Available online at www.sciencedirect.com
ScienceDirect
Computers and Mathematics with Applications 55 (2008) 368–380
An International Journal
computers & mathematics
with applications
www.elsevier.com/locate/camwa

Stabilization of explicit methods for convection diffusion equations by discrete mollification☆

Carlos D. Acosta[a], Carlos E. Mejía[b,*]

[a] *Universidad Nacional de Colombia, Department of Mathematics and Statistics, Manizales, Colombia*
[b] *Universidad Nacional de Colombia, Department of Mathematics, Medellín, Colombia*

Received 22 May 2006; received in revised form 21 March 2007; accepted 5 April 2007

Abstract

The main goal of this paper is to show that discrete mollification is a simple and effective way to speed up explicit time-stepping schemes for partial differential equations. The second objective is to enhance the mollification method with a variety of alternatives for the treatment of boundary conditions. The numerical experiments indicate that stabilization by mollification is a technique that works well for a variety of explicit schemes applied to linear and nonlinear differential equations.
© 2007 Elsevier Ltd. All rights reserved.

Keywords: Convection–diffusion; Wave propagation; Discrete mollification; Stability analysis; Explicit schemes

1. Introduction

Since the 1980s, the mollification method has been used by a considerable number of authors as a regularization method for ill-posed problems [1–3]. More recently, in [4,5], the method was introduced as a stabilizer of the forward-time central-space explicit scheme for linear parabolic equations. By stabilizer we mean a technique that provides a way to increase the stability bound of the explicit method.

The main goal of this paper is to show that discrete mollification is a simple and effective way to speed up explicit time-stepping schemes for partial differential equations. To fulfill this objective, we concentrate on convection–diffusion equations and present a mollified forward-time central-space method. We establish stability bounds for linear equations and show through encouraging numerical experiments, that the same bounds are valid for some nonlinear cases.

The second goal of this paper is to enhance the mollification method with updated convergence results and a wide variety of alternatives for the treatment of boundary conditions. This is a delicate matter when dealing with convolutions; our approach is inspired by the methods of digital image processing.

☆ This work has been partially supported by COLCIENCIAS, project number 1118-11-16705 and DIME, project number 30802867.
* Corresponding author.
E-mail addresses: cdacostam@unal.edu.co (C.D. Acosta), cemejia@unalmed.edu.co (C.E. Mejía).

0898-1221/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2007.04.019

The outline of this paper is as follows: We begin with the definition and main properties of mollification in Section 2. The study of boundary conditions appears in Section 2.3, which is followed by the results on stabilization in Section 3. The last section presents illustrative numerical experiments.

2. Mollification

The mollification method is a filtering procedure, based on convolution, that is appropriate for the regularization of a variety of ill-posed problems, namely, inverse heat conduction problems, numerical differentiation, coefficient identification, problems related to digital signal processing, etc.

When mollification is applied to digital signal processing, it is convenient to have the possibility of working with different mollification kernels. However, in this paper, we restrict our attention to a Gaussian kernel. For an overview of the many applications of this method, we recommend [1,4] and the references therein.

2.1. Abstract setting

Let $\delta > 0$, $p > 0$ and

$$A_{p\delta} = \left(\int_{-p/\delta}^{p/\delta} \exp(-s^2) \mathrm{d}s \right)^{-1}.$$

We work with the following truncated Gaussian kernel:

$$\kappa_{\delta p}(t) = \begin{cases} A_{p\delta} \delta^{-1} \exp(-t^2/\delta^2), & |t| \le p \\ 0, & |t| > p. \end{cases}$$

This kernel satisfies: $\kappa_{\delta p} \ge 0$, $\kappa_{\delta p} \in C^\infty(-p, p)$, $\kappa_{\delta p}$ is zero outside $[-p, p]$ and $\int_{\mathbb{R}} \kappa_{\delta p} = 1$.

Given $f : \mathbb{R} \to \mathbb{R}$ locally integrable, we define its δp-mollification, denoted $J_{\delta p} f$, as the convolution of f with the kernel $\kappa_{\delta p}$. That is,

$$\begin{aligned} J_{\delta p} f(t) &= \left(\kappa_{\delta p} * f \right)(t) \\ &= \int_{-\infty}^{\infty} \kappa_{\delta p}(t - s) f(s) \mathrm{d}s \\ &= \int_{t-p}^{t+p} \kappa_{\delta p}(t - s) f(s) \mathrm{d}s \\ &= \int_{-p}^{p} \kappa_{\delta p}(-s) f(t + s) \mathrm{d}s. \end{aligned} \tag{1}$$

Roughly speaking, the parameters δ and p are related to shape and support of the mollification kernel respectively. Unless otherwise stated, we assume $p = 3\delta$ for mollification in the abstract setting. Thus

$$A_{p\delta} = \left(\int_{-p/\delta}^{p/\delta} \exp(-s^2) \mathrm{d}s \right)^{-1} = \left(\int_{-3}^{3} \exp(-s^2) \mathrm{d}s \right)^{-1} \tag{2}$$

is independent of δ.

If f is defined on a bounded set Y, the computation of the convolution $J_{\delta p} f$ requires either an extension of f to points out of Y or the restriction of f to a proper subset of Y. A useful approach for closed intervals was presented in [2]. Furthermore, in the last paragraph of Section 5 of [4], Murio says: "In general, if the initial and/or boundary conditions are known, they should be incorporated to the code to improve accuracy". Consequently, in this paper we deal with several data extension procedures, based on the ideas presented in [6,7]. We also include a scaling technique

as an alternative for computations near the boundary. Following [6], we refer to all the extensions and to the scaling procedure as *boundary conditions*. The details on this topic are presented in Section 2.3.

2.2. Discrete mollification

Definition 1. Let $X = \{x_j : x_j = x_0 + jh,\ j \in \mathbb{Z}\}$ be a discrete domain with x_0 and h given real numbers and $h > 0$. Let $G : X \to \mathbb{R}$ be a function defined by $G(x_j) = y_j$. Set

$$\begin{aligned} S_j &= \left(x_{j-1} + x_j\right)/2, \quad j \in \mathbb{Z} \\ I_j &= \left[S_j, S_{j+1}\right), \quad j \in \mathbb{Z} \\ f(t) &= \sum_{j=-\infty}^{\infty} y_j \chi_j(t), \quad t \in \mathbb{R}, \end{aligned} \tag{3}$$

with χ_j the characteristic function of I_j. Then for $\delta > 0$ and η a given non-negative integer, we define the $\delta\eta$-mollification of G as the δp-mollification of f with

$$p = (\eta + 1/2)\, h, \tag{4}$$

That is,

$$J_{\delta\eta} G(x) = J_{\delta p} f(x).$$

We are particularly interested in the value of $J_{\delta\eta} G$ at the points in X. Let

$$t_j = (j - 1/2)\, h, \quad j \in \mathbb{Z}. \tag{5}$$

Then we can write

$$\begin{aligned} J_{\delta\eta} G(x_j) &= J_{\delta p} f(x_j) \\ &= \int_{-p}^{p} \kappa_{\delta p}(-s) f(x_j + s) \mathrm{d}s \\ &= \sum_{i=-\eta}^{\eta} \int_{t_i}^{t_{i+1}} \kappa_{\delta p}(-s) f(x_j + s) \mathrm{d}s. \end{aligned}$$

Furthermore,

$$t_i < s < t_{i+1} \text{ if and only if } S_{i+j} < x_j + s < S_{i+j+1}.$$

Thus,

$$J_{\delta\eta} G(x_j) = \sum_{i=-\eta}^{\eta} w_i y_{j+i}, \quad \text{where } w_i = \int_{t_i}^{t_{i+1}} \kappa_{\delta p}(-s) \mathrm{d}s. \tag{6}$$

That is, the discrete mollification of G is the discrete convolution of the vector y with a kernel vector w of weights obtained from the kernel $k_{\delta p}$. Notice that w satisfies

$$\sum_{i=-\eta}^{\eta} w_i = 1.$$

For discrete mollification, the integer η is the parameter related to the support of the discrete mollification kernel.

Theorem 1 (*Convergence*). *Let g be a function defined on $\mathbb{R}$ with fourth derivative $g^{(4)}$ continuous and bounded in $\mathbb{R}$. Let G be its discrete version defined on X. If G^{ε} is another discrete function defined on X such that*

$$|G^{\varepsilon}(x_j) - G(x_j)| \leq \varepsilon, \quad \text{for } x_j \in X,$$

then, for each compact set $K = [a, b]$, there exists a constant $C = C(K)$ such that if $x_j \in K$

$$|J_{\delta\eta}G^{\varepsilon}(x_j) - J_{\delta\eta}G(x_j)| \leq \varepsilon,$$
$$|J_{\delta\eta}G(x_j) - g(x_j)| \leq Ch^2.$$

Additionally,

$$\begin{aligned} &|D_+ J_{\delta\eta}G(x_j) - g'(x_j)| \leq Ch, \\ &|D_0 J_{\delta\eta}G(x_j) - g'(x_j)| \leq Ch^2, \\ &|D_- D_+ J_{\delta\eta}G(x_j) - g''(x_j)| \leq Ch^2, \end{aligned} \tag{7}$$

where D_+, D_- and D_0 are the forward, backward and central finite difference operators respectively.

Proof. (Stability) In fact,

$$\begin{aligned} |J_{\delta\eta}G^{\varepsilon}(x_j) - J_{\delta\eta}G(x_j)| &= \left| \sum_{i=-\eta}^{\eta} w_i \left(G^{\varepsilon}(x_{j+i}) - G(x_{j+i}) \right) \right| \\ &\leq \sum_{i=-\eta}^{\eta} w_i |G^{\varepsilon}(x_{j+i}) - G(x_{j+i})| \leq \varepsilon. \end{aligned}$$

(Consistency) Let $x_j \in K = [a, b]$; then, from Taylor's theorem, for each $i = -\eta, \ldots, \eta$ there exists $\xi_i \in \tilde{K} = [a - p, b + p]$ such that

$$g(x_{j+i}) = g(x_j) + (ih)\, g'(x_j) + \frac{(ih)^2}{2} g''(\xi_i).$$

Then

$$\begin{aligned} J_{\delta\eta}G(x_j) &= \sum_{i=-\eta}^{\eta} w_i g(x_{j+i}) \\ &= \sum_{i=-\eta}^{\eta} w_i \left(g(x_j) + (ih)\, g'(x_j) + \frac{(ih)^2}{2} g''(\xi_i) \right) \\ &= g(x_j) + \frac{h^2}{2} \sum_{i=-\eta}^{\eta} i^2 w_i g''(\xi_i). \end{aligned} \tag{8}$$

Because, from symmetry in the kernel

$$\begin{aligned} \sum_{i=-\eta}^{\eta} w_i\, (ih)\, g'(x_j) &= hg'(x_j) \sum_{i=-\eta}^{\eta} i w_i \\ &= hg'(x_j) \sum_{i=1}^{\eta} i\, (w_i - w_{-i}) = 0. \end{aligned}$$

Hence,

$$\begin{aligned} |J_{\delta\eta}G(x_j) - g(x_j)| &= \frac{h^2}{2} \sum_{i=-\eta}^{\eta} i^2 w_i |g''(\xi_i)| \\ &\leq \frac{h^2}{2} \sum_{i=-\eta}^{\eta} i^2 w_i \|g''\|_{\infty,\tilde{K}}. \end{aligned}$$

(Numerical derivatives) As above, if $x_{j-1}, x_{j+1} \in K$, there exist $\xi_i, \zeta_i \in \tilde{K}$ such that

$$J_{\delta\eta}G(x_{j+1}) = g(x_{j+1}) + \frac{h^2}{2}g''(x_{j+1})\sum_{i=-\eta}^{\eta} i^2 w_i + \frac{h^4}{24}\sum_{i=-\eta}^{\eta} i^2 w_i g^{(4)}(\xi_i),$$
$$J_{\delta\eta}G(x_{j-1}) = g(x_{j-1}) + \frac{h^2}{2}g''(x_{j-1})\sum_{i=-\eta}^{\eta} i^2 w_i + \frac{h^4}{24}\sum_{i=-\eta}^{\eta} i^2 w_i g^{(4)}(\zeta_i).$$

Now, for some ϑ_j and ν_j between x_{j-1} and x_{j+1}, we have

$$\left|\frac{g(x_{j+1}) - g(x_{j-1})}{2h} - g'(x_j)\right| = \frac{h^2}{12}|g'''(\vartheta_j)| \le \frac{h^2}{12}\|g^{(3)}\|_{\infty,\tilde{K}},$$
$$\left|\frac{g''(x_{j+1}) - g''(x_{j-1})}{2h}\right| = |g'''(\nu_j)| \le \|g^{(3)}\|_{\infty,\tilde{K}}$$
$$\left|\frac{g^{(4)}(\xi_i) - g^{(4)}(\zeta_i)}{2h}\right| \le \frac{\|g^{(4)}\|_{\infty,\tilde{K}}}{h}.$$

Thus

$$\frac{J_{\delta\eta}G(x_{j+1}) - J_{\delta\eta}G(x_{j-1})}{2h} = g'(x_j) + O(h^2).$$

The other bounds in (7) are obtained in the same way. □

2.3. Boundary conditions

In this section we present the numerical procedure for the computation of the $\delta\eta$-discrete mollification of a data vector $y \in \mathbb{R}^m$. This mollification is denoted $y_{\delta\eta}$.

The procedure makes use of extensions of the data outside the domain and, in this sense, follows the ideas in [4] and other references on mollification, in which two constant extensions of the data are computed by solving two optimization problems. However, we take a different approach, based on the techniques for image reconstruction and digital signal processing described in [6], Section 7 of [7] and Chapter 24 of [8].

Let us assume, following Definition 1, that

$$y_j = G\left(x_j\right),$$

where $a = x_1 < x_2 < \cdots < x_m = b$ and $x_j - x_{j-1} = h > 0$ for $j = 2, \ldots m$. By (6),

$$\left[y_{\delta\eta}\right]_j = J_{\delta\eta}G(x_j) = \sum_{i=-\eta}^{\eta} w_i y_{j+i}, \quad \text{where } w_i = \int_{t_i}^{t_{i+1}} \kappa_{\delta p}(-s)\mathrm{d}s$$

and the $t_i's$ are given by (5), that is,

$$t_i = (i - 1/2)\,h, \quad i \in \mathbb{Z}.$$

This yields

$$y_{\delta\eta} = T_l y_l + T y + T_r y_r, \tag{9}$$

where

$$T_l = \begin{bmatrix} w_{-\eta} & \cdots & w_{-1} \\ & \ddots & \vdots \\ & & w_{-\eta} \\ 0 & & \end{bmatrix}_{m\times\eta}, \qquad y_l = \begin{bmatrix} y_{-\eta+1} \\ y_{-\eta+2} \\ \vdots \\ y_{-1} \\ y_0 \end{bmatrix}_{\eta\times 1},$$

$$T = \begin{bmatrix} w_0 & \cdots & w_\eta & & 0 \\ \vdots & \ddots & & \ddots & \\ w_{-\eta} & & \ddots & & w_\eta \\ & \ddots & & \ddots & \vdots \\ 0 & & w_{-\eta} & \cdots & w_0 \end{bmatrix}_{m\times m}, \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{m-1} \\ y_m \end{bmatrix}_{m\times 1}, \tag{10}$$

$$T_r = \begin{bmatrix} & & 0 \\ & & \\ w_\eta & & \\ \vdots & \ddots & \\ w_1 & \cdots & w_\eta \end{bmatrix}_{m\times\eta}, \qquad y_r = \begin{bmatrix} y_{m+1} \\ y_{m+2} \\ \vdots \\ y_{m+\eta-1} \\ y_{m+\eta} \end{bmatrix}_{\eta\times 1}.$$

The vectors y_l and y_r are our concern now. How do we define them? Some additional hypotheses are in order. They are called *Boundary Conditions*, and we describe them according to [6].

2.3.1. The zero (Dirichlet) boundary condition

In this case, y_l and y_r are null and $y_{\delta\eta}$ is computed from y by the Toeplitz matrix T. More precisely,

$$y_{\delta\eta} = Ty, \tag{11}$$

where T and y are given by (10).

This boundary condition is useful when it is known that y decays rapidly to zero outside the domain $[a, b]$. Of course, it becomes a liability if y does not have this property, as is clearly stated in [7,8] for digital image processing.

2.3.2. Scaled boundary condition

In this case, y_l and y_r are null, and $y_{\delta\eta}$ is computed from y by a scaled version $T_{\delta\eta}$ of matrix T given by

$$T_{\delta\eta} = D^{-1}T,$$

where $D = \operatorname{diag}[d_1, d_2, \ldots, d_m]$ and d_k is the sum of the elements in the kth row of T. Thus,

$$y_{\delta\eta} = T_{\delta\eta}y. \tag{12}$$

This scaling is recommended when there is no knowledge of the behavior of the function G outside the domain.

2.3.3. The periodic boundary condition

In this case,

$$y_l = \begin{bmatrix} y_{-\eta+1} \\ y_{-\eta+2} \\ \vdots \\ y_{-1} \\ y_0 \end{bmatrix} = \begin{bmatrix} y_{m-\eta+1} \\ y_{m-\eta+2} \\ \vdots \\ y_{m-1} \\ y_m \end{bmatrix}, \qquad y_r = \begin{bmatrix} y_{m+1} \\ y_{m+2} \\ \vdots \\ y_{m+\eta-1} \\ y_{m+\eta} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{\eta-1} \\ y_\eta \end{bmatrix}$$

and it is possible to compute discrete mollification in a straightforward way. The associated linear operator is the circulant matrix

$$C_{\delta\eta} = \left[0_{m\times(m-\eta)}|T_l\right] + T + \left[T_r|0_{m\times(m-\eta)}\right]$$

and

$$y_{\delta\eta} = C_{\delta\eta} y.$$

Since $C_{\delta\eta}$ is circulant, it can be diagonalized by the discrete Fourier matrix. This is an important advantage of using periodic boundary conditions.

Another common approach is to extend the data by reflection of the information in the domain. This reflection can be even or odd. In the even case, the mollification operator is a Toeplitz plus Hankel matrix. Section 3 of [6] is an appropriate reference at this point.

2.4. Parameter selection

When working with mollification as a regularization procedure, the parameter δ may be automatically selected by GCV, Generalized Cross Validation [9], and the integer η is obtained from δ as

$$\eta = \left\lfloor \frac{3\delta}{h} - \frac{1}{2} \right\rfloor,$$

which is consistent with Eq. (4). In this paper we establish that, even when boundary conditions are assumed, discrete mollification is defined by a linear operator. Therefore GCV can be applied in all cases. For the application of GCV to mollification, we recommend [4].

However, for stabilization, it is convenient to keep η as the main parameter. This means that we know in advance the number of points to take into account as support for the discrete mollification kernel. Given η, the value of δ is found from (4) as

$$\delta = \frac{\left(\eta + \frac{1}{2}\right) h}{3}.$$

3. Stabilization

Explicit methods for partial differential equations are subject to restrictions on the time step. By stabilization of an explicit scheme, we mean a procedure that speeds up computations by allowing greater time steps. Additionally, it is desirable that the method prevents the appearance of spurious oscillations.

To comply with the first task, we show that our stabilized schemes allow greater time steps than those established by regular stability restrictions of the CFL type; this is obtained by using the Theorem 2, below. For references on the subject, we recommend [10–12,5,4]. The second task is relevant when dealing with non-linear equations like the viscous Burgers' equation.

Our approach is stabilization via discrete mollification applied to explicit schemes for convection–diffusion. Our results generalize those in [5] and Section 5 of [4].

Theorem 2. *Let*

$$v_m^{n+1} = \sum_{j=-\eta-1}^{\eta+1} W_j v_{m+j}^n,$$

be a time-stepping numerical scheme for solving the convection–diffusion equation

$$u_t + a u_x = b u_{xx},$$

on a equally spaced discrete domain

$$x_1 < x_2 < \cdots < x_N.$$

If the entries' modulus of the Fourier transform of the $\mathbb{R}^N$ vector

$$\begin{bmatrix} W_0 & W_1 & \cdots & W_{\eta+1} & 0 & \cdots & 0 & W_{-\eta-1} & \cdots & W_{-1} \end{bmatrix} \tag{13}$$

are all less than or equal to one, then the scheme is stable under periodic boundary conditions.

Proof. Under periodic boundary conditions, the numerical solution at t_{n+1} can be computed from the solution at t_n, by multiplying the latter by the circulant matrix with first row (13). But, the eigenvalues of such a matrix are those entries in the Fourier transform of (13). Then the spectral radius of the iteration matrix is the infinity norm of the Fourier transform of (13). So, from the hypothesis, the iteration matrix has spectral radius less than or equal to one, thus the scheme is stable. □

This simple result gives a sufficient condition for stability. Furthermore, the conditions for the theorem are very easy to check.

3.1. Convection–diffusion equation

Consider the Convection–Diffusion Equation

$$u_t + au_x = bu_{xx} \tag{14}$$

and the forward-time central-space finite difference scheme (FTCS) for this equation,

$$\frac{v_m^{n+1} - v_m^n}{k} + a\frac{v_{m+1}^n - v_{m-1}^n}{2h} = b\frac{v_{m+1}^n - 2v_m^n + v_{m-1}^n}{h^2}. \tag{15}$$

Here, v_m^n is the discrete approximation to the value of u at the nodal point (x_m, t_n), h and k are the uniform mesh sizes for the space and the time variables x and t respectively. Notice that (15) is equivalent to

$$v_m^{n+1} = v_m^n + b\mu(1+\alpha)v_{m-1}^n - 2b\mu v_m^n + b\mu(1-\alpha)v_{m+1}^n, \tag{16}$$

where

$$\mu = \frac{k}{h^2}, \qquad \lambda = \frac{k}{h} \quad \text{and} \quad \alpha = \frac{ha}{2b} = \frac{a\lambda}{2b\mu}.$$

The stability bounds for this scheme are

$$b\mu \le 1/2, \qquad \alpha \le 1.$$

Let $z_m^n = J_{\delta\eta}v^n(x_m)$ be the $\delta\eta$-mollification (in space) of v^n evaluated at x_m. We consider two mollified versions for (16), the first one

$$v_m^{n+1} = v_m^n + b\mu(1+\alpha)z_{m-1}^n - 2b\mu z_m^n + b\mu(1-\alpha)z_{m+1}^n, \tag{17}$$

results from replacing the data coming from spatial discrete differentiation for their mollified versions. The second mollified scheme is obtained from (16) by replacing the right hand side terms by their mollified versions. This yields

$$v_m^{n+1} = z_m^n + b\mu(1+\alpha)z_{m-1}^n - 2b\mu z_m^n + b\mu(1-\alpha)z_{m+1}^n. \tag{18}$$

To analyze the convergence of (17), we rewrite it in terms of the definition of discrete mollification. We begin by setting $w_j = 0$ for $|j| > \eta$, so that

$$z_m^n = \sum_{j=-\eta}^{\eta} w_j v_{m+j}^n = \sum_{j=-\eta-1}^{\eta+1} w_j v_{m+j}^n,$$

$$z_{m-1}^n = \sum_{j=-\eta}^{\eta} w_j v_{m-1+j}^n = \sum_{j=-\eta-1}^{\eta+1} w_{j+1} v_{m+j}^n,$$

$$z_{m+1}^n = \sum_{j=-\eta}^{\eta} w_j v_{m+1+j}^n = \sum_{j=-\eta-1}^{\eta+1} w_{j-1} v_{m+j}^n.$$

This yields

$$v_m^{n+1} = v_m^n + b\mu \sum_{j=-\eta-1}^{\eta+1} \left\{(1+\alpha)w_{j+1} - 2w_j + (1-\alpha)w_{j-1}\right\} v_{m+j}^n,$$

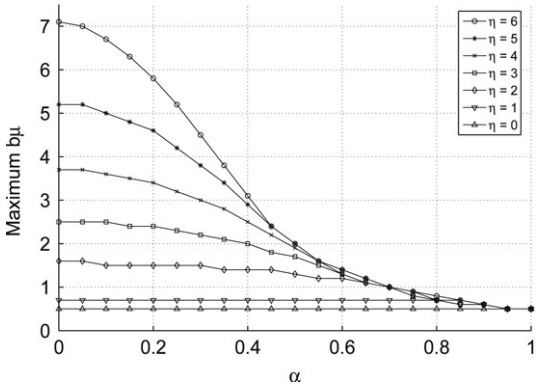

Fig. 1. CFL condition for the mollified FTCS Scheme (19).

or equivalently

$$v_m^{n+1} = \sum_{j=-\eta-1}^{\eta+1} \left[\delta_{0,j} + b\mu\left\{(1+\alpha)w_{j+1} - 2w_j + (1-\alpha)w_{j-1}\right\}\right] v_{m+j}^n, \tag{19}$$

where $\delta_{0,j}$ represents the Kronecker's delta. Similarly, we can write (18) in the form

$$v_m^{n+1} = \sum_{j=-\eta-1}^{\eta+1} \left[w_j + b\mu\left\{(1+\alpha)w_{j+1} - 2w_j + (1-\alpha)w_{j-1}\right\}\right] v_{m+j}^n. \tag{20}$$

3.1.1. Stability

Theorem 2 can be applied to schemes (19) and (20). Fig. 1 shows the maximum $b\mu$ allowed for the scheme (19). From this figure, some additional remarks have to be made:

(1) For values of α close to 1, there is no essential improvement in the stability of the original scheme.
(2) For values of α close or equal to 0, the stability bound is greater than $\frac{1}{2}$ whenever mollification is used ($\eta > 0$). As a consequence, when solving the convection–diffusion equation with α close to 1, one can increase the space resolution by reducing h without having to select a very small k.
(3) The case $\alpha = 0$ corresponds to the heat equation, for which stabilization by mollification was considered by Murio in [4]. There is agreement between our results and Murio's.

The CFL condition for the mollified scheme (20) is shown in Fig. 2. The main feature here is stabilization for α close to 1, which is missing from the previous scheme.

3.1.2. Consistency

For consistency with the problem

$$u_t + au_x = bu_{xx},$$

we rewrite (17) and (18) as

$$\frac{v_m^{n+1} - v_m^n}{k} + aD_0 z_m^n = bD_2 z_m^n, \tag{21}$$

$$\frac{v_m^{n+1} - z_m^n}{k} + aD_0 z_m^n = bD_2 z_m^n, \tag{22}$$

respectively, and note that

$$u_t - \frac{v_m^{n+1} - v_m^n}{k} = \mathcal{O}(k)$$

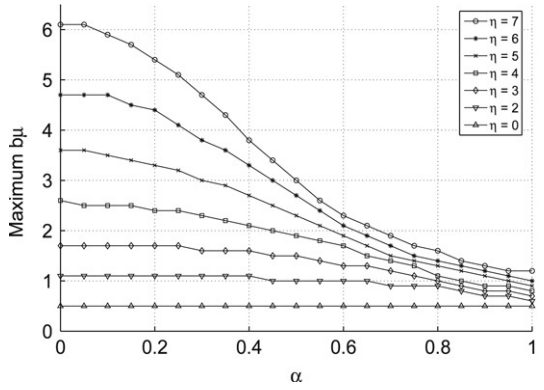

Fig. 2. CFL condition for the mollified FTCS scheme (20).

$$u_t - \frac{v_m^{n+1} - z_m^n}{k} = \mathcal{O}\left(k + \frac{h^2}{k}\right)$$
$$u_x(x_m, t_n) - D_0 z_m^n = \mathcal{O}\left(h^2\right)$$
$$u_{xx}(x_m, t_n) - D_2 z_m^n = \mathcal{O}\left(h^2\right).$$

From here, we obtain the consistency of the schemes as $k, h \to 0$, as long as $h^2/k \to 0$ as well.

4. Numerical examples

In this section we illustrate the stabilization property of the mollified explicit schemes by testing them with a variety of examples, based on the convection–diffusion equation

$$u_t + a u_x = b u_{xx}. \tag{23}$$

Some of the examples are illustrations of the theory presented above and the others are preliminary encouraging experiments that show the potential of discrete mollification as stabilizer of explicit marching schemes for partial differential equations. All of the examples were implemented using MATLAB 7.0 SP1 under Windows XP SP2 on a PC with a 3.02GHz Intel Pentium 4 processor and 512 Mb of RAM.

Example 1 (*Diffusion-Dominated*). Mollified Forward-time Central-space scheme (19) for the homogeneous heat equation

$$u_t = u_{xx}, \quad 0 < x < 1,\ 0 < t$$
$$u(x, 0) = \sin(\pi x), \quad 0 < x < 1,$$
$$u(0, t) = u(1, t) = 0, \quad 0 < t.$$

The exact solution is known. For the table, we use $h = 1/64$, the Neumann (odd extension) boundary condition and the maximum μ allowed. The error norms are computed for all x and a set of t-values on the discrete grid.

η	$b\mu$	Inf error	L_2 error	Time (s)
0	0.5	1.4563e−4	2.3612e−4	1.532
1	0.6475	2.5998e−3	7.3468e−4	1.234
2	1.48	6.4994e−3	1.8757e−3	0.547
3	2.3125	8.3427e−3	2.5916e−3	0.344
4	3.4225	8.6844e−3	3.1920e−3	0.234
5	4.68	8.7602e−3	3.8941e−3	0.172
6	6.2125	7.9251e−3	4.8629e−3	0.125

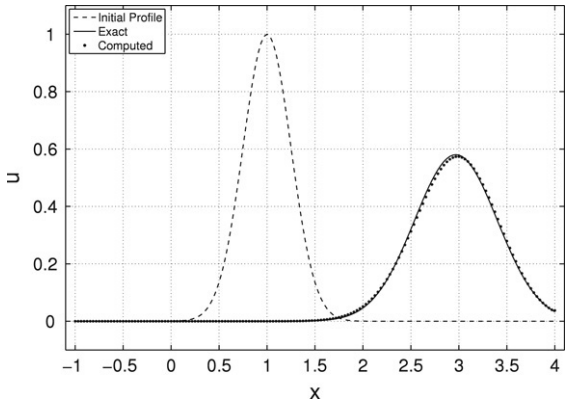

Fig. 3. Example 02 at $t = 2$, with $\alpha = 0.5$, $\eta = 6$ and $b\mu = 1.5$.

Example 2 (*Convection-Dominated*). Mollified Forward-time Central-space scheme for the convection–diffusion equation (23) with

$$u(x,t) = \frac{1}{\sqrt{1+t}} \exp\left(-\frac{(x-a(1+t))^2}{4b(1+t)}\right).$$

This example is presented at [13] as a test problem for studying the effect of dissipation and dispersion in two particular schemes. Here, we use this example to compare schemes (19) and (20). Actually, this problem requires us to choose a proper value for the parameter $b\mu$ inside the stability region in order to get good accuracy. The original scheme FTCS (16) yields acceptable results for $b\mu = 0.125$. Scheme (19) provides good results for $b\mu = 0.25$ with $\alpha \leq 0.5$ but for a greater α it is required a smaller $b\mu$. In contrast, scheme (20) is capable of solving the problem with good accuracy and using $b\mu$ values greater than 0.5. So, only in this case can we talk of an effective stabilization. The following tables show the maximum absolute error (ε) at $t = 2$ for scheme (20) with several values of α, $b\mu$ and η. The experiments were run with $x \in [-1, 4]$, $t \in [0, 2]$, $a = 1$, $h = 1/64$ and $b = ha\alpha^{-1}/2$. For other values of h, the same value of $b\mu$ will perform well. Fig. 3 illustrates this example for $\alpha = 0.5$, $\eta = 6$ and $b\mu = 1.5$.

α	(20), $\eta = 4$	(20), $\eta = 6$	(20), $\eta = 8$
0.25	$\varepsilon = 9.4212\text{e}{-003}$ $b\mu = 2.25$	$\varepsilon = 1.1833\text{e}{-002}$ $b\mu = 3.0$	$\varepsilon = 1.2065\text{e}{-002}$ $b\mu = 4.0$
0.5	$\varepsilon = 1.0714\text{e}{-002}$ $b\mu = 1.125$	$\varepsilon = 1.2710\text{e}{-002}$ $b\mu = 1.5$	$\varepsilon = 7.1431\text{e}{-003}$ $b\mu = 2.0$
0.75	$\varepsilon = 1.0637\text{e}{-002}$ $b\mu = 0.75$	$\varepsilon = 1.3850\text{e}{-002}$ $b\mu = 1.0$	$\varepsilon = 1.7370\text{e}{-002}$ $b\mu = 1.375$
1.0	$\varepsilon = 1.2062\text{e}{-002}$ $b\mu = 0.5625$	$\varepsilon = 2.0333\text{e}{-002}$ $b\mu = 0.75$	$\varepsilon = 1.9226\text{e}{-002}$ $b\mu = 1.0$

Example 3 (*Viscous Burger's Equation*). Mollified Forward-time Central-space scheme for the nonlinear equation

$$u_t + \frac{1}{2}\left(u^2\right)_x = bu_{xx}$$

with

$$u(x,t) = a - c\tanh\left(\frac{c}{2b}(x-at)\right). \tag{24}$$

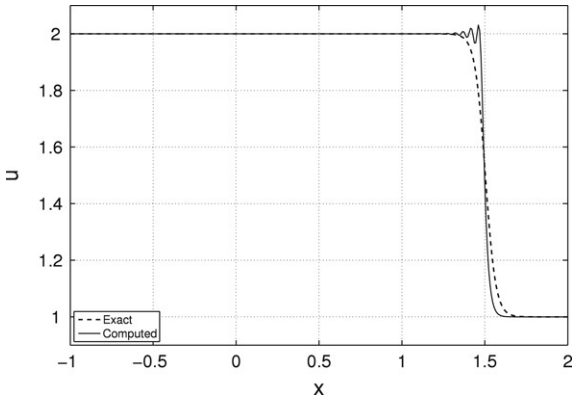

Fig. 4. Example 03 at $t = 1.0$, mollified FTCS (19) with $\eta = 4$ and $b\mu = 1.9$.

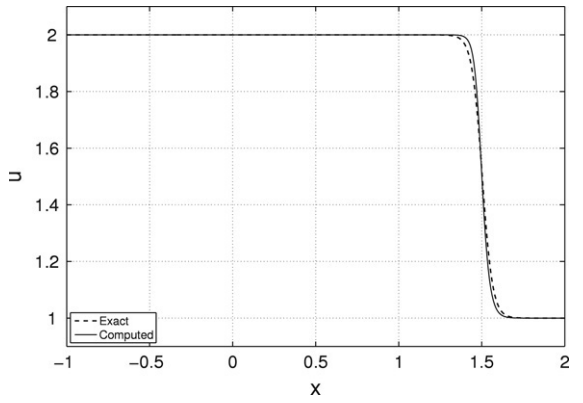

Fig. 5. Example 03 at $t = 1.0$, mollified FTCS (20) with $\eta = 4$ and $b\mu = 1.9$.

In this case, the two different implementations (19) and (20) stabilize but the first one does not prevent oscillations. Figs. 4 and 5 illustrate this behavior at $t = 1.0$ for $a = 1.5$, $c = 0.5$, $h = 1/64$, $b = h(a + c)$, $\eta = 4$ and $b\mu = 1.9$. This selection implies $\alpha = 0.5$.

The reason for this behavior is that, unlike (19), scheme (20) is a TVD method. We visualize discrete mollification as an effective way to stabilize computations and, at the same time, control spurious oscillations. Our results in this area will be the subject of a forthcoming paper. Finally, some remarks should be pointed out. The first example illustrates how stabilization can reduce the computing time by relaxing the stability bound. The second example shows that scheme (20) is more desirable than scheme (19) for convection dominated problems. The third example shows that the stabilization procedure presented here can also be successfully implemented for non-linear problems.

References

[1] D.A. Murio, The Mollification Method and the Numerical Solution of Ill-Posed Problems, John Wiley, 1993.
[2] C.E. Mejía, D. Murio, Mollified hyperbolic method for coefficient identification problems, Comput. Math. Appl. 26 (1993) 1–12.
[3] D. Murio, C.E. Mejía, S. Zhan, Discrete mollification and automatic numerical differentiation, Comput. Math. Appl. 35 (1998) 1–16.
[4] D.A. Murio, Mollification and space marching, in: K. Woodbury (Ed.), Inverse Engineering Handbook, CRC Press, 2002.
[5] L.J. Montoya, C.E. Mejía, F.M. Toro, Estabilización de esquemas por molificación discreta, Adv. Recur. Hidrául. 7 (2000) 102–116.
[6] M.K. Ng, R.H. Chan, W. Tang, A fast algorithm for deblurring models with neumann boundary conditions, SIAM J. Sci. Comput. 21 (3) (1999) 851–866.
[7] D. Calvetti, L. Reichel, Q. Zhang, Iterative solution methods for large linear discrete ill-posed problems, Appl. Comput. Control, Signals Circuits 1 (1999) 317–374.
[8] S.W. Smith, The Scientist and Engineer's Guide to Digital Signal Processing, second ed., California Technical Publishing, San Diego, California, 1999. URL: http://www.dspguide.com.
[9] G. Golub, M. Heath, G. Wahba, Generalized cross validation as a method for choosing a good ridge parameter, Tecnometrics 21 (2) (1979) 215–223.

[10] F.W. Wubs, Stabilization of explicit methods for hyperbolic partial differential equations, Internat. J. Numer. Methods Fluids 6 (1986) 641–657.
[11] V. Alexiades, G. Amienz, P. Gremaud, Super-time-stepping acceleration of explicit schemes for parabolic problems, Comm. Numer. Methods Engrg. 12 (1996) 31–42.
[12] K. Eriksson, C. Johnson, A. Logg, Explicit time-stepping for stiff odes, SIAM J. Sci. Comput. 25 (2003) 1142–1157.
[13] M.M. Cecchi, M.A. Pirozzi, High-order finite difference numerical methods for time-dependent convection-dominated problems, Appl. Numer. Math. 55 (2005) 334–356.