On efficient numerical methods for an initial-boundary value problem with nonlocal boundary conditions

J. Martín–Vaquero, B. A. Wade

\textsuperscript{a}Departamento de Matemática Aplicada, E.T.S.I.I. Béjar, Universidad de Salamanca
E37700, Bejar, Salamanca, Spain. jesmarva@usal.es

\textsuperscript{b}Department of Mathematical Sciences
University of Wisconsin–Milwaukee
Milwaukee, Wisconsin, 53201–0413, USA. wade@uwm.edu

Abstract

Many physical phenomena are modeled by nonclassical hyperbolic boundary value problems with nonlocal boundary conditions. In this paper, the problem of solving the one-dimensional wave equation subject to given initial and non-local boundary conditions is considered. These non-local conditions arise mainly when the data on the boundary can not be measured directly. Several finite difference methods with low order have been proposed in other papers for the numerical solution of this one dimensional non-classic boundary value problem. Here, we derive a new family of efficient three-level algorithms with higher order to solve the wave equation and also use a Simpson formula with higher order than before to approximate the integral conditions. Additionally, the fourth-order formula is also adapted to nonlinear equations, in particular to the well-known nonlinear Klein-Gordon equations which many physical problems can be modelled with. Numerical results are presented and are compared with some existing methods showing the efficiency of the new algorithms.

Keywords: Klein-Gordon equations, Nonlocal boundary conditions, Numerical methods, Wave equation

1. Introduction

In the last few years, many physical phenomena have been formulated with non-local mathematical models. These physical phenomena are modelled by non-classical parabolic [6, 7, 11, 12, 13, 21, 25, 27], elliptic [16],

The theoretical study of initial-boundary value problems in one space variable, which involve an integral over the spatial domain of a function of the desired solution that may appear in a boundary condition, is investigated in several articles [3, 4, 9, 15, 19]. In the last years, the development of numerical techniques for the solution of the hyperbolic nonlocal boundary value problems has been an important research topic in many branches of science and engineering. Particularly viscoelasticity has been the subject of some recent works [8, 24].

Here, we consider the following hyperbolic problem with two non-local constraints (in place of the standard boundary conditions):

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + q(x, t); \quad 0 \leq x \leq 1, \quad 0 \leq t \leq t_f,$$

(1)

(where $q(x, t)$ is a function sufficiently differentiable, $t_f \in R^+$), with initial condition

$$u(x, 0) = f_1(x), \quad 0 \leq x \leq 1,$$

(2)

$$u_t(x, 0) = f_2(x), \quad 0 \leq x \leq 1,$$

(3)

subject to the boundary conditions

$$u(0, t) = \int_0^1 \phi(x, t)u(x, t)dx + g_1(t), \quad 0 \leq t \leq t_f,$$

(4)

$$u(1, t) = \int_0^1 \psi(x, t)u(x, t)dx + g_2(t), \quad 0 \leq t \leq t_f,$$

(5)

where $q$, $f$, $g_1$, $g_2$, $\phi$ and $\psi$ are known functions (we will assume enough smoothness in the functions to get the desire orders of convergency of the algorithms).

In [1], a numerical procedure for solving a similar problem by reducing it into a system of linear algebraic equations. In [14], three computational techniques are compared. In this paper, we build a family of higher-order methods for a more general kind of problem and compare the new sixth-order method with the one called optimal in [14] in linear problems.
We also adapt the fourth-order algorithm for nonlinear equations, in particular we shall show the efficiency of the new formula with the dimensionless form of the phi-four equation \cite{26, 28}. This is a particular case of the Klein-Gordon nonlinear equation that arises in theoretical physics and it is the relativistic version of the Schrödinger’s equation:

\[
\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + F(u),
\]

(6)

It represents the equation of motion of a quantum scalar or a pseudo-scalar field, which is a field whose quanta are spinless particles. If \(F(u) = \sin(u)\), it is known as the Sine-Gordon equation:

\[
\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + \sin u,
\]

(7)

which also arises in a large number of areas of physics, for example, crystal dislocation theory, self-induced transparency, laser physics and particle physics \cite{5, 10, 18}; if \(F(u) = e^u\), (6) is the Liouville equation; if \(F(u) = \sinh(u)\), gives the sinh-Gordon equation.

2. The new three-level explicit methods

In \cite{14} three different schemes were proposed to solve a specific hyperbolic problem with non-local conditions. The author stated that the following so-called “optimal explicit technique”

\[
u_{i}^{n+1} = \frac{r(r-1)}{12}u_{i-2}^{n} + \frac{r(4-r)}{3}u_{i-1}^{n} + \frac{(4-r)(r-1)}{2}u_{i}^{n} + \frac{r(4-r)}{3}u_{i+1}^{n} + \frac{r(r-1)}{12}(u_{i+2}^{n} - u_{i-1}^{n}) + k^2 q_{i}^{n},
\]

(8)

was fourth-order, but the leading terms of the error are

\[
(4 - 5r + r^2)uk^{-2} + \frac{k^2(u^{(4,0)} - u^{0,4})}{12}
\]

(9)

and the order depends on the value of \(r\) \((r = k^2/h^2 = (\Delta t)^2/(\Delta x)^2)\); we are using the classic notation, see \cite{21, 22}, for example). Additionally, this technique requires four additional equations, except when \(r = 1\) or \(4\) (when only two are necessary). If \(r = 1\), both equations can be given through
numerical approximations of the conditions (4) and (5) as we are showing below.

The three procedures in [14] and the one that we shall derive at this paper are three-level techniques and, therefore, the solution on the initial time-step after is required. This is usually done through the Taylor expansion in time and equation (3); in this way, Dehghan gave [14]

\[ u_i^1 = u_i^0 + k(u_t)_i^0 + \frac{k^2(u_{i-1}^0 - 2u_i^0 + u_{i+1}^0)}{2h^2} + q(x_i, k). \]  

However, formula (10) is not convergent since if we apply Taylor series to the previous equation we obtain that the leading terms are

\[ -q + \frac{k^2q}{2} + O(k^3). \]  

There are better procedures for the first step, for example,

\[ u_i^1 = u_i^0 + k(u_t)_i^0 + \frac{k^2(u_{i-1}^0 - 2u_i^0 + u_{i+1}^0)}{2h^2} + \frac{k^2q(x_i, 0)}{2}. \]  

2.1. The new family of higher-order methods

Suppose that we have (1), when \( q(x, t) \) is cancelled, the following formula:

\[ u_i^{n+1} = u_{i-1}^n + u_{i+1}^n - u_i^{n-1}, \]

is exact for the homogeneous problem; therefore, it is possible to derive higher-order finite difference schemes for the non-homogenous problem, cancelling the Taylor expansions with the derivatives of \( q(x, t) \).

For example, a fourth-order method for (1) is

\[ u_i^{n+1} = u_{i-1}^n + u_{i+1}^n - u_i^{n-1} + k^2 \left( q_i^n + \frac{k^2((q_{xx})_i^n + (q_t)_i^n)}{12} \right). \]  

This algorithm is derived vanishing the two first terms of the truncation error when \( u_i^{n+1} = u_{i-1}^n + u_{i+1}^n - u_i^{n-1} \) is applied to the non-homogenous problem (1), and now, the leading term of the truncation error is

\[ \frac{k^4(q_{uuu} + q_{xxx} + q_{xxxx})}{360} + O(k^5), \]  

4
therefore a sixth-order method would be

\[ u_i^{n+1} = u_i^n + u_{i+1}^n - u_i^{n-1} + k^2 \left( q_i^n + \frac{k^2((q_{xx})_n^i) + (q_{tt})_n^i)}{12} + \frac{k^4((q_{xxxx})_n^i) + (q_{ttt})_n^i)}{360} \right), \]

whose leading term of the truncation error is

\[ \frac{k^6(u_{uuuu} - u_{xxxxxx})}{20160} + O(k^7). \] (16)

Again, we can also express this truncation error in terms of \( q \) and vanishing it we would get an eight-order formula and so on.

However, this kind of formulae is not frequently given in terms of \( q_{xx} \) or \( q_{tt} \), because it is possible to solve a more general problem if we are able to use \( q_n^i, q_{i+1}^n, q_{i-1}^n, \ldots \) instead. In this case, we can also solve efficiently the very interesting non-linear equation

\[ \frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + q(x, t, u); \quad 0 \leq x \leq 1, \quad 0 \leq t \leq t_f, \] (17)

for some \( q \) depending also on \( u \). For example, we can build a fourth-order algorithm (for linear equations) if we replace \( q_i^n + \frac{k^2((q_{xx})_n^i) + (q_{tt})_n^i)}{12} \) in (13) by

\[ \frac{q_i^{n+1} + q_{i+1}^n + q_{i-1}^n + q_i^{n-1}}{12} + \frac{2q_i^n}{3}, \]

and the leading term of the truncation error for equation

\[ u_i^{n+1} = u_i^{n-1} + u_{i+1}^n - u_i^{n-1} + k^2 \left( \frac{q_i^{n+1} + q_{i+1}^n + q_{i-1}^n + q_i^{n-1}}{12} + \frac{2q_i^n}{3} \right) \] (19)

is then

\[ \frac{k^4(-3q_{ttt} + 2q_{xtt} - 3q_{xxtt})}{720} + O(k^5), \] (20)

obviously, it is possible to do something similar with the sixth- and eight-order formulas, but in this case, it would be necessary to evaluate \( q \) in more points.
2.2. Numerical approximation of the nonlocal conditions

Equations (13), (15) or (19) present \( M - 1 \) (for \( i = 1, \ldots, M - 1 \)) linear equations in \( M + 1 \) (for \( i = 0, \ldots, M \)) unknown variables for each \( n \). In order to solve the linear system, we need two more equations, which we can obtain (as in [22]) by approximating the integrals (4) and (5) numerically by the fourth- or the sixth-order Simpson’s composite formulas.

The fourth-order Simpson’s composite formula can be formulated in the following way:

\[
\begin{align*}
u_0^{n+1} &= u(0, t^{n+1}) = \int_0^1 \phi(x, t^{n+1}) u(x, t^{n+1}) dx + g_1(t^{n+1}) = \\
&= \frac{h}{3} \left( \phi_0^{n+1} u_0^{n+1} + 4 \sum_{j=1}^{M/2} \phi_{2j-1}^{n+1} u_{2j-1}^{n+1} + 2 \sum_{j=1}^{M/2-1} \phi_{2j}^{n+1} u_{2j}^{n+1} + \phi_M^{n+1} u_M^{n+1} \right) + g_1^{n+1} + O(h^4),
\end{align*}
\]

(21)

and

\[
\begin{align*}
u_M^{n+1} &= u(1, t^{n+1}) = \int_0^1 \psi(x, t^{n+1}) u(x, t^{n+1}) dx + g_2(t^{n+1}) = \\
&= \frac{h}{3} \left( \psi_0^{n+1} u_0^{n+1} + 4 \sum_{j=1}^{M/2} \psi_{2j-1}^{n+1} u_{2j-1}^{n+1} + 2 \sum_{j=1}^{M/2-1} \psi_{2j}^{n+1} u_{2j}^{n+1} + \psi_M^{n+1} u_M^{n+1} \right) + g_2^{n+1} + O(h^4),
\end{align*}
\]

(22)

\( M \) being even.

Therefore, we have

\[
(h \phi_0^{n+1} - 3) u_0^{n+1} + 4 h \phi_1^{n+1} u_1^{n+1} + \ldots + h \phi_M^{n+1} u_M^{n+1} \simeq -3 g_1^{n+1},
\]

(23)

and

\[
h \psi_0^{n+1} u_0^{n+1} + 4 h \psi_1^{n+1} u_1^{n+1} + \ldots + (h \psi_M^{n+1} - 3) u_M^{n+1} \simeq -3 g_2^{n+1}.
\]

(24)

Accordingly, we can write the two linear equations that we need as:

\[
u_0^{n+1} = Y^{-1} (W_1 (h \psi_M^{n+1} - 3) - W_2 h \phi_M^{n+1}),
\]

(25)

and

\[
u_M^{n+1} = Y^{-1} (W_2 (h \phi_0^{n+1} - 3) - W_1 h \phi_M^{n+1}),
\]

(26)
where

\[ W_1 = -2h \left( \sum_{i=1}^{M/2} \phi_{2i-1}^{n+1} u_{2i-1}^{n+1} \right) + -4h \left( \sum_{i=1}^{M/2-1} \phi_{2i}^{n+1} u_{2i}^{n+1} \right) - 3g_1^{n+1}, \quad (27) \]

\[ W_2 = -2h \left( \sum_{i=1}^{M/2} \psi_{2i-1}^{n+1} u_{2i-1}^{n+1} \right) + -4h \left( \sum_{i=1}^{M/2-1} \psi_{2i}^{n+1} u_{2i}^{n+1} \right) - 3g_2^{n+1}, \quad (28) \]

and

\[ Y = (h\phi_0^{n+1} - 3)(h\psi_M^{n+1} - 3) - h^2\phi_0^{n+1}\psi_M^{n+1} = 9 - 3h(\phi_0^{n+1} + \psi_M^{n+1}). \quad (29) \]

We can also derive a sixth-order formula to approximate the integrals:

\[ u_0^{n+1} = u(0, t^{n+1}) = \int_0^1 \phi(x, t^{n+1})u(x, t^{n+1})dx + g_1(t^{n+1}) = \]

\[ \frac{2h}{65}(7\phi_0^{n+1} u_0^{n+1} + 32 \sum_{j=1}^{M/2} \phi_{2j-1}^{n+1} u_{2j-1}^{n+1} + 12 \sum_{j=0}^{M/4-1} \phi_{4j+2}^{n+1} u_{4j+2}^{n+1} + 14 \sum_{j=0}^{M/4-2} \phi_{4j+4}^{n+1} u_{4j+4}^{n+1} + 7\phi_M^{n+1} u_M^{n+1}) + g_1^{n+1} + O(h^6), \quad (30) \]

and

\[ u_M^{n+1} = u(1, t^{n+1}) = \int_0^1 \psi(x, t^{n+1})u(x, t^{n+1})dx + g_2(t^{n+1}) = \]

\[ \frac{2h}{65}(7\psi_0^{n+1} u_0^{n+1} + 32 \sum_{j=1}^{M/2} \psi_{2j-1}^{n+1} u_{2j-1}^{n+1} + 12 \sum_{j=0}^{M/4-1} \psi_{4j+2}^{n+1} u_{4j+2}^{n+1} + 14 \sum_{j=0}^{M/4-2} \psi_{4j+4}^{n+1} u_{4j+4}^{n+1} + 7\psi_M^{n+1} u_M^{n+1}) + g_2^{n+1} + O(h^6), \quad (31) \]

in this case, we need \( M \) being a multiple of 4.

Hence, we can write the two linear relations that we needed as:

\[ u_0^{n+1} = Y_6^{-1}(W_{6,1}(14h\psi_M^{n+1} - 45) - W_{6,2}14h\phi_M^{n+1}), \quad (32) \]

and

\[ u_M^{n+1} = Y_6^{-1}(W_{6,2}(14h\phi_0^{n+1} - 45) - W_{6,1}14h\psi_M^{n+1}), \quad (33) \]

where

\[ W_{6,1} = -64h \left( \sum_{i=1}^{M/2} \phi_{2i-1}^{n+1} u_{2i-1}^{n+1} \right) - 24h \left( \sum_{i=0}^{M/4-1} \phi_{4i+2}^{n+1} u_{4i+2}^{n+1} \right) \]

\[ -28 \left( \sum_{i=0}^{M/4-2} \phi_{4i+4}^{n+1} u_{4i+4}^{n+1} \right) - 45g_1^{n+1}, \quad (34) \]
\[ W_{6,2} = -64h \left( \sum_{i=1}^{M/2} \psi_{2i-1}^{n+1} u_{2i-1}^{n+1} \right) - 24h \left( \sum_{i=0}^{M/4-1} \psi_{4i+2}^{n+1} u_{4i+2}^{n+1} \right) \]
\[ - 28 \left( \sum_{i=0}^{M/4-2} \psi_{4i+4}^{n+1} u_{4i+4}^{n+1} \right) - 45g_2^{n+1}, \]  
(35)

and

\[ Y_6 = 2025 - 630h(\phi_0^{n+1} + \psi_M^{n+1}) + 196h^2(\phi_0^{n+1}\psi_M^{n+1} - \phi_M^{n+1} + \psi_0^{n+1}). \]  
(36)

3. Numerical examples

In this section we give some results of numerical experiments with the methods shown on the preceding sections and compare with algorithms proposed in [14], to support our theoretical discussion. Since the principal goal is showing the efficiency of the new methods, then we have considered that the solution in the first step, \( u(k, x_i) \) is known for both methods for a better comparison of the algorithms.

**Test 1.** We will begin by considering the simple test problem (1)-(5) with

\[ g_1(t) = -(-2 + e)\ e^{-t-1}t, \quad 0 < t < 1, \]
\[ g_2(t) = -g_1(t), \quad 0 < t < 1, \]
\[ \phi(x, t) = 1, \quad 0 < x < 1, \]
\[ \psi(x, t) = 1 - x, \quad 0 < x < 1, \]
\[ q(x, t) = 2e^{-t-x}(t-x), \quad 0 \leq t \leq 1, \quad 0 < x < 1, \]

and with solution

\[ u(x, t) = xte^{-t-x}. \]

In Figure 1 we compare the step size employed with the maximum error at \( t \in [0, 1] \), that is, \( \max_{i=1, \ldots, M; j=1, \ldots, N} |U(ih, jk)_{aprox} - U(ih, jk)_{exact}| \) using the “optimal explicit technique” in [14] and the sixth-order algorithm that we derived in the previous section.

In Figure 2 we compare the CPU time employed and the maximum error at \( t \in [0, 1] \) \( (\max_{i=1, \ldots, M; j=1, \ldots, N} |U(ih, jk)_{aprox} - U(ih, jk)_{exact}|) \) using the same both methods. We can clearly check that the new method gets better results.

**Test 2.** In this case, we are going to choose the Klein-Gordon nonlinear equation with \( F(u) = u - u^3 \). This is called the dimensionless form of the
Figure 1: Numerical comparison of the step size employed versus the numerical errors in test 1. The “optimal explicit technique” is the algorithm proposed in [14].

Figure 2: Numerical comparison of the CPU time (seconds) versus the numerical errors in test 1. The “optimal explicit technique” is the algorithm proposed in [14].
phi-four equation [26, 28]. This equation arises in quantum field theory in the study of quartic interaction theory and its 1-soliton solution is given by

\[ u(x, t) = \frac{A}{\cosh \lambda}, \]

where \( A = \sqrt{2}, \lambda = B(x - \mu t), \) \( B = \frac{1}{\sqrt{\mu^2 - 1}}. \) Here, \( A \) is the amplitude of the soliton, \( B \) is the inverse width of the soliton and \( \nu \) is its velocity.

In this case, we have the exact solution so we can compare the numerical results, however it also can have perturbation terms [26]

\[ u_{tt} - u_{xx} - u + u^3 = \varepsilon(\alpha u + \beta u_t + \gamma u_x + \delta u_{xt} + \zeta u_{tt} + \eta u_{xxt} + \theta_{xxxx}) \] (37)

and being much more difficult to solve.

We have solved

\[ u_{tt} - u_{xx} - u + u^3 = 0, \quad 0 \leq x \leq 1, \quad 0 \leq t \leq 1, \] (38)

with the following boundary conditions

\[ u(0, t) = g_1(t), \quad 0 \leq t \leq 1, \] (39)

\[ u(1, t) = g_2(t), \quad 0 \leq t \leq 1, \] (40)

\( g_1(t) \) and \( g_2(t) \) being those that

\[ u(x, t) = \frac{\sqrt{2}}{\cosh \left( \frac{x - \mu t}{\sqrt{3}} \right)} \]

is the solution of the problem.

In Figure 3 we compare the CPU time employed with the maximum error at \( t \in [0, 1] \), that is, \( \max_{i=1,...,M;j=1,...,N} |U(ih, jk)_{aprox} - U(ih, jk)_{exact}| \) using the “optimal explicit technique” in [14] and the fourth-order algorithm that we derived in the previous section.

In Figure 4 we compare the CPU time employed and the maximum error at \( t = 1 \) \( \max_{i=1,...,M} |U(ih, 1)_{aprox} - U(ih, 1)_{exact}| \) using the same both methods. We can clearly check that the new method yields better results.

Acknowledgments: This work has supported by grant FS13-2008 from Fundación Memoria de D. Samuel Solórzano Barruso.
Figure 3: Numerical comparison of the CPU time (seconds) employed versus the numerical error at \( t \in [0, 1] \), in test 2. The “optimal explicit technique” is the algorithm proposed in [14].

Figure 4: Numerical comparison of the CPU time (seconds) employed versus the numerical errors at \( t = 1 \) in test 2. The “optimal explicit technique” is the algorithm proposed in [14].
References


