26. Discrete Fourier transform

Last time

- Approximating functions globally
- Fourier series
- Fourier coefficients
- Rate of decay of Fourier coefficients $\leftrightarrow$ differentiability

Goals for today

- Approximating functions using Fourier analysis
- Discrete Fourier transform

Review: Fourier series

- Setting: Continuous, periodic function $f(t)$ with period 2π
- Think of t as living in circle $\mathbb{S}^1$, i.e. $t \in [0, 2\pi)$ with $0 \equiv 2\pi$

Review: Fourier series

- Setting: Continuous, periodic function $f(t)$ with period 2π
- Think of t as living in circle $\mathbb{S}^1$, i.e. $t \in [0, 2\pi)$ with $0 \equiv 2\pi$

- Can write f as **Fourier series**
- Infinite linear combination

$$f = \sum_{n=-\infty}^{\infty} \hat{f}_n \phi_n$$

with $\phi_n(t) := \exp(int)$ and $\hat{f}_n \in \mathbb{C}$

Review: Inner product

- Define **inner product** between two functions f, g:

$$(f, g) := \int_0^{2\pi} \overline{f(x)}\, g(x)\, dx$$

Review: Inner product

- Define **inner product** between two functions f, g:

$$(f, g) := \int_0^{2\pi} \overline{f(x)}\, g(x)\, dx$$

- Have $(\phi_n, \phi_m) = 2\pi\, \delta_{n,m}$

Review: Inner product

- Define **inner product** between two functions f, g:

$$(f, g) := \int_0^{2\pi} \overline{f(x)}\, g(x)\, dx$$

- Have $(\phi_n, \phi_m) = 2\pi\, \delta_{n,m}$
- Take (ϕ_m, f) to find

$$\hat{f}_m = \frac{1}{2\pi} \int_0^{2\pi} e^{-imx} f(x) dx$$

Approximating functions using Fourier series

- How approximate a function f?

Approximating functions using Fourier series

- How approximate a function f?
- According to the above, calculate a finite number of its Fourier coefficients and use

$$f \simeq f_N := \sum_{n=-N}^{N} \hat{f}_n \phi_n$$

Approximating functions using Fourier series

- How approximate a function f?
- According to the above, calculate a finite number of its Fourier coefficients and use

$$f \simeq f_N := \sum_{n=-N}^{N} \hat{f}_n \phi_n$$

- We saw that $\|f - f_N\|$ decays exponentially fast if f is analytic

Approximating functions using Fourier series

- How approximate a function f?
- According to the above, calculate a finite number of its Fourier coefficients and use

$$f \simeq f_N := \sum_{n=-N}^{N} \hat{f}_n \phi_n$$

- We saw that $\|f - f_N\|$ decays exponentially fast if f is analytic
- But calculating integrals numerically is not particularly pleasant
- Is there an alternative?

Approximating functions using Fourier series

- How approximate a function f?
- According to the above, calculate a finite number of its Fourier coefficients and use

$$f \simeq f_N := \sum_{n=-N}^{N} \hat{f}_n \phi_n$$

- We saw that $\|f - f_N\|$ decays exponentially fast if f is analytic
- But calculating integrals numerically is not particularly pleasant
- Is there an alternative?

Interpolation using trigonometric functions

- Return to idea from start of course: **interpolation**

Interpolation using trigonometric functions

- Return to idea from start of course: **interpolation**
- Let's try to **interpolate** f using trigonometric functions
- (Previously used only polynomials for interpolation)

Interpolation using trigonometric functions

- Return to idea from start of course: **interpolation**
- Let's try to **interpolate** f using trigonometric functions
- (Previously used only polynomials for interpolation)
- Recall setting of interpolation:

  Given t_j *and* $f_j = f(t_j)$ *for* $j = 0, \ldots, N$, *find* g *in some class such that* $g(t_j) = f_j$

- In context of periodic functions take $t_N = t_0$

Interpolation using trigonometric functions

- Return to idea from start of course: **interpolation**
- Let's try to **interpolate** f using trigonometric functions
- (Previously used only polynomials for interpolation)
- Recall setting of interpolation:

  Given t_j *and* $f_j = f(t_j)$ *for* $j = 0, \ldots, N$*, find* g *in some class such that* $g(t_j) = f_j$

- In context of periodic functions take $t_N = t_0$
- We will look for trigonometric functions

$$g = \sum^{N-1} g_k \, \phi_k$$

Interpolation II

- Condition for interpolation:

$$g(t_j) = f_j$$

- So interpolate N points (t_j, f_j) with a sum of basis functions with N unknown coefficients g_n:

$$\sum_{k=0}^{N-1} g_k \, \phi_k(t_j) = f_j$$

Interpolation II

- Condition for interpolation:

$$g(t_j) = f_j$$

- So interpolate N points (t_j, f_j) with a sum of basis functions with N unknown coefficients g_n:

$$\sum_{k=0}^{N-1} g_k \, \phi_k(t_j) = f_j$$

- Which points should we interpolate in?

Interpolation II

- Condition for interpolation:

$$g(t_j) = f_j$$

- So interpolate N points (t_j, f_j) with a sum of basis functions with N unknown coefficients g_n:

$$\sum_{k=0}^{N-1} g_k \, \phi_k(t_j) = f_j$$

- Which points should we interpolate in?
- For polyomials, equally-spaced points were **bad**

Interpolation III

- So interpolation condition becomes

$$\sum_{k=0}^{N-1} g_k \exp(ikjh) = f_j$$

Interpolation III

- So interpolation condition becomes

$$\sum_{k=0}^{N-1} g_k \, \mathrm{exp}(ikjh) = f_j$$

- What kind of equations are these for the unknown coefficients g_n?

Interpolation III

- So interpolation condition becomes

$$\sum_{k=0}^{N-1} g_k \, \mathsf{exp}(ikjh) = f_j$$

- What kind of equations are these for the unknown coefficients g_n?
- It's a **linear system** for the g_n:

$$\sum_k M_{jk} \, g_k = f_j$$

- I.e. a matrix equation

Interpolation IV

- Solution is $\mathbf{g} = \mathsf{M}^{-1}\mathbf{f}$
- Can we find an analytical solution for this?

Interpolation IV

- Solution is $\mathbf{g} = \mathsf{M}^{-1}\mathbf{f}$
- Can we find an analytical solution for this?
- Under what circumstance is this equation easy to solve?

Interpolation IV

- Solution is $\mathbf{g} = \mathrm{M}^{-1}\mathbf{f}$
- Can we find an analytical solution for this?
- Under what circumstance is this equation easy to solve?
- Easy to solve if M is an **orthogonal** matrix

Interpolation IV

- Solution is $\mathbf{g} = \mathsf{M}^{-1}\mathbf{f}$
- Can we find an analytical solution for this?
- Under what circumstance is this equation easy to solve?
- Easy to solve if M is an **orthogonal** matrix
- It turns out that it almost is:

$$\mathsf{M}^*\mathsf{M} = N\mathsf{I}$$

  where M^* is the **conjugate transpose** $\mathsf{M}^* := \overline{\mathsf{M}^\top}$
- Hence

$$\mathbf{g} = \frac{1}{N}\mathsf{M}^*\mathbf{f}$$

Discrete Fourier transform (DFT)

- **Discrete Fourier transform** $\mathcal{F}$ maps *from* function values *to* Fourier coefficients:

$$g_k = \frac{1}{N} \sum_j e^{-ijk\frac{2\pi}{N}} f_j$$

- **Inverse discrete Fourier transform** maps *from* Fourier coefficients *to* function values:

$$f_j = \sum_k e^{ijk\frac{2\pi}{N}} g_k$$

Fast Fourier transform

- It looks like we need to do a matrix–vector multiplication to calculate the DFT
- $\mathcal{O}(n^2)$ operations

Fast Fourier transform

- It looks like we need to do a matrix–vector multiplication to calculate the DFT
- $\mathcal{O}(n^2)$ operations

- However, one of the great algorithm discoveries of the 20th (also 19th) centuries was the **fast Fourier transform**
- Uses *structure* in M to calculate in $\mathcal{O}(n \log n)$ time
- Has myriad applications throughout applied mathematics, signal processing, physics, …

Trapezium rule

- Let's go back to the trapezium rule for a periodic function
- Take $f = \sum_{k=-\infty}^{\infty} \hat{f}_k \phi_k$

Trapezium rule

- Let's go back to the trapezium rule for a periodic function
- Take $f = \sum_{k=-\infty}^{\infty} \hat{f}_k \phi_k$
- Trapezium rule approximation of $I := \int f$ with N points is

$$S_N = \frac{h}{2} \sum_{j=0}^{N-1} f(jh)$$

with $h := \frac{2\pi}{N}$

Trapezium rule

- Let's go back to the trapezium rule for a periodic function
- Take $f = \sum_{k=-\infty}^{\infty} \hat{f}_k \phi_k$
- Trapezium rule approximation of $I := \int f$ with N points is

$$S_N = \frac{h}{2} \sum_{j=0}^{N-1} f(jh)$$

with $h := \frac{2\pi}{N}$

$$= \frac{\pi}{N} \sum_{j=0}^{N-1} \sum_{k=-\infty}^{\infty} \hat{f}_k e^{ijkh}$$

Trapezium rule

- Let's go back to the trapezium rule for a periodic function
- Take $f = \sum_{k=-\infty}^{\infty} \hat{f}_k \phi_k$
- Trapezium rule approximation of $I := \int f$ with N points is

$$S_N = \frac{h}{2} \sum_{j=0}^{N-1} f(jh)$$

with $h := \frac{2\pi}{N}$

$$= \frac{\pi}{N} \sum_{j=0}^{N-1} \sum_{k=-\infty}^{\infty} \hat{f}_k e^{ijkh}$$

Trapezium rule II

- $\sum_j = 0$ unless j is integer multiple of N

Trapezium rule II

- $\sum_j = 0$ unless j is integer multiple of N
- So get $S_N = \sum_{k=-\infty}^{\infty} \hat{f}_{kN}$

Trapezium rule II

- $\sum_j = 0$ unless j is integer multiple of N
- So get $S_N = \sum_{k=-\infty}^{\infty} \hat{f}_{kN}$
- $I = \hat{f}_0$

Trapezium rule II

- $\sum_j = 0$ unless j is integer multiple of N
- So get $S_N = \sum_{k=-\infty}^{\infty} \hat{f}_{kN}$
- $I = \hat{f}_0$
- So $|I - S_N| = \sum_{k=1\infty} (\hat{f}_{kN} + \hat{f}_{-kN})$

Trapezium rule II

- $\sum_j = 0$ unless j is integer multiple of N
- So get $S_N = \sum_{k=-\infty}^{\infty} \hat{f}_{kN}$
- $I = \hat{f}_0$
- So $|I - S_N| = \sum_{k=1\infty} (\hat{f}_{kN} + \hat{f}_{-kN})$
- So if $\hat{f}_n$ decay exponentially fast then so does $|I - S_N$

Summary

- Can interpolate using trigonometric polynomials
- Get fast Fourier transform relating function values to Fourier coefficients
- Trapezium rule error has spectral decay