

ELK313E - Power Transmission Lines Project Assignment

Instructor: Zehra Elif BATMAN

Sena ERSOY
040200434

9 January 2025

Digit	conductor	region	σ	h	σ_{max}	E_{th}
L ₁	Haub	IV	450m	25m	$\sigma_{max} = 11 \text{ kg/mm}^2$	$E_{th} = 18,9 \times 10^{-6} \text{ 1/C}^\circ$
					$d = 21,77 \text{ mm}$	$\frac{1}{E_{cl}} = E = 7700 \text{ kg/mm}^2$
			$h = 0.5$		$q = 280,84 \text{ mm}^2$	$H = \sigma \cdot q$
			$w_c = 972,8 \text{ kg/km}$ $= 0,9728 \text{ kg/m}$		$w_c = h \sqrt{\sigma} = 0,5 \sqrt{21,77}$ $= 2,33 \text{ kg/m}$	$H_{max} = 11 \cdot 280,84 = 3089 \text{ kg}$
			$t_{min} = -30^\circ \text{C}$			
			$t_{max} = 40^\circ \text{C}$			

$$s_h = 2 H_{max} \sqrt{\frac{6 E_{th} (t_{min} + 5)}{w_c^2 - (w_c + w_c)^2}} \quad (\text{for region IV})$$

$$= 2 \cdot 3089 \sqrt{\frac{6 \cdot 18,9 \times 10^{-6} (-30 + 5)}{(0,9728)^2 - (0,9728 + 2,33)^2}} = 6178 \sqrt{\frac{-1,835 \times 10^{-3}}{-9,962}} = 104,22 \text{ m (critical span)}$$

$\sigma_h < \sigma$, so max tensile occurs when there is additional load on the conductor

$$t_h = \frac{E_{cl} \cdot w_c \cdot H_{ice}}{E_{th} (w_c + w_c) q} - 5 \quad (\text{for region IV}) \quad H_{ice} = H_{max}$$

$$t_h = 48,32^\circ \text{C} > 40^\circ \text{C} \quad (\text{critical temperature})$$

$t_h > t_{max}$, so maximum sag occurs when there is additional load on the conductor

Conductor State Change Equation

$$H_2^2 \left[H_2 + q \frac{E_{th}}{E_{cl}} (t_h - t_f) + \frac{\sigma^2 (w_c)^2 q}{24 H_1^2 E_{cl}} - H_1 \right] = \frac{\sigma^2 w_c^2 q}{24 E_{cl}}$$

Figure 1: Critical Span and Temperature Calculations

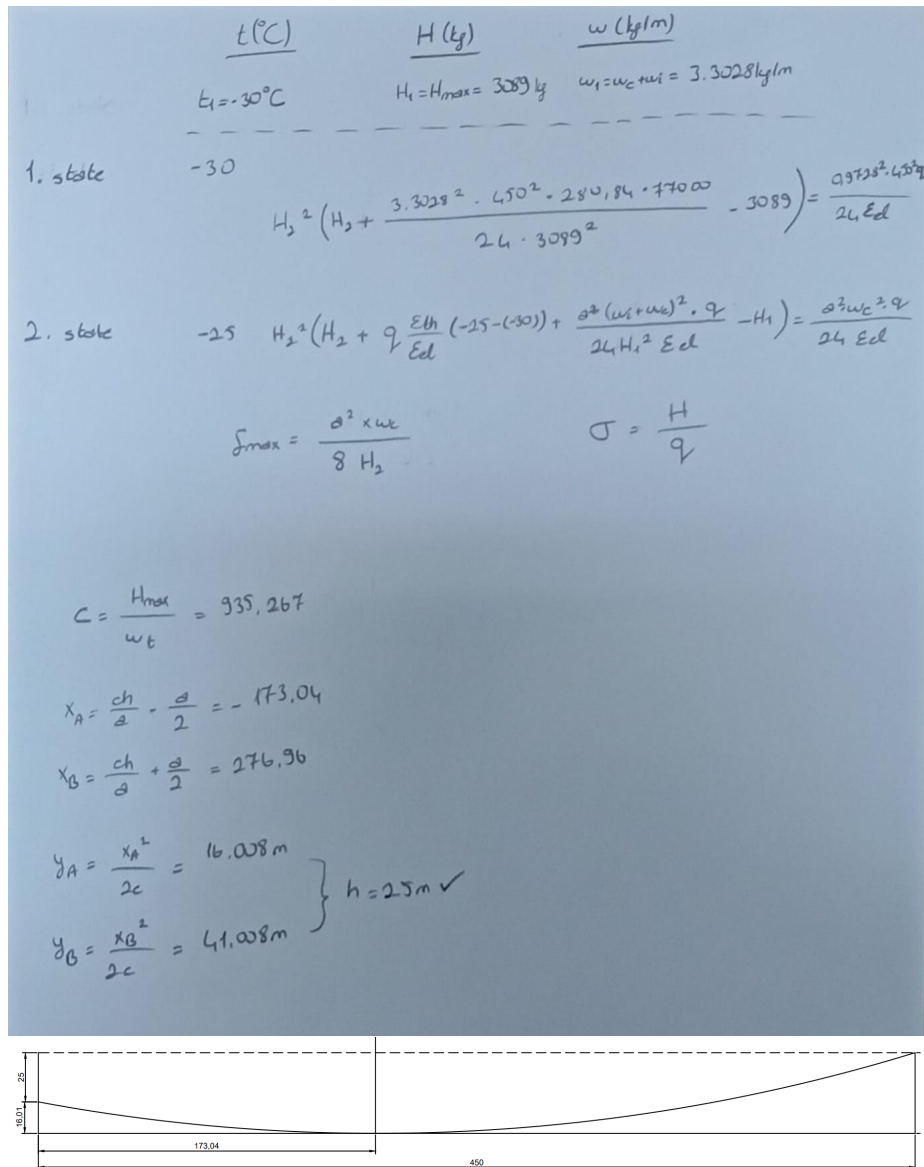


Figure 2: Scaling the drawing by 1/1500 resulted in annotations becoming excessively large, and I was unable to adjust them appropriately. Therefore, the drawing is presented in its unscaled form.

Python Code

```
import numpy as np
import pandas as pd
from scipy.optimize import fsolve

# Define the constants
H1 = 3089
q = 280.84
wi = 2.33
wc = 0.9728
a = 450
Eel = 0.00012987
Eth = 0.0000189
t1 = -30

# Generate t2 values from -30 to 40 in steps of 5
t2_values = np.arange(-30, 41, 5)

# Define the equation to solve for H2
def equation(H2, t2):
    term1 = H2 + q * (Eth / Eel) * (t2 - t1)
    term2 = (a**2 * (wi + wc)**2 * q) / (24 * H1**2 * Eel)
    term3 = H1
    rhs = (a**2 * wc**2 * q) / (24 * Eel)
    return H2**2 * (term1 + term2 - term3) - rhs

# Solve for H2 for each t2 value
H2_values = []
for t2 in t2_values:
    # Use fsolve to find the root of the equation
    H2_solution = fsolve(equation, H1, args=(t2,))[0] # Initial
    # guess is H1
    H2_values.append(H2_solution)

# Create a DataFrame for the results
results = pd.DataFrame({
    "t2": t2_values,
    "H2": H2_values
})

# Add "stress" column (H2 / q)
results["stress"] = results["H2"] / q

# Add "sag" column (a^2 * wc / (8 * H2))
results["sag"] = (a**2 * wc) / (8 * results["H2"])

# Save the final results to an Excel file
file_path_final_excel = "H2_Stress_Sag_Results_Final.xlsx"
results.to_excel(file_path_final_excel, index=True, index_label="
    Index", sheet_name="Results")

# Inform the user of the file creation
file_path_final_excel
```

Exported Results in Excel Format

The following table shows the exported results from the Python code above:

t2	H2	stress	sag
-30	960.14	3.42	25.65
-25	955.07	3.40	25.78
-20	950.08	3.38	25.92
-15	945.17	3.37	26.05
-10	940.33	3.35	26.19
-5	935.56	3.33	26.32
0	930.86	3.31	26.45
5	926.23	3.30	26.59
10	921.67	3.28	26.72
15	917.17	3.27	26.85
20	912.74	3.25	26.98
25	908.36	3.23	27.11
30	904.05	3.22	27.24
35	899.80	3.20	27.37
40	895.61	3.19	27.49

Table 1: Exported results showing t_2 , H_2 , stress, and sag.