pyOZ

Theory: Fourier transformation

previous
theory index
next (Ornstein-Zernike equation and closure relations)

In order to solve the Ornstein-Zernike equation, pyOZ transforms it into the Fourier space, where the solution is easier to find (see the section devoted to the equation itself for more details). This text concerns different types of transformations, that are being used in the program. It is by no means complete-consult some specialized text if you need more detailed information.

Fourier transformation

Let's start with the definition of the **Fourier transformation** (FT). In one dimension, the FT of a function f(x) can be written as

$$F(k) = \int_{-\infty}^{\infty} f(x) e^{ikx} dx$$

where x and k are distance and reciprocal distance, respectively, and the function F(k) is called a Fourier transform of f(x). The inverse FT (iFT) reads

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(k) e^{-ikx} dk$$

i.e., we are getting the original function back. The normalization factor $1/2\pi$ can be also distributed evenly among both transforms, yielding unitary transformation pair

$$F(k) = \sqrt{\frac{1}{2\pi}} \int_{-\infty}^{\infty} f(x) e^{ikx} dx$$
$$f(x) = \sqrt{\frac{1}{2\pi}} \int_{-\infty}^{\infty} F(k) e^{-ikx} dk$$

or the whole factor can be used within the forward transformation.

In the case of 3D function f(**r**) where **r** is a position vector, the FT/iFT pair reads

$$F(\mathbf{k}) = \int\int\int_{-\infty}^{\infty} f(\mathbf{r}) e^{i\mathbf{k}\cdot\mathbf{r}} d\mathbf{r}$$
$$f(\mathbf{r}) = \frac{1}{(2\pi)^3} \int\int\int_{-\infty}^{\infty} F(\mathbf{k}) e^{i\mathbf{k}\cdot\mathbf{r}} d\mathbf{k}$$

with **k** being the position vector in the reciprocal space (wave vector). The three integrals are all evaluated over the whole range of the respective coordinate (x, y, z in real and reciprocal space). Please note the scalar product of **k** and **r** in the exponent

and the original normalization constant 1/2π raised to the power of three (the dimensionality of the problem). Unitary FT pair involves the original normalization constant raised to the power of 3/2.

Fourier-Bessel transformation

Let's now assume a function f(**r**), which is, however, dependent only on the length of the vector **r** and not on its orientation, i.e., a spherically symmetric function.

$$f(\mathbf{r}) = f(|\mathbf{r}|) = f(r)$$

By converting the problem into spherical coordinates, we obtain

$$F(\mathbf{k}) = F(k,\theta,\phi) = \int_0^{\infty}\int_0^{\pi}\int_0^{2\pi} e^{i\mathbf{k}\cdot\mathbf{r}} f(r) r^2 dr \sin\theta d\theta d\phi$$

Let's also assume that the angle between vectors **r** and **k** is equal to ϑ. Then, the scalar product in the exponential can be replaced by

$$\mathbf{k}\cdot\mathbf{r} = kr\cos\theta$$

with k and r being now **scalar** quantities. We can integrate over the angle variable φ and upon introducing a substitution

$$x = \cos\theta, dx = -\sin\theta d\theta$$

with integration limits changed from 0,π to 1,-1 (we swap the limits to get -1,1 and at the same time remove the minus sign in front of the sinus function in dx) we arrive at expression

$$F(k) = 2\pi \int_0^{\infty} f(r) r^2 \left[\int_{-1}^{1} e^{ikrx} dx\right] dr = 2\pi \int_0^{\infty} f(r) r^2 \left[\frac{e^{ikr} - e^{-ikr}}{ikr}\right] dr$$

which can be easily converted to the final result

$$F(k) = 2\pi \int_0^{\infty} f(r) r^2 \frac{2\sin(kr)}{kr} dr = 4\pi \int_0^{\infty} f(r) r^2 \frac{\sin(kr)}{kr} dr$$

The resulting expression is the so called **Fourier-Bessel transformation** (FBT) of the spherically symmetric function f(r). The name comes from the fact, that the spherical Bessel function sin(kr)/kr is involved. This procedure decreases the complexity of the problem from 3-dimensional to one-dimensional.

The same argument and slightly modified procedure (changing the sign in the exponential factor from plus to minus) leads to the expression for the inverse Fourier-Bessel transformation

$$f(r) = \frac{1}{(2\pi)^3} 4\pi \int_0^\infty F(k) k^2 \frac{\sin(kr)}{kr} dk$$

where the extra factor is the normalization factor of the Fourier transformation itself. In case of an unitary pair, the factor is the same for both forward and reverse transformation, and it is $\sqrt{2/\pi}$.

(Fourier-)Sine transformation

The **Fourier-sine transformation** (FST) can be derived from the general formula for the Fourier transformation which is applied to odd function. The exponential factor can be written as a sum of sine and cosine terms. Due to symmetry properties, the cosine term vanishes. The sine transformation is than defined as

$$F(k) = \int_0^\infty f(r) \sin(kr) dr$$
$$f(r) = 2/\pi \int_0^\infty F(k) \sin(kr) dk$$

or as a unitary transform pair with the normalization constant distributed among both forward and inverse transformaion

$$F(k) = \sqrt{2/\pi} \int_0^\infty f(r) \sin(kr) dr$$
$$f(r) = \sqrt{2/\pi} \int_0^\infty F(k) \sin(kr) dk$$

It can easily be seen, that there is a close relationship between the Fourier-Bessel and sine transformations. Namely

$$\mathrm{FBT}(f(r)) = N_{FB} \cdot 4\pi \cdot N_{FS}^{-1} \cdot \mathrm{FST}(f(r) \cdot r)/k$$
$$\mathrm{iFBT}(F(k)) = N_{iFB} \cdot 4\pi \cdot N_{iFS}^{-1} \cdot \mathrm{iFST}(F(k) \cdot k)/r$$

where ‘i’ indicates inverse transformation and Nxx are the respective normalization constants. The reason why the sine transformation is discussed here is, that it can be easily calculated in the computer using the Fast Fourier Transform algorithms (see below for more details).

Convolution theorem

One important property of the Fourier transformation that is used in pyOZ is described by the co called **convolution theorem**. The convolution (f*g)(t) is an integral of product of two functions f(t) and g(t), one of whose was firstly reversed:

$$(f * g)(t) = \int_{-\infty}^\infty f(\theta) g(t - \theta) d\theta$$

The convolution theorem states, that

$$\mathrm{FT}((f * g)(t)) = N \cdot \mathrm{FT}(f) \cdot \mathrm{FT}(g)$$

The constant normalization factor N is the reciprocal of the normalization factor used for the forward Fourier transformation.

Discrete Fourier transformation

All information written on this page so far concerned the transformations of continuous functions. This part of the text concerns the situation where the data is only sampled at discrete intervals (which is the case of pyOZ).

When the continuous Fourier transform is rewritten by means of sums instead of integrals, the following expression emerges

$$F(k_m) = \Delta r \sum_{n=0}^{N-1} f_n e^{\frac{2\pi i}{N} mn} = \Delta r F_m$$

F_m is called the **discrete Fourier transform** (DFT) of a set of numbers $f(r_n)$. This transform does not depend on any dimensional parameter (such as the step size Δr). To get an equivalent of the continuous FT of the function f(r) when it is sampled at an interval Δr, it is necessary to multiply every member of the resulting DFT set with this interval.

For the inverse transformation (iDFT) f_n, similar expression can be written

$$f(r_n) = \Delta k \frac{1}{N} \sum_{m=0}^{N-1} F_m e^{-\frac{2\pi i}{N} mn} = \Delta k f_n$$

which also contains the normalization factor, given by the reciprocal of the number of sampling points. Also here, when one wants to have an equivalent of the continuous iFT of the function F(k) sampled at discrete interval Δk, it is necessary to multiply every member of the resulting iDFT set with this interval.

There are three things to note here. Firstly, the step size in the Fourier space is given by

$$\Delta k = \frac{2\pi}{N \Delta r}$$

Secondly, the DFT/iDFT pair is normalized, therefore iDFT(DFT(f)) = f. When the transforms are multiplied with the sampling intervals, however, the normalization is lost, since

$$x = \mathrm{iDFT}(\mathrm{DFT}(f)\Delta r)\Delta k = f \Delta r \Delta k = f \frac{2\pi}{N}$$

Therefore, the resulting set of function samples has to be divided by the extra factor in order to get the original values back.

Thirdly, due to sampling, only the values of the F(k) for n ≤ N/2 agree with the Fourier transformation of the function f(r). This is a consequence of the fact, that in order to define a function through its samples, at least 2 points are needed (the sampling

theorem). Therefore, any signal sampled with “frequency” f = N/r provides information about the spectrum for frequencies smaller than N/2r = f/2 (the so called Nyquist frequency).

Discrete Fourier transform routines in the scientific libraries are **usually** defined so, that the normalization is taken care of automatically. It is, therefore, necessary to carefully check how exactly are the library routines implemented (by using a function where the analytical FT is known and comparing with the data coming from the DFT).

Discrete sine transformation

The last issue to be discussed here is the so called **discrete sine transformation**, due to its specific properties. Both forward and inverse transformations are the same (just exchange the positions of f and F)

$$F_m = \sum_{n=0}^{N-1} f_n \sin(\frac{\pi nm}{N})$$

which resembles the imaginary part of the DFT (recall that an exponential can be written as a sum of a sine and a cosine term) up to the factor of 2 in the nominator. Due to the requirement of an odd function, the first sample has to be zero. This transformation can be quite efficiently evaluated using the Fast Fourier Transform technique.

Firstly, the function is extended beyond its last sample (to the total number of 2N samples). This is actually easy to do, since we can use its odd property, so

$$f_{2N-n} = -f_n$$

for j in the interval 0, N-1 (the original sampling range). This function is then Fourier-transformed.

$$F_m = \sum_{n=0}^{2N-1} f_n e^{\frac{2\pi inm}{2N}}$$

The sum can be splitted in half. The terms for n ≥ N yield then (using a substitution n = 2N-q)

$$\sum_{n=N}^{2N-1} f_n e^{\frac{2\pi inm}{2N}} = \sum_{q=1}^{N} f_{2N-q} e^{\frac{2\pi i(2N-q)m}{2N}} = -\sum_{n=0}^{N-1} f_n e^{-\frac{2\pi inm}{2N}}$$

Putting the two parts of the sum back together yields

$$F_m = \sum_{n=0}^{N-1} f_n e^{\frac{2\pi inm}{2N}} - e^{-\frac{2\pi inm}{2N}} = 2i \sum_{n=0}^{N-1} f_n \sin\left(\frac{\pi nm}{N}\right)$$

As can be seen here, this is (up to a factor of 2i) the expression for the DST. Despite the fact that this procedure introduces a factor of 2 inefficiency, the FFT routines are usually very well optimized, and can save a considerable amount of time.

Due to the fact that 2N points are actually used, the sampling interval in the Fourier space is changed to

$$\Delta k = \frac{2\pi}{2N\Delta r} = \frac{\pi}{N\Delta r}$$

When working with discrete sine transform, this fact must be taken care of.

---

previous
theory index
next (Ornstein-Zernike equation and closure relations)