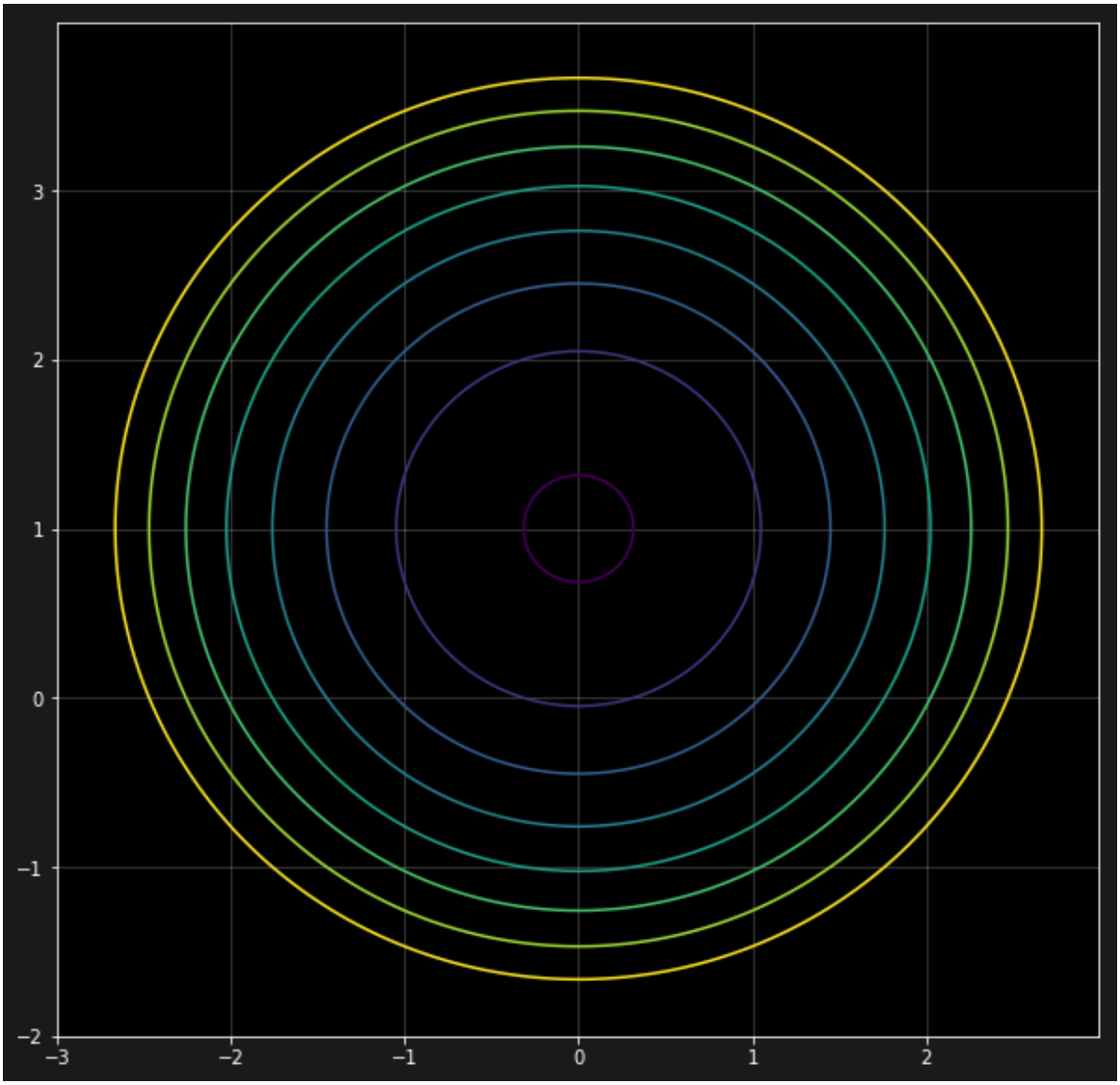
Метод Пауэлла

Решаемая задача - поиск локального минимума функции $f(x) : \mathbb{R}^n \to \mathbb{R}$

Рассмотрим следующую функцию: $f(x, y) = x^2 + (y - 1)^2$.

In [1]:

```
from matplotlib import pyplot as plt
import numpy as np
%matplotlib inline
plt.style.use('dark_background')

f = lambda x, y : np.power(x, 2) + (y-1)**2

def plot_func(X, Y, Z):
    plt.figure(facecolor='0.1', figsize=(10, 10))
    plt.grid(True, alpha=0.25)
    plt.contour(X, Y, Z, levels=levels)

X = np.arange(-3, 3, 0.01)
Y = np.arange(-2, 4, 0.01)
levels = np.arange(0.1, 8, 1)
X, Y = np.meshgrid(X, Y)
Z = f(X, Y)

plot_func(X, Y, Z)
plt.show()
```


3
2
1
0
−1
−2
−3
−2
−1
0
1
2

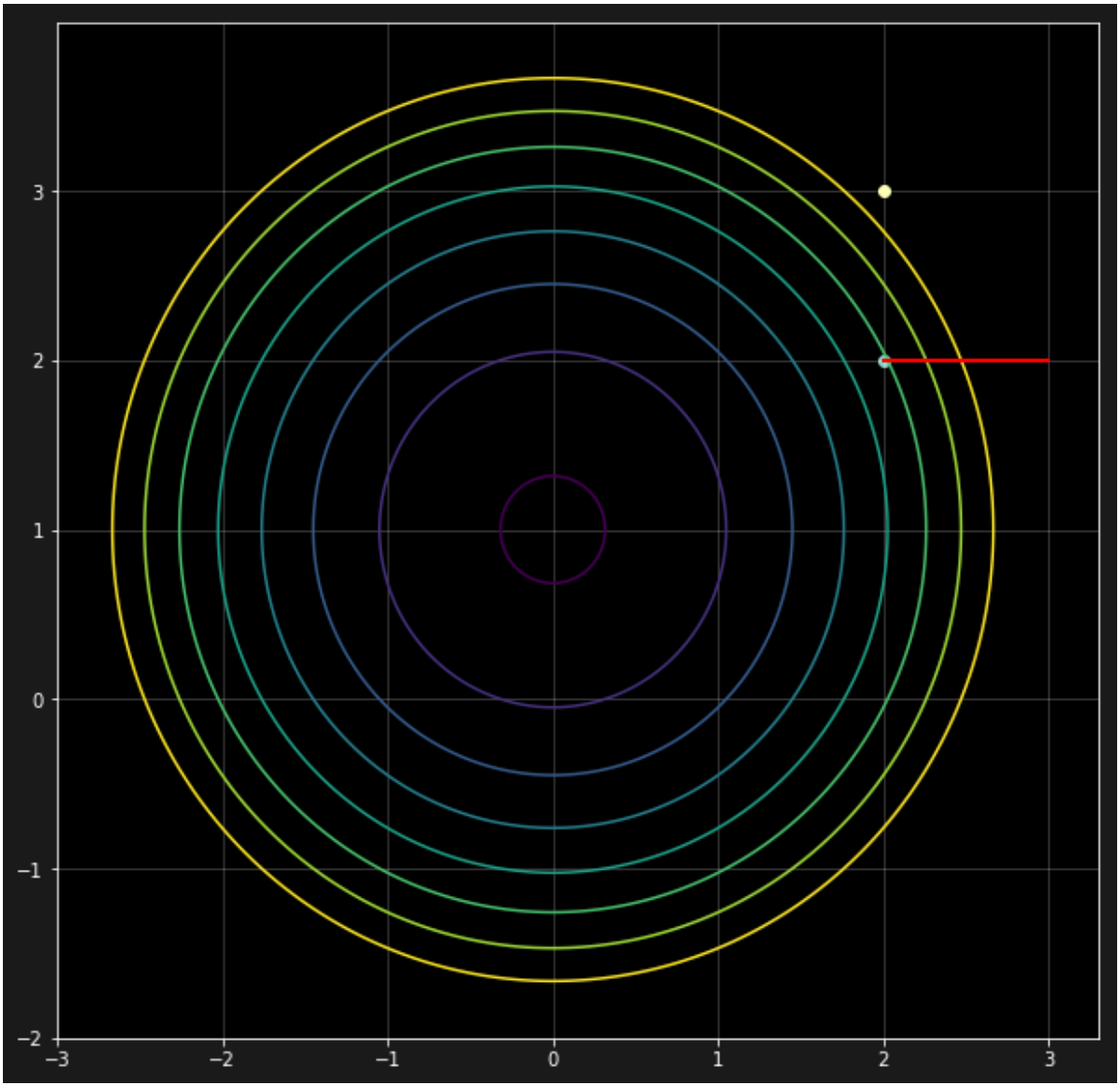
Метод Пауэлла

Выбор начальных точек и направления

$x_1 = (2, 2)$
$x_2 = (2, 3)$
$\Delta x = (1, 0)$

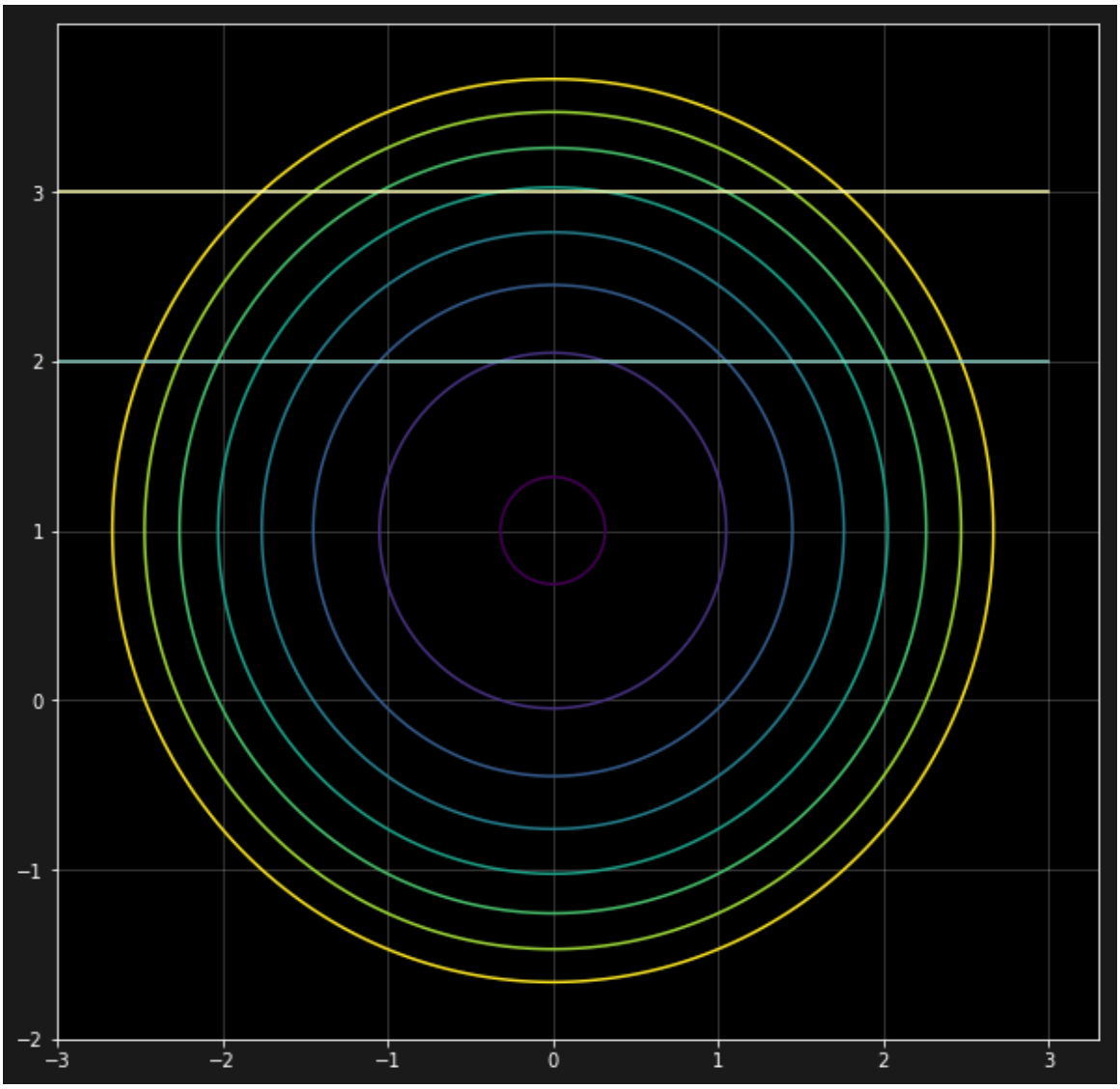
In [2]:

```
x_1 = np.array((2, 2))
x_2 = np.array((2, 3))
dx = np.array((1, 0))

plot_func(X, Y, Z)
plt.scatter(*x_1)
plt.scatter(*x_2)
vec = np.array([x_1, x_1 + dx])
plt.plot(vec[:, 0], vec[:, 1], linewidth=2, color='r')
pass
```



In [3]:

```
def func_mod(f, point, coef):
    return f(*(point * coef))

line1 = np.array([x_1 + a * dx for a in np.linspace(-5, 1, 1000)])
line2 = np.array([x_2 + a * dx for a in np.linspace(-5, 1, 1000)])
plot_func(X, Y, Z)
plt.plot(line1[:, 0], line1[:, 1])
plt.plot(line2[:, 0], line2[:, 1])
pass
```



In [4]:

```
def get_min_on_line(f, line):
    func_vals = np.array([f(*x) for x in line])
    min_ = np.argmin(func_vals)
    min_point = line[min_]
    return min_point

p1 = get_min_on_line(f, line1)
p2 = get_min_on_line(f, line2)
new_vec = p2 - p1
line3 = np.array([p1 + new_vec * c for c in np.linspace(-4, 2, 1000)])
p_min = get_min_on_line(f, line3)

plot_func(X, Y, Z)
plt.plot(line1[:, 0], line1[:, 1], linewidth=2, color='r', zorder=1)
plt.plot(line2[:, 0], line2[:, 1], linewidth=2, color='r', zorder=1)
plt.plot(line3[:, 0], line3[:, 1], linewidth=3, color='y', zorder=1)
plt.scatter(*x_1, zorder=2)
plt.scatter(*x_2, zorder=2)
plt.scatter(*p1, color='r', zorder=2)
plt.scatter(*p2, color='r', zorder=2)
plt.scatter(*p_min, color='r', zorder=2, s=80)
pass
```