Paul Simmerling
pws17001

Homework 5

1 Question 29

$$\int_0^{\pi/4} \frac{dx}{\sqrt{\sin x}} \approx 1.7911613389539645 \tag{1.1}$$

2 Question 30

$$h(\theta_0 = 0^o) \int_0^{\pi/2} \frac{d\theta}{\sqrt{1-\sin^2(\theta_0/2)\sin^2\theta}} \approx 1.5707963267948966 \tag{2.1}$$

$$h(\theta_0 = 15^o) \int_0^{\pi/2} \frac{d\theta}{\sqrt{1-\sin^2(\theta_0/2)\sin^2\theta}} \approx 1.5775516530701426 \tag{2.2}$$

$$h(\theta_0 = 30^o) \int_0^{\pi/2} \frac{d\theta}{\sqrt{1-\sin^2(\theta_0/2)\sin^2\theta}} \approx 1.598142002573657 \tag{2.3}$$

$$h(\theta_0 = 45^o) \int_0^{\pi/2} \frac{d\theta}{\sqrt{1-\sin^2(\theta_0/2)\sin^2\theta}} \approx 1.6335863090850953 \tag{2.4}$$

$sin(x) \approx x$ is a good approximation even at 45^o as it only results in a percent error of -3.8%.

3 Question 31

$$g(u) = u^3 \int_0^{1/u} \frac{x^4 e^x}{(e^x-1)^2} dx \tag{3.1}$$

$g(0) \approx 0$	$g(0.05) \approx 0.003247$	$g(0.1) \approx 0.025274$	$g(0.15) \approx 0.070997$
$g(0.2) \approx 0.122878$	$g(0.25) \approx 0.167686$	$g(0.3) \approx 0.202568$	$g(0.35) \approx 0.228858$
$g(0.4) \approx 0.248618$	$g(0.45) \approx 0.263608$	$g(0.5) \approx 0.275136$	$g(0.55) \approx 0.284136$
$g(0.6) \approx 0.291265$	$g(0.65) \approx 0.296992$	$g(0.7) \approx 0.301651$	$g(0.75) \approx 0.305487$
$g(0.8) \approx 0.308678$	$g(0.85) \approx 0.311359$	$g(0.9) \approx 0.313631$	$g(0.95) \approx 0.315573$
$g(1.0) \approx 0.317244$			

4 Question 32

$$E = \int_0^{\infty} 0.5\Omega \cdot \left[100A \cdot e^{-\frac{t}{0.01s}} \sin\left(\frac{2\cdot t}{0.01s}\right)\right]^2 dt \approx 9.999991J \tag{4.1}$$

5 Question 33

$$\int_0^{\infty} \frac{xdx}{e^x+1} \approx 0.822469 \tag{5.1}$$

6 Question 34

$$erf(1.0) = \frac{2}{\sqrt{\pi}} \int_0^1 .0e^{-t^2} dt \approx 0.842701 \tag{6.1}$$

7 Question 35

$$I = \int_0^{\pi/2} \ln(\sin x) dx \approx \int_0^{0.01} \ln(x) dx + \int_{0.01}^{0.2} \ln(\sin x) dx + \int_{0.2}^{\pi/2} \ln(\sin x) dx \tag{7.1}$$

$$\int_0^{0.01} \ln(x) dx \approx -0.056052 \tag{7.2}$$

$$\int_{0.01}^{0.2} \ln(\sin x) dx \approx -0.466280 \tag{7.3}$$

$$\int_{0.2}^{\pi/2} \ln(\sin x) dx \approx -0.566461 \tag{7.4}$$

$$I = \int_0^{\pi/2} \ln(\sin x) dx \approx -1.088792 \tag{7.5}$$

```
#HW5
#P29: Page 213 Problem 10
import numpy as np
from math import *

def trapezoid(f,a,b,Iold,k):
  #First panel
  if k == 1: Inew = (f(a) + f(b))*(b-a)/2.0
  else:
    #make math readable
    n = 2**(k-2)#num of new points
    h = (b - a)/n#spacing of new points
    x = a + h/2.0
    sum = 0.0
    #Use 6.9a Ik=I_{k-1}/2 + H/2^{k-1}*SUM(f(a+(2i-1)H/(2^{k-1}))
    for i in range(n):
      sum += f(x)
      x += h
    Inew = (Iold + h*sum)/2.0
  return Inew

def romberg(f,a,b,tol=1.0e-6):
  def richardson(r,k):
    #Extrapolate with
    #R'_j = (4^{k-j}R'_{j+1}-R'_j)/(4^{k-j}-1),j=k-1,k-2,..,1
    for j in range(k-1, 0, -1):
      const = 4.0**(k-j)
      r[j] = (const*r[j+1]-r[j])/(const - 1.0)
    return r

  r = np.zeros(21)
  #Mkae first extrapolation based on trapezoid
  r[1] = trapezoid(f,a,b,0.0,1)
  r_old = r[1]#Save it, will be rewritted
  for k in range(2,21):
    #Repeat trap but use previous extrap+integrate
    r[k] = trapezoid(f,a,b,r[k-1],k)
    r = richardson(r,k)#Repeat improved extrapolation
    if abs(r[1]-r_old) < tol*max(abs(r[1]),1.0):#test for tol
      return r[1],2**(k-1)
    r_old = r[1]
  print("Romberg quadrature did not converge")

#Write a function that returns the integration value.
```

```
#It should not take any input and return a single value
#'x' using (your) romberg.py and trapezold.py modules.
def Q29():
  print("+------+")
  print("| P29 |")
  print("+------+")  #a, b = 0, pi/4
  #def f(x): return sin(x)**(-1/2)
  a, b = 0.0, sqrt(sqrt(2)/2)#Using t^2=sinx
  def f(t): return 2.0/sqrt(1-t**4)
  return romberg(f, a, b)[0]#Integrate



#P30: Page 213 Problem 11
def Q30():
  a, b = 0, pi/2

  print("+------+")
  print("| P30 |")
  print("+------+")
  #Integrate with different  theta_0
  def f0(x): return (1-sin(0/2)**2*sin(x)**2)**(-1/2)
  print(romberg(f0, a, b))
  def f15(x): return (1-sin((15*pi/180)/2)**2*sin(x)**2)**(-1/2)
  print(romberg(f15, a, b))
  def f30(x): return (1-sin((30*pi/180)/2)**2*sin(x)**2)**(-1/2)
  print(romberg(f30, a, b))
  def f45(x): return (1-sin((45*pi/180)/2)**2*sin(x)**2)**(-1/2)
  print(romberg(f45, a, b))


#P31: Page 214 Problem 14
#Write a function that returns values g(u) in the interval
#u=0 to u=1.0 in 0.05 increments. You do not
#need to plot the results for this problem.
def Q31():
  g = [0]
  #if u = 0 -> g=0
  def f(x): return x**4*exp(x)/(exp(x)-1)**2 if x != 0 else 0
  for u in np.arange(0.05,  1.05, 0.05):
    g.append(u**3*romberg(f,0,1/u)[0])#make array of g

  print("+------+")
  print("| P31 |")
  print("+------+")
```

```
  print(g)
#P32: Page 214 Problem 15

#Write a function that returns the value E using the parameters
#listed in the problem. For this function you should use the
#recursive trapezoid rule with 1024 panels.
def Q32():
  def f(x): return 0.5*(100*exp(-x/0.01)*sin(2*x/0.01))**2
  i0, R, t0 = 100, 0.5, 0.01
  b = -t0*log((10e-8/i0)**2)#go until power is 10e-6% of final val
  told=0
  for k in range(1, 12):
    told = trapezoid(f,0,b,told,k)#integrate to 1024 panels

  print("+-----+")
  print("| P32 |")
  print("+-----+")
  print(romberg(f,0,b))
  print(told)


#P33: Page 230 Problem 10
def gaussNodes(m,tol=1.e-9):
  def legendre(t, m):
    p0, p1 = 1.0, t#inital pols
    for k in range(1,m):
      #legendre poly at t and m using the recurrence in 6.19
      #a_n*phi_{n+1} = (b_n+c_n*x)*phi_n - d_n*phi_{n-1}
      #a_n=n+1,b_n=0,c_n=2n+1,dn=n
      #(n+1)*phi_{n+1} = (2n+1)*x*phi_n - n*phi_{n-1}
      #t=x, k=n
      p=((2.0*k + 1.0)*t*p1 - k*p0)/(1.0 + k)
      p0=p1; p1 = p
    #eq 6.21 -> deriv of legendre poly
    dp = m*(p0 - t*p1)/(1.0 - t**2)
    return p,dp

  A, x = np.zeros(m), np.zeros(m)
  #calc num of roots
  nRoots = int((m+1)/2)
  for i in range(nRoots):
    #approx abcissas xi = cos(pi(i+3/4)/(m+1/2)), m=nodes+1
    t = cos(pi*(i+0.75)/(m+0.5))
    for j in range(30):
      p,dp=legendre(t,m)#find pol
```

```
      dt = -p/dp#find dt
      t = t+dt
      if abs(dt)<tol:
        x[i]=t
        x[m-i-1] = -t
        #use eq 6.25 for A
        A[i] = 2.0/(1.0-t**2)*1/(dp**2)
        #update A for next iteration
        A[m-i-1] = A[i]
        break
  return x,A

def gaussQuad(f,a,b,m):
  c1,c2 = (b+a)/2.0,(b-a)/2.0
  x,A=gaussNodes(m)
  s = 0.0
  for i in range(len(x)):
    #sum the eq using weights from pol eq 6.26
    s += A[i]*f(c1+c2*x[i])
  return c2*s


def integrate(f, a, b, tol=1e-6):
  old = gaussQuad(f,a,b,2)
  for m in range(3,1001):
    curr = gaussQuad(f,a,b,m)#keep repeating with increased acc
    if abs(old-curr)<tol:#check if in tol
      return curr
    old = curr
  print("Could not converge")

def Q33():
  print("+-----+")
  print("| P33 |")
  print("+-----+")
  def f(x): return -1/(1+x)
  def f(x): return x/(exp(x)+1)#sub not working?
  print(integrate(f, 0, 100))
  #def f(x): return log(-log(x))/((x*log(x)*(1-log(x))))
  #print(integrate(f, exp(-1), 0))


#P34: Page 231 Problem 12
def erf(x):
  def f(t): return 2/sqrt(pi)*exp(-1*(t**2))#erf
```

```
    return integrate(f, 0, x)

def Q34():
    print("+------+")
    print("| P34 |")
    print("+------+")
    print(erf(1.0))

#P35: Page 231 Problem 15

def Q35():
    print("+------+")
    print("| P35 |")
    print("+------+")

    def f(x): return log(sin(x))#use piecewise approach
    print(0.01*(log(0.01)-1))
    print(integrate(f,0.01,0.2))
    print(integrate(f,0.2, pi/2))
    print(0.01*(log(0.01)-1)+integrate(f,0.01,0.2)+integrate(f,0.2, pi/2))


if __name__ == "__main__":
    print(Q29())
    print(Q30())
    print(Q31())
    print(Q32())
    print(Q33())
    print(Q34())
    print(Q35())
```