nickel

24 сентября 2024 г.

1 Оптическое определение никеля

[50]:
```
                       import numpy as np
                       import matplotlib.pyplot as plt
                       from scipy.stats import linregress
```

[49]:
```
                       def subtract_baseline(signal_x, signal_y, baseline_x,␣
↪baseline_y):
                           """
                           Subtracts the baseline from the signal after␣
↪interpolating the baseline.

                           Parameters:
                           - signal_x: Array of x values for the signal.
                           - signal_y: Array of y values for the signal.
                           - baseline_x: Array of x values for the baseline.
                           - baseline_y: Array of y values for the baseline.

                           Returns:
                           - result_y: Array of y values after subtracting the␣
↪baseline from the signal.
                           """

                       # Create an interpolation function for the baseline
                       interpolate_baseline = interp1d(baseline_x, baseline_y,␣
↪bounds_error=False, fill_value="extrapolate")

                       # Interpolate the baseline values at the x points of the␣
↪signal
                       interpolated_baseline_y = interpolate_baseline(signal_x)

                       # Subtract the interpolated baseline from the signal
                       result_y = signal_y - interpolated_baseline_y

                       return result_y
                       def f(X, A, B):
                       return A * X + B
                       def f_1(Y, A, B):
                           '''
                           Y = AX + B
                           '''
                       return (Y - B) / A
```

1.1 Калибровка

[ ]:

```
                       c_std =
                       V_flask_cal = 100.00 #ml
                       V_cal = np.array([ 4.,  6.,  8., 10., 12., 14., 16.]) #ml
                       c_cal = V_cal * c_std
```

1.1.1 Недифференциальная

[21]:

```
                       l_cal = 0.3 #Cuvette length cm
                       A_cal = np.array([]) # Optical density

                       #A_0 = np.float64(INPUT) # Needed if no baseline
                       #A = A - A_0
```

[ ]:

```
                       slope_cal, intercept_cal = linregress(c_cal, A_cal)[0],␣
↪linregress(c_cal, A_cal)[1]
```

[ ]:

```
                       plt.scatter(c_cal, A_cal)
                       c_cal_fine = np.linspace(np.min(c_cal), np.max(c_cal),␣
↪100)
                       plt.plot(c_cal_fine, f(c_cal_fine, slope_cal,␣
↪intercept_cal))
```

1.1.2 Дифференциальная

[ ]:

```
                       l_dif_cal =
                       A_dif_cal = np.array([]) # Needed if no baseline
                       #A_0_dif = A_dif[3]
                       #A_dif = A_dif - A_0_dif
```

[ ]:

```
                       slope_dif_cal, intercept_dif_cal = linregress(c_dif_cal,␣
↪A_dif_cal)[0], linregress(c_dif_cal, A_dif_cal)[1]
```

[ ]:

```
                       plt.scatter(c_dif_cal, A_dif_cal)

                       c_dif_cal_fine = np.linspace(np.min(c_dif_cal), np.
↪max(c_dif_cal), 100)
                       plt.plot(c_dif_cal_fine, f(c_dif_cal_fine,␣
↪slope_dif_cal, intercept_dif_cal))
```

1.2 Определение

1.2.1 Недифференциальное

[ ]:

```
m_steel_1 =
m_steel_2 =
V_exer_flask = 250.0
V_aliq = 25.00
```

[ ]:

```
A_1 = np.array([])
f_1(A_1, slope_cal, intercept_cal)
```

1.2.2 Дифференциальное

[ ]:

```
A_2 = np.array([])
f_1(A_2, slope_cal, intercept_cal)
```