subroutine RK(Pfreq\_in, E\_in, f\_in, p\_in, n, &

Pfreq\_out, E\_out, f\_out, p\_out)

double precision ::n !time step

double precision, dimension(Ny), intent(in) ::f\_in

double precision, dimension(Ny), intent(out) ::f\_out

double precision, dimension(Ny) ::kf1, kf2, kf3, kf4

complex\*16, dimension(Ny), intent(in) ::p\_in

complex\*16, dimension(Ny), intent(out) ::p\_out

complex\*16, dimension(Ny) ::kp1, kp2, kp3, kp4

complex\*16, intent(in) ::E\_in(N\_freq)

complex\*16, intent(out) ::E\_out(N\_freq)

complex\*16 ::kE1(N\_freq), kE2(N\_freq), kE3(N\_freq)&

, kE4(N\_freq)

complex\*16, intent(in) ::Pfreq\_in(N\_freq)

complex\*16, intent(out) ::Pfreq\_out(N\_freq)

complex\*16 ::kPfreq1(N\_freq), kPfreq2(N\_freq), kPfreq3(N\_freq)&

, kPfreq4(N\_freq), kPfreq1\_hlp(N\_freq), &

kPfreq2\_hlp(N\_freq), kPfreq3\_hlp(N\_freq), &

kPfreq4\_hlp(N\_freq)

integer ::num = 1 , i1

kp1 = dt \* funcp(n,f\_in, p\_in)

kf1 = dt \* funcf(n,p\_in)

kE1 = dt \* Etime(n) \* Ebind \* zexp((0.0d0, 1.0d0)\*(dt\*n+tstart)\*freqgrid)

kPfreq1\_hlp = dy\*sum(y\*kp1)/(2.0d0\*pi)

kPfreq1 = dt \* kPfreq1\_hlp \* zexp((0.0d0, 1.0d0)\*(tstart+dt\*n)\*freqgrid)

kp2 = dt \* funcp(n+0.50d0, f\_in + kf1/2.0d0, p\_in + kp1/2.0d0)

kf2 = dt \* funcf(n+0.5d0, p\_in + kp1/2.0d0)

kE2 = dt \* Etime(n+0.5d0) \* Ebind \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+0.5d0)))

kPfreq2\_hlp = dy\*sum(y\*(p\_in + kp1/2.0d0))/(2.0d0\*pi)

kPfreq2 = dt \* kPfreq2\_hlp \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+0.5d0)))

kp3 = dt \* funcp(n+0.5d0, f\_in + kf2/2.0d0, p\_in + kp2/2.0d0)

kf3 = dt \* funcf(n+0.5d0, p\_in + kp2/2.0d0)

kE3 = dt \* Etime(n+0.5d0) \* Ebind \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+0.5d0)))

kPfreq3\_hlp = dy\*sum(y\*(kp2/2.0d0))/(2.0d0\*pi)

kPfreq3 = dt \* kPfreq3\_hlp \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+0.5d0)))

kp4 = dt \* funcp(n+1.0d0, f\_in + kf3, p\_in + kp3)

kf4 = dt \* funcf(n+1.0d0, p\_in + kp3)

kE4 = dt \* Etime(n+1.0d0) \* Ebind \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+1.0d0)))

kPfreq4\_hlp = dy\*sum(y\*(p\_in + kp3))/(2.0d0\*pi)

kPfreq4 = dt \* kPfreq4\_hlp \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+1.0d0)))

p\_out = p\_in + kp1/6.0d0 +kp2/3.0d0 + kp3/3.0d0 + kp4/6.0d0

f\_out = f\_in + kf1/6.0d0 +kf2/3.0d0 + kf3/3.0d0 + kf4/6.0d0

E\_out = E\_in + kE1/6.0d0 +kE2/3.0d0 + kE3/3.0d0 + kE4/6.0d0

Pfreq\_out = Pfreq\_in + kPfreq1/6.0d0 +kPfreq2/3.0d0 + kPfreq3/3.0d0 + kPfreq4/6.0d0

if (int(n) == (num\*2500)) then

write(format\_V, '(A12, I6, A18)') '(SE24.16e3, ', Ny, '(", ",SE24.16e3))'

do i1 = 1, 1

write(700, format\_V) int(n)

end do

num = num+1

end if

end subroutine RK

function funcp(nt\_via, f\_via, p\_via)

double precision ::nt\_via

complex\*16 ::p\_via(Ny), p\_sum(Ny), test(Ny)

double precision ::f\_via(Ny), f\_sum(Ny)

complex\*16 ::funcp(Ny)

f\_sum = matmul(TRANSPOSE(coul\_mat), f\_via)

p\_sum = matmul(TRANSPOSE(coul\_mat), p\_via)

funcp = -(0.0d0,1.0d0)\*(y\*y\*p\_via - ((2.0d0\*f\_sum)\*p\_via) + shift\*p\_via - (0.0d0,1.0d0) \* &

gamma \* p\_via/Ebind - (1.0d0-2.0d0\*f\_via)\*(p\_sum+Etime(nt\_via)))/hbar\*Ebind

end function funcp

function funcf(nt\_via, p\_via)

double precision ::nt\_via

complex\*16, dimension(Ny) ::p\_via

double precision, dimension(Ny) ::funcf

! funcf = dble(aimag(conjg(Etime(nt\_via)+matmul(TRANSPOSE(coul\_mat),p\_via))\*p\_via\*2.0d0))/hbar\*Ebind

funcf = 0d0

end function funcf

module constants

implicit none

integer, parameter ::Nt = 700000

double precision, parameter ::t\_end = 35.0d0, &

dt = t\_end/dble(Nt)

integer, parameter ::Ny = 400, N\_fine = 50, Nphi = 100

double precision, parameter ::ymax = 40.0d0, dy = ymax/dble(Ny)

double precision, parameter ::pi = 4.0d0\*datan(1.0d0), &

hbar = 0.6582119514d0, e = 1.60217662d-19

double precision, parameter ::Ebind = 4.18d0, gamma = 0.39d0

double precision, parameter ::sigmat = 0.15d0, tstart = -5.0d0, &

E\_excit = 1.0d-3, shift = 4.0d0

double precision ::coul\_mat(Ny, Ny) = 0.0d0, Et(Nt+1) = 0.0d0, &

time(Nt+1) = 0.0d0, argument(Nt+1) = 0.0d0, &

y(Ny)=0.0d0, dexp\_argument(Nt+1) = 0.0d0, &

exp\_argument(Nt+1) = 0.0d0, &

y\_fine(N\_fine) = 0.0d0, dy\_fine, f(Ny) = 0.0d0, &

ft(Nt+1) = 0.0d0

complex\*16 ::p(Ny) = 0.0d0, pt(Nt+1) = 0.0d0

integer, parameter ::N\_freq = 800

complex\*16 ::p\_freq(N\_freq) = 0.0d0, E\_freq(N\_freq) = 0.0d0

double precision ::freqgrid(N\_freq) = 0.0d0, test(Nphi) = 0.0d0

integer ::i1

character(80) ::list\_file

character(len=100) ::format\_V

contains

double precision function Etime(tvia)

double precision ::tvia

Etime = E\_excit\*dexp(-(tvia\*dt+tstart)\*(tvia\*dt+tstart)/(sigmat\*sigmat)) !no E\_bind

end function Etime

subroutine readdata

integer ::i1

character(len = 100) ::format\_V

write(format\_V, '(A12, I6, A18)') '(SE24.16e3, ', N\_freq, '(", ",SE24.16e3))'

open(unit = 100, file = 'Efreq\_Ben.dat', status = 'old', action = 'read')

do i1 = 1,N\_freq

read(100, format\_V) freqgrid(i1)

end do

close(100)

freqgrid = (freqgrid\*Ebind+4d0\*Ebind)/hbar

write(list\_file, '(A)') 'freqgrid.dat' !p(t)

open(unit=700,file=list\_file)

write(format\_V, '(A12, I6, A18)') '(SE24.16e3, ', N\_freq, '(", ",SE24.16e3))'

do i1 = 1, N\_freq

write(700, format\_V) freqgrid(i1)

end do

close(700)

end subroutine readdata

subroutine constant

integer ::Ndo

do Ndo = 1, Nt+1

time(Ndo) = (dble(Ndo)-1.0d0)\*dt

Et(Ndo) = Etime(dble(Ndo))

argument(Ndo) = -(dble(Ndo)\*dt+tstart)\*(dble(Ndo)\*dt+tstart)/(sigmat\*sigmat)

dexp\_argument(Ndo) = dexp(-(dble(Ndo)\*dt+tstart)\*(dble(Ndo)\*dt+tstart)/(sigmat\*sigmat))

exp\_argument(Ndo) = exp(-(dble(Ndo)\*dt+tstart)\*(dble(Ndo)\*dt+tstart)/(sigmat\*sigmat))

end do

do Ndo = 1,Ny

y(Ndo) = dy\*(dble(Ndo) - 0.50d0)

end do

dy\_fine = dy/dble(N\_fine) !y step for finer grid

end subroutine constant

subroutine coul\_matrix

implicit none

double precision, dimension(Nphi) ::a, b, c, w1

integer ::Ndo, Ndo\_in

double precision, dimension(N\_fine) ::fine\_grid

double precision ::b\_mid1, b\_mid2(Nphi), b\_mid3(Nphi)

call constant

do Ndo = 1, Nphi

a(Ndo) = dble(Ndo)

end do

do Ndo = 1, N\_fine

fine\_grid(Ndo) = (dble(Ndo)-0.50d0)\*dy\_fine

end do

b\_mid1 = ( dble(Nphi) + 1.0d0 ) / (2.0d0 \* pi )

b\_mid2 = dsin ( (2.0d0 \* pi \* a ) / ( dble(Nphi)+ 1.0d0 ))

b\_mid3 = ( a - b\_mid1 \* b\_mid2)

b = pi / ( dble(Nphi) + 1.0d0 ) \* b\_mid3

w1 = pi / ( dble(Nphi) + 1.0d0 ) \* ( 1.0d0 - dcos( 2.0d0 \* pi \* a / ( dble(Nphi) + 1.0d0)))

do Ndo = 1, Ny

y\_fine = y(Ndo) - dy/2.0d0 + fine\_grid

do Ndo\_in = 1, Nphi

coul\_mat(:, Ndo)=coul\_mat(:,Ndo)+(1.0d0/(dsqrt(y(Ndo)\*y(Ndo)+y\*y -2.0d0\*y\*y(Ndo)\*dcos(b(Ndo\_in)))))\*2.0d0\*y\*w1(Ndo\_in)

end do

coul\_mat(Ndo, Ndo) = 0.0d0

do Ndo\_in = 1, N\_fine

coul\_mat(Ndo,Ndo)=coul\_mat(Ndo,Ndo) + sum( 1.0d0/ dsqrt( y(Ndo) \*y(Ndo) + y\_fine(Ndo\_in) \*y\_fine(Ndo\_in) - 2.0d0 \* y\_fine(Ndo\_in) \* y(Ndo) \* dcos( b ) ) / dble(N\_fine) \* 2.0d0 \* y\_fine(Ndo\_in) \* w1)

end do

end do

coul\_mat = coul\_mat/pi\*dy

end subroutine coul\_matrix

function funcp(nt\_via, f\_via, p\_via)

double precision ::nt\_via

complex\*16 ::p\_via(Ny), p\_sum(Ny), test(Ny)

double precision ::f\_via(Ny), f\_sum(Ny)

complex\*16 ::funcp(Ny)

f\_sum = matmul(TRANSPOSE(coul\_mat), f\_via)

p\_sum = matmul(TRANSPOSE(coul\_mat), p\_via)

funcp = -(0.0d0,1.0d0)\*(y\*y\*p\_via - ((2.0d0\*f\_sum)\*p\_via) + shift\*p\_via - (0.0d0,1.0d0) \* &

gamma \* p\_via/Ebind - (1.0d0-2.0d0\*f\_via)\*(p\_sum+Etime(nt\_via)))/hbar\*Ebind

end function funcp

function funcf(nt\_via, p\_via)

double precision ::nt\_via

complex\*16, dimension(Ny) ::p\_via

double precision, dimension(Ny) ::funcf

! funcf = dble(aimag(conjg(Etime(nt\_via)+matmul(TRANSPOSE(coul\_mat),p\_via))\*p\_via\*2.0d0))/hbar\*Ebind

funcf = 0d0

end function funcf

subroutine RK(Pfreq\_in, E\_in, f\_in, p\_in, n, &

Pfreq\_out, E\_out, f\_out, p\_out)

double precision ::n !time step

double precision, dimension(Ny), intent(in) ::f\_in

double precision, dimension(Ny), intent(out) ::f\_out

double precision, dimension(Ny) ::kf1, kf2, kf3, kf4

complex\*16, dimension(Ny), intent(in) ::p\_in

complex\*16, dimension(Ny), intent(out) ::p\_out

complex\*16, dimension(Ny) ::kp1, kp2, kp3, kp4

complex\*16, intent(in) ::E\_in(N\_freq)

complex\*16, intent(out) ::E\_out(N\_freq)

complex\*16 ::kE1(N\_freq), kE2(N\_freq), kE3(N\_freq)&

, kE4(N\_freq)

complex\*16, intent(in) ::Pfreq\_in(N\_freq)

complex\*16, intent(out) ::Pfreq\_out(N\_freq)

complex\*16 ::kPfreq1(N\_freq), kPfreq2(N\_freq), kPfreq3(N\_freq)&

, kPfreq4(N\_freq), kPfreq1\_hlp(N\_freq), &

kPfreq2\_hlp(N\_freq), kPfreq3\_hlp(N\_freq), &

kPfreq4\_hlp(N\_freq)

integer ::num = 1 , i1

kp1 = dt \* funcp(n,f\_in, p\_in)

kf1 = dt \* funcf(n,p\_in)

kE1 = dt \* Etime(n) \* Ebind \* zexp((0.0d0, 1.0d0)\*(dt\*n+tstart)\*freqgrid)

kPfreq1\_hlp = dy\*sum(y\*kp1)/(2.0d0\*pi)

kPfreq1 = dt \* kPfreq1\_hlp \* zexp((0.0d0, 1.0d0)\*(tstart+dt\*n)\*freqgrid)

kp2 = dt \* funcp(n+0.50d0, f\_in + kf1/2.0d0, p\_in + kp1/2.0d0)

kf2 = dt \* funcf(n+0.5d0, p\_in + kp1/2.0d0)

kE2 = dt \* Etime(n+0.5d0) \* Ebind \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+0.5d0)))

kPfreq2\_hlp = dy\*sum(y\*(p\_in + kp1/2.0d0))/(2.0d0\*pi)

kPfreq2 = dt \* kPfreq2\_hlp \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+0.5d0)))

kp3 = dt \* funcp(n+0.5d0, f\_in + kf2/2.0d0, p\_in + kp2/2.0d0)

kf3 = dt \* funcf(n+0.5d0, p\_in + kp2/2.0d0)

kE3 = dt \* Etime(n+0.5d0) \* Ebind \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+0.5d0)))

kPfreq3\_hlp = dy\*sum(y\*(kp2/2.0d0))/(2.0d0\*pi)

kPfreq3 = dt \* kPfreq3\_hlp \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+0.5d0)))

kp4 = dt \* funcp(n+1.0d0, f\_in + kf3, p\_in + kp3)

kf4 = dt \* funcf(n+1.0d0, p\_in + kp3)

kE4 = dt \* Etime(n+1.0d0) \* Ebind \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+1.0d0)))

kPfreq4\_hlp = dy\*sum(y\*(p\_in + kp3))/(2.0d0\*pi)

kPfreq4 = dt \* kPfreq4\_hlp \* zexp((0.0d0, 1.0d0)\*freqgrid\*(tstart+dt\*(n+1.0d0)))

p\_out = p\_in + kp1/6.0d0 +kp2/3.0d0 + kp3/3.0d0 + kp4/6.0d0

f\_out = f\_in + kf1/6.0d0 +kf2/3.0d0 + kf3/3.0d0 + kf4/6.0d0

E\_out = E\_in + kE1/6.0d0 +kE2/3.0d0 + kE3/3.0d0 + kE4/6.0d0

Pfreq\_out = Pfreq\_in + kPfreq1/6.0d0 +kPfreq2/3.0d0 + kPfreq3/3.0d0 + kPfreq4/6.0d0

if (int(n) == (num\*2500)) then

write(format\_V, '(A12, I6, A18)') '(SE24.16e3, ', Ny, '(", ",SE24.16e3))'

do i1 = 1, 1

write(700, format\_V) int(n)

end do

num = num+1

end if

end subroutine RK

end module constants