DCT filtering

May 8, 2013

1 1D case

1.1 Decomposing of x

Given a signal $x[n]$, defined for $n = 0, \ldots, N-1$, we can define the discrete cosine transform $Cx[k]$ as

$$Cx[k] = 2\sum_{n=0}^{N-1} x(n)\cos\left(\frac{2\pi\left(n+\frac{1}{2}\right)k}{2N}\right)$$

Note that this corresponds to an unnormalized version of the DCT calculated by the `dct` function in MATLAB. From x, we can define the symmetrized signal $y[n]$ by

$$y[n] = \begin{cases} x[n], & n = 0, \ldots, N-1 \\ x(2N-n-1), & n = N, \ldots, 2N-1 \end{cases},$$

and its discrete Fourier transform $\hat{y}[k]$

$$\hat{y}[k] = \sum_{n=0}^{2N-1} y[n] e^{\frac{-2\pi i n k}{2N}}.$$

One can then derive the following relation between $\hat{y}$ and Cx [1]:

$$\hat{y}[k] = e^{\frac{2\pi i k/2}{2N}} Cx[k], \quad k = 0, \ldots, N-1.$$

Note that since the signal $y[n]$ is symmetric not about $n = 0$, but $n = -1/2$, its Fourier transform is not real, but has a linear phase corresponding to this shift of symmetry axis. We also note that since y is real, the remaining coefficients of $\hat{y}$ can be obtained through conjugate (Hermitian) symmetry

$$\hat{y}[2N-k] = \overline{\hat{y}[k]}, \quad k = 1, \ldots, N-1$$

and the fact that $\hat{y}[N] = 0$.

1.2 Filtering with a Hermitian filter

If we want to calculate the circular convolution of x with a filter h using symmetric boundary conditions, this can be obtained by filtering y with h using periodic boundary conditions, which is thus reduced to multiplication of Fourier transforms. For a general h, this breaks the symmetry of y and required to calculate the convolution for all $n = 0, \ldots, 2N-1$. However, if h possesses symmetry properties, for example conjugate symmetry about $n = 0$, we only need to compute transforms of size N.

Let h be defined on $n = 0, \ldots, 2N-1$ with conjugate symmetry about $n = 0$ (even symmetry in the real part and odd symmetry in the imaginary part). Its Fourier transform $\hat{h}$ is therefore real. Calculating the product

$$\hat{z}[k] = \hat{y}[k]\hat{h}[k]$$

the phase remains unchanged, so the signal z whose Fourier transform is $\hat{z}$ has a conjugate symmetry about $n = -1/2$. Note that in extending $Cx[k]$ into $\hat{y}[k]$, we've multiplied the number of real coefficients by 2 (disregarding the phase, which is fixed). However, since h is a complex filter as opposed to purely real, it is unavoidable that the number of coefficients would double at some point.

We can exploit the fact that $\hat{z}[k]$ is real times a linear phase, reducing its inversion from a complex inverse Fourier transform of size $2N$ to one of size N [2].

Let us divide $\hat{z}$ into its even-numbered coefficients $\hat{z}_0$ and its odd-numbered coefficients $\hat{z}_1$ (shifted in phase):

$$\hat{z}_0[k] = \hat{z}[2k] \qquad k = 0, \ldots, N-1$$
$$\hat{z}_1[k] = e^{\frac{-2\pi i/2}{2N}} \hat{z}[2k+1] \qquad k = 0, \ldots, N-1$$

We can now create a signal $\hat{s}[k]$, defined by

$$\hat{s}[k] = \hat{z}_0[k] + i\hat{z}_1[k].$$

Here, $\hat{s}[k]$ is a complex signal of length N. If we compute its inverse Fourier transform, we obtain

$$s[n] = z_0[n] + iz_1[n],$$

where z_0 and z_1 are the inverse Fourier transforms of $\hat{z}_0$ and $\hat{z}_1$, respectively. Since $\hat{z}_0[k]$ and $\hat{z}_1[k]$ both have the phase $e^{\frac{2\pi ik/2}{N}}$, their inverses $z_0[k]$ and $z_1[k]$ are conjugate symmetric about $n = -1/2$. As a result, the real even and the imaginary odd parts of s belong to z_0 while the real odd and the imaginary even parts of s belong to z_1. Specifically, if we define

$$s_e[n] = \frac{1}{2}\left(s[n] + s[N-1-n]\right)$$
$$s_o[n] = \frac{1}{2}\left(s[n] - s[N-1-n]\right),$$

we obtain

$$z_0[n] = \mathrm{Re}\{s_e\} + i\mathrm{Im}\{s_o\} z_1[n] \quad = \mathrm{Im}\{s_e\} - i\mathrm{Re}\{s_o\}.$$

Inverting the phase shift of z_1, we can now reconstruct z:

$$z[n] = z_0[n] + e^{2\pi in/2N}\left(e^{-\frac{2\pi i/2}{2N}} z_1[n]\right),$$

for $n = 0, \ldots, N-1$. Since z is conjugate symmetric about $n = -1/2$, we don't need to calculate the second half.

1.3 Filtering with an even or odd filter

TODO! Instead of considering an analytic Hermitian filter, we can consider two (orthogonal) filters: the real and imaginary part. The real part is even, so can be decomposed with a DCT, while the imaginary part is odd, so can be decomposed with a DST. Computing the convolution of even and odd functions can be done by multiplying their DCTs/DSTs [3]. The issue is then how to optimize the inversion so that the inverse DCT/DST can be computed without padding. To do this, we should be able to adapt the methods outlined in [1].

References

[1] J. Makhoul, "A fast cosine transform in one and two dimensions," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 1, pp. 27–34, 1980.

[2] E. O. Brigham, *The fast Fourier transform.* Englewood Cliffs, New Jersey, PrenticeçHall, 1974.

[3] S. A. Martucci, "Symmetric convolution and the discrete sine and cosine transforms," *Signal Processing, IEEE Transactions on*, vol. 42, no. 5, pp. 1038–1051, 1994.