Numerical Methods II
(Pseudo)Spectral Methods for PDEs

Aleksandar Donev
Courant Institute, NYU[1]
donev@courant.nyu.edu

[1]MATH-GA.2020-001 / CSCI-GA.2421-001, Spring 2019

Feb 5th, 2019

Outline

Convolutions using FFT

Filtering using FFTs

- Because FFT is a very fast, almost linear algorithm, it is used often to accomplish tasks in data processing, e.g., **noise filtering** (see example in previous lecture), computing **(auto)correlation** functions, etc.
- Denote the (continuous or discrete) Fourier transform with

$$\hat{\mathbf{f}} = \mathcal{F}(\mathbf{f}) \text{ and } \mathbf{f} = \mathcal{F}^{-1}\left(\hat{\mathbf{f}}\right).$$

- Plain FFT is used in signal processing for **digital filtering** (**low-pass**, **high-pass**, or **band-pass filters**)
- How to do it: Multiply the spectrum by a filter $\hat{S}(k)$ discretized as $\hat{\mathbf{s}} = \left\{\hat{S}(k)\right\}_k$:

$$\mathbf{f}_{\text{filtered}} = \mathcal{F}^{-1}\left(\hat{\mathbf{s}} \boxdot \hat{\mathbf{f}}\right) = \mathbf{f} \circledast \mathbf{s},$$

where $\boxdot$ denotes element-wise product, and $\circledast$ denotes convolution.

Convolution

- For continuous function, an important type of operation found in practice is **convolution** (smoothing) of a (periodic) function $f(x)$ with a (periodic) **kernel** $K(x)$:

$$(K \circledast f)(x) = \int_0^{2\pi} f(y)K(x-y)dy.$$

- It is not hard to prove the **convolution theorem**:

$$\mathcal{F}(K \circledast f) = \mathcal{F}(K) \boxdot \mathcal{F}(f).$$

- Importantly, this remains true for **discrete convolutions**:

$$(\mathbf{K} \circledast \mathbf{f})_j = \frac{1}{N} \sum_{j'=0}^{N-1} f_{j'} K_{j-j'} \quad \Rightarrow$$

$$\mathcal{F}(\mathbf{K} \circledast \mathbf{f}) = \mathcal{F}(\mathbf{K}) \boxdot \mathcal{F}(\mathbf{f}) \quad \Rightarrow \quad \mathbf{K} \circledast \mathbf{f} = \mathcal{F}^{-1}(\mathcal{F}(\mathbf{K}) \boxdot \mathcal{F}(\mathbf{f}))$$

Proof of Discrete Convolution Theorem

Assume that the normalization used is a factor of N^{-1} in the forward and no factor in the inverse DFT:

$$f_j = \sum_{k=0}^{N-1} \hat{f}_k \exp\left(\frac{2\pi ijk}{N}\right), \text{ and } \quad \hat{f}_k = \frac{1}{N}\sum_{j=0}^{N-1} f_j \exp\left(-\frac{2\pi ijk}{N}\right)$$

$$\left[\mathcal{F}^{-1}\left(\mathcal{F}\left(\mathbf{K}\right) \boxdot \mathcal{F}\left(\mathbf{f}\right)\right)\right]_k = \sum_{k=0}^{N-1} \hat{f}_k \hat{K}_k \exp\left(\frac{2\pi ijk}{N}\right) =$$

$$N^{-2}\sum_{k=0}^{N-1}\left(\sum_{l=0}^{N-1} f_l \exp\left(-\frac{2\pi ilk}{N}\right)\right)\left(\sum_{m=0}^{N-1} K_m \exp\left(-\frac{2\pi imk}{N}\right)\right)\exp\left(\frac{2\pi ijk}{N}\right)$$

$$= N^{-2}\sum_{l=0}^{N-1} f_l \sum_{m=0}^{N-1} K_m \sum_{k=0}^{N-1} \exp\left[\frac{2\pi i\left(j-l-m\right)k}{N}\right]$$

contd.

Recall the key discrete orthogonality property

$$\forall \Delta k \in \mathbb{Z}: \quad N^{-1} \sum_j \exp\left[i \frac{2\pi}{N} j \Delta k\right] = \delta_{\Delta k} \quad \Rightarrow$$

$$N^{-2} \sum_{l=0}^{N-1} f_l \sum_{m=0}^{N-1} K_m \sum_{k=0}^{N-1} \exp\left[\frac{2\pi i \, (j-l-m)\, k}{N}\right] = N^{-1} \sum_{l=0}^{N-1} f_l \sum_{m=0}^{N-1} K_m \delta_{j-l-m}$$

$$= N^{-1} \sum_{l=0}^{N-1} f_l K_{j-l} = (\mathbf{K} \circledast \mathbf{f})_j$$

Computing convolutions requires 2 forward FFTs, one element-wise product, and one inverse FFT, for a total cost $N \log N$ instead of N^2.

We can use this to solve **periodic integro-differential equations** involving convolutions, for example (recall that trapezoidal rule for the convolution is spectrally accurate for analytic functions)!

Spectral Differentiation

Spectral Derivative

- Consider approximating the derivative of a periodic function $f(x)$, computed at a set of N equally-spaced nodes, $\mathbf{f}$.
- We can differentiate the spectral approximation: **Spectral derivative**

$$f'(x) \approx \phi'(x) = \frac{d}{dx}\phi(x) = \frac{d}{dx}\left(\sum_{k=0}^{N-1} \hat{f}_k e^{ikx}\right) = \sum_{k=0}^{N-1} \hat{f}_k \frac{d}{dx} e^{ikx}$$

$$= \sum_{k=0}^{N-1} \left(ik\hat{f}_k\right) e^{ikx} = \sum_{k=0}^{N-1} \widehat{(\phi')}_k e^{ikx} \quad \Rightarrow$$

$$\widehat{(\phi')}_k = ik\hat{f}_k \quad \Rightarrow \quad \phi' = \mathcal{F}^{-1}\left(i\mathbf{k} \boxdot \hat{\mathbf{f}}\right)$$

- **Differentiation, like convolution, becomes multiplication in Fourier space.**
  Indeed $-\int f(y)\delta'(x-y)\,dy = \int f'(y)\delta(x-y)\,dy = f'(x)$.

Unmatched mode

- Recall that for even N there is one unmatched mode, the one with the highest frequency and amplitude $\hat{f}_{N/2}$.
- We need to choose what we want to do with that mode; see notes by S. G. Johnson (MIT) linked on webpage for details:

$$\phi(x) = \hat{f}_0 + \sum_{0<k<N/2} \left(\hat{f}_k e^{ikx} + \hat{f}_{N-k} e^{-ikx} \right) + \hat{f}_{N/2} \cos\left(\frac{Nx}{2}\right).$$

  This is the unique "**minimal oscillation**" trigonometric interpolant.
- Differentiating this we get

$$\widehat{(\phi')}_k = \hat{f}_k \begin{cases} 0 & \text{if } k = N/2 \\ ik & \text{if } k < N/2 \\ i\,(k-N) & \text{if } k > N/2 \end{cases}.$$

- Real valued interpolation samples result in **real-valued** $\phi(x)$ for all x.

FFT-based differentiation

```
% From Nick Trefethen's Spectral Methods book
% Differentiation of exp(sin(x)) on (0,2*pi]:
  N = 8; % Even number!
  h = 2*pi/N;  x = h*(1:N)';
  v = exp(sin(x));  vprime = cos(x).*v;
  v_hat = fft(v);
  ik = 1i*[0:N/2-1 0 -N/2+1:-1]'; % Zero special mode
  w_hat = ik .* v_hat;
  w = real(ifft(w_hat));
  error = norm(w-vprime,inf)
```

Differentiation matrices

- Writing $g = f'$ we can denote this in matrix notation $\hat{\mathbf{g}} = \widehat{\mathbf{D}}_1 \hat{\mathbf{f}}$, where $\widehat{\mathbf{D}}_1$ is a **diagonal differentiation matrix** with *ik* on its diagonal (why does it have to be a matrix?).
- Observe that $\widehat{\mathbf{D}}_1^{\star} = -\widehat{\mathbf{D}}_1$ (anti-Hermitian).
- In **real space** $\mathbf{g} = \mathbf{D}\mathbf{f}$and in **Fourier space** $\hat{\mathbf{g}} = \widehat{\mathbf{D}}\hat{\mathbf{f}}$, related by

$$\mathbf{D} = \mathbf{F}^{-1}\widehat{\mathbf{D}}\mathbf{F} = \mathbf{F}^{\star}\widehat{\mathbf{D}}\mathbf{F},$$

  where $\mathbf{F}$ is the unitary DFT matrix.
  Observe this is a similarity transformation!
- Here $\mathbf{F}\mathbf{f}$ and $\mathbf{F}^{\star}\hat{\mathbf{f}}$ are computed using the (forward/inverse) FFT in nearly linear time.

Second derivative

- Differentiating the interpolant twice we get

$$\widehat{(\phi'')}_k = \hat{f}_k \begin{cases} -k^2 & \text{if } k < N/2 \\ -(k-N)^2 & \text{if } k \geq N/2 \end{cases}.$$

- Similarly, if $g = f''$ then $\hat{\mathbf{g}} = \widehat{\mathbf{D}}_2 \hat{\mathbf{f}}$, where $\widehat{\mathbf{D}}_2$ has $-k^2$ on its diagonal, $\widehat{\mathbf{D}}_2^\star = \widehat{\mathbf{D}}_2$ (Hermitian, same for $\mathbf{D}_2$).
- Double differentiating is different from differentiating twice in sequence, i.e., $\mathbf{D}_2 \neq \mathbf{D}_1^2$.
- Why is $\mathbf{D}_2$ "better" than $\mathbf{D}_1^2$? They have the same spectral accuracy.
- $\mathbf{D}_1^2$ has a nontrivial null space of $\mathbf{1}$ and $\mathbf{F}^{-1}\mathbf{e}_{N/2}$, while $\mathbf{D}_2$ has only $\mathbf{1}$.
- So $\mathbf{D}_2$ is closer to the **continuum Laplacian operator** in periodic domains (having only constant functions in its null space). This is important when solving elliptic/parabolic PDEs.

Discrete Matrices vs Continuum Operators

- The lesson learned from $\mathbf{D}_2 \neq \mathbf{D}_1^2$ is quite general: **Continuum identities don't always translate to discrete identities.**
- Many properties that seem obvious in continuum, may not work for discretizations:
  - Chain and product rules e.g., $(cu)' = c'u + cu'$.
  - Integration by parts (including boundary terms).
  - Operators commute, e.g., $\partial_x(\partial_y f) = \partial_y(\partial_x f)$.
  - Null spaces, eigenvalue spectrum properties (e.g., positive definiteness, symmetry, etc.).
- **Mimetic discretizations** try to mimic some of the properties of continuum operators.

Sturm-Louville Problems

- As an example, consider the periodic Sturm-Louville (SL) operator appearing in many boundary-value problems (BVPs):

$$\mathcal{L} = -\frac{d}{dx} c(x) \frac{d}{dx}, \quad c(x) > 0.$$

- From PDE class we know that this is a **symmetric positive semidefinite (SPsD) differential operator** with only constant functions in its null space; proving this uses integration by parts.
- When discretized, this will become a matrix $\mathbf{L}$. We want this matrix to be SPsD with only $\mathbf{e}$ in its null space.
- It is a bad idea is to use the chain rule and discretize:

$$-\mathcal{L}f = \frac{d}{dx} c(x) \frac{d}{dx} f(x) = c'f' + cf''$$

$$-\mathbf{Lf} = (\mathbf{D}_1\mathbf{c}) \boxdot (\mathbf{D}_1\mathbf{f}) + c \boxdot (\mathbf{D}_2\mathbf{f}) \quad \text{(BAD!)}$$

since this is not an SPsD $\mathbf{L}$.

Pseudospectral SL operator

- Another possibility is the **pseudospectral algorithm** that does not use the chain rule:

$$\mathbf{L}\mathbf{f} = -\mathbf{D}_1\left(\mathbf{c} \boxdot \mathbf{D}_1\mathbf{f}\right).$$

$$\mathbf{L}\mathbf{f} = -\mathcal{F}^{-1}\left(i\mathbf{k} \boxdot \mathcal{F}\left(\mathbf{c} \boxdot \left(\mathcal{F}^{-1}\left(i\mathbf{k} \boxdot \left(\mathcal{F}\mathbf{f}\right)\right)\right)\right)\right).$$

- In words: Go to Fourier space using the FFT, multiply coefficients by ik, go back to real space with iFFT, multiply by $c(x)$ in real-space, then go back to Fourier space (FFT) and multiply coefficients by $-ik$, and then go back to real space again (iFFT).
- Why does this work? In matrix notation

$$\mathbf{L} = -\left(\mathbf{F}^\star\widehat{\mathbf{D}}_1\mathbf{F}\right)\mathbf{C}\left(\mathbf{F}^\star\widehat{\mathbf{D}}_1\mathbf{F}\right) = \mathbf{D}_1\mathbf{C}\mathbf{D}_1^\star,$$

where $\mathbf{C}$ is a diagonal matrix with $\mathbf{c} > 0$ on its diagonal.
- This is obviously SPsD since $\mathbf{C}$ is SPD (why?).

Pseudospectral SL algorithm

For even N the pseudo-spectral $\mathbf{L}$ has a nontrivial null space just like $\mathbf{D}_1^2$ does (think $c = 1$), but this can be fixed (see article by Johnson):

1. Compute $\mathbf{f}'$ using FFT/iFFT but save the coefficient $\hat{f}_{N/2}$ (two FFTs).
2. Compute $\mathbf{g} = \mathbf{c} \boxdot \mathbf{f}'$ in real space (pseudospectral part).
3. Compute $\hat{\mathbf{g}}$ using FFT.
4. Compute $\widehat{(\mathbf{Lf})}$ in Fourier space as:

$$\widehat{(\mathbf{Lf})}_k = \begin{cases} \hat{c}_0 \left(\frac{N}{2}\right)^2 \hat{f}_{N/2} & \text{if } k = N/2 \\ -ik\hat{g}_k & \text{if } k < N/2 \\ -i\left(k - N\right)\hat{g}_k & \text{if } k > N/2 \end{cases}.$$

5. Compute $\mathbf{Lf}$ in real space using an iFFT.

Solving PDEs using FFTs

KdV equation

- Consider as an example the periodic Korteweg – de Vries equation on $[0, 2\pi)$,

$$\partial_t \phi = -\partial_{xxx} \phi + 6\phi \left(\partial_x \phi \right),$$

  which models waves in a channel and has **soliton** solutions.

- First note that $\phi\phi_x = \partial_x \left(\phi^2 / 2 \right)$ and this is the right form to use because **KdV is a conservation law** and $\phi^2/2$ is a flux.

- **Not all forms of PDEs equivalent on paper are equivalent numerically!** We prefer

$$\partial_t \phi = -\partial_{xxx} \phi + 3\partial_x \left(\phi^2 \right).$$

- The idea is to use a Fourier series representation,

$$\phi(x, t) = \sum_k \hat{\phi}_k(t) e^{ikx}.$$

Spectral spatial discretization

- If we go to Fourier space we get a **system of coupled (nonlinear) ODEs**:

$$\frac{d\hat{\phi}_k}{dt} = ik^3\hat{\phi}_k + 3ik\widehat{\left(\phi^2\right)}_k \quad \Rightarrow$$

$$\frac{d\hat{\phi}}{dt} = ik^3 \boxdot \hat{\phi} + 3i\mathbf{k} \boxdot \mathcal{F}\left(\left(\mathcal{F}^{-1}\hat{\phi}\right)^{\boxed{2}}\right).$$

- Note that the unmatched mode $N/2$ should be set to zero for the third derivative (all odd derivatives in fact).
- This is a **pseudo-spectral spatial discretization** and will be spectrally accurate for analytic solutions.
- In order to actually compute solutions we need methods to solve systems of ODEs (coming up soon)!

Nonlinear PDEs

- Observe that if the nonlinear term was not there, we could write the solution right away:

$$\hat{\phi}_k(t) = \hat{\phi}_k(0) \exp\left(ik^3 t\right) \text{ for all } k.$$

- This is called an **exponential temporal integrator** and can be used to build accurate integrators for the nonlinear KdV equation.
- If the equation were linear, then $\hat{\phi}_k(t) = 0$ if $\hat{\phi}_k(0) = 0$: **linear PDEs do not generate new Fourier components.**
- But this is not true for nonlinear equations: in general, the solution will have nonzero components for all k for sufficiently long times, and **aliasing** becomes a problem.
- An extreme example is Burger's equation, which **develops singularities** (shocks), leading to the Gibbs phenomenon and **loss of spectral accuracy**.

Aliasing

- As an example, consider the product (or square)

$$w(x) = u(x)v(x) \quad \Rightarrow$$

$$w(x) = \left(\sum_{k''=-n}^{n} \hat{u}_{k''} e^{ik''x} \right) \left(\sum_{k=-n}^{n} \hat{u}_{k'} e^{ik'x} \right) = \sum_{k=-2n}^{2n} \hat{w}_k e^{ikx}$$

- So we doubled the number of Fourier modes, and handling this would require growing our FFT grid along the way!
- What we want to compute is the truncated Fourier series

$$w(x) \approx \tilde{w}(x) = \sum_{k=-n}^{n} \hat{w}_k e^{ikx}.$$

- If we do this naively using FFTs on a grid of $N = 2n + 1$ points, however, we will alias the modes $|k| > n$ with those with $|k| < n$ and this will introduce aliasing error.

Anti-aliasing via oversampling

- But there is an easy fix using **oversampling**. Take $u = v$ for simplicity and even N:

1. Evaluate $u(x)$ on a grid of N points, take the FFT to compute $\hat{\mathbf{u}}$.
2. Padd the FFT to size $M = 2N$, avoiding fftshift (see fftinterp):

$$(\hat{\mathbf{u}})_{\text{padded}} = [\hat{\mathbf{u}}\,(1 : N/2) \quad \text{zeros}(1, M-N) \quad \hat{\mathbf{u}}\,(N/2+1 : \text{end})]\,.$$

   Note: It can be shown that $M = 3N/2$ also gives the same result.

3. Compute an oversampled $u_{\text{os}}(x)$ on a grid of size $2N$ by taking the iFFT of $(\hat{\mathbf{u}})_{\text{padded}}$.
4. Compute $\mathbf{u}_{\text{os}}^{\boxed{2}}$ in real space, and take the FFT to compute $\hat{\mathbf{w}}$.
5. Truncate to N Fourier coefficients by returning $[\hat{\mathbf{w}}\,(1 : N/2) \quad \hat{\mathbf{w}}\,(M - N/2 + 1 : \text{end})]$.

Chebyshev Series via FFTs

Chebyshev Polynomials

- If we are solving PDEs on a bounded interval, say $[-1, 1]$ for simplicity, we need other orthogonal polynomials, not trig ones.
- Recall from Numerical Methods I the Chebyshev polynomials:

$$T_n\left(x \in [-1, 1]\right) = \cos\left(n\theta\right) \quad \text{where} \quad x = \cos\left(\theta \in [0, 2\pi]\right).$$

$$T_0\left(x\right) = 1,\ T_1(x) = x,\ T_2(x) = 2x^2 - 1,\ T_3(x) = 4x^3 - 3x, \ldots$$

- These are orthogonal with respect to the weighted inner/dot product:

$$\int_{-1}^{1} T_m(x)\, T_n(x) \frac{dx}{\sqrt{1 - x^2}} = \begin{cases} \pi & m = n = 0 \\ \pi/2 & m = n > 0 \\ 0 & m \neq n \end{cases}.$$

Chebyshev Interpolants

- We can represent functions using these polynomials as basis functions,

$$f(x)=\sum_{n=0}^{\infty}\breve{f}_n T_n(x) \quad \Rightarrow$$
$$\breve{f}_{n>0}=\frac{2}{\pi}\int_{-1}^{1} f(x)T_n(x)\frac{dx}{\sqrt{1-x^2}}.$$

- We discretize the function pointwise at $N+1$ **Chebyshev nodes**

$$\theta_j = j\pi/N, \quad j=0\ldots N$$
$$x_j=\cos\theta_j$$

- This gives us the **Chebyshev interpolant** (approximation):

$$\phi(x)=\sum_{n=0}^{N}\breve{f}_n T_n(x).$$

Chebyshev Nodes

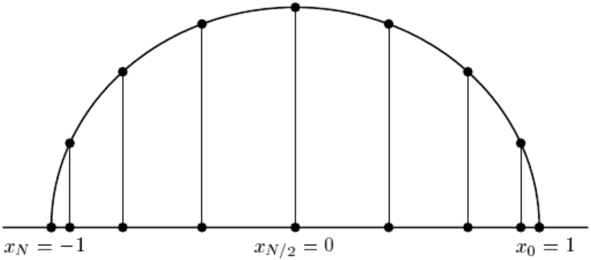

Fig. 5.1. Chebyshev points are the projections onto the x-axis of equally spaced points on the unit circle. Note that they are numbered from right to left.

Chebyshev via Fourier

- Changing variables from x to θ we get

$$\begin{aligned}
\breve{f}_{n>0} &= \frac{2}{\pi}\int_{-1}^{1} f(x)\,T_n(x)\frac{dx}{\sqrt{1-x^2}}.\\
&= \int_{-\pi}^{\pi} f\left(\cos\theta\right)\cos\left(n\theta\right)d\theta\\
&= \int_{-\pi}^{\pi} f\left(\cos\theta\right)\left(\frac{\exp\left(in\theta\right)+\exp\left(-in\theta\right)}{2}\right)d\theta.
\end{aligned}$$

- So if we consider instead of $f(x)$ the function

$$g(\theta) = f\left(\cos\theta\right)$$

then we can go from **Fourier coefficients** of g to **Chebyshev** for f:

$$\breve{f}_{n>0} = \hat{g}_{-n} + \hat{g}_{n}$$

Chebyshev-Fourier transformation

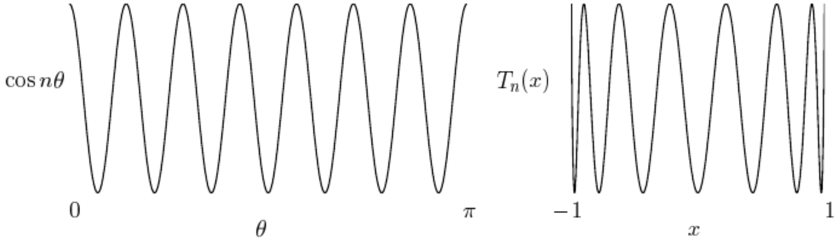

Fig. 8.2. The Chebyshev polynomial T_n can be interpreted as a sine wave "wrapped around a cylinder and viewed from the side".

Chebyshev via FFT

- This means that we can do **FFTs in equispaced points on** $\theta \in [0, 2\pi]$ instead of Chebyshev on non-equispaced nodes.
- Note that we want to extend this to $\theta \in [0, 2\pi]$ to be periodic and not $\theta \in [0, \pi]$, so we **double the number of points** and do the FFTs on vectors of length $2N$.
- If $f(x)$ can be extended analytically just outside of $[-1, 1]$, then we get **spectral accuracy**.
- Intuition: **Chebyshev polynomials are sine waves "wrapped around a cylinder and viewed from the side"**.
- One can **approximate derivatives using the FFT**; all that is needed is change of variables from x to θ using the chain rule.
- The chain of variables adds factors of the form $\left(1 - x^2\right)^{-p/2}$ (where p is an integer) when converting from Fourier coefficients derivatives of g to derivatives of f.

Conclusions

Function Norms

- Consider a one-dimensional interval $I = [a, b]$. Standard norms for functions similar to the usual vector norms:
  - **Maximum norm**: $\|f(x)\|_\infty = \max_{x \in I} |f(x)|$
  - L_1 **norm**: $\|f(x)\|_1 = \int_a^b |f(x)|\, dx$
  - **Euclidian** L_2 **norm**: $\|f(x)\|_2 = \left[\int_a^b |f(x)|^2\, dx\right]^{1/2}$
- **Different function norms are not equivalent!**
- An **inner or scalar product** (equivalent of dot product for vectors):

$$(f, g) = \int_a^b f(x) g^\star(x) dx$$

- Formally, function spaces are **infinite-dimensional linear spaces**. Numerically we always **truncate and use a finite basis**.

Discrete Function Norms

- Consider a set of m **nodes** $x_i = a + ih$ with a constant grid spacing $h = (b-a)/m$, and evaluate the function at those nodes **pointwise**

$$\mathbf{f} = \{f(x_0), f(x_1), \cdots, f(x_m)\}.$$

- We define the discrete "function norms" and "dot products", with periodic BCs:

$$\|f(x)\|_2 \approx \left[h \sum_{i=0}^{m-1} |f(x_i)|^2\right]^{1/2} = \sqrt{h}\,\|\mathbf{f}\|_2,$$

$$\|f(x)\|_1 \approx h \sum_{i=0}^{m-1} |f(x_i)| = h\,\|\mathbf{f}\|_1,$$

$$\|f(x)\|_\infty \approx \max_i |f(x_i)| = \|\mathbf{f}\|_\infty$$

$$(f,g) \approx \mathbf{f} \cdot \mathbf{g} = h \sum_{i=0}^{m-1} f(x_i) g^\star(x_i).$$

Conclusions/Summary

- **Convolution** in real space becomes **multiplication** in Fourier space, and vice versa.
- **Spectrally-accurate derivatives** $f^{(\nu)}$ of analytic functions f can be done by multiplication by $(ik)^{\nu}$ in Fourier space, zeroing out the unmatched mode for even N and odd ν.
- Not all forms of operators and PDEs equal on paper are equal numerically. **Choose the form that preserves the important properties of the continuum PDE**: conservation laws, self-Hermitian operators, completeness (this is where understanding PDEs is crucial beyond superficial: functional analysis).
- Nonlinear PDEs can be **discretized spectrally in space to a system of coupled nonlinear ODEs**. Non-periodic domains can be handled by using orthogonal polynomials but boundary conditions need to be thought about some more!