An efficient approach for solving stiff nonlinear boundary value problems

D. V. Dragunov

October 2015

Abstract

A new method for solving stiff boundary value problems is described and compared to other known approaches using the Troesch's problems as a test example. A c++ implementation of the proposed method is available at https://github.com/imathsoft/MathSoftDevelopment.

1 Introduction

In the current paper we consider the following nonlinear boundary value problem

$$\frac{d^2 u(x)}{dx^2} = N\left(u(x), x\right) u(x), \quad x \in [a, b], \; N(u, x) \in C^2(\mathbb{R} \times [a, b]), \tag{1}$$

$$u(a) = u_l \in \mathbb{R}, \quad u(b) = u_r \in \mathbb{R},) \tag{2}$$

which arises in many areas of physics and mathematics. Although, there is a huge variety of known methods for solving problems of type (1), (2), almost none of them fill comfortable when the problem turns out to be stiff.

As a quite famous example of stiff problems it is worth to mention the Troesch's problem:

$$\frac{d^2 u(x)}{dx^2} = \lambda \sinh\left(\lambda u(x)\right), \quad x \in [0, 1] \tag{3}$$

$$u(0) = 0, \quad u(1) = 1, \tag{4}$$

which is a partial case of problem (1), (2) with $N(u(x), x) \equiv \lambda \sinh\left(\lambda u(x)\right)/u(x)$, $a = u_l = 0$, $b = u_r = 1$. It is well known, that the problem (3), (4) is inherently unstable and difficult (see [1], [2], [7], [4], [8] and the references therein), in the sense that

$$\lim_{\substack{\lambda \to +\infty \\ x \to 0+}} u_x'(x, \lambda) = 0, \quad \lim_{\substack{\lambda \to +\infty \\ x \to 1-}} u_x'(x, \lambda) = +\infty.$$

Talking about the known methods for solving BVPs, it is impossible not to mention the *simple shooting method* (SSM) and the *multiple shooting method* (MSM) [6, Section 7.3] which are two the must simple and reliable techniques to deal with boundary value problems of type (1), (2). By calling them *techniques* and not just *methods* we would like

1

to emphasize a fact that the basic idea behind them is very broad and can be used in many different modifications, which, in turn, might be called the *methods*. Since definitions of both SSM and MSM essentially relay on using methods for solving *initial value problems* (IVP), one of the ways to modify the methods consists in using different IVP solvers. Below we are going to adopt (or modify if you wish) the SSM and MSM for using a specific approach for numerical solution of IVP's called Straight-Inverse method (or, simply SI-method). The true meaning of the method's name will be explained in the coming sections, and for now we just note that the SI-method presented below has a genuine "immunity" towards problems whose solutions possess rapid variation (i.e. big values of the first derivative). In what follows we will use the Troesch's problem as a testing instrument to reveal strengths and weaknesses of the proposed modifications of the SSM and MSM via using SI-method.

2 Generic functions

In this section we are going to introduce a pair of, so called, *generic functions*, which play a crucial role throughout the rest of the paper.

Let us define the generic function $X(h)$ to be the solution for the following initial value problem:

$$\frac{d^2 X(h)}{dh^2} = (Ah + B)X(h), \quad X(0) = D; \quad X'(0) = C, \quad h \in \mathbb{R}. \tag{5}$$

It is easy to check that the function $X(h)$ can be expressed explicitly through the Airy functions (see [5, 283]) in the following way:

$$X(h) = C_1 \, \mathbf{Ai}\left(\frac{Ah + B}{(-A)^{2/3}}\right) + C_2 \, \mathbf{Bi}\left(\frac{Ah + B}{(-A)^{2/3}}\right), \tag{6}$$

where

$$C_1 = \left(-\mathbf{Bi}\left(\frac{B}{(-A)^{2/3}}\right)C(-A)^{2/3} + DA\,\mathbf{Bi}\left(1, \frac{B}{(-A)^{2/3}}\right)\right)E,$$

$$C_2 = \left(-C(-A)^{2/3}\,\mathbf{Ai}\left(\frac{B}{(-A)^{2/3}}\right) + DA\,\mathbf{Ai}\left(1, \frac{B}{(-A)^{2/3}}\right)\right)E,$$

$$E = A^{-1}\left(\mathbf{Bi}\left(\frac{B}{(-A)^{2/3}}\right)\mathbf{Ai}\left(1, \frac{B}{(-A)^{2/3}}\right) - \mathbf{Bi}\left(1, \frac{B}{(-A)^{2/3}}\right)\mathbf{Ai}\left(\frac{B}{(-A)^{2/3}}\right)\right)^{-1}.$$

Similarly to this, we define generic function $Y(h)$ to be the solution for the problem

$$\frac{d^2 Y(h)}{dh^2} = (Ah + B)\frac{dY(h)}{dh}, \quad Y(0) = D, \quad Y'(0) = C, \quad h \in \mathbb{R}, \tag{7}$$

possessing explicit general solution in the form of

$$Y(h) = \frac{1}{2A}\left(2DA - \left(\mathbf{erf}\left(\frac{B}{\sqrt{-2A}}\right) - \mathbf{erf}\left(\frac{Ah + B}{\sqrt{-2A}}\right)\right)C\sqrt{-2A\pi}\exp\left(\frac{B^2}{2A}\right)\right) \tag{8}$$

$$\mathbf{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2)dt.$$

Though explicit, formulas (6) and (8) are quite difficult to evaluate, especially when $\|A\|$ is quite small. However, in what follows, we are going to use functions $X(h)$ and $Y(h)$ with $\|h\|$ quite small, which allows us to use more convenient approach for evaluating them instead of formulas (6), (8). The approach naturally follows from the theorems below.

Theorem 1
$$X(h) = X(A, B, C, D, h) = \lim_{n \to +\infty} X_n(A, B, C, D, h), \tag{9}$$

where function $X_n(A, B, C, D, h)$ can be found recursively

$$X_n(A, B, C, D, h) = \int_0^h \left(\int_0^\xi (A\eta + B)X_{n-1}(A, B, C, D, \eta)d\eta \right) d\xi + Ch + D, \tag{10}$$

$$X_0(A, B, C, D, h) \equiv 0$$

and the following estimation holds true:

$$\|X_{n+1}(A, B, C, D, h) - X_n(A, B, C, D, h)\| \le \frac{(\|Ah\| + \|B\|)^n(\|Ch\| + \|D\|)h^{2n}}{(2n!)}. \tag{11}$$

Theorem 2
$$Y(h) = Y(A, B, C, D, h) = \lim_{n \to +\infty} Y_n(A, B, C, D, h), \tag{12}$$

where function $Y_n(A, B, C, D, h)$ can be found recursively

$$Y'_n(A, B, C, D, h) = \int_0^h (Ah + B)Y'_{n-1}(A, B, C, D, h) + C, \tag{13}$$

$$Y_n(A, B, C, D, h) = \int_0^h Y'_n(A, B, C, D, \xi)d\xi + D,$$

$$Y'_0(A, B, C, D, h) \equiv 0,$$

and the following estimation holds true:

$$\|Y_{n+1}(A, B, C, D, h) - Y_n(A, B, C, D, h)\| \le \frac{(\|Ah\| + \|B\|)^n\|C\|h^{n+1}}{n!}. \tag{14}$$

The importance of the *generic functions* introduced above can be explained by the approximation properties they possess. The properties are described in the theorems below.

Theorem 3 *Let $u(x)$ be the solution to equation (1) supplemented by the following initial conditions:*

$$u(a) = u_l \in \mathbb{R}, \quad u'(a) = u'_l \in \mathbb{R}. \tag{15}$$

Then

$$\|u(a + x) - X(A, B, C, D, x)\|_{[0,h]} = \mathcal{O}(h^3), \tag{16}$$

$$\|u'(a + x) - X'_x(A, B, C, D, x)\|_{[0,h]} = \mathcal{O}(h^2), \tag{17}$$

3

where

$$
\begin{aligned}
A &= N_u'(u_l, a)u_l' + N_x'(u_l, a), \\
B &= N(u_l, a), \\
C &= u_l', \\
D &= u_l
\end{aligned}
\tag{18}
$$

and

$$
\|f(x)\|_{[a,b]} \overset{def}{=} \max_{\forall x \in [a,b]} |f(x)|, \ f(x) \in C([a,b]).
$$

Lemma 1 *Let $u_l' > 0$. Then in some neighborhood $B(u_l, h) = (u_l - h; u_l + h)$ of the point u_l there exists the unique solution $x(u)$ to the IVP*

$$
\frac{d^2 x(u)}{du^2} = N(u, x(u))\, u \left(\frac{dx(u)}{du}\right)^3,
\tag{19}
$$

$$
x(u_l) = a, \quad x'(u_l) = x_l' = 1/u_l'
\tag{20}
$$

and $x(u)$ is the inverse function to $u(x)$ satisfying (1), (15), i.e. $\forall u_1 \in B(u_l, h)$ we have that $x(u_i) \in \bar{B}(a, h_1)$ for some $h > 0$ and $u_1 \equiv u(x(u_1))$.

Theorem 4 *Let $x(u)$ be a function satisfying IVP (19), (20) in some neighborhood $B(u_l, h)$. Then*

$$
\|x(u_l + u) - Y(A, B, C, D, u)\|_{[0,h]} = \mathcal{O}(h^3),
\tag{21}
$$

$$
\|x'(u_l + u) - Y_u'(A, B, C, D, u)\|_{[0,h]} = \mathcal{O}(h^2),
\tag{22}
$$

where

$$
\begin{aligned}
A &= \left(N_u'(u_l, a) + N_x'(u_l, a)x_l' + N(u_l, a)\right)(x_l')^2 + 2\left(N(u_l, a)u_l\right)^2 (x_l')^3, \\
B &= N(u_l, a)u_l (x_l')^2, \\
C &= x_l', \\
D &= a.
\end{aligned}
\tag{23}
$$

3 Straight-Inverse method for solving IVPs for the second order differential equations

3.1 Preliminary comments

In this section we describe the SI-method for solving equation (1) subjected to the following initial conditions (15). However, before doing that, let us first introduce a list of requirements which we expect the SI-method to fulfill, and which, historically, led us to the SI-method as such:

1. the method should approximate the exact solution $u(x)$ for IVP (1), (15) on a discrete mesh $\varnothing \neq \omega \in [a, b]$, which should depend on the problem itself and on the desired accuracy of approximation;

2. for a given positive $h \in \mathbb{R}$, the method should provide an algorithm to construct a mesh $\omega(h) \in [a, b]$, containing $N_\omega \leq h^{-1} \int_a^b \sqrt{1 + (u'(x))^2} dx$ points, such that, the SI-method's approximation $u_\omega(x)$ of the solution $u(x)$ on the mesh $\omega(h)$ satisfies the following asymptotic equalities:

$$\|u(x) - u_\omega(x)\|_\omega = \mathcal{O}(h^2), \tag{24}$$

$$\|u'(x) - u'_\omega(x)\|_\omega = \mathcal{O}(h^2); \tag{25}$$

3. from the method's point of view, there should be no qualitative difference between solving IVP (1), (15) with respect to $u(x)$ or with respect to its inverse $x(u)$, i.e. the method being applied to the IVP (19), (20), which is the "inverse" equivalent of IVP (1), (15) (in the sense that the graph of x(u) (wherever it exists exists) coincide with that of u(x)), gives the same result (or almost the same result) as when it is applied to the "straight" IVP[1].

3.2   Definition

Denoting by $u_\omega(x)$ the SI-method's approximation of the solution to IVP (1), (15), we define it by the following recurrence chain of equalities:

$$x_0 = a, \quad u_0 = u_l, \quad u'_0 = u'_l; \tag{26}$$

if $|u_{i-1}| \leq 1$ then

$$
\begin{aligned}
x_i &= x_{i-1} + h, \\
u_i &= X(A_{i-1}, B_{i-1}, C_{i-1}, D_{i-1}, h), \\
u'_i &= X'_h(A_{i-1}, B_{i-1}, C_{i-1}, D_{i-1}, h), \\
A_{i-1} &= N'_u(u_{i-1}, x_{i-1})u'_{i-1} + N'_x(u_{i-1}, x_{i-1}), \\
B_{i-1} &= N(u_{i-1}, x_{i-1}), \\
C_{i-1} &= u'_{i-1}, \\
D_{i-1} &= u_{i-1},
\end{aligned} \tag{27}
$$

otherwise, if $|u_{i-1}| > 1$

$$
\begin{aligned}
x_i &= Y(A_{i-1}, B_{i-1}, C_{i-1}, D_{i-1}, h), \\
u_i &= u_{i-1} + h, \\
u'_i &= 1/Y'_h(A_{i-1}, B_{i-1}, C_{i-1}, D_{i-1}, h), \\
A_{i-1} &= \big(N'_u(u_{i-1}, x_{i-1}) + N'_x(u_{i-1}, x_{i-1})x'_{i-1} + \\
&\quad + N(u_{i-1}, x_{i-1})\big)\left(x'_{i-1}\right)^2 + \\
&\quad + 2\left(N(u_{i-1}, x_{i-1})u_{i-1}\right)^2 \left(x'_{i-1}\right)^3, \\
B_{i-1} &= N(u_{i-1}, x_{i-1})u_{i-1}\left(x'_{i-1}\right)^2, \\
C_{i-1} &= 1/u'_{i-1}, \\
D_{i-1} &= x_{i-1},
\end{aligned} \tag{28}
$$

---

[1]Apparently, the statement, as it is now, has a lack of rigor and it might even seem contradictory. However, its meaning will become more clear as we get more familiar with the SI-method and its properties.

where

$$u_i \stackrel{def}{=} u_\omega(x_i), \quad u_i' \stackrel{def}{=} u_\omega'(x_i)$$

and h — some fixed positive[2] real number hereinafter referenced to as a *step size* of the SI-method.

As a matter of the fact, the recurrence equalities (26), (27), (28) provide us a way to construct an ordered set of two dimensional points

$$\omega(h) = \{(x_i, u_i), i = 0, 1, 2 \ldots\} \tag{29}$$

which will be referenced to as a *mesh* of the SI-method.

Theorem 5 *For the mesh $\omega(h)$ obtained through the recurrence formulas (26), (27), (28) the asymptotic equalities (24), (25) holds true.*

4 Straight-Inverse method for solving BVPs for second order differential equations.

In this section we introduce the SI-method for solving boundary value problems. As a matter of the fact we are going to talk about the *simple* and *multiple shooting techniques* supplemented by the SI-method for solving IVPs described above.

Since any *simple shooting technique* is nothing but a bisection algorithm supplemented by an IVP solver, the meaning of the *SI simple shooting* method is evident and self-explanatory. At the same time the *SI multiple shooting method* requires considerably deeper introduction and the rest of the current section is devoted strictly to it.

4.1 SI multiple shooting method.

Talking about the *multiple shooting technique* for solving boundary value problems we (as a rule) mean a way how the given BVP can be transformed to a system of nonlinear algebraic equations together with an algorithm how to solve the system.

Assume that we have fixed some positive parameter $\mathbf{h}$, hereinafter referenced to as a *step size*. In addition to that we need to have some initial guess Ω_k which is some discrete approximation[3] of the exact solution $u(x)$ of the BVP (1), (2):

$$\Omega_k = \big\{\omega_{k,i} = (u_{k,i}', u_{k,i}, x_{k,i})|$$
$$x_{k,i} \in [a, b], \ u(x_{k,i}) \approx u_{k,i}, \ u'(x_{k,i}) \approx u_{k,i}', \ i \in \overline{0, N_k}, \ x_{k,i} < x_{k,j} \Leftrightarrow i < j < N_k \tag{30}$$
$$u_{k,0} = u_l, \ u_{k,N_0} = u_r, \ x_{k,0} = 0, \ x_{k,N_0} = 1\big\},$$

From now on, we assume that

$$\begin{aligned}
h_{k,i} &\stackrel{def}{=} x_{k,i+1} - x_{k,i}, \ i \in \overline{0, N_k - 1}, \\
\bar{h}_{k,i} &\stackrel{def}{=} u_{k,i+1} - u_{k,i}, \ i \in \overline{0, N_k - 1}, \\
\max\{h_{k,i}, \bar{h}_{k,i}\} &\leq \mathbf{h}, \ \forall i \in \overline{0, N_k - 1}.
\end{aligned} \tag{31}$$

---

[2]Actually it might be negative as well, the only really important requirement for h is to be nonzero.

[3]Here we avoid talking about how close the approximation should be, however, in practice, if the approximation is too bed the method described below can do not work at all. At the same time, using the simple shooting approach described above it is always possible to get the desired approximation.

Given that, we transform the set Ω_0 into an ordered set of nonlinear equations

$$\Sigma_0 = \{\sigma_{0,i,j}, \ i \in \overline{0, N_0 - 1}, \ j = 0, 1\}$$

using the rule described below.

For the first two equations we have

$$\sigma_{k,0,0}(\omega_{k,0}, \ \omega_{k,1}) \stackrel{def}{=}$$
$$\left\{ \begin{array}{ll} X(A_X(x_{k,0}, u_{k,0}, \mathbf{u}_0'), \ B_X(x_{k,0}, u_{k,0}, \mathbf{u}_0'), \ \mathbf{u}_0', u_{k,0}, h_{k,0}) = & \mathbf{u_1}, \ |u_{k,0}'| \leq 1, \\ Y(A_Y(u_{k,0}, x_{k,0}, \mathbf{x}_0'), \ B_Y(u_{k,0}, x_{k,0}, \mathbf{x}_0'), \ \mathbf{x}_0', x_{k,0}, \bar{h}_{k,0}) = & \mathbf{x_1}, \ |u_{k,0}'| > 1, \end{array} \right. \quad (32)$$

$$\sigma_{k,0,1}(\omega_{k,0}, \ \omega_{k,1}) \stackrel{def}{=}$$
$$\left\{ \begin{array}{ll} X_h'(A_X(x_{k,0}, u_{k,0}, \mathbf{u}_0'), \ B_X(x_{k,0}, u_{k,0}, \mathbf{u}_0'), \ \mathbf{u}_0', u_{k,0}, h)|_{h=h_{k,0}}, & |u_{k,0}'| \leq 1, \\ Y_h'(A_Y(u_{k,0}, x_{k,0}, \mathbf{x}_0'), \ B_Y(u_{k,0}, x_{k,0}, \mathbf{x}_0'), \ \mathbf{x}_0', x_{k,0}, h)|_{h=\bar{h}_{k,0}}, & |u_{k,0}'| > 1, \end{array} \right. =$$
$$= \left\{ \begin{array}{ll} \mathbf{u}_1', & |u_{k,0}'| \leq 1, \ |u_{k,1}'| \leq 1, \\ 1/\mathbf{x}_1', & |u_{k,0}'| \leq 1, \ |u_{k,1}'| > 1, \\ \mathbf{x}_1', & |u_{k,0}'| > 1, \ |u_{k,1}'| > 1, \\ 1/\mathbf{u}_1', & |u_{k,0}'| > 1, \ |u_{k,1}'| \leq 1. \end{array} \right. \quad (33)$$

As you can see, the equations are dependent on the absolute values of $|u_{k,0}'|$ and $|u_{k,1}'|$, which (in general) represent values of the first derivative of the unknown function $u(x)$. The same is true for rest of the equations, although their descriptions seem even more complicated. For $i \in \overline{1, N_k - 1}$. we have:

$$\sigma_{k,i,0}(\omega_{k,i-1}, \ \omega_{k,i}, \ \omega_{k,i+1}) \stackrel{def}{=}$$
$$\left\{ \begin{array}{ll} X(A_X(x_{k,i}, \mathbf{u}_i, \mathbf{u}_i'), \ B_X(x_{k,i}, \mathbf{u}_i, \mathbf{u}_i'), \ \mathbf{u}_i', \mathbf{u}_i, h_{k,i}), & |u_{k,i}'| \leq 1, \ |u_{k,i-1}'| \leq 1, \\ X(A_X(\mathbf{x}_i, u_{k,i}, \mathbf{u}_i'), \ B_X(\mathbf{x}_i, u_{k,i}, \mathbf{u}_i'), \ \mathbf{u}_i', u_{k,i}, x_{k,i+1} - \mathbf{x}_i), & |u_{k,i}'| \leq 1, \ |u_{k,i-1}'| > 1, \\ Y(A_Y(u_{k,i}, \mathbf{x}_i, \mathbf{x}_i'), \ B_Y(u_{k,i}, \mathbf{x}_i, \mathbf{x}_i'), \ \mathbf{x}_i', \mathbf{x}_i, \bar{h}_{k,i}), & |u_{k,i}'| > 1, \ |u_{k,i-1}'| > 1, \\ Y(A_Y(\mathbf{u}_i, x_{k,i}, \mathbf{x}_i'), \ B_Y(\mathbf{u}_i, x_{k,i}, \mathbf{x}_i'), \ \mathbf{x}_i', x_{k,i}, u_{k,i+1} - \mathbf{u}_i), & |u_{k,i}'| > 1, \ |u_{k,i-1}'| \leq 1, \end{array} \right. =$$
$$= \left\{ \begin{array}{ll} \left\{ \begin{array}{ll} \mathbf{u}_{i+1}, & i \in \overline{1, N_k - 2} \\ u_{k,N_k}, & \text{otherwise} \end{array} \right. , & |u_{k,i}'| \leq 1, \\ \left\{ \begin{array}{ll} \mathbf{x}_{i+1}, & i \in \overline{1, N_k - 2} \\ x_{k,N_k}, & \text{otherwise} \end{array} \right. & |u_{k,i}'| > 1, \end{array} \right. \quad (34)$$

$$\sigma_{k,i,1}(\omega_{k,i-1}, \ \omega_{k,i}, \ \omega_{k,i+1}) \stackrel{def}{=}$$
$$\left\{ \begin{array}{ll} X_h'(A_X(x_{k,i}, \mathbf{u}_i, \mathbf{u}_i'), \ B_X(x_{k,i}, \mathbf{u}_i, \mathbf{u}_i'), \ \mathbf{u}_i', \mathbf{u}_i, h)|_{h=h_{k,i}}, & |u_{k,i-1}'| \leq 1, \ |u_{k,i}'| \leq 1, \\ X_h'(A_X(\mathbf{x}_i, u_{k,i}, \mathbf{u}_i'), \ B_X(\mathbf{x}_i, u_{k,i}, \mathbf{u}_i'), \ \mathbf{u}_i', u_{k,i}, h)|_{h=x_{k,i+1}-\mathbf{x}_i}, & |u_{k,i-1}'| > 1, |u_{k,i}'| \leq 1, \\ Y_h'(A_Y(u_{k,i}, \mathbf{x}_i, \mathbf{x}_i'), \ B_Y(u_{k,i}, \mathbf{x}_i, \mathbf{x}_i'), \ \mathbf{x}_i', \mathbf{x}_i, h)|_{h=\bar{h}_{k,i}}, & |u_{k,i-1}'| > 1, \ |u_{k,i}'| > 1, \\ Y_h'(A_Y(\mathbf{u}_i, x_{k,i}, \mathbf{x}_i'), \ B_Y(\mathbf{u}_i, x_{k,i}, \mathbf{x}_i'), \ \mathbf{x}_i', x_{k,i}, h)|_{h=u_{k,i+1}-\mathbf{u}_i}, & |u_{k,i-1}'| \leq 1, \ |u_{k,i}'| > 1, \end{array} \right. =$$
$$= \left\{ \begin{array}{ll} \mathbf{u}_{i+1}', & |u_{k,i}'| \leq 1, \ |u_{k,i+1}'| \leq 1, \\ 1/\mathbf{x}_{i+1}', & |u_{k,i}'| \leq 1, \ |u_{k,i+1}'| > 1, \\ \mathbf{x}_{i+1}', & |u_{k,i}'| > 1, \ |u_{k,i+1}'| > 1, \\ 1/\mathbf{u}_{i+1}', & |u_{k,i}'| > 1, \ |u_{k,i+1}'| \leq 1, \end{array} \right. \quad (35)$$

where

$$A_X(x,u,u') = N'_u(u,x)u' + N'_x(u,x),$$
$$B_X(x,u,u') = N(u,x),$$
$$A_Y(u,x,x') = \left(N'_u(u,x) + N'_x(u,x)x' + N(u,x)\right)(x')^2 + 2\left(N(u,x)u\right)^2(x')^3, \tag{36}$$
$$B_Y(u,x,x') = N(u,x)u\,(x')^2,$$

bold variables describe unknowns and

$$\mathbf{u}'_i \stackrel{def}{=} 1/\mathbf{x}'_i, \ \forall i \in \overline{0,N_k}.$$

Using Ω_k as an initial guess and applying a single iteration of the generalized Newton's method (see, for example, [6, p. 293]) to the system Σ_k (32), (33), (34), (35) we will get a new set Ω_{k+1} as a combination of Ω_k and the results brought by the Newton's method's iteration. In practice it (quite often) happens that the set Ω_{k+1} obtained as described above needs to be sorted out (to fulfill the requirement $x_{k+1,i} < x_{k+1,j} \Leftrightarrow i < j < N_{k+1}$) and then refined (by linear interpolation to satisfy the inequality $\max\{h_{k+1,i}, \bar{h}_{k+1,i}\} \leq \mathbf{h}, \ \forall i \in \overline{0,N_{k+1}-1}$). Once it is done, we can use Ω_{k+1} to construct a new system Σ_{k+1} and, applying a single Newton's method's iteration to it, get Ω_{k+2} and so on and so forth, until the difference between to subsequent Ω's is not small enough[4].

5 Numerical examples

In this section we are going to apply the SI-method to the Troesch's problem (3), (4).

λ	Value	[8]	[1]	[3]	SI-method, $h = 10^{-4}$	SI-method, $h = 10^{-5}$
20	$u'(0)$	1.648773182e-08	—	1.6487734e-8	1.648773647e-008	1.648773188e-008
20	$u'(1)$	22026.29966	22026.4657	—	22026.4657494062	22026.4657494068
30	$u'(0)$	7.486093793e-13	—	7.4861194e-13	7.486098431e-013	7.486093844e-013
50	$u'(0)$	1.542999878e-21	—	1.5430022e-21	1.543002448e-021	1.542999906e-021
61	$u'(0)$	—	—	2.5770722e-26	2.577078525e-026	2.577072299e-026
100	$u'(0)$	2.976060781e-43	—	—	2.976079583e-043	2.976060996e-043

Table 1: Troesch's problem solved by different approaches.

Value	[2]	[8]	SI-method, $h = 10^{-4}$	SI-method, $h = 10^{-5}$
$u(0.1)$	4.211183679705e-05	4.211189927237e-05	4.21119023173e-05	4.21118993037e-05
$u(0.2)$	1.299639238293e-04	1.299641158237e-04	1.29964125220e-04	1.29964115920e-04
$u(0.3)$	3.589778855481e-04	3.589784013896e-04	3.58978427345e-04	3.58978401657e-04
$u(0.4)$	9.779014227050e-04	9.779027718029e-04	9.77902842508e-04	9.77902772532e-04
$u(0.5)$	2.659017178062e-03	2.659020490351e-03	2.659020682593-03	2.65902049234e-03
$u(0.999)$	8.889931171768e-01	8.889931181558e-01	8.89035025083e-01[5]	8.88994612232e-01[6]

Table 2: Solution to the Troesch's problem with $\lambda = 10$ obtained by different approaches.

---

[4]Of course, it might not always be the case and requires the iteration process to be convergent.

[5]For $x = 0.999000491899$

[6]For $x = 0.999000017539$

[7]Number of knots in the final mesh.

[8]System capabilities : Intel(R) Core(TM) i3-3120M, 2.5 GHz, 8 Gb RAM. Single thread implementation.

[9]Relative difference as compared to $u'(0)$ calculated in [8].

Source	$u'(0)$	$\|\Omega(h)\|$[7]	CPU time, sec.[8]	Rel. diff. to [8][9]
SIM, $h = 10^{-4}$	2.976079584e-43	19844	1.00	0.63e-05
SIM, $h = 10^{-5}$	2.976060996e-43	197679	10.52	0.72e-07
SIM, $h = 10^{-6}$	2.976060783e-43	1976024	100.40	0.67e-09
[8]	2.976060781e-43	—	—	0.0

Table 3: Solution to the Troesch's problem with $\lambda = 100$.

References

[1] Snyman J. A. Continuous and discontinuous numerical solutions to the troesch problem. *Journal of Computational and Applied Mathematics*, 5(3):171–175, 1979.

[2] Shih-Hsiang Chang. A variational iteration method for solving troesch's problem. *Journal of Computational and Applied Mathematics*, 234(10):3043 – 3047, 2010.

[3] I. P. Gavrilyuk, M. Hermann, M. V. Kutniv, and V. L. Makarov. Difference schemes for nonlinear bvps using runge-kutta ivp-solvers. *Advances in Difference Equations*, 2006:Article ID 12167, 29 pages, 2006.

[4] D.J Jones. Solution of troesch's, and other, two point boundary value problems by shooting techniques. *Journal of Computational Physics*, 12(3):429 – 434, 1973.

[5] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 3 edition, 2007.

[6] J. Stoer and R. Bulirsch. *Introduction to numerical analysis*. Springer-Verlag, 2002.

[7] B.A Troesch. A simple approach to a sensitive two-point boundary value problem. *Journal of Computational Physics*, 21(3):279 – 290, 1976.

[8] Hector Vazquez-Leal, Yasir Khan, Guillermo Fernández-Anaya, Agustín Herrera-May, Arturo Sarmiento-Reyes, Uriel Filobello-Nino, Víctor-M. Jimenez-Fernández, and Domitilo Pereyra-Díaz. A general solution for Troesch's problem. *Mathematical Problems in Engineering*, 2012.