MATH 226, HOMEWORK 2, DUE OCT. 16, 2015

SHENG XU

(1) Derive the Peano kernel for the midpoint rule on interval $[a, b]$ and show the error for the midpoint rule is

$$E(f) = \frac{(b-a)^3}{24} f''(\xi),$$

where $\xi \in [a, b]$.

Proof: Note that for Midpoint Rule $k = 1$, So we can define function using Taylor Expansion:

$$\begin{aligned} f(x) &= p_1(x) + \int_a^x f^{''}(t)(x-t)^2 \mathrm{d}t \\ &= p_1(x) + \int_a^b f^{''}(t)(x-t)_+^2 \mathrm{d}t \end{aligned}$$

Where $(x-t)_+$ is defined as:

$$(x-t)_+ = \begin{cases} 0, & \text{if } x < t \\ x-t, & \text{if } x \geq t \end{cases}$$

Now define $R(x)$ as:

$$R(x) = \int_a^b f^{''}(t)(x-t)_+^2 \mathrm{d}t$$

Thus, now we have $f(x) = p_1(x) + R(x)$. And if we continue to calculate $E(f)$

$$\begin{aligned} E(f) &= \int_a^b f(x)\mathrm{d}t - \frac{b-a}{2} f(\frac{b+a}{2}) \\ &= E(p_1 + R) = E(R) \\ E(R) &= E[\int_a^b f^{''}(t)(x-t)_+^2 \mathrm{d}t] \\ &= \int_a^b f^{''}(t)[\int_a^b (x-t)_+^2 \mathrm{d}x]\mathrm{d}t - \int_a^b f^{''}(t) \sum w_i (x_i - t)_+ \mathrm{d}t \\ &= \int_a^b f^{''}(t)[\int_a^b (x-t)_+^2 \mathrm{d}x - \sum w_i (x_i - t)_+]\mathrm{d}t \end{aligned}$$

So we can derived the Peano kernel $K(t)$ with $b \geq t \geq a$

$$\begin{aligned} K(t) &= \int_a^b (x-t)_+^2 \mathrm{d}x - \sum w_i (x_i - t)_+ \\ &= \frac{1}{2}(b-t)^2 - (b-a)(\frac{b+a}{2} - t)_+ \end{aligned}$$

Now we need to discuss the situation $a \leq t \leq \dfrac{b+a}{2}$ and $\dfrac{b+a}{2} < t \leq b$.
For $a \leq t \leq \dfrac{b+a}{2}$:

$$\begin{aligned} K(t) &= \frac{1}{2}(b-t)^2 - (b-a)(\frac{b+a}{2} - t) \\ &= \frac{1}{2}[(b-t)^2 - (b-a)(b+a-2t)] \\ &= \frac{1}{2}(t-a)^2 \end{aligned}$$

For $\dfrac{b+a}{2} < t \leq b$:

$$K(t) = \frac{1}{2}(b-t)^2$$

Therefore,

$$K(t) = \begin{cases} \frac{1}{2}(t-a)^2, & \text{if } a \leq t \leq \dfrac{b+a}{2} \\ \frac{1}{2}(b-t)^2, & \text{if } \dfrac{b+a}{2} < t \leq b \end{cases}$$

Finally, Let's Calculate $E(f)$:

$$E(f) = E(R) = \int_a^b f^{''}(t) K(t) \mathrm{d}t$$

With Intermediate Value Theorem,

$$E(f) = f^{''}(\xi)\int_a^b K(t)\mathrm{d}t$$
$$\xi\epsilon(a,b)$$
$$\therefore E(f) = f^{''}(\xi)[\int_a^{\frac{b+a}{2}} \frac{1}{2}(t-a)^2\mathrm{d}t + \int_{\frac{b+a}{2}}^b \frac{1}{2}(b-t)^2\mathrm{d}t]$$
$$= \frac{1}{2}f^{''}(\xi)[\frac{1}{3}(\frac{b-a}{2})^3 + \frac{1}{3}(\frac{b-a}{2})^3]$$
$$= \frac{(b-a)^3}{24}f''(\xi)$$
$$(\text{where } \xi\epsilon(a,b))$$

(2) Prove that $(n+1)$-point Gauss quadrature is the only $(n+1)$-point quadrature rule with degree of precision $2n+1$.

Proof: Suppose We have $n+1$ distinguished point in $[a,b]$ which are $a \leq x_1 < x_2 < \ldots < x_n \leq b$ So we compose a function $\phi(x) = \Pi_{i=0}^n(x-x_i)$ Now we find an $f(x)\epsilon P_{2n+1}$ Using Dividing Rule of polymers, we can find $q(x)$ and $R(x)$ from P_n which makes $f(x) = q(x)\phi(x) + R(x)$ Thus the integration of f can be written as:

$$\int_a^b f(x)\mathrm{d}x = \int_a^b q(x)\phi(x) + R(x)\mathrm{d}x = \int_a^b q(x)\phi(x)\mathrm{d}x + \int_a^b R(x)\mathrm{d}x$$

Since the precision is $2n+1$

$$\int_a^b f(x)\mathrm{d}x = \sum_{i=0}^n w_i f(x_i) \qquad w_i \text{ is defined by Legendre}$$
$$= \sum_{i=0}^n w_i q(x_i)\phi(x_i) + \sum_{i=0}^n w_i R(x_i)$$
$$= \sum_{i=0}^n w_i R(x_i) \qquad \phi(x_i) = 0 \text{ for } i = 0,1,\ldots,n$$

Since $R(x)\epsilon P_n$, we will have $\int_a^b R(x)\mathrm{d}x = \sum_{i=0}^n w_i R(x_i)$, which leads to:

$$\int_a^b q(x)\phi(x) = 0$$

This is true for all $q(x)$ in P_n, since we can always find a $f(x)$ in P_{2n+1} correspond to $q(x)$, where $f(x) = q(x)\phi(x)$ Note that $\phi(x)$ can be written into the form with Legendre Basis $Ł_i(x)$, where:

$$\phi(x) = \sum_{i=0}^{n+1} a_i Ł_i(x)$$

Now we define an $T(x)$ on P_n:

$$T(x) = \sum_{j=0}^{n} Ł_j(x)$$

Note that:

$$\int Ł_i(x)Ł_j(x)\mathrm{d}x = \begin{cases} 0, & \text{if } i \neq j \\ \| Ł_i(x) \|^2, & \text{if } i = j \end{cases}$$

Hence with $\int_a^b q(x)\phi(x) = 0$ is true for all $q(x)$ in Pn, we will have:

$$\begin{aligned} 0 = \int_a^b T(x)\phi(x)\mathrm{d}x &= \sum_{i=0}^{n} a_i \| Ł_i(x) \|^2 \\ \Rightarrow a_i &= 0; (i = 0, 1, \ldots, n) \\ \Rightarrow \phi(x) &= a_{n+1}Ł_{n+1}(x) \end{aligned}$$

So we proved that $\phi(x)$ has to be the $(n+1)$-degree Legendre Polynomial times a real number. With the restriction on the coefficient of highest degree x^{n+1} equals to 1, we can get a unique polynomial, which satisfies Gauss Quadrature. Thus we get the conclusion.

(3) Suppose that an numerical quadrature $I_h(f)$ has the following asymptotic expansion

$$I(f) - I_h(f) = c_1h^{r_1} + c_2h^{r_2} + c_3h^{r_3} + \cdots$$

Here $0 < r_1 < r_2 < r_3 < \cdots$ and c_i are independent of h. Assume that we have computed $I_h(f)$, $I_{h/2}(f)$, and $I_{h/4}(f)$. Show how Richardson extrapolation can be used to the maximum extent to combine these values to get a higher order approximation to $I(f)$. What is the order of the new approximation?

Proof:We define:

$$\begin{aligned} E_h^0 &= I(f) - I_h(f) = c_1h^{r_1} + c_2h^{r_2} + c_3h^{r_3} + \cdots \oplus \\ E_{\frac{h}{2}}^0 &= I(f) - I_{\frac{h}{2}}(f) = c_1(\frac{h}{2})^{r_1} + c_2(\frac{h}{2})^{r_2} + c_3(\frac{h}{2})^{r_3} + \cdots \otimes \\ E_{\frac{h}{4}}^0 &= I(f) - I_{\frac{h}{4}}(f) = c_1(\frac{h}{4})^{r_1} + c_2(\frac{h}{4})^{r_2} + c_3(\frac{h}{4})^{r_3} + \cdots \odot \end{aligned}$$

Then we get rid of h^{r_1}:

$2^{r_1} \otimes - \oplus$:

$$(2^{r_1}-1)I(f) - [2^{r_1} I_{\frac{h}{2}}(f) - I_h(f)] = c_2(2^{r_1}-2^{r_2})(\frac{h}{2})^{r_2} + c_3(2^{r_1}-2^{r_3})(\frac{h}{2})^{r_3} + \cdots$$

$2^{r_1} \odot - \otimes$:

$$(2^{r_1}-1)I(f) - [2^{r_1} I_{\frac{h}{4}}(f) - I_{\frac{h}{2}}(f)] = c_2(2^{r_1}-2^{r_2})(\frac{h}{4})^{r_2} + c_3(2^{r_1}-2^{r_3})(\frac{h}{4})^{r_3} + \cdots$$

Again we define

$$E^1_{\frac{h}{2}} = I(f) - \frac{2^{r_1} I_{\frac{h}{2}}(f) - I_h(f)}{2^{r_1}-1} \qquad E^1_{\frac{h}{4}} = I(f) - \frac{2^{r_1} I_{\frac{h}{4}}(f) - I_{\frac{h}{2}}(f)}{2^{r_1}-1}$$

And get rid of h^{r_2}:

$$2^{r_2} E^1_{\frac{h}{4}} - E^1_{\frac{h}{2}} = \frac{1}{2^{r_1}-1}[c_3(2^{r_1}-2^{r_3})(2^{r_2}-2^{r_3})(\frac{h}{4})^{r_3} + \cdots]$$

Still we can define:

$$\begin{aligned} E^2_{\frac{h}{4}} &= \frac{1}{2^{r_2}-1}[2^{r_2} E^1_{\frac{h}{4}} - E^1_{\frac{h}{2}}] \\ &= I(f) - \frac{2^{r_1+r_2} I_{\frac{h}{4}}(f) - (2^{r_1}+2^{r_2}) I_{\frac{h}{2}}(f) + I_h}{(2^{r_1}-1)(2^{r_2}-1)} \\ &= \frac{1}{(2^{r_1}-1)(2^{r_2}-1)}[c_3(2^{r_1}-2^{r_3})(2^{r_2}-2^{r_3})(\frac{h}{4})^{r_3} + \cdots] \end{aligned}$$

So the higher order approximation we get is:

$$\frac{2^{r_1+r_2} I_{\frac{h}{4}}(f) - (2^{r_1}+2^{r_2}) I_{\frac{h}{2}}(f) + I_h}{(2^{r_1}-1)(2^{r_2}-1)}$$

The order of error is r_3 with coefficient for h^{r_3} of:

$$\frac{1}{(2^{r_1}-1)(2^{r_2}-1)4^{r_3}}[c_3(2^{r_1}-2^{r_3})(2^{r_2}-2^{r_3})]$$

(4) Write a code that uses adaptive composite Simpson's rule to approximate the integral

$$\int_1^{\pi} x^2 \sin x \mathrm{d}x.$$

Given the tolerance $\varepsilon = 10^{-8}$, report the approximation value.
Answer: 5.646360126534591 with recursive and 5.646360126534589 with Cursive method.
For cursive method, we need to calculate the error under Simpson's rule in interval(a, b) with mid point c: $|I_a^c + I_c^b - I_a^b| \leq 15 * eps * (b-a)$. When it is satisfied,

we calculate it Integration approximately with Simpson method:$I = I_a^c + I_c^b$ and add I into the sum using the code :$sumI = sumI + I$. If the condition is not satisfied, we continue to divide the interval until it is satisfied and add all the value to "SumI". Then we can out sumI as the result. The speed of Calculation is a little quicker but not so obvious in this question since there are just a few steps.

```
function I=AdaptiveSimpsonC(f,a,b,eps) %Cursive
format long %type AdaptiveSimpsonC('x*x*sin(x)',1,pi) for test
if(nargin==3)
   eps=1.0e-8;     %缺省精度为0.00000001
end;
I=double(quadSimpson(f,a,b,eps));
function q=quadSimpson(f,a,b,eps)
flag=0;
SumI=0;
pt(1,1)=a;
pt(1,2)=b;
while flag==0
 flag=1;
 n=1;
 for i=1 : size(pt,1)
 a=pt(i,1);
 b=pt(i,2);
 QA=IntSimpson(f,a,b);
 QLeft=IntSimpson(f,a,(a+b)/2);
 QRight=IntSimpson(f,(a+b)/2,b);
   if(abs(QLeft+QRight-QA)<=15*eps*(b-a))
       Ipart=double(QLeft+QRight);
       SumI=SumI+Ipart;
   else
       flag=0;
       pttemp(n,1)=a;
       pttemp(n,2)=(a+b)/2;
       pttemp(n+1,1)=(a+b)/2;
       pttemp(n+1,2)=b;
       n=n+2;
   end
 end
pt=pttemp;
pttemp=0;
end
q=SumI;
function I = IntSimpson(f,a,b)
format long
       I=((b-a)/6)*(subs(sym(f),findsym(sym(f)),a)+...
           4*subs(sym(f),findsym(sym(f)),(a+b)/2)+...
           subs(sym(f),findsym(sym(f)),b));
```

```
function I=AdaptiveSimpson(f,a,b,eps) %Recursive
format long  %type AdaptiveSimpson('x*x*sin(x)',1,pi) for test
if(nargin==3)
   eps=1.0e-8;     %缺省精度为0.00000001
end;
I=double(quadSimpson(f,a,b,eps));

function q=quadSimpson(f,a,b,eps)
QA=IntSimpson(f,a,b);
QLeft=IntSimpson(f,a,(a+b)/2);
QRight=IntSimpson(f,(a+b)/2,b);
if(abs(QLeft+QRight-QA)<=15*eps*(b-a))
   q=QLeft+QRight;
else
   q=quadSimpson(f,a,(a+b)/2,eps)+quadSimpson(f,(a+b)/2,b,eps);
end
function I = IntSimpson(f,a,b)
format long
       I=((b-a)/6)*(subs(sym(f),findsym(sym(f)),a)+...
          4*subs(sym(f),findsym(sym(f)),(a+b)/2)+...
          subs(sym(f),findsym(sym(f)),b));
```