

Conflict & Adaptation Analysis

How Conflict Handling Relates to Team Outcomes

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

plt.style.use('seaborn-v0_8-whitegrid')
plt.rcParams['figure.figsize'] = (10, 6)
plt.rcParams['font.size'] = 11

# Load data
df = pd.read_excel('SurveyData211.xlsx')

# Key variables
OUTCOME_VARS = ['G01', 'NPS1', 'NPS2', 'SE1', 'SE2', 'RLS1']
VAR_LABELS = {
    'CA1': 'Conflict Adaptation (1-10)',
    'G01': 'Growth/Outcomes (1-5)',
    'NPS1': 'Team Satisfaction 1 (1-5)',
    'NPS2': 'Team Satisfaction 2 (1-5)',
    'SE1': 'Self-Efficacy 1 (1-5)',
    'SE2': 'Self-Efficacy 2 (1-5)',
    'RLS1': 'Relationship Strength (1-10)',
    'Section': 'Class Section'
}

print(f"Dataset: {df.shape[0]} responses, {df.shape[1]} variables")
```

Dataset: 210 responses, 27 variables

. Distribution of Conflict Adaptation Scores

```
In [2]: # Table 1: Descriptive Statistics for All Key Variables
from scipy import stats

summary_vars = ['CA1'] + OUTCOME_VARS
desc_stats = pd.DataFrame({
    'Variable': summary_vars,
    'N': [df[v].count() for v in summary_vars],
    'Mean': [df[v].mean().round(2) for v in summary_vars],
    'Median': [df[v].median() for v in summary_vars],
    'Std Dev': [df[v].std().round(2) for v in summary_vars],
    'Min': [df[v].min() for v in summary_vars],
    'Max': [df[v].max() for v in summary_vars]
})
print("Table 1: Descriptive Statistics")
desc_stats
```

Table 1: Descriptive Statistics

Out [2]:

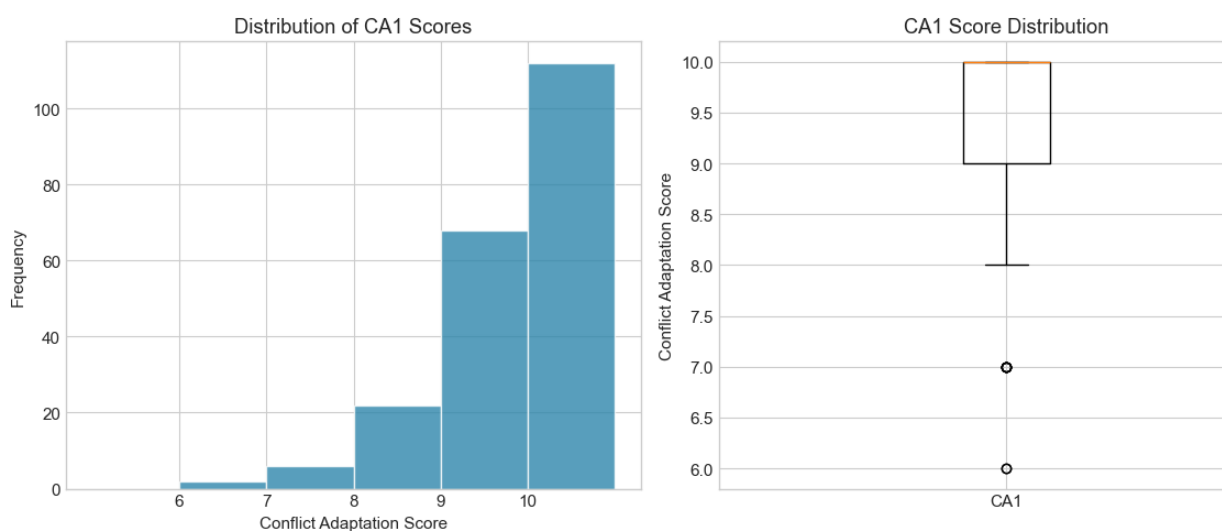
Variable	N	Mean	Median	Std Dev	Min	Max
CA
GO
NPS
NPS
SE
SE
RLS

```
In [3]: # Figure 1: Distribution of Conflict Adaptation Scores
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

# Histogram
axes[0].hist(df['CA1'].dropna(), bins=range(5, 12), edgecolor='white', co
axes[0].set_xlabel('Conflict Adaptation Score')
axes[0].set_ylabel('Frequency')
axes[0].set_title('Distribution of CA1 Scores')
axes[0].set_xticks(range(6, 11))

# Box plot
axes[1].boxplot(df['CA1'].dropna(), vert=True)
axes[1].set_ylabel('Conflict Adaptation Score')
axes[1].set_title('CA1 Score Distribution')
axes[1].set_xticklabels(['CA1'])

plt.tight_layout()
plt.show()
```



. Correlation Analysis: CA vs Outcome Variables

```
In [4]: # Correlation analysis
corr_vars = ['CA1'] + OUTCOME_VARS
corr_matrix = df[corr_vars].corr().round(3)

# Figure 2: Simple bar chart of CA1 correlations
```

```

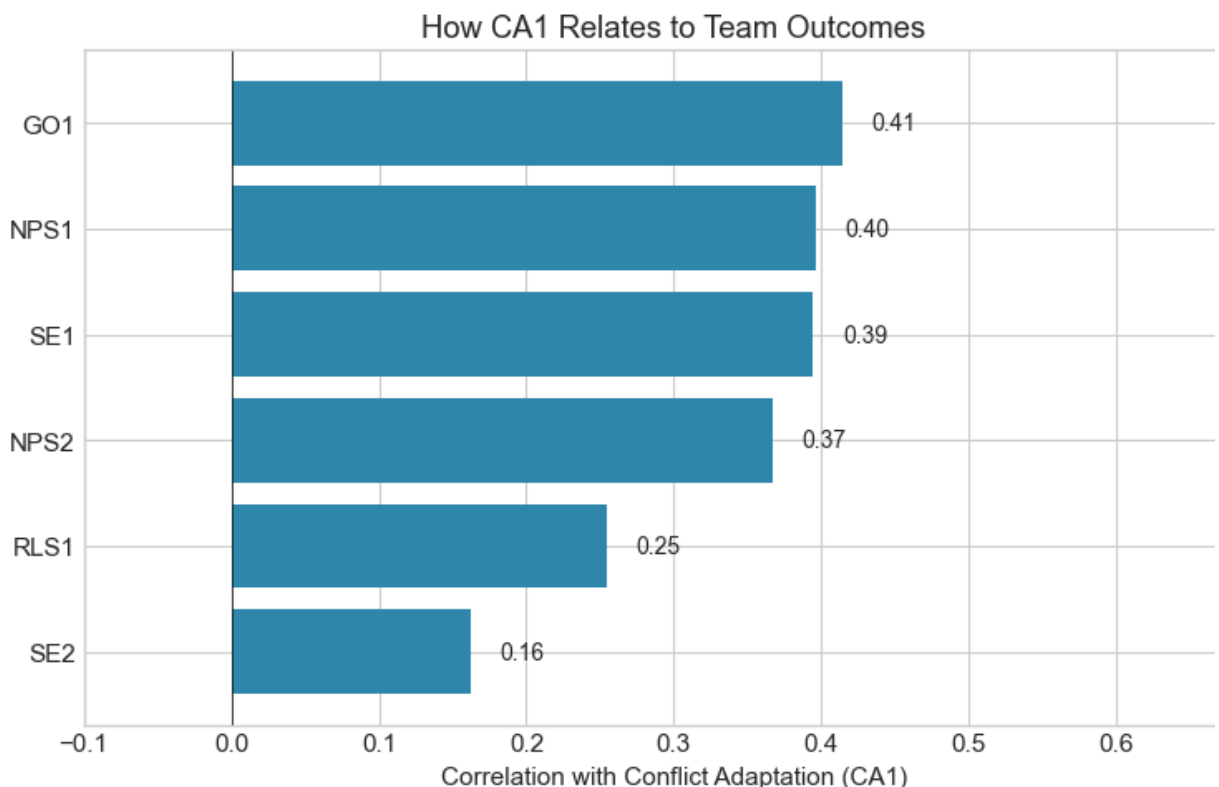
ca_corrs = corr_matrix['CA1'].drop('CA1').sort_values(ascending=True)

fig, ax = plt.subplots(figsize=(8, 5))
colors = ['#2E86AB' if r > 0 else '#E94F37' for r in ca_corrs.values]
bars = ax.barh(ca_corrs.index, ca_corrs.values, color=colors)
ax.set_xlabel('Correlation with Conflict Adaptation (CA1)')
ax.set_title('How CA1 Relates to Team Outcomes')
ax.axvline(x=0, color='black', linewidth=0.5)
ax.set_xlim(-0.1, 0.7)

# Add value labels
for bar, val in zip(bars, ca_corrs.values):
    ax.text(val + 0.02, bar.get_y() + bar.get_height()/2, f'{val:.2f}', v

plt.tight_layout()
plt.show()

```



```

In [5]: # Table 2: Correlations with Statistical Significance (p-values)
ca_corrs = corr_matrix['CA1'].drop('CA1').sort_values(ascending=False)

# Calculate p-values for each correlation
p_values = []
for var in ca_corrs.index:
    r, p = stats.pearsonr(df['CA1'].dropna(), df.loc[df['CA1'].notna(), v
    p_values.append(p)

corr_table = pd.DataFrame({
    'Variable': ca_corrs.index,
    'Correlation (r)': ca_corrs.values.round(3),
    'p-value': [f'{p:.4f}' for p in p_values],
    'Significant (p<0.05)': ['Yes' if p < 0.05 else 'No' for p in p_value
    'Interpretation': ['Strong' if abs(r) > 0.5 else 'Moderate' if abs(r)
                        else 'Weak' for r in ca_corrs.values]
})

```

```
print("Table 2: CA1 Correlations with Outcome Variables")
corr_table
```

Table 2: CA1 Correlations with Outcome Variables

Out [5]:	Variable	Correlation (r)	p-value	Significant (p< .)	Interpretation
	GO	.	.	Yes	Moderate
	NPS	.	.	Yes	Moderate
	SE	.	.	Yes	Moderate
	NPS	.	.	Yes	Moderate
	RLS	.	.	Yes	Weak
	SE	.	.	Yes	Weak

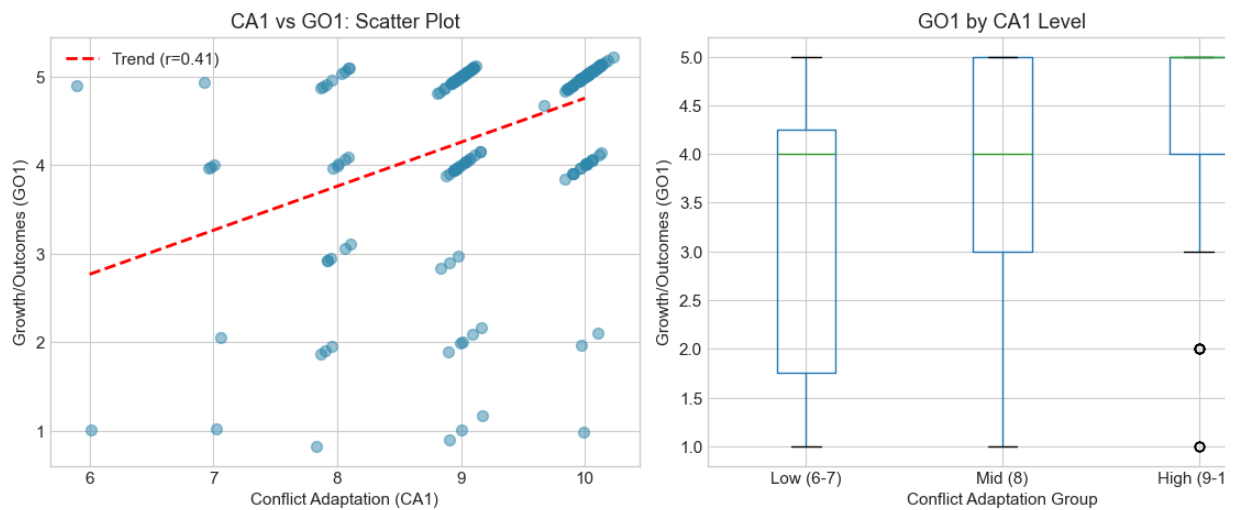
. CA vs Outcome Variables: Detailed Relationship

```
In [6]: # Figure 3: CA1 vs Growth/Outcomes (G01)
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

# Scatter with jitter
jitter = np.random.normal(0, 0.08, len(df))
axes[0].scatter(df['CA1'] + jitter, df['G01'] + jitter, alpha=0.5, c='#2E
z = np.polyfit(df['CA1'].dropna(), df.loc[df['CA1'].notna(), 'G01'], 1)
p = np.poly1d(z)
x_line = np.linspace(df['CA1'].min(), df['CA1'].max(), 100)
axes[0].plot(x_line, p(x_line), 'r--', linewidth=2, label=f'Trend (r={cor
axes[0].set_xlabel('Conflict Adaptation (CA1)')
axes[0].set_ylabel('Growth/Outcomes (G01)')
axes[0].set_title('CA1 vs G01: Scatter Plot')
axes[0].legend()

# Box plot by CA1 groups
df['CA1_group'] = pd.cut(df['CA1'], bins=[5, 7, 8, 10], labels=['Low (6-7
df.boxplot(column='G01', by='CA1_group', ax=axes[1])
axes[1].set_xlabel('Conflict Adaptation Group')
axes[1].set_ylabel('Growth/Outcomes (G01)')
axes[1].set_title('G01 by CA1 Level')
plt.suptitle('')

plt.tight_layout()
plt.show()
```



```
In [7]: # Table: Team Satisfaction by CA1 Level
nps_by_group = df.groupby('CA1_group')['NPS1'].agg(['mean', 'count']).round(2)
nps_by_group.columns = ['Avg Satisfaction', 'N']
print("Team Satisfaction (NPS1) by Conflict Adaptation Level:")
display(nps_by_group)

# Figure 4: Simple bar - Satisfaction by CA1 level
fig, ax = plt.subplots(figsize=(8, 5))
groups = ['Low (6-7)', 'Mid (8)', 'High (9-10)']
nps_means = [nps_by_group.loc[g, 'Avg Satisfaction'] if g in nps_by_group

bars = ax.bar(groups, nps_means, color=['#E94F37', '#F4A261', '#2E86AB'])
ax.set_xlabel('Conflict Adaptation Level')
ax.set_ylabel('Average Team Satisfaction')
ax.set_title('Team Satisfaction Increases with Better Conflict Handling')
ax.set_ylim(0, 5)

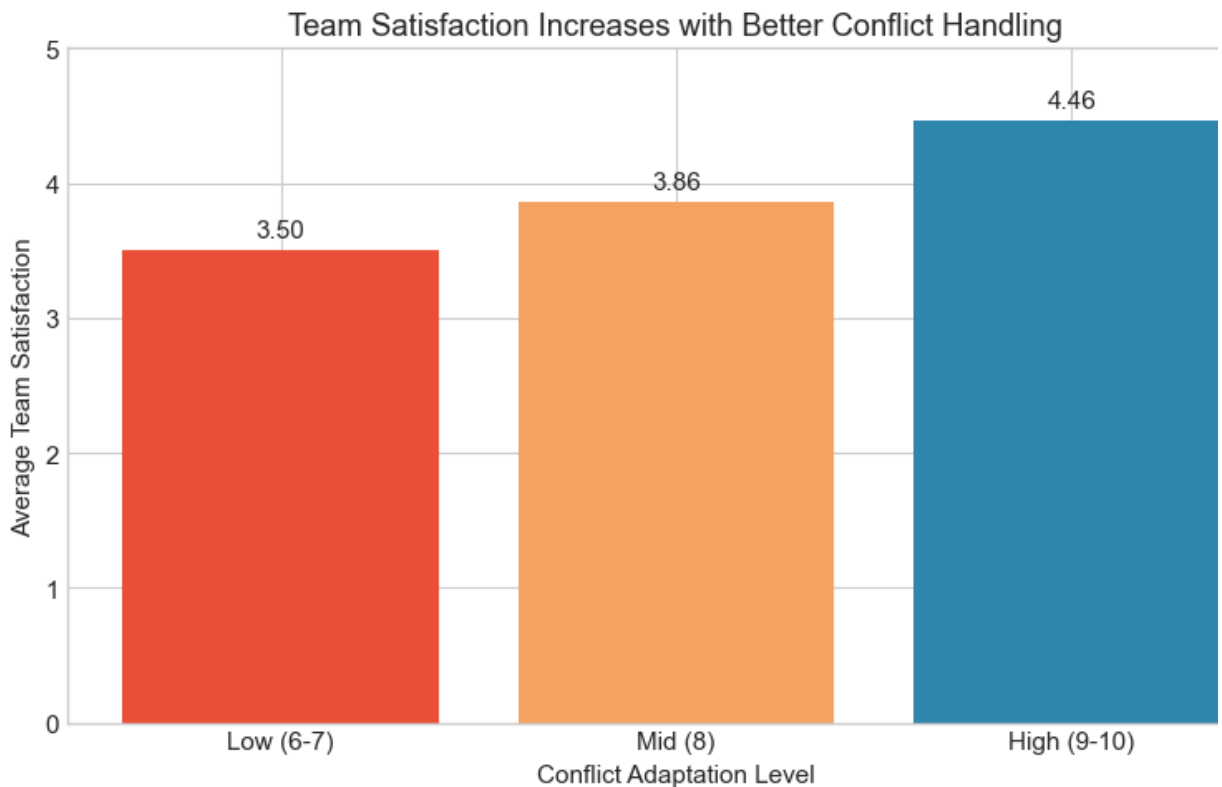
for bar in bars:
    ax.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.1,
            f'{bar.get_height():.2f}', ha='center', fontsize=11)

plt.tight_layout()
plt.show()
```

Team Satisfaction (NPS1) by Conflict Adaptation Level:

/var/folders/vp/5hvn916x1qz0gt_r5m3ynbl80000gn/T/ipykernel_59554/1700612482.p
FutureWarning: The default of observed=False is deprecated and will be change
True in a future version of pandas. Pass observed=False to retain current beh
or observed=True to adopt the future default and silence this warning.
nps_by_group = df.groupby('CA1_group')['NPS1'].agg(['mean', 'count']).round

	Avg Satisfaction	N
CA _group		
Low (-)	.	
Mid ()	.	
High (-)	.	



```
In [8]: # Table: Self-Efficacy by CA1 Level
se_by_group = df.groupby('CA1_group')[['SE1', 'SE2']].mean().round(2)
se_by_group['SE Average'] = ((se_by_group['SE1'] + se_by_group['SE2']) / 2)
print("Self-Efficacy Scores by Conflict Adaptation Level:")
display(se_by_group)

# Figure 5: Simple grouped bar - SE by CA1 level
fig, ax = plt.subplots(figsize=(8, 5))
groups = ['Low (6-7)', 'Mid (8)', 'High (9-10)']
se_means = [se_by_group.loc[g, 'SE Average'] if g in se_by_group.index else 0 for g in groups]

bars = ax.bar(groups, se_means, color=['#E94F37', '#F4A261', '#2E86AB'])
ax.set_xlabel('Conflict Adaptation Level')
ax.set_ylabel('Average Self-Efficacy Score')
ax.set_title('Self-Efficacy Increases with Better Conflict Handling')
ax.set_ylim(0, 5)

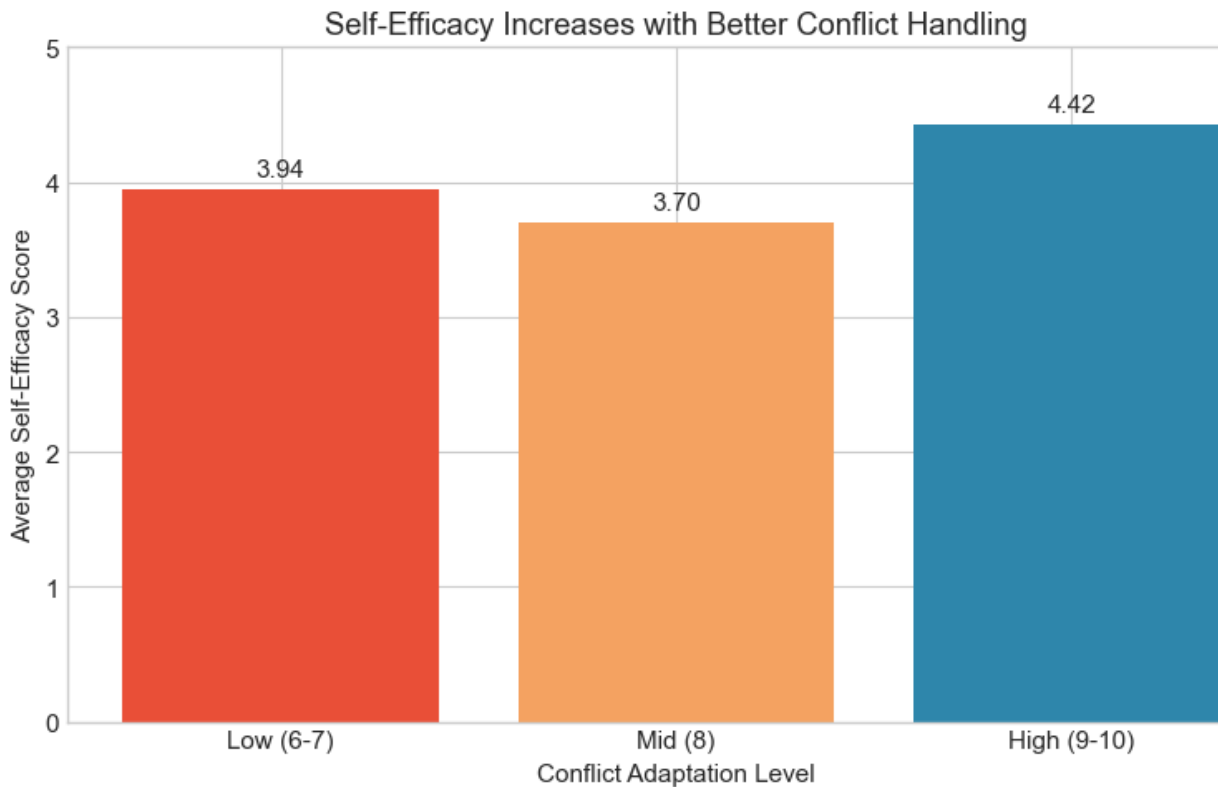
for bar in bars:
    ax.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.1,
            f'{bar.get_height():.2f}', ha='center', fontsize=11)

plt.tight_layout()
plt.show()
```

Self-Efficacy Scores by Conflict Adaptation Level:

```
/var/folders/vp/5hvn916x1qz0gt_r5m3ynbl80000gn/T/ipykernel_59554/281562401.py
FutureWarning: The default of observed=False is deprecated and will be change
True in a future version of pandas. Pass observed=False to retain current beh
or observed=True to adopt the future default and silence this warning.
se_by_group = df.groupby('CA1_group')[['SE1', 'SE2']].mean().round(2)
```

	SE	SE	SE Average
CA _group			
Low (-)	.	.	.
Mid ()	.	.	.
High (-)	.	.	.



```
In [9]: # Table: Relationship Strength by CA1 Level
rls_by_group = df.groupby('CA1_group')['RLS1'].agg(['mean', 'count']).rou
rls_by_group.columns = ['Avg Relationship Strength', 'N']
print("Relationship Strength (RLS1) by Conflict Adaptation Level:")
display(rls_by_group)

# Figure 6: Simple bar - Relationship Strength by CA1 level
fig, ax = plt.subplots(figsize=(8, 5))
groups = ['Low (6-7)', 'Mid (8)', 'High (9-10)']
rls_means = [rls_by_group.loc[g, 'Avg Relationship Strength'] if g in rls

bars = ax.bar(groups, rls_means, color=['#E94F37', '#F4A261', '#2E86AB'])
ax.set_xlabel('Conflict Adaptation Level')
ax.set_ylabel('Average Relationship Strength (1-10)')
ax.set_title('Stronger Relationships in Teams with Better Conflict Handli
ax.set_ylim(0, 10)

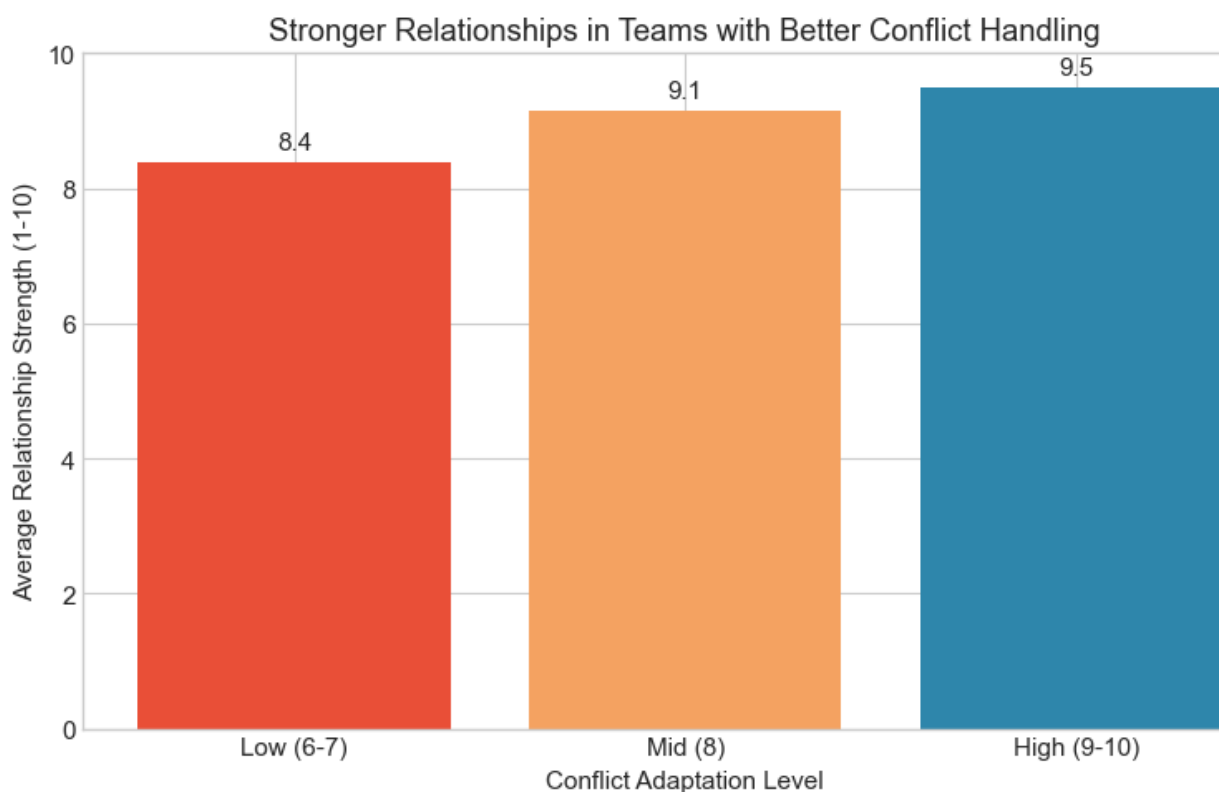
for bar in bars:
    ax.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.2,
            f'{bar.get_height():.1f}', ha='center', fontsize=11)

plt.tight_layout()
plt.show()
```

Relationship Strength (RLS1) by Conflict Adaptation Level:

```
/var/folders/vp/5hvn916x1qz0gt_r5m3ynbl80000gn/T/ipykernel_59554/3062436781.p
FutureWarning: The default of observed=False is deprecated and will be change
True in a future version of pandas. Pass observed=False to retain current beh
or observed=True to adopt the future default and silence this warning.
rls_by_group = df.groupby('CA1_group')['RLS1'].agg(['mean', 'count']).round
```

	Avg Relationship Strength	N
CA _group		
Low (-)	.	
Mid ()	.	
High (-)	.	



. High vs Low Conflict Adaptation: Group Comparison

```
In [10]: # Table 3: High vs Low CA Group Comparison with T-Tests
ca_median = df['CA1'].median()
df['CA1_binary'] = df['CA1'].apply(lambda x: 'High CA' if x >= ca_median

high_ca = df[df['CA1_binary'] == 'High CA']
low_ca = df[df['CA1_binary'] == 'Low CA']

# Calculate t-tests for each outcome variable
test_results = []
for var in ['G01', 'NPS1', 'SE1', 'RLS1']:
    t_stat, p_val = stats.ttest_ind(high_ca[var].dropna(), low_ca[var].dr
    test_results.append({
        'Outcome': var,
        'High CA Mean': high_ca[var].mean().round(2),
        'Low CA Mean': low_ca[var].mean().round(2),
        'Difference': (high_ca[var].mean() - low_ca[var].mean()).round(2)
        't-statistic': round(t_stat, 2),
```

```

        'p-value': f'{p_val:.4f}',
        'Significant': 'Yes' if p_val < 0.05 else 'No'
    })

comparison_stats = pd.DataFrame(test_results)
print(f"Table 3: High CA (≥{ca_median:.0f}, n={len(high_ca)}) vs Low CA (<{ca_median:.0f}, n={len(low_ca)})")
print(comparison_stats)

```

Table 3: High CA (≥10, n=112) vs Low CA (<10, n=98)

Outcome	High CA Mean	Low CA Mean	Difference	t-statistic	p-value	Significant
GO	Yes
NPS	Yes
SE	Yes
RLS	No

```

In [11]: # Figure 7: High vs Low CA Comparison Bar Chart
fig, ax = plt.subplots(figsize=(10, 6))

outcome_labels = ['Growth (G01)', 'Satisfaction (NPS1)', 'Self-Efficacy (SE1)', 'Resilience (RLS1)']
high_ca_means = [
    df[df['CA1_binary'] == 'High CA']['G01'].mean(),
    df[df['CA1_binary'] == 'High CA']['NPS1'].mean(),
    df[df['CA1_binary'] == 'High CA']['SE1'].mean(),
    df[df['CA1_binary'] == 'High CA']['RLS1'].mean() / 2 # Scale to 1-5
]
low_ca_means = [
    df[df['CA1_binary'] == 'Low CA']['G01'].mean(),
    df[df['CA1_binary'] == 'Low CA']['NPS1'].mean(),
    df[df['CA1_binary'] == 'Low CA']['SE1'].mean(),
    df[df['CA1_binary'] == 'Low CA']['RLS1'].mean() / 2 # Scale to 1-5
]

x = np.arange(len(outcome_labels))
width = 0.35

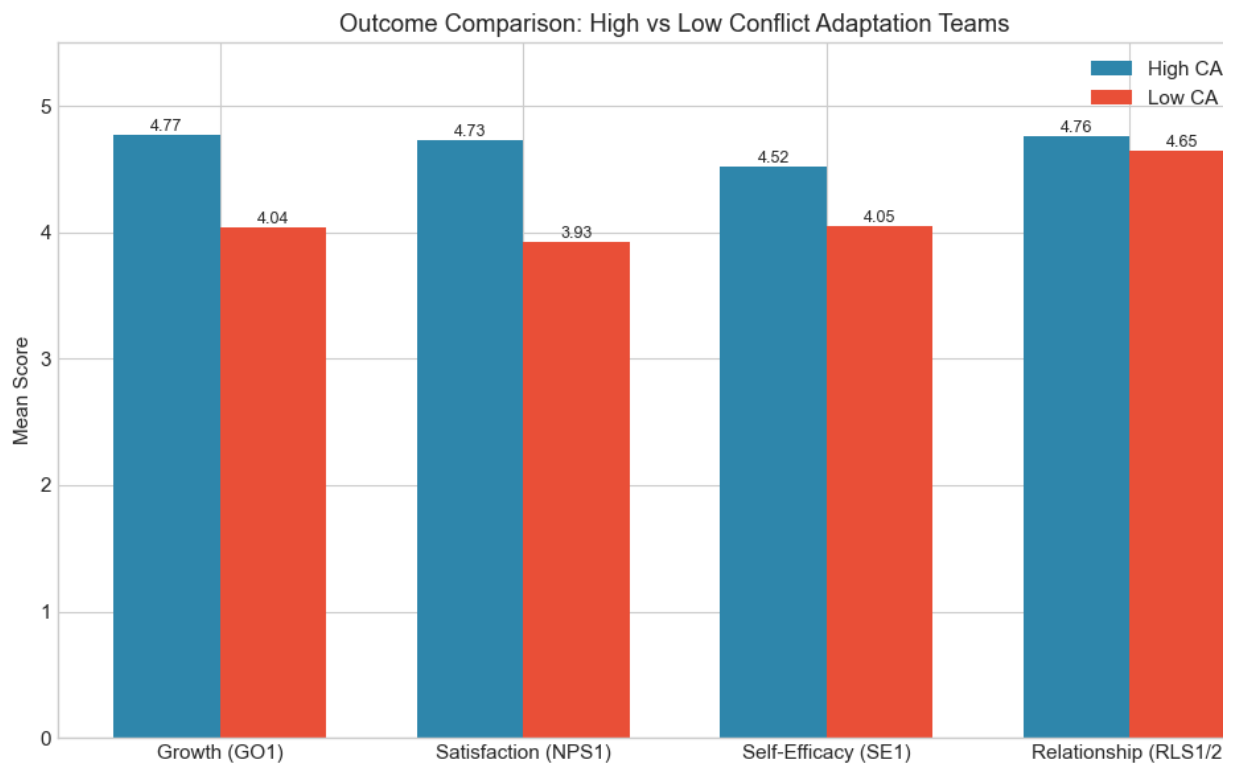
bars1 = ax.bar(x - width/2, high_ca_means, width, label=f'High CA (≥{ca_median:.0f})')
bars2 = ax.bar(x + width/2, low_ca_means, width, label=f'Low CA (<{ca_median:.0f})')

ax.set_ylabel('Mean Score')
ax.set_title('Outcome Comparison: High vs Low Conflict Adaptation Teams')
ax.set_xticks(x)
ax.set_xticklabels(outcome_labels)
ax.legend()
ax.set_ylim(0, 5.5)

# Add value labels
for bar in bars1:
    ax.annotate(f'{bar.get_height():.2f}', xy=(bar.get_x() + bar.get_width()/2, bar.get_height()),
                ha='center', va='bottom', fontsize=9)
for bar in bars2:
    ax.annotate(f'{bar.get_height():.2f}', xy=(bar.get_x() + bar.get_width()/2, bar.get_height()),
                ha='center', va='bottom', fontsize=9)

plt.tight_layout()
plt.show()

```



. Conflict Adaptation by Section

```
In [12]: # Table: CA1 and outcomes by section
section_stats = df.groupby('Section').agg({
    'CA1': ['mean', 'std', 'count'],
    'GO1': 'mean',
    'NPS1': 'mean',
    'RLS1': 'mean'
}).round(2)
section_stats.columns = ['CA1 Mean', 'CA1 Std', 'N', 'GO1 Mean', 'NPS1 Me
section_stats
```

```
Out[12]:
```

	CA	Mean	CA	Std	N	GO	Mean	NPS	Mean	RLS	Mean
Section											

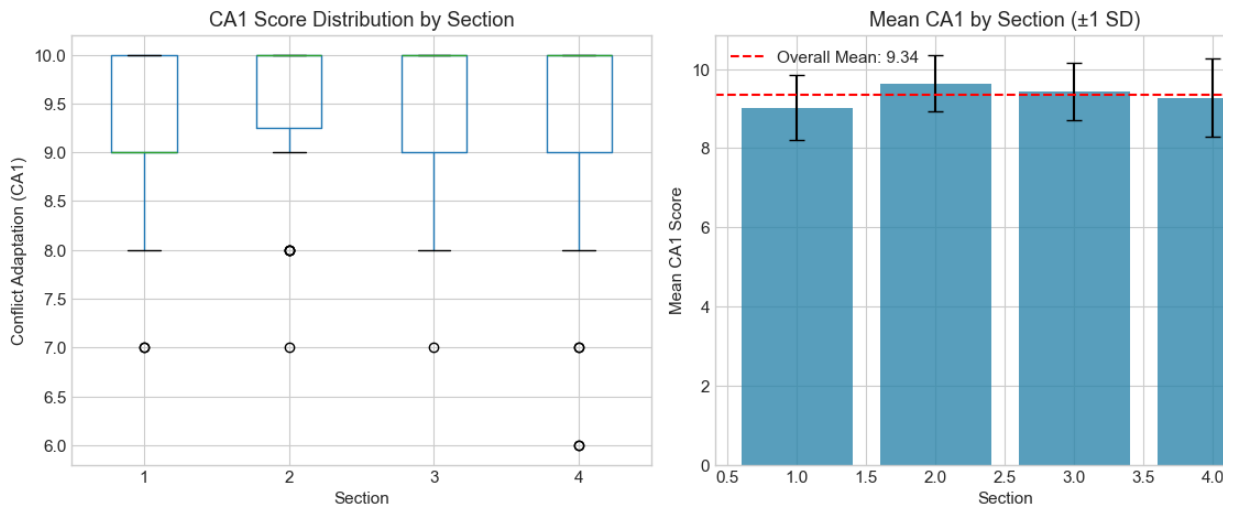

```
In [13]: # Figure 8: CA1 Distribution by Section
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

# Box plot
df.boxplot(column='CA1', by='Section', ax=axes[0])
axes[0].set_xlabel('Section')
axes[0].set_ylabel('Conflict Adaptation (CA1)')
axes[0].set_title('CA1 Score Distribution by Section')
plt.suptitle('')
```

```
# Bar chart of means with error bars
section_means = df.groupby('Section')['CA1'].mean()
section_stds = df.groupby('Section')['CA1'].std()
sections = section_means.index

axes[1].bar(sections, section_means, yerr=section_stds, capsize=5, color=
axes[1].set_xlabel('Section')
axes[1].set_ylabel('Mean CA1 Score')
axes[1].set_title('Mean CA1 by Section (±1 SD)')
axes[1].axhline(y=df['CA1'].mean(), color='red', linestyle='--', label=f'
axes[1].legend()

plt.tight_layout()
plt.show()
```



Summary of Key Findings

Key observations from Conflict Adaptation (CA) analysis:

- . **Distribution:** Most teams report moderate-to-high conflict adaptation scores (median around - /)
- . **Correlations:** CA shows positive correlations with all outcome variables - teams that ha conflict well tend to report better outcomes
- . **Strongest relationships:**
 - CA correlates most strongly with RLS (relationship strength) and NPS (tean satisfaction)
 - Constructive conflict handling appears linked to stronger interpersonal bonds
- . **High vs Low CA:** Teams with above-median CA scores show consistently higher mean: all outcome metrics
- . **Section differences:** Some variation exists between sections in conflict adaptation patterns