ch_16_assignment

May 22, 2023

Ch_16_assignment

[ ]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = 'all'
```

1 Integration

1.1 Importing Modules

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
from scipy import integrate
import sympy
import mpmath

sympy.init_printing()
```

1.2 Numerical Integration Methods

[ ]:
```
a, b, X = sympy.symbols("a, b, x")
f = sympy.Function("f")

x = a, (a+b)/2, b # for Simpson's rule
w = [sympy.symbols("w_%d" % i) for i in range(len(x))]

q_rule = sum([w[i] * f(x[i]) for i in range(len(x))])
q_rule
```

[ ]:
$$w_0 f(a) + w_1 f\left(\frac{a}{2} + \frac{b}{2}\right) + w_2 f(b)$$

[ ]:
```
phi = [sympy.Lambda(X, X**n) for n in range(len(x))]
phi
```

[ ]:
$$\left[(x \mapsto 1),\ (x \mapsto x),\ (x \mapsto x^2)\right]$$

[ ]:
```
eqs = [q_rule.subs(f, phi[n]) - sympy.integrate(phi[n](X), (X, a, b)) for n in␣
↪range(len(phi))]
eqs
```

[ ]:
$$\left[a - b + w_0 + w_1 + w_2,\ \frac{a^2}{2} + aw_0 - \frac{b^2}{2} + bw_2 + w_1\left(\frac{a}{2} + \frac{b}{2}\right),\ \frac{a^3}{3} + a^2w_0 - \frac{b^3}{3} + b^2w_2 + w_1\left(\frac{a}{2} + \frac{b}{2}\right)^2\right]$$

[ ]:
```
w_sol = sympy.solve(eqs, w)
w_sol

q_rule.subs(w_sol).simplify()
```

[ ]:
$$\left\{w_0 : -\frac{a}{6} + \frac{b}{6},\ w_1 : -\frac{2a}{3} + \frac{2b}{3},\ w_2 : -\frac{a}{6} + \frac{b}{6}\right\}$$

[ ]:
$$\frac{(a-b)\left(-f(a) - f(b) - 4f\left(\frac{a}{2} + \frac{b}{2}\right)\right)}{6}$$

1.3 Numerical Integration with Scipy

[ ]:
```
def f(x):
    return np.exp(-x**2)
val, err = integrate.quad(f, -1, 1)

val

err
```

[ ]:
1.49364826562485

[ ]:
$1.65828269518814 \cdot 10^{-14}$

[ ]:
```
def f(x, a, b, c):
    return a * np.exp(-((x-b)/c)**2)

val, err = integrate.quad(f, -1, 1, args = (1, 2, 3))
val
err
```

[ ]:
1.27630683510222

[ ]:
$1.41698523481695 \cdot 10^{-14}$

[ ]:
```
from scipy.special import jv
f = lambda x: jv(0, x)
val, err = integrate.quad(f, 0, 5)
val
err
```

[ ]:
0.715311917784768

[ ]:

$2.47260738289741 \cdot 10^{-14}$

[ ]:
```
f = lambda x: np.exp(-x**2)
val, err = integrate.quad(f, -np.inf, np.inf)
val

err
```

[ ]: 1.77245385090552

[ ]: $1.42026370594529 \cdot 10^{-8}$

[ ]:
```
f = lambda x: 1/np.sqrt(abs(x))
a, b = -1, 1
integrate.quad(f, a, b)

integrate.quad(f, a, b, points=[0])
```

```
/var/folders/r1/8vnnkyjn3h3b_tnp2010w6nm0000gn/T/ipykernel_12186/2733451217.py:1
: RuntimeWarning: divide by zero encountered in scalar divide
  f = lambda x: 1/np.sqrt(abs(x))
```

[ ]: $(\infty,\ \infty)$

[ ]: $(4.0,\ 2.04281036531029 \cdot 10^{-14})$

[ ]:
```
f = lambda x: np.sqrt(x)
a, b = 0, 2
x = np.linspace(a, b, 25)
y = f(x)
fig, ax = plt.subplots(figsize=(8, 3))
ax. plot(x, y)
xx = np.linspace(a, b, 500)
ax.plot(xx, f(xx), 'b-')
ax.fill_between(xx, f(xx), color='green', alpha= 0.5)
ax.set_xlabel(r"$x$", fontsize=18)
ax.set_ylabel(r"$f(x)$", fontsize=18)

val_trapz = integrate.trapz(y, x)
val_trapz

val_simps = integrate.simps(y, x)
val_simps
```

[ ]: `[<matplotlib.lines.Line2D at 0x138892b80>]`

[ ]: `[<matplotlib.lines.Line2D at 0x12e60beb0>]`

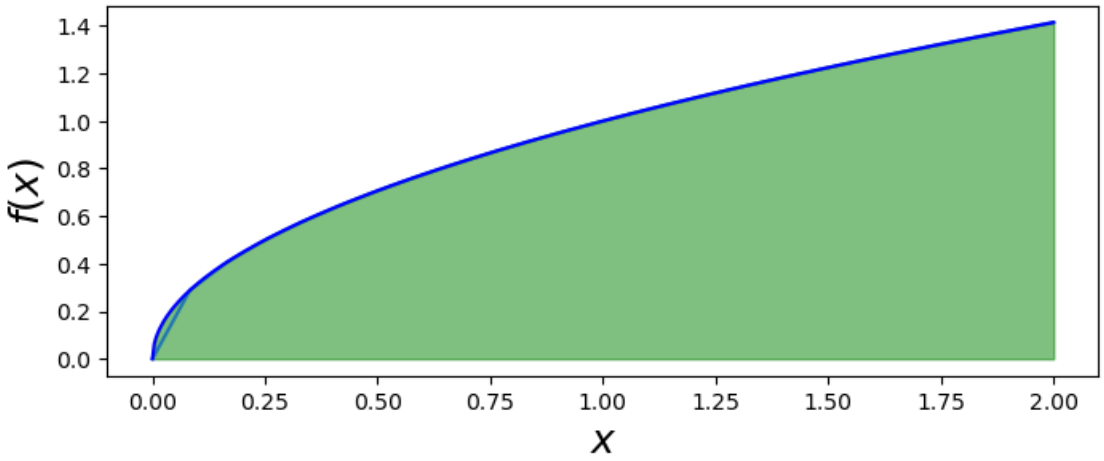
[ ]: `<matplotlib.collections.PolyCollection at 0x12e60f490>`

[ ]:
```
Text(0.5, 0, '$x$')
```

[ ]:
```
Text(0, 0.5, '$f(x)$')
```

[ ]:
1.88082171605085

[ ]:
1.88366510244871



Reference * Title: Physics Programming Lecture Note (INU) * Author: Jeongwoo Kim, Ph.D. * Availability: https://sites.google.com/view/jeongwookim

---