A SIMPLE PROGRAM FOR SOLVING NONLINEAR INITIAL VALUE PROBLEM USING ADOMIAN DECOMPOSITION METHOD

F.A. Hendi, H.O. Bakodah, M. Almazmumy & H. Alzumi
Department of Mathematics, Science Faculty for Girls, King Abdulaziz University, Saudi Arabia
falhendi@kau.edu.sa, hbakodah@kau.edu.sa, malmazmumy@kau.edu.sa, hzaizumi@kau.edu.sa

ABSTRACT
The Adomian method is widely used in approximate calculation, its main demerit is that it is very difficult and complex to calculate Adomian's polynomials. Many researchers have suggested different methods and algorithm for computing these polynomials. In this paper, a mathematica program is prepared to solve the initial value problem in ordinary differential equation of the first order. Simplicity and efficiency of the algorithm presented in this paper are illustrated briefly in the examples.

Keywords: Adomian decomposition method, Adomian polynomials, Initial value problem.
2000 Mathematics Subject Classification : 49M27,34A12

1. INTRODUCTION
Adomian decomposition method (ADM) is an approximate approach for solving nonlinear differential equations by substitution of nonlinear parts of equation with Adomian polynomials and use a step by step method for finding the solutions [1]. This method is a powerful approach in nonlinear differential equations and an accuracy of it depends on number of used partial solutions. Also, the solution of this method has a fast convergence to the exact solution generally. In recent years, some modifications on this method have been presented [2,3], these modifications affect on convergence of the method. In some papers [4-7], authors modified the method of computing Adomian's polynomials, which are rapidly convergence by least number of components as compared than standard (ADM). (ADM) has been successfully applied to solve many nonlinear equations, ordinary or partial, deterministic or stochastic, with approximants which converges rapidly to accurate solutions. We shall deal here only with the simple first order differential equation. ADM is one of the new method for solving initial value problem in ordinary differential equation of various kind arising not only in the field of medicine, physical and biological science but also in the area of engineering [8]. In this paper, we present many cases of nonlinearities of I.V.P. and applying (ADM) to get the solution. Adomian and Rach considered nonlinearities for polynomial [9], Negative power [10], decimal power [11], Redicals [12] and composite nonlinearities [13]. They are also used transform techniques with different equations containing convolution product nonlinearities to yield an algebraic equations. ADM has been used to solve nonlinear initial value problems, but applying this method needs some computations which is sometimes boring, having a program to do all computations would be interesting and helpful. In this artical a mathematica program is prepared to solve I.V.P. with different nonlinearity.

2. ADOMIAN DECOMPOSITION METHOD
We use (ADM) for solving first order ordinary differential equations of the form

\[ y' = f(x, y), \quad y(a) = y_0 \]  

(2.1)

The basic concepts of (ADM) [14]:
we assume that \( y(x) \) is sufficiently differentiable and that the solution of (2.1) exists and satisfy the lipschitz condition. (ADM) usaully defines an equation in an operator form by considering the highest -order derivative in the problem. In an operator form, equation (2.1) can be written as

\[ Ly = f(x, y) \]  

(2.2)

where the differential operator \( L \) is given as

\[ L = \frac{d}{dx} \]  

(2.3)

The inverse operator \( L^{-1} \) is considered a one fold integral operator defined by

\[ L^{-1} = \int_a^x f(x)dx \]  

(2.4)
If we operate $L^{-1}$ on both sides of (2.2) and use the initial condition $y(a) = y_0$, we have

$$ y(x) = y_0 + L^{-1}f(x, y) $$

(2.5)

The (ADM) introduces the solution $y(x)$ in an infinite series form as

$$ y(x) = \sum_{n=0}^{\infty} y_n(x) $$

(2.6)

where the components $y_n(x)$ will be determined recursively. Moreover, the method defines the nonlinear function $f(x, y)$ by the infinite series of the form

$$ f(x, y) = \sum_{n=0}^{\infty} A_n $$

(2.7)

If we now use equation (2.6) and (2.7) in (2.5), we have

$$ \sum_{n=0}^{\infty} y_n(x) = y_0 + L^{-1}\sum_{n=0}^{\infty} A_n $$

(2.8)

The next step is to seek a way to determine the component $y_n(x)$. We first identify the zeroth component $y_0(x)$ by the term that arises from the initial condition. The remaining component is determined by using the preceding component. Each term of the series (2.6) is given by the recurrence relation.

$$ y_0(x) = y_0 $$

(2.9)

$$ y_{n+1}(x) = L^{-1}(A_n), \quad n \geq 0 $$

(2.10)

We must state here that in practice all term of the series in (2.6) cannot be determined and the solution will be approximated by the series of the form

$$ \phi_n(x) = \sum_{n=0}^{N-1} y_n(x) \quad , \quad y(x) = \lim_{n \to \infty} \phi_n(x) $$

(2.11)

with (2.11), we obtain the series of the solutions for our system (2.1).

3. SOME CASES OF NONLINEARITIES

In the 1980’s, G. Adomian introduced a new powerful method for solving nonlinear functional equations. The technique is based on a decomposition of a solution of nonlinear operator equation in a series of functions. Each term of the series is obtained from a polynomial generated from an expansion of analytic function into a power series. The Adomian technique is very simple in an abstract formulation but the difficulty arises in calculating the polynomials. The purpose of this section is to study many cases of nonlinearities of this method in application to the initial value problems for nonlinear ordinary differential equations.

3.1 polynomial nonlinearities

Adomian and Rach [9] solved equations of the form

$$ Ly + N y = x(t) $$

(3.1)

where the nonlinear term $Ny = y^m$ for positive integer $m$. With the application of the decomposition method, the solutions to nonlinear differential equations can be made without linearization procedures. Adomian introduced these polynomials, for functions with one variable, by the following formula [15]

$$ A_n(y_0, y_1, y_2, \ldots, y_n) = \frac{1}{n!} \frac{d^n}{d\lambda^n} \left[ f \left( \sum_{i=0}^{n} \lambda^i y_i \right) \right]_{\lambda=0}, \quad n = 0, 1, 2, \ldots $$

(3.2)

Hence, the solution for any equation involving a polynomial nonlinearity is quick and easy.

3.2 Negative power nonlinearities

Differential equations involving the term $y^{-m}$, where $m$ is a positive integer, are solved by decomposition method
[10], Then the $A_n$ are given by

$$A_0 = y_0^{-m}$$

$$A_1 = -my_0^{-(m+1)} y_1$$

$$A_2 = \frac{1}{2} m(m+1)y_0^{-(m+2)} y_1^2 - my_0^{-(m+1)} y_2$$

$$A_3 = -\frac{1}{6} m(m+1)(m+2)y_0^{-(m+3)} y_1^3 + m(m+1)y_0^{-(m+2)} y_1 y_2 - my_0^{-(m+1)} y_3$$

... .

Consequently,

$$y_n = (-1)^n \prod_{v=0}^{n-1} (vm + v - 1)^{-k^{-(m+n-1)}} t^n$$

so that

$$y(t) = \prod_{v=0}^{n-1} (-1)^n \sum_{v=0}^{n-1} (vm + v - 1)^{-k^{-(m+n-1)}} t^n$$

is immediately calculable from expressions for $A_n$ for negative powers of $y$ remembering $L^{-1}$ represents $m$-fold integration from 0 to $t$.

### 3.3 Decimal power nonlinearities

If $Ny$ is a nonlinear term of the form $y^\gamma$ with $\gamma$ a decimal number [11], Let $y + y^\gamma = 0$ and $y(0) = k$ (extension to cases $\frac{d^m y}{dt^m} + y^\gamma = x(t)$ with appropriate given conditions is a simple generalization). The Adomian polynomials are

$$A_0 = y_0^\gamma$$

$$A_1 = \gamma y_0^{\gamma-1} y_1$$

$$A_2 = \gamma y_0^{\gamma-2} y_2 + \frac{1}{2} \gamma (\gamma - 1) y_0^{\gamma-2} y_1^2$$

$$A_3 = \gamma y_0^{\gamma-3} y_3 + \gamma (\gamma - 1) y_0^{\gamma-2} y_1 y_2 + \frac{1}{5} \gamma (\gamma - 1) (\gamma - 2) y_0^{\gamma-3} y_1^3$$

... .

Consequently,

$$y(t) = \sum_{m=0}^{\infty} (-1)^m \left( \prod_{\mu=0}^{m-1} \left( \mu \gamma - (\mu - 1) \right) \right) k^{m\gamma - (m-1)} \frac{t^m}{(mn)!}$$

### 3.4 Equations containing radicals

Suppose we are considering equation with
We treat the nonlinear or radical term \( (\alpha y^2 + \beta)^{\frac{1}{2}} \) as the composite nonlinearity \([12]\) with

\[
N_y = \left(\alpha y^2 + \beta\right)^{\frac{1}{2}}
\]

Calculated the \( A_n^0 \) polynomials

\[
A_0^0 = \left(u_0^0\right)^\frac{1}{2} = \left(\beta + \alpha A_0^1\right)^{\frac{1}{2}}
A_1^0 = \frac{1}{2} \left(u_0^0\right)^\frac{3}{2} (u_1^0) = \frac{1}{2} \left(\beta + \alpha A_1^1\right)^{\frac{3}{2}} (\alpha A_1^1)
A_2^0 = \frac{1}{2} \left(u_0^0\right)^\frac{5}{2} (u_2^0) - \frac{1}{8} \left(u_0^0\right)^\frac{7}{2} (u_1^0)^2
\]

Then, the solution for

\[
\frac{dy}{dt} + \sqrt{\alpha y^2 + \beta} = 0, \quad y(0) = k
\]

is

\[
y = k - \left(\alpha k^2 + \beta\right)^{\frac{1}{2}} t + \alpha k^2 - \alpha \left(\alpha k^2 + \beta\right)^{\frac{3}{2}} \frac{t^3}{6} + ...
\]

4. **COMPOSITE NONLINEARITIES**

The composite nonlinearities of the form \( N(x) = N_0(N_1(N_2(...(x)...))) \) where \( N(x) \) a nonlinear term in an equation to be solved by decomposition. Terms such as \( x^2, e^x, \sin x \) etc are viewed as \( N_0u^0 \) where \( u^0 = x \), and expanded in Adomian's polynomial \([13]\). Now identified as \( A_n^0 \) to correspond to the \( N_0 \) nonlinear operator.

Thus, \( N_0u^0 = \sum_{n=0}^{\infty} A_n^0 \).

A first order composite nonlinearity is defined as \( \boxed{N_1.x = N_0\left(N_1u^1\right)} \) or as \( N_0N_1u^1 \) with \( u^0 = x \) and \( u_0 = N_1u^1 \) with \( N_0u^0 = \sum_{n=0}^{\infty} A_n^0 \) and \( N_0u^1 = \sum_{n=0}^{\infty} A_n^1 \).

In general, \( N_nu^n = \sum_{n=0}^{\infty} A_n^0 = u^{n-1} \) for \( 1 \leq n \leq m \) with \( u^{m} = x \) and \( u^n = \sum_{n=0}^{\infty} u^{n} \) when the results apply to differential equations in the form

\[
Ly + Ny = g(x)
\]

where \( Ny \) is a composite nonlinearity, we get

\[
L^{-1} Ly = L^{-1} g(x) - L^{-1} Ny
= L^{-1} g(x) - L^{-1} \sum_{n=0}^{\infty} A_n
\]

If \( Ly = \frac{dy}{dx}, y(0) = k \), then

400
where

\[ y = \sum_{n=0}^{\infty} y_n = k + L^{-1}g - L^{-1} \sum_{n=0}^{\infty} A_n \]

\[ y_0 = k + L^{-1}g \]

\[ y_{n+1} = -L^{-1}A_n \quad \text{for} \quad n \geq 0 \]

and the \( A_n \) are calculated by the methods discussed.

5. A SIMPLE ALGORITHM FOR SOLVING I.V.P.

Many researchers have suggested different methods and algorithm for computing Adomian polynomials. Wazwaz suggestion [4] is to substitute the series solution (2.6) into (2.2) and then by manipulation terms, based on algebraic operations, trigonometric identities and Taylor series as appropriate, all terms can be collected such that the subscripts of the components of in each terms is the same. With this step preformed, the calculation of the Adomian polynomials is thus completed. In [5], Rach introduced a new definition of Adomian polynomial to provide a new proof of convergence of the Adomian decomposition series for solving nonlinear ordinary and partial differential equations. Adomian polynomials are expressed in terms of new objects called reduced polynomials. In [6], these new objects, which carry two subscripts, are independent of the form of the nonlinear operator. In [7], the authors modified the method of computing Adomian's polynomial to find the numerical solution for nonlinear systems of partial differential equations with less number of components, more accuracy and faster convergence when compared with the standard Adomian decomposition method.

The main goal of this work is to present a mathematica program to compute easily these polynomials for nonlinear terms in solving I.V.P. in ordinary differential equations and to compute \( A_n \) in general for any kind of nonlinearities.

6. MATHEMATICA PROGRAM

Consider the following first order initial value problem of the form

\[ y = f(x, y), \quad y(a) = y_0 \]

In Adomian decomposition method, we can consider the solution as summation of a series

\[ Y_{\lambda} = \sum_{j=0}^{\infty} y_{\lambda_j} \]

and \( f(x, y) \), as a series, say:

\[ f_{\lambda}(x, y) = \sum_{j=0}^{\infty} A_{\lambda_j} \lambda_j \]

where \( A_{\lambda_j}(y_0, y_1, ..., y_n) \), which depends on \( y_0, y_1, ..., y_j \) called Adomian polynomials and are defined as

\[ A_{\lambda_n} = \frac{1}{n!} \frac{d^n}{d\lambda^n} \left[ f \left( x, \sum_{j=0}^{n} \lambda_j y_j \right) \right]_{\lambda=0} \quad , \quad n = 0, 1, 2, ... \]

The inputs of the algorithm are the following:

\( n \), the number of terms of the approximations of the solutions.

\( g(x) \), the first term in the right hand side of the canonical form

\[ Ly + Ny = g(x) \]

\( n = 5 \)

\( S[\lambda] := \sum_{i=0}^{\infty} \lambda_i y_i \)

\( \text{Ad}=\text{Table} \left[ \frac{1}{n!} D\left[ f[S[\lambda]], \{\lambda, n\}\right], \lambda \rightarrow 0, \{n, 0, 5\} \right] \);
Table Form \[{ad, Table Alignments \rightarrow Left, TableHeadings\}
\rightarrow \{\{A_0^", \ A_1^", \ A_2^", \ A_3^", \ A_4^"\}, None\}].

A few cases of nonlinear operators are given.

6.1 Nonlinear polynomials \( f(y) = y^m \)
Substituting \([S[\lambda]] \rightarrow 42\) instead of \( f[S[\lambda]] \) in the above program we get
\[
A_0[x] = y_0[x]^2;
A_1[x] = 2y_0[x]y_1[x];
A_2[x] = \frac{1}{2} \left( 2y_1[x]^2 + 4y_0[x]y_2[x] \right);
A_3[x] = \frac{1}{6} \left( 12y_1[x]y_2[x] + 12y_0[x]y_3[x] \right);
A_4[x] = \frac{1}{24} \left( 24y_2[x]^2 + 48y_1[x]y_3[x] + 48y_0[x]y_4[x] \right);
A_5[x] = \frac{1}{120} \left( 240y_2[x]y_3[x] + 240y_1[x]y_4[x] + 240y_0[x]y_5[x] \right);
\]

6.2 Negative power nonlinearities
\( f(y) = y^{-m} \)
\[
A_0[x] = y_0^{-m};
A_1[x] = -my_0^{-1-m}y_1;
A_2[x] = \frac{1}{2} \left( -m(-1-m)y_0^{-2-m}y_1^2 - 2my_0^{-1-m}y_2 \right);
A_3[x] = \frac{1}{6} \left( (-2-m)(-1-m)m \ y_0^{-3-m}y_1^3 - 6(-1-m)my_0^{-2-m}y_1y_2 - 6my_0^{-1-m}y_3 \right);
A_4[x] = \frac{1}{24} \left( (-3-m)(-2-m)(-1-m)m \ y_0^{-4-m}y_1^4 - 12(-2-m)(-1-m)m \ y_0^{-3-m}y_1^2y_2 - 12(-1-m)my_0^{-2-m}y_1y_3 - 24my_0^{-1-m}y_4 \right);
A_5[x] = \frac{1}{120} \left( (-4-m)(-3-m)(-2-m)(-1-m)m \ y_0^{-5-m}y_1^5 - 20(-3-m)(-2-m)(-1-m)m \ y_0^{-4-m}y_1^3y_2 - 60(-2-m)(-1-m)m \ y_0^{-3-m}y_1y_3^2 - 60(-1-m)m \ y_0^{-2-m}y_2y_3 - 120(-1-m)m \ y_0^{-1-m}y_4 - 120m \ y_0^{-1-m}y_5 \right);
\]

6.3 Decimal power nonlinearities
\( f(y) = y^\frac{1}{3} \)
\[
A_0[x] = y_0^\frac{1}{3};
\]
\[ A_1[x] = \frac{y_1}{3y_0^3} \]
\[ A_2[x] = \frac{1}{2} \left( \frac{2y_1^2}{9y_0^{\frac{5}{3}}} + \frac{2y_2}{2y_0^{\frac{5}{3}}} \right) \]
\[ A_3[x] = \frac{1}{6} \left( \frac{10y_1^3}{27y_0^{\frac{5}{3}}} - \frac{4y_1y_2}{3y_0^{\frac{5}{3}}} + \frac{2y_3}{y_0^{\frac{5}{3}}} \right) \]
\[ A_4[x] = \frac{1}{24} \left( \frac{80y_1^4}{81y_0^3} + \frac{40y_1^2y_2}{9y_0^3} - \frac{8y_2^2}{3y_0^3} - \frac{16y_1y_3}{3y_0^3} + \frac{8y_4}{2y_0^3} \right) \]
\[ A_5[x] = \frac{1}{120} \left( \frac{880y_1^5}{243y_0^3} - \frac{1600y_1^3y_2}{81y_0^3} + \frac{200y_1^2y_3}{9y_0^3} + \frac{200y_1y_4}{y_0^3} - \frac{80y_2y_3}{3y_0^3} - \frac{80y_1y_4}{3y_0^3} + \frac{40y_3}{y_0^3} \right) \]

6.4 Equations containing radicals

\[ Ny = (\beta + \alpha y^2) \frac{1}{y} \]
\[ A_0[x] = \sqrt{\beta + \alpha y_0^2} \]
\[ A_1[x] = \frac{\alpha y_0}{\sqrt{\beta + \alpha y_0^2}} \]
\[ A_2[x] = \frac{1}{2} \left( -\frac{\alpha^2y_0^2y_1^2}{(\beta + \alpha y_0^2)^{\frac{3}{2}}} + \frac{\alpha y_1^2}{\sqrt{\beta + \alpha y_0^2}} + \frac{2\alpha y_0y_2}{\sqrt{\beta + \alpha y_0^2}} \right) \]
\[ A_3[x] = \frac{1}{6} \left( \frac{3\alpha^3y_0y_1y_2}{(\beta + \alpha y_0^2)^{\frac{3}{2}}} \frac{3\alpha^2y_0y_1y_3}{(\beta + \alpha y_0^2)^{\frac{3}{2}}} \frac{6\alpha^2y_0y_2y_3}{(\beta + \alpha y_0^2)^{\frac{3}{2}}} \frac{6\alpha y_0y_2}{\sqrt{\beta + \alpha y_0^2}} \frac{6\alpha y_0y_3}{\sqrt{\beta + \alpha y_0^2}} \right) \]
\[ A_4[x] = \frac{1}{24} \left( -\frac{15\alpha^2y_0y_1^4}{(\beta + \alpha y_0^2)^{\frac{3}{2}}} + \frac{18\alpha^2y_0y_1y_2y_3}{(\beta + \alpha y_0^2)^{\frac{3}{2}}} - \frac{3\alpha^2y_1^4}{(\beta + \alpha y_0^2)^{\frac{3}{2}}} + \frac{36\alpha^2y_0y_1^2y_3}{(\beta + \alpha y_0^2)^{\frac{3}{2}}} - \frac{36\alpha y_0y_1y_2}{\sqrt{\beta + \alpha y_0^2}} + \frac{24\alpha y_0y_3}{\sqrt{\beta + \alpha y_0^2}} \right) \]
\[ A_5[x] = \frac{1}{120} \left( \frac{105\alpha^5y_0y_1^5}{(\beta + \alpha y_0^2)^{\frac{5}{2}}} + \frac{150\alpha^4y_0y_1^3y_2}{(\beta + \alpha y_0^2)^{\frac{5}{2}}} + \frac{45\alpha^3y_0y_1y_3}{(\beta + \alpha y_0^2)^{\frac{5}{2}}} - \frac{300\alpha^4y_0y_1^2y_3}{(\beta + \alpha y_0^2)^{\frac{5}{2}}} - \frac{360\alpha^3y_0y_1y_4}{(\beta + \alpha y_0^2)^{\frac{5}{2}}} \right) \]

6.5 Composite nonlinearities
\[ f(y) = \exp(\sin y) \]
\[ A_0[x] = e^{\sin[y_0]} \]
\[ A_1[x] = e^{\sin[y_0]} \cos[y_0]y_1 \]
\[ A_2[x] = \frac{1}{2} \left( e^{\sin[y_0]} \cos[y_0]y_1^2 - e^{\sin[y_0]} \sin[y_0]y_1^2 + 2e^{\sin[y_0]} \cos[y_0]y_2 \right) \]
\[ A_3[x] = \frac{1}{6} \left( -e^{\sin[y_0]} \cos[y_0]y_1^3 + e^{\sin[y_0]} \cos[y_0]y_1^3 - 3e^{\sin[y_0]} \cos[y_0]y_1^3 + 6e^{\sin[y_0]} \cos[y_0]y_1y_2 - 6e^{\sin[y_0]} \sin[y_0]y_1y_2 + 6e^{\sin[y_0]} \cos[y_0]y_3 \right) \]
\[ A_4[x] = \frac{1}{24} \left( -4e^{\sin[y_0]} \cos[y_0]y_1^4 + e^{\sin[y_0]} \cos[y_0]y_1^4 + 3e^{\sin[y_0]} \sin[y_0]y_1^4 - 12e^{\sin[y_0]} \cos[y_0]y_1^3y_2 - 6e^{\sin[y_0]} \sin[y_0]y_1^3y_2 + 12e^{\sin[y_0]} \cos[y_0]y_3^2 - 2e^{\sin[y_0]} \sin[y_0]y_3^2 + 24e^{\sin[y_0]} \cos[y_0]y_1y_3 + 24e^{\sin[y_0]} \sin[y_0]y_1y_3 + 24e^{\sin[y_0]} \cos[y_0]y_4 \right) \]

7. NUMERICAL EXAMPLES

Adomian decomposition method as a possible approach can be applied to the analysis of equation modeling water flows [8] from an inverted conical tank with circular orifice at the rate

\[ \frac{dy}{dt} = -0.6\pi r^2 (-2g) \frac{\sqrt{y}}{A(y)} \]

where \( r \) is the radius of the orifice, \( x \) is the height of the liquid level from the vertex of the cone, and \( A(y) \) is the area of the cross section of the tank \( y \) unit above the orifice. Suppose \( r = 0.1 \text{ ft} \), \( g = -32.17 \text{ ft/s}^2 \), and the tank has an initial water level of 8 ft and initial volume of \[ \frac{512}{3} \text{ ft}^3 \].

We need to compute the water level after 10 min with \( h = 205 \), and determine, to within 1 min, when the tank will be empty.

Solution of the problem:
If \( H = 8 \) and \( V = 512 \frac{\pi}{3} \), and the cone volume is \( V = \frac{1}{3} \pi R^2 H \).

We can obtained that \( R = 8 \) and \( \tan \theta = \frac{\pi}{4} \). This implies that \( R = y \) for all \( y \).

Using this results, we have

\[
\frac{dy}{dt} = 0.048y^\frac{3}{2}(t) \quad \text{or} \quad y(t) = -0.048\int_0^t y^\frac{3}{2}(s)ds
\]

To solve this equation by Adomian's decomposition method, let \( y(t) = \sum_{m=0}^\infty y_m(t) \) and

\[
-0.048\int_0^t y^\frac{3}{2}(s)ds = \sum_{m=0}^\infty A_m \quad \text{where} \quad A_m, m = 0,1,2,... \quad \text{are Adomian polynomials}.
\]

Hence, we can rewrite the equation as

\[
\sum_{m=0}^\infty y_m(t) = y(0) + \sum_{m=0}^\infty A_m
\]

From this equation, the solution is then obtained by the following Adomian scheme

\[
y(0) = 8
\]

\[
y_{n+1}(t) = 0.048L^{-1}A_i, \quad i = 0,1,2,...
\]

Using the mathematica program, Adomian polynomials given by

\[
A_0 = 1/y_0^2
\]

\[
A_1 = -3y_1/2y_0^5
\]

\[
A_2 = \frac{1}{2} \left( \frac{15y_1^5}{4y_0^9} - \frac{3y_2}{y_0^5} \right)
\]

\[
A_3 = \frac{1}{6} \left( \frac{-105y_1^7}{8y_0^9} + \frac{45y_1^5y_2}{2y_0^7} - \frac{9y_3}{y_0^5} \right)
\]

\[
A_4 = \frac{1}{24} \left( \frac{945y_1^9}{16y_0^{11}} - \frac{315y_1^7y_2}{2y_0^9} + \frac{45y_2^5}{y_0^7} + \frac{90y_1^5y_3}{y_0^5} - \frac{36y_4}{y_0^5} \right)
\]

\[
A_5 = \frac{1}{120} \left( \frac{-10395y_1^{11}}{32y_0^{13}} + \frac{4725y_1^9y_2}{3y_0^{11}} - \frac{1575y_1^7y_2^2}{2y_0^9} + \frac{1575y_1^5y_3^2}{y_0^7} + \frac{450y_1^3y_4^2}{y_0^7} + \frac{450y_1y_5^2}{y_0^7} - \frac{180y_5}{y_0^5} \right)
\]

we can extracted the polynomials as we want. The approximate solution including five terms is

\[
y(t) = 8 - 0.002121327 - 4.21875 \times 10^{-7} t^2 + 1.491155 \times 10^{-10} t^3 - 6.427 \times 10^{-14} t^4 + 3.06759 \times 10^{-17} t^5 - 1.55905 \times 10^{-20} t^6.
\]

Approximate solution, for value \( t = 10 \) is \( 8.02117 \).

8. CONCLUSION
The Adomian decomposition method is a powerful device for solving a large class of nonlinear problems in science. The main part of this method is calculating Adomian polynomials for nonlinear terms. In this paper, we considered many cases of nonlinearities of initial value problem in ordinary differential equation of the first order. We introduced a simple algorithm for calculating Adomian polynomials. This algorithm is very easy to implement and may extended to calculate Adomian polynomials for nonlinear functionals of several problems. We applied the program for solving some example of nonlinearities type to compute Adomian polynomials, those are familiar with Adomian polynomials are well aware of the amount of computational work of this method. The program is checked by calculating the water flows problem.

REFERENCES