found in nearly all cases we have studied. The estimation procedure yields accurate values for the $\beta$s even if we only use measurements on the area, and even if the measurements are temporally distant one from another.

# Chapter 1

# Blood flow simulation

## 1.1 Model

In order to simulate the blood flow we have chosen to use the following 1D model, which is obtained by integrating the Navier-Stokes equations on the transverse section of the artery:

$$\begin{cases} \frac{\partial A}{\partial t} + \frac{\partial Q}{\partial x} = 0 \\ \frac{\partial Q}{\partial t} + \frac{\partial}{\partial x}\left(\frac{Q^2}{A}\right) + \frac{A}{\rho}\frac{\partial p}{\partial x} = 0 \end{cases} \tag{1.1}$$

where $A(x,t)$ is the area of the vessel section, $Q(x,t)$ the mass flux through this surface, $\rho$ the density of the fluid (that we assume to be constant, equal to 1.021) and $p$ the average pressure on the section. In order to solve these two equations in three variables we are forced to add a relationship between two of the variables. In particular we have chosen to relate the area and the pressure through the following:

$$p = p_{ext} + \beta(\sqrt{A} - \sqrt{A_0}) \tag{1.2}$$

having represented the equilibrium area of the vessel by $A_0 = A_0(x)$. This value is reached when the vessel is empty, which corresponds to having a pressure that is equal to the external pressure, as can easily be seen from (1.2). The external pressure $p_{ext}$ is a constant value of reference. In our case it is arbitrary, since the pressure only enters the equation (1.1) under a partial derivative. Therefore, without losing generality, we will assume that it is equal to zero.

The term $\beta$ which appears in the equation (1.2) is a parameter which is related to the elasticity of the vessel wall. It is assumed to be constant in each artery, but different from a vessel to another. However, a variation of $\beta$ in the $x$ direction can easily be inserted in the blood flow simulation, to achieve a more realistic modeling or for example to study the effects of inserting a stent.

### 1.1.1 Alternative models

The set of equations (1.1) is not the only way in which we can face the problem. Indeed, by manipulating these we can obtain equivalent systems of equations, for instance in the variables $(A, u, p)$ instead of $(A, Q, p)$. Here the variable $u$ denotes the average velocity on the section $A$, so the change in variables takes place through the relationship $Q = Au$. Therefore we obtain:

$$\begin{cases} \frac{\partial A}{\partial t} + \frac{\partial Au}{\partial x} = 0 \\ \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{1}{\rho}\frac{\partial p}{\partial x} = 0 \end{cases} \tag{1.3}$$

This system of equations can be rewritten as a general conservation law:

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} = 0 \tag{1.4}$$

Here we have defined the variable vectors $\mathbf{U}$ and $\mathbf{F}$ as follows:

$$\mathbf{U} = \begin{bmatrix} A \\ u \end{bmatrix}, \qquad \mathbf{F} = \mathbf{F}(\mathbf{U}) = \begin{bmatrix} Au \\ p_t \end{bmatrix} \tag{1.5}$$

where $p_t$ denotes the total pressure, defined as:

$$p_t = \frac{u^2}{2} + \frac{p}{\rho}$$

As we will see further on, in order to impose the border conditions and information transfer, it will be more useful to have another manipulation of (1.4) at hand. In particular we want the conservation law to be in a quasi-linear form, which can easily be attained by rewriting (1.4) in the following way:

$$\frac{\partial \mathbf{U}}{\partial t} + \overline{\mathbf{H}}\frac{\partial \mathbf{U}}{\partial x} = 0 \tag{1.6}$$

where the $\overline{\mathbf{H}}$ matrix clearly is defined as:

$$\overline{\mathbf{H}} = \begin{bmatrix} \frac{\partial \mathbf{F_1}}{\partial \mathbf{U}_1} & \frac{\partial \mathbf{F_1}}{\partial \mathbf{U}_2} \\ \frac{\partial \mathbf{F_2}}{\partial \mathbf{U}_1} & \frac{\partial \mathbf{F_2}}{\partial \mathbf{U}_2} \end{bmatrix} = \begin{bmatrix} u & A \\ \frac{\beta}{2\rho\sqrt{A}} & u \end{bmatrix}$$

Here the subscript $(i)$ in $\mathbf{F}_i$ and $\mathbf{U}_i$ denotes the extraction of the i-th component from the vector. For further details see [SFP03].

### 1.1.2 Simulation domain

The problem is solved on a domain which is made up of three arteries: the ascending aorta (artery number 1), the aortic arch (artery number 2) and the brachiocephalic artery (artery number 3). The input signal is imposed on the first node of artery 1, and then transmits to the two arteries in which this one splits.

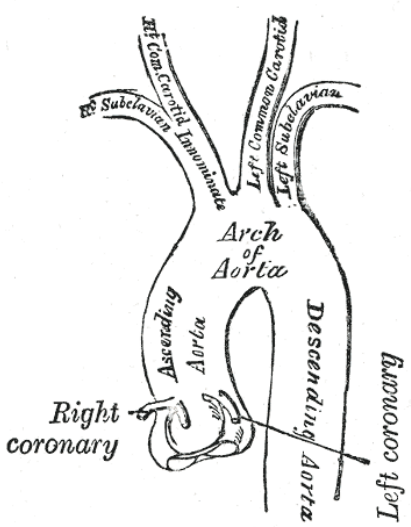


Figure 1.1: Anatomical drawing of the arteries, from [GH08]. Here the brachiocephalic artery is called "Innominate".

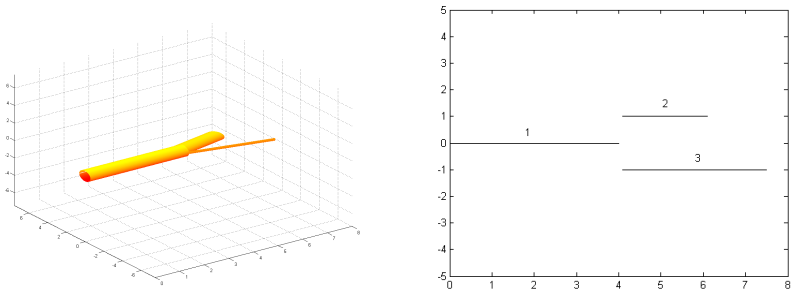


Figure 1.2: Left: the bifurcation we consider. Right: the 1D simulation domain. The proportions between vessels is kept in both images.

In particular, the numerical method we have chosen gives us the opportunity of considering the three arteries as separate domains, which only communicate through their interface conditions, in a way which we will explain further on.

### 1.1.3 Numerical method

Our problem is not a classical modeling problem on a single domain, hence we are forced to take some time to think about which method to use. Indeed we are making a mono-dimensional simulation of a naturally three-dimensional problem, which brings some difficulties in treating the bifurcation. In particular we have to face the issue of connecting the arteries: in one dimension it is impossible to have a real bifurcation. In addition we have to transfer the signal from the vessel before the bifurcation to the two that are after it in a way that resembles the actual physical behaviour as much as possible.

Furthermore in our case we have that the physical parameters vary between the vessels, and in particular $A_0$ is discontinuous across the bifurcation. Therefore it is unreasonable to impose that the solution is continuous at that point. As a consequence, in order to avoid problems at the bifurcation we will use a discontinuous Galerkin method.

Bearing in mind that this is our final goal, we multiply the conservation law (1.4) by a generic test vector function $\varphi = \begin{bmatrix} \varphi_1 \\ \varphi_2 \end{bmatrix}$ defined upon the whole domain $\Omega$ and integrate on the simulation domain to obtain its weak form:

$$\int_\Omega \frac{\partial \mathbf{U}}{\partial t} \cdot \varphi dx + \int_\Omega \frac{\partial \mathbf{F}}{\partial x} \cdot \varphi dx = 0$$

Through the additivity of the integral we can rewrite this equation in the following way, which will be of higher interest further on:

$$\sum_{e=1}^{N_e} \left( \int_{\Omega_e} \frac{\partial \mathbf{U}}{\partial t} \cdot \varphi dx + \int_{\Omega_e} \frac{\partial \mathbf{F}}{\partial x} \cdot \varphi dx \right) = 0$$

where $N_e$ denotes the total number of elements in which we have divided $\Omega$ and $\int_{\Omega_e}$ denotes the integral on the element $\Omega_e$.

At this point we can integrate the second term by parts, which yields the following form:

$$\sum_{e=1}^{N_e} \left( \int_{\Omega_e} \frac{\partial \mathbf{U}}{\partial t} \cdot \varphi dx - \int_{\Omega_e} \mathbf{F} \cdot \frac{\partial \varphi}{\partial x} dx + [\mathbf{F} \cdot \varphi]_{x_e(left)}^{x_e(right)} \right) = 0$$

where we have used the notation $x_e(left)$ and $x_e(right)$ to respectively denote the first and the last node of the element $\Omega_e$. We observe that the functions are all defined in their natural functional spaces, and haven't been numerically approximated yet. In order to take this further step we use a Galerkin method, and in particular we choose a functional space which brings to the following discontinuous Galerkin method:

find $\mathbf{U}_h \in W_h^r = \left\{ \mathbf{v}_h \in L^2(\Omega) \; s.t. \; \mathbf{v}_{h|\Omega_e} \in \mathbb{P}_r, \; \forall \Omega_e \in \mathcal{T}_h \right\}$, where $\mathcal{T}_h$ is the triangulation of the domain $\Omega$, such that

$$\begin{cases} \sum_{e=1}^{N_e} \left( \int_{\Omega_e} \frac{\partial \mathbf{U}_h}{\partial t} \cdot \varphi dx - \int_{\Omega_e} \mathbf{F}(\mathbf{U}_h) \cdot \frac{\partial \varphi}{\partial x} dx + [\mathbf{F^u} \cdot \varphi]_{x_e(left)}^{x_e(right)} \right) = 0 \\ b.c. \end{cases}$$

holds for all $\varphi \in W_h^r$.

We have denoted the flux at the border of every element by $\mathbf{F^u}$, and here we will insert the transfer of information between elements, as we will see shortly. Finally, we can counter-integrate the term we integrated by parts before. Having inserted $\mathbf{F^u}$ in the expression the two terms we obtain will generally not cancel out, and we will attain the following problem:

find $\mathbf{U}_h \in W_h^r$ such that

$$\begin{cases} \sum_{e=1}^{N_e} \left( \int_{\Omega_e} \frac{\partial \mathbf{U}_h}{\partial t} \cdot \varphi dx + \int_{\Omega_e} \frac{\partial \mathbf{F}(\mathbf{U}_h)}{\partial x} \cdot \varphi dx + [(\mathbf{F^u} - \mathbf{F}(\mathbf{U}_h)) \cdot \varphi]_{x_e(left)}^{x_e(right)} \right) = 0 \\ b.c. \end{cases} \tag{1.7}$$

holds for all $\varphi \in W_h^r$.

This final form highlights the fact that it is preferable to choose a consistent $\mathbf{F^u}$, which eliminates the last term whenever the method is applied on the exact solution. In addition it is interesting to see that now the problem can be split into $N_e$ separated problems which only interact through the flux $\mathbf{F^u}$, which gives us the possibility of parallelising the problem.

## 1.2 Matrix form

In order to follow the standard Galerkin procedures we now have to define a basis for the functional space $W_h^r$. We have chosen to consider characteristic polynomials of degree $r$, defined on the Gauss-Legendre-Lobatto nodes. They have the following trends:

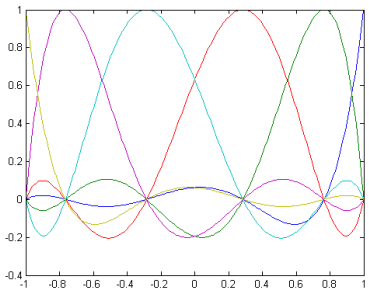


Figure 1.3: Characteristic basis functions of degree 5 on GLL nodes.

By characteristic on the i-th node we mean a function whose value is zero on all nodes except the i-th, where its value is 1. Having defined a basis,

every function of the space $W_h^r$ can be expressed as a linear combination of the basis functions. In particular this holds for $\mathbf{U}_h$ and $\varphi$, that can be decomposed as

$$\mathbf{U}_h(t,x) = \sum_{j=1}^{r+1} \mathbf{u}_j(t)\Psi_j(x)$$

$$\varphi(x) = \sum_{j=1}^{r+1} \varphi_j \Psi_j(x)$$

where $\Psi_j(x) = \begin{bmatrix} \psi_j^1 \\ \psi_j^2 \end{bmatrix}$ denotes the j-th basis function, $\mathbf{u}_j(t)$ and $\varphi_j$ the corresponding coefficients. Since the modified conservation law (1.7) must hold for every function $\varphi \in W_h^r$, just as in standard Galerkin methods we impose the law only to hold for every basis function. This does not bring to a loss of generality, since all operations involving $\varphi$ in (1.7) are linear, meaning that if it holds for all basis functions it will also hold for all linear combinations of these.

Substituting $\mathbf{U}_h$ with its expansion $\mathbf{U}_h^d$ upon the basis we reach the matrix form of the problem, written separately on each $\Omega_e$ of the triangulation $\mathcal{T}_h$:

$$\mathbf{M}\frac{\partial \mathbf{U}_h^d}{\partial t} + \mathbf{S}\mathbf{H}\mathbf{U}_h^d + \mathbf{H}_{\partial\Omega_e}\mathbf{U}_h^d + \mathbf{P}\mathbf{F}_h^{\mathbf{u}} = \mathbf{0} \tag{1.8}$$

All the terms of this equation will be explained in detail in the following subsections. In this equation the matrices have to multiply $\mathbf{U}_h^d = \begin{bmatrix} \mathbf{A}_h^d \\ \mathbf{u}_h^d \end{bmatrix}$, where now $\mathbf{A}_h^d$ and $\mathbf{u}_h^d$ have $r+1$ components each. As a consequence all matrices have to be of the size $2(r+1) \times 2(r+1)$, and virtually partitioned into four blocks of equal size, $(r+1) \times (r+1)$. Every block shows how one of the variables $(A, u)$ influences one of the equations in (1.4). In $\mathbf{M}$ and $\mathbf{S}$, as we will see shortly, there is no difference between the blocks, so we will call each submatrix $\mathbf{m}$ or $\mathbf{s}$ and will drop the superscripts of the scalar basis functions $\psi_i^1$ and $\psi_i^2$. The resulting matrices will be structured as follows:

$$M = \begin{bmatrix} \mathbf{m} & 0 \\ 0 & \mathbf{m} \end{bmatrix} \qquad S = \begin{bmatrix} \mathbf{s} & 0 \\ 0 & \mathbf{s} \end{bmatrix}$$

As for $\mathbf{H}$ and $\mathbf{H}_{\partial\Omega}$ this does not hold, and we will be forced to write the matrix as a block matrix made up of four different submatrices.

### 1.2.1 The mass matrix

The submatrix $\mathbf{m}$ is a mass matrix, with the components $\mathbf{m}_{ij} = \int_{\Omega_e} \psi_j \psi_i dx$. With the current choice of basis and quadrature rule, it becomes diagonal, since the basis functions are characteristic. In order to compute the value of this integral we use a GLL quadrature rule, of order $N = r$:

$$\int_{\Omega_e} \psi_j \psi_i dx \approx \sum_{k=1}^{N+1} w_k \psi_j(x_k)\psi_i(x_k) = \sum_{k=1}^{N+1} w_k \delta_{jk}\delta_{ik} \tag{1.9}$$

where $x_k$ are the GLL nodes and $w_k$ the corresponding weights, defined as $w_i = \int_{\Omega_e} \psi_i dx$. In the notation, $\delta_{jk}$ and $\delta_{ik}$ are Kronecker deltas, and show that the i-th basis function is null on all nodes except the i-th, where its value is 1. The result of the integral approximation will only be different from zero if $i = j$, as to say if we are on the diagonal of the matrix $\mathbf{m}$.

It is known that the GLL quadrature rule gives an exact result if the polynomial is at most of degree $2N-1$ if we use $N+1$ nodes. Thus, in our case we are actually approximating, since both $\psi_i$ and $\psi_j$ are of degree $N$.

In order to speed up the computation it is a common choice to work on a reference element, for instance such that $\Omega_{ref} = \begin{bmatrix} -1 & 1 \end{bmatrix}$, and then transfer the result on the actual domain. By doing this, the values on the diagonal of the $\mathbf{m}$ matrix on the reference element, as can be seen in the equation 1.9, will simply be the quadrature weights $\hat{w}_i = \int_{-1}^{1} \hat{\psi}_i dx$, where $\hat{\psi}_i$ denotes the basis on the reference element. Moreover, we will not be forced to change them throughout the simulation.

### 1.2.2 Transfer from the reference element to the current element

In order to get from the value of $\mathbf{m}$ that we have attained on the reference element to the value which is found on the real domain, we have to take some facts into consideration:

- the reference element has $\xi \in \begin{bmatrix} -1 & 1 \end{bmatrix}$ as its variable, and we have to compute $\int_{-1}^{1} \hat{\psi}_i(\xi)\hat{\psi}_j(\xi)d\xi$

- the element $\Omega_e$ of the domain $\Omega$ has $x \in \begin{bmatrix} x_{e0} & x_{eN} \end{bmatrix}$ as its variable, and the integral is $\int_{x_{e0}}^{x_{eN}} \psi_i(x)\psi_j(x)dx$

We can observe that the basis functions are different in these two cases. This is easily understandable, since the domain has been "stretched" from one case to the other, while the basis functions have to stay characteristic on the nodes, which have moved. However, the trend of these functions is the same, since their definition is unambiguous once we have defined the nodes. As a consequence we can state that $\hat{\psi}_i(\xi) = \psi_i(x(\xi)) \quad \forall i$ .

The relationship between $\xi$ and $x$ is easy to find: $x = \frac{(\xi+1)}{2}(x_{eN} - x_{e0})$, which yields the relationship between the differentials: $dx = \frac{(x_{eN}-x_{e0})}{2}d\xi$. This justifies the following manipulations:

$$\begin{aligned} \int_{x_{e0}}^{x_{eN}} \psi_i(x)\psi_j(x)dx &= \tfrac{(x_{eN}-x_{e0})}{2} \int_{-1}^{1} \psi_i(x(\xi))\psi_j(x(\xi))d\xi \\ &= \tfrac{(x_{eN}-x_{e0})}{2} \int_{-1}^{1} \hat{\psi}_i(\xi)\hat{\psi}_j(\xi)d\xi \end{aligned}$$

Therefore, to transfer the value from the reference element to $\Omega_e$ we only have to multiply $\mathbf{m}$ by half the length of $\Omega_e$.

If instead we have to transfer the value of an integral like $\int_{\Omega_e} \frac{d\psi_j}{dx}\psi_i dx$, as soon will be the case, we also have to find a way to convert the derived term:

$$\frac{d\psi_j(x(\xi))}{dx} = \frac{d\hat{\psi}_j(\xi)}{d\xi}\frac{d\xi}{dx} = \frac{2}{(x_{eN} - x_{e0})} \cdot \frac{d\hat{\psi}_j(\xi)}{d\xi}$$

By following the same line of thought as above, we find out that we have to multiply and divide by $\frac{2}{(x_{eN}-x_{e0})}$, so the value is actually the same on the reference element and the real one, and it is not necessary to convert it.

### 1.2.3 The spatial derivation matrix

The submatrix $\mathbf{s}$, made up of the components $\mathbf{s}_{ij} = \int_{\Omega_e} \frac{d\psi_j}{dx}\psi_i dx$, can be obtained from the differentiation matrix $\mathbf{d}$ through the operation $\mathbf{s} = \mathbf{md}$. Indeed, $\mathbf{d}$ has the components $\mathbf{d}_{ij} = \frac{d\psi_j}{dx}(x_i)$, as to say the value of the spatial derivative of the j-th basis function in the i-th node, and we can construct the following component-by-component proof:

$$\begin{aligned}(\mathbf{md})_{ij} &= \sum_{k=1}^{r+1} m_{ik}d_{kj} = \sum_{k=1}^{r+1} \int_{\Omega_e} \psi_i\psi_k dx \frac{d\psi_j}{dx}(x_k) \\ &= \int_{\Omega_e} \psi_i \sum_{k=1}^{r+1} \left(\psi_k \frac{d\psi_j}{dx}(x_k)\right) dx = \int\limits_{\Omega_e} \psi_i \frac{d\psi_j}{dx} dx = \mathbf{s}_{ij}\end{aligned}$$

The next to last step is justified by the fact that the function $\frac{d\psi_j}{dx}$ is of degree $N - 1 \leq N$, and is therefore exactly represented by its projection upon the basis.

### 1.2.4 The flux definition matrix

The matrix $\mathbf{H}$ that figures in (1.8) is the Galerkin version of the matrix $\overline{\mathbf{H}}$ that we had in the equation (1.6). Indeed, now instead of the vector $\mathbf{U} = \begin{bmatrix} A \\ u \end{bmatrix}$ we have the vector $\mathbf{U}_h^d = \begin{bmatrix} \mathbf{A}_h^d \\ \mathbf{u}_h^d \end{bmatrix}$, and we have to multiply $\mathbf{A}_h^d$ and $\mathbf{u}_h^d$ adequately to compute the fluxes. Writing these operations as matrix multiplications may seem excessively complicated, since calculating the fluxes is a punctual operation, and we could simply write a function which finds the numerical fluxes at each node. However this way of facing the problem will turn out to be of fundamental importance further on.

$\overline{\mathbf{H}}$ was structured as follows:

$$\overline{\mathbf{H}} = \begin{bmatrix} u & A \\ \frac{\beta}{2\rho\sqrt{A}} & u \end{bmatrix}$$

In order to obtain a similar result, we build the matrix $\mathbf{H}$ in this way:

$$\mathbf{H}(\mathbf{U}_h^d) = \mathbf{H} = \begin{bmatrix} \mathbf{diag}(\mathbf{u}_h^d) & \mathbf{diag}(\mathbf{A}_h^d) \\ \mathbf{diag}(\frac{\beta}{2\rho\sqrt{\mathbf{A}_h^d}}) & \mathbf{diag}(\mathbf{u}_h^d) \end{bmatrix}$$

where **diag** does not denote the extraction of the diagonal, but the construction of a diagonal square matrix, with a number of rows (and columns) equal to the length of the vector, and with the values of the vector on the diagonal.

### 1.2.5 The border flux matrix

The matrix $\mathbf{H}_{\partial\Omega_e}$ in equation (1.8) comes from the expression $[-\mathbf{F}(\mathbf{U}_h)\cdot\varphi]_{x_e(left)}^{x_e(right)}$. This matrix is very sparse, since it represents the border values. Indeed the only rows that are not equal to zero are those that correspond to the fluxes at the first and last node (each has two fluxes), and the only non-zero columns are those that multiply the area and the velocity of the first and last node. The sparsity pattern is as follows:

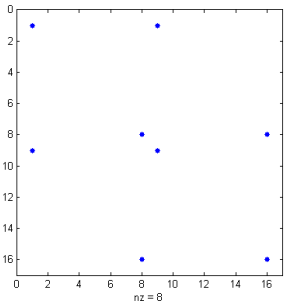


Figure 1.4: Sparsity pattern of the matrix $H_{d\Omega_e}$. This figure shows an example with 9 nodes on one vessel.

In considering the matrix $\mathbf{H}_{d\Omega_e}$ we have not included $\mathbf{F^u}$, since the latter involves not only the element on which we are working, but also the bordering ones, as we will see in depth further on. An alternate strategy to using the separate elements $\Omega_e$ is that of considering the whole domain, which brings us to construct matrices thrice the size of those we have seen so far. This gives us the possibility of putting vessel interactions in matrix notation too.

This procedure will be mandatory when we arrive to the implementation of the Kalman filter, but has several drawbacks when we simulate the blood flow. Indeed, as we will see, $\mathbf{F^u}$ is calculated through the resolution of a set of non-linear equations, which could be manipulated into a quasi-linear form involving the values at border nodes. However this is only a complicated way of translating what the MATLAB function `fsolve` does excellently, and adds nothing to our reasoning, since in the Kalman filter case we will have linearised this set, and matrix notation will be much easier. In addition, creating a global matrix has another major disadvantage, since the problem cannot be parallelised as easily anymore.

Indeed, this means that if we consider a larger tree of arteries than a single bifurcation we cannot simply solve the problem on different vessels concurrently. Anyhow, the matrix is "almost" block diagonal, so operations can be split on several processors with little communication between them.

### 1.2.6 The upwind flux matrix

Finally, the term $\mathbf{PF}_h^{\mathbf{u}}$ comes from the term $[\mathbf{F}^{\mathbf{u}} \cdot \varphi]_{x_e(left)}^{x_e(right)}$, and simply is the expansion of the vector made up of the values of $\mathbf{F}^{\mathbf{u}}$ at the borders (the way in which these are calculated will soon be explained in detail) in order to make it long $2r$. The fact that we only have the values at the borders is justified by the scalar product, which will only be different from zero for $\varphi_i$ such that $\varphi_i(x_e(right)) \neq 0$ or $\varphi_i(x_e(left)) \neq 0$. The basis functions for which this holds are the first and the last, so $\mathbf{PF}_h^{\mathbf{u}}$ has to be a vector of only zeros except the first and last entries. If we build $\mathbf{F}_h^{\mathbf{u}}$ this way:

$$\mathbf{F}_h^{\mathbf{u}} = \begin{bmatrix} F_1(x_e(left)) \\ F_2(x_e(left)) \\ F_1(x_e(right)) \\ F_2(x_e(right)) \end{bmatrix}$$

the matrix $\mathbf{P}$ has to be as follows:

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \mathbf{0}_{r-2} & \mathbf{0}_{r-2} & \mathbf{0}_{r-2} & \mathbf{0}_{r-2} \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ \mathbf{0}_{r-2} & \mathbf{0}_{r-2} & \mathbf{0}_{r-2} & \mathbf{0}_{r-2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.10}$$

where $\mathbf{0}_{r-2}$ denotes a vector long $r-2$ made up of all zeros.

## 1.3 Time discretisation

The problem (1.8) is still continuous in time. It clearly cannot be solved as it is, but we have to set up a time advancing scheme. For the blood flow simulation, if we were to follow [SFP03], we should choose a second order Adams-Bashforth method. This method is completely explicit, and for $\frac{\partial U}{\partial t} = f(t, U)$ uses the following formula:

$$U_{n+2} = U_{n+1} + \frac{3}{2}\triangle t \cdot f(t_{n+1}, U_{n+1}) - \frac{1}{2}\triangle t \cdot f(t_n, U_n)$$

However this approach does not fit our needs completely. In fact we must bear in mind our final purpose: finding a model to insert in the Kalman filter. As we will find out further on, the Kalman filter is a Markov model, meaning that the current state can only depend on the previous state, and not from values further back in time. Because of this, we have chosen to use a forward Euler method instead:

$$U_{n+1} = U_n + \triangle t \cdot f(t_n, U_n)$$

This of course reduces the time accuracy to first order.

## 1.4 Boundary conditions

If we only had one vessel, with one element, the resolution of (1.8) with some appropriate physical boundary conditions would be enough. However our case is slightly more complicated, with different kinds of boundaries, and on each of these we have to bear in mind that we want the equations to be physically reasonable.

Indeed, in addition to the bifurcation issue, on each artery we can use an arbitrary number of elements. The discontinuous Galerkin approach we have chosen lets us treat every element separately from the others, creating additional boundaries, and $\mathbf{F^u}$ ensures the information transfer between elements. The superscript $\mathbf{u}$ denotes the fact that this flux is calculated in an upwind manner, the meaning of which will soon be clear. This strategy has to be applied to our three types of boundaries: physical boundaries of $\Omega$, the boundary between three elements in a bifurcation and the boundary between two elements on the same vessel.

### 1.4.1 Bifurcation conditions

Whenever the pipe in which a fluid is flowing bifurcates, the mass flux splits up. The way in which it divides depends on several variables, for instance the pipe diameters, the angle at which they are connected and how the pipe network is made up downstream. However, we can always write down a mass flux conservation law:

$$A_p u_p = A_{ch1} u_{ch1} + A_{ch2} u_{ch2} \tag{1.11}$$

where the subscript $p$ denotes the parent vessel, while $ch1$ and $ch2$ are the two child arteries.

In addition, if we suppose not to have any friction loss, two other equations can be found by imposing continuity of total pressure between the parent vessel and the children:

$$\frac{u_p^2}{2} + \frac{\beta_p}{\rho}(\sqrt{A_p} - \sqrt{A_{0p}}) = \frac{u_{ch1}^2}{2} + \frac{\beta_{ch1}}{\rho}(\sqrt{A_{ch1}} - \sqrt{A_{0ch1}}) \tag{1.12}$$

$$\frac{u_p^2}{2} + \frac{\beta_p}{\rho}(\sqrt{A_p} - \sqrt{A_{0p}}) = \frac{u_{ch2}^2}{2} + \frac{\beta_{ch2}}{\rho}(\sqrt{A_{ch2}} - \sqrt{A_{0ch2}}) \tag{1.13}$$

this assumption brings the equation (1.4) to hold in stationary conditions ($A = A_0$, $u = u_0$) not only on points inside the elements, but also on the point where the vessels bifurcate.

The three equations we have so far have six variables, the area and velocity at each side of the bifurcation, and are therefore not sufficient to find an unambiguous solution. In order to face this issue we can add some equations that derive from mathematical considerations.

Having a set of conservation laws as in (1.4), we can define the characteristic variables, which are obtained from the $\overline{\mathbf{H}}$ matrix of the quasi-linear form. Indeed, the eigenvalues and the corresponding left eigenmatrix of $\overline{\mathbf{H}}$ are:

$$\lambda_{1,2} = u \pm \sqrt{\tfrac{\beta}{2\rho}} A^{1/4} \qquad \mathbf{L} = \begin{bmatrix} \sqrt{\frac{\beta}{2\rho}} A^{-3/4} & 1 \\ -\sqrt{\frac{\beta}{2\rho}} A^{-3/4} & 1 \end{bmatrix}$$

and the characteristic variables $\mathbf{W} = \mathbf{W}(A,u) = \begin{bmatrix} W_1 \\ W_2 \end{bmatrix}$ are found by integrating the equation that defines them: $\frac{\partial \mathbf{W}}{\partial \mathbf{U}} = \mathbf{L}$. Thus we find the following:

$$W_1 = u + 4A^{1/4}\sqrt{\frac{\beta}{2\rho}} \tag{1.14}$$

$$W_2 = u - 4A^{1/4}\sqrt{\frac{\beta}{2\rho}} \tag{1.15}$$

The main advantage in using characteristic variables is that they give the opportunity of decoupling the equations in the set. Indeed, if we multiply $\mathbf{L}$ at the left of (1.6) and exploit the fact that $\frac{\partial \mathbf{W}}{\partial x} = \frac{\partial \mathbf{W}}{\partial \mathbf{U}} \cdot \frac{\partial \mathbf{U}}{\partial x}$ and that $\mathbf{L}$ is an eigenmatrix we find the following:

$$\frac{\partial W_1}{\partial t} + \lambda_1 \frac{\partial W_1}{\partial x} = 0$$

$$\frac{\partial W_2}{\partial t} + \lambda_2 \frac{\partial W_2}{\partial x} = 0$$

*In vivo* measurements have shown that the flux generally stays subsonic, as to say $|u| < \sqrt{\frac{\beta}{2\rho}} A^{1/4}$, so one eigenvalue is positive whereas the other is negative. We can therefore assert that $W_1$ travels at velocity $\lambda_1 > 0$, which means rightwards on our domain, while $W_2$ travels leftwards at velocity $\lambda_2 < 0$. As a consequence we can state that at the boundary the upwinded $W_1$ is the one coming from the element to the left of the boundary, while the upwinded $W_2$ is the one coming from the right.

This gives us the possibility to write down the three equations we were missing, simply by imposing the conservation of the upwinded characteristic variables over the boundary. In particular we want the parent's $W_1$ to be conserved, along with both children's $W_2$. This translates mathematically into the following:

$$W_1 = u_p + 4A_p^{1/4}\sqrt{\frac{\beta_p}{2\rho}} \tag{1.16}$$

$$W_{21} = u_{ch1} - 4A_{ch1}^{1/4}\sqrt{\frac{\beta_{ch1}}{2\rho}} \tag{1.17}$$

$$W_{22} = u_{ch2} - 4A_{ch2}^{1/4}\sqrt{\frac{\beta_{ch2}}{2\rho}} \tag{1.18}$$

In order to understand how to use these equations, we have to bear in mind that the resolution of the set of equations at the bifurcation aims at modifying the values we have found through the finite elements methods. This means that at each time-step we have the values of $(A, u)$ on the boundary for each vessel. These values are used to compute the parent's $W_1$ and the children's $W_2$, which will then be inserted as constants in equations (1.16)-(1.18), while the variables are $(A_p, u_p)$, $(A_{ch1}, u_{ch1})$, $(A_{ch2}, u_{ch2})$.

Since we now have six equations (1.11)-(1.13) (1.16)-(1.18) in six variables we can solve the set, finding new values to insert into the boundary fluxes $\mathbf{F^u}$. Through this procedure we have that if a signal arrives at the parent's boundary, it will be transferred to the children realistically. The fluxes are computed through the definitions (1.5), with each vessel using its own modified values (e.g. the parent artery uses the new values of $(A_p, u_p)$ we have found). The set can be solved through Newton's method.

### 1.4.2 Interface conditions between elements

Since we are looking for possible discontinuous solutions, we are also forced to find a realistic way to transfer signals from one element to the next. This can be done in a fashion which is very similar to that used for the bifurcations, with the difference that we now only have to take two elements into account. The set we have to solve is the following:

$$A_1 u_1 = A_2 u_2$$

$$\frac{u_1^2}{2} + \frac{\beta_1}{\rho}(\sqrt{A_1} - \sqrt{A_{01}}) = \frac{u_2^2}{2} + \frac{\beta_2}{\rho}(\sqrt{A_2} - \sqrt{A_{02}})$$

$$W_1 = u_1 + 4A_1^{1/4}\sqrt{\frac{\beta_1}{2\rho}}$$

$$W_2 = u_2 - 4A_2^{1/4}\sqrt{\frac{\beta_2}{2\rho}}$$

Exactly as before, the values we find through this method have to be inserted into the fluxes $\mathbf{F^u}$ in the corresponding nodes.

### 1.4.3 Discussions on the flux $\mathbf{F}_h^{\mathbf{u}}$

It is useful to check whether the scheme is able to produce a constant solution. Let's therefore observe what happens at the bifurcation if we find all arteries in the state $(A_0, 0)$. In this case the equation (1.11) automatically holds with the original data. The two pressure continuity equations (1.12) and (1.13) are satisfied trivially, and the conservation of the characteristic variables obviously holds. We therefore do not need to solve the set of equations, since we already are at an equilibrium point. As a consequence $\mathbf{F}^{\mathbf{u}}$ will be equal to $-\mathbf{H}_{\partial\Omega_e}\mathbf{U}_h$, and there will be no changes (in area and velocity) at the bifurcation, which does not move from the equilibrium.

These arguments also hold for boundaries between elements on the same vessel, where we can even take a further step. Indeed we know that the real solution is continuous over these boundaries, and that $\beta$ generally does not have points of discontinuity (even in the case of stents we will model the discontinuity with a smooth function). This means that for the real solution the bifurcation conditions are automatically satisfied, and once again $\mathbf{F}^{\mathbf{u}}$ and $-\mathbf{H}_{\partial\Omega_e}\mathbf{U}_h$ cancel out.

### 1.4.4 Conditions on the physical boundary

The borders (the first node of vessel 1 and the last node of vessels 2 and 3) of the physical domain $\Omega$ can also be considered analogously to interfaces between elements, and will be treated as such to impose the desired input and output conditions. Even in this case we will use upwind information, and in particular we will impose the boundary conditions weakly. This means that the values the variables have at the border will not be exactly the ones we impose, but will derive from these.

In this case we can still hold on to the considerations we have made above: $W_1$ travels leftwards, while $W_2$ travels rightwards. This means that we only can impose one of these variables on each border, while the other information we need must come from inside the domain. In order to bring this into light let us focus on the input border. If we could impose both $W_1$ and $W_2$ we would have that at one point, the first node, we have two different values of $W_2$: the one coming from inside the domain and the one we impose. This brings to a point of discontinuity of $W_2$, which could bring a reflection of the wave carrying this variable. This leads to an oscillatory behaviour, which is only numerical, and therefore unwanted.

In order to avoid this problem, at the input we only impose variables concerning $W_1$, while $W_2$ is calculated from the values $(A_{root}, u_{root})$ inside the domain. The subscript "root" is used to denote the fact that the input signal is imposed on the root of the tree of arteries (in this case the tree only has two leaves, but a more general case can be envisaged, as has been done for the bloodflow simulation in the project that precedes this thesis). In addition it is important that $W_2$ is conserved during the border condition imposition, as to say that the values $(A^*, u^*)$ that we insert in $\mathbf{F}^{\mathbf{u}}$ have to give the same $W_2$ as the one we had originally.

From equations (1.14) and (1.15) we can obtain a way to calculate the

velocity and the area:

$$u = \frac{W_1 + W_2}{2} \qquad A = \left( \frac{W_1 - W_2}{8\sqrt{\frac{\beta_{root}}{2\rho}}} \right)^4 \tag{1.19}$$

If for example we want to impose $A_{bc}$ as an input signal, we also have to assign a value to the corresponding $u$. We will choose the unperturbed value $u_{initial}$, for reasons that will be clear further on. We then calculate $W_1$ from these values, along with $W_2$ from the values inside the domain, and use equations (1.19) to find the values at the border $(A^*, u^*)$. These can then be inserted in $\mathbf{F^u}$ in the usual manner.

As we stated before, it is important that the outwards-traveling wave exits the domain without being reflected. The fact that this holds can easily be proven:

$$W_2^{new} = u^* - 4A^{*1/4}\sqrt{\frac{\beta_{root}}{2\rho}} = \frac{W_1 + W_2}{2} - 4\sqrt{\frac{\beta_{root}}{2\rho}}\frac{W_1 - W_2}{8\sqrt{\frac{\beta_{root}}{2\rho}}} = W_2$$

At the output the condition only differs slightly from this one. Indeed we have no precise signal to impose, we only want the flow to continue steadily, without reflections. This can be attained by imposing the conservation of the outgoing characteristic variable, $W_1$. At the outside of the output section we suppose that we are in an equilibrium state, for instance $(A_{initial}, u_{initial})$. This choice is very important, since any other choice of values of $(A, u)$ creates a signal at the output boundary, which then travels upstream.

From equations (1.19) we can obtain $(A^*, u^*)$ exactly as before. However we have to bear in mind that now $(A_{initial}, u_{initial})$ is found at the right of the domain, so must be inserted in $W_2$ instead of $W_1$.

Now it is important that $W_1$ is not reflected:

$$W_1^{new} = u^* + 4A^{*1/4}\sqrt{\frac{\beta_{leaf}}{2\rho}} = \frac{W_1 + W_2}{2} + 4\sqrt{\frac{\beta_{leaf}}{2\rho}}\frac{W_1 - W_2}{8\sqrt{\frac{\beta_{leaf}}{2\rho}}} = W_1$$

Let's consider the calculation of $u^*$ in more detail:

$$u^* = \frac{u_{initial} + u_{leaf}}{2} + 2\sqrt{\frac{\beta_{leaf}}{2\rho}}\left(A_{leaf}^{1/4} - A_{initial}^{1/4}\right) \tag{1.20}$$

where the subscript "leaf" denotes the variable on the border, calculated from the numerical method on the corresponding element.

Here we see that if we start our simulation from an initial condition $(A_{initial}, u_{initial})$ and we impose values that are different from these outside the domain, the border value will change. Indeed, equation (1.20) will not yield $u_{initial}$ as we expect it to do (the input signal has not reached the end

of the domain yet, so we expect it to stay unperturbed), and equation (1.19) then modifies $A^*$ too as a consequence. Since our conservation law causes waves to travel both upstream and downstream, this perturbation will act exactly as an input signal and influence neighboring nodes. In addition, in this case, if we place the system in a constant initial condition and don't vary the input signal, the wave propagating backwards from the output will after a while arrive at the bifurcation, and influence the other vessels. In this way, the whole system will be perturbed by a simple misfitting choice of values.

## 1.5 Input signal

The input somehow has to simulate the variations in pressure, area or velocity that a heartbeat causes. We have chosen to use a simple signal, a half sine followed by a constant part, and have chosen to impose this signal on the area. The signal is imposed weakly, in the way we saw in section (1.4.4). The first choice, which well suited some cases but was inadequate in others, was to use $A_0$ of the root vessel for the constant part of the signal. The equation of the input is:

$$A(t) = max\{A_0, \quad A_0 + a_1 A_0(sin(2\pi t + a_2) - a_3)\}$$

Where $a_1 = 0.597$, $a_2 = 0.628$ and $a_3 = 0.588$.
This gives the following trend:

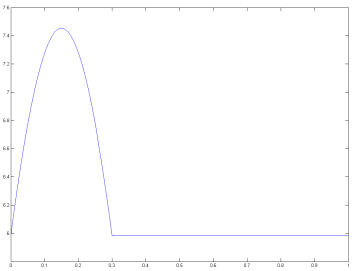

Figure 1.5: First envisaged input signal

This kind of input is not realistic, since we suppose that between heartbeats the artery is completely empty, with a pressure equal to the external pressure. Additionally, if we use this function we have problems applying the linearised Kalman filter, as we will see further on. Therefore, a second input signal can be envisaged.

There are certain properties that we want our input to have. To begin with, it is preferable not to have input functions with discontinuities, since they are physically unrealistic. To avoid these discontinuities it is necessary to ensure compatibility between initial and boundary data. This simply means that if the input signal starts at the value $A_1$, the initial condition

on the first node of the root vessel also has to be $A_1$. If we do not impose this, we would have a transient, where the system responds to the discontinuity, which propagates along the vessel. On the contrary, if we chose compatible conditions the behaviour would be smooth.

To construct our input function we simply add a constant to the previous one. In particular we have chosen to impose the sine wave to depart from the area that corresponds to $p_t = 70\, mmHg$, with $u = 0$. This value represents a realistic pressure in-between heartbeats (i.e. the constant part). As we have seen before, this value is tightly related to the initial conditions we will impose.

To find the initial conditions we give the (arbitrary) values $Q = 0$ and $p = 70\, mmHg$ to the two fluxes on each vessel. The fact that the mass flux cancels out and the pressure is equal on all three elements that are connected through the bifurcation means that the bifurcation conditions are automatically verified. In addition, if the initial condition is spatially constant on each artery (meaning $\frac{\partial \mathbf{F}}{\partial x} = 0$) and we have the same value $(A_{initial}, u_{initial})$ outside the terminal elements, we have that the system is at rest. To find which area corresponds to these conditions we simply use the definition (1.2) and the fact that $Q = Au = 0 \implies u = 0$:

$$A_e = \left(\frac{70}{\beta} + \sqrt{A_0}\right)^2$$

In the units we have chosen, $[A] = cm^2$, $[u] = \frac{cm}{s}$, $[\rho] = \frac{g}{cm^3}$, $[\beta] = \frac{dyn}{cm^3}$ the pressure cannot be used in mmHg, but must be converted into $\frac{dyn}{cm^2}$.

The resulting input has the same trend as the previous one, but a different base value. The equation of this function is:

$$A(t) = max\{A_e, \quad A_e + 0.597A_e(sin(2\pi t + 0.628) - 0.588)\}$$

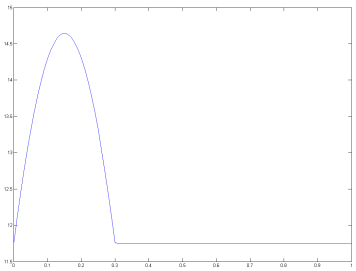

Figure 1.6: Alternative input function

## 1.6 Results

Our simulations are made with a time-step of $\Delta t = 10^{-5}$, in order to ensure that we respect the CFL condition, and we simulate ten heartbeats (i.e. $T_{final} = 10$). The polynomials are chosen of degree nine. The values of $\beta$ and $A_0$, along with the lengths of the vessels, are the following:

| | $\beta$ | $A_0$ | Length |
|---|---|---|---|
| Artery 1 | 97000 | 5.983 | 4 cm |
| Artery 2 | 87000 | 5.147 | 2 cm |
| Artery 3 | 233000 | 1.219 | 3.4 cm |

Table 1.1: Values of the parameters for each vessel

The results are only shown for the first heartbeat, since all periods will have the exact same behaviour. Indeed, there will not be any difference between the first heartbeat and the following ones, since the initial conditions are compatible with the input function and the boundary conditions . We can notice that the input trend is kept quite precisely, with only small overshoots corresponding to the sharp bend between the half-sine and the constant part.

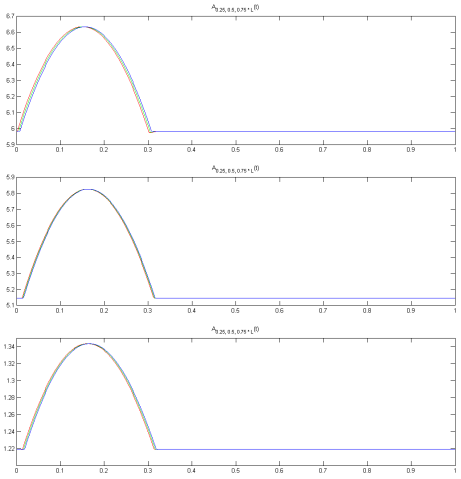

Figure 1.7: The system's response to the first input

The answer to the translated sine wave is the following. It is identical to the previous one in trend, but translated to the new base value, and stretched over the $y$ axis, since we have $A_e$ instead of $A_0$ multiplying the half-sine.

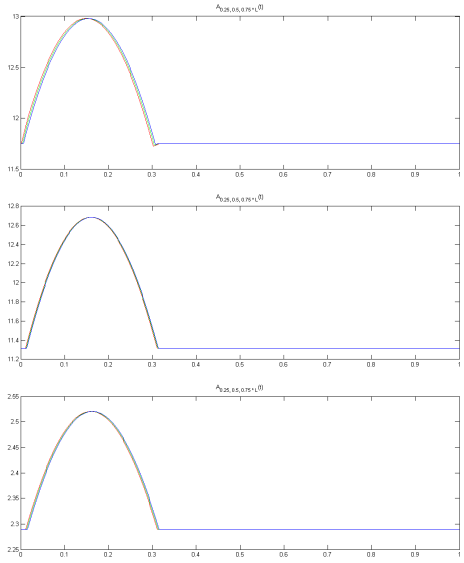

Figure 1.8: The system's response to the second input

# Chapter 2

# Kalman filter

The Kalman filter is a classical method of pursuit, used for instance in augmenting the accuracy of GPS devices and ballistic calculations. It is based upon simple Bayesian procedures, under the assumption that all considered distributions are Gaussian. In order to use it we need to have an idea of how a state vector evolves in time (prediction), and we need to measure a function of part of the actual state at some points in time in order to have a feedback on our suppositions (correction).

In our case the final goal is to find an estimate of the parameter $\beta$, so a slightly modified version of the Kalman filter will be used, in which the state has been "augmented" by making $\beta$ part of it. In this way at each time-step we estimate both the actual state and the parameter. The classical Kalman filter supposes that both the state evolution and the measurement function are linear, but this can be generalised through the Extended Kalman Filter (EKF), in which these functions are linearised at each time-step, or through other variants and generalisations.

## 2.1 Variables and equations

The Kalman filter assumes that we have a state vector $\mathbf{x} \in \mathbb{R}^n$ which evolves in the following way:

$$\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_k + \mathbf{q}_k \tag{2.1}$$

where $\mathbf{A}_{k-1} \in \mathbb{R}^{n\times n}$ is the state transition matrix, $\mathbf{q}_k \in \mathbb{R}^n$ is a noise which models the fact that our state evolution model may not be exact, $\mathbf{B}_{k-1} \in \mathbb{R}^{n\times n}$ is a matrix that multiplies the control vector $\mathbf{u}_k \in \mathbb{R}^n$. The control vector is the input of the system and must be treated independently of the state since it is a source function. In our case the input is one of the functions discussed in section 1.5, and $\mathbf{B}_{k-1}$ denotes the weak imposition of the border conditions. The subscripts of $\mathbf{A}_{k-1}$ and $\mathbf{B}_{k-1}$ denotes the fact that these matrices may vary in time.

In addition we need to have some measures $\mathbf{y}_k \in \mathbb{R}^m$, modeled as follows:

$$\mathbf{y}_k = \mathbf{H}\mathbf{x}_k + \mathbf{r}_k \tag{2.2}$$

Here $\mathbf{H} \in \mathbb{R}^{m \times n}$ denotes the measure matrix, and it being constant means that we measure the same linear combinations of the state-vector components at each time. The vector $\mathbf{r}_k$ is another noise, which models measurement errors.

If we could measure the whole state at all times the filter would clearly be of limited use: the beauty in the Kalman filter lies in the fact that few measures at few time-steps may be enough. This means that the matrix $\mathbf{H}$ may have a much smaller rank than $\mathbf{A}_{k-1}$, as to say $m \ll n$.

In order to use the Kalman filter we need to suppose that the initial distribution $\mathbf{x}_0$ from which it starts is Gaussian, and that the noises are Gaussian zero-mean:

$$\mathbf{x}_0 \sim N(\mu_0, \Sigma_0) \tag{2.3}$$

$$\mathbf{q}_k \sim N(0, \Sigma_q) \tag{2.4}$$

$$\mathbf{r}_k \sim N(0, \Sigma_r) \tag{2.5}$$

Additionally we need to suppose that the noises are not temporally correlated, as to say that $\mathbb{E}[\mathbf{q}_k \mathbf{q}_j^T] = \Sigma_q \delta_{kj}$ and $\mathbb{E}[\mathbf{r}_k \mathbf{r}_j^T] = \Sigma_r \delta_{kj}$, and that the noises and the state are mutually statistically independent, as to say $\mathbb{E}[\mathbf{x}_k \mathbf{r}_j^T] = 0 \quad \forall k, j$ and $\mathbb{E}[\mathbf{x}_k \mathbf{q}_j^T] = 0 \quad \forall k < j$ (the state clearly is correlated with "previous" noises, $k > j$, since the noise appears in the state evolution equation (2.1)). Finally the two noises are required to be mutually independent : $\mathbb{E}[\mathbf{r}_k \mathbf{q}_j^T] = 0 \quad \forall k, j$.

### 2.1.1 State and measurement distributions

As a consequence of the definitions (2.3)-(2.5), $\mathbf{y}_k|\mathbf{x}_k$ and $\mathbf{x}_k|\mathbf{x}_{k-1}$ (meaning $\mathbf{y}_k$ given $\mathbf{x}_k$ and $\mathbf{x}_k$ given $\mathbf{x}_{k-1}$ respectively), will be Gaussian too, since they are linear combinations of Gaussian distributions. The control matrix and vector are for our purposes considered as constants, since they only vary over time but are deterministic at each time-step. This means that $\mathbf{u}_k$ isn't drawn from a probability distribution, and $\mathbb{E}[\mathbf{u}_k] = \mathbf{u}_k \quad \forall k$. We have that equation (2.1) yields:

$$\mathbb{E}[\mathbf{x}_k|\mathbf{x}_{k-1}] = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_k + \mathbb{E}[\mathbf{q}_k] = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_k$$

$$Cov[\mathbf{x}_k|\mathbf{x}_{k-1}] = \mathbf{A}_{k-1} Cov[\mathbf{x}_{k-1}] \mathbf{A}_{k-1}^T + \Sigma_q$$

The same holds for the measurement equation (2.2), which gives:

$$\mathbb{E}[\mathbf{y}_k|\mathbf{x}_k] = \mathbf{H}\mathbf{x}_k + \mathbb{E}[\mathbf{r}_k] = \mathbf{H}\mathbf{x}_k$$

$$Cov[\mathbf{y}_k|\mathbf{x}_k] = \Sigma_r$$

## 2.2 Bayesian foundations of the Kalman filter

### 2.2.1 Applying Bayes' theorem

The Kalman filter is an optimal Bayesian filter, applied under Gaussian assumptions. Its aim is finding a MAP (Maximum A Posteriori) estimate for the state, which we will denote $\hat{\mathbf{x}}_k^{MAP}$. The fact that we can call a MAP estimate optimal derives from the fact that our posterior distribution will be Gaussian, thus some different definitions of *optimal* are equivalent (MAP and MMSE for instance, as is shown in Appendex A). A secondary output of our method is the posterior covariance, and thus we will have a full description of the posterior distribution.

To start with, we need to recall Bayes's theorem, which shows us how to formally find the *a posteriori* distribution $p(\mathbf{x}_k|\mathbf{Y}_k)$, having defined the vector of past measures $\mathbf{Y}_k = (\mathbf{y}_k,\ \mathbf{y}_{k-1},\ \ldots)$:

$$p(\mathbf{x}_k|\mathbf{Y}_k) = \frac{p(\mathbf{x}_k, \mathbf{Y}_k)}{p(\mathbf{Y}_k)} = \frac{p(\mathbf{x}_k, \mathbf{y}_k, \mathbf{Y}_{k-1})}{p(\mathbf{y}_k, \mathbf{Y}_{k-1})} \tag{2.6}$$

Here $p(\mathbf{x}_k, \mathbf{y}_k, \mathbf{Y}_{k-1})$ denotes the joint probability density function, and $p(\mathbf{x}_k|\mathbf{Y}_k)$ is the probability density of $\mathbf{x}_k$ given all the previous measurements $\mathbf{Y}_k$.

The second step in (2.6)simply isolates the $k$-th measurement from the set of all measurements prior to the time $k$. The numerator can be further manipulated through the probability product law to become:

$$p(\mathbf{x}_k, \mathbf{y}_k, \mathbf{Y}_{k-1}) = p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{Y}_{k-1})p(\mathbf{x}_k|\mathbf{Y}_{k-1})p(\mathbf{Y}_{k-1})$$

If we then use the fact that $\mathbf{r}_k$ is independent from previous measures and states by hypothesis (which leads to $\mathbf{y}_k|\mathbf{x}_k$ depending only from $\mathbf{x}_k$, i.e. being independent from $\mathbf{Y}_{k-1}$), we can take a further step, arriving to:

$$p(\mathbf{x}_k, \mathbf{y}_k, \mathbf{Y}_{k-1}) = p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Y}_{k-1})p(\mathbf{Y}_{k-1})$$

In this way, and by using the product law once again, we can rewrite Bayes's law as follows:

$$p(\mathbf{x}_k|\mathbf{Y}_k) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Y}_{k-1})p(\mathbf{Y}_{k-1})}{p(\mathbf{y}_k|\mathbf{Y}_{k-1})p(\mathbf{Y}_{k-1})} = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Y}_{k-1})}{p(\mathbf{y}_k|\mathbf{Y}_{k-1})}$$

### 2.2.2 Finding the optimum

In order to get a MAP estimate, we now have to find the maximum of this probability function (its argmax on $\mathbf{x}_k$). This offers the advantage that we can ignore the denominator, since it is a strictly positive value. The only two terms we have to consider are thus $p(\mathbf{y}_k|\mathbf{x}_k)$ and $p(\mathbf{x}_k|\mathbf{Y}_{k-1})$. The first one is simply

$$p(\mathbf{y}_k|\mathbf{x}_k) \sim N(H\mathbf{x}_k,\ \Sigma_r)$$

as we have seen before. The second one needs one more manipulation, since $\mathbf{x}_k$ is not directly related to $\mathbf{Y}_{k-1}$ (but $\mathbf{x}_{k-1}$ is):

$$p(\mathbf{x}_k|\mathbf{Y}_{k-1}) = p(\mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_k + \mathbf{q}_k|\mathbf{Y}_{k-1})$$

which leads to:

$$p(\mathbf{x}_k|\mathbf{Y}_{k-1}) \sim N(\mathbf{A}_{k-1}\mathbb{E}\left[\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}\right]+\mathbf{B}_{k-1}\mathbf{u}_k,\ \mathbf{A}_{k-1}Cov[\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}]\mathbf{A}_{k-1}^T+\Sigma_q)$$

In order to simplify the notation, we can define the prediction estimate $\hat{\mathbf{x}}_{k,k-1} = \mathbf{A}_{k-1}\mathbb{E}\left[\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}\right] + \mathbf{B}_{k-1}\mathbf{u}_k$. The covariance of $\mathbf{x}_k|\mathbf{Y}_{k-1}$ may also be rewritten as $\mathbf{P}_{k,k-1} = \mathbf{A}_{k-1}Cov[\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}]\mathbf{A}_{k-1}^T + \Sigma_q$, and represents the prediction covariance.

Since all probability functions we are using are Gaussian, the posterior density will also be Gaussian. However, for our purposes it is enough only to consider the numerator, which brings us to an improper density (the denominator is the constant that makes it proper):

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{Y}_k) \quad \propto \quad & \exp\left(-\frac{1}{2}\left(\left(\mathbf{y}_k - \mathbf{H}\mathbf{x}_k\right)^T \Sigma_r^{-1}\left(\mathbf{y}_k - \mathbf{H}\mathbf{x}_k\right) + \right.\right. \\ & \left.\left. + \left(\mathbf{x}_k - \hat{\mathbf{x}}_{k,k-1}\right)^T \mathbf{P}_{k,k-1}^{-1}\left(\mathbf{x}_k - \hat{\mathbf{x}}_{k,k-1}\right)\right)\right) \end{aligned}$$

This expression is obtained by simply multiplying the two Gaussian probability functions $p(\mathbf{y}_k|\mathbf{x}_k)$ and $p(\mathbf{x}_k|\mathbf{Y}_{k-1})$.

We are looking for the maximum of this function:

$$\left.\frac{\partial p(\mathbf{x}_k|\mathbf{Y}_k)}{\partial \mathbf{x}_k}\right|_{\mathbf{x}_k=\hat{\mathbf{x}}_{k,k}^{MAP}} = 0$$

Here we have chosen to use the notation $\hat{\mathbf{x}}_{k,k}^{MAP}$ to denote the MAP estimate, calculated at time $k$ through the measurements at times $t \leq k$.

By simple derivation we obtain:

$$\hat{\mathbf{x}}_{k,k}^{MAP} = \left(\mathbf{P}_{k,k-1}^{-1} + \mathbf{H}^T\Sigma_r^{-1}\mathbf{H}\right)^{-1}\left(\mathbf{H}^T\Sigma_r^{-1}\mathbf{y}_k + \mathbf{P}_{k,k-1}^{-1}\hat{\mathbf{x}}_{k,k-1}\right) \tag{2.7}$$

Now we can use Woodbury's matrix identity, which states:

$$\left(\mathbf{A} + \mathbf{BCD}\right)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}\left(\mathbf{C}^{-1} + \mathbf{DA}^{-1}\mathbf{B}\right)^{-1}\mathbf{DA}^{-1} \tag{2.8}$$

This brings us to the expression:

$$\hat{\mathbf{x}}_{k,k}^{MAP} = \hat{\mathbf{x}}_{k,k-1} + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{H}\hat{\mathbf{x}}_{k,k-1}\right) \tag{2.9}$$

where $\mathbf{K}_k \in R^{n\times m}$ denotes the Kalman gain matrix, defined as follows:

$$\mathbf{K}_k = \mathbf{P}_{k,k-1}\mathbf{H}^T\left(\Sigma_r + \mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\right)^{-1} \tag{2.10}$$

The expression of $\mathbf{K}_k$ shows that the choice of $\Sigma_r$ is not completely arbitrary. Indeed we expect $\|\mathbf{P}_{k,k-1}\|_\infty$ to decrease considerably through

the Kalman filter, so the imposition of $\Sigma_r$ must be strictly positive (we cannot take it to be too small in norm in our test cases), in order not to let the matrix inversion become too ill-conditioned. In real cases, where $\Sigma_r$ is given, we must bear this in mind, and even if the measurements are very accurate the matrix has to be invertible. All the steps through which the expression of $\hat{\mathbf{x}}_{k,k}^{MAP}$ was calculated can be found in Appendix A. The value $\hat{\mathbf{x}}_{k,k}^{MAP}$ we have found through this procedure will be the "corrected" value for the state at time $k$, and is the guess we will use for the next iteration.

It is interesting to notice that in equation (2.9) the term $(\mathbf{y}_k - \mathbf{H}\hat{\mathbf{x}}_{k,k-1})$ represents a residual of the measurement equation: the higher the difference between the estimated measure $\mathbf{H}\hat{\mathbf{x}}_{k,k-1}$ and the real measurement, the more $\hat{\mathbf{x}}_{k,k-1}$ will be modified to become $\hat{\mathbf{x}}_{k,k}^{MAP}$. The term $K_k$ takes the name of *gain* because it modulates these fluctuations.

We also need to know how to update the covariance matrix in order to continue our estimation loop. Since $\hat{\mathbf{x}}_{k,k}^{MAP}$ is obtained through a linear combination of normally distributed variables, the definition of $\mathbf{y}_k$ (equation (2.2)) leads to:

$$\mathbf{P}_{k,k} = Cov\left[\hat{\mathbf{x}}_{k,k}^{MAP}\right] = Cov\left[(I - \mathbf{K}_k\mathbf{H})\hat{\mathbf{x}}_{k,k-1}\right] + Cov\left[\mathbf{K}_k\mathbf{r}_k\right] =$$

$$= (I - \mathbf{K}_k\mathbf{H})\mathbf{P}_{k,k-1}(I - \mathbf{K}_k\mathbf{H})^T + \mathbf{K}_k\Sigma_r\mathbf{K}_k^T =$$

$$= (I - \mathbf{K}_k\mathbf{H})\mathbf{P}_{k,k-1} - \mathbf{P}_{k,k-1}\mathbf{H}^T\mathbf{K}_k^T + \mathbf{K}_k\left(\Sigma_r + \mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\right)\mathbf{K}_k^T \quad (2.11)$$

In order to simplify this expression, we can use equation (2.10): let's rename the matrix $\mathbf{S}_k = \left(\Sigma_r + \mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\right)$. Equation (2.10) then states that

$$\mathbf{K}_k = \mathbf{P}_{k,k-1}\mathbf{H}^T\mathbf{S}_k^{-1}$$

If we right-multiply this equation by $\mathbf{S}_k\mathbf{K}_k^T$ , we find out that

$$\mathbf{K}_k\mathbf{S}_k\mathbf{K}_k^T = \mathbf{P}_{k,k-1}\mathbf{H}^T\mathbf{K}_k^T$$

which means that the last to terms of (2.11) cancel out. This gives us the final expression of our updated covariance matrix:

$$\mathbf{P}_{k,k} = Cov\left[\hat{\mathbf{x}}_{k,k}^{MAP}\right] = (I - \mathbf{K}_k\mathbf{H})\mathbf{P}_{k,k-1}$$

### 2.2.3 Kalman filter algorithm

What we have described up to this point is one step of the iterative Kalman method. In order to describe the whole method we need to use an estimated probability function for $\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}$, and in particular we will use $\hat{\mathbf{x}}_{k-1,k-1}^{MAP}$ for its mean (remembering that MMSE and MAP are equivalent under Gaussian assumptions) and $\mathbf{P}_{k-1,k-1}$ for its covariance. Its density is Gaussian, because in turn it yields from an initial Gaussian distribution.

Summing up, given an initial guess $\hat{\mathbf{x}}_{0,0}^{MAP}$ and an initial covariance matrix $\mathbf{P}_{0,0}$, the steps we have to take iteratively are:

$$\begin{cases} \hat{\mathbf{x}}_{k,k-1} = \mathbf{A}_{k-1}\hat{\mathbf{x}}_{k-1,k-1}^{MAP} + \mathbf{B}_{k-1}\mathbf{u}_k & \text{(Predicting the state)} \\ \mathbf{P}_{k,k-1} = \mathbf{A}_{k-1}\mathbf{P}_{k-1,k-1}\mathbf{A}_{k-1}^T + \Sigma_q & \text{(Calculating the prediction covariance)} \\ \mathbf{S}_k = \left(\Sigma_r + \mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\right) & \text{(Calculating the measurement covariance)} \\ \mathbf{K}_k = \mathbf{P}_{k,k-1}\mathbf{H}^T\mathbf{S}_k^{-1} & \text{(Calculating the Kalman gain matrix)} \\ \hat{\mathbf{x}}_{k,k}^{MAP} = \hat{\mathbf{x}}_{k,k-1} + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{H}\hat{\mathbf{x}}_{k,k-1}\right) & \text{(Correcting the estimation)} \\ \mathbf{P}_{k,k} = (I - \mathbf{K}_k\mathbf{H})\mathbf{P}_{k,k-1} & \text{(Updating the covariance matrix)} \end{cases}$$

## 2.3 Generalising the Kalman filter

As we have seen, the classical Kalman filter has the drawback of only considering linear evolution and measurement laws. This means that if we want to use it for our blood simulation problem we need to linearise the problem, which yields results with low precision, as we will see further on. If we want to increase accuracy we can use the extended Kalman filter, which uses non-linear state evolutions, and linearises the problem at each time-step (instead of once and for all) in order to update the mean and covariance estimates.

### 2.3.1 Extended Kalman Filter (EKF)

Let's suppose that we have an evolution function

$$\mathbf{x}_k = \mathbf{a}(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{q}_k$$

and a measurement function

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{r}_k$$

In these expressions the two noise functions $\mathbf{q}_k$ and $\mathbf{r}_k$ are exactly the same as before, zero-mean Gaussian. It is important that these noises are not included in the functions $\mathbf{a}$ and $\mathbf{h}$, they need to be additive.

Next, we calculate the Jacobian of these two functions at each time-step, and evaluate them with the appropriate state estimates:

$$\mathbf{A}_{k-1} = \left.\frac{\partial \mathbf{a}}{\partial \mathbf{x}_{k-1}}\right|_{\hat{\mathbf{x}}_{k-1,k-1}, \mathbf{u}_k} \qquad \mathbf{H}_k = \left.\frac{\partial h}{\partial \mathbf{x}_k}\right|_{\hat{\mathbf{x}}_{k,k-1}}$$

These functions are constant at each time-step, and we can apply the Kalman filter in a slightly modified version, as follows:

Given an initial guess $\hat{\mathbf{x}}_{0,0}$ and an initial covariance matrix $\mathbf{P}_{0,0}$, iteratively compute

$$\begin{cases} \hat{\mathbf{x}}_{k,k-1} = \mathbf{a}(\hat{\mathbf{x}}_{k-1,k-1}, \mathbf{u}_k) & \text{(Predicting the state)} \\ \mathbf{P}_{k,k-1} = \mathbf{A}_{k-1}\mathbf{P}_{k-1,k-1}\mathbf{A}_{k-1}^T + \Sigma_q & \text{(Calculating the prediction covariance)} \\ \mathbf{S}_k = \left(\Sigma_r + \mathbf{H}_k\mathbf{P}_{k,k-1}\mathbf{H}_k^T\right) & \text{(Calculating the measurement covariance)} \\ \mathbf{K}_k = \mathbf{P}_{k,k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1} & \text{(Calculating the Kalman gain matrix)} \\ \hat{\mathbf{x}}_{k,k} = \hat{\mathbf{x}}_{k,k-1} + \mathbf{K}_k\left(\mathbf{y}_k - h\left(\hat{\mathbf{x}}_{k,k-1}\right)\right) & \text{(Correcting the estimation)} \\ \mathbf{P}_{k,k} = (I - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k,k-1} & \text{(Updating the covariance matrix)} \end{cases}$$

In this algorithm we have dropped the superscripts "MAP" on the estimated values, since this approach is no longer optimal. Indeed we are no longer certain that the posterior distribution is Gaussian, and since some of our derivations were based on the Gaussianity of linear combinations of Gaussian variables, the optimality results no longer hold.

### 2.3.2 Kalman filter with time-varying matrices

Instead of the EKF we can use a less refined version of it, as has been done further on to validate and debug the code that has been produced. The algorithm is the following:

Given an initial guess $\hat{\mathbf{x}}_{0,0}$ and an initial covariance matrix $\mathbf{P}_{0,0}$, iteratively compute

$$\begin{cases} \hat{\mathbf{x}}_{k,k-1} = \mathbf{A}_{k-1}\hat{\mathbf{x}}_{k-1,k-1} + \mathbf{B}_{k-1}\mathbf{u}_k & \text{(Predicting the state)} \\ \mathbf{P}_{k,k-1} = \mathbf{A}_{k-1}\mathbf{P}_{k-1,k-1}\mathbf{A}_{k-1}^T + \Sigma_q & \text{(Calculating the prediction covariance)} \\ \mathbf{S}_k = \left(\Sigma_r + \mathbf{H}_k\mathbf{P}_{k,k-1}\mathbf{H}_k^T\right) & \text{(Calculating the measurement covariance)} \\ \mathbf{K}_k = \mathbf{P}_{k,k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1} & \text{(Calculating the Kalman gain matrix)} \\ \hat{\mathbf{x}}_{k,k} = \hat{\mathbf{x}}_{k,k-1} + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k,k-1}\right) & \text{(Correcting the estimation)} \\ \mathbf{P}_{k,k} = (I - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k,k-1} & \text{(Updating the covariance matrix)} \end{cases}$$

Here we have introduced the matrix $\mathbf{B}_k$, the linearised control matrix, defined as:

$$\mathbf{B}_{k-1} = \left.\frac{\partial \mathbf{a}}{\partial \mathbf{u}_k}\right|_{\hat{\mathbf{x}}_{k-1,k-1}, \mathbf{u}_k}$$

This approach has the advantage that we can use matrix calculations in the whole process. However, in this case the Kalman gain is not optimal, and the posterior distribution is not Gaussian, since the matrix $\mathbf{A}_{k-1}$ depends upon the state $\hat{\mathbf{x}}_{k-1,k-1}$.

### 2.3.3 Applying the method

What we have seen in this chapter is a general description of the Kalman filter. The values we have to insert into $\mathbf{A}_k$, $\mathbf{B}_k$ and $\mathbf{H}_k$ depend on the particular problem we are solving. In our case $\mathbf{A}_k$ is given by the numerical scheme we are using, and the evolution equation comes from equation (1.8)

by applying the time discretisation we have chosen. $\mathbf{B}_k$ on the other hand is defined by the way in which the boundary conditions are imposed on the domain, and $\mathbf{u}_k$ is a vector that contains the values outside the domain. Finally, $\mathbf{H}_k$ is a constant matrix in our case, which shows which values we are measuring.

The precise way in which the matrices are computed will be explained in the following chapter. However, the general idea is translating the numerical scheme we have found in the first chapter into a form that makes a Kalman filter applicable and profitable.

# Chapter 3

# Joint state-parameter estimation

Having introduced the tools we are going to use, we can finally introduce the method we will put in place to find an estimate of $\beta$. This simply is a Kalman filter (classical, extended or with time-varying matrices) applied on an augmented state vector. For our purposes, uniquely finding an approximate value of the parameter would be enough, but unfortunately this is not possible through Kalman filtering. Indeed, the Kalman filter needs to use the interaction between state and parameter to narrow the distribution of $\beta$, so both must be estimated simultaneously.

## 3.1 General description

Having a state vector evolution and a measurement function

$$\mathbf{x}_k = \mathbf{a}(\mathbf{x}_{k-1}, \beta_{k-1}, \mathbf{u}_k) + \mathbf{q}_k$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \beta_k) + \mathbf{r}_k$$

which give us the possibility to use a Kalman filter, and wanting to find an estimate for a vector of parameters $\beta \in \mathbb{R}^d$, we define the augmented state-vector as:

$$\mathbf{x}_{aug} = \begin{bmatrix} \mathbf{x} \\ \beta \end{bmatrix}$$

with $\mathbf{x}$ being the regular state. Since the parameter is constant throughout time, we can define its evolution function simply as:

$$\beta_k = \beta_{k-1} \tag{3.1}$$

This function has to be appropriately inserted in the new evolution matrix $\tilde{\mathbf{A}}_{k-1}$ (this is the matrix that will be used in the algorithm):

$$\tilde{\mathbf{A}}_{k-1} = \begin{bmatrix} \mathbf{A}_{k-1} & \mathbf{D}_{k-1} \\ 0 & \mathbf{I}_d \end{bmatrix}$$

Where we have defined the matrices

$$\mathbf{A}_{k-1} = \frac{\partial \mathbf{a}}{\partial \mathbf{x}_{k-1}}(\hat{\mathbf{x}}_{k-1,k-1}, \mathbf{u}_k, \beta_{k-1,k-1})$$

$$\mathbf{D}_{k-1} = \frac{\partial \mathbf{a}}{\partial \beta}(\hat{\mathbf{x}}_{k-1,k-1}, \mathbf{u}_k, \beta_{k-1,k-1})$$

Here the term $\mathbf{I}_d$ denotes the identity matrix in $\mathbb{R}^d$. The matrix $\mathbf{D}_k \in \mathbb{R}^{n \times d}$ defines how the parameter influences the state vector, and is of fundamental importance. Indeed, if this matrix is made up of all zeros, the parameters will not evolve through the Kalman filter. This is quite reasonable, since a null $\mathbf{D}_k$ matrix means that the state is not influenced by the parameter (i.e. is independent from it).

The matrix which defines how the measurements are taken also needs to be modified to match the new conditions:

$$\tilde{\mathbf{H}}_k = \begin{bmatrix} \mathbf{H}_k & \mathbf{G}_k \end{bmatrix}$$

Where we have defined the matrices

$$\mathbf{H}_k = \frac{\partial \mathbf{h}}{\partial \mathbf{x}_k}(\hat{\mathbf{x}}_{k,k-1}, \beta_{k,k-1}) \qquad \mathbf{G}_k = \frac{\partial \mathbf{h}}{\partial \beta}(\hat{\mathbf{x}}_{k,k-1}, \beta_{k,k-1})$$

## 3.2 Vessel wall elasticity estimation

In our specific problem, the state vector is $\mathbf{U}_h^d = \begin{bmatrix} \mathbf{A}_h^d \\ \mathbf{u}_h^d \end{bmatrix}$. In building up the Kalman filter we need to find a global evolution matrix, which brings from the solution at a given time-step to the next one. Unfortunately, because of the interactions between elements (through boundaries between elements and through the bifurcation) we cannot treat each element separately, but will end up with a much larger matrix. From this point onwards, in order to simplify notation, we will treat a simple bifurcation, made up of three elements. However, generalising from our results is not a hard task. The actual state we will consider is:

$$\mathbf{U}_{global} = \begin{bmatrix} \mathbf{A}_h^1 \\ \mathbf{A}_h^2 \\ \mathbf{A}_h^3 \\ \mathbf{u}_h^1 \\ \mathbf{u}_h^2 \\ \mathbf{u}_h^3 \end{bmatrix}$$

where the superscripts indicate the vessel to which the variable (which denotes the vector of Galerkin node values) refers. Additionally, we want

to estimate the parameter vector $\beta$, $\in \mathbb{R}^3$ (to start with we suppose that each vessel has a different $\beta$, which is constant on each artery). Thus, the augmented state vector is $\mathbf{U}_{aug} \in \mathbb{R}^{6(r+1)+3}$ (where $r$ is the polynomial degree we have chosen in the Discontinuous Galerkin method):

$$\mathbf{U}_{aug} = \begin{bmatrix} \mathbf{U}_{global} \\ \beta \end{bmatrix}$$

In order to define the evolution equation in our particular case we need to start from equation (1.8), which through time-discretization becomes:

$$\mathbf{U}_{k+1}^d = \mathbf{U}_k^d - \Delta t \mathbf{M}^{-1}(\mathbf{S}\mathbf{H}\mathbf{U}_k^d + \mathbf{H}_{\partial\Omega_e}\mathbf{U}_k^d + \mathbf{P}\mathbf{F}_k^{\mathbf{u}})$$

Here the subscript $h$, which in equation (1.8) shows that we are considering the Galerkin approximation, has been dropped to make place for $k$, which denotes the time-step. However this is only due to notation constraints, we are still in the Galerkin case. Each of the matrices involved in this equation has to be adapted to fit the augmented-state form, and the fluxes have to be linearised (once and for all or at each time step depending on the method) in order to use a Kalman filter.

Indeed, as we have seen in the previous chapter, the covariance update is always done by multiplying the previous matrix by $\mathbf{A}_k$, which is the constant or time-varying (depending on the method) linearisation of the fluxes. If we then choose to use a linearised Kalman (with time-varying matrices or not) this matrix will also be used to make the prediction step. If instead we use an extended Kalman filter the prediction is done through the non-linear model.

Additionally, we need to make a distinction between state-evolution and external input, and define the matrix $\mathbf{B}_k$, along with the control vector $\mathbf{u}_k$. In the following subsections we will use $\mathbf{H}$ to denote the quasi-linear matrices, and $\mathbf{H}_{meas}$ to denote the measurement matrix. The problem is written on three vessels, communicating through a bifurcation, but could be expanded effortlessly.

### 3.2.1 The flux definition matrix & border flux matrix

In order to find the global linearized matrix $\mathbf{H}_{aug}$ (which is the augmented-state version of the matrix $\mathbf{H}$ in (1.8)) we need to linearise the fluxes in (1.5):

$$Au \approx A_e u_e + A_e(u - u_e) + u_e(A - A_e) \tag{3.2}$$

$$\frac{u^2}{2} + \frac{\beta}{\rho}\left(\sqrt{A} - \sqrt{A_0}\right) \approx \frac{u_e^2}{2} + \frac{\beta_e}{\rho}\left(\sqrt{A_e} - \sqrt{A_0}\right) + u_e(u - u_e) +$$

$$+ \frac{\beta_e}{2\rho\sqrt{A_e}}(A - A_e) + \frac{\left(\sqrt{A_e} - \sqrt{A_0}\right)}{\rho}(\beta - \beta_e) \tag{3.3}$$

Here we have used the subscript "e" to denote the state around which we are linearising. We take these values to be constant in each vessel, but different from one vessel to another. The values we insert in these parameters determine which method we are using: if we take a time-constant value we are using the classical Kalman filter, while if we use the previous state estimate we are using the Kalman filter with time-varying matrices. These cases have been studied in sections 4.1.1 and 4.1.2.

Through the linearised fluxes, by considering the variables ($A$, $u$, $\beta$), and not the variables ($A - A_e$, $u - u_e$, $\beta - \beta_e$), we can build the augmented quasi-linear matrix $\mathbf{H}_i$, which holds for vessel $i$. This matrix will then, after being translated into its Galerkin version, be inserted in the global quasi-linear matrix $\mathbf{H}_{aug}$.

$$\mathbf{H}_i = \begin{bmatrix} u_e & A_e & 0 \\ \frac{\beta_e}{2\rho\sqrt{A_e}} & u_e & \frac{\left(\sqrt{A_e}-\sqrt{A_0}\right)}{\rho} \\ 0 & 0 & 0 \end{bmatrix}$$

The third line is null because $\beta$'s evolution function (3.1) has no term that is derived spatially. We also obtain a constant vector from the linearised fluxes (3.2) and (3.3):

$$\mathbf{f}_{const} = \begin{bmatrix} -A_e u_e \\ -\frac{u_e^2}{2} - \frac{\beta_e\sqrt{A_e}}{2\rho} \\ 0 \end{bmatrix}$$

This vector plays no role in the wave propagation inside each element, since the space-derivative eliminates it (we must bear in mind that $A_e$, $u_e$ and $\beta_e$ are constant throughout each vessel), but intervenes at element borders (bifurcation, between elements or physical boundaries).

In order to find the global quasi-linear matrix, we first need to expand each $\mathbf{H}_i$ so that it describes the finite elements case. This is done in a straightforward manner, exactly as we did in the first chapter. Then we have to assemble the three resulting matrices appropriately, to yield the following:

$$\mathbf{H}_{aug} = \begin{bmatrix} \tilde{\mathbf{H}}_1 & \tilde{\mathbf{H}}_2 & \mathbf{0}_{3(r+1)\times 3} \\ \tilde{\mathbf{H}}_3 & \tilde{\mathbf{H}}_4 & \tilde{\mathbf{D}} \\ \mathbf{0}_{3\times 3(r+1)} & \mathbf{0}_{3\times 3(r+1)} & \mathbf{0}_{3\times 3} \end{bmatrix}$$

Where $\mathbf{0}_{3\times 3(r+1)}$ denotes a matrix $\in \mathbb{R}^{3\times 3(r+1)}$ of all zeros, and the matrices $\tilde{\mathbf{H}}_1$ are defined as:

$$\tilde{\mathbf{H}}_1 = \tilde{\mathbf{H}}_4 = \begin{bmatrix} \mathbf{diag}(\mathbf{u}_e^1) & \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{0}_{(r+1)\times(r+1)} \\ \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{diag}(\mathbf{u}_e^2) & \mathbf{0}_{(r+1)\times(r+1)} \\ \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{diag}(\mathbf{u}_e^3) \end{bmatrix}$$

$$\tilde{\mathbf{H}}_2 = \begin{bmatrix} \mathbf{diag}(\mathbf{A}_e^1) & \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{0}_{(r+1)\times(r+1)} \\ \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{diag}(\mathbf{A}_e^2) & \mathbf{0}_{(r+1)\times(r+1)} \\ \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{diag}(\mathbf{A}_e^3) \end{bmatrix}$$

$$\tilde{\mathbf{H}}_3 = \begin{bmatrix} \mathbf{diag}(\frac{\beta_1}{2\rho\sqrt{\mathbf{A}_e^1}}) & \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{0}_{(r+1)\times(r+1)} \\ \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{diag}(\frac{\beta_2}{2\rho\sqrt{\mathbf{A}_e^2}}) & \mathbf{0}_{(r+1)\times(r+1)} \\ \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{0}_{(r+1)\times(r+1)} & \mathbf{diag}(\frac{\beta_3}{2\rho\sqrt{\mathbf{A}_e^3}}) \end{bmatrix}$$

Where $\mathbf{0}_{(r+1)\times(r+1)}$ denotes a matrix $\in \mathbb{R}^{(r+1)\times(r+1)}$ of all zeros.

The matrix $D \in \mathbb{R}^{3(r+1)\times 3}$ which defines the way in which the $\beta$s influence the fluxes is defined as:

$$\tilde{\mathbf{D}} = \begin{bmatrix} \frac{\left(\sqrt{\mathbf{A}_e^1}-\sqrt{A_0^1}\right)}{\rho} & \mathbf{0}_{(r+1)} & \mathbf{0}_{(r+1)} \\ \mathbf{0}_{(r+1)} & \frac{\left(\sqrt{\mathbf{A}_e^2}-\sqrt{A_0^2}\right)}{\rho} & \mathbf{0}_{(r+1)} \\ \mathbf{0}_{(r+1)} & \mathbf{0}_{(r+1)} & \frac{\left(\sqrt{\mathbf{A}_e^3}-\sqrt{A_0^3}\right)}{\rho} \end{bmatrix}$$

where $\frac{\left(\sqrt{\mathbf{A}_e^i}-\sqrt{A_0^i}\right)}{\rho}$ denotes the vector resulting from calculating $\beta$'s coefficient for each component of the vector $\mathbf{A}_e^i$, and $\mathbf{0}_{(r+1)}$ denotes a vector $\in \mathbb{R}^{(r+1)}$ of all zeros.

The vector $\mathbf{f}_{const}$ also has to be expanded, to become:

$$\mathbf{f}_{const}^{aug} = \begin{bmatrix} -\mathbf{A}_e^1 \cdot \mathbf{u}_e^1 \\ -\mathbf{A}_e^2 \cdot \mathbf{u}_e^2 \\ -\mathbf{A}_e^3 \cdot \mathbf{u}_e^3 \\ -\frac{\left(\mathbf{u}_e^1\right)^2}{2} - \frac{\beta_e^1\sqrt{\mathbf{A}_e^1}}{2\rho} \\ -\frac{\left(\mathbf{u}_e^2\right)^2}{2} - \frac{\beta_e^2\sqrt{\mathbf{A}_e^2}}{2\rho} \\ -\frac{\left(\mathbf{u}_e^3\right)^2}{2} - \frac{\beta_e^3\sqrt{\mathbf{A}_e^3}}{2\rho} \\ \mathbf{0}_3 \end{bmatrix}$$

where as above, $\mathbf{0}_3$ denotes a vector of length 3 of all zeros, and $-\frac{\left(\mathbf{u}_e^i\right)^2}{2} - \frac{\beta_e^i\sqrt{\mathbf{A}_e^i}}{2\rho}$ is the pointwise calculation, applied to each component of $\mathbf{A}_e^i$ and $\mathbf{u}_e^i$. In the case of the border fluxes it is enough to suppose that for all $i$:

$$\mathbf{A}_e^i = \begin{bmatrix} A_{e1}^i \\ \mathbf{0}_8 \\ A_{e(r+1)}^i \end{bmatrix} \qquad \mathbf{u}_{ei} = \begin{bmatrix} u_{e1}^i \\ \mathbf{0}_8 \\ u_{e(r+1)}^i \end{bmatrix}$$

in order to get to the correct $\mathbf{f}_{const}^{\partial\Omega_e}$. The same holds for $\mathbf{H}_{aug}^{\partial\Omega_e}$. The border fluxes are then calculated in the usual way:

$$\mathbf{F}^{\partial\Omega_e} = \mathbf{H}_{aug}^{\partial\Omega_e}\mathbf{U}_{aug} + \mathbf{f}_{const}^{\partial\Omega_e}$$

### 3.2.2 The bifurcation matrix

Since we are using global matrices (that consider all three vessels simultaneously), we now can describe the bifurcation conditions using a single

matrix. In order to do this we need to use the linearised fluxes (equations (3.2) and (3.3)), and apply the conditions on these fluxes instead of on the non-linear ones. If we define the coefficients:

$$s = \frac{\beta_e}{2\rho\sqrt{A_e}} \qquad d = \frac{\left(\sqrt{A_e} - \sqrt{A_0}\right)}{\rho} \qquad k = \sqrt{\frac{\beta_e}{2\rho}} A_e^{-3/4} \qquad t = -\frac{u_e^2}{2} - sA_e$$

the set of equations we have to solve becomes:

$$u_e^p u^p + t^p + s^p A^p + d^p \beta^p = u_e^{ch1} u^{ch1} + t^{ch1} + s^{ch1} A^{ch1} + d^{ch1} \beta^{ch1}$$

$$u_e^p u^p + t^p + s^p A^p + d^p \beta^p = u_e^{ch2} u^{ch2} + t^{ch2} + s^{ch2} A^{ch2} + d^{ch2} \beta^{ch2}$$

$$A_e^p u^p + u_e^p A^p - A_e^p u_e^p = A_e^{ch1} u^{ch1} + u_e^{ch1} A^{ch1} - A_e^{ch1} u_e^{ch1} + A_e^{ch2} u^{ch2} + u_e^{ch2} A^{ch2} - A_e^{ch2} u_e^{ch2}$$

$$\hat{W}_1 = u^p + k^p A^p$$

$$\hat{W}_{21} = u^{ch1} - k^{ch1} A^{ch1}$$

$$\hat{W}_{22} = u^{ch2} - k^{ch2} A^{ch2}$$

In the last three equations we have linearised the characteristic variables, and then we have neglected the constant terms (including $\beta$, which is supposed constant). Indeed these appear on both sides of the equations, thus cancel out. The notation $\hat{W}$ indicates this different way of calculating $W$.

As we have seen previously, in order to apply the last three equations, we have to compute $\hat{W}$ from our finite elements solution at the boundary, and then use it as a constant in the set of equations, whose variables are $(A, u)$ of each vessel. This translates into defining the matrix:

$$\mathbf{C}_{char} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 3} \\ k^p & 0 & 0 & 1 & 0 & 0 & \mathbf{0}_3^T \\ 0 & -k^{ch1} & 0 & 0 & 1 & 0 & \mathbf{0}_3^T \\ 0 & 0 & -k^{ch2} & 0 & 0 & 1 & \mathbf{0}_3^T \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 3} \end{bmatrix}$$

This matrix has to multiply the vector which contains the bifurcation values of $\mathbf{U}$:

$$\mathbf{U}_{bif} = \begin{bmatrix} A^p \\ A^{ch1} \\ A^{ch2} \\ u^p \\ u^{ch1} \\ u^{ch2} \\ \beta^p \\ \beta^{ch1} \\ \beta^{ch2} \end{bmatrix}$$

We then have to define the "compatibility" matrix, which actually codes the bifurcation compatibility conditions:

$$\mathbf{C}_{comp} = \begin{bmatrix} s^p & -s^{ch1} & 0 & u_e^p & -u_e^{ch1} & 0 & d^p & -d^{ch1} & 0 \\ s^p & 0 & -s^{ch2} & u_e^p & 0 & -u_e^{ch2} & d^p & 0 & -d^{ch2} \\ u_e^p & -u_e^{ch1} & -u_e^{ch2} & A_e^p & -A_e^{ch1} & -A_e^{ch2} & 0 & 0 & 0 \\ k^p & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & k^{ch1} & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & k^{ch2} & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

We also end up with a constant vector:

$$\mathbf{const}_{comp} = \begin{bmatrix} t^{ch1} - t^p \\ t^{ch2} - t^p \\ u_e^p A_e^p - u_e^{ch1} A_e^{ch1} - u_e^{ch2} A_e^{ch2} \\ \mathbf{0}_3 \\ \beta^p \\ \beta^{ch1} \\ \beta^{ch2} \end{bmatrix}$$

The solution of the set of equation then simply is:

$$\mathbf{U}_{bif}^{new} = \mathbf{C}_{comp}^{-1}(\mathbf{C}_{char}\mathbf{U}_{bif} + \mathbf{const}_{comp})$$

This expression then needs to be expanded appropriately in order to match the dimensions of the Galerkin problem. This can be done through the matrices $\mathbf{P}_1 \in \mathbb{R}^{(6(r+1)+3)\times 9}$ and $\mathbf{P}_2 \in \mathbb{R}^{9\times(6(r+1)+3)}$ (whose definition is simple, knowing how $\mathbf{U}_{bif}^{new}$ and $\mathbf{U}_{aug}$ are structured). Indeed they simply have to put the values in the correct positions, leaving the rest of the matrix values equal to zero, much like matrix P in (1.10) does.

$$\mathbf{U}_{aug}^{bif} = \mathbf{P}_1\mathbf{C}_{comp}^{-1}\mathbf{C}_{char}\mathbf{P}_2\mathbf{U}_{aug} + \mathbf{P}_1\mathbf{C}_{comp}^{-1}\mathbf{const}_{comp}$$

This vector then has to be multiplied by $\mathbf{H}_{aug}$ in order to find the upwind flux in this case

$$\mathbf{F}^{bif}_{upwind} = \mathbf{H}_{aug}\mathbf{U}^{bif}_{aug} + \mathbf{f}^{bif}_{const}$$

where the vector $\mathbf{f}^{bif}_{const}$ is found from the general expression of $\mathbf{f}^{aug}_{const}$ simply by setting

$$\mathbf{A}^{p}_{e} = \begin{bmatrix} \mathbf{0}_r \\ A^{p}_{e(r+1)} \end{bmatrix} \qquad \mathbf{u}_{ei} = \begin{bmatrix} \mathbf{0}_r \\ u^{p}_{e(r+1)} \end{bmatrix}$$

for the parent vessel, while for the children vessels we have

$$\mathbf{A}^{ch}_{e} = \begin{bmatrix} A^{ch}_{e1} \\ \mathbf{0}_r \end{bmatrix} \qquad \mathbf{u}_{ei} = \begin{bmatrix} u^{ch}_{e1} \\ \mathbf{0}_r \end{bmatrix}$$

In conclusion, the upwind flux at the bifurcation can be expressed using a matrix-vector multiplication

$$\mathbf{F}^{bif}_{upwind} = \mathbf{H}^{bif}_{aug}\mathbf{U}_{aug} + \mathbf{f}^{bif}_{const} + \mathbf{H}_{aug}\mathbf{P}_1\mathbf{C}^{-1}_{comp}\mathbf{const}_{comp}$$

where the new quasi-linear matrix is defined as

$$\mathbf{H}^{bif}_{aug} = \mathbf{H}_{aug}\mathbf{P}_1\mathbf{C}^{-1}_{comp}\mathbf{C}_{char}\mathbf{P}_2$$

### 3.2.3 The control matrix

We also have to put the border condition impositions in the form of a matrix-vector multiplication. To start with, we define the border vector and the input vector:

$$\mathbf{U}_{\partial\Omega} = \begin{bmatrix} A^r \\ A^{l1} \\ A^{l2} \\ u^r \\ u^{l1} \\ u^{l2} \\ \beta^r \\ \beta^{l1} \\ \beta^{l2} \end{bmatrix} \qquad \mathbf{U}_{input} = \begin{bmatrix} A^r_{in} \\ A^{l1}_{in} \\ A^{l2}_{in} \\ u^r_{in} \\ u^{l1}_{in} \\ u^{l2}_{in} \\ \beta^r_{in} \\ \beta^{l1}_{in} \\ \beta^{l2}_{in} \end{bmatrix}$$

In this case we will use the same $\beta$ inside and outside the domain (this is the most reasonable choice, since the real $\beta$s are unknown, the only value we have is $\beta_e$), $\beta = \beta_e$, so the linearised definitions of $W_1$ and $W_2$ simply are $\hat{W}_1$ and $\hat{W}_2$. The expressions of $\hat{W}_1$ and $\hat{W}_2$ bring to the equations that define the area and velocity at the boundary:

$$u = \frac{\hat{W}_1 + \hat{W}_2}{2} \qquad A = \frac{\hat{W}_1 - \hat{W}_2}{2k}$$

Recalling that $\hat{W}_1$ is calculated from the state at the left of the node, while $\hat{W}_2$ is calculated from the values at the right, the augmented state vector $\mathbf{U}_{upwind}$ can be computed as:

$$\mathbf{U}_{upwind} = \mathbf{C}_{control}\mathbf{U}_{input} + \mathbf{C}_{\partial\Omega}\mathbf{U}_{\partial\Omega} \tag{3.4}$$

where we have defined the matrices

$$\mathbf{C}_{control} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2k^r} & 0 & 0 & \mathbf{0}_3^T \\ 0 & \frac{1}{2} & 0 & 0 & -\frac{1}{2k^{l1}} & 0 & \mathbf{0}_3^T \\ 0 & 0 & \frac{1}{2} & 0 & 0 & -\frac{1}{2k^{l2}} & \mathbf{0}_3^T \\ \frac{k^r}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 & \mathbf{0}_3^T \\ 0 & -\frac{k^{l1}}{2} & 0 & 0 & \frac{1}{2} & 0 & \mathbf{0}_3^T \\ 0 & 0 & -\frac{k^{l2}}{2} & 0 & 0 & \frac{1}{2} & \mathbf{0}_3^T \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 3} \end{bmatrix}$$

$$\mathbf{C}_{\partial\Omega} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & -\frac{1}{2k^r} & 0 & 0 & \mathbf{0}_3^T \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2k^{l1}} & 0 & \mathbf{0}_3^T \\ 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2k^{l2}} & \mathbf{0}_3^T \\ -\frac{k^r}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 & \mathbf{0}_3^T \\ 0 & \frac{k^{l1}}{2} & 0 & 0 & \frac{1}{2} & 0 & \mathbf{0}_3^T \\ 0 & 0 & \frac{k^{l2}}{2} & 0 & 0 & \frac{1}{2} & \mathbf{0}_3^T \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_{3\times 3} \end{bmatrix}$$

The vector $\mathbf{U}_{upwind}$ then has to be multiplied by $\mathbf{H}_{aug}^{borders}$ (the quasi-linear matrix that defines the flux, evaluated at the border nodes) in order to find the upwinded fluxes. As we have seen before this operation also involves a constant term $\mathbf{f}_{const}^{\partial\Omega}$ which comes from the linearised forms.

$$\mathbf{F}_{upwind}^{\partial\Omega} = \mathbf{H}_{aug}^{borders}\mathbf{P}_3\mathbf{U}_{upwind} + \mathbf{f}_{const}^{\partial\Omega}$$

Where the matrix $\mathbf{P}_3 \in \mathbb{R}^{(6(r+1)+3)\times 9}$ is another expansion matrix, which brings $\mathbf{U}_{upwind}$ to be of the appropriate dimensions, and with the entries at the appropriate positions. The vector $\mathbf{f}_{const}^{\partial\Omega}$ most not be confused for $\mathbf{f}_{const}^{\partial\Omega_e}$.

In order to use the Kalman filter properly we need to separate the contributions from inside and outside the domain:

$$\mathbf{F}_{upwind}^{\partial\Omega} = \mathbf{B}_{control}\mathbf{P}_3\mathbf{U}_{input} + \mathbf{H}_{\partial\Omega}^{borders}\mathbf{U}_{aug} + \mathbf{f}_{const}^{\partial\Omega}$$

Where $\mathbf{P}_3\mathbf{U}_{input}$ is the expanded input vector. The matrices simply become:

$$\mathbf{B}_{control} = \mathbf{H}_{aug}^{borders}\mathbf{P}_3\mathbf{C}_{control}\mathbf{P}_4$$

$$\mathbf{H}_{\partial\Omega}^{borders} = \mathbf{H}_{aug}^{borders}\mathbf{P}_3\mathbf{C}_{\partial\Omega}\mathbf{P}_4$$

The matrix $\mathbf{P}_4$ is yet another expansion matrix, whose purpose is to make the $\mathbf{C}$ matrices of the right number of columns.

If we observe the two matrices $\mathbf{C}_{control}$ and $\mathbf{C}_{\partial\Omega}$ we see that one has an identity matrix as last entry, while the other has a zero matrix. The choice of whether to put the identity matrix in the first or the second matrix

is arbitrary, but it is important that only one of them has it, otherwise equation (3.4) does not hold.

The matrix $\mathbf{H}_{\partial\Omega}^{borders}$ is different from $\mathbf{H}_{aug}^{\partial\Omega_e}$, since it only has entries that correspond to the physical boundaries of the domain. The way in which it is calculated is also different, so these terms will not cancel out in the final expression.

### 3.2.4 The mass matrix and the derivation matrix

The conservation law that concerns $\beta$ simply is:

$$\frac{\partial \beta}{\partial t} = 0 \tag{3.5}$$

This can be reinterpreted as

$$\beta_{k+1} = \beta_k$$

Which means that the evolution matrix of the parameters simply is the identity. This matrix has to be inserted in the mass matrix:

$$\mathbf{M}_{aug} = \begin{bmatrix} \mathbf{M} & 0 \\ 0 & \mathbf{I}_3 \end{bmatrix}$$

Augmenting the derivation matrix is even easier. Indeed equation (3.5) does not contain any spatial derivatives, so we only need to make $\mathbf{S}_{aug}$ of the right dimensions by zero-padding of $\mathbf{S}$.

### 3.2.5 Assembling the global evolution matrix

In the case of the blood flow simulation we chose an Adams-Bashforth approach for the time discretisation. However, now we are about to use a Kalman filter, so this choice no longer suits our needs. Indeed, for a Kalman filter to work we need to have a Markov process, meaning that the future state can only depend on the current one, and not on ones further back in time. Therefore we have chosen to use a simple Euler forward method

$$\frac{\partial u}{\partial t} = f(u,t) \implies u_{n+1} = u_n + \Delta t f(u_n, t_n)$$

If we recall equation (1.8), we then find the following:

$$\mathbf{U}_{aug}^{k+1} = \mathbf{U}_{aug}^{k} - \Delta t \mathbf{M}_{aug}^{-1} \left( \mathbf{S}_{aug} \mathbf{H}_{aug}^{k} \mathbf{U}_{aug}^{k} + \mathbf{P}_{sign} \left( \mathbf{F}_{upwind}^{\partial\Omega} + \mathbf{F}_{upwind}^{bif} - \mathbf{F}^{\partial\Omega_e} \right) \right)$$

Here the matrix $\mathbf{P}_{sign}$ is a matrix which changes the signs of the fluxes appropriately. Indeed we had the expression $[(\mathbf{F^u} - \mathbf{F}(\mathbf{U}_h)) \cdot \varphi]_{x_e(left)}^{x_e(right)}$, which means that the fluxes on the right side of each element have the sign "+", while those on the left have the sign "-". The superscript "k" hasn't been applied to $\mathbf{F}_{upwind}^{\partial\Omega}$, $\mathbf{F}_{upwind}^{bif}$ and $\mathbf{F}^{\partial\Omega_e}$ simply not to make the notation too

complicated, but all these matrices can change at each time step $k$. The same holds for the various $\mathbf{f}_{const}$ vectors.

Our final evolution equation is:

$$\mathbf{U}_{aug}^{k+1} = \mathbf{A}_{renew}^{k}\mathbf{U}_{aug}^{k} + \hat{\mathbf{B}}_{control}^{k}\mathbf{U}_{input} + \mathbf{f}_{const}^{k}$$

where we have defined the matrices

$$\mathbf{A}_{renew}^{k} = \mathbf{I} - \Delta t \mathbf{M}_{aug}^{-1}\left(\mathbf{S}_{aug}\mathbf{H}_{aug}^{k} + \mathbf{P}_{sign}\left(\mathbf{H}_{\partial\Omega}^{borders} + \mathbf{H}_{aug}^{bif} - \mathbf{H}_{aug}^{\partial\Omega_e}\right)\right)$$

$$\hat{\mathbf{B}}_{control} = -\Delta t \mathbf{M}_{aug}^{-1}\mathbf{P}_{sign}\mathbf{B}_{control}^{k}\mathbf{P}_3$$

and the constant vector:

$$\mathbf{f}_{const}^{k} = -\Delta t \mathbf{M}_{aug}^{-1}\mathbf{P}_{sign}\left(\mathbf{H}_{aug}^{k}\mathbf{P}_1\mathbf{C}_{comp}^{-1}\mathbf{const}_{comp}^{k} + \mathbf{f}_{const}^{\partial\Omega} + \mathbf{f}_{const}^{bif} - \mathbf{f}_{const}^{\partial\Omega_e}\right)$$

### 3.2.6 Measure equation

The final step towards applying the Kalman filter to our problem is defining the measure matrix $\mathbf{H}_{meas}$. Its definition obviously depends on which kind of data we have. In our case we have chosen to generate the measures (depending on the case from the linear, with time varying matrices or non-linear blood flow simulation), and have chosen to take three measures from each vessel, for both $A$ and $u$. The measures are taken at the nodes closest to 25%, 50% and 75% of the number of nodes of the artery. With 10 nodes this means at the second, fifth and seventh node.

Therefore the measure matrix becomes:

$$\mathbf{H}_{meas} = \begin{bmatrix} \mathbf{h}_{meas} & & \mathbf{0} & & \cdots & & \mathbf{0} \\ & \mathbf{h}_{meas} & & & & & \\ \mathbf{0} & & \mathbf{h}_{meas} & & \ddots & & \vdots \\ \vdots & \ddots & & \mathbf{h}_{meas} & & & \\ & & \ddots & & \mathbf{h}_{meas} & & \mathbf{0} \\ \mathbf{0} & \cdots & & \mathbf{0} & & \mathbf{h}_{meas} & \mathbf{0} \end{bmatrix}$$

with the submatrices defined as:

$$\mathbf{h}_{meas} = \begin{bmatrix} 0 & 1 & \mathbf{0}_2^T & 0 & 0 & 0 & \mathbf{0}_3^T \\ 0 & 0 & \mathbf{0}_2^T & 1 & 0 & 0 & \mathbf{0}_3^T \\ 0 & 0 & \mathbf{0}_2^T & 0 & 0 & 1 & \mathbf{0}_3^T \end{bmatrix}$$

Further on we will also test whether having measures only on the variable $A$ may be sufficient. The measure matrix changes accordingly, in an obvious manner.

The measures we generate from our simulations are then modified by adding a Gaussian noise, in order to make them more realistic. The variance

of this noise is chosen depending on what we want to test, but the noise is always chosen to have zero-mean, according to the Kalman filter hypotheses.

# Chapter 4

# Results

## 4.1 Linearised Kalman filter

In this case the values of $\mathbf{u}_e$ and $\mathbf{A}_e$ are chosen once and for all. As a consequence all matrices are time-constant, and we can use the classical Kalman filter. Two different choices for these variables have been studied: $(\mathbf{A}_0, \mathbf{u}_0)$ and $\mathbf{u}_e = \mathbf{0}$ and $\mathbf{A}_e$ that yield a pressure of 70 mmHg. In both cases the values have been chosen constant throughout the vessel, in order not to create perturbations. The initial conditions, which are at rest, are taken to be $(\mathbf{A}_e, \mathbf{u}_e)$, and are also equal to the values we assign to the unperturbed states outside our domain. In all the cases we will test the real values of the parameters are $\beta_1 = 9.7 \cdot 10^4$, $\beta_2 = 8.7 \cdot 10^4$, $\beta_3 = 2.33 \cdot 10^5$.

### 4.1.1 Linearising around $(\mathbf{A}_0, \mathbf{u}_0, \beta_0)$

This choice is reasonable if we insert the first input, the half sine that rises from the equilibrium $A_0$. Indeed with a quasi-linear matrix structured as follows:

$$\mathbf{H} = \begin{bmatrix} u_0 & A_0 & 0 \\ \frac{\beta_0}{2\rho\sqrt{A_0}} & u_0 & \frac{\left(\sqrt{A_0}-\sqrt{A_0}\right)}{\rho} \\ 0 & 0 & 0 \end{bmatrix} \tag{4.1}$$

we have that when we have a constant input equal to $A_0$ the system is at rest. This happens because the bifurcation conditions are automatically satisfied (for every choice of $\beta_0$), and the border conditions find the same value on the border and outside the domain (and therefore leaves it unperturbed). This means that when the input half sine has returned to its base value, and after a short period in which the system relaxes, the arteries return to the exact values $(A_0, u_0)$.

However, if we take a look at matrix (4.1), we can see that the value $\mathbf{H}_{32}$ is zero. This means that the variable $\beta$ does not appear in any flux, thus does not appear in the conservation law we are considering. The only value that appears is $\beta_0$, a constant guess we make. In this case any hope of

refining the estimation is vain, since we are trying to do so on an independent parameter. The only thing we can achieve by this procedure is filtering the measurement noise, making a state estimation instead of a joint state-parameter estimation.

### 4.1.2 Linearising around $(\mathbf{A}_e, \mathbf{u}_e, \beta_e)$

If we come back to observing matrix (4.1) we see that the problem that $\beta$'s coefficient is zero is easily solved. Indeed, if we simply linearise around another point the coefficient $\mathbf{H}_{32}$ no longer will be zero. However this poses several other problems.

Firstly, the values of $u_e$ and $A_e$ have to be chosen wisely, otherwise the bifurcation conditions will not be satisfied, and a spurious signal will be created at this point. Therefore we can for instance choose them to yield a mass flux $Q = 0$ and a pressure $p = 70mmHg$ (this value is arbitrary), as we have seen before. However, we have a problem in the selection of $A_e$ since the expression

$$A_e = \left(\frac{70}{\beta} + \sqrt{A_0}\right)^2$$

depends on $\beta$, which is the parameter we want to estimate. Which $\beta$ should we choose in calculating $A_e$? The most reasonable choice is the initial guess of $\beta$, since it is the only value we have, but this is a dangerous option. In fact, the value we use influences the problem, both in choosing the linearisation point and in defining the various matrices. This leads to the fact that different initial guesses on $\beta$ lead to different results, yielding stability issues (and we cannot hope to obtain convergence to the real value in a system in which the final estimation depends on the initial guess).

In addition, let's observe what happens if we insert a constant input $A_{in} = A_0$ to the system in the initial conditions $(A_{initial}, u_{initial}) = (A_0, u_0)$. As we have seen before, in the non-linear blood flow simulation we have that the system is at rest, and remains unchanged. In the linearised case this no longer holds, because now the bifurcation conditions are not satisfied automatically, which leads to this state no longer being an equilibrium. Indeed, whereas the mass flux equation holds since $u_e = 0$ for all the vessels, the pressure equations do not hold if we use $A = A_0$:

$$\frac{\beta_e^p}{\rho}(\sqrt{A_e^p}-\sqrt{A_0^p})+\frac{\beta_e^p}{2\rho\sqrt{A_e^p}}(A_0^p-A_e^p) = \frac{\beta_e^{ch}}{\rho}(\sqrt{A_e^{ch}}-\sqrt{A_0^{ch}})+\frac{\beta_e^{ch}}{2\rho\sqrt{A_e^{ch}}}(A_0^{ch}-A_e^{ch})$$

$$\beta_e^p\left(\frac{\sqrt{A_e^p}}{2} - \sqrt{A_0^p} + \frac{A_0^p}{2\sqrt{A_e^p}}\right) = \beta_e^{ch}\left(\frac{\sqrt{A_e^{ch}}}{2} - \sqrt{A_0^{ch}} + \frac{A_0^{ch}}{2\sqrt{A_e^{ch}}}\right)$$

Since $A_e \neq A_0$ (which is clear from how it is defined), the parentheses will never become zeros:

$$\begin{aligned} \frac{\sqrt{A_e}}{2} - \sqrt{A_0} + \frac{A_0}{2\sqrt{A_e}} &= 0 \\ 2\sqrt{A_0 A_e} &= A_e + A_0 \\ (\sqrt{A_e} - \sqrt{A_0})^2 &= 0 \end{aligned}$$

This means that unless we are in a very fortunate case of choices of $\beta_e$, which certainly is too restrictive for our purposes, the pressure equations do not hold. This leads to the creation of a signal, which then travels back to the root and forward to the leaves, as can be seen in the following images. The figures "A vessel" and "u vessel" show the variables at the current time-step, while A_history and u_history show the time-evolution of the variables.

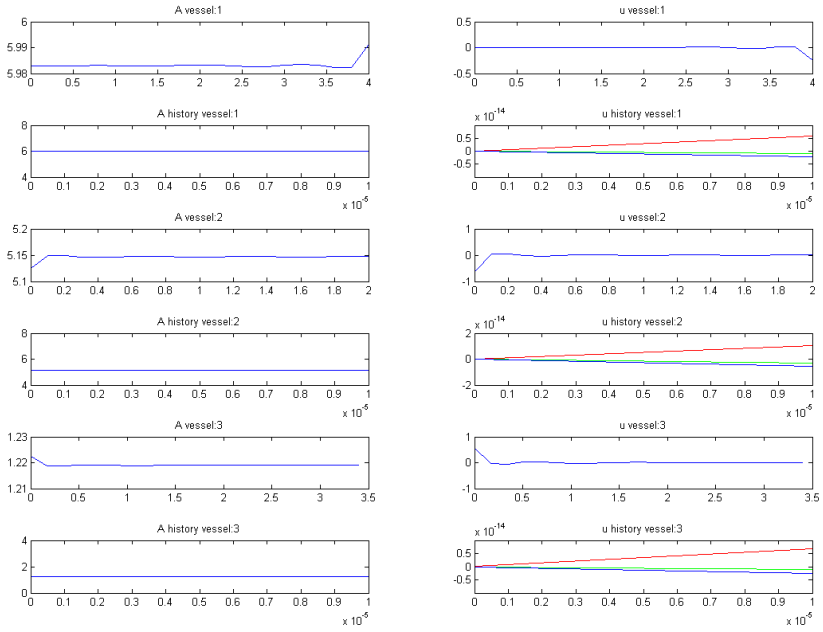


Figure 4.1: The first step, where immediately a signal is created at the bifurcation.

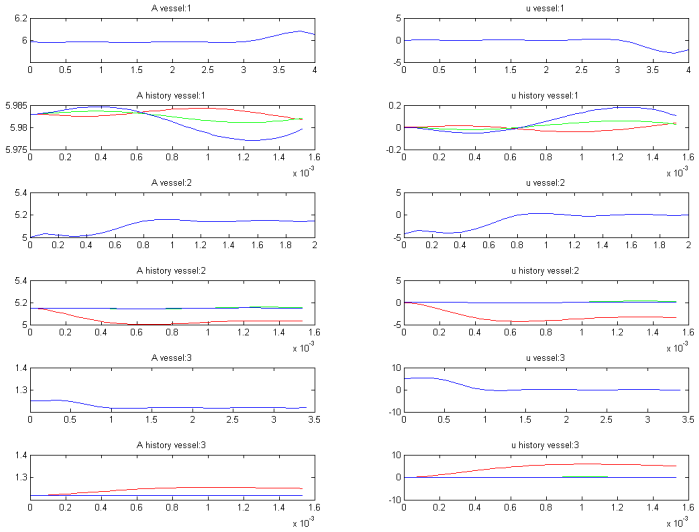


Figure 4.2: The signal then is propagated, just as any other wave

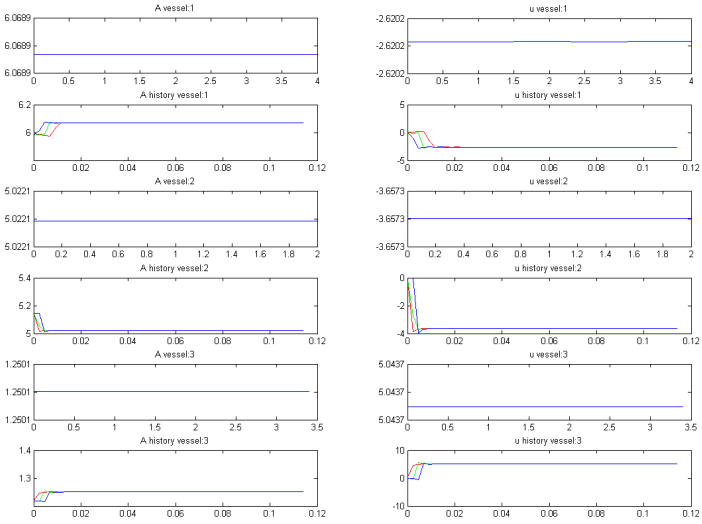


Figure 4.3: The signal then travels backward and forward, stabilising on an equilibrium

Finally the system attains an equilibrium, which obviously is different from $(A_0, u_0)$ (which is the equilibrium point it reaches in the non-linear case), but for which the bifurcation conditions hold. The importance of these observations lay in the fact that the results are different from those found in the non-linear simulation, from which we draw the measurements. This means that there is a positive inferior bound to the residual, meaning that it cannot tend to zero. In turn, this leads to convergence issues.

It is also interesting to observe how the input signal is imposed weakly: at the end of this simulation the value we find at the input section is different from what we are trying to impose.

The fact that $(A_0, u_0)$ is not an equilibrium leads to the system behaving differently when we impose the sine wave too. Indeed it does not return to $A_0$, but to the equilibrium values we found previously:

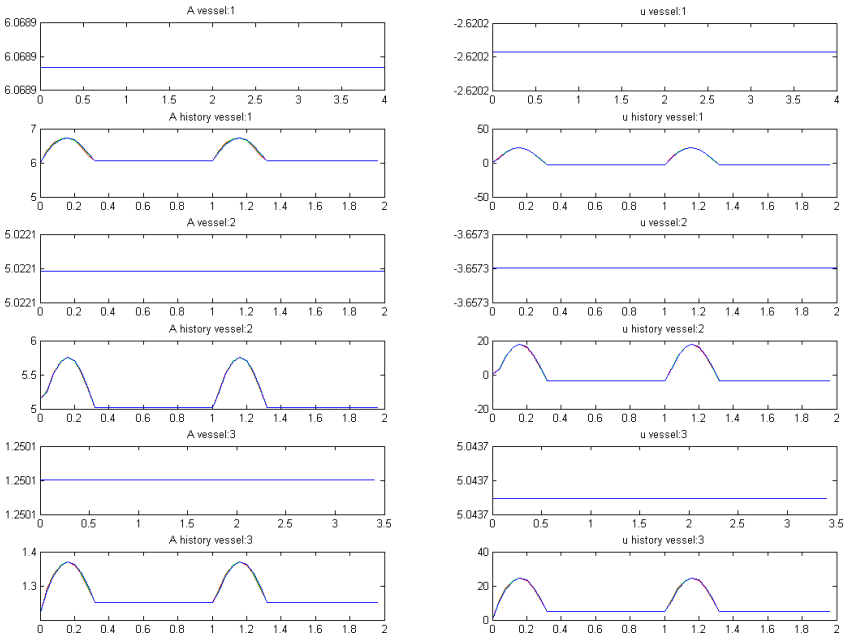


Figure 4.4: The linearised (around $(A_e, u_e)$) system's response to a half-sine wave input.

### 4.1.3 Linearising around $(\mathbf{A}_e, \mathbf{u}_e, \beta_e)$, changing the input

In order to avoid the aforementioned problems, a possible solution is using a different input function. Indeed, using $A_0$ as base value is a particular case, and not a very realistic one. If we use the second input we introduced in section 1.5 instead, we have that $\beta$'s coefficient is not zero, and that the base value is an equilibrium state. However, we still have the problem that linearising by using our guess of $\beta$ instead of the real value makes us solve a different problem, with a systematic linearisation error.

The linearised problem actually has a very restricted use. If we consider the ideal case in which we use the actual $\beta$ in calculating the matrices, the measures aren't perturbed and are generated from the linearised model and we use the real $\beta$ outside the domain, the method converges excellently. However this case does not need such a complicated procedure, and is of limited interest since we suppose that we already know the value we are estimating.

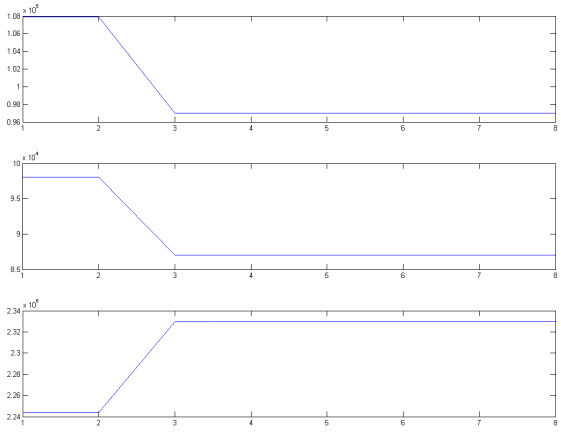

Figure 4.5: Parameter estimation without perturbing the measures. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step. The values are $\beta_1 = 9.7000 \cdot 10^4$, $\beta_2 = 8.7000 \cdot 10^4$, $\beta_3 = 2.3300 \cdot 10^5$ after 8 steps. The estimation error is due to the fact that the measure covariance matrix cannot be imposed null, since it must be invertible.

Even if we perturb the measures considerably (with a variance of 0.1, but keeping the error zero-mean), make a guess that differs from the real value by $10^4$, and keep the real value outside the domain (this is not a realistic case), the precision is still very good:

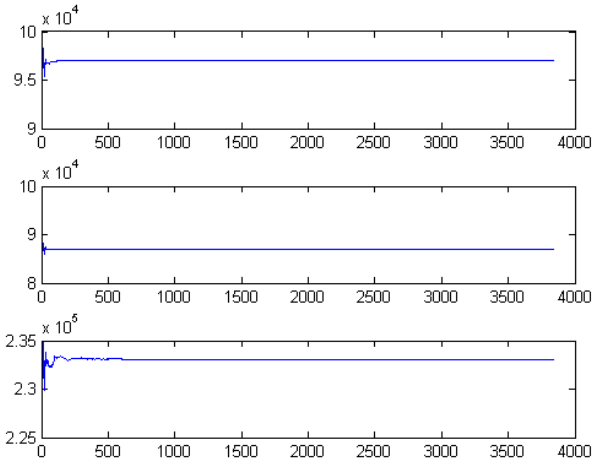


Figure 4.6: Parameter estimation with perturbed measures. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step. The values are $\beta_1 = 9.7000 \cdot 10^4$, $\beta_2 = 8.7001 \cdot 10^4$, $\beta_3 = 2.3300 \cdot 10^4$.

However, if we realistically suppose not to know the real value of $\beta$, the performance is much worse, and in some cases we are not even ensured that the Kalman filter betters the estimate. This is the case in the following example, where the initial guesses are $\beta_1 = 9.8 \cdot 10^4$, $\beta_2 = 8.8 \cdot 10^4$, $\beta_3 = 2.34 \cdot 10^5$, and the measures have not been perturbed. The stable values we find through Kalman filtering are $\beta_1 = 9.8139 \cdot 10^4$, $\beta_2 = 8.8218 \cdot 10^4$, $\beta_3 = 2.3566 \cdot 10^5$, which all are further from the real values than the initial guesses.

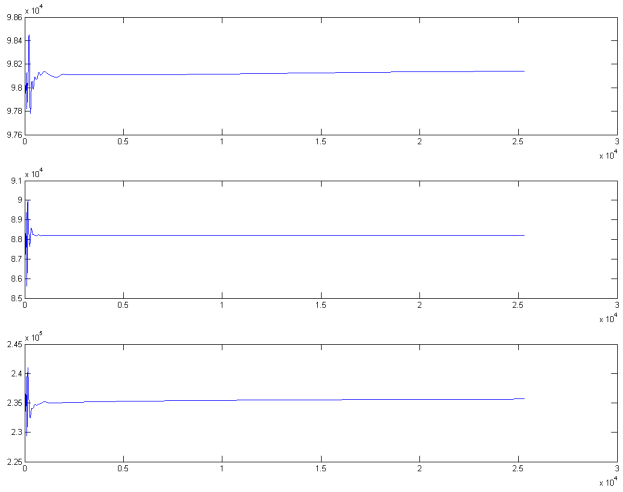

Figure 4.7: The linearised Kalman filter does not ensure a lowering of the estimation error. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

This fact is certainly not related to the Kalman filter itself, which as we have proven above is optimal. Instead it is due to the fact that we use an approximated problem, i.e. we solve a different problem from the one we started off with.

In addition there is another issue, regarding the foundations of Kalman filtering. Indeed, the quasi-linear matrix in equation (1.6) contains the parameter $\beta$. This means that if we perturb the parameter in order to get the guess $\beta_e = \beta + \delta\beta$, the perturbation $\delta\beta$ will multiply the variable $A$. This modeling error is multiplicative rather than additive, and cannot be inserted in the evolution equation, since it is in contrast with the hypotheses of the Kalman filter.

## 4.2 Kalman filter with time-varying matrices

The next step towards getting better estimates of the parameters is letting the matrices vary, following the variation of the estimated value. This means that they are recalculated at each time-step, so that the linearisation takes place repeatedly instead of only once for the entire simulation time. In this case we can choose to use any of the two inputs, since $\beta$'s coefficient will not be zero throughout the entire heartbeat in either of the cases. However, as we will see, the time-varying matrix approach gives rise to new stability problems.

### 4.2.1 Coding validation

In order to control if the program is correctly implemented, a step-wise control has been made. The first test consists in checking that if the measures are not perturbed, convergence is found quickly and precisely. As a start, we have chosen to use measures that are generated in the exact same time-varying matrix way in which the prediction evolves (clearly using an exact guess for the $\beta$ parameters). The first result is that if we make an exact initial guess the estimates aren't perturbed, since the residual is zero.

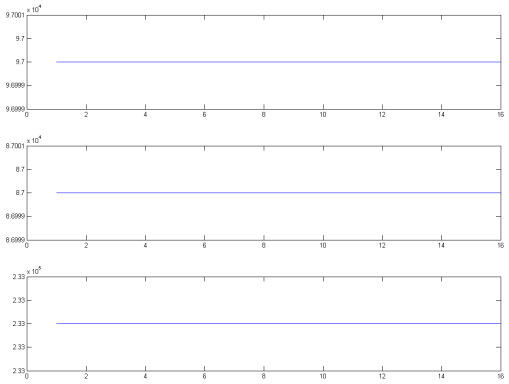

Figure 4.8: No matter which covariance matrices we choose, the estimates are not worsened if we make an exact guess. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

However this only tells us that the Kalman filter is implemented correctly, at least in some parts, and says nothing about the resolution of the conservation laws. These can also be tested stepwise, in order to validate the linearisation at every step. The first thing to control is that no numerical signals are created at the output and at the bifurcation. This can be noted by observing the variables throughout the simulation, by plotting them on a regular basis. A snapshot of this can be seen in the following figure:

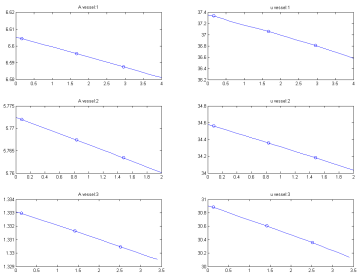

Figure 4.9: The instantaneous values of A and u on the three vessels

In this figure we can see that there is no peculiar behaviour on the borders, the signal is transferred without discontinuity at the bifurcation, and the flow continues steadily at the output, without reflections.

We then also have to control that the time-varying matrix signal is not too dissimilar to the non-linear flow, in order to hope for the Kalman filter to work. This can be done by plotting the difference between the two signals during the first heartbeat (all the others are identical).

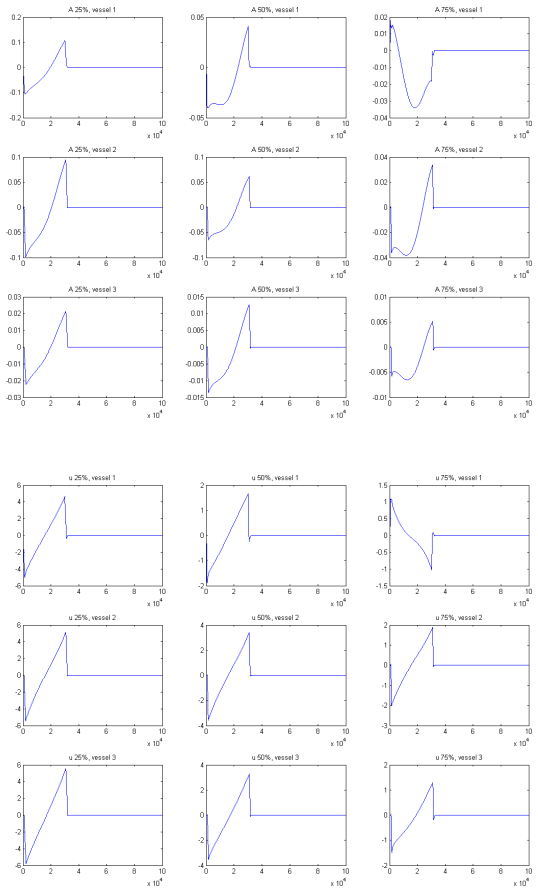


Figure 4.10: Difference between the values found through the non-linear and the time-varying matrix simulation at the measurement nodes.

The differences are all below 1.6% of the maximum value for the areas, and 16% of the maximum velocity. However we can see that the velocity errors are not far from having a zero integral, which means that the error is almost zero-mean.

### 4.2.2 Results with the first input

Another important test is seeing where an exact guess leads us in the case of perturbed measures. In the following figure we can see that the estimated value oscillates around the real value, but staying within a range of less than 0.01% of the real value:

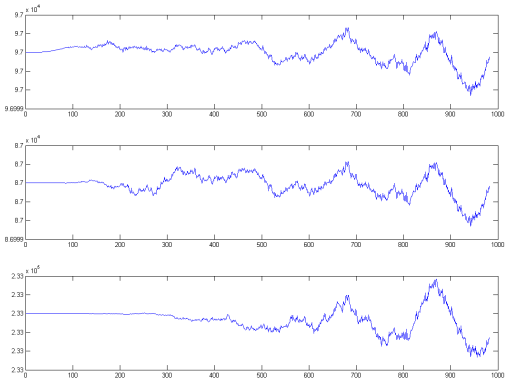

Figure 4.11: Estimation with an exact guess and perturbed measures. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

However, if we let the simulation continue, we observe a problem that will also be present further on: the estimated values suddenly and simultaneously increase or decrease considerably, until the $\beta$ parameters or the areas become negative.

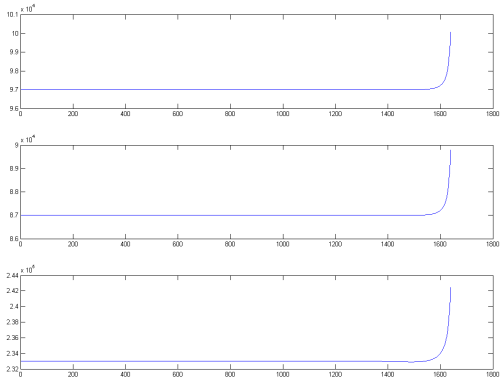

Figure 4.12: Unstable behaviour. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

The same behaviour can be observed if we make a random guess of $\beta$ (with a given perturbation covariance). Indeed the estimation first converges to the correct values, then suddenly the estimated values increase or decrease, without returning to the equilibrium. The values found in the following figure are $\beta_1 = 9.7001 \cdot 10^4$, $\beta_2 = 8.7003 \cdot 10^4$, $\beta_3 = 2.3301 \cdot 10^5$.

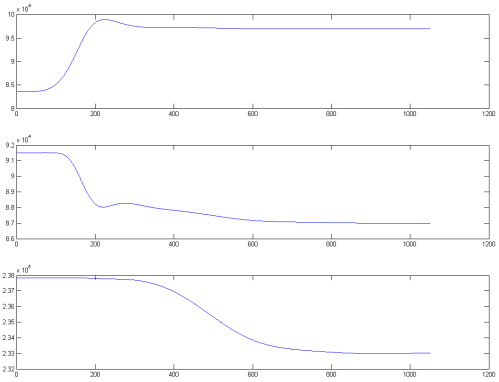

Figure 4.13: Convergence of the method with perturbed measures. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

However, if we let the simulation continue, we find that it exits the equilibrium state, diverging towards unacceptable values of $A$ and $\beta$.

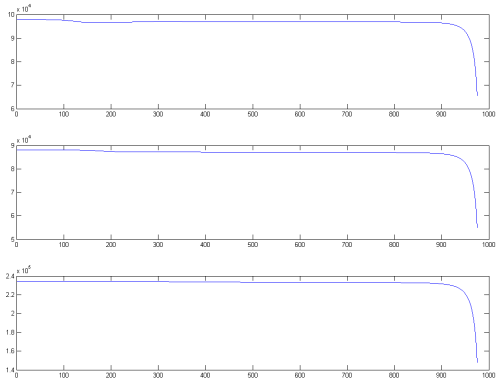

Figure 4.14: The method arrives to good estimates, then diverges. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

### 4.2.3 Changing the input

This stability problem can also be observed if we change the input signal. This rules out the hypothesis that the instability could be due to the fact that $\beta$'s coefficient could change sign when the input value is near $A_0$. Otherwise this could have been an explanation, since the coefficient multiplies the covariance matrix, making covariating variables countervariate.

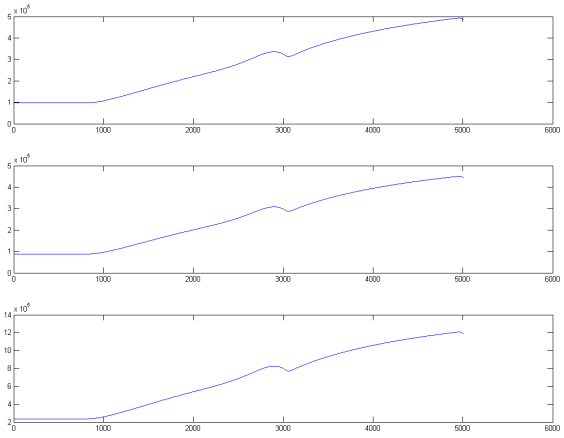

Figure 4.15: Estimates from perturbed measures, with an exact initial guess. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

It is also interesting to observe the first section of this behaviour, which is shown in the following figure. Indeed, the estimate begins oscillating, following the measure perturbation, but then stabilises on the correct value before diverging.

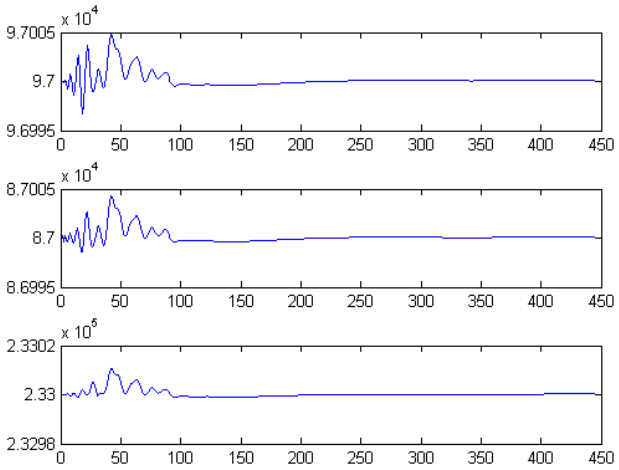


Figure 4.16: First part of the estimation evolution. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

By observing the same phenomenon in a case where the coefficient never goes below zero, we have to find alternative explanations. One possibility could be that the system is self-sustaining: when the state-covariances have reached a very low value (which always happens after a while, since we are adding new measures) a slightly too high (or low) guess for $\beta$ is made. As can be seen in most figures, all $\beta$s tend to have a similar trend (they are all covariating), which can be explained by the fact that a pressure continuity is imposed.

Since the system is "sure" about the state, it reduces the corresponding residuals (bringing the state to correspond to the measured value). This "pins" down the solution in certain points, but the other points do not necessarily follow. The high order polynomial then propagates, following the conservation law, which tends to increase the residuals at the measure points. Since the state cannot be corrected considerably, the $\beta$s are corrected instead. If underestimated $\beta$s give underestimated states (which often is the case), the estimation diverges.

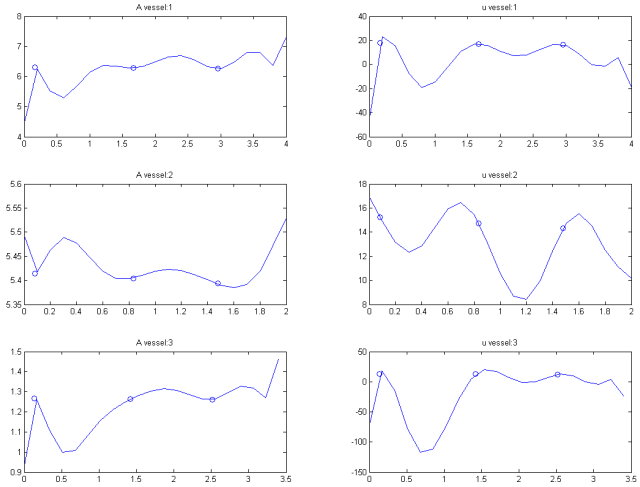


Figure 4.17: The state sometimes is pinned down at the measure points

However, this is only one of the possible explanations, and the problem has not been investigated further, since the behaviour disappears in considering the actual Extended Kalman filter.

### 4.2.4 Using measures from the non-linear process

All the aforementioned tests were made using measures that were generated in a time-varying matrix manner. However, the method we wanted to implement uses the output of the non-linear simulation as its measures. However, we still have the same stability problems, as can be seen in the following figures, obtained with perturbed measures and with non-exact initial guesses for $\beta$.

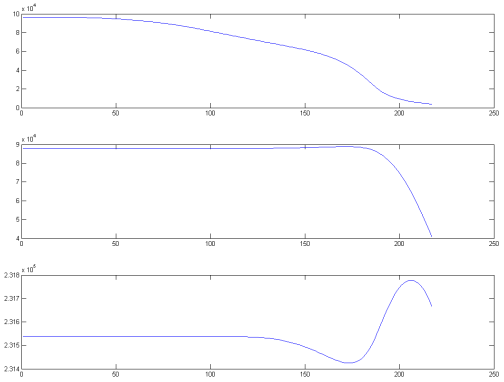

Figure 4.18: Estimation with the first input. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

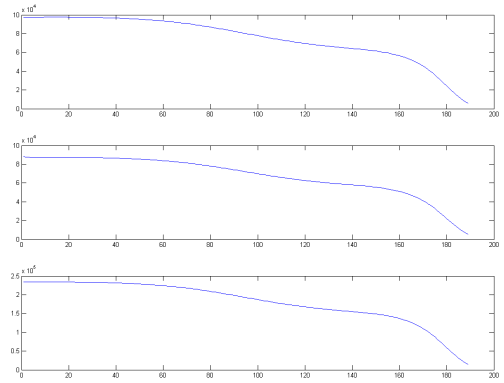

Figure 4.19: Estimation with the second input. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

### 4.2.5 Measuring only one state-variable

In real-life cases it may sometimes be impossible to obtain measures on both area and velocity. Additionally, if we have these measures, they may not be taken at the exact same time instants, which could cause additional problems. In order to face these issues, tests with only one measured variable have been made. In particular we have chosen only to consider measurements of the area.

This only needs slight changes in the coding: the measurement matrix has to be reduced in dimensions, and the vector of the measurements is half its prior length. The results are that the simulation takes much longer to

converge, as could easily be foreseen, but the accuracy is very good.

The final estimates are $\beta_1 = 9.7000 \cdot 10^4$, $\beta_2 = 8.7000 \cdot 10^4$, $\beta_3 = 2.3300 \cdot 10^5$. These are obtained with initial guesses that are 1000 above the actual values, with a measurement covariance of $10^{-2}$, an initial covariance of

$$\mathbf{P}_{00} = \mathbf{diag}(ones(length(\mathbf{U})); \ 1000^2 ones(3))$$

and the measurements, taken from the non-linear case, were perturbed with a standard deviation of $10^{-6}$. The second input was chosen.

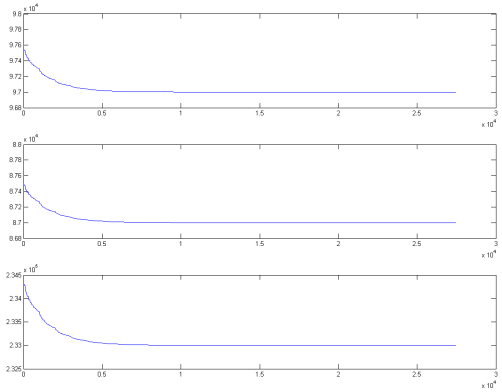

Figure 4.20: Estimates through measurements on the area alone. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

However, given the prior stability problems, one could think that these results are only partial, and that if the simulation continued it would diverge exactly as before. In order to check that this would not happen, it is useful to test the behaviour if we make an exact guess with perturbed measures. The results can be seen in the following figure:

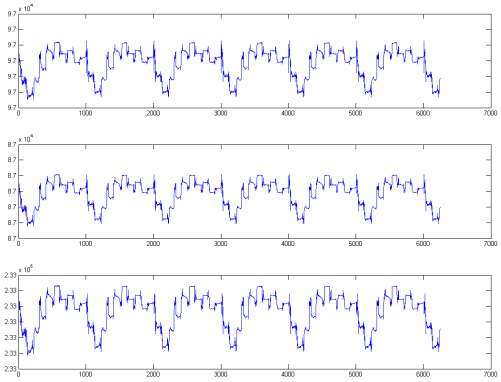

Figure 4.21: Estimation through an exact guess of $\beta$, with perturbed measures. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

The values found at the last step, when the simulation was interrupted, are $\beta_1 = 9.7000 \cdot 10^4$, $\beta_2 = 8.7000 \cdot 10^4$, $\beta_3 = 2.3300 \cdot 10^5$. The errors are therefore about $10^{-5}$ percent of the values. In addition it is very interesting to see that the trend is approximately periodical. This is given by the fact that the program uses the same measurements several times, repeatedly, in order to dramatically increase the actual number of values available. This strategy can be used since the parameters we are estimating are constant, and since the input signal is periodical, and we have measurements over the entire period.

### 4.2.6 Estimating at rarefied time-steps

In order to make the estimation process more realistic, we also have to consider that hoping to have measures at the frequency of the time-step we have chosen ($\Delta t = 10^{-5}$) may be too optimistic. Indeed this value was chosen to ensure stability in the simulation, by satisfying the CFL condition. However, taking measures at this rate means ending up with $10^5$ measures during a single heartbeat, which could be too costly or even impossible in some cases.

This leads to the urgency to adapt the Kalman filter to work with less measures. Fortunately this task is not difficult to implement, it's enough to take the prediction step at all times, while the correction step is only taken every $n$ time-steps. This strategy has been put in place in the previous case, where only measurements of the area were at hand. A measurement covariance of $10^{-2}$ has been chosen, along with a perturbation of the measures with standard deviation of $10^{-5}$ and an initial error on the guess of $\beta$ with a standard deviation of $10^3$. If we only take the correction step every $10^3$ steps (resulting in $10^3$ measures in 10 heartbeats) the resulting estimates are $\beta_1 = 9.7000 \cdot 10^4$, $\beta_2 = 8.7000 \cdot 10^4$, $\beta_3 = 2.3300 \cdot 10^5$. The measure-

ments are taken from a time-varying matrix model as a start, in order not to introduce too many changes at once. The convergence is shown in the following figure:

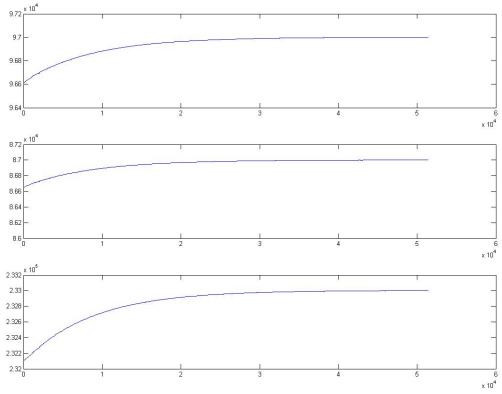

Figure 4.22: Estimates with measurements taken every $10^3$ time-steps. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

If instead we correct the state by the Kalman filter every $5 \cdot 10^3$ time-steps (resulting in 200 measures on 10 heartbeats), we find $\beta_1 = 9.6998 \cdot 10^4$, $\beta_2 = 8.6998 \cdot 10^4$, $\beta_3 = 2.3300 \cdot 10^5$ and the following trend:

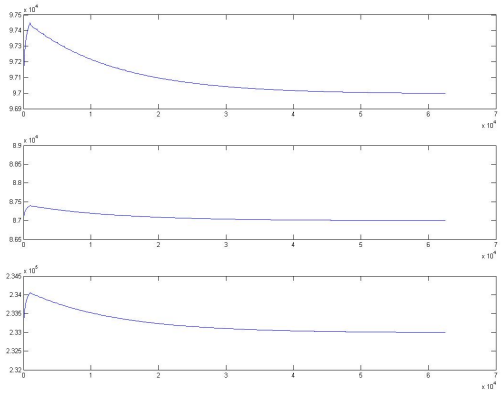

Figure 4.23: Estimates with measurements taken every $5 \cdot 10^3$ time-steps. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

If we take measurements from the non-linear model instead, approaching a real-life case even further, we have the following trend:

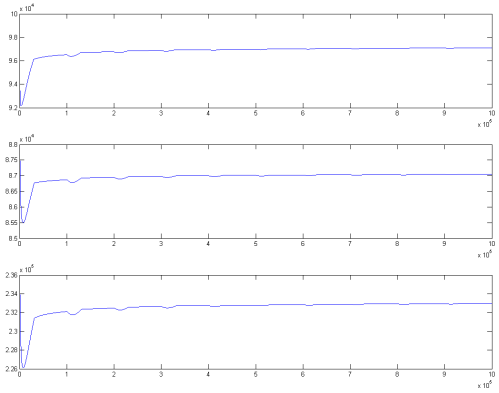

Figure 4.24: Estimates with measurements (from the non-linear model) taken every $5 \cdot 10^3$ time-steps. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

In this case we have used a measurement covariance of $10^{-2}$, an initial covariance guess of

$$\mathbf{P}_{00} = \mathbf{diag}(ones(length(\mathbf{U}));\ 1000^2 ones(3))$$

and initial guesses of $\beta$ which are $10^3$ higher (not random) than the real ones. The values we end up with after 10 heartbeats are $\beta_1 = 9.7101 \cdot 10^4$, $\beta_2 = 8.7045 \cdot 10^4$ and $\beta_3 = 2.3297 \cdot 10^5$. Considering that we are in a case in which we only measure one of the main variables, a different structure of the covariance matrix could turn out to be more realistic. Indeed the estimates of the area will always be more accurate than those on the velocity, and we could use a matrix like this:

\beta

$$\mathbf{P}_{00} = \mathbf{diag}(ones(length(\mathbf{A}));\ 10 * ones(length(\mathbf{u}));\ 1000^2 ones(3))$$

this also gives the possibility of making a more "flat" guess on the velocity, as to say increase its covariance. This was not possible before since the areas must remain positive (a Gaussian model is not very accurate), which limits the maximum covariance value. The same holds for the $\beta$s: having too high initial covariances could lead to negative elasticity parameters, which is unacceptable for our model.

## 4.3 Extended Kalman filter

Upon considering the extended Kalman filter, we need to go through a control phase similar to the one above. The first thing to check is whether exact guesses on the parameters and unperturbed measures yield exact results. This actually happens, which reassures us on the correct loading of

the measurements and on the way in which the residual is calculated. This also means that the way in which we calculate everything is identical in the two cases: upon taking the measures and upon applying the prediction step of the Kalman filter. In the EKF case we clearly only use the non-linear measurements, other kinds of measurements would introduce unnecessary errors. We have chosen to use the most realistic input signal, as to say the second one.

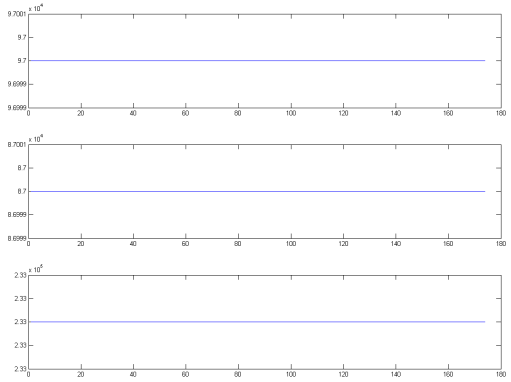

Figure 4.25: Estimates with exact guesses and unperturbed measures. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

### 4.3.1 Introducing measurement errors or inaccurate guesses

The second test consists in observing what happens if the measures are perturbed, but keeping exact guesses on the variables. We expect that the estimates become perturbed too, since the system does not know that the non-null residual is only due to measurement errors. However, we also hope to have a certain robustness, leading the estimates back to their correct values. This is what we actually observe, and after many iterations the estimates stabilize on the values $\beta_1 = 9.7000 \cdot 10^4$, $\beta_2 = 8.7000 \cdot 10^4$ and $\beta_3 = 2.3300 \cdot 10^5$, upon having perturbed the measurements with a standard deviation of $10^{-5}$. This leads us to the conclusion that the Kalman filter effectively filters the noise if left active long enough.

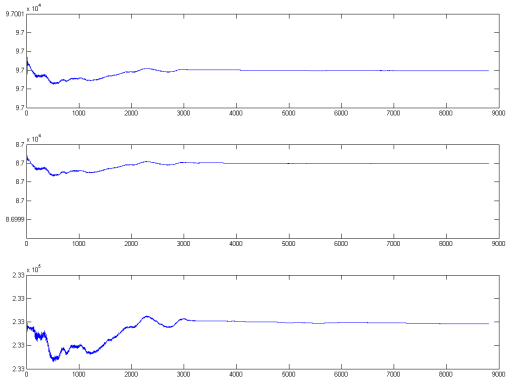

Figure 4.26: Estimates with perturbed measures and exact guesses on the $\beta$s. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

If on the other hand we only make inaccurate guesses, but keep unperturbed measures, we also want the system to converge. This is equally important, since afterwards we will have to face real-life cases, with perturbed measures and unknown parameters, and a failure or major inaccuracy in this test would leave little or no hope for convergence further on. Here we find a good convergence trend, and the values $\beta_1 = 9.6993 \cdot 10^4$, $\beta_2 = 8.6994 \cdot 10^4$ and $\beta_3 = 2.3298 \cdot 10^5$. Here the guesses have an initial error with a standard deviation of $10^3$.

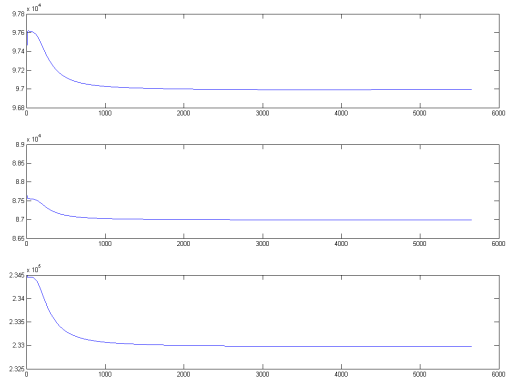

Figure 4.27: Estimates with inaccurate guesses and unperturbed measurements. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

In order to push the test even further, we could try to use a standard

deviation of $10^5$ on the initial guesses of $\beta$. This resembles the case in which we actually have no clue of their real values more. The values we find in this case are $\beta_1 = 9.7045 \cdot 10^4$, $\beta_2 = 8.7040 \cdot 10^4$ and $\beta_3 = 2.3310 \cdot 10^5$.

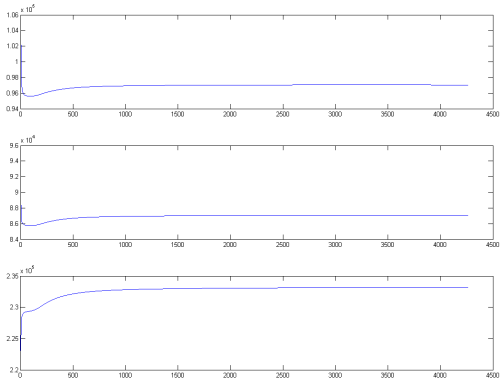

Figure 4.28: Estimates with highly inaccurate guesses and unperturbed measurements. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

### 4.3.2 Inaccurate guesses and perturbed measurements

The next step is testing the case in which both perturbations are present. If we use a measure error that has a standard deviation of $10^{-5}$ and make initial guesses on the parameters that are drawn from $\mathcal{N}(\beta, 10^8)$, where $\beta$ denotes the real value, we find the estimates: $\beta_1 = 9.7008 \cdot 10^4$, $\beta_2 = 8.7007 \cdot 10^4$ and $\beta_3 = 2.3302 \cdot 10^5$.

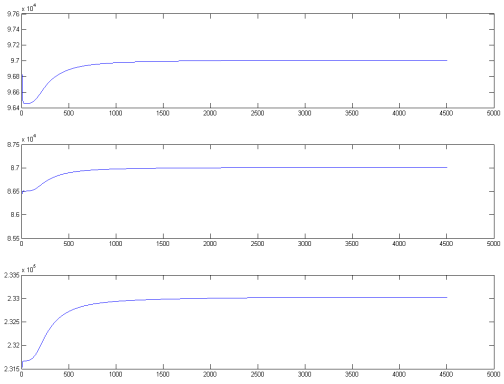

Figure 4.29: Estimates with inaccurate guesses and perturbed measurements. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

If we increase the standard deviations of both the measurement error (to $10^{-3}$) and the initial guesses (to $10^5$), we still obtain acceptable results: $\beta_1 = 9.7029 \cdot 10^4$, $\beta_2 = 8.7026 \cdot 10^4$ and $\beta_3 = 2.3307 \cdot 10^5$.

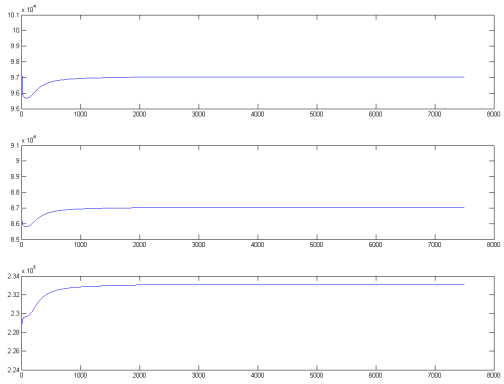

Figure 4.30: Estimates with highly inaccurate guesses and perturbed measurements. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

### 4.3.3 Measuring only one variable

Exactly as in the time-varying matrix case, it is interesting to test whether the method works adequately if we only measure one of the physical variables $A$ and $u$. We have again chosen only to measure the area. With a standard deviation of $10^{-5}$ and a guess inaccuracy of 1000 we find the

following values: $\beta_1 = 9.6940 \cdot 10^4$, $\beta_2 = 8.6946 \cdot 10^4$ and $\beta_3 = 2.3286 \cdot 10^5$. The following figure shows that initially the convergence rate is quite high, but soon slows down, arriving to an asymptotic behaviour.

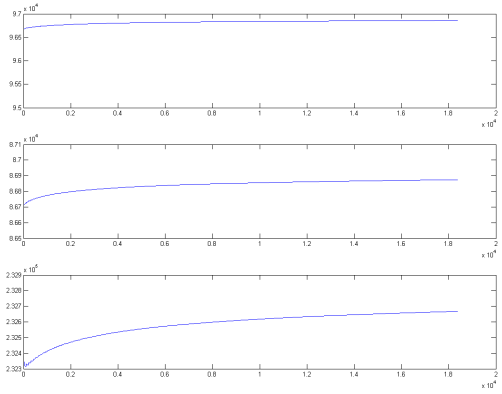

Figure 4.31: Estimates by measuring only the area, with inaccurate guesses and perturbed measurements. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

If we take measurements at rarefied time-steps, one every $10^3$ for instance, and only measure the area, we arrive to a close-to-real case. The values we find are $\beta_1 = 9.6860 \cdot 10^4$, $\beta_2 = 8.6874 \cdot 10^4$ and $\beta_3 = 2.3268, \cdot 10^5$, with a trend that is as follows:

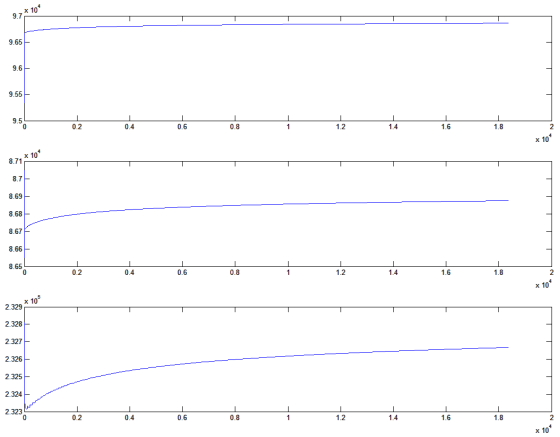

Figure 4.32: Estimates at rarefied time-steps (one measure every 1000 steps), with measurements only on the area. This is a plot of the values of $\beta$ (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

### 4.3.4 Using a $\beta$ that varies with x

In real cases the compliance of the wall may vary, it is clearly an approximation to assume that it is constant in the entire vessel. In order to describe this case the first, optimistic, attempt is that of letting $\beta$ be different on each node.

This leads to little or no change in the concept we applied previously; simply the augmented state vector is longer, as are the various covariance matrices and the evolution matrix. By doing this we allow the parameter to have a $r$-th grade polynomial trend. However, this approach (in its first naive version) turns out to be ineffective. Indeed, the simulations end because of a negative area, due to instability problems, as can be seen in the following figure:

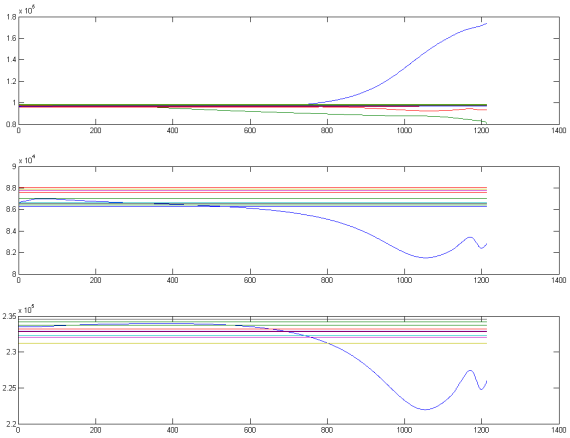

Figure 4.33: Estimates of one $\beta$ for every node. This is a plot of the 10 values of $\beta$ on each vessel (in order: vessel 1, vessel 2 and vessel 3) against the estimation step.

Here, each parameter has been perturbed with a different value and the measurements have been slightly perturbed (with a standard deviation of $10^{-5}$). The node on which the $\beta$ changes the most is the first one for all the vessels. On the first vessel the values on the second, third and last node also move from their initial guesses. It is interesting to notice how all the other estimates remain perfectly unperturbed.

This behaviour suggests that the system tends to change its first $\beta$ in order to match the measurements, but the information is not transferred to neighbouring nodes. This was clearly not the case when we only had one $\beta$ for each vessel, where a change in its value had effects all along the vessel. In addition, if we watch the time-evolution of the areas and velocities, we see that the systems "prefers" not to change these variables, but tends to modify the $\beta$s instead. It is possible that this behaviour disappears through

an accurate tuning of the covariance matrices, which however differs from the "flat" uninformative conditions in which we would be in a real case.

An interesting study would be that of testing how many different values of $\beta$ we can estimate on a vessel without having ambiguity problems. If we use fewer than $r + 1$ parameters we also have to think about a way to interpolate them in order to find out the values of $\beta$ on the remaining nodes. In imposing these trends we have to consider that continuous functions are preferable, since they are more realistic and generate fewer reflections of the waves. For the same reason smooth functions are preferable.

# Chapter 5

# Conclusions

Kalman filtering has turned out to be an efficient way to estimate the compliance of the vessel walls. Its time-varying and extended versions are accurate, robust and neither need many measurements, nor need them to come from different sources (measurements on one variable are enough). The estimated values can subsequently be inserted in the non-linear blood flow simulation, bringing to a patient-specific forecast of the effects of a local increase in $\beta$.

During the development of this project, several problems of different nature have appeared. These range from trivial, but very time-consuming, implementation errors (for instance looping with an increment of $10^{-5}$ yields roundoff errors thay may unphase the Kalman filter steps) to the task of interpreting abnormal behaviour. For instance, the stability problem that was highlighted in section (4.2.2) could at first analysis be misinterpreted as a bug in the code. This could then be very hard to exclude without seeing the very same code behave correctly in other cases. In addition, the mere fact of reinterpreting a CFD problem in a statistical perspective is not immediate, and deriving the matrix notation, which is original to this work, required many reformulations (even though it may seem trivial).

Some interesting further developments of this thesis could be a more detailed study of the case in which $\beta$ varies along the vessel and the estimation of $\beta$ together with the area $A_0$. Additionally, the study may be extended to a wider portion of the human arterial tree, and completed with terminal resistances, which simulate the reflections caused by smaller arteries or capillaries at the end of the tree.

# Appendix A

# The correction equation

## A.1 MAP and MMSE equivalency

As we have seen in section 2.2.1, the MAP estimation procedure arrives to the result:

$$\hat{\mathbf{x}}_{k,k}^{MAP} = \left(\mathbf{P}_{k,k-1}^{-1} + \mathbf{H}^T\Sigma_r^{-1}\mathbf{H}\right)^{-1}\left(\mathbf{H}^T\Sigma_r^{-1}\mathbf{y}_k + \mathbf{P}_{k,k-1}^{-1}\hat{\mathbf{x}}_{k,k-1}\right)$$

the exact same result can be found if we look for the a posteriori mean, instead of the mode. Indeed we had the posterior distribution in the following form:

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{Y}_k) \quad \propto \quad & \exp\left(-\frac{1}{2}\left(\left(\mathbf{y}_k - \mathbf{H}\mathbf{x}_k\right)^T\Sigma_r^{-1}\left(\mathbf{y}_k - \mathbf{H}\mathbf{x}_k\right) + \right.\right. \\ & \left.\left. + \left(\mathbf{x}_k - \hat{\mathbf{x}}_{k,k-1}\right)^T\mathbf{P}_{k,k-1}^{-1}\left(\mathbf{x}_k - \hat{\mathbf{x}}_{k,k-1}\right)\right)\right) \end{aligned}$$

the exponent can be manipulated into making the distribution Gaussian. If we expand all terms, leaving out $-\frac{1}{2}$:

$$\left(\mathbf{y}_k - \mathbf{H}\mathbf{x}_k\right)^T\Sigma_r^{-1}\left(\mathbf{y}_k - \mathbf{H}\mathbf{x}_k\right) + \left(\mathbf{x}_k - \hat{\mathbf{x}}_{k,k-1}\right)^T\mathbf{P}_{k,k-1}^{-1}\left(\mathbf{x}_k - \hat{\mathbf{x}}_{k,k-1}\right) =$$

$$\begin{aligned} = \quad & \mathbf{x}_k^T\mathbf{H}^T\Sigma_r^{-1}\mathbf{H}\mathbf{x}_k - 2\mathbf{x}_k^T\mathbf{H}^T\Sigma_r^{-1}\mathbf{y}_k + \mathbf{y}_k^T\Sigma_r^{-1}\mathbf{y}_k + \\ & + \mathbf{x}_k^T\mathbf{P}_{k,k-1}^{-1}\mathbf{x}_k - 2\hat{\mathbf{x}}_{k,k-1}^T\mathbf{P}_{k,k-1}^{-1}\mathbf{x}_k + \hat{\mathbf{x}}_{k,k-1}^T\mathbf{P}_{k,k-1}^{-1}\hat{\mathbf{x}}_{k,k-1} \end{aligned}$$

If we impose this expression to be of the form

$$\left(\mathbf{x}_k - \widetilde{\mathbf{x}}\right)^T\widetilde{\Sigma}^{-1}\left(\mathbf{x}_k - \widetilde{\mathbf{x}}\right)$$

we find that

$$\widetilde{\Sigma} = \left(\mathbf{H}^T\Sigma_r^{-1}\mathbf{H} + \mathbf{P}_{k,k-1}^{-1}\right)^{-1}$$

$$\widetilde{\mathbf{x}} = \widetilde{\Sigma}(\mathbf{H}^T\Sigma_r^{-1}\mathbf{y}_k + \mathbf{P}_{k,k-1}^{-1}\hat{\mathbf{x}}_{k,k-1})$$

this last expression is exactly the same that we found when we were looking for a MAP estimate. This proves that in the case of Gaussian distributions MAP and MMSE estimates are equivalent.

## A.2 Kalman gain derivation

We want to prove that expression (2.7) is equivalent to expression (2.9). Starting from (2.7) we apply the matrix inversion identity (2.8):

$$\left(\mathbf{P}_{k,k-1}^{-1} + \mathbf{H}^T\Sigma_r^{-1}\mathbf{H}\right)^{-1}\left(\mathbf{H}^T\Sigma_r^{-1}\mathbf{y}_k + \mathbf{P}_{k,k-1}^{-1}\hat{\mathbf{x}}_{k,k-1}\right) =$$

$$= \left(\mathbf{P}_{k,k-1} - \mathbf{P}_{k,k-1}\mathbf{H}^T\left(\Sigma_r + \mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\right)^{-1}\mathbf{H}\mathbf{P}_{k,k-1}\right)\left(\mathbf{H}^T\Sigma_r^{-1}\mathbf{y}_k + \mathbf{P}_{k,k-1}^{-1}\hat{\mathbf{x}}_{k,k-1}\right) =$$

$$= \left(\mathbf{P}_{k,k-1}\mathbf{H}^T\Sigma_r^{-1} - \mathbf{P}_{k,k-1}\mathbf{H}^T\left(\Sigma_r + \mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\right)^{-1}\mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\Sigma_r^{-1}\right)\mathbf{y}_k +$$

$$-\mathbf{P}_{k,k-1}\mathbf{H}^T\left(\Sigma_r + \mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\right)^{-1}\mathbf{H}\hat{\mathbf{x}}_{k,k-1} + \hat{\mathbf{x}}_{k,k-1}$$

Since the coefficients that multiply $\hat{\mathbf{x}}_{k,k-1}$ already correspond to those in equation (2.9), we only have to prove that $\mathbf{y}_k$'s coefficient matches:

$$\left(\mathbf{P}_{k,k-1}\mathbf{H}^T\Sigma_r^{-1} - \mathbf{P}_{k,k-1}\mathbf{H}^T\left(\Sigma_r + \mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\right)^{-1}\mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\Sigma_r^{-1}\right) =$$

$$= \mathbf{P}_{k,k-1}\mathbf{H}^T\left(\Sigma_r^{-1} - \left(\Sigma_r + \mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\right)^{-1}\mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\Sigma_r^{-1}\right)$$

In this expression we can recognise the use of Woodbury's identity. If we now use its inverse, with $\mathbf{A} = \Sigma_r$, $\mathbf{B} = \mathbf{C} = \mathbf{I}$ and $\mathbf{D} = \mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T$, and bear in mind that $(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$, we obtain:

$$\mathbf{P}_{k,k-1}\mathbf{H}^T\left(\Sigma_r + \mathbf{H}\mathbf{P}_{k,k-1}\mathbf{H}^T\right)^{-1}$$

which is the exact coefficient we find in expression (2.9).

# Bibliography


[CHE03] Chen Z. Bayesian filtering: From Kalman filters to particle filters, and beyond. *Adaptive Systems Lab., McMaster University, Hamilton, ON, Canada*, 2003.

[SFP03] Sherwin SJ, Formaggia L, Peiró J, Franke V. Computational modelling of 1D blood ow with variable mechanical properties and its application to the simulation of wave propagation in the human arterial system. *Int. J. Numer. Meth. Fluids* 2003; 43:673-700.

[FNQ02] Formaggia L, Nobile F, Quarteroni A. A One Dimensional Model for Blood Flow: Application to Vascular Prosthesis. *In: I. Babuška, T. Miyoshi, and P.G. Ciarlet, (ed.) Mathematical Modeling and Numerical Simulation in Continuum Mechanics; volume 19 of Lecture Notes in Computational Science and Engineering. Berlin: Springer-Verlag (2002)* pp. 137-153.

[HW08] Hesthaven JS, Warburton T. Nodal Discontinous Galerkin Methods; Algorithms, Analysis, and Applications. *Springer-Verlag* 2008.

[FF09] Fagnola F. Processi stocastici. *Lecture notes* 2009.

[GH08] Gray H. Gray's Anatomy: The Anatomical Basis of Clinical Practice, *40th edition, Churchill-Livingstone, Elsevier* 2008.

[VV05] Vele S, Villa U. Modelli 1D per la dinamica del usso ematico e dei suoi soluti. Tesi di Laurea presso il MOX-Politecnico di Milano, 2005.

[CHQ87] Canuto C, Hussaini MY, Quarteroni A, Tang TA. Spectral Methods in Fluid Dynamics. *Section 2.3 Springer-Verlag,* 1987.

[FSP03] Sherwin SJ, Franke V, Peiró J, Parker K. One-dimensional modelling of a vascular network in space-time variables. *J Engrg. Math.,* 2003; 47:217 - 250.

[CMT09] Chapelle D, Moireau P, Le Tallec P. Robust filtering for joint state-parameter estimation in distributed mechanical systems. *DCDS-A pg 65-84*, 2009.

[BDN08] Blum J, Le Dimet FX, Navon M. Data Assimilation for Geophysical Fluids. *Computational Methods for the Atmosphere and the Oceans, Elsevier*, 2008.

[PVR96] Pham DT, Verron J, Roubaud MC. A singular evolutive extended Kalman filter for data assimilation in oceanography. *Journal of Marine Systems, Elsevier,* 1996.

[MCT08] Moireau P, Chapelle D, Le Tallec P. Joint state and parameter estimation for distributed mechanical systems. *Computer methods in applied mechanics and engineering, Elsevier* 2008.