プログラミング

DFT の演算中には複素数が含まれていますので、C 言語などでプログラミングして DFT 係数を求めるためには

$$\mathrm{C}[k] = \mathrm{A}[k] + j \cdot \mathrm{B}[k]$$

の様に DFT 係数 C[k] を実数成分 A[k] と虚数成分 B[k] に分けて求めると楽です。

そこでオイラー公式を使って DFT の演算中に含まれる複素数を sin と cos の和に分解してみると、A[k] と B[k] は次のようになります。

DFT をプログラミングで求める

$$\mathrm{A}[k] = \frac{1}{\mathrm{N}} \sum_{i=0}^{\mathrm{N}-1} \left\{ f[i] \cdot \cos(k \cdot \frac{2\pi}{\mathrm{N}} \cdot i) \right\} \ , \ (k = 0, 1, 2, \cdots, \mathrm{N}-1)$$

$$\mathrm{B}[k] = -\frac{1}{\mathrm{N}} \sum_{i=0}^{\mathrm{N}-1} \left\{ f[i] \cdot \sin(k \cdot \frac{2\pi}{\mathrm{N}} \cdot i) \right\} \ , \ (k = 0, 1, 2, \cdots, \mathrm{N}-1)$$

$f[i]$ ⋯ 周期 N [点] の周期性ディジタル信号

A[k] ⋯ k 番目の DFT 係数の実数成分、実数の <u>定数</u>、個数は $\mathrm{A}[0], \mathrm{A}[1], \cdots, \mathrm{A}[\mathrm{N}-1]$ の N 個

B[k] ⋯ k 番目の DFT 係数の虚数成分、実数の <u>定数</u>、個数は $\mathrm{B}[0], \mathrm{B}[1], \cdots, \mathrm{B}[\mathrm{N}-1]$ の N 個

このようにすると DFT の演算中に複素数が含まれなくなるので、普通に外側が k、内側が i の 2 重 for ループを作って A[k] と B[k] を求められます。また

$$|\mathrm{C}[k]| = \sqrt{\mathrm{A}^2[k] + \mathrm{B}^2[k]}$$

$$\angle\, \mathrm{C}[k] = \mathrm{atan2}(\mathrm{B}[k], \mathrm{A}[k])$$

より DFT 係数 C[k] の絶対値と偏角も求まります。

同様に IDFT を A[k] と B[k] を使って書き直すと次のようになります。

IDFT をプログラミングで求める

$$f[i] = \sum_{k=0}^{\mathrm{N}-1} \left\{ \mathrm{A}[k] \cdot \cos(k \cdot \frac{2\pi}{\mathrm{N}} \cdot i) - \mathrm{B}[k] \cdot \sin(k \cdot \frac{2\pi}{\mathrm{N}} \cdot i) \right\} \ , \ (i = 0, 1, 2, \cdots, \mathrm{N}-1)$$

$f[i]$ ⋯ 周期 N [点] の周期性ディジタル信号

A[k] ⋯ k 番目の DFT 係数の実数成分、実数の <u>定数</u>、個数は $\mathrm{A}[0], \mathrm{A}[1], \cdots, \mathrm{A}[\mathrm{N}-1]$ の N 個

B[k] ⋯ k 番目の DFT 係数の虚数成分、実数の <u>定数</u>、個数は $\mathrm{B}[0], \mathrm{B}[1], \cdots, \mathrm{B}[\mathrm{N}-1]$ の N 個

この場合は外側が i、内側が k の 2 重 for ループを作って A[k] と B[k] から $f[i]$ を復元できます。