

Выполнил: Литти Тимофей

Теоретическая основа

Для двухкомпонентной системы используется приближение эффективной среды.

Уравнение баланса полей имеет вид:

$$p_1 \epsilon_{eff} \frac{3E_0}{2\epsilon_{eff} + \epsilon_1} + p_2 \epsilon_{eff} \frac{3E_0}{2\epsilon_{eff} + \epsilon_2} = E_0$$

Для численного решения методом простой итерации ($x = g(x)$) используются выраженные формы уравнения:

1. Для $p \rightarrow 0$ (малая концентрация компонента 1) удобна формула $g_2(x, p)$:

$$\epsilon_{eff} = \frac{1}{3(1-p)} (2\epsilon_{eff} + \epsilon_2) \left(1 - p \frac{3\epsilon_{eff}}{2\epsilon_{eff} + \epsilon_1} \right)$$

2. Для $p \rightarrow 1$ (высокая концентрация компонента 1) удобна формула $g_1(x, p)$:

$$\epsilon_{eff} = \frac{2\epsilon_{eff} + \epsilon_1}{3p} \left(1 - (1-p) \frac{3\epsilon_{eff}}{2\epsilon_{eff} + \epsilon_2} \right)$$

Код

```
import numpy as np
import matplotlib.pyplot as plt

def solve_effective_permittivity(p, eps1, eps2, tolerance=1e-6,
max_iter=1000):
    """
    Вычисляет эффективную диэлектрическую проницаемость методом простой
    итерации.

    Параметры:
    p : float - объемная концентрация компонента 1 (от 0 до 1)
    eps1 : float - диэлектрическая проницаемость компонента 1
    eps2 : float - диэлектрическая проницаемость компонента 2
    """
    if p == 0:
        return eps2
    if p == 1:
        return eps1
```

```

x = (eps1 * p) + (eps2 * (1 - p))

for _ in range(max_iter):
    x_old = x

    if p < 0.5:
        term1 = 1 / (3 * (1 - p))
        term2 = 2 * x + eps2
        term3 = 1 - p * (3 * x / (2 * x + eps1))
        x = term1 * term2 * term3
    else:
        term1 = (2 * x + eps1) / (3 * p)
        term2 = 1 - (1 - p) * (3 * x / (2 * x + eps2))
        x = term1 * term2

    if abs(x - x_old) < tolerance:
        return x

return x

```

```
eps_sio2 = 1.46
```

```
eps_si = 15.6
```

```
p_values = np.linspace(0, 1, 100)
```

```
eps_eff_2_1 = [solve_effective_permittivity(p, eps_sio2, eps_si) for p in
p_values]
```

```
plt.figure(figsize=(10, 5))
```

```
plt.plot(p_values, eps_eff_2_1, label="Si - SiO2 Mixture", color="blue")
```

```
# plt.title(r"Task 2.1: Effective Permittivity  $\epsilon_{\text{eff}}(p)$  for
SiO2 in Si")
```

```
plt.xlabel(r"Volume Concentration of SiO2 ( $p$ )")
```

```
plt.ylabel(r"Effective Permittivity ( $\epsilon_{\text{eff}}$ )")
```

```
plt.grid(True)
```

```
plt.legend()
```

```
plt.show()
```

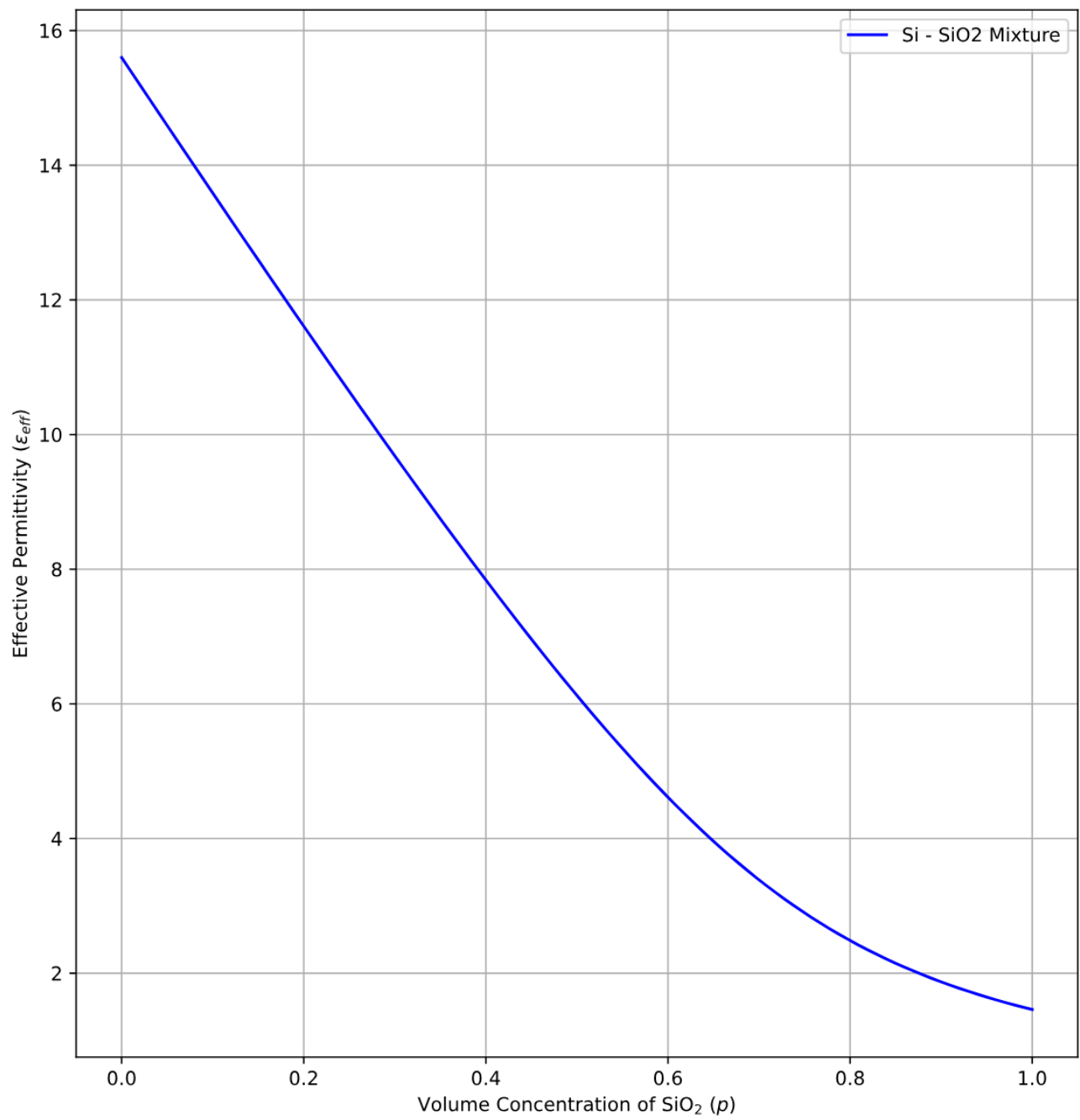
```
eps_ferrite = 5.2
```

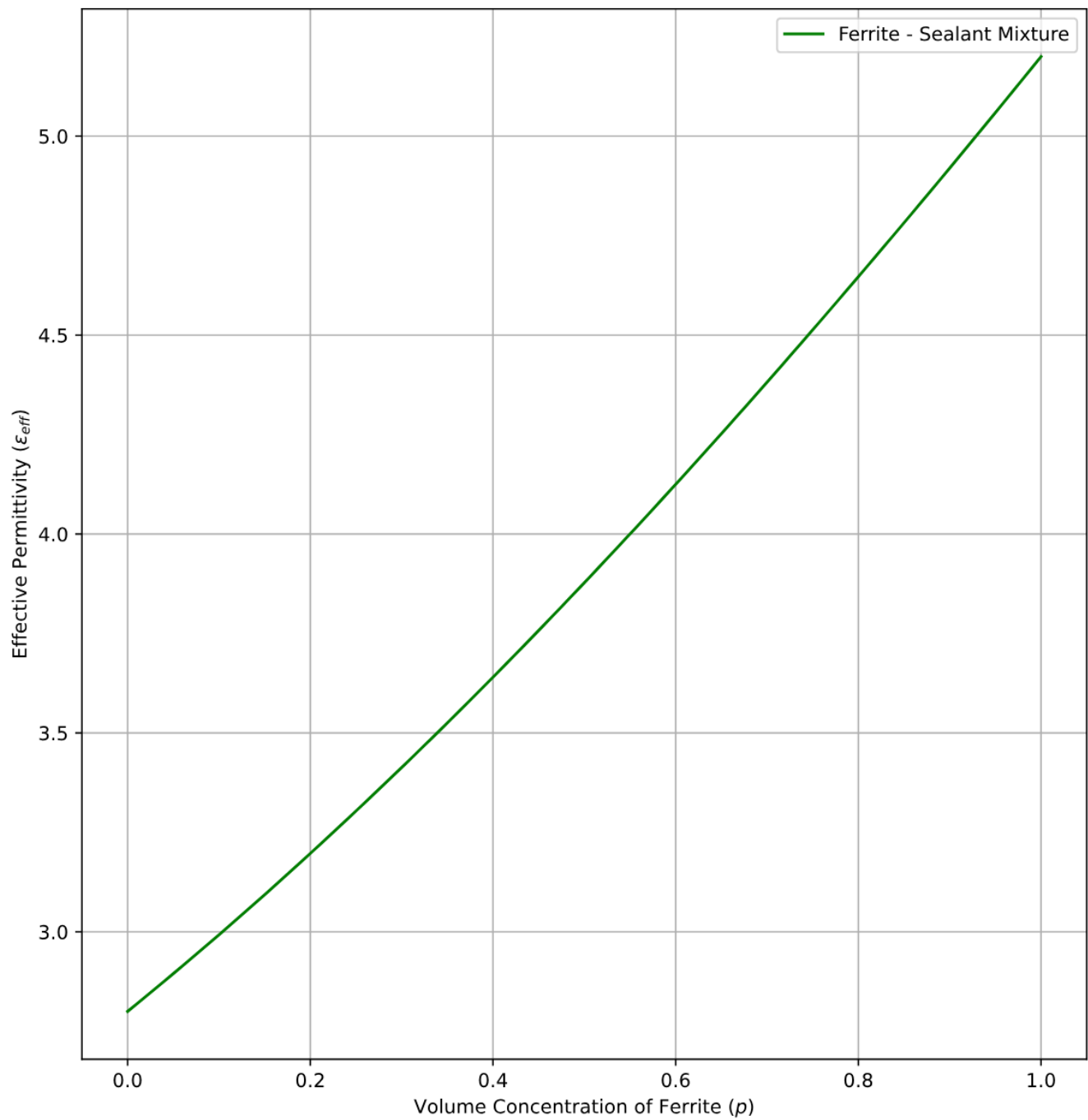
```
eps_matrix = 2.8
```

```
eps_eff_2_2 = [
    solve_effective_permittivity(p, eps_ferrite, eps_matrix) for p in
p_values
]
```

```
plt.figure(figsize=(10, 5))
plt.plot(p_values, eps_eff_2_2, label="Ferrite - Sealant Mixture",
color="green")
# plt.title(r"Task 2.2: Effective Permittivity  $\epsilon_{eff}(p)$  for
Ferrite in Matrix")
plt.xlabel(r"Volume Concentration of Ferrite ( $p$ )")
plt.ylabel(r"Effective Permittivity ( $\epsilon_{eff}$ )")
plt.grid(True)
plt.legend()
plt.show()
```

График





Пояснение к решению

1. **Алгоритм:** Реализована функция `solve_effective_permittivity`, которая решает уравнение методом итераций.

- Для обеспечения стабильности алгоритма при $p \rightarrow 0$ и $p \rightarrow 1$ программа автоматически переключается между двумя эквивалентными формулами ($g1$ и $g2$), приведенными в документе. Это предотвращает деление на ноль.

2. **Задача 2.1:** Изучается процесс окисления кремния.

- $\epsilon_1 (\text{SiO}_2) = 1.46$.
- $\epsilon_2 (\text{Si}) = 15.6$.
- Параметр p соответствует концентрации SiO_2 .

Вывод: Поскольку p — это концентрация SiO_2 , увеличение p означает замещение материала с высокой проницаемостью (15.6) на материал с низкой проницаемостью (1.46). Поэтому общее значение ϵ_{eff} **падает**.

3. **Задача 2.2:** Композит Mn-Zn феррита.

- $\epsilon_1 (\text{Феррит}) = 5.2$.
- $\epsilon_2 (\text{Матрица}) = 2.8$.
- Параметр p соответствует концентрации феррита.

Вывод: Здесь увеличение p означает добавление более проницаемого материала в менее проницаемую среду. Поэтому общее значение ϵ_{eff} **растет**.