Complex Integration (AG mam)

19th May, 2023

The integration is done by Simpson's 1/3 rule.

$$\int_a^b f(x)dx = \frac{h}{3}[f(a) + 4(f(a+h) + f(a+3h) + \ldots) + 2(f(a+2h) + f(a+4h) + \ldots) + f(b)]$$

In [1]:

```
def simp13z(f, pr, a, b, tol):
    n = 10
    I1 = 0
    while True:
        h = (b-a)/n
        I2 = 0
        for i in range(n+1):
            if i==0 or i==n:
                I2 += f(pr, a+i*h)
            elif (i%2)==0:
                I2 += 2*f(pr, a+i*h)
            else:
                I2 += 4*f(pr, a+i*h)
        I2 = (h/3)*I2
        if abs(I2-I1) <= tol:
            break
        else:
            I1 = I2
            n += 10
    return I2
```

Example:

$$\int_0^{\pi+2j} \cos(\frac{z}{2})dz$$

In [2]:

```
from cmath import *
def f1(pr, z):
    return cos(z/2)  # function
tol = 1e-6
intgsp1 = simp13z(f1, None, 0, pi +2j, tol)
print(intgsp1)

import numpy as np
def f1(pr, z):
    return np.cos(z/2)  # function
tol = 1e-6
intgsp2 = simp13z(f1, None, 0, np.pi +2j, tol)
print(intgsp2)
```

```
(3.086161217931016+6.174435784878085e-08j)
(3.086161217931016+6.174435762673625e-08j)
```

Example:

$$\int_0^{j} \frac{z^2+1}{z+1}dz$$

In [3]:
```
import numpy as np
def f2(pr, z):
    return (z**2 +1)/(z +1) # function
tol = 1e-6
intgsp2 = simp13z(f2, None, 0, 1j, tol)
print(intgsp2)
```

```
(0.1931472836593496+0.5707963267676798j)
```

Contour Integration

We need to evaluate $\oint_c f(z)dz$ from $z = z_0$ to $z = z_1$ along the curve c and $z = c(t)$.

We can get the integration as, $(z = c(t) = x(t) + jy(t))$

$$\int_{t_0}^{t_1} f(x(t) + jy(t))(x'(t) + jy'(t))dt = \int_{t_0}^{t_1} F(t)dt$$

Differentiation (3 points):

$$\frac{df}{dx} = \frac{f(x+h) - f(x-h)}{2h}$$

In [4]:
```

def dfdz3(f, pr, z, tol):
    h = 0.1
    ch = complex(h,h)
    dfdz1 = (f(pr, z+ch) - f(pr, z-ch))/(2*ch)
    while True:
        h = h/2
        ch = complex(h,h)
        dfdz2 = (f(pr, z+ch) - f(pr, z-ch))/(2*ch)
        if abs(dfdz2 -dfdz1) <= tol:
            break
        else:
            dfdz1 = dfdz2
    return dfdz2
```

Formation of integrand and integration:

In [5]:
```
def fzdz(fnpr, t):
    f, prf, c, prc, tol = fnpr
    z = c(prc, t)
    Ft = f(prf, z)* dfdz3(c, prc, t, tol)
    return Ft

def simp13cont(f, prf, c, prc, t0, t1, tol):
    fnpr = [f, prf, c, prc, tol]
    contintg = simp13z(fzdz, fnpr, t0, t1, tol)
    return contintg
```

Examples

(a) Example: $f(z) = \bar{z}$, $z_0 = 0$, $z_1 = 4 + 2j$ and $c(t) = t^2 + tj$.

Solution: By solving we can get t varies from 0 to 2.

In [6]:
```
def f(prf, z):
    return z.conjugate() # input the function

def c(prc, t):
    return t**2 +t*1j  # input the curve
```

In [7]:
```
tol = 1e-6  # tolerance
t0, t1 = 0, 2  # integration limits
prf, prc = None, None
contintg1 = simp13cont(f, prf, c, prc, t0, t1, tol)
print(contintg1)
```

```
(10.000000000000005-2.6666666666666763j)
```

29 May, 2023 (online)

(b) Example: $f(z) = z^{1/2}$, $(z_0 = 3, z_1 = -3)$ and $c(\theta) = 3e^{j\theta}$.

Solution: We can get, $(t_0 = 0, t_1 = \pi)$. $(t = \theta)$

In [8]:
```
import numpy as np
def f(prf, z):
    return z**0.5 # input the function

def c(prc, t):
    return 3*np.exp(t*1j)  # input the curve
```

In [9]:
```
tol = 1e-6  # tolerance
t0, t1 = 0, np.pi  # integration limits
prf, prc = None, None
contintg1 = simp13cont(f, prf, c, prc, t0, t1, tol)
print(contintg1)
```

```
(-3.4641021868120188-3.464101834425901j)
```

(c) Example: $f(z) = exp((a-1)log(z))$, $(-\pi \leq \theta \leq \pi)$ and $c(\theta) = Re^{j\theta}$.

Solution: We have, $(t_0 = -\pi, t_1 = \pi)$. $(t = \theta)$

In [10]:
```
import numpy as np
def f(prf, z):
    a = prf
    return np.exp((a-1)*np.log(z)) # input the function

def c(prc, t):
    R = prc
    return R*np.exp(t*1j)  # input the curve
```

In [11]:
```python
tol = 1e-6  # tolerance
t0, t1 = -np.pi, np.pi  # integration limits
prf, prc = [-1,-0.5,0.5,1], 1  # prf = a, prc = R (input values)

for a in prf:
    contintg1 = simp13cont(f, a, c, prc, t0, t1, tol)
    print('for a = %f and R = %f, integral = %f + %fj'
          %(a, prc, contintg1.real, contintg1.imag))
```

```
for a = -1.000000 and R = 1.000000, integral = -0.000000 + -0.000000j
for a = -0.500000 and R = 1.000000, integral = 0.000001 + 4.000000j
for a = 0.500000 and R = 1.000000, integral = 0.000001 + 4.000000j
for a = 1.000000 and R = 1.000000, integral = -0.000000 + -0.000000j
```

(d) Example: $f(z) = \pi exp(\pi \bar{z})$ and c is the boundary of square with vertices $0, 1, 1+j, j$ in anticlockwise direction.

Solution: We have the paths,

1. $c_1 : (z_0 = 0, z_1 = 1)$.
2. $c_2 : (z_0 = 1, z_1 = 1+j)$.
3. $c_3 : (z_0 = 1+j, z_1 = j)$.
4. $c_4 : (z_0 = j, z_1 = 0)$.

In [12]:
```python
import numpy as np
def f(prf, z):
    return np.pi*np.exp(np.pi*z.conjugate()) # input the function

def c1(prc, z):  # curve (path) 1
    return z
def c2(prc, z):  # curve (path) 2
    return z
def c3(prc, z):  # curve (path) 3
    return z
def c4(prc, z):  # curve (path) 4
    return z
```

In [13]:
```python
tol = 1e-6
prf, prc = None, None

intg1 = simp13cont(f, prf, c1, prc, 0, 1, tol)
print('I_c1 =', intg1)
intg2 = simp13cont(f, prf, c2, prc, 1, 1+1j, tol)
print('I_c2 =', intg2)
intg3 = simp13cont(f, prf, c3, prc, 1+1j, 1j, tol)
print('I_c3 =', intg3)
intg4 = simp13cont(f, prf, c4, prc, 1j, 0, tol)
print('I_c4 =', intg4)

intg = intg1 + intg2 + intg3 + intg4
print('result I_c =', intg)
```

```
I_c1 = (22.14069355699097-5.899692905713894e-15j)
I_c2 = (46.281386308945336-1.827109891954543e-14j)
I_c3 = (22.14069355699098+6.033653069004802e-15j)
I_c4 = (-2.000000423093183+1.4802973661668754e-17j)
result I_c = (88.56277299983411-1.8122335782592855e-14j)
```

(e) Example: $f(z) = \frac{1}{(z-z_0)^n}$, $(n = 2, 3, 4, \ldots)$; $c(\theta) = Re^{j\theta}$ and $z_0 = \frac{R}{2} exp(\frac{j\pi}{4}), (R = 1)$.

Solution:

In [14]:
```
import numpy as np
def f(prf, z):
    z0, n = prf
    return 1/(z-z0)**n # input the function

def c(prc, th):
    R = prc
    return R*np.exp(th*1j)  # input the curve
```

In [15]:
```
tol = 1e-6
R = 1
z0 = (R/2)* np.exp(1j*np.pi/4)
for n in range(2,5):
    intg = simp13cont(f, [z0,n], c, R, 0, 2*np.pi, tol)
    print('n = %d, I =' %(n), intg)
```

```
n = 2, I = (4.266343353926582e-09+1.391659530705444e-13j)
n = 3, I = (-5.943228934898735e-09+1.241528591044285e-13j)
n = 4, I = (6.757301247986022e-09-4.962590971092578e-14j)
```

(f) Example: $f(z) = (z^2 + 1)^2, \quad [x = a(\theta - \sin\theta), y = a(1 - \cos\theta)], \quad (0 \leq \theta \leq 2\pi)$ and $c(\theta) = x + yj$.

Solution: We have, $(t_0 = -\pi, t_1 = \pi)$. $(t = \theta)$

In [16]:
```
import numpy as np
def f(prf, z):
    a = prf
    return (z**2 + 1)**2 # input the function

def c(a, th):
    return a*(th-np.sin(th)) +1j*a*(1-np.cos(th))  # input the curve
```

In [17]:
```
tol = 1e-6
a = 1
intg = simp13cont(f, None, c, a, 0, 2*np.pi, tol)
print('result =', intg)
```

```
result = (2130.175678928536+8.8889985479623e-05j)
```

(g) Example: Calculate the definite integral,

$$G(z) = \int_{\pi - j\pi}^{z} \cos 3\xi d\xi$$

at an arbitary point $z = 2 + 3j$. Then show $G'(z) = \cos 3z$ at $z = 2 + 3j$.

In [18]:
```
import numpy as np
def f(prf, xi): # integrand
    return np.cos(3*xi)
def G(pr, z):
    return simp13z(f, pr, np.pi - 1j*np.pi, z, tol)

tol = 1e-6
z0 = 2 + 3*1j
G0 = G(None, z0)
print('value of the integration at z0 is', G0)

rhs = np.cos(3*z0)
lhs = dfdz3(G, None, z0, tol)
print('dG/dz (z0) =', lhs, 'and \ncos(3z0) =', rhs)
```

```
value of the integration at z0 is (-377.354541469566-768.5512486606746j)
dG/dz (z0) = (3890.170288681984+1132.063643168658j) and
cos(3z0) = (3890.1702679932287+1132.0635990442863j)
```

(h) Example: $f(z) = \sqrt{z}$ and c is the boundary broken into 3 parts c_1, c_2, c_3 in anticlockwise direction.

1. $c_1 : z = re^0; (0 \leq r \leq 1)$.
2. $c_2 : z = 1e^{j\theta}; (0 \leq \theta \leq \pi)$.
3. $-c_3 : z = re^0; (0 \leq r \leq 1)$.

Solution:

In [19]:
```
import numpy as np
def f(prf, z):
    return z**0.5 # input the function

def c1(prc, r):  # curve (path) 1
    return r
def c2(prc, th):  # curve (path) 2
    return np.exp(1j*th)
def c3(prc, r):  # curve (path) 3
    return -r
```

In [20]:
```
tol = 1e-6
intg1 = simp13cont(f, None, c1, None, 0, 1, tol)
intg2 = simp13cont(f, None, c2, None, 0, np.pi, tol)
intg3 = simp13cont(f, None, c3, None, 1, 0, tol)
intg = intg1 +intg2 +intg3
print(intg)
```

```
(-1.7539275942214797e-05-1.7268008757120867e-05j)
```

Question-7:

$$\int \frac{1}{1+x^2} dx$$

In [21]:
```
def f(prf, z):
    return 1/(1+z**2)
def c(prc, th):
    return 1.1*np.exp(1j*th)
tol = 1e-6
intg = simp13cont(f, None, c, None, 0, np.pi, tol)
print(intg)
```

```
(1.4756298702810504-3.0021770148508094e-07j)
```

In [22]:
```
from scipy.integrate import quad
fx = lambda x: 1/(1+x**2)
intgv = quad(fx, -np.inf, np.inf)
print(intgv)
```

```
(3.141592653589793, 5.155583041103855e-10)
```

In [ ]:
```
```