Ústav fyzikální elektroniky Přírodovědecké fakulty Masarykovy univerzity

FYZIKÁLNÍ PRAKTIKUM

Fyzikální praktikum 1

Zpracoval: Milan Suk **Naměřeno:** 19. března 2018

Obor: F **Skupina:** PO 8:00 **Testováno:**

Úloha č. **5:** **Měření modul pružnosti pevných látek**

1. Úvod

V první části jsem měl určovat modul pružnosti drátu přímou metodou. Na drát toušťky d se pustepně zavěšovala závaží a měřila se závislost prověšení Δl na celkové hmotnosti m.

$$\Delta l = \frac{4gl}{\pi d^2 E} m \tag{1}$$

Ze směrnice $k = \frac{4gl}{\pi d^2 E}$ pak můřu určit modul pružnosti E.

V druhé části jsem měřil modul pružnosti z průhybu nosníku. Ze změřené závislosti y jsem určil modul pružnosti E.

$$y = \frac{mgl^3}{4Ea^3b} \tag{2}$$

Ze směrnice jsem potom podobně jako u předchozího experimentu získat samotnou hodnotu E.

V poslední části se pak měl určit modul pružnosti drátu ve smyku dynamickou metodou. Ze změřené periody kmitů T lze modul pružnosti G určit jako

$$G = \frac{16\pi m R^2 l}{5r^4T^2} \tag{3}$$

2. Výsledky

2.1. Modul pružnosti v tahu měřený přímou metodou

Z naměřených stačí spočítat lineární regresí sklon, ten je roven konstantě k.

```python
import numpy
import matplotlib.pyplot as plt
from scipy import stats

data = [float(i.strip()) for i in open('data_1').readlines()[:21]]
weights = [float(i.strip()) for i in open('vahy_1').readlines()[:10]]

x = list()
last = 0
for i in weights:
    last = last + i
    x.append(last)

x.insert(0, 0)
```

```

x_rev = list(reversed(x))

y = data[:11]
y_rev = data[10:]

slope, intercept, r_value, p_value, std_err = stats.linregress(x, y)
print(slope, std_err)
slope, intercept, r_value, p_value, std_err = stats.linregress(x_rev, y_rev)
print(slope, std_err)
```

Tento script výhodí směrnici i s odchylkou.

$$k_1 = (432 \pm 3) \cdot 10^{-6} m \cdot kg^{-1}$$
$$k_2 = (425 \pm 1) \cdot 10^{-6} m \cdot kg^{-1}$$

tloušťka drátu je $d = 5.0 \cdot 10^{-4}\, m$ a výchozí délka drátu $l = 1.565\, m$. Modul pružnosti lze pak určit pomocí rovnice

$$E = \frac{4gl}{\pi d^2 k} \tag{4}$$

A odtud vychází modul pružnosti

$$E_1 = (18 \pm 4) \cdot 10^{10} Pa$$
$$E_2 = (18 \pm 4) \cdot 10^{10} Pa$$

2.2. Modul pružnosti v tahu z průhybu nosníku

Při zpracování dat jsem postupoval podobně jako u předchozího měření, zjistil jsem směrnici k a z ní jsem určil modul pružnosti E.

```
import numpy
import matplotlib.pyplot as plt
from scipy import stats

def is_number(n):
    try:
        float(n)
    except ValueError:
        return False
    return True

data_raw = [float(i.strip()) * 1e-3 for i in open('data_2').readlines() if is_number(i)
    ]
weights = [float(i.strip()) * 1e-3 for i in open('vahy_2').readlines()[:10]]
sizes = [float(i.strip()) * 1e-3 for i in open('data_2_rozmery').readlines() if
    is_number(i)]

a, ua, b, ub = list(), list(), list(), list()

for i in range(5):
    a.append(numpy.average(sizes[20 * i : 20 * i + 10]))
    b.append(numpy.average(sizes[20 * i + 10 : 20 * i + 20]))

    ub.append(numpy.average((sizes[20 * i : 20 * i + 10] - a[-1])**2))
    ua.append(numpy.average((sizes[20 * i + 10 : 20 * i + 20] - b[-1])**2))

data_points = [(0, 9), (9, 30), (30, 51), (51, 72), (72, 93)]
data = list()
for i, j in data_points:
    data.append(data_raw[i:j])

_x = list()
```

```
_x.append(0)
last = 0
for i in weights:
    last = last + i
    _x.append(last)

x = [_x[:5]]
for _ in range(4):
    x.append(_x)


def analyze(x, y):
    x_rev = list(reversed(x))
    k1, _, _, _, err1 = stats.linregress(x, y[:len(x)])
    k2, _, _, _, err2 = stats.linregress(x_rev, y[len(x) - 1:])
    return (k1, err1, k2, err2)


def calculate_e(a, ua, b, ub, l, ul, k, uk):
    from math import pow, sqrt
    g = 9.81
    pow2 = lambda x: pow(x, 2)
    E = g * pow(l, 3) / (4 * k * pow(a, 3) * b)
    uE = g * pow(l, 3) / (4 * k * pow(a, 3) * b) * sqrt(pow2(3 * ua / a) + pow2(ub / b)
     + pow2(3 * ul / l) + pow2(uk / k))

    return (E, uE)


l, ul = (0.898, 0.0005)
for i in range(5):
    k1, uk1, k2, uk2 = analyze(x[i], data[i])
    print(calculate_e(a[i], ua[i], b[i], ub[i], l, ul, k1, uk1))
    print(calculate_e(a[i], ua[i], b[i], ub[i], l, ul, k2, uk2))
```

výsledné moduly pružnosti jsou

$$E_{uhlik} = (9 \pm 1)\ GPa$$

$$E_{mosaz} = (4603 \pm 12)\ GPa$$

$$E_{ocel} = (8372 \pm 1)\ GPa$$

$$E_{hlinik} = (12108 \pm 25)\ GPa$$

$$E^{(2)}_{mosaz} = (96.9 \pm 0.2)\ GPa$$

2.3. Modul pružnosti ve smyku

Následujícím scriptem jsem určil hodnty neznámých veličin a jejich nejistoty, script nakonec vypočítá výsledný modul pružnosti.

```
import numpy
from math import pi, sqrt, pow

m = 5.905
um = 0.0005
_T = [3.96, 3.99, 4.07, 3.99, 3.99, 4.07, 4.02, 3.99, 3.96, 3.91] # s
_R = [9.576, 9.562, 9.562, 9.572, 9.560, 9.524, 9.570, 9.578, 9.514, 9.568] # cm
_l = [51.5, 51.4, 51.6, 51.4, 51.5, 51.5, 51.4, 51.5, 51.4, 51.6] # cm
_r = [1.00, 1.00, 0.99, 1.00, 1.00, 0.99, 0.99, 1.00, 0.99, 1.00] # mm

# uprava na zakladni jednotky
_R = [i * 1e-2 for i in _R]
_l = [i * 1e-2 for i in _l]
_r = [i * 1e-3 for i in _r]
```

```

T = numpy.average(_T)
R = numpy.average(_R)
l = numpy.average(_l)
r = numpy.average(_r)

uT = numpy.average((_T - T)**2)
uR = numpy.average((_R - R)**2)
ul = numpy.average((_l - l)**2)
ur = numpy.average((_r - r)**2)

pow2 = lambda x: pow(x, 2)
G = 16 * pi * m * pow(R, 2) * l / (5 * pow(r, 4) * pow(T, 2))
uG = G * sqrt(pow2(um /m) + pow2(2 * uR / R) + pow2(ul / l) + pow2(4 * ur / r) + pow2(2
    * uT / T))

print(G, uG)
```

$$G = (1778 \pm 2) \cdot 10^7 \, Pa$$

3. Zhodnocení měření, závěr

U měřeních, kde bylo potřeba měřit hmotnostiválečků, je zajímavé, že přesto, že jednotlivé hmotnosti se liší nepatrně, celková hmotnost by vyrobila rozdíl několik gramů. Výsledky se vůči tabulkovým hodnotám někdy liší docela výrazně, zejména hodnoty modulu pružnosti v tahu pohybující se v tisisích GPa jsou pravděpodobně zatíženy nějakou chybou. Zbytek se alespoň řádově shoduje.