29. Chebyshev methods III

Last time

- Chebyshev interpolation
- Discrete Cosine transform
- Barycentric Lagrange interpolation

Chebyshev interpolation

- Given $f : [-1, +1] \to \mathbb{R}$

Chebyshev interpolation

- Given $f : [-1, +1] \to \mathbb{R}$
- Approximate $f = \sum_{k=0}^{N} \alpha_k T_k$
- **Chebyshev polynomials** $T_n(x) = \cos(n \arccos(x))$

Chebyshev interpolation

- Given $f : [-1, +1] \to \mathbb{R}$
- Approximate $f = \sum_{k=0}^{N} \alpha_k T_k$
- **Chebyshev polynomials** $T_n(x) = \cos(n \arccos(x))$
- Interpolate in Chebyshev points $t_j := \cos\left(\frac{\pi j}{N}\right)$
- $f_j := f(t_j)$ at $(N+1)$ points t_j with $j = 0, \ldots, N$

Chebyshev interpolation

- Given $f : [-1, +1] \to \mathbb{R}$
- Approximate $f = \sum_{k=0}^{N} \alpha_k T_k$
- **Chebyshev polynomials** $T_n(x) = \cos(n \arccos(x))$
- Interpolate in Chebyshev points $t_j := \cos\left(\frac{\pi j}{N}\right)$
- $f_j := f(t_j)$ at $(N+1)$ points t_j with $j = 0, \ldots, N$
- Discrete Cosine Transformation (DCT):

$$\sum_k \alpha_k \cos\left(\frac{j\,k\pi}{n}\right) = f_j$$

where $f_j := f(t_j)$

Goals for today

- Choosing the number of interpolation points
- Operations using Chebyshev representation
- Derivatives
- Integrals
- Roots

Choosing number of interpolation points

- Fundamental idea:

  Represent / approximate function f by Chebyshev interpolant in Chebyshev points

  . . .

- But *how many* points should we choose?

Choosing number of interpolation points

- Fundamental idea:

  Represent / approximate function f by Chebyshev interpolant in Chebyshev points

  . . .

- But *how many* points should we choose?
- Enough that Chebyshev coefficients have decayed to ϵ_{mach}

Choosing number of interpolation points II

. . .

- Start with $N+1$, e.g. $N=8$

Choosing number of interpolation points II

. . .

- Start with $N+1$, e.g. $N=8$
- Interpolate f to get $f_j^{(N)}$

Choosing number of interpolation points II

. . .

- Start with $N+1$, e.g. $N=8$
- Interpolate f to get $f_j^{(N)}$
- Calculate α_k using DCT or preferably fast FCT
- Check if have decayed, e.g. last two are $< 10^{-13}

Choosing number of interpolation points II

. . .

- Start with $N + 1$, e.g. $N = 8$
- Interpolate f to get $f_j^{(N)}$
- Calculate α_k using DCT or preferably fast FCT
- Check if have decayed, e.g. last two are \$< 10^{-13}
- If not, **double** N and try again
- Can reuse: $f_{2j}^{(2N)} = f_j^{(N)}$

Differentiation

- Suppose have $f = \sum_{k=0}^{N} \alpha_k T_k$
- How can we calculate the derivative f'?

Differentiation

- Suppose have $f = \sum_{k=0}^{N} \alpha_k T_k$
- How can we calculate the derivative f'?
- Will also be a sum of T_k:

$$f' = \sum_{k=1}^{N} \alpha_k T_k'$$

Differentiation

- Suppose have $f = \sum_{k=0}^{N} \alpha_k T_k$
- How can we calculate the derivative f'?
- Will also be a sum of T_k:

$$f' = \sum_{k=1}^{N} \alpha_k T_k'$$

- This will be polynomial of degree $N - 1$

Differentiation II

- Let p be the unique polynomial that interpolates the f_j
- Set $w_j := p'(t_j)$

Differentiation II

- Let p be the unique polynomial that interpolates the f_j
- Set $w_j := p'(t_j)$
- Calculate using Lagrange interpolation since have explicit formulae (unlike for Chebyshev polynomials)

Differentiation II

- Let p be the unique polynomial that interpolates the f_j
- Set $w_j := p'(t_j)$
- Calculate using Lagrange interpolation since have explicit formulae (unlike for Chebyshev polynomials)
- Then

$$\mathbf{w} = \mathsf{D}_N \mathbf{f}$$

- Where D_N is $(N+1) \times (N+1)$ **Chebyshev differentiation matrix**
- Chapter 6 of Trefethen, *Spectral Methods in MATLAB* has explicit formulae for D_N

Differentiation III

- The matrices D_N are *dense* matrices

Differentiation III

- The matrices D_N are *dense* matrices
- Modern approach: Olver & Townsend 2014, “A fast and well-conditioned spectral method”

Differentiation III

- The matrices D_N are *dense* matrices

- Modern approach: Olver & Townsend 2014, "A fast and well-conditioned spectral method"
- Derivative T_k' is ultraspherical polynomial
- Get banded matrix if use correct bases

Differentiation III

- The matrices D_N are *dense* matrices

- Modern approach: Olver & Townsend 2014, “A fast and well-conditioned spectral method”
- Derivative T_k' is ultraspherical polynomial
- Get banded matrix if use correct bases
- “Differentiating scales the coefficients and changes the basis”

Differentiation IV

- Note that differentiation lowers the degree of the polynomial
- So repeated differentiation is a bad idea

Differentiation IV

- Note that differentiation lowers the degree of the polynomial
- So repeated differentiation is a bad idea

- Alternative: Use **automatic differentiation**!
- Calculate $f'(t_j)$ using dual numbers exactly (up to rounding error)

Differentiation IV

- Note that differentiation lowers the degree of the polynomial
- So repeated differentiation is a bad idea

- Alternative: Use **automatic differentiation**!
- Calculate $f'(t_j)$ using dual numbers exactly (up to rounding error)
- Higher-order derivatives using Taylor methods

Differentiation IV

- Note that differentiation lowers the degree of the polynomial
- So repeated differentiation is a bad idea

- Alternative: Use **automatic differentiation**!
- Calculate $f'(t_j)$ using dual numbers exactly (up to rounding error)
- Higher-order derivatives using Taylor methods
- Then interpolate again!

Recurrence relation

- 3-term **recurrence relation** relating T_k to T_{k-1} and T_{k_2}:

  $\$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$

Recurrence relation

- 3-term **recurrence relation** relating T_k to T_{k-1} and T_{k_2}:

  \$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$

- Where does this come from?

Recurrence relation

- 3-term **recurrence relation** relating T_k to T_{k-1} and T_{k_2}:

  \$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$

- Where does this come from?

- Consider $xT_k(x)$

Recurrence relation

- 3-term **recurrence relation** relating T_k to T_{k-1} and T_{k_2}:

  \$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$

- Where does this come from?

- Consider $xT_k(x)$
- This is a polynomial of degree $k+1$, so

$$xT_k(x) = \sum_{j=0}^{k+1} \alpha_j T_j(x)$$

Recurrence relation

- 3-term **recurrence relation** relating T_k to T_{k-1} and T_{k_2}:

  \$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$

- Where does this come from?

- Consider $xT_k(x)$
- This is a polynomial of degree $k+1$, so

$$xT_k(x) = \sum_{j=0}^{k+1} \alpha_j T_j(x)$$

- α_j are given by (xT_k, T_j)

Recurrence relation II

- We have

$$(xT_k, T_j) = \int_{-1}^{-1} xT_k(x)T_j(x)dx$$

- Change variables using $x = \cos(\theta)$:

$$(xT_k, T_j) = \int_0^{2\pi} c_1 c_k c_j d\theta$$

where $c_j(\theta) := \cos(j\theta)$

Recurrence relation II

- We have

$$(xT_k, T_j) = \int_{-1}^{-1} xT_k(x)T_j(x)dx$$

- Change variables using $x = \cos(\theta)$:

$$(xT_k, T_j) = \int_0^{2\pi} c_1 c_k c_j d\theta$$

  where $c_j(\theta) := \cos(j\theta)$

- Use trigonometric relation

$$\cos(A)\cos(B) = \frac{1}{2}[\cos(A+B) + \cos(A-B)]$$

Recurrence relation II

- We have

$$(xT_k, T_j) = \int_{-1}^{-1} xT_k(x)T_j(x)dx$$

- Change variables using $x = \cos(\theta)$:

$$(xT_k, T_j) = \int_0^{2\pi} c_1 c_k c_j d\theta$$

where $c_j(\theta) := \cos(j\theta)$

- Use trigonometric relation

$$\cos(A)\cos(B) = \tfrac{1}{2}[\cos(A+B) + \cos(A-B)]$$

Recurrence relation III

- We have

$$(xT_k, T_j) = \frac{1}{2}\int_0^{2\pi} c_1[c_{k+j} + c_{k-j}]$$

Recurrence relation III

- We have

$$(xT_k, T_j) = \frac{1}{2}\int_0^{2\pi} c_1[c_{k+j} + c_{k-j}]$$

- But $\int c_l c_m = 0$ if $l \neq m$

Recurrence relation III

- We have

$$(xT_k, T_j) = \frac{1}{2}\int_0^{2\pi} c_1[c_{k+j} + c_{k-j}]$$

- But $\int c_l c_m = 0$ if $l \neq m$
- So (xT_k, T_j) is 0 unless $j = k + 1$ or $j = k - 1$

Recurrence relation III

- We have

$$(xT_k, T_j) = \frac{1}{2}\int_0^{2\pi} c_1[c_{k+j} + c_{k-j}]$$

- But $\int c_l c_m = 0$ if $l \neq m$
- So (xT_k, T_j) is 0 unless $j = k + 1$ or $j = k - 1$
- Hence $xT_k = \alpha T_{k+1} + \beta T_{k-1}$

  and we can calculate the constants α and β

Recurrence relation III

- We have

$$(xT_k, T_j) = \frac{1}{2}\int_0^{2\pi} c_1[c_{k+j} + c_{k-j}]$$

- But $\int c_l c_m = 0$ if $l \neq m$
- So (xT_k, T_j) is 0 unless $j = k + 1$ or $j = k - 1$
- Hence $xT_k = \alpha T_{k+1} + \beta T_{k-1}$

  and we can calculate the constants α and β

- Can show that *any* orthogonal polynomials have a similar 3-term recurrence

Integration

- Now suppose want to integrate f
- Approximate f as $f = \sum_{k=0}^{N} \alpha_k T_k$ such that error is small

Integration

- Now suppose want to integrate f
- Approximate f as $f = \sum_{k=0}^{N} \alpha_k T_k$ such that error is small
- Now integrate the resulting polynomial
- **Clenshaw–Curtis integration**

Integration

- Now suppose want to integrate f
- Approximate f as $f = \sum_{k=0}^{N} \alpha_k T_k$

  such that error is small
- Now integrate the resulting polynomial
- **Clenshaw–Curtis integration**
- Will get spectral accuracy due to spectral accuracy of the polynomial interpolation!

Integration II

- Two possibilities:

$$\int f = \sum_{k=0}^{N} \alpha_k \int T_k$$

- Pre-calculate and store $\int T_k$ – integral of a polynomial

Integration II

- Two possibilities:

$$\int f = \sum_{k=0}^{N} \alpha_k \int T_k$$

- Pre-calculate and store $\int T_k$ – integral of a polynomial
- Calculate explicit polynomial using recurrence relation

Integration II

- Two possibilities:

$$\int f = \sum_{k=0}^{N} \alpha_k \int T_k$$

- Pre-calculate and store $\int T_k$ – integral of a polynomial
- Calculate explicit polynomial using recurrence relation
- Integral becomes a dot product!

Integration II

- Alternative: use Lagrange interpolation in f_j
- Integrate Lagrange interpolant

Integration II

- Alternative: use Lagrange interpolation in f_j
- Integrate Lagrange interpolant
- Can find explicit formulae for the result

Summary

- Fundamental mathematical operations become "easy" once we have spectral approximation
- Spectral convergence gives excellent approximation of function
- This is (mostly) maintained by operations like differentiation, integration
- Orthogonal polynomials satisfy 3-term recurrence relations