Symbolic Generation of Adomian Polynomials for Different Nonlinearities by Python

ABSTRACT. The Adomian decomposition method (ADM) is a powerful mathematical technique to find closed-form solutions to nonlinear functional equations including ODEs, PDEs, differential-difference, integral, integro-differential, algebraic, and transcendental equations or systems of such equations. It features a particular infinite series for the representation of nonlinear terms of the equation under study, referred to as the Adomian polynomials. Nevertheless, the computation of such polynomials manually, devoid of any assistance from computational resources, can often be a laborious and protracted endeavor. In this paper, an innovative Python code is proposed, which exploits the SymPy library to perform the involved symbolic calculus operations to generate the Adomian polynomials of any given nonlinear expressions. The use of the code would substantially facilitate the implementation of the ADM to the equations arising in various branches of science and engineering. A number of nonlinear expressions are decomposed to their relevant Adomian polynomials for the sake of demonstration.

Keywords: Adomian Decomposition Method, Adomian Polynomials, Differential Equations, Nonlinear Equations, Python.

Introduction

Years of studying nature and making efforts to quantify real world problems by means of mathematics has proven that physical and engineering models usually lead to equations that incorporate nonlinearities. The more accuracy is demanded, the more complicated equations are derived. As a result, there is always a need for more efficient solution methods. These methods are inexhaustively categorized into two groups: analytical methods and numerical methods. Numerical methods will not provide an exact answer for all the domains of the problem nor for a specific point of the problem domain. In other words, there will be no closed-form solution using the numerical methods. On the other hand, analytical solutions will provide a closed-form solution to the problem. However, they are usually achievable for specific cases, like linear equations or some specific nonlinearities which are rather simple. Therefore, in 1983, George Adomian introduced a potent analytical technique which can be implemented to find the solution to all kinds of nonlinear functional equations, called the Adomian Decomposition Method (ADM) [1]. Needless to say, this method can also be used to find the solutions to linear equations. The ADM does not impose any linearization, perturbation or discretization and leads to convergent solutions quickly. Many problems from various branches of science and engineering have been solved by the ADM. Areas that the ADM have been applied in include heat transfer [2-9], population dynamics [10-13], fluid dynamics [14-17], thermodynamics [18-22], applied chemistry [23-27], optics [28, 29], etc. Particularly, the ADM has been applied to problems arising from chemical engineering; for instance, in catalysis and reactor engineering [30], gas absorption [31], bioreactors [32], and electrochemical systems [33]. Particularly, The

Hammerstein integral equation manifests in chemical reactor engineering subsequent to employing the Green's function technique and has been proficiently solved using the ADM [34].

As it will be discussed, the ADM requires a particular series representation for the nonlinearities involved in the equation, namely the Adomian Polynomials. Several efforts have been made to derive procedures for computing these polynomials. However, some of them are restricted to only special cases of nonlinearity and many of them involve complexity. A number of algorithms to calculate the Adomian polynomials have been developed so far [35-40]. They consist intricate structures and all are coded in Maple or MATLAB.

The present paper, proposes a code to efficiently compute the Adomian polynomials in Python, an open source, high level, widely used programming language, getting its basic concept from a straightforward technique. Illustrative examples are provided to show the reliability of the program.

Methodology

Consider, without loss of generality, the following functional equation:

$$u - N(u) = f \tag{1}$$

where N is a nonlinear operator on a Banach space E, f is a specified element of E and we are seeking $u \in E$, which satisfies Eq. (1). Assuming that Eq. (1) has a unique solution for every $f \in E$, then the ADM decomposes the solution u as an infinite series $u = \sum_{i=0}^{\infty} u_i$ and the nonlinearity as $N(u) = \sum_{i=0}^{\infty} A_i$, where the A_i are called the Adomian polynomials and are defined as:

$$A_i = \frac{1}{i!}\frac{d^i}{d\lambda^i} N\left(\sum_{k=0}^{\infty} u_k \lambda^k\right)\Bigg|_{\lambda=0} \tag{2}$$

By selecting the initial solution component as $u_0 = f$, the ADM uses the following expression to generate components of the solution as:

$$\begin{cases} u_0 = f, \\ u_{i+1} = A_i, i \geq 0 \end{cases} \tag{3}$$

The convergence and reliability of the ADM have been ascertained in prior research [41,42].

Among all the different formulas for the Adomian polynomials, we are going to use the mentioned definitional one, i.e. Eq. (2) due to its simplicity and succinctness. The code will be generated in the Python programming language, which is nowadays really popular among researchers and is implemented globally, since its syntax is very close to human's speech and is an object-oriented open-source programming language with many practical libraries.

The Python Code

```python
#Copyright (c) [2023] [A. Houshmand, M. Noorimohammad, and H. Fatoorehchi]

#The code presented in this paper is protected by copyright law.
#Permission is granted for personal or educational use, subject to
#the following conditions: (1) Include the copyright notice in all
#copies, (2) Prohibit commercial use and distribution without
#written permission, and (3) Disclaimer of warranties.
#For other usage or questions, contact [hfatoorehchi@ut.ac.ir].

# Import the necessary mathematical libraries
import math
import sympy as sym

# Define the order of the Adomian series expansion
n = 10

# Create variables u0, u1, ..., un-1 for the coefficients in the series
u = {}

for i in range(n):
  key = 'u{}'.format(i)
  u[key] = sym.symbols(key)
u_list = list(u.values())
Lambda = sym.symbols('Lambda')


# Calculate the series representation S(Lambda)
s = 0
for i in range(n):
  s = s + pow(Lambda,i)*u_list[i]

# Compute the nonlinearity term N(s)
N_s = s**2

# Calculate the Adomian polynomials A0, A1, ..., An-1
kk = range(n)
A_K = [None] * len(kk)
A_KK = [None] * len(kk)  # A_KK stores the Adomian result
for i in kk:
  A_K [i] = (1/(sym.factorial(i)))*(sym.diff(N_s, Lambda,i))
  A_KK[i] = A_K[i].subs(Lambda,0)
```

```
# Print the Adomian polynomials
print (A_KK)
```

The *Python* code can be downloaded from this address:
https://hfatoorehchi.com/Admpolypython.ipynb

Illustrative Examples

To illustrate the program results, several Adomian polynomials related to the most frequent, in real-world applications, nonlinearities are presented here. Note that in our code, the variable ‘*s*’ has been used to express the nonlinearity term $N(s) = s^2$ as an example. Hence, one can replace it with any desired nonlinearity operating on variable ‘*s*’.

Table 1. First ten Adomian polynomials for $N(u) = u^2$ and $N(u) = u^3$

$N(u) = u^2$	$N(u) = u^3$
$A_0 = u_0^2$	$A_0 = u_0^3$
$A_1 = 2u_0u_1$	$A_1 = 3u_0^2u_1$
$A_2 = 2u_0u_2 + u_1^2$	$A_2 = 3u_0(u_0u_2 + u_1^2)$
$A_3 = 2u_0u_3 + 2u_1u_2$	$A_3 = 3u_0^2u_3 + 6u_0u_1u_2 + u_1^3$
$A_4 = 2u_0u_4 + 2u_1u_3 + u_2^2$	$A_4 = 3u_0^2u_4 + 6u_0u_1u_3 + 3u_0u_2^2 + 3u_1^2u_2$
$A_5 = 2u_0u_5 + 2u_1u_4 + 2u_2u_3$	$A_5 = 3u_0^2u_5 + 6u_0u_1u_4 + 6u_0u_2u_3 + 3u_1^2u_3 + 3u_1u_2^2$
$A_6 = 2u_0u_6 + 2u_1u_5 + 2u_2u_4 + u_3^2$	$A_6 = 3u_0^2u_6 + 6u_0u_1u_5 + 6u_0u_2u_4 + 3u_0u_3^2 + 3u_1^2u_4$ $+6u_1u_2u_3 + u_2^3$
$A_7 = 2u_0u_7 + 2u_1u_6 + 2u_2u_5 + 2u_3u_4$	$A_7 = 3u_0^2u_7 + 6u_0u_1u_6 + 6u_0u_2u_5 + 6u_0u_3u_4 + 3u_1^2u_5$ $+ 6u_1u_2u_4 + 3u_1u_3^2 + 3u_2^2u_3$
$A_8 = 2u_0u_8 + 2u_1u_7 + 2u_2u_6 + 2u_3u_5 + u_4^2$	$A_8 = 3u_0^2u_8 + 6u_0u_1u_7 + 6u_0u_2u_6 + 6u_0u_3u_5 + 3u_0u_4^2$ $+3u_1^2u_6 + 6u_1u_2u_5 + 6u_1u_3u_4 + 3u_2^2u_4 + 3u_2u_3^2$
$A_9 = 2u_0u_9 + 2u_1u_8 + 2u_2u_7 + 2u_3u_6 + 2u_4u_5$	$A_9 = 3u_0^2u_9 + 6u_0u_1u_8 + 6u_0u_2u_7 + 6u_0u_3u_6 + 6u_0u_4u_5$ $+ 3u_1^2u7 + 6u_1u_2u_6 + 6u_1u_3u_5 + 3u_1u_4^2 + 3u_2^2u_5$ $+6u_2u_3u_4 + u_3^3$

Table 2. First ten Adomian polynomials for $N(u) = u^4$

$N(u) = u^4$
$A_0 = u_0^4$
$A_1 = 4u_0^3u_1$
$A_2 = 2u_0^2(2u_0u_2 + 3u_1^2)$
$A_3 = 4u_0(u_0^2u_3 + 3u_0u_1u_2 + u_1^3)$
$A_4 = 4u_0^3u_4 + 12u_0^2u_1u_3 + 6u_0^2u_2^2 + 12u_0u_1^2u_2 + u_1^4$
$A_5 = 4u_0^3u_5 + 12u_0^2u_1u_4 + 12u_0^2u_2u_3 + 12u_0u_1^2u_3 + 12u_0u_1u_2^2 + 4u_1^3u_2$
$A_6 = 4u_0^3u_6 + 12u_0^2u_1u_5 + 12u_0^2u_2u_4 + 6u_0^2u_3^2 + 12u_0u_1^2u_4 + 24u_0u_1u_2u_3 + 4u_0u_2^3 + 4u_1^3u_3 + 6u_1^2u_2^2$
$A_7 = 4u_0^3u_7 + 12u_0^2u_1u_6 + 12u_0^2u_2u_5 + 12u_0^2u_3u_4 + 12u_0u_1^2u_5 + 24u_0u_1u_2u_4 + 12u_0u_1u_3^2 + 12u_0u_2^2u_3$ $+ 4u_1^3u_4 + 12u_1^2u_2u_3 + 4u_1u_2^3$
$A_8 = 4u_0^3u_8 + 12u_0^2u_1u_7 + 12u_0^2u_2u_6 + 12u_0^2u_3u_5 + 6u_0^2u_4^2 + 12u_0u_1^2u_6 + 24u_0u_1u_2u_5 + 24u_0u_1u_3u_4$ $+ 12u_0u_2^2u_4 + 12u_0u_2u_3^2 + 4u_1^3u_5 + 12u_1^2u_2u_4 + 6u_1^2u_3^2 + 12u_1u_2^2u_3 + u_2^4$
$A_9 = 4u_0^3u_9 + 12u_0^2u_1u_8 + 12u_0^2u_2u_7 + 12u_0^2u_3u_6 + 12u_0^2u_4u_5 + 12u_0u_1^2u_7 + 24u_0u_1u_2u_6 + 24u_0u_1u_3u_5$ $+ 12u_0u_1u_4^2 + 12u_0u_2^2u_5 + 24u_0u_2u_3u_4 + 4u_0u_3^3 + 4u_1^3u_6 + 12u_1^2u_2u_5 + 12u_1^2u_3u_4 + 12u_1u_2^2u_4$ $+ 12u_1u_2u_3^2 + 4u_2^3u_3$

Table 3. First ten Adomian polynomials for $N(u) = e^u$,

$N(u) = e^u$
$A_0 = e^{u_0}$
$A_1 = u_1e^{u_0}$
$A_2 = \dfrac{(u_1^2 + 2u_2)e^{u_0}}{2}$
$A_3 = \dfrac{(u_1^3 + 6u_1u_2 + 6u_3)e^{u_0}}{6}$
$A_4 = \dfrac{(u_1^4 + 12u_1^2u_2 + 24u_1u_3 + 12u_2^2 + 24u_4)e^{u_0}}{24}$

$A_5 = \dfrac{\left(u_1^5 + 20u_1^3u_2 + 60u_1^2u_3 + 60u_1u_2^2 + 120u_1u_4 + 120u_2u_3 + 120u_5\right)e^{u_0}}{120}$
$A_6 = \dfrac{\left(u_1^6 + 30u_1^4u_2 + 120u_1^3u_3 + 180u_1^2u_2^2 + 360u_1^2u_4 + 720u_1u_2u_3 + 720u_1u_5 + 120u_2^3 + 720u_2u_4 + 360u_3^2 + 720u_6\right)e^{u_0}}{720}$
$A_7 = \dfrac{\begin{pmatrix} u_1^7 + 42u_1^5u_2 + 210u_1^4u_3 + 420u_1^3u_2^2 + 840u_1^3u_4 + 2520u_1^2u_2u_3 + 2520u_1^2u_5 + 840u_1u_2^3 \\ +5040u_1u_2u_4 + 2520u_1u_3^2 + 5040u_1u_6 + 2520u_2^2u_3 + 5040u_2u_5 + 5040u_3u_4 + 5040u_7 \end{pmatrix} e^{u_0}}{5040}$
$A_8 = \dfrac{\begin{pmatrix} u_1^8 + 56u_1^6u_2 + 336u_1^5u_3 + 840u_1^4u_2^2 + 1680u_1^4u_4 + 6720u_1^3u_2u_3 + 6720u_1^3u_5 + 3360u_1^2u_2^3 \\ +20160u_1^2u_2u_4 + 10080u_1^2u_3^2 + 20160u_1^2u_6 + 20160u_1u_2^2u_3 + 40320u_1u_2u_5 + 40320u_1u_3u_4 \\ +40320u_1u_7 + 1680u_2^4 + 20160u_2^2u_4 + 20160u_2u_3^2 + 40320u_2u_6 + 40320u_3u_5 + 20160u_4^2 \\ +40320u_8 \end{pmatrix} e^{u_0}}{40320}$
$A_9 = \dfrac{\begin{pmatrix} u_1^9 + 72u_1^7u_2 + 504u_1^6u_3 + 1512u_1^5u_2^2 + 3024u_1^5u_4 + 15120u_1^4u_2u_3 + 15120u_1^4u_5 + 10080u_1^3u_2^3 \\ +60480u_1^3u_2u_4 + 30240u_1^3u_3^2 + 60480u_1^3u_6 + 90720u_1^2u_2^2u_3 + 181440u_1^2u_2u_5 + 181440u_1^2u_3u_4 \\ +181440u_1^2u_7 + 15120u_1u_2^4 + 181440u_1u_2^2u_4 + 181440u_1u_2u_3^2 + 362880u_1u_2u_6 + 362880u_1u_3u_5 \\ +181440u_1u_4^2 + 362880u_1u_8 + 60480u_2^3u_3 + 181440u_2^2u_5 + 362880u_2u_3u_4 + 362880u_2u_7 \\ +60480u_3^3 + 362880u_3u_6 + 362880u_4u_5 + 362880u_9 \end{pmatrix} e^{u_0}}{362880}$

Table 4. First ten Adomian polynomials for $N(u) = \dfrac{au}{b+u}$

$N(u) = \dfrac{au}{b+u}$
$A_0 = \dfrac{au_0}{b+u_0}$
$A_1 = \dfrac{abu_1}{(b+u_0)^2}$
$A_2 = \dfrac{ab\left(bu_2 + u_0u_2 - u_1^2\right)}{(b+u_0)^3}$
$A_3 = \dfrac{ab}{(b+u_0)^4}\left(b^2u_3 + 2bu_0u_3 - 2bu_1u_2 + u_0^2u_3 - 2u_0u_1u_2 + u_1^3\right)$

$$A_4 = \frac{ab}{(b+u_0)^5}\left(\begin{array}{l} b^3u_4 + 3b^2u_0u_4 - 2b^2u_1u_3 - b^2u_2^2 + 3bu_0^2u_4 - 4bu_0u_1u_3 - 2bu_0u_2^2 \\ +3bu_1^2u_2 + u_0^3u_4 - 2u_0^2u_1u_3 - u_0^2u_2^2 + 3u_0u_1^2u_2 - u_1^4 \end{array}\right)$$

$$A_5 = \frac{ab}{(b+u_0)^6}\left(\begin{array}{l} b^4u_5 + 4b^3u_0u_5 - 2b^3u_1u_4 - 2b^3u_2u_3 + 6b^2u_0^2u_5 - 6b^2u_0u_1u_4 \\ -6b^2u_0u_2u_3 + 3b^2u_1^2u_3 + 3b^2u_1u_2^2 + 4bu_0^3u_5 - 6bu_0^2u_1u_4 \\ -6bu_0^2u_2u_3 + 6bu_0u_1^2u_3 + 6bu_0u_1u_2^2 - 4bu_1^3u_2 + u_0^4u_5 - 2u_0^3u_1u_4 \\ -2u_0^3u_2u_3 + 3u_0^2u_1^2u_3 + 3u_0^2u_1u_2^2 - 4u_0u_1^3u_2 + u_1^5 \end{array}\right)$$

$$A_6 = \frac{ab}{(b+u_0)^7}\left(\begin{array}{l} b^5u_6 + 5b^4u_0u_6 - 2b^4u_1u_5 - 2b^4u_2u_4 - b^4u_3^2 + 10b^3u_0^2u_6 - 8b^3u_0u_1u_5 \\ -8b^3u_0u_2u_4 - 4b^3u_0u_3^2 + 3b^3u_1^2u_4 + 6b^3u_1u_2u_3 + b^3u_2^3 + 10b^2u_0^3u_6 \\ -12b^2u_0^2u_1u_5 - 12b^2u_0^2u_2u_4 - 6b^2u_0^2u_3^2 + 9b^2u_0u_1^2u_4 + 18b^2u_0u_1u_2u_3 \\ +3b^2u_0u_2^3 - 4b^2u_1^3u_3 - 6b^2u_1^2u_2^2 + 5bu_0^4u_6 - 8bu_0^3u_1u_5 - 8bu_0^3u_2u_4 \\ -4bu_0^3u_3^2 + 9bu_0^2u_1^2u_4 + 18bu_0^2u_1u_2u_3 + 3bu_0^2u_2^3 - 8bu_0u_1^3u_3 - 12bu_0u_1^2u_2^2 \\ +5bu_1^4u_2 + u_0^5u_6 - 2u_0^4u_1u_5 - 2u_0^4u_2u_4 - u_0^4u_3^2 + 3u_0^3u_1^2u_4 + 6u_0^3u_1u_2u_3 \\ +u_0^3u_2^3 - 4u_0^2u_1^3u_3 - 6u_0^2u_1^2u_2^2 + 5u_0u_1^4u_2 - u_1^6 \end{array}\right)$$

$$A_7 = \frac{ab}{(b+u_0)^8}\left(\begin{array}{l} b^6u_7 + 6b^5u_0u_7 - 2b^5u_1u_6 - 2b^5u_2u_5 - 2b^5u_3u_4 + 15b^4u_0^2u_7 - 10b^4u_0u_1u_6 \\ -10b^4u_0u_2u_5 - 10b^4u_0u_3u_4 + 3b^4u_1^2u_5 + 6b^4u_1u_2u_4 + 3b^4u_1u_3^2 + 3b^4u_2^2u_3 \\ +20b^3u_0^3u_7 - 20b^3u_0^2u_1u_6 - 20b^3u_0^2u_2u_5 - 20b^3u_0^2u_3u_4 + 12b^3u_0u_1^2u_5 \\ +24b^3u_0u_1u_2u_4 + 12b^3u_0u_1u_3^2 + 12b^3u_0u_2^2u_3 - 4b^3u_1^3u_4 - 12b^3u_1^2u_2u_3 \\ -4b^3u_1u_2^3 + 15b^2u_0^4u_7 - 20b^2u_0^3u_1u_6 - 20b^2u_0^3u_2u_5 - 20b^2u_0^3u_3u_4 + 18b^2u_0^2u_1^2u_5 \\ +36b^2u_0^2u_1u_2u_4 + 18b^2u_0^2u_1u_3^2 + 18b^2u_0^2u_2^2u_3 - 12b^2u_0u_1^3u_4 - 36b^2u_0u_1^2u_2u_3 \\ -12b^2u_0u_1u_2^3 + 5b^2u_1^4u_3 + 10b^2u_1^3u_2^2 + 6bu_0^5u_7 - 10bu_0^4u_1u_6 - 10bu_0^4u_2u_5 \\ -10bu_0^4u_3u_4 + 12bu_0^3u_1^2u_5 + 24bu_0^3u_1u_2u_4 + 12bu_0^3u_1u_3^2 + 12bu_0^3u_2^2u_3 \\ -12bu_0^2u_1^3u_4 - 36bu_0^2u_1^2u_2u_3 - 12bu_0^2u_1u_2^3 + 10bu_0u_1^4u_3 + 20bu_0u_1^3u_2^2 \\ -6bu_1^5u_2 + u_0^6u_7 - 2u_0^5u_1u_6 - 2u_0^5u_2u_5 - 2u_0^5u_3u_4 + 3u_0^4u_1^2u_5 + 6u_0^4u_1u_2u_4 \\ +3u_0^4u_1u_3^2 + 3u_0^4u_2^2u_3 - 4u_0^3u_1^3u_4 - 12u_0^3u_1^2u_2u_3 - 4u_0^3u_1u_2^3 + 5u_0^2u_1^4u_3 \\ +10u_0^2u_1^3u_2^2 - 6u_0u_1^5u_2 + u_1^7 \end{array}\right)$$

$$
A_8 = \frac{ab}{\left(b+u_0\right)^9}\left(\begin{aligned}
&b^7u_8+7b^6u_0u_8-2b^6u_1u_7-2b^6u_2u_6-2b^6u_3u_5-b^6u_4^2+21b^5u_0^2u_8-12b^5u_0u_1u_7\\
&-12b^5u_0u_2u_6-12b^5u_0u_3u_5-6b^5u_0u_4^2+3b^5u_1^2u_6+6b^5u_1u_2u_5+6b^5u_1u_3u_4\\
&+3b^5u_2^2u_4+3b^5u_2u_3^2+35b^4u_0^3u_8-30b^4u_0^2u_1u_7-30b^4u_0^2u_2u_6-30b^4u_0^2u_3u_5\\
&-15b^4u_0^2u_4^2+15b^4u_0u_1^2u_6+30b^4u_0u_1u_2u_5+30b^4u_0u_1u_3u_4+15b^4u_0u_2^2u_4\\
&+15b^4u_0u_2u_3^2-4b^4u_1^3u_5-12b^4u_1^2u_2u_4-6b^4u_1^2u_3^2-12b^4u_1u_2^2u_3-b^4u_2^4\\
&+35b^3u_0^4u_8-40b^3u_0^3u_1u_7-40b^3u_0^3u_2u_6-40b^3u_0^3u_3u_5-20b^3u_0^3u_4^2+30b^3u_0^2u_1^2u_6\\
&+60b^3u_0^2u_1u_2u_5+60b^3u_0^2u_1u_3u_4+30b^3u_0^2u_2^2u_4+30b^3u_0^2u_2u_3^2-16b^3u_0u_1^3u_5\\
&-48b^3u_0u_1^2u_2u_4-24b^3u_0u_1^2u_3^2-48b^3u_0u_1u_2^2u_3-4b^3u_0u_2^4+5b^3u_1^4u_4\\
&+20b^3u_1^3u_2u_3+10b^3u_1^2u_2^3+21b^2u_0^5u_8-30b^2u_0^4u_1u_7-30b^2u_0^4u_2u_6-30b^2u_0^4u_3u_5\\
&-15b^2u_0^4u_4^2+30b^2u_0^3u_1^2u_6+60b^2u_0^3u_1u_2u_5+60b^2u_0^3u_1u_3u_4+30b^2u_0^3u_2^2u_4\\
&+30b^2u_0^3u_2u_3^2-24b^2u_0^2u_1^3u_5-72b^2u_0^2u_1^2u_2u_4-36b^2u_0^2u_1^2u_3^2-72b^2u_0^2u_1u_2^2u_3\\
&-6b^2u_0^2u_2^4+15b^2u_0u_1^4u_4+60b^2u_0u_1^3u_2u_3+30b^2u_0u_1^2u_2^3-6b^2u_1^5u_3-15b^2u_1^4u_2^2\\
&+7bu_0^6u_8-12bu_0^5u_1u_7-12bu_0^5u_2u_6-12bu_0^5u_3u_5-6bu_0^5u_4^2+15bu_0^4u_1^2u_6\\
&+30bu_0^4u_1u_2u_5+30bu_0^4u_1u_3u_4+15bu_0^4u_2^2u_4+15bu_0^4u_2u_3^2-16bu_0^3u_1^3u_5\\
&-48bu_0^3u_1^2u_2u_4-24bu_0^3u_1^2u_3^2-48bu_0^3u_1u_2^2u_3-4bu_0^3u_2^4+15bu_0^2u_1^4u_4\\
&+60bu_0^2u_1^3u_2u_3+30bu_0^2u_1^2u_2^3-12bu_0u_1^5u_3-30bu_0u_1^4u_2^2+7bu_1^6u_2+u_0^7u_8\\
&-2u_0^6u_1u_7-2u_0^6u_2u_6-2u_0^6u_3u_5-u_0^6u_4^2+3u_0^5u_1^2u_6+6u_0^5u_1u_2u_5+6u_0^5u_1u_3u_4\\
&+3u_0^5u_2^2u_4+3u_0^5u_2u_3^2-4u_0^4u_1^3u_5-12u_0^4u_1^2u_2u_4-6u_0^4u_1^2u_3^2-12u_0^4u_1u_2^2u_3-u_0^4u_2^4\\
&+5u_0^3u_1^4u_4+20u_0^3u_1^3u_2u_3+10u_0^3u_1^2u_2^3-6u_0^2u_1^5u_3-15u_0^2u_1^4u_2^2+7u_0u_1^6u_2-u_1^8
\end{aligned}\right)
$$

$$
A_9 = \frac{ab}{(b+u_0)^{10}} \left(\begin{array}{l}
b^8u_9 + 8b^7u_0u_9 - 2b^7u_1u_8 - 2b^7u_2u_7 - 2b^7u_3u_6 - 2b^7u_4u_5 + 28b^6u_0^2u_9 \\
-14b^6u_0u_1u_8 - 14b^6u_0u_2u_7 - 14b^6u_0u_3u_6 - 14b^6u_0u_4u_5 + 3b^6u_1^2u_7 \\
+6b^6u_1u_2u_6 + 6b^6u_1u_3u_5 + 3b^6u_1u_4^2 + 3b^6u_2^2u_5 + 6b^6u_2u_3u_4 + b^6u_3^3 + 56b^5u_0^3u_9 \\
-42b^5u_0^2u_1u_8 - 42b^5u_0^2u_2u_7 - 42b^5u_0^2u_3u_6 - 42b^5u_0^2u_4u_5 + 18b^5u_0u_1^2u_7 \\
+36b^5u_0u_1u_2u_6 + 36b^5u_0u_1u_3u_5 + 18b^5u_0u_1u_4^2 + 18b^5u_0u_2^2u_5 + 36b^5u_0u_2u_3u_4 \\
+6b^5u_0u_3^3 - 4b^5u_1^3u_6 - 12b^5u_1^2u_2u_5 - 12b^5u_1^2u_3u_4 - 12b^5u_1u_2^2u_4 - 12b^5u_1u_2u_3^2 \\
-4b^5u_2^3u_3 + 70b^4u_0^4u_9 - 70b^4u_0^3u_1u_8 - 70b^4u_0^3u_2u_7 - 70b^4u_0^3u_3u_6 - 70b^4u_0^3u_4u_5 \\
+45b^4u_0^2u_1^2u_7 + 90b^4u_0^2u_1u_2u_6 + 90b^4u_0^2u_1u_3u_5 + 45b^4u_0^2u_1u_4^2 + 45b^4u_0^2u_2^2u_5 \\
+90b^4u_0^2u_2u_3u_4 + 15b^4u_0^2u_3^3 - 20b^4u_0u_1^3u_6 - 60b^4u_0u_1^2u_2u_5 - 60b^4u_0u_1^2u_3u_4 \\
-60b^4u_0u_1u_2^2u_4 - 60b^4u_0u_1u_2u_3^2 - 20b^4u_0u_2^3u_3 + 5b^4u_1^4u_5 + 20b^4u_1^3u_2u_4 \\
+10b^4u_1^3u_3^2 + 30b^4u_1^2u_2^2u_3 + 5b^4u_1u_2^4 + 56b^3u_0^5u_9 - 70b^3u_0^4u_1u_8 - 70b^3u_0^4u_2u_7 \\
-70b^3u_0^4u_3u_6 - 70b^3u_0^4u_4u_5 + 60b^3u_0^3u_1^2u_7 + 120b^3u_0^3u_1u_2u_6 + 120b^3u_0^3u_1u_3u_5 \\
+60b^3u_0^3u_1u_4^2 + 60b^3u_0^3u_2^2u_5 + 120b^3u_0^3u_2u_3u_4 + 20b^3u_0^3u_3^3 - 40b^3u_0^2u_1^3u_6 \\
-120b^3u_0^2u_1^2u_2u_5 - 120b^3u_0^2u_1^2u_3u_4 - 120b^3u_0^2u_1u_2^2u_4 - 120b^3u_0^2u_1u_2u_3^2 \\
-40b^3u_0^2u_2^3u_3 + 20b^3u_0u_1^4u_5 + 80b^3u_0u_1^3u_2u_4 + 40b^3u_0u_1^3u_3^2 + 120b^3u_0u_1^2u_2^2u_3 \\
+20b^3u_0u_1u_2^4 - 6b^3u_1^5u_4 - 30b^3u_1^4u_2u_3 - 20b^3u_1^3u_2^3 + 28b^2u_0^6u_9 - 42b^2u_0^5u_1u_8 \\
-42b^2u_0^5u_2u_7 - 42b^2u_0^5u_3u_6 - 42b^2u_0^5u_4u_5 + 45b^2u_0^4u_1^2u_7 + 90b^2u_0^4u_1u_2u_6 \\
+90b^2u_0^4u_1u_3u_5 + 45b^2u_0^4u_1u_4^2 + 45b^2u_0^4u_2^2u_5 + 90b^2u_0^4u_2u_3u_4 + 15b^2u_0^4u_3^3 \\
-40b^2u_0^3u_1^3u_6 - 120b^2u_0^3u_1^2u_2u_5 - 120b^2u_0^3u_1^2u_3u_4 - 120b^2u_0^3u_1u_2^2u_4 \\
-120b^2u_0^3u_1u_2u_3^2 - 40b^2u_0^3u_2^3u_3 + 30b^2u_0^2u_1^4u_5 + 120b^2u_0^2u_1^3u_2u_4 + 60b^2u_0^2u_1^3u_3^2 \\
+180b^2u_0^2u_1^2u_2^2u_3 + 30b^2u_0^2u_1u_2^4 - 18b^2u_0u_1^5u_4 - 90b^2u_0u_1^4u_2u_3 - 60b^2u_0u_1^3u_2^3 \\
+7b^2u_1^6u_3 + 21b^2u_1^5u_2^2 + 8bu_0^7u_9 - 14bu_0^6u_1u_8 - 14bu_0^6u_2u_7 - 14bu_0^6u_3u_6 \\
-14bu_0^6u_4u_5 + 18bu_0^5u_1^2u_7 + 36bu_0^5u_1u_2u_6 + 36bu_0^5u_1u_3u_5 + 18bu_0^5u_1u_4^2 \\
+18bu_0^5u_2^2u_5 + 36bu_0^5u_2u_3u_4 + 6bu_0^5u_3^3 - 20bu_0^4u_1^3u_6 - 60bu_0^4u_1^2u_2u_5 \\
-60bu_0^4u_1^2u_3u_4 - 60bu_0^4u_1u_2^2u_4 - 60bu_0^4u_1u_2u_3^2 - 20bu_0^4u_2^3u_3 + 20bu_0^3u_1^4u_5 \\
+80bu_0^3u_1^3u_2u_4 + 40bu_0^3u_1^3u_3^2 + 120bu_0^3u_1^2u_2^2u_3 + 20bu_0^3u_1u_2^4 - 18bu_0^2u_1^5u_4 \\
-90bu_0^2u_1^4u_2u_3 - 60bu_0^2u_1^3u_2^3 + 14bu_0u_1^6u_3 + 42bu_0u_1^5u_2^2 - 8bu_1^7u_2 + u_0^8u_9 \\
-2u_0^7u_1u_8 - 2u_0^7u_2u_7 - 2u_0^7u_3u_6 - 2u_0^7u_4u_5 + 3u_0^6u_1^2u_7 + 6u_0^6u_1u_2u_6 \\
+6u_0^6u_1u_3u_5 + 3u_0^6u_1u_4^2 + 3u_0^6u_2^2u_5 + 6u_0^6u_2u_3u_4 + u_0^6u_3^3 - 4u_0^5u_1^3u_6 \\
-12u_0^5u_1^2u_2u_5 - 12u_0^5u_1^2u_3u_4 - 12u_0^5u_1u_2^2u_4 - 12u_0^5u_1u_2u_3^2 - 4u_0^5u_2^3u_3 \\
+5u_0^4u_1^4u_5 + 20u_0^4u_1^3u_2u_4 + 10u_0^4u_1^3u_3^2 + 30u_0^4u_1^2u_2^2u_3 + 5u_0^4u_1u_2^4 - 6u_0^3u_1^5u_4 \\
-30u_0^3u_1^4u_2u_3 - 20u_0^3u_1^3u_2^3 + 7u_0^2u_1^6u_3 + 21u_0^2u_1^5u_2^2 - 8u_0u_1^7u_2 + u_1^9
\end{array} \right)
$$

Conclusion

The proposed algorithm intended to calculate the Adomian polynomials enjoys a great deal of simplicity, succinctness and efficiency. As it is converted to a function file in Python, it can easily be integrated into any code dealing with solutions of nonlinear functional equations and be called anywhere in the program to compute the desired Adomian polynomial component of an interested nonlinearity. By taking advantage of the symbolic infrastructure of the *SymPy* library in Python, we have managed to shorten the code considerably. The authors are working on the application of this computer code to tackle nonlinear equations in chemical engineering, especially in the field of equations of state in thermodynamics.

References

[1] Adomian G, Rach R. On linear and nonlinear integro-differential equations Journal of Mathematical Analysis and Applications. 1986;113(1):199-201. DOI: https://doi.org/10.1016/0022-247X(86)90343-4

[2] Turkyilmazoglu M. A reliable convergent Adomian decomposition method for heat transfer through extended surfaces. International Journal of Numerical Methods for Heat & Fluid Flow. 2018;28(11):2551-2566. DOI: https://doi.org/10.1108/HFF-01-2018-0003

[3] Saifi H, Sari MR, Kezzar M, Ghazvini M, Sharifpur M, Sadeghzadeh M. Heat transfer through converging-diverging channels using Adomian decomposition method. Engineering Applications of Computational Fluid Mechanics. 2020;14(1):1373-1384.
DOI: https://doi.org/10.1080/19942060.2020.1830857

[4] Birajdar GA. A new approach for non-linear fractional heat transfer model by Adomian decomposition method. Mathematical Analysis and Computing. 2021;344:333-343. DOI: https://doi.org/10.1007/978-981-33-4646-8_28

[5] Acharya S, Nayak B, Mishra S, Jena S. Adomian decomposition method for the MHD flow of a viscous fluid with the influence of dissipative heat energy. Heat Transfer. 2020;49(8):4612-4625. DOI: https://doi.org/10.1002/htj.21844

[6] Shamshuddin M, Mishra S, Beg OA, Kadir A. Adomian decomposition method simulation of Von Karman swirling bioconvection nanofluid flow. Journal of Central South University: Science & Technology of Mining and Metallurgy. 2019;26(10):2797-2813. DOI: https://doi.org/10.1007/s11771-019-4214-4

[7] Ostadhossein R, Hoseinzadeh S. The solution of Pennes' bio-heat equation with a convection term and nonlinear specific heat capacity using Adomian decomposition. Journal of Thermal Analysis and Calorimetry. 2022;147(22):12739-12747. DOI: https://doi.org/10.1007/s10973-022-11445-x

[8] Yusuf A, Gupa MI, Sayeed NH, Bolarin G. Thermo-diffusion effects of a stagnation point flow in a nanofluid with convection using the Adomian decomposition method. Covenant Journal of Physical and Life Sciences. 2021;9(2).

[9] Ajibade AO, Gambo JJ. Adomian decomposition method to magnetohydrodynamics natural convection heat generating/absorbing slip flow through a porous medium. Multidiscipline Modeling in Materials and Structures. 2019;15(3):673-684. DOI: https://doi.org/10.1108/MMMS-08-2018-0153

[10] Putranto YW, Mungkasi S. Adomian decomposition method for solving the population dynamics model of two species. Journal of Physics: Conference Series. 2017;795. DOI: https://doi.org/10.1088/1742-6596/795/1/012045

[11] Yunus AO, Olayiwola MO, Adedokun KA, Adedeji JA, Alaje IA. Mathematical analysis of fractional-order Caputo's derivative of coronavirus disease model via Laplace Adomian decomposition method. Beni-Suef University Journal of Basic and Applied Sciences. 2022;11(1):144. https://doi.org/10.1186/s43088-022-003269

[12] Ullah A, Ullah A, Ahmad S, Ahmad I, Akgül A. On solutions of fuzzy fractional order complex population dynamical model. Numerical Methods for Partial Differential Equations. 2020;1-21. DOI: https://doi.org/10.1002/num.22654

[13] Lede Y, Mungkasi S. Adomian decomposition method used to solve a SIR epidemic model of dengue fever. In: Proceedings of the 2nd International Conference of Science and Technology for the Internet of Things, ICSTI 2019; September 3rd, 2019; Yogyakarta, Indonesia.

[14] Gahgah M, Sari MR, Kezzar M, Eid MR. Duan–Rach modified Adomian decomposition method (DRMA) for viscoelastic fluid flow between nonparallel plane walls. The European Physical Journal Plus. 2020;135(2):250. DOI: https://doi.org/10.1140/epjp/s13360-020-00250-w

[15] Ul Ain Q, Zaheer M. An analysis of an underground water flow using Adomian decomposition method. Water Conservation & Management. 2019;3(1):27-29.
DOI: https://doi.org/10.26480/wcm.01.2019.27.29

[16] Mahmood S, Shah R, Khan H, Arif M. Laplace Adomian decomposition method for multi-dimensional time fractional model of Navier-Stokes equation. Symmetry. 2019;11(2):149. DOI: https://doi.org/10.3390/sym11020149

[17] Momani S, Odibat Z. Analytical solution of a time-fractional Navier–Stokes equation by Adomian decomposition method. Applied Mathematics and Computation. 2006;177(2):488-494. DOI: https://doi.org/10.1016/j.amc.2005.11.025

[18] Fatoorehchi H, Abolghasemi H, Rach R. An accurate explicit form of the Hankinson–Thomas–Phillips correlation for prediction of the natural gas compressibility factor. Journal of Petroleum Science and Engineering. 2014;117:46-53. DOI: https://doi.org/10.1016/j.petrol.2014.03.004

[19] Fatoorehchi H, Abolghasemi H. Approximating the minimum reflux ratio of multicomponent distillation columns based on the Adomian decomposition method Journal of the Taiwan Institute of Chemical Engineers. 2014;45(3):880-886. DOI: https://doi.org/10.1016/j.jtice.2013.09.032

[20] Fatoorehchi H, Rach R, Sakhaeinia H. Explicit Frost- Kalkwarf type equations for calculation of vapor pressure of liquids from triple to critical point by the Adomian decomposition method. The Canadian Journal of Chemical Engineering. 2017;95(11):2199-2208.
DOI: https://doi.org/10.1002/cjce.22853

[21] Fatoorehchi H, Abolghasemi H, Rach R. A new parametric algorithm for isothermal flash calculations by the Adomian decomposition of Michaelis–Menten type nonlinearities. Fluid Phase Equilibria. 2015;395:44-50. DOI: https://doi.org/10.1016/j.fluid.2015.03.024

[22] Fernandes PR. Fast dynamic simulation of distillation columns with local thermodynamic models and Adomian decomposition. IFAC-PapersOnLine. 2021;54(3):383-388.
DOI: https://doi.org/10.1016/j.ifacol.2021.08.272

[23] Rach R, Duan J-S, Wazwaz A-M. Solving coupled Lane–Emden boundary value problems in catalytic diffusion reactions by the Adomian decomposition method. Journal of Mathematical Chemistry. 2014;52:255-267. DOI: https://doi.org/10.1007/s10910-013-0260-6

[24] Fatoorehchi H, Gutman I, Abolghasemi H. Computing graph energy: an alternative approach. Kragujevac Journal of Science. 2014;36:69-78. DOI: https://doi.org/10.5937/KgJSci1436069F

[25] Younker JM. Numerical integration of the chemical rate equations via a discretized Adomian decomposition. Industrial & engineering chemistry research. 2011;50(6):3100-3109.
DOI: https://doi.org/10.1021/ie1008647

[26] Goličnik M. Solution of the extended Michaelis-Menten equation for enzyme kinetics with spontaneous substrate depletion using the Adomian decomposition method. Match Commun Math Comput Chem. 2016;75(3):613-626.

[27] Fatoorehchi H, Gutman I, Abolghasemi H. A combined technique for computation of energy-effect of cycles in conjugated molecules. Journal of Mathematical Chemistry. 2015;53:1113-1125.
DOI: https://doi.org/10.1007/s10910-015-0473-y

[28] Asma M, Othman WAM, Wong BR, Biswas A. Optical soliton perturbation with quadratic-cubic nonlinearity by Adomian decomposition method. Optik. 2018;164:632-641.
DOI: https://doi.org/10.1016/j.ijleo.2018.03.008

[29] Biswas A, Asma M, Alqahtani RT. Optical soliton perturbation with Kerr law nonlinearity by Adomian decomposition method. Optik. 2018;16:253-270.
DOI: https://doi.org/10.1016/j.ijleo.2018.04.025

[30] Rach R, Duan J-S, Wazwaz A-M. On the solution of non-isothermal reaction-diffusion model equations in a spherical catalyst by the modified Adomian method. Chemical Engineering Communications. 2015;202(8):1081-1088.
DOI: https://doi.org/10.1080/00986445.2014.900054

[31] Duan J-S, Rach R, Wazwaz A-M. Steady-state concentrations of carbon dioxide absorbed into phenyl glycidyl ether solutions by the Adomian decomposition method. Journal of Mathematical Chemistry. 2015;53:1054-1067.
DOI: https://doi.org/10.1007/s10910-014-0469-z

[32] Jeyabarathi P, Kannan M, Rajendran L. Approximate analytical solutions of biofilm reactor problem in applied biotechnology. Theoretical Foundations of Chemical Engineering. 2021;55:851-861.
DOI: https://doi.org/10.1134/S0040579521050213

[33] Cheng M, Scott K, Sun Y, Wu B. Explicit solution of nonlinear electrochemical models by the decomposition method. Chemical Engineering & Technology. 2002;25(12):1155-1160.
DOI: https://doi.org/10.1002/1521-4125(20021210)25:12<1155::AID-CEAT1155>3.0.CO;2-A

[34] Madbouly NM, McGhee DF, Roach GF. Adomian's method for Hammerstein integral equations arising from chemical reactor theory. Applied Mathematics and Computation. 2001;117(2-3):241-249.
DOI: https://doi.org/10.1016/S0096-3003(99)00177-0

[35] Biazar J, Pourabd M. A Maple program for computing Adomian polynomials. International Mathematical Forum. 2006;39(1):1919-1924. DOI: https://doi.org/10.13140/RG.2.1.3473.4565

[36] Duan J-S. Recurrence triangle for Adomian polynomials. Applied Mathematics and Computation. 2010;216(4):1235-1241. DOI: https://doi.org/10.1016/j.amc.2010.07.046

[37] Babolian E, Javadi S. New method for calculating Adomian polynomials. Applied Mathematics and Computation. 2004;153(1):253-259. DOI: https://doi.org/10.1016/S0096-3003(03)00629-5

[38] Biazar J, Babolian E, Kember G, Nouri A, Islam R. An alternate algorithm for computing Adomian polynomials in special cases. Applied Mathematics and Computation. 2003;138(2-3):523-529. DOI: https://doi.org/10.1016/S0096-3003(02)00174-1

[39] Wazwaz A-M. A new algorithm for calculating Adomian polynomials for nonlinear operators. Applied Mathematics and Computation. 2000;111(1):33-51. DOI: https://doi.org/10.1016/S0096-3003(99)00063-6

[40] Fatoorehchi H, Abolghasemi H. On calculation of Adomian polynomials by MATLAB. J Appl Comput Sci Math. 2011;5:85-88.

[41] Abbaoi K, Cherruault Y. Convergence of Adomian's method applied to differential equations. Computers & Mathematics with Applications. 1994;28(5):103-109. DOI: https://doi.org/10.1016/0898-1221(94)00144-8

[42] Babolian E, Biazar J. On the order of convergence of Adomian method. Applied Mathematics and Computation. 2002;130(2-3):383-387. DOI: https://doi.org/10.1016/S0096-3003(01)00103-5

Nomenclature	
A_i	*i*-th Adomian polynomial component
d	differentiation operator
E	Banach space
f	known function in E
u_k	*k*-th solution component of functional equation (1)
u	analytical solution of functional equation (1)
$N(u)$	nonlinear operator acting on u
λ	auxiliary variable