HW2-plot-JS

November 19, 2020

1 Plotting Jazayeri & Shadlen (2010) data

Code to reproduce the plot in their paper: https://www.nature.com/articles/nn.2590.pdf

[3]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[4]:

```
dat = pd.read_csv("dataJS.csv")
dat.head()
```

[4]:

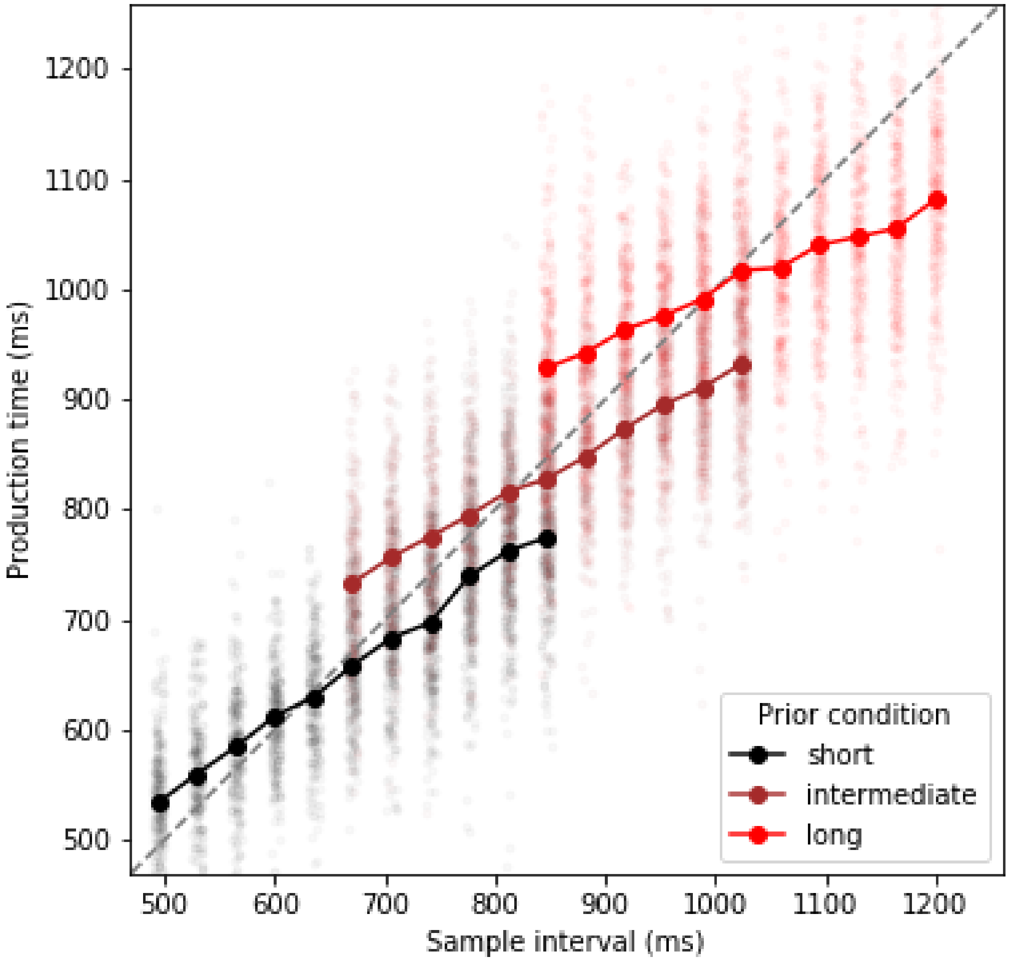
```
   Subj  Cond  line  Trial      Ts      Tp  MaxTrial   Main
0     1     1     1      1  847.01  863.26      1489  False
1     1     1     2      2  705.84  683.01      1489  False
2     1     1     3      3  670.55  711.81      1489  False
3     1     1     4      4  847.01  829.67      1489  False
4     1     1     5      5  705.84  695.06      1489  False
```

[5]:

```
# Remove training trials
dat = dat[dat['Main'] == True]

# Calculate mean Tp by condition
mean_tp = dat.groupby(['Cond', 'Ts'])['Tp'].mean().reset_index()


yrange = np.multiply((min(mean_tp['Ts']), max(mean_tp['Ts'])), [0.95, 1.05])

# Subset data for plotting

cond1 = mean_tp.loc[mean_tp['Cond'] == 1]
cond2 = mean_tp.loc[mean_tp['Cond'] == 2]
cond3 = mean_tp.loc[mean_tp['Cond'] == 3]

# Add jitter noise
jitter = dat.copy()
jitter['Ts'] = jitter['Ts'] + np.random.uniform(-5, 5, len(dat))
cond1_jitter = jitter.loc[jitter['Cond'] == 1]
cond2_jitter = jitter.loc[jitter['Cond'] == 2]
```

```
cond3_jitter = jitter.loc[jitter['Cond'] == 3]

# Make plot
f, ax = plt.subplots(figsize = (6,6))

ax.set(xlim = yrange, ylim = yrange)
f.gca().set_aspect('equal', adjustable = 'box')

ax.set_xlabel('Sample interval (ms)')
ax.set_ylabel('Production time (ms)')

ax.plot(yrange, yrange, linestyle = '--', color ='gray')

ax.scatter(cond1_jitter['Ts'], cond1_jitter['Tp'], marker = '.', color =␣
 ↪'black', alpha = 0.025, label = None)
ax.scatter(cond2_jitter['Ts'], cond2_jitter['Tp'], marker = '.', color =␣
 ↪'brown', alpha = 0.025, label = None)
ax.scatter(cond3_jitter['Ts'], cond3_jitter['Tp'], marker = '.', color = 'red',␣
 ↪alpha = 0.025, label = None)

ax.plot(cond1['Ts'], cond1['Tp'], color = 'black', marker = 'o', label =␣
 ↪"short")
ax.plot(cond2['Ts'], cond2['Tp'], color = 'brown', marker = 'o', label =␣
 ↪"intermediate")
ax.plot(cond3['Ts'], cond3['Tp'], color = 'red', marker = 'o', label = "long")

ax.legend(title = 'Prior condition', loc = 4)
```

```
[5]: <matplotlib.legend.Legend at 0x7fac9956c2b0>
```


Production time (ms)
1200
1100
1000
900
800
700
600
500
Sample interval (ms)
500
600
700
800
900
1000
1100
1200
Prior condition
short
intermediate
long

```
[ ]:
```