Justin Nguyen

Problem 16A

language-python

```
     x    h  true_val        w1         error
0  1.1  0.1  1.092629  1.092629  2.393596e-09
1  1.2  0.1  1.187085  1.187085  3.573025e-09
2  1.3  0.1  1.283382  1.283382  4.324595e-09
3  1.4  0.1  1.381446  1.381446  4.927043e-09
4  1.5  0.1  1.481159  1.481159  5.480356e-09
5  1.7  0.1  1.685014  1.685014  5.786400e-10
6  1.9  0.1  1.893930  1.893930  1.300705e-09
7  2.0  0.1  2.000000  2.000000  6.951240e-11
```

Problem 16B

language-python

```
      x    h  true_val        w1         error
0   1.0  0.1  1.000000  1.000000  0.000000e+00
1   1.1  0.1  1.092629  1.092629  1.343586e-07
2   1.2  0.1  1.187085  1.187085  1.336928e-07
3   1.3  0.1  1.283382  1.283382  9.782392e-08
4   1.4  0.1  1.381446  1.381446  6.019883e-08
5   1.5  0.1  1.481159  1.481159  3.067557e-08
6   1.6  0.1  1.582392  1.582392  1.081805e-08
7   1.7  0.1  1.685014  1.685014  4.897331e-10
8   1.8  0.1  1.788899  1.788899  4.990896e-09
9   1.9  0.1  1.893930  1.893930  4.346022e-09
10  2.0  0.1  2.000000  2.000000  6.951240e-11
```

Code Source

language-python

```python
import numpy as np

import pandas as pd



"""

Function RK_Vec_Var_11_1_A



In BlackBox:

- Compute gamma
- Compute approximations

- Compute Y true

- Compute Error


Format short g

Show:

- XX = a+(i*H[i])

- H

- YA

- Error


Values are as follows:

X H YA

1.1 0.1 1.0926 1.0926 2.4045E-9

1.2 0.1 1.1871 1.1871 3.5951E-9

1.3 0.1

1.4 0.1

1.5 0.1

1.7 0.2

1.9 0.2

2 0.1 2 2 0



HW 16B

X H YA
```

```
1.1 0.1 1.0926 1.0926 1.3435E-7

1.2 0.1 1.1871 1.1871 1.3367E-7

1.3 0.1

1.4 0.1

1.5 0.1

1.7 0.2

1.9 0.2

2 0.1 2 2 8.8818E-16


"""

#u1,u2,u3,u4

# a = 1

# b = 2

# alpha = 1

# beta = 2 #

# w = np.array([alpha, 0, 0, 1])


p = lambda x : -2/x

q = lambda x : (2 / x**2)

r = lambda x : (np.sin(np.log(x)))/(x**2)


c2 = -0.0392070132

c1 = 1.1392070132

true_function = lambda x : (c1*x) + (c2/(x**2)) \
- ((0.3)*np.sin(np.log(x))) \
- ((0.1)*np.cos(np.log(x)))

def set_pandas_display_options() -> None:
"""Set pandas display options."""
# Ref: https://stackoverflow.com/a/52432757/
display = pd.options.display

display.max_columns = 1000
display.max_rows = 1000
display.max_colwidth = 199
display.width = 1000

def vector_function(t,w):
f1 = lambda x,w : w[1]
#(p(x)*u2+q(x)*u1+r(x));
f2 = lambda x,w : p(x) * w[1] + q(x) * w[0] + r(x)
f3 = lambda x,w : w[3]
# k12 = h*(p(x)*v2+q(x)*v1);
f4 = lambda x,w : p(x)* w[3] + q(x)* w[2]
fv = np.zeros(len(w))
fv[0] = f1(t,w)
fv[1] = f2(t,w)
fv[2] = f3(t,w)
fv[3] = f4(t,w)
```

```
return fv


def RK4Vector(a,b, w, N, h=[False,0.1], condition= 1):
"""
RK4 for a vector function
"""


T = np.zeros(N+1)
W = np.zeros((N+1, len(w)))
W[0] = w
T[0] = a
if h[0] == False:
h[1] = (b-a)/N
else:
h[1] = h[1]

h_val = h[1]

if condition == 1:
function = vector_function

for i in range(N):
s1 = function(T[i], W[i])
s2 = function(T[i] + h_val/2, W[i] + h_val*s1/2)
```

```
s3 = function(T[i] + h_val/2, W[i] + h_val*s2/2)

s4 = function(T[i] + h_val, W[i] + h_val*s3)

W[i+1] = W[i] + (h_val*(s1 + 2*s2 + 2*s3 + s4)/6)

T[i+1] = T[i] + h_val

return T,W



def AdaptiveRKVector(a,b,w,tol,condition=1):

T = []

W = []

H = []

t = a

w = w

h = 0.1

if a + h > b:

h = b - a

while t < b - 10E-12:

x1,w1 = RK4Vector(t, b, w,1, h=[True, h], condition=condition)#RK4Vector(t, w, h, N)

x2, w2 = RK4Vector(t, b, w, 2, h=[True,h/2], condition=condition)



last_w1 = w1[-1]

last_w2 = w2[-1]



w3 = (16*last_w2 - last_w1)/15

error = max(abs(w3 - last_w1))
```

```
if error < tol:
w = w3
t = t + h
T.append(t)
W.append(w)
H.append(h)
if error < tol/128:
h = 2*h
if t + h > b:
h = b - t

else:
h = h/2
if h < 10E-4:
print("Step size has become too small at N = ", len(T))
return T,W,H

return T,W,H

def sixteenB():
a = 1
b = 2
N = 10
h = (b-a)/N
```

```
w = np.array([a, 0, 0, 1])

T,W = RK4Vector(a,b,w,N)
error = []
true_val = []
h_list = []
x_list = []
w1_list = []
z = (b - W[-1][0]) / (W[-1][2])

for i in range(len(T)):

X = a+i*h
w1 = W[i][0] + (z*W[i][2])
w2 = W[i][1] + (z*W[i][3])

w1_list.append(w1)
x_list.append(X)
h_list.append(h)
true_val.append(true_function(X))
error.append(abs(true_function(X) - w1))
info_dict = {
'x': x_list,
'h': h_list,
'true_val': true_val,
'w1': w1_list,
'error': error
}

df = pd.DataFrame(info_dict)
print(df)
def sixteenA():
a = 1
b = 2
tol = 10E-5
N = 10
h = (b-a)/N
w = np.array([a, 0, 0, 1])

T,W,H = AdaptiveRKVector(a,b,w,tol,condition=1)

error = []
true_val = []
h_list = []
x_list = []
w1_list = []
z = (b - W[-1][0]) / (W[-1][2])

for i in range(len(T)):
```

```
X = T[i]#a+(i*H[i])
w1 = W[i][0] + (z*W[i][2])
w2 = W[i][1] + (z*W[i][3])

w1_list.append(w1)
x_list.append(X)
h_list.append(h)
true_val.append(true_function(X))
error.append(abs(true_function(X) - w1))
info_dict = {
'x': x_list,
'h': h_list,
'true_val': true_val,
'w1': w1_list,
'error': error
}

df = pd.DataFrame(info_dict)
print(df)

if __name__=='__main__':
set_pandas_display_options()
sixteenA()
sixteenB()
```