Untitled7

April 16, 2022

[1]:
```
import matplotlib.pyplot as plt
import pandas as pd
import torch
import torch.nn as nn
from torch.optim import Adam
from torch.utils.data import DataLoader

from binarypredictor import split_functions
from binarypredictor.dataset import FunctionPairDataset
from binarypredictor.net import DerivativeNet, TangentNet
```

[2]:
```
out_features = 500
in_features = out_features
```

[3]:
```
net_0 = DerivativeNet(train=False, net='FirstDerivativeNet_250_s.pth')
net_1 = DerivativeNet(train=False, net='SecondDerivativeNet_250_s.pth')
```

[131]:
```
fpd = FunctionPairDataset(n_functions=100000, filename="test.csv",␣
↪overwrite=True, step=1/in_features)
fpd.create_functions()
```

[132]:
```
loader = DataLoader(fpd, batch_size=4096)
net = TangentNet(train=True, in_features=in_features * 2,␣
↪out_features=out_features, hidden_size_linear=500, hidden_layers=2)
```

[133]:
```
x = torch.arange(1e-10, 1., step=fpd.step)
```

[134]:
```
lr = 1e-3

# Workers
loss_func = nn.MSELoss()
optimizer = Adam(net.parameters(), lr=lr)

best_loss = 1000
best_net = net

for i in range(2500):
    for d in loader:
```

```
        inp = torch.hstack((d[0][:, :, 0], d[0][:, :, 1]))
        out = net(inp)
        out = torch.clamp(out, min=1e-10, max=1.-1e-4)

        f_der = fpd.first_derivative(**d[1][0], x=out)/d[2].unsqueeze(-1)
        g_der = fpd.first_derivative(**d[1][1])/d[2].unsqueeze(-1)

        g_der = torch.clamp(g_der, min=-100, max=100)

        optimizer.zero_grad()
        loss = loss_func(f_der, g_der)

        if loss.isinf():
            print('max: ', torch.max(out))
            print('inf')
            break

        if loss.isnan():
            print('nan')
            break
        elif loss < 0.2:
            break

        loss.backward()
        optimizer.step()

    if not loss.isnan() and loss < best_loss:
            best_net = net
            best_loss = loss
    elif loss.isnan() or loss.isinf():
        break

    if i % 10 == 0:
        print(loss)
```

```
tensor(2.0971, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.7734, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.6611, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.6494, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.5981, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.5869, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.5534, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.5274, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.5110, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.5116, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.5240, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4959, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4861, dtype=torch.float64, grad_fn=<MseLossBackward0>)
```

```
tensor(0.5228, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.5112, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4698, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4524, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4770, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4426, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4409, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4607, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4366, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4527, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4469, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4515, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4334, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4179, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4423, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4202, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4281, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4200, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4247, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4351, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4273, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4173, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4152, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4120, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3990, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4024, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3878, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4049, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3938, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3923, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3999, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4011, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3882, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3817, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4008, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3838, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3925, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3868, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3729, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3890, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3766, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3743, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3835, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3932, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3860, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3687, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3910, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3868, dtype=torch.float64, grad_fn=<MseLossBackward0>)
```

```
tensor(0.3734, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3771, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4083, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3805, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3791, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3628, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3850, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3715, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3587, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3686, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3853, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3594, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.4156, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3631, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3733, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3638, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3901, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3743, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3745, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3833, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3696, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3625, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3563, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3678, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3558, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3636, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3553, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3626, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3574, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3577, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3646, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3676, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3553, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3638, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3558, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3426, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3402, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3568, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3568, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3451, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3479, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3475, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3451, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3729, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3381, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3471, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3517, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3382, dtype=torch.float64, grad_fn=<MseLossBackward0>)
```

```
tensor(0.3344, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3518, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3429, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3429, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3469, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3448, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3409, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3382, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3425, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3552, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3289, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3379, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3581, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3336, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3219, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3297, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3263, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3260, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3302, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3436, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3281, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3258, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3369, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3322, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3787, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3421, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3352, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3379, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3179, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3217, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3101, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3146, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3212, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3356, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3312, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3131, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3262, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3423, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3248, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3316, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3136, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3181, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3266, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3140, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3158, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3183, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3367, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3348, dtype=torch.float64, grad_fn=<MseLossBackward0>)
```

```
tensor(0.3287, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3202, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3161, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3263, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3112, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3076, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3140, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3086, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3086, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3078, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3223, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3064, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3112, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3080, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3024, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3437, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3257, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3248, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3150, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3170, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3164, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3088, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3332, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3110, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3126, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3143, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.2974, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3170, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3030, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3049, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3020, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3228, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3021, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3038, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3001, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3063, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.2981, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.2996, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3277, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3167, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3029, dtype=torch.float64, grad_fn=<MseLossBackward0>)
tensor(0.3110, dtype=torch.float64, grad_fn=<MseLossBackward0>)
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-134-2184c708c4aa> in <module>
     33             break
```

```
     34
---> 35         loss.backward()
     36         optimizer.step()
     37

~\anaconda3\envs\5_Programmcodes\lib\site-packages\torch\_tensor.py in␣
↪backward(self, gradient, retain_graph, create_graph, inputs)
    305                 create_graph=create_graph,
    306                 inputs=inputs)
--> 307         torch.autograd.backward(self, gradient, retain_graph,␣
↪create_graph, inputs=inputs)
    308
    309     def register_hook(self, hook):

~\anaconda3\envs\5_Programmcodes\lib\site-packages\torch\autograd\__init__.py i␣
↪backward(tensors, grad_tensors, retain_graph, create_graph, grad_variables,␣
↪inputs)
    152         retain_graph = create_graph
    153
--> 154     Variable._execution_engine.run_backward(
    155         tensors, grad_tensors_, retain_graph, create_graph, inputs,
    156         allow_unreachable=True, accumulate_grad=True)  #␣
↪allow_unreachable flag

KeyboardInterrupt:
```

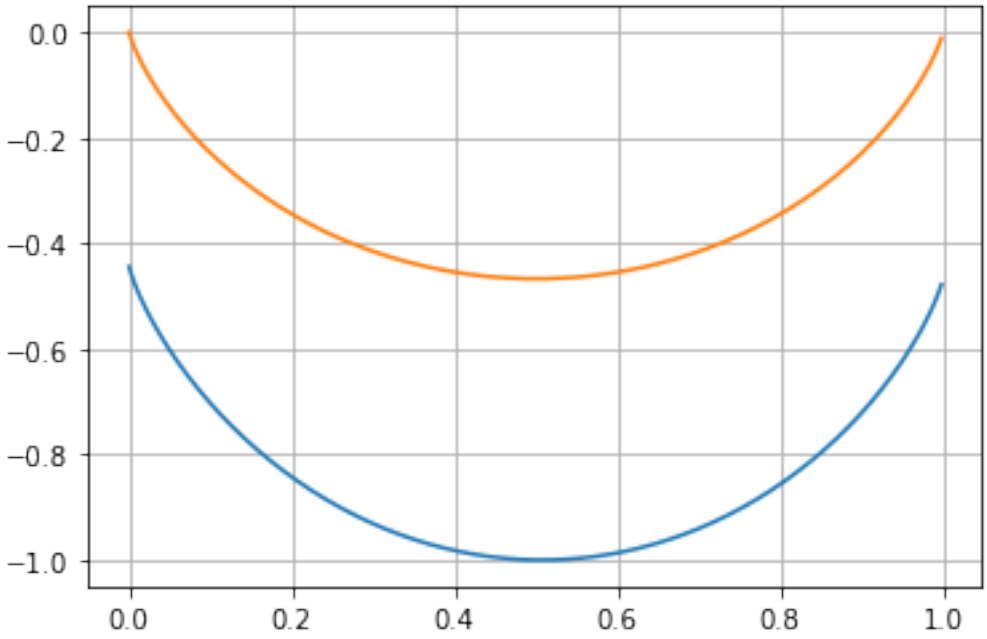
[135]:
```
torch.save(best_net, 'TangentNet_500_.pth')
```

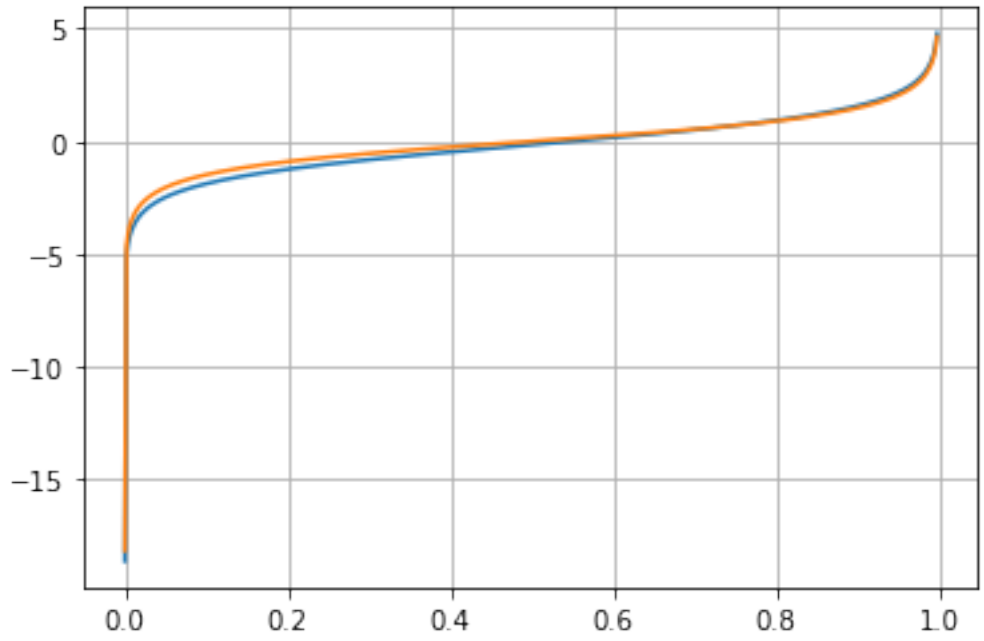
[136]:
```
net = best_net
```

[151]:
```
d = fpd[2]
scale = d[2].unsqueeze(-1)
inp = torch.hstack((d[0][:, 0], d[0][:, 1]))
```

[152]:
```
f = fpd.base_function(**d[1][0])/scale
g = fpd.base_function(**d[1][1])/scale
plt.plot(x, f)
plt.plot(x, g)
plt.grid()
plt.show()

f_d = fpd.first_derivative(**d[1][0])/scale
g_d = fpd.first_derivative(**d[1][1])/scale
plt.plot(x, f_d)
plt.plot(x, g_d)
plt.grid()
```

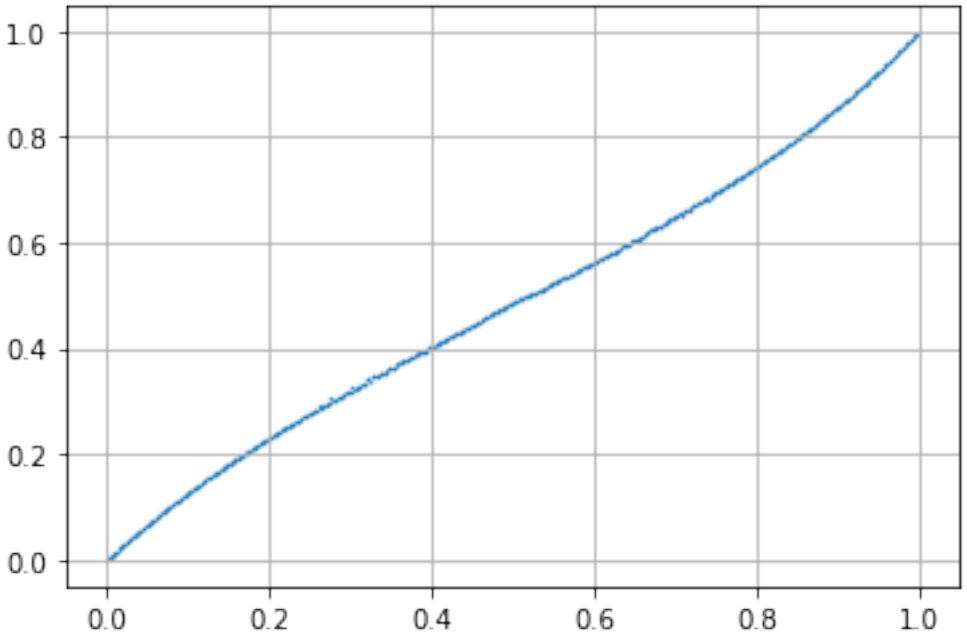
```
plt.show()
```





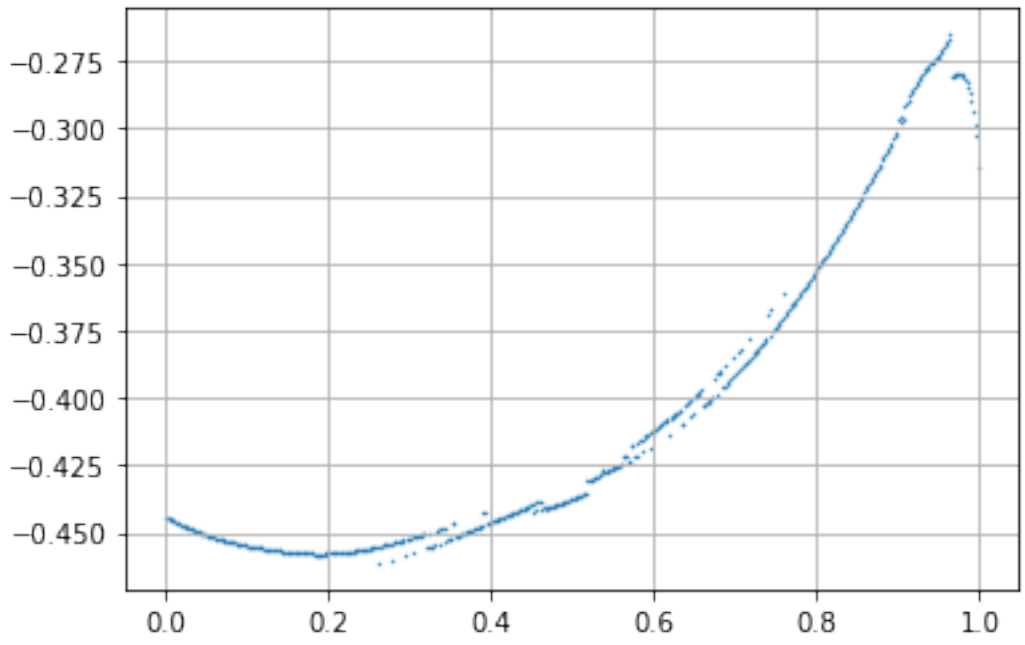
[153]:
```
out = net(inp)
```

```
plt.scatter(x, out.detach(), s=0.2)
plt.grid()
plt.show()
```



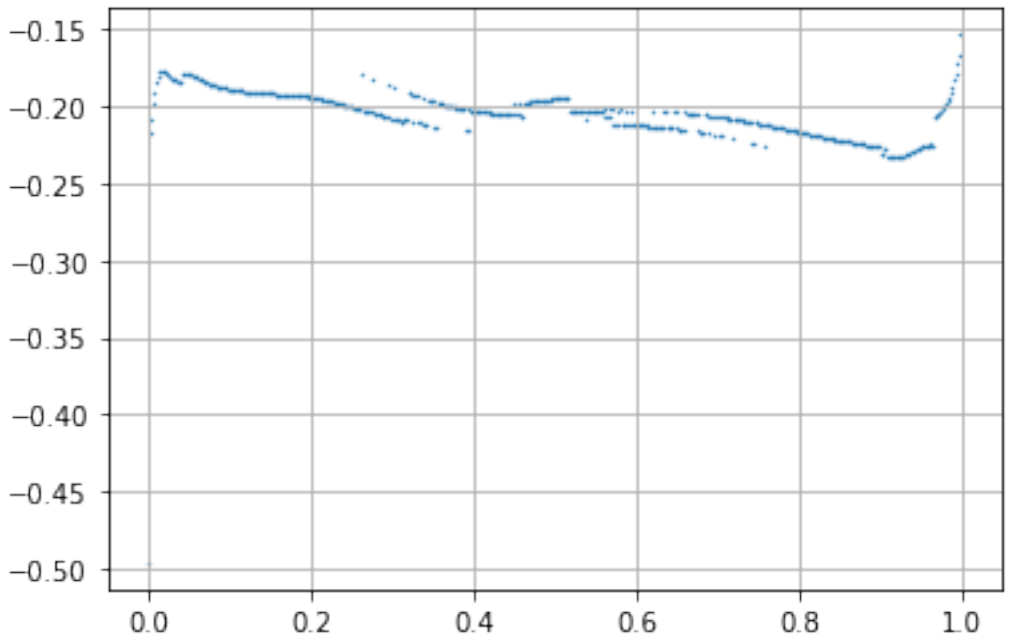
[154]:
```
eq = lambda x_, k: (fpd.base_function(**k, x=x_) - x_ * fpd.
 ↪first_derivative(**k, x=x_))/scale

eqn = eq(out, d[1][0]) - eq(x, d[1][1])

i_w_eqn = torch.where(abs(eqn) < 0.1)[0]
print(len(i_w_eqn))
plt.scatter(x, eqn.detach(), s=0.2)
plt.grid()
plt.show()
```

```
0
```



[155]:
```
dd = lambda x1, x2: (fpd.first_derivative(**d[1][0],x=x1) - fpd.
 ↪first_derivative(**d[1][1],x=x2))/scale

diff_der = dd(out, x) #/ max(abs(g_d))
i_w_der = torch.where(abs(diff_der) < 0.1)[0]

print(len(i_w_der))

plt.scatter(x, diff_der.detach(), s=.2)
plt.grid()
plt.show()
```

0

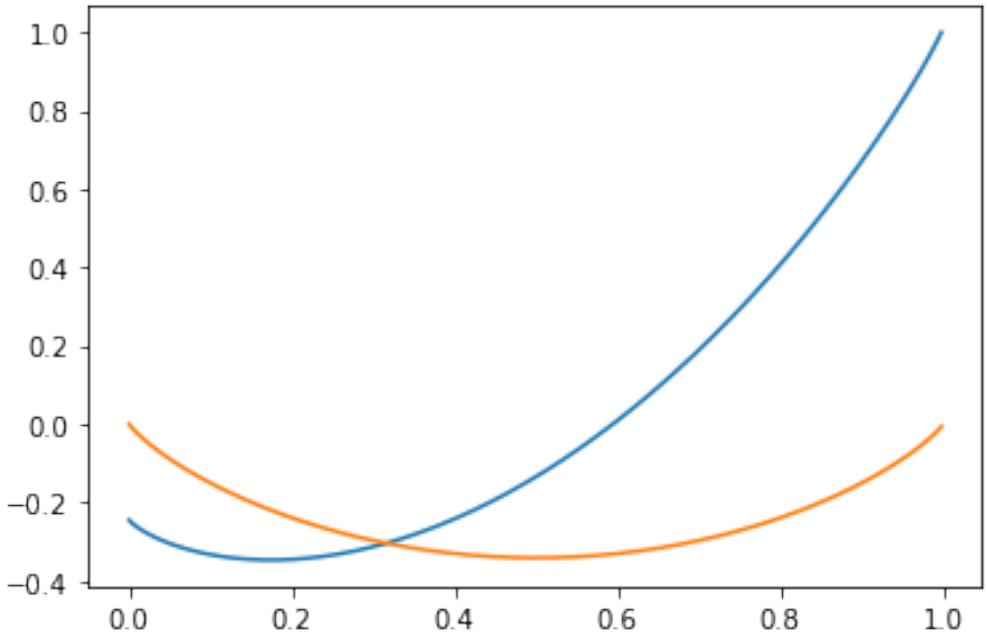


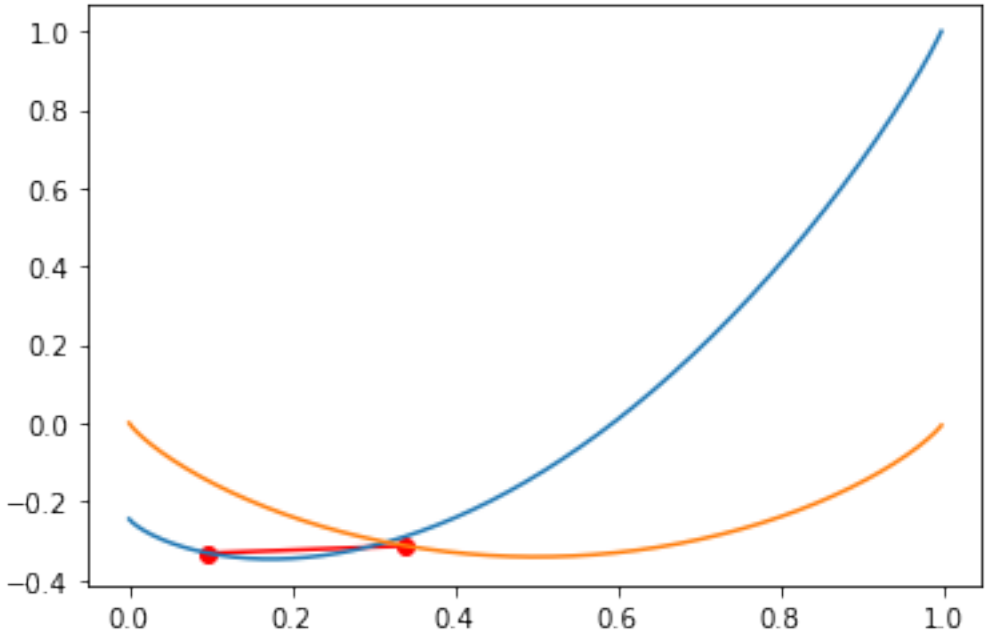
[129]:
```
i_w = torch.tensor([idx_ for idx_ in i_w_eqn if idx_ in i_w_der], dtype=torch.
 ↪int64)
print(i_w)
```

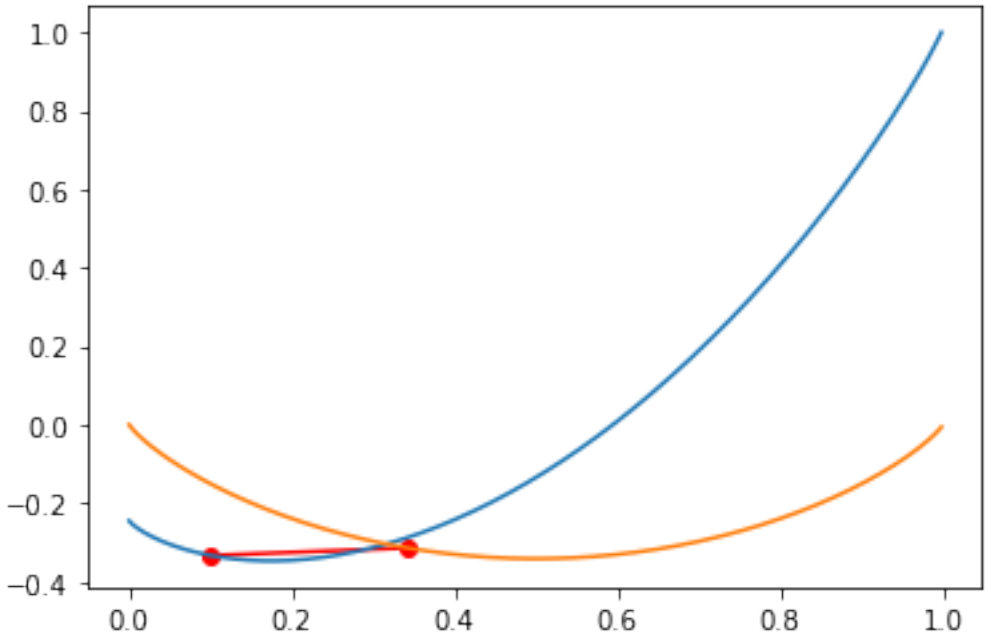
```
tensor([170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183,
        184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197,
        198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211,
        212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 225, 226,
        227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240,
        241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254,
        255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268,
        269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282,
        283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296,
        297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310,
        311, 312, 314, 316])
```

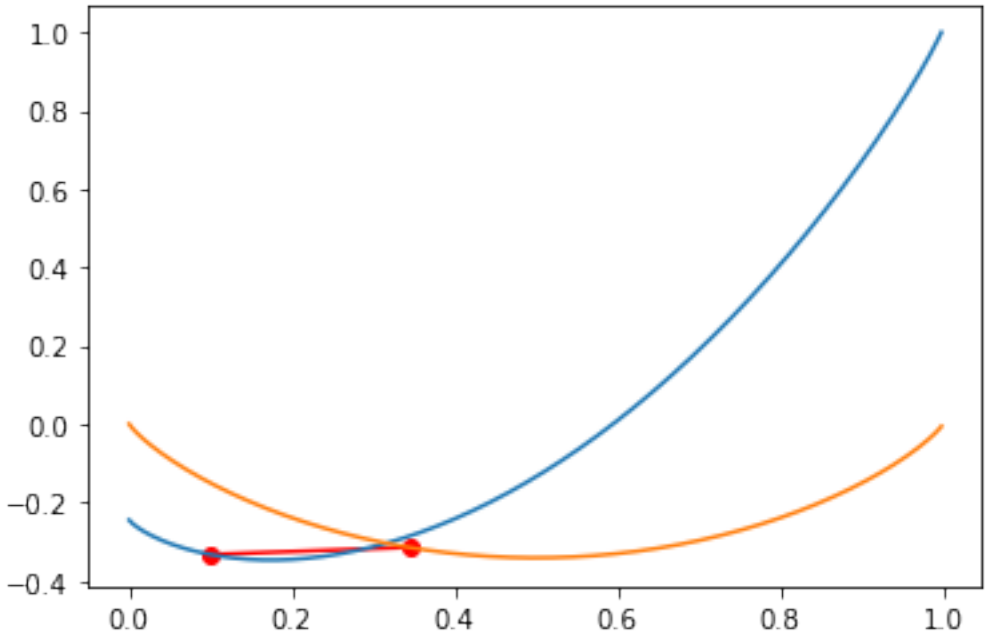
[130]:
```
plt.plot(x, f)
plt.plot(x, g)
plt.show()
x_f = out[i_w]
x_g = x[i_w]
y_f = fpd.base_function(**d[1][0], x=x_f)/scale
y_g = fpd.base_function(**d[1][1], x=x_g)/scale

for o, xx, yy, yy_ in zip(x_f.detach(), x_g, y_f.detach(), y_g):
    plt.plot([o, xx], [yy, yy_], 'ro-')
```

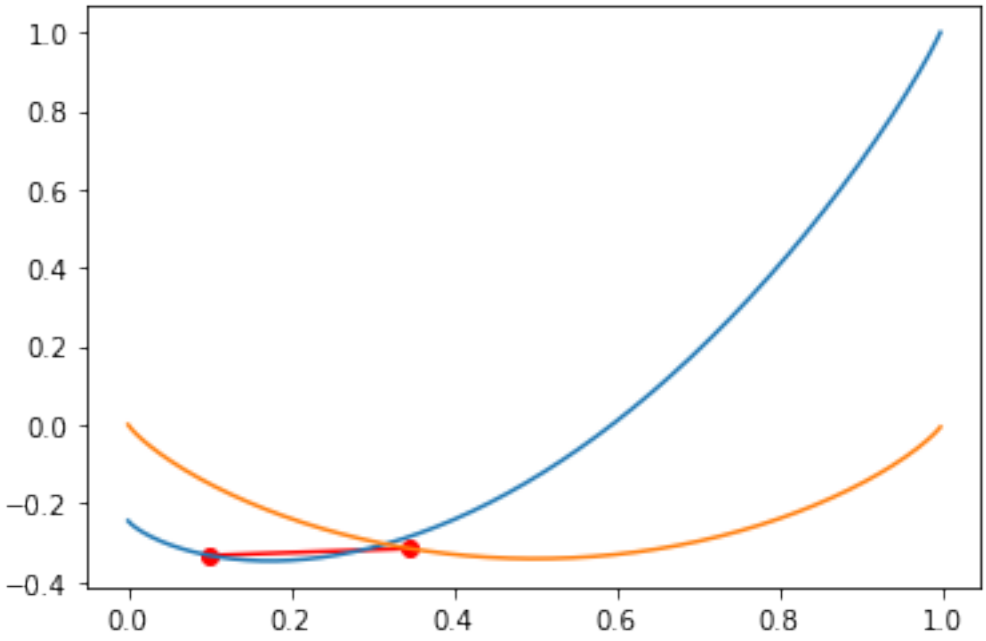
```
    plt.plot(x, f)
    plt.plot(x, g)
    plt.show()
#plt.show()
```

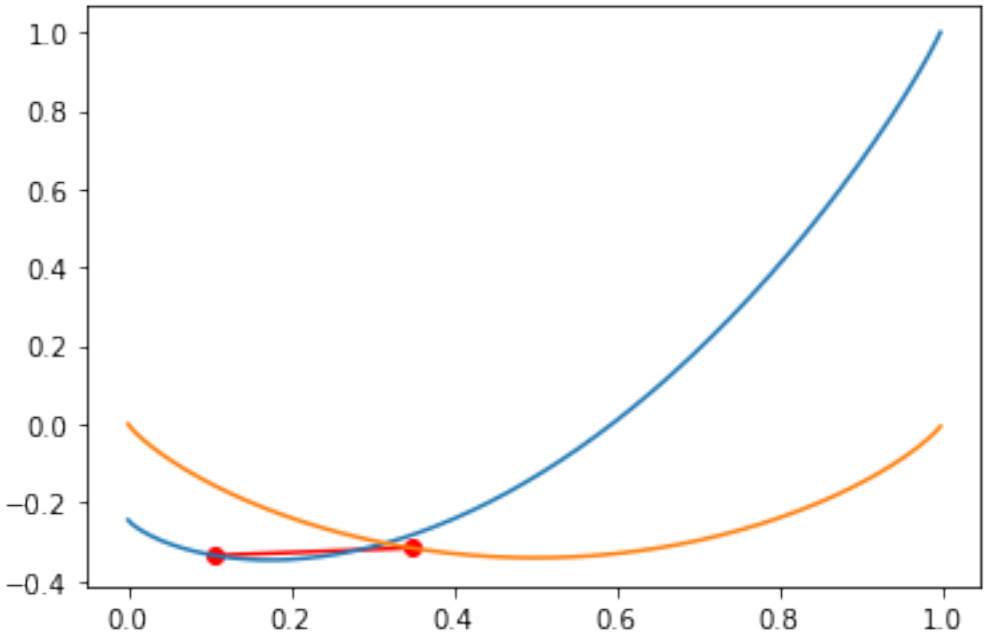


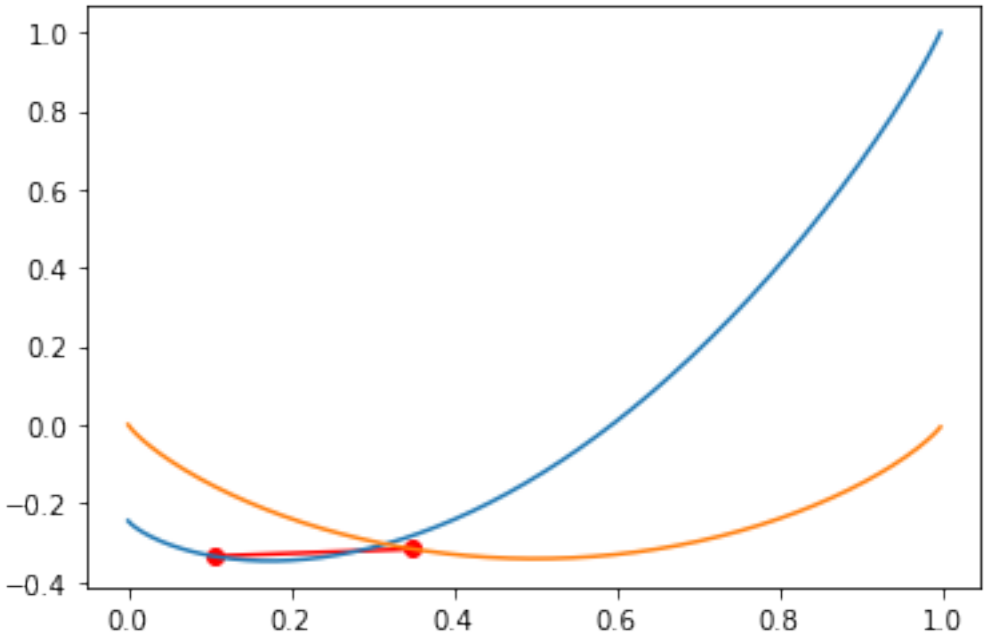


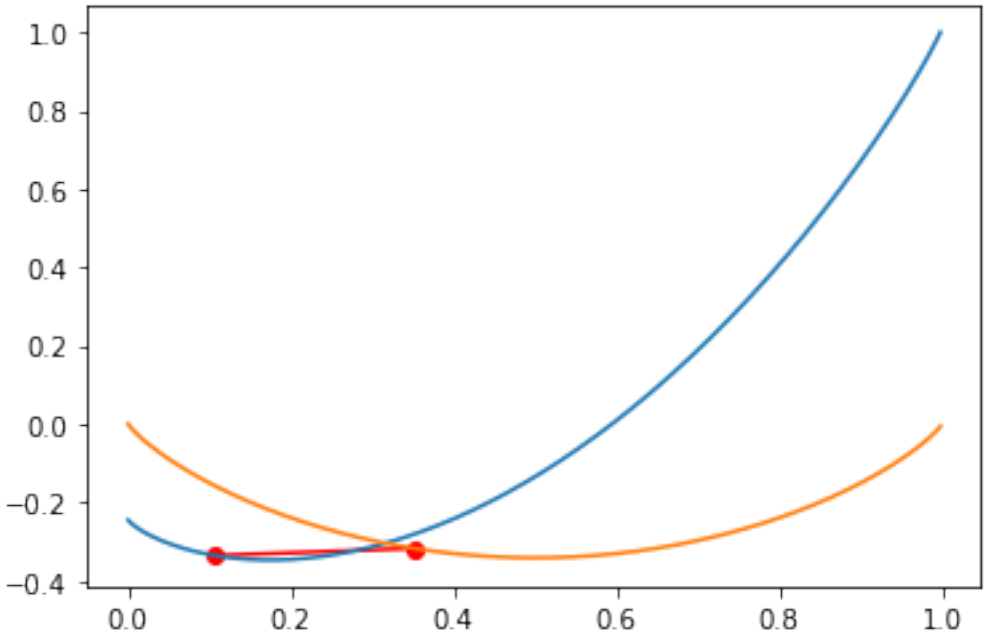

1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0

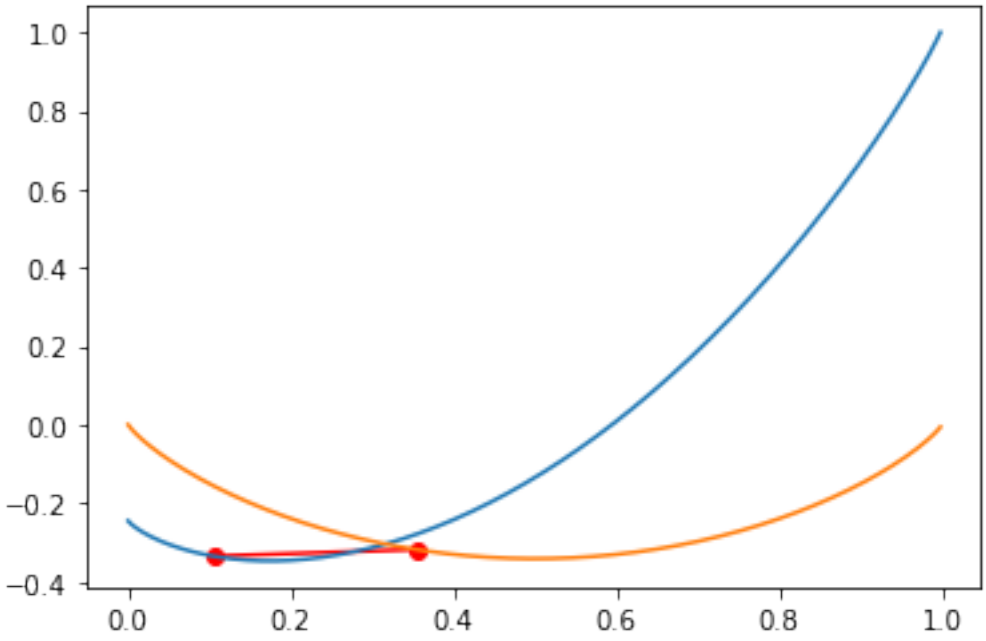

1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0

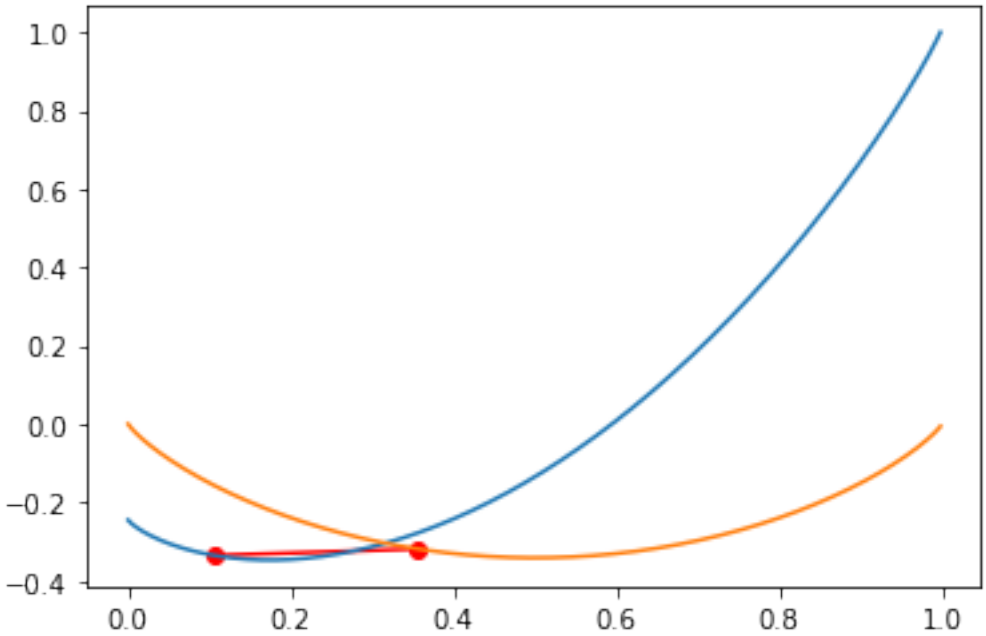

1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0

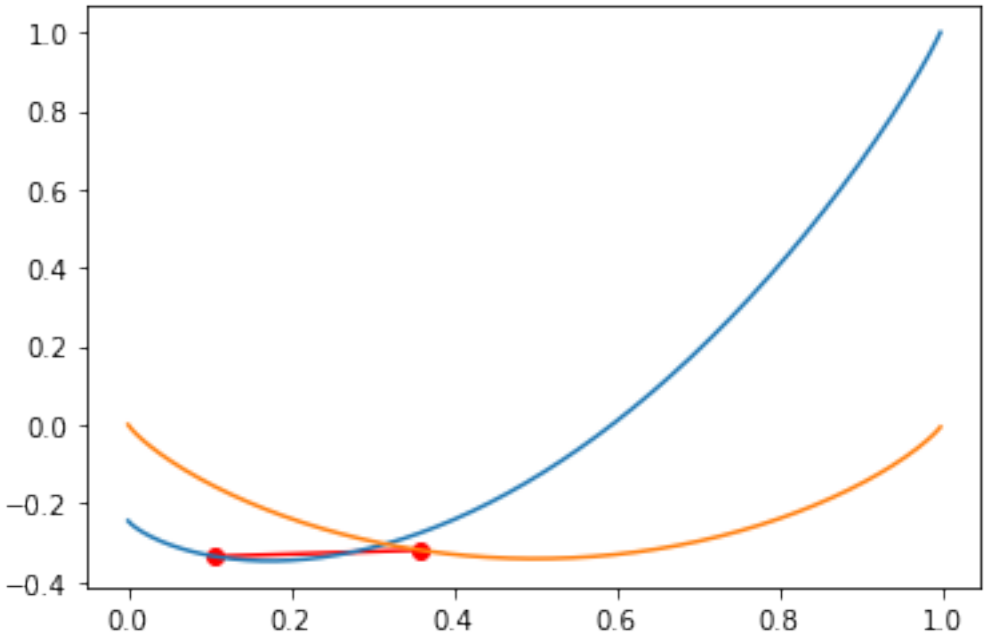

1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0

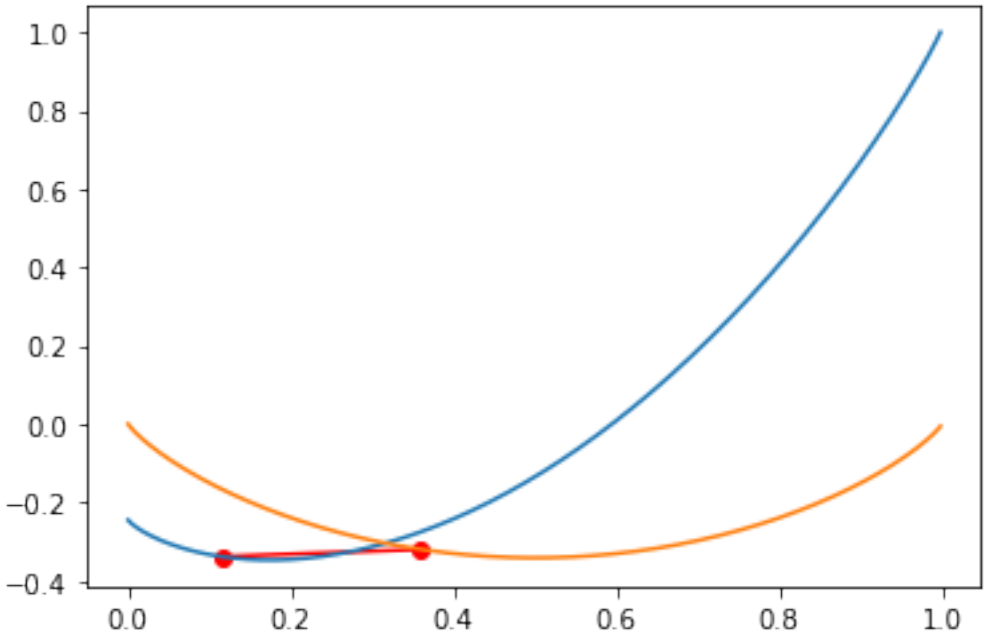

1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0

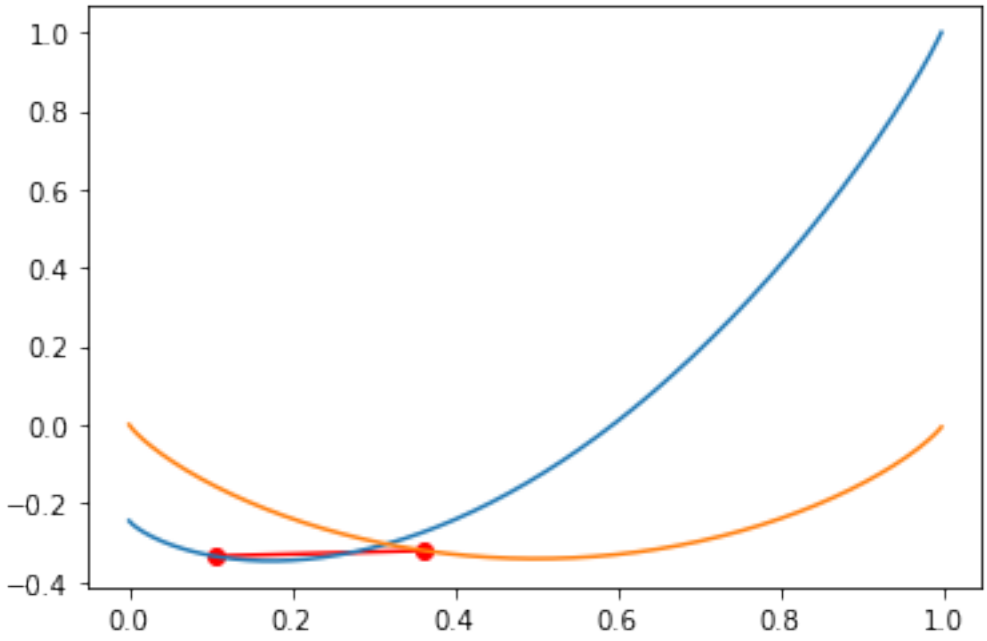

1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0

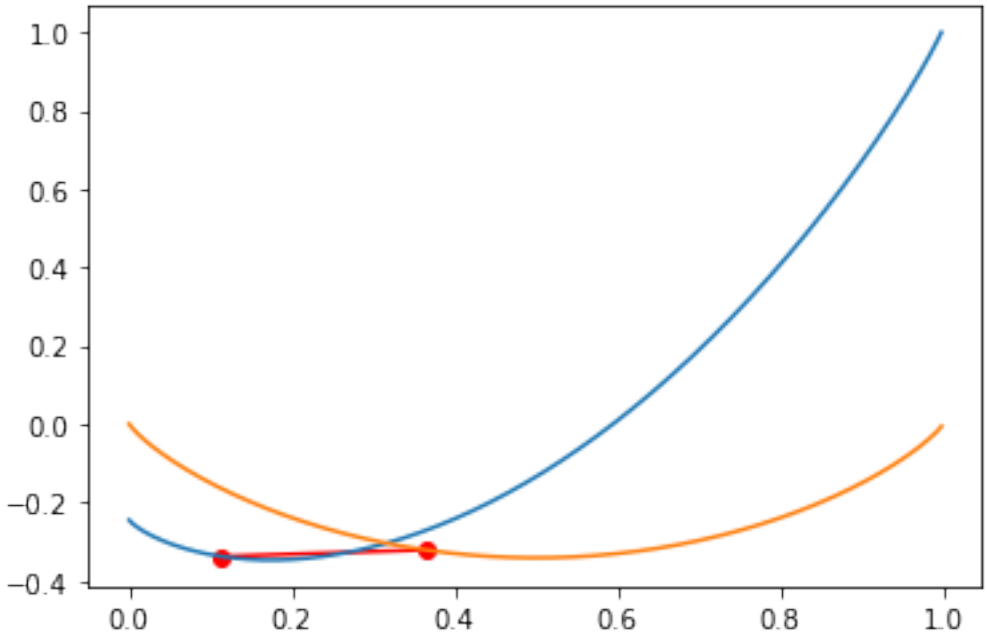

1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0

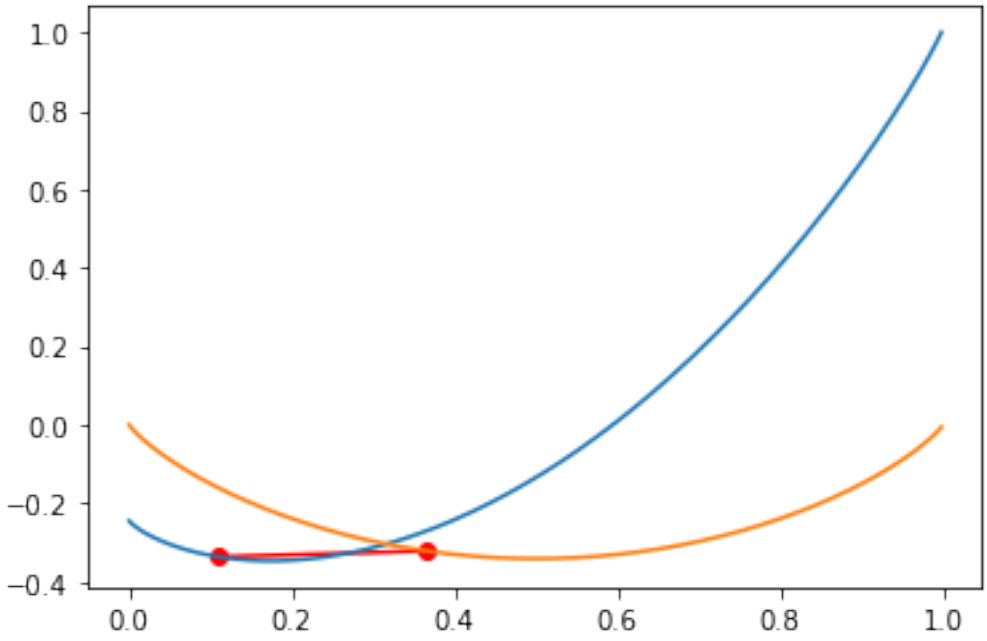

1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0

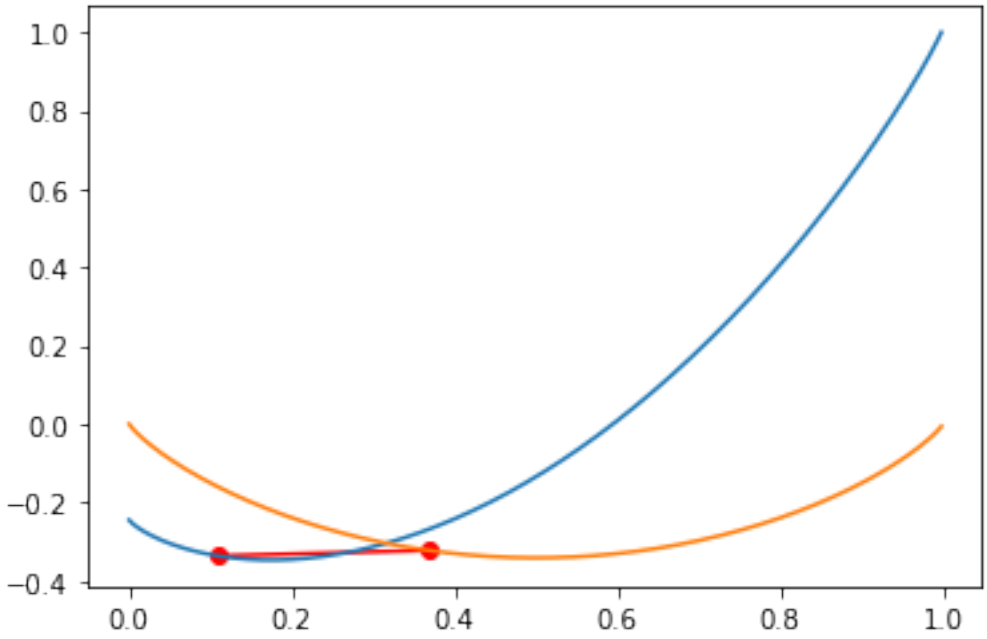

1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0

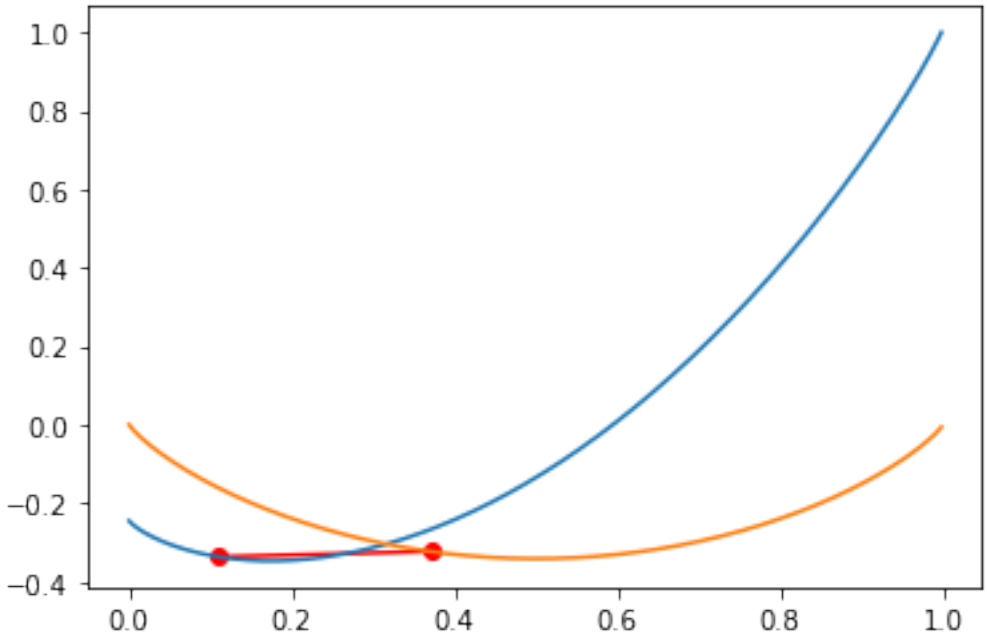

1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0


1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0


1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0


1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0


1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.2
0.4
0.6
0.8
1.0




```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-130-b7ae90c80d43> in <module>
     11     plt.plot(x, f)
     12     plt.plot(x, g)
---> 13     plt.show()
     14 #plt.show()

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\pyplot.py in␣
 ↪show(*args, **kwargs)
    366     """
    367     _warn_if_gui_out_of_main_thread()
--> 368     return _backend_mod.show(*args, **kwargs)
    369
    370

~\AppData\Roaming\Python\Python38\site-packages\ipykernel\pylab\backend_inline.
 ↪py in show(close, block)
     40             display(
     41                 figure_manager.canvas.figure,
---> 42                 metadata=_fetch_figure_metadata(figure_manager.canvas.
 ↪figure)
     43             )
     44     finally:
```

```
~\AppData\Roaming\Python\Python38\site-packages\ipykernel\pylab\backend_inline.
↪py in _fetch_figure_metadata(fig)
    226     if _is_transparent(fig.get_facecolor()):
    227         # the background is transparent
--> 228         ticksLight = _is_light([label.get_color()

    229                                 for axes in fig.axes
    230                                 for axis in (axes.xaxis, axes.yaxis)

~\AppData\Roaming\Python\Python38\site-packages\ipykernel\pylab\backend_inline.
↪py in <listcomp>(.0)
    229                                 for axes in fig.axes
    230                                 for axis in (axes.xaxis, axes.yaxis)
--> 231                                 for label in axis.get_ticklabels()])

    232         if ticksLight.size and (ticksLight == ticksLight[0]).all():
    233             # there are one or more tick labels, all with the same␣
↪lightness

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\axis.py in␣
↪get_ticklabels(self, minor, which)
   1247         if minor:
   1248             return self.get_minorticklabels()
-> 1249         return self.get_majorticklabels()
   1250
   1251     def get_majorticklines(self):

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\axis.py in␣
↪get_majorticklabels(self)
   1199     def get_majorticklabels(self):
   1200         """Return this Axis' major tick labels, as a list of `~.text.
↪Text`."""
-> 1201         ticks = self.get_major_ticks()
   1202         labels1 = [tick.label1 for tick in ticks if tick.label1.
↪get_visible()]
   1203         labels2 = [tick.label2 for tick in ticks if tick.label2.
↪get_visible()]

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\axis.py in␣
↪get_major_ticks(self, numticks)
   1369         r"""Return the list of major `.Tick`\s."""
   1370         if numticks is None:
-> 1371             numticks = len(self.get_majorticklocs())
   1372
   1373         while len(self.majorTicks) < numticks:

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\axis.py in␣
↪get_majorticklocs(self)
   1275     def get_majorticklocs(self):
```

```
   1276         """Return this Axis' major tick locations in data coordinates." "
-> 1277         return self.major.locator()
   1278
   1279     def get_minorticklocs(self):

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\ticker.py in␣
↪__call__(self)
   2112     def __call__(self):
   2113         vmin, vmax = self.axis.get_view_interval()
-> 2114         return self.tick_values(vmin, vmax)
   2115
   2116     def tick_values(self, vmin, vmax):

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\ticker.py in␣
↪tick_values(self, vmin, vmax)
   2120         vmin, vmax = mtransforms.nonsingular(
   2121             vmin, vmax, expander=1e-13, tiny=1e-14)
-> 2122         locs = self._raw_ticks(vmin, vmax)
   2123
   2124         prune = self._prune

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\ticker.py in␣
↪_raw_ticks(self, vmin, vmax)
   2059         if self._nbins == 'auto':
   2060             if self.axis is not None:
-> 2061                 nbins = np.clip(self.axis.get_tick_space(),
   2062                                 max(1, self._min_n_ticks - 1), 9)
   2063             else:

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\axis.py in␣
↪get_tick_space(self)
   2523     def get_tick_space(self):
   2524         ends = mtransforms.Bbox.from_bounds(0, 0, 1, 1)
-> 2525         ends = ends.transformed(self.axes.transAxes -
   2526                                 self.figure.dpi_scale_trans)
   2527         length = ends.height * 72

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\transforms.py in␣
↪transformed(self, transform)
    490         """
    491         pts = self.get_points()
--> 492         ll, ul, lr = transform.transform(np.array(
    493             [pts[0], [pts[0, 0], pts[1, 1]], [pts[1, 0], pts[0, 1]]]))
    494         return Bbox([ll, [lr[0], ul[1]]])
```

```
~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\transforms.py in␣
↪transform(self, values)
   1501
   1502         # Transform the values
-> 1503         res = self.transform_affine(self.transform_non_affine(values))
   1504
   1505         # Convert the result back to the shape of the input values.

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\transforms.py in␣
↪transform_affine(self, points)
   2417     def transform_affine(self, points):
   2418         # docstring inherited
-> 2419         return self.get_affine().transform(points)
   2420
   2421     def transform_non_affine(self, points):

~\anaconda3\envs\5_Programmcodes\lib\site-packages\matplotlib\transforms.py in␣
↪get_affine(self)
   2444             return self._b.get_affine()
   2445         else:
-> 2446             return Affine2D(np.dot(self._b.get_affine().get_matrix(),
   2447                                    self._a.get_affine().get_matrix()))
   2448

<__array_function__ internals> in dot(*args, **kwargs)

KeyboardInterrupt:
```

[86]:
```
print(eq(x_f, d[1][0]) - eq(x_g, d[1][1]))
print(dd(x_f, x_g))
```

```
tensor([-0.0955,  0.0591,  0.0608], grad_fn=<SubBackward0>)
tensor([-0.0982, -0.0966, -0.0876], grad_fn=<DivBackward0>)
```

[66]:
```
print((y_g - y_f)/(x_g - x_f))
print(fpd.first_derivative(**d[1][0],x=x_f)/scale)
print(((y_g - y_f)/(x_g - x_f) - fpd.first_derivative(**d[1][0],x=x_f)/scale))
```

```
tensor([0.0369, 0.0463, 0.0386, 0.0270, 0.0249, 0.0190, 0.0194, 0.0261, 0.0196,
        0.0241, 0.0185, 0.0225, 0.0165, 0.0244, 0.0232, 0.0174, 0.0130, 0.0170,
        0.0132, 0.0144, 0.0171, 0.0162, 0.0169, 0.0191, 0.0195, 0.0208, 0.0218,
        0.0223, 0.0334, 0.0382], grad_fn=<DivBackward0>)
tensor([-0.3485, -0.2881, -0.2258, -0.2450, -0.2406, -0.2673, -0.2423, -0.1902,
        -0.1959, -0.1525, -0.1896, -0.1491, -0.1583, -0.0404, -0.0341, -0.1019,
        -0.1292, -0.0550, -0.1103, -0.0802,  0.0054,  0.0701, -0.0363,  0.0336,
         0.0123, -0.0022,  0.0273,  0.0188,  0.1098,  0.1526],
       grad_fn=<DivBackward0>)
```

```
tensor([ 0.3853,  0.3344,  0.2644,  0.2720,  0.2655,  0.2863,  0.2617,  0.2163,
         0.2155,  0.1766,  0.2081,  0.1716,  0.1748,  0.0648,  0.0574,  0.1193,
         0.1423,  0.0720,  0.1235,  0.0946,  0.0117, -0.0538,  0.0532, -0.0145,
         0.0072,  0.0230, -0.0055,  0.0035, -0.0764, -0.1144],
       grad_fn=<SubBackward0>)
```

[ ]: