Proceedings of the Federated Conference on Computer Science and Information Systems pp. 641–647

DOI: 10.15439/2015F286
ACSIS, Vol. 5

A K-iterated scheme for the first-order Gaussian recursive filter with boundary conditions

Salvatore Cuomo
University of Naples Federico II
Department of Mathematics and Applications ”R. Caccioppoli”, Italy
Email: salvatore.cuomo@unina.itl

Ardelio Galletti
University of Naples ”Parthenope”
Department of Science and Technology, Italy
Email: ardelio.galletti@uniparthenope.it

Raffaele Farina
Institute for high performance computing and networking
CNR, Italy
Email: raffaele.farina@na.icar.cnr.it

Livia Marcellino
University of Naples ”Parthenope”
Department of Science and Technology, Italy
Email: livia.marcellino@uniparthenope.it

***Abstract*—Recursive Filters (RFs) are a well known way to approximate the Gaussian convolution and are intensively used in several research fields. When applied to signals with support in a finite domain, RFs can generate distortions and artifacts, mostly localized at the boundaries of the computed solution. To deal with this issue, heuristic and theoretical end conditions have been proposed in literature. However, these end conditions strategies do not consider the case in which a Gaussian RF is applied more than once, as often happens in several realistic applications. In this paper, we suggest a way to use the end conditions for such a K-iterated Gaussian RF and propose an algorithm that implements the described approach. Tests and numerical experiments show the benefit of the proposed scheme.**

I. Introduction

Recursive filters (RFs) have achieved a central role in several research fields, as in data assimilation for operational three-dimensional variational analysis schemes (3Dvar) [4], [10], and in Electrocardiogram (ECG) denoising [1], [2]. Among RFs, the Gaussian RFs are particularly suitable for digital image processing [13] and applications of the scale-space theory [8], [15]. Gaussian RFs are an efficient computational tool for approximating Gaussian-based convolutions [3], [14], [10], [11], [12]. Gaussian RFs are mainly derived in three different ways: the Deriche strategy uses an approximation of the Gaussian function in the space domain [5]; the approximation procedure of Jin et al. is carried out in the z-domain, i.e. it is based on an approximation of the z-transform of the Gaussian function by means of rational polynomial functions [9]; and the approach followed by Vliet et al. [12], [16] approximates the Gaussian function in the Fourier domain. The latter approach is particularly attractive since the Fourier transform of a Gaussian function is a Gaussian.

From a mathematical point of view, a Gaussian RF consists of two infinite sequences of equations (forward and backward equations) that involve the entries of both the input and output signals. However, algorithms that implement RFs need to consider only a finite number of these equations, i.e. they take into account a finite number of input and output signal samples. Without additional assumptions, such a reduction (from the infinite to the finite) introduces distortions and artifacts on the computed finite output signals. In particular, these solutions present an error that affects the entries near to the boundaries. This phenomenon has been recognized as an edge effect in [3] and some authors, such as Purser et al. [10] and Triggs et al. [14], suggest how to avoid this by simulating the effect of the continuation of the neglected equations. This results in inserting the so-called boundary conditions, or end conditions, i.e. in modifying some of the filter equations so that the backward equations can be primed. These strategies are based on some heuristic assumptions on the input signal and seem to fix the edge effect problem.

In some real applications [3], [6], [7], Gaussian RFs are used iteratively. This relies on a more general definition of RFs, named K-iterated RFs, which have been formally introduced in [3]. So far, a comprehensive study of the conditions for the K-iterated RFs is still lacking. Moreover, as noticed in [3], and recalled in this work (see Section 3), edge effects reappear when the number K of filter iterations increases, even though the classic RF end conditions are used. In this context, the purpose of this work is to provide an algorithm which combines classic end conditions with a suitable oversizing and reduction of both the input and the output signals, so that the edge effects are strongly mitigated. The paper is organized as follows. In Section 2, we recall the definitions of the discrete Gaussian convolution and Gaussian recursive filters, then we derive classic end conditions in a general way and show their specific features for the first-order Gaussian RF used in [3], [6], [7]. The benefit of the end conditions is shown through some numerical examples. In Section 3, we give the definition of the K-iterated RFs and point out how the filter coefficients of a K-iterated first-order RF have to be taken. Finally, we propose a numerical scheme that implements a K-iterated first-order RF with end conditions and a strategy for preventing the occurrence of edge effects. In Section 4, we report some experiments to confirm the reliability of the proposed algorithm and give suggestions on how to set the parameters of the proposed K-iterated scheme. Finally, Section 5 contains some concluding remarks.

II. Mathematical Background

Let

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$$

be the Gaussian function and

$$s^{(0)} = \{s_j^{(0)}\}_{j=-\infty}^{+\infty} = \left(\ldots, s_{-2}^{(0)}, s_{-1}^{(0)}, s_0^{(0)}, s_1^{(0)}, s_2^{(0)}, \ldots\right)$$

a signal (i.e. a complex function defined on the set of integers **Z**). The discrete Gaussian convolution of $s^{(0)}$ gives rise to an output signal $s^{(g)}$ whose entries are defined by:

$$s_j^{(g)} \overset{\text{def}}{=} \left(g * s^{(0)}\right)_j = \sum_{t=-\infty}^{+\infty} g(j-t) s_t^{(0)}. \tag{1}$$

The entries $s_j^{(g)}$ of $s^{(g)}$ in (1) can be approximated by means of Gaussian recursive filters. The n-order Gaussian RF filter gives an output signal s, which is an approximation of $s^{(g)}$, whose entries solve the infinite sequences of equations:

$$p_j = \beta_j s_j^{(0)} + \sum_{t=1}^{n} \alpha_{j,t} p_{j-t}, \qquad j = -\infty, \ldots, +\infty, \tag{2}$$

$$s_j = \beta_j p_j + \sum_{t=1}^{n} \alpha_{j,t} s_{j+t}, \qquad j = -\infty, \ldots, +\infty. \tag{3}$$

The equations in (2) and (3) are conveniently referred to as the advancing and backing filters, respectively, since in the former the index j must be treated as in increasing order while, in the latter, it must be treated in decreasing order [10]. The values $\alpha_{j,t}$ and β_j are called *smoothing coefficients* and satisfy:

$$\beta_j = 1 - \sum_{t=1}^{n} \alpha_{j,t}.$$

The smoothing coefficients depend on σ, n and, in a more general setting, even on the index j. Hereafter, we only consider the homogeneous case, i.e. we set:

$$\beta_j \equiv \beta, \qquad \alpha_{j,t} \equiv \alpha_t, \tag{4}$$

where the advancing and backing filters take the form:

$$p_j = \beta s_j^{(0)} + \sum_{t=1}^{n} \alpha_t p_{j-t}, \qquad j = -\infty, \ldots, +\infty, \tag{5}$$

$$s_j = \beta p_j + \sum_{t=1}^{n} \alpha_t s_{j+t}, \qquad j = -\infty, \ldots, +\infty. \tag{6}$$

If we assume that the support of the input signal $s^{(0)}$ is in the grid $\{1, 2, \ldots, N\}$, then equations (5) and (6) can be implemented in an algorithm in which the index j increases from 1 to N, for the advancing filter, and decreases from N to 1, for the backing filter. This scheme needs to prime the advancing filter, by setting the values p_j, for $j \in \{0, -1, \ldots, 1-n\}$, and the backing filter, by setting the values s_j for $j \in \{N+1, N+2, \ldots, N+n\}$. For example, a common choice is to set at zero the required p_j and s_j values, i.e.:

$$\begin{aligned} p_{-n+1} = p_{-n+2} = \ldots = p_0 = 0; \\ s_{N+1} = s_{N+2} = \ldots = s_{N+n} = 0. \end{aligned} \tag{7}$$

However, this assumption gives rise to a well-known edge effect, already noticed in [14], and discussed in detail in [3]. An outline of such a scheme, implementing (5), (6) and (7), is provided in **Algorithm 1**.

Algorithm 1 Scheme of an n-order Recursive Filter with zero end constraints

```
Input: $s^{(0)}$, $\sigma$      Output: $s$
 1: set $\beta, \alpha_1, \ldots, \alpha_n$      % smoothing coefficient precomputation
 2: for $j = 1, 2, \ldots, n$      % left zero end conditions
 3:     $p_{j-n} := 0$
 4: endfor
 5: for $j = 1, 2, \ldots, N$      % advancing filter
 6:     $p_j := \beta s_j^{(0)}$
 7:     for $t = 1, 2, \ldots, n$
 8:         $p_j := p_j + \alpha_t p_{j-t}$
 9:     endfor
10: endfor
11: for $j = 1, 2, \ldots, n$      % right zero end conditions
12:     $s_{N+j} := 0$
13: endfor
14: for $j = N, \ldots, 1$      % backing filter
15:     $s_j := \beta p_j$
16:     for $t = 1, 2, \ldots, n$
17:         $s_j := s_j + \alpha_t s_{j+t}$
18:     endfor
19: endfor
```

Example 1. Now we provide some insights into the edge effect through the following example. Let $s^{(0)}$ be the input signal with entries:

$$s_j^{(0)} = \begin{cases} 0 & \text{for } j \le 0; \\ 1 & \text{for } 1 \le j \le N = 30; \\ 0 & \text{for } j \ge N+1 = 31; \end{cases} \tag{8}$$

In Figure 1, we report the results obtained by applying the first-order RF and third-order RF to the signal $s^{(0)}$. In particular, the red line represents the signal $s^{(g)}$, i.e. the discrete Gaussian convolution of $s^{(0)}$ with $\sigma = 4$. Conversely, the white squares and gray diamonds are the values of the output signals computed by means of the first-order RF and third-order RF, respectively. Notice that both computed solutions differ from $s^{(g)}$ mostly on the boundary entries. In particular, the third-order filter seems to not well approximate the entries of $s^{(g)}$ close to the right boundary. This phenomenon can be better

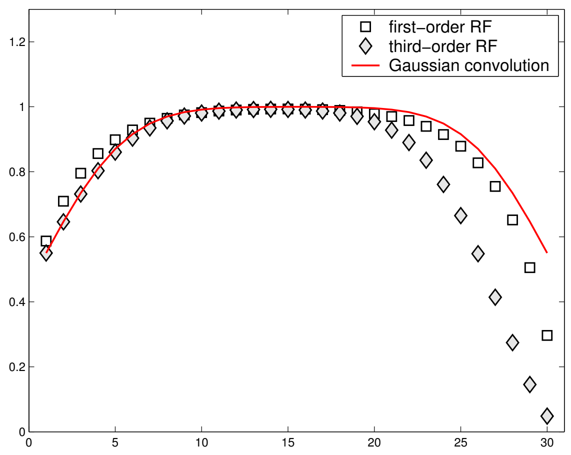

Fig. 1. Red line: discrete Gaussian. White squares: first-order RF solution. Gray diamonds: third-order RF solution.

understood by looking at the relative errors

$$e_j = \frac{|s_j - s_j^{(g)}|}{|s_j^{(g)}|}. \tag{9}$$

The value of the errors e_j, for the first-order RF and third-order RF, are shown in Figure 2. Observing the order of magnitude of the values e_j, we can conclude that, in general, the output signals present a larger error at the boundaries.

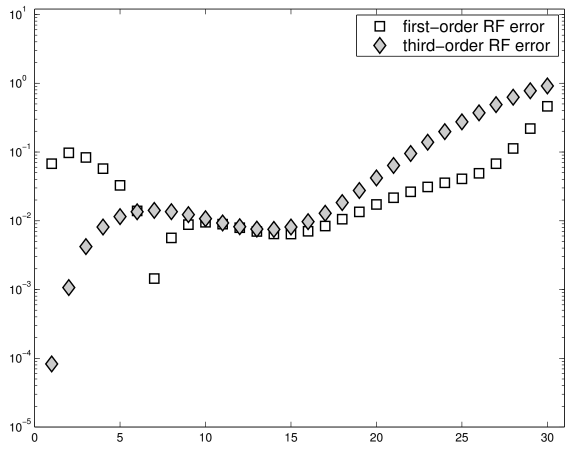

Fig. 2. White squares: first-order RF relative errors. Gray diamonds: third-order RF relative errors.

A. *Introducing end conditions*

The edge effect, shown in the previous example, can be partly explained by observing that, in the transition from the infinite sequences of equations to the finite algorithm, the computed solution s has been constrained to assume the value zero on the right off-grid points $N+1, N+2, \ldots, N+n$, while this property is not true for $s^{(g)}$. Hence, assumption (7) introduces a sort of perturbation error on the output signal s of the RF. This drawback can be avoided by simulating, in the finite setting, the effect of the continuation of the neglected equations (for $j > N$). To achieve this aim, we adapt to our setting and notations the derivation of the end conditions (e.c.) described in [14].

We consider an n-order recursive filter with smoothing coefficients $\alpha_1, \ldots, \alpha_n$ and β_j. Let C and A be the following $n \times n$ matrices:

$$\mathbf{A} = \begin{pmatrix} \alpha_1 & \ldots & \alpha_{n-1} & \alpha_n \\ 1 & \ldots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \ldots & 1 & 0 \end{pmatrix} \tag{10}$$

$$\mathbf{C} = \begin{pmatrix} \beta & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 0 & 0 \end{pmatrix} \tag{11}$$

Now, by setting:

$$\begin{aligned} \vec{p}_t &= \left(p_t, p_{t-1}, \ldots, p_{t-n+1}\right)^T, \\ \vec{s}_t &= \left(s_t, s_{t+1}, \ldots, s_{t+n-1}\right)^T, \\ \vec{s}_t^{(0)} &= \left(s_t^{(0)}, s_{t+1}^{(0)}, \ldots, s_{t+n-1}^{(0)}\right)^T, \end{aligned}$$

the forward and backward filters in (5) and (6), take the form:

$$\vec{p}_{t+1} = C\vec{s}_{t+1}^{(0)} + A\vec{p}_t, \tag{12}$$

$$\vec{s}_t = C\vec{p}_t + A\vec{s}_{t+1}. \tag{13}$$

Then, applying recursively (12) and (13), we obtain:

$$\vec{p}_{t+k} = \sum_{l=0}^{k-1} A^l C\vec{s}_{t+k-l}^{(0)} + A^k\vec{p}_t, \tag{14}$$

$$\vec{s}_t = \sum_{l=0}^{k-1} A^l C\vec{p}_{t+l} + A^k\vec{s}_{t+k}. \tag{15}$$

Assuming that the support of the input signal $s^{(0)}$ is in $\{1, 2, \ldots, N\}$, the equation (14), for $k > 0$ and $t \geq N$, becomes:

$$\vec{p}_{t+k} = A^k\vec{p}_t. \tag{16}$$

Now, using (16) in (15), and taking the limit for $l \to \infty$, provided that s is bounded, we obtain:

$$\vec{s}_t = M\vec{p}_t, \qquad \text{with} \qquad M \overset{\text{def}}{=} \sum_{l=0}^{\infty} A^l C A^l. \tag{17}$$

For $t = N$, equation (17) provides a way of priming the backing filter. In other words, (17) behaves as a sort of *turning* condition that accounts for the infinite sequence of neglected equations and allows us to skip from the advancing filter to the backing filter. Indeed, the values $s_N, s_{N+1}, \ldots, s_{N+n-1}$ are linear functions of the values $p_N, p_{N-1}, \ldots, p_{N-n+1}$ which are computed by the advancing filter. In this approach, we need to precompute the matrix M. This can be done in several ways: in [14], it is observed that the recursive definition of M implies that it solves the equation:

$$M = C + AMA;$$

otherwise, one could compute an approximation of M as the partial sum $M_k = \sum_{l=0}^{k} A^l C A^l$, provided that $A^{k+1}CA^{k+1}$ is negligible. Observe that, for the first-order RF with smoothing coefficients $\alpha \equiv \alpha_1$ and $\beta = 1 - \alpha$, we simply have:

$$A = \alpha, \quad C = \beta, \quad M = \sum_{l=0}^{\infty} \beta \cdot \alpha^{2l} = \frac{\beta}{1-\alpha^2} = \frac{1}{1+\alpha}$$

and for $t = N$ in (17),

$$s_N = \frac{1}{1+\alpha} p_N.$$

An exact expression of the matrix M, in terms of the coefficients α_1, α_2, α_3 and β, can be derived for the third-order RF [14] and will be used in the example below. In order to take into account the end conditions, steps 11. 12. and 13. of **Algorithm 1** must be replaced with the computation of M and the computation of $\vec{s}_N$. Moreover, the last loop (steps 14.-19.) must start from $N - 1$.

Example 2. To see the effect of the e.c. on the RFs, we consider the input signal $s^{(0)}$ in (8). The results obtained by applying the first-order and the third-order RFs, with and without e.c., are plotted in Figure 3 and Figure 4, respectively.

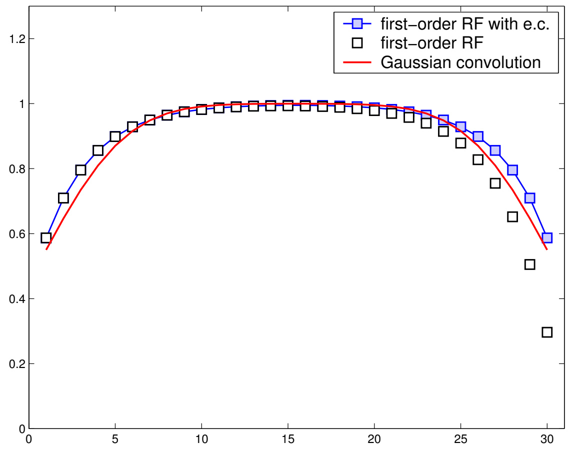

Fig. 3. Red line: discrete Gaussian. White squares: first-order RF without end conditions. Blue squares: first-order RF with end conditions.

In Figure 3 the solutions of the first-order RFs with and without e.c. are compared; analogously, Figure 4 deals with the third-order RFs. In both cases, one can notice the improvement of the computed solutions, especially on their right boundary entries, when the e.c. are considered. In particular, the output signal of the third-order RF with e.c. is very close to the Gaussian convolution output $s^{(g)}$.

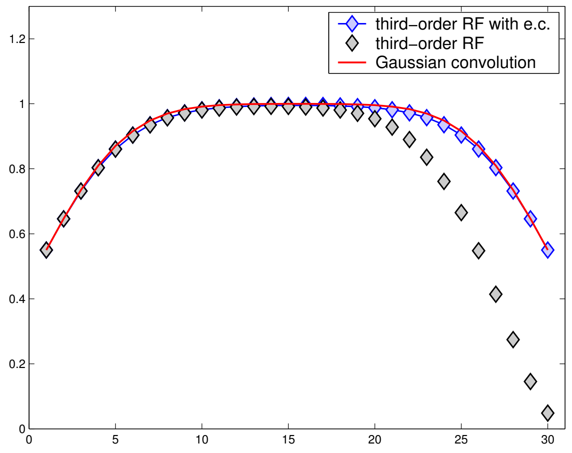

Fig. 4. Red line: discrete Gaussian. Gray diamonds: third-order RF without end conditions. Blue diamonds: third-order RF with end conditions.

III. K-ITERATED GAUSSIAN RECURSIVE FILTERS

The latter example highlights that the third-order filter mimics very well the effect of the discrete Gaussian convolution. Conversely, the output signal of the first-order RF, even with e.c., is still a poor approximation of $s^{(g)}$. Purser et al. [10] suggest that applying several times the same RF can improve the accuracy of the approximation. This fact could be seen as a consequence of the central limit theorem applied to the Gaussian convolution. This idea has been discussed and implemented in [3], [6] and [7] where the authors iteratively used the first-order RF (without e.c.) to solve a three-dimensional variational data assimilation problem. Here, we are interested in formalizing such an iterative approach.

A K-iterated n-order Gaussian RF filter computes the output signal $s^{(K)}$, i.e. the K-iterated approximation of $s^{(g)}$, as follows:

$$p_j^{(k)} = \beta s_j^{(k-1)} + \sum_{t=1}^{n} \alpha_t p_{j-t}^{(k)}, \qquad j = -\infty, \dots, +\infty, \quad (18)$$

$$s_j^{(k)} = \beta p_j^{(k)} + \sum_{t=1}^{n} \alpha_t s_{j+t}^{(k)}, \qquad j = -\infty, \dots, +\infty. \quad (19)$$

The filter iteration counter k goes from 1 to K, where K is the total number of filter iterations. For $K > 1$, the problem of triggering the advancing filter at iteration $k = 2, \dots, K$ can be faced with the same strategy used to generate the (right) end conditions for a classic RF ($K = 1$). Assuming indeed that, at the iteration k:

$$\begin{aligned} p_{-n+1}^{(k)} &= p_{-n+2}^{(k)} = \dots = p_0^{(k)} = 0, \\ s_{N+1}^{(k)} &= s_{N+2}^{(k)} = \dots = s_{N+n}^{(k)} = 0, \end{aligned} \quad (20)$$

and rearranging the equations (12) and (13), it results:

$$\vec{p}_1^{(k)} = M \vec{s}_1^{(k-1)}, \quad (21)$$

with M as in (17). Similarly, the backing filter, for iterations $k = 2, \dots, K$, can be triggered by setting

$$\vec{s}_N^{(k)} = M \vec{p}_N^{(k)}. \quad (22)$$

The smoothing coefficients depend on σ, n and even on K. The correct setting of the smoothing coefficients is crucial for the convergence. We remark that, in the Fourier domain, the Gaussian convolution in (1) becomes

$$S^{(g)}(\omega) = G(\omega, \sigma) \cdot S^{(0)}(\omega), \tag{23}$$

where

$$S^{(g)}(\omega) = \Big(\mathcal{F}\big(s^{(g)}\big)\Big)(\omega), \qquad S^{(0)}(\omega) = \Big(\mathcal{F}\big(s^{(0)}\big)\Big)(\omega)$$

and

$$G(\omega, \sigma) = \big(\mathcal{F}(g)\big)(\omega) = \exp\left(-\frac{\omega^2\sigma^2}{2}\right)$$

are the Fourier Transforms of the signals $s^{(g)}$, $s^{(0)}$ and g, respectively. Rewriting (23) as:

$$S^{(g)}(\omega) = \left(G\left(\omega, \frac{\sigma}{\sqrt{K}}\right)\right)^K \cdot S^{(0)}(\omega), \tag{24}$$

we obtain, in the signal domain, that $s^{(g)}$ can be seen as the result of K successive Gaussian convolutions with an identical Gaussian function with a standard deviation:

$$\sigma_K = \frac{\sigma}{\sqrt{K}}.$$

This argument suggests that σ_K must replace σ in the expression of the smoothing coefficients of a K-iterated RF. For example, the smoothing coefficients α and β of the (one-iterated) first-order Gaussian RF, in the homogeneous case, are set as:

$$\alpha = 1 + E_\sigma - \sqrt{E_\sigma(E_\sigma + 2)}, \qquad \beta = \sqrt{E_\sigma(E_\sigma + 2)} - E_\sigma. \tag{25}$$

with

$$E_\sigma = \frac{1}{\sigma^2}.$$

Then, for the K-iterated first-order Gaussian RF, the value of E_σ must be replaced by:

$$E_{\sigma_K} = \frac{1}{\sigma_K^2} = \frac{K}{\sigma^2}.$$

Example 3. To see the behaviour of such a K-iterated filter, we apply it to a random input signal with support in $\{1, 2, \ldots, 30\}$. In Figure 5, the RF output signals for three different values of the number iterations ($K = 12, 25, 50$) are reported. The results show that, despite the fact that the accuracy on the central entries does not seem to change, the edge effects reappear when the number of filter iterations increases. In particular (see Figure 6), the relative errors e_j, defined as in (9), increase both in the left and right boundary entries as K increases. Conversely, the convergence to the Gaussian convolution output signal may be observed by looking at the central entries of the errors in Figure 6. We highlight that, in the previous test, the end conditions (21) and (22) have been used. The fact that the e.c. do not work as expected, mostly at the edges, can be simply explained as a consequence of the wrong assumptions in (20).

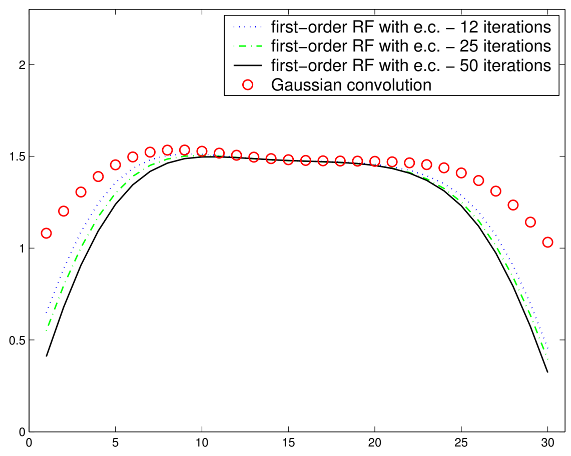

Fig. 5. Discrete Gaussian (red circles) versus first-order RF solutions with 12 iterations (dotted blue line), 25 iterations (dashdot green line) and 50 iterations (solid black line).

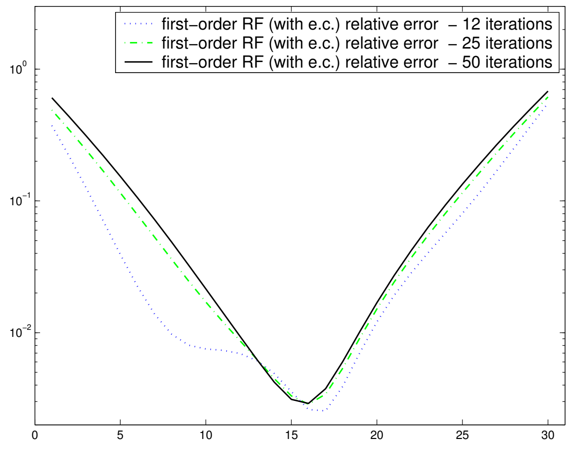

Fig. 6. First-order RF relative errors for 12 iterations (dotted blue line), 25 iterations (dashdot green line) and 50 iterations (solid black line).

Here, we suggest a scheme that allows the use of the K-iterated first-order Gaussian RF and prevents the edge effect. Our idea consists in three steps:

(i) *extending* the given input signal $s^{(0)}$, with support in $\{1, 2, \ldots, N\}$, by adding artificial zero entries at the left and right boundaries. More specifically, we introduce the extended signal:

$$s^{(0),m} = \big(0, \ldots, 0, s_1^{(0)}, \ldots, s_N^{(0)}, 0, \ldots, 0\big), \tag{26}$$

which is obtained by placing m zeros before $s_1^{(0)}$ and m zeros after $s_N^{(0)}$;

(ii) applying the K-iterated first-order Gaussian RF to $s^{(0),m}$;

(iii) reducing the output signal $s^{(K),m}$, by removing its first and last m entries.

The underlying idea of that scheme is to shift the edge effects on the artificially added entries. Steps (i)-(iii) are summarized in the following algorithm. For the sake of simplicity, we consider the case of the first-order RF. Nevertheless, the same algorithm could be easily modified for any K-iterated n-order RF.

Algorithm 2 Scheme of the K-iterated first-order recursive filter with end conditions

```
Input: $s^{(0)}, \sigma, m, K$     Output: $s^{(K)}$
 1: extend $s^{(0)}$ to $s^{(0),m}$
 2: set $s^{(0)} := s^{(0),m}$; $\beta, \alpha$ as in (25); $M := 1/(1+\alpha)$
 3: for $k = 1, 2, \ldots, K$                          % filter loop
 4:     compute $p_1^{(k)} := M s_1^{(k-1)}$              % left end conditions
 5:     if $k = 1$ then
 6:         $p_1^{(k)} := \beta s_1^{(k-1)}$
 7:     end
 8:     for $j = 2, \ldots, N$                         % advancing filter
 9:         $p_j^{(k)} := \beta s_j^{(k-1)} + \alpha p_{j-1}^{(k)}$
10:     endfor
11:     compute $s_N^{(k)} := M p_N^{(k)}$               % right end conditions
12:     for $j = N-1, \ldots, 1$                       % backing filter
13:         $s_j^{(k)} := \beta p_j^{(k)} + \alpha s_{j+1}^{(k)}$
14:     endfor
15: endfor
16: reduce $s^{(K)}$ as described in step (iii)
```

In the next section, through some numerical examples, it will be pointed out how the parameter m has to be set, in order to obtain a satisfactory accuracy on the computed RF output signals, even at their boundary entries.

IV. Numerical Experiments

In this section, we present two experiments. The aim of the first test is to prove the effectiveness of **Algorithm 2** in removing the edge effects. In the second test, we show that a suitable value of m can guarantee a negligible edge error, regardless of the filter number iterations K and the size N of the input signal.

A. Convergence at the edges

We consider the random input signal of **Example 3.**. We use **Algorithm 2** varying both the number of iterations and the size of $s^{(0),m}$. More precisely, in Figure 7, we report the output signals obtained with $K = 12$ iterations and by extending the input signal $s^{(0)}$ with $m = 1, 3, 6$ zeros at each boundary. Figure 8 shows the results obtained with the same values of m and increasing the number of iteration to $K = 25$.

Looking at Figure 7 and Figure 8, we observe that the edge effects seem to disappear as m increases. Notice that, comparing the behaviour of the blue curves ($m = 1$), the results worsen as K increases. This drawback can be avoided by suitably increasing the value of m. Figure 9 shows pictorially that, taking $m = 3\sigma = 12$, the accuracy is preserved also at the boundaries. Moreover, Figure 10 indicates that the component-wise convergence holds as K increases.

B. Suitable choice of m

In this test we measure the error $\|s^{(g)} - s^{(K)}\|_2$, where $s^{(K)}$ is the output signal obtained by applying the **Algorithm 2** to the random input signal $s^{(0)}$ of **Example 3.** varying the values of m and K. The results summarized in Table I, for $\sigma = 4$, show that the convergence is reached, as K increases, provided that $m \geq 2\sigma$. For $m \leq \sigma$ the convergence is not achieved due to the edge effects. We also note that increasing m beyond 3σ does not improve significantly the accuracy.

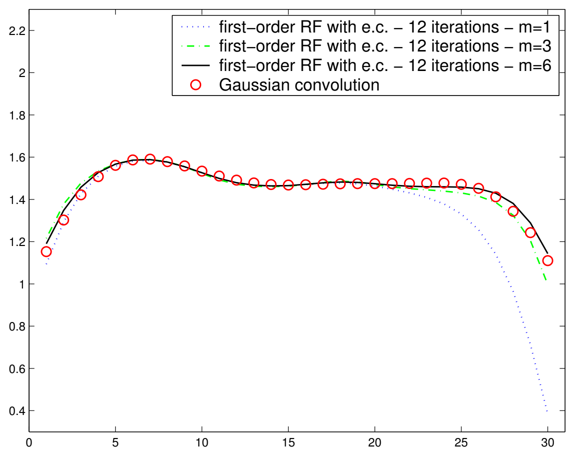

Fig. 7. Discrete Gaussian (red circles) versus first-order RF solutions with 12 iterations and $m = 1$ (dotted blue line), $m = 3$ (dashdot green line) and $m = 6$ (solid black line).

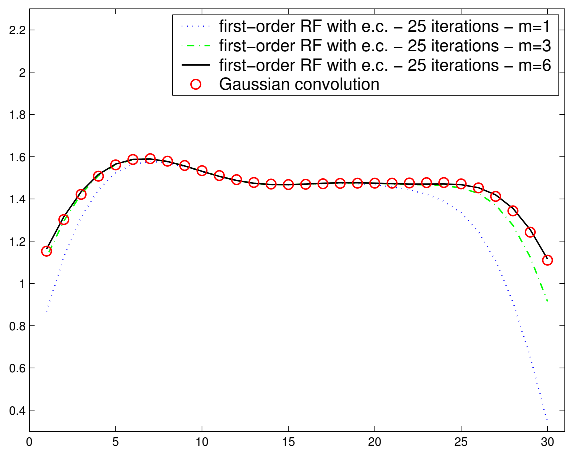

Fig. 8. Discrete Gaussian (red circles) versus first-order RF solutions with 25 iterations and $m = 1$ (dotted blue line), $m = 3$ (dashdot green line) and $m = 6$ (solid black line).

TABLE I
NORMS $\|s^{(g)} - s^{(K)}\|_2$ FOR SEVERAL VALUES OF K AND m. $\sigma = 4, N = 30$.

$K \backslash m$	0.25σ	0.5σ	σ	2σ	3σ	6σ
5	9.85e-01	4.70e-01	1.00e-01	1.23e-01	1.42e-01	1.16e-01
15	1.14e+00	5.66e-01	7.75e-02	4.96e-02	6.00e-02	5.73e-02
30	1.19e+00	6.07e-01	8.14e-02	3.41e-02	3.48e-02	3.00e-02
50	1.23e+00	6.18e-01	8.61e-02	2.48e-02	2.49e-02	2.54e-02
100	1.26e+00	6.40e-01	9.09e-02	1.71e-02	1.70e-02	1.70e-02

Table II shows that the above results about convergence and accuracy hold true also for different values of σ. We remark

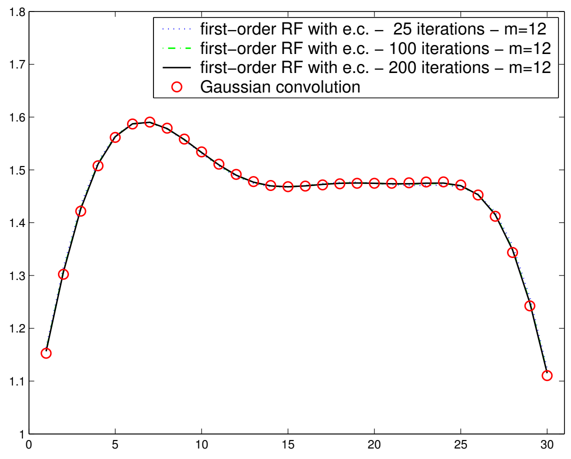

Fig. 9. Discrete Gaussian (red circles) versus first-order RF solutions with 25 iterations (dotted blue line), 100 iterations (dashdot green line) and 200 iterations (solid black line). m is set to 12 and the computed solutions can not be distinguished.

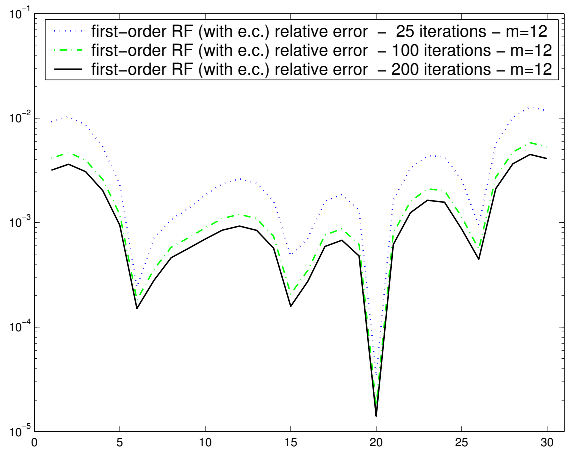

Fig. 10. First-order RF relative errors for 25 iterations (dotted blue line), 100 iterations (dashdot green line) and 200 iterations (solid black line).

that, for any fixed value of K, the value $m = 3\sigma$ represents a good trade-off between accuracy and efficiency.

TABLE II
NORMS $\|s^{(g)} - s^{(K)}\|_2$ FOR SEVERAL VALUES OF K AND m. $\sigma = 6, N = 30$.

$K \backslash m$	0.25σ	0.5σ	σ	2σ	3σ	6σ
5	1.02e+00	4.88e-01	1.01e-01	1.13e-001	1.61e-001	1.18e-01
15	1.13e+00	5.63e-01	7.71e-02	4.57e-002	4.98e-002	5.43e-02
30	1.20e+00	6.02e-01	8.31e-02	3.24e-002	3.65e-002	3.48e-02
50	1.23e+00	6.21e-01	8.62e-02	2.31e-002	2.41e-002	2.30e-02
100	1.26e+00	6.42e-01	9.08e-02	1.64e-002	1.74e-002	1.65e-02

The results in Table III prove that the above conclusions do not depend on the size of the input signal.

V. CONCLUSIONS

In this paper, we have discussed a way to overcome the edge effect in K-iterated Gaussian RFs, by a suitable choice of the end conditions. We have introduced an algorithm that implements the described approach. We have shown by means of several numerical experiments the effectiveness of the proposed K-iterated scheme. Finally, we have discussed in detail several issues related to the suitable choice of the parameter of our method.

TABLE III
NORMS $\|s^{(g)} - s^{(K)}\|_2$ FOR SEVERAL VALUES OF K AND m. $\sigma = 4, N = 2000$.

$K \backslash m$	0.25σ	0.5σ	σ	2σ	3σ	6σ
5	1.13e+00	5.84e-01	3.07e-01	4.68e-01	2.87e-01	3.47e-01
15	1.20e+00	6.00e-01	1.39e-01	1.33e-01	1.25e-01	1.32e-01
30	1.25e+00	6.25e-01	1.17e-01	8.23e-02	9.08e-02	9.11e-02
50	1.29e+00	6.44e-01	1.03e-01	6.27e-02	5.90e-02	6.48e-02
100	1.32e+00	6.67e-01	1.01e-01	4.28e-02	4.29e-02	4.28e-02

REFERENCES

[1] S. Cuomo, G. De Pietro, R. Farina, A. Galletti, and G. Sannino - *Novel O(n) Numerical Scheme for ECG Signal Denoising*. Procedia Computer Science , Volume 51, pp. 775784, doi: 10.1016/j.procs.2015.05.198, 2015.

[2] S. Cuomo, G. De Pietro, R. Farina, A. Galletti, and G. Sannino - *A framework for ECG denoising for mobile devices*. PETRA 15 ACM. ISBN 978-1-4503-3452-5/15/07, doi: 10.1145/2769493.2769560, 2015.

[3] S. Cuomo, R. Farina, A. Galletti, L. Marcellino -*An error estimate of Gaussian Recursive Filter in 3Dvar problem*, Federated Conference on Computer Science and Information Systems, FedCSIS 2014, doi: 10.15439/2014F279, pp. 587-595, 2014.

[4] L. D'Amore, R. Arcucci, L. Marcellino, A. Murli- *HPC computation issues of the incremental 3D variational data assimilation scheme in OceanVarsoftware*. Journal of Numerical Analysis, Industrial and Applied Mathematics, 7(3-4), pp. 91-105, 2013.

[5] R. Deriche - *Recursively implementing the Gaussian and its derivatives*. INRIA Research Report RR-1893, 1993.

[6] S. Dobricic, N. Pinardi - *An oceanographic three-dimensional variational data assimilation scheme*. Ocean Modeling 22, pp. 89-105, doi: 10.1016/j.ocemod.2008.01.004, 2008.

[7] Farina, R., Dobricic, S., Storto, A., Masina, S., Cuomo, S. -*A revised scheme to compute horizontal covariances in an oceanographic 3D-VAR assimilation system*. Journal of Computational Physics, 284, pp. 631-647, doi: 10.1016/j.jcp.2015.01.003, 2015.

[8] T. Lindeberg *Scale-space theory in computer vision*. Dordrecht: Kluwer Academic Publishers; 1994.

[9] J. S. Jin, Y. Gao - *Recursive implementation of Log Filtering*. Real-Time Imaging pp. 59-65, doi: 10.1006/rtim.1996.0045, 1997.

[10] R.J. Purser, W.-S. Wu, D.F. Parrish, N.M. Roberts - *Numerical Aspects of the Application of Recursive Filters to Variational Statistical Analysis. Part I: Spatially Homogeneous and Isotropic Gaussian Covariances*. Monthly Weather Review 131, pp. 1524-1535, doi: 10.1175//2543.1, 2003.

[11] C. Hayden, R. Purser - *Recursive filter objective analysis of meteorological field: applications to NESDIS operational processing*. Journal of Applied Meteorology 34, pp. 3-15, doi: 10.1175/1520-0450-34.1.3, 1995.

[12] L.V. Vliet, I. Young, P. Verbeek - *Recursive Gaussian derivative filters*. International Conference Recognition, pp. 509-514, doi: 10.1109/ICPR.1998.711192, 1998.

[13] L.J. van Vliet, P.W. Verbeek - *Estimators for orientation and anisotropy in digitized image*. Proc. ASCI'95, Heijen , pp. 442-450, 1995.

[14] B. Triggs, M. Sdika - *Boundary conditions for Young-van Vliet recursive filtering*. IEEE Transactions on Signal Processing, 54 (6 I), pp. 2365-2367, doi: 10.1109/TSP.2006.871980, 2006.

[15] A. Witkin - *Scale-space filtering*. Proc. Internat. Joint Conf. on Artificial Intelligence, Karlsruhe, germany, pp. 1019-1021, 1983.

[16] I.T. Young, L.J. van Vliet - *Recursive implementation of the Gaussian filter*. Signal Processing 44, pp. 139-151, doi: 10.1016/0165-1684(95)00020-E, 1995.