Authors: S.I. Vinitsky, A.A. Gusev and O. Chuluunbaatar

Language: Fortran

The subroutine TIME6T 1.0 calculates numerical solutions $\psi(x,t)$ of the Cauchy problem for the time-dependent Schrödinger equation in the finite spatial variable interval $[x_{\min}, x_{\max}]$

$$\imath\frac{\partial\psi(x,t)}{\partial t} = \left(-\frac{1}{2}\frac{\partial^2}{\partial x^2} + f(x,t)\right)\psi(x,t), \tag{1}$$

with the initial condition

$$\psi_0(x) = \psi(x,t_0) \tag{2}$$

up to $2M(M=1,2,3)$ order of accuracy in the time step τ of a uniform grid $\Omega_\tau[t_0,T] = \{t_0, t_{k+1} = t_k + \tau, (k=0,1,\ldots,K-1), t_K = T\}$, covering the time interval $t \in [t_0,T]$, using the explicit Magnus expansion [1] of the evolution operator with the additional gauge transformations and their Padé approximations.

We require the solution $\psi(x,t)$ to be continuous and to have general first derivatives integrable with square, and to belong to the Sobolev space $\mathbf{W}_2^1([x_{\min},x_{\max}]\otimes[t_0,T])$ $(\psi_0(x) \in \mathbf{W}_2^1([x_{\min},x_{\max}]))$. Also we suppose that the function $f(x,t)$ has partial derivatives to the order of $2M$ to be continuous by time and spatial variables. The normalization condition reads

$$\|\psi(x,t)\|^2 = \int_{x_{\min}}^{x_{\max}} |\psi(x,t)|^2 dx = 1, \quad t \in [t_0,T]. \tag{3}$$

For solving the numerical solution of the problem (1)–(3) at each transforming $\psi(x,t_k)$ into $\psi(x,t_{k+1})$, we use the following implicit numerical scheme [2] with the symmetric operators $A_k^{(M)}$, $S_k^{(M)}$:

$$\begin{aligned}
&\psi_k^0 = \psi(x,t_k),\\
&\left(I - \frac{\overline{\alpha}_\eta^{(L)} S_k^{(M)}}{2L}\right)\psi_k^{\eta/L} = \left(I - \frac{\alpha_\eta^{(L)} S_k^{(M)}}{2L}\right)\psi_k^{(\eta-1)/L}, \quad \eta = 1,\ldots,L,\\
&\tilde{\psi}_k^0 = \psi_k^1,\\
&\left(I + \frac{\tau\overline{\alpha}_\zeta^{(M)} A_k^{(M)}}{2M}\right)\tilde{\psi}_k^{\zeta/M} = \left(I + \frac{\tau\alpha_\zeta^{(M)} A_k^{(M)}}{2M}\right)\tilde{\psi}_k^{(\zeta-1)/M}, \ \zeta = 1,\ldots,M,\\
&\psi_k^0 = \tilde{\psi}_k^1,\\
&\left(I + \frac{\overline{\alpha}_\eta^{(L)} S_k^{(M)}}{2L}\right)\psi_k^{\eta/L} = \left(I + \frac{\alpha_\eta^{(L)} S_k^{(M)}}{2L}\right)\psi_k^{(\eta-1)/L}, \quad \eta = 1,\ldots,L,\\
&\psi(x,t_{k+1}) = \psi_k^1.
\end{aligned} \tag{4}$$

Here $L = [\frac{M}{2}]$, I is the unit operator. The coefficients, $\alpha_\zeta^{(M)}$ $(\zeta = 1,\ldots,M,\ M \geq 1)$, stand for the roots of the polynomial equation ${}_1F_1(-M,-2M,2M\imath/\alpha) = 0$, where ${}_1F_1(a,b,x)$ is the confluent hypergeometric function and the overline indicates the complex conjugate. At $M = 1,2,3$ the operators $A_k^{(M)}$, $S_k^{(M)}$ are given by the relations

$$\begin{aligned}
A_k^{(1)} &= -\frac{1}{2}\frac{d^2}{dx^2} + f(x,t_c), \quad S_k^{(1)} = 0,\\
A_k^{(2)} &= A_k^{(1)} + \frac{\tau^2}{24}\ddot{f}, \quad S_k^{(2)} = S_k^{(1)} + \frac{\tau^2}{12}\dot{f},\\
A_k^{(3)} &= A_k^{(2)} + \frac{\tau^4}{1920}\ddddot{f} + \frac{\tau^4}{1440}\left(\frac{\partial}{\partial x}\dot{f}\right)^2 - \frac{\tau^4}{2880}\left(\frac{\partial^4}{\partial x^4}\ddot{f}\right)\\
&\quad - \frac{\tau^4}{720}\left(\frac{\partial}{\partial x}\ddot{f}\right)\left(\frac{\partial}{\partial x}f\right) - \frac{\tau^4}{720}\frac{\partial}{\partial x}\left(\frac{\partial^2}{\partial x^2}\ddot{f}\right)\frac{\partial}{\partial x},
\end{aligned} \tag{5}$$

$$S_k^{(3)} = S_k^{(2)} + \frac{\tau^4}{480}\dddot{f} + \frac{\tau^4}{720}\left(\frac{\partial}{\partial x}\dot{f}\right)\left(\frac{\partial}{\partial x}f\right) + \frac{\tau^4}{2880}\left(\frac{\partial^4}{\partial x^4}\dot{f}\right) + \frac{\tau^4}{720}\frac{\partial}{\partial x}\left(\frac{\partial^2}{\partial x^2}\dot{f}\right)\frac{\partial}{\partial x},$$

where $f \equiv f(x, t_c)$, $\dot{f} \equiv \partial_t f(x,t)|_{t=t_c}$, ... и $t_c = t_k + \tau/2$. Note that in the case M =1, the scheme corresponds to the well-known Crank- Nicolson scheme [3].

Methods:

The considered scheme was constructed on the basis of the algorithms published in paper [2]. Further discretization of the resulting problem in each layer $t = t_k$ is implemented by means of the finite-element method [4] by spatial variable $x \in [x_{\min}, x_{\max}]$, using Lagrange interpolation polynomials up to the order $p = 8$ at suitable smoothness of the solution.

References:

[1] Magnus W. Commun. Pure Appl. Math., 1954, v. 7, pp. 649–673.
[2] Chuluunbaatar O. et al. J. Phys. A, 2008, v. 41, pp. 295203–1–25.
[3] Crank J. and Nicolson P. Proc. Cambridge Philos. Soc., 1947, v. 43, pp. 50–67.
[4] Bathe K.J. *Finite element procedures in engineering analysis*. Englewood Cliffs, Prentice Hall, New York, 1982.

```
Structure:

      SUBROUTINE
      Name: TIME6T
      Internal subroutines: TMSOLV, ASSMBS, ASSMBM, EMASSD, ESTIFD,ADDVEC,
                            BOUNDC, COLMHT, ERRDIM, FEGRID,MAXHT, GAULEG,
                            NODGEN, SHAPEF, MULTC, MULTCC, REDBAC, DECOMC
      External subroutines: POTCAL, DINIT (user-supplied subroutines)

Usage:

      CALL TIME6T(TITLE,NPOL,TMIN,TMAX,TAU,ITORDR,IPRINT,IPRSTP,
     1            IPRITT,NMESH,RMESH,IBOUND,FNOUT,IOUT,FMATR,
     2            IOUM,EVWFN,IOUF,TOT,ITOT,ZTOT,MTOT,MITOT,MZTOT)
      INPUT:      TITLE,NPOL,TMIN,TMAX,TAU,ITORDR,IPRINT,IPRSTP,
                  IPRITT,NMESH,RMESH,IBOUND,FNOUT,IOUT,FMATR,
                  IOUM,EVWFN,IOUF,TOT,ITOT,ZTOT,MTOT,MITOT,MZTOT:

 TITLE   -- character symbol, title of the run to be printed on the output
            listing.
 NPOL    -- integer number, order of interpolating Lagrange polynomials.
 TMIN    -- double precision number, begin of time interval.
 TMAX    -- double precision number, begin of time interval.
 TAU     -- double precision number, time step of time variable t.
 ITORDR  -- integer number, order of Magnus expansion, 1 , 2 , 3.
 IPRINT  -- integer number, level of print, 0 , 1 , 2.
 IPRSTP  -- integer number, step of spatial variable x with which solutions
            are printed out.
 IPRITT  -- integer number, step of time variable t with which solutions
            are printed out.
 NMESH   -- integer number, dimension of array RMESH. NMESH always should be
            odd and > = 3.
```

```
 RMESH   -- double precision array, array RMESH contains information about
            subdivision of interval [x_min,x_max] of space variable x on
            subintervals. The whole interval [x_min,x_max] is divided as
            follows: RMESH(1) = x_min, RMESH(NMESH) = x_max, and the values
            of RMESH(I) set the number of elements for each subinterval
            [RMESH(I-1), RMESH(I+1)], where I=2, 4, ... , NMESH-1.
 IBOUND  -- integer number, parameter defining the type of boundary
            conditions set in the boundary points x = x_min and x = x_max:
            = 1 -- the Dirichlet -- Dirichlet boundary conditions;
            = 2 -- the Dirichlet -- Neumann boundary conditions;
            = 3 -- the Neumann -- Dirichlet boundary conditions;
            =  4 -- the Neumann -- Neumann boundary conditions.
 FNOUT   -- character symbol, name of the output file for printing out the results
            of the calculation.
 IOUT    -- integer number, number of the output logical device for printing
            out the results of the calculation.
 FMATR   -- character symbol, name of the scratch file for storing matrices.
 IOUM    -- integer number, number of the logical device for storing matrices.
 EVWFN   -- character symbol, name of the output file for storing the results
            of the calculation, namely, finite-element grid points, and
            solutions. It is used only if IOUF > 0.
 IOUF    -- integer number, number of the logical device for storing data
C           into file EVWFN.
 TOT     -- double precision array, working vector.
 ITOT    -- integer array, working vector.
 ZTOT    -- double complex array, working vector.
 MTOT    -- integer number, dimension of the working array TOT.
 MITOT   -- integer number, dimension of the working array ITOT.
 MZTOT   -- integer number, dimension of the working array ZTOT.

     OUTPUT:
            WRITE(IOUF) NN,NGRID,TT,TAU,(XGRID(J),J=1,NGRID)
    1                  ,(ZU(J),J=1,NN)

 NGRID -- integer number, the number of finite-element grid points.
 NN    -- integer number, the number of finite-element grid points of solution.
 TT    -- double precision number, the value of time variable t.
 TAU   -- double precision number, the time step of time variable t.
 XGRID -- double precision array, contains the values of the finite-element grid points.
 ZU    -- double complex array, contains calculated solution.

Example: subroutine POTCAL for the given potential
      f(x,t) = (4 - 3 * exp(-t)) * x**2 / 2:

      SUBROUTINE POTCAL(RG,TT,TAU,DA,HH,DS,SS,ITORDR)
      IMPLICIT REAL*8 (A-H,O-Z)
      DATA ZERO / 0.D0 /, ONE / 1.D0 /
C
      HH = (4 - 3 * DEXP(-TT)) * RG**2 / 2
      SS = ZERO
      DA = ONE / 2
      DS = ZERO
C
      IF (ITORDR .GE. 2) THEN
C
        F1 = -3 * DEXP(-TT) / 24 * RG**2 / 2
```

```
        HH = HH + TAU**2 * F1
        F1 =  3 * DEXP(-TT) / 12 * RG**2 / 2
        SS = SS + TAU**2 * F1
C
      END IF
C
      IF (ITORDR .GE. 3) THEN
C
        F1 = -3 * DEXP(-TT) * RG**2 / 2 / 1920
     1    + ( 3 * DEXP(-TT) * RG)**2 / 1440
     2    - (-3 * DEXP(-TT) * RG) * (4 - 3 * DEXP(-TT)) * RG / 720
        HH = HH + TAU**4 * F1
        F1 = 3 * DEXP(-TT) * RG**2 / 2 / 480
     1     + (3 * DEXP(-TT) * RG) * (4 - 3 * DEXP(-TT)) * RG / 720
        SS = SS + TAU**4 * F1
C
        DA = DA + (- 3 * DEXP(-TT)) * TAU**4 / 720
        DS = DS - (  3 * DEXP(-TT)) * TAU**4 / 720
C
      END IF
      RETURN
      END
```

```
 RG    -- double precision number, the value of spatial variable x.
 TT    -- double precision number, the value of time variable t.
 TAU   -- double precision number, the time step of time variable t.
 HH    -- double precision number, contains the operator A^(M)_k without
          differential operators from Eq. (5) at M=ITORDR.
 SS    -- double precision number, contains the operator S^(M)_k without
          differential operators from Eq. (5) at M=ITORDR.
 DA    -- double precision number, contains the coefficient of differential
          operators of operator A^(M)_k from Eq. (5) at M=ITORDR. DA = 1/2
          at ITORDR = 1 , 2, while DA = 1/2 + f_{xxtt}(x,t) * TAU**4 / 720
          at ITORDR = 3.
 DS    -- double precision number, contains the coefficient of differential
          operators of operator S^(M)_k from Eq. (5) at M=ITORDR. DS = 0
          at ITORDR = 1 , 2, while DS = - f_{xxt}(x,t) * TAU**4 / 720
          at ITORDR = 3.
```

Here parameters RG, TT, TAU should not be changed by users.

Example: subroutine DINIT for given initial condition (2)

$$psi\_0(x) = \sqrt{\sqrt{1/\pi}} * \exp(- (x - \sqrt{2}))^{**}2 / 2)$$

```
      SUBROUTINE DINIT(KEY,NN,NGRID,XGRID,ZU1)
      IMPLICIT REAL*8 (A-H,O-Y)
      IMPLICIT COMPLEX*16 (Z)
      DIMENSION ZU1(NN),XGRID(NGRID)
      DATA ONE / 1.D0 /, TWO / 2.D0 /
      PI = DACOS(- ONE)
      FG = DSQRT(DSQRT(ONE / PI))
      DO I = 1 , NN
        X = XGRID(I + KEY)
        ZU1(I) = FG * DEXP( - (X - DSQRT(TWO))**2 / 2)
      END DO
      RETURN
```

```
      END
```

```
 KEY   -- integer number, If IBOUND >= 3 then KEY = 0, else KEY = 1.
```

Here parameters KEY, NN, NGRID, XGRID should not be changed by users.

Remarks:

1. In subroutine TIME6T 1.0 is used the dynamic distribution storages as one-dimensional arrays TOT, ZTOT ITOT, RMESH. In test example their dimension is set by the values MTOT=100000, MZTOT=100000, MITOT=100000, NMESH1=5. If user has set insufficient values of dimension above arrays the message about an error is printed and the execution of the program is aborted. In the last case, in order to carry out the required calculation it is necessary to increase the dimensions of arrays to the quantity taken from the message.

2. For the accuracy control over the numerical solution by step TAU, the Runge coefficient on four twice condensed grids is additionally calculated, using the subroutine RUNGE.