Algorithms of Scientific Computing

Discrete Sine Transform (DST)

Michael Bader
Summer Term 2013
# DFT and Symmetry

<table>
<thead>
<tr>
<th>INPUT</th>
<th>TRANSFORM</th>
</tr>
</thead>
<tbody>
<tr>
<td>real symmetry</td>
<td>( f_n \in \mathbb{R} )                     →  Real DFT (RDFT)</td>
</tr>
<tr>
<td>even symmetry</td>
<td>( f_n = f_{-n} )                  →  Discrete Cosine Transform (DCT)</td>
</tr>
<tr>
<td>odd symmetry</td>
<td>( f_n = -f_{-n} )                  →  Discrete Sine Transform (DST)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>“QUARTER-WAVE”</th>
<th>INPUT</th>
<th>TRANSFORM</th>
</tr>
</thead>
<tbody>
<tr>
<td>even symmetry</td>
<td>( f_n = f_{n-1} )                     →  QW-DCT</td>
<td></td>
</tr>
<tr>
<td>odd symmetry</td>
<td>( f_n = -f_{n-1} )                     →  QW-DST</td>
<td></td>
</tr>
</tbody>
</table>
Real-valued Input Data with “Odd” Symmetry

Given: $2N$ input data $f_{-N+1}, \ldots, f_N$, all $f_n \in \mathbb{R}$, with

$$f_{-n} = -f_n,$$

in particular $f_0 = f_N = f_{-N} = 0$

The DFT then has the following form:

$$F_k = \frac{1}{2N} \sum_{n=-N+1}^{N} f_n \omega_{2N}^{-nk}$$

$$= \frac{1}{2N} \left( f_0 \sum_{n=0}^{N-1} \left( f_n \omega_{2N}^{-nk} + f_{-n} \omega_{2N}^{nk} \right) + f_N \left( \omega_{2N}^{-Nk} \right) \right)$$

$$= \frac{1}{2N} \sum_{n=1}^{N-1} f_n \left( \omega_{2N}^{-nk} - \omega_{2N}^{nk} \right) = -i \frac{N-1}{N} \sum_{n=1}^{N-1} f_n \sin \left( \frac{\pi nk}{N} \right).$$
Symmetry in the Coefficients

Transform to $f_n$ with symmetry $f_{-n} = -f_n$ gives:

$$F_k = \frac{-i}{N} \sum_{n=1}^{N-1} f_n \sin \left( \frac{\pi nk}{N} \right) \quad \text{for} \quad k = -N + 1, \ldots, N.$$  

Same symmetry in the coefficients $F_k$:

$$F_{-k} = \frac{-i}{N} \sum_{n=1}^{N-1} f_n \sin \left( \frac{\pi n(-k)}{N} \right) = \frac{-i}{N} \sum_{n=1}^{N-1} f_n \left( - \sin \frac{\pi nk}{N} \right) = -F_k$$

$\Rightarrow$ leads to the same (up to scaling) “discrete sine transform”
Discrete Sine Transform (DST)

From DFT of real-valued, odd symmetric data:

\[ F_k = -\frac{i}{N} \sum_{n=1}^{N-1} f_n \sin \left( \frac{\pi nk}{N} \right), \quad k = 1, \ldots, N - 1. \]

Analogue calculation for IDFT gives:

\[ f_n = 2i \sum_{k=1}^{N-1} F_k \sin \left( \frac{\pi nk}{N} \right), \quad n = 1, \ldots, N - 1. \]

⇒ definition of the discrete sine transform (\( \hat{F}_k := iF_k \)):

\[ \hat{F}_k = \frac{1}{N} \sum_{n=1}^{N-1} f_n \sin \left( \frac{\pi nk}{N} \right), \quad f_n = 2 \sum_{k=1}^{N-1} \hat{F}_k \sin \left( \frac{\pi nk}{N} \right), \]
Computation of the Discrete Sine Transform

Via pre-/postprocessing:

(1) generate $2N$ vector with odd symmetry

\[ x_{-k} = -x_k \quad \text{for } k = 1, \ldots, N - 1 \]
\[ x_0 = x_N = 0 \]

(2) coefficients $X_k$ via fast, real-valued FFT on vector $x$

(3) postprocessing: $\hat{X}_k = -\text{Im}\{X_k\}$ for $k = 1, \ldots, N - 1$.

(4) if necessary: scaling
Summary: Survey on DCT/DST Variants

Symmetry properties $\Leftrightarrow$ how is data continued at boundaries:

<table>
<thead>
<tr>
<th>beg. \ end</th>
<th>even</th>
<th>odd</th>
</tr>
</thead>
<tbody>
<tr>
<td>even</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>odd</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

$\Rightarrow$ 4 possibilities

<table>
<thead>
<tr>
<th>beg. \ end</th>
<th>mirror</th>
<th>copy</th>
</tr>
</thead>
<tbody>
<tr>
<td>mirror</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>copy</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

$\Rightarrow$ 4 possibilities

$\Rightarrow$ in total: 16 possibilities (8 DCT, 8 DST)
Summary: Survey on DCT/DST Variants (2)

Common schemes of DCT (left) and DST (right) (images: Wikipedia):

DCT-I:

DCT-II:

DCT-III:

DCT-IV:

DST-I:

DST-II:

DST-III:

DST-IV:
Application: DCT/DST for PDE (Spectral Methods)

nice application in exercise sessions: DST for Fast Poisson Solver

Space domain \rightarrow \text{Problem} \rightarrow \text{Solution} \rightarrow \text{Frequency domain} \rightarrow \text{DST/DCT}

Attention: limits/problems for using DFT with PDE include
- irregular (i.e. non-rectangular) domains
- variable coefficients in problem

⇒ other methods: FVM, FEM (fast linear solvers, multigrid, etc.)