

VASE__data

January 2, 2026

1 VASE data: Fits and data

- The folder ‘./VASE’ has all the ellipsometry data
- Here we will open and plot the data, to act as a visualisation but also as a tool to navigate the datasets
- Contained are the datasets for the substrate and underlying layers too: It can be refit by users using software of choice

```
[1]: import numpy as np
from matplotlib import pyplot as plt
# Colours for plotting
colours = ["#fd7f6f", "#7eb0d5", "#b2e061", "#bd7ebe", "#ffb55a", "#ffee65",
↪ "#beb9db", "#fdcce5", "#8bd3c7"]

def plot_ellipsometry(data_path, trans_data_path, angles, savepath=None,
↪ title=None):
    '''Plot ellipsometry data'''
    fig = plt.figure(figsize=(7.2, 4))

    ax_psi = plt.subplot2grid((2,3),(0,0), colspan=2)
    ax_delta = plt.subplot2grid((2,3),(1,0), colspan=2)
    ax_trans = plt.subplot2grid((2,3),(0,2), rowspan=2)

    data = np.loadtxt(data_path, skiprows=2)
    data_trans = np.loadtxt(trans_data_path, skiprows=2)

    for i, angle in enumerate(angles):
        # Psi
        ax_psi.plot(data[:, 0], data[:, 4*i + 1], linewidth=2, color=colours[i])
        ax_psi.plot(data[:, 0], data[:, 4*i +
↪ 3], linewidth=2, color='k', linestyle=':')
        ax_psi.set_xticks([])
        ax_psi.set_ylabel(r"$\Psi$")

        # Delta
        ax_delta.plot(data[:, 0], data[:, 4*i +
↪ 2], color=colours[i], linewidth=2, label=f"{angle}$^\circ$")
```

```

        ax_delta.plot(data[:, 0], data[:, 4*i + 4], linewidth=2, color='k', linestyle=':')
        ax_delta.set_xlabel("Wavelength (nm)")
        ax_delta.set_ylabel(r"$\delta$")

    # Transmission
    ax_trans.plot(data_trans[:, 0], data_trans[:, 1], color="#ea5545", linewidth=2)
    ax_trans.plot(data_trans[:, 0], data_trans[:, 2], linewidth=2, color='k', linestyle=':')
    ax_trans.set_xlabel("Wavelength (nm)")
    ax_trans.set_ylabel("Transmission Intensity")

    # Legend entry for model
    ax_delta.plot([], [], color='k', linewidth=4, linestyle=':', label='Model')
    ax_delta.legend(fontsize=10)

    fig.tight_layout()
    if savepath:
        plt.savefig(f'{savepath}.svg', bbox_inches='tight', transparent=True)
    if title:
        plt.suptitle(title, y=1.02, fontsize=12)
    plt.show()

def plot_nk(title, datapath, savepath=None):
    '''Plot n and k data'''
    fig, ax1 = plt.subplots()

    nk_data = np.loadtxt(datapath, skiprows=2)
    ax1.plot(nk_data[:,0], nk_data[:,1], linewidth=2, color="#b30000", label='n')
    ax2 = ax1.twinx()
    ax2.plot(nk_data[:,0], nk_data[:,2], linewidth=2, color="#5ad45a", label='k')
    ax1.plot([], [], color="#5ad45a", linewidth=2, label='k')
    ax1.legend(frameon=False)
    ax1.set_xlabel('Wavelength (nm)')
    ax1.set_ylabel('n')
    ax2.set_ylabel('k')

    fig.suptitle(title, fontsize=12)
    if savepath:
        plt.savefig(f'{savepath}.svg', bbox_inches='tight', transparent=True)
    plt.show()

def plot_nk_graded(title, datapath, savepath=None):
    '''Plot n and k data'''
    fig, ax1 = plt.subplots()

```

```

nk_data = np.loadtxt(datapath, skiprows=2)
ax1.plot(nk_data[:,0], nk_data[:,1], linewidth=2, color="#b30000", label='n_
↳(top)')
ax1.plot(nk_data[:,0], nk_data[:,3], linewidth=2, color="#b84c4cff",
↳label='n (bottom)')
ax2 = ax1.twinx()
ax2.plot(nk_data[:,0], nk_data[:,2], linewidth=2, color="#5ad45a", label='k_
↳(top)')
ax2.plot(nk_data[:,0], nk_data[:,4], linewidth=2, color="#099609", label='k_
↳(bottom)')
ax1.plot([],[], color="#5ad45a", linewidth=2, label='k (top)')
ax1.plot([],[], color="#099609", linewidth=2, label='k (bottom)')
ax1.legend(frameon=False)
ax1.set_xlabel('Wavelength (nm)')
ax1.set_ylabel('n')
ax2.set_ylabel('k')
fig.suptitle(title, fontsize=12)
if savepath:
    plt.savefig(f'{savepath}.svg', bbox_inches='tight', transparent=True)
plt.show()

```

1.1 Neat Pb: Substrate fits

- These are the base layers for the neat Pb compositions
- We fit just glass, glass/SnO2 and kept those constant
- The SnO2 thickness was roughly 40nm
- We used a scotch tape back-scatter layer so the glass was considered infinite

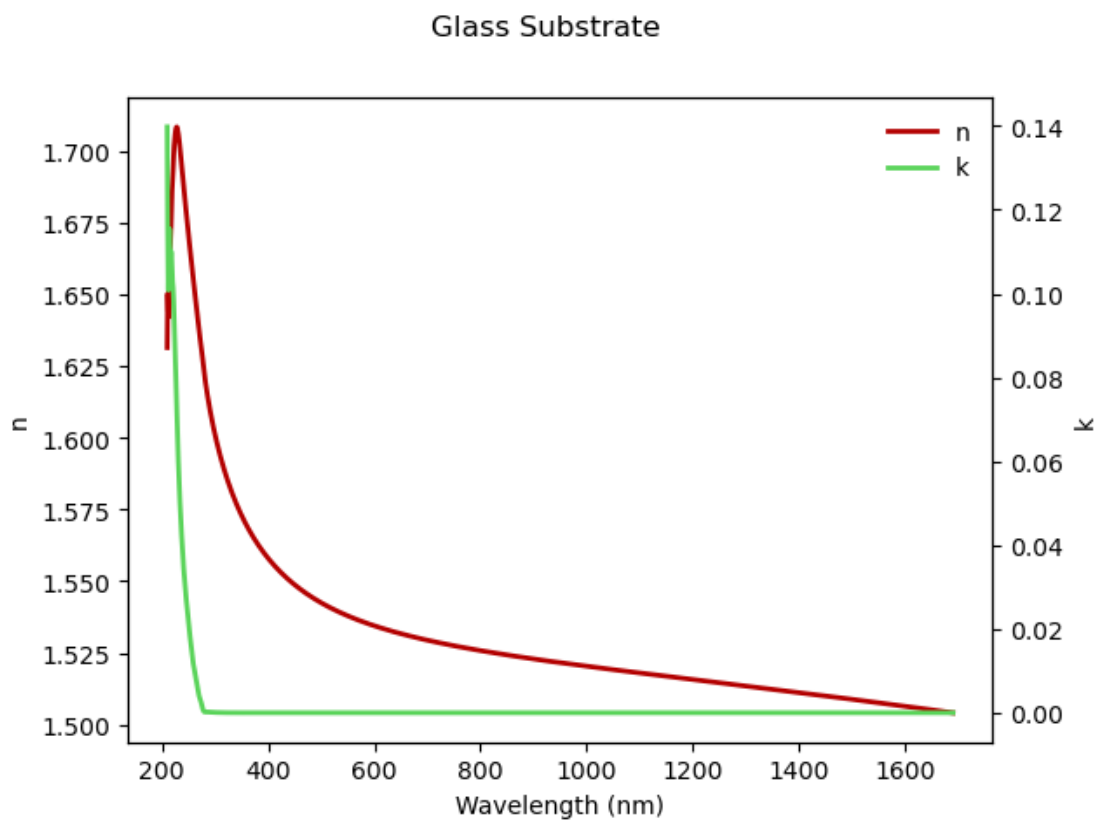
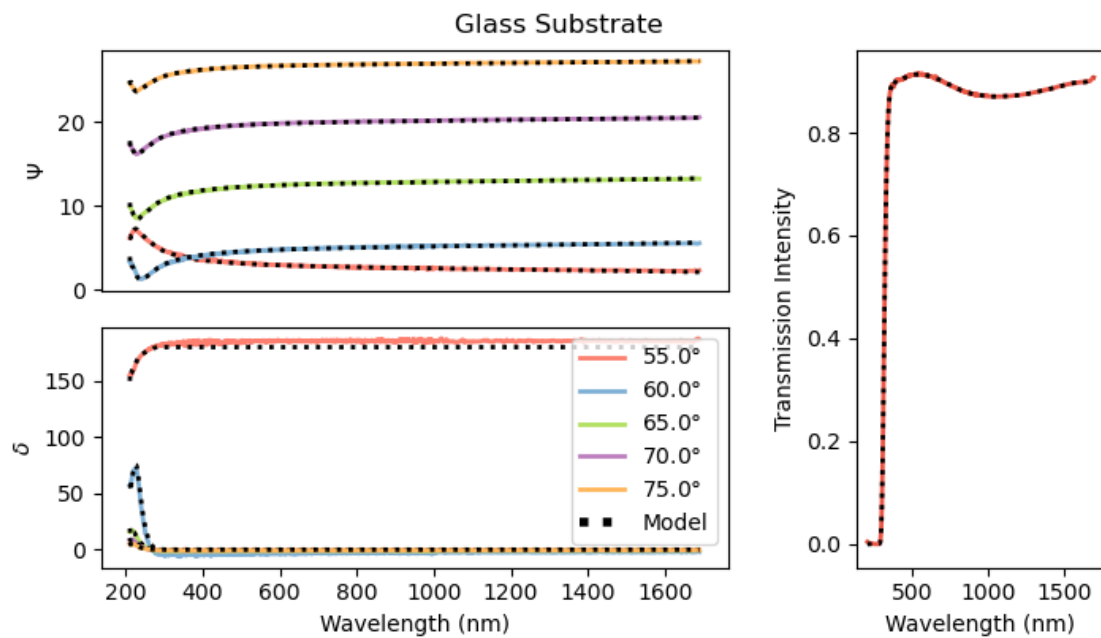
```

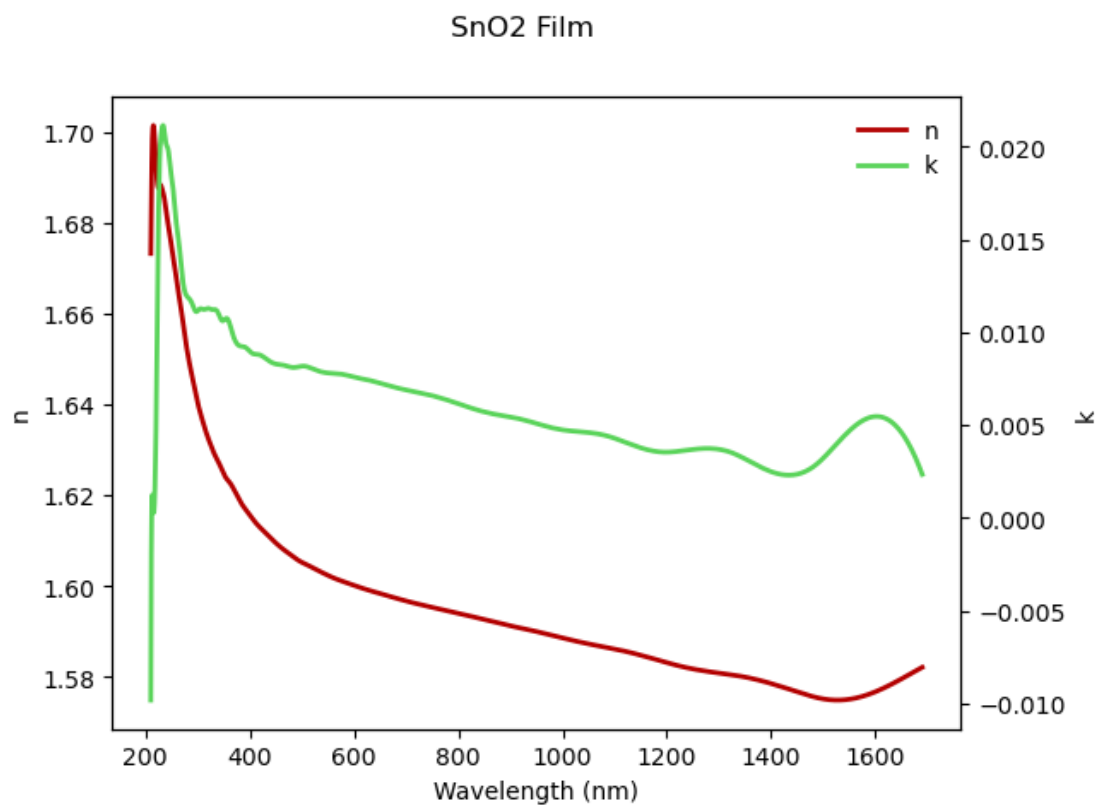
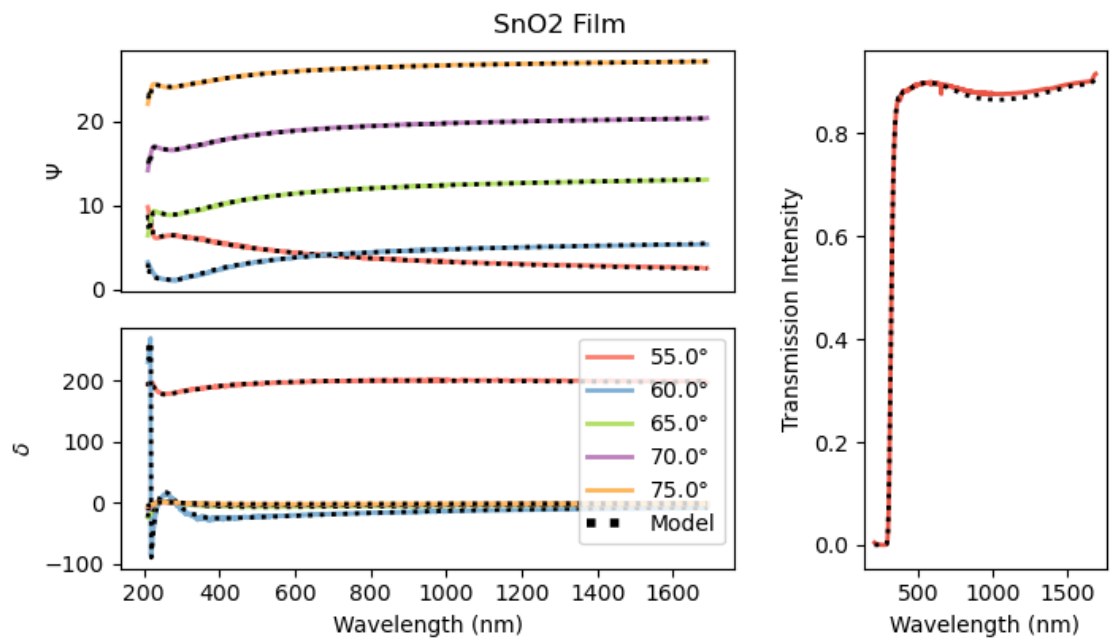
[2]: angles= [55.00, 60.00, 65.00, 70.00, 75.00]

glass_data_path = "VASE/glass_data.txt"
glass_trans_data_path = "VASE/glass_data_trans.txt"
glass_nk_data_path = "VASE/glass_nk.txt"
sno2_data_path = "VASE/snox_data.txt"
sno2_trans_data_path = "VASE/snox_data_trans.txt"
sno2_nk_data_path = "VASE/snox_nk.txt"

plot_ellipsometry(glass_data_path, glass_trans_data_path, angles, title="Glass_
↳Substrate")
plot_nk(title="Glass Substrate", datapath=glass_nk_data_path)
plot_ellipsometry(sno2_data_path, sno2_trans_data_path, angles, title="SnO2_
↳Film")
plot_nk(title="SnO2 Film", datapath=sno2_nk_data_path)

```



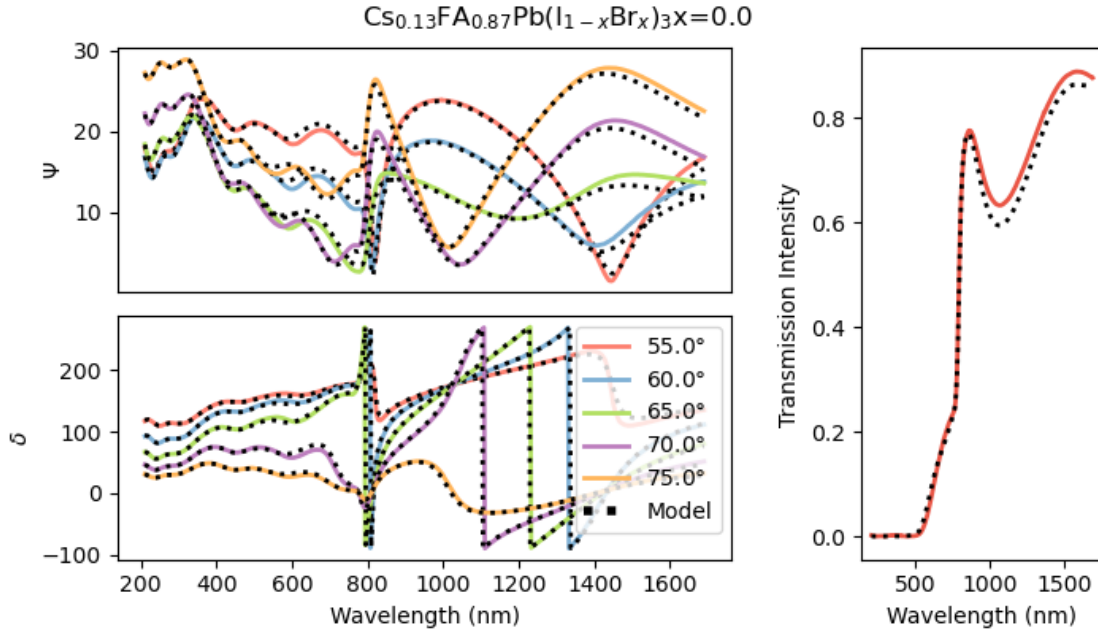


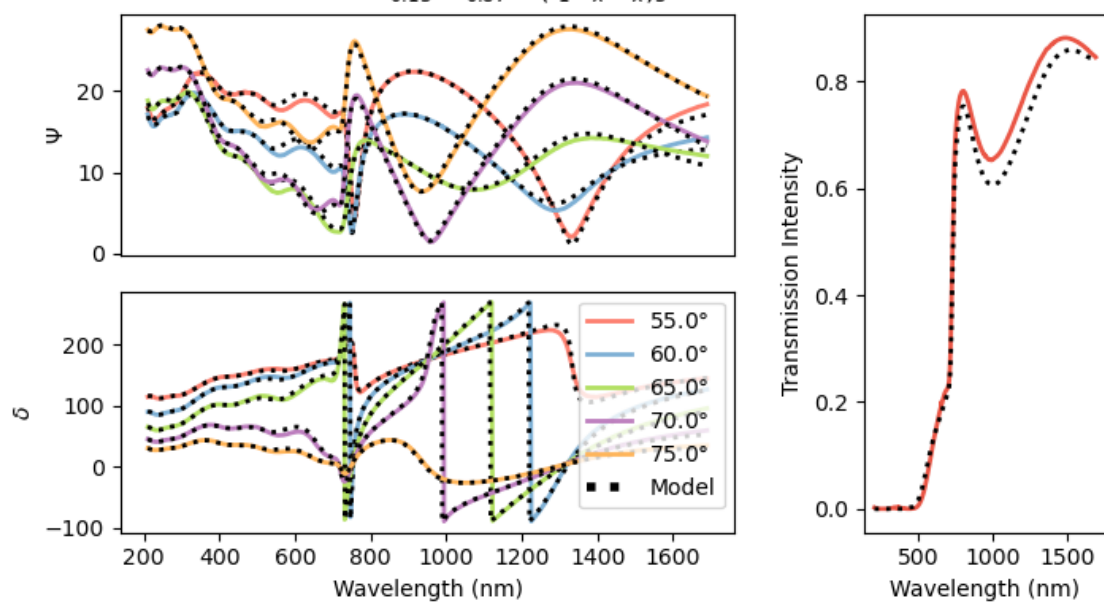
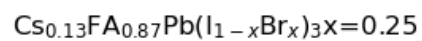
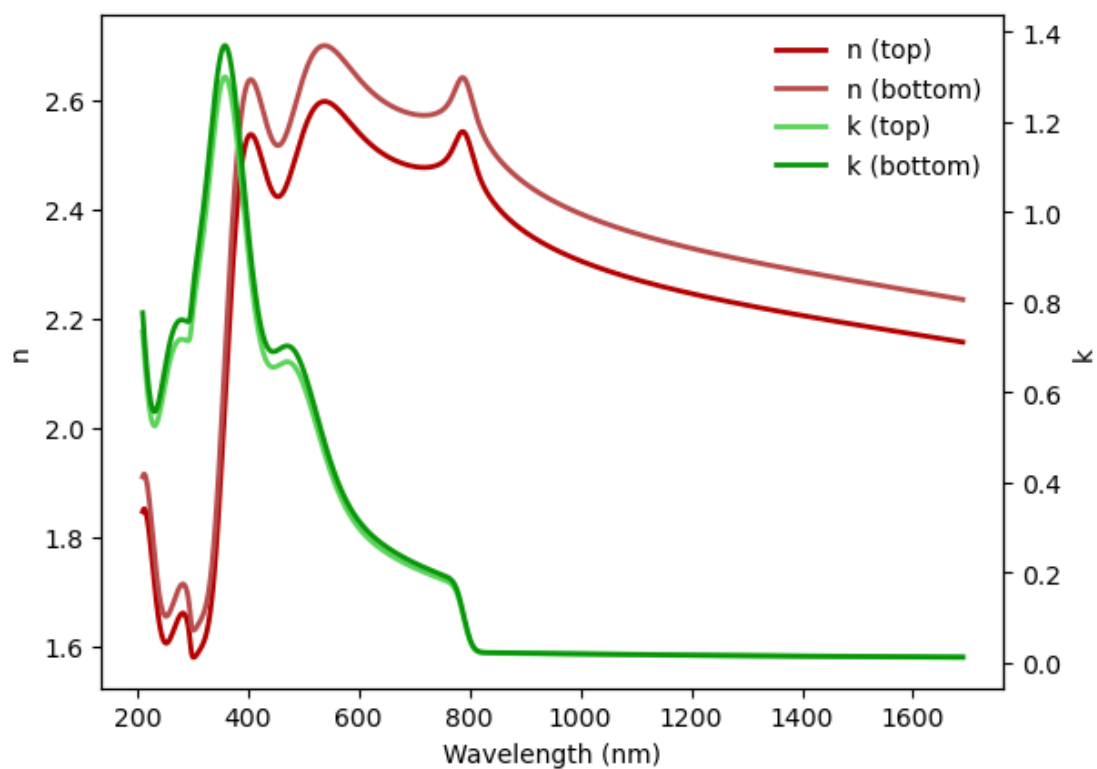
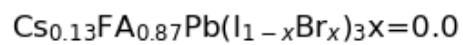
1.2 Neat Pb perovskite fits

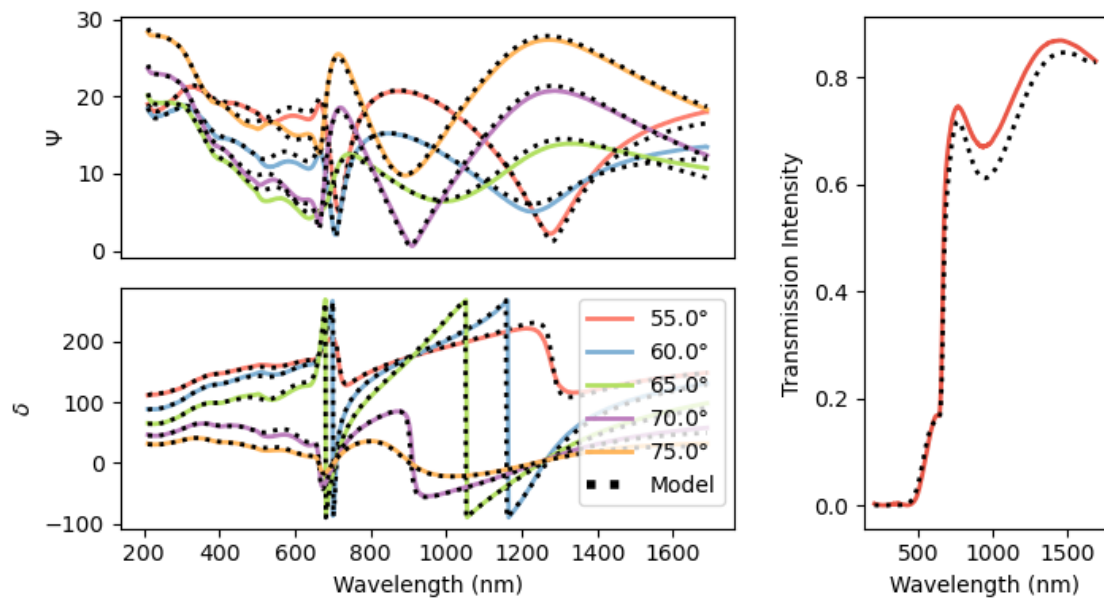
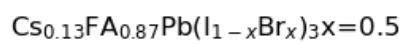
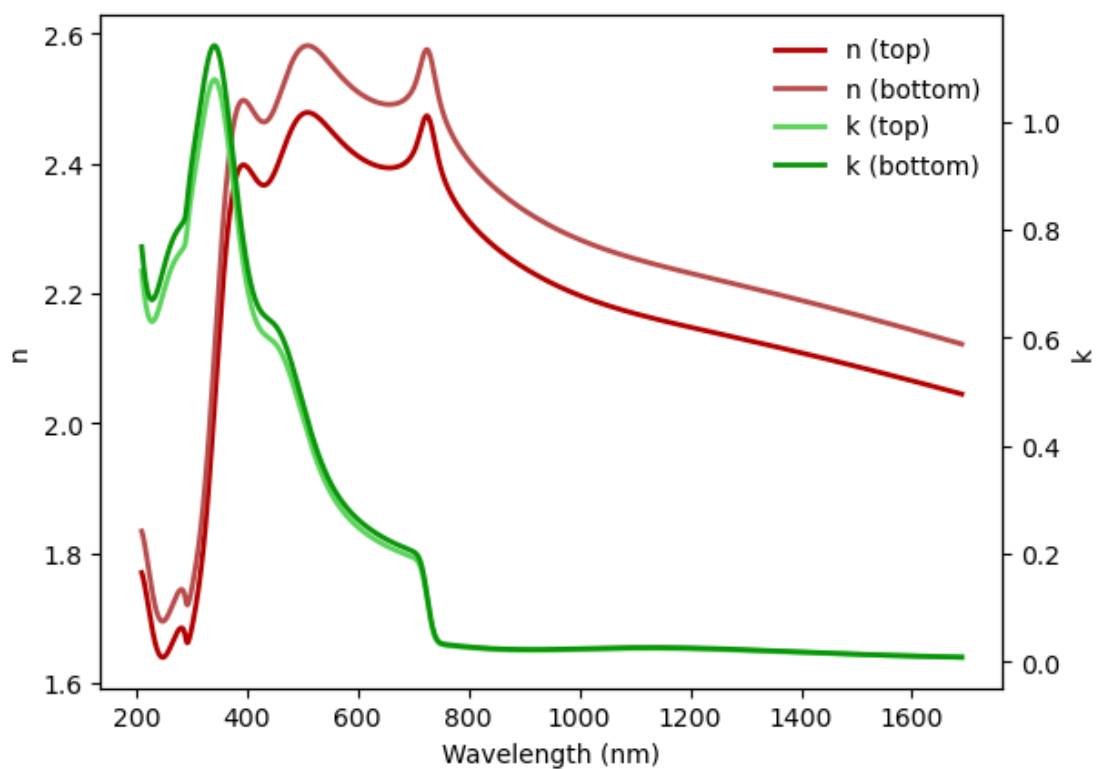
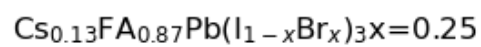
- For this layer we used a graded layer (Although the grading didn't have a huge impact)
- In the paper we will use the bottom layer values

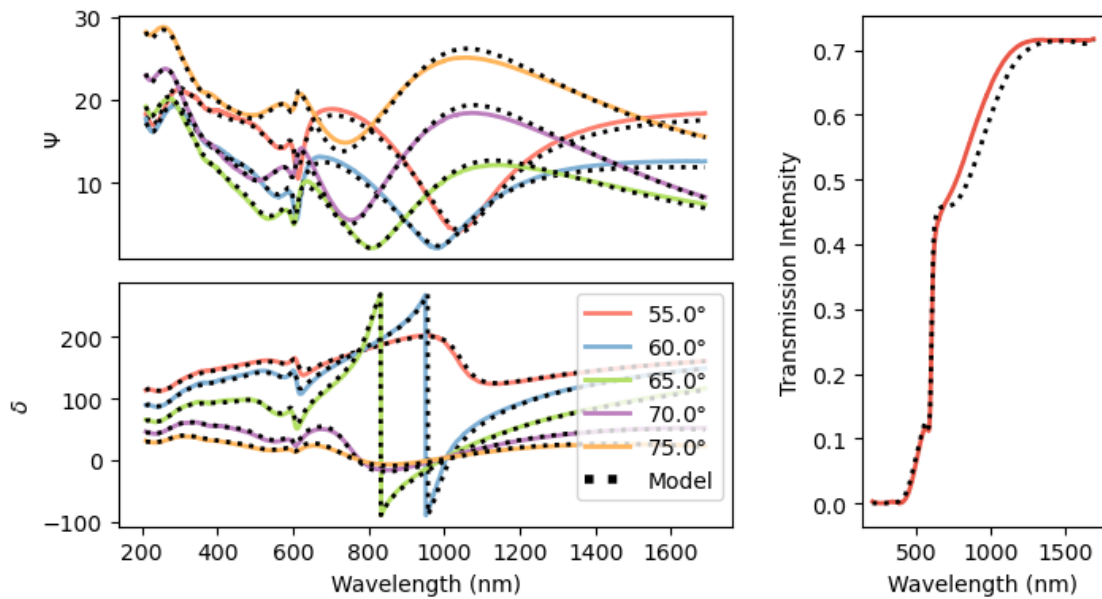
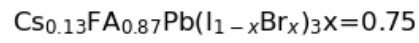
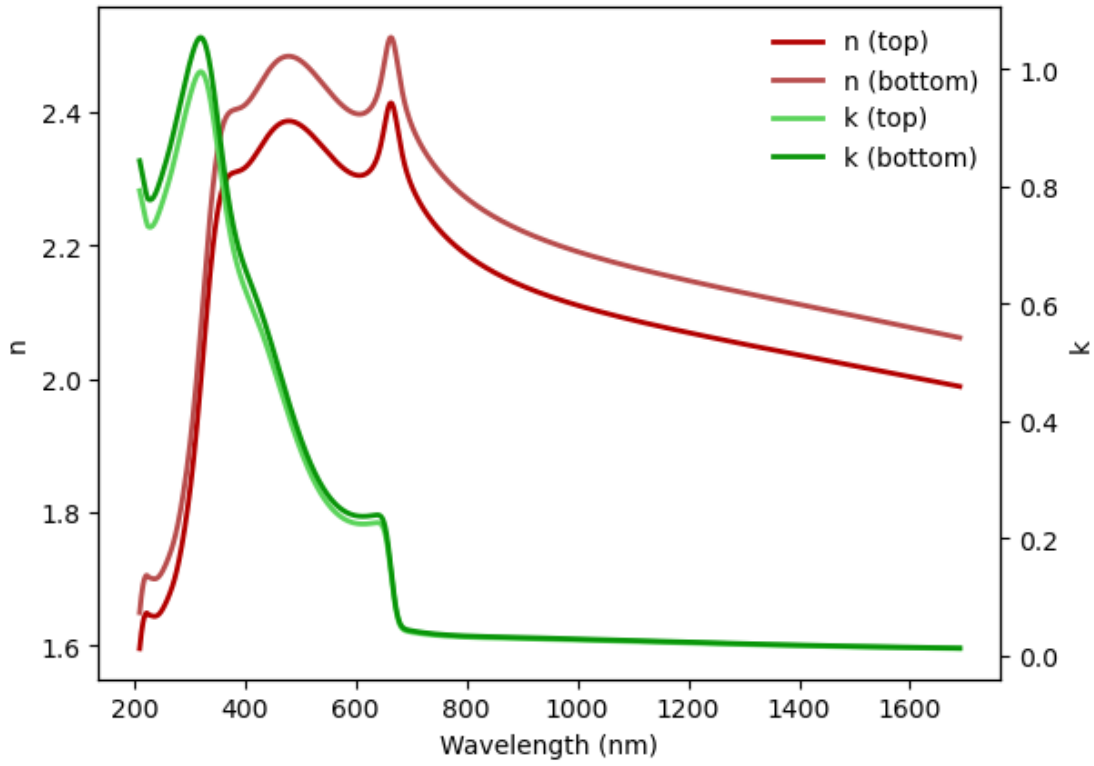
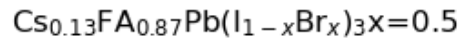
```
[3]: angles= [55.00, 60.00, 65.00, 70.00, 75.00]
br_contents = ['000', '025', '050', '075', '100']

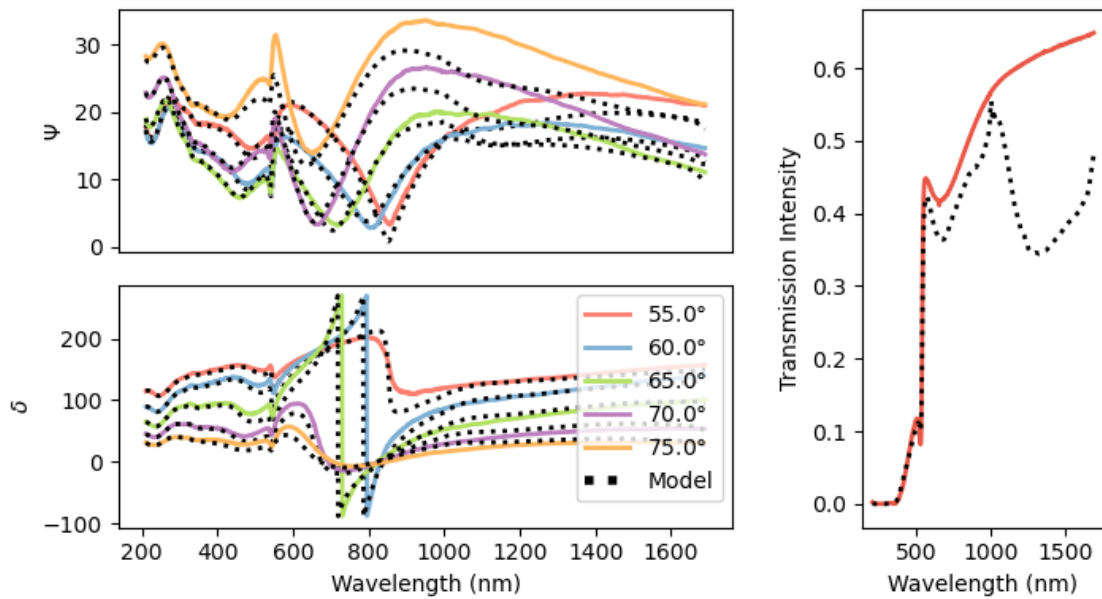
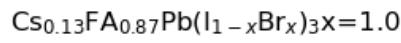
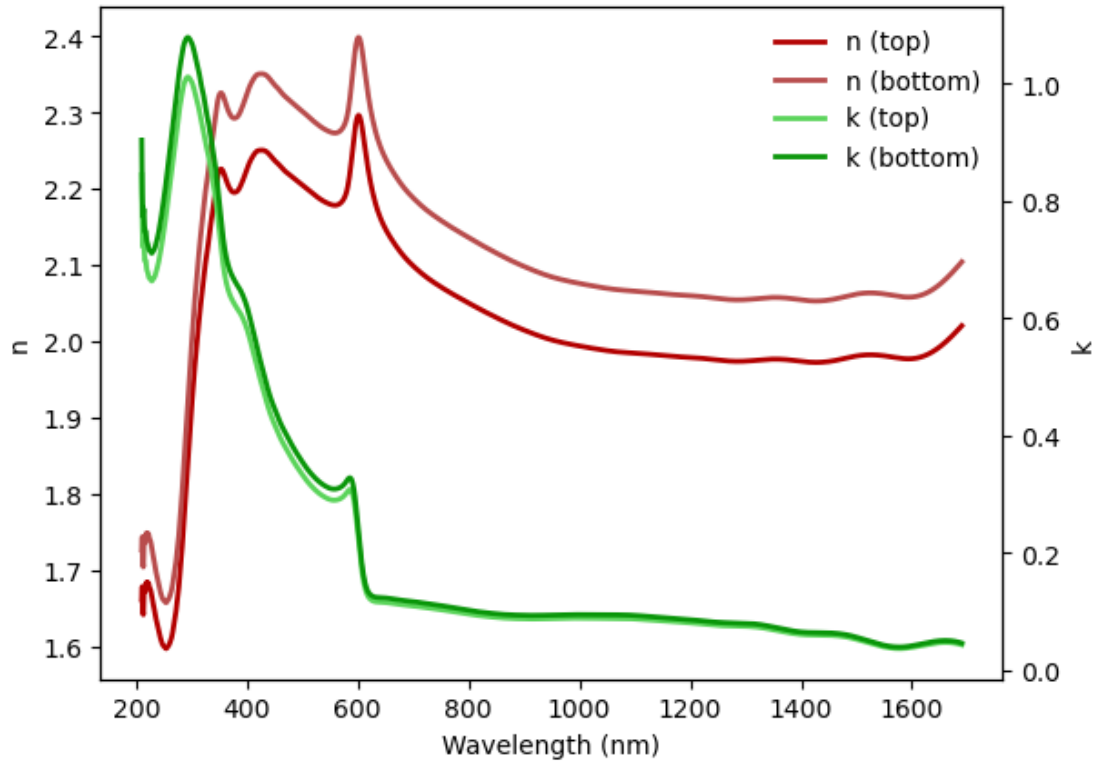
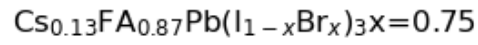
for i,br_content in enumerate(br_contents):
    pero_data_path = f'VASE/br{br_contents[i]}_data.txt'
    pero_data_trans_path = f"VASE/br{br_contents[i]}_data_trans.txt"
    pero_data_nk_path = f'VASE/br{br_contents[i]}_nk.txt'
    title = 'Cs$_{0.13}$FA$_{0.87}$Pb(I$_{1-x}$Br$_x$)$_3$'+ 'x={}'.format(int(br_contents[i])/100)
    plot_ellipsometry(pero_data_path, pero_data_trans_path, angles, title=title)
    plot_nk_graded(title, pero_data_nk_path)
```

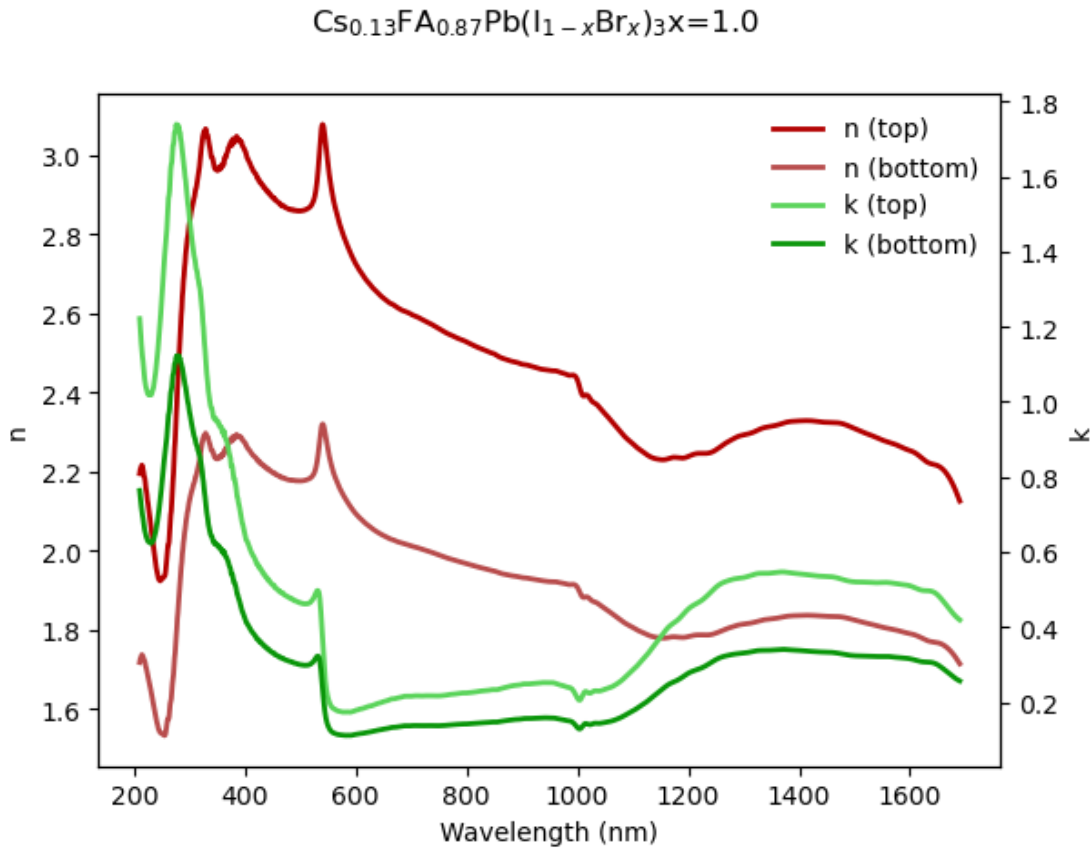












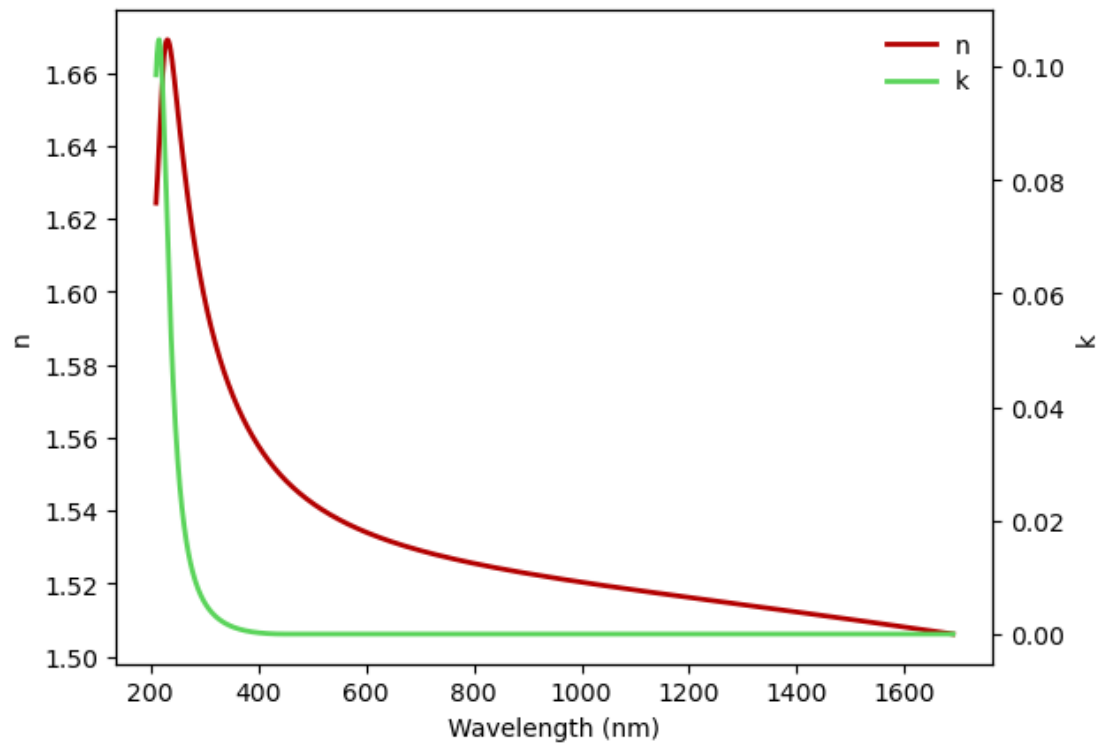
1.3 Substrate layers for PbSn

- Here we have the same as before so we didn't store the full VASE data, but we can replot the fitted n and k (It's roughly the same)
 - The SnO_2 here was deposited differently and is not as transparent: it has some high energy absorption. Still, $k < 0.5$ so it's still quite transparent
- There is an additional SAM layer between the SnO_2 and perovskite but it's barely measurable: The measurement is not sensitive to it, it's too thin. Still we measured and included it although it doesn't make much difference
- We didn't worry too much about these layers being very accurate as they just need to be good enough to act as a base for the perovskite fits

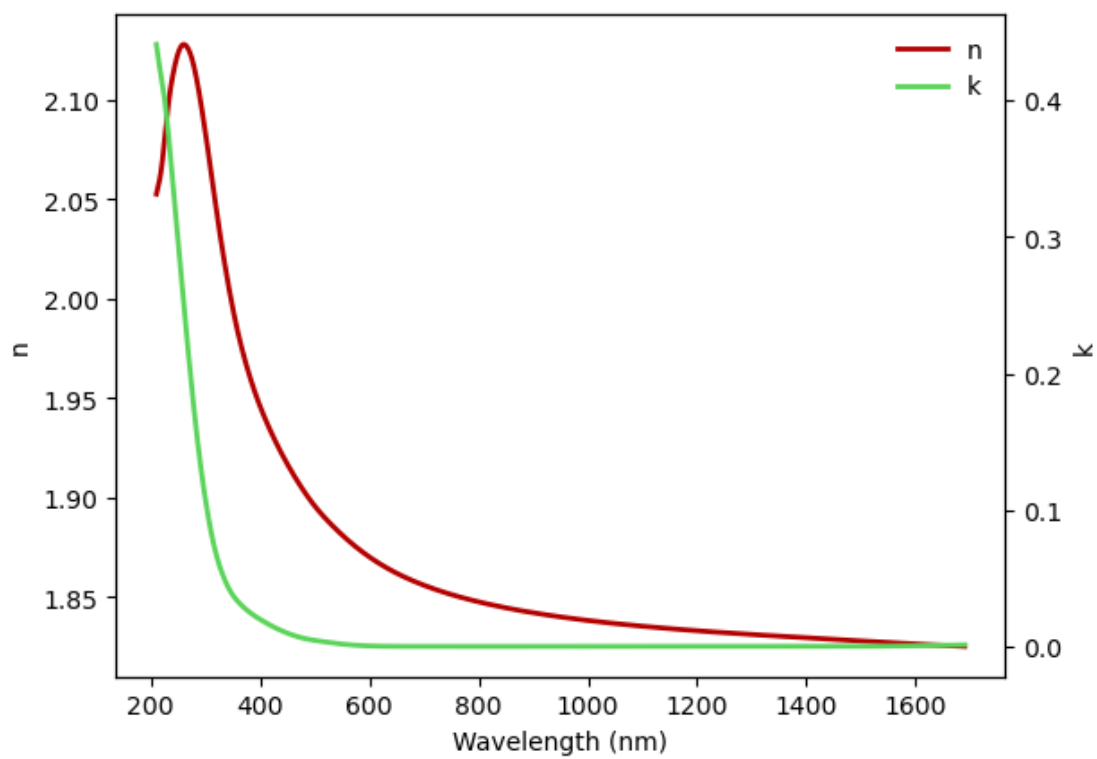
```
[4]: substrate_layer_names = ["glasspbsn", "SnO2_pbsn", "SAM"]
titles = ['Glass (for PbSn samples)', 'SnO2 (for PbSn samples)', 'SAM']

for title, substrate_name in zip(titles, substrate_layer_names):
    nk_path = f"VASE/{substrate_name}_nk.txt"
    plot_nk(title=title, datapath=nk_path)
```

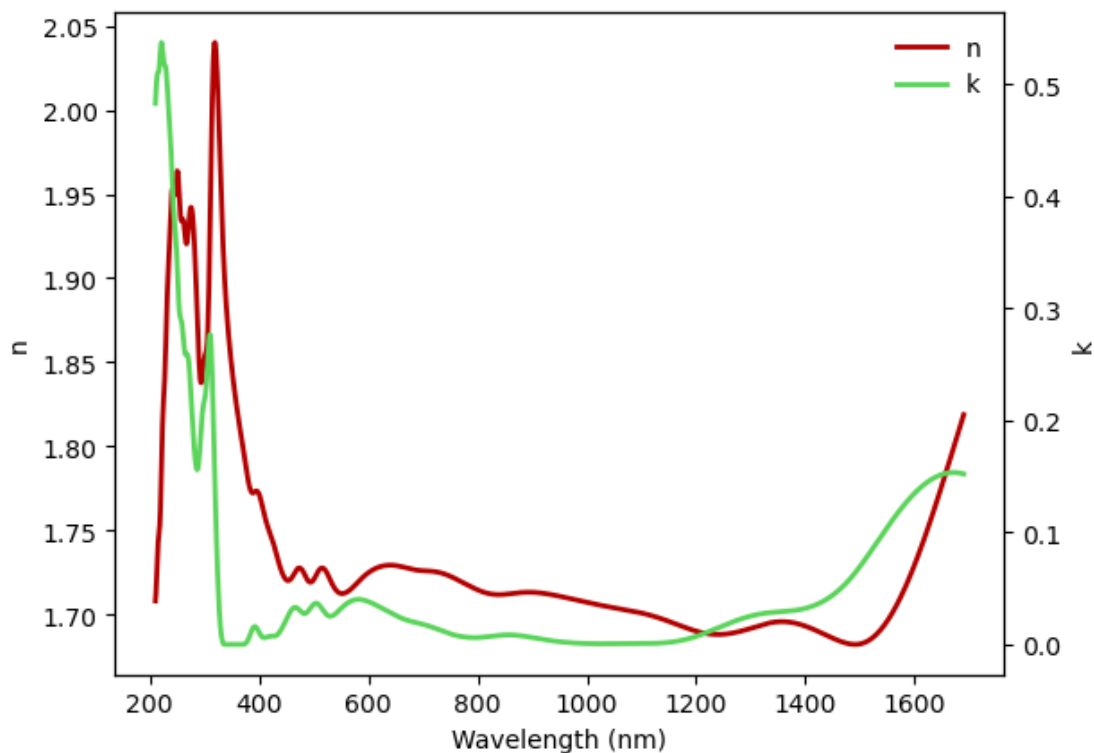
Glass (for PbSn samples)



SnO₂ (for PbSn samples)



SAM



1.4 PbSn Perovskite fits

```
[5]: angles= [50, 60, 70]
sn_contents = ['010', '020', '030', '040', '050']

for i,sn_content in enumerate(sn_contents):
    pero_data_path = f'VASE/sn{sn_contents[i]}_data.txt'
    pero_data_trans_path = f'VASE/sn{sn_contents[i]}_trans.txt'
    pero_data_nk_path = f'VASE/sn{sn_contents[i]}_nk.txt'
    title = "Cs$_{0.1}$FA$_{0.61}$MA$_{0.3}$Sn$_{x}$Pb$_{(1-x)}$I$_3$"+f"_{int(sn_content)/100}"
    plot_ellipsometry(pero_data_path, pero_data_trans_path, angles, title=title)
    plot_nk_graded(title, pero_data_nk_path)
```

