# MATH 408: Computational Methods for Differential Equations

## Periodic Grids: The Discrete FT and Fast FT

Greg Fasshauer

Department of Applied Mathematics & Statistics
Colorado School of Mines

Fall 2021

## Outline

## Periodic Grids: The DFT and FFT

We now consider the case of a bounded grid with periodic data and will explain how to find the entries in the $m \times m$ dense matrix $\mathsf{D}_m$ from slide #19 in `Notes408_15_HigherOrderMethods2ptBVPs`, i.e.,

$$\mathsf{D}_m = \begin{bmatrix} 0 & & & & & -\frac{1}{2}\cot\frac{1h}{2} \\ -\frac{1}{2}\cot\frac{1h}{2} & \ddots & & \ddots & & \frac{1}{2}\cot\frac{2h}{2} \\ \frac{1}{2}\cot\frac{2h}{2} & & \ddots & & & -\frac{1}{2}\cot\frac{3h}{2} \\ -\frac{1}{2}\cot\frac{3h}{2} & & & \ddots & & \vdots \\ \vdots & & \ddots & & \ddots & \frac{1}{2}\cot\frac{1h}{2} \\ \frac{1}{2}\cot\frac{1h}{2} & & & & & 0 \end{bmatrix}.$$

For simplicity we consider the interval $[0, 2\pi]$ only, and assume we are given $m$ (even[1]) uniformly spaced grid points $x_j = jh$, $j = 1, \ldots, m$, with $h = \frac{2\pi}{m}$.

[1] Formulas for odd $m$ also exist, but are slightly different.

## The Discrete Fourier Transform (DFT)

Let's look at the Fourier transform of the discrete and periodic data $\boldsymbol{u} = [u_1, \ldots, u_m]^T$ with

$$u_j = u(x_j) = u(jh) = u(j\frac{2\pi}{m}), \quad j = 1, \ldots, m.$$

Aliasing again ensures that the Fourier domain will be bounded.
Periodicity of the data implies that the Fourier domain is also discrete (since only waves $e^{ikx}$ with integer wavenumber $k$ have period $2\pi$).
Thus, the discrete Fourier transform (DFT) is given by

$$\hat{u}_k = h \sum_{j=1}^{m} e^{-ikx_j} u_j, \quad k = -\frac{m}{2} + 1, \ldots, \frac{m}{2}. \tag{1}$$

### Remark

*The (continuous) Fourier domain $[-\pi/h, \pi/h]$ used earlier now translates to the discrete domain $\{-\frac{m}{2} + 1, \ldots, \frac{m}{2}\}$ since $h = \frac{2\pi}{m}$ is equivalent to $\pi/h = m/2$.*

## Variants of the FT used thus far

1. Fourier transform, $u \in L_2(\mathbb{R})$, arbitrary

$$\hat{u}(\xi) = \int_{-\infty}^{\infty} \mathrm{e}^{-\mathrm{i}\xi x} u(x)\mathrm{d}x, \quad \xi \in \mathbb{R}$$

Physical space: $x \in \mathbb{R}$, unbounded & continuous
Fourier space: $\xi \in \mathbb{R}$, unbounded & continuous

2. Semi-discrete Fourier transform, $\boldsymbol{u} \in \ell_2(\mathbb{Z})$, arbitrary

$$\hat{u}(\xi) = h \sum_{j=-\infty}^{\infty} \mathrm{e}^{-\mathrm{i}\xi x_j} u_j, \quad \xi \in [-\frac{\pi}{h}, \frac{\pi}{h}]$$

Physical space: $x_j \in h\mathbb{Z}$, unbounded & discrete
Fourier space: $\xi \in [-\frac{\pi}{h}, \frac{\pi}{h}]$, bounded & continuous

3. Discrete Fourier Transform, $\boldsymbol{u}$ $2\pi$-periodic

$$\hat{u}_k = h \sum_{j=1}^{m} \mathrm{e}^{-\mathrm{i}k x_j} u_j, \quad k = -\frac{m}{2} + 1, \ldots, \frac{m}{2}.$$

Physical space: $x_j \in \{h, 2h, \ldots, 2\pi - h, 2\pi\}$, bounded & discrete
Fourier space: $k \in \{-\frac{m}{2} + 1, \ldots, \frac{m}{2}\}$, bounded & discrete

## The Inverse Discrete Fourier Transform (iDFT)

The inverse discrete Fourier transform (iDFT) is defined by

$$u_j = \frac{1}{2\pi} \sum_{k=-m/2+1}^{m/2} e^{ikx_j} \hat{u}_k, \quad j = 1, \ldots, m. \tag{2}$$

The spectral derivative of a finite periodic data vector $\boldsymbol{u}$ is obtained by the same procedure as before:

- Define the band-limited interpolant in terms of the iDFT.
- Represent $\boldsymbol{u}$ as a linear combination of shifts of periodic delta functions (details skipped).
- Show that band-limited interpolants for the periodic delta functions are periodic sinc functions.
- Get the band-limited interpolant for an arbitrary periodic data function in terms of periodic sinc functions.
- Differentiate periodic sinc functions to get entries of $\mathsf{D}_m$.

The band-limited interpolant of the data is given by filling in the iDFT of the data

$$p(x) = \frac{1}{2\pi} {\sum_{k=-m/2}^{m/2}}' e^{ikx} \hat{u}_k, \quad x \in [0, 2\pi]. \tag{3}$$

To ensure the band-limited interpolant works properly we define $\hat{u}_{-m/2} = \hat{u}_{m/2}$, and the prime on the sum indicates that we add the first and last terms in the sum with weight $\frac{1}{2}$.

### Remark

*The band-limited interpolant is actually a trigonometric polynomial of degree $\frac{m}{2}$, i.e., $p(x)$ can be written as a linear combination of*

$$1, \sin(x), \cos(x), \sin(2x), \cos(2x), \ldots, \sin(\frac{m}{2}x), \cos(\frac{m}{2}x).$$

*We will come back to this fact when we discuss non-periodic data.*

Next, we represent an arbitrary periodic data vector $\boldsymbol{u}$ as a linear combination of shifts of periodic delta functions.
We omit the details here (they can be found in [3]).

The band-limited interpolant of the periodic delta function ends up being

$$p(x) = S_m(x) = \frac{\sin(\pi x/h)}{(2\pi/h)\tan(x/2)},$$

which is known as the periodic sinc function $S_m$.

Now, just as in the infinite case (see slide #22 in `Notes408_16_UnboundedGrids`), the band-limited interpolant for an arbitrary periodic data function can be written as

$$p(x) = \sum_{k=1}^{m} u_k S_m(x - x_k).$$

Finally, using the same arguments and similar elementary calculations as earlier, we get the derivative of the periodic sinc function

$$S_m'(x_j) = \begin{cases} 0, & j \equiv 0 \pmod{m}, \\ \frac{1}{2}(-1)^j \cot(\frac{jh}{2}), & j \not\equiv 0 \pmod{m}. \end{cases}$$

Since

$$p'(x) = \sum_{k=1}^{m} u_k S_m'(x - x_k).$$

and

$$u_j' = p'(x_j) = \sum_{k=1}^{m} u_k S_m'(x_j - x_k)$$

the $S_m'(x_j)$ are the entries of the $m$th column of the Toeplitz matrix $\mathsf{D}_m$.

**Example**

The MATLAB script `SpectralDiffDemo.m` illustrates the use of spectral differentiation for the not so smooth hat function and for the infinitely smooth function $u(x) = e^{\sin x}$.

## Using the Fast Fourier Transform (FFT)

The most efficient computational approach is to view spectral differentiation in the Fourier domain (Procedure #2 earlier) and then implement the DFT via the fast Fourier transform (FFT).
The general outline is as follows:

1. Sample the function $u$ at the (finite set of) discrete points $x_j$, $j = 1, \ldots, m$ to obtain the data vector $\boldsymbol{u}$ with components $u_j$.
2. Compute the discrete Fourier transform of the (finite) data vector via (1):
$$\hat{u}_k = h \sum_{j=1}^{m} e^{-ikx_j} u_j, \qquad k = -\frac{m}{2} + 1, \ldots, \frac{m}{2}.$$
3. Compute the Fourier transform of the derivative as in (6) of `Notes408_16_UnboundedGrids`, i.e.,
$$\widehat{u'}_k = \begin{cases} 0, & k = m/2, \\ ik\hat{u}_k, & \text{otherwise.} \end{cases}$$
4. Find the derivative vector via iDFT (see (2)), i.e.,
$$u'_j = \frac{1}{2\pi} \sum_{k=-m/2+1}^{m/2} e^{ikx_j} \widehat{u'}_k, \quad j = 1, \ldots, m.$$

## Remark

- *Cooley and Tukey (1965) are usually given credit for discovering the FFT. However, the same algorithm was already known to Gauss in 1805 (even before Fourier completed his work on what is known today as the Fourier transform in 1822) (see [1]).*
- *A detailed discussion of the FFT algorithm goes beyond the scope of this course. We use the MATLAB implementations `fft` and `ifft` which are based on the current state-of-the-art FFTW algorithm (the "fastest Fourier transform in the West") developed by Matteo Frigo and Steven G. Johnson at MIT in 1998.*

## Example

The MATLAB script `SpectralDiffFFTDemo.m` is an FFT version of the earlier script `SpectralDiffDemo.m`. The FFT implementation is considerably faster than the implementation based on differentiation matrices. The FFT requires roughly $\mathcal{O}(m \log m)$ operations, while the differentiation matrix approach is $\mathcal{O}(m^2)$.

## References I

M. T. Heideman, et al.
Gauss and the History of the Fast Fourier Transform.
*Archive for History of Exact Sciences*, Vol. 34, no. 3, 1985, 265–277.

R. LeVeque.
Finite Difference Methods for Ordinary and Partial Differential Equations.
SIAM, Philadelphia, 2007.

L. N. Trefethen.
Spectral Methods in MATLAB.
SIAM, Philadelphia, 2000.