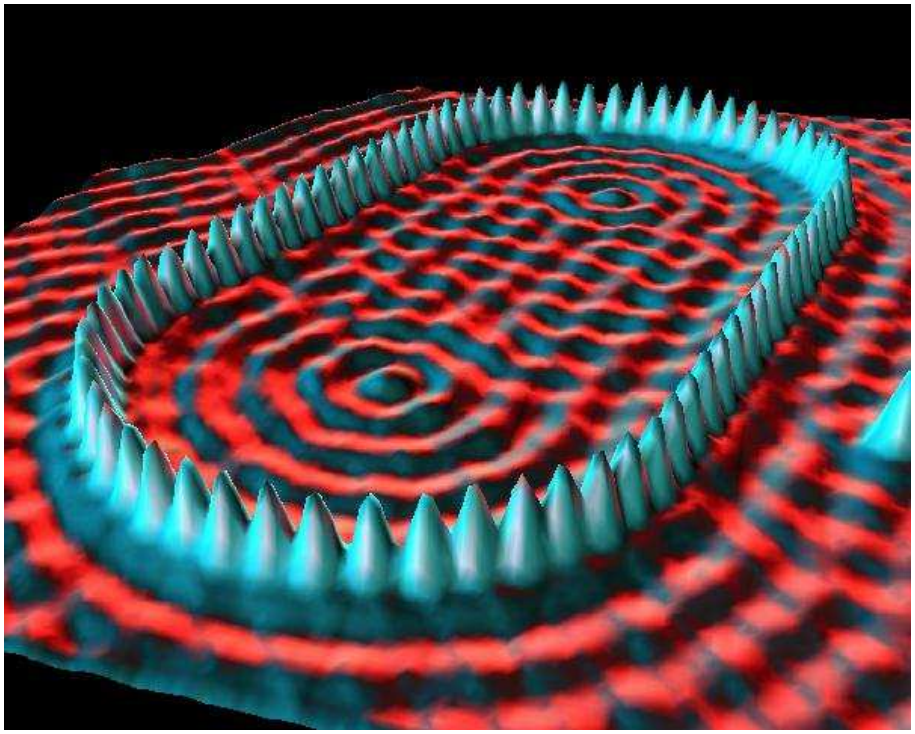
Quantum mechanics
Computation activities

David Viennot



Contents

Instructions

The practice works of the quantum mechanics course must be realised with the environment software Jupyter. This one permits to have in the same document the source code, the results (including figures) and the text of the report. Jupyter is available with the Anaconda distribution:
`https://www.anaconda.com/products/individual`
Anaconda is a general software environment (based on Python language) dedicated to scientific computing. The codes must be programmed in Python language. Exceptionally, PICS students who have never learned Python can code in Octave (a language with syntax similar to Matlab). To add an Octave kernel to Jupyter, open a terminal in the file directory of Anaconda and use the following command:
`pip install octave_kernel`
In the Jupyter Notebook (or the JupyterLab) there are two kinds of cells. "Code cells" which are preceded by `[ ]:` in which you can enter Python (or Octave) source codes. These cells can be evaluated by pressing `[shift]+[enter]`. The results of the evaluation (compilation) appears after the cell (including graphs issued from Matplotlib instructions). The "text cells" (called "Markdown") are not preceded by any symbol. You can find the syntax to format your text (headings, bold font,...) at:
`https://www.markdownguide.org/basic-syntax/`
In markdown cells you can enter LaTeX instructions to write equations, by using the usual tags: $ $ for inline equations, and $$ $$ for isolated equations. Markdown cells are also evaluated by pressing `[shift]+[enter]`. The nature of a cell can be modified in the top toolbar (drop-down menu displaying "Code" or "Markdown").

For the three practice works, you must return a single file `.ipynb` (Jupyter notebook) as your report including your source code (in compilable cells) and your results. This report must be written by respecting the rules of this kind of document (title, authors, section titles, introduction, conclusion, ...). The physical interpretations and comments, and the answers to the questions must written directly in the ipynb document. The scientific rigour concerning equations, result analyses and graphs must be respected. This work is an exercise of physics, do not be satisfied with having a running code and some figures, the main part of the work consists in the physical interpretations. The evaluation criteria are the following:

	Assessed skills		***Marking scheme***
Technical aspects	**Quality of the code**	The program runs withour error, its outputs are correct. The structure of the program is efficient. The source code is sufficiently commented.	/5
	Completeness	All the requested work has been done.	
Formatting aspects	**Quality of the report**	The report is well structured (with an introduction, a conclusion, transition sentences), is well written and well organised.	/4
	Quality of the figures	The figures are readable, the choices of colours, sizes and thicknesses are pertinent. The figures are endowed with titles and legends. The axes are named and the units are written.	
Scientific aspects	**Understanding of quantum mechanics and of mathematics**	No error about the physical principles of quantum mechanics or about the associated mathematics. Relevance of the realised computations and of the physical comparisons. Identification of fundamental physical phenomena.	/11
	Understanding of the used numerical methods	Analysis, discussion and comments concerning the numerical methods. Answers to the questions about the numerical methods. Comparisons between different numerical methods.	
	Interpretation of the physical results	Comments of the figures. Discussion about the meaning of the results. Answers to the questions about the physical aspects.	
	Understanding of the goals of the subject	Discussion about the modelling of the studied quantum systems. Inserting the subject into its scientific context. Summary of the different results and final physical discussion.	

The files must be uploaded on the CompuPhys moodle page (Quantum Physics heading):
`https://moodle.univ-fcomte.fr/course/view.php?id=4778&section=6`
Please upload your file only once per group of students (think to write the name of all members of the group in the document header).
Comments and marks concerning your works will be available on this same page. It is strongly recommended to wait until have the comments on the previous report before submitting the next one, so as not to repro-

duce the same errors.

The computation activities in the quantum mechanics course are realised in Python language (by using the Jupyter). The Python libraries *numpy*, *scipy* and *matplotlib* will be used. A complete online documentation can be found at:
`https://docs.python.org/`
`https://docs.scipy.org/doc/`
`https://matplotlib.org`
`http://doc.sagemath.org/html/en/reference/index.html`

In the numerical simulations of quantum systems, we always use the atomic units. These units are such that $\hbar = m_e = 1$ a.u. (m_e is the electron mass). The translation of these units to usual units is the following:

	1. a.u. =
mass	9.1×10^{-31} kg
length	0.53 Å
energy	27.2 eV
time	24 as
temperature	3.2×10^5 K

Chapter 1

Numerical integration of the Schrödinger equation

We consider a quantum dynamical system described by a Hamiltonian $t \mapsto H(t)$ in the Hilbert space $\mathcal{H}$. We are interested by the Schrödinger equation for the system state:

$$\imath\hbar\dot{\psi}(t) = H(t)\psi(t) \qquad \psi \in \mathcal{H}$$

or by the one for the evolution operator:

$$\imath\hbar\dot{U}(t,0) = H(t)U(t,0) \qquad U \in \mathcal{U}(\mathcal{H})$$

In this chapter, we introduce numerical integrators for these equations. We suppose that the integration concerns the time interval $[t_0, T]$. Let $\{t_0, t_1, ..., t_N\}$ be a regular partition of this interval with $t_{i+1} - t_i = \Delta t$ the time step. We suppose that the partition is sufficiently thin to $H(t)$ does not significantly change from t_i to t_{i+1}.

1.1 Theory

1.1.1 The Richardson method

We set $\psi_n = \psi(t_n)$. The SOD (second order differencing) method is based on second order finite differences which are obtained by Taylor developments:

$$\psi(t + \Delta t) = \psi(t) + \dot{\psi}(t)\Delta t + \ddot{\psi}(t)\frac{\Delta t^2}{2} + \mathcal{O}(\Delta t^3)$$

$$\psi(t - \Delta t) = \psi(t) - \dot{\psi}(t)\Delta t + \ddot{\psi}(t)\frac{\Delta t^2}{2} + \mathcal{O}(\Delta t^3)$$

where

$$\psi(t + \Delta t) - \psi(t - \Delta t) = 2\dot{\psi}(t)\Delta t + \mathcal{O}(\Delta t^3)$$

The (two steps) propagation scheme is the following:

$$\psi_{n+1} = \psi_{n-1} - 2\imath\hbar^{-1}H(t_n)\psi_n\Delta t$$

The error at each step has order of Δt^2 and the algorithm is conditionally stable, that is to say that it exists a value Δt_{max} under which the algorithm converges and above which it diverges. To verify the algorithmic convergence, it is advisable to monitor the state norm $\|\psi_n\|$ which diverges to infinity (often quickly) when $\Delta t > \Delta t_{max}$.
The two steps recurrence is initialised as follows: we find $\psi_{1/2}$ (at time $\Delta t/2$) by a first order propagation scheme (Euler method):

$$\psi_{1/2} = \psi_0 - \imath\hbar^{-1}H(t_0)\psi_0\frac{\Delta t}{2}$$

next we find ψ_1 from ψ_0 and $\psi_{1/2}$ by the second order scheme (Richardson method)

$$\psi_1 = \psi_0 - \imath\hbar^{-1}H(t_0 + \Delta t/2)\psi_{1/2}\Delta t$$

On the whole of the integration, we do never use the first order scheme (Euler method) with the Schrödinger equation because this scheme is strongly unstable when complex numbers are involved.

1.1.2 The split operator method

The split operator algorithm is based on the group rule of the evolution operator:

$$U(T, t_0) = U(t_N, t_{N-1})U(t_{N-1}, t_{N-2})...U(t_1, t_0)$$

On a small duration Δt, the time ordered exponential is almost equal to a matrix exponential:

$$U(t_{n+1}, t_n) \simeq e^{-\imath\hbar^{-1}H(t_n)\Delta t}$$

The propagation scheme is then:

$$\psi_{n+1} = e^{-\imath\hbar^{-1}H(t_n)\Delta t}\psi_n$$

The problem is then to compute the matrix exponential. This computation can be realised by using some numerical computation libraries. The quality of the approximation depends on the method used to compute the exponential. The algorithm is unconditionally stable, that is to say it converges for all Δt (this does not mean that it converges to an accurate solution if Δt is too large). The norm of ψ_n is automatically preserved except if the method used to compute the exponentials introduces errors violating the unitarity.

1.1.3 The discrete adiabatic transport method

This method is a variant of the split operator method where the computation of the matrix exponentials is assumed by diagonalisation of $H(t_n)$. We denote by $D(t_n)$ the diagonal matrix of $H(t_n)$ and by $\Phi(t_n)$ the matrix of the eigenvectors:

$$H(t_n)\Phi(t_n) = \Phi(t_n)D(t_n)$$

We have the following integration scheme:

$$\psi_{n+1} = \Phi(t_{n+1})^\dagger\Phi(t_n)e^{-\imath\hbar^{-1}D(t_n)\Delta t}\psi_n$$

where ψ_n is expressed in the eigenvector basis of $H(t_n)$ (and not in the work basis). The computation of $e^{-\imath\hbar^{-1}D(t_n)\Delta t}$ is trivial (the exponential of a diagonal matrix is the diagonal matrix of the exponentials of the eigenvalues). The algorithm is unconditionally stable but needs a diagonalisation at each step (inducing a large computation time cost). This method is usually used when an adiabatic approximation provides a restriction to a small number of eigenstates. We can then used partial diagonalisation at each step (needing specific diagonalisation methods). The error depends then on the validity of the adiabatic approximation.

The more fundamental adiabatic approximation is the following: let $\phi_a(t)$ be an instantaneous eigenvector:

$$H(t)\phi_a(t) = \lambda_a(t)\phi_a(t) \qquad \lambda_a(t) \in \mathbb{R}$$

The eigenvalue λ_a being supposed no globally degenerate. If the variations of $H(t)$ with respect to the time are slow, and if λ_a does not cross another eigenvalue during the evolution, or more precisely if

$$\sup_t \max_{b\neq a} \left| \frac{\langle\phi_b(t)|\dot{H}(t)|\phi_a(t)\rangle}{\lambda_a(t) - \lambda_b(t)} \right| \ll 1$$

then we have $\forall t > 0$

$$\psi(t) \simeq e^{\imath\varphi_a(t)}\phi_a(t)$$

if $\psi(0) = \phi_a(0)$, where φ_a is just a phase. This approximation means that if we start on an instantaneous equilibrium state (ϕ_a) under slow transformations (quasi-static regime) the system remains on the same equilibrium state, except if an eigenvalue crossing (a resonance) occurs.

1.2 Application

The goal of this problem is the study of a dynamical quantum system with comparison between the Richardson and the split operator methods. The result will be analyse by an adiabatic approach.

1.2.1 Algorithm

Problem data

```
tmin ← -30.                  # starting time (in atomic unit)
tmax ← 80.                   # ending time (in atomic unit)
ntime ← 400                  # number of steps in the time discretisation
Deltat ← (tmax-tmin)/ntime   # duration of the time step (in atomic unit)
```

We suppose that it exists a list of vectors denoted psi such that psi(0) is the initial condition and a function H which returns the value of the Hamiltonian matrix with respect to the time.

Richardson integrator

```
epsilon ← 0.1                # tolerable error concerning the norm conservation
psihalf ← psi(0) - imag *H(tmin) · psi(0)*Deltat/2   # imag is the pure imaginary number
psi(1) ← psi(0) - imag * H(tmin+Deltat/2) · psihalf*Deltat
for i from 1 to ntime-1 do
        psi(i+1) ← psi(i-1) - 2*imag *H(tmin+i*Deltat) · psi(i)*Deltat
        if squareroot(conjugation(psi(i+1)) · psi(i+1)) > 1+epsilon then
                display 'divergence'
                stop the loop        # security to stop the program in the divergent cases
        end
end
```

Attention: do not confuse the matrix product '·' with the number product '*'!

Split operator integrator

```
for i from 0 to ntime-1 do
        psi(i+1) ← matrix_exp(-imag*H(tmin+i*Deltat)*Deltat) · psi(i)
end
```

1.2.2 Useful Python functions

To code this algorithm, it needs to load the following libraries:

```
import numpy as np;
from numpy import linalg as LA;
from scipy import linalg as LA2;
```

To define matrices of type `matrix` of SageMath, we have the functions: `matrix`, `matrix.identity` and `diagonal_matrix`. To define matrices of type `array` of numpy, we have the functions: `np.array`, `np.identity`, and `np.diag`.
The computation of the spectrum of a hermitian matrix is provided by the numpy function `LA.eigvalsh`, to obtain the eigenvalues and the eigenvectors we have `LA.eigh`. To compute the matrix exponential we can use the scipy function `LA2.expm`. The matrix product (action of a matrix onto a vector, product of two matrices, scalar product of two vectors – without conjugation of the left vector –) is obtained with the numpy function `np.dot`
To draw the graphs, we can use the MatPlotLib function `matplotlib.pyplot`.

1.2.3 STIRAP

We consider a three level atom interacting with two laser fields, a first one called "pump" which is quasi-resonant with the transition $|1\rangle \to |2\rangle$ and a second one called "Stokes" which is quasi-resonant with the

transition $|2\rangle \to |3\rangle$. The system Hamiltonian in the one photon rotating wave approximation (RWA) is

$$H(\Omega_P, \Omega_S) = \frac{\hbar}{2} \begin{pmatrix} 0 & \Omega_P & 0 \\ \Omega_P & 2\Delta_P & \Omega_S \\ 0 & \Omega_S & 2(\Delta_P - \Delta_S) \end{pmatrix}$$

$\Omega_P = |\langle 1|\vec{\mathcal{E}}_P \cdot \vec{\mu}|2\rangle|$ and $\Omega_S = |\langle 2|\vec{\mathcal{E}}_S \cdot \vec{\mu}|3\rangle|$ where $\vec{\mathcal{E}}_{P/S}$ are the laser electric fields and $\vec{\mu}$ is the atom electric dipolar moment. $\Delta_P = E_2 - E_1 - \hbar\omega_P$ and $\Delta_S = E_3 - E_2 - \hbar\omega_S$ are the detunings (the gaps between the atom transition energies and the photon energy). $\Delta_{P/S}$ are supposed to be constant ($\Delta_P = 0.5$ a.u. and $\Delta_S = -0.5$ a.u.). The laser fields are gaussian pulses:

$$\Omega_{P/S}(t) = \Omega_0 e^{-(t-t_{P/S})^2/\tau_{S/P}^2}$$

with $\Omega_0 = 3.5$ a.u. and $\tau_{S/P} = 15$ a.u.. We have the choice to send in first the pump laser pulse ($t_P = 10$ a.u.) and in second the Stokes pulse ($t_S = 35$ a.u.), or to choice the contrary ($t_P = 35$ a.u. and $t_S = 10$ a.u.).

1. After the construction of the functions H and $\Omega_{P/S}$, draw the gaussian pulses with respect to the time (in a first time we choose to send first the pump pulse before the Stokes pulse).

2. At the initial time $t_{min} = -30$ a.u., the atom is in the state $|1\rangle$, intuitively in what state should the atom be at the final time $t_{max} = 80$ a.u. (when the two pulses are turned off)?

3. a. Integrate the Schrödinger equation by using the Richardson and the split operator methods.

   b. Draw on a single figure the time-dependent populations $t \mapsto |\langle 1|\psi(t)\rangle|^2$, $t \mapsto |\langle 2|\psi(t)\rangle|^2$ and $t \mapsto |\langle 3|\psi(t)\rangle|^2$.

   c. Compare the two integration methods with respect to the time step Δt (or equivalently with respect to N (ntime)). Comment.

4. After the numerical solving of the Schrödinger equation, in what state is the atom?

5. Integrate anew the Schrödinger equation but by sending first the Stokes pulse before the pump pulse. What is the result? Comment.

6. Draw on a single figure the time-dependent eigenvalues of $H(\Omega_P(t), \Omega_S(t))$ (the eigenvalues must at each time by ordered by increasing values) for the two cases (pump then Stokes and Stokes then pump).

7. Knowing that we can show that the dynamics is adiabatic except at the times where the energy levels cross, and that at these times the system state follows the more regular path, interpret the simulation outcomes. The counter-intuitive quantum effect enlighten by these simulations is called STIRAP (*STImulated Raman Adiabatic Passage*).

Chapter 2

Numerical representation of infinite dimensional systems

We consider a quantum particle of mass m described by the Hilbert space $\mathcal{H} = L^2(\mathbb{R}, dx)$ and the Hamiltonian $H = -\frac{\hbar^2}{2m}\frac{d^2}{dx^2} + V(x)$. In this chapter, we introduce an approximate finite representation of the system which can be numerically implemented.

We supposed to begin that the particle remains localised into the interval $[0, L] \subset \mathbb{R}$ (the wave function locally prepared during the experiment, cannot visit during a reasonable time interval the whole of $\mathbb{R}$). The choice consisting to represent the system only on $[0, L]$ constitutes a spatial truncation (we can always redefine the coordinates to the interval begins at 0).

Let $\{x_0, x_1, ..., x_N\}$ be a regular partition of the interval with $x_{i+1} - x_i = \Delta x$ being the partition step ($x_0 = 0$ and $x_N = L$). We suppose that the partition is sufficiently thin to the induced discretisation (pixilation) does not induce the loss of too much details. The numerical representation is based on this partition. The points of the partition are called collocation points.

2.1 Theory

2.1.1 Finite difference representation

The finite difference representation is based on the $|x\rangle$-representation of the states of $\mathcal{H}$. We choose $(|x_i\rangle)_{i=0,...,N}$ as the basis of the representation. We have then for $\psi \in \mathcal{H}$, $|\psi\rangle \simeq \sum_{i=0}^{N} \psi_i |x_i\rangle$ (with $\psi_i = \langle x_i|\psi\rangle = \psi(x_i)$). We have then

$$|\psi\rangle_{FD} = \begin{pmatrix} \psi_0 \\ \vdots \\ \psi_N \end{pmatrix}$$

Note the special normalisation of $|\psi\rangle_{FD}$:

$$\int_{x_0}^{x_N} |\psi(x)|^2 dx \simeq \sum_{i=0}^{N} |\psi_i|^2 \Delta x = 1$$

The action of the operator $\hat{V}$ on ψ, $\hat{V}\psi(x) = V(x)\psi(x)$, can be written in the following matrix form:

$$\begin{pmatrix} V(x_0) & & 0 \\ & \ddots & \\ 0 & & V(x_N) \end{pmatrix} \begin{pmatrix} \psi_0 \\ \vdots \\ \psi_N \end{pmatrix} = \begin{pmatrix} V(x_0)\psi_0 \\ \vdots \\ V(x_N)\psi_N \end{pmatrix}$$

This induces the matrix representation:

$$\hat{V}_{FD} = \begin{pmatrix} V(x_0) & & 0 \\ & \ddots & \\ 0 & & V(x_N) \end{pmatrix}$$

To represent the momentum operator $\hat{p} = -\imath\hbar\frac{d}{dx}$, it needs to use a first order Taylor development:

$$\begin{aligned} \psi(x_{i+1}) &= \psi(x_i + \Delta x) \\ &= \psi(x_i) + \left.\frac{d\psi}{dx}\right|_{x=x_i} \Delta x + \mathcal{O}(\Delta x^2) \end{aligned}$$

We have then

$$-\imath\hbar \left.\frac{d\psi}{dx}\right|_{x=x_i} \simeq -\imath\hbar\frac{\psi_{i+1} - \psi_i}{\Delta x}$$

This induces the matrix representation

$$\hat{p}_{FD} = -\frac{\imath\hbar}{\Delta x} \begin{pmatrix} -1 & 1 & 0 & & 0 \\ 0 & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & 0 & -1 & 1 \\ 0 & & 0 & 0 & -1 \end{pmatrix}$$

Note that this representation is a non-selfadjoint matrix! To solve this problem, we can consider the first other Taylor development:

$$\begin{aligned} \psi(x_{i-1}) &= \psi(x_i - \Delta x) \\ &= \psi(x_i) - \left.\frac{d\psi}{dx}\right|_{x=x_i} \Delta x + \mathcal{O}(\Delta x^2) \end{aligned}$$

By using the average of the two developments, we have:

$$-\imath\hbar \left.\frac{d\psi}{dx}\right|_{x=x_i} \simeq -\imath\hbar\frac{\psi_{i+1} - \psi_{i-1}}{2\Delta x}$$

This induces the selfadjoint matrix representation:

$$\hat{p}_{FD} = -\frac{\imath\hbar}{2\Delta x} \begin{pmatrix} 0 & 1 & 0 & & 0 \\ -1 & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ 0 & & 0 & -1 & 0 \end{pmatrix}$$

To represent the kinetic Hamiltonian $H_K = -\frac{\hbar^2}{2m}\frac{d^2}{dx^2}$ it needs to use a second order Taylor development:

$$\begin{aligned} \psi(x_{i+1}) &= \psi(x_i) + \left.\frac{d\psi}{dx}\right|_{x=x_i} \Delta x + \left.\frac{d^2\psi}{dx^2}\right|_{x=x_i} \frac{\Delta x^2}{2} + \mathcal{O}(\Delta x^3) \\ \psi(x_{i-1}) &= \psi(x_i) - \left.\frac{d\psi}{dx}\right|_{x=x_i} \Delta x + \left.\frac{d^2\psi}{dx^2}\right|_{x=x_i} \frac{\Delta x^2}{2} + \mathcal{O}(\Delta x^3) \end{aligned}$$

The sum of the two equations provides:

$$-\frac{\hbar^2}{2m} \left.\frac{d^2\psi}{dx^2}\right|_{x=x_i} \simeq -\frac{\hbar^2}{2m}\frac{\psi_{i+1} + \psi_{i-1} - 2\psi_i}{\Delta x^2}$$

and then

$$H_{K,FD} = -\frac{\hbar^2}{2m\Delta x^2} \begin{pmatrix} -2 & 1 & 0 & & 0 \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ 0 & & 0 & 1 & -2 \end{pmatrix}$$

2.1.2 DVR/FBR representations

We consider two representation bases of functions, a DVR (*Discrete Variable Representation*) basis in which the operators $V(x)$ are diagonal, and a FBR (*Finite Basis Representation*) basis in which the differential operators are diagonal.
The DVR basis $(u_i(x))_{i=0,...,N}$ is composed by functions such that $u_i(x_j) = \delta_{ij}$. The behaviour of $u_i(x)$ is almost the same than the one of the Dirac distribution (or more precisely $\lim_{N\to+\infty} u_i(x) = \delta(x - x_i)$ where the limit is defined for a discretisation step tending to 0). The FBR basis $(\phi_n)_{n=0,...,N}$ is chosen as being the Fourier transform of the DVR basis. By using a discrete Fourier transform, we have:

$$u_j(x) = \frac{1}{\sqrt{N+1}} \sum_{n=0}^{N} e^{-\imath k_n x_j} \phi_n(x)$$

where $\{k_n\}_{n=0,...,N}$ are the points of the reciprocal lattice of $\{x_0, ..., x_N\}$: $k_n = \frac{2\pi}{L}(n - \frac{N+1}{2})$ (if $N \in 2\mathbb{N}+1$) or $k_n = \frac{2\pi}{L}(n - \frac{N}{2})$ (if $N \in 2\mathbb{N}$). We have then for a function $\psi \in \mathcal{H}$:

$$\psi(x) \simeq \sum_{j=0}^{N} \psi_j^{DVR} u_j(x) = \sum_{n=0}^{N} \psi_n^{FBR} \phi_n(x)$$

with $\psi_j^{DVR} = \langle u_j|\psi\rangle$ and $\psi_n^{FBR} = \langle \phi_n|\psi\rangle$. In order to speed up the computation, we often use the approximation $\psi_j^{DVR} \simeq \psi(x_j)$ (but the price to pay is a decrease of the accuracy). The two representations are related by:

$$\psi_n^{FBR} = \frac{1}{\sqrt{N+1}} \sum_{j=0}^{N} \psi_j^{DVR} e^{-\imath k_n x_j}$$

$$\psi_j^{DVR} = \frac{1}{\sqrt{N+1}} \sum_{n=0}^{N} \psi_n^{FBR} e^{\imath k_n x_j}$$

The discrete Fourier transforms are poor approximations of the Fourier transform, in practice we prefer to use a fast Fourier transform (FFT) algorithm. Such an algorithm is relatively complicated but we can find FFT functions or subroutines in a lot of numerical computation libraries.
The matrix representations are then:

$$|\psi\rangle_{DVR} = \begin{pmatrix} \psi_0^{DVR} \\ \vdots \\ \psi_N^{DVR} \end{pmatrix} \xrightarrow{FFT} |\psi\rangle_{FBR} = \begin{pmatrix} \psi_0^{FBR} \\ \vdots \\ \psi_N^{FBR} \end{pmatrix}$$

$$\hat{V}_{DVR} = \begin{pmatrix} V(x_0) & & 0 \\ & \ddots & \\ 0 & & V(x_N) \end{pmatrix}$$

$$\hat{p}_{FBR} = \hbar \begin{pmatrix} k_0 & & 0 \\ & \ddots & \\ 0 & & k_N \end{pmatrix}$$

$$H_{K,FBR} = -\frac{\hbar^2}{2m} \begin{pmatrix} k_0^2 & & 0 \\ & \ddots & \\ 0 & & k_N^2 \end{pmatrix}$$

Moreover, the kinetic Hamiltonian can be written in the DVR basis as

$$[H_{K,DVR}]_{ij} = \sum_{n=0}^{N} \left.\frac{\partial^2 \phi_n}{\partial x^2}\right|_{x=x_i} \overline{\phi_n(x_j)}$$

The choice of the DVR basis depends on the circumstances ($u_j(x)$ can be a sinus function, a cardinal sinus, a Tchebychev polynomial, a Laguerre polynomial, an Hermitte polynomial, a Legendre polynomial,...).

According to the treated system, a basis choice could provide better results than another one. A choice relatively universal is the following:

$$u_j(x) = \frac{1}{\sqrt{(N+1)L}} \sum_{n=0}^{N} e^{\imath k_n (x - x_j)}$$

$$\phi_n(x) = \frac{1}{\sqrt{L}} e^{\imath k_n x}$$

This choice is rather natural because it consists to choice the FBR basis as constituted by the plane waves of the $|\hbar k\rangle$-representation. Note that in this case, we have an analytic expression of the representation of the kinetic operator in the DVR basis:

$$[H_{K,DVR}]_{ij} = \frac{\hbar^2}{2m} \begin{cases} \frac{\pi^2((N+1)^2+2)}{3L^2} & \text{if } i = j \\ \frac{(-1)^{j-i} 2\pi^2}{L^2 \sin^2((j-i)\pi/(N+1))} & \text{if } i \neq j \end{cases}$$

2.1.3 Numerical computation of the spectrum

H is numerically represented by an order $N+1$ matrix. The wave number k and the spatial coordinate x being related by Fourier transform, the spatial discretisation induces a cut-off of the wave numbers $[-\frac{1}{2\Delta x}, \frac{1}{2\Delta x}]$. Moreover, the spatial cut-off induces a lost of resolution for the wave numbers. We cannot distinguish two wave numbers with a gap lower than $\Delta k = \frac{1}{L}$. This induces for the spectrum the following properties:

- The numerical spectrum presents a spectral cut-off $[-\frac{\hbar^2}{8m\Delta x^2}, \frac{\hbar^2}{8m\Delta x^2}]$. It needs then that the discretisation step Δx be sufficiently thin to do not lose the low energy spectrum (the atomic and molecular spectra are always lower bounded). For all space discretisation step, the very high energy spectrum is lost.

- The numerical spectrum presents a spectral resolution with accuracy $\Delta\lambda = \frac{\hbar^2}{m}|k|\Delta k = \frac{\hbar\sqrt{2}}{\sqrt{mL}}\sqrt{|\lambda|}$. Two spectral values with gap lower than $\Delta\lambda$ are not numerically distinguishable. The space cut-off must be chosen sufficiently far to the eigenvalues of the pure point spectrum be separated. In contrast, the continuum spectrum is discretised (it is replaced by a sequence of discrete values spaced by $\Delta\lambda$). The space cut-off must be chosen sufficiently far to the values representing the continuum be sufficiently close to each other, in order to $\mathrm{Sp}_{cont}(H)$ be distinguishable of $\mathrm{Sp}_{pp}(H)$. We note that $\Delta\lambda$ increases with $|\lambda|$, the spectral resolution is then more poor at the boundaries of the spectrum.

- The numerical representation of the Hamiltonian being a $(N+1)$-order matrix, the sum of the eigenvalue multiplicities is necessarily $N+1$. If it exists less of $N+1$ eigenvalues in the spectral window $[-\frac{\hbar^2}{8m\Delta x^2}, \frac{\hbar^2}{8m\Delta x^2}]$ then artefacts appear (a false zero eigenvalue or an artificial increase of the degeneracy of the zero eigenvalue). If the number of eigenvalues in the spectral window is grower than $N+1$, some eigenvalues are lost in the numerical representation (in general a lost eigenvalue is very close to a kept eigenvalue as is the two ones have merged without increase of the degeneracy degree).

The continuous spectrum being discretised, the associated states have a numerical behaviour similar to bound states, they do not have scattering. This is an artefact of the space cut-offs. These ones are indeed similar to infinite wall potentials at 0 and L. The propagative waves associated with the continuum states are then reflected by these walls. The reflected waves interfere with the propagative ones, producing stationary waves associated with the discretised continuum spectrum.

2.2 Applications

The goal of this problem is the comparison of the finite difference representation with a DVR representation for three known quantum systems.

2.2.1 Algorithm

Problem data

```
xmax ← 10.                  # maximal value of x (in atomic unit): [0,xmax]
m ← 1.                      # particle mass (in atomic unit)
Nstep ← 30                  # number of cells in the partition
DeltaX ← xmax/Nstep         # Δx, partition step (in atomic unit)
Xd ← {i*DeltaX for i from 0 to Nstep}   # list of points of the partition
```

To define the system, we need to define the potential.

```
valV ← {V(Xd(i)) for i from 0 to Nstep}
```

The function V depends on the treated system (see the sequel).

Hamiltonian construction

We begin by the construction of the kinetic Hamiltonian with the finite difference method:

```
Hfd ← -2.*identity_matrix(Nstep+1)      # the matrix have order equal to Nstep+1
for i from 0 to Nstep-1 do
        Hfd(i,i+1) ← 1.
        Hfd(i+1,i) ← 1.
end
Hdf ← -1./(2*m*DeltaX^2)*Hfd
```

We built also the Hamiltonian with the DVR method:

```
Hdvr ← matrix(Npas+1)       # Hdvr must be initialized as a matrix
for i from 0 to Nstep do
        for j from 0 to Nstep do
                if i=j then
                        Hdvr(i,i) ← pi^2*((Nstep+1)^2+2)/(3*xmax^2*2*m)
                else
                        Hdvr(i,j) ← (-1)^(j-i)*2*pi^2/(xmax^2 *sin((j-i)*pi/(Nstep+1))^2*2*m)
                end
        end
end
```

We add the potential:

```
Vmat ← diagonal_matrix(valV)
Hdf ← Hdf + Vmat
Hdvr ← Hdvr + Vmat
```

2.2.2 Useful Python functions

To code this algorithm it needs to load the following libraries:

```
import numpy as np;
from numpy import linalg as LA;
```

To define matrices of type `matrix` of SageMath, we have the functions: `matrix`, `matrix.identity` and `diagonal_matrix`. To define matrices of type `array` of numpy, we have the functions: `np.array`, `np.identity`, and `np.diag`.
The computation of the spectrum of a hermitian matrix is provided by the numpy function `LA.eigvalsh`, to obtain the eigenvalues and the eigenvectors we have `LA.eigh`.
To draw the graphs, we can use the MatPlotLib function `matplotlib.pyplot`.

2.2.3 Particle in a box

The first system is a particle in a box $[0, x_{max}]$, the potential is then

$$V(x) = \begin{cases} +\infty & \text{si } x = 0 \\ 0 & \text{si } x \in]0, x_{max}[\\ +\infty & \text{si } x = x_{max} \end{cases}$$

The description of the infinite wall is difficult. We replace the infinity by a very large value (as for example 10^{10}).

1. After the construction of the numerical Hamiltonians of the system, compute the two approximations of the spectrum: $\mathrm{Sp}^{FD}(H)$ and $\mathrm{Sp}^{DVR}(H)$ (the eigenvalues must be ordered by increasing values).

2. Compare $\mathrm{Sp}^{FD}(H)$, $\mathrm{Sp}^{DVR}(H)$ and $\mathrm{Sp}^{th}(H) = \{\frac{\hbar^2 n^2 \pi^2}{2m x_{max}^2}, n \in \mathbb{N}^*\}$ (theoretical spectrum of a particle in a box). We can draw the three sequences $(\lambda_n^{FD/DVR/th})_n$ onto the same figure. Comment.

3. Compute the eigenvectors of the numerical Hamiltonians. It needs to well attribute the numbers of these eigenvectors. Do not forget the renormalisation of the eigenvectors by the factor $1/\sqrt{\Delta x}$.

4. Draw the wave functions corresponding to $n = 1, 2$ and 3. Make a comparison between $\phi_n^{FD}(x)$, $\phi_n^{DVR}(x)$, and $\phi_n^{th}(x) = \sqrt{\frac{2}{x_{max}}}\sin(\frac{n\pi x}{x_{max}})$. Comment.

2.2.4 Harmonic oscillator

The second system is an harmonic oscillator with potential

$$V(x) = \frac{1}{2}k(x - \frac{x_{max}}{2})^2$$

with $k = 1$ a.u.. We denote $\omega_0 = \sqrt{k/m}$

Treat the same questions that for the particle in a box, knowing that:

$$\mathrm{Sp}^{th}(H) = \{\hbar\omega_0(n + \frac{1}{2}), n \in \mathbb{N}\}$$

$$\phi_n^{th}(x) = \frac{1}{\sqrt{2^n n!}}\left(\frac{m\omega_0}{\pi\hbar}\right)^{1/4} H_n\left(\sqrt{\hbar^{-1}m\omega_0}(x - \frac{x_{max}}{2})\right) e^{-\frac{m\omega_0}{2\hbar}(x-\frac{x_{max}}{2})^2}$$

where H_n is the n-th Hermitte polynomial.

2.2.5 H_2^+ molecule

The third system is the molecular ion H_2^+ in the bound electronic state ${}^2\Sigma_g^+$ viewed only from the point of view of its vibrational states. To this system, the data must be modified with $x_{max} = 15$ a.u. and $m = 911.489$ a.u.. We must choose at least Nstep=50. The (anharmonic) vibration potential is

$$V(x) = V_0(e^{-2a(x-x_0)} - 2e^{-a(x-x_0)})$$

with $V_0 = 0.10262$ a.u., $a = 0.72$ a.u. and $x_0 = 2$ a.u..

I. Draw the potential $V(x)$ and compare it to a **CONSISTENT** harmonic oscillator potential.

II. Treat the same questions that for the particle in a box.
There are not theoretical spectrum for the vibration of H_2^+, but an accurate computation shows that:

n	$\mathrm{Sp}_{pp}(H)$ (a.u.)
1	-9.731×10^{-2}
2	-8.711×10^{-2}
3	-7.748×10^{-2}
4	-6.841×10^{-2}
5	-5.991×10^{-2}
6	-5.195×10^{-2}
7	-4.459×10^{-2}
8	-3.778×10^{-2}
9	-3.153×10^{-2}
10	-2.585×10^{-2}
11	-2.073×10^{-2}
12	-1.618×10^{-2}
13	-1.218×10^{-2}
14	-0.876×10^{-2}
15	-0.590×10^{-2}
16	-0.360×10^{-2}
17	-0.187×10^{-2}
18	-0.070×10^{-2}
19	-0.062×10^{-2}

$$Sp_{cont}(H) = [0, +\infty[$$

III. Study the numerical representation of $\mathrm{Sp}_{cont}(H)$ with respect to x_{max} and Nstep for the two methods (finite differences and DVR). Comment.

Chapter 3

Wave packet propagation

We consider a dynamical system described by the Hilbert space $\mathcal{H} = L^2(\mathbb{R}, dx)$ and the Hamiltonian $t \mapsto H(t) = -\frac{\hbar^2}{2m}\frac{d^2}{dx^2} + V(x,t)$. We are interested by the Schrödinger equation $\imath\hbar\dot{\psi}(t) = H(t)\psi(t)$ with initial condition a wave packet $\psi(x, t=0) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty}\psi_k(t=0)e^{\imath kx}dk$. This situation groups the two problems viewed in the previous activities: the integration of the Schrödinger equation and the representation of infinite dimensional systems. We will work then with a space partition $\{x_0, ..., x_M\}$ of step Δx and with a time partition $\{t_0, ..., t_N\}$ of step Δt.

3.1 Theory

3.1.1 The split operator method with DVR/FBR bases

We consider a variant of the split operator method based on the fact that H_K is diagonal in FBR basis whereas $V(x,t)$ is diagonal in DVR basis. We can compute the matrix exponentials as matrices of exponentials by passing from a basis to the other one.
To compute $e^{-\imath\hbar^{-1}H(t_n)\Delta t_n} = e^{-\imath\hbar^{-1}(H_K+V(t_n))\Delta t_n}$ it needs to split the exponential into terms where H_K and V are separated. To this, we see by using the Baker-Campbell-Hausdorff formula at second order that:

$$\begin{aligned} e^{\lambda A}e^{\lambda B}e^{\lambda A} &= e^{\lambda A+\lambda B+\frac{\lambda^2}{2}[A,B]+\mathcal{O}(\lambda^3)}e^{\lambda A} \\ &= e^{2\lambda A+\lambda B+\frac{\lambda^2}{2}[A,B]+\frac{\lambda^2}{2}[B,A]+\mathcal{O}(\lambda^3)} \\ &= e^{2\lambda A+\lambda B+\mathcal{O}(\lambda^3)} \end{aligned}$$

We conclude that

$$e^{-\imath\hbar^{-1}H(t_n)\Delta t_n} = e^{-\imath\hbar^{-1}V(t_n)\frac{\Delta t}{2}}e^{-\imath\hbar^{-1}H_K\Delta t}e^{-\imath\hbar^{-1}V(t_n)\frac{\Delta t}{2}}$$

The integration scheme is then the following:

$$\begin{aligned} \psi_{n,1}^{DVR} &= e^{-\imath\hbar^{-1}V_{DVR}(t_n)\frac{\Delta t}{2}}\psi_n^{DVR} \\ \psi_{n,1}^{DVR} &\xrightarrow{FFT} \psi_{n,1}^{FBR} \\ \psi_{n,2}^{FBR} &= e^{-\imath\hbar^{-1}H_{K,FBR}\Delta t}\psi_{n,1}^{FBR} \\ \psi_{n,2}^{FBR} &\xrightarrow{FFT^{-1}} \psi_{n,2}^{DVR} \\ \psi_{n+1}^{DVR} &= e^{-\imath\hbar^{-1}V_{DVR}(t_n)\frac{\Delta t}{2}}\psi_{n,2}^{DVR} \end{aligned}$$

We remark that we can compute $e^{-\imath\hbar^{-1}H_{K,FBR}\Delta t}$ only once, because this matrix does not change with the time.
This algorithm is unconditionally stable and unitary, the error at each step has order of Δt^3.

3.2 Applications

The goal of this problem is the study of wave packet propagations in different situations.

3.2.1 Algorithm

General data

```
xmax ← 10.                  # maximal value of x (in atomic unit): [0,xmax]
m ← 1.                      # particle mass (in atomic unit)
Nstep ← 100                 # number of cells in the space partition
DeltaX ← xmax/Nstep         # Δx, space step (in atomic unit)
Xd ← {i*DeltaX for i from 0 to Nstep}   # list of the points of the space partition
tmax ← 16.                  # ending time (in atomic unit)
ntime ← 300                 # number of steps in the time partition
Deltat ← tmax/ntime         # time step (in atomic unit)
```

(We suppose that $t_{min} = 0$).

Hamiltonian construction (in DVR basis)

```
HK ← matrix(Nstep+1)        # HK must be initialized as a matrix
for i from 0 to Nstep do
        for j from 0 to Nstep do
                if i=j then
                        HK(i,i) ← pi^2*((Nstep+1)^2+2)/(3*xmax^2*2*m)
                else
                        HK(i,j) ← (-1)^(j-i)*2*pi^2/(xmax^2 *sin((j-i)*pi/(Nstep+1))^2*2*m)
                end
        end
end
Vmat ← diagonal_matrix({V(Xd(j)) for j from 0 to Nstep})   # only if V is time-independent
H ← HK + Vmat               # only if V is time-independent
```

Propagator

In the case where H is time-independent:

```
expH = matrix_exp(-imag*H*Deltat)
for i from 0 to ntime-1 do
        psi(i+1) ← expH · psi(i)
end
```

In the case where H is time-dependent:

```
expHK ← matrix_exp(-imag*HK*Deltat)
for i from 0 to ntime-1 do
        ExpV ← diagonal_matrix({exp(-imag*V(Xd(j),i*Deltat)*Deltat/2) for j from 0 to Nstep})
        ExpH ← expV · expHK · expV
        psi(i+1) ← expH · psi(i)
end
```

3.2.2 Useful Python functions

To code this algorithm, it needs to load the following libraries:

```
import numpy as np;
from numpy import linalg as LA;
from scipy import linalg as LA2;
import matplotlib.pyplot as py;
import mpl_toolkits.mplot3d as mp3d;
```

To draw 3D graphs of two variable functions, we can use the matplotlib function `plot_surface`. To draw density colour graphs, we can use the matplotlib function `pcolor`.

3.2.3 Wave packet in a box and quantum carpet

We consider a particle of mass $m = 1$ a.u. in a box $[0, x_{max}]$ ($x_{max} = 10$ u.a). At the initial time, the particle is in a state which is a gaussian wave packet with standard deviation $\sigma_x = 0.5$ a.u., centred at the box middle, and with zero initial momentum:

$$\psi(x, t = 0) = \frac{1}{\sqrt{\sigma_x \sqrt{2\pi}}} e^{-\frac{(x - \frac{x_{max}}{2})^2}{4\sigma_x^2}}$$

1. After the construction of the system Hamiltonian (we set $V = 0$ without enforcing the strict boundary conditions), propagate the wave packet during the time until $t_{max} = 16$ a.u..
2. Draw the particle position probability with respect to the time $(x, t) \mapsto |\psi(x, t)|^2$. (we can draw a 3D graph or a colour density graph).
3. Enlighten the phenomenon of wave packet dispersion.
4. By comparison of the obtained quantum dynamics with the classical dynamics of a purely corpuscular particle, discuss the differences between the notions of stationary regime in quantum mechanics and the notion of rest in classical mechanics.
5. What is happening around $t = 2$ a.u. concerning the information on the initial wave packet? But what is happening around $t = 16$ a.u.? (This phenomenon is called wave packet revival). Why seems this phenomenon naively be in contradiction with the second principle of thermodynamics?
6. Explain why it is not a contradiction, by using the colour density graph of $(x, t) \mapsto |\psi(x, t)|^2$ (which is called a quantum carpet).

3.2.4 Wave packet in front of a potential barrier

We consider a particle with mass $m = 1$ a.u. onto $\mathbb{R}$ (represented by $[0, x_{max}]$ with $x_{max} = 20$ a.u.). A potential barrier is erected from $x_0 = 10$ a.u. to $x_1 = 11$ a.u.:

$$V(x) = \begin{cases} 0 & \text{if } x < x_0 \\ V_0 & \text{if } x \in [x_0, x_1] \\ 0 & \text{if } x > x_1 \end{cases}$$

The barrier height is $V_0 = 3$ a.u.. The particle initial state is a gaussian wave packet with standard deviation $\sigma_x = 0.5$ a.u., centred at $\frac{x_0}{2} = 5$ a.u., and with initial momentum $k_0 > 0$ (propagation to the right):

$$\psi(x, t = 0) = \frac{1}{\sqrt{\sigma_x \sqrt{2\pi}}} e^{-\frac{(x - \frac{x_0}{2})^2}{4\sigma_x^2}} e^{\imath k_0 x}$$

1. Compute $k_E = \frac{\sqrt{2mV_0}}{\hbar}$. What is the physical meaning of $\hbar k_E$ in classical dynamics?
2. After the construction of the system Hamiltonian, propagate the wave function during the time until $t_{max} = 4$ a.u. (we can choose ntime equal to 100), for different values of k_0 below and above k_E.
3. Draw the particle position probability with respect to the time $(x, t) \mapsto |\psi(x, t)|^2$ for different values of k_0 (as colour density graphs and/or as 3D graphs).
4. Interpret the different results.

3.2.5 Quantum Fermi accelerator

We consider a particle of mass $m = 1$ a.u. in a box $[0, x_{max}]$ ($x_{max} = 20$ a.u.). A potential barrier with time-dependent width is erected on the right of the box:

$$V(x, t) = \begin{cases} 0 & \text{if } x < x_{max} - \frac{(x_{max} - x_{min})}{2}(1 - \cos(\omega t)) \\ V_0 & \text{if } x \geq x_{max} - \frac{(x_{max} - x_{min})}{2}(1 - \cos(\omega t)) \end{cases}$$

with $V_0 = 10^{15}$ a.u., $\omega = \frac{2\pi}{\tau}$ and $x_{min} = \frac{3}{4}x_{max}$. At the initial time, the particle is in an eigenvector of $H(t = 0)$:

$$\psi(x, t = 0) = \sqrt{\frac{2}{x_{max}}} \sin\left(\frac{p\pi x}{x_{max}}\right)$$

with $p \in \mathbb{N}^*$.

1. After the construction of the system Hamiltonian, propagate the wave packet during the time until $t_{max} = 30$ a.u. (we can choose ntime equal to 400), for different values of τ included between 0.5 and 30 a.u., for $p = 1, 2$ or 3.

2. Draw the particle position probability with respect to the time $(x, t) \mapsto |\psi(x, t)|^2$. (We can draw a 3D graph or a density colour graph).

3. Discuss the simulation outcomes.