Math 270b: Numerical Approximation and Nonlinear Equations

Instructor: Randolph E. Bank

Winter Quarter 2020

Homework Assignment #10
Due Friday, March 13, 2020

Exercise 10.1. Write an adaptive quadrature subroutine; test it on the following functions:

- $f(x) = e^x$ on $[0, 1]$
- $f(x) = \log x$ on $[0, 1]$
- $f(x) = \sin x$ on $[0, \pi/2]$
- $f(x) = \sin 10x$ on $[0, \pi/2]$
- $f(x) = \sqrt{x}$ on $[0, 1]$
- $f(x) = x^2$ on $[0, 1]$
- $f(x) = x^{10}$ on $[0, 1]$

as well as several functions of your own choosing. Report the number of function evaluations as well as the error. Try several values of the error tolerance. Compare at least two different fundamental quadrature rules.

Here is a sample adaptive quadrature routine (written in FORTRAN). It uses a 1-point (order 2) or 6-point (order 12) Gaussian Quadrature formula as the fundamental rule. The maximum depth of the search tree is 12 (which is arbitrary).

```
        subroutine adapt(a,b,f,tol,nofun,quad,error,iflag)
c
            real quadlr(13,2)
            integer idfs(13),left,right
            external f
            save maxd,left,right
            data maxd,left,right/12,1,2/
c
c       input:
c           (a,b)  : interval of intergation
c           f=f(x) : function to be integrated (note external statement)
c           tol    : error tolerance
c       output:
c           nofun  : number of function evaluations
c           quad   : value of quadrature
```

```
c           error  : error estimate for quadrature
c           iflag  : error flag (iflag=0 is normal return)
c
        iflag=0
c
c       order of quadrature rule (2 or 12 in this example code)
c
        iorder=2
c
c       number of function evaluations used in the basic rule
c       (1 or 6 in this example code)
c
        ieval=1
c
c       length of integration inteval
c
        h0=b-a
c
c       level
c
        level=1
c
c       interval number on this level
c
        intnum=0
c
c       number of intervals on this level (if the mesh were uniform!)
c
        numint=1
c
c       error reduction constant depends on the order
c
        const=1.0e0/(2.0e0**(iorder)-1.0e0)
c
c       initialize on level=1
c
        idfs(level)=right
        quadlr(level,right)=gauss(a,b,f)
        nofun=ieval
        tol1=abs(tol/h0)
        quad=0.0e0
        error=0.0e0
c
c       evaluate the error on level+1
c
   10   h=h0/float(numint)
        alevel=a+float(intnum)*h
        blevel=alevel+h
        clevel=(alevel+blevel)/2.0e0
c
        quadlr(level+1,left)=gauss(alevel,clevel,f)
```

```
        quadlr(level+1,right)=gauss(clevel,blevel,f)
        nofun=nofun+2*ieval
c
c       qq is the approximation using the mesh on level+1
c
        qq=quadlr(level+1,left)+quadlr(level+1,right)
c
c       ee is the error estimate for this interval
c
        ee=(qq-quadlr(level,idfs(level)))*const
c
c       if the error is too large, go to the next level
c
        if(abs(ee) > h*tol1) then
c
c       if we have not reached maximum depth in the tree...
c
            if(level < maxd) then
                level=level+1
                idfs(level)=left
                intnum=intnum*2
                numint=numint*2
                go to 10
            else
c
c       if we reach maximum depth, accept qq, but set error flag
c
                iflag=iflag+1
            endif
        endif
c
c       update quad, error if we acept qq
c
        quad=quad+qq+ee
        error=error+abs(ee)
        intnum=intnum+1
c
c       decide what to do next
c
   20   if(idfs(level) == left) then
c
c       do right interval on same level
c
            idfs(level)=right
            go to 10
        else
c
c       go up one level
c
            intnum=intnum/2
            numint=numint/2
```

```
            level=level-1
            if(level > 0) go to 20
         endif
         return
         end
c
c
c
         function gauss(a,b,f)
c
            external f
c
c        for this rule:
c        iorder=2
c        ieval=1
c
         h=b-a
         r=(b+a)/2.0e0
         gauss=h*f(r)
         return
         end
c
         function gauss6(a,b,f)
c
            external f
            save x1,x2,x3,w1,w2,w3
            data x1,x2,x3/.238619186083197e0,
     1           .661029386466265e0,.932469514201352e0/
            data w1,w2,w3/.467913934572691e0,
     1           .360761573048139e0,.171324492379170e0/
c
c        for this rule:
c        iorder=12
c        ieval=6
c
         h=(b-a)/2.0e0
         r=(b+a)/2.0e0
         t1=x1*h
         t2=x2*h
         t3=x3*h
         s=w1*(f(r-t1)+f(r+t1))
     1    +w2*(f(r-t2)+f(r+t2))
     2    +w3*(f(r-t3)+f(r+t3))
         gauss=s*h
         return
         end
```