AE 370: HW4

Linyi Hou

February 27, 2020

Problem 1

a.- c. See figures and code attached in Appendices A and B, respectively.

d. Global approximation using equispaced points did not work well because we are using a polynomial to approximate a non-polynomial function. By increasing the number of points where the approximation must match the exact function value, we introduce large errors between those sample points, therefore causing the error to grow.

Global approximation using Chebyshev points worked well because the linear system becomes trivial to solve in the Lagrange basis, where the A matrix in $Ac = f$ simply becomes the identity matrix. This guarantees the solution $c = f$. Furthermore, Chebyshev points better constrain the function near the ends of the approximation bounds, which loosens the behavior of the approximation polynomial in the domain.

The trapezoid method also worked well since it is a local approximation method with first order approximations on each segment of the function. This means is is not susceptible to the errors caused by attempting global approximation using a singular function.

Problem 2

a. We rewrite the integral as a piece-wise function:

$$\int_a^b f(x)dx = \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} f(x)dx \tag{1}$$

b. Next, for each $f(x)$ where $x \in [x_{j-1}, x_j]$:

$$f(x) \approx f(x_j - 1) \tag{2}$$

Substituting back into the summation:

$$\sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} f(x_{j-1})dx = \sum_{j=2}^{n+1} (x_j - x_{j-1}) f(x_{j-1}) \tag{3}$$

c. If we assume $(x_j - x_{j-1}) = \Delta x$ for $j \in [2, n+1]$:

$$\sum_{j=2}^{n+1} (x_j - x_{j-1}) f(x_{j-1}) = \Delta x \sum_{j=2}^{n+1} f(x_{j-1}) = T(\Delta x) \tag{4}$$

d. We define the error as

$$\begin{aligned}
E(\Delta x) &= \left| \int_a^b f(x)dx - T(\Delta x) \right| && (5)\\
&= \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} f(x)dx - \Delta x \sum_{j=2}^{n+1} f(x_{j-1}) \right| && (6)\\
&= \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} f(x)dx - \Delta x f(x_{j-1}) \right| && (7)\\
&= \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} f(x) - f(x_{j-1})dx \right| && (8)
\end{aligned}$$

The Taylor series expansion of $f(x_{j-1})$ about x is:

$$f(x_{j-1}) = f(x) + f'(x)(x - x_{j-1}) + f''(x)(x - x_{j-1})^2 + \text{H.O.T.} \tag{9}$$

The error then becomes

$$\begin{aligned}
E(\Delta x) &= \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} f(x) - f(x_{j-1})dx \right| && (10)\\
&= \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} \left(f'(x)(x - x_{j-1}) + f''(x)(x - x_{j-1})^2 + \text{H.O.T.} \right) dx \right| && (11)
\end{aligned}$$

Since $|x - x_{j-1}| \le \Delta x$, $f'(x) \le \max(f'(x^*))$:

$$\begin{aligned}
E(\Delta x) &\le \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} \max(f'(x^*)) \Delta x dx \right| && (12)\\
&= \left| \sum_{j=2}^{n+1} \max(f'(x^*)) \Delta x^2 \right| && (13)\\
&= \left| \frac{b-a}{\Delta x} \max(f'(x^*)) \Delta x^2 \right| && (14)\\
&= O(\Delta x) && (15)
\end{aligned}$$

Problem 3

See figures and code attached in Appendices A and B, respectively.

Problem 4

a.- c. See figures and code attached in Appendices A and B, respectively.

Appendix A: Figures

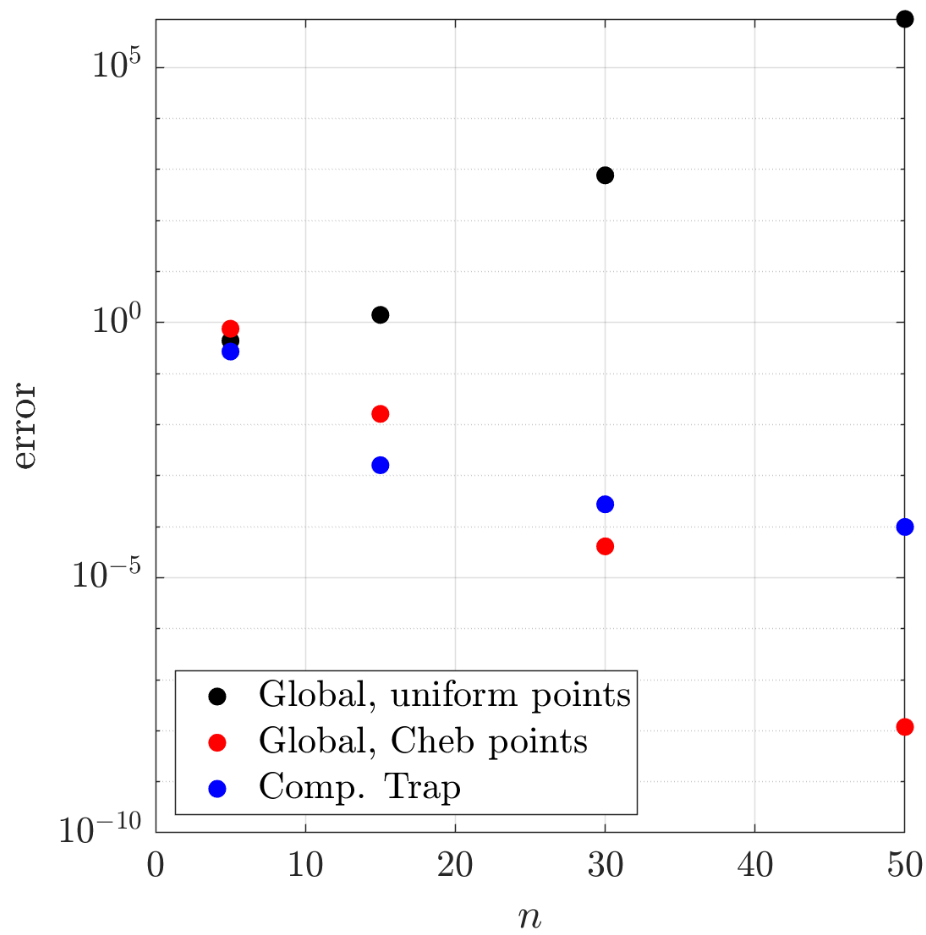

Figure 1: Error Graph for Problem 1

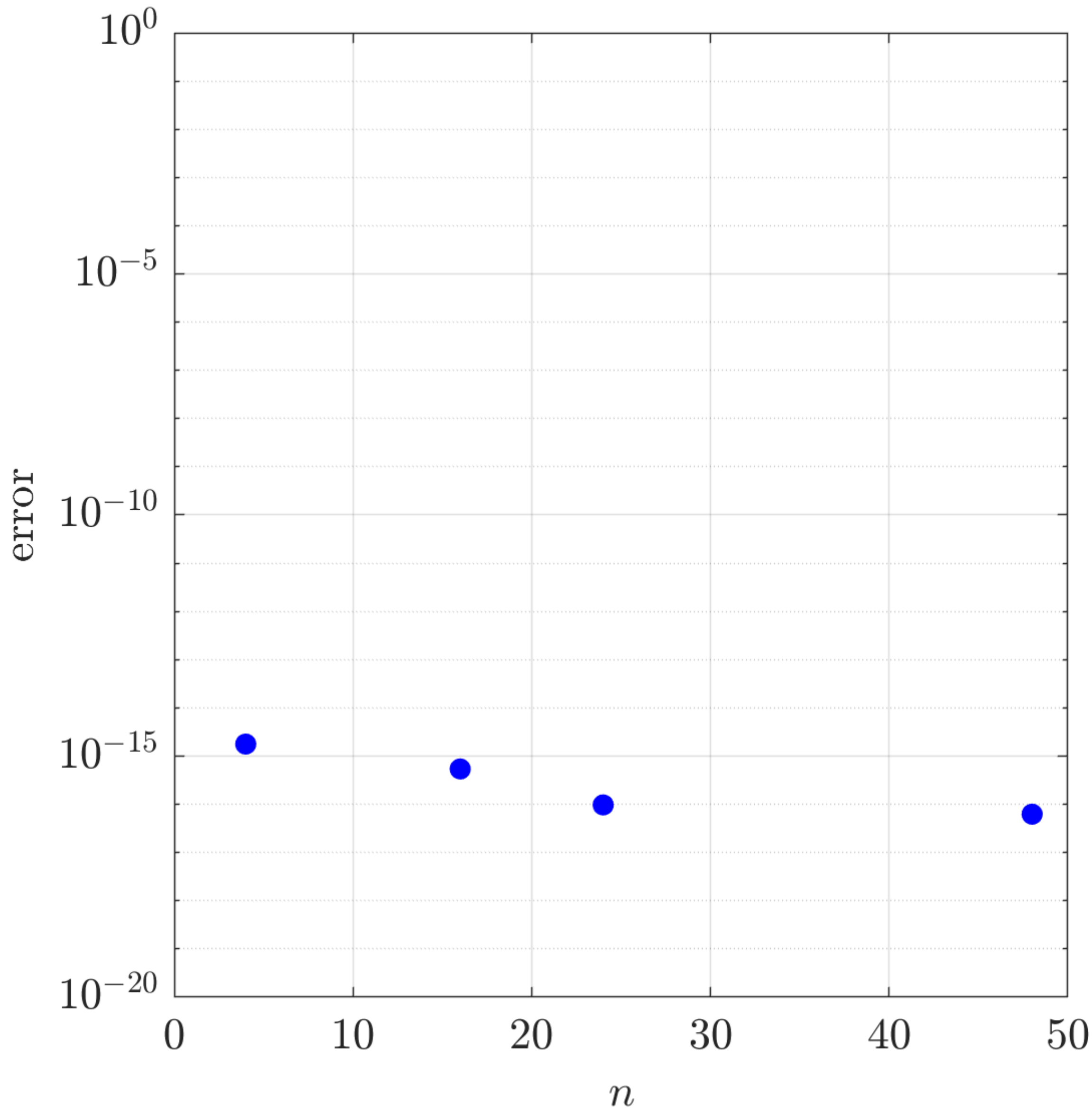

Figure 2: Error Graph for Problem 3

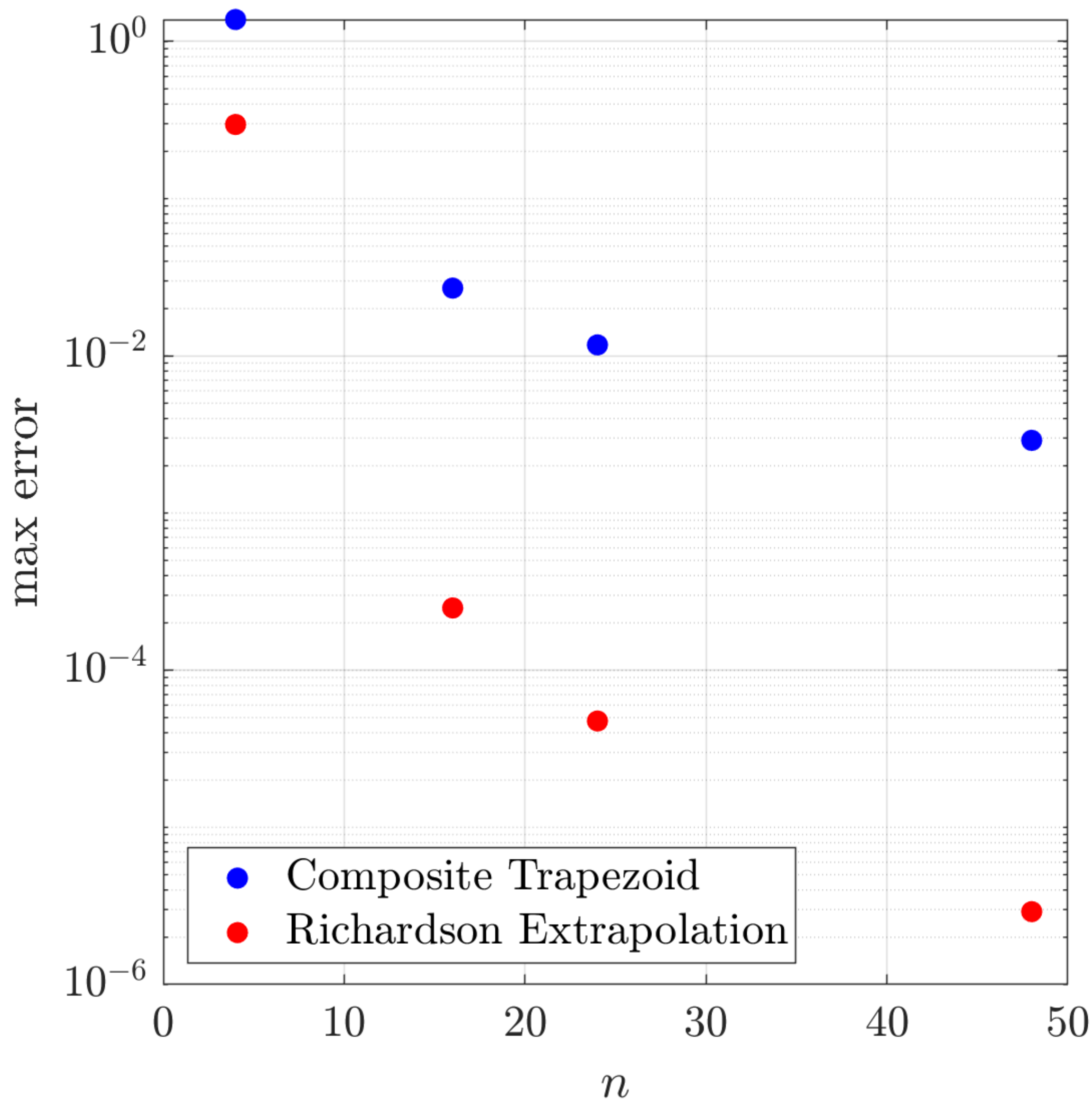

Figure 3: Error Graph for Problem 4

Appendix B: Code

```
%% Problem 1
close all hidden
clear;clc
format long

% a) Equispaced Points with Lagrange Basis

nvect = [5, 15, 30, 50];

fcn = @(x) 1 ./ (1 + x.^2);

%exact integral
xs = sym( 'x' );
%Can use int command to analytically integrate fcn over domain [-5,5]
int_exact = int( fcn( xs ), xs, -5, 5 );

err = zeros( length( nvect ), 1 );

for j = 1 : length( nvect )

    n = nvect( j );

    xj = linspace(-5,5,n+1);

    %define Lagrange basis vectors
    intval = 0;
    for i = 1 : n + 1
        L_i = 1;

        %vector indexing can't start at zero, so go from 1 to n+1
        for k = 1 : n + 1
            if k ~= i
                L_i = ( xs - xj( k ) )./( xj( i ) - xj( k ) ) .* L_i;
            end
        end

        %Again, use int (this time to compute integral of Lagrange fcn)
        Li_int = int( L_i, xs, -5, 5 );
```

```

        %Update integral
        intval = intval + Li_int * fcn(xj(i));

    end

    err( j ) = abs( int_exact - intval );

end

%plot error
figure(100)
semilogy( nvect, err, 'k.', 'markersize', 26 )
grid on


% b) Chebyshev Points with Lagrange Basis

err = zeros( length( nvect ), 1 );

for j = 1 : length( nvect )

    n = nvect( j );

    xj = 5*cos( (0:n)*pi/n );

    %define Lagrange basis vectors
    intval = 0;
    for i = 1 : n + 1
        %How to get Li???
        L_i = 1;

        %vector indexing can't start at zero, so go from 1 to n+1
        for k = 1 : n + 1
            if k ~= i
                L_i = ( xs - xj( k ) )./( xj( i ) - xj( k ) ) .* L_i;
            end
        end

        Li_int = int( L_i, xs, -5, 5 );
```

```
        intval = intval + Li_int * fcn(xj(i));

    end

    err( j ) = abs( intval - int_exact );

end

%plot error
figure(100), hold on
semilogy( nvect, err, 'r.', 'markersize', 26 )
grid on


% c) Equispaced Points with Trapezoid Rule

err = zeros( length( nvect ), 1 );

for j = 1 : length( nvect )

    n = nvect( j );

    xj = linspace(-5,5,n+1);

    dx = xj( 2 ) - xj( 1 ); %spacing between points

    %do end points first
    intval = dx / 2 * (fcn( xj( 1 ) ) + fcn( xj( n+1 ) ));

    %remaining points
    for jj = 2 : n

        intval = intval + dx * fcn( xj( jj ) );

    end

    err( j ) = abs( intval - int_exact );

end
```

```
%plot error
figure(100), hold on
semilogy( nvect, err, 'b.', 'markersize', 26 )
grid on


%make plot pretty
h = legend( 'Global, uniform points', 'Global, Cheb points', 'Comp. Trap'
    );
set( h, 'location', 'SouthWest', 'interpreter', 'latex', 'fontsize', 16)
xlabel( '$n$', 'interpreter', 'latex', 'fontsize', 16)
ylabel( 'error', 'interpreter', 'latex', 'fontsize', 16)

set(gca, 'TickLabelInterpreter','latex', 'fontsize', 16 )

set(gcf, 'PaperPositionMode', 'manual')
set(gcf, 'Color', [1 1 1])
set(gca, 'Color', [1 1 1])
set(gcf, 'PaperUnits', 'centimeters')
set(gcf, 'PaperSize', [15 15])
set(gcf, 'Units', 'centimeters' )
set(gcf, 'Position', [0 0 15 15])
set(gcf, 'PaperPosition', [0 0 15 15])

svnm = 'error_compare';
print( '-dpng', svnm, '-r200' )


%% Problem 3

close all hidden
clear;clc
format long

nvect = [4, 16, 24, 48];

fcn = @(x) sin(10.*pi*x);

%exact answer
```

```
xs = sym( 'x' );
int_exact = int( fcn( xs ), xs, 0, 2 );

% a) Equispaced Points with Trapezoid Rule

err = zeros( length( nvect ), 1 );
% intvalh = zeros( length( nvect ), 1 );

for j = 1 : length( nvect )

    n = nvect( j );

    dx = 2/n; %spacing between points

    xj = 0 : dx : 2;

    %do end points first
    intval = dx / 2 * (fcn( xj( 1 ) ) + fcn( xj(n+1) ));

    %remaining points
    for jj = 2 : n

        intval = intval + dx * fcn( xj( jj ) );

    end

    err( j ) = abs( int_exact - intval );

end

%plot error
figure(100)
semilogy( nvect, err, 'b.', 'markersize', 26 )
grid on
set(gca, 'TickLabelInterpreter','latex', 'fontsize', 16 )
ylim([10^(-20) 1])
xlabel( '$n$', 'interpreter', 'latex', 'fontsize', 16)
ylabel( 'error', 'interpreter', 'latex', 'fontsize', 16)

set(gcf, 'PaperPositionMode', 'manual')
```

```
set(gcf, 'Color', [1 1 1])
set(gca, 'Color', [1 1 1])
set(gcf, 'PaperUnits', 'centimeters')
set(gcf, 'PaperSize', [15 15])
set(gcf, 'Units', 'centimeters' )
set(gcf, 'Position', [0 0 15 15])
set(gcf, 'PaperPosition', [0 0 15 15])

svnm = 'error_q3';
print( '-dpng', svnm, '-r200' )


%% Problem 4

close all hidden
clear;clc
format long

%Assumes a vector intvalh, of length length(nvect), is computed from
%part a) using the Trapezoid rule with length h between points. Part b)
%will compute the trapezoid rule at length h/2 and combine the two to
%produce an improved estimate of the integral.

nvect = [4, 16, 24, 48];

fcn = @(x) (x.^2).*sin(10.*x);

%exact answer
xs = sym( 'x' );
int_exact = int( fcn( xs ), xs, 0, 2 );


% a) Equispaced Points with Trapezoid Rule

err = zeros( length( nvect ), 1 );
intvalh = zeros( length( nvect ), 1 );

for j = 1 : length( nvect )

    n = nvect( j );
```

```
    dx = 2/n; %spacing between points

    xj = 0 : dx : 2;

    %do end points first
    intval = dx / 2 * (fcn( xj( 1 ) ) + fcn( xj( n + 1 ) ));

    %remaining points
    for jj = 2 : n

        intval = intval + dx * fcn( xj( jj ) );

    end

    err( j ) = abs( int_exact - intval );
    intvalh( j ) = intval;

end

%plot error
figure(100)
semilogy( nvect, err, 'b.', 'markersize', 26 )


% b) Richardson Extrapolation by adding to Part a)

err = zeros( length( nvect ), 1 );

%--evaluate trap rule at h/2 and combine this with result from a) to get
%  Richardson extrapolated value.
intvalhb2 = zeros( length( nvect ), 1 );

for j = 1 : length( nvect )

    n = nvect( j );

    %h/2
    dx = 2/n / 2; %spacing between points

```

```
    %corresponding x points
    xj = 0 : dx : 2;

    %do end points first
    intvalhb2( j ) = dx / 2 * (fcn( xj( 1 ) ) + fcn( xj( 2*n + 1 ) ));

    %remaining points
    for jj = 2 : 2*n

        intvalhb2( j ) = intvalhb2( j ) + dx * fcn( xj( jj ) );

    end

    %Richardson extrapolated value
    int_rich = ( 4 * intvalhb2( j ) - intvalh( j ) ) / 3;

    err( j ) = abs( int_rich - int_exact );

end


%plot error
figure(100), hold on
semilogy( nvect, err, 'r.', 'markersize', 26 )
grid on

%make plot pretty
h = legend( 'Composite Trapezoid', 'Richardson Extrapolation' );
set( h, 'location', 'SouthWest', 'interpreter', 'latex', 'fontsize', 16)
xlabel( '$n$', 'interpreter', 'latex', 'fontsize', 16)
ylabel( 'max error', 'interpreter', 'latex', 'fontsize', 16)

set(gca, 'TickLabelInterpreter','latex', 'fontsize', 16 )

set(gcf, 'PaperPositionMode', 'manual')
set(gcf, 'Color', [1 1 1])
set(gca, 'Color', [1 1 1])
set(gcf, 'PaperUnits', 'centimeters')
set(gcf, 'PaperSize', [15 15])
set(gcf, 'Units', 'centimeters' )
```

```
set(gcf, 'Position', [0 0 15 15])
set(gcf, 'PaperPosition', [0 0 15 15])

svnm = 'error_q4';
print( '-dpng', svnm, '-r200' )
```