Diffusive Realization of a Lyapunov Equation Solution, and its FPGA Implementation

Y. Yakoubi, Michel Lenczner, G. Goavec-Merou, R. Couturier, J.-M. Friedt

Diffusive Realization of a Lyapunov Equation Solution, and its FPGA Implementation

Y. Yakoubi * **M. Lenczner** ** **G. Goavec-Merou** **
R. Couturier *** **J.M. Friedt** **

* *UPMC Univ Paris 06, Laboratoire Jacques-Louis Lions, F-75005, Paris CEDEX, FRANCE, (e-mail: yyakoubi@ann.jussieu.fr)*
** *FEMTO-ST, Time-Frequency Department, 26, chemin de l'Epitaphe, 25030 Besançon, FRANCE, (e-mail: michel.lenczner@utbm.fr, gwenhael.goavec@femto-st.fr, jean-michel.friedt@femto-st.fr)*
*** *University of Franche-Comte, LIFC, IUT Belfort-Montbéliard, rue Engel Gros, 90000 Belfort, FRANCE, (e-mail: raphael.couturier@iut-bm.univ-fcomte.fr)*

Abstract: In Yakoubi [2010] and Lenczner et al. [2010] we developed a theoretical framework of diffusive realization for state-realizations of some linear operators. Those are solutions to certain operator linear differential equations in one-dimensional bounded domains. We also illustrated the theory and developed a numerical method for a Lyapunov equation arising from optimal control theory of the heat equation. However, the principles of our numerical methods were only sketched, and now we provide more details. Then, we do not only provide validation results of the method, but we also report our experience in its implementation on a Field Programmable Gate Arrays (FPGA), for the purpose of promoting embedded real-time computation.

Keywords: Diffusive Realization; Lyapunov Equation; Distributed Control; FPGA.

1. INTRODUCTION

One of the main recognized advantages of the diffusive realization of a linear operator is its very low computational cost, see the papers of G. Montseny and of D. Matignon, e.g. Laudebat et al. [2004], and Hélie et al. [2007], for representations of various pseudodifferential operators, and for their approximation. Those of C. Lubich and collaborators, e.g. López-Fernández et al. [2005], apply a similar idea to convolution operators, and they introduced optimized adaptive numerical methods. Notice that fast operator realization is essential for real-time control. Until now, realization of a linear operator $u \mapsto z = Pu$, by the diffusive realization method, has been addressed to causal operators when the kernel p of P is explicitely known, and analytic in its second variable, see Montseny [2005]. Here, we cover the case where P is solution of an operator equation, so p is not explicitly given nor analytic. Our approach is presented and illustrated with the example of P solution to the Lyapunov equation,

$$\frac{d^2}{dx^2}Pu + P\frac{d^2}{dx^2}u = Qu \tag{1}$$

in $\omega = (0,1)$, for all u vanishing at the boundary, and where Q is another linear operator. This problem comes from optimal filtering or control theory of the heat equation,

$$\frac{\partial T}{\partial t} - \frac{\partial^2 T}{\partial x^2} = q \text{ in } \omega$$

with Dirichlet boundary conditions. Our new method was announced in Lenczner and Montseny [2005], it was fully developed in Lenczner et al. [2010], and the theoritical part with some numerical results were presented in Yakoubi [2010]. The present paper focus mainly on the numerical method which is not published yet. A second aim of this paper, is implementation in view of embeded real-time computation. Field Programmable Gate Arrays (FPGA) is the best today choice for real-time, embeded, massive, and low-cost computation. The main drawback of these processors, compared to usual computers, is that they require a good expertise in digital electronics for application implementation. Helpful dedicated software exist to help FPGA implementation, but until today, due to FPGA complexity, the solutions that they give are often not very efficient. So, we have elaborated a solution from scratch to execute our algorithm in a small FPGA, namely the Spartan3A by Xilinx.

The paper is organized as follows. The diffusive realization of P is recalled in Section 2, then in Section 3 and 4, we propose a numerical method, and we present related numerical results. Last, we discuss our FPGA implementation in Section 5.

2. DIFFUSIVE REPRESENTATION OF P

In this section we recall the diffusive realization of the solution P to the Lyapunov equation (1) published in

Lenczner et al. [2010]. We consider the kernel formulation of the operator P

$$Pu(x) = \int_0^1 p(x,y)u(y)\ dy,$$

and its unique decomposition $(Pu) = z^+ + z^-$ into causal and anti-causal parts,

$$z^+ = \int_0^x p(x,y)u(y)\ dy \quad \text{and} \quad z^- = \int_x^1 p(x,y)u(y)\ dy.$$

Throughout this paper, we shall use the superscripts $+$ or $-$ to refer to causal or anti-causal operators, and the convention $\mp = -(\pm)$.

We recall that the kernel p is the unique solution to the boundary value problem

$$\begin{cases} -\Delta p = q & \text{in the square } (0,1)^2, \\ \text{and } p = 0 & \text{on the square boundary } (0,1)^2, \end{cases}$$

where q is the kernel of Q. The realization of z^+ and of z^- may be formulated thanks to the diffusive representation, see Lenczner and Montseny [2005] and Montseny [2005], in the form

$$\begin{aligned} z^+(x) &= \int_{\mathbb{R}} \mu^+(x,\xi)\psi^+(x,\xi)\ d\xi \\ \text{and } z^-(x) &= \int_{\mathbb{R}} \mu^-(x,\xi)\psi^-(x,\xi)\ d\xi, \end{aligned} \tag{2}$$

where both ψ^+ and ψ^- store a part of the *history* of the input data u. They are respectively solution to the forward and backward ordinary differential equation in x,

$$\begin{aligned} \partial_x\psi^+(x,\xi) + \theta^+(\xi)\psi^+(x,\xi) &= u(x) \\ \text{with } \psi^+(0,\xi) &= 0, \end{aligned} \tag{3}$$

$$\begin{aligned} \text{and } \partial_x\psi^-(x,\xi) - \theta^-(\xi)\psi^-(x,\xi) &= u(x) \\ \text{with } \psi^-(1,\xi) &= 0, \end{aligned} \tag{4}$$

parametrized by $\xi \in \mathbb{R}$. We notice that ψ^+ and ψ^- are defined independently of P. Conversely, the coefficients μ^+ and μ^-, called *diffusive symbols*, depend on P but not on u. The functions $\xi \mapsto \theta^+(\xi)$ and $\theta^-(\xi)$ parametrize two closed paths in the complex plane, satisfying the cone condition, and enlacing the singularities of the Laplace transform $\mathcal{P}^+$ and $\mathcal{P}^-$ defined hereafter. The diffusive symbol derivation is achieved within several steps. First, the functions

$$y \mapsto p(x, x-y) \quad \text{and} \quad y \mapsto p(x, x+y), \tag{5}$$

corresponding to the causal part and the anti-causal part of the impulse response, are analytically extended to $\mathbb{R}^+$. Then, we assume that the Laplace transforms $\mathcal{P}^+$ and $\mathcal{P}^-$, with respect to y, of the extended causal and anti-causal parts of the impulse response, are well-defined in $\mathbb{C}^+$, and that they admit holomorphic extensions vanishing at infinity. Finally, we show that the diffusive symbols are given by

$$\mu^\pm(x,\xi) = \mp\frac{\theta^{\pm\prime}(\xi)}{2i\pi}\mathcal{P}^\pm\left(x, -\theta^\pm(\xi)\right). \tag{6}$$

3. DIFFUSIVE REALIZATION APPROXIMATION

The approximation presented in this section are formulated in the particular case of Lyapunov Equation (1). In Subsection 3.1, we recall a Petrov-Galerkin method to approximate the symbols $\mu^\pm$. Then, computational algorithms for *history functions* $\psi^\pm$ and for $z^\pm$ are derived in Subsection 3.2 and 3.3.

3.1 Symbol Approximation

First, we choose the contours $-\theta^\pm$ proposed by J.A.C. Weideman and L.N. Trefethen in Weideman and Trefethen [2007], in the context of inverse Laplace transform computation, namely a parabola

$$-\theta^\pm(\xi) = \theta_P\left(i\xi + 1\right)^2 \quad \text{for } \xi \in \mathbb{R}, \tag{7}$$

and a hyperbola

$$-\theta^\pm(\xi) = \theta_H\left(1 + \sin\left(i\xi - \alpha\right)\right) \quad \text{for } \xi \in \mathbb{R}, \tag{8}$$

for some positive real numbers θ_P, θ_H, and α the hyperbola asymptotic angle. In Lenczner et al. [2010], we have derived two Petrov-Galerkin formulations satisfied by two symbols μ^{N+} and μ^{N-} yielding approximations $\int_{\mathbb{R}} \mu^{N\pm}\psi^\pm(u)\ d\xi$ of $z^\pm$. Here we simply recall them without repeating their justification. The symbols $\mu^{N\pm}$ are linear combinations,

$$\mu^{N\pm}(x,\xi) = \mp\frac{(\theta^\pm(\xi))'}{2i\pi}\sum_{k=0,\ell=0}^{N_1,N_2} \varphi_k^1(x)\zeta_\ell^\pm(x,\xi)\mu_{k\ell}^\pm, \tag{9}$$

of base functions which are products of polynomials in x, $\varphi_k^1(x) = (1-x)x^{k+1}$, satisfying the Dirichlet condition, with rational fractions in ξ,

$$\zeta_\ell^\pm(x,\xi) = \frac{1}{\ell + 1 - \theta^\pm(\xi)} - \frac{e^{-h^\pm(x)}}{\ell - \theta^\pm(\xi)},$$

where $h^+(x) = x$, $h^-(x) = 1-x$. Then, the test functions are linear combinations,

$$\widetilde{v}^{N\pm}(x,y) = \sum_{k=0,\ell=0}^{N_1,N_2} \varphi_k^1(x)\varphi_\ell^{3\pm}(x,y)v_{k\ell}^\pm, \tag{10}$$

with $\varphi_\ell^{3\pm}(x,y) = (e^y - e^{h^\pm(x)})e^{\ell y}$. The right-hand sides are decomposed as linear combinations,

$$q(x, x \mp y) \approx \sum_{\ell=0}^{N_2} \varphi_\ell^{2\pm}(x,y)q_\ell^{N_2}(x), \tag{11}$$

of exponential polynomials $\varphi_\ell^{2\pm}(x,y) = (e^{-y} - e^{-h^\pm(x)})e^{-\ell y}$ in the y-variable. Finally, for the matrix $K^\pm = \begin{pmatrix} 1 & \pm 1 \\ 0 & \mp 1 \end{pmatrix}$, the linear operator $L^\pm(w) = \int_0^{h^\pm(x)} w\ e^{-\theta^\pm y}\ dy$, and the differential operator $D_\lambda = (\partial_x, \lambda)$, the symbols $\mu^{N\pm}$ are solutions to the weak formulation,

$$\begin{aligned} &\int_\omega \int_{\mathbb{R}} K^\pm D_{-\theta^\pm}^T \mu^{N\pm} K^\pm L^\pm(\nabla\widetilde{v}^{N\pm})\ d\xi dx \\ &= \int_\omega \int_{\mathbb{R}} \nu^{N\pm}, L^\pm(\widetilde{v}^{N\pm})\ d\xi dx, \end{aligned} \tag{12}$$

for all $\widetilde{v}^{N\pm}$as in (10).

Remark 2 We have used a spectral method to discretize both x- and y-directions. In the y-direction we actually need to use global basis functions so that they can be analytically extended. On the contrary, there is no particular restriction regarding approximations in the x-direction. For instance a local basis as a finite element basis might be used.

3.2 Discretization of ψ with respect to x

Two x-discretizations have been considered. They are based on two different interpolations of discrete inputs $(u_n)_n$ located at regularly spaced nodes $(x_n)_n$ separated by a distance h. In the interval $[x_n, x_{n+1})$, the first one is piecewise constant $u(x) = u_n$, and the second one is piecewise linear and continuous, $u(x) = u_n + \frac{u_{n+1}-u_n}{h}(x - x_n)$. We detail the calculation for both causal and anti-causal parts, since it is not so trivial to deduce from each other. To proceed, we firstly consider the integral forms of (3-4), i.e.

$$\psi^+(x,\xi) = \int_0^x e^{-\theta^+(\xi)(x-y)} u(y)\ dy, \tag{13}$$

$$\psi^-(x,\xi) = -\int_x^1 e^{\theta^-(\xi)(x-y)} u(y)\ dy. \tag{14}$$

In particular, at a point $x = x_{n+1}$, we have

$$\begin{aligned}\psi^+(x_{n+1},\xi) &= \int_0^{x_{n+1}} e^{-\theta^+(\xi)(x_{n+1}-y)} u(y)\ dy \\ &= \int_0^{x_n} e^{-\theta^+(\xi)(x_{n+1}-y)} u(y)\ dy \\ &+ \int_{x_n}^{x_{n+1}} e^{-\theta^+(\xi)(x_{n+1}-y)} u(y)\ dy,\end{aligned}$$

and at a point $x = x_n$,

$$\begin{aligned}\psi^-(x_n,\xi) &= -\int_{x_n}^1 e^{\theta^-(\xi)(x_n-y)} u(y)\ dy \\ &= -\int_{x_{n+1}}^1 e^{\theta^-(\xi)(x_n-y)} u(y)\ dy \\ &- \int_{x_n}^{x_{n+1}} e^{\theta^-(\xi)(x_n-y)} u(y)\ dy.\end{aligned}$$

We deduce the recurrence relations

$$\begin{aligned}\psi^+(x_{n+1},\xi) &= e^{-\theta^+(\xi)h}\psi^+(x_n,\xi) \\ &+ \int_{x_n}^{x_{n+1}} e^{-\theta^+(\xi)(x_{n+1}-y)} u(y)\ dy, \quad \psi^+(u)(0,\xi) = 0, \\ \text{and } \psi^-(x_n,\xi) &= e^{-\theta^-(\xi)h}\psi^-(x_{n+1},\xi) \\ &- \int_{x_n}^{x_{n+1}} e^{\theta^-(\xi)(x_n-y)} u(y)\ dy, \quad \psi^-(u)(1,\xi) = 0.\end{aligned}$$

► *Piecewise constant interpolation of u* Defining the parameters $\alpha^\pm(\xi) = e^{-\theta^\pm(\xi)h}$, and $\beta^\pm(\xi) = \frac{\alpha^\pm(\xi)-1}{-\theta^\pm(\xi)}$, the integrals turns to be equal to:

$$\int_{x_n}^{x_{n+1}} e^{-\theta^+(\xi)(x_{n+1}-y)} u(y)\ dy \simeq \beta^+(\xi) u_n,$$

and

$$\int_{x_n}^{x_{n+1}} e^{\theta^-(\xi)(x_n-y)} u(y)\ dy \simeq \beta^-(\xi) u_n.$$

So, the recurrence relations yields

$$\begin{aligned}\psi^+(x_{n+1},\xi) &\simeq \alpha^+(\xi)\psi^+(x_n,\xi) + \beta^+(\xi)u_n, \\ \psi^+(0,\xi) &= 0, \\ \text{and } \psi^-(x_n,\xi) &\simeq \alpha^-(\xi)\psi^-(x_{n+1},\xi) - \beta^-(\xi)u_n, \\ \psi^-(1,\xi) &= 0.\end{aligned} \tag{15}$$

Notice that this recurence relation was already found by Casenave [2009] for the causal part.

► *Piecewise linear interpolation of u* Here we pose $\eta^\pm(\xi) = \pm\frac{\alpha^\pm(\xi)}{-\theta^\pm(\xi)} - \frac{\beta^\pm(\xi)}{-\theta^\pm(\xi)h}$, and $\delta^\pm(\xi) = \mp\frac{1}{-\theta^\pm(\xi)} + \frac{\beta^\pm(\xi)}{-\theta^\pm(\xi)h}$, so the integrals are

$$\int_{x_n}^{x_{n+1}} e^{-\theta^+(\xi)(x_{n+1}-y)} u(y)\ dy \simeq \eta^+(\xi)u_n + \delta^+(\xi)u_{n+1},$$

and

$$\int_{x_n}^{x_{n+1}} e^{\theta^-(\xi)(x_n-y)} u(y)\ dy \simeq \delta^-(\xi)u_n + \eta^-(\xi)u_{n+1},$$

so the recurrence relations are rewritten as

$$\begin{aligned}\psi^+(x_{n+1},\xi) &\simeq \alpha^+(\xi)\psi^+(x_n,\xi) + \eta^+(\xi)u_n + \delta^+(\xi)u_{n+1}, \\ &\text{with } \psi^+(0,\xi) = 0, \\ \psi^-(x_n,\xi) &\simeq \alpha^-(\xi)\psi^-(x_{n+1},\xi) + \delta^-(\xi)u_n + \eta^-(\xi)u_{n+1}, \\ &\text{with } \psi^-(1,\xi) = 0.\end{aligned} \tag{16}$$

López-Fernández et al. [2005] have developed a computation of ψ, in the linear interpolation case, which presents similarities with ours.

3.3 Approximation of the integrals in z^+ and z^- in (2)

Using the above recurrence relations, we establish the final approximations z^+_{n+1} and z^-_n of z^+ and z^- at the input nodes. We notice that a direct application of the residu theorem yields to eliminate the terms without exponential,

$$\int_{\mathbb{R}} \frac{\mu^{N\pm}(x,\xi)}{\theta^\pm(\xi)}\ d\xi = \int_{\mathbb{R}} \frac{\mu^{N\pm}(x,\xi)}{\theta^{\pm 2}(\xi)}\ d\xi = 0. \tag{17}$$

► *Piecewise constant interpolation of u* From the recurrence relation (15),

$$\begin{aligned}z^+(x_{n+1}) &\simeq \int_{\mathbb{R}} \mu^{N+}(x_{n+1},\xi)\,\psi^+(x_{n+1},\xi)\ d\xi \\ &\simeq \int_{\mathbb{R}} \mu^{N+}(x_{n+1},\xi)\left(\alpha^+(\xi)\psi^+(x_n,\xi) + \beta^+(\xi)u_n\right)\ d\xi \\ &= \int_{\mathbb{R}} \mu^{N+}(x_{n+1},\xi)\left(\alpha^+(\xi)\psi^+(x_n,\xi) + \gamma^+(\xi)u_n\right)\ d\xi,\end{aligned}$$

and similarly

$$\begin{aligned}z^-(x_n) &\simeq \int_{\mathbb{R}} \mu^{N-}(x_n,\xi) \\ &\times \left(\alpha^-(\xi)\psi^-(x_{n+1},\xi) - \gamma^-(\xi)u_n\right)\ d\xi\end{aligned}$$

for the anti-causal part, whith $\gamma^\pm(\xi) = \frac{\alpha^\pm(\xi)}{-\theta^\pm(\xi)}$, $\overline{\gamma}^\pm(\xi) = \pm\frac{\alpha^\pm(\xi)}{-\theta^\pm(\xi)} - \frac{\gamma^\pm(\xi)}{-\theta^\pm(\xi)h}$, and $\overline{\delta}^\pm(\xi) = \frac{\gamma^\pm(\xi)}{-\theta^\pm(\xi)h}$. Evaluating the integrals thanks to the trapezoidale rule with $2M+1$ quadrature nodes regularly spaced at a distance h_ξ yields the final aproximations,

$$z^+_{n+1} = h_\xi \sum_{k=-M}^{M} \mu^{N+}_{n+1,k}\left(\alpha^+_k\psi^+_{n,k} + \gamma^+_k u_n\right),$$

$$z^-_n = h_\xi \sum_{k=-M}^{M} \mu^{N-}_{n,k}\left(\alpha^-_k\psi^-_{n+1,k} - \gamma^-_k u_n\right).$$

► *Piecewise linear interpolation of u* We follow the same route to find

$$z^+_{n+1} = h_\xi \sum_{k=-M}^{M} \mu^{N+}_{n+1,k} \left(\alpha^+_k \psi^+_{n,k} + \overline{\gamma}^+_k u_n + \overline{\delta}^+_k u_{n+1} \right),$$

$$z^-_n = h_\xi \sum_{k=-M}^{M} \mu^{N-}_{n,k} \left(\alpha^-_k \psi^-_{n+1,k} + \overline{\delta}^-_k u_n + \overline{\gamma}^-_k u_{n+1} \right),$$

with $\theta^\pm_k = \theta^\pm(\xi_k)$, $\alpha^\pm_k = \alpha^\pm(\xi_k)$, $\overline{\beta}^\pm_k = \overline{\beta}^\pm(\xi_k)$, $\overline{\gamma}^\pm_k = \overline{\gamma}^\pm(\xi_k)$, and $\overline{\delta}^\pm_k = \overline{\delta}^\pm(\xi_k)$.

3.4 Balance of Error Estimates

We replace $\mu^\pm$ (or $\mathcal{P}^\pm$) and $\psi^\pm$ by their approximations $\mu^{N\pm}$ (or $\mathcal{P}^{N\pm}$) and the x-discretization of $\psi^\pm$ in the integral (2). We establish that the approximation of $z^\pm$ the realization in (2) can be written like a linear combination of inverse Laplace transform $\mathcal{L}^{-1}$.

► *In the piecewise linear interpolation case:*

$$z^\pm_n \approx \sum_{j \in J^\pm_n} \mathcal{L}^{-1}$$
$$\left[F^\pm_{1n}(-\theta^\pm) u_j + F^\pm_{2n}(-\theta^\pm) \frac{u_{j+1} - u_j}{h} \right] (\pm(x_n - x_j))$$
$$- \left[F^\pm_{1n}(-\theta^\pm) u_{j+1} + F^\pm_{2n}(-\theta^\pm) \frac{u_{j+1} - u_j}{h} \right] (\pm(x_n - x_{j+1})) .$$

► *In the piecewise constant interpolation case:*

$$z^\pm_n \approx \sum_{j \in J^\pm_n} u_j \mathcal{L}^{-1}$$

$\left[F^\pm_{1n}\left(-\theta^\pm\right)\right](\pm(x_n - x_j)) - \left[F^\pm_{1n}\left(-\theta^\pm\right)\right](\pm(x_n - x_{j+1}))$, for all $n \in \{0, 1..., \mathcal{N}\}$, with $\mathcal{N} = 1/h - 1$ the number of intervals in the x-variable, $J^+_n = \{0, ..., n-1\}$, $J^-_n = \{n, ..., \mathcal{N}-1\}$, $x_j = jh$, $u_j = u(x_j)$, $z^\pm_n = z^\pm(x_n)$, $F^\pm_{1n}(-\theta^\pm) = \frac{\mathcal{P}^{N\pm}\left(x_n, -\theta^\pm(\xi)\right)}{-\theta^\pm(\xi)}$ and $F^\pm_{2n}(-\theta^\pm) = \frac{\mathcal{P}^{N\pm}\left(x_n, -\theta^\pm(\xi)\right)}{\theta^{\pm 2}(\xi)}$. Following Weideman and Trefethen [2007], we approximate the inverse Laplace transforms

$$\mathcal{L}^{-1}[F^\pm_{1n}(-\theta^\pm)](\overline{x}^\pm), \quad \mathcal{L}^{-1}[F^\pm_{2n}(-\theta^\pm)](\overline{x}^\pm)$$

at $\overline{x}^\pm = \pm(x_n - x_j)$ and $\overline{x}^\pm = \pm(x_n - x_{j+1})$ using a numerical integration formula along the integration contours (7-8). The optimization of these contours is founded on a balance between the truncation error estimate and the discretization error estimate for the numerical integration of the Laplace inversion at points $\overline{x}^\pm \in I = \{h,\ 2h, .., 1\}$ excluding the point 0. At $\overline{x}^\pm = 0$, we know that $\mathcal{L}^{-1}[F^\pm_{1n}(-\theta^\pm)](0) = \mathcal{L}^{-1}[F^\pm_{2n}(-\theta^\pm)](0) = 0$ thanks to the formula (17). The ratio between the upper and lower bounds of the set I, i.e. $\Lambda = \frac{1}{h}$ is very large for a fine mesh and the numerical inversion of Laplace transform is relatively expensive. To avoid this problem, we observed an improvement by changing the formulas obtained in Weideman and Trefethen [2007] by fixing $\Lambda = 6$. The minimum quadrature interval lengths are to be equal to

$$L_P = \sqrt{8\Lambda + 1} = 7, \quad L_H(\alpha^*) = \cosh^{-1} \frac{5\pi - 8\alpha^*}{(4\alpha^* - \pi)\sin\alpha^*},$$

where α^* is the unique argument of the maximization problem,

$$\alpha^* = \arg \max_{\alpha \in (0, \frac{\pi}{2})} \frac{\pi(\pi - 2\alpha)}{L_H(\alpha)} = 1.0641.$$

Then, it is shown that for the optimal contour parameters,

$$\theta_P = \frac{\pi M}{4 L_P}, \quad \alpha = \alpha^*, \quad \theta_H = \frac{\pi(4\alpha^* - \pi) M}{L_H(\alpha^*)},$$

the quadrature error estimate e^P_M and e^H_M, for parabolic (P) and hyperbolic (H) paths, are exponentially decreasing with respect to M the number of points of the integration contours,

$$e^P_M = C_P e^{-AM} \quad \text{and} \quad e^H_M = C_H e^{-BM}.$$

The two constants C_P and C_H are uniform with respect to M and the decay rates depends on the quadrature interval length,

$$A = \frac{2\pi}{L_P} = 0.8976, \quad B = \frac{\pi(\pi - 2\alpha^*)}{L_H(\alpha^*)} = 1.1846.$$

We notice that hyperbolic paths always yield faster computation over parabolic ones. Moreover, the error e^C_h and e^L_h, in the exact integral (2), for constant (C) and linear (L) interpolations of u are linear and quadratic in h. So, there exists two constants C_C and C_L such that,

$$e^C_h = C_C h \quad \text{and} \quad e^L_h = C_L h^2.$$

In each couple of approximation, the quadrature and the interpolation errors can be balanced by equating the related errors e^X_M and e^Y_h. This yields the four relations between h and M, stated in Table 1, where $E(.)$ stands for the integer part, that allows to parameterize the numerical method by h only.

	e^C_h	e^L_h
e^P_M	$M = E(-\frac{1}{A}\log(\frac{C_C}{C_P}h))$	$M = E(-\frac{1}{A}\log(\frac{C_L}{C_P}h^2))$
e^H_M	$M = E(-\frac{1}{B}\log(\frac{C_C}{C_H}h)$	$M = E(-\frac{1}{B}\log(\frac{C_L}{C_H}h^2))$

Table 1. Relations between M and h

4. NUMERICAL RESULTS

In our presentation of numerical experiments, we discuss only causal parts. Similar results have been observed for anti-causal parts. We have considered the kernel $q(x, y) = 2(1-3x)(1-y)y^2 + 2(1-x)x^2(1-3y)$ and the input variable $u(x) = \sin(j\pi x)$ with $j \in \mathbb{N}^*$. The number of base functions of the Petrov-Galerkin method are fixed at sufficiently large values, $N_1 = N_2 = 15$, so that the error on μ the diffusive symbols is negligible in comparison with the other error sources. In Figure 1 we report the relative errors in logarithmic scale between z^+ an exact realization and its approximations z^{h+}, parameterized by h only,

$$e^{h+} = \frac{||z^+ - z^{h+}||_{L^2_h(\omega)}}{||z^+||_{L^2_h(\omega)}}$$

for $u(x) = \sin(\pi x)$. The error is measured in the discrete $L^2(\omega)$-norm, and the discretization step h ranges from 0.005 to 0.25. The errors decay rate is proportional to h for piecewise constant (C) interpolation, and proportional to h^2 for piecewise linear (L) interpolation. We notice that balancing the errors e^X_M and e^Y_h yields the same global error for both the parabolic (P) and the hyperbolic (H) path, this is the reason why both errors are plotted with a same curve.

To speed up the computation, we take into account the fact that for a real valued operator P, only half of the integrals need to be computed, i.e.

$$z^{\pm}(x) = 2\Re \int_{\mathbb{R}^+} \mu^{\pm}(x,\xi)\psi^{\pm}(x,\xi)\, d\xi,$$

which reduces, by a factor of 2, the number of quadrature points. Figure 2 provides a comparison of computation times between the direct (D) quadrature method,

$$Pu(x_n) \approx h \sum_j p(x_n, y_j)u(y_j) \quad \text{for all } n,$$

and the diffusive realization methods with hyperbolic (H) and parabolic (P) contours with piecewise constant interpolation of the input variable $u(x) = \sin(10\pi x)$ presenting ten oscillations. The gain in using diffusive realization over direct quadrature increases for finer spatial discretization points. The implementation is done in Matlab version 7.9, hence the timing-results have to be taken with caution.

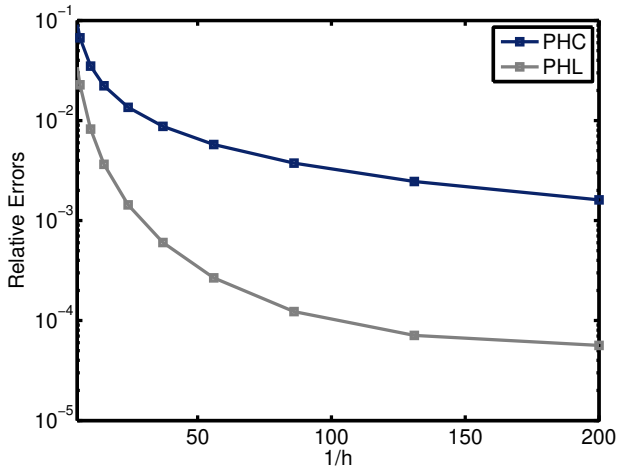

Fig. 1. Errors between z^+ and z^{h+}

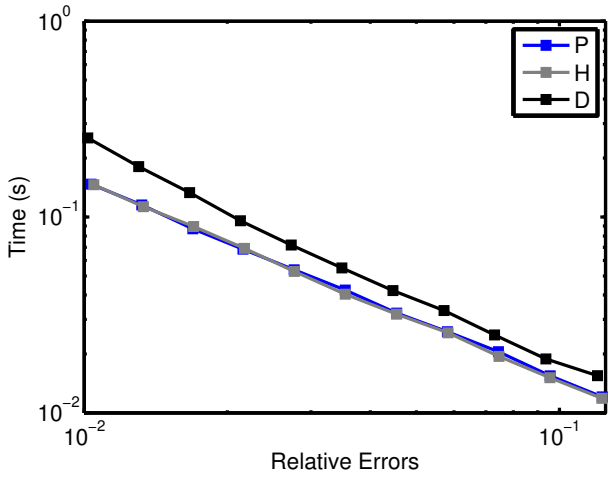

Fig. 2. Computation time in seconds versus relative error E^{h+}

5. FPGA IMPLEMENTATION

In this section we describe our implementation in a FPGA of the algorithms established for the *history functions* ψ^+, and for the diffusive realizations z^+ when the input $u(x) = \sin(\pi x)$ is interpolated with a piecewise constant functions For the sake of clarity, we synthetize them in Algorithm 1.

Algorithm 1 Diffusive Realization of $z^+(x)$

Offline Computation of diffusive symbol $\mu^{N+}(x,\xi)$
Online Computation
for $n = 0, ..., \mathcal{N}$ **do**
  for $k = 1, ..., M$ **do**
    $\psi^+_{n+1,k} = \alpha^+_k \psi^+_{n,k} + \beta^+_k u_n, \quad \psi^+_{0,k} = 0,$
  end for
  $z^+_{n+1} = 2h_\xi \Re\left(\sum_{k=1}^M \mu^{N+}_{n+1,k}\left(\alpha^+_k \psi^+_{n,k} + \gamma^+_k u_n\right)\right)$
end for

Note that the implementation of the anti-causal part is done in a similar way, and will not be described. Consequently, we will drop all upper indices "+" without any risk of confusion.

In order to process the highest number of operations in parallel, quantization of numbers must be optimized. To do so, we first require that all variables belong to a neighborhood of 1. This is achieved by scaling β, γ, μ, u, ψ, and z by a factor corresponding to an estimate of their larger value $\beta^* = \beta/\gamma_{\max}$, $\gamma^* = \gamma/\gamma_{\max}$, $\mu^* = \mu/\mu_{\max}$, $u^* = u/u_{\max}$, $\psi^* = \psi/(\beta_{\max}u_{\max})$, and $z^* = z/(u_{\max}\beta_{\max}\mu_{\max})$. We notice that α does not require any scaling, and that β has been scaled by $\gamma_{\max}$ since the latter is larger than $\beta_{\max}$. Then, Algorithm 1 has been rewritten based on the scaled variables, and the impact of quantization has been evaluated. In Table 2, we report absolute and relative errors, in the maximum norm, on the output z for quantizations varying between 11 bits to 16 bits.

Number of bits	Maximum error	Relative error
16	9.21e-5	1.05%
15	2.08e-4	2.51%
14	1.38e-4	1.51%
13	4.82e-4	6.79%
12	6.87e-4	9.23%
11	1.4e-3	18.69%

Table 2. Error with respect to the quantization

Regarding FPGA hardware implementations, three variants have been studied, namely a sequential, a parallel, plus a pipelined architecture. The parallel and the pipeline solution have been implemented. Before to discuss them, we underline some features of our Algorithm. We observe that each couple $(\pi_{n,k}, \nu_{n,k}) = (\alpha_k\psi_{n+1,k} + \beta_k u_n, \alpha_k\psi_{n+1,k} + \gamma_k u_n)$ of complex numbers can be computed independently, and that the same real number u_n is used for their evaluation. The dependency flow, related to the computation of the vector $(\psi_{n+1,k})_k$, is depicted in Figure 3.

We do not discuss further the sequential implementations since they are not exploiting the specific FPGA resources. In our parallel implementation, all $(\pi_{n,k}, \nu_{n,k})$ are computed independently using the same inputs $(u_n)_n$. For

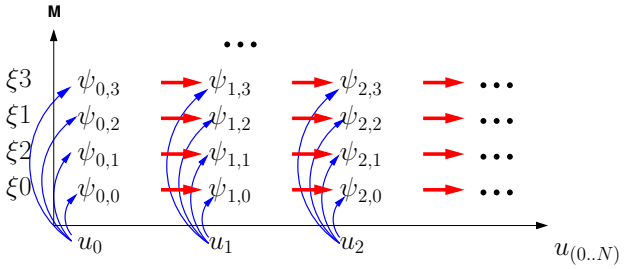

Fig. 3. Dependency flow for the computation of ψ

given u_n and $\psi_{n,k}$, eleven multiplications are required to determine a $\psi_{n+1,k}$. Due to the limited number of multipliers available in a FPGA, for large N and M a full parallel implementation is not always possible and the limited resources must be shared. To optimize resource sharing, a finite state machine was used. To design it, we deduce from the algorithm a number of states depending on the number of multipliers in a branch and on the number of required multiplications in a step. In our implementation of a state, the output of each multiplier is affected to a register before to be entered as an input of a subsequent addition or substraction, and the multiplier inputs are refreshed with new data. Finally, the state machine is designed to control all state evolution and the RAM, to load data in multipliers, and to store the results. Assuming infinite resources, the necessary time to build a complete vector $(z_n)_n$ is $N \times n_z \times t_{clock}$ where $n_z = 4$ is the number of time steps necessary to compute a new occurence z_n, and t_{clock} is the clock period.

Our implementation of a pipeline architecture takes advantage of the already underlined independence of the $(\pi_{n,k}, \nu_{n,k})$. For a given input u_n, the computation of a sequence of couples $(\pi_{n,k}, \nu_{n,k})_k$, and the updating of z_n are executed through the pipeline. Thus, in regular functioning (i.e. except initial and final periods), the computation of a couple takes one clock period. When the computation of $\psi_{n+1,k}$ is complete, u_{n+1} is taken as a new input parameter. Finally, the time required to build a realization $(z_n)_n$ is $N \times M \times t_{clock}$.

We have successfully implemented the parallel and the pipeline architectures in a Xilinx Spartan3A XILINX [2010] comprising 200 kgates, 16 multipliers with 18-bit input data and 36-bit output data, 16 RAM blocks of 16 kbits each, and a 100 MHz clock. For a vector of data $(u_n)_n$ with $N = 8$ components, for a quadrature formula with $M = 4$ nodes, and for 9-bit encoded integers, the parallel computation (with $n_z = 9$) and the pipeline computation (with $n_z = 4$) of a vector $(z_n)_n$ takes $0.72\mu s$ and $0.32\mu s$ respectively, which fit with the theoritical values. The same computation with a C program on a laptop computer with a x86 1.6 GHz processor takes about $60\mu s$. This yields a speed-up of about 10^2.

6. CONCLUSION

Until now, the diffusive realization of operators has been applied to operators with analytically known kernels. From the references in the field, it is known to be a very efficient method requiring little computation for real time realizations since small M (compared to N) are generally enough to yield good approximations. In Lenczner et al. [2010] we have introduced a mathematical framework allowing for its derivation when an operator is a solution to a linear operator partial differential equation. A complete justification of the numerical method was not included, so it constitutes the main focus of the present paper. Here, our general approach is presented through the example of a Lyapunov equation arising in optimal control theory of the one-dimensional heat equation. In view of real-time applications, we have also implemented this method in a FPGA with a parallel and with a pipeline architecture. The theoretical (optimal) computation time for the pipeline implementation is found to be $N * M * t_{cloc}$ and is confirmed by our experiment. Further extensions of this method are now in development: we study how to encompass Riccati equations coming from more general partial differential equations in higher dimensional domains.

ACKNOWLEDGEMENTS

This work is partially supported by the European Territorial Cooperation Programme INTERREG IV A France-Switzerland 2007-2013.

REFERENCES

C. Casenave. *Représentation diffusive et inversion opératorielle pour l'analyse et la résolution de problèmes dynamiques non locaux*. PhD thesis, Université Toulouse III - Paul Sabatier, 2009.

T. Hélie, D. Matignon, and R. Mignot. Criterion design for optimizing low-cost approximations of infinite-dimensional systems: towards efficient real-time simulation. *Int. J. Tomogr. Stat.*, 7(F07):13–18, 2007.

L. Laudebat, P. Bidan, and G. Montseny. Modeling and optimal identification of pseudodifferential electrical dynamics by means of diffusive representation - Part 1: Modeling. *IEEE Transactions on Circuits and Systems I-Regular Papers*, 51(9):1801–1813, 2004.

M. Lenczner and G. Montseny. Diffusive realization of operator solutions of certain operational partial differential equations. *C. R. Math. Acad. Sci. Paris*, 341(12): 737–740, 2005.

M. Lenczner, G. Montseny, and Y. Yakoubi. Diffusive realizations for solutions of some operator equations. *Accepted in Math. of Comp.*, 2010.

M. López-Fernández, C. Lubich, C. Palencia, and A. Schädle. Fast Runge-Kutta approximation of inhomogeneous parabolic equations. *Numer. Math.*, 102(2):277–291, 2005.

G. Montseny. *Représentation diffusive*. Hermès-Sciences, 2005.

J. A. C. Weideman and L. N. Trefethen. Parabolic and hyperbolic contours for computing the Bromwich integral. *Math. Comp.*, 76(259):1341–1356, 2007.

XILINX. Spartan-3a fpga family: Data sheet, 2010. `http://www.xilinx.com/support/documentation/data_sheets/ds529.pdf`.

Y. Yakoubi. *Deux Méthodes d'Approximation pour un Contrôle Optimal Semi-Décentralisé pour des Systèmes Distribués*. PhD thesis, Université de Franche-Comté, 2010.