Numerical Solution of Linear, Nonhomogeneous Differential Equation Systems via Padé Approximation

Kenneth C. Johnson

KJ Innovation

kjinnovation@earthlink.net

(First posted November 1, 2016, revised November 30, 2016.)

http://vixra.org/abs/1611.0002

Abstract

This paper generalizes an earlier investigation of linear differential equation solutions via Padé approximation (viXra:1509.0286), for the case of nonhomogeneous equations. Formulas are provided for Padé polynomial orders 1, 2, 3, and 4, for both constant-coefficient and functional-coefficient cases. The scale-and-square algorithm for the constant-coefficient case is generalized for nonhomogeneous equations. Implementation details including step size initialization and tolerance control are discussed.

1. Introduction

An earlier study [1] investigated solutions of the linear differential equation $F'[x] = D[x]F[x]$ via Padé approximation: $F[h] \approx Q[h]^{-1} Q[-h]F[-h]$, where $Q[h]$ is a polynomial, D and Q are square matrices, and F may be a column vector or a multi-column matrix. (In this paper, square braces "[…]" delimit function arguments while round braces "(…)" are reserved for grouping.)

We consider here the more general nonhomogeneous equation,

$$F'[x] = D[x]F[x] + C[x]. \tag{1}$$

where C is a vector or matrix, size-matched to F. Eq. (1) can be recast in the form of a homogeneous equation,

$$\frac{d}{dx}\begin{pmatrix} F[x] \\ \mathbf{I} \end{pmatrix} = \begin{pmatrix} D[x] & C[x] \\ \mathbf{0} & \mathbf{0} \end{pmatrix}\begin{pmatrix} F[x] \\ \mathbf{I} \end{pmatrix}. \tag{2}$$

where $\mathbf{I}$ is an identity matrix. For this case, Eq's. (7) and (8) in [1] result in relations of the form

$$\begin{pmatrix} Q[h] & R[h] \\ \mathbf{0} & \mathbf{I} \end{pmatrix}\begin{pmatrix} F[h] \\ \mathbf{I} \end{pmatrix} - \begin{pmatrix} Q[-h] & R[-h] \\ \mathbf{0} & \mathbf{I} \end{pmatrix}\begin{pmatrix} F[-h] \\ \mathbf{I} \end{pmatrix} = O\,h^{2n+1};\ Q[0] = \mathbf{I},\ R[0] = \mathbf{0}. \tag{3}$$

where Q and R are matrix polynomials. This simplifies to

$$R[h] - R[-h] + Q[h]F[h] - Q[-h]F[-h] = O\,h^{2n+1}. \tag{4}$$

The Q polynomial has the form given in [1]; it is determined from D and has no dependence on C. The R polynomial depends on both D and C and has a linear dependence on C. In some cases $R[h]$ is an odd function of h (i.e., $R[-h] = -R[h]$), in which case the $R[h] - R[-h]$ term in Eq. (4) is replaced by $2\,R[h]$.

The homogeneous equation ($C = \mathbf{0}$ in Eq. (1)) has solutions of the form $F[x] = \Phi[x]F[0]$, where $\Phi[x]$ is the solution of the initial value problem,

$$\Phi'[x] = D[x]\Phi[x], \quad \Phi[0] = \mathbf{I}. \tag{5}$$

($\mathbf{I}$ is an identity matrix.) For the nonhomogeneous case, general solutions of Eq. (1) are of the form

$$F[x] = \Phi[x]\left(\int_0^x \Phi[t]^{-1}\,C[t]\,dt + F[0]\right). \tag{6}$$

For the special case of constant D, $\Phi[x]$ is an exponential matrix,

$$\Phi[x] = \exp[D\,x] \quad (\text{constant } D). \tag{7}$$

If C is also constant, Eq. (6) reduces to $F[x] = D^{-1}(\exp[D\,x] - \mathbf{I})\,C + \exp[D\,x]F[0]$, or more generally,

$$F[x + \Delta x] = D^{-1}(\exp[D\,\Delta x] - \mathbf{I})\,C + \exp[D\,\Delta x]F[x] \quad (\text{constant } D \text{ and } C). \tag{8}$$

This formula cannot be used when D is singular (even though the first left-hand term is well defined by its Taylor series), and it has poor numerical precision when D is near-singular or x is very small. But the Padé approximation method based on Eq. (4) does not have this limitation. The method can be used, for example, to robustly calculate $D^{-1}(\exp[D] - \mathbf{I})$, even for singular D, by setting $F[0] = \mathbf{0}$, $C = \mathbf{I}$, and $x = 1$.

If $C[x]$ can be an arbitrary linear combination of basis functions within a finite basis set, then particular solutions of Eq. (1) can be efficiently calculated by setting $F[0] = \mathbf{0}$ and setting $C[x]$ to a matrix containing all basis functions in its columns. The resulting $F[x]$ columns can be linearly combined to obtain particular solutions for any combination of $C[x]$ basis functions. The result can then be added to $\Phi[x]F[0]$ to obtain general solutions $F[x]$ for any $F[0]$.

Eq. (4) is used to integrate $F[x]$ across a small interval, from $x = -h$ to $x = h$. The independent variable x can be scaled and shifted to convert this to an integration from $x = x_0$ to $x = x_0 + \Delta x$ for a sufficiently small Δx, and multiple such integrations are concatenated to calculate $F[x]$ over a large integration interval. For the homogeneous, constant-coefficient case (D constant, $C = \mathbf{0}$), the concatenation can be efficiently implemented using a "scale-and-square" technique based on the relation

$$\exp[D\,x] = (\ldots((\exp[2^{-j}\,D\,x])\overbrace{^2)^2\ldots)^2}^{j\times}. \tag{9}$$

(For some sufficiently large integer j, a Padé approximant is used to calculate $\exp[2^{-j} D x]$, and the result is squared j times to obtain $\exp[D x]$.) This algorithm can be generalized for the nonhomogeneous case with constant D and C.

Section 2 lists polynomial functions Q and R in Eq. (4) for various Padé polynomial orders. Section 3 outlines the scale-and-square algorithm, generalized for the nonhomogeneous case. Section 4 discusses the choice of integration interval size. Appendix A discusses MATLAB® implementation details for the constant-coefficient case, and Appendix B provides Mathematica code validating the results of section 2.

MATLAB® implementation code and application test cases are posted on the MathWorks File Exchange [2]. The algorithms and code incorporate and extend the functionality of MATLAB's **`expm`** function [3-5], and provide an efficient alternative to MATLAB's differential equation solvers [6] (e.g., **`ode45`**) for linear equations.

2. Padé-approximation formulas

The Q and R polynomials in Eq. (4) are listed below for Padé polynomial orders 1, 2, 3, and 4, first for the case of constant D and C, and then for the non-constant case. For Padé order n, the approximation order is $2n$ (i.e., the approximation error is of order h^{2n+1}.) The constant-coefficient formulas (Eq's. (10)-(13)) include an estimate of the approximation error, which is useful for choosing the integration step size. For the non-constant-coefficient case (Eq's. (18)-(21)), similar error approximations would be too complex to be of much use, but the constant-coefficient error formulas can be used for step size initialization as described in section 4.

Padé order 1, constant D, C:

$$\begin{aligned} &Q[h] = \mathbf{I} - h D \\ &R[h] = -h C \\ &2R[h] + Q[h]F[h] - Q[-h]F[-h] = -\tfrac{2}{3} h^3 D^2 (C + D F[0]) + O h^5 \end{aligned} \tag{10}$$

Padé order 2, constant D, C:

$$\begin{aligned} &Q[h] = \left(\mathbf{I} + \tfrac{1}{3} h^2 D^2\right) - h D \\ &R[h] = -h C \\ &2R[h] + Q[h]F[h] - Q[-h]F[-h] = \tfrac{2}{45} h^5 D^4 (C + D F[0]) + O h^7 \end{aligned} \tag{11}$$

Padé order 3, constant D, C:

$$\begin{aligned} &Q[h] = \left(\mathbf{I} + \tfrac{2}{5} h^2 D^2\right) - \left(\mathbf{I} + \tfrac{1}{15} h^2 D^2\right) h D \\ &R[h] = -\left(\mathbf{I} + \tfrac{1}{15} h^2 D^2\right) h C \\ &2R[h] + Q[h]F[h] - Q[-h]F[-h] = -\tfrac{2}{1575} h^7 D^6 (C + D F[0]) + O h^9 \end{aligned} \tag{12}$$

Padé order 4, constant D, C:

$$
\begin{aligned}
&Q[h] = \left(\mathbf{I} + \tfrac{3}{7} h^2\, D^2 + \tfrac{1}{105} h^4\, D^4\right) - \left(\mathbf{I} + \tfrac{2}{21} h^2\, D^2\right) h\, D \\
&R[h] = -\left(\mathbf{I} + \tfrac{2}{21} h^2\, D^2\right) h\, C \\
&2\, R[h] + Q[h]\, F[h] - Q[-h]\, F[-h] = \tfrac{2}{99225} h^9\, D^8\, (C + D\, F[0]) + O\, h^{11}
\end{aligned}
\tag{13}
$$

Eq's. (10)-(13) are specializations of the following general formula, in which the n subscript is applied to the Q and R matrices to identify the Padé order:

Padé order n, constant D, C:

$$
\begin{aligned}
&Q_n[h] = \sum_{j=0}^{n} \frac{(2n-j)!\, n!}{j!\, (2n)!\, (n-j)!} (-2\, h\, D)^j \\
&R_n[h] = \sum_{1 \le j \le n,\ j \text{ odd}} \frac{(2n-j)!\, n!}{j!\, (2n)!\, (n-j)!} (-2\, h\, D)^{j-1}\, (-2\, h\, C) \\
&residual_n[h] = \frac{(-1)^n\, (n!)^2\, (2\, h)^{2n+1}}{(2n)!\, (2n+1)!} D^{2n}\, (C + D\, F[0]) \\
&2\, R_n[h] + Q_n[h]\, F[h] - Q_n[-h]\, F[-h] = residual_n[h] + O\, h^{2n+3}
\end{aligned}
\tag{14}
$$

The subscripted functions in Eq's. (14) can be efficiently calculated by using the following recursion relations,

$$
\begin{aligned}
&Q_0[h] = \mathbf{I}, \\
&Q_1[h] = \mathbf{I} - h\, D, \\
&Q_{n+1}[h] = Q_n[h] + \frac{h^2\, D^2}{(2n+1)(2n-1)} Q_{n-1}[h]
\end{aligned}
\tag{15}
$$

$$
\begin{aligned}
&R_0[h] = \mathbf{0}, \\
&R_1[h] = -h\, C, \\
&R_{n+1}[h] = R_n[h] + \frac{h^2\, D^2}{(2n+1)(2n-1)} R_{n-1}[h]
\end{aligned}
\tag{16}
$$

$$
\begin{aligned}
&residual_0[h] = 2\, h\, (C + D\, F[0]) \\
&residual_{n+1}[h] = \frac{-h^2\, D^2}{(2n+1)(2n+3)} residual_n[h]
\end{aligned}
\tag{17}
$$

For non-constant D and C, general formulas such as Eq's. (14) have not been developed, but several special cases are listed below.

Padé order 1, non-constant D, C :

$$\begin{aligned} &Q[h] = \mathbf{I} - h\, D[0] \\ &R[h] = -h\, C[0] \\ &2\, R[h] + Q[h]\, F[h] - Q[-h]\, F[-h] = O\, h^3 \end{aligned} \tag{18}$$

Padé order 2, non-constant D, C :

$$\begin{aligned} &Q[h] = \mathbf{I} - h\,(-\tfrac{1}{6} D[-h] + \tfrac{2}{3} D[0] + \tfrac{1}{2} D[h]) + \tfrac{1}{3} h^2\, D[h]^2 \\ &R[h] = -h\,(-\tfrac{1}{6} C[-h] + \tfrac{2}{3} C[0] + \tfrac{1}{2} C[h]) + \tfrac{1}{3} h^2\, D[h]\, C[h] \\ &R[h] - R[-h] + Q[h]\, F[h] - Q[-h]\, F[-h] = O\, h^5 \end{aligned} \tag{19}$$

Padé order 3, non-constant D, C :

$$\begin{aligned} &Q[h] = \mathbf{I} - h\,(\tfrac{2}{45} D[-\tfrac{1}{2} h] + \tfrac{2}{15} D[0] + \tfrac{2}{3} D[\tfrac{1}{2} h] + \tfrac{7}{45} D[h]) + \\ &\quad (\tfrac{1}{15} D[-\tfrac{1}{2} h] + \tfrac{1}{5} D[0] + \tfrac{11}{15} D[\tfrac{1}{2} h]) \\ &\quad (\tfrac{2}{5} h^2\,(\tfrac{1}{9} D[-\tfrac{1}{2} h] - \tfrac{1}{2} D[0] + D[\tfrac{1}{2} h] + \tfrac{7}{18} D[h]) - \tfrac{1}{15} h^3\, D[h]^2) \\ &R[h] = -h\,(\tfrac{2}{45} C[-\tfrac{1}{2} h] + \tfrac{2}{15} C[0] + \tfrac{2}{3} C[\tfrac{1}{2} h] + \tfrac{7}{45} C[h]) + \\ &\quad (\tfrac{1}{15} D[-\tfrac{1}{2} h] + \tfrac{1}{5} D[0] + \tfrac{11}{15} D[\tfrac{1}{2} h]) \\ &\quad (\tfrac{2}{5} h^2\,(\tfrac{1}{9} C[-\tfrac{1}{2} h] - \tfrac{1}{2} C[0] + C[\tfrac{1}{2} h] + \tfrac{7}{18} C[h]) - \tfrac{1}{15} h^3\, D[h]\, C[h]) \\ &R[h] - R[-h] + Q[h]\, F[h] - Q[-h]\, F[-h] = O\, h^7 \end{aligned} \tag{20}$$

Padé order 4, non-constant D, C:

$$
\begin{aligned}
L_1[h,X] &= \tfrac{403}{16800}X[-h] - \tfrac{279}{2800}X[-\tfrac{2}{3}h] + \tfrac{99}{800}X[-\tfrac{1}{3}h] \\
&\quad + \tfrac{34}{105}X[0] - \tfrac{333}{5600}X[\tfrac{1}{3}h] + \tfrac{1719}{2800}X[\tfrac{2}{3}h] + \tfrac{1237}{16800}X[h] \\
L_2[h,X] &= \tfrac{57}{1120}X[-h] - \tfrac{243}{560}X[-\tfrac{2}{3}h] + \tfrac{1269}{1120}X[-\tfrac{1}{3}h] - \tfrac{3}{4}X[0] \\
&\quad + \tfrac{891}{1120}X[\tfrac{1}{3}h] + \tfrac{27}{112}X[\tfrac{2}{3}h] - \tfrac{41}{1120}X[h] \\
L_3[h,X] &= -\tfrac{2067}{9680}X[-h] + \tfrac{6021}{4840}X[-\tfrac{2}{3}h] - \tfrac{5805}{1936}X[-\tfrac{1}{3}h] + \tfrac{1863}{484}X[0] \\
&\quad - \tfrac{5697}{1936}X[\tfrac{1}{3}h] + \tfrac{10341}{4840}X[\tfrac{2}{3}h] - \tfrac{727}{9680}X[h] \\
L_4[h,X] &= \tfrac{63}{16}X[-h] - \tfrac{1809}{40}X[-\tfrac{2}{3}h] + \tfrac{2295}{16}X[-\tfrac{1}{3}h] - \tfrac{801}{4}X[0] \\
&\quad + \tfrac{2133}{16}X[\tfrac{1}{3}h] - \tfrac{297}{8}X[\tfrac{2}{3}h] + \tfrac{233}{80}X[h] \\
L_5[h,X] &= \tfrac{123}{160}X[-h] - \tfrac{135}{8}X[-\tfrac{2}{3}h] + \tfrac{2295}{32}X[-\tfrac{1}{3}h] - 132\,X[0] \\
&\quad + \tfrac{3861}{32}X[\tfrac{1}{3}h] - \tfrac{1917}{40}X[\tfrac{2}{3}h] + \tfrac{149}{32}X[h] \\
L_6[h,X] &= -\tfrac{6}{35}X[-h] + \tfrac{27}{10}X[-\tfrac{2}{3}h] - \tfrac{1053}{112}X[-\tfrac{1}{3}h] + \tfrac{57}{4}X[0] \\
&\quad - \tfrac{621}{56}X[\tfrac{1}{3}h] + \tfrac{729}{140}X[\tfrac{2}{3}h] - \tfrac{277}{560}X[h] \\
Q[h] &= \mathbf{I} - h\,L_1[h,D] + L_2[h,D]\left(\tfrac{121}{315}h^2\,L_3[h,D] - \tfrac{2}{315}h^3\,L_4[h,D]\,L_5[h,D]\right) \\
&\quad + \left(\tfrac{2}{45}h^2\,L_6[h,D] + L_2[h,D]\left(-\tfrac{4}{45}h^3\,L_6[h,D] + \tfrac{1}{105}h^4\,D[h]^2\right)\right)D[h] \\
R[h] &= -h\,L_1[h,C] + L_2[h,D]\left(\tfrac{121}{315}h^2\,L_3[h,C] - \tfrac{2}{315}h^3\,L_4[h,D]\,L_5[h,C]\right) \\
&\quad + \left(\tfrac{2}{45}h^2\,L_6[h,D] + L_2[h,D]\left(-\tfrac{4}{45}h^3\,L_6[h,D] + \tfrac{1}{105}h^4\,D[h]^2\right)\right)C[h] \\
R[h] &- R[-h] + Q[h]\,F[h] - Q[-h]\,F[-h] = O\,h^9
\end{aligned}
\tag{21}
$$

Note the commonality of subexpressions in the $Q[h]$ and $R[h]$ formulas in Eq's. (20) and (21). Also, the product factors $D[h]^2$ in Eq's. (19)-(21) can be re-used in the subsequent integration step as $D[-h]^2$ for calculating $Q[-h]$ and $R[-h]$. The products $D[h]C[h]$ in Eq's. (19) and (20) can similarly be carried over to the next step.

3. Scale-and-square algorithm

For the constant-coefficient case, Eq. (8) can be formulated as

$$F[x_0 + \Delta x] = \Omega + \Phi\,F[x_0] \tag{22}$$

where

$$\Omega = D^{-1}\left(\exp[D\,\Delta x] - \mathbf{I}\right)C\,, \tag{23}$$

$$\Phi = \exp[D\,\Delta x]\,. \tag{24}$$

The Ω and Φ matrices, which depend on Δx but not on x_0, can be obtained from the Padé approximation, Eq. (4), for small Δx,

$$F[h] \approx Q[h]^{-1}\left(Q[-h]\,F[-h] - 2\,R[h]\right). \tag{25}$$

(The term $R[h]-R[-h]$ in Eq. (4) has been replaced by $2\,R[h]$ because $R[h]$ is an odd function of h for the constant-coefficient case.) Eq. (25) is applied with $\Delta x = 2\,h$ and with the x coordinate origin shifted so that $x_0 = -h$; thus $F[x_0 + \Delta x]$ is determined from $F[x_0]$. The following approximations result from Eq's. (22) and (25),

$$\Omega \approx -2\,Q[h]^{-1}\,R[h], \tag{26}$$

$$\Phi \approx Q[h]^{-1}\,Q[-h]. \tag{27}$$

Eq. (22) is applied recursively to integrate $F[x]$ over a large, m-step integration interval,

$$F[x_0 + m\,\Delta x] = \Omega_m + \Phi^m\,F[x_0] \tag{28}$$

where

$$\Omega_m = (\mathbf{I} + \Phi + \Phi^2 + \ldots + \Phi^{m-1})\,\Omega \quad (\Omega_1 = \Omega). \tag{29}$$

Given Ω_m and Φ^m for any particular integer m, Ω_{2m} and Φ^{2m} are obtained as

$$\Omega_{2m} = \Omega_m + \Phi^m\,\Omega_m, \tag{30}$$

$$\Phi^{2m} = (\Phi^m)^2. \tag{31}$$

Eq. (31) is the basis of the standard scale-and-square algorithm for homogeneous linear differential equations, and Eq. (30) generalizes the method for nonhomogeneous equations.

In implementing the Padé approximation it is advantageous to calculate the even and odd parts of $Q[h]$ separately so that $Q[h]$ and $Q[-h]$ can be calculated with minimal computational overhead,

$$\begin{aligned} &Q[\pm h] = Q^{[\text{even}]}[h] \pm Q^{[\text{odd}]}[h], \\ &Q^{[\text{even}]}[h] = \tfrac{1}{2}(Q[h] + Q[-h]), \quad Q^{[\text{odd}]}[h] = \tfrac{1}{2}(Q[h] - Q[-h]). \end{aligned} \tag{32}$$

Note that the $Q[h]$ definitions in Eq's. (10)-(13) are formatted with the even and odd polynomials separated. Also note that the $R[h]$ function is similar to $Q^{[\text{odd}]}[h]$ except for replacement of the last D factor by C. These matrices have a common matrix left-factor L,

$$R[h] = L\,C, \quad Q^{[\text{odd}]}[h] = L\,D. \tag{33}$$

Eq. (33) is applied in Eq's. (26) and (27) to obtain

$$\Omega = -2\,Q[h]^{-1}\,L\,C. \tag{34}$$

$$\Phi - \mathbf{I} = -2\,Q[h]^{-1}\,Q^{[\text{odd}]}[h] = -2\,Q[h]^{-1}\,L\,D. \tag{35}$$

For small Δx the matrix Ω is approximately proportional to Δx (Eq. (23)), but Φ is approximately equal to $\mathbf{I}$ with a small Δx-proportionate increment (Eq. (24)). To avoid possible precision loss in the Φ diagonal elements, Φ can be calculated with the dominant $\mathbf{I}$ component subtracted off. The $\mathbf{I}$ separation is preserved through the scale-and-square process by modifying Eq's. (30) and (31) as follows,

$$\Omega_{2m} = 2\,\Omega_m + (\Phi^m - \mathbf{I})\,\Omega_m\,, \tag{36}$$

$$(\Phi^{2m} - \mathbf{I}) = (\Phi^m - \mathbf{I})^2 + 2\,(\Phi^m - \mathbf{I})\,. \tag{37}$$

4. Error analysis and tolerance control

Continuing with the constant-coefficient case, Eq. (22) will be slightly in error due to the inaccuracy of the Padé approximation. Denoting error terms (approximation minus exact value) by the prefix "δ", the calculated error in $F[x_0 + \Delta x]$ is

$$\delta F[x_0 + \Delta x] = \delta\Omega + (\delta\Phi)\,F[x_0]\,. \tag{38}$$

The errors in Ω and Φ can be obtained from Eq's. (14), in which $F[h]$ is calculated by ignoring the residual term ($residual_n[h]$),

$$Q_n[h]\,\delta F[h] \approx -residual_n[h]\,. \tag{39}$$

For small h, $Q_n[h]$ is close to $\mathbf{I}$ (i.e., $Q_n[h] = \mathbf{I} + O\,h$) and Eq. (39) simplifies to

$$\delta F[h] \approx -residual_n[h]\,. \tag{40}$$

A comparison of Eq's. (38) and (40), with $x_0 = -h$, $\Delta x = 2\,h$, and with $residual_n[h]$ defined in Eq's. (14), yields the following expressions for $\delta\Gamma$ and $\delta\Phi$ (from Eq. (14)).

$$\delta\Omega = -\frac{(-1)^n\,(n!)^2\,(\Delta x)^{2n+1}}{(2\,n)!(2\,n+1)!}\,D^{2n}\,C \quad (\Delta x = 2\,h)\,, \tag{41}$$

$$\delta\Phi = -\frac{(-1)^n\,(n!)^2\,(\Delta x)^{2n+1}}{(2\,n)!(2\,n+1)!}\,D^{2n+1}\,. \tag{42}$$

(The $F[0]$ term in Eq. (14) does not differ significantly from $F[-h]$ for the purpose of error estimation.)

The cumulative errors in m integration steps (Eq. (28)) are represented as

$$\delta F[x_0 + m\,\Delta x] = \delta\Omega_m + (\delta(\Phi^m))\,F[x_0]\,. \tag{43}$$

The $\delta(\Phi^m)$ error has the approximate form

$$\delta(\Phi^m) \approx (\delta\Phi)\,\Phi^{m-1} + \Phi\,(\delta\Phi)\,\Phi^{m-2} + \ldots + \Phi^{m-1}\,(\delta\Phi)\,. \tag{44}$$

Φ is close to $\mathbf{I}$ ($\Phi = \mathbf{I} + O\,\Delta x$, Eq. (24)), so Eq. (44) simplifies to

$$\delta(\Phi^m) \approx m\,\delta\Phi\,. \tag{45}$$

A similar relation is applied to the Φ powers on the right side of Eq. (29),

$$\delta\Omega_m \approx (\delta\Phi + 2\,\delta\Phi + \ldots + (m-1)\,\delta\Phi)\,\Omega + (\mathbf{I} + \Phi + \Phi^2 + \ldots + \Phi^{m-1})\,\delta\Omega\,. \tag{46}$$

Making the approximations $\Phi \approx \mathbf{I}$ and $\Omega \approx C\,\Delta x$ (from Eq. (23)), Eq. (46) simplifies to

$$\delta\Omega_m \approx \tfrac{1}{2} m(m-1)(\delta\Phi) C \Delta x + m\,\delta\Omega . \tag{47}$$

The factors $\delta\Phi$ and $\delta\Omega$ are both of order $(\Delta x)^{2n+1}$ (cf. Eq's. (41) and (42)). The first term on the right side of Eq. (47) contains an extra factor of Δx relative to the second term and can hence be neglected,

$$\delta\Gamma_m \approx m\,\delta\Gamma . \tag{48}$$

Eq's. (45) and (48) are substituted in Eq. (43),

$$\delta F[x_0 + m\Delta x] \approx m(\delta\Omega + (\delta\Phi) F[x_0]) . \tag{49}$$

(To a first-order approximation the single-step error $\delta F[x_0 + \Delta x]$ is simply multiplied by m in taking m integration steps.) Eq. (49) includes two error factors: an additive factor $m\,\delta\Omega$, and a multiplicative factor $m\,\delta\Phi$ that is applied to $F[x_0]$. The $m\,\delta\Phi$ factor is dimensionless (cf. Eq. (42)) whereas $m\,\delta\Omega$ has a linear dependence on C and has the same dimensional units as $C\Delta x$ (Eq. (41)). Δx can be chosen to impose an approximate tolerance bound on both error terms,

$$m\|\delta\Omega\| \le \|C\| x_{\text{range}}\, tol, \quad m\|\delta\Phi\| \le tol \tag{50}$$

where $\|\ldots\|$ is the Frobenius norm, tol is a specified dimensionless tolerance bound, and x_{range} is the total integration range,

$$x_{\text{range}} = m|\Delta x| . \tag{51}$$

(tol is a "relative tolerance", which scales the $(\delta\Phi) F[x_0]$ error in proportion to $F[x_0]$, and the $\delta\Omega$ error in proportion to C.) With substitution from Eq's. (41) and (42), the following conditions are obtained from Eq's. (50),

$$m\frac{(n!)^2 (x_{\text{range}}/m)^{2n+1}}{(2n)!(2n+1)!}\left\|D^{2n} C\right\| \le \|C\| x_{\text{range}}\, tol, \quad m\frac{(n!)^2 (x_{\text{range}}/m)^{2n+1}}{(2n)!(2n+1)!}\|D^{2n+1}\| \le tol . \tag{52}$$

The first of Eq's. (52) can be strengthened by omitting the C factor because $\left\|D^{2n} C\right\| \le \|D^{2n}\| \cdot \|C\|$. The following limit on m implies Eq's. (52),

$$m \ge \left(\frac{(n!)^2}{(2n)!(2n+1)!\,tol} \max\left[(x_{\text{range}})^{2n} \|D^{2n}\|, (x_{\text{range}})^{2n+1} \|D^{2n+1}\| \right] \right)^{1/(2n)} . \tag{53}$$

With the scale-and-square algorithm, m is a power of 2 and Eq. (53) translates to

$$m = 2^j, \quad j \ge \frac{1}{2n} \log_2 \left[\frac{(n!)^2}{(2n)!(2n+1)!\,tol} \max\left[(x_{\text{range}})^{2n} \|D^{2n}\|, (x_{\text{range}})^{2n+1} \|D^{2n+1}\| \right] \right]. \tag{54}$$

The first max argument is only applicable if C is non-zero; if C is zero the max[…] factor can be replaced by $(x_{range})^{2n+1} \|D^{2n+1}\|$. (If D is zero, Eq. (1) has a trivial solution for constant C and no error analysis is required.)

The minimum j satisfying Eq. (54) defines the integration step $|\Delta x| = 2h = 2^{-j} x_{\text{range}}$. In some cases the step may not be small enough to justify the differential approximations made in the above derivation (e.g. the assumptions that $Q_n[h]$ and Φ are close to $\mathbf{I}$). But nevertheless, Eq. (54) typically results in good computational accuracy.

The above formulas are not directly applicable to the non-constant-coefficient case, but Eq. (53) can be used to obtain an initial integration step size $|\Delta x| = x_{range} / m$, using values of $D[x]$ and $C[x]$ at the beginning of the integration interval. This initialization is inapplicable when D is zero or when D or C varies significantly over the Δx range. (When D is identically zero, the errors in Eq's. (18)-(21) are proportional to the order-$2n$ derivative of C for Padé order n.) An alternative step initialization criterion may need to be used to accommodate spatial variability of D and C.

After Δx is initialized, it is dynamically varied at each integration step to limit the error, which can be estimated by determining $F[x_0 + \Delta x]$ from $F[x_0]$ by two estimation methods and applying Richardson extrapolation to the estimates. A first estimate $F_1[x_0 + \Delta x]$ is obtained by making a single-step Padé approximation with step size Δx, and a second estimate $F_2[x_0 + \Delta x]$ is obtained by making two Padé approximation steps with step size $\frac{1}{2}\Delta x$. The errors in these estimates are approximately

$$\delta F_1[x_0 + \Delta x] \approx A \Delta x^{2n+1}, \quad \delta F_2[x_0 + \Delta x] \approx 2 A (\tfrac{1}{2}\Delta x)^{2n+1} \tag{55}$$

where n is the Padé order, A is an undetermined matrix, and the factor of 2 is included in the second equality to account for the two steps. The following relation is obtained by eliminating A between Eq's. (55),

$$\delta F_1[x_0 + \Delta x] \approx 2^{2n}\, \delta F_2[x_0 + \Delta x]. \tag{56}$$

Subtracting the error from both estimates should give the same error-corrected result,

$$F_1[x_0 + \Delta x] - \delta F_1[x_0 + \Delta x] = F_2[x_0 + \Delta x] - \delta F_2[x_0 + \Delta x]. \tag{57}$$

$\delta F_1[x_0 + \Delta x]$ is eliminated from Eq's. (56) and (57) to obtain

$$\delta F_2[x_0 + \Delta x] = \frac{F_1[x_0 + \Delta x] - F_2[x_0 + \Delta x]}{2^{2n} - 1}. \tag{58}$$

The integration step Δx is decreased or increased by factors of 2 to keep this estimated error within allowed tolerance bounds (i.e. the step is halved if the error significantly exceeds the tolerance, and is doubled if the error times 2^{2n+1} is within the tolerance). Some excursion of the estimated error over the tolerance limit can be allowed because the calculated $F_2[x_0 + \Delta x]$ can be decremented by the error estimate $\delta F_2[x_0 + \Delta x]$ to improve its accuracy.

The $F[h]$ value calculated from Eq. (4) can be represented as in Eq's. (22), (26) and (27),

$$F[h] = \Omega + \Phi\, F[-h], \tag{59}$$

where

$$\Omega = -Q[h]^{-1}\,(R[h] - R[-h])\,, \tag{60}$$

$$\Phi = Q[h]^{-1}\,Q[-h]\,. \tag{61}$$

(In this context $R[h]$ is not generally an odd function of h.) With $x_0 = -h$ and $\Delta x = 2h$, the $F_1[x_0 + \Delta x]$ term in Eq. (58) has the form

$$F_1[x_0 + \Delta x] = \Omega_1 + \Phi_1\,F[x_0]\,. \tag{62}$$

The same separation is made for $F_2[x_0 + \Delta x]$ in two steps,

$$\begin{aligned} &F_2[x_0 + \tfrac{1}{2}\Delta x] = \Omega_{2,1} + \Phi_{2,1}\,F[x_0], \\ &F_2[x_0 + \Delta x] = \Omega_{2,2} + \Phi_{2,2}\,F[x_0 + \tfrac{1}{2}\Delta x] = \Omega_{2,2} + \Phi_{2,2}\,(\Omega_{2,1} + \Phi_{2,1}\,F[x_0]). \end{aligned} \tag{63}$$

This expression is of the form

$$F_2[x_0 + \Delta x] = \Omega_2 + \Phi_2\,F[x_0] \quad \text{with} \quad \Omega_2 = \Omega_{2,2} + \Phi_{2,2}\,\Omega_{2,1},\ \ \Phi_2 = \Phi_{2,2}\,\Phi_{2,1}. \tag{64}$$

The estimated error $\delta F_2[x_0 + \Delta x]$ in Eq. (58) is correspondingly separated into Ω and Φ components,

$$\delta F_2[x_0 + \Delta x] = \delta\Omega_2 + (\delta\Phi_2)\,F[x_0] \quad \text{with} \quad \delta\Omega_2 = \frac{\Omega_1 - \Omega_2}{2^{2n} - 1},\ \ \delta\Phi_2 = \frac{\Phi_1 - \Phi_2}{2^{2n} - 1}. \tag{65}$$

Eq. (65) is similar to Eq. (38). The following tolerance specifications are analogous to Eq's. (50) (with m defined by Eq. (51)),

$$\frac{x_{\text{range}}}{|\Delta x|}\left\|\delta\Omega_2\right\| \le \|C\|_{\text{rms}}\,x_{\text{range}}\,tol, \quad \frac{x_{\text{range}}}{|\Delta x|}\left\|\delta\Phi_2\right\| \le tol\,. \tag{66}$$

"$\|C\|_{\text{rms}}$" represents the root-mean-square of $\|C[x]\|$ over the $C[x]$ matrices appearing in any of Eq's. (18)-(21). (If $\|C\|_{\text{rms}}$ is zero the first limit in Eq. (66) is inapplicable.) These conditions can be used to control the integration step size Δx for the non-constant coefficient case.

References

[1] K. Johnson, *Numerical Solution of Linear, Homogeneous Differential Equation Systems via Padé Approximation* (v2, posted April 22, 2016). http://vixra.org/abs/1509.0286.

[2] K. Johnson, Linear differential equation solver (lde.m), posted Nov. 30, 2016. http://www.mathworks.com/matlabcentral/fileexchange/60475-linear-differential-equation-solver--lde-m-.

[3] N. J. Higham, *The Scaling and Squaring Method for the Matrix Exponential Revisited*, SIAM Review, 51 (2009), pp. 747–764.

[4] A. H. Al-Mohy and N. J. Higham, *A new scaling and squaring algorithm for the matrix exponential*, SIAM J. Matrix Anal. Appl., 30 (2009), pp. 970–989.

[5] MATLAB **expm** function, https://www.mathworks.com/help/matlab/ref/expm.html.

[6] MATLAB Ordinary Differential Equation solvers, https://www.mathworks.com/help/matlab/math/choose-an-ode-solver.html.

Appendix A: MATLAB Implementation notes, constant-coefficient case

The even part of the $Q[h]$ polynomial ($Q^{[\text{even}]}[h]$, Eq's. (14), (32)) is of the form

$$Y = \sum_{j=1}^{N} c_j X^{j-1}; \;\; X = (h D)^2 . \tag{67}$$

(X is a square matrix; the coefficients c_j are scalar.) The odd part ($Q^{[\text{odd}]}[h]$) has a similar form, but with an extra factor of $h D$. (Omitting the D factor yields the L matrix in Eq's. (33), from which $R[h]$ is obtained.) The polynomial coefficients c_j can be initialized by using the recursion relations in Eq's. (15).

For large N the polynomial can be efficiently evaluated by zero-padding and reshaping the coefficient vector to a rectangular array and reorganizing Eq. (67) as

$$Y = \sum_{k=1}^{N_2} \left(\sum_{j=1}^{N_1} c_{j,k} X^{j-1} \right) (X^{N_1})^{k-1}; \;\; N_1 = \text{round}[\sqrt{N}], \;\; N_2 = \text{ceil}[N / N_1] \; . \tag{68}$$

A direct implementation of Eq. (67) would require $N-2$ matrix multiplies, whereas Eq. (68) typically requires $N_1 + N_2 - 2$, a reduction by a factor of approximately $\frac{1}{2}\sqrt{N}$. The powers X^2 … X^{N_1-1} in the j sum, and the outer factor X^{N_1} in the k sum, are pre-computed, and the outer sum is implemented using Horner's method. The number of multiplies is reduced by 1 (to $N_1 + N_2 - 3$) in two special cases: if $N_2 = 1$ (in which case X^{N_1} is not needed), or if $N_2 > 1$ and $c_{j,N_2} = 0$ for $j > 1$ (in which the first step of Horner's method multiplies X^{N_1} by c_{1,N_2}, a scalar).

For example, an order-12 polynomial can be implemented with 5 matrix multiplies as follows,

$$\begin{aligned} c_1 + c_2 X + \ldots + c_{13} X^{12} = {} & c_1 + c_2 X + c_3 X^2 + c_4 X^3 + \\ & \left(c_5 + c_6 X + c_7 X^2 + c_8 X^3 + \left(c_9 + c_{10} X + c_{11} X^2 + c_{12} X^3 + c_{13} X^4 \right) X^4 \right) X^4 . \end{aligned} \tag{69}$$

(Three matrix-matrix multiplies are required for X^2, X^3, and X^4; and two are required for the $(\ldots) X^4$ products.)

Eq. (54) contains additional matrix powers D^{2n} and D^{2n+1}, but calculation of these powers can be avoided by using the following bounding estimate for the norm of a matrix power,

$$\| D^{j_1 + j_2 + \ldots} \| \le \| D^{j_1} \| \cdot \| D^{j_2} \| \cdot \ldots \tag{70}$$

The pre-computed powers of D used for the Q polynomial evaluation are used on the right side of Eq. (70), and products of this form are substituted in Eq. (54). The number j of squaring operations may be increased by this substitution, but j is proportional to the logarithm of the matrix norm so the additional squaring steps are typically not significant compared to the

runtime penalty of computing D^{2n} and D^{2n+1}. Furthermore, direct computation of $\|D^{2n}\|$ and $\|D^{2n+1}\|$ could potentially overflow the floating-point range limit (2^{1024}) when D and n are large. This problem can be circumvented by separating a $\log_2\left[\|D^{2n}\|\right]$ term out of Eq. (54) and replacing it with the following bounding estimate,

$$\log_2\left[\|D^{j_1+j_2+\cdots}\|\right] \le \log_2\left[\|D^{j_1}\|\right] + \log_2\left[\|D^{j_2}\|\right] + \ldots \quad (j_1 + j_2 + \ldots = 2n) \tag{71}$$

Over-estimation of $\|D^N\|$ can potentially lead to numeric precision loss due to overscaling [4], but the $\mathbf{I}$-separation method (Eq's. (35)-(37)) avoids this problem. The following MATLAB test case illustrates the benefit of $\mathbf{I}$ separation:

```
a = -1e20;
b = eps;
c = 1;
A = [a,0,b;0,c,0;-b,0,a];
% Exact matrix exponential:
expA = exp(a)*( ...
    [1,0,0;0,0,0;0,0,1]*cos(b)+ ...
    [0,0,1;0,0,0;-1,0,0]*sin(b))+ ...
    [0,0,0;0,exp(c),0;0,0,0];
disp(num2str(expA))
0           0           0
0      2.7183           0
0           0           0
```

(72)

The standard MATLAB **expm** function completely loses numeric precision on this example:

```
disp(num2str(expm(A)))
0  0  0
0  1  0
0  0  0
```

(73)

However, a simple code modification implementing the squaring algorithm as in Eq. (37) reduces the error to machine precision ($4\cdot10^{-16}$).

Appendix B: Mathematica verification of Eq's. (10)-(13) and (18)-(21)

The calculations underlying Eq's. (10)-(13) and (18)-(21) require non-commutative symbolic algebra. The following results are obtained using the NCAlgebra package for Mathematica, from the University of California, San Diego (http://math.ucsd.edu/~ncalg/). The Mathematica code loads the NCAlgebra package, adds some functionality, and verifies the equations. A Mathematica notebook containing the following code is posted at https://figshare.com/articles/Appendix_2016_11_29_nb/4269584.

```
(* Load NCAlgebra package (http://math.ucsd.edu/~ncalg/) *)
<< NC`
<< NCAlgebra`

(* Make all variables commutative by default.
  (Override the default noncommutativity of single-letter lowercase variables.) *)
Remove[a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z]

(* D0, C0, Dfn, Cfn, F, Q, and R represent matrices. D0 and C0 represent constants;
Dfn, Cfn, F, Q, and R represent functions, and "1" represents the identity matrix. *)
SetNonCommutative[D0, C0, Dfn, Cfn, F, Q, R];

(* Series and O (e.g. O[h]^n) do not work with NC types
 (e.g.: try Dfn[h]**F[h]+O[h]^2 or Series[Dfn[h]**F[h],{h,0,1}]). Define a variant that does work. *)
NCSeries[f_, {x_, x0_, n_}] := NCExpand[Sum[(D[f, {x, j}] / j! /. x → x0) (x - x0) ^ j, {j, 0, n}]] + O[x - x0] ^ (n + 1);

(* substD is a substitution rule for reducing derivatives of F using the relation F'[h]==Dfn[h]**F[h]+Cfn[h].
    Use "... //. substD" to eliminate all F derivatives.
    (The substD definition uses ":>",
     not "->" otherwise the substitutions will not work when x or n has a preassigned value.) *)
substD = Derivative[n_][F][x_] :> Derivative[n - 1][Dfn[#] ** F[#] + Cfn[#] &][x];

(* substD0 is a substitution rule for reducing derivatives of F using the relation F'[h]==
 D0**F[h]+C0. This specializes substD for the case where Dfn and Cfn are constant. *)
substD0 = Derivative[n_][F][x_] :> Derivative[n - 1][D0 ** F[#] + C0 &][x];


(* Eq 10 *)
Q[h_] := 1 - h D0;
R[h_] := -h C0;
Factor[NCExpand[Normal[NCSeries[2 R[h] + Q[h] ** F[h] - Q[-h] ** F[-h], {h, 0, 4}]] //. substD0]]
```

$$-\frac{2}{3} h^3 \,(D0 ** D0 ** C0 + D0 ** D0 ** D0 ** F[0])$$

```
(* Eq 11 *)
Q[h_] := (1 + 1/3 h^2 D0 ** D0) - h D0;
R[h_] := -h C0;
Factor[NCExpand[Normal[NCSeries[2 R[h] + Q[h] ** F[h] - Q[-h] ** F[-h], {h, 0, 6}]] //. substD0]]
```

$$\frac{2}{45} h^5 \,(D0 ** D0 ** D0 ** D0 ** C0 + D0 ** D0 ** D0 ** D0 ** D0 ** F[0])$$

```
(* Eq 12 *)
Q[h_] := (1 + 2/5 h^2 D0 ** D0) - (1 + 1/15 h^2 D0 ** D0) ** (h D0);
R[h_] := -(1 + 1/15 h^2 D0 ** D0) ** (h C0);
Factor[NCExpand[Normal[NCSeries[2 R[h] + Q[h] ** F[h] - Q[-h] ** F[-h], {h, 0, 8}]] //. substD0]]
```

$$-\frac{2\, h^7 \,(D0 ** D0 ** D0 ** D0 ** D0 ** D0 ** C0 + D0 ** D0 ** D0 ** D0 ** D0 ** D0 ** D0 ** F[0])}{1575}$$

```
(* Eq 13 *)
Q[h_] := (1 + 3/7 h^2 D0 ** D0 + 1/105 h^4 D0 ** D0 ** D0 ** D0) - (1 + 2/21 h^2 D0 ** D0) ** (h D0);
R[h_] := -(1 + 2/21 h^2 D0 ** D0) ** (h C0);
Factor[NCExpand[Normal[NCSeries[2 R[h] + Q[h] ** F[h] - Q[-h] ** F[-h], {h, 0, 10}]] //. substD0]]
```

$$\frac{1}{99225} 2\, h^9 \,(D0 ** D0 ** D0 ** D0 ** D0 ** D0 ** D0 ** D0 ** C0 + D0 ** D0 ** D0 ** D0 ** D0 ** D0 ** D0 ** D0 ** D0 ** F[0])$$

```
(* Eq 18 *)
Q[h_] := 1 - h Dfn[0];
R[h_] := -h Cfn[0];
NCExpand[Normal[NCSeries[2 R[h] + Q[h] ** F[h] - Q[-h] ** F[-h], {h, 0, 2}]] //. substD]
```

```
0
```

```
(* Eq 19 *)
Q[h_] := 1 - h (-1/6 Dfn[-h] + 2/3 Dfn[0] + 1/2 Dfn[h]) + 1/3 h^2 Dfn[h] ** Dfn[h];
R[h_] := -h (-1/6 Cfn[-h] + 2/3 Cfn[0] + 1/2 Cfn[h]) + 1/3 h^2 Dfn[h] ** Cfn[h];
NCExpand[Normal[NCSeries[R[h] - R[-h] + Q[h] ** F[h] - Q[-h] ** F[-h], {h, 0, 4}]] //. substD]
```

```
0
```

```
(* Eq 20 *)
Q[h_] := 1 - h (2/45 Dfn[-h/2] + 2/15 Dfn[0] + 2/3 Dfn[h/2] + 7/45 Dfn[h]) +
    (1/15 Dfn[-h/2] + 1/5 Dfn[0] + 11/15 Dfn[h/2]) **
     (2/5 h^2 (1/9 Dfn[-h/2] - 1/2 Dfn[0] + Dfn[h/2] + 7/18 Dfn[h]) - 1/15 h^3 Dfn[h] ** Dfn[h]);
R[h_] := -h (2/45 Cfn[-h/2] + 2/15 Cfn[0] + 2/3 Cfn[h/2] + 7/45 Cfn[h]) +
    (1/15 Dfn[-h/2] + 1/5 Dfn[0] + 11/15 Dfn[h/2]) **
     (2/5 h^2 (1/9 Cfn[-h/2] - 1/2 Cfn[0] + Cfn[h/2] + 7/18 Cfn[h]) - 1/15 h^3 Dfn[h] ** Cfn[h]);
NCExpand[Normal[NCSeries[R[h] - R[-h] + Q[h] ** F[h] - Q[-h] ** F[-h], {h, 0, 6}]] //. substD]
```

```
0
```

```
(* Eq 21 *)
L1[h_, X_] := 403/16800 X[-h] - 279/2800 X[-2 h/3] + 99/800 X[-h/3] + 34/105 X[0] - 333/5600 X[h/3] + 1719/2800 X[2 h/3] + 1237/16800 X[h];
L2[h_, X_] := 57/1120 X[-h] - 243/560 X[-2 h/3] + 1269/1120 X[-h/3] - 3/4 X[0] + 891/1120 X[h/3] + 27/112 X[2 h/3] - 41/1120 X[h];
L3[h_, X_] := -2067/9680 X[-h] + 6021/4840 X[-2 h/3] - 5805/1936 X[-h/3] + 1863/484 X[0] - 5697/1936 X[h/3] + 10341/4840 X[2 h/3] - 727/9680 X[h];
L4[h_, X_] := 63/16 X[-h] - 1809/40 X[-2 h/3] + 2295/16 X[-h/3] - 801/4 X[0] + 2133/16 X[h/3] - 297/8 X[2 h/3] + 233/80 X[h];
L5[h_, X_] := 123/160 X[-h] - 135/8 X[-2 h/3] + 2295/32 X[-h/3] - 132 X[0] + 3861/32 X[h/3] - 1917/40 X[2 h/3] + 149/32 X[h];
L6[h_, X_] := -6/35 X[-h] + 27/10 X[-2 h/3] - 1053/112 X[-h/3] + 57/4 X[0] - 621/56 X[h/3] + 729/140 X[2 h/3] - 277/560 X[h];
Q[h_] := 1 - h L1[h, Dfn] + L2[h, Dfn] ** (121/315 h^2 L3[h, Dfn] - 2/315 h^3 L4[h, Dfn] ** L5[h, Dfn]) +
    (2/45 h^2 L6[h, Dfn] + L2[h, Dfn] ** (-4/45 h^3 L6[h, Dfn] + 1/105 h^4 Dfn[h] ** Dfn[h])) ** Dfn[h];
R[h_] := -h L1[h, Cfn] + L2[h, Dfn] ** (121/315 h^2 L3[h, Cfn] - 2/315 h^3 L4[h, Dfn] ** L5[h, Cfn]) +
    (2/45 h^2 L6[h, Dfn] + L2[h, Dfn] ** (-4/45 h^3 L6[h, Dfn] + 1/105 h^4 Dfn[h] ** Dfn[h])) ** Cfn[h];
NCExpand[Normal[NCSeries[R[h] - R[-h] + Q[h] ** F[h] - Q[-h] ** F[-h], {h, 0, 6}]] //. substD]

0
```