Faraday_cage

May 5, 2022

Zhuikov Artyom, Lopatenko George

Modeling along the course of general physics (ISaT), 2 module

1 Faraday's cage model

Task: visualize the principle of the Faraday's cage.

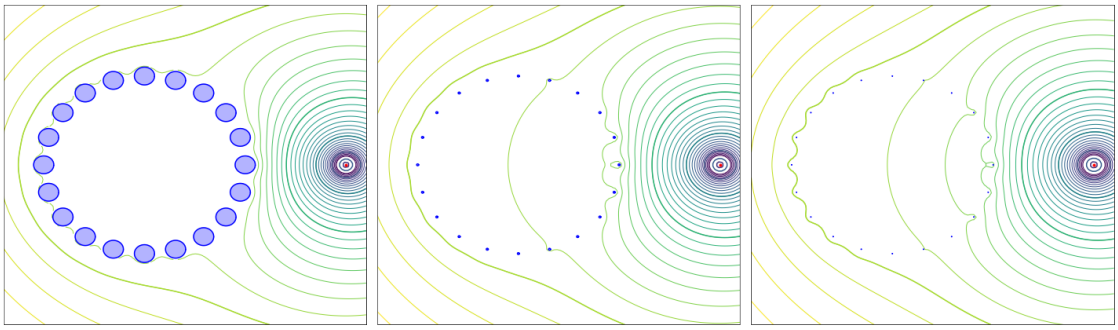
[38]:
```
import matplotlib.pyplot as plt
import numpy as np
import random
import math
```

[39]:
```
class FaradayCage:
    def __init__(self, wire_center, radius, zs):
        wire_center = np.asarray(wire_center)
        n = len(wire_center)
        N = max(0, int(np.round(4 + 0.5 * np.log10(radius))))
        M, K = 3 * N + 2, n * N
        d = np.exp(2j * np.pi * np.arange(M) / M)
        z = wire_center[:,np.newaxis] + radius * d
        z = z.reshape(-1)
        k = np.arange(1, N + 1)
        zc = z[:,np.newaxis] - wire_center
        zck = zc[...,np.newaxis] ** (-k)
        zck = zck.reshape(len(z), -1)
        b = np.hstack([0, -np.log(np.abs(z - zs))])
        A = np.column_stack([np.hstack([0, -np.ones(len(z))]), np.vstack([np.
 ↪ones(n), np.log(np.abs(zc))]), np.vstack([np.zeros(K), np.real(zck)]),
                np.vstack([np.zeros(K), np.imag(zck)])])
        x = np.linalg.lstsq(A, b, rcond=None)[0]
        e, x = x[0], x[1:]  # potential on wires
        d, x = x[:n], x[n:] # charge on wires
        a = x[:K].reshape(n,-1)
```

```
        b = x[K:].reshape(n,-1)
        self.a, self.b, self.wire_center, self.d, self.e, self.k, self.radius,␣
↪self.zs = a, b, wire_center, d, e, k, radius, zs
    def potential(self, arr):
        arr = np.asarray(arr)
        arrc = arr[...,np.newaxis] - self.wire_center
        arrck = arrc[...,np.newaxis] ** (-self.k)
        rc = np.abs(arrc)
        compl = np.log(np.abs(arr - self.zs))
        compl += np.dot(np.log(rc), self.d) + np.einsum('...ij,ij', np.
↪real(arrck), self.a) + np.einsum('...ij,ij', np.imag(arrck), self.b)
        compl[np.any(rc < self.radius, -1)] = np.nan
        return compl
    def field(self, arr):
        arr = np.asarray(arr)
        arrc = arr[...,np.newaxis] - self.wire_center
        arrck = -self.k * arrc[...,np.newaxis] ** (-self.k - 1)
        compl = 1 / (arr - self.zs)
        compl += np.dot(1 / arrc, self.d) + np.einsum('...ij,ij', arrck, self.
↪a) - np.einsum('...ij,ij', arrck, self.b) * 1j
        return np.conj(compl)
```

[40]:
```
n, radius, zs, N = 20, [0.1, 0.01, 0.001], 2, 64
wire_center = np.exp(2j * np.pi * np.arange(n) / n)
d = np.exp(2j * np.pi * np.arange(N) / N)
x, y = np.meshgrid(np.linspace(-1.4, 2.2, 120), np.linspace(-1.8, 1.8, 120))
plt.figure(figsize=(20, 6))
for i, r in enumerate(radius):
    u = FaradayCage(wire_center, r, zs).potential(x + 1j * y)
    disks = wire_center[:,np.newaxis] + r * d
    plt.subplot(1, 3, i + 1)
    plt.contour(x, y, u, linewidths = 1.5)
    plt.contour(x, y, u, levels = np.linspace(-2, 1.2, 33), linewidths = 1)
    plt.fill(np.real(disks.T), np.imag(disks.T), color = (0.7, 0.7, 1))
    plt.plot(np.real(disks.T), np.imag(disks.T), 'b')
    plt.plot(np.real(zs), np.imag(zs), '.r')
    plt.xticks([])
    plt.yticks([])
    # plt.axis('equal')
    plt.xlim([np.min(x), np.max(x)])
    plt.ylim([np.min(y), np.max(y)])

plt.tight_layout()
plt.show()
```



[42]:
```
# quantity of dots, radius of dots, external point charge, number of plot␣
 ↪points on a disk
n, r, zs, N = [20, 40, 60], 0.01, 2, 64
d = np.exp(2j * np.pi * np.arange(N) / N)
x, y = np.meshgrid(np.linspace(-1.4, 2.2, 120), np.linspace(-1.8, 1.8, 120))
plt.figure(figsize=(20, 6))
for i, n in enumerate(n):
    c = np.exp(2j * np.pi * np.arange(n) / n)
    disks = c[:,np.newaxis] + r * d
    plt.subplot(1, 3, i + 1)
    plt.contour(x, y, FaradayCage(c, r, zs).potential(x + 1j * y), levels = np.
 ↪linspace(-2, 1.2, 33), linewidths = 1)
    plt.fill(np.real(disks.T), np.imag(disks.T), color = (0.7, 0.7, 1))
    plt.plot(np.real(disks.T), np.imag(disks.T), 'b')
    plt.plot(np.real(zs), np.imag(zs), '.r')
    plt.xticks([])
    plt.yticks([])
    plt.xlim([np.min(x), np.max(x)])
    plt.ylim([np.min(y), np.max(y)])
plt.tight_layout()
plt.show()
```