In [1]:

```
import numpy as np
import pandas as pd
from scipy.interpolate import interp1d
import re

import Spectra as sp
from fittings import fit_single_shape, fit_single_shape_colnames

def fitToModel(data, model, prange, pwrref = None, pwrvar = None, absref = None):
    ptrn = '-?\ *[0-9]+\.?[0-9]*(?:[Ee]\ *[+-]?\ *[0-9]+)?' #kümnendmurru pattern re jaoks
    model = (list(range(*prange)),model[prange[0]:prange[1]])
    result = pd.DataFrame()
    a = []
    wl= []
    for n in data:
        a+= [n.spc[prange[0]:prange[1]]]
        wl += [float(re.search(ptrn,n.comment).group(0))]
    result['wl'] = wl
    result['data'] = a
    fitted,params =  fit_single_shape(np.array(a), list(range(*prange)), model)
    result['fitted'] = list(fitted)
    #fittingu parameetrid
    for i,col in enumerate(fit_single_shape_colnames()):
        result[col] = params.T[i]
    #neeldumise koef, kui vaja
    if absref is None:
        result['coef'] = 1
    else:
        result['coef'] = pow(10,absref(result['wl']))
    #võimsusreferents
    if pwrref is None:
        #proovime ekstraheerida commendist
        result['power'] = [float(re.search('[uW]'+ptrn,n.comment).group(0)[1:]) for n in data]
        result['pwr-Err'] = [float(re.search('[var]'+ptrn,n.comment).group(0)[1:]) for n in data]
    else:
        result['power'] = pwrref(result['wl'])
        result['pwr-Err'] = pwrvar(result['wl'])*2.58
    result['Excit'] = result['a']/(result['power']*result['coef'])

    return result
```

In [2]:

```
data = sp.Spectra("750m.spc")
model = sp.Spectra("calib210326.spc")[3]
abssig = sp.Spectra("calib210326.spc")[0]
absref = sp.Spectra("calib210326.spc")[2]
prange = (680,1020)
p750m = fitToModel(data,model.spc, prange, None,None,interp1d(abssig.xVect(),abssig.spc ))

p750m
```

Out[2]:

	wl	data	fitted	a	a-Err	b	b-Err	Noise	S/N	coef	power	pwr-Err	
0	700.02	[462.0, 527.0, 547.0, 574.0, 612.0, 609.0, 670...	[473.91665005143494, 517.6544959306786, 549.80...	12201.860298	34.525626	-76.540704	15.830696	77.980068	154.315833	1.125019	257.50	0.034700	42.1
1	700.02	[99.0, 143.0, 139.0, 169.0, 198.0, 173.0, 193....	[140.19035108750177, 150.808156160144, 158.613...	2962.125170	12.557619	6.561253	5.757921	28.346932	103.054127	1.125019	82.50	0.005930	31.9
2	701.96	[141.0, 156.0, 126.0, 149.0, 170.0, 156.0, 160...	[126.43396626751863, 136.43213184267424, 143.7...	2789.259899	12.259312	0.603266	5.621126	27.692072	99.334836	1.121996	80.09	0.030200	31.0
3	704.03	[120.0, 121.0, 106.0, 144.0, 149.0, 182.0, 198...	[121.12735356633316, 130.09536262427306, 136.6...	2501.869753	11.901945	8.261563	5.457283	26.873080	91.815352	1.117663	74.13	0.000747	30.1

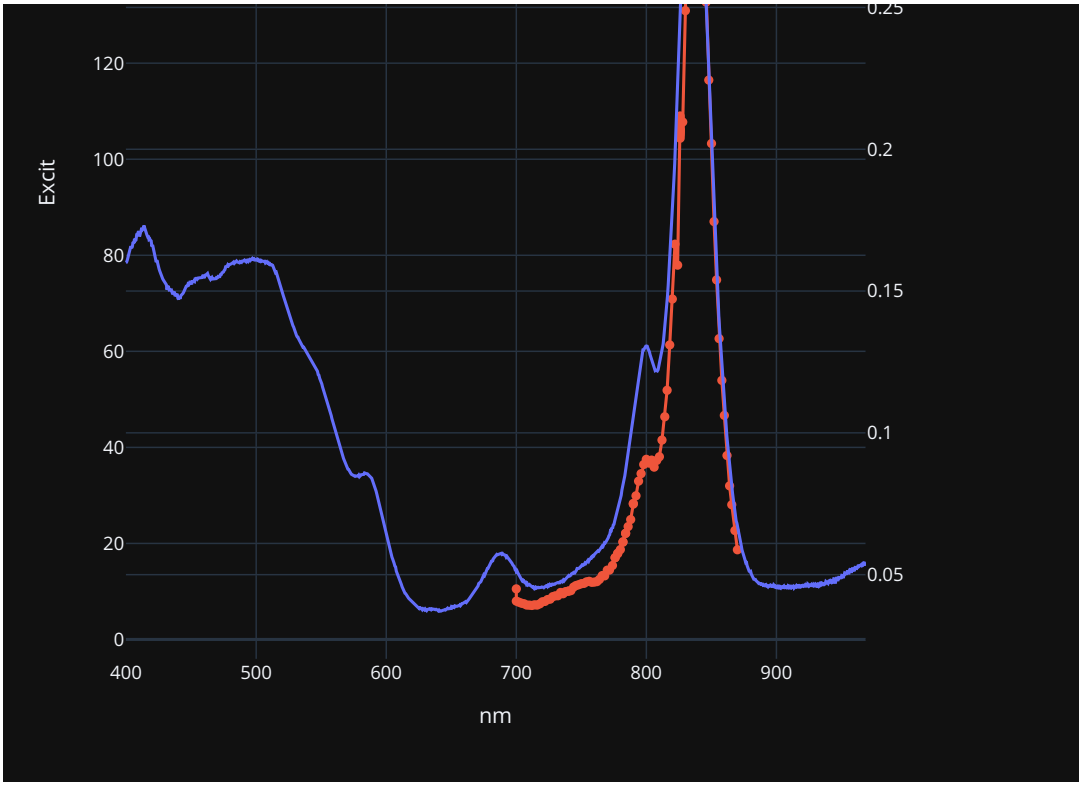
4	705.99	[134.0, 117.0, 126.0, 137.0, 132.0, 168.0, 180...	[121.90521133105666, 130.1597303750467, 136.22...	2302.822331	11.326405	18.018963	5.193384	25.581275	88.778191	1.115475	69.50	0.000741	29.7(
...	...	...	...	...	...	...	...	...	...	...	...	...	
102	862.02	[108.0, 142.0, 143.0, 161.0, 195.0, 202.0, 180...	[123.00764447390864, 130.64675926751823, 136.2...	2131.138597	8.878020	26.866492	4.070751	20.202036	104.036228	1.254207	11.09	0.000058	153.2
103	864.01	[154.0, 121.0, 159.0, 159.0, 161.0, 175.0, 196...	[130.19140234178118, 136.46368719275307, 141.0...	1749.824253	9.753498	51.252322	4.472175	22.020724	78.366579	1.225503	11.16	0.000018	127.9
104	865.69	[143.0, 120.0, 155.0, 144.0, 178.0, 147.0, 197...	[110.11652576467145, 115.61142005271162, 119.6...	1532.950037	10.090181	40.961200	4.626551	22.774853	66.380491	1.203532	11.35	0.000129	112.2
105	868.11	[139.0, 138.0, 182.0, 173.0, 183.0, 187.0, 183...	[112.7657109472801, 117.3164090070556, 120.661...	1269.540849	10.342608	55.493452	4.742294	23.360663	53.595653	1.180909	11.85	0.000027	90.7
106	870.02	[182.0, 209.0, 194.0, 232.0, 209.0, 229.0, 244...	[153.47799733433146, 157.33522556038344, 160.1...	1076.078600	19.247524	104.933319	8.825377	43.448714	24.425031	1.166545	12.35	0.000133	74.6

107 rows × 13 columns

In [3]:

```
import plotly.graph_objects as go
from plotly.subplots import make_subplots
fig1 = make_subplots(specs=[[{"secondary_y": True}]])

fig1.add_trace(go.Scatter(x =abssig.xVect(),y=abssig.spc, mode = 'lines', name = "abs"),  secondary_y=True)
fig1.add_trace(go.Scatter(x =p750m['wl'], y=p750m['Excit']/4, mode = 'lines+markers', name = "FES750"))
#fig1.add_trace(go.Scatter(x =absref.xVect(),y=absref.spc, mode = 'lines', name = "abs_ref"))
fig1.update_layout(height = 800, title = "Ergastusspekter (R.Sp. ???)",template = "plotly_dark")
#fig1.layout.yaxis.range = [-10,8000]
fig1.layout.xaxis.title = "nm"
fig1.layout.yaxis.title = "Excit"
fig1
```



In [4]:

```
fig2 = go.FigureWidget(data=go.Scatter(x=list(range(*prange)), y=p750m['data'][0] , mode = 'lines', name = 'sig')
fig2.add_trace(go.Scatter(x=list(range(*prange)), y=p750m['fitted'][0] , mode = 'lines', name = 'ref'))
#fig2.add_trace(go.Scatter(x =sig[0].xVect(), y=np.array(sig[0].spc) - np.array(ref[0].spc), mode = 'lines', name
#fig2.layout.yaxis.range = [-10,4500]
fig2.update_layout(height = 600,template = "plotly_dark")
fig2
```

In [5]:

```
from ipywidgets import IntSlider


slider = IntSlider(
    orientation='horizontal',
    value=0,
    min=0,max=len(p750m['data']) - 1
)

def update(spno):

    #fig2.data[0].x = sig[slider.value].xVect()
    fig2.data[0].y = p750m['data'][slider.value]
    #fig2.data[1].x = ref[slider.value].xVect()
    fig2.data[1].y = p750m['fitted'][slider.value]
    #fig2.data[2].y = np.array(sig[slider.value].spc) - np.array(ref[slider.value].spc)
    fig2.layout.title = p750m['wl'][slider.value]

slider.observe(update, names='value')
slider
```

In [ ]: