2. Lecture 2

2.1. Review of Some Linear Algebra. A system of m linear equations with n unknowns $\{x_1, ..., x_n\}$ is of the form

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n = b_m, \end{cases}$$

can be rewritten in a matrix-vector form

$$Ax = b,$$

where

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, \qquad b = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$$

and

$$A = \begin{pmatrix} a_{11} & a_{12} & ... & a_{1n} \\ a_{21} & a_{22} & ... & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{m1} & a_{m2} & ... & a_{mn} \end{pmatrix}.$$

We will often deal with square systems, i.e. $m = n$. For a square system:

Theorem 2.1 (Invertibility)**.** *Let A be a $n \times n$ matrix. The following are equivalent*

- $\det(A) \neq 0$*;*
- *A is invertible, i.e. there exists a $n \times n$ matrix B satisfying*

$$BA = AB = I,$$

  (I denotes the identity matrix);
- $\text{Range}(A) = \mathbb{R}^n$*;*
- $\text{Ker}(A) = \{x \in \mathbb{R}^n \ : \ Ax = 0\} = \{0\}$*;*
- $\text{Rank}(A) = n$*;*
- *For any $b \in \mathbb{R}^n$, the system $Ax = b$ has a unique solution $x \in \mathbb{R}^n$.*

2.2. Polynomial Interpolation (see Chapt. 6).

Definition 2.1 (Polynomial Space)**.** *We define $\mathbb{P}^k$ to be the collection of all polynomials of degree at most k, i.e.*

$$\mathbb{P}^k = \text{span}\{1, x, ..., x^k\}.$$

In particular $\dim(\mathbb{P}^k) = k + 1$.

Interpolation problem: Given a function f and interpolation points $x_1, ..., x_m$, find $p \in \mathbb{P}^k$ such that

$$p(x_i) = f(x_i), \qquad i = 1, ..., m.$$

Some remarks are in order.

Remark 2.1 (Distinct Interpolation Points)*.* The $x_i's$ should be distinct otherwise the equations are redundant.

Remark 2.2 (Number of Interpolation Points). $\mathbb{P}^k$ has dimension $k+1$ so we need at least $m = k+1$ snapshots.

Remark 2.3 (Approximation). The interpolating polynomial $p(x)$ provides an approximation to f.

Theorem 2.2 (Interpolation)**.** *Let $x_0, ..., x_n$ be distinct real numbers. Then for arbitrary snapshots $y_0, ..., y_n$ there exists a unique $p \in \mathbb{P}^n$ satisfying*

$$p(x_i) = y_i, \qquad i = 0, ..., n.$$

Proof. Let $p(x) = a_0 + a_1 x + ... + a_n x_n$ such that $p(x_j) = y_j$ for $j = 0, ..., n$. The latter conditions can be rewritten

$$a_0 + a_1 x_j + a_2 x_j^2 + ... + a_n x_j^n = y_j, \qquad j = 0, ..., n.$$

This is a linear system of $n+1$ equations and $n+1$ unknowns $\{a_0, ..., a_n\}$. This system corresponds to

$$Ba = y,$$

where

$$y = \begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix}, \qquad a = \begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix}$$

and

$$B = \begin{pmatrix} 1 & x_0 & x_0^2 & ... & x_0^n \\ 1 & x_1 & x_1^2 & ... & x_1^n \\ \vdots & & \ddots & \vdots & \\ 1 & x_n & x_n^2 & ... & x_n^n \end{pmatrix}.$$

According to Theorem 2.1, we shall show that B is invertible by checking that $\mathrm{Ker}(B) = \{0\}$. Suppose that $Bc = 0$ for some

$$c = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{pmatrix}.$$

Consider $q(x) = c_0 + c_1 x + ... + c_n x^n$. Then $(Bc)_j = 0$ is equivalent to

$$c_0 + c_1 x_j + c_2 x_j^2 + ... + c_n x_j^n = 0,$$

i.e.

$$q(x_j) = 0.$$

The only polynomial of degree n with $n+1$ distinct roots is the zero polynomial. Thus $c_0 = c_1 = ... = c_n = 0$, i.e. $c = 0$ and $\mathrm{Ker}(B) = \{0\}$. □

Remark 2.4 (Function Interpolation). This implies that there exists a unique polynomial $p \in \mathbb{P}^n$ interpolating f at $x_0, ..., x_n$.

The following method construct recursively the polynomial $p \in \mathbb{P}^n$ satisfying $p(x_i) = y_i$ for $i = 0, ..., n$.

Newton Form: Given $x_0, ..., x_n$ distincts and $y_0, ..., y_n$.

(1) if $n = 0$ then $\mathbb{P}^0 = \text{span}\{1\}$ and $p_0 \in \mathbb{P}^0$ satisfying $p_0(x_0) = y_0$ is the constant polynomial $p_0(x) = y_0$.

(2) Suppose that $p_k \in \mathbb{P}^k$ has been constructed satisfying $p + k(x_i) = y_i$, $i = 0, ..., k$. We look for $p_{k+1} \in \mathbb{P}^{k+1}$ of the form

$$p_{k+1}(x) = p_k(x) + \underbrace{c_{k+1}(x - x_0)(x - x_1) \cdot ... \cdot (x - x_k)}_{\in \mathbb{P}^{k+1}},$$

where c_{k+1} is to be determined. Note that

$$p_{k+1}(x_i) = p_k(x_i) = y_i, \qquad i = 0, ..., k.$$

Therefore, we determine c_{k+1} imposing the last condition

$$y_{k+1} = p_{k+1}(x_{k+1}) = p_k(x_{k+1}) + c_{k+1}(x_{k+1} - x_1) \cdot ... \cdot (x_{k+1} - x_k)$$

i.e.

$$c_{k+1} = \frac{y_{k+1} - p_k(x_{k+1})}{(x_{k+1} - x_1) \cdot ... \cdot (x_{k+1} - x_k)}.$$

Example 2.1 (Newton form)**.** *Find $p \in \mathbb{P}^3$ interpolating*

$$p(i) = 1/i, \qquad i = 1, 2, 3, 4.$$

The initialization of the algorithm consists in setting $p_0(x) = 1$. We then look for $p_1(x) = 1 + c_1(x - 1)$ using the second interpolation point 2:

$$1/2 = p_1(2) = 1 + c_1(2 - 1), \quad c_1 = -1/2 \quad \Longrightarrow \quad p_1(x) = 1 + \frac{1}{2}(x - 1).$$

Similarly for the third interpolation point: $p_2(x) = p_1(x) + c_2(x - 1)(x - 2)$ such that

$$1/3 = p_2(3) = 1 - \frac{1}{2}(2) + c_2(2)(1), \quad c_2 = \frac{1}{6} \quad \Longrightarrow \quad p_2(x) = 1 - \frac{1}{2}(x-1) + \frac{1}{6}(x-1)(x-2).$$

Finally, using the last interpolation point: $p_3(x) = p_2(x) + c_3(x - 1)(x - 2)(x - 3)$

$$1/4 = p_3(4) = 1 - \frac{3}{2} + \frac{1}{6}(3 \cdot 2) + c_3(3 \cdot 2 \cdot 1), \quad c_3 = -\frac{1}{24},$$

which leads to the desired polynomial

$$p_3(x) = 1 - \frac{1}{2}(x - 1) + \frac{1}{6}(x - 1)(x - 2) - \frac{1}{24}(x - 1)(x - 2)(x - 3).$$

You can plot this polynomial in matlab

```
x=.5:0.05:4;
y=1-(x-1)/2+times(x-1,x-2)/6 - times(x-1,times(x-2,x-3));
plot(x,y,x,rdivide(1,x));
```

3. Lecture 3

3.1. Newton form in *matlab*. Recall that the Newton's form algorithm proceeds recursively.

(1) $p_0(x) = y_0$;
(2) $p_{k+1}(x) = p_k(x) + c_{k+1}(x - x_0) \cdot ... \cdot (x - x_k)$, where

$$c_{k+1} = \frac{y_{k+1} - p_k(x_k)}{(x_{k+1} - x_0) \cdot ... \cdot (x_{k+1} - x_k)}.$$

We will need three *matlab* functions.

```
function c=CGEN(n,x,y)
```

which generates $c_0, ..., c_n$ from $x_0, ..., x_n$ and $y_0, ..., y_n$,

```
function Px=EVAL(n,c,x,y0,t)
```

which computes $p_n(t)$ given $c = (c_0, ..., c_n), x = (x_0, ..., x_n)$, $y0 = y_0$ and

```
function PLOTINTP(x0,xf,NP,c,x,n,y0,FN),
```

which plots $p(x)$ and the interpolated function, where $x0, xf$ are the plot limits, NP is the number of points used for plotting, c, x, n are as above and FN is the analytic function interpolated.

Notice that *matlab* array indices start at 1! Therefore, we rewrite the Newton form algorithm with index starting at 1.

Newton Form with Shifted index: Given $x_1, ..., x_{n+1}$ distincts and $y_1, ..., y_{n+1}$ find $p_{n+1} \in \mathbb{P}^n$ satisfying

$$p_{n+1}(x_j) = y_j, \qquad j = 1, ..., n+1.$$

(1) $p_1(x) = y_1$;
(2) $p_{k+1}(x) = p_k(x) + c_k(x - x_1) \cdot ... \cdot (x - x_k)$, where

$$c_k = \frac{y_{k+1} - p_k(x_{k+1})}{(x_{k+1} - x_1) \cdot ... \cdot (x_{k+1} - x_k)}.$$

We now provide the *matlab* code for the $CGEN$ routine

```
function c=CGEN(n,x,y)
        for j=1:n    % compute c(j)
        % compute the denominator of c(j)
        PROD=1.0;
        for I=1:j
                PROD = PROD*(x(j+1)-x(I));
        end
        % compute pj(x(j+1))
        if (j==1)
                pj=y(1);
        else
                pj=EVAL(j-1,c,x,y(1),x(j+1));
        end
        c(j) = (y(j+1)-pj)/PROD;
        end
```

Links to this code and the $PLOTINTP$ code are given on ecampus. These are *matlab* m-file function codes and need to be in the subdirectory where you will run *matlab*. There is one file per function.

Problem 3.1. *Write and debug* EVAL.m*, a* matlab *m-file code for the function* EVAL *above.*

Problem 3.2. *For* $n = 2, 3, 4, 5, 6$ *define* $\{x_0 = 1, x_1, x_2, ..., x_n = 4\}$ *with* x_0,...,x_n *uniformly spaced on* $[1, 4]$*. Using the above routines, for each* n*, compute c for the polynomial in* $\mathbb{P}^n$ *interpolating* e^x *and plot on* $[0, 5]$ *using* $NP = 200$*. Report the* $L-$*infinity error computed by* $PLOTINTP$ *using the call*

```
PLOTINTP(0,5,200,c,x,y0,n,inline('exp(x)'));
```

Handin plots for $n = 2$ *and* $n = 6$*.*

Problem 3.3. *Repeat the above problem using the interpolation interval* $[0, \pi]$ *but instead interpolating* $\sin(x)$*. Plot on* $[0, \pi]$ *with* 200 *points.*

3.2. **Lagrange Form of the Interpolating Polynomial.** We consider again the problem of finding $p \in \mathbb{P}^n$ satisfying

$$p(x_i) = y_i, \qquad i = 0, ..., n.$$

We saw that there is a unique $l_j \in \mathbb{P}^n$ satisfying (fix $j \in \{0, 1, ..., n\}$)

$$l_j(x_i) = \delta_{ij},$$

where

$$\delta_{ij} := \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

is the Kronecker Delta. In fact,

$$l_j(x) = \prod_{i=0, i\neq j}^{n} \frac{(x - x_i)}{(x_j - x_i)}.$$

These *Lagrange* polynomials allows us to solve the interpolation problem easily. Indeed, the interpolant is given by

$$p(x) = \sum_{i=0}^{n} y_i l_i(x)$$

(check it!).

Example 3.1 (Lagrange Polynomials)**.** *Let* $x_0 = 1$, $x_1 = 3/2$ *and* $x_2 = 2$*. Compute* $l_i(x)$ *for* $i = 0, 1, 2$*. They are given by*

$$\begin{aligned}
l_0(x) &= \frac{(x - 3/2)(x - 2)}{(1 - 3/2)(1 - 2)} = 2(x - 3/2)(x - 2) \\
l_1(x) &= \frac{(x - 1)(x - 2)}{(3/2 - 1)(3/2 - 2)} = -4(x - 1)(x - 2) \\
l_2(x) &= \frac{(x - 1)(x - 3/2)}{(2 - 1)(2 - 3/2)} = 2(x - 1)(x - 3/2).
\end{aligned}$$

Example 3.2 (Lagrange Interpolation)**.** *Use l_0, l_1 and l_2 from the previous example to find $p \in \mathbb{P}^2$ satisfying*

$$p(1) = 1, \quad p(2) = -1, \quad p(3/2) = 2.$$

The desired polynomial is directly given by

$$p(x) = 1l_0(x) + 2l_1(x) - 1l_2(x) = 2(x-3/2)(x-2) - 8(x-1)(x-2) - 2(x-1)(x-3/2).$$

4. Lecture 4

The Lagrange form can be used to deduce properties of polynomial interpolation.

We recall that $C[a,b]$ denote the set of continuous functions defined on $[a,b]$. The space $C[a,b]$ is a linear (vector) space, with vector operations given by

$$(f+g)(x) = f(x) + g(x), \qquad x \in [a,b]$$

and

$$(\alpha f)(x) = \alpha f(x), \qquad x \in [a,b]$$

(for all $f, g \in C[a,b]$ and $\alpha \in \mathbb{R}$). The set $\mathbb{P}^k$ is also a linear space (check it!). In addition, for $\{x_0, ..., x_k\} \subset [a,b]$ define

$$L : C[a,b] \to \mathbb{P}^k$$

by Lf to be the polynomial in $\mathbb{P}^k$ interpolating f at x_i, $i = 0, ..., k$.

Lemma 4.1 (Property of L)**.** *The transformation L is a linear transformation, i.e.*

$$L(\alpha f + \beta g) = \alpha L(f) + \beta L(g)$$

for all $\alpha, \beta \in R$ and $f, g \in C[a,b]$.

Proof. Let $\{l_i\}_{i=0}^k$ be the Lagrange polynomials associated with $x_0, ..., x_k$. Then

$$(Lf)(x) = \sum_{i=0}^k f(x_i) l_i(x) \quad \text{and} \quad (Lg)(x) = \sum_{i=0}^k g(x_i) l_i(x).$$

This implies

$$\begin{aligned} L(\alpha f + \beta g) &= \sum_{i=0}^k (\alpha f(x_i) + \beta g(x_i)) l_i(x) = \alpha \sum_{i=0}^k f(x_i) l_i(x) + \beta \sum_{i=0}^k g(x_i) l_i(x) \\ &= \alpha (Lf)(x) + \beta (Lg)(x). \end{aligned}$$

□

Lemma 4.2. *Let $\{l_i\}_{i=0}^n$ be the Lagrange polynomials corresponding to the nodes $\{x_0, x_1, ..., x_n\}$. Then every $p \in \mathbb{P}^n$ reads*

$$p(x) = \sum_{i=0}^n p(x_i) l_i(x).$$

Proof. The polynomial $q \in \mathbb{P}^k$ given by

$$q(x) = \sum_{i=0}^n p(x_i) l_i(x)$$

interpolates $p(x)$. Since p trivially interpolates itself, the unicity of the interpolant implies that $q = p$. □

Remark 4.1 (Polynomial Integration). Using the above representation lemma, we directly deduce an integration formula working simultaneously for all polynomials of degree n:

$$\int_a^b p(x) dx = \int_a^b \sum_{i=0}^n p(x_i) l_i(x) = \sum_{i=0}^n p(x_i) A_i,$$

where $A_i := \int_a^b l_i(x) dx$.

The next theorem is central to derive approximation properties of the interpolant.

Theorem 4.1 (Inteprolation Estimates)**.** *Assume that* $f \in C^{n+1}[a,b]$ *and* $p \in \mathbb{P}^n$ *interpolates* f *at* $\{x_0, ..., x_n\} \subset [a,b]$ *(distincts). To each* $x \in [a,b]$*, there is a* $\xi_x \in (a,b)$ *such that*

$$f(x) - p(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi_x) \prod_{i=1}^{n} (x - x_i).$$

Proof. If $x = x_i$ for some $i = 0, ..., n$ then the assertion is trivial. Therefore, we assume that $x \neq x_i$. Set

$$w(t) = (t - x_0)(t - x_1) \cdot ... \cdot (t - x_n)$$

and

$$\phi(t) = f(t) - p(t) - \lambda w(t), \tag{3}$$

with

$$\lambda = \frac{f(x) - p(x)}{w(x)}.$$

Note that $\phi \in C^{n+1}[a,b]$, $\phi(x_i) = 0$, $i = 0, ..., n$ and $\phi(x) = 0$. Hence, ϕ has $n+2$ distinct roots. Rolles theorem implies that ϕ' has at least $n+1$ distinct roots. Repeating the argument: ϕ'' has at least n roots,..., $\phi^{(n+1)}$ has at least 1 root, denoted $\xi_x \in (a,b)$. Differentiating (3) $n+1$ times, we get

$$0 = \phi^{(n+1)}(\xi_x) = f^{(n+1)}(\xi_x) - \underbrace{\frac{d^{n+1}}{dt^{n+1}} p(t)}_{=0} |_{t=\xi_x} - \lambda \underbrace{\frac{d^{n+1}}{dt^{n+1}} w(t)}_{=(n+1)!} |_{t=\xi_x}.$$

In view of the definition of λ, this simplifies to

$$f(x) - p(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi_x) \prod_{i=1}^{n} (x - x_i),$$

which is the desired result. ☐

Let us now see if we can try to optimize $(x_0, ..., x_n)$ distinct such that

$$\max_{x \in [a,b]} |(x - x_0)...(x - x_n)| \leq \max_{x \in [a,b]} |(x - y_0)...(x - y_n)|$$

for any $y_0, ..., y_n$ distinct. Such points would make the right hand side the smallest possible (in magnitude). We need to choose $x_0, ..., x_n$ so that

$$q(x) = (x - x_0)...(x - x_n)$$

has minimal absolute value on $[a,b]$.

Suppose that $[a,b] = [-1,1]$. If $p(x)$ has all distinct real roots in $[x_0, x_n]$ (reorder the roots such that $x_0 < x_1 < ... < x_n$), then $p(x)$ is monotone (increasing or decreasing) for $x > x_n$ and $x < x_0$. One might guess that all of the roots should be in $[-1,1]$ to achieve the minimal absolute value.

To gain more insight, consider $n = 0$. In that case,

$$\max_{x \in [-1,1]} |x - x_0| = \begin{cases} 1 + x_0 & \text{if} x_0 \geq 0, \\ 1 - x_0 & \text{if} x_0 \leq 0 \end{cases}$$

and the minimum occurs at $x_0 = 0$.

Next consider the quadratic case. Intuitively, the two points should be symmetric, i.e. $x_1 = -x_0$ and so

$$q(x) = (x - x_0)(x + x_0) = x^2 - x_0^2.$$

Hence,

$$\max_{x \in [-1,1]} |(x - x_0)(x + x_0)| = \max\{|q(0)|, |q(1)|\} = \max\{x_0^2, 1 - x_0^2\}.$$

The optimum choice satisfies $x_0^2 = 1 - x_0^2$, i.e. $1 = 2x_0^2$ or $x_0 = \sqrt{1/2}$. With this choice, $q(x_0) = \frac{1}{2} = 1 - x_0^2$.

In the two examples, q has $n+1$ extreme points with equal magnitude. We need a polynomial $T_n(x)$ with n zeros in $[-1, 1]$ and $n + 1$ extrema's (with oscillating signs). More later.

5. Lecture 5

The cosine function has a lots of zeros and alternating extrema but unfortunately, it is not a polynomial.

5.1. **Chebyschev Polynomials.** Define for $x \in [-1, 1]$ and integer $n \geq 0$

$$T_n(x) := \cos(n \cos^{-1}(x)).$$

Recall that

$$\cos^{-1} : [-1, 1] \to [0, \pi]$$

so that

$$n \cos^{-1} : [-1, 1] \to [0, n\pi].$$

As a consequence, $T_n(x)$ has $n+1$ extrema oscillating between ± 1. Moreover, $T_n(x)$ has n zeros. Less obvious, $T_n(x)$ is actually a polynomial. Indeed,

$$T_0(x) = \cos(0) = 1$$

$$T_1(x) = \cos(\cos^{-1}(x)) = x.$$

Also,

$$T_{n+1}(x) = \cos((n+1)\theta), \quad \theta = \cos^{-1}(x)$$

and so

$$T_{n+1}(x) = \cos(\theta)\cos(n\theta) - \sin(\theta)\sin(n\theta).$$

Similarly

$$T_{n-1}(x) = \cos(\theta)\cos(n\theta) + \sin(\theta)\sin(n\theta)$$

and therefore

$$T_{n+1}(x) + T_{n-1}(x) = 2\cos(\theta)\cos(n\theta) = 2xT_n(x).$$

We just proved a recurrence formula

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x),$$

which implies

(1) $T_n(x)$ is a polynomial of degree n;
(2) The leading coefficient for $T_n(x)$ is $2^{n-1}x^n$ for $n \geq 1$;
(3) For odd n, $T_n(x)$ is an odd function (contain only x^j with j odd) while for even n, $T_n(x)$ is even.

The roots of T_n are related to the zeros of $\cos(n\theta)$, $\theta \in [0, \pi]$ ($\theta = \cos^{-1}(x)$). Since $\cos(y) = 0$ for $y = \pi(j + \frac{1}{2})$, the roots x_j of $T_n(x)$ are such that

$$n \cos^{-1}(x_j) = \pi(j + \frac{1}{2})$$

or

$$x_j = \cos\left(\frac{\pi}{n}(j + \frac{1}{2})\right), \qquad j = 0, 1, ..., n-1.$$

Example 5.1 ($n = 1$)**.** *When $n = 1$, $T_1(x) = x$ and $x_0 = 0$.*

Example 5.2 ($n = 2$)**.** *When $n = 2$, $T_2(x) = 2x^2 - 1$ and $x_0 = -\frac{1}{\sqrt{2}}$, $x_1 = \frac{1}{\sqrt{2}}$.*

Example 5.3 ($n = 3$)**.** *When $n = 3$, $T_3(x) = 4x^3 - 3x$ and $x_0 = -\sqrt{\frac{3}{4}}$, $x_1 = 0$, $x_2 = \sqrt{\frac{3}{4}}$.*

The following theorem provides a share estimate on the interpolation error (see Theorem 4.1).

Theorem 5.1 (Interpolation Error)**.** *Given n, let $x_0, ..., x_n$ be the roots of $T_{n+1}(x)$, i.e.*

$$x_j = \cos\left(\frac{\pi}{n+1}(j+\frac{1}{2})\right), \qquad j = 0, .., n.$$

Let $\{y_0, ..., y_n\} \subset \mathbb{R}^{n+1}$, then

$$m_x := 2^{-n} = \max_{x\in[-1,1]} \left|\prod_{i=0}^{n}(x - x_i)\right| \leq \max_{x\in[-1,1]} \left|\prod_{i=0}^{n}(x - y_i)\right| =: m_y.$$

Proof. We proceed by contradiction. Suppose the theorem does not hold. Then, there exists $\{y_0, ..., y_n\}$ with $m_y < m_x$. Set

$$P(x) := \prod_{i=0}^{n}(x - x_i) = 2^{-n}T_{n+1}(x),$$

where we used the fact that x_i are the roots of $T_{n+1}(x)$. Also, we set

$$Q(x) := \prod_{i=0}^{n}(x - y_i) \qquad \text{and} \qquad R(x) := P(x) - Q(x).$$

Now, $m_x = 2^{-n}$ since $T_{n+1}(x)$ oscillate between -1 and 1. Moreover, $m_y < m_x$ implies that $R(x)$ has the same sign as $P(x)$ at each extrema of $P(x)$ (which coincide with those of $T_{n+1}(x)$).

There are $n + 2$ extrema with oscillating signs. Therefore $R(x)$ has oscillating signs at the extrema of $T_{n+1}(x)$. The intermediate value theorem implies that there is a root of $R(x)$ between each pair (of oscillating signs). In turn, $n + 2$ extrema implies that $R(x)$ has $n + 1$ roots. Note however, that both $P(x)$ and $Q(x)$ are monic so their difference, i.e., $R(x)$ is a polynomial of degree n. The only polynomial of degree n with $n + 1$ distinct roots is the zero polynomial, which implies that $P(x) = Q(x)$. This contradicts $m_y < m_x$. ☐

Example 5.4 (Error Theorem)**.** *Let $x_0, ..., x_n$ be distinct in $[0, \pi]$ and $p \in \mathbb{P}^n$ interpolate $\sin(x)$ at $x_0, ..., x_n$. Derive a bound for the error $|\sin(x) - p(x)|$ for $x \in [0, \pi]$. According to Theorem 4.1, we have*

$$|\sin(x) - p(x)| = \frac{|f^{(n+1)}(\xi_x)|}{(n+1)!}\left|\prod_{i=0}^{n}(x - x_i)\right|.$$

Without information on the distribution of $\{x_i\}$

$$\left|\prod_{i=0}^{n}(x - x_i)\right| = \prod_{i=0}^{n}|(x - x_i)| \leq \pi^{n+1}$$

and $|f^{(n+1)}(\xi_x)| \leq 1$ since

$$|f^{(n+1)}(\xi_x)| = \begin{cases} |\cos(\xi_x)| & \textit{when } n+1 \textit{ is odd,} \\ |\sin(\xi_x)| & \textit{when } n+1 \textit{ is even.} \end{cases}$$

As a consequence, we obtain

$$|\sin(x) - p(x)| \leq \frac{\pi^{n+1}}{(n+1)!} \to 0 \qquad \textit{when } n \to 0.$$