Inverse_problem_steady_state_bc

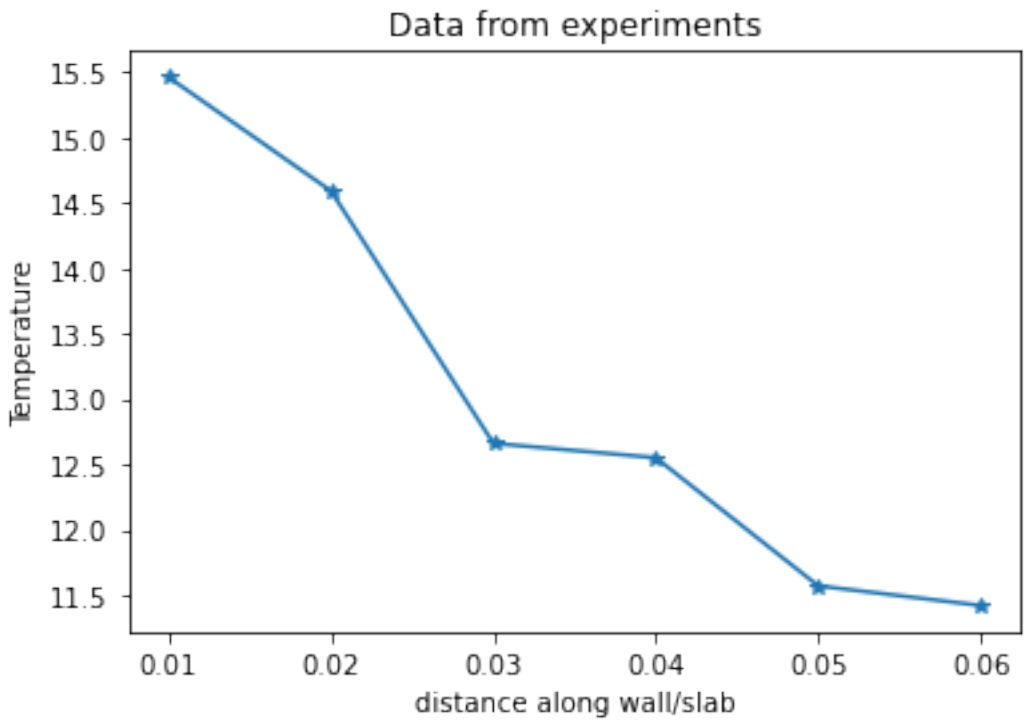
November 19, 2023

[20]:
```
import numpy as np
import matplotlib.pyplot as plt
import sciann as sn
```

[22]:
```
##Number of thermocouples attached
N_t=6
##distance along bar
x_data=np.zeros((N_t,1))
x_data[0]=0.01
x_data[1]=0.02
x_data[2]=0.03
x_data[3]=0.04
x_data[4]=0.05
x_data[5]=0.06
##thermal conductivity of material
k=14.4
##length of slab
L=0.07
```

[23]:
```
##Temperature readings from thermocouple
T_data=np.zeros((N_t,1))
T_data[0]=15.46
T_data[1]=14.59
T_data[2]=12.66
T_data[3]=12.55
T_data[4]=11.57
T_data[5]=11.42
```

[24]:
```
plt.plot(x_data,T_data,'-*')
plt.xlabel('distance along wall/slab')
plt.ylabel('Temperature')
plt.title('Data from experiments')
plt.show()
```



[25]:
```
##Compute mean
T_m=np.mean(T_data)
##product
T_x=T_data*x_data
##square distance
x_2=x_data*x_data
# ##distance from mean
T_Tm=(T_data-T_m)**2
```

[27]:
```
##find co-efficients
a=(N_t*np.sum(T_x)-np.sum(x_data)*np.sum(T))/(N_t*np.sum(x_2)-(np.
 ↪sum(x_data))**2)
print(a)
b=(np.sum(T_data)-a*np.sum(x_data))/(N_t)
print(b)
```

```
-83.91428571428625
15.978666666666689
```

[28]:
```
##Compute curve fitting temperature
T_fit=a*x_data+b
```

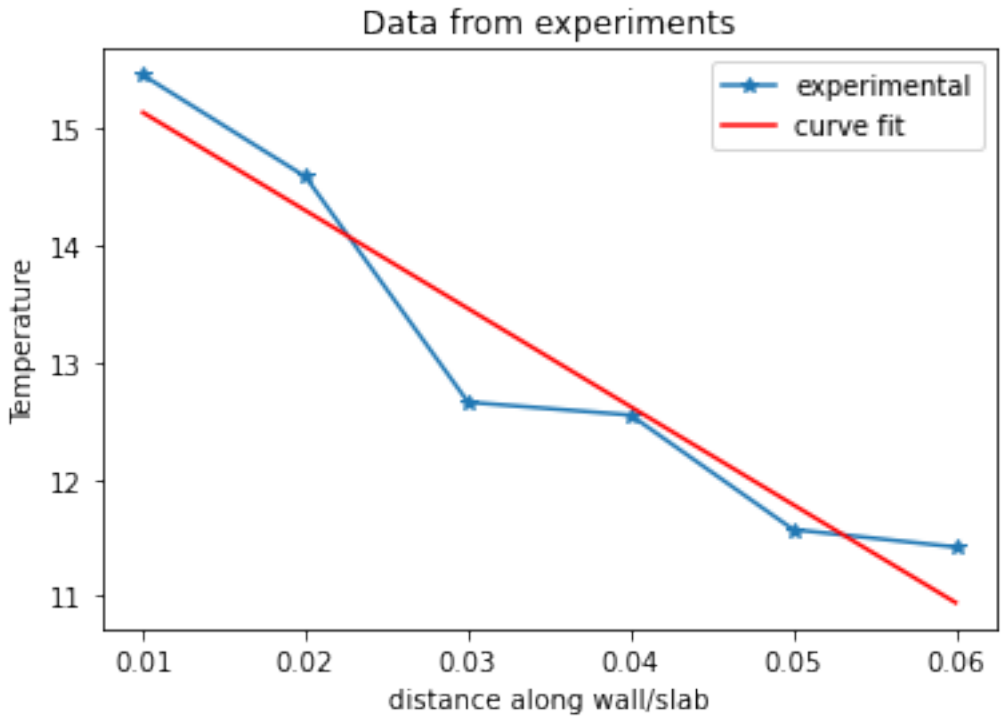
[29]:
```
Table=np.concatenate((x_data,T_data,T_x,x_2,T_Tm,T_fit),axis=1)
```

[30]:
```
print(np.round(Table,2))
```

```
[[1.000e-02 1.546e+01 1.500e-01 0.000e+00 5.850e+00 1.514e+01]
 [2.000e-02 1.459e+01 2.900e-01 0.000e+00 2.400e+00 1.430e+01]
 [3.000e-02 1.266e+01 3.800e-01 0.000e+00 1.500e-01 1.346e+01]
 [4.000e-02 1.255e+01 5.000e-01 0.000e+00 2.400e-01 1.262e+01]
 [5.000e-02 1.157e+01 5.800e-01 0.000e+00 2.170e+00 1.178e+01]
 [6.000e-02 1.142e+01 6.900e-01 0.000e+00 2.630e+00 1.094e+01]]
```

[31]:
```
##find co-efficient of determination
## find correlation co-efficient
S_fit=np.sum((T_data-T_fit)**2)
S_t=np.sum((T_data-T_m)**2)
gamma_2=(S_t-S_fit)/(S_t)
r=np.sqrt(gamma_2)
```

[33]:
```
##Curve fit plot
plt.plot(x_data,T_data,'-*',label='experimental')
plt.plot(x_data,T_fit,'r',label='curve fit')
plt.xlabel('distance along wall/slab')
plt.ylabel('Temperature')
plt.title('Data from experiments')
plt.legend()
plt.show()
```

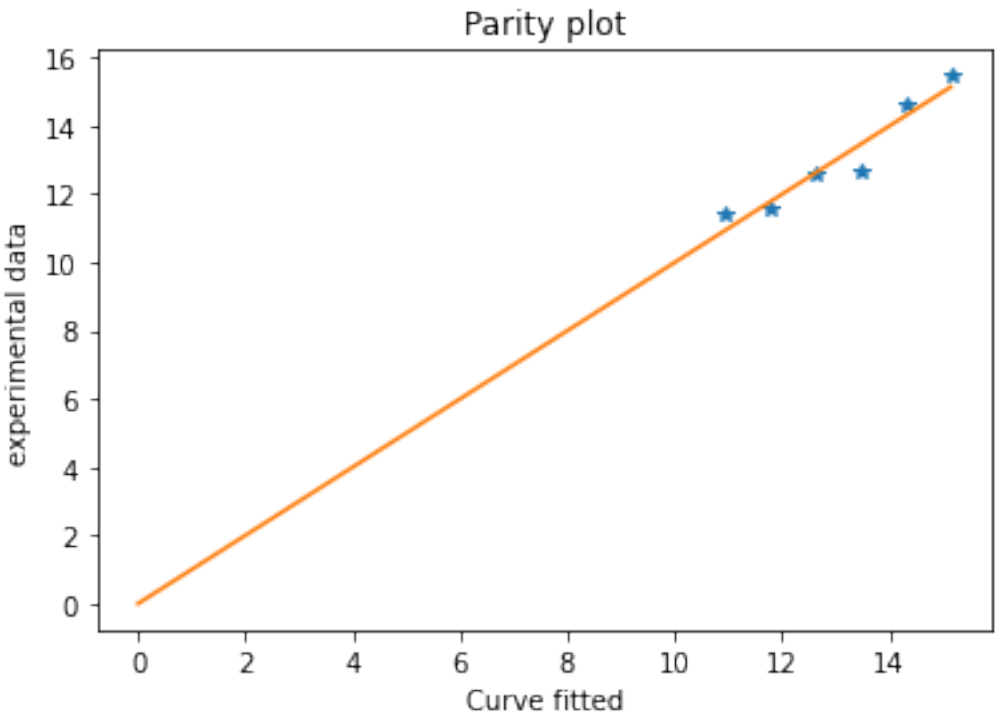


[34]:
```
##heat flux at left boundary
q=-k*a
## Temperature at right boundary x=0.07
T_r=a*L+b
```

[35]:
```
print(q)
print(T_r)
```

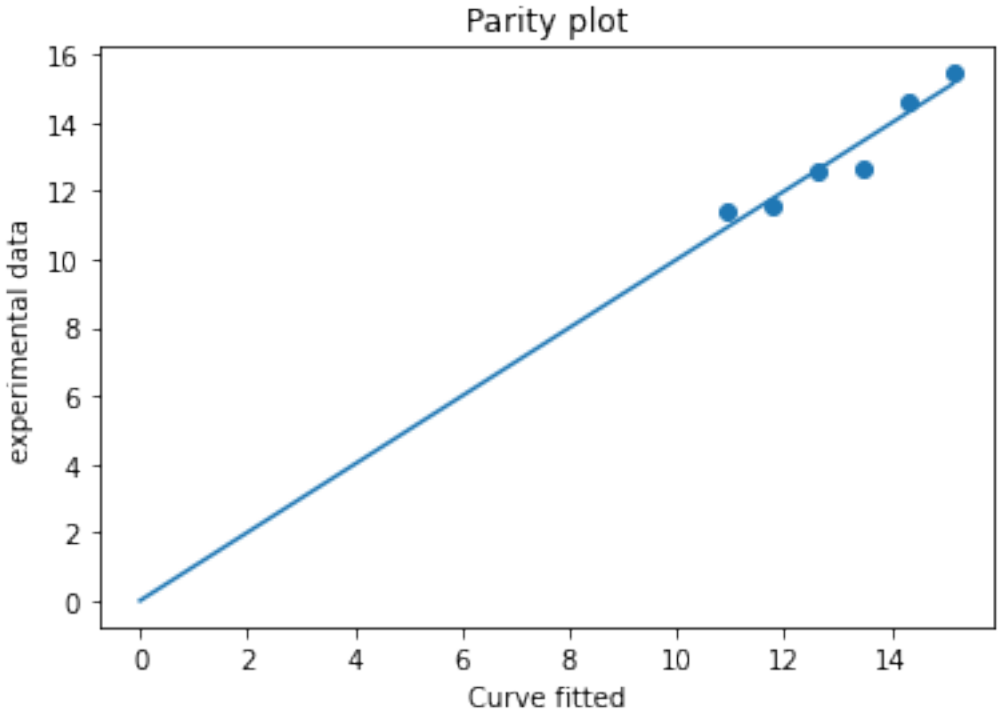
```
1208.365714285722
10.10466666666665
```

[37]:
```
x_p=np.linspace(0,np.max(T_fit),6)
y_p=x_p
plt.plot(T_fit,T_data,'*')
plt.plot(x_p,y_p)
plt.title('Parity plot')
plt.xlabel('Curve fitted')
plt.ylabel('experimental data')
plt.show()
```



[38]:
```
plt.scatter(T_fit,T_data)
plt.plot(x_p,y_p)
plt.title('Parity plot')
plt.xlabel('Curve fitted')
```

```
plt.ylabel('experimental data')
plt.show()
```



[22]:
```
T_fit
```

[22]:
```
array([[15.13952381],
       [14.30038095],
       [13.4612381 ],
       [12.62209524],
       [11.78295238],
       [10.94380952]])
```

[42]:
```
print(gamma_2)
```

```
0.9176486295238319
```

[44]:
```
print(r)
```

```
0.9579397838715291
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: