

# Algorithms of Scientific Computing

## Discrete Sine Transform (DST)

Tobias Neckel, Dirk Pflüger

Summer Term 2011



# DFT and Symmetry

INPUT

TRANSFORM

**real symmetry** $f_n$  is real

→

Real-valued DFT (RDFT)

**even symmetry** $f_n = f_{-n}$ 

→

Discrete Cosine Transform (DCT)

**odd symmetry** $f_n = -f_{-n}$ 

→

Discrete Sine Transform (DST)

## Real-valued Input Data with “Odd” Symmetry

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

$$f_{-n} = -f_n, \quad \text{in particular} \quad f_0 = f_N = f_{-N} = 0$$

The DFT has the following form:

$$\begin{aligned} F_k &= \frac{1}{2N} \sum_{n=-N+1}^N f_n \omega_{2N}^{-nk} \\ &= \frac{1}{2N} \left( \underbrace{f_0}_{=0} + \sum_{n=1}^{N-1} \left( f_n \omega_{2N}^{-nk} + f_{-n} \omega_{2N}^{nk} \right) + \underbrace{f_N}_{=0} \omega_{2N}^{-Nk} \right) \\ &= \frac{1}{2N} \sum_{n=1}^{N-1} f_n \left( \omega_{2N}^{-nk} - \omega_{2N}^{nk} \right) = \frac{-i}{N} \sum_{n=1}^{N-1} f_n \sin \left( \frac{\pi nk}{N} \right). \end{aligned}$$

# Symmetry in the Coefficients

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

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

Same symmetrie in the coefficients  $F_k$ :

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

⇒ leads to the same (up to scaling) sine transform

# Discrete Sine Transform (DST)

From DFT of real-valued, odd symmetric data:

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

Analogue calculation for IDFT gives:

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

⇒ 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

$$x_{-k} = -x_k \quad \text{for } k = 1, \dots, N-1$$

$$x_0 = x_N = 0$$

- (2) coefficients  $X_k$  via fast, real-valued FFT on vector  $x$
- (3) postprocessing:  $\widehat{X}_k = -\text{Im}\{X_k\}$  for  $k = 1, \dots, N-1$ .
- (4) if necessary: scaling

# Summary: Survey on DCT/DST Variants

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

beg. \ end	even	odd
even	x	x
odd	x	x

$\Rightarrow$  4 possibilities

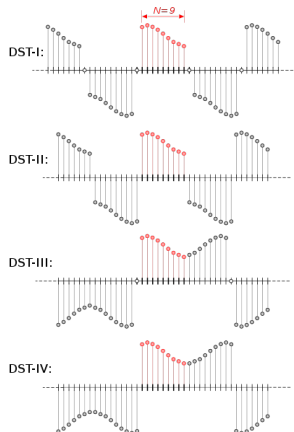
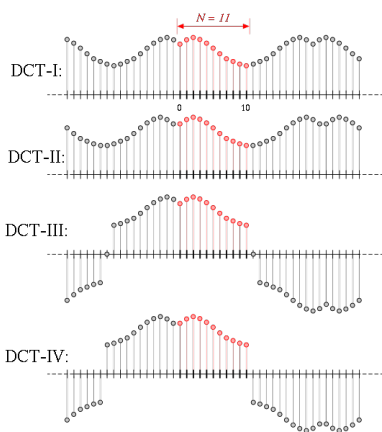
beg. \ end	mirror	copy
mirror	x	x
copy	x	x

$\Rightarrow$  4 possibilities

$\Rightarrow$  **total:** 16 possibilities (8 DCT, 8 DST)

# Summary: Survey on DCT/DST Variants (2)

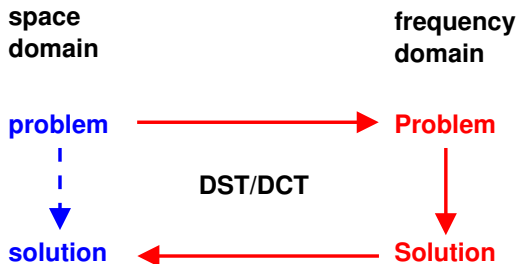
Common schemes of DCT (left) and DST (right) (source: wikipedia):





# Application: DCT/DST for PDE (Spectral Methods)

nice application in exercise sessions: DST for Fast Poisson Solver



**Attention:** also limits/problems for using DFT with PDE:

- irregular (i.e. non-rectangular) domains
  - variable coefficients in problem
- ⇒ other methods: FVM, FEM (fast linear solvers, multigrid, etc.)