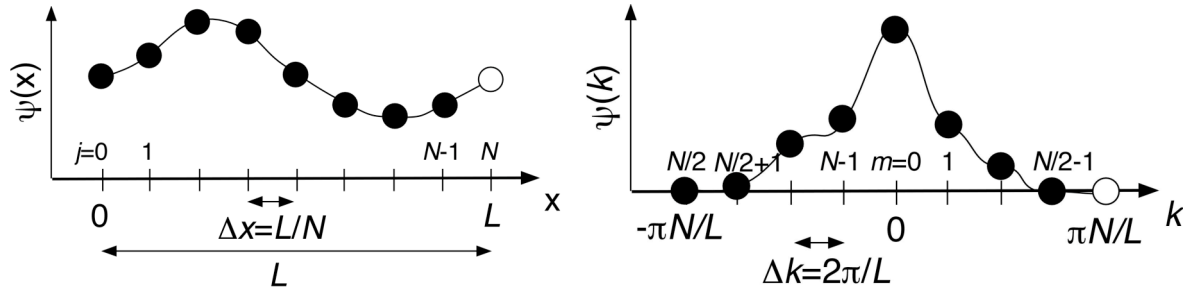
Quantum Dynamics Basics—Spectral Method

In this chapter, we will solve the time-dependent Schrödinger equation using another numerical technique, i.e., the spectral method, which is based on Fourier transformation.

§1. Discrete Fourier Transform

Consider a complex-valued function, $\psi(x) \in \mathbf{C}$, in the range, $x \in [0, L]$. We assume the periodic boundary condition: $\psi(x + L) = \psi(x)$. Let us discretize $\psi(x)$ on N mesh points, $x_j = j\Delta x$ ($j = 0, ..., N-1$), with equal mesh spacing, $\Delta x = L/N$ (see the left figure below). We denote the discrete function values as $\psi_j = \psi(x_j)$.



Discrete Fourier transformation represents $\psi(x)$ as a linear combination of trigonometric functions, $\exp(ikx) = \cos(kx) + i \sin(kx)$, with different wave numbers, k:

$$\psi_j = \sum_{m=0}^{N-1} \tilde{\psi}_m \exp\left(ik_m x_j\right), \tag{1}$$

where the discrete wave numbers, k_m, are defined as

$$k_m = \begin{cases} 2\pi m / L & (m = 0,1,\ldots,N/2-1) \\ 2\pi(m-N)/L & (m = N/2, N/2+1,\ldots,N-1) \end{cases}, \tag{2}$$

and the expansion coefficients are given by

$$\tilde{\psi}_m = \frac{1}{N} \sum_{j=0}^{N-1} \psi_j \exp\left(-ik_m x_j\right). \tag{3}$$

Note that the choice of wave numbers in Eq. (2) guarantees that $\psi(x)$ has the periodicity of L. Also, because of the discrete sampling in the real space, wave numbers separated by $2\pi nN/L = 2\pi n/\Delta x$ ($n = \pm 1$, ± 2, ...) are all equivalent. (Higher wave numbers oscillate more, but come back to the same value as their lower wave number counterparts at x_j.) Among these equivalent wave numbers, we use the smallest-magnitude wave number, since physically it represents the lowest-energy state and, mathematically, the discrete mesh points in the real space cannot represent higher wave numbers. Accordingly, in Eq. (2), the wave numbers for the higher indices, $m = N/2, N/2+1, ..., N-1$, are folded back by $2\pi N/L$, so that all the wave numbers are in the range, $[-\pi/\Delta x, \pi/\Delta x]$, see the right figure above. (For simplicity, we assume that N is an even number.)

To prove the correctness of the above Fourier expansion, it is convenient to think the discrete function, ψ_j, as a vector in the N-dimensional vector space: $|\psi\rangle = (\psi_0, \psi_1, \ldots, \psi_{N-1})$. In this vector space, we define the plane-wave basis set,

$$\left\{ |m\rangle = b_m(x_j) = \frac{1}{\sqrt{N}} \exp(ik_m x_j) \middle| m = 0,1,\ldots,N-1 \right\}, \tag{4}$$

which is **orthonormal**, i.e., the inner products of the basis functions are

$$\langle m|n\rangle \equiv \sum_{j=0}^{N-1} b_m^*(x_j) b_n(x_j) = \delta_{m,n} = \begin{cases} 1 & m = n \\ 0 & m \neq n \end{cases}. \tag{5}$$

∵For $m \neq n$, carry out the sum of geometric series; otherwise ($m = n$), all N summands are $1/N$.

$$\langle m|n\rangle = \frac{1}{N}\sum_{j=0}^{N-1} \exp\left(i(k_n - k_m)x_j\right) = \frac{1}{N}\sum_{j=0}^{N-1} \exp\left(i\frac{2\pi}{N}(n-m)j\right)$$

$$= \begin{cases} \dfrac{1}{N}\dfrac{\exp\left(i2\pi(n-m)\right)-1}{\exp\left(i\frac{2\pi}{N}(n-m)\right)-1} = 0 & m \neq n \\ \dfrac{1}{N} \bullet N = 1 & m = n \end{cases} \quad .//$$

The above basis set is also **complete**, i.e., any discrete function, ψ_j, in this N-dimensional vector space can be represented as a linear combination of N basis set functions, $b_m(x_j)$. Specifically,

$$|\psi\rangle = \sum_{m=0}^{N-1} |m\rangle\langle m|\psi\rangle, \tag{6}$$

or

$$1 = \sum_{m=0}^{N-1} |m\rangle\langle m|. \tag{7}$$

∵Suppose the function is expanded as

$$|\psi\rangle = \sum_{n=0}^{N-1} c_n |n\rangle.$$

Multiplying both sides by $\langle m|$ and using the orthonormality, Eq. (5), we get $\langle m|\psi\rangle = c_m$.//

The Fourier coefficients, $\tilde{\psi}_m$, in Eq. (3) are readily obtained from Eq. (6). Substituting the definitions of the basis functions and the inner product in Eq. (6), we obtain

$$\psi_j = \sum_{m=0}^{N-1} \exp\left(ik_m x_j\right) \frac{1}{N} \sum_{l=0}^{N-1} \exp\left(-ik_m x_l\right)\psi_l .$$

Comparing this equation with Eq. (1) identifies the expansion coefficients, $\tilde{\psi}_m$, in Eq. (1) as Eq. (3).

§2. Spectral Method for Integrating Time-Dependent Schrödinger Equation

HAMILTONIAN OPERATOR

Consider the time-dependent Schrödinger equation in one dimension in atomic unit,

$$i\frac{\partial}{\partial t}\psi(x,t) = H\psi(x,t), \tag{8}$$

where the Hamiltonian operator, H, is defined as

$$H = -\frac{1}{2}\frac{\partial^2}{\partial x^2} + V(x), \tag{9}$$
$$= T + V$$

with T and V being the kinetic- and potential-energy operators, respectively.

The kinetic energy operator is diagonal in the Fourier (or momentum) space. To see this, we operate T on the wave function in its Fourier representation, Eq. (1):

$$-\frac{1}{2}\frac{\partial^2}{\partial x^2}\sum_{m=0}^{N-1}\tilde{\psi}_m \exp(ik_m x) = \sum_{m=0}^{N-1}\frac{k_m^2}{2}\tilde{\psi}_m \exp(ik_m x), \tag{10}$$

i.e., the kinetic energy operator multiplies the factor, $k_m^2/2$, to the Fourier coefficient of the wave function:

$$\tilde{\psi}_m \xrightarrow[T]{} \frac{k_m^2}{2}\tilde{\psi}_m .$$

Recall, on the other hand, the potential energy operator is diagonal in the real space, i.e., it multiplies the factor, $V_j = V(x_j)$ to the wave function:

$$\psi_j \xrightarrow[V]{} V_j\psi_j .$$

SPLIT-OPERATOR TECHNIQUE AND SPECTRAL METHOD

The above observation, that the kinetic- and potential-energy operators are diagonal in the real- and momentum-spaces, respectively, suggests an efficient algorithm for the time evolution of the wave function. Recall the Trotter expansion (also called the split-operator technique):

$$\psi(x,t+\Delta t) = \exp(-iV\Delta t/2)\exp(-iT\Delta t)\exp(-iV\Delta t/2)\psi(x,t) + O\left(\left[\Delta t\right]^3\right). \tag{11}$$

The time evolution operator, $\exp(-iV\Delta t/2)$, arising from the potential energy, is easily operated in the real space,

$$\exp(-iV\Delta t/2)\psi_j = \exp(-iV_j\Delta t/2)\psi_j , \tag{12}$$

or

$$\psi_j \xrightarrow[\exp(-iV\Delta t/2)]{} \exp(-iV_j\Delta t/2)\psi_j .$$

On the other hand, the time evolution operator, $\exp(-iT\Delta t)$, arising from the kinetic energy, is operated in the Fourier space as

$$\exp(-iT\Delta t)\tilde{\psi}_m = \exp(-ik_m^2\Delta t/2)\tilde{\psi}_m , \tag{13}$$

or

$$\tilde{\psi}_m \xrightarrow[\exp(-iT\Delta t)]{} \exp(-ik_m^2\Delta t/2)\tilde{\psi}_m .$$

The spectral method is formally represented in terms of the forward and inverse Fourier transformation operators, F and F^{-1},

$$\psi_j \xrightarrow[F^{-1}]{} F^{-1}\psi_j = \tilde{\psi}_m = \frac{1}{N}\sum_{j=1}^{N}\psi_j \exp(-ik_m x_j), \tag{14}$$

$$\tilde{\psi}_m \xrightarrow[F]{} F\tilde{\psi}_m = \psi_j = \sum_{m=1}^{N}\tilde{\psi}_m \exp(ik_m x_j), \tag{15}$$

as follows

$$\psi(t+\Delta t) = \exp(-iV\Delta t/2)F\exp(-iT\Delta t)F^{-1}\exp(-iV\Delta t/2)\psi(t). \tag{16}$$

Equation (16) amounts to the following algorithm.

Spectral Split-Operator Algorithm

1. $\psi_j \leftarrow \exp\left(-iV_j \Delta t/2\right)\psi_j$
2. $\tilde{\psi}_m \xleftarrow[F^{-1}]{} \psi_j$
3. $\tilde{\psi}_m \leftarrow \exp\left(-ik_m^2 \Delta t/2\right)\tilde{\psi}_m$
4. $\psi_j \xleftarrow[F]{} \tilde{\psi}_m$
5. $\psi_j \leftarrow \exp\left(-iV_j \Delta t/2\right)\psi_j$

COMPUTATION OF THE ENERGY

The total energy is a conserved quantity for the time-dependent Schrödinger equation, Eq. (8), and is useful for estimating the discretization error. The total energy can be calculated as follows:

$$\begin{aligned}\langle H\rangle &= \langle T\rangle + \langle V\rangle \\ &= \int dx\psi^*(x)\left(-\frac{1}{2}\frac{\partial^2}{\partial x^2}\right)\psi(x) + \int dx\psi^*(x)V(x)\psi(x). \\ &\cong dx\sum_{j=0}^{N-1}\psi_j^*\left(-\frac{1}{2}\frac{\partial^2}{\partial x^2}\right)\psi_j + dx\sum_{j=0}^{N-1}V_j\left|\psi_j\right|^2\end{aligned} \tag{17}$$

To calculate the first term (i.e., the kinetic energy) in Eq. (17), let us expand the wave function in terms of its Fourier components as in Eq. (1):

$$\begin{aligned}\langle T\rangle &= dx\sum_{j=0}^{N-1}\sum_{m=0}^{N-1}\tilde{\psi}_m^*\exp\left(-ik_m x_j\right)\left(-\frac{1}{2}\frac{\partial^2}{\partial x^2}\right)\sum_{n=0}^{N-1}\tilde{\psi}_n\exp\left(ik_n x_j\right) \\ &= dx\sum_{j=0}^{N-1}\sum_{m=0}^{N-1}\tilde{\psi}_m^*\exp\left(-ik_m x_j\right)\frac{k_n^2}{2}\sum_{n=0}^{N-1}\tilde{\psi}_n\exp\left(ik_n x_j\right) \\ &= dx\sum_{m=0}^{N-1}\tilde{\psi}_m^*\frac{k_n^2}{2}\sum_{n=0}^{N-1}\tilde{\psi}_n\sum_{j=0}^{N-1}\exp\left(i\left(k_n-k_m\right)x_j\right) \\ &= dx\sum_{m=0}^{N-1}\tilde{\psi}_m^*\frac{k_n^2}{2}\sum_{n=0}^{N-1}\tilde{\psi}_n N\delta_{m,n} \\ &= dxN\sum_{m=0}^{N-1}\frac{k_m^2}{2}\left|\tilde{\psi}_m\right|^2\end{aligned} \tag{18}$$

By substituting Eq. (18) in (17), we obtain

$$\begin{aligned}\langle H\rangle &= \langle T\rangle + \langle V\rangle \\ &= dxN\sum_{m=0}^{N-1}\frac{k_m^2}{2}\left|\tilde{\psi}_m\right|^2 + dx\sum_{j=0}^{N-1}V_j\left|\psi_j\right|^2.\end{aligned} \tag{19}$$

§3. Fast Fourier Transform

The bottleneck in implementing the above spectral method is the computational cost associated with the discrete Fourier transform. Since the computation of each of the N Fourier coefficients, $\tilde{\psi}_m$, involves summation over N terms, the computational time grows as $O(N^2)$. The fast Fourier transform (FFT) algorithm[1] reduces this complexity to $O(N\log N)$, and makes the quantum-dynamics simulation less compute-intensive. The discussion in this lecture note follows Chapter 12 in the *Numerical Recipes*.[2] First, download and read sections 12.1 "Fourier Transform of Discretely Sampled Data" and 12.2 "Fast Fourier Transform" at http://www.library.cornell.edu/nr/bookcpdf.html.

DANIELSON-LANCZOS ALGORITHM

The summation in the Fourier transform can be split into two partial sums as follows:

$$\begin{aligned}\psi_j &= \sum_{m=0}^{N-1}\tilde{\psi}_m \exp\left(ik_m x_j\right) = \sum_{m=0}^{N-1}\tilde{\psi}_m \exp\left(i2\pi mj/N\right)\\ &= \sum_{m=0}^{N/2-1}\tilde{\psi}_{2m} \exp\left(i2\pi(2m)j/N\right) + \sum_{m=0}^{N/2-1}\tilde{\psi}_{2m+1} \exp\left(i2\pi(2m+1)j/N\right)\\ &= \sum_{m=0}^{N/2-1}\tilde{\psi}_{2m} \exp\left(i2\pi mj/(N/2)\right) + \exp\left(i2\pi j/N\right)\sum_{m=0}^{N/2-1}\tilde{\psi}_{2m+1} \exp\left(i2\pi mj/(N/2)\right)\end{aligned}.$$

Therefore,

$$\psi_j = \psi_j^0 + W_N^j \psi_j^1, \tag{20}$$

where

$$\begin{cases}\psi_j^0 = \sum_{j=0}^{N/2-1}\tilde{\psi}_{2m} \exp\left(i2\pi mj/(N/2)\right)\\ \psi_j^1 = \sum_{j=0}^{N/2-1}\tilde{\psi}_{2m+1} \exp\left(i2\pi mj/(N/2)\right).\\ W_N = \exp\left(i2\pi/N\right)\end{cases} \tag{21}$$

Note that ψ_j^0 and ψ_j^1 represent $N/2$-element Fourier transforms consisting of even and odd sub-arrays, respectively. Accordingly, in the sub-array Fourier components, ψ_j^0 and ψ_j^1, j should be read as j mod (N/2), i.e., an index in a N/2-long Fourier transform.

The top part of the figure below illustrates this decomposition due to Danielson and Lanczos for an 8-element Fourier transform, specifically for

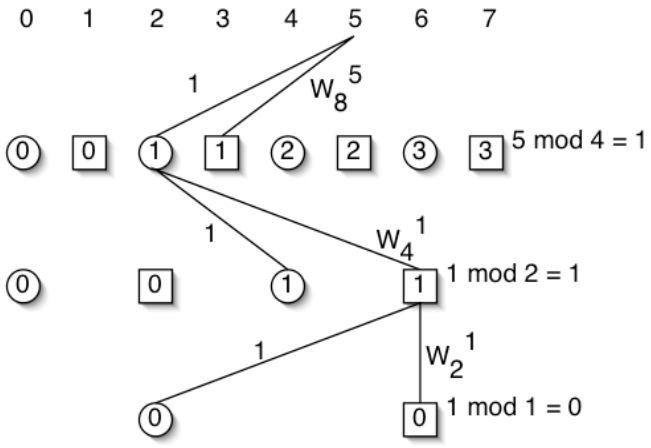
$$\psi_5 = \psi_{5\bmod 4=1}^0 + W_8^5 \psi_{5\bmod 4=1}^1,$$

where circles and squares denote the even and odd sub-arrays, respectively.

---

[1] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," Math. Comput. **19**, 297 (1965).

[2] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C, 2nd Ed.* (Cambridge U Press, 1993).



In a similar manner, each $N/2$-element Fourier transform is further decomposed into two $N/4$-element Fourier transforms, e.g.,

$$\begin{aligned}\psi_j^0 &= \sum_{m=0}^{N/4-1}\tilde{\psi}_{2(2m)}\exp\left(i2\pi(2m)j/(N/2)\right) + \sum_{m=0}^{N/4-1}\tilde{\psi}_{2(2m+1)}\exp\left(i2\pi(2m+1)j/(N/2)\right)\\ &= \sum_{m=0}^{N/4-1}\tilde{\psi}_{2(2m)}\exp\left(i2\pi mj/(N/4)\right) + \exp\left(i2\pi j/(N/2)\right)\sum_{m=0}^{N/4-1}\tilde{\psi}_{2(2m+1)}\exp\left(i2\pi mj/(N/4)\right)\end{aligned}.$$

Therefore,

$$\psi_j^0 = \psi_j^{00} + W_{N/2}^j\psi_j^{01}, \tag{22}$$

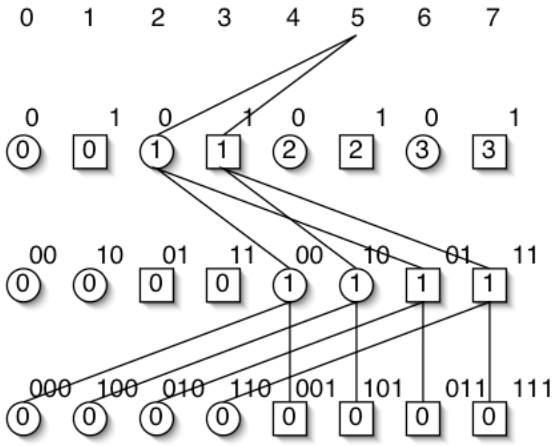
where

$$\begin{cases}\psi_j^{00} = \sum_{m=0}^{N/4-1}\tilde{\psi}_{2(2m)}\exp\left(i2\pi mj/(N/4)\right)\\ \psi_j^{01} = \sum_{m=0}^{N/4-1}\tilde{\psi}_{2(2m+1)}\exp\left(i2\pi mj/(N/4)\right).\\ W_{N/2} = \exp\left(i2\pi/(N/2)\right)\end{cases} \tag{23}$$

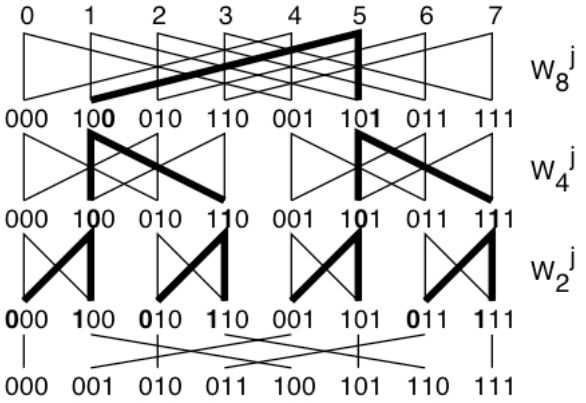
In ψ_j^{00} and ψ_j^{01} (see Eq. (23)), j should be read as j mod (N/4).

The decomposition of Fourier transform into sums of two sub-array Fourier transforms, e.g., Eqs. (20) and (22), continues recursively. The figure above illustrates how the even (circles) and odd (squares) sub-array Fourier components are combined. Eventually, the sub-array contains only one element, at which stage the recursion terminates, and the sub-array Fourier components is a function value at some grid point. For example, in the figure above, the bottom circle and square represent ψ_j^{010} and ψ_j^{011}, which turn out to be $\tilde{\psi}_2$ and $\tilde{\psi}_6$, respectively. In general, we can obtain the wave function index from the bit sequence to specify the recursive sub-array by reversing the bit sequence and converting it to decimal. This works because successive subdivisions of the data into even and odd are tests of successive low-order (least significant) bits of the wave function index.

The figure below shows all the wave functions that participate in the construction of ψ_5, and how they are combined to construct ψ_5.



In the FFT algorithm, the input wave function values are first re-ordered by applying the bit-reversal operation to each wave function index. The Danielson-Lanczos procedures, such as Eqs. (20) and (22), are then applied recursively, starting from the smaller sub-arrays up. The figure below shows all the combinations of sub-array Fourier coefficients to construct all the Fourier components in the bit-reversed scheme, in which the combinations to construct ψ_5 are represented by bold lines.



The figure above shows that, to compute all N Fourier transforms, the sub-array Fourier transforms can be re-used. Consequently, there are N complex multiplications and N complex additions at each recursive step. (Note that the Danielson-Lanczos procedure, e.g., in Eq. (20) and (22), involves one multiplication and one addition, and is represented by two lines in the figure above.) To compute all the Fourier components, every array element is connected to two (even and odd) sub-array elements at each recursive step. Since there are $\log_2 N$ recursive steps, the number of complex floating-point operations in the FFT algorithm is $2 \log_2 N$.

The program, four1(double data[], unsigned long nn, int isign), in *Numerical Recipes in C* implements the above algorithm. On input, the data[] array contains 2*nn elements that represent nn complex function values, such that data[2*j−1] and data[2*j] (j = 1, ..., nn) are the real and imaginary parts of the function value on the j-th grid point. If isign = 1, four1() performs the Fourier transform,

$$data_j \leftarrow \sum_{m=0}^{N-1} data_m \exp\left(i 2\pi m j / N\right),$$

and, on output, data[] contains the transformed function values. Else if isign = −1, four1() performs a part of inverse Fourier transform,

$$data_m \leftarrow \sum_{j=0}^{N-1} data_j \exp\left(-i2\pi mj/N\right),$$

without dividing the result by nn. To complete the inverse Fourier transform, the caller of the four1() function needs to divide the resulting data[] array by nn.

In your 1D quantum dynamics program, you may define

double psi[2*N],

where psi[2*j] and psi[2*j+1] (j = 0, ..., N−1) are the real and imaginary parts of the wave function on the j-th grid point. Since four1() expects the index to start from 1, instead of 0 in the above psi[] array, we need to call four1() with psi−1 as the first argument. (Note in C, the array name is a pointer to its first element.) The following shows typical calls to four1() in your quantum dynamics program:

```
/* Fourier transform */
four1(psi-1, (unsigned long) N, 1)

/* Inverse Fourier transform */
four1(psi-1, (unsigned long) N, -1)
for (j=0; j< 2*N; j++)
   psi[j] /= N;
```