Fractional Fourier Transform

Implementation details

Giuliano Graziani and Pietro Rossi*

April 4, 2019

*Prometeia SpA, Bologna

1 Fractional Fourier Transform

1.1 The Fractional Fourier Transform

Our task is to compute an infinite sum of the type:

$$p(x) = \frac{1}{2X_c} \sum_{n=-\infty}^{+\infty} \hat{p}_n e^{-i2\pi nx/2X_c} \quad 0 \le x \le 2X_c, \tag{1}$$

The p_N approximation to $p(x)$ is given by:

$$p_N(x) = \frac{1}{2X_c} \sum_{n=-N/2}^{N/2} \hat{p}_n e^{-i2\pi nx/2X_c}. \tag{2}$$

If we confine our interest to the discrete set of values:

$$x_m = m\frac{2X_c}{N}, \quad -N/2 \le m < N/2$$

we get:

$$\begin{aligned}
p_N(x_m) &= \frac{1}{2X_c} \sum_{n=-N/2}^{N/2} \hat{p}_n e^{-i2\pi nm/N} \\
&= \frac{1}{2X_c} \sum_{n=0}^{N/2} \hat{p}_n e^{-i2\pi nm/N} + \frac{1}{2X_c} \sum_{n=-N/2}^{-1} \hat{p}_n e^{-i2\pi nm/N} \\
&= \frac{1}{2X_c} \sum_{n=0}^{N/2} \hat{p}_n e^{-i2\pi nm/N} + \frac{1}{2X_c} \sum_{n=-N/2}^{-1} \hat{p}_n e^{-i2\pi (n+N)m/N} \\
&= \frac{1}{2X_c} \sum_{n=0}^{N/2} \hat{p}_n e^{-i2\pi nm/N} + \frac{1}{2X_c} \sum_{N/2}^{N-1} \hat{p}_{n-N} e^{-i2\pi nm/N} \\
&= \frac{1}{2X_c} \sum_{n=0}^{N-1} \hat{q}_n e^{-i2\pi nm/N} \qquad (3) \\
q_n &= \begin{cases} p_n & 0 \le n < N/2, \\ p_{N/2} + p_{-N/2} & n = N/2, \\ p_{n-N} & N/2 < n < N. \end{cases}
\end{aligned}$$

we can compute very efficiently the N-number $p_N(x_m)$, performing the N-sums in eq.(3) using the FFT. The convenience of the FFT brings along some rigidity. Namely the highest resolution we can get is given by:

$$\delta x = \frac{2X_c}{N}, \tag{4}$$

and this sometimes is just too coarse. We always have the option to increase the number N of Fourier modes, but this has a cost. The alternative, that should always be weighted with care, is to resort to the fractional Fourier transform.
Let's decide that the spacing we want for the set x_m is given by:

$$\delta \hat{x} = \epsilon \delta x, \quad 0 < \epsilon \le 1, \tag{5}$$

where ϵ is the percentage of the origina interval $[-X_c, X_c]$ that we are going to represent, and the set of points we will compute will be:

$$X_m : \{\hat{x}_m = m\delta\hat{x}\}, \quad -\epsilon X_c \leq \hat{x}m \leq \epsilon X_c.$$

To achieve this we would need to compute:

$$p_N(\hat{x}_m) = \frac{1}{2X_c} \sum_{n=-N/2}^{N/2-1} \hat{p}_n e^{-i2\pi nm\epsilon/N}. \tag{6}$$

In the following we will show how to compute efficiently the sum (6) for an arbitrary real value $\eta := \epsilon/N$.
Rewriting

$$2nm \quad \text{as} \quad n^2 + m^2 - (n-m)^2$$

eq. (6) can be written as:

$$p_N(\hat{x}_m)e^{i\pi m^2\eta} \quad = \quad \frac{1}{2X_c} \sum_{n=-N/2}^{N/2} e^{i\pi(n-m)^2\eta - i\pi n^2\eta}\, \hat{p}_n.$$

or, which is the same:

$$p_N(\hat{x}_{j-N/2})e^{i\pi(j-N/2)^2\eta} \quad = \quad \frac{1}{2X_c} \sum_{l=0}^{N-1} e^{i\pi(l-j)^2\eta - i\pi(l-N/2)^2\eta}\, \hat{p}_{l-N/2}.$$

If we define:

$$\begin{aligned} f_j &:= p_N(\hat{x}_{j-N/2})e^{i\pi(j-N/2)^2\eta}, \quad 0 \leq j < N \\ q_l &:= \frac{1}{2X_c} e^{-i\pi(l-N/2)^2\eta}\, \hat{p}_{l-N/2}, \quad 0 \leq l < N \\ T_{jl} &:= e^{i\pi(l-j)^2\eta} \end{aligned}$$

Then: in matrix notation we have:

$$\mathbf{f} = \mathbf{Tq}.$$

The matrix $\mathbf{T}$ has a peculiar form, it is in fact only a function of the difference between the two indices:

$$T_{nm} = T(n-m)$$

and such a matrix is a well known object in the computational literature, it is known as a Toeplitz matrix. We will take now a brief detour in the world of Toeplitz matrices.

1.2 Circular Matrix

Before we tackle Toeplitz matrices we must describe an other special kind of matrix, that is a circular matrix. A circular matrix $\mathbf{C}$ is a matrix of the form:

$$\mathbf{C} = \begin{pmatrix} c_0 & c_{N-1} & c_{N-2} & \cdots & c_1 \\ c_1 & c_0 & c_{N-1} & \cdots & c_2 \\ & & \cdots & & \\ c_{N-1} & c_{N-2} & \cdots & & c_0 \end{pmatrix}$$

The matrix $\mathbf{C}$ is fully specified by its first column

$$\mathbf{c} = \begin{pmatrix} c_0 \\ c_1 \\ \cdots \\ c_{N-1} \end{pmatrix}$$

and the generic element C_{ij} can be written in the form

$$C_{ij} = g(\, i - j\,|_N\,)$$

where $i - j\,|_N$ selects the value of $0 \leq i - j < N$ with the usual mod N arithmetics.

Theorem 1.1. *The N functions f_n defined by:*

$$f_n(j) := \exp\left(\frac{i2\pi nj}{N}\right)$$

are eigenfunctions of any circular matrix $\mathbf{C}$.
Eigenvalues are given by:

$$\lambda_n = \sum_{j=0}^{N-1} \mathbf{c}_j f_n^\dagger(j)$$

Proof.

$$\begin{aligned}
\sum_{j=0}^{N-1} C_{ij} f_n(j) &= \sum_{j=0}^{N-1} g(i - j\,|_N) f_n(j) \\
&= \sum_{j=i}^{i-N+1} g(j) f_n(i-j) \\
&= f(i) \sum_{j=i}^{i-N+1} g(j) f_n^\dagger(j) \\
&= f(i) \sum_{j=0}^{N-1} g(j) f_n^\dagger(j) = \lambda_n f_n(i).
\end{aligned}$$

□

Since:

$$\sum_{n=0}^{N-1} f_n(i) f_n^\dagger(j) = N\delta_{ij}$$

we have:

$$C_{ij} = \sum_{n=0}^{N-1} \lambda_n f_n(i) f_n^\dagger(j)$$

1.3 Matrix vector multiplication

We want to compute:

$$u_i = \sum_{j=0}^{N-1} C_{ij} v_j$$

using the decomposition we get:

$$u_i = \sum_{n=0}^{N-1} \lambda_n f_n(i) \sum_{j=0}^{N-1} f_n^\dagger(j) v_j, \tag{7}$$

and this is where FFT comes into play given that we use it to compute efficiently the various pieces of eq (7).
In fact:

$$\begin{aligned}
\lambda_n &= [\overline{\mathcal{F}}\mathbf{c}]_n \\
u_i &= \sum_{n=0}^{N-1} f_n(i)\lambda_n[\overline{\mathcal{F}}v]_n = [\mathcal{F}(\lambda\overline{\mathcal{F}}v)]_i
\end{aligned}$$

with a total computation cost of :

$$3N\log(N) + N.$$

1.4 Toeplitz Matrix

A Toeplitz matrix **T** is a matrix of the form:

$$\mathbf{T} = \begin{pmatrix} t_0 & t_{-1} & t_{-2} & \cdots & t_{-(N-1)} \\ t_1 & t_0 & t_{-1} & \cdots & t_{-(N-2)} \\ & & \cdots & & \\ t_{N-1} & t_{N-2} & \cdots & & t_0 \end{pmatrix}$$

The matrix **T** is fully specified by its first column

$$\mathbf{t}^1 = \begin{pmatrix} t_0 \\ t_1 \\ \cdots \\ t_{N-1} \end{pmatrix}$$

and its first row:

$$\mathbf{t}_1 = \left(t_0, t_{-1}, \cdots, t_{-(N-1)}\right)$$

and the generic element T_{ij} can be written in the form

$$T_{ij} = t(j-i), \quad 0 \le i, j < N$$

1.5 Embedding in a circular matrix

Let's consider a column vector **r** with elements::

$$r_i = \begin{cases} t_i & 0 \le i < N \\ 0 & N \le i < N+Q \\ t_{-[(2N-1+Q)-i]} & N+Q \le i < 2N-1+Q \end{cases} \qquad 2N-1+Q = 2^M,$$

that is:

$$\mathbf{r} = \begin{pmatrix} t_0 \\ t_1 \\ \vdots \\ t_{N-1} \\ 0 \\ \vdots \\ 0 \\ t_{-(N-1)} \\ t_{-(N-2)} \\ \vdots \\ t_{-1} \end{pmatrix} \tag{8}$$

and build the circular matrix $\mathbf{C}(r)$ based on $\mathbf{r}$.
Let's compute $\mathbf{C}(r)_{ij} 0 \leq i, j < N$ that is the top left corner of $\mathbf{C}(r)$.

$$\begin{array}{llcl} i \geq j & \mathbf{C}(r)_{ij} & = & r(i-j) = t(i-j) \\ i < j & \mathbf{C}(r)_{ij} & = & r(i-j) = r(2N-1+Q-(j-i)) \end{array}$$

Clearly:

$$1 \leq j - i \leq N - 1,$$

therefore:

$$N + Q < 2N + Q - 1 - (j - i) \leq 2N + Q - 1$$

and:

$$r(2N-1+Q-(j-i)) = t_{-[2N-1+Q-(2N-1+Q-(j-i))]} = t(i-j)$$

The top left corner is therefore the original Toeplitz Matrix. This result is pretty obvious if we build the circular matrix from the array $\mathbf{r}$ in eq.(8)
If we are interested in computing:

$$\mathbf{z} = \mathbf{T}\mathbf{x}$$

we can compute:

$$\begin{pmatrix} \mathbf{z} \\ \mathbf{u} \end{pmatrix} = \mathbf{C}(\mathbf{r}) \begin{pmatrix} \mathbf{x} \\ \mathbf{0} \end{pmatrix}$$