Phase function methods for second order inhomogeneous linear ordinary differential equations

Kirill Serkh[1,2*] and James Bremer[2*]

[1*]Department of Computer Science, University of Toronto.

[2]Department of Mathematics, University of Toronto.

*Corresponding author(s). E-mail(s): bremer@math.toronto.edu;

Abstract

It is well known that second order homogeneous linear ordinary differential equations with slowly varying coefficients admit slowly varying phase functions. This observation underlies the Liouville-Green method and many other techniques for the asymptotic approximation of the solutions of such equations. It is also the basis of a recently developed numerical algorithm for solving second order linear ordinary differential equations that, in many cases of interest, runs in time independent of the magnitude of the equation's coefficients and achieves accuracy on par with that predicted by its condition number. Here we point out that a large class of second order inhomogeneous linear ordinary differential equations can be efficiently and accurately solved by combining phase function methods for second order homogeneous linear ordinary differential equations with a variant of the adaptive Levin method for evaluating oscillatory integrals.

1 Introduction

We say that α is a phase function for the second order homogeneous linear ordinary differential equation

$$y''(t) + q(t)y(t) = 0, \qquad a < t < b, \tag{1}$$

provided α' is positive on (a, b) and the pair

$$u(t) = \frac{\cos(\alpha(t))}{\sqrt{\alpha'(t)}} \quad \text{and} \quad v(t) = \frac{\sin(\alpha(t))}{\sqrt{\alpha'(t)}} \tag{2}$$

is a basis in its space of solutions. It is well known that when q is a slowly varying function, (1) admits slowly varying phase functions, and that this is true even when q is of large magnitude and the solutions of (1) are highly oscillatory or behave as combinations of rapidly increasing and decreasing exponential functions. This observation underlies the Liouville-Green method (see, for instance, Chapter 6 of [1] or Chapter 7 of [2]) and many other techniques for the asymptotic approximation of the solutions of such equations (for example, [3, 4, 5]).

It is also the basis of the algorithm of [6] for solving equations of the form (1) where the coefficient q is positive. In this case, the solutions of (1) are oscillatory and the frequency of their oscillations grows with the magnitude of q. Consequently, when they are used in this regime, the running times of standard solvers for ordinary differential equations grow with the magnitude of q. By contrast, in many cases of interest, the numerical algorithm of [6] is able to calculate a nonoscillatory phase function for (1) with near machine precision accuracy in time independent of the magnitude of q. The companion paper [7] gives a bound on the complexity of the phase function produced by the algorithm of [6] under mild conditions on the coefficient q.

In this article, we describe a solver for second order inhomogeneous liner ordinary differential equations of the form

$$y''(t) + q(t)y(t) = f(t), \qquad a < t < b, \tag{3}$$

where f is a slowly varying real-valued function and q is slowly varying and positive. The algorithm of [6] is first used to construct a nonoscillatory phase function α such that the functions u and v defined via (2) form a basis in the solution space of the corresponding homogeneous problem (1). A particular solution of (3) is then given by the formula

$$z(t) = u(t) \int_a^t v(s)f(s)\, ds + v(t) \int_a^t u(s)f(s)\, ds, \tag{4}$$

and any solution of (3) is of the form

$$y(t) = c_1 u(t) + c_2 v(t) + z(t) \tag{5}$$

with c_1 and c_2 constants. We then use an adaptive Levin method, similar to that introduced in [8], to construct a sequence of auxillary functions $p_1, \ldots, p_m$, defined on a partition of $[a, b]$, which allow us to efficiently evaluate

the integrals appearing in (4). In many cases of interest, α and the sequence of functions $p_1, \ldots, p_m$ can be constructed in time independent of the magnitude of q and, once this has been done, constants c_1 and c_2 such that (5) satisfies essentially any reasonable boundary conditions can be readily computed.

The Levin method, which was introduced in [9], is a classical technique for evaluating integrals of the form

$$\int_a^b \exp(ig(s))f(s)\ ds \tag{6}$$

where f is slowly varying and g is real-valued and slowly varying. It was long believed that it suffers from "low-frequency breakdown," meaning that accuracy is lost when g' is of insufficient magnitude. In [8], it is shown that when the associated linear system is solved using spectral collocation and a truncated singular value decomposition, the Levin method is applicable regardless of the magnitude of g'. Moreover, this observation is used to develop an adaptive integration scheme for integrals of the form (6). One of the key advantages of the resulting "adaptive Levin method" is its ability to efficiently evaluate oscillatory integrals in the presence of saddle points — that is, locations at which the first $l \geq 1$ derivatives of g vanish.

Although the method of [6] can be extended to the case in which (1) has turning points (this is done in [10]), the algorithm presented here often encounters numerical difficulties when used in this regime. They stem from the fact that one or both of the functions u and v must be rapidly increasing in regions in which q is negative and of large magnitude, and this makes (5) a numerically unstable representation of the desired solution of (3). Of course, in many cases in which additional information about the desired solution is known *a priori*— for instance, if it is a linear combination of the particular solution and a solution u of (1) whose magnitude does not become large — a numerically stable formula for representing the desired solution can be devised and (3) can be solved to high accuracy using a slightly modified version of the algorithm presented here.

The remainder of this paper is structured as follows. In Section 2, we discuss phase functions for second order homogeneous linear ordinary differential equations and describe the algorithm of [6] for the numerical calculation of nonoscillatory phase functions. Section 3 discusses the adaptive Levin method for the evaluation of oscillatory integrals and explains how a variant of it can be used to construct the sequence of functions $p_1, \ldots, p_m$ which enable the rapid evaluation of the particular solution (4). In Section 4, we present the results of numerical experiments designed to demonstrate the properties of our approach to solving (3). We close with a few brief remarks in Section 5.

2 Phase functions for second order homogeneous linear ordinary differential equations

2.1 Kummer's equation

It can be easily verified that if α is a phase function for the second order homogeneous linear ordinary differential equation (1) — so that the functions u and v defined in (2) form a basis in its space of solutions — then α' solves

$$q(t) - (\alpha'(t))^2 + \frac{3}{4}\left(\frac{\alpha''(t)}{\alpha'(t)}\right)^2 - \frac{1}{2}\frac{\alpha'''(t)}{\alpha'(t)} = 0, \qquad a < t < b. \tag{7}$$

Equation (7) is often referred to as Kummer's equation, after E. E. Kummer who studied it in [11]. Kummer's equation is intimately related to the better known Riccati equation

$$r'(t) + (r(t))^2 + q(t) = 0, \qquad a < t < b, \tag{8}$$

satisfied by the logarithmic derivatives of the solutions of (1). In particular, if α is a phase function for (1), then

$$r(t) = i\alpha'(t) - \frac{1}{2}\frac{\alpha''(t)}{\alpha'(t)} \tag{9}$$

is a solution of (8).

When q is positive, almost all solutions of Kummer's equation are oscillatory. However, it was recognized long ago that there exist some solutions which can be asymptotically approximated by nonoscillatory functions, provided q is slowly varying. This is the basis of many classical asymptotic techniques, including the Liouville-Green method. If q is strictly positive on the interval $[a, b]$, then

$$u_0(t) = \frac{\cos(\alpha_0(t))}{\sqrt{\alpha_0'(t)}} \quad \text{and} \quad v_0(t) = \frac{\sin(\alpha_0(t))}{\sqrt{\alpha_0'(t)}}, \tag{10}$$

where α_0 is defined via

$$\alpha_0(t) = \int_a^t \sqrt{q(s)}\, ds, \tag{11}$$

are a pair of Liouville-Green approximates for solutions of (1). When q is slowly varying, obviously so too is the function defined via (11), and this is the case regardless of the magnitude of q. For a careful discussion of the Liouville-Green method, including rigorous error bounds for the approximates (10), we refer the reader to Chapter 6 of [1]. A higher order asymptotic method which operates by iteratively refining the Liouville-Green phase α_0 is developed in [3, 4, 5].

A theorem showing the existence of a nonoscillatory phase function for (1) in the case in which $q(t) = \lambda^2 q_0(t)$ with q_0 smooth and strictly positive on $[a, b]$ appears in [7]. It applies when the function $p(x) = \widetilde{p}(t(x))$, where $\widetilde{p}(t)$ is

defined via

$$\widetilde{p}(t) = \frac{1}{q_0(t)} \left(\frac{5}{4} \left(\frac{q_0'(t)}{q_0(t)} \right)^2 - \frac{q_0''(t)}{q_0(t)} \right) = 4 \left(q_0(t) \right)^{\frac{1}{4}} \frac{d}{dt} \left(\frac{1}{\left(q_0(t) \right)^{\frac{1}{4}}} \right) \tag{12}$$

and $t(x)$ is the inverse function of

$$x(t) = \int_a^t \sqrt{q_0(s)}\, ds, \tag{13}$$

has a rapidly decaying Fourier transform. More explicitly, the theorem asserts that if the Fourier transform of p satisfies a bound of the form

$$|\widehat{p}(\xi)| \leq \Gamma \exp\left(-\mu \left| \xi \right| \right), \tag{14}$$

then there exist functions ν and δ such that

$$|\nu(t)| \leq \frac{\Gamma}{2\mu} \left(1 + \frac{4\Gamma}{\lambda} \right) \exp(-\mu\lambda), \tag{15}$$

$$\left| \widehat{\delta}(\xi) \right| \leq \frac{\Gamma}{\lambda^2} \left(1 + \frac{2\Gamma}{\lambda} \right) \exp(-\mu|\xi|) \tag{16}$$

and

$$\alpha(t) = \lambda \sqrt{q_0(t)} \int_a^t \exp\left(\frac{\delta(u)}{2} \right)\, du \tag{17}$$

is a phase function for

$$y''(t) + \lambda^2 \left(q_0(t) + \frac{\nu(t)}{4\lambda^2} \right) y(t) = 0. \tag{18}$$

We note that when λ is of moderate size, Equation (18) is identical to (1) for the purposes numerical computation. The definition of the function $p(x)$ is ostensibly quite complicated; however, $p(x)$ is, in fact, simply equal to twice the Schwarzian derivative of the inverse function $t(x)$ of (13). This theorem ensures that, even for relatively modest values of λ, (1) admits a phase function which is nonoscillatory for the purposes of numerical computation. Of course, when λ is small, the frequency of oscillation of any phase function for (1) is modest.

Remark 1. *When discussing phase functions for second order linear ordinary differential equations it is standard practice to restrict attention to the special form (1). This is because Kummer's equation and the phase function α are invariant under the standard transformation which takes the more general ODE*

$$y''(t) + p(t)y'(t) + q(t)y(t) = 0, \qquad a < t < b, \tag{19}$$

to the form (1). To see this, we suppose that α is a phase function for (19) so that α' is positive on (a, b) and

$$\sqrt{\frac{\omega(t)}{\alpha'(t)}} \cos\left(\alpha(t) \right) \quad \textit{and} \quad \sqrt{\frac{\omega(t)}{\alpha'(t)}} \sin\left(\alpha(t) \right), \tag{20}$$

where

$$\omega(t) = \exp\left(-\int p(t)\ dt\right), \tag{21}$$

form a basis in its space of solutions. We note that by Abel's theorem, the Wronskian of any pair of independent solutions of (19) is a constant multiple of ω, and so the factors of $\sqrt{\omega(t)}$ appearing in (20) are necessary. It can be easily shown that α' satisfies

$$q(t) - \frac{(p(t))^2}{4} - \frac{p'(t)}{2} - (\alpha'(t))^2 + \frac{3}{4}\left(\frac{\alpha''(t)}{\alpha'(t)}\right)^2 - \frac{1}{2}\frac{\alpha'''(t)}{\alpha'(t)} = 0, \qquad a < t < b. \tag{22}$$

If we let

$$z(t) = \exp\left(-\frac{1}{2}\int p(t)\ dt\right) y(t), \tag{23}$$

where the choice of antiderivative is immaterial, then z solves the second order linear ordinary differential equation

$$z''(t) + \left(q(t) - \frac{(p(t))^2}{4} - \frac{p'(t)}{2}\right) z(t) = 0 \tag{24}$$

and substituting the coefficient in (24) into Kummer's equation (7) simply yields (22).

2.2 A numerical algorithm for constructing nonoscillatory phase functions

In this subsection, we describe the numerical method of [6] for constructing a nonoscillatory phase function for equations of the form (1) in the event that the coefficient q is slowly varying and positive on the interval $[a, b]$. The principal difficulty in designing such an algorithm is that while there exists a nonoscillatory solution of (7), almost all solutions of Kummer's equation are oscillatory and some mechanism must be used to identify a nonoscillatory solution. The algorithm of [6] addresses this problem by introducing a "windowed" version of the coefficient $\widetilde{q}$ which is equal to a constant ν^2 on the right quarter of the interval $[a, b]$ and nearly equal to the original coefficient q on the left quarter of $[a, b]$. The equation

$$y''(t) + \widetilde{q}(t)y(t) = 0 \tag{25}$$

admits a nonoscillatory phase function $\widetilde{\alpha}$ which is equal to $i\nu t$ on the right quarter of $[a, b]$ and closely approximates a nonoscillatory phase function for q on the left quarter of $[a, b]$. Accordingly, the algorithm solves an initial value problem for Kummer's equation to calculate the values of a nonoscillatory phase function for (1) and its first few derivatives at the point b. A terminal value problem for Kummer's equation is then solved to construct the desired phase function over the entire interval $[a, b]$.

We now give a more detailed description of the algorithm of [6]. It takes as inputs the endpoints a and b, an external subroutine for evaluating the coefficient q, an integer parameter k which determines the order of the piecewise Chebyshev expansions used to represent the outputs and a real-valued parameter ϵ which specifies the desired precision for the calculation. The outputs of the algorithm comprise three k^{th} order piecewise Chebyshev expansions which represent the phase function α and its first two derivatives α' and α''. By an k^{th} order piecewise Chebyshev expansions on the interval $[a, b]$, we mean a sum of the form

$$\begin{aligned}\sum_{i=1}^{m-1} \chi_{[x_{i-1},x_i)}(t) \sum_{j=0}^{k} c_{ij}\ T_j\left(\frac{2}{x_i - x_{i-1}}t + \frac{x_i + x_{i-1}}{x_i - x_{i-1}}\right)\\ +\chi_{[x_{m-1},x_m]}(t) \sum_{j=0}^{k} c_{mj}\ T_j\left(\frac{2}{x_m - x_{m-1}}t + \frac{x_m + x_{m-1}}{x_m - x_{m-1}}\right),\end{aligned} \tag{26}$$

where $a = x_0 < x_1 < \cdots < x_m = b$ is a partition of $[a, b]$, χ_I is the characteristic function on the interval I and T_j denotes the Chebyshev polynomial of degree j.

We let

$$\nu = \sqrt{q\left(\frac{a+b}{2}\right)} \tag{27}$$

and define $\widetilde{q}$ via the formula

$$\widetilde{q}(t) = \phi(t)\nu^2 + (1 - \phi(t))q(t), \tag{28}$$

where ϕ is given by

$$\phi(t) = \frac{1 + \operatorname{erf}\left(\frac{12}{b-a}\left(t - \frac{a+b}{2}\right)\right)}{2}. \tag{29}$$

The constant in (29) is chosen so that

$$|\phi(a)|\,,\ |\phi(b) - 1| < \epsilon_0, \tag{30}$$

where ϵ_0 denotes IEEE double precision machine zero. We next solve the terminal value problem

$$\begin{cases}\widetilde{q}(t) - (\widetilde{\alpha}'(t))^2 + \dfrac{3}{4}\left(\dfrac{\widetilde{\alpha}''(t)}{\widetilde{\alpha}'(t)}\right)^2 - \dfrac{1}{2}\dfrac{\widetilde{\alpha}'''(t)}{\widetilde{\alpha}'(t)} = 0, \qquad a < t < b,\\ \widetilde{\alpha}'(b) = \nu\\ \widetilde{\alpha}''(b) = 0\end{cases} \tag{31}$$

using the algorithm described in Appendix A. We take the precision parameter for that algorithm to be the input parameter ϵ and the integer parameter specifying the order of the piecewise Chebyshev expansions it outputs to be the input parameter k. Although the algorithm produces k^{th} order piecewise Chebyshev expansions representing the functions $\widetilde{\alpha}'$ and $\widetilde{\alpha}''$, it is only the values of these functions at the point a which concern us.

Next, we use the method of Appendix A to solve the initial value problem

$$\begin{cases} q(t) - (\alpha'(t))^2 + \frac{3}{4}\left(\frac{\alpha''(t)}{\alpha'(t)}\right)^2 - \frac{1}{2}\frac{\alpha'''(t)}{\alpha'(t)} = 0, & a < t < b, \\ \alpha'(a) = \widetilde{\alpha}'(a) \\ \alpha''(a) = \widetilde{\alpha}''(a). \end{cases} \tag{32}$$

Once again, the precision parameter is taken to be ϵ and the integer parameter is taken to be k. The outputs of the procedure consist of k^{th} order piecewise Chebyshev expansions representing α' and α''. A k^{th} order piecewise Chebyshev expansion representing the phase function α itself is constructed via spectral integration; the particular choice of antiderivative is largely irrelevant, and we determine it through the requirement that $\alpha(a) = 0$.

3 Levin methods

3.1 The classical Levin method

The classical Levin method [9] is a technique for rapidly evaluating integrals of the form

$$\int_a^b \exp(ig(t))f(t)\ dt \tag{33}$$

in the event that f is a slowly varying function, g is a slowly varying real-valued function and g' is of large magnitude. It proceeds by constructing a solution to the ordinary differential equation

$$p'(t) + ig'(t)p(t) = f(t), \qquad a < t < b. \tag{34}$$

Then

$$\frac{d}{dt}\left(\exp(ig(t))p(t)\right) = \exp(ig(t))f(t) \tag{35}$$

and the value of (33) is given by

$$p(b)\exp(ig(b)) - p(a)\exp(ig(a)). \tag{36}$$

This procedure is more efficient than standard methods under the above assumptions because, as shown in [9], (34) admits a nonoscillatory solution p_0 in this case.

The differential equation (34) is typically solved via a spectral collocation method. Although the operator

$$D\left[p\right](t) = p'(t) + ig'(t)p(t) \tag{37}$$

appearing on the left-hand side of (34) has a nontrivial nullspace, the resulting discretization of the operator will be well-conditioned as long an appropriate discretization scheme is used. In particular, it is necessary to chose a collocation grid sufficient to resolve f and g but not the nullspace of the operator D, which consists of all multiples of the function

$$\eta(t) = \exp(-ig(t)). \tag{38}$$

When g' is not of sufficiently large magnitude, it is not possible to choose such a grid of collocation points and, in this event, the matrix discretizing (34) will have a small singular value and be ill-conditioned. This phenomenon is known as "low-frequency breakdown" and it has long been viewed as a barrier to applying the Levin method in such cases.

3.2 The adaptive Levin method

The articles [12, 13] present experimental evidence indicating that when a Chebyshev spectral method is used to discretize (34) and the resulting linear system is solved via a truncated singular value decomposition, no low-frequency breakdown seems to occur. In [8], a proof that this is, in fact, the case is presented and it is observed that the lack of low-frequency breakdown makes it possible to use the Levin method as the basis of an adaptive integration scheme. More explicitly, because subdividing the interval $[a, b]$ over which (33) is given has the practical effect of reducing the magnitude of g', an adaptive algorithm is only viable if high accuracy can be achieved even when the magnitude of g' is small. One of the remarkable features of the adaptive Levin method introduced in [8] is that it is still quite efficient when g has saddle points; i.e., locations where one or more of the derivatives of g vanishes.

The procedure of [8] is quite simple — it is completely analogous to an adaptive Gaussian quadrature scheme, but on each subinterval $[a_0, b_0]$ considered, it uses a Chebyshev spectral method to solve

$$p'(t) + ig'(t)p(t) = f(t), \qquad a_0 < t < b_0, \tag{39}$$

and estimates the value of

$$\int_{a_0}^{b_0} \exp(ig(x))f(x)\ dx \tag{40}$$

via the difference

$$p(b_0)\exp(ig(b_0)) - p(a_0)\exp(ig(a_0)) \tag{41}$$

rather than using a Gaussian quadrature rule to estimate (40) more directly.

3.3 Numerical algorithm

In this subsection, we describe a variant of the adaptive Levin algorithm for efficiently evaluating the particular solution (4). It operates by constructing a collection of functions $p_1, \ldots, p_m$ that allow for the efficient calculation of

$$\int_a^t \exp(ig(s))\widetilde{f}(s)\ ds, \tag{42}$$

where

$$g(s) = \alpha(s) \tag{43}$$

and

$$\widetilde{f}(s) = \frac{f(s)}{\sqrt{\alpha'(s)}}. \tag{44}$$

Since

$$\begin{aligned} \int_a^t \exp(ig(s))\widetilde{f}(s)\ ds &= \int_a^t \frac{\exp(i\alpha(s))}{\sqrt{\alpha'(s)}} f(s)\ ds \\ &= \int_a^t u(s)f(s)\ ds + i\int_a^t v(s)f(s)\ ds, \end{aligned} \tag{45}$$

the particular solution defined via (4) can be evaluated at a point t if the values of $u(t)$, $v(t)$ and the integral (42) are known.

The algorithm takes as input a subroutine for evaluating the phase function α and its first derivative, as well as the input function f, a positive integer k and a real-valued parameter ϵ specifying the desired accuracy for the computations. The output of the algorithm comprises a partition

$$a = a_0 < a_1 < \ldots < a_m = b \tag{46}$$

of the interval $[a, b]$ and a collection of m Chebyshev expansions. Each Chebyshev expansion is of order $(k-1)$ and the j^{th} expansion represents the j^{th} output function p_j, which is defined over the interval $[a_{j-1}, a_j]$ and closely approximates a solution of the differential equation

$$p'(t) + ig'(t)p(t) = \widetilde{f}(t), \qquad a_{j-1} < t < a_j. \tag{47}$$

It follows that (42) can be calculated by finding the least positive integer j such that $t \le a_j$ and then computing the sum

$$\begin{aligned} &\Big[p_j(t)\exp(ig(t)) - p_j(a_{j-1})\exp(ig(a_{j-1}))\Big] \\ &+ \sum_{i=1}^{j-1}\Big[p_i(a_i)\exp(ig(a_i)) - p_i(a_{i-1})\exp(ig(a_{i-1}))\Big]. \end{aligned} \tag{48}$$

The algorithm maintains a list of subintervals of $[a, b]$ which have been processed and a list of "accepted" subintervals of $[a, b]$. Upon completion of the procedure, the list of "accepted subintervals" determines the partition (46). Initially, the list of subintervals to process contains $[a, b]$ and the list of accepted subintervals is empty. The following sequence of operations is performed repeatedly until the list of subintervals to process is empty:

1. Remove a subinterval $[a_0, b_0]$ from the list of subintervals to process.
2. Form the k-point extremal Chebyshev grid $x_1, \ldots, x_k$ on the interval $[a_0, b_0]$; that is, let

$$x_j = \frac{b_0 - a_0}{2}\cos\left(\pi\frac{k-j}{k-1}\right) + \frac{b_0 + a_0}{2}, \qquad j = 1, \ldots, k. \tag{49}$$

3. Form the $k \times k$ Chebyshev spectral differentiation matrix B which takes the vector

$$\begin{pmatrix} h(x_1) \\ h(x_2) \\ \vdots \\ h(x_k) \end{pmatrix} \tag{50}$$

of values of any expansion of the form

$$h(t) = \sum_{j=0}^{k-1} c_j T_j \left(\frac{2}{b_0 - a_0} t + \frac{b_0 + a_0}{b_0 - a_0} \right) \tag{51}$$

at the extremal Chebyshev nodes to the vector

$$\begin{pmatrix} h'(x_1) \\ h'(x_2) \\ \vdots \\ h'(x_k) \end{pmatrix} \tag{52}$$

of the values of its derivative at the extremal Chebyshev nodes.

4. Call the external subroutine provided as input to evaluate g, g' and f at the points $x_1, \ldots, x_k$ and form the $k \times k$ matrix

$$A = B + i \begin{pmatrix} g'(x_1) & & & \\ & g'(x_2) & & \\ & & \ddots & \\ & & & g'(x_k) \end{pmatrix} \tag{53}$$

which discretizes the restriction of the differential operator D defined in (37) to the interval $[a_0, b_0]$.

5. Solve the linear system

$$A \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{pmatrix} = \begin{pmatrix} \widetilde{f}(x_1) \\ \widetilde{f}(x_2) \\ \vdots \\ \widetilde{f}(x_k) \end{pmatrix}, \tag{54}$$

where $\widetilde{f}$ is defined in (44), using a truncated singular value decomposition. To do so, first construct the singular value decomposition

$$A = \begin{pmatrix} U_1 & U_2 & \cdots & U_k \end{pmatrix} \begin{pmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_k \end{pmatrix} \begin{pmatrix} V_1 & V_2 & \cdots & V_k \end{pmatrix}^* \tag{55}$$

of the matrix A. Then, find the least integer $1 \leq l \leq k$ such that $\sigma_l > 10\epsilon_0 \|A\|_F$, where ϵ_0 is machine zero and $\|A\|_F$ is the Frobenius norm of A.

Finally, let

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{pmatrix} = \begin{pmatrix} V_1 & \cdots & V_l \end{pmatrix} \begin{pmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_l \end{pmatrix}^{-1} \begin{pmatrix} U_1 & \cdots & U_l \end{pmatrix}^* \begin{pmatrix} \widetilde{f}(x_1) \\ \widetilde{f}(x_2) \\ \vdots \\ \widetilde{f}(x_k) \end{pmatrix}. \tag{56}$$

6. Calculate the coefficients $c_0, \ldots, c_{k-1}$ such that the Chebyshev expansion defined via

$$\widetilde{p}(t) = \sum_{j=0}^{k-1} c_j T_j \left(\frac{2}{b_0 - a_0} t + \frac{b_0 + a_0}{b_0 - a_0} \right) \tag{57}$$

satisfies

$$\widetilde{p}(x_1) = y_1, \ \widetilde{p}(x_2) = y_2, \ \ldots, \ \widetilde{p}(x_k) = y_k. \tag{58}$$

7. If

$$\frac{\sum_{j=\lfloor k/2 \rfloor}^{k-1} c_j^2}{\sum_{j=0}^{k-1} c_j^2} < \epsilon^2, \tag{59}$$

then add the interval $[a_0, b_0]$ to the list of accepted intervals. In this case, the interval $[a_0, b_0]$ becomes one of the subintervals $[a_j, a_{j+1}]$ in the partition (46) and (57) becomes the output function p_j. Otherwise, add the intervals $[a_0, c_0]$ and $[c_0, b_0]$, where $c_0 = (a_0 + b_0)/2$, to the list of subintervals to process.

When the spectral discretization on an interval $[a_0, b_0]$ suffices to resolve f, g' and the nonoscillatory solution p_0 whose existence is established in [9], the $(k-1)^{st}$ order Chebyshev expansion which agrees with p_0 at the k-point Chebyshev extremal grid on $[a_0, b_0]$ approximates a solution of (54) with high accuracy. If, in addition, (53) is invertible — which is the case when g' is of sufficiently large magnitude — it follows that the obtained solution (57) will closely approximate p_0. As a result, when g' is of sufficient magnitude, the functions $p_1, \ldots, p_m$ produced by the procedure just described comprise a piecewise Chebyshev discretization of the Levin solution p_0 of the equation (34).

However, when g' is not of sufficient magnitude, the matrix (53) is noninvertible and the obtained solution (57) is a linear combination of p_0 and some other function (when the nullspace is fully resolved, that other function is a multiple of (38), but this need not be the case when the discretization fails to fully resolve η). Consequently, when g' is of insufficient magnitude, there is no guarantee that the functions $p_1, \ldots, p_m$ will approximate p_0 or even that $p_j(b_j)$ will equal $p_{j+1}(a_j)$. Nonetheless, Formula (48) remains an accurate and efficient mechanism for evaluating (42). We note that because $p_j(b_j)$ need not be equal to $p_{j+1}(a_j)$, it is not possible to apply the obvious simplification to (42).

Remark 2. *The truncated singular value decomposition is quite expensive. In our implementation of the method described in this subsection, we used a rank-revealing QR decomposition in lieu of the truncated singular value decomposition to solve the linear system which results from discretizing the Levin equation. This was found to be about 5 times faster and lead to no apparent loss in accuracy.*

4 Numerical experiments

In this section, we present the results of numerical experiments which were conducted to illustrate the properties of the algorithm of this paper. The code for these experiments was written in Fortran and compiled with version 12.10 of the GNU Fortran compiler. They were performed on a workstation computer equipped with an AMD 3995WX processor and 512GB of memory. No attempt was made to parallelize our code (i.e., only a single processor core was used).

In the course of conducting the experiments for this article we found that, in most cases, our method obtains higher accuracy than the conventional solver described in Appendix A. Accordingly, in almost all of the experiments described here, we measured the accuracy of the solutions obtained via our algorithm by comparison with reference solutions calculated by running the conventional solver of Appendix A using quadruple precision arithmetic (i.e., using Fortran REAL*16 numbers). These extended precision calculations required a great deal of time and memory, which is the reason that we conducted the experiments on a workstation computer equipped with a large amount of memory. The experiments of Subsection 4.1, which concern a terminal value problem whose solution is explicitly known, are the exceptions.

In all of the experiments discussed here, 15^{th} order piecewise Chebyshev expansions were used to represent phase functions as well as the functions $p_1, \ldots, p_m$, and the tolerance parameter ϵ passed to the algorithms of Subsections 2.2 and 3.3 was taken to be to 10^{-13}. Whenever the conventional solver was deployed, 15^{th} order piecewise Chebyshev expansions were used to represent the obtained solutions.

4.1 A terminal value problem with a known solution

In our first set of experiments, we used both the phase function method and the conventional approach described in Appendix A to solve the terminal value problem

$$\begin{cases} y''(t) - \lambda^2 t y(t) = \lambda^2 t^2, \quad -10 < t < 0, \\ y(0) = \dfrac{1}{3^{\frac{2}{3}}\Gamma\left(\frac{2}{3}\right)} \\ y'(0) = -1 - \dfrac{\lambda^{\frac{2}{3}}\Gamma\left(-\frac{1}{3}\right)}{2\pi\ 3^{\frac{5}{6}}}. \end{cases} \tag{60}$$

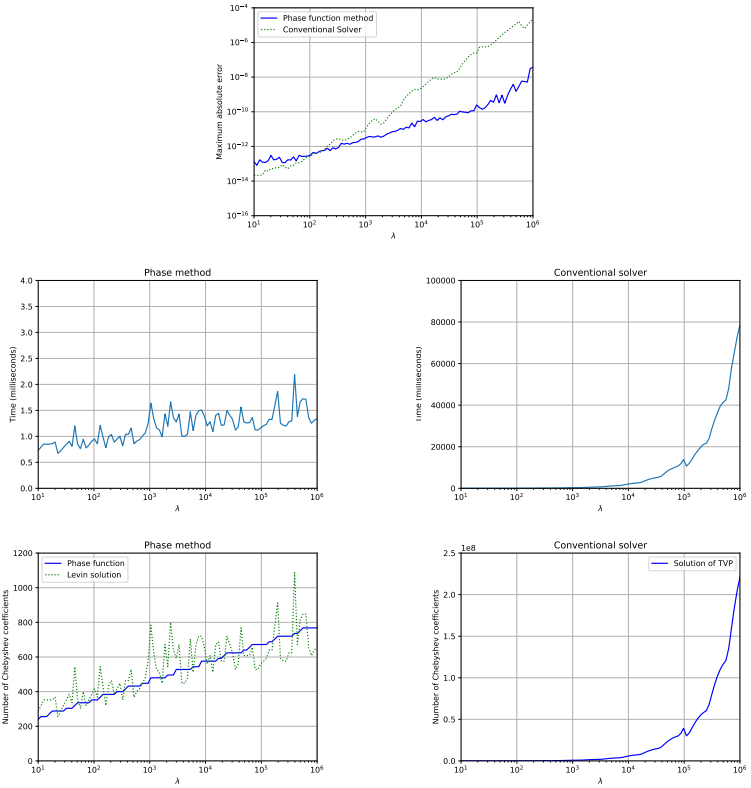

Fig. 1: The results of the first set of experiments of Subsection 4.1. The plot in the first row gives the maximum observed absolute errors in the solutions obtained by the phase function method and the conventional solver as functions of λ. The plot on the left-hand side of the second row gives the time required by the phase method as a function of λ, while the plot on the right-hand side of the second row gives the time required by the conventional solver as a function of λ. The plot on the left-hand side of the third row gives the number of coefficients in the piecewise Chebyshev expansions of the phase function and the Levin solution as functions of λ, whereas the plot on the right-hand side of the third row gives the number of coefficients in the piecewise Chebyshev expansion of the solution of (60) produced by the conventional solver.

It can be easily verified that the solution of (60) is

$$y(t) = -t + \mathrm{Ai}\left(\lambda^{\frac{2}{3}} t\right), \tag{61}$$

where Ai denotes the Airy function of the first kind (see, for instance, [1] for a discussion of the Airy functions and a definition of the function Ai).

Because the condition number of the problem (60) and the condition number of evaluation of the function (61) increase with λ, we took the precision parameter

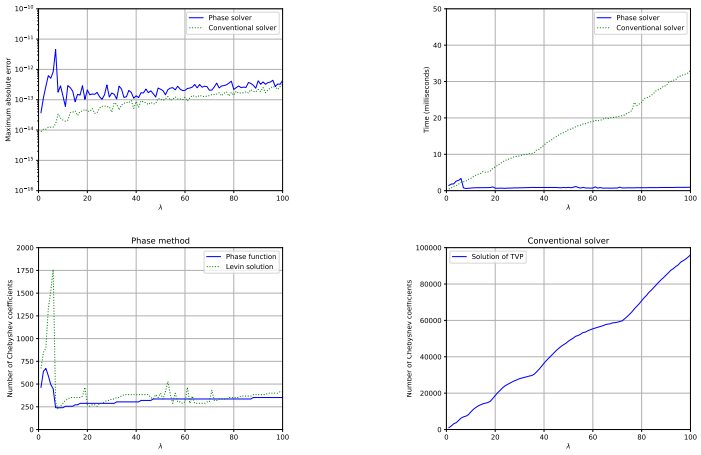

Fig. 2: The results of the second set of experiments of Subsection 4.1, which demonstrate the behavior of our algorithm for values of the parameter λ between 1 and 100. The plots in the first row compare the running time and accuracy of the phase method with a conventional solver. The plot in the lower left gives the number of Chebyshev coefficients used to represent the phase function and the Levin solution, while the plot in the lower right gives the number of Chebyshev coefficients used to represent the solution produced by the conventional solver.

for the conventional solver to be

$$\epsilon_{\text{conventional}} = \max\left\{10^{-13}, \epsilon_0 \times \lambda\right\}, \tag{62}$$

where $\epsilon_0 \approx 2.220446049250313 \times^{-16}$ is machine zero for IEEE double precision arithmetic. As in all of the experiments discussed in this paper, we set the parameter specifying the desired precision for the phase function and for the algorithm of Subsection 3.3 to be $\epsilon = 10^{-13}$.

We began our first experiment by sampling $m = 100$ equispaced points $x_1, \ldots, x_m$ in the interval $[1, 6]$. Then, for each $\lambda = 10^{x_1}, 10^{x_2}, \ldots, 10^{x_m}$, we solved (60) using both the phase function method and the conventional solver. The wall clock time required by each method was measured, as was the number of coefficients in the piecewise Chebyshev expansion used to represent the phase function and the total number of Chebyshev coefficients in the expansions of the functions $p_1, \ldots, p_m$ produced by the algorithm of Subsection 3.3. In a mild abuse of terminology, we refer to this latter quantity in the rest of this section and in our figures as the number of piecewise Chebyshev coefficients needed to represent the Levin solution. Each of the two obtained solutions was then evaluated at $10,000$ equispaced points in the interval $(-10, 0)$ and the largest observed absolute errors recorded. The errors were, of course, measured by comparison with the known solution (61). We measured absolute errors rather than relative errors because (61) is an oscillatory function with many zeros on the interval $(-10, 0)$. The results are presented in Figure 1. The plot

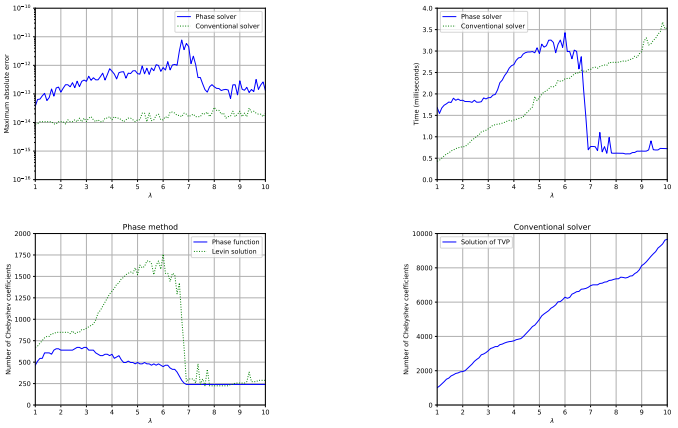

Fig. 3: The results of the third set of experiments of Subsection 4.1, which demonstrate the behavior of our algorithm for values of the parameter λ between 1 and 10. The plots in the first row compare the accuracy and running time of the phase method with a conventional solver. The plot in the lower left gives the number of Chebyshev coefficients used to represent the phase function and the Levin solution, while the plot in the lower right gives the number of Chebyshev coefficients used to represent the solution produced by the conventional solver.

at the top of that figure gives the maximum observed absolute errors in the solutions obtained by the phase function method and the conventional solver as functions of λ. The plot on the left-hand side of the second row gives the time required by the phase method as a function of λ, while the plot on the right-hand side of the second row gives the time required by the conventional solver as a function of λ. The plot on the left-hand side of the third row gives the number of coefficients in the piecewise Chebyshev expansions of the phase function and the Levin solution. The plot on the right-hand side of the third row gives the number of coefficients in the piecewise Chebyshev expansion of the solution of (60) produced by the conventional solver.

Two more experiments concerning the problem (60) were performed to test the behavior of the algorithm of this paper in the case of small values of λ. In the first of these experiments, we sampled $m = 100$ equispaced points $\lambda_1, \lambda_2, \ldots, \lambda_m$ in the interval $[1, 100]$. For each obtained value of λ, we repeated the procedure described above. The results are given in Figure 2. In the second of these experiments, we sampled $m = 100$ equispaced points $\lambda_1, \lambda_2, \ldots, \lambda_m$ in the interval $[1, 10]$ and repeated the experiments described above for each obtained value of λ. The results are shown in Figure 3.

4.2 An initial value problem

In our next experiment, we solved the initial value problem

$$\begin{cases} y''(t) + \left(\dfrac{\lambda^2}{0.01 + t^2}\right) y(t) = \lambda^2 (1 + t) \cos(13t^2), \quad 0 < t < 1, \\ y(0) = y'(0) = 1. \end{cases} \tag{63}$$

for various values of λ using the phase function method. More explicitly, we first sampled $m = 100$ points $x_1, x_2, \ldots, x_m$ in the interval $[1, 6]$. Then, for each $\lambda = 10^{x_1}, 10^{x_2}, \ldots, 10^{x_m}$, we used the algorithm of this paper to solve (63). For each λ, we recorded the wall clock time spent solving the problem and measured the error in the obtained solution at $10,000$ equispaced points in the interval $[0, 1]$. The reference solution was obtained by solving (63) using the conventional solver running in quadruple precision (REAL*16) arithmetic. The results are presented in Figure 4.

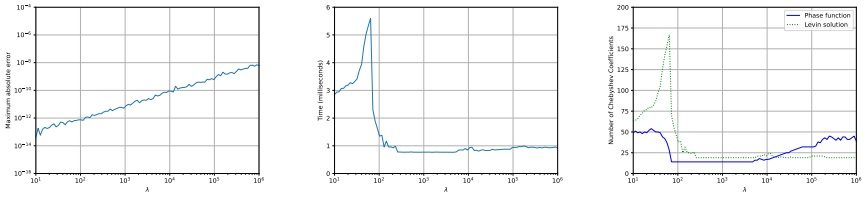

Fig. 4: The results of the experiments of Subsection 4.2. The plot on the left gives the maximum observed absolute error as a function of λ; the middle plot gives the wall clock time required to solve (63) as a function of λ; and the plot on the right-hand side gives the number of coefficients in the piecewise Chebyshev expansions of the phase function and the Levin solution as functions of the parameter λ.

4.3 A boundary value problem

In the experiment described in this subsection, we solved the boundary value problem

$$\begin{cases} y''(t) + \left(\dfrac{\lambda^3 \left(\frac{3}{2} + \cos\left(\log(\lambda) t\right)\right)}{1 + \lambda \exp(t)}\right) y(t) = \dfrac{\lambda^2}{\sqrt{2 + t}}, \quad -1 < t < 1, \\ y(-1) = y(1) = 0 \end{cases} \tag{64}$$

for various values of λ. More explicitly, we first sampled $m = 100$ points $x_1, x_2, \ldots, x_m$ in the interval $[1, 6]$. Then, for each $\lambda = 10^{x_1}, 10^{x_2}, \ldots, 10^{x_m}$, we used the algorithm of this paper to solve (64). For each λ, we recorded the wall clock time spent solving the problem and measured the error in the obtained solution at $10,000$ equispaced points in the interval $[-1, 1]$. The reference solution was obtained by solving (64) using the conventional solver running in quadruple precision (REAL*16) arithmetic. The results are presented in Figure 5. We note that the coefficient in the differential equation

appearing in (64) becomes more oscillatory with λ, so we expect the running time of our algorithm to grow with λ. And while this is, in fact, the case, the effect is remarkably mild.

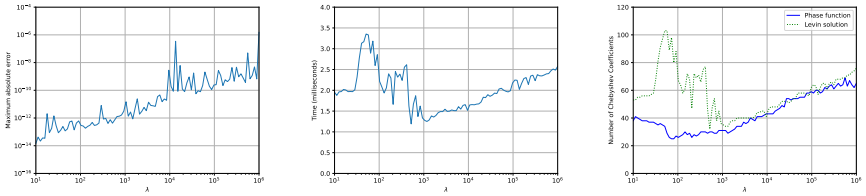

Fig. 5: The results of the experiments of Subsection 4.3. The plot on the left gives the maximum observed absolute error as a function of λ; the middle plot gives the wall clock time required to solve (64) as a function of λ; and the plot on the right-hand side gives the number of coefficients in the piecewise Chebyshev expansions of the phase function and the Levin solution as functions of the parameter λ.

4.4 A boundary value problem with more complicated boundary conditions

In a final experiment, we solved the boundary value problem

$$\begin{cases} y''(t) + \left(\dfrac{\lambda^2 \left(2 + t^2 \cos(\lambda)\right)}{1 + t^2} \right) y(t) = \lambda^2 \cos(3t^2), \quad -1 < t < 1, \\ y(-1) = y(1) \\ y'(-1) = y'(1) \end{cases} \tag{65}$$

for various values of λ. We first sampled $m = 100$ points $x_1, x_2, \ldots, x_m$ in the interval $[1, 6]$. Then, for each $\lambda = 10^{x_1}, 10^{x_2}, \ldots, 10^{x_m}$, we used the algorithm of this paper to solve (65). For each λ, we recorded the wall clock time spent solving the problem and measured the error in the obtained solution at $10,000$

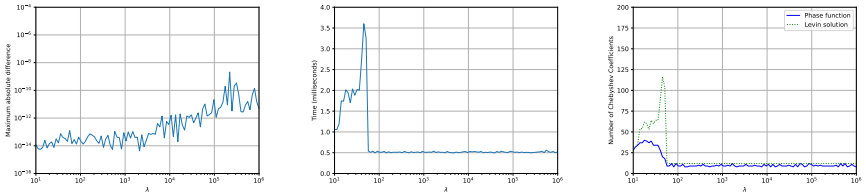

Fig. 6: The results of the experiments of Subsection 4.4. The plot on the left gives the maximum observed absolute error as a function of λ; the middle plot gives the wall clock time required to solve (65) as a function of λ; and the plot on the right-hand side gives the number of coefficients in the piecewise Chebyshev expansions of the phase function and the Levin solution as functions of the parameter λ.

equispaced points in the interval $[-1, 1]$. The reference solution was obtained by solving (65) using the conventional solver running in quadruple precision (REAL*16) arithmetic. The results are presented in Figure 6.

5 Conclusions

We have introduced a scheme for solving second order inhomogeneous linear ordinary differential equations of the form

$$y''(t) + q(t)y(t) = f(t), \qquad a < t < b, \tag{66}$$

in the case in which f is real-valued and slowly varying and q is positive and slowly varying. Unlike standard solvers, its running time is independent of the magnitude of q and, unlike asymptotic methods, the accuracy it obtains is consistent with the condition number of the problem even when q is of small magnitude.

We chose the approach of this paper over another, more direct, Levin-type method precisely because we wanted an algorithm which achieves high-accuracy regardless of the magnitude of q. When f and q are slowly varying and q is of sufficiently large magnitude, the equation (3) admits a nonoscillatory solution and applying an algorithm analogous to that of Subsection 2.2 to it will result in a piecewise Chebyshev expansion representing that solution. However, such an approach will fail when q is of small magnitude for the reasons discussed at the end of Subsection 2.2. Nonetheless, it would be of some interest to determine when this more direct approach is preferable to the algorithm of this paper.

As mentioned in the introduction, the representation (5) of the general form of the solution of (3) is numerically unstable when q is negative and of large magnitude on some or all of the interval $[a, b]$. In the authors' view, it is unlikely that an algorithm of the type described this paper can be successfully applied to such problems in general. Nonetheless, it is most likely possible to further develop the method of this article so that it applies to important classes of physically relevant problems for which the coefficient q is negative on some or all of the interval $[a, b]$.

6 Acknowledgments

KS was supported in part by the NSERC Discovery Grants RGPIN-2020-06022 and DGECR-2020-00356. JB was supported in part by NSERC Discovery grant RGPIN-2021-02613.

References

[1] Olver, F.W.J.: Asymptotics and Special Functions. A.K. Peters, Wellesley, Massachusetts (1997)

[2] Miller, P.D.: Applied Asymptotic Analysis. American Mathematical Society, Providence, Rhode Island (2006)

[3] Spigler, R., Vianello, M.: A numerical method for evaluating the zeros of solutions of second-order linear differential equations. Mathematics of Computation **55**, 591–612 (1990)

[4] Spigler, R., Vianello, M.: The phase function method to solve second-order asymptotically polynomial differential equations. Numerische Mathematik **121**, 565–586 (2012)

[5] Spigler, R.: Asymptotic-numerical approximations for highly oscillatory second-order differential equations by the phase function method. Journal of Mathematical Analysis and Applications **463**, 318–344 (2018)

[6] Bremer, J.: On the numerical solution of second order differential equations in the high-frequency regime. Applied and Computational Harmonic Analysis **44**, 312–349 (2018)

[7] Bremer, J., Rokhlin, V.: Improved estimates for nonoscillatory phase functions. Discrete and Continuous Dynamical Systems, Series A **36**, 4101–4131 (2016)

[8] Chen, S., Serkh, K., Bremer, J.: The adaptive Levin method. arXiv **2209.14561** (2022)

[9] Levin, D.: Procedures for computing one- and two-dimensional integrals of functions with rapid irregular oscillations. Mathematics of Computation **38**, 531–5538 (1982)

[10] Bremer, J.: Phase function methods for second order linear ordinary differential equations with turning points. arXiv **2209.14561** (2022)

[11] Kummer, E.E.: De generali quadam aequatione differentiali tertti ordinis. Progr. Evang. Köngil. Stadtgymnasium Liegnitz (1834)

[12] Li, J., Wang, X., Wang, T.: A universal solution to one-dimensional oscillatory integrals. Science in China Series F: Information Sciences **51**, 1614–1622 (2008)

[13] Li, J., Wang, X., Wang, T., Xiao, S.: An improved Levin quadrature method for highly oscillatory integrals. Applied Numerical Mathematics **60**(8), 833–842 (2010)

[14] Greengard, L.: Spectral integration and two-point boundary value problems. SIAM Journal of Numerical Analysis **28**, 1071–1080 (1991)

[15] Ascher, U.M., Petzold, L.R.: Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations. Society for Industrial and Applied Mathematics, USA (1998)

A An adaptive Chebyshev spectral solver for ordinary differential equations

The algorithm of this paper entails solving several ordinary differential equations. We use a fairly standard adaptive Chebyshev spectral solver to do so. We now briefly describe its operation in the case of the initial value problem

$$\begin{cases} \boldsymbol{y}'(t) = F(t, \boldsymbol{y}(t)), \quad a < t < b, \\ \boldsymbol{y}(a) = \boldsymbol{v} \end{cases} \tag{67}$$

where $F : \mathbb{R}^{n+1} \to \mathbb{R}^n$ is smooth and $\boldsymbol{v} \in \mathbb{R}^n$. The solver can be easily modified to apply to a terminal value problem.

The solver takes as input a positive integer k, a tolerance parameter ϵ, an interval (a, b), the vector $\boldsymbol{v}$ and a subroutine for evaluating the function F. It outputs n piecewise k^{th} order Chebyshev expansions, one for each of the components $y_i(t)$ of the solution $\boldsymbol{y}$ of (67).

The solver maintains two lists of subintervals of (a, b): one consisting of what we term "accepted subintervals" and the other of subintervals which have yet to processed. A subinterval is accepted if the solution is deemed to be adequately represented by a k^{th} order Chebyhev expansion on that subinterval. Initially, the list of accepted subintervals is empty and the list of subintervals to process contains the single interval (a, b) It then operates as follows until the list of subintervals to process is empty:

1. Find, in the list of subinterval to process, the interval (c, d) such that c is as small as possible and remove this subinterval from the list.
2. Solve the initial value problem

$$\begin{cases} \boldsymbol{u}'(t) = F(t, \boldsymbol{u}(t)), \quad c < t < d, \\ \boldsymbol{u}(c) = \boldsymbol{w} \end{cases} \tag{68}$$

   If $(c, d) = (a, b)$, then we take $\boldsymbol{w} = \boldsymbol{v}$. Otherwise, the value of the solution at the point c has already been approximated, and we use that estimate for $\boldsymbol{w}$ in (68).
   If the problem is linear, a straightforward integral equation method (see, for instance, [14]) is used to solve (68). Otherwise, the trapezoidal method (see, for instance, [15]) is first used to produce an initial approximation $\boldsymbol{y_0}$ of the solution and then Newton's method is applied to refine it. The linearized problems are solved using an integral equation method.
   In any event, the result is a set of k^{th} order Chebyshev expansions

$$u_i(t) \approx \sum_{j=0}^{k} \lambda_{ij} \, T_j \left(\frac{2}{d-c} t + \frac{c+d}{c-d} \right), \quad i = 1, \ldots, n, \tag{69}$$

   approximating the components $u_1, \ldots, u_n$ of the solution of (68).

3. Compute the quantities

$$\frac{\sqrt{\sum_{j=\lfloor k/2 \rfloor+1}^{k} \lambda_{ij}^2}}{\sqrt{\sum_{j=0}^{k} \lambda_{ij}^2}}, \quad i = 1, \ldots, n, \tag{70}$$

where the λ_{ij} are the coefficients in the expansions (69). If any of the resulting values is larger than ϵ, then we split the subinterval into two halves $\left(c, \frac{c+d}{2}\right)$ and $\left(\frac{c+d}{2}, d\right)$ and place them on the list of subintervals to process. Otherwise, we place the subinterval (c, d) on the list of accepted subintervals.

At the conclusion of this procedure, we have k^{th} order Chebyshev expansions for each component of the solution, with the list of accepted subintervals determining the partition for each expansion.