ShootingPi(1)

January 13, 2025

[1]:
```
import numpy as np
from numpy.random import rand
import matplotlib.pyplot as plt
import time
from IPython import display
```

Estimation of pi

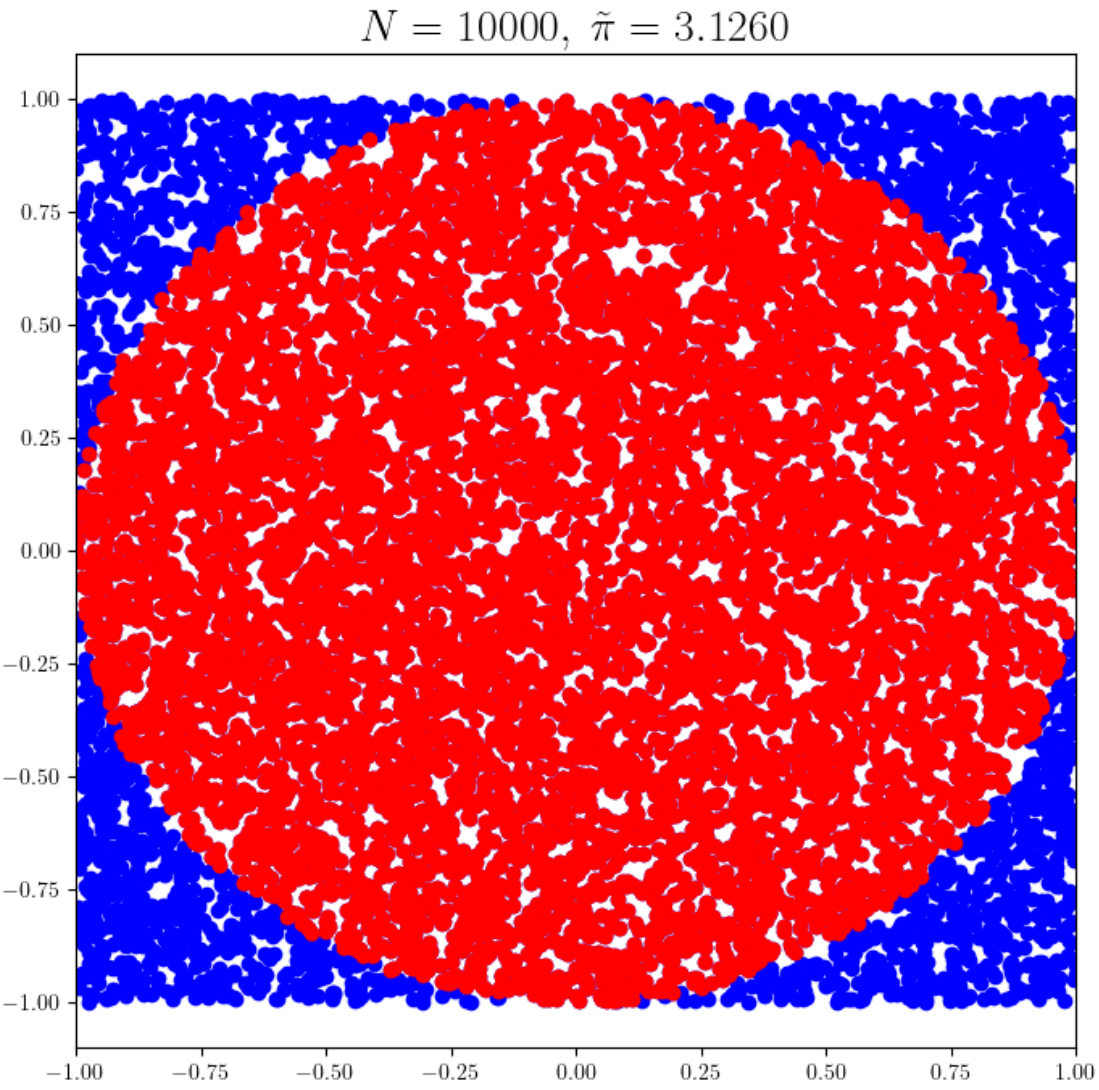
[10]:
```
def plot_tool(points, points_in):

    x,y=np.asarray(points)[:,0], np.asarray(points)[:,1]
    x_in,y_in=np.asarray(points_in)[:,0], np.asarray(points_in)[:,1]

    plt.clf()
    plt.figure("scatter", figsize=(8,8))
    plt.scatter(x,y, color='b')
    plt.scatter(x_in, y_in, color='r')

    plt.title(r"$N=%i,\; \tilde{\pi}=%.4f$"%(len(points),4*len(points_in)/
 ↪len(points)), fontsize=20)
    plt.xlim(-1,1)
    plt.ylim=(-1,1)


    display.display(plt.gcf())
    display.clear_output(wait=True)
    time.sleep(0.01)
    #plt.show()
```

[11]:
```
shots=10000
R=1
A_square=4*R**2
A_circle=np.pi*R*R
```

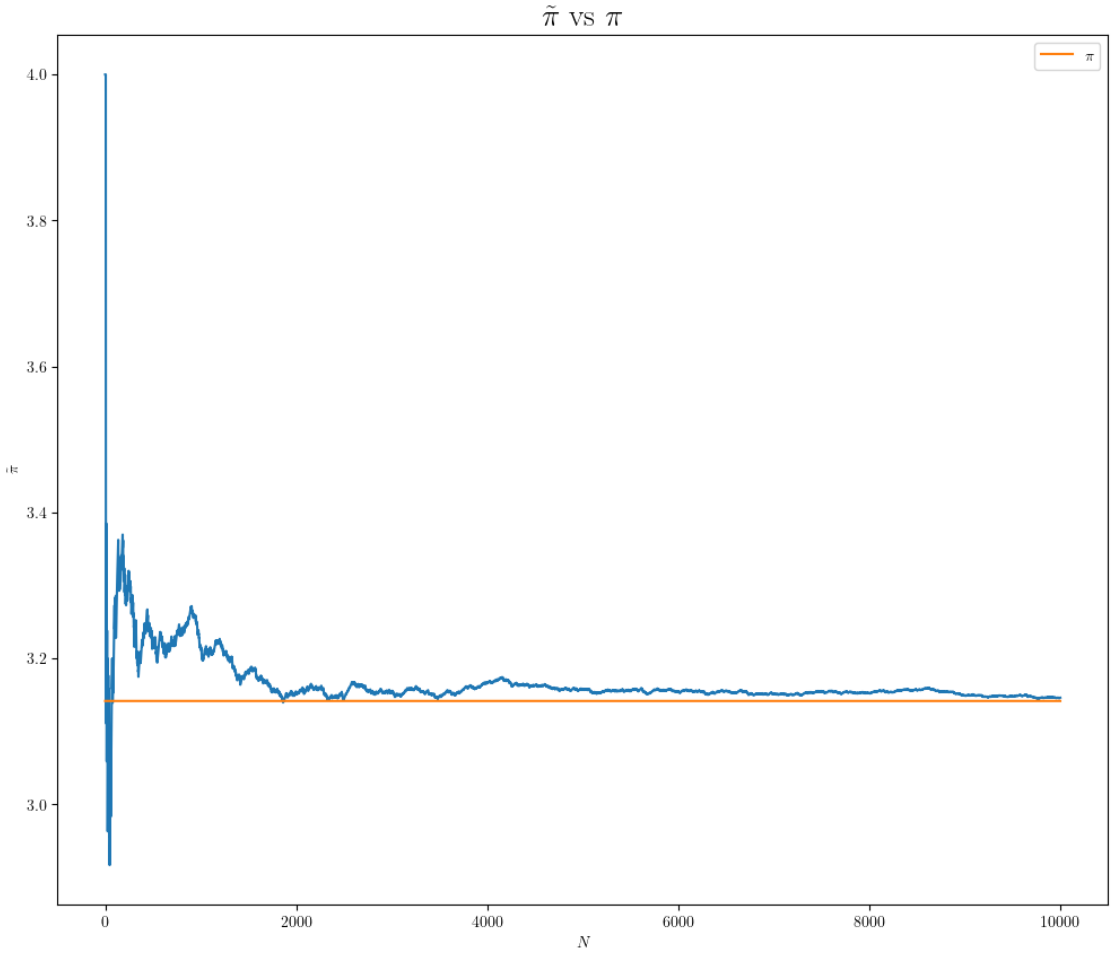
[12]:
```
points=[]
frac=[]
freq=100
```

```
for n in range(shots):
    sample=2*rand(2)-np.asarray([1, 1])
    points+=[sample]

    points_in=[p for p in points if p[0]**2+p[1]**2 <=R**2]

    frac+=[len(points_in)/len(points)]

    if (n+1)%freq==0:
        plot_tool(points, points_in)
```

<Figure size 640x480 with 0 Axes>



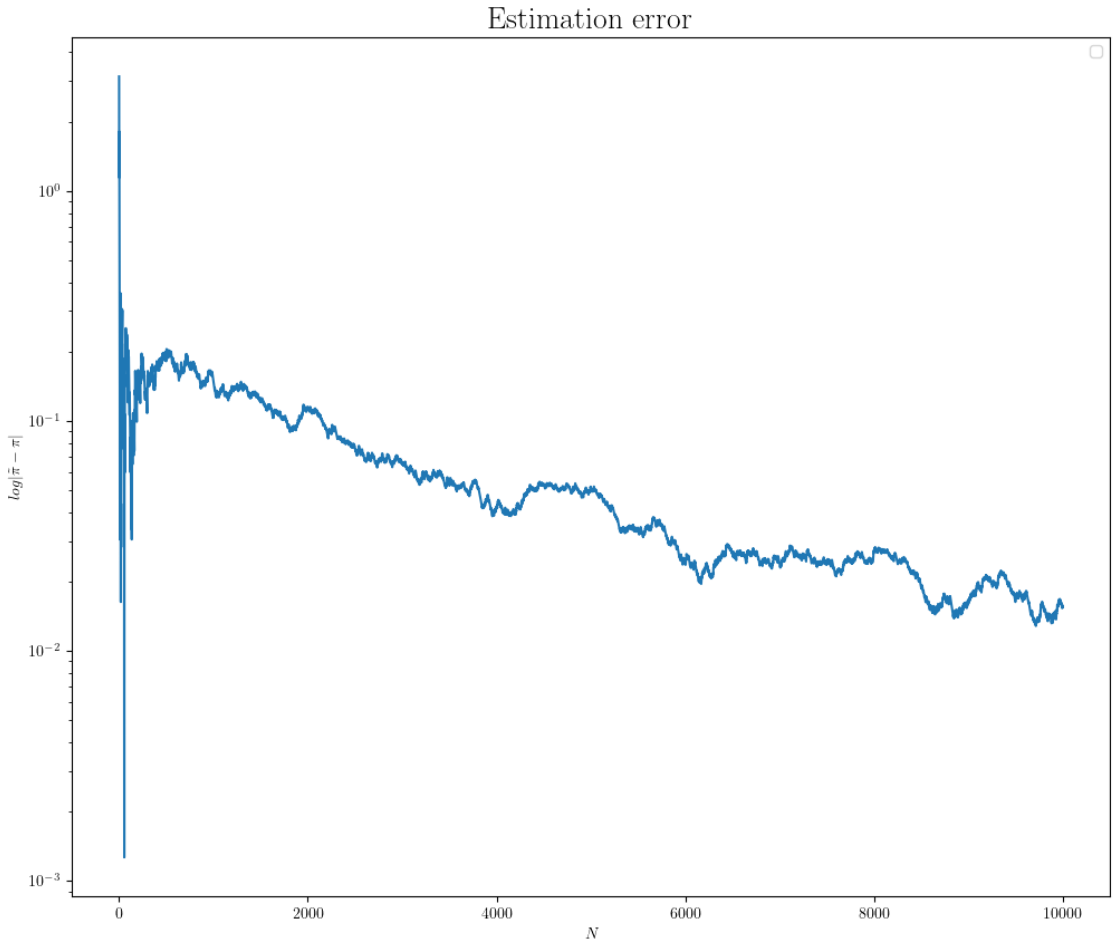
[21]:
```
plt.figure('evol',figsize=(12,10))

plt.plot(range(len(frac)), 4*np.asarray(frac))
plt.plot(range(len(frac)), np.pi*np.ones(len(frac)), label=r'$\pi$')
plt.xlabel(r'$N$')
plt.ylabel(r'$\tilde{\pi}$')
plt.title(r"$\tilde{\pi}$ vs $\pi$", fontsize=20)

plt.legend()
plt.show()
```



[13]:
```
plt.figure('error',figsize=(12,10))

plt.plot(abs(4*np.asarray(frac) -  np.pi*np.ones(len(frac))))
plt.yscale('log')
plt.xlabel(r'$N$')
plt.ylabel(r'$log|\tilde{\pi} - \pi|$')
```

```
plt.title(r"Estimation error", fontsize=20)

plt.legend()
plt.show()
```

```
/tmp/ipykernel_63942/1090897698.py:9: UserWarning: No artists with labels found
to put in legend.  Note that artists whose label start with an underscore are
ignored when legend() is called with no argument.
  plt.legend()
```



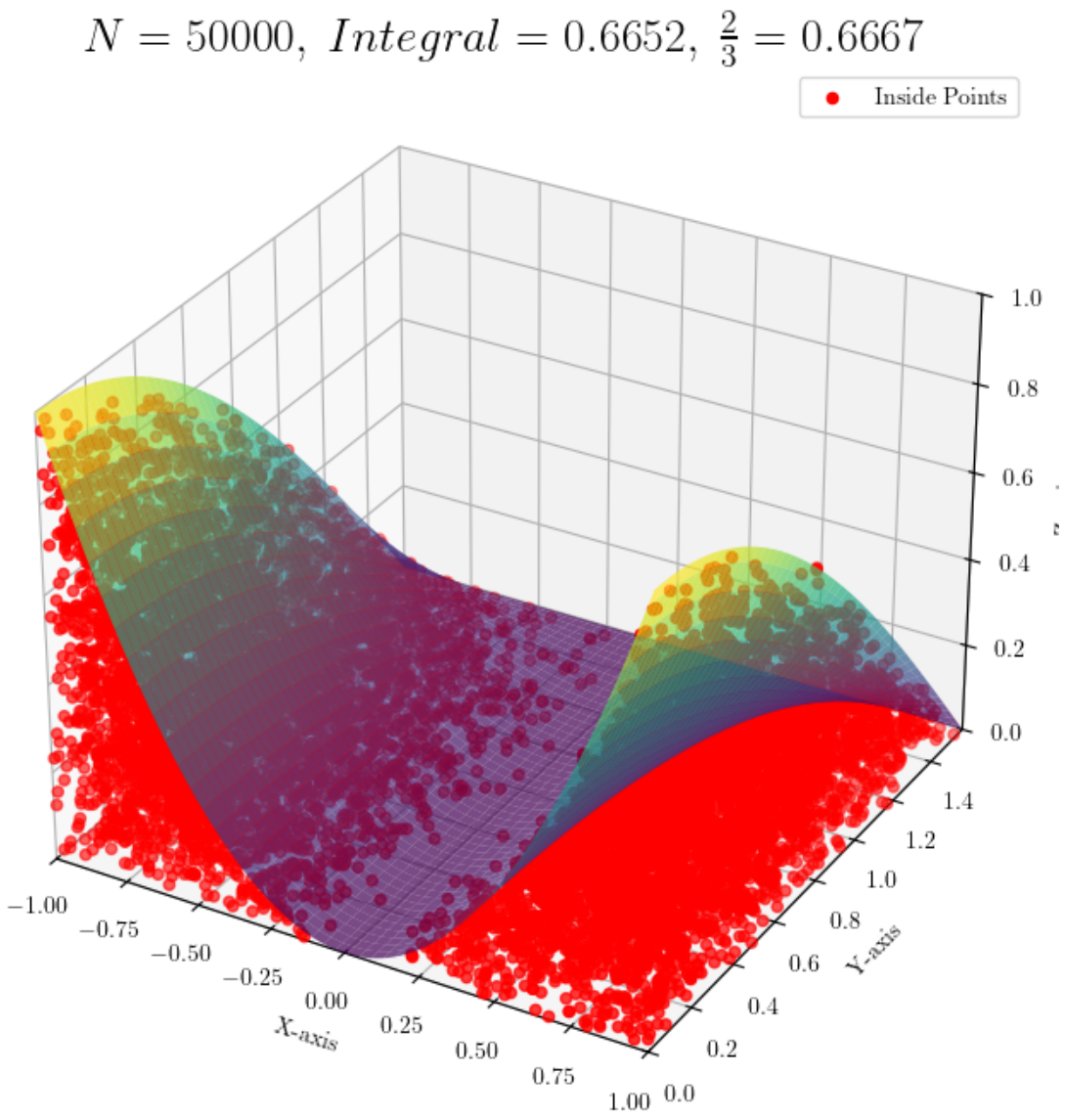
Estimation of z = f(x,y) = x^2.cos(y)

[2]:
```
from mpl_toolkits.mplot3d import Axes3D

def plot_tool(points, points_in):
    x = np.linspace(-1, 1, 100)  # x domain
    y = np.linspace(0, np.pi / 2, 100)  # y domain
    X, Y = np.meshgrid(x, y)  # Create grid
```

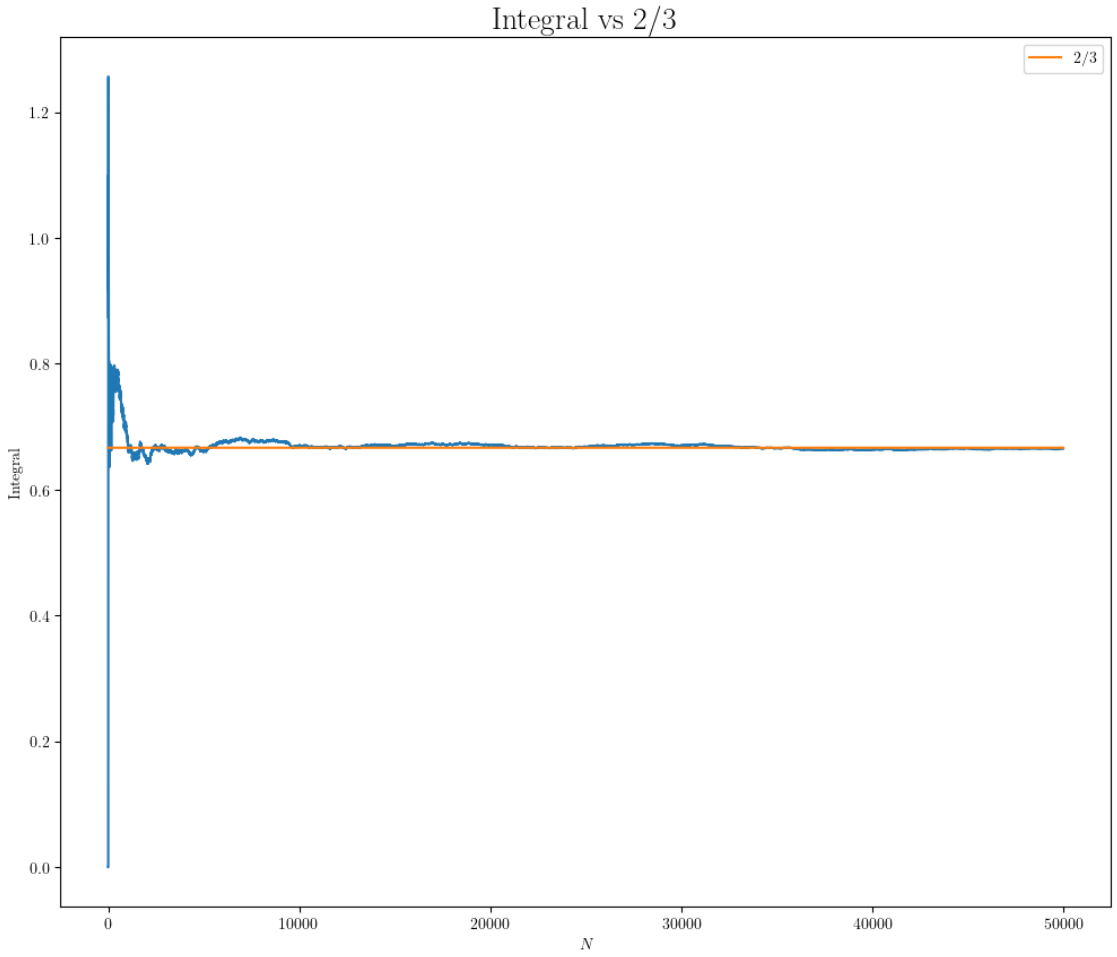
```
    Z = X**2 * np.cos(Y)  # Compute f(x, y)

    x,y,z=np.asarray(points)[:,0], np.asarray(points)[:,1], np.asarray(points)[:
↪,2]
    x_in,y_in, z_in=np.asarray(points_in)[:,0], np.asarray(points_in)[:,1], np.
↪asarray(points_in)[:,2]

    plt.clf()
    fig = plt.figure("3D Scatter", figsize=(8, 8))
    ax = fig.add_subplot(111, projection='3d')
    # ax.scatter(x, y, z, color='b', label='Points')
    ax.scatter(x_in, y_in, z_in, color='r', label='Inside Points')

    ax.plot_surface(X, Y, Z, cmap='viridis', alpha=0.7, edgecolor='none')

    ax.set_title(r"$N=%i,\; Integral=%.4f,\; \frac{2}{3}=%.4f$" % (len(points),␣
↪np.pi * len(points_in) / len(points), 2 / 3), fontsize=20)
    ax.set_xlim([-1, 1])
    ax.set_ylim([0, np.pi/2])
    ax.set_zlim([0, 1])
    ax.set_xlabel('X-axis')
    ax.set_ylabel('Y-axis')
    ax.set_zlabel('Z-axis')

    ax.legend()
    display.display(plt.gcf())
    display.clear_output(wait=True)
    time.sleep(0.01)
```

[4]:
```
shots=50000
points=[]
frac=[]
freq=1000

for n in range(shots):
    x = 2*rand() - 1
    y = np.pi/2 * rand()
    z = rand()
    points+=[np.asarray([x,y,z])]

    points_in=[p for p in points if p[2] <=p[0]**2*np.cos(p[1])]

    frac+=[len(points_in)/len(points)]

    if (n+1)%freq==0:
        plot_tool(points, points_in)
```

```
<Figure size 640x480 with 0 Axes>
```

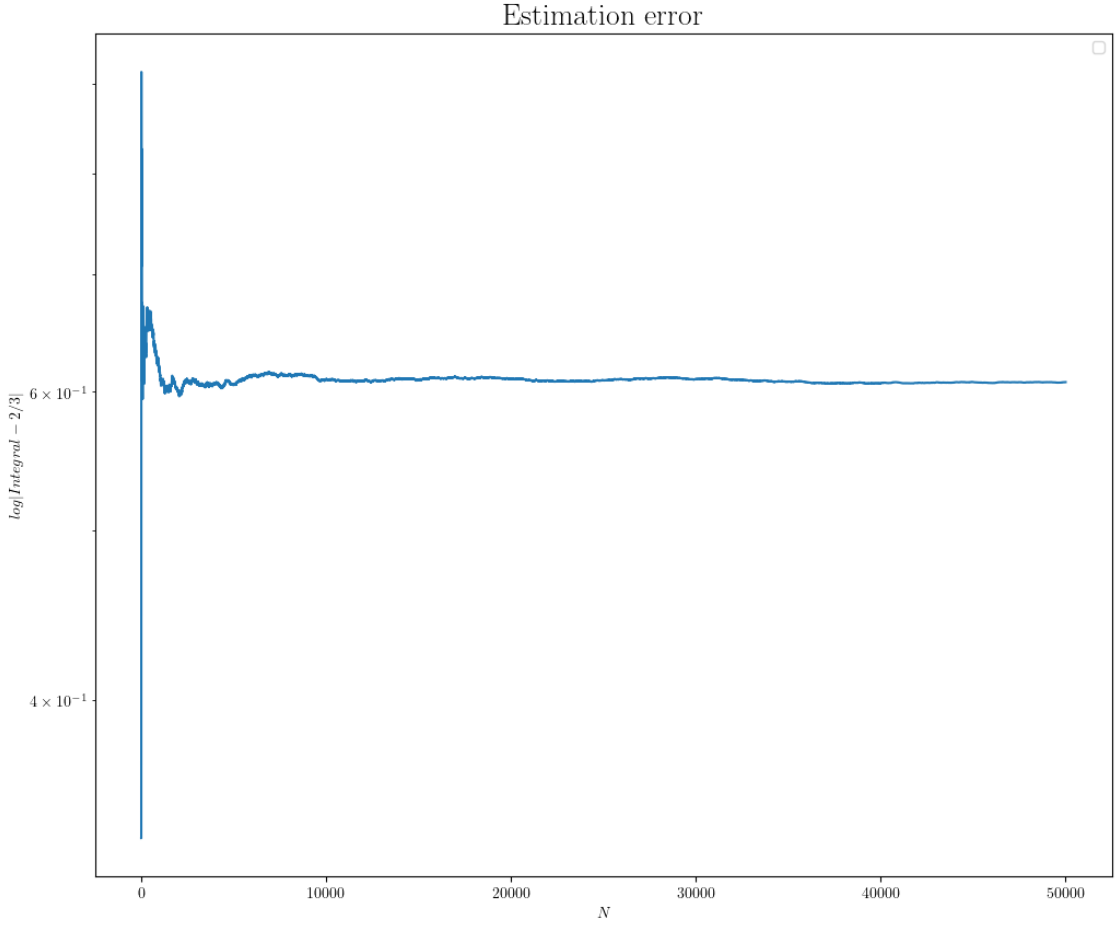


[5]:
```
plt.figure('evol',figsize=(12,10))

plt.plot(range(len(frac)), np.pi*np.asarray(frac))
plt.plot(range(len(frac)), 2/3*np.ones(len(frac)), label=r'2/3')
plt.xlabel(r'$N$')
plt.ylabel(r"Integral")
plt.title(r"Integral vs 2/3", fontsize=20)

plt.legend()
plt.show()
```



[9]:
```
plt.figure('error',figsize=(12,10))

plt.plot(abs(np.pi**np.asarray(frac) -  2/3*np.ones(len(frac))))
plt.yscale('log')
plt.xlabel(r'$N$')
plt.ylabel(r'$log|Integral - 2/3|$')
plt.title(r"Estimation error", fontsize=20)

plt.legend()
plt.show()
```

```
/tmp/ipykernel_63942/2033949171.py:9: UserWarning: No artists with labels found
to put in legend.  Note that artists whose label start with an underscore are
ignored when legend() is called with no argument.
  plt.legend()
```


Estimation error
$log|Integral - 2/3|$
6×10^{-1}
4×10^{-1}
0
10000
20000
30000
40000
50000
N