Computer Physics Communications 184 (2013) 2136–2144

Contents lists available at SciVerse ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

An iterative HAM approach for nonlinear boundary value problems in a semi-infinite domain

Yinlong Zhao [c], Zhiliang Lin [c], Shijun Liao [a,b,c,d,*]

[a] Department of Mathematics, Shanghai Jiao Tong University, Shanghai 200240, China
[b] State Key Laboratory of Ocean Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
[c] School of Naval Architecture, Ocean and Civil Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
[d] Nonlinear Analysis and Applied Mathematics Research Group (NAAM), King Abdulaziz University, Jeddah, Saudi Arabia

ARTICLE INFO

Article history:
Received 16 October 2012
Received in revised form
11 April 2013
Accepted 19 April 2013
Available online 29 April 2013

Keywords:
Homotopy analysis method
Truncation technique
Iteration technique
Orthonormal functions
Approximate analytical solutions

ABSTRACT

In this paper, we propose an iterative approach to increase the computation efficiency of the homotopy analysis method (HAM), a analytic technique for highly nonlinear problems. By means of the Schmidt–Gram process (Arfken et al., 1985) [15], we approximate the right-hand side terms of high-order linear sub-equations by a finite set of orthonormal bases. Based on this truncation technique, we introduce the Mth-order iterative HAM by using each Mth-order approximation as a new initial guess. It is found that the iterative HAM is much more efficient than the standard HAM without truncation, as illustrated by three nonlinear differential equations defined in an infinite domain as examples. This work might greatly improve the computational efficiency of the HAM and also the Mathematica package BVPh for nonlinear BVPs.

1. Introduction

Ordinary differential equations (ODEs) are widely used in mathematics, science and engineering. The so-called initial value problems (IVPs) specify some restrictions only at a single point. This kind of problem is often solved by means of a numerical approach based on integration, such as the Runge–Kutta method. However, in many cases, a solution is described in a more complicated way. The so-called boundary value problems (BVPs) specify some restrictions at more than one point. Unlike IVPs, BVPs might have multiple solutions, or may satisfy some restrictions at infinity. Hence, BVPs are often more difficult to solve than IVPs.

Thanks to the development of mathematical software, such as Maple, Mathematica and MATLAB, some packages have been developed to solve nonlinear BVPs, such as BVP4c, Chebfun and BVPh. The famous MATLAB package BVP4c [1,2] implements a collocation method, instead of a shooting method. The collocation polynomial provides a C^1-continuous solution that is fourth-order accurate uniformly in $[a, b]$. Mesh selection and error control are based on the residual of the continuous solution. However, it is not easy for BVP4c to resolve the singularity in governing equations and/or boundary conditions. Besides, BVP4c regards an infinite interval as a kind of singularity and replaces it by a finite one: this leads to the additional inaccuracy and uncertainty of solutions.

Chebfun [3,4] is a collection of algorithms on the "chebfun" objects written in MATLAB. It aims to combine the feel of symbolics with the speed of numerics. The basis of Chebfun is Chebyshev expansions, fast Fourier transform, barycentric interpolation and so on. The idea "computing numerically with functions instead of numbers" [4] behind Chebfun makes it have the potential to handle unbounded domains and singularities in a easy way. Although linear differential equations can be solved in a single step by Chebfun without iteration, only a few examples for nonlinear differential equations are given [3,4]. Actually, Chebfun uses Newton's iteration to solve nonlinear problems. However, it is well known that the convergence of Newton's iteration is strongly dependent upon initial guesses and thus is not guaranteed. Besides, Chebfun searches for multiple solutions of nonlinear differential equations by using different guess approximations. However, it is not very clear how to choose these different guess approximations.

Based on the homotopy in topology, the so-called homotopy analysis method (HAM) has been proposed by Liao [5–10] to gain analytic approximations of highly nonlinear problems. The HAM has some advantages over other traditional analytic approximation methods. First, unlike perturbation techniques, the HAM is independent of small/large physical parameters, and thus is valid in

* Correspondence to: Shanghai Jiao Tong University, Shanghai 200240, China. Tel.: +86 21 34204445.
E-mail address: sjliao@sjtu.edu.cn (S. Liao).

0010-4655/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.cpc.2013.04.009

more general cases. Besides, different from all other analytic techniques, the HAM provides us with a convenient way to guarantee the convergence of series solution. Furthermore, the HAM provides extremely large freedom to choose the initial guess and equation type of linear sub-problems. It is found [7,5] that lots of nonlinear BVPs in science, engineering and finance can be solved conveniently by means of the HAM, no matter whether the interval is finite or not.

Based on the HAM, a Mathematica package BVPh 1.0 for nonlinear boundary value/eigenvalue problems with singularity and/or multi-point boundary conditions was issued by Liao [5] in May 2012, which is free and available online (http://numericaltank.sjtu.edu.cn/BVPh.htm). Its aim is to develop a kind of analytic tool for as many nonlinear BVPs as possible such that multiple solutions of highly nonlinear BVPs can be conveniently found out, and that the infinite interval and singularities of governing equations and/or boundary conditions at multi-points can be easily resolved. As illustrated by Liao [5], BVPh 1.0 is valid for lots of nonlinear BVPs and thus is a useful tool in practice.

Based on the HAM, the Maple package NOPH [11] for periodically oscillating systems of center and limit cycle types is currently being developed, which delivers accurate approximations of frequency, mean of motion and amplitude of oscillation automatically. NOPH combines Wu's elimination method and the homotopy analysis method (HAM). It is free and available online (http://numericaltank.sjtu.edu.cn/NOPH.htm). Unlike BVPh, NOPH is only applicable to periodic oscillations. This illustrates the general validity of the HAM for nonlinear problems.

However, based on Mathematica, BVPh suffers from the common problem in symbolic computation: the intermediate expressions "swell" so fast that it needs more and more CPU time to gain high-order approximations. To obtain accurate approximations in a short time, Yabushita et al. [12] suggested an optimal HAM approach by minimizing the squared residual of governing equations. Niu et al. [13] and Liao [14] further developed some optimal approaches based on the HAM. Especially, Liao [5] suggested an iterative analytic technique, in which the initial guess is replaced by truncated analytic approximations over and over. For differential equations in a finite interval, trigonometric functions (or polynomial functions) are usually used to express solutions. In this case, orthonormal trigonometric functions (or Chebyshev polynomials) are used to approximate the right-hand side of high-order deformation equations (see [5] for details). As illustrated by Liao [5], accurate approximations can be gained more efficiently by means of the HAM in this way.

In this paper, we generalize this kind of iterative analytic approach for nonlinear differential equations in a semi-infinite interval by means of the Schmidt–Gram process [15]. The orthonormal functions derived from the Schmidt–Gram process are used to approximate the right-hand side of high-order deformation equations during computation, and the Mth-order approximation is then used as a new initial guess to gain a better approximation. In this way, the computational efficiency of the HAM can be greatly improved, as illustrated in this paper.

This paper is arranged as follows. In Section 2 the basic ideas of the standard HAM are briefly described. In Section 3 the ideas of the truncation and iteration technique based on the Schmidt–Gram process [15] are explained, and the HAM-based iterative analytic approach is proposed. Section 4 gives some key points of the implementation of this HAM-based iterative approach. In Section 5 three different boundary-layer flow problems are used as examples to demonstrate the validity and efficiency of the iteration approach. Some conclusions and discussions are made in the final section.

2. Basic ideas of the HAM

Consider a system of S equations,

$$\mathcal{N}_j[u_1(x), \ldots, u_S(x)] = 0, \quad j = 1, \ldots, S, x \in \Omega, \tag{1}$$

where $\mathcal{N}_j$ is a nonlinear differential operator, x and u_j are the independent and dependent variables, and Ω denotes the solution interval, respectively.

The HAM is based on the concept of homotopy in topology, which describes a continuous deformation or variation in general. Let $q \in [0, 1]$ denote an embedding parameter for homotopy, $c_{0,j} \neq 0$ an auxiliary parameter (called a convergence-control parameter), $H_j(x)$ an auxiliary function, $\mathcal{L}_j$ an auxiliary linear operator, and $u_{j,0}(x)$ an initial guess of the solution $u_j(x)$, respectively. To build a continuous deformation (or variation) from the initial guess $u_{j,0}(x)$ to the true solution $u_j(x)$, we construct the so-called zeroth-order deformation equations

$$\begin{aligned}&(1-q)\mathcal{L}_j[\phi_j(x;q) - u_{j,0}(x)]\\ &\quad = q\, c_{0,j}\, H_j(x)\, \mathcal{N}_j[\phi_1(x;q), \ldots, \phi_S(x;q)].\end{aligned} \tag{2}$$

Obviously, we have the solution $\phi_j(x; 0) = u_{j,0}(x)$ when $q = 0$, and $\phi_j(x; 1) = u_j(x)$ when $q = 1$, respectively. In other words, $\phi_j(x; q)$ denotes a continuous variation from the initial guess $u_{j,0}(x)$ to the solution $u_j(x)$, as q increases from 0 to 1. Then, we can expand $\phi_j(x; q)$ in Taylor series with respect to the embedding parameter q, i.e.

$$\phi_j(x;q) = u_{j,0}(x) + \sum_{k=1}^{+\infty} u_{j,k}(x) q^k. \tag{3}$$

It should be emphasized that we have great freedom to choose the auxiliary linear operator $\mathcal{L}_j$, the initial guess $u_{j,0}(x)$ and especially the so-called convergence-control parameter $c_{0,j}$, as pointed out by Liao [16]. Assuming that all of them are properly chosen so that the series (3) converges at $q = 1$, we get the series solution

$$u_j(x) = u_{j,0}(x) + \sum_{k=1}^{+\infty} u_{j,k}(x). \tag{4}$$

The mth-order approximation of $u_j(x)$ is given by

$$u_j(x) \approx U_{j,m}(x) = \sum_{k=0}^{m} u_{j,k}(x). \tag{5}$$

Applying the so-called homotopy-derivative operator [17]

$$D_k(\square) = \frac{1}{k!} \frac{\partial^k \square}{\partial q^k}\bigg|_{q=0} \tag{6}$$

to both sides of Eqs. (2) and using its properties, it is straightforward to derive the kth-order deformation equations ($k \geq 1$)

$$\begin{aligned}&\mathcal{L}_j[u_{j,k}(x) - \chi_k u_{j,k-1}(x)]\\ &\quad = c_{0,j}\, H_j(x)\, R_{j,k-1}(x), \quad j = 1, \ldots, S,\end{aligned} \tag{7}$$

where

$$\begin{aligned}R_{j,k-1}(x) &= D_{k-1}\{\mathcal{N}_j[\phi_1(x;q), \ldots, \phi_S(x;q)]\}\\ &= D_{k-1}\left\{\mathcal{N}_j\left[\sum_{i=0}^{k-1} u_{1,i}(x) q^i, \ldots, \sum_{i=0}^{k-1} u_{S,i}(x) q^i\right]\right\}\end{aligned} \tag{8}$$

and

$$\chi_k = \begin{cases} 0, & k \leq 1,\\ 1, & k > 1.\end{cases} \tag{9}$$

Note that $R_{j,k-1}$ is only dependent upon $u_{j,0}(x), \ldots, u_{j,k-1}(x)$, $j = 1, \ldots, S$. Thus, it is easy to gain $u_{j,k}(x)$, $j = 1, \ldots, S$, by solving

the linear equations (7). The convergence of the series (3) depends upon the initial guess $u_{j,0}(x)$, the auxiliary linear operator $\mathcal{L}_j$, the auxiliary function $H_j(x)$, and especially the convergence-control parameter $c_{0,j}, j = 1, \ldots, S$.

To measure the accuracy of the mth-order approximation $U_{j,m}(x)$, where $j = 1, \ldots, S$, the squared residual for the system (1)

$$E_m = \max_{1\le j\le S} \int_\Omega |\mathcal{N}_j[U_{1,m}, \ldots, U_{S,m}]|^2 dx \tag{10}$$

is defined. Such kind of error (10) has been considered in other works [14,18–21]. It is tied closely to the optimal HAM. The smaller E_m, the more accurate the mth-order approximation $U_{j,m}(x)$. Given the initial guess $u_{j,0}(x)$, the auxiliary linear operator $\mathcal{L}_j$ and the auxiliary function $H_j(x)$, the squared residual E_m is dependent on the convergence-control parameters $c_{0,j}$, where $j = 1, 2, \ldots, S$, whose optimal values are determined by the minimum of E_m. Thus, unlike other analytic methods, the HAM provides us with a convenient way to guarantee the convergence of the series solution. In fact, it is the so-called convergence-control parameter that distinguishes the HAM from all other analytic techniques, as pointed out by Liao [5].

3. Modification of the HAM

However, as the order of approximation increases, it needs in general more and more CPU time to solve the linear high-order deformation equations (7) since, due to the nonlinearity, the number of terms on its right-hand side often grows rapidly. To increase the computational efficiency, we develop a truncation technique (called tHAM) and an iteration approach (called iHAM) in the frame of the HAM. These two techniques can greatly decrease the CPU time, as shown below.

Briefly speaking, the HAM provides us great freedom to choose base functions of the solution and equation type of the linear sub-problems, as pointed out by Liao [7,5]. Using this freedom, one can choose proper base functions and the corresponding auxiliary linear operator, and determine the optimal convergence-control parameter by means of the minimum of the squared residual E_m. Due to the nonlinearity, the right-hand side term $R_{j,k-1}$ of Eqs. (7) becomes more and more complicated so that it often costs most of the CPU time to solve Eqs. (7).

The basic idea of the truncation technique (tHAM) is to approximate $R_{j,k-1}$ by a set of finite base functions before solving Eqs. (7). Suppose that $R_{j,k-1}$ can be expressed by a finite linear combination of linearly independent functions $\varphi_1(x), \varphi_2(x), \ldots, \varphi_n(x)$. Here, "linearly independent" means that

$$\alpha_1\varphi_1 + \alpha_2\varphi_2 + \cdots + \alpha_n\varphi_n = 0 \tag{11}$$

is true only when all α_i are zero. Define a proper inner product

$$\langle \varphi_s, \varphi_l \rangle = \int_\Omega \rho(x)\varphi_s(x)\varphi_l(x)dx, \quad 1 \le s, l \le N \tag{12}$$

in the linear space spanned by $\varphi_1, \ldots, \varphi_N$, where $\rho(x)$ is a weight function and Ω is the interval of interest. Since $\varphi_1, \ldots, \varphi_N$ are not necessarily orthonormal, we can apply the Schmidt–Gram process [15] to them and obtain N *orthonormal functions* $e_1, \ldots, e_N$, which satisfy

$$\langle e_s, e_l \rangle = \delta_{s,l}, \quad 1 \le s, l \le N, \tag{13}$$

where $\delta_{s,l}$ is the Kronecker delta. Therefore, as long as $R_{j,k-1}$ is obtained, we first approximate it by the orthonormal bases $e_1, \ldots, e_N$ so that the right-hand side of Eq. (7) contains only N terms. Mathematically speaking, we replace $R_{j,k-1}$ by its approximation

$$R_{j,k-1} \approx \tilde{R}_{j,k-1} = \sum_{l=1}^{N} \langle R_{j,k-1}, e_l \rangle e_l, \tag{14}$$

before we solve the kth-order deformation equations (7). In this way, the solution $u_{j,k}(x)$ of Eqs. (7) is greatly simplified and shortened so that much less CPU time is needed to solve it. Obviously, the larger the number of N, the more accurate the solution is, but the more CPU time is needed. Therefore, the number N should be properly chosen to ensure both the accuracy of solution and the efficiency of computation.

The computation efficiency can be further greatly improved by means of an iteration technique, which is based on the truncation technique mentioned above. This is mainly because, unlike other analytic techniques, the HAM provides us great freedom to choose the initial guess. Obviously, the better the initial guess, the faster the solution series converges. Here, a "better" initial guess means that it is "closer" to the solution. Mathematically, an initial guess "closer" to the solution corresponds to a "smaller" squared residual. Thus, if the Mth-order approximation (5) is not accurate enough, we can use it as a new initial guess to gain a better Mth-order approximation. This provides us with an iteration approach. In this way, we can gain accurate approximations by means of much less CPU time, as illustrated by Liao [5] (see Chapter 2). It should be emphasized that such kind of iteration HAM approach, namely iHAM, must be based on the truncation of the right-hand side of (7). Otherwise, the solution expression quickly becomes so lengthy and complicated that the CPU time increases exponentially.

4. The implementation of the modification

We have to efficiently calculate the expansion coefficients $\langle R_{j,k-1}, e_l \rangle$ in (14). If the inner product $\langle \cdot, \cdot \rangle$ is hard to calculate, or the approximation $\tilde{R}_{j,k-1}$ is difficult to get, then the time wasted will outweigh the time saved, making the modification meaningless. We explain below how to deal with this problem and give the formulas of fast integration for two typical kinds of base functions.

Let $\varphi_1, \varphi_2, \ldots$ be linearly independent functions, and $e_1, \ldots, e_N$ be orthonormal functions obtained from $\varphi_1, \ldots, \varphi_N$ by means of the Schmidt–Gram process [15]. Write

$$R_{j,k-1} = \sum_{k=1}^{N^*} c_l \varphi_l, \tag{15}$$

where c_l are coefficients. Consider the two cases of $\tilde{R}_{j,k-1}$:

(i) If $N^* \le N$, then

$$\tilde{R}_{j,k-1} = R_{j,k-1}. \tag{16}$$

(ii) If $N^* > N$, then

$$\tilde{R}_{j,k-1} = \sum_{i=1}^{N} \left\langle \sum_{l=1}^{N^*} c_l\varphi_l, e_i \right\rangle e_i \tag{17}$$

$$= \sum_{l=1}^{N} c_l\varphi_l + \sum_{i=1}^{N} \left\langle \sum_{l=N+1}^{N^*} c_l\varphi_l, e_i \right\rangle e_i \tag{18}$$

$$= \sum_{l=1}^{N} c_l\varphi_l + \sum_{i=1}^{N} \sum_{l=N+1}^{N^*} c_l \langle \varphi_l, e_i \rangle e_i. \tag{19}$$

It is necessary to calculate $\langle \varphi_l, e_i \rangle$ only once and store the value in a table. Further more, write

$$e_i = \sum_{s=1}^{i} \alpha_s \varphi_s, \tag{20}$$

where α_s is a constant determined in the Schmidt–Gram process. According to the linear property of the inner product, it holds

$$\langle \varphi_l, e_i \rangle = \left\langle \varphi_l, \sum_{s=1}^{i} \alpha_s \varphi_s \right\rangle = \sum_{s=1}^{i} \alpha_s \langle \varphi_l, \varphi_s \rangle. \tag{21}$$

According to (19) and (21), the calculation of $\langle \varphi_l, \varphi_s \rangle$ is a primitive operation. The above approach greatly saves CPU time.

In the frame of the HAM, two kinds of base functions are often used for boundary value problems (BVPs) in a semi-infinite interval. Luckily, for these two typical kinds of base functions we have fast integration formulas to calculate the inner product.

First, let us consider the case that $R_{j,k-1}$ can be expressed by a finite linear combination of linearly independent functions

$$\{x^n e^{-mx} | n \geq 0, m \geq 1\}, \tag{22}$$

which depend on two parameters m and n. To access all of the functions in (22), an order is given to them as

$$x^n e^{-mx} = \varphi_{n+1+(m+n-1)(m+n)/2}, \tag{23}$$

i.e. $x^n e^{-mx}$ is the $n+1+(m+n-1)(m+n)/2$th function in the list $\varphi_1, \varphi_2, \ldots$. The corresponding inner product is defined in a natural way

$$\langle \varphi_l, \varphi_s \rangle = \int_0^{+\infty} \varphi_l(x) \varphi_s(x) dx, \quad l, s \geq 1, \tag{24}$$

whose exact value can be easily calculated by means of the formula

$$\int_0^{+\infty} e^{-mx} x^n dx = \frac{n!}{m^{n+1}}, \tag{25}$$

where m is a positive integer and n is a non-negative integer. In this way, it is efficient to calculate the inner product $\langle \varphi_l, \varphi_s \rangle$.

Secondly, let us consider the case that $R_{j,k-1}$ can be expressed by a finite linear combination of linearly independent functions

$$\{x^{-n} | n \geq 1\}. \tag{26}$$

Write $\varphi_n = x^{-n}$ and define the corresponding inner product

$$\langle \varphi_l, \varphi_s \rangle = \int_1^{+\infty} \varphi_l(x) \varphi_s(x) dx, \tag{27}$$

whose exact value can be easily gained by means of the formula

$$\int_1^{+\infty} x^{-n} dx = \frac{1}{n-1}, \quad n \geq 2. \tag{28}$$

5. Some examples

To illustrate the computational efficiency and validity of the iterative approach described above, three boundary value problems in a semi-infinite interval are used as examples, including two similarity boundary-layer flow problems and a non-similarity boundary-layer flow problem. The codes are written in Mathematica 7.0, and a desktop system with an Intel Core 2 Quad 2.66 GHz CPU (4 GB memory) is used.

5.1. Example 1: Blasius boundary-layer flow

First of all, let us consider the famous Blasius boundary-layer flow in fluid mechanics, governed by the nonlinear differential equation (see [22] for details)

$$f'''(\eta) + \frac{1}{2} f(\eta) f''(\eta) = 0, \qquad f(0) = f'(0) = 0, \quad f'(+\infty) = 1, \tag{29}$$

where η is a similarity variable, $f(\eta)$ is related to the stream-function, and the prime denotes the derivative with respect to η. Let $\lambda > 0$ denote a kind of scale parameter and introduce the transformation

$$f(\eta) = \lambda^{-1} F(\xi), \quad \xi = \lambda \eta. \tag{30}$$

Then, Eq. (29) becomes

$$F'''(\xi) + \frac{1}{2\lambda^2} F(\xi) F''(\xi) = 0, \qquad F(0) = F'(0) = 0, \quad F'(+\infty) = 1, \tag{31}$$

where the prime denotes the derivative with respect to ξ. This equation has been solved by means of the HAM in a high level of accuracy [14,23]. Following Liao [14,23], we choose the same auxiliary linear operator

$$\mathcal{L} = \frac{\partial^3}{\partial \xi^3} + \frac{\partial^2}{\partial \xi^2}, \tag{32}$$

the same initial approximation $F_0(\xi) = \xi - 1 - e^{-\xi}$, the same convergence-control parameter $c_0 = -1$ (if not mentioned), and the same value $\lambda = 4$. According to the boundary condition of (31), its solution $F(\xi)$ can be expressed in the form

$$F(\xi) = A_{0,0} + \xi + \sum_{m=1}^{+\infty} \sum_{n=0}^{+\infty} A_{m,n} \xi^n e^{-m\xi}, \tag{33}$$

where $A_{m,n}$ is a constant. Thus, the base function (22) is used here.

By mans of the standard HAM, we gain the convergent solution: at the 50th-order approximation, the solution converges to the correct result $f''(0) = 0.332057$ with the squared residual 4.0×10^{-8}, as shown in Table 1. However, the 50th-order approximation is rather lengthy: it has 2603 terms. Besides, it takes 7270.9 s CPU time.

Using the 3rd-order iterative HAM with $N = 40$, we gain a much more accurate approximation by means of much less CPU time: the 40th iteration gives the accurate value $f''(0) = 0.332057$ with the squared residual 1.6×10^{-12} in only 61.9 s CPU time, as shown in Table 2. Note that the CPU time used by the standard HAM is more than 117 times longer than that by the iterative HAM, even though the squared residual of the former is 25 000 times larger than that of the latter. This clearly illustrates the efficiency and validity of the iterative HAM.

Let us compare the accuracy and the CPU time used of the standard and the iterative HAM in detail. It is found that the CPU time increases *linearly* with the iteration times for the iterative

Table 1
The squared residual, the number of terms in the approximations and the approximations of $f''(0)$ using the standard HAM for Example 1.

m, order of approx.	E_m	Number of terms	$f''(0)$
5	0.2	38	0.256390
10	2.5×10^{-2}	123	0.327756
15	8.5×10^{-3}	258	0.331256
20	7.5×10^{-4}	443	0.331851
25	1.5×10^{-4}	678	0.332004
30	3.8×10^{-5}	963	0.332040
35	5.4×10^{-6}	1298	0.332052
40	7.2×10^{-7}	1683	0.332055
45	1.6×10^{-7}	2118	0.332057
50	4.0×10^{-8}	2603	0.332057

Table 2
The squared residual and approximations of $f''(0)$ using the 3rd-order iterative HAM approach with $N = 40$ for Example 1.

Iteration times (m)	E_m	Number of terms	$f''(0)$
5	2.2×10^{-2}	43	0.367709
10	3.2×10^{-4}	43	0.329884
15	6.2×10^{-6}	43	0.331085
20	1.2×10^{-7}	43	0.331991
25	1.9×10^{-9}	43	0.332068
30	4.2×10^{-11}	43	0.332060
35	2.5×10^{-11}	43	0.332057
40	1.6×10^{-12}	43	0.332057
45	1.6×10^{-12}	43	0.332057
50	1.6×10^{-12}	43	0.332057

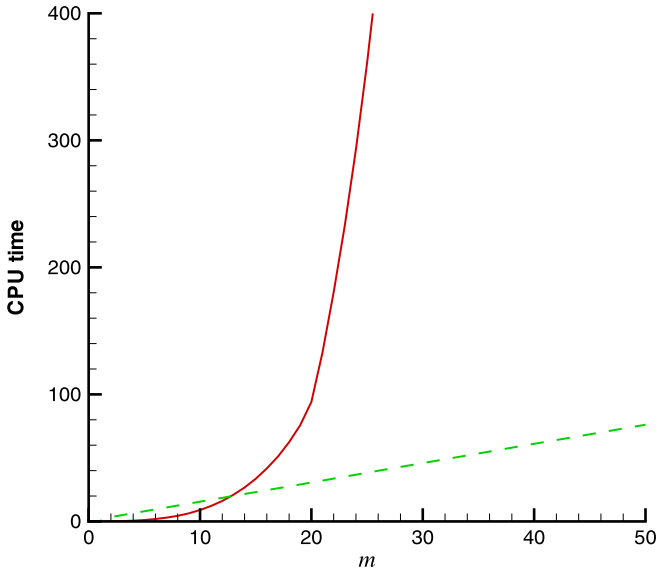

Fig. 1. CPU time versus m for Example 1. Solid line: standard HAM without iteration (m denotes the order of approximation); dashed line: 3rd-order iterative HAM approach with $N = 40$ (m denotes the iteration times).

HAM, but *exponentially* with the order of approximation for the standard HAM, as shown in Fig. 1. Thus, the iterative HAM is computationally much more efficient. As shown in Fig. 2, the squared residual by the iterative HAM decreases more quickly than that by the standard HAM, until the minimum of the squared residual $E_{\min} = 1.6 \times 10^{-12}$ is arrived at, which is mainly determined by the value of N for the truncation. This clearly indicates that the approximations given by iterative HAM converge much faster than the standard HAM.

The squared residual versus the CPU time of the standard and iterative HAM are given in Fig. 3. It is found that the squared residual given by the iterative HAM decreases *exponentially* with the CPU time until its minimum value $E_{\min}$ is reached, where the squared residual given by the non-iterative HAM decreases much more slowly. Thus, using the same CPU time, one can obtain much more accurate approximations by means of the iterative HAM. All of the above conclusions have general meanings: they are correct for different-order iterative HAM, as shown in Fig. 3. Note that the approximations given by the 1st-order iterative HAM converges a little faster than those given by the 3rd-order iterative HAM, whereas the approximations given by the 3rd-order iterative HAM converge a little faster than those given by the 5th-order iterative HAM. The approximations given by the 1st, 3rd and 5th-order iterative HAM converge faster than those given by the truncated HAM without iteration. All of these indicate that, in the frame of the HAM, the iteration can greatly accelerate the convergence of approximations of BVPs in an infinite interval.

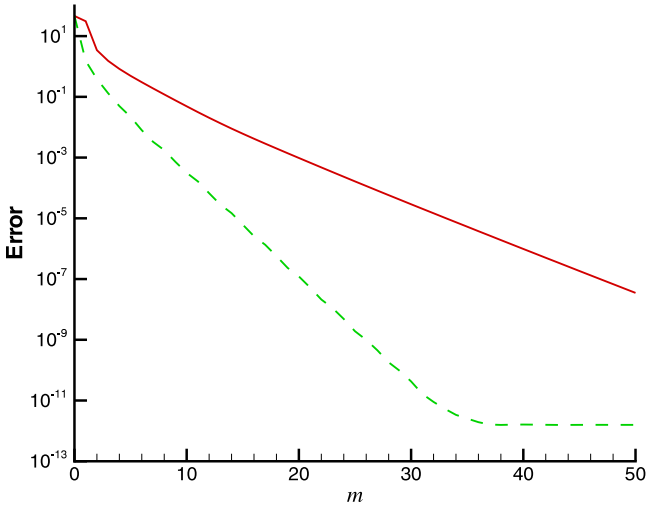

Fig. 2. Squared residual E_m versus m for Example 1. Solid line: standard HAM without iteration (m denotes the order of approximation); dashed line: 3rd-order iterative HAM approach with $N = 40$ (m denotes the iteration times).

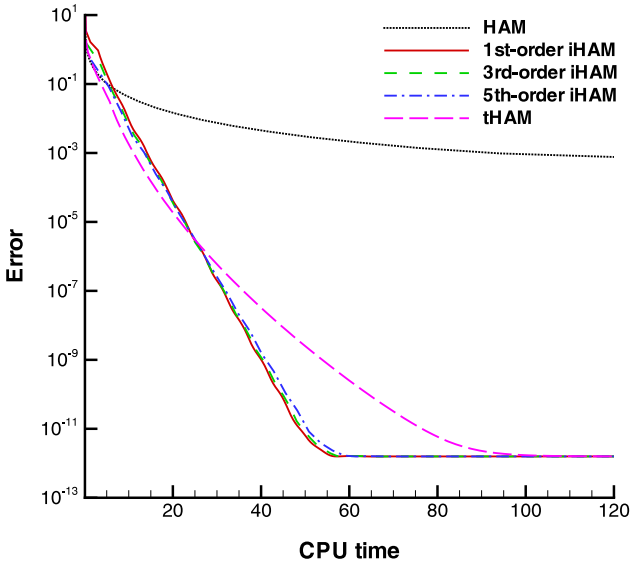

Fig. 3. Squared residual versus CPU time for Example 1. Dotted line: standard HAM without iteration; solid line: 1st-order iterative HAM; dashed line: 3rd-order iterative HAM; dash dot line: 5rd-order iterative HAM; long dash line: the truncated HAM without iteration.

As shown in Figs. 2 and 3, when the minimum squared residual $E_{\min} = 1.6 \times 10^{-12}$ is reached, we cannot get better approximations by more iterations. Theoretically speaking, the minimum squared residual $E_{\min}$ of the iterative HAM is dependent upon the value of N used for the truncation. Obviously, the larger N corresponds to the smaller $E_{\min}$. This is indeed true in practice, as shown in Table 3 and Fig. 4, which illustrate that: (i) $E_{\min}$ is dependent upon N, the number of terms reserved in the right-hand side of the high-order deformation equation, but has nothing to do with the order of iteration formula; and (ii) $E_{\min}$ decreases *exponentially* with the increase of N (it is found that $E_{\min} \approx e^{-0.8402N+5.3712}$ in this case).

It is found that the convergence-control parameter c_0 also has some influence on the convergence-rate. As shown in Fig. 5, the approximations by the 3rd-order iterative HAM when $c_0 = -3/2$ converge faster than those when $c_0 = -1$, and approximations when $c_0 = -1$ converge faster than those when $c_0 = -1/2$,

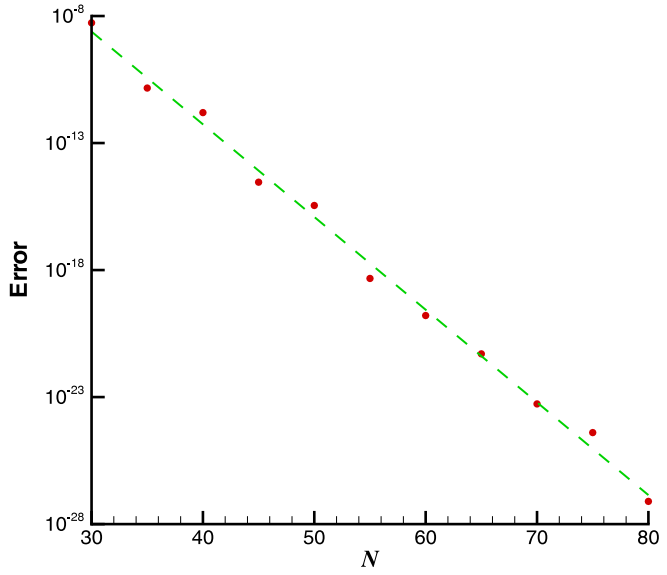

Fig. 4. Solid dot: the minimum squared residual $E_{\min}$ versus the number of truncated terms N by the iterative HAM for Example 1. Dashed line: the curve of $e^{-0.8402N+5.3712}$.

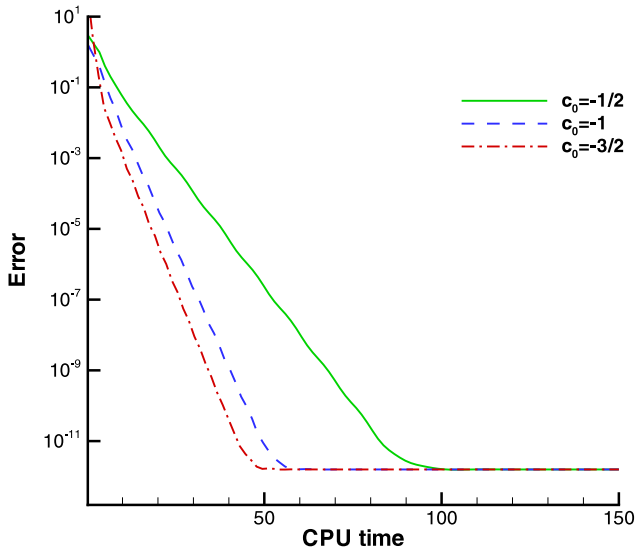

Fig. 5. Squared residual versus CPU time by the 3rd-order iterative HAM with different values of convergence-control parameter c_0 for Example 1. Solid line: $c_0 = -1/2$; dashed line: $c_0 = -1$; dash dot line: $c_0 = -3/2$.

Table 3
The minimum of the squared residual by means of the different order iterative HAM with the different number of terms N for Example 1.

N	1st-order iteration	3rd-order iteration	5th-order iteration
30	5.39×10^{-9}	5.39×10^{-9}	5.39×10^{-9}
40	1.58×10^{-12}	1.58×10^{-12}	1.58×10^{-12}
50	3.49×10^{-16}	3.49×10^{-16}	3.49×10^{-16}
60	1.62×10^{-20}	1.62×10^{-20}	1.62×10^{-20}
70	5.38×10^{-24}	5.38×10^{-24}	5.38×10^{-24}

respectively. Note that the squared residual of the three different approaches decrease to the *same* level $E_{\min} = 1.6 \times 10^{-12}$, indicating once again that $E_{\min}$ is only determined by N for the truncation.

5.2. Example 2: Kuiken's equation

Let us consider a set of two coupled nonlinear differential equations (see [24] for details)

$$f'''(\eta) + \theta(\eta) - f'^2(\eta) = 0, \tag{34}$$

$$\theta''(\eta) - 3\sigma f'(\eta)\theta(\eta) = 0, \tag{35}$$

with the boundary conditions

$$f(0) = f'(0) = 0, \qquad \theta(0) = 1, \qquad f'(+\infty) = \theta(+\infty) = 0,$$

where σ is the Prandtl number. Under the transformation

$$\xi = 1 + \lambda\eta, \qquad F(\xi) = f'(\eta), \qquad S(\xi) = \theta(\eta),$$

Eqs. (34) and (35) become

$$\lambda^2 F''(\xi) + S(\xi) - F^2(\xi) = 0, \tag{36}$$

$$\lambda^2 S''(\xi) - 3\sigma F(\xi)S(\xi) = 0, \tag{37}$$

with the boundary conditions

$$F(1) = 0, \qquad S(1) = 1, \qquad F(+\infty) = S(+\infty) = 0.$$

Considering their algebraic property at infinity, we seek the solution $F(\xi)$ and $S(\xi)$ in the form

$$F(\xi) = \sum_{n=2}^{+\infty} a_n \xi^{-n}, \qquad S(\xi) = \sum_{n=4}^{+\infty} b_n \xi^{-n},$$

respectively, where a_n, b_n are coefficients. Thus, the base function (26) is used here.

Following Liao [7], we choose the same auxiliary linear operators

$$\mathcal{L}_F = \left(\frac{\xi}{3}\right)\frac{\partial^2}{\partial\xi^2} + \frac{\partial}{\partial\xi}, \qquad \mathcal{L}_S = \left(\frac{\xi}{5}\right)\frac{\partial^2}{\partial\xi^2} + \frac{\partial}{\partial\xi},$$

the same initial guesses

$$F_0(\xi) = \gamma(\xi^{-2} - \xi^{-3}), \qquad S_0(\xi) = \xi^{-4},$$

and the same parameters $\lambda = 1/3, \gamma = 3$. Then, the corresponding high-order deformation equations read

$$\mathcal{L}_F[F_n(\xi) - \chi_n F_{n-1}(\xi)] = c_F R^F_{n-1},$$

$$\mathcal{L}_S[S_n(\xi) - \chi_n S_{n-1}(\xi)] = c_S R^S_{n-1},$$

subject to the homogeneous boundary conditions

$$F_n(1) = S_n(1) = F_n(+\infty) = S_n(+\infty) = 0,$$

where c_F, c_S are convergence-control parameters, and

$$R^F_{n-1} = \lambda^2 F''_{n-1}(\xi) + S_{n-1}(\xi) - \sum_{j=0}^{n-1} F_j(\xi)F_{n-1-j}(\xi),$$

$$R^S_{n-1} = \lambda^2 S''_{n-1}(\xi) - 3\sigma \sum_{j=0}^{n-1} F_j(\xi)S_{n-1-j}(\xi).$$

It is found that R^F_{n-1} and R^S_{n-1} are expressed by a finite combination of functions

$$\{\xi^{-n} | n \geq 4\}, \tag{38}$$

and

$$\{\xi^{-n} | n \geq 6\}, \tag{39}$$

respectively. Applying the Schmidt–Gram process to the first N functions in (38) and (39), respectively, we obtain two sets of N orthonormal functions, which are used to calculate the corresponding inner product.

Table 4
The squared residual, the number of terms in the approximations, and the approximations of $f''(0)$ using the standard HAM for Example 2 (in the case of $\sigma = 1$ and $c_F = c_S = -1/2$).

m, order of approx.	E_m	Number of terms	$f''(0)$
15	9.9×10^{-5}	32	0.702546920
30	1.7×10^{-7}	62	0.693537549
45	7.2×10^{-10}	92	0.693227314
60	8.9×10^{-12}	122	0.693211627
75	1.0×10^{-12}	152	0.693211551
90	2.3×10^{-13}	182	0.693211633
105	6.4×10^{-14}	212	0.693211633
120	2.1×10^{-14}	242	0.693211632
135	7.5×10^{-15}	272	0.693211632
150	3.0×10^{-15}	302	0.693211632

Table 5
The squared residual and the approximations of $f''(0)$ by means of the 3rd-order iterative HAM with $N = 30$ for Example 2 (in the case of $\sigma = 1$ and $c_F = c_S = -1/2$).

m, iteration times	E_m	Number of terms	$f''(0)$
15	1.1×10^{-9}	31	0.693227799
30	2.0×10^{-14}	31	0.693211615
45	4.1×10^{-16}	31	0.693211632
60	3.6×10^{-17}	31	0.693211632
75	6.0×10^{-18}	31	0.693211632
90	1.2×10^{-18}	31	0.693211632
105	3.1×10^{-19}	31	0.693211632
120	1.7×10^{-19}	31	0.693211632
135	1.6×10^{-19}	31	0.693211632
150	1.7×10^{-19}	31	0.693211632

To compare the computational efficiency of the standard and iterative HAM for Example 2, we follow Liao [7] to consider the case of $\sigma = 1$ and use the same parameters $\lambda = 1/3, \gamma = 3$ and the same convergence-control parameters $c_F = c_S = -1/2$. The approximations given by the standard HAM converge: it takes 237 s CPU time to gain the 150th-order of approximation with the squared residual 3.0×10^{-15} and the convergent value $f''(0) = 0.693211632$, as shown in Table 4.

For the iterative HAM, we use the truncation number $N = 30$, say, the right-hand sides of the high-order deformation equations are approximated by the first 30 terms, i.e. from ξ^{-4} to ξ^{-33} for the 1st equation, and from ξ^{-6} to ξ^{-35} for the 2nd one, respectively. As shown in Table 5, the approximations given by the 3rd-order iterative HAM converge much faster than those by the standard HAM: the 45th iteration gives the convergent value $f''(0) = 0.693211632$ in only 29 s CPU time even with the less squared residual 4.1×10^{-16}. This is mainly because the approximations given by the standard HAM become more and more complicated, however, the approximations given by the iterative HAM always have 43 terms but are accurate enough. This confirms once again that the iterative HAM is indeed much more efficient than the standard HAM.

Let us compare the accuracy and the CPU time used of the standard and iterative HAM. It is found that the CPU time increases *linearly* for the iterative HAM with the iteration times, but *exponentially* for the standard HAM with the order of approximation, as shown in Fig. 6. Besides, as shown in Fig. 7, the squared residual for the iterative HAM decreases much faster, until the minimum of the squared residual $E_{\min} = 1.7 \times 10^{-19}$ is arrived. This confirms once again that the approximations given by iterative HAM indeed converge much faster. These conclusions about the iterative HAM have general meanings: they are correct for different orders of iterative approaches, as shown in Fig. 8. Therefore, using the same CPU time, one can obtain much more accurate approximations by means of the iterative HAM.

As shown in Figs. 7 and 8, when the minimum of the squared residual $E_{\min} = 1.7 \times 10^{-19}$ (when $N = 30$) arrives, one cannot get better approximations by more iterations. Theoretically, the minimum of the squared residual is dependent upon the truncation number N. This is indeed true, as shown in Table 6 and Fig. 9. Therefore, $E_{\min}$ decreases with the increase of the truncation number N. Note that the larger truncation number N corresponds to more CPU time but more accurate approximations. In practice,

Table 6
The minimum of squared residual by means of the Mth-order iterative HAM with different truncation number N for Example 2 (in the case of $\sigma = 1$ and $c_F = c_S = -1/2$).

N	$M = 1$	$M = 3$	$M = 5$
10	4.7×10^{-12}	4.7×10^{-12}	4.7×10^{-12}
30	1.7×10^{-19}	1.7×10^{-19}	1.7×10^{-19}
40	3.3×10^{-21}	3.3×10^{-21}	3.3×10^{-21}
50	1.5×10^{-22}	1.5×10^{-22}	1.5×10^{-22}
70	1.4×10^{-24}	1.4×10^{-24}	1.4×10^{-24}

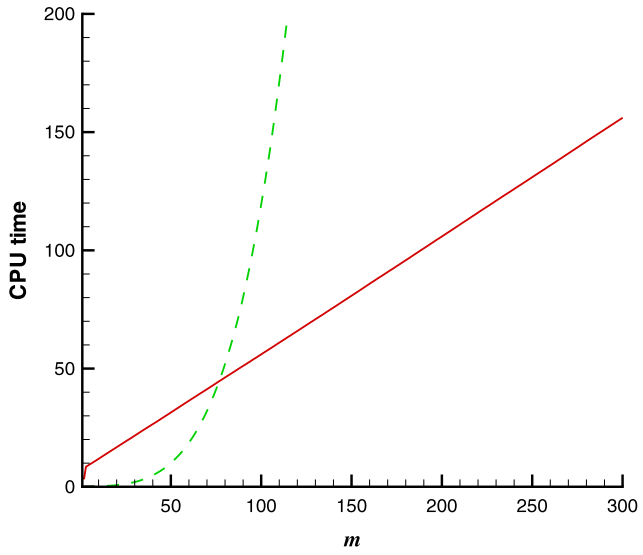

Fig. 6. CPU time versus m for Example 2 (in the case of $\sigma = 1$ and $c_F = c_S = -1/2$). Dotted line: standard HAM (m denotes the order of approximation); solid line: 3rd-order iterative HAM (m denotes the iteration times).

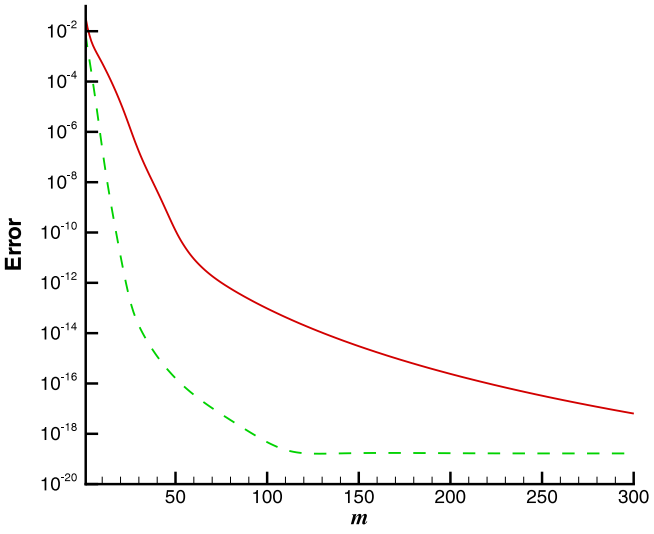

Fig. 7. Squared residual E_m versus m for Example 2 (in the case of $\sigma = 1$ and $c_F = c_S = -1/2$). Solid line: standard HAM (m denotes the order of approximation); dashed line: 3rd-order iterative HAM (m denotes the iteration times).

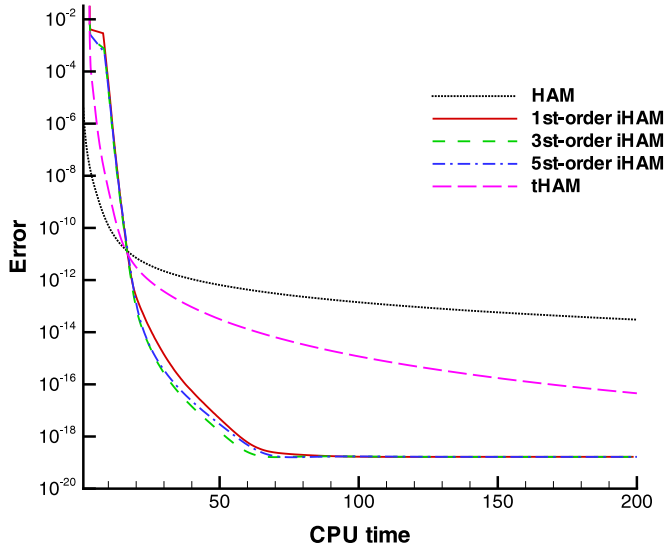

Fig. 8. Squared residual versus the CPU time for Example 2 (in the case of $\sigma = 1$ and $c_F = c_S = -1/2$). Dotted line: standard HAM; solid line: 1st-order iterative HAM; dashed line: 3rd-order iterative HAM; dash dot line: the truncated HAM without iteration.

we should choose a proper truncation number N for the iterative HAM, so that accurate enough approximations can be gained with high enough computational efficiency.

5.3. Example 3: non-similarity boundary-layer flow

The iterative HAM can be applied to some nonlinear PDEs. For instance, let us consider a nonlinear PDE [25]

$$\frac{\partial^3 f}{\partial \eta^3} + \frac{1}{2} f \frac{\partial^2 f}{\partial \eta^2} + (1-\xi)\left(\frac{\partial f}{\partial \eta}\frac{\partial^2 f}{\partial \eta^2} - \frac{\partial f}{\partial \eta}\frac{\partial^2 f}{\partial \xi \partial \eta}\right) = 0, \tag{40}$$

subject to the boundary conditions

$$f(\xi, 0) = 0, \qquad f_\eta(\xi, 0) = \xi, \qquad f_\eta(\xi, +\infty) = 0. \tag{41}$$

It describes the non-similarity boundary-layer flows over a stretching flat sheet [25].

Table 7
The squared residual and the number of terms in the approximations by means of the standard HAM (without truncation, with linear operator (42), initial guess $f_0(\xi, \eta) = \xi(1 - e^{-\eta})$ and the convergence-control parameter $c_0 = -1/2$) for Example 3.

m, order of approx.	E_m	Number of terms
5	3.0×10^{-4}	62
10	8.9×10^{-6}	297
15	1.7×10^{-6}	832
20	4.2×10^{-7}	1792
25	1.8×10^{-7}	3302
30	1.0×10^{-7}	5487

As pointed out by Liao [25], the solution $f(\xi, \eta)$ can be expressed in the form

$$f(\xi, \eta) = \sum_{m=0}^{+\infty} \sum_{n=0}^{+\infty} \sum_{s=0}^{+\infty} a_{m,n,s} \quad \xi^n \eta^s \exp(-m\eta),$$

where $a_{m,n,s}$ is a constant coefficient to be determined. Using the standard HAM (without truncation), Liao [25] gained convergent approximations by means of the initial guess $f_0(\xi, \eta) = \xi(1-e^{-\eta})$, the auxiliary linear operator

$$\mathcal{L}[f] = \frac{\partial^3 f}{\partial \eta^3} - \frac{\partial f}{\partial \eta} \tag{42}$$

and the convergence-control parameter $c_0 = -1/2$. The corresponding squared residual and the number of terms appearing in the approximations are listed in Table 7. Note that the 30th-order approximation has 5487 terms, with the corresponding squared residual 1.0×10^{-7}. Thus, it needs more and more CPU time to get high-order approximations because of exponentially increasing intermediate expressions.

Since Eq. (40) is a PDE, the term R_{k-1} on the right-hand side of the high-order deformation equation depends on the two variables ξ and η. The Chebyshev polynomials in the ξ-direction and the orthonormal functions (22) in the η-direction are used to approximate the term R_{k-1}. Let N_ξ and N_η denote the truncation numbers in the ξ and η direction, respectively. In theory, the larger the truncation numbers N_ξ and N_η, the more complicated the expression of approximations, but the smaller the minimum of squared residuals.

However, if we use the same initial guess $f_0(\xi, \eta)$, the same auxiliary linear operator $\mathcal{L}$ and the same convergence-control

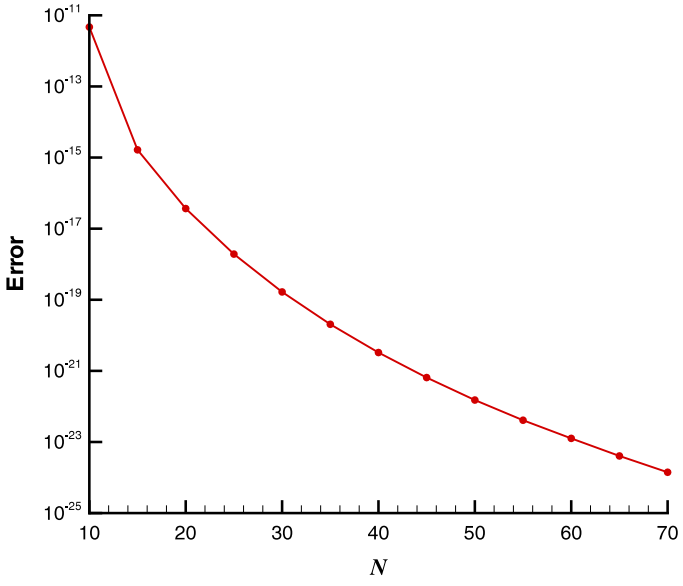

Fig. 9. The minimum squared residual versus the truncation number N of the iterative HAM for Example 2 (in the case of $\sigma = 1$ and $c_F = c_S = -1/2$).

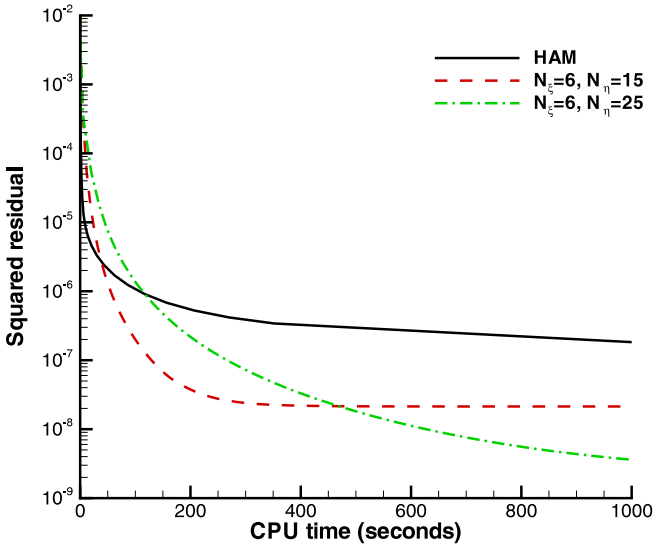

Fig. 10. Squared residual versus CPU time for Example 3. Solid line: the standard HAM with linear operator (42), initial guess $f_0(\xi,\eta) = \xi(1-e^{-\eta})$ and the convergence-control parameter $c_0 = -1/2$; dashed line: 1st-order iterative HAM with linear operator (43), initial guess $f_0(\xi,\eta) = \xi(1-e^{-\eta})$, $c_0 = -2$, $N_\xi = 6$ and $N_\eta = 15$; long dash line: 1st-order iterative HAM with linear operator (43), initial guess $f_0(\xi,\eta) = \xi(1-e^{-\eta})$, $c_0 = -2$, $N_\xi = 6$ and $N_\eta = 25$.

Table 8
The minimum of squared residual by means of 1st-order iterative HAM with different N_ξ and N_η for Example 3.

N_η	$N_\xi = 15$	$N_\xi = 20$	$N_\xi = 25$	$N_\xi = 30$	$N_\xi = 40$
5	2.6×10^{-8}	2.0×10^{-8}	7.5×10^{-9}	6.7×10^{-9}	6.4×10^{-9}
6	2.1×10^{-8}	5.3×10^{-9}	2.2×10^{-9}	1.3×10^{-9}	9.6×10^{-10}
10	2.0×10^{-8}	4.4×10^{-9}	1.3×10^{-9}	4.8×10^{-10}	1.0×10^{-10}
15	2.0×10^{-8}	4.4×10^{-9}	1.3×10^{-9}	4.8×10^{-10}	9.8×10^{-11}

parameter c_0 as those by Liao [25], we cannot gain convergent results by means of the iterative HAM. It is found that the truncation error in the ξ direction increases quickly during the iteration. Since the HAM provides us great freedom to choose the auxiliary linear operator, we use such a new auxiliary linear operator

$$\widetilde{\mathcal{L}}[f] = \frac{\partial}{\partial\xi}\left[\xi\left(\frac{\partial^3 f}{\partial\eta^3} - \frac{\partial f}{\partial\eta}\right)\right] \tag{43}$$

with a property

$$\widetilde{\mathcal{L}}^{-1}[\xi^n] = \mathcal{L}^{-1}\left[\frac{\xi^n}{n+1}\right], \quad n \geq 0, \tag{44}$$

which leads to the convergence of the approximations given by the iterative HAM when $c_0 = -2$. As shown in Fig. 10, the iterative HAM is more efficient than the standard HAM without truncation. For example, it takes about 400 s for the 1st-order iterative HAM (with $N_\xi = 6, N_\eta = 15$ and $c_0 = -2$) to reach its minimum squared residual $E_{\min} = 2.1\times10^{-8}$, but for the standard HAM (with linear operator (42) and the convergence-control parameter $c_0 = -1/2$) it obtains a much more complicated approximation with the squared residual larger than 1.0×10^{-7} in the same time. As shown in Fig. 10, when the minimum of squared residual is arrived at, one cannot get better approximations by more iterations. In order to get more accurate approximations, larger truncation numbers should be used, as shown in Table 8.

This example illustrates that the iterative HAM is valid even for some nonlinear PDEs, and thus has general meanings.

6. Conclusions

The homotopy analysis method (HAM) has been successfully applied to solve a lot of nonlinear ODEs and PDEs in science and engineering. Based on the HAM, the Mathematica package BVPh 1.0 was issued for nonlinear ODEs, which is free and available online (http://numericaltank.sjtu.edu.cn/BVPh.htm) and is becoming an easy-to-use analytic tool. As pointed out by Trfethen [4], "computing numerically with functions instead of numbers" has many advantages over numerical ones. For example, the analytic approaches can exactly handle an infinite interval, since boundary conditions at infinity can be easily satisfied by means of proper base functions.

Briefly speaking, the HAM transfers a nonlinear problem into an infinite number of linear sub-problems by means of the homotopy in topology. However, due to the nonlinearity, the right-hand side of the high-order deformation equation (corresponding to the sub-problems) becomes more and more complicated so that it needs more and more CPU time to gain a better approximation. For equations defined in a finite interval, it is convenient to use polynomial or Fourier series to approximate the right-hand side, as illustrated by Liao in Section 7.2 of his book [5]. For equations defined in an infinite interval, the Schmidt–Gram process [15] is used in this paper to approximate the right-hand side by means of a finite set of orthonormal bases. Based on this kind of truncation technique, we introduce the Mth-order iterative HAM by using each Mth-order approximation as a new initial guess. It is found that the iterative HAM is much more efficient than the standard HAM without truncation, as illustrated by the three nonlinear differential equations defined in an infinite interval as examples.

In summary, this iterative HAM approach can greatly improve the computational efficiency of the HAM (and also the Mathematica package BVPh) for many nonlinear BVPs.

Acknowledgments

Thanks to the anonymous reviewer for their constructive comments and suggestions. This work is partly supported by the National Natural Science Foundation of China (Approval No. 11272209) and the State Key Lab of Ocean Engineering (Approval No. GKZD010056).

References

[1] L.F. Shampine, I. Gladwell, S. Thompson, Solving ODEs with MATLAB, Cambridge University Press, Cambridge, 2003.
[2] J. Kierzenka, L.F. Shampine, ACM Trans. Math. Softw. (TOMS) 27 (3) (2001) 299.
[3] Z. Battles, L.N. Trefethen, Math. Comput. Sci. 25 (2004) 1743.
[4] L.N. Trefethen, Math. Comput. Sci. 1 (2007) 9.
[5] S.J. Liao, Homotopy Analysis Method in Nonlinear Differential Equations, Springer-Verlag, New York, 2012.
[6] S.J. Liao, Proposed homotopy analysis techniques for the solution of nonlinear problem, Ph.D. Thesis, Shanghai Jiao Tong University, 1992.
[7] S.J. Liao, Beyond Perturbation: Introduction to the Homotopy Analysis Method, Chapman & Hall/CRC, Boca Raton, 2003.
[8] S.J. Liao, J. Fluid Mech. 385 (1999) 101.
[9] S.J. Liao, J. Fluid Mech. 488 (2003) 189.
[10] S.J. Liao, Stud. Appl. Math. 117 (2006) 239.
[11] Y. Liu, S. Liao, Z. Li, J. Symbolic Comput. 55 (2013) 72.
[12] K. Yabushita, M. Yamashita, K. Tsuboi, J. Phys. A 40 (29) (2007) 8403.
[13] Z. Niu, C. Wang, Commun. Nonlinear Sci. Numer. Simul. 15 (8) (2010) 2026.
[14] S.J. Liao, Commun. Nonlinear Sci. Numer. Simul. 15 (8) (2010) 2003.
[15] G. Arfken, Mathematical Methods for Physicists, 3rd ed., Academic Press, New York, 1985.
[16] S.J. Liao, Y. Tan, Stud. Appl. Math. 119 (4) (2007) 297.
[17] S.J. Liao, Commun. Nonlinear Sci. Numer. Simul. 14 (4) (2009) 983.
[18] M. Ghoreishi, A. Ismail, A. Alomari, A.S. Bataineh, Commun. Nonlinear Sci. Numer. Simul. 17 (3) (2012) 1163.
[19] S. Abbasbandy, E. Shivanian, K. Vajravelu, Commun. Nonlinear Sci. Numer. Simul. 16 (11) (2011) 4268.
[20] R.V. Gorder, Commun. Nonlinear Sci. Numer. Simul. 17 (3) (2012) 1233.
[21] R.V. Gorder, Int. J. Nonlinear Mech. 47 (3) (2012) 1.
[22] F. White, Viscous Fluid Flow, McGraw-Hill, New York, 1991.
[23] S.J. Liao, Int. J. Nonlinear Mech. 34 (4) (1999) 759.
[24] H. Kuiken, Quart. J. Mech. Appl. Math. 34 (3) (1981) 397.
[25] S.J. Liao, Commun. Nonlinear Sci. Numer. Simul. 14 (5) (2009) 2144.