

# 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 it 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:

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

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

$$f(x) \approx f(x_j - 1) \quad (2)$$

Substituting back into the summation:

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

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

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

d. We define the error as

$$E(\Delta x) = \left| \int_a^b f(x) dx - T(\Delta x) \right| \quad (5)$$

$$= \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} f(x) dx - \Delta x \sum_{j=2}^{n+1} f(x_{j-1}) \right| \quad (6)$$

$$= \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} f(x) dx - \Delta x f(x_{j-1}) \right| \quad (7)$$

$$= \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} f(x) - f(x_{j-1}) dx \right| \quad (8)$$

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

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

The error then becomes

$$E(\Delta x) = \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} f(x) - f(x_{j-1}) dx \right| \quad (10)$$

$$= \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} (f'(x)(x - x_{j-1}) + f''(x)(x - x_{j-1})^2 + \text{H.O.T.}) dx \right| \quad (11)$$

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

$$E(\Delta x) \leq \left| \sum_{j=2}^{n+1} \int_{x_{j-1}}^{x_j} \max(f'(x^*)) \Delta x dx \right| \quad (12)$$

$$= \left| \sum_{j=2}^{n+1} \max(f'(x^*)) \Delta x^2 \right| \quad (13)$$

$$= \left| \frac{b-a}{\Delta x} \max(f'(x^*)) \Delta x^2 \right| \quad (14)$$

$$= O(\Delta x) \quad (15)$$

## 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

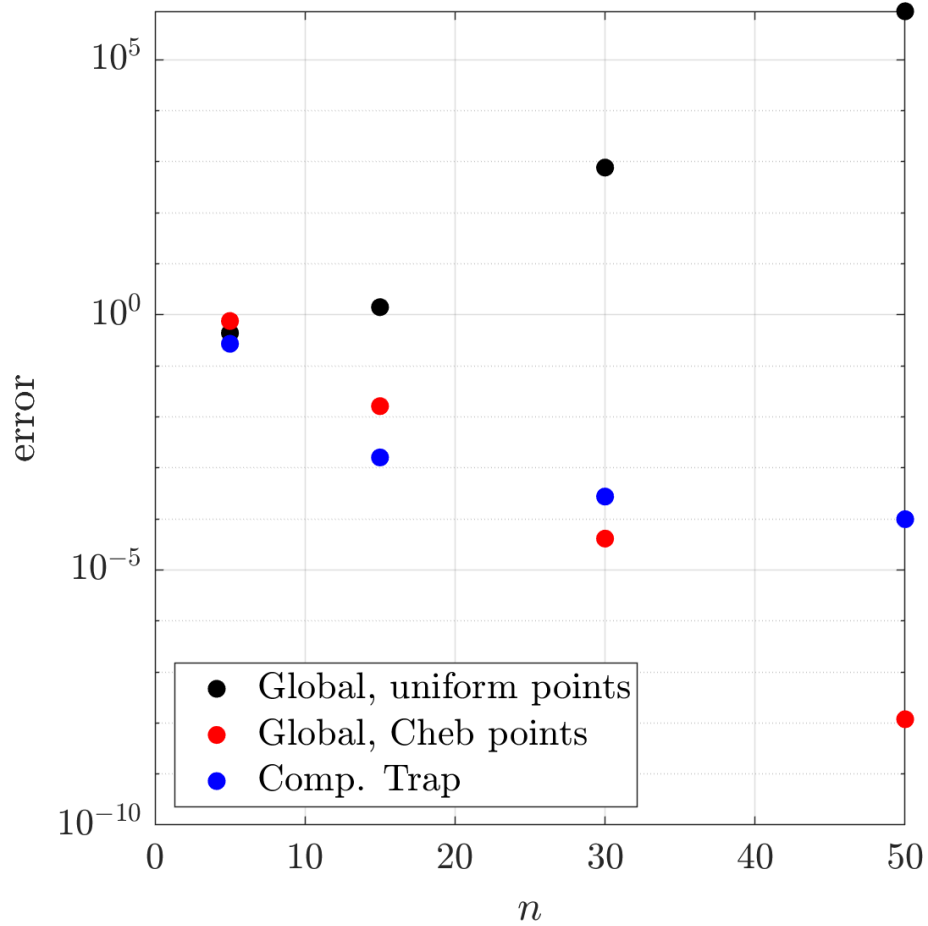


Figure 1: Error Graph for Problem 1

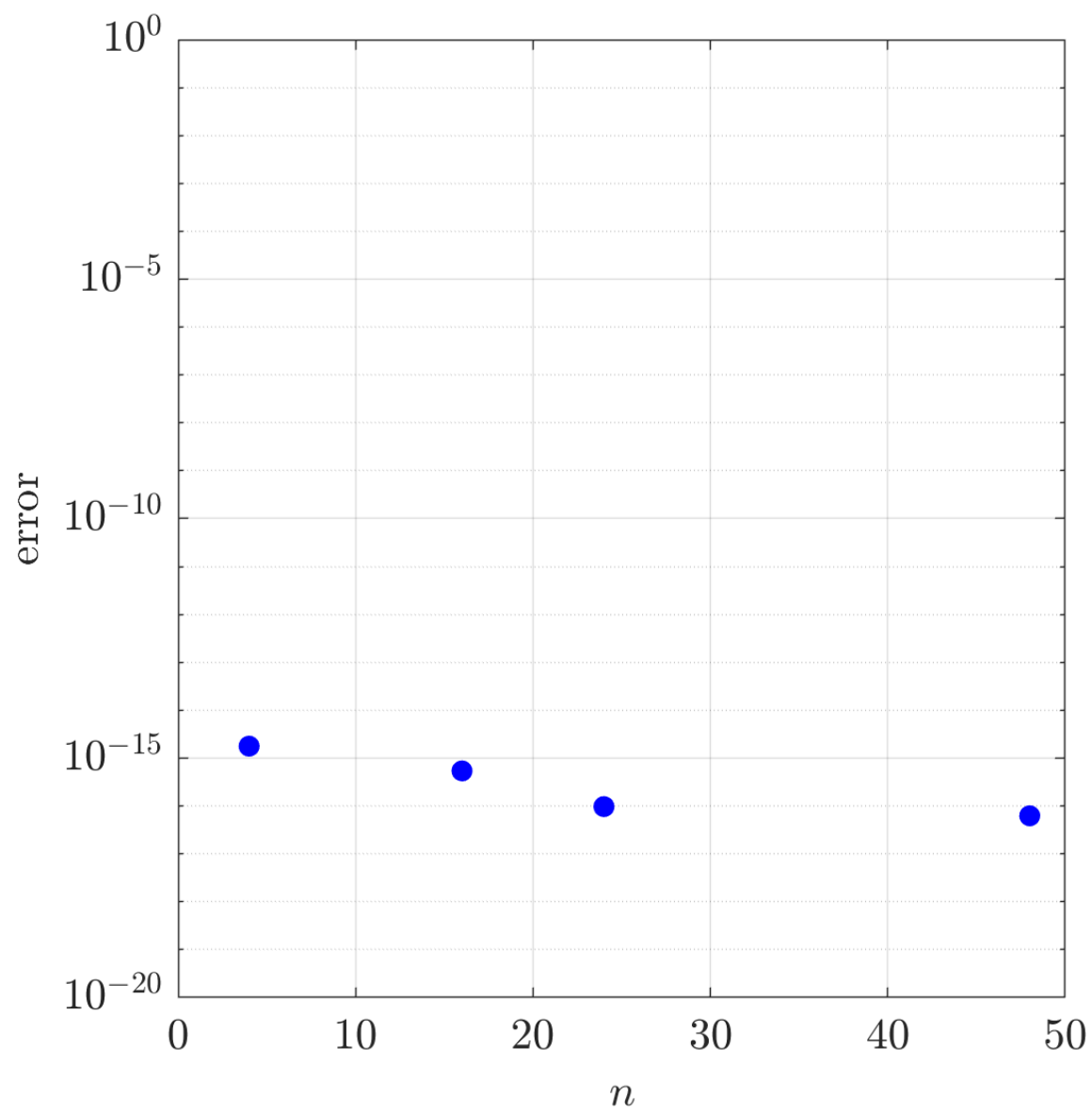


Figure 2: Error Graph for Problem 3

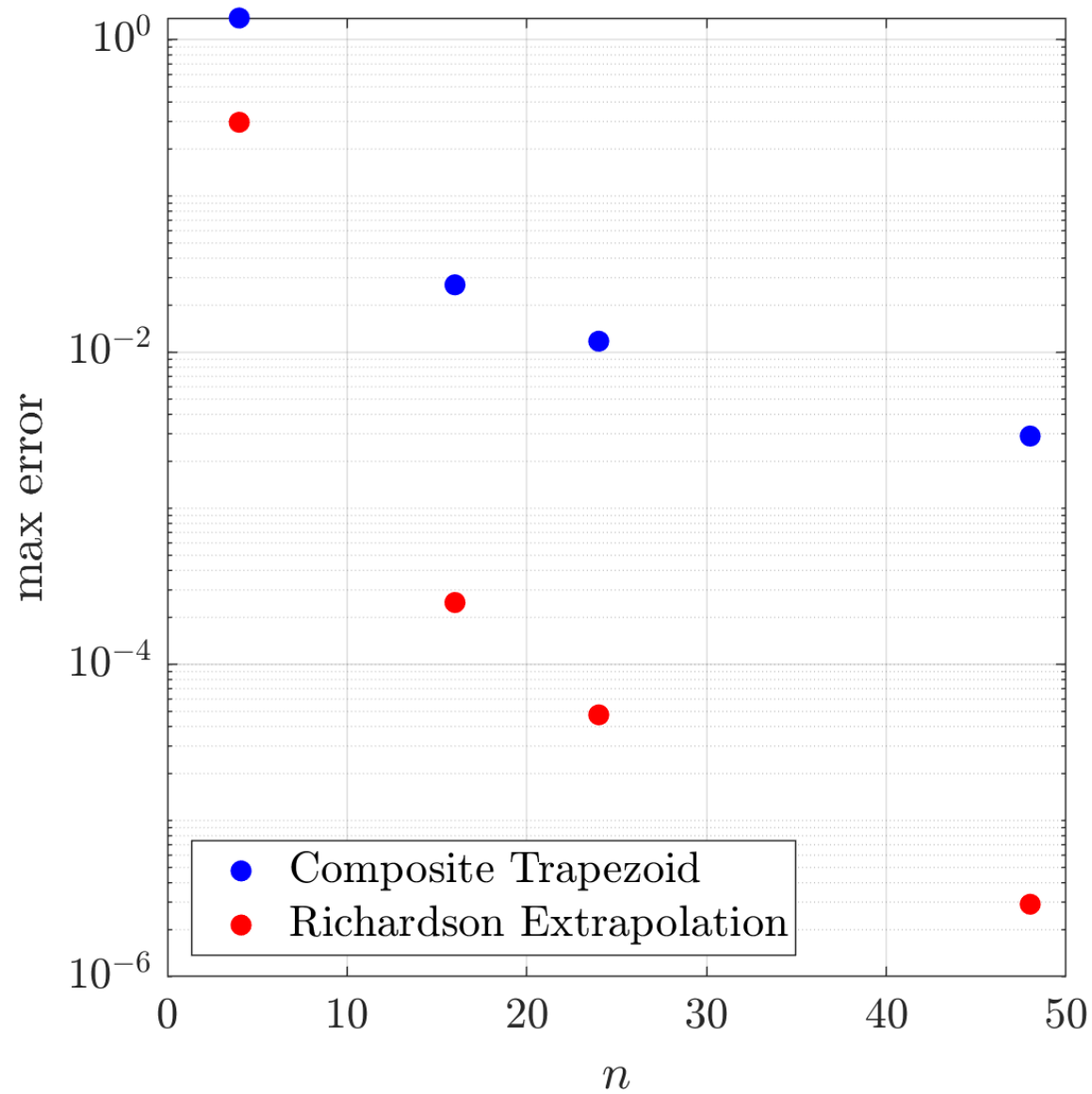


Figure 3: Error Graph for Problem 4

## Appendix B: Code

```
1 %% Problem 1
2 close all hidden
3 clear;clc
4 format long
5
6 % a) Equispaced Points with Lagrange Basis
7
8 nvect = [5, 15, 30, 50];
9
10 fcn = @(x) 1 ./ (1 + x.^2);
11
12 %exact integral
13 xs = sym( 'x' );
14 %Can use int command to analytically integrate fcn over domain [-5,5]
15 int_exact = int( fcn( xs ), xs, -5, 5 );
16
17 err = zeros( length( nvect ), 1 );
18
19 for j = 1 : length( nvect )
20
21     n = nvect( j );
22
23     xj = linspace(-5,5,n+1);
24
25     %define Lagrange basis vectors
26     intval = 0;
27     for i = 1 : n + 1
28         Li = 1;
29
30         %vector indexing can't start at zero, so go from 1 to n+1
31         for k = 1 : n + 1
32             if k ~= i
33                 Li = ( xs - xj( k ) )./( xj( i ) - xj( k ) ) .* Li;
34             end
35         end
36
37         %Again, use int (this time to compute integral of Lagrange fcn)
38         Li_int = int( Li, xs, -5, 5 );
```

```

39
40     %Update integral
41     intval = intval + Li_int * fcn(xj(i));
42
43     end
44
45     err( j ) = abs( int_exact - intval );
46
47 end
48
49 %plot error
50 figure(100)
51 semilogy( nvect, err, 'k.', 'markersize', 26 )
52 grid on
53
54
55 % b) Chebyshev Points with Lagrange Basis
56
57 err = zeros( length( nvect ), 1 );
58
59 for j = 1 : length( nvect )
60
61     n = nvect( j );
62
63     xj = 5*cos( (0:n)*pi/n );
64
65     %define Lagrange basis vectors
66     intval = 0;
67     for i = 1 : n + 1
68         %How to get Li???
69         Li = 1;
70
71         %vector indexing can't start at zero, so go from 1 to n+1
72         for k = 1 : n + 1
73             if k ~= i
74                 Li = ( xs - xj( k ) )./( xj( i ) - xj( k ) ) .* Li;
75             end
76         end
77
78         Li_int = int( Li, xs, -5, 5 );

```

```

79
80         intval = intval + Li_int * fcn(xj(i));
81
82     end
83
84     err( j ) = abs( intval - int_exact );
85
86 end
87
88 %plot error
89 figure(100), hold on
90 semilogy( nvect, err, 'r.', 'markersize', 26 )
91 grid on
92
93
94 % c) Equispaced Points with Trapezoid Rule
95
96 err = zeros( length( nvect ), 1 );
97
98 for j = 1 : length( nvect )
99
100     n = nvect( j );
101
102     xj = linspace(-5,5,n+1);
103
104     dx = xj( 2 ) - xj( 1 ); %spacing between points
105
106     %do end points first
107     intval = dx / 2 * (fcn( xj( 1 ) ) + fcn( xj( n+1 ) ));
108
109     %remaining points
110     for jj = 2 : n
111
112         intval = intval + dx * fcn( xj( jj ) );
113
114     end
115
116     err( j ) = abs( intval - int_exact );
117
118 end

```



```

119
120 %plot error
121 figure(100), hold on
122 semilogy( nvect, err, 'b.', 'markersize', 26 )
123 grid on
124
125
126 %make plot pretty
127 h = legend( 'Global, uniform points', 'Global, Cheb points', 'Comp. Trap'
128            );
129 set( h, 'location', 'SouthWest', 'interpreter', 'latex', 'fontsize', 16)
130 xlabel( '$n$', 'interpreter', 'latex', 'fontsize', 16)
131 ylabel( 'error', 'interpreter', 'latex', 'fontsize', 16)
132
133 set(gca, 'TickLabelInterpreter','latex', 'fontsize', 16 )
134
135 set(gcf, 'PaperPositionMode', 'manual')
136 set(gcf, 'Color', [1 1 1])
137 set(gca, 'Color', [1 1 1])
138 set(gcf, 'PaperUnits', 'centimeters')
139 set(gcf, 'PaperSize', [15 15])
140 set(gcf, 'Units', 'centimeters' )
141 set(gcf, 'Position', [0 0 15 15])
142 set(gcf, 'PaperPosition', [0 0 15 15])
143
144 svnm = 'error_compare';
145 print( '-dpng', svnm, '-r200' )
146
147 %% Problem 3
148
149 close all hidden
150 clear;clc
151 format long
152
153 nvect = [4, 16, 24, 48];
154
155 fcn = @(x) sin(10.*pi*x);
156
157 %exact answer

```

```

158 xs = sym( 'x' );
159 int_exact = int( fcn( xs ), xs, 0, 2 );
160
161 % a) Equispaced Points with Trapezoid Rule
162
163 err = zeros( length( nvect ), 1 );
164 % intvalh = zeros( length( nvect ), 1 );
165
166 for j = 1 : length( nvect )
167
168     n = nvect( j );
169
170     dx = 2/n; %spacing between points
171
172     xj = 0 : dx : 2;
173
174     %do end points first
175     intval = dx / 2 * ( fcn( xj( 1 ) ) + fcn( xj(n+1) ) );
176
177     %remaining points
178     for jj = 2 : n
179
180         intval = intval + dx * fcn( xj( jj ) );
181
182     end
183
184     err( j ) = abs( int_exact - intval );
185
186 end
187
188 %plot error
189 figure(100)
190 semilogy( nvect, err, 'b.', 'markersize', 26 )
191 grid on
192 set(gca, 'TickLabelInterpreter','latex', 'fontsize', 16 )
193 ylim([10^(-20) 1])
194 xlabel( '$n$', 'interpreter', 'latex', 'fontsize', 16)
195 ylabel( 'error', 'interpreter', 'latex', 'fontsize', 16)
196
197 set(gcf, 'PaperPositionMode', 'manual')

```

```

198 set(gcf, 'Color', [1 1 1])
199 set(gca, 'Color', [1 1 1])
200 set(gcf, 'PaperUnits', 'centimeters')
201 set(gcf, 'PaperSize', [15 15])
202 set(gcf, 'Units', 'centimeters' )
203 set(gcf, 'Position', [0 0 15 15])
204 set(gcf, 'PaperPosition', [0 0 15 15])
205
206 svnm = 'error_q3';
207 print( '-dpng', svnm, '-r200' )
208
209
210 %% Problem 4
211
212 close all hidden
213 clear;clc
214 format long
215
216 %Assumes a vector intvalh, of length length(nvect), is computed from
217 %part a) using the Trapezoid rule with length h between points. Part b)
218 %will compute the trapezoid rule at length h/2 and combine the two to
219 %produce an improved estimate of the integral.
220
221 nvect = [4, 16, 24, 48];
222
223 fcn = @(x) (x.^2).*sin(10.*x);
224
225 %exact answer
226 xs = sym( 'x' );
227 int_exact = int( fcn( xs ), xs, 0, 2 );
228
229
230 % a) Equispaced Points with Trapezoid Rule
231
232 err = zeros( length( nvect ), 1 );
233 intvalh = zeros( length( nvect ), 1 );
234
235 for j = 1 : length( nvect )
236
237     n = nvect( j );

```

```

238
239     dx = 2/n; %spacing between points
240
241     xj = 0 : dx : 2;
242
243     %do end points first
244     intval = dx / 2 * (fcn( xj( 1 ) ) + fcn( xj( n + 1 ) ));
245
246     %remaining points
247     for jj = 2 : n
248
249         intval = intval + dx * fcn( xj( jj ) );
250
251     end
252
253     err( j ) = abs( int_exact - intval );
254     intvalh( j ) = intval;
255
256 end
257
258 %plot error
259 figure(100)
260 semilogy( nvect, err, 'b.', 'markersize', 26 )
261
262
263 % b) Richardson Extrapolation by adding to Part a)
264
265 err = zeros( length( nvect ), 1 );
266
267 %—evaluate trap rule at h/2 and combine this with result from a) to get
268 % Richardson extrapolated value.
269 intvalhb2 = zeros( length( nvect ), 1 );
270
271 for j = 1 : length( nvect )
272
273     n = nvect( j );
274
275     %h/2
276     dx = 2/n / 2; %spacing between points
277

```

```

278 %corresponding x points
279 xj = 0 : dx : 2;
280
281 %do end points first
282 intvalhb2( j ) = dx / 2 * ( fcn( xj( 1 ) ) + fcn( xj( 2*n + 1 ) ) );
283
284 %remaining points
285 for jj = 2 : 2*n
286
287     intvalhb2( j ) = intvalhb2( j ) + dx * fcn( xj( jj ) );
288
289 end
290
291 %Richardson extrapolated value
292 int_rich = ( 4 * intvalhb2( j ) - intvalh( j ) ) / 3;
293
294 err( j ) = abs( int_rich - int_exact );
295
296 end
297
298
299 %plot error
300 figure(100), hold on
301 semilogy( nvect, err, 'r.', 'markersize', 26 )
302 grid on
303
304 %make plot pretty
305 h = legend( 'Composite Trapezoid', 'Richardson Extrapolation' );
306 set( h, 'location', 'SouthWest', 'interpreter', 'latex', 'fontsize', 16)
307 xlabel( '$n$', 'interpreter', 'latex', 'fontsize', 16)
308 ylabel( 'max error', 'interpreter', 'latex', 'fontsize', 16)
309
310 set(gca, 'TickLabelInterpreter','latex', 'fontsize', 16 )
311
312 set(gcf, 'PaperPositionMode', 'manual')
313 set(gcf, 'Color', [1 1 1])
314 set(gca, 'Color', [1 1 1])
315 set(gcf, 'PaperUnits', 'centimeters')
316 set(gcf, 'PaperSize', [15 15])
317 set(gcf, 'Units', 'centimeters' )

```

```
318 set(gcf, 'Position', [0 0 15 15])
319 set(gcf, 'PaperPosition', [0 0 15 15])
320
321 svnm = 'error_q4';
322 print( '-dpng', svnm, '-r200' )
```