Fair-IRT_01

June 12, 2024

[1]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
import numpy as np
import pandas as pd

from matplotlib import rcParams

from irt import Beta3
```

[2]:
```
n_rows = 50 # number of individual
n_cols = 20 # number of prediction models

np.random.seed(0)
abil = np.random.rand(n_cols)
diff = np.random.rand(n_rows)

discr = np.random.normal(1,1,size = n_rows)

pij = pd.DataFrame(index=range(n_rows), columns=range(n_cols))

for i in range(n_rows):
    for j in range(n_cols):
        alpha = (abil[j] / diff[i]) ** discr[i]
        beta_val = ((1 - abil[j]) / (1 - diff[i])) ** discr[i]
        pij.iloc[i, j] = (alpha)/(alpha+beta_val)
```

[3]:
```
normalized_df = pij.astype(np.float32)
```

[4]:
```
def ICC_function(abilities, difficulties, discriminations):
    a = ((1-abilities)/ abilities)
    b = (difficulties / (1-difficulties))
    c = a*b
    d = c**discriminations
    return (1 / (d+1))
```

[5]:
```
b4 = Beta3(
        learning_rate=1,
        epochs=5000,
        n_respondents=normalized_df.shape[1],
        n_items=normalized_df.shape[0],
        n_workers=-1,
        random_seed=1,
    )
b4.fit(normalized_df.values)
```

```
100%|      | 5000/5000 [00:07<00:00, 686.68it/s]
```

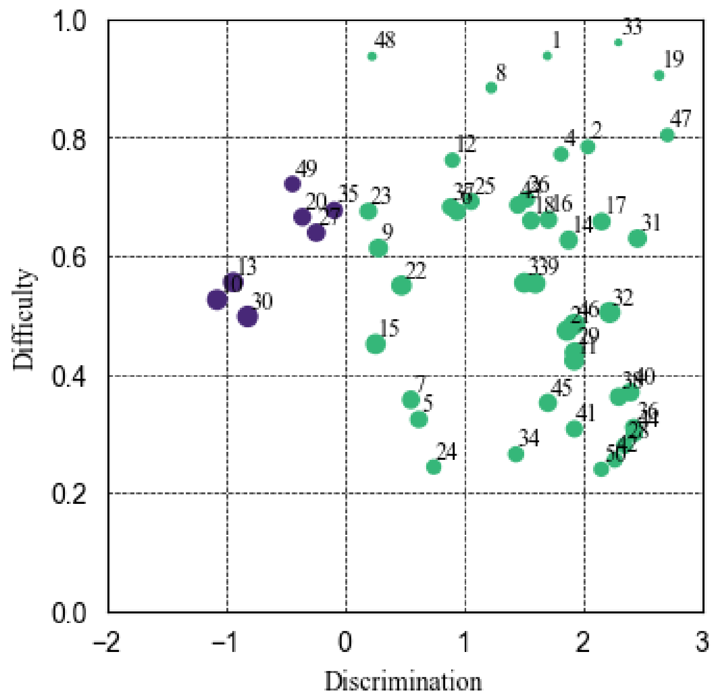
[5]:
```
<irt.Beta3 at 0x337409b80>
```

[6]:
```
new_pij = pd.DataFrame(index=range(n_rows), columns=range(n_cols))

for i in range(n_rows):
    for j in range(n_cols):
        alpha = (b4.abilities[j] / b4.difficulties[i]) ** b4.discriminations[i]
        beta_val = ((1 - b4.abilities[j]) / (1 - b4.difficulties[i])) ** b4.
↪discriminations[i]
        new_pij.iloc[i, j] = (alpha)/(alpha+beta_val)
```

1 Figure 2

[7]:
```
def plot_discriminations_difficulties(discriminations, difficulties,␣
↪normalized_df, font_size=10, font_ann_size=5, base_point_size=500):

    rcParams['font.family'] = 'serif'
    rcParams['font.serif'] = ['Times New Roman']

    sns.set_style('whitegrid')
    fig, ax = plt.subplots(figsize=(4, 4))
    ax.grid(color='black', linestyle='--', linewidth=0.5)

    ax.spines['bottom'].set_color('black')
    ax.spines['left'].set_color('black')
    ax.spines['right'].set_color('black')
    ax.spines['top'].set_color('black')
    ax.xaxis.label.set_color('black')
    ax.yaxis.label.set_color('black')
    ax.tick_params(axis='x', colors='black')
    ax.tick_params(axis='y', colors='black')

    point_sizes = base_point_size * (1 - np.abs(difficulties - 0.5) * 2)

    colors = []
```

```
    for disc, diff in zip(discriminations, difficulties):
        base_color = '#482878' if disc < 0 else '#35b779'
        intensity = 1
        color = mcolors.to_rgba(base_color, intensity)
        colors.append(color)

    scatter = ax.scatter(discriminations, difficulties, s=point_sizes, c=colors)

    for i in range(normalized_df.shape[0]):
        ax.text(x=discriminations[i]+0.015, y=difficulties[i]+0.015,␣
↪s=f'{normalized_df.index[i]+1}', fontsize=font_ann_size, family='Times New␣
↪Roman', color='black')

    plt.xlabel(r'Discrimination', fontsize=font_size, family='Times New Roman',␣
↪color='black')
    plt.ylabel(r'Difficulty', fontsize=font_size, family='Times New Roman',␣
↪color='black')

    ax.set_ylim(0, 1)

    if discriminations.min() < 0:
        x_min = int(discriminations.min()) - 1
    else:
        x_min = int(discriminations.min())

    if discriminations.max() > 0:
        x_max = int(discriminations.max()) + 1
    else:
        x_max = int(discriminations.max())

    ax.set_xlim(x_min, x_max)

plot_discriminations_difficulties(b4.discriminations, b4.difficulties, new_pij,␣
↪font_size=10, font_ann_size=9, base_point_size=50)
```

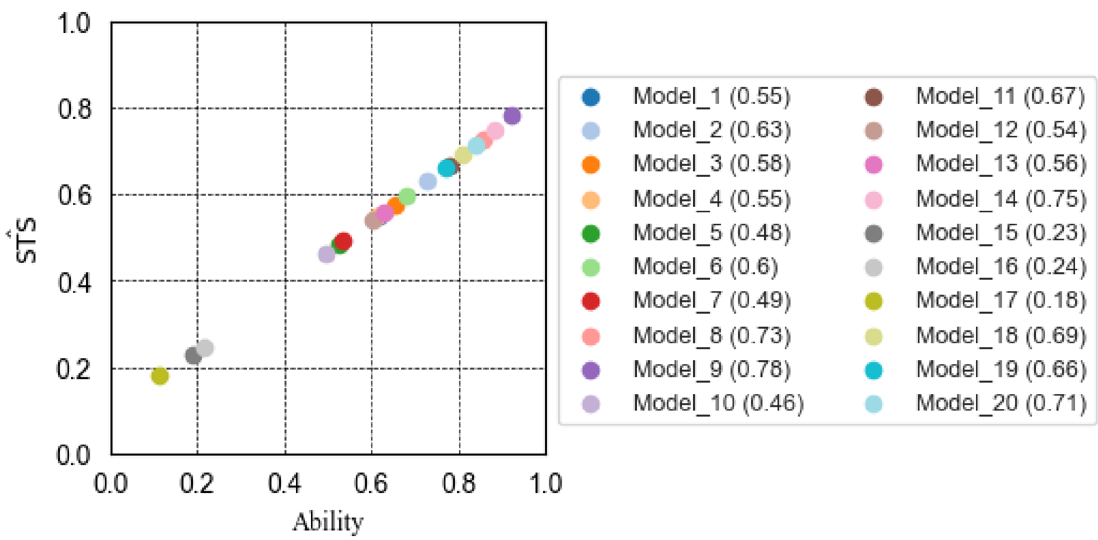


[8]:
```
fairness_model = new_pij.apply(np.mean,axis=0).to_numpy()
```

2 Figure 5

[9]:
```
def plot_abilities_fairness(abilities, fairness_model, font_size=10):
    plt.rcParams["font.family"] = "Times New Roman"

    sns.set_style('whitegrid')
    fig, ax = plt.subplots(figsize=(3, 3))
    ax.grid(color='black', linestyle='--', linewidth=0.5)

    ax.spines['bottom'].set_color('black')
    ax.spines['left'].set_color('black')
    ax.spines['right'].set_color('black')
    ax.spines['top'].set_color('black')
    ax.xaxis.label.set_color('black')
    ax.yaxis.label.set_color('black')
    ax.tick_params(axis='x', colors='black')
    ax.tick_params(axis='y', colors='black')

    colors = sns.color_palette('tab20', n_colors=20)

    for i in range(fairness_model.shape[0]):
```

```
        plt.scatter(abilities[i], fairness_model[i], label=f'Model_{i+1}␣
↪({round(fairness_model[i], 2)})', color=colors[i])

    plt.xlabel(r'Ability', fontsize=font_size, family='Times New Roman',␣
↪color='black')
    plt.ylabel(r'$\hat{\text{STS}}$', fontsize=font_size, family='Times New␣
↪Roman', color='black')
    plt.legend(title='', fontsize=9, bbox_to_anchor=(1, 0.9), loc='upper left',␣
↪ncol=2)

    ticks = np.arange(0, 1.1, 0.2)
    plt.xlim(0, 1)
    plt.ylim(0, 1)
    plt.xticks(ticks)
    plt.yticks(ticks)

    plt.show()

plot_abilities_fairness(b4.abilities, fairness_model, font_size=10)
```

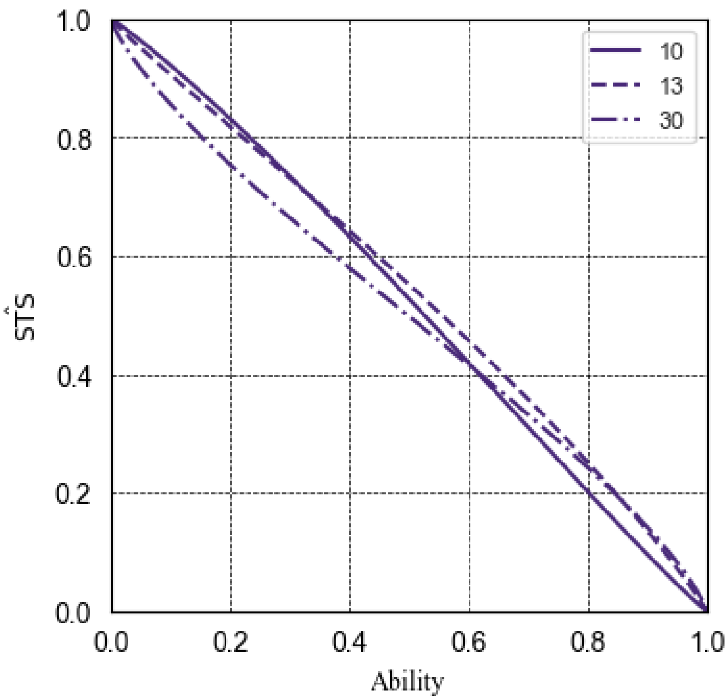


[10]:
```
abilities = np.linspace(0.001, 0.999, 1000)
```

3 Figure 3(a)

[11]:
```
plt.figure(figsize=(4, 4))
plt.rcParams["font.family"] = "Times New Roman"

sns.set_style('whitegrid')
fig, ax = plt.subplots(figsize=(4, 4))
ax.grid(color='black', linestyle='--', linewidth=0.5)

ax.spines['bottom'].set_color('black')
ax.spines['left'].set_color('black')
ax.spines['right'].set_color('black')
ax.spines['top'].set_color('black')
ax.xaxis.label.set_color('black')
ax.yaxis.label.set_color('black')
ax.tick_params(axis='x', colors='black')
ax.tick_params(axis='y', colors='black')

linestyles = ['-', '--', '-.', ':']
num_linestyles = len(linestyles)
i =0

for index in range (50):
    if b4.difficulties[index] < 0.6 and b4.difficulties[index] > 0.4:
        if b4.discriminations[index] < 0:
            fairness = ICC_function(abilities, b4.difficulties[index], b4.
 ↪discriminations[index])
            linestyle = linestyles[i % num_linestyles]
            plt.plot(abilities, fairness, label=f'{index+1}', color='#482878',␣
 ↪linestyle=linestyle)
            i = i+1

plt.xlim(0, 1)
plt.ylim(0, 1)

plt.legend(loc='upper right', fontsize=9)

plt.xlabel(r'Ability', fontsize=10, family='Times New Roman')
plt.ylabel(r'$\hat{\text{STS}}$', fontsize=10, family='Times New Roman')

plt.grid(True)
plt.show()
```

```
<Figure size 400x400 with 0 Axes>
```



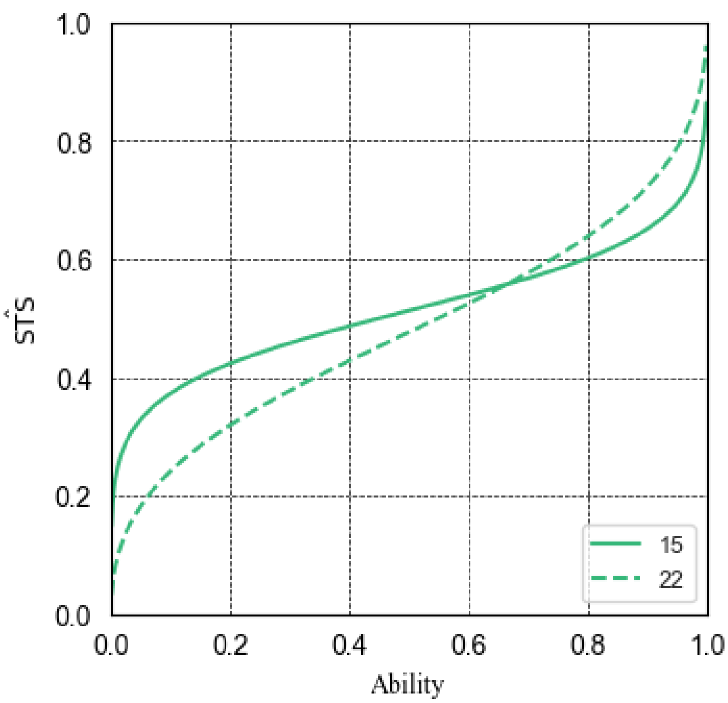
4 Figure 3(b)

[12]:

```
plt.figure(figsize=(4, 4))
plt.rcParams["font.family"] = "Times New Roman"

sns.set_style('whitegrid')
fig, ax = plt.subplots(figsize=(4, 4))
ax.grid(color='black', linestyle='--', linewidth=0.5)

ax.spines['bottom'].set_color('black')
ax.spines['left'].set_color('black')
ax.spines['right'].set_color('black')
ax.spines['top'].set_color('black')
ax.xaxis.label.set_color('black')
ax.yaxis.label.set_color('black')
ax.tick_params(axis='x', colors='black')
ax.tick_params(axis='y', colors='black')

linestyles = ['-', '--', '-.', ':']
num_linestyles = len(linestyles)
i = 0

for index in range (50):
```

```
    if b4.difficulties[index] < 0.6 and b4.difficulties[index] > 0.4:
        if b4.discriminations[index] < 1 and b4.discriminations[index] > 0:
            fairness = ICC_function(abilities, b4.difficulties[index], b4.
 ↪discriminations[index])
            linestyle = linestyles[i % num_linestyles]
            plt.plot(abilities, fairness, label=f'{index+1}', color='#35b779',␣
 ↪linestyle=linestyle)
            i = i+1

plt.xlim(0, 1)
plt.ylim(0, 1)

plt.legend(loc='lower right', fontsize=9)

plt.xlabel(r'Ability', fontsize=10, family='Times New Roman')
plt.ylabel(r'$\hat{\text{STS}}$', fontsize=10, family='Times New Roman')

plt.grid(True)
plt.show()
```

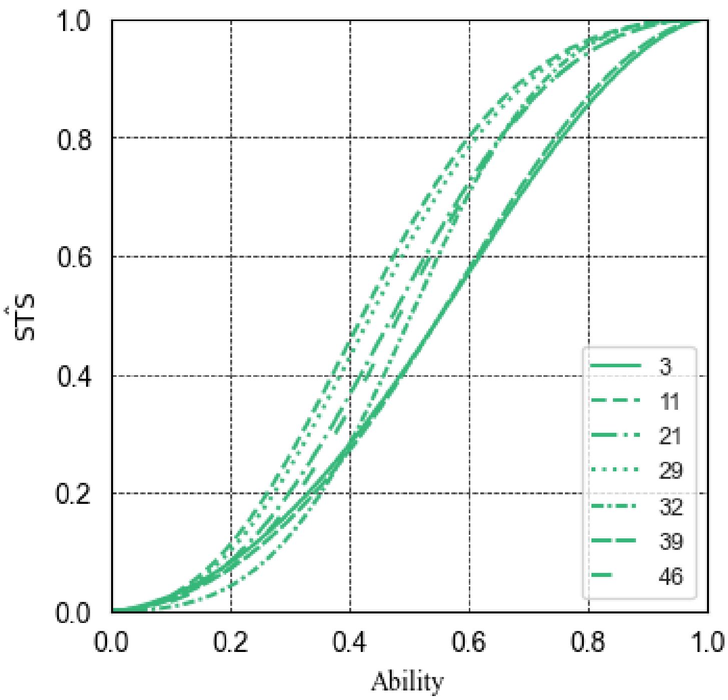
```
<Figure size 400x400 with 0 Axes>
```



5 Figure 3(c)

[13]:
```
plt.figure(figsize=(4, 4))
plt.rcParams["font.family"] = "Times New Roman"

sns.set_style('whitegrid')
fig, ax = plt.subplots(figsize=(4, 4))
ax.grid(color='black', linestyle='--', linewidth=0.5)

ax.spines['bottom'].set_color('black')
ax.spines['left'].set_color('black')
ax.spines['right'].set_color('black')
ax.spines['top'].set_color('black')
ax.xaxis.label.set_color('black')
ax.yaxis.label.set_color('black')
ax.tick_params(axis='x', colors='black')
ax.tick_params(axis='y', colors='black')

linestyles = ['-', '--', '-.', ':', (0, (3, 1, 1, 1)), (0, (5, 1)), (0, (5,␣
↪10))]
num_linestyles = len(linestyles)
i =0

for index in range (50):
    if b4.difficulties[index] < 0.6 and b4.difficulties[index] > 0.4:
        if b4.discriminations[index] > 1:
            fairness = ICC_function(abilities, b4.difficulties[index], b4.
↪discriminations[index])
            linestyle = linestyles[i % num_linestyles]
            plt.plot(abilities, fairness, label=f'{index+1}', color='#35b779',␣
↪linestyle=linestyle)
            i =i+1

plt.xlim(0, 1)
plt.ylim(0, 1)

plt.legend(loc='lower right', fontsize=9)

plt.xlabel(r'Ability', fontsize=10, family='Times New Roman')
plt.ylabel(r'$\hat{\text{STS}}$', fontsize=10, family='Times New Roman')

plt.grid(True)
plt.show()
```

```
<Figure size 400x400 with 0 Axes>
```

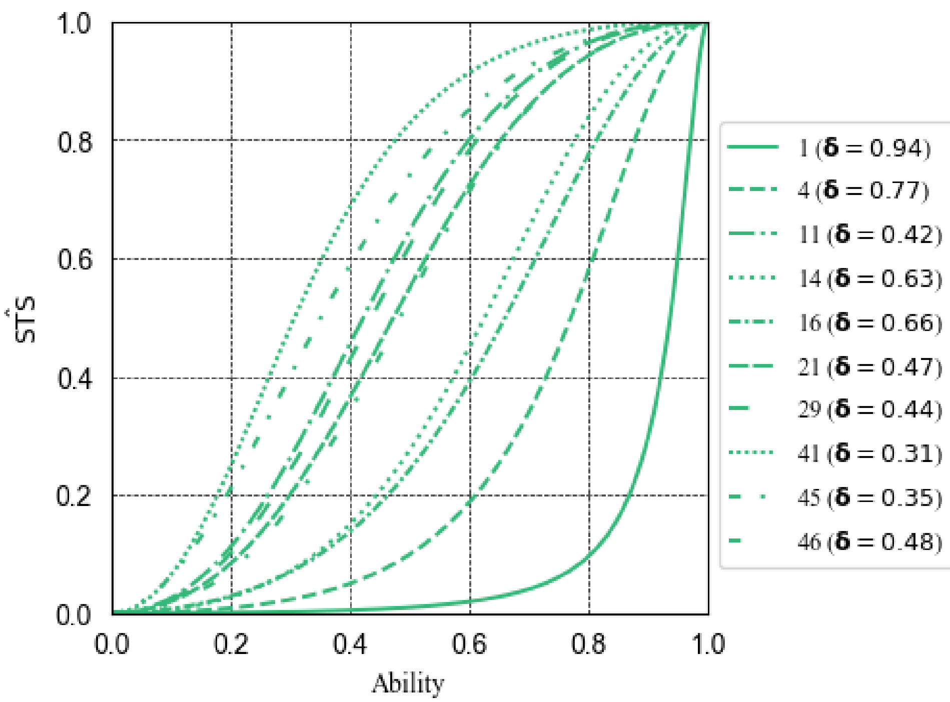


6 Figure 4

[14]:
```
plt.figure(figsize=(4, 4))
plt.rcParams["font.family"] = "Times New Roman"

sns.set_style('whitegrid')
fig, ax = plt.subplots(figsize=(4, 4))
ax.grid(color='black', linestyle='--', linewidth=0.5)

ax.spines['bottom'].set_color('black')
ax.spines['left'].set_color('black')
ax.spines['right'].set_color('black')
ax.spines['top'].set_color('black')
ax.xaxis.label.set_color('black')
ax.yaxis.label.set_color('black')
ax.tick_params(axis='x', colors='black')
ax.tick_params(axis='y', colors='black')

linestyles = ['-', '--', '-.', ':', (0, (3, 1, 1, 1)),
              (0, (5, 1)), (0, (5, 10)), (0, (1, 1)),
              (0, (3, 5, 1, 5)), (0, (3, 10, 1, 10))]
num_linestyles = len(linestyles)
i =0
```

```python
for index in range (50):
    if b4.discriminations[index] > 1.7 and b4.discriminations[index]<2:
        fairness = ICC_function(abilities, b4.difficulties[index], b4.discriminations[index])
        linestyle = linestyles[i % num_linestyles]
        plt.plot(abilities, fairness, label=f'{index+1} ($\mathbf{{\delta}}={b4.difficulties[index]:.2f}$)', color='#35b779', linestyle=linestyle)
        i += 1

plt.xlim(0, 1)
plt.ylim(0, 1)

legend = plt.legend(loc='upper left', fontsize=9, bbox_to_anchor=(1, 0.85), ncol=1)

for text in legend.get_texts():
    text.set_fontname('Times New Roman')
    text.set_color('black')

plt.xlabel(r'Ability', fontsize=10, family='Times New Roman')
plt.ylabel(r'$\hat{\text{STS}}$', fontsize=10, family='Times New Roman')

plt.grid(True)
plt.show()
```

```
<Figure size 400x400 with 0 Axes>
```



[15]:
```
def f(theta,delta_j,a_j):
    term1 = (delta_j / (1 - delta_j)) ** a_j
    term2 = (theta / (1 - theta)) ** (-a_j - 1)
    numerator = a_j * term1 * term2
    denominator = (1 + term1 * (theta / (1 - theta)) ** -a_j) ** 2
    return numerator / denominator * (1 / (1 - theta) ** 2)
```

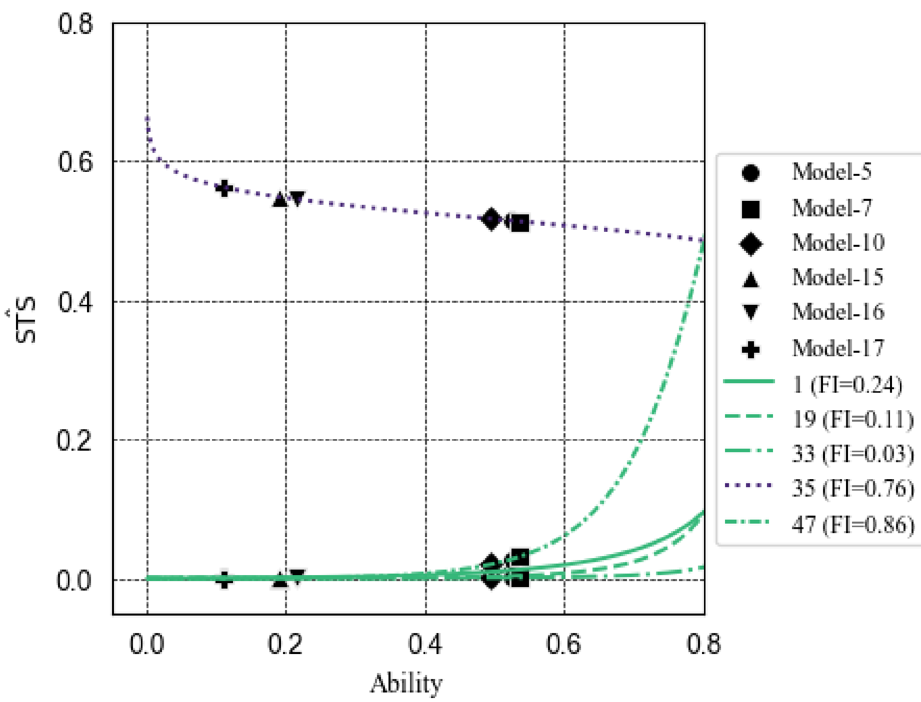
7 Figure 6 (Theorem 1)

[16]:
```
plt.figure(figsize=(4, 4))
plt.rcParams["font.family"] = "Times New Roman"

sns.set_style('whitegrid')
fig, ax = plt.subplots(figsize=(4, 4))
ax.grid(color='black', linestyle='--', linewidth=0.5)

ax.spines['bottom'].set_color('black')
ax.spines['left'].set_color('black')
ax.spines['right'].set_color('black')
ax.spines['top'].set_color('black')
ax.xaxis.label.set_color('black')
ax.yaxis.label.set_color('black')
ax.tick_params(axis='x', colors='black')
ax.tick_params(axis='y', colors='black')

markers = ['o', 's', 'D', '^', 'v', 'P']
num_markers = len(markers)
colors = ['red', 'blue', 'green', 'orange', 'purple', 'brown']
num_colors = len(colors)
linestyles = ['-', '--', '-.', ':', (0, (3, 1, 1, 1)),
              (0, (5, 1)), (0, (5, 10)), (0, (1, 1)),
              (0, (3, 5, 1, 5)), (0, (3, 10, 1, 10))]
num_linestyles = len(linestyles)
i = 0
j = 0
id = 0
list = [0.25,0.11,0.02,0.76,0.86]

added_labels = set()

for index in [0, 18, 32, 34, 46]:
    total_f_theta = 0
    for x_idx, x in enumerate([4, 6, 9, 14, 15, 16]):
        marker = markers[x_idx % num_markers]
```

```
        fairness = ICC_function(b4.abilities[x], b4.difficulties[index], b4.
↪discriminations[index])
        f_theta = abs(f(b4.abilities[x], b4.difficulties[index], b4.
↪discriminations[index]))
        total_f_theta += f_theta
        total_f_theta = round(total_f_theta, 2)

        label = f'Model-{x+1}'
        if label not in added_labels:
            sns.scatterplot(x=[b4.abilities[x]], y=[fairness], marker=marker,␣
↪s=50, color='black', label=label)
            added_labels.add(label)
        else:
            sns.scatterplot(x=[b4.abilities[x]], y=[fairness], marker=marker,␣
↪s=50, color='black')


        i += 1
    linestyle = linestyles[j % num_linestyles]
    fairness_2 = ICC_function(abilities, b4.difficulties[index], b4.
↪discriminations[index])
    if b4.discriminations[index]>0:
        plt.plot(abilities, fairness_2, label=f'{index+1}␣
↪(FI={total_f_theta})',color='#35b779', linestyle=linestyle)
    else:
        plt.plot(abilities, fairness_2, label=f'{index+1}␣
↪(FI={total_f_theta})',color='#482878', linestyle=linestyle)
    j += 1
    id += 1


plt.xlim(-0.05, 0.8)
plt.ylim(-0.05, 0.8)

plt.yticks(np.arange(0, 1, 0.2))

plt.gca().set_aspect('equal', adjustable='box')

legend = plt.legend(loc='upper left', fontsize=9, bbox_to_anchor=(1, 0.8),␣
↪ncol=1)

for text in legend.get_texts():
    text.set_fontname('Times New Roman')
    text.set_color('black')

plt.xlabel(r'Ability', fontsize=10, family='Times New Roman')
```

```
plt.ylabel(r'$\hat{\text{STS}}$', fontsize=10, family='Times New Roman')

plt.grid(True)
plt.show()
```

<Figure size 400x400 with 0 Axes>



8 Theorem 2 (Individual "1" on Prediction Model "5")

[17]:
```
from irt_special import Beta3
```

[18]:
```
def ICC_function_special(abilities, difficulties):
    a = ((1-abilities)/ abilities)
    b = (difficulties / (1-difficulties))
    c = a*b
    d = c
    return (1 / (d+1))
```

[19]:
```
b4_special = Beta3(
        learning_rate=1,
        epochs=5000,
        n_respondents=normalized_df.shape[1],
        n_items=normalized_df.shape[0],
```

```
        n_workers=-1,
        random_seed=1,
    )
b4_special.fit(normalized_df.values)
```

```
100%|      | 5000/5000 [00:06<00:00, 809.62it/s]
```

[19]:
```
<irt_special.Beta3 at 0x337b3f7c0>
```

$$g(\hat{\mathrm{STS}}) = 2.83$$

[20]:
```
new_pij_special = pd.DataFrame(index=range(n_rows), columns=range(n_cols))

for i in range(n_rows):
    for j in range(n_cols):
        alpha = (b4_special.abilities[j] / b4_special.difficulties[i])
        beta_val = ((1 - b4_special.abilities[j]) / (1 - b4_special.
 ↪difficulties[i]))
        new_pij_special.iloc[i, j] = (alpha)/(alpha+beta_val)
```

[21]:
```
res = np.log(1-new_pij_special[4][0])-np.log(new_pij_special[4][0])
res
```

[21]:
```
2.8271178869682236
```

$$\log \Delta = 3.07$$

[22]:
```
delta_j_values = b4_special.difficulties[0]
Delta_j_values = delta_j_values / (1 - delta_j_values)

# log(Delta_j)
log_Delta_j_values = np.log(Delta_j_values)
log_Delta_j_values
```

[22]:
```
3.0683646146457577
```

$$\log \Theta = -0.24$$

[23]:
```
theta_i_values = b4_special.abilities[4]
Theta_i_values = (1 - theta_i_values) / theta_i_values

#   log(Delta_j)
log_Telta_j_values = np.log(Theta_i_values)
log_Telta_j_values
```

[23]:
```
-0.24124673261181603
```

$$\log \Delta + \log \Theta = g(\mathrm{S\hat{T}S}) = 2.83$$

[24]:
```
log_Telta_j_values + log_Delta_j_values
```

[24]: 2.8271178820339418