Week 2

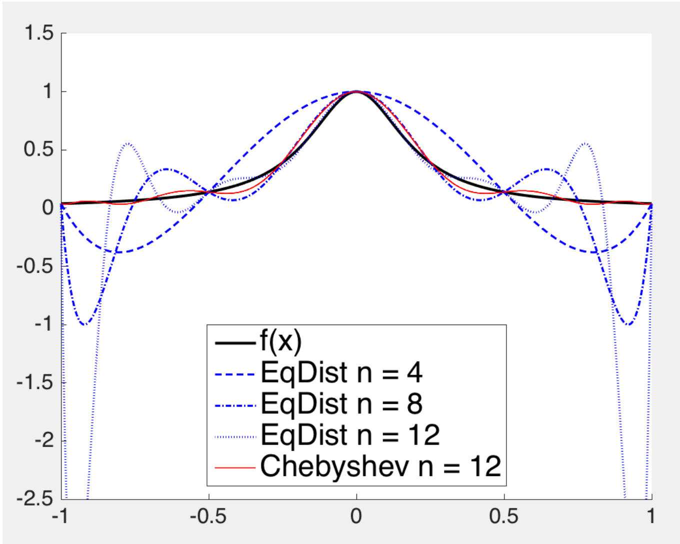
Andy

1 Runge 现象及等距节点插值和 Chebyshev 多项式插值逼近差异（P41/6）

对 $f(x)=\frac{1}{1+25x^2}$ 在 $[0,1]$ 区间内进行等距节点的 4、8、12 阶多项式插值，和 Chebyshev 极值点的 12 阶多项式插值，并对比原函数图像。代码如下：

```
% Week 2 Problem 1
% Runge's Phenomenon
% cheng yu ge
% 12307110079

% The function
f = @( x ) ones( size(x) ) ./ ( 1 + 25 * x .^ 2 ) ;

% Plotting Parameters
xx = -1 :.001:1 ;

figure ;
hold on ;
plot( xx, f( xx ), 'k-', 'Linewidth', 2 ) ;

% polinomial interpolation on Equidistant Nodes
n = 2 * [ 2, 4, 6 ] + 1 ;
linsty = { '--', '-.', ':' } ;
for i = 1 : 3
    x = linspace( -1, 1, n( i ) ) ;
    y = f(x) ;
    [ p, s ] = polyfit( x, y, n(i) - 1 ) ;
    plot( xx, polyval( p, xx, s ), strcat( 'b', linsty{ i } ), 'Linewidth', 1.5 ) ;
end

% polinomial interpolation on Chebyshev Extreme Points
x = -cos( (0 : 12) * pi ./ 12) ;
y = f(x) ;
[ p, s ] = polyfit( x, y, 12 ) ;
plot( xx, polyval( p, xx, s ), 'r-', 'Linewidth', 1 ) ;

h = legend( 'f(x)', 'EqDist n = 4', 'EqDist n = 8', 'EqDist n = 12', 'Chebyshev n = 12' , 'Location', 'South' ) ;
axis( [ -1, 1, -2.5, 1.5 ]) ;
set(h,'Fontsize', 22);
set(gca,'Fontsize', 16);
```

得到的图像如下：



可以看到，随着节点书的增多，等距节点插值越来越不准确（Runge 现象），当 n=12 时等距节点的插值在区间的两端已经出现了非常大的误差；而相对来说同阶的 Chebyshev 多项式插值则仍然有较好的逼近（但也难逃 Runge 现象），出现这样的差异主要因为 Chebyshev 极值点在区间两端附近节点密度增长的速率 $\mathcal{O}(n^2)$ 比等距节点的速度 $\mathcal{O}(n)$ 要快。

2 用 Newton 公式计算 x^n 的积分并展示误差（P43/表 1.4）

对 $f(x) = x^n(n = 1, 2, 3, 4, 5, 6, 7)$ 在 $[0, 1]$ 区间上进行 Newton 法数值积分。代码如下：

```
% Week 2 Problem 2
% Compute Integral of x^n [0,1] by Newton Formula
% cheng yu ge
% 12307110079

% Newton Integral Formuae where n = 0, 1, 2, 3, 4
Midpoint = @(f,a,b) (b - a) * f( (a + b) ./ 2 ) ;
Trapezoidal = @(f,a,b) (b - a)/2 * (f(a) + f(b)) ;
Simpson = @(f,a,b) (b - a)/6 * ( f(a) + 4*f( (a + b) ./ 2 ) + f(b) ) ;
Rule38 = @(f,a,b) (b - a)/8 * ( f(a) + 3*f( (2*a + b) ./ 3 ) ...
    + 3*f( (a + 2*b) ./ 3 ) + f(b) ) ;
Cotes = @(f,a,b) (b - a)/90 * ( 7*f(a) + 32*f( (3*a + b) ./ 4 ) ...
    + 12*f( (a + b) ./ 2 ) + 32*f( (a + 3*b) ./ 4 ) + 7*f(b) ) ;

% show table 1.4 on P43
E = (1 : 7) ;
F = @(x) x .^ E ;
Exact = 1 ./ (E + 1) ;
T = horzcat  (    Exact' , ...
    Midpoint   (F,0,1)' , ...
    Trapezoidal(F,0,1)' , ...
    Simpson    (F,0,1)' , ...
    Rule38     (F,0,1)' , ...
    Cotes      (F,0,1)' ) ;
disp(T);
```

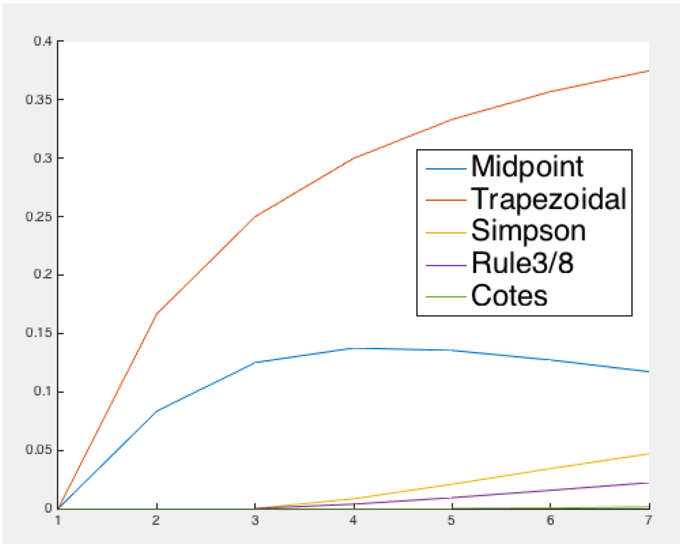
```
% show error
figure ;
hold on ;
for i = 2 : 6
    plot ( E , abs(T(:,i)' - T(:,1)') );
end
h = legend('Midpoint','Trapezoidal','Simpson','Rule3/8','Cotes');
set(h,'Fontsize',22);
```

得到的数据如下：

Table 1: 用 Newton 公式计算 x^n 的积分

x^n	精确值	中点公式	梯形公式	Simpson 公式	$\frac{3}{8}$ 规则	Cotes 公式
x^1	0.5000	**0.5000**	**0.5000**	0.5000	0.5000	0.5000
x^2	0.3333	0.2500	0.5000	**0.3333**	0.3333	0.3333
x^3	0.2500	0.1250	0.5000	**0.2500**	**0.2500**	0.2500
x^4	0.2000	0.0625	0.5000	0.2083	0.2037	**0.2000**
x^5	0.1667	0.0312	0.5000	0.1875	0.1759	**0.1667**
x^6	0.1429	0.0156	0.5000	0.1771	0.1584	0.1432
x^7	0.1250	0.0078	0.5000	0.1719	0.1471	0.1263

表中用粗体标出的数据和精确值吻合，由此可以看出来偶数 n 阶的数值积分公式对 n+1 次多项式也精确成立。误差如下：



3 用 Gauss 求积公式计算 x^n 的积分并展示误差（P49/表 1.5）

由书 48 页公式可计算出 n+1 点的 Gauss 求积公式的节点和相对应权重：

```
function [ x , w ] = LegendreGauss( n )
%LEGENDREGAUSS returns Nodes and Weight of Gauss Integral Formula
```

```
%   input : n , the number of Nodes
%   output: [ x , w ] , Nodes and their Weight
%   NOTE : the integral interval is [ -1 , 1 ] by default,
%          a linear transformation is needed to [ a , b ].

b = 0.5 ./ sqrt( 1 - (2*(1:n)) .^ (-2) );
[ V , Lambda ] = eig( diag(b,1) + diag(b,-1) );
[ x , i ] = sort( diag(Lambda) );
w = 2 * V(1, i) .^ 2 ;
end
```

上式求积区间为 $[0,1]$，进行线性变换后便得到任意区间 $[a,b]$ 上的 Gauss 求积公式：

```
function [ G ] = GaussInt( f , n , a , b )
%GAUSSINT : Integral of f over [a,b] by n point Gauss Integral Formula
%   input : f , n , a , b .
%           f : function ,
%           n : number of nodes ,
%           a , b : interval [ a , b ] .
%   output: G : Gauss Integral

% validating input
if nargin < 2
    n = 4 ;
    warning('n set to 4 by default.');
end
if nargin < 3
    a = -1 ;
    warning('a set to -1 by default.');
end
if nargin < 4
    b = a + 2 ;
    warning('b set to a + 2 by default.');
end

[ x , w ] = LegendreGauss ( n - 1 );
G = (b-a)/2 * w * f( (b-a)/2 .* x + (b+a)/2 ) ;
end
```

由此便可以计算出 x^n 的 k 点 Gauss 求积结果：

```
% Week 2 Problem 3
% Compute Integral of x^n [0,1] by Gauss Formula
% cheng yu ge
% 12307110079

% show table 1.5 on P49
E = (1 : 7);
Exact = 1 ./ (E + 1) ;
T = zeros(7,5);
for i = E
    for j = 1 : 5
        T(i,j) = GaussInt( @(x) x .^ i , j , 0 , 1 );
    end
end
disp( horzcat( Exact' , T ) );

% show error
figure ;
hold on ;
for i = 1 : 5
    plot ( E , abs(T(:,i)' - Exact) );
end
```

```
h = legend('1 point','2 points','3 points','4 points','5 points')
axis([1,7,-0.01,0.14]);
set(h,'Fontsize',22);
title('error of 1-5 point(s) Gauss integral formula');
```

得到的数据如下：

Table 2: 用 Gauss 公式计算 x^n 的积分

x^n	精确值	1 点	2 点	3 点	4 点	5 点
x^1	0.5000	**0.5000**	0.5000	0.5000	0.5000	0.5000
x^2	0.3333	0.2500	**0.3333**	0.3333	0.3333	0.3333
x^3	0.2500	0.1250	**0.2500**	0.2500	0.2500	0.2500
x^4	0.2000	0.0625	0.1944	**0.2000**	0.2000	0.2000
x^5	0.1667	0.0312	0.1528	**0.1667**	0.1667	0.1667
x^6	0.1429	0.0156	0.1204	0.1425	**0.1429**	0.1429
x^7	0.1250	0.0078	0.0949	0.1238	**0.1250**	0.1250

由加粗的精确值可以印证，n 点的 Gauss 求积公式具有 2n-1 阶的代数精度。
1 到 5 点 Gauss 求积的误差如下：(4 点与 5 点重合，皆为 0)