INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 5, Number 3, Pages 466–490

AN IMPROVED EXPLICIT SCHEME FOR AGE-DEPENDENT POPULATION MODELS WITH SPATIAL DIFFUSION

GALENA PELOVSKA

(Communicated by Lubin G. Vulkov)

Abstract. In this work we present three age-structured models with spatial dependence. We introduce an improved explicit method, namely Super-Time-Stepping (STS) developed for parabolic problems and we use its modification for the numerical treatment of our models. We explain how the acceleration scheme can be adapted to the age-dependent models. We prove convergence of the method in case of Dirichlet boundary conditions and we demonstrate the accuracy and the efficiency of the Modified STS comparing it with other numerical algorithms of same or higher order, namely the explicit, fully implicit and Crank-Nicolson standard schemes.

Key Words. population dynamics, age-dependence, linear diffusion, linear and nonlinear models, finite difference method, numerical acceleration, modified super-time-stepping

1. Introduction.

During the last years, when modeling how populations change in time, it has been common to take into account not only the age structure of the species, but also their distribution in space. In 1973 Gurtin introduced spatial spread in age-dependent populations [9]. Later on many other authors have investigated the analytical aspects of various age-structured models with linear or nonlinear diffusion (for instance [4], [5], [6], [12], [16], [17]). Concerning the numerical treatment of the models arising in the field of population dynamics, numerical methods for models including only age or space structure, have been studied extensively (see [20, 21, 22] and the references cited therein). Much less research work has been done on models that include both - age and space. In the works of Kim [10], Kim-Park [11] and Milner [18], nonlinear models with nonlinear diffusion are treated. They propose and analyze some mixed numerical algorithms combining finite difference methods along the characteristic lines and finite element methods in the spatial variables. In the case of linear fertility and mortality functions, Lopez and Trigiante [15] have developed a finite difference scheme for an age-dependent model with Dirichlet boundary conditions and linear population flux. Ayati [3] proposes a numerical method for a nonlinear model with nonlinear diffusion which allows the use of variable time steps and independent age and time discretization. In [19] Pelovska and Boyadzhiev show how an additional acceleration of the Modified STS scheme can be obtained.

Received by the editors September 2, 2007 and, in revised form, October 31, 2007.
2000 *Mathematics Subject Classification.* 35M10, 65N06, 65N12, 92B99.

In the present paper we propose a variation of the standard explicit scheme for the heat equation adapted for solving age-dependent population models with linear spatial diffusion. One of the main ideas used in the article is that along characteristics in the age-time direction the models below can be viewed as parabolic differential equations. The super-time-stepping algorithm that we employ is an acceleration method for explicit schemes for parabolic problems. STS relaxes the condition of stability at the end of every time step that is imposed for the normal explicit scheme and demands stability at the end of every super-step, where a super-step consists of K sub-steps. It implies that we can take larger time steps and consequently the total number of steps is reduced speeding up the computations, compared with the standard explicit scheme. The intermediate steps (sub-steps) are chosen non-uniformly from a formula that is given later.

The organization of the paper is as follows. In Section 2 we present the continuous models and our assumptions on them. In Section 3 the Super-Time-Stepping algorithm for the heat equation is presented and we show the connection between the problems with and without age structure. In Section 4 we give a modified version of STS adapted to our problems and we discuss its implementation. The convergence of the method is proved in Section 5. In Section 6 we give computational examples illustrating the benefits of the numerical scheme and we analyze our results. In Section 7 we make a brief discussion on the different approaches to the models with Neumann boundary conditions and in Section 8 we make some final remarks.

2. The continuous models.

The model we consider is similar to the Lotka-McKendrick's problem [20], but involving also the spatial structure of the individuals [19]. Let $p(a,t,x)$ be the **density of the population** where $a \in [0,a_+]$ denotes age and a_+ is the **maximum age**; $t>0$ denotes time and $x \in (0,1)$ denotes **spatial position**, then we have the following system:

$$
(1)\qquad \begin{cases} 1)\, p_t + p_a + \mu(a)p = Dp_{xx},\ a \in [0,a_+],\ t>0,\ x \in (0,1) \\ 2)\, p(0,t,x) = \displaystyle\int_0^{a_+} \beta(a)p(a,t,x)\ \mathrm{d}a = B(t,x),\ t>0,\ x \in (0,1) \\ 3)\, p(a,0,x) = p_0(a,x),\ a \in [0,a_+],\ x \in (0,1) \\ 4)\, p(a,t,0) = p(a,t,1) = 0,\ a \in [0,a_+],\ t>0 \end{cases}
$$

The functions $\beta(a)$ and $\mu(a)$ represent the **age specific fertility** and the **age specific mortality respectively**; $p_0(a,x)$ is the **initial distribution**; D is the **dispersal modulus** being constant; $B(t,x)$ is the **birth rate**, which gives the total number of offspring in one time unit at position x. Since homogeneous Dirichlet conditions on the boundaries of the region (0,1) are considered as "extremely inhospitable" (see [6]), we consider the same model but with Neumann boundary conditions which are imposed to describe a population without immigration or emigration:

$$
(2)\qquad \begin{cases} 1)\, p_t + p_a + \mu(a)p = Dp_{xx},\ a \in [0,a_+],\ t>0,\ x \in (0,1) \\ 2)\, p(0,t,x) = \displaystyle\int_0^{a_+} \beta(a)p(a,t,x)\ \mathrm{d}a = B(t,x),\ t>0,\ x \in (0,1) \\ 3)\, p(a,0,x) = p_0(a,x),\ a \in [0,a_+],\ x \in (0,1) \\ 4)\, p_x(a,t,0) = p_x(a,t,1) = 0,\ a \in [0,a_+],\ t>0 \end{cases}
$$

In order to present a more realistic case where the species are with a finite life span and to allow the mathematical treatment of our problems, we want the maximum age a_+ to be finite (a $\in [0, a_+]$, where $a_+ < +\infty$) and we require that the **survival probability**

$$\pi(a) = e^{-\int_0^a \mu(\tau)d\tau} \tag{3}$$

vanishes at a_+. We assume:

- $\beta(.)$ is non-negative and belongs to $L^\infty(0, a_+)$;
- $\mu(.)$ is non-negative and belongs to $L^1_{loc}(0, a_+)$;
- $\int_0^{a_+} \mu(\tau)d\tau = +\infty$ (in order for the survival probability to vanish at a_+);
- $p_0 \in L^2([0, a_+]; (0, 1))$, $p_0(a, x) \geq 0$ for a.e. (a,x)$\in [0, a_+] \times (0, 1)$

We add the following compatibility condition that ensure the continuity of $p(a, t, x)$ along the characteristic lines, namely

$$p_0(0, x) = \int_0^{a_+} \beta(a)p_0(a, x)da, \tag{4}$$

where $p_0(a, x)$ satisfies the conditions of Dirichlet (1-4) or Neumann (2-4). Under these assumptions it is shown (see [4, 12, 13]), that there exists a unique solution $p \in C(0, T; L^2((0, a_+) \times (0, 1)))$, verifying problems (1) and (2).

Furthermore we consider the following variables:

$$\begin{cases} 1)\ w(a, t, x) = \dfrac{p(a, t, x)}{P(t)} \ \textbf{(age profile)} \\ 2)\ P(t) = \displaystyle\int_0^1 \int_0^{a_+} p(a, t, x)dadx \ \textbf{(total population)} \end{cases} \tag{5}$$

Then, substituting in (2) we obtain a nonlinear model which is a modified version of our initial-boundary value problem with Neumann conditions

$$\begin{cases} 1)\ w_t + w_a + \mu(a)w + Y(t)w = Dw_{xx}, a \in [0, a_+],\ t > 0,\ x \in (0, 1) \\ 2)\ w(0, t, x) = \displaystyle\int_0^{a_+} \beta(a)w(a, t, x)da,\ t > 0,\ x \in (0, 1) \\ 3)\ \displaystyle\int_0^1 \int_0^{a_+} w(a, t, x)dadx = 1,\ t > 0 \\ 4)\ w(a, 0, x) = w_0(a, x),\ a \in [0, a_+],\ x \in (0, 1) \\ 5)\ w_x(a, t, 0) = w_x(a, t, 1) = 0,\ a \in [0, a_+],\ t > 0 \end{cases} \tag{6}$$

$$\begin{cases} \dfrac{d}{dt}P(t) = Y(t)P(t) \\ P(0) = P_0 \end{cases} \tag{7}$$

where

$$w_0(a, x) = \frac{p_0(a, x)}{\displaystyle\int_0^1 \int_0^{a_+} p_0(a, x)dadx}, \quad P_0 = \int_0^1 \int_0^{a_+} p_0(a, x)dadx \tag{8}$$

and

$$Y(t) = \int_0^1 \int_0^{a_+} [\beta(a) - \mu(a)]w(a, t, x)dadx \tag{9}$$

This formulation has the advantage that "the function w is "smoother" than p ([5, 6])" and obviously, once $w(a,t,x)$ and $P(t)$ have been approximated, an approximation of $p(a,t,x)$ can be obtained by formula (5.1) and vice versa.

In order to approximate our models we have to use quadrature formulas such as the end-point rule or the trapezoidal rule for the integral terms. In [20] it is shown that this creates problems every time when an evaluation of the integrated function at the right endpoint a_+ of the interval is required, since $\lim_{a\to a_+}\mu(a)=\infty$. Following [20] we take

$$\begin{aligned} u(a,t,x) &= \frac{p(a,t,x)}{\pi(a)} \\ v(a,t,x) &= \frac{w(a,t,x)}{\pi(a)} \end{aligned} \tag{10}$$

and then substituting with the new variables u and v in equations (1), (2) and (6) respectively, we obtain:

(11)
$$\begin{cases} 1)\, u_t + u_a = Du_{xx}, \quad a\in[0,a_+],\ t>0,\ x\in(0,1) \\ 2)\, u(0,t,x) = \displaystyle\int_0^{a_+} \beta(a)\pi(a)u(a,t,x)\ \mathrm{d}a = B(t,x),\ t>0,\ x\in(0,1) \\ 3)\, u(a,0,x) = u_0(a,x),\ a\in[0,a_+],\ x\in(0,1) \\ 4)\, u(a,t,0) = u(a,t,1) = 0 \quad or \quad u_x(a,t,0) = u_x(a,t,1) = 0,\ a\in[0,a_+],\ t>0 \end{cases}$$

(12)
$$\begin{cases} 1)\, v_t + v_a + v\displaystyle\int_0^1\int_0^{a_+} [(\beta(a)-\mu(a))\pi(a)]v(a,t,x)dadx = Dv_{xx}, \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad a\in[0,a_+],\ t>0,\ x\in(0,1) \\ 2)\, v(0,t,x) = \displaystyle\int_0^{a_+} \beta(a)\pi(a)v(a,t,x)da,\ t>0,\ x\in(0,1) \\ 3)\, \displaystyle\int_0^1\int_0^{a_+} v(a,t,x)\pi(a)dadx = 1,\ t>0 \\ 4)\, v(a,0,x) = v_0(a,x),\ a\in[0,a_+],\ x\in(0,1) \\ 5)\, v_x(a,t,0) = v_x(a,t,1) = 0,\ a\in[0,a_+],\ t>0 \end{cases}$$

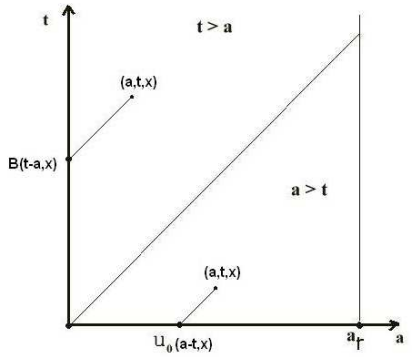
Written in this way, the qualitative features of the discussed models are preserved but there are no more problems with their numerical treatment (see [20] for details). Because of this reason in the sequel we shall apply numerical schemes on equations (11) and (12) and once knowing their approximate solution $u(a,t,x)$ and $v(a,t,x)$ respectively, we shall multiply it with $\pi(a)$ in order to yield results for $p(a,t,x)$ and $w(a,t,x)$ respectively.

3. Super-Time-Stepping method.

In this section we give the connection between (11) and the heat equation and we present the super-time-stepping method for parabolic equations. Let us make the following substitution

$$\gamma(s,x) = u(a_0+s, t_0+s, x) \tag{13}$$

where (a_0, t_0) is a certain point on the characteristic line $t = a + s$ (or $a = t + s$) as shown on the figure below.



Then, we obtain:

$$\begin{cases} \gamma_s = u_a + u_t = Du_{xx}(a_0 + s, t_0 + s, x) = D\gamma_{xx} \\ \gamma(s,0) = \gamma(s,1) = 0 \quad or \quad \gamma_x(s,0) = \gamma_x(s,1) = 0 \\ \gamma(0,x) = u(a_0, t_0, x), \end{cases} \tag{14}$$

where if $a_0 > t_0$, then $u(a_0 + s, t_0 + s, x) = u_0(a, x)$ and $u(a_0 + s, t_0 + s, x) = B(t, x)$ vice versa (see the figure above). This means that along the characteristic lines we can rewrite the governing equation in (11) as

$$\gamma_t = D\gamma_{xx}, \quad t > 0,\ x \in (0,1) \tag{15}$$

coupled with Dirichlet or Neumann boundary conditions

$$\gamma(t,0) = \gamma(t,1) = 0 \quad or \quad \gamma_x(t,0) = \gamma_x(t,1) = 0, \quad t > 0 \tag{16}$$

and initial conditions

$$\gamma(0,x) = \gamma_0(x), \quad x \in (0,1) \tag{17}$$

Consequently an approximation of this problem by a standard Euler explicit scheme is given by

(18)

$$\begin{cases} 1)\, \gamma_i^{n+1} = \dfrac{D\tau}{h^2}\gamma_{i-1}^n + (1 - \dfrac{2D\tau}{h^2})\gamma_i^n + \dfrac{D\tau}{h^2}\gamma_{i+1}^n, \quad i = 1, \ldots, M-1; n = 0, \ldots, N-1 \\ 2)\, \gamma_0^{n+1} = 0 \quad or \quad \gamma_0^{n+1} = (1 - \dfrac{2D\tau}{h^2})\gamma_0^n + \dfrac{2D\tau}{h^2}\gamma_1^n, \quad n = 0, \ldots, N-1 \\ 3)\, \gamma_M^{n+1} = 0 \quad or \quad \gamma_M^{n+1} = \dfrac{2D\tau}{h^2}\gamma_{M-1}^n + (1 - \dfrac{2D\tau}{h^2})\gamma_M^n, \quad n = 0, \ldots, N-1 \\ 4)\, \gamma(0,x) = \gamma_0(x), \end{cases}$$

where τ is the step size in time; h is the mesh size in space; by γ_i^{n+1} we have denoted an approximation of the exact solution $\gamma(s,x)$ at the mesh point (t^{n+1}, x_i) and for the end points $i = 0$ and $i = M$ we have employed a central-difference approximation (with a "fictitious" point) (see [21, 22]) in case of Neumann boundary conditions. Rewriting the scheme in a suitable way, we get

$$\begin{cases} \gamma^{n+1} = A\gamma^n, \quad n = 0, \ldots, N-1 \\ \gamma^0 = \gamma_0, \end{cases} \tag{19}$$

where A is an *(M-1)*$\times$ *(M-1)* symmetric, three-diagonal, positive definite matrix [22] and γ^n denotes a column vector with *M-1* elements.

Even though this scheme is computationally simple, it has a serious drawback - the scheme is stable if the time step is very small, namely $\tau \leq \frac{2}{\lambda_{max}}$ (λ_{max} is the biggest eigenvalue of the matrix A). This is the so called Courant-Friedrichs-Lewy condition (CFL), which in our case is: $\tau \leq \frac{h^2}{2D}$. Aiming to overcome this drawback and to increase the efficiency of the method by a slight modification in the code while keeping the accuracy at the same time we use the STS method for parabolic problems (see [2], [8]) whose idea is to require stability only at the end of a super-step ΔT, consisting of K sub-steps $\tau_1, \tau_2, ..., \tau_K$ with different length. These inner steps have no particular approximation properties and can be chosen explicitly in such a way that stability is ensured over the super-step

$$\Delta T = \sum_{k=1}^{K} \tau_k \tag{20}$$

Consequently the method can be associated with Runge-Kutta type methods with K stages.

Alexiades [2] uses the optimality properties of some modified Chebishev polynomials to give the following formula for τ_k:

$$\tau_k = \tau((-1+\nu)\cos(\frac{(2k-1)\pi}{2K}) + 1 + \nu)^{-1}, \quad k = 1, \ldots, K \tag{21}$$

where τ is the time step for the explicit scheme, calculated in such a way that the CFL (stability) condition is satisfied; ν is a number in the interval $(0, \frac{\lambda_{min}}{\lambda_{max}}]$ with λ_{min} and λ_{max} being the smallest and the biggest eigenvalues respectively of the matrix A in (19).

From the equation above it can be shown, [2], that:

$$\Delta T \to K^2\tau \quad as \quad \nu \to 0 \tag{22}$$

Analyzing this result we can conclude that for ν being close to 0 the super-step ΔT is K times faster than an explicit time step, i.e. the length of the time interval covered when executing K explicit steps $K\tau$ is K times shorter than a super-step, consisting of K sub-steps. It means that by a proper choice of $K$and ν, STS can accelerate an explicit scheme for parabolic equations up to K times.

In [2] it is shown that for each choice of K the standard explicit scheme (18-1), applied to equation (15), coupled with Dirichlet boundary conditions is stable and accurate for larger ν (< 1). But the larger the damping factor ν is, the shorter ΔT becomes, trading accuracy for speed. When ν is small, the method is faster but less accurate which is to be expected, since the time steps become larger. The same subordination can also be seen by our experiments.

4. Modified Super-Time-Stepping for age-structured problems.

Inspired by the approach described above, we note that the age of the species changes at the same rate as the time passes, so we take the step size in age identical to the step size in time and along the characteristic lines, we have the following numerical grid

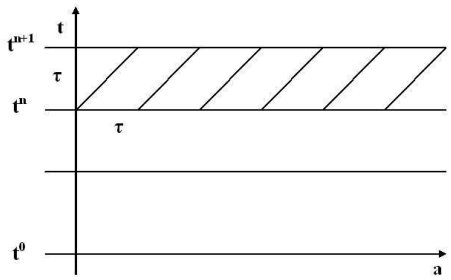


FIGURE 1. Along the characteristics

where $\tau > 0$ is the age and time discretization parameter and $\tau = \frac{a_+}{N}$ (N is the total number of subintervals in time). Let $h = \frac{1}{M}$ be the discretization step in space where M is the number of subintervals in space. Let $L = \frac{a_+}{\tau}$ be the number of discrete age steps. Then for each time level $t^n = n\tau,\ n = 0, ..., N$ we have the following grid: $\Gamma = \{(a^j, x_i) : a^j = j\tau,\ j = 0, ..., L; x_i = ih, i = 0, ..., M\}$. Let the discrete function U_i^j be an approximation of the solution of (15) at time level t^n at grid point (a^j, x_i) and $\hat{U}_i^{j+1}$ - at time level t^{n+1} at grid point (a^{j+1}, x_i). Then we approximate the directional derivative $\frac{\partial}{\partial t} + \frac{\partial}{\partial a}$, setting

$$\left(\frac{\partial}{\partial t} + \frac{\partial}{\partial a}\right) u(a^j, t^n, x_i) \approx \frac{\hat{U}_i^{j+1} - U_i^j}{\tau} \tag{23}$$

and the discrete Laplace operator

$$U_{xx} = \frac{U_{i-1}^j - 2U_i^j + U_{i+1}^j}{h^2} \tag{24}$$

Thus, an approximation of problem (11) (analogous to the one we applied to the heat equation) is as follows (see Figure 1):

(25)

$$\begin{cases} 1)\, \hat{U}_i^{j+1} = \frac{D\tau}{h^2} U_{i-1}^j + (1 - \frac{2D\tau}{h^2}) U_i^j + \frac{D\tau}{h^2} U_{i+1}^j,\ i = 1, \ldots, M-1;\ j = 0, \ldots, L-1 \\ 2)\, \hat{U}_0^{j+1} = 0 \quad or \quad \hat{U}_0^{j+1} = (1 - \frac{2D\tau}{h^2}) U_0^j + \frac{2D\tau}{h^2} U_1^j,\ j = 0, \ldots, L-1 \\ 3)\, \hat{U}_M^{j+1} = 0 \quad or \quad \hat{U}_M^{j+1} = \frac{2D\tau}{h^2} U_{M-1}^j + (1 - \frac{2D\tau}{h^2}) U_M^j,\ j = 0, \ldots, L-1 \end{cases}$$

Proceeding as in the case without age-structure we rewrite (25) in the form

$$\hat{U}^{j+1} = AU^j,\ j = 0, \ldots, L-1 \tag{26}$$

where A is the same *(M-1) × (M-1)* symmetric, tri-diagonal, positive definite matrix as in (19).

At the initial time $t = 0$ we consider $U_i^j = \frac{p_0(a^j, x_i)}{\pi(a^j)},\ j = 0, \ldots, L,\ i = 0, \ldots, M$ and for the boundary condition (11-2) we apply the trapezoidal rule, obtaining

$$\hat{U}_i^0 = \tau \sum_{j=1}^{L-1} \beta_j \pi(a_j) \hat{U}_i^j + \frac{\tau}{2} [\beta_0 \pi(a_0) \hat{U}_i^0 + \beta_L \pi(a_L) \hat{U}_i^L], \quad i = 0, \ldots, M \tag{27}$$

The previous description concerns a uniform grid (see Figure 1). Wishing to adapt STS to the age-structured problems, we consider a mesh as shown in Figure 2, which shows how one super-step looks like.

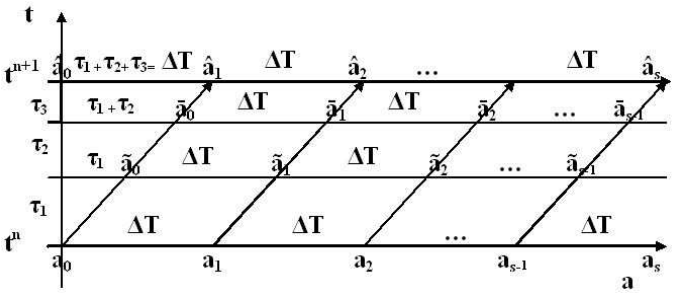


FIGURE 2. One super-time-step with 3 intermediate steps

The vertical and horizontal axes present the time and age distributions respectively; τ_k, $k = 1, \ldots, K$ are the inner-time-steps (on the graph we have drawn one super-step, consisting of three sub-steps). The first difference with STS for parabolic problems is that when stepping in time, we move also in age. Another particularity is that we have to calculate the solution at the boundary points as well. Because of these reasons, for the implementation of the modified STS scheme we proceed as follows: first we choose the value of ν and the number of intermediate steps, K; then we calculate τ_k, $k = 1, \ldots, K$, the length of ΔT and $s = \dfrac{a_+ K}{T}$, the number of age-nodes (see Figure 2), which depends on the choice of K. To initialize the procedure, we consider the discrete solution being identical to the analytical solution at time $t = 0$. Next we start using sub-steps τ_k, $k = 1, \ldots, K$ in time. Since we have multiple age nodes at each time level, we re-number the "inner" age-nodes in a convenient way (as shown on Figure 3.2) and we calculate the discrete solution at the k^{th} inner time level $k = 1, \ldots, K-1$ as follows

(28)

$$\hat{U}_i^j = \frac{D\tau_k}{h^2} U_{i-1}^j + (1 - \frac{2D\tau_k}{h^2}) U_i^j + \frac{D\tau_k}{h^2} U_{i+1}^j,\ i = 1, ..., M-1; j = 0, ..., s-1,$$
$$\hat{U}_0^j = \hat{U}_M^j = 0,\ j = 0, ..., s-1 \quad or$$
$$\hat{U}_0^j = (1 - \frac{2D\tau_k}{h^2}) U_0^j + \frac{2D\tau_k}{h^2} U_1^j,\ \hat{U}_M^j = \frac{2D\tau_k}{h^2} U_{M-1}^j + (1 - \frac{2D\tau_k}{h^2}) U_M^j,\ j = 0, s-1,$$

where U_i^j is the discrete solution at the $(k-1)^{st}$ time level and we consider it as known. One can see that within one super-step, we do not use the value of the discrete solution at the boundary points and that's why we do not calculate it. Another specification is that the discrete solution at the inner time levels has no particular approximation properties and consequently we do not output it. We use the approximation only at the last inner-time K corresponding to time level t^{n+1}, found by formula (25), but with time step τ_K, i.e. $\tau = \tau_K$. At this level we calculate the solution at the boundary point as well, by formula (27) and time step ΔT. The described procedure is repeated for each super-step and continues until we reach the end of the time interval.

In the nonlinear case (12), we proceed in an analogous way. The only difference here is that we need to approximate the double integral in the first equation with

a proper quadrature formula, namely we use the trapezoidal rule. Since we employ an explicit scheme, its application is trivial. The modified STS procedure is easy to use once we have developed it for the linear model. The difficulty comes from the fact that for the nonlinear problem no theoretical results are available. This implies that we cannot determine the upper bound for the parameter ν, but the results show that this is not a real obstacle since we can choose ν as a suitable number in the interval $(0,1)$, [1], and apply the modified STS also in this case. The good performance of STS-like algorithms for degenerate nonlinear parabolic problems is given in [7]. A method for an automatic time-step selection for STS applied to a Stefan-like problem, is proposed in [14].

While with a proper choice of K and ν, STS can accelerate an explicit scheme for parabolic equations up to K times, (22), in our case, as we do steps in time and in age, modified STS can speed up the explicit scheme up to K^2 times. If we consider also the fact that by the STS modification, we calculate the boundary condition much less frequently (we calculate it only at the end of each super-step) in comparison with the explicit scheme, this means the acceleration is even bigger.

5. Convergence of the method.

We shall show in this Section that under certain conditions on regularity of the coefficients of (11), the approximate solution defined by the modified STS converges to u, uniformly in ΔT, as $\Delta T \to 0$.
Let us first note that the formula of integration by parts

$$f(\Delta T) - f(0) = \Delta T f'(0) + \int_0^{\Delta T} (\Delta T - s) f''(s) ds$$

is equivalent to

$$f'(0) = \frac{f(\Delta T) - f(0)}{\Delta T} - \frac{1}{\Delta T} \int_0^{\Delta T} (\Delta T - s) f''(s) ds \tag{29}$$

Using this formula and equation (11-1) it can be shown that

$$\frac{u(a+\Delta T, t+\Delta T, x) - u(a,t,x)}{\Delta T} = D\frac{\partial^2 u}{\partial x^2} + \frac{1}{\Delta T}\int_0^{\Delta T} (\Delta T - s)\frac{\partial^2 u}{\partial \xi^2}(a+s, t+s, x) ds, \tag{30}$$

where we have considered the directional derivative in the characteristic direction $\xi = \frac{1}{\sqrt{2}}(1,1)$:

$$\frac{\partial}{\partial \xi} = (\frac{\partial}{\partial t}, \frac{\partial}{\partial a}).\xi$$

Thus, for $0 \le i \le M$, $0 \le j \le N_s$, $0 \le n \le N_t$ (N_s and N_t represent the number of subintervals in age and time respectively; $N_t = TN_s$ (since for simplicity we assume $a_+ = 1$) and we use the notation introduced in Section 4, replacing the step-size τ in age and time by ΔT) we have

$$\frac{\hat{u}_i^{j+1} - u_i^j}{\Delta T} - D\frac{\partial^2 u_i^j}{\partial x_i^2} = \frac{1}{\Delta T}\int_0^{\Delta T} (\Delta T - s)\frac{\partial^2 u}{\partial \xi^2}(a^j + s, t^n + s, x_i) ds, \tag{31}$$

which can be rewritten in the following form

$$\hat{u}_i^{j+1} = u_i^j + D\Delta T\frac{\partial^2 u_i^j}{\partial x_i^2} + \int_0^{\Delta T} (\Delta T - s)\frac{\partial^2 u}{\partial \xi^2}(a^j + s, t^n + s, x_i) ds \tag{32}$$

Let us now introduce the approximation error ε, defined by

$$\begin{cases} \varepsilon_i^j = u(a^j, t^n, x_i) - U_i^j = u_i^j - U_i^j, \\ \hat{\varepsilon}_i^{j+1} = u(a^{j+1}, t^{n+1}, x_i) - \hat{U}_i^{j+1} = \hat{u}_i^{j+1} - \hat{U}_i^{j+1}, \end{cases} \tag{33}$$

for $0 \le i \le M$, $0 \le j \le N_s$, $0 \le n \le N_t$.
For readers' convenience we give the definition of some norms, associated with ε:

$$\begin{aligned} ||\varepsilon_i||_{l^1} &= \Delta T \sum_{j=0}^{N_s} |\varepsilon_i^j|, \\ ||\varepsilon||_{l^1}^{l^\infty} &= \Delta T \max_{0\le i\le M} \sum_{j=0}^{N_s} |\varepsilon_i^j|, \\ ||\varepsilon||_{l^\infty}^{l^\infty} &= \max_{0\le i\le M} \max_{0\le j\le N_s} \{|\varepsilon_i^j|\} \end{aligned} \tag{34}$$

Here is the result about the convergence of the modified STS:

Theorem 1. *Let the solution $u(a,t,x)$ of (11) with conditions of Dirichlet on the boundary be continuously differentiable for (a,t,x) $\in (0, a_+) \times (0,T) \times (0,1)$ and its derivatives are bounded. Then there exists a constant $C > 0$ (independent of ΔT and depending on the norms indicated below), such that:*

$$\begin{aligned} &a)\, ||\varepsilon||_{l^1}^{l^\infty} \le C \left(||\tilde{\beta}||_{L^\infty} + ||\frac{\partial u}{\partial t}||_{L^\infty} + ||\frac{\partial u}{\partial a}||_{L^\infty} + ||\frac{\partial^2 u}{\partial \xi^2}||_{L^\infty} \right) \Delta T, \\ &b)\, ||\varepsilon||_{l^\infty}^{l^\infty} \le C \left(||\tilde{\beta}||_{L^\infty} + ||\frac{\partial u}{\partial t}||_{L^\infty} + ||\frac{\partial u}{\partial a}||_{L^\infty} + ||\frac{\partial^2 u}{\partial \xi^2}||_{L^\infty} \right) \Delta T, \end{aligned} \tag{35}$$

where with $\tilde{\beta}$ we have denoted the function $\tilde{\beta}(a) = \beta(a)\pi(a)$.

Proof: **1) i > K**
Let us consider K intermediate steps within one super-step (see Figure 2). Then, after one step in time ΔT, we have the following more convenient form of the modified STS scheme:

$$\begin{aligned} \hat{U}_i^{j+1} = {} & c_K(U_{i-K}^j + U_{i+K}^j) + c_{K-1}(U_{i-K+1}^j + U_{i+K-1}^j) + \dots \\ & + c_1(U_{i-1}^j + U_{i+1}^j) + c_0(U_i^j),\ i = 1, \dots, M-1, j = 0, \dots, N_s - 1 \end{aligned} \tag{36}$$

where c_k, $k = 0, \dots, K$ are **positive** (and **bounded**) coefficients which can be obtained explicitly by formula (38) given below. Subtracting (36) from (32) we obtain, for $i = 1, \dots, M-1, j = 0, \dots, N_s - 1, 0 \le n \le N_t - 1$, the error equation:

$$\begin{aligned} \hat{\varepsilon}_i^{j+1} = \hat{u}_i^{j+1} - \hat{U}_i^{j+1} = {} & \\ u_i^j + D\Delta T \frac{\partial^2 u_i^j}{\partial x_i^2} + \int_0^{\Delta T} (\Delta T - s) \frac{\partial^2 u}{\partial \xi^2}(a^j + s, t^n + s, x_i) ds - c_K(U_{i-K}^j + U_{i+K}^j) & \\ - c_{K-1}(U_{i-K+1}^j + U_{i+K-1}^j) - \dots - c_i(U_0^j + U_{2i}^j) - \dots - c_1(U_{i-1}^j + U_{i+1}^j) - c_0(U_i^j) & \end{aligned}$$

We add and subtract the exact solution in mesh points $(a^j, t^n, x_{i-K}), (a^j, t^n, x_{i+K})$, $(a^j, t^n, x_{i-K+1}), \ldots, (a^j, t^n, x_0)$, multiplied by proper coefficients and thus, we obtain:

$$\begin{aligned}\hat{\varepsilon}_i^{j+1} = u_i^j + D\Delta T\frac{\partial^2 u_i^j}{\partial x_i^2} + \int_0^{\Delta T}(\Delta T - s)\frac{\partial^2 u}{\partial \xi^2}(a^j + s, t^n + s, x_i)ds + c_K(\varepsilon_{K+i}^j + \varepsilon_{i-K}^j)\\ +c_{K-1}(\varepsilon_{i+K-1}^j + \varepsilon_{i-K+1}^j) + \ldots + c_i(\varepsilon_{2i}^j + \varepsilon_0^j) + \ldots + c_1(\varepsilon_{i-1}^j + \varepsilon_{i+1}^j) + c_0(\varepsilon_i^j) -\\ -c_K(u_{i-K}^j + u_{i+K}^j) - c_{K-1}(u_{i-K+1}^j + u_{i+K-1}^j) - \ldots - c_i(u_0^j + u_{2i}^j) - \ldots\\ -c_1(u_{i-1}^j + u_{i+1}^j) - c_0(u_i^j)\end{aligned}$$

Using Tailor expansions we have:

$$u_{i+k}^j = u(a^j, t^n, x_i + kh) = u_i^j + kh\frac{\partial u_i^j}{\partial x_i} + \frac{k^2h^2}{2}\frac{\partial^2 u_i^j}{\partial x_i^2} + \frac{k^3h^3}{3!}\frac{\partial^3 u_i^j}{\partial x_i^3} + O(h^4),$$

$$u_{i-k}^j = u(a^j, t^n, x_i - kh) = u_i^j - kh\frac{\partial u_i^j}{\partial x_i} + \frac{k^2h^2}{2}\frac{\partial^2 u_i^j}{\partial x_i^2} - \frac{k^3h^3}{3!}\frac{\partial^3 u_i^j}{\partial x_i^3} + O(h^4),$$

which implies

$$u_{i+k}^j + u_{i-k}^j = 2u_i^j + k^2h^2\frac{\partial^2 u_i^j}{\partial x_i^2} + O(h^3)$$

Hence we see that:

(37)

$$\begin{aligned}\hat{\varepsilon}_i^{j+1} = u_i^j + D\Delta T\frac{\partial^2 u_i^j}{\partial x_i^2} + \int_0^{\Delta T}(\Delta T - s)\frac{\partial^2 u}{\partial \xi^2}(a^j + s, t^n + s, x_i)ds + c_K(\varepsilon_{i+K}^j + \varepsilon_{i-K}^j)\\ +c_{K-1}(\varepsilon_{i+K-1}^j + \varepsilon_{i-K+1}^j) + \ldots + c_i(\varepsilon_{2i}^j + \varepsilon_0^j) + \ldots + c_1(\varepsilon_{i-1}^j + \varepsilon_{i+1}^j) + c_0(\varepsilon_i^j) -\\ -c_K(2u_i^j + K^2h^2\frac{\partial^2 u_i^j}{\partial x_i^2}) - c_{K-1}(2u_i^j + (K-1)^2h^2\frac{\partial^2 u_i^j}{\partial x_i^2}) - \ldots\\ -c_1(2u_i^j + h^2\frac{\partial^2 u_i^j}{\partial x_i^2}) - c_0(u_i^j) + O(h^3)\\ = u_i^j + D\Delta T\frac{\partial^2 u_i^j}{\partial x_i^2} + \int_0^{\Delta T}(\Delta T - s)\frac{\partial^2 u}{\partial \xi^2}(a^j + s, t^n + s, x_i)ds + c_K(\varepsilon_{i+K}^j + \varepsilon_{i-K}^j)\\ +c_{K-1}(\varepsilon_{i+K-1}^j + \varepsilon_{i-K+1}^j) + \ldots + c_i(\varepsilon_{2i}^j + \varepsilon_0^j) + \ldots + c_1(\varepsilon_{i-1}^j + \varepsilon_{i+1}^j) +\\ +c_0(\varepsilon_i^j) - h^2(K^2c_K + (K-1)^2c_{K-1} + \ldots c_1)\frac{\partial^2 u_i^j}{\partial x_i^2}\\ -(2c_K + 2c_{K-1} + \ldots 2c_1 + c_0)u_i^j + O(h^3)\end{aligned}$$

Now we give some results about the coefficients (36) of the method: the coefficients c_l^k, $l = 0, \ldots, k$ at the intermediate time level t^k, $k = 1, \ldots, K$ can be obtained by the following recursive formulas:

$$\begin{aligned}&c_0^k = (1 - 2\sigma_k)c_0^{k-1} + 2\sigma_k c_1^{k-1},\\ &c_l^k = \sigma_k(c_{l-1}^{k-1} + c_{l+1}^{k-1}) + (1 - 2\sigma_k)c_l^{k-1}, l = 1, \ldots, k-1,\\ &c_k^k = \sigma_k c_{k-1}^{k-1},\end{aligned} \tag{38}$$

where $\sigma_k = \dfrac{D\tau_k}{h^2}$; c_l^{k-1}, $l = 0, \ldots, k-1$ are the (non-zero) coefficients of the previous $(k-1)^{\text{st}}$ intermediate time level and c_l^{k-1}=0 for $l \geq k$, i.e. if a super-step consists

of K sub-steps, then for each sub-step k, $k = 1, \ldots, K$ we calculate the discrete solution by using formula (38), where all c_l^k, $l = 0, \ldots, k$ are dependent on the coefficients c_l^{k-1}, $l = 0, \ldots, k-1$ of the $(k-1)^{\text{st}}$ sub-step as shown above and to initialize this procedure we assume that in the beginning $c_0^0 = 1$, $c_l^0 = 0$, $l \geq 1$. Then, under these conditions the following relations hold:

Proposition 1. *At each (also intermediate) time level t^k, we have:*

$$2c_k^k + 2c_{k-1}^k + \ldots + 2c_1^k + c_0^k = 1,\ k = 1, \ldots, K \tag{39}$$

Proof: We will prove this statement by induction. Let $k = 1$, then using formula (38) we get:

$$c_0^1 = 1 - 2\sigma_1,$$
$$c_1^1 = \sigma_1,$$

i.e. $2c_1^1 + c_0^1 = 1$. Let us assume that the equality holds for the first $k-1$ intermediate steps; we shall show it is valid for the k^{th}. Using again (38) we obtain:

$$\begin{aligned}
c_0^k + 2c_1^k + \ldots + 2c_k^k &= (1-2\sigma_k)c_0^{k-1} + 2\sigma_k c_1^{k-1} + 2\sigma_k c_{k-1}^{k-1} + 2\sigma_k \sum_{l=1}^{k-1}(c_{l-1}^{k-1} + c_{l+1}^{k-1}) \\
&\quad + 2(1-2\sigma_k)\sum_{l=1}^{k-1} c_l^{k-1} \\
&= (1-2\sigma_k)c_0^{k-1} + 2\sigma_k c_1^{k-1} + 2\sigma_k c_{k-1}^{k-1} + 2\sigma_k \sum_{l=0}^{k-2} c_l^{k-1} + 2\sigma_k \sum_{l=2}^{k} c_l^{k-1} \\
&\quad + 2(1-2\sigma_k)\sum_{l=1}^{k-1} c_l^{k-1} \\
&= (1-2\sigma_k)c_0^{k-1} + 2\sigma_k \sum_{l=1}^{k-1} c_l^{k-1} + 2\sigma_k c_0^{k-1} + 2\sigma_k \sum_{l=1}^{k-1} c_l^{k-1} + 2\sigma_k c_k^{k-1} + 2\sum_{l=1}^{k-1} c_l^{k-1} \\
&\quad - 4\sigma_k \sum_{l=1}^{k-1} c_l^{k-1} \\
&= c_0^{k-1} + 2\sum_{l=1}^{k-1} c_l^{k-1} + 2\sigma_k c_k^{k-1} = c_0^{k-1} + 2\sum_{l=1}^{k-1} c_l^{k-1} = 1,
\end{aligned}$$

where we have used that since c_l^{k-1}, $l = 0, \ldots, k-1$ are the coefficients of level $k-1$, then $c_l^{k-1} = 0$ for $l \geq k$.

Proposition 2. *At each time level t^k, we have:*

$$k^2 c_k^k + (k-1)^2 c_{k-1}^k + \ldots + c_1^k = \frac{D(\tau_1 + \tau_2 + \ldots + \tau_k)}{h^2},\ k = 1, \ldots, K$$

In particular, for the end sub-level $k = K$, this sum is exactly $\dfrac{D\Delta T}{h^2}$ (since a super-step ΔT consists of K sub-steps).

Proof: The proof of Proposition 2 is analogous to that of Proposition 1. In order to verify it, we shall again use mathematical induction, adopting the notation we introduced above. Let $k = 1$, then we obtain the following result for the coefficients of (36):

$$c_1^1 = \sigma_1 = \frac{D\tau_1}{h^2}$$

We assume the equality holds for $k-1$ intermediate steps and using (38) we shall verify it for the k^{th} step:

$$k^2c_k^k+(k-1)^2c_{k-1}^k+\ldots+c_1^k=\sigma_k k^2c_{k-1}^{k-1}+\sigma_k\sum_{l=1}^{k-1}l^2(c_{l-1}^{k-1}+c_{l+1}^{k-1})$$
$$+(1-2\sigma_k)\sum_{l=1}^{k-1}l^2c_l^{k-1}$$
$$=\sigma_k\Big[1^2c_0^{k-1}+2^2c_1^{k-1}+(1^2+3^2)c_2^{k-1}+\ldots+\Big((k-3)^2+(k-1)^2\Big)c_{k-2}^{k-1}$$
$$+\Big((k-2)^2+k^2\Big)c_{k-1}^{k-1}\Big]+(1-2\sigma_k)\sum_{l=1}^{k-1}l^2c_l^{k-1}$$
$$=2\sigma_k\left[c_1^{k-1}+2^2c_2^{k-1}+3^2c_3^{k-1}+\ldots+(k-2)^2c_{k-2}^{k-1}+(k-1)^2c_{k-1}^{k-1}\right]$$
$$+\sigma_k\left[c_0^{k-1}+2c_1^{k-1}+2c_2^{k-1}+2c_3^{k-1}+\ldots+2c_{k-2}^{k-1}+2c_{k-1}^{k-1}\right]+(1-2\sigma_k)\sum_{l=1}^{k-1}l^2c_l^{k-1}$$

Using the result obtained in Proposition 1, we have:

$$k^2c_k^k+(k-1)^2c_{k-1}^k+\ldots+c_1^k=\sigma_k+\sum_{l=1}^{k-1}l^2c_l^{k-1}=\sigma+\frac{D(\tau_1+\tau_2+...+\tau_{k-1})}{h^2}=$$
$$\frac{D(\tau_1+\tau_2+...+\tau_k)}{h^2}=\frac{D\Delta T}{h^2},$$

and thus, we conclude the proof.

Remark: We underline that everywhere in the article, by c_l, $l=0,\ldots,K$ we have denoted the coefficients of the last level (the K^{th}) of each super-step and that's why we have skipped their upper index.

Substituting these results into the error equation (37), we readily obtain:

(40)
$$\hat{\varepsilon}_i^{j+1}=\int_0^{\Delta T}(\Delta T-s)\frac{\partial^2u}{\partial\xi^2}(a^j+s,t^n+s,x_i)ds+c_K(\varepsilon_{i+K}^j+\varepsilon_{i-K}^j)$$
$$+c_{K-1}(\varepsilon_{i+K-1}^j+\varepsilon_{i-K+1}^j)+\ldots+c_i(\varepsilon_{2i}^j+\varepsilon_0^j)+\ldots$$
$$c_1(\varepsilon_{i-1}^j+\varepsilon_{i+1}^j)+c_0(\varepsilon_i^j)+O(h^3)$$

We take the absolute values of (40) to deduce that:

(41)
$$|\hat{\varepsilon}_i^{j+1}|\le\int_0^{\Delta T}|(\Delta T-s)\frac{\partial^2u}{\partial\xi^2}(a^j+s,t^n+s,x_i)|ds+|c_K||\varepsilon_{i+K}^j+\varepsilon_{i-K}^j|$$
$$+|c_{K-1}||\varepsilon_{i+K-1}^j+\varepsilon_{i-K+1}^j|+\ldots+|c_i||\varepsilon_{2i}^j+\varepsilon_0^j|+\ldots+|c_1||\varepsilon_{i-1}^j+\varepsilon_{i+1}^j|$$
$$+|c_0||\varepsilon_i^j|+O(h^3)$$
$$\le\sup_{s\in[0,\Delta T]}\left\{|\frac{\partial^2u}{\partial\xi^2}(a^j+s,t^n+s,x_i)|\right\}\frac{\Delta T^2}{2}+|c_K||\varepsilon_{i+K}^j+\varepsilon_{i-K}^j|$$
$$+|c_{K-1}||\varepsilon_{i+K-1}^j+\varepsilon_{i-K+1}^j|+\ldots+|c_i||\varepsilon_{2i}^j+\varepsilon_0^j|+\ldots+|c_1||\varepsilon_{i-1}^j+\varepsilon_{i+1}^j|$$
$$+|c_0||\varepsilon_i^j|+O(h^3)$$
$$\le C_1\Delta T^2+\max_{1\le i\le M-1}|\varepsilon_i^j|(|\underbrace{2c_K+\ldots+2c_1+c_0}_{1}|)+O(h^3)$$
$$=C_1\Delta T^2+\max_{1\le i\le M-1}|\varepsilon_i^j|+O(h^3)$$

Concerning $n = 0$ we see that:

$$\varepsilon_i^j = u_0(a^j, x_i) - u_0(a^j, x_i) = 0, \tag{42}$$

while for the approximation of the newborn ($j = 0$) we have:

$$\hat{\varepsilon}_i^0 = \int_0^{a_+} \tilde{\beta}(a^{j+1}) u(a^{j+1}, t^{n+1}, x_i) da - \Big(\Delta T \sum_{j=0}^{N_s-1} \tilde{\beta}_{j+1} \hat{U}_i^{j+1} + \frac{\Delta T}{2} \tilde{\beta}_0 \hat{U}_i^0\Big) \tag{43}$$

Using the standard error estimate for the trapezoidal rule, an immediate consequence is

$$|\hat{\varepsilon}_i^0| \le \Delta T^2 ||\tilde{\beta}||_{l^\infty} C_2 \Big(||\frac{\partial u}{\partial t}||_{l^\infty} + ||\frac{\partial u}{\partial a}||_{l^\infty}\Big), \tag{44}$$

where C_2 does not depend neither on ΔT nor on u.
From (41) and (44) it follows that

$$\max_{1\le i\le M-1} |\hat{\varepsilon}_i^{j+1}| \le \max_{1\le i\le M-1} |\varepsilon_i^j| + C_1 \Delta T^2 + O(h^3), \tag{45}$$

and

$$\max_{1\le i\le M-1} |\hat{\varepsilon}_i^0| \le C_2 \Delta T^2 \tag{46}$$

Consequently, multiplying (45) and (46) by ΔT and summing on j, $j = 0, ..., N_s$, we obtain

$$||\hat{\varepsilon}||_{l^1}^{l^\infty} \le ||\varepsilon||_{l^1}^{l^\infty} + C^* \Delta T^2, \tag{47}$$

where C^* depends on $\Big(||\frac{\partial u}{\partial t}||_{L^\infty}, ||\frac{\partial u}{\partial a}||_{L^\infty}, ||\frac{\partial^2 u}{\partial \xi^2}||_{L^\infty}, ||\tilde{\beta}||_{L^\infty}, T \text{ and } a_+\Big)$. Substituting this relation into itself n times and using (42), we see that

$$||\varepsilon||_{l^1}^{l^\infty} \le C^* \Delta T, \tag{48}$$

which is exactly part a) of Theorem 1.
In order to derive the $||.||_{l^\infty}^{l^\infty}$ estimate, we use (45) and (46) to see that:

$$\max_{0\le j\le N_s} \max_{1\le i\le M-1} |\hat{\varepsilon}_i^j| \le \max_{0\le j\le N_s} \max_{1\le i\le M-1} |\varepsilon_i^j| + C_3 \Delta T^2$$

and then using the same iterative procedure as in case a), we obtain the second part of the proof:

$$||\varepsilon||_{l^\infty}^{l^\infty} \le C^{**} \Delta T \tag{49}$$

2) $\mathbf{K > i}$ and $\mathbf{K < 2i}$
In this case, it can easily be checked that we can present the modified STS scheme at time level t^{n+1} in the following form:

$$\begin{aligned} \hat{U}_i^{j+1} = {} & c_K(U_{i+K}^j - U_{i-(2i-K)}^j) + c_{K-1}(U_{i+(K-1)}^j - U_{i-(2i-K+1)}^j) + \dots \\ & c_{i+1}(U_{i+(i+1)}^j - U_{i-(i-1)}^j) + c_i(U_{i+i}^j - U_{i-i}^j) + c_{i-1}(U_{i+(i-1)}^j + U_{i-(i-1)}^j) + \dots \\ & c_1(U_{i+1}^j + U_{i-1}^j) + c_0(U_i^j),\ i = 1, \dots, M-1, j = 0, \dots, N_s - 1, \end{aligned} \tag{50}$$

where c_l, $l = 0, \ldots, K$ are the same coefficients as in case 1. Subtracting (50) from (32) we obtain, for $i = 1, \ldots, M-1, j = 0, \ldots, N_s - 1, 0 \leq n \leq N_t - 1$, the following error equation

(51)
$$\begin{aligned}\hat{\varepsilon}_i^{j+1} = u_i^j + D\Delta T\frac{\partial^2 u_i^j}{\partial x_i^2} + I - c_K(U_{i+K}^j - U_{i-(2i-K)}^j)\\ -c_{K-1}(U_{i+(K-1)}^j - U_{i-(2i-K+1)}^j) - \ldots - c_{i+1}(U_{i+(i+1)}^j - U_{i-(i-1)}^j)\\ -c_i(U_{i+i}^j - U_{i-i}^j) - \ldots - c_1(U_{i+1}^j + U_{i-1}^j) - c_0 U_i^j,\end{aligned}$$

where we have used the notation $I = \displaystyle\int_0^{\Delta T}(\Delta T - s)\frac{\partial^2 u}{\partial \xi^2}(a^j + s, t^n + s, x_i)ds$. We add and subtract the exact solution in mesh points $(a^j, t^n, x_{i+K}), (a^j, t^n, x_{K-i})$, $(a^j, t^n, x_{i+K-1}), \ldots, (a^j, t^n, x_0)$, multiplied by proper coefficients and thus, we obtain

$$\hat{\varepsilon}_i^{j+1} = \varepsilon_1 - \varepsilon_2, \tag{52}$$

where

(53)
$$\begin{aligned}\varepsilon_1 = u_i^j + D\Delta T\frac{\partial^2 u_i^j}{\partial x_i^2} + I + c_K(\varepsilon_{i+K}^j - \varepsilon_{i-(2i-K)}^j) + c_{K-1}(\varepsilon_{i+(K-1)}^j - \varepsilon_{i-(2i-K+1)}^j)\\ + \ldots + c_{i+1}(\varepsilon_{i+(i+1)}^j - \varepsilon_{i-(i-1)}^j) + c_i(\varepsilon_{i+i}^j - \varepsilon_{i-i}^j) + \ldots + c_1(\varepsilon_{i+1}^j + \varepsilon_{i-1}^j) + c_0\varepsilon_i^j\end{aligned}$$

(54)
$$\begin{aligned}\varepsilon_2 = c_K(u_{i+K}^j - u_{i-(2i-K)}^j) + c_{K-1}(u_{i+(K-1)}^j - u_{i-(2i-K+1)}^j) + \ldots\\ +c_{i+1}(u_{i+(i+1)}^j - u_{i-(i-1)}^j) + c_i(u_{i+i}^j - u_{i-i}^j) + \ldots + c_1(u_{i+1}^j + u_{i-1}^j) + c_0 u_i^j\end{aligned}$$

Proceeding as in the previous case, i.e. using the Tailor expansions and simplifying afterwards, we readily obtain:

$$\begin{aligned}\varepsilon_2 = u_i^j\left[c_0 + 2c_1 + \ldots + 2c_i\right] + 2ih\frac{\partial u_i^j}{\partial x_i}\left[c_{i+1} + c_{i+2} + \ldots + c_K\right] +\\ h^2\frac{\partial^2 u_i^j}{\partial x_i^2}\left[c_1 + 2^2c_2 + \ldots + i^2c_i\right] + 2ih^2\frac{\partial^2 u_i^j}{\partial x_i^2}\left[1c_{i+1} + 2c_{i+2} + \ldots + (K-i)c_K\right] +\\ \frac{2ih^3}{3!}\frac{\partial^3 u_i^j}{\partial x_i^3}\left[(i^2 + 3.1^2)c_{i+1} + (i^2 + 3.2^2)c_{i+2} + \ldots + (i^2 + 3.(K-i)^2)c_K\right] + O(h^4)\end{aligned}$$

We add and subtract the expression $2u_i^j[c_{i+1} + c_{i+2} + \ldots + c_K]$ and using the fact that the solution of (11) (for Dirichlet boundary conditions) is identically 0 at the boundary points $x = 0$ and $x = 1$, we only add $2u_0^j[c_{i+1} + c_{i+2} + \ldots + c_K]$.

We also use the following formulas:

$$\begin{aligned}2u_0^j[c_{i+1} + c_{i+2} + \ldots + c_K] =\\ 2[c_{i+1} + c_{i+2} + \ldots + c_K]\left[u_i^j - ih\frac{\partial u_i^j}{\partial x_i} + \frac{i^2h^2}{2!}\frac{\partial^2 u_i^j}{\partial x_i^2} - \frac{i^3h^3}{3!}\frac{\partial^3 u_i^j}{\partial x_i^3} + O(h^4)\right]\end{aligned}$$

$$u_i^j[c_0 + 2c_1 + \ldots + 2c_i] + 2u_i^j[c_{i+1} + c_{i+2} + \ldots + c_K] = u_i^j \quad \text{(see Proposition 1)}$$

Thus, we arrive to the following result:

$$\begin{aligned}
\varepsilon_2 &= u_i^j - 2u_i^j[c_{i+1} + c_{i+2} + \ldots + c_K] \\
&+2[c_{i+1} + c_{i+2} + \ldots + c_K]\left[u_i^j - ih\frac{\partial u_i^j}{\partial x_i} + \frac{i^2h^2}{2!}\frac{\partial^2 u_i^j}{\partial x_i^2} - \frac{i^3h^3}{3!}\frac{\partial^3 u_i^j}{\partial x_i^3}\right] \\
&+2ih\frac{\partial u_i^j}{\partial x_i}\left[c_{i+1} + c_{i+2} + \ldots + c_K\right] + h^2\frac{\partial^2 u_i^j}{\partial x_i^2}\left[c_1 + 2^2c_2 + \ldots + i^2c_i\right] \\
&+2ih^2\frac{\partial^2 u_i^j}{\partial x_i^2}\left[1c_{i+1} + 2c_{i+2} + \ldots + (K-i)c_K\right] \\
&+\frac{ih^3}{3}\frac{\partial^3 u_i^j}{\partial x_i^3}\left[(i^2 + 3.1^2)c_{i+1} + (i^2 + 3.2^2)c_{i+2} + \ldots + (i^2 + 3.(K-i)^2)c_K\right] + O(h^4) \\
&= u_i^j + h^2\frac{\partial^2 u_i^j}{\partial x_i^2}\left[c_1 + 2^2c_2 + \ldots + i^2c_i\right] \\
&+h^2\frac{\partial^2 u_i^j}{\partial x_i^2}\left[(i+1)^2c_{i+1} + (i+2)^2c_{i+2} + \ldots + K^2c_k\right] \\
&-h^2\frac{\partial^2 u_i^j}{\partial x_i^2}\left[1^2c_{i+1} + 2^2c_{i+2} + \ldots + (K-i)^2c_K\right] \\
&+ih^3\frac{\partial^3 u_i^j}{\partial x_i^3}\left[1^2c_{i+1} + 2^2c_{i+2} + \ldots + (K-i)^2c_K\right] + O(h^4)
\end{aligned}$$

Using the result obtained in Proposition 2 and the fact that since $u_0^j = 0$, it follows $\dfrac{\partial^2 u_0^j}{\partial x_0^2} = 0$, we get:

$$\begin{aligned}
\varepsilon_2 &= u_i^j + D\Delta T\frac{\partial^2 u_i^j}{\partial x_i^2} - h^2\left[c_{i+1} + 2^2c_{i+2} + ...(K-i)^2c_K\right]\left[\frac{\partial^2 u_0^j}{\partial x_0^2} + ih\frac{\partial^3 u_0^j}{\partial x_0^3} + O(h^2)\right] \\
&+ih^3\left[1^2c_{i+1} + 2^2c_{i+2} + \ldots + (K-i)^2c_K\right]\left[\frac{\partial^3 u_0^j}{\partial x_0^3} + O(h)\right] + O(h^4) \\
&= u_i^j + D\Delta T\frac{\partial^2 u_i^j}{\partial x_i^2} + O(h^4)
\end{aligned}$$

Combining this result with (53), we yield:

(55)

$$\begin{aligned}
\hat{\varepsilon}_i^{j+1} &= \varepsilon_1 - \varepsilon_2 = u_i^j + D\Delta T\frac{\partial^2 u_i^j}{\partial x_i^2} + I + c_K(\varepsilon_{i+K}^j - \varepsilon_{i-(2i-K)}^j) \\
&+c_{K-1}(\varepsilon_{i+(K-1)}^j - \varepsilon_{i-(2i-K+1)}^j) + \ldots + c_{i+1}(\varepsilon_{i+(i+1)}^j - \varepsilon_{i-(i-1)}^j) \\
&+c_i(\varepsilon_{i+i}^j - \varepsilon_{i-i}^j) + \ldots + c_1(\varepsilon_{i+1}^j + \varepsilon_{i-1}^j) + c_0\varepsilon_i^j - [u_i^j + D\Delta T\frac{\partial^2 u_i^j}{\partial x_i^2} + O(h^4)] \\
&= \int_0^{\Delta T}(\Delta T - s)\frac{\partial^2 u}{\partial \xi^2}(a^j + s, t^n + s, x_i)ds + c_K(\varepsilon_{i+K}^j - \varepsilon_{i-(2i-K)}^j) \\
&+c_{K-1}(\varepsilon_{i+(K-1)}^j - \varepsilon_{i-(2i-K+1)}^j) + \ldots + c_{i+1}(\varepsilon_{i+(i+1)}^j - \varepsilon_{i-(i-1)}^j) \\
&+c_i(\varepsilon_{i+i}^j - \varepsilon_{i-i}^j) + \ldots + c_1(\varepsilon_{i+1}^j + \varepsilon_{i-1}^j) + c_0\varepsilon_i^j + O(h^4)
\end{aligned}$$

We can take the absolute values in (55), yielding:

$$\begin{aligned}|\hat{\varepsilon}_i^{j+1}| = |\int_0^{\Delta T}(\Delta T - s)\frac{\partial^2 u}{\partial \xi^2}(a^j + s, t^n + s, x_i)ds| + |c_K(\varepsilon^j_{i+K} - \varepsilon^j_{i-(2i-K)})| \\ +|c_{K-1}(\varepsilon^j_{i+(K-1)} - \varepsilon^j_{i-(2i-K+1)})| + \ldots + |c_{i+1}(\varepsilon^j_{i+(i+1)} - \varepsilon^j_{i-(i-1)})| \\ +|c_i(\varepsilon^j_{i+i} - \varepsilon^j_{i-i})| + \ldots + |c_1(\varepsilon^j_{i+1} + \varepsilon^j_{i-1})| + |c_0\varepsilon^j_i| + O(h^4) \\ \leq |\int_0^{\Delta T}(\Delta T - s)\frac{\partial^2 u}{\partial \xi^2}(a^j + s, t^n + s, x_i)ds| + |c_K(\varepsilon^j_{i+K} + \varepsilon^j_{i-(2i-K)})| \\ +|c_{K-1}(\varepsilon^j_{i+(K-1)} + \varepsilon^j_{i-(2i-K+1)})| + \ldots + |c_{i+1}(\varepsilon^j_{i+(i+1)} + \varepsilon^j_{i-(i-1)})| \\ +|c_i(\varepsilon^j_{i+i} + \varepsilon^j_{i-i})| + \ldots + |c_1(\varepsilon^j_{i+1} + \varepsilon^j_{i-1})| + |c_0\varepsilon^j_i| + O(h^4)\end{aligned}$$

Since the rest of the proof is identical to what we did in case 1), we are not going to demonstrate it. The case ($\mathbf{K > i}$ and $\mathbf{K > 2i}$) is analogous to ($\mathbf{K > i}$ and $\mathbf{K < 2i}$) and we leave it to the reader.

6. Performance on linear and nonlinear test problems.

In this section we investigate the performance of the modified super-time-stepping scheme on three exactly solvable test problems. Relying on the stability results given in Section 5, we believe the algorithm will work well also in the case of Neumann boundary conditions and we demonstrate this empirically. We choose the parameter ν as a random number in the interval $(0, 1)$ and we perform a large number of experiments. In each case we compare modified STS with schemes of the same or higher order, namely the explicit, fully implicit and Crank-Nicolson standard schemes, in terms of efficiency and approximation. For the implementation of the implicit schemes we use either direct or iterative methods for solving the resulting system. Since SOR iterations and Thomas' algorithm showed best results, we list comparisons only with them. In the modified super-time-stepping scheme we vary the parameter ν and the number of sub-steps we do, while in the implicit schemes we increase the number of steps in time. Moreover, when applying SOR iterations we report the value of ω (found by trial) which gives the best approximation and we use a tolerance of 10^{-6} for convergence.

6.1 Problem 1 - linear models

We consider a population with a finite age and for simplicity we take the maximum age of the individuals $a_+ = 1$ [19]. The mortality and the survival probability are $\mu(a) = \dfrac{1}{1-a}$, $\pi(a) = 1 - a$ respectively. We choose the following initial conditions:

$$p_0(a, x) = e^{-\alpha^* a}(1 - a)\sin(\pi x), \text{ for Dirichlet boundary conditions} \tag{56}$$

$$p_0(a, x) = e^{-\alpha^* a}(1 - a)(2 + \cos(\pi x)), \text{ for Neumann boundary conditions} \tag{57}$$

where α^* is the intrinsic Malthusian parameter which determines the population growth via the birth rate $B(t, x)$. We assume the fertility $\beta(a) = \beta$ and by choosing an appropriate value of $\alpha^* = 2$, we calculate it by formula (4), which provides continuity of the solution $p(a, t, x)$. The solution of system (1) is given by

$$p(a, t, x) = e^{\alpha^*(t-a)}(1 - a)e^{-\pi^2 Dt}\sin(\pi x), \tag{58}$$

and the one of problem (2)

$$p(a,t,x) = e^{\alpha^*(t-a)}(1-a)[2+e^{-\pi^2 Dt}\cos(\pi x)], \tag{59}$$

We assume the diffusion constant $D = 1$ for simplicity. In order to satisfy the CFL condition $\tau \le \dfrac{h^2}{2D}$ for the explicit scheme, we choose $\tau = 0.00125$ and $h = 0.05$. In the other schemes we fix $M = 20$ and we vary the number of the steps in time. All calculations in the first three tables are done for $T = 3$ (since the solution (58) of (1) does not grow in time). In case of Neumann conditions on the boundary, we have exponential growth of the solution and we give results only for $T = 1$.

In the tables below we use the following notations:
$\mathbf{t_{steps}}$ - total number of steps in time; $\mathbf{a_{steps}}$ - total number of steps in age;
$\mathbf{N}_{\int}$ - number of calculations of the birth integral; $\mathbf{N_{STS}}$ - total number of super-steps;
K - number of intermediate steps per one super-step; **iter** - number of iterations;
ω- factor, used in SOR iterations; **CPU** - time (in seconds), needed for computations;
$\mathbf{E_{abs}}$ - the max L^∞ error; $\mathbf{E_{L^1}}$ - the max L^1 error; $\mathbf{E_{rel}}$ - the max relative error;
IS - pure implicit scheme; **CN** - Crank-Nicolson scheme

a) Drichlet boundary conditions: $T = 3$

Table 1: MODIFIED SUPER - TIME - STEPPING

ν	N_{STS}	K	t_{steps}	a_{steps}	$N_{\int}$	E_{abs}	E_{L^1}	CPU
0.00	2400	1	2400	36480000	45600	3.069E-3	6.102E-4	24.18
0.0003	66	6	396	159258	1254	1.163E-1	2.361E-2	0.07
0.0006	150	4	600	561450	2850	4.933E-2	9.898E-3	0.24
0.001	99	5	495	302841	1881	7.628E-2	1.540E-2	0.14
0.001	27	10	270	41553	513	3.817E-1	7.937E-2	0.01
0.004	279	3	837	1468377	5301	2.615E-2	5.227E-3	0.48
0.006	168	4	672	705432	3192	4.237E-2	8.502E-3	0.31
0.006	114	5	570	402876	2166	6.149E-2	1.240E-2	0.19
0.009	177	4	708	783579	3363	3.919E-2	7.861E-3	0.28
0.02	150	5	750	701100	2850	3.882E-2	7.811E-3	0.30
0.07	126	10	1260	983934	2394	1.502E-2	3.133E-3	0.25
0.07	48	25	1200	342912	912	8.339E-3	2.673E-3	0.08
0.095	48	30	1440	411312	912	5.810E-3	1.874E-3	0.09
0.20	69	30	2070	866571	1311	2.657E-3	8.412E-4	0.18

Table 2a: IS + THOMAS' ALGORITHM

t_{steps}	$N_{\int}$	a_{steps}	E_{abs}	E_{L^1}	CPU
450	8550	1282500	2.594E-2	5.142E-3	1.09
600	11400	2280000	1.986E-2	3.939E-3	1.76
1200	22800	9120000	1.072E-2	2.127E-3	6.85
1800	34200	20520000	7.659E-3	1.521E-3	15.12
2400	45600	36480000	6.130E-3	1.217E-3	26.37

Table 2b: IS + SOR ITERATIONS							
t_{steps}	$N_{\int}$	a_{steps}	ω	iter	E_{abs}	E_{L^1}	CPU
450	8550	1282500	1.57	597	2.460E-2	4.887E-3	1.11
600	11400	2280000	1.46	778	2.420E-2	4.808E-3	1.88
1200	22800	9120000	1.35	1389	1.905E-2	3.784E-3	7.25
1800	34200	20520000	1.27	2030	1.491E-2	2.962E-3	15.87
2400	45600	36480000	1.22	2641	1.417E-2	2.904E-3	26.55

Table 3a: CN + THOMAS' ALGORITHM					
t_{steps}	$N_{\int}$	a_{steps}	E_{abs}	E_{L^1}	CPU
300	5700	570000	1.197E-3	2.118E-4	0.61
450	8550	1282500	1.384E-3	2.634E-4	1.24
600	11400	2280000	1.450E-3	2.815E-4	2.01
1200	22800	9120000	1.514E-3	2.991E-4	6.83
1800	34200	20520000	1.526E-3	3.023E-4	15.43

Table 3b: CN + SOR ITERATIONS							
t_{steps}	$N_{\int}$	a_{steps}	ω	iter	E_{abs}	E_{L^1}	CPU
300	5700	570000	1.1	667	3.623E-2	5.203E-3	1.05
450	8550	1282500	1.17	913	2.325E-2	4.620E-3	1.26
600	11400	2280000	1.14	1109	1.071E-2	2.127E-3	2.04
1200	22800	9120000	1.36	1687	1.702E-3	3.382E-4	7.32
1800	34200	20520000	1.22	2327	1.085E-3	2.155E-4	16.28

b) Neumann boundary conditions: $T = 1$

Table 4: MODIFIED SUPER - TIME - STEPPING									
ν	N_{STS}	K	t_{steps}	a_{steps}	$N_{\int}$	E_{abs}	E_{L^1}	E_{rel}	CPU
0.00	800	1	800	13440000	16800	1.116E-2	1.853E-3	9.659E-4	8.54
0.0001	22	6	132	58674	462	1.220E-1	2.630E-2	4.122E-2	0.03
0.0005	50	4	200	206850	1050	5.037E-2	1.001E-2	1.635E-2	0.09
0.0009	32	5	160	104832	672	7.869E-2	1.564E-2	2.582E-2	0.05
0.006	38	5	190	148428	798	6.307E-2	1.254E-2	2.068E-2	0.07
0.008	58	4	232	278922	1218	4.091E-2	8.131E-3	1.314E-2	0.15
0.03	58	5	290	348348	1218	3.090E-2	6.138E-3	9.827E-3	0.15
0.08	56	8	448	518616	1176	1.423E-2	2.827E-3	4.469E-3	0.17
0.08	75	6	450	700875	9450	4.792E-2	1.007E-2	9.783E-3	0.24
0.1	50	10	500	515550	1050	1.129E-2	2.243E-3	3.545E-3	0.16
0.22	34	22	748	519078	714	1.148E-2	1.917E-3	1.521E-3	0.13

Table 5a: IS + THOMAS' ALGORITHM						
t_{steps}	$N_{\int}$	a_{steps}	E_{abs}	E_{L^1}	E_{rel}	CPU
150	3150	472500	3.112E-1	5.144E-2	7.861E-3	0.47
200	4200	840000	1.756E-1	2.906E-2	6.056E-3	0.72
400	8400	3360000	4.454E-2	7.391E-3	3.298E-3	2.76
600	12600	7560000	2.009E-2	3.351E-3	2.365E-3	6.17
800	16800	13440000	1.149E-2	1.933E-3	1.896E-3	10.82

Table 5b: IS + SOR ITERATIONS

t_{steps}	$N_{\int}$	a_{steps}	ω	iter	E_{abs}	E_{L^1}	E_{rel}	CPU
150	3150	472500	1.5	1492	2.346E-1	3.635E-2	7.833E-3	0.92
200	4200	840000	1.2	2838	1.664E-1	3.889E-2	6.276E-3	1.75
400	8400	3360000	1.36	2533	1.545E-2	2.405E-3	3.247E-3	3.96
600	12600	7560000	1.3	3096	1.572E-2	3.771E-3	2.275E-3	8.41
800	16800	13440000	1.6	6258	1.004E-2	1.839E-3	2.101E-3	15.81

Table 6a: CN + THOMAS' ALGORITHM

t_{steps}	$N_{\int}$	a_{steps}	E_{abs}	E_{L^1}	E_{rel}	CPU
100	2100	210000	1.314E-3	1.849E-4	2.892E-4	0.33
150	3150	472500	1.267E-3	2.516E-4	3.935E-4	0.54
200	4200	840000	1.384E-3	2.749E-4	4.299E-4	0.79
400	8400	3360000	1.497E-3	2.974E-4	4.652E-4	2.92
600	12600	7560000	1.518E-3	3.016E-4	4.717E-4	6.54

Table 6b: CN + SOR ITERATIONS

t_{steps}	$N_{\int}$	a_{steps}	ω	iter	E_{abs}	E_{L^1}	E_{rel}	CPU
100	2100	210000	1.57	820	3.302E-2	6.552E-3	6.944E-3	0.44
150	3150	472500	1.5	914	1.732E-2	3.442E-3	1.101E-3	0.65
200	4200	840000	1.45	1089	7.741E-3	1.154E-3	7.279E-4	1.12
400	8400	3360000	1.48	2320	3.719E-3	7.388E-4	4.835E-4	3.98
600	12600	7560000	1.15	3403	3.167E-3	7.278E-4	4.720E-4	8.84

From the data reported in the first table, we can see that the smaller the ν is, the larger the errors are, but the computations are fast performed. One way to improve accuracy is by increasing the damping factor. Thus the duration ΔT of the super-step decreases and the errors we get are with better accuracy, but the cost goes up. One can see that when decreasing the number of intermediate steps we also obtain better accuracy (compare results for $\nu = 0.0003$, $K = 6$ and $\nu = 0.0006$, $K = 4$ or $\nu = 0.001$, $K = 10$ and $\nu = 0.001$, $K = 5$ in Table 1), but computational time increases sensitively. Similar behavior can be observed in Table 4.

Since the upper bound of the parameter ν in case of Dirichlet b.c. is approximately 0.006 ($\frac{\lambda_{min}}{\lambda_{max}} \approx 0.00619$), we increased the damping factor to 0.004 and 0.006 and we obtained an error comparable to the error given by the fully implicit scheme + Thomas' algorithm and to the one of the fully implicit scheme + SOR, but modified STS appeared to be much more effective (see Tables 1, 2a, 2b, 3a and 3b). Furthermore, relying on the fact that in practice computations work far beyond the theoretical limits, we tried the modified STS algorithm with larger values of ν and the results show that it behaves extremely well. It follows that the analytical restriction $\nu \in (0, \frac{\lambda_{min}}{\lambda_{max}}]$ is too strong and we can choose the values of ν in the interval (0,1) randomly. Doing so, we could decrease the number of super-steps N_{STS} and consequently the number of computations of the boundary condition $N_{\int}$ too. Thus, for $\nu \approx 0.2$ (also in case of Neumann b.c., Table 4) we obtained accuracy comparable to the accuracy of the explicit scheme (note that for $\nu = 0$ and $K = 1$, we have exactly the explicit scheme itself) and to that one of the second order Crank-Nicolson scheme (see Tables 3a and 3b). Regarding the CPU time, one can see that the speed up given by the modified STS is enormous.

In case of Neumann boundary conditions, the excessive restrictiveness of the theoretical condition imposed on ν is even more obvious (since $\lambda_{min} = 0$) and it is proved empirically (see Table 11) that the scheme works for $\nu \in (0, 1)$. It implies, we can also apply the modified STS to problems (for example nonlinear problems) where the eigenvalues of the matrix A in (26) are not known. This fact is confirmed by our experiments done for the nonlinear model (6) and listed below.

The results obtained in case of Neumann b.c. are similar to these for Dirichlet b.c. In addition to the absolute and L^1 errors, we show also the relative error. It seems that in both cases for smaller values of ν we achieve best accuracy for smaller number of inner steps, which corresponds to number of super-steps between 30 and 60 per unit time. By increasing ν, we yield better results as we increased the number of the inner steps as well (compare error values for $\nu = 0.07$, $K = 10$ and $K = 25$ in Table 1 and $\nu = 0.08$, $K = 6$ or $K = 8$ in Table 4). Another interesting observation is that while the implicit scheme combined with Thomas' algorithm is as exact and efficient as the implicit scheme combined with SOR iterations in case of Dirichlet b.c. (Tables 2a and 2b), in the other case the implicit scheme with SOR iterations, appears to be much slower with respect to the implicit scheme with Thomas' algorithm (see Tables 5a and 5b). Concerning the Crank-Nicolson scheme in both cases its combination with Thomas' algorithm is better (as error values and time consumption) than the combination with SOR iterations (compare results from Tables 3a, 3b, 6a and 6b). Moreover, in case of Neumann b.c. SOR iterations are more expensive than Thomas' algorithm (Tables 5a and 5b; 6a and 6b).

6.2 Problem 2 - nonlinear model

Here we consider the same values for the fertility and mortality as in the linear case and our initial conditions are the following

$$w_0(a, x) = \frac{\beta e^{-\alpha^* a}(1 - a)(2 + \cos(\pi x))}{2}, \tag{60}$$

where we assume α^* is again equal to 2.
The solution of system (6) is given by

$$w(a, t, x) = \frac{\beta e^{\alpha^*(t-a)}(1 - a)(2 + e^{-\pi^2 Dt}\cos(\pi x))}{2e^{\alpha^* t}}, \tag{61}$$

where the exact value of the total population (7) is:

$$P(t) = \frac{2e^{\alpha^* t}}{\beta} \tag{62}$$

In order to obtain stability for the explicit scheme we keep the values of N and M as in the previous case, i.e. $N = 800$ and $M = 20$ and we use the same notations. We underline that since the solution of (6) is bounded, it implies the errors remain stable in time and we present results only for $T = 1$.

Table 7: MODIFIED SUPER - TIME - STEPPING: NONLINEAR CASE

ν	N_{STS}	K	t_{steps}	a_{steps}	$N_{\int}$	E_{abs}	E_{L^1}	E_{rel}	CPU
0.00	800	1	800	13440000	16800	2.239E-2	4.457E-3	3.314E-3	10.13
0.0005	50	4	200	206850	462	2.400E-1	4.963E-2	3.608E-2	0.14
0.004	26	6	156	82446	546	4.370E-1	9.401E-2	6.754E-2	0.05
0.004	54	4	216	241542	1134	2.143E-1	4.432E-2	3.227E-2	0.16
0.008	58	4	232	278922	1218	1.909E-1	3.941E-2	2.866E-2	0.19
0.03	58	5	290	348348	1218	1.285E-1	2.700E-2	1.960E-2	0.18
0.08	64	7	448	594048	1344	5.173E-2	9.985E-3	1.047E-2	0.30
0.1	50	10	500	515550	1050	5.094E-2	8.819E-3	9.907E-3	0.22
0.20	59	12	708	863583	1239	3.048E-2	5.107E-3	5.759E-3	0.28
0.22	57	13	741	872613	1197	2.969E-2	4.876E-3	5.544E-3	0.26
0.24	55	14	770	874335	1155	2.916E-2	4.705E-3	5.382E-3	0.24

Table 8a: IS + THOMAS' ALGORITHM: NONLINEAR CASE

t_{steps}	$N_{\int}$	a_{steps}	iter	E_{abs}	E_{L^1}	E_{rel}	CPU
150	3150	472500	309	1.321E-1	2.602E-2	1.938E-2	0.65
200	4200	840000	396	9.971E-2	1.969E-2	1.459E-2	0.97
400	8400	3360000	634	5.107E-2	1.012E-2	7.438E-3	3.48
600	12600	7560000	918	3.481E-2	6.902E-3	5.055E-3	7.51
800	16800	13440000	1137	2.668E-2	5.292E-3	3.864E-3	12.65

Table 8b: IS + SOR ITERATIONS: NONLINEAR CASE

t_{steps}	$N_{\int}$	a_{steps}	ω	iter	E_{abs}	E_{L^1}	E_{rel}	CPU
150	3150	472500	1.5	442	1.243E-1	2.447E-2	1.486E-2	0.52
200	4200	840000	1.45	667	9.888E-2	1.952E-2	1.486E-2	0.90
400	8400	3360000	1.35	1026	5.073E-2	1.004E-2	7.388E-3	3.22
600	12600	7560000	1.25	1198	3.457E-2	6.855E-3	4.969E-3	6.48
800	16800	13440000	1.2	1297	2.534E-2	5.026E-3	3.785E-3	10.65

Table 9a: CN + THOMAS' ALGORITHM: NONLINEAR CASE

t_{steps}	$N_{\int}$	a_{steps}	iter	E_{abs}	E_{L^1}	E_{rel}	CPU
100	2100	210000	189	2.503E-3	7.239E-4	2.174E-3	0.39
150	3150	472500	282	1.846E-3	4.066E-4	9.681E-4	0.67
200	4200	840000	361	2.058E-3	3.893E-4	5.452E-4	1.18
400	8400	3360000	616	2.283E-3	4.471E-4	4.512E-4	3.82
600	12600	7560000	810	2.326E-3	4.591E-4	4.657E-4	7.58

Table 9b: CN + SOR ITERATIONS: NONLINEAR CASE

t_{steps}	$N_{\int}$	a_{steps}	ω	iter	E_{abs}	E_{L^1}	E_{rel}	CPU
100	2100	210000	1.5	377	5.842E-3	1.161E-3	6.936E-4	0.34
150	3150	472500	1.3	448	5.353E-3	1.064E-3	8.649E-4	0.61
200	4200	840000	1.25	528	3.709E-3	7.369E-4	5.969E-4	1.12
400	8400	3360000	1.3	917	2.288E-3	4.545E-4	4.612E-4	3.57
600	12600	7560000	1.18	1101	1.503E-3	2.986E-4	3.122E-4	7.47

What we see from Table 7 is that the behavior of the modified STS we observed in the linear models, does not change here - the method remains very accurate and efficient also in the nonlinear case. For small ν it is fast, but the accuracy is poor.

In order to achieve accuracy we have to proceed as before, i.e. taking larger values of ν which increases sensitively the computational time. The best performance of the modified STS here (as CPU time) was obtained for $\nu = 0.004$ and $N_{STS} = 26$ with maximum absolute error ≈ 0.437, L^1 error ≈ 0.094 and relative error ≈ 0.0675. Increasing the damping factor up to 0.22 and 0.24 we obtained errors similar to the errors of the explicit scheme but with much less costs.

We have to remark that in the nonlinear case the both implicit schemes (and especially the fully implicit scheme) combined with SOR iterations are more efficient than their combination with Thomas' algorithm (compare Tables 8a and 9a with 8b and 9b respectively). Moreover they are more efficient even than the implicit schemes with SOR iterations in the linear case (see Tables 8b and 5b; Tables 9b and 6b). It seems that the integral term in the leading equation of model (6) enables the fast convergence of these schemes. The fully implicit scheme with Thomas' algorithm shows results (as error values) similar to these of the fully implicit scheme with SOR iterations (see Tables 8a and 8b). The same is valid for the Crank-Nicolson method (Tables 9a and 9b). As CPU time, the worst result of the modified STS (CPU=0.30) is better than the best result for Crank-Nicolson + SOR iterations (CPU=0.34). As errors, the best values are obtained by Crank-Nicolson scheme (Tables 9a and 9b). Although the errors of the modified STS are of the same range as these of the fully implicit scheme (compare Tables 7, 8a and 8b), their computational times are vastly different.

7. Different approaches.

Proceeding as in the case without diffusion [20], we want to compare different ways of finding the solution of the linear model (2). In [20] it is shown that the indirect ways, i.e. via the integral equation and via the non-linear formulation, work better than the direct approach. In the diffusion dependent case we observe different behavior and as all the schemes showed similar results we present results only for one of them, i.e. the fully implicit scheme with Thomas' algorithm.

Let us introduce the following notations:
- $\mathbf{E_{absp}}$ - the maximum absolute error for $p(a, x, t)$;
- $\mathbf{E_{absw}}$ - the maximum absolute error for $w(a, x, t)$;
- $\mathbf{E_{relp}}$ - the maximum relative error for $p(a, x, t)$;
- $\mathbf{E_{relw}}$ - the maximum relative error for $w(a, x, t)$;

Table 10: IS + THOMAS' ALGORITHM - via p(a,x,t)

T	N	M	E_{absp}	E_{absw}	E_{relp}	E_{relw}
1	200	20	1.756E-1	3.009E-2	6.056E-3	6.056E-3
1	600	20	2.009E-2	1.182E-2	2.365E-3	2.365E-3
1	400	200	4.436E-2	1.427E-2	2.833E-3	2.833E-3
1	1000	100	7.333E-3	5.772E-3	1.158E-3	1.158E-3

Table 11: IS + THOMAS' ALGORITHM - via w(a,x,t)

T	N	M	E_{absp}	E_{absw}	E_{relp}	E_{relw}
1	200	20	3.745	9.971E-2	1.459E-2	1.459E-2
1	600	20	1.236	3.481E-2	5.055E-3	5.055E-3
1	400	200	1.858	4.891E-2	7.156E-3	7.156E-3
1	1000	100	7.40E-1	1.967E-2	2.872E-3	2.872E-3

From Table 10 and Table 11 it is obvious that in the case with diffusion the better way to obtain $p(a, x, t)$ is the direct treatment of the linear model (2). Moreover it seems that finding $w(a, x, t)$ by $p(a, x, t)$ is better (as accuracy) than obtaining it directly from the non-linear system.

8. Conclusions.

The analysis and computations presented show that modified Super-Time-Stepping is very effective and accurate method in case of age-structured models. It is applicable and simple to employ in an existing explicit code for such problems. Its features - speed, accuracy and implementation simplicity when compared to the implicit schemes, make it preferable also for higher-dimensional and nonlinear problems.

Acknowledgments

I am very thankful to Professor Alexiades for the given support and precious advices. I am grateful to my advisor - Professor Mimmo Iannelli for the useful comments and ideas which led to a better presentation of the article.

References

[1] V. Alexiades, Overcoming the stability restriction of explicit schemes via super-time-stepping, *2nd International Conference on Dynamic Systems and Applications,* (1995), Atlanta.

[2] V. Alexiades, G. Amiez and P-A Gremaud, Super-Time-Stepping acceleration of explicit schemes for parabolic problems, *Communications in numerical methods in engineering,* **12** (1996), 31-42.

[3] B. Ayati, A variable time step method for an age-dependent population model with non-linear diffusion, *Siam Journal of Numerical Analysis,* **37** (5) (2000), 1571-1589.

[4] G. di Blasio, Non-linear age-dependent population diffusion, *Journal of Math. Biology,* **8** (1979), 265-284.

[5] S. Busenberg and M. Iannelli, A class of nonlinear diffusion problems in age-dependent population dynamics, *Nonlinear Analysis, Theory, Methods and Applications,* **7** (5) (1983), 501-529.

[6] S. Busenberg and M. Iannelli, A degenerated nonlinear diffusion problem in age-structured population dynamics, *Nonlinear Analysis, Theory, Methods and Applications,* **7** (12) (1983), 1411-1429.

[7] J. J. Droux, Three-dimensional numerical simulation of solidification by an improved explicit scheme, *Comput. Methods Appl. Mech. Eng.,* **85** (1991), 57-74.

[8] W. Gentzsch, Numerical solution of linear and non-linear parabolic differential equations by a time-discretisation of third order accuracy, *Proceedings of the Third GAMM-Conference on Numerical Methods in Fluid Mechanics* - Vieweg and Son (1979), 109-117.

[9] M. Gurtin, A system of equations for age-dependent population diffusion, *Journal of Theoretical Biology,* **40** (1973), 389-392.

[10] M.-Y. Kim, Galerkin methods for a model of population dynamics with nonlinear diffusion, *Numerical methods for partial differential equations,* **12** (1996), 59-73.

[11] M.-Y. Kim and E.-J. Park, Mixed approximation of a population diffusion equation, *Computers Math. Applic.,* **30** (12) (1995), 23-33.

[12] M. Kubo and M. Langlais, Periodic solutions for a population dynamics problem with age-dependence and spatial structure, *Journal of Math. Biology,* **27** (1991), 363-378.

[13] M. Langlais, Large time behavior in nonlinear age-dependent population dynamics problem with spatial diffuzion, *Journal of Math. Biology,* **26** (1988), 319-346.

[14] R. W. Lewis, I. Masters and J.T. Cross, Automatic timestep selection for the super-time stepping acceleration on unstructured grids using object-oriented programming, *Communications in numerical methods in engineering,* **13** (1997), 249-260.

[15] L. Lopez and D. Trigiante, A finite-difference scheme for a stiff problem arising in the numerical solution of a population dynamical model with spatial diffusion, *Nonlinear Analysis, Theory, Methods and Applications,* **9**(1) (1985), 1-12.

[16] R.C. MacCamy, A population model with non-linear diffusion, *Journal of Differential Equations,* **39** (1981), 52-72.

[17] P. Marcati and R. Serafini, Asymptotic Behaviour in age dependent population dynamics with spatial diffusion, *Bollettino U.M.I.,* **5** (16-B) (1979), 734-753.

[18] F. A. Milner, A numerical method for a model of population dynamics with spatial diffusion, *Computers Math. Applic.,* **19**(4) (1990), 31-43.

[19] G. Pelovska and D. Boyadzhiev, Increasing the Calculation Speed of an Acceleration Scheme for an Age-Structured Diffusion Model, (submitted).

[20] G. Pelovska and M. Iannelli, Numerical methods for the Lotka-McKendrick's equation, *Journal of Computational and Applied Mathematics,* **197** (2006), 534-557.

[21] A. A. Samarskii and A. V. Gulin, Stability of Difference Schemes, *"Nauka"* - Moscow (1973).

[22] G. D. Smith, Numerical Solution of Partial Differential Equations, *Oxford University Press* (1969).

Universitá degli Studi di Trento, via Sommarive 14, I-38050, Povo (Trento), Italy
E-mail: galena@science.unitn.it