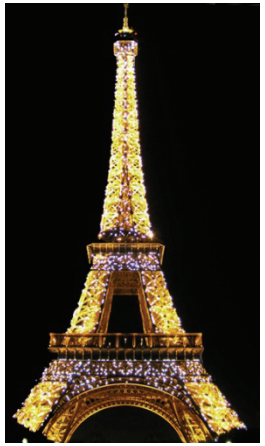
Numerical methods for attosecond strong field dynamics:

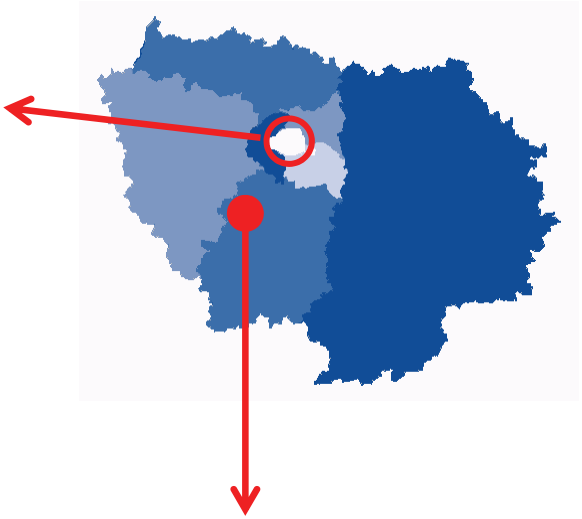
Introduction
&
Single Active Electron (SAE) dynamics

I – Grid methods

Eric Charron – Université Paris-sud

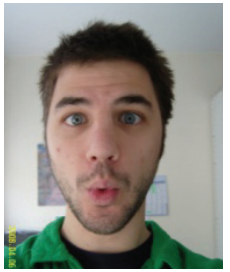
Our location





Our group

Osman



Michel

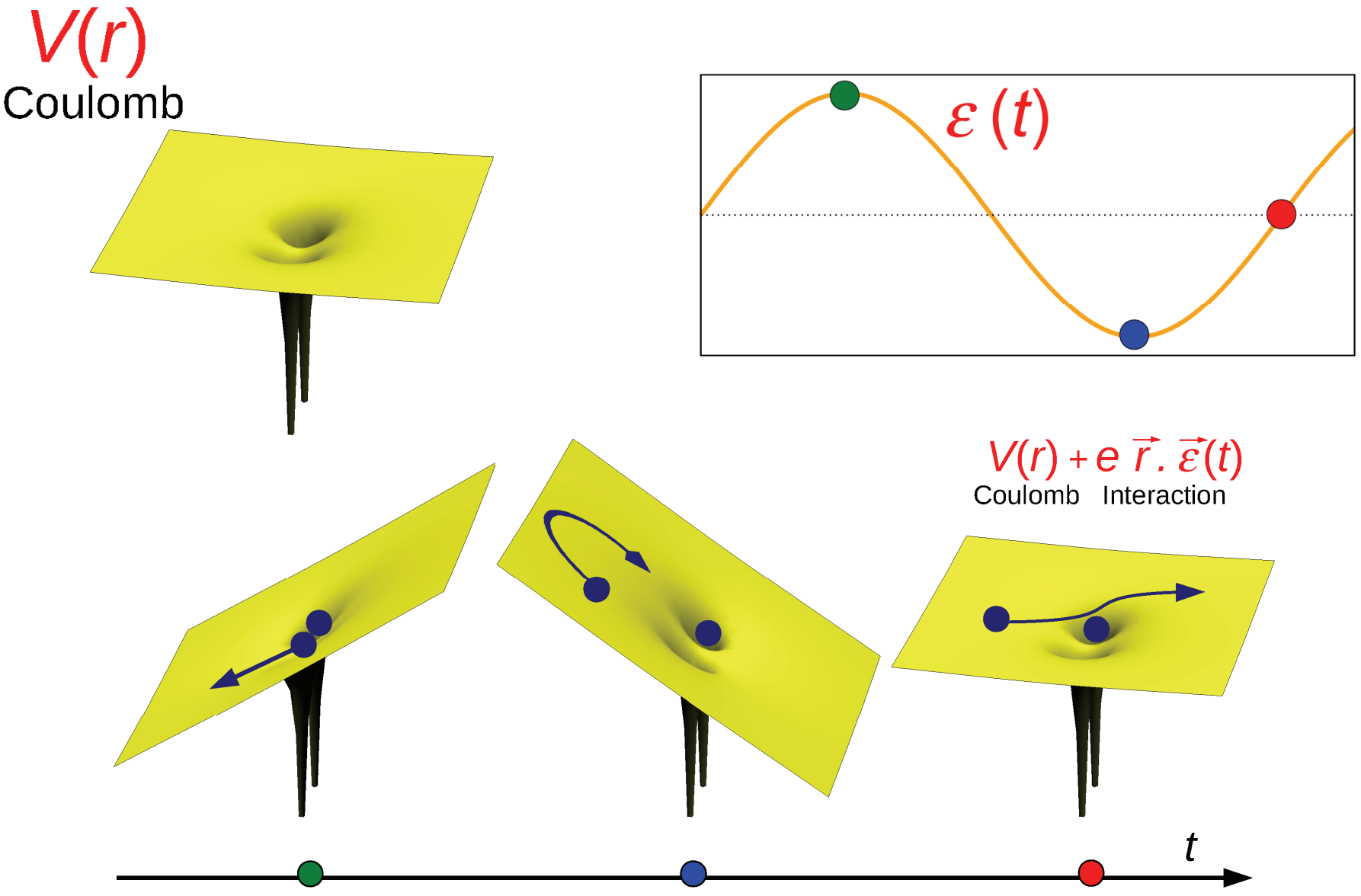
Raijumon

Arne

Eric

Jeremy

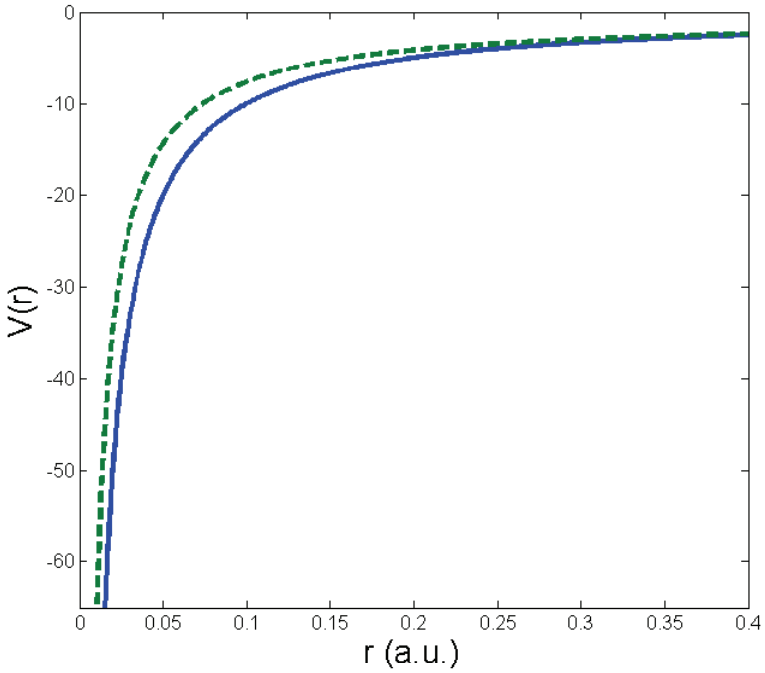
Introduction: at the origin of the SAE model…

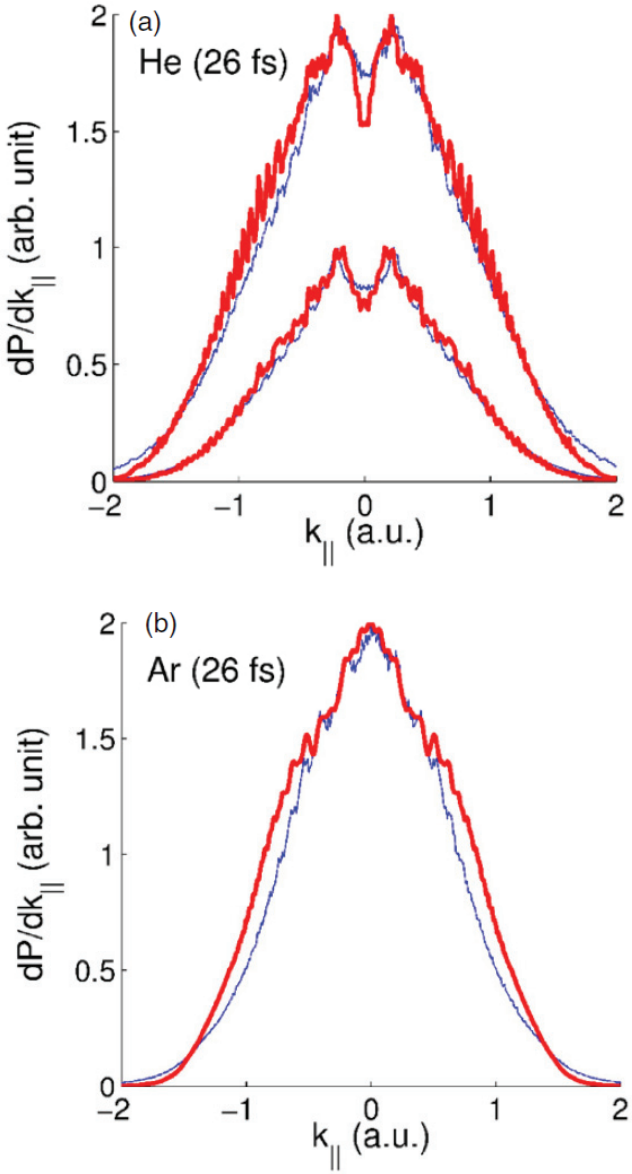


P. Corkum, Phys. Rev. Lett. **71**, 1994 (1993)

Introduction: a few specific examples… Atoms

$$V_{eff}(r) = \frac{-1 + \sum_{j=1}^{N} c_j \, e^{-\alpha_j r}}{r}$$





M. Abu-Samha, L. B. Madsen J. Phys. B 44, 235601 (2011)

```
%Time-dependent propagation in 1D using the Split-Operator method
%in a 1D grid representation - Everything in atomic units (a.u.)

%First define the 1D spatial grid z
nz=128;    %Number of grid points
zmin=-10; %Spatial grid minimum (a.u.)
zmax=10;  %Spatial grid maximum (a.u.)
dz=(zmax-zmin)/(nz-1); %Spatial grid step (a.u.) NB (nz-1)= # of intervals
z=[zmin:dz:zmax]; %Value of z (a.u.) at each grid point

%Now define the potential (harmonic oscillator here)
kspring=1.0; %Spring constant for the harmonic oscillator (a.u.)
V=0*z; %Creates null vector of size nz
for j=1:nz,
    V(j)=0.5*kspring*(z(j))^2;
end

%Now prepare an initial wavefunction
psi=0*z; %Creates null vector of size nz
psi(:)=sqrt(1/(nz*dz)); %Constant normalized initial guess

%Define time step
dt_as=0.01; %Time step in as
dt=dt_as/24.188843; %Time step in a.u.
```

```
%Now define the potential energy propagator during time interval dt
propV=0*z; %Creates null vector of size nz
for j=1:nz,
    propV(j)=exp(-V(j)*dt); %Potential energy propagator
end
```

```
%Now define the kinetic energy propagator during time interval dt/2
%First define the grid in k
dk=2*pi/(nz*dz); %Momentum grid step (a.u.)
k=0*z; %Creates null vector of size nz
for j=1:nz/2-1,
    k(j)=(j-1)*dk;     %Value of k (a.u.) at each grid point
end
for j=nz/2:nz,
    k(j)=-(nz+1-j)*dk; %Value of k (a.u.) at each grid point
end
%Then define the value of kinetic energy (a.u.) at each momentum grid point
KE=0*z; %Creates null vector of size nz
for j=1:nz,
    KE(j)=(k(j)*k(j))/(2*mass); %value of kinetic energy (a.u.)
end

%And then define the kinetic energy propagator
propT=0*z; %Creates null vector of size nz
for j=1:nz,
    propT(j)=exp(-KE(j)*dt/2); %Kinetic energy propagator
end
```

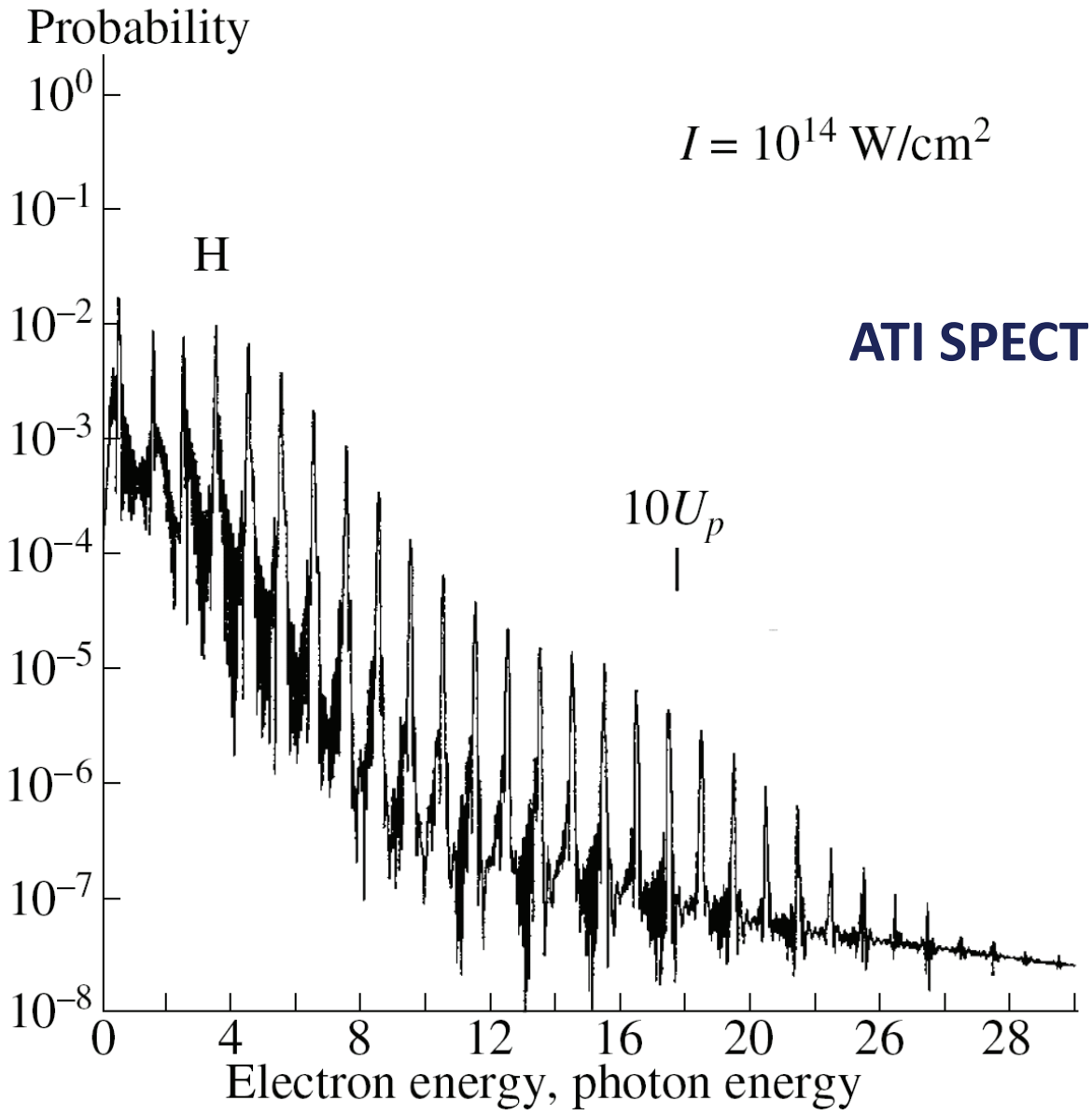
```
%Start wave packet evolution
for time=1:6000,

    %Kinetic energy propagation during time interval dt/2
    psi_momentum=fft(psi); %Fourrier transform
    for j=1:nz,
        psi_momentum(j)=propT(j)*psi_momentum(j); %Propagation
    end
    psi=ifft(psi_momentum); %Inverse Fourrier transform

    %Potential energy propagation during time interval dt
    for j=1:nz,
        psi(j)=propV(j)*psi(j); %Propagation
    end

    %Kinetic energy propagation during time interval dt/2
    psi_momentum=fft(psi); %Fourrier transform
    for j=1:nz,
        psi_momentum(j)=propT(j)*psi_momentum(j); %Propagation
    end
    psi=ifft(psi_momentum); %Inverse Fourrier transform

    %Renormalize
    Norm=sum(psi(1:nz).^2)*dz;
    psi(:)=psi(:)/sqrt(Norm);
end
```



ATI SPECTRUM

S. Chelkowski, A. D. Bandrauk, Las. Phys. ***7****, 797 (1997)*

Exercise session

– form small groups of 2 students –

1) **Pen & paper**

Show that :

$$\exp\left[-i(T+V)\Delta t\right] = \exp\left[-iT\frac{\Delta t}{2}\right]\exp\left[-iV\Delta t\right]\exp\left[-iT\frac{\Delta t}{2}\right] + o(\Delta t^3)$$

2) MATLAB

a. Read and execute the IT1D matlab code ;

b. Propose a simple way to calculate the first excited state of the harmonic oscillator within this code. Implement it, check the eigenfunction and the eigenenergy ;

c. Propose a (more complex) way to calculate the second excited state of the harmonic oscillator within this code. Implement it, check the eigenfunction and the eigenenergy ;

d. Modify the code to solve the TISE (3 lowest eigenstates) with a 1D soft-Coulomb potential. Check the influence of the softening parameter a ;

e. Modify the code to solve now the TDSE starting from the ground state of a 1D soft-Coulomb potential with a <u>single</u> optical cycle E(t) = E_0 sin(ω_Lt). Take E_0 around 0.01-0.05 a.u. and ω_L = 0.06 a.u. ;

f. Propose a simple way to calculate the <u>total</u> ionization signal. Implement it and check the influence of E_0 .

Eric CHARRON
ISMO – ORSAY
eric.charron@u-psud.fr

<u>1D Harmonic oscillator</u>

Ground state: $u_0(x) = \left(\frac{m\omega}{\pi\hbar}\right)^{\frac{1}{4}} e^{-m\omega x^2/2\hbar}$

First excited state: $u_1(x) = \left(\frac{m\omega}{\pi\hbar}\right)^{\frac{1}{4}} \sqrt{\frac{2m\omega}{\hbar}} x e^{-m\omega x^2/2\hbar}$

Second excited state: $u_2(x) = C\left(1 - 2\frac{m\omega x^2}{\hbar}\right) e^{-m\omega x^2/2\hbar}$

```
%Time-dependent propagation in 1D using the Split-Operator method
%in a 1D grid representation - Everything in atomic units (a.u.)

%First define the 1D spatial grid z
nz=128;   %Number of grid points
zmin=-10; %Spatial grid minimum (a.u.)
zmax=10;  %Spatial grid maximum (a.u.)
dz=(zmax-zmin)/(nz-1); %Spatial grid step (a.u.) NB (nz-1)= # of intervals
z=[zmin:dz:zmax]; %Value of z (a.u.) at each grid point

%Now define the potential (harmonic oscillator here)
kspring=1.0; %Spring constant for the harmonic oscillator (a.u.)
V=0*z; %Creates null vector of size nz
for j=1:nz,
    V(j)=0.5*kspring*(z(j))^2;
end
plot(z,V) %Plot the potential
pause %Wait until you press a key

%Define the exact ground state
psi_exact=0*z; %Creates null vector of size nz
mass=1; %Particle mass in a.u.
for j=1:nz,
    %Harmonic oscillator ground state
    psi_exact(j)=((sqrt(mass*kspring)/pi)^(0.25))*exp(-(z(j)*z(j))/2);
end
plot(z,psi_exact,'r','LineWidth',2)
hold on                                  %Fix the axis scaling of the plot
axis([-10 10 0 0.8])                   %Define manually the axix scaling
pause %Wait until you press a key

%Now prepare an initial wavefunction
psi=0*z; %Creates null vector of size nz
psi(:)=sqrt(1/(nz*dz)); %Constant normalized initial guess
Norm=(sum(psi(1:nz).^2))*dz %Verify norm

wavepacket_evolution=plot(z,real(psi),'-.','LineWidth',2); %Plot wavefunct

%Define time step
dt_as=0.01; %Time step in as
dt=dt_as/24.188843; %Time step in a.u.

%Now define the potential energy propagator during time interval dt
propV=0*z; %Creates null vector of size nz
for j=1:nz,
    propV(j)=exp(-V(j)*dt); %Potential energy propagator
end

%Now define the kinetic energy propagator during time intervat dt/2
%First define the grid in k
dk=2*pi/(nz*dz); %Momentum grid step (a.u.)
k=0*z; %Creates null vector of size nz
for j=1:nz/2-1,
    k(j)=(j-1)*dk;     %Value of k (a.u.) at each grid point
end
for j=nz/2:nz,
    k(j)=-(nz+1-j)*dk; %Value of k (a.u.) at each grid point
end
%Then define the value of kinetic energy (a.u.) at each momentum grid point
KE=0*z; %Creates null vector of size nz
for j=1:nz,
    KE(j)=(k(j)*k(j))/(2*mass); %value of kinetic energy (a.u.)
end

%And then define the kinetic energy propagator
propT=0*z; %Creates null vector of size nz
```

```
for j=1:nz,
    propT(j)=exp(-KE(j)*dt/2); %Kinetic energy propagator
end

%Start wave packet evolution
for time=1:6000,

    %Kinetic energy propagation during time interval dt/2
    psi_momentum=fft(psi); %Fourrier transform
    for j=1:nz,
        psi_momentum(j)=propT(j)*psi_momentum(j); %Propagation
    end
    psi=ifft(psi_momentum); %Inverse Fourrier transform

    %Potential energy propagation during time interval dt
    for j=1:nz,
        psi(j)=propV(j)*psi(j); %Propagation
    end

    %Kinetic energy propagation during time interval dt/2
    psi_momentum=fft(psi); %Fourrier transform
    for j=1:nz,
        psi_momentum(j)=propT(j)*psi_momentum(j); %Propagation
    end
    psi=ifft(psi_momentum); %Inverse Fourrier transform

    %Renormalize
    Norm=sum(psi(1:nz).^2)*dz;
    psi(:)=psi(:)/sqrt(Norm);

    set(wavepacket_evolution,'YData',real(psi)); %Redefine the y-value
    drawnow                                      %And replot the wavefunct

end

%Calculate potential energy
pen=0;
for j=1:nz,
    pen=pen+V(j)*psi(j)^2;
end
pen=pen*dz;

%Calculate kinetic energy
psi_momentum=fft(psi); %Fourrier transform
for j=1:nz,
    psi_momentum(j)=KE(j)*psi_momentum(j); %Propagation
end
psi2=ifft(psi_momentum); %Inverse Fourrier transform
ken=0;
for j=1:nz,
    ken=ken+psi(j)*psi2(j);
end
ken=ken*dz;
energy=pen+ken
```