Exploratory Data Analysis — Duflo, Dupas & Kremer ( )

Goal: Understand the data, find correlations, identify promising causal questions.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import warnings
warnings.filterwarnings('ignore')

sns.set_style('whitegrid')
plt.rcParams['figure.figsize'] = (12, 6)
plt.rcParams['figure.dpi'] = 100

BASE = '112446-V1/data'
st = pd.read_stata(f'{BASE}/student_test_data.dta')
sp = pd.read_stata(f'{BASE}/student_pres_data.dta')
tp = pd.read_stata(f'{BASE}/teacher_pres_data.dta')

print(f'student_test: {st.shape}')
print(f'student_pres: {sp.shape}')
print(f'teacher_pres: {tp.shape}')
```

```
student_test: (7022, 106)
student_pres: (97100, 10)
teacher_pres: (2484, 12)
```

. Basic Structure & Missingness

In [2]:
```
# Treatment assignment counts
print('=== Treatment Assignment ===')
print('Tracking:', st['tracking'].value_counts().to_dict())
print('ETP teacher:', st['etpteacher'].value_counts().to_dict())
print('SBM:', st['sbm'].value_counts().to_dict())
print(f'\nSchools: {st["schoolid"].nunique()}')
print(f'Students: {len(st)}')
print(f'Districts: {st["district"].unique()}')
```

```
=== Treatment Assignment ===
Tracking: {1.0: 3613, 0.0: 3409}
ETP teacher: {0.0: 3537, 1.0: 3485}
SBM: {1.0: 3570, 0.0: 3452}

Schools: 121
Students: 7022
Districts: <StringArray>
['BUNGOMA', 'BUTERE/M']
Length: 2, dtype: str
```

In [3]:
```
# Missingness for key variables
key_vars = ['tracking', 'sbm', 'etpteacher', 'girl', 'agetest', 'std_mark
```

```
             'percentile', 'totalscore', 'litscore', 'mathscoreraw',
             'r2_totalscore', 'r2_litscore', 'r2_mathscoreraw', 'attrition
miss = st[key_vars].isnull().sum().to_frame('n_missing')
miss['pct_missing'] = (miss['n_missing'] / len(st) * 100).round(1)
print(miss.to_string())
```

```
                 n_missing  pct_missing
tracking                 0          0.0
sbm                      0          0.0
etpteacher               0          0.0
girl                    27          0.4
agetest                522          7.4
std_mark               758         10.8
percentile             591          8.4
totalscore            1227         17.5
litscore              1226         17.5
mathscoreraw          1226         17.5
r2_totalscore         1533         21.8
r2_litscore           1527         21.7
r2_mathscoreraw       1533         21.8
attrition                0          0.0
```

In [4]:
```
# Attrition by treatment
print('=== Attrition rates ===')
print(st.groupby('tracking')['attrition'].mean().round(3))
print()
print('Attrition by tracking x bottomhalf:')
print(st.groupby(['tracking', 'bottomhalf'])['attrition'].mean().round(3)
```

```
=== Attrition rates ===
tracking
0.0    0.174
1.0    0.175
Name: attrition, dtype: float32

Attrition by tracking x bottomhalf:
bottomhalf    0.0    1.0
tracking
0.0         0.162  0.189
1.0         0.163  0.186
```

. Summary Statistics by Treatment Group

In [5]:
```
# Baseline balance check
baseline_vars = ['girl', 'agetest', 'std_mark', 'percentile']

balance = []
for v in baseline_vars:
    ctrl = st.loc[st['tracking'] == 0, v].dropna()
    treat = st.loc[st['tracking'] == 1, v].dropna()
    t_stat, p_val = stats.ttest_ind(ctrl, treat)
    balance.append({
        'variable': v,
        'control_mean': ctrl.mean(),
        'treat_mean': treat.mean(),
        'diff': treat.mean() - ctrl.mean(),
        't_stat': t_stat,
        'p_value': p_val
    })
```

```
balance_df = pd.DataFrame(balance).round(4)
print('=== Baseline Balance: Control vs Tracking ===')
print(balance_df.to_string(index=False))
```

```
=== Baseline Balance: Control vs Tracking ===
  variable  control_mean  treat_mean    diff  t_stat  p_value
      girl        0.4919    0.498600  0.0067 -0.5578   0.5770
   agetest        9.1811    9.362700  0.1815 -4.9835   0.0000
  std_mark        0.0227    0.001200 -0.0215  0.8514   0.3946
percentile       51.2542   50.441299 -0.8130  1.1301   0.2585
```

In [6]:

```
# Outcome summary stats by treatment
outcomes = ['totalscore', 'litscore', 'mathscoreraw',
            'r2_totalscore', 'r2_litscore', 'r2_mathscoreraw']

summary = st.groupby('tracking')[outcomes].agg(['mean', 'std', 'count']).
print('=== Outcome Means by Tracking Status ===')
print('(r2_ = long-run follow-up scores)\n')
print(summary.T)
```

```
=== Outcome Means by Tracking Status ===
(r2_ = long-run follow-up scores)

tracking                     0.0          1.0
totalscore      mean      12.256    13.514000
                std        9.010     9.165000
                count   2814.000  2981.000000
litscore        mean       4.999     5.677000
                std        5.472     5.606000
                count   2815.000  2981.000000
mathscoreraw    mean       7.256     7.837000
                std        4.647     4.617000
                count   2814.000  2982.000000
r2_totalscore   mean      18.914    20.759001
                std       11.254    11.297000
                count   2661.000  2828.000000
r2_litscore     mean       8.507     9.671000
                std        7.217     7.338000
                count   2663.000  2832.000000
r2_mathscoreraw mean      10.403    11.078000
                std        5.262     5.128000
                count   2661.000  2828.000000
```

. Standardised Test Scores — Treatment Effects (Rav

In [7]:

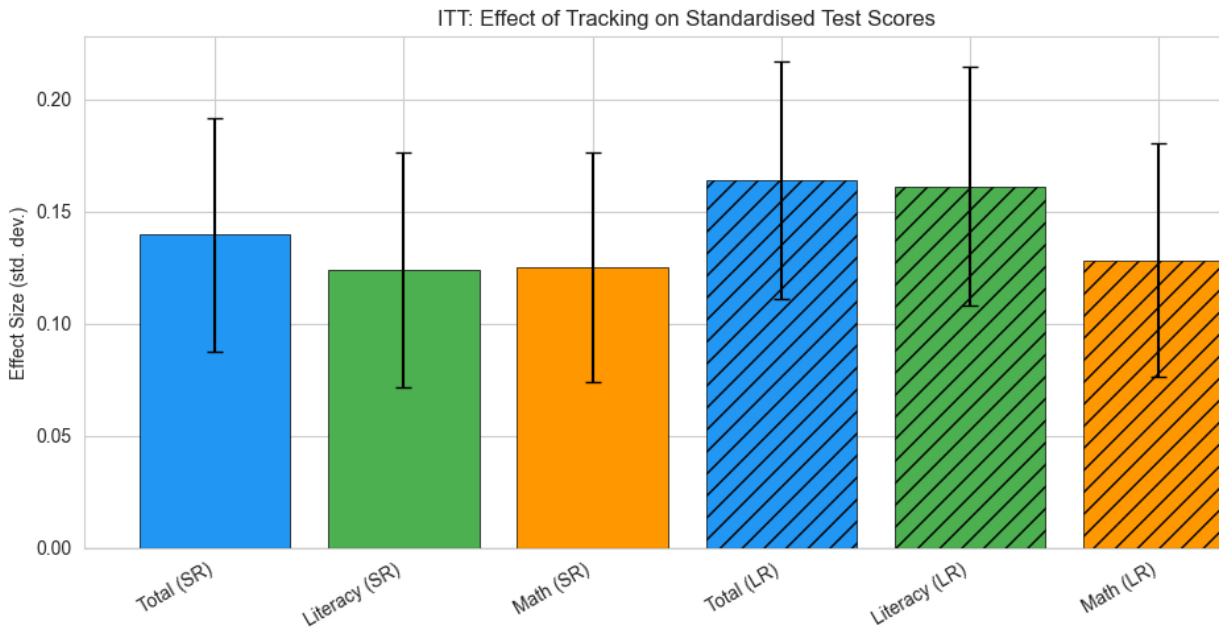
```
# Standardise scores relative to control group (as in the original paper)
for col in ['totalscore', 'litscore', 'mathscoreraw',
            'r2_totalscore', 'r2_litscore', 'r2_mathscoreraw']:
    ctrl = st.loc[st['tracking'] == 0, col]
    st[f'z_{col}'] = (st[col] - ctrl.mean()) / ctrl.std()

# Simple difference in standardised means
z_outcomes = [f'z_{c}' for c in outcomes]
diff_table = []
for v in z_outcomes:
    c = st.loc[st['tracking'] == 0, v].dropna()
    t = st.loc[st['tracking'] == 1, v].dropna()
    tstat, pval = stats.ttest_ind(c, t)
```

```
    diff_table.append({'outcome': v, 'ctrl_mean': c.mean(), 'treat_mean':
                        'ATE': t.mean() - c.mean(), 't_stat': tstat, 'p_va

print('=== ITT (simple difference in standardised means) ===')
print(pd.DataFrame(diff_table).round(4).to_string(index=False))
```

```
=== ITT (simple difference in standardised means) ===
          outcome  ctrl_mean  treat_mean    ATE  t_stat  p_value
     z_totalscore       -0.0      0.1396 0.1396 -5.2660      0.0
       z_litscore        0.0      0.1239 0.1239 -4.6571      0.0
   z_mathscoreraw       -0.0      0.1251 0.1251 -4.7744      0.0
  z_r2_totalscore       -0.0      0.1640 0.1640 -6.0595      0.0
    z_r2_litscore        0.0      0.1613 0.1613 -5.9233      0.0
z_r2_mathscoreraw        0.0      0.1283 0.1283 -4.8118      0.0
```

In [8]:

```
# Bar chart: ITT effect sizes with confidence intervals
fig, ax = plt.subplots(figsize=(10, 5))
df_eff = pd.DataFrame(diff_table)
labels = ['Total (SR)', 'Literacy (SR)', 'Math (SR)',
          'Total (LR)', 'Literacy (LR)', 'Math (LR)']
colors = ['#2196F3', '#4CAF50', '#FF9800'] * 2
hatches = [''] * 3 + ['//'] * 3

bars = ax.bar(range(len(labels)), df_eff['ATE'], color=colors, edgecolor=
for bar, h in zip(bars, hatches):
    bar.set_hatch(h)

for i, row in df_eff.iterrows():
    c = st.loc[st['tracking'] == 0, row['outcome']].dropna()
    t = st.loc[st['tracking'] == 1, row['outcome']].dropna()
    se = np.sqrt(c.var()/len(c) + t.var()/len(t))
    ax.errorbar(i, row['ATE'], yerr=1.96*se, color='black', capsize=4, li

ax.set_xticks(range(len(labels)))
ax.set_xticklabels(labels, rotation=30, ha='right')
ax.set_ylabel('Effect Size (std. dev.)')
ax.set_title('ITT: Effect of Tracking on Standardised Test Scores')
ax.axhline(0, color='black', linewidth=0.8)
plt.tight_layout()
plt.show()
```



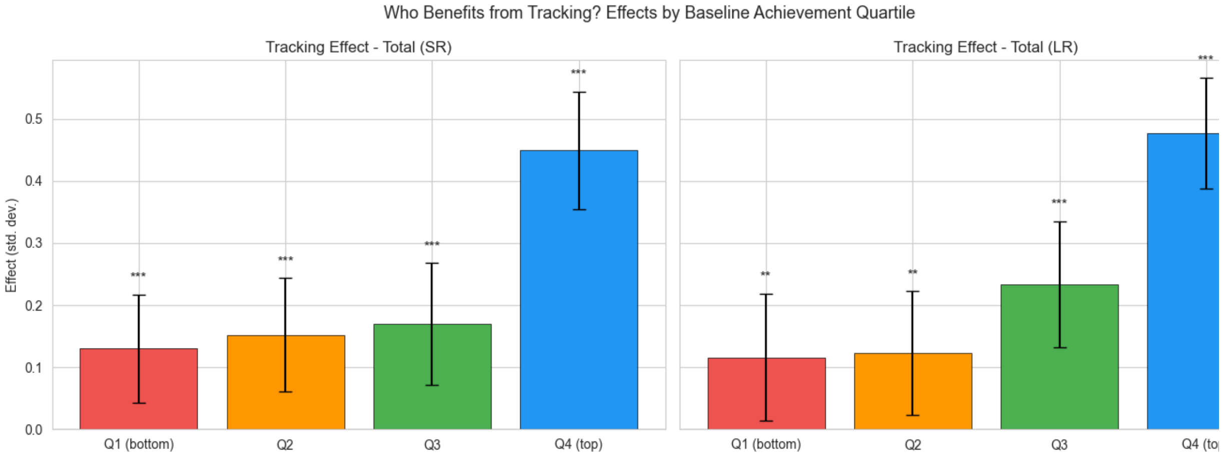
. Heterogeneous Effects by Baseline Achievement

In [9]:
```
# Effect of tracking by baseline achievement quartile
conditions = [
    st['bottomquarter'] == 1,
    st['secondquarter'] == 1,
    st['thirdquarter'] == 1,
    st['topquarter'] == 1
]
choices = ['Q1 (bottom)', 'Q2', 'Q3', 'Q4 (top)']
st['quartile'] = np.select(conditions, choices, default='')
st.loc[st['quartile'] == '', 'quartile'] = np.nan

het = []
for q in ['Q1 (bottom)', 'Q2', 'Q3', 'Q4 (top)']:
    sub = st[st['quartile'] == q]
    for outcome_label, outcome_col in [('Total (SR)', 'z_totalscore'),
                                        ('Total (LR)', 'z_r2_totalscore')
        c = sub.loc[sub['tracking'] == 0, outcome_col].dropna()
        t = sub.loc[sub['tracking'] == 1, outcome_col].dropna()
        if len(c) > 1 and len(t) > 1:
            diff = t.mean() - c.mean()
            se = np.sqrt(c.var()/len(c) + t.var()/len(t))
            tstat, pval = stats.ttest_ind(c, t)
            het.append({'quartile': q, 'outcome': outcome_label,
                        'ATE': diff, 'SE': se, 'p_value': pval, 'N_ctrl':

het_df = pd.DataFrame(het)
print('=== Heterogeneous ITT by Baseline Quartile ===')
print(het_df.round(4).to_string(index=False))
```

```
=== Heterogeneous ITT by Baseline Quartile ===
   quartile    outcome    ATE     SE  p_value  N_ctrl  N_treat
Q1 (bottom) Total (SR) 0.1300 0.0444   0.0036     520      699
Q1 (bottom) Total (LR) 0.1157 0.0522   0.0272     503      663
         Q2 Total (SR) 0.1519 0.0466   0.0012     579      757
         Q2 Total (LR) 0.1229 0.0507   0.0153     538      719
         Q3 Total (SR) 0.1697 0.0499   0.0007     606      759
         Q3 Total (LR) 0.2337 0.0516   0.0000     565      728
   Q4 (top) Total (SR) 0.4493 0.0483   0.0000    1109      766
   Q4 (top) Total (LR) 0.4769 0.0453   0.0000    1055      718
```

In [10]:
```
fig, axes = plt.subplots(1, 2, figsize=(14, 5), sharey=True)

for ax, outcome in zip(axes, ['Total (SR)', 'Total (LR)']):
    sub = het_df[het_df['outcome'] == outcome]
    x = range(len(sub))
    ax.bar(x, sub['ATE'], yerr=1.96*sub['SE'], capsize=5,
           color=['#EF5350', '#FF9800', '#4CAF50', '#2196F3'], edgecolor=
    ax.set_xticks(x)
    ax.set_xticklabels(sub['quartile'])
    ax.axhline(0, color='black', linewidth=0.8)
    ax.set_title(f'Tracking Effect — {outcome}')
    ax.set_ylabel('Effect (std. dev.)' if ax == axes[0] else '')
    for i, (_, row) in enumerate(sub.iterrows()):
        star = '***' if row['p_value'] < 0.01 else '**' if row['p_value']
        ax.text(i, row['ATE'] + 1.96*row['SE'] + 0.02, star, ha='center',
```

```
fig.suptitle('Who Benefits from Tracking? Effects by Baseline Achievement
plt.tight_layout()
plt.show()
```



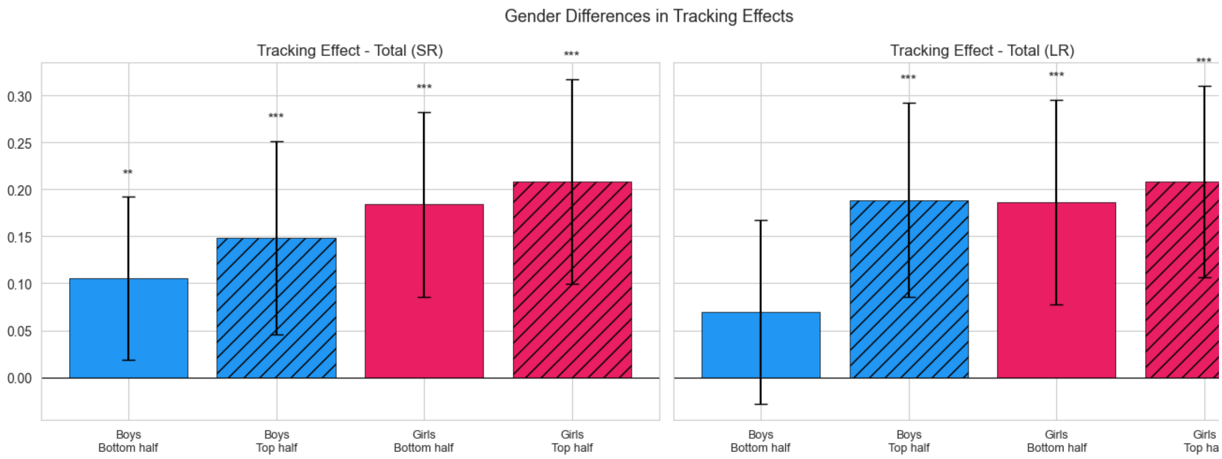
. Heterogeneous Effects by Gender

In [11]:
```
gender_het = []
for gender_label, gender_val in [('Boys', 0.0), ('Girls', 1.0)]:
    for half_label, half_val in [('Bottom half', 1.0), ('Top half', 0.0)]
        for out_label, out_col in [('Total (SR)', 'z_totalscore'), ('Tota
            sub = st[(st['girl'] == gender_val) & (st['bottomhalf'] == ha
            c = sub.loc[sub['tracking'] == 0, out_col].dropna()
            t = sub.loc[sub['tracking'] == 1, out_col].dropna()
            if len(c) > 1 and len(t) > 1:
                diff = t.mean() - c.mean()
                se = np.sqrt(c.var()/len(c) + t.var()/len(t))
                _, pval = stats.ttest_ind(c, t)
                gender_het.append({'gender': gender_label, 'half': half_l
                                   'outcome': out_label, 'ATE': diff, 'SE

gdf = pd.DataFrame(gender_het)
print('=== Gender x Achievement Half x Tracking ===')
print(gdf.round(4).to_string(index=False))
```

```
=== Gender x Achievement Half x Tracking ===
gender        half    outcome    ATE     SE  p_value
  Boys Bottom half Total (SR) 0.1054 0.0442   0.0176
  Boys Bottom half Total (LR) 0.0693 0.0499   0.1631
  Boys    Top half Total (SR) 0.1484 0.0526   0.0052
  Boys    Top half Total (LR) 0.1887 0.0529   0.0004
 Girls Bottom half Total (SR) 0.1840 0.0502   0.0003
 Girls Bottom half Total (LR) 0.1865 0.0556   0.0009
 Girls    Top half Total (SR) 0.2085 0.0555   0.0002
 Girls    Top half Total (LR) 0.2083 0.0520   0.0001
```

In [12]:
```
fig, axes = plt.subplots(1, 2, figsize=(14, 5), sharey=True)

for ax, outcome in zip(axes, ['Total (SR)', 'Total (LR)']):
    sub = gdf[gdf['outcome'] == outcome]
    x = np.arange(len(sub))
    colors = ['#2196F3', '#2196F3', '#E91E63', '#E91E63']
    hatches = ['', '//', '', '//']
    bars = ax.bar(x, sub['ATE'], yerr=1.96*sub['SE'], capsize=5,
                  color=colors, edgecolor='black', linewidth=0.5)
```

```
    for bar, h in zip(bars, hatches):
        bar.set_hatch(h)
    ax.set_xticks(x)
    labels = [f"{r['gender']}\n{r['half']}" for _, r in sub.iterrows()]
    ax.set_xticklabels(labels, fontsize=9)
    ax.axhline(0, color='black', linewidth=0.8)
    ax.set_title(f'Tracking Effect - {outcome}')
    for i, (_, row) in enumerate(sub.iterrows()):
        star = '***' if row['p_value'] < 0.01 else '**' if row['p_value']
        offset = 1.96*row['SE'] + 0.02 if row['ATE'] >= 0 else -(1.96*row
        ax.text(i, row['ATE'] + offset, star, ha='center', fontsize=11)

fig.suptitle('Gender Differences in Tracking Effects', fontsize=13)
plt.tight_layout()
plt.show()
```



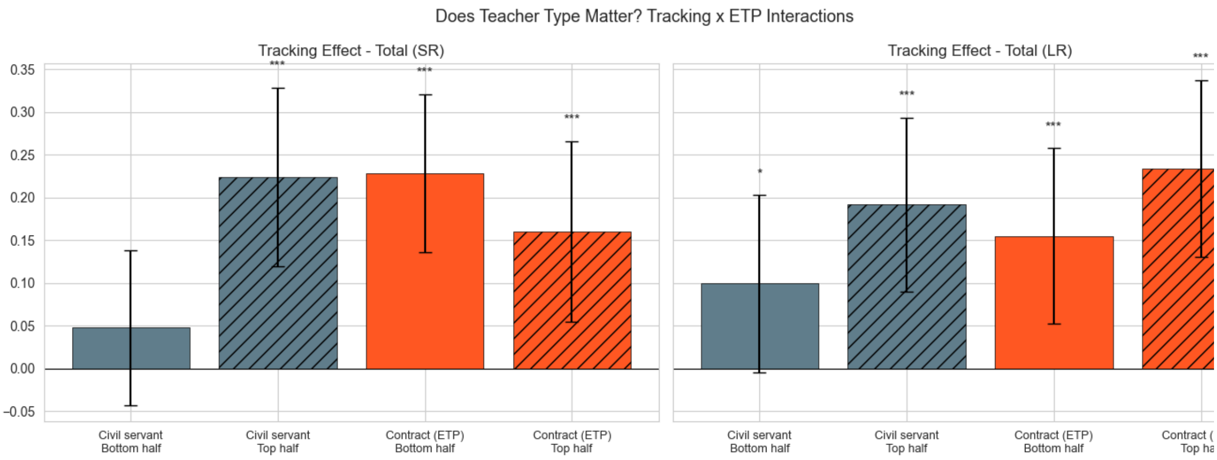
. Contract Teacher (ETP) Interactions

In [13]:
```
etp_het = []
for etp_label, etp_val in [('Civil servant', 0.0), ('Contract (ETP)', 1.0
    for half_label, half_val in [('Bottom half', 1.0), ('Top half', 0.0)]
        for out_label, out_col in [('Total (SR)', 'z_totalscore'), ('Tota
            sub = st[(st['etpteacher'] == etp_val) & (st['bottomhalf'] ==
            c = sub.loc[sub['tracking'] == 0, out_col].dropna()
            t = sub.loc[sub['tracking'] == 1, out_col].dropna()
            if len(c) > 1 and len(t) > 1:
                diff = t.mean() - c.mean()
                se = np.sqrt(c.var()/len(c) + t.var()/len(t))
                _, pval = stats.ttest_ind(c, t)
                etp_het.append({'teacher': etp_label, 'half': half_label,
                                'outcome': out_label, 'ATE': diff, 'SE':

etp_df = pd.DataFrame(etp_het)
print('=== Tracking Effect by Teacher Type x Achievement Half ===')
print(etp_df.round(4).to_string(index=False))
```

```
=== Tracking Effect by Teacher Type x Achievement Half ===
        teacher        half     outcome    ATE     SE  p_value
 Civil servant Bottom half Total (SR) 0.0476 0.0462   0.2973
 Civil servant Bottom half Total (LR) 0.0993 0.0528   0.0607
 Civil servant    Top half Total (SR) 0.2242 0.0533   0.0000
 Civil servant    Top half Total (LR) 0.1918 0.0518   0.0002
Contract (ETP) Bottom half Total (SR) 0.2288 0.0472   0.0000
Contract (ETP) Bottom half Total (LR) 0.1551 0.0524   0.0031
Contract (ETP)    Top half Total (SR) 0.1603 0.0540   0.0031
Contract (ETP)    Top half Total (LR) 0.2341 0.0528   0.0000
```

In [14]:

```python
fig, axes = plt.subplots(1, 2, figsize=(14, 5), sharey=True)

for ax, outcome in zip(axes, ['Total (SR)', 'Total (LR)']):
    sub = etp_df[etp_df['outcome'] == outcome]
    x = np.arange(len(sub))
    colors = ['#607D8B', '#607D8B', '#FF5722', '#FF5722']
    hatches = ['', '//', '', '//']
    bars = ax.bar(x, sub['ATE'], yerr=1.96*sub['SE'], capsize=5,
                  color=colors, edgecolor='black', linewidth=0.5)
    for bar, h in zip(bars, hatches):
        bar.set_hatch(h)
    ax.set_xticks(x)
    labels = [f"{r['teacher']}\n{r['half']}" for _, r in sub.iterrows()]
    ax.set_xticklabels(labels, fontsize=9)
    ax.axhline(0, color='black', linewidth=0.8)
    ax.set_title(f'Tracking Effect - {outcome}')
    for i, (_, row) in enumerate(sub.iterrows()):
        star = '***' if row['p_value'] < 0.01 else '**' if row['p_value']
        offset = 1.96*row['SE'] + 0.02 if row['ATE'] >= 0 else -(1.96*row
        ax.text(i, row['ATE'] + offset, star, ha='center', fontsize=11)

fig.suptitle('Does Teacher Type Matter? Tracking x ETP Interactions', fon
plt.tight_layout()
plt.show()
```



. Peer Composition Effects (Non-tracking Schools)

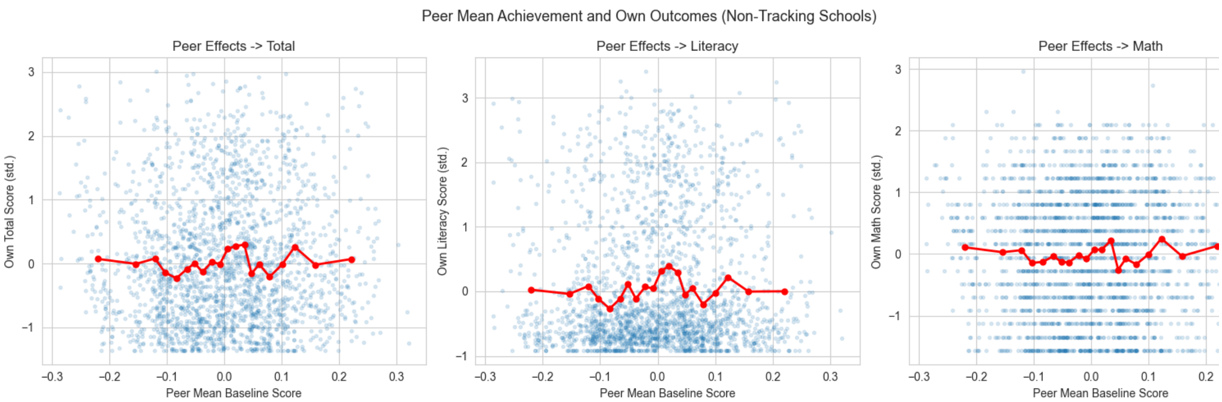
In [15]:

```python
# In non-tracking schools, students are randomly assigned to sections
nt = st[st['tracking'] == 0].copy()

peer_vars = ['rMEANstream_std_baselinemark', 'rSDstream_std_baselinemark'
             'MEANstream_std_mark', 'SDstream_std_mark']
outcome_vars = ['z_totalscore', 'z_litscore', 'z_mathscoreraw']
```

```
print('=== Correlations: Peer Composition -> Outcomes (Non-tracking schoo
corr_rows = []
for p in peer_vars:
    for o in outcome_vars:
        sub = nt[[p, o]].dropna()
        if len(sub) > 2:
            r, pval = stats.pearsonr(sub[p], sub[o])
            corr_rows.append({'peer_var': p, 'outcome': o, 'corr': r, 'p_

print(pd.DataFrame(corr_rows).round(4).to_string(index=False))
```

```
=== Correlations: Peer Composition -> Outcomes (Non-tracking schools) ===
                      peer_var         outcome    corr  p_value    N
rMEANstream_std_baselinemark    z_totalscore  0.0197   0.3542 2210
rMEANstream_std_baselinemark      z_litscore  0.0225   0.2906 2211
rMEANstream_std_baselinemark z_mathscoreraw  0.0114   0.5910 2210
  rSDstream_std_baselinemark    z_totalscore  0.0006   0.9786 2210
  rSDstream_std_baselinemark      z_litscore  0.0150   0.4802 2211
  rSDstream_std_baselinemark z_mathscoreraw -0.0171   0.4227 2210
         MEANstream_std_mark    z_totalscore  0.0888   0.0000 2210
         MEANstream_std_mark      z_litscore  0.0759   0.0004 2211
         MEANstream_std_mark z_mathscoreraw  0.0827   0.0001 2210
           SDstream_std_mark    z_totalscore  0.0087   0.6844 2210
           SDstream_std_mark      z_litscore  0.0266   0.2117 2211
           SDstream_std_mark z_mathscoreraw -0.0151   0.4789 2210
```

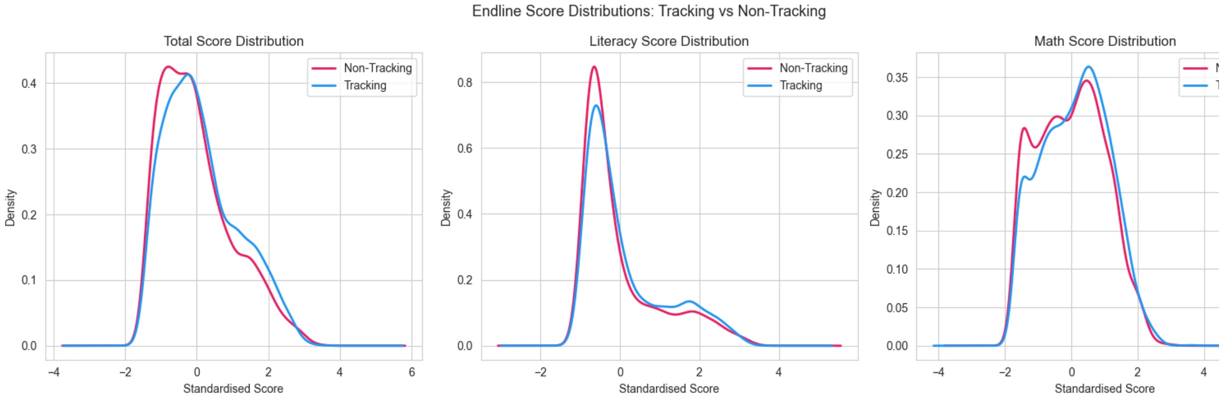
In [16]:
```
fig, axes = plt.subplots(1, 3, figsize=(16, 5))
for ax, (out_col, label) in zip(axes, [('z_totalscore', 'Total'), ('z_lit
    sub = nt[['rMEANstream_std_baselinemark', out_col]].dropna()
    ax.scatter(sub['rMEANstream_std_baselinemark'], sub[out_col], alpha=0
    sub['bin'] = pd.qcut(sub['rMEANstream_std_baselinemark'], 20, duplica
    binned = sub.groupby('bin', observed=True)[out_col].mean()
    bin_x = sub.groupby('bin', observed=True)['rMEANstream_std_baselinema
    ax.plot(bin_x, binned, 'r-o', markersize=5, linewidth=2)
    ax.set_xlabel('Peer Mean Baseline Score')
    ax.set_ylabel(f'Own {label} Score (std.)')
    ax.set_title(f'Peer Effects -> {label}')

fig.suptitle('Peer Mean Achievement and Own Outcomes (Non-Tracking School
plt.tight_layout()
plt.show()
```



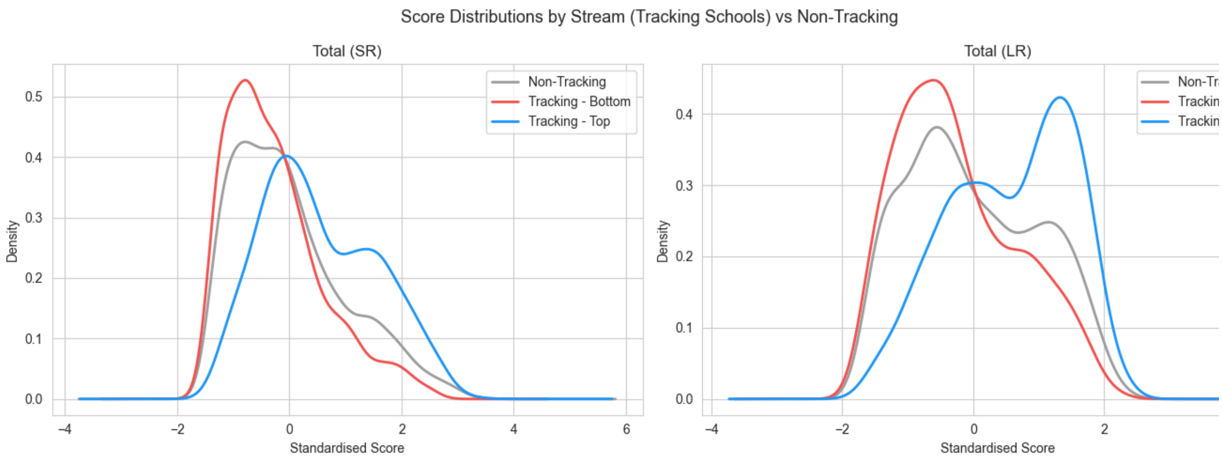
. Score Distributions: Tracking vs Non-Tracking

In [17]:

```
fig, axes = plt.subplots(1, 3, figsize=(16, 5))

for ax, (col, label) in zip(axes, [('z_totalscore', 'Total'), ('z_litscor
    for track_val, track_label, color in [(0, 'Non-Tracking', '#E91E63'),
        data = st.loc[(st['tracking'] == track_val), col].dropna()
        data.plot.kde(ax=ax, label=track_label, color=color, linewidth=2)
    ax.set_title(f'{label} Score Distribution')
    ax.set_xlabel('Standardised Score')
    ax.legend()

fig.suptitle('Endline Score Distributions: Tracking vs Non-Tracking', fon
plt.tight_layout()
plt.show()
```



In [18]:
```
fig, axes = plt.subplots(1, 2, figsize=(14, 5))

for ax, (col, label) in zip(axes, [('z_totalscore', 'Total (SR)'), ('z_r2
    for grp, lbl, color in [
        ((st['tracking'] == 0), 'Non-Tracking', '#9E9E9E'),
        ((st['tracking'] == 1) & (st['bottomhalf'] == 1), 'Tracking - Bot
        ((st['tracking'] == 1) & (st['bottomhalf'] == 0), 'Tracking - Top
    ]:
        data = st.loc[grp, col].dropna()
        if len(data) > 10:
            data.plot.kde(ax=ax, label=lbl, color=color, linewidth=2)
    ax.set_title(label)
    ax.set_xlabel('Standardised Score')
    ax.legend()

fig.suptitle('Score Distributions by Stream (Tracking Schools) vs Non-Tra
plt.tight_layout()
plt.show()
```



. Teacher & Student Attendance

In [19]:
```
print('=== Teacher Presence Rates ===')
print(tp.groupby(['tracking']).agg(
    presence_rate=('pres', 'mean'),
    inclass_rate=('inclass', 'mean'),
    n_obs=('pres', 'count')
).round(4))
print()
print('Teacher presence by tracking x stream:')
print(tp.groupby(['tracking', 'lowstream']).agg(
    presence=('pres', 'mean'),
    inclass=('inclass', 'mean'),
    n=('pres', 'count')
).round(4))
print()
print('Teacher presence by ETP status x tracking:')
print(tp.groupby(['tracking', 'etpteacher']).agg(
    presence=('pres', 'mean'),
    inclass=('inclass', 'mean'),
    n=('pres', 'count')
).round(4))
```

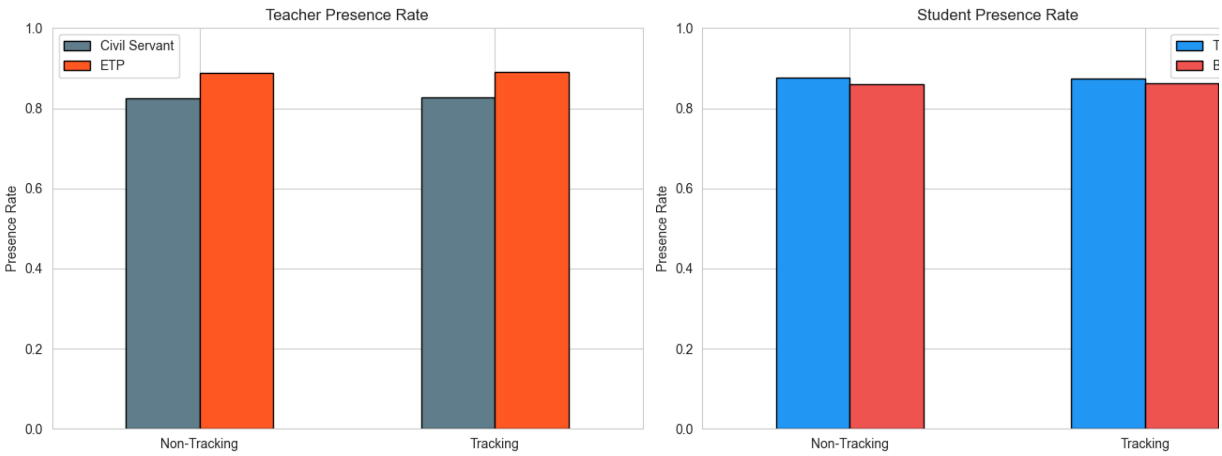
```
=== Teacher Presence Rates ===
          presence_rate  inclass_rate  n_obs
tracking
0.0              0.8375        0.5093   1243
1.0              0.8399        0.5710   1212

Teacher presence by tracking x stream:
                    presence  inclass     n
tracking lowstream
0.0      0.0          0.8370   0.5053  1227
1.0      0.0          0.8576   0.5934   632
         1.0          0.8212   0.5434   565

Teacher presence by ETP status x tracking:
                     presence  inclass    n
tracking etpteacher
0.0      0.0           0.8248   0.4491  993
         1.0           0.8880   0.7480  250
1.0      0.0           0.8269   0.5223  965
         1.0           0.8907   0.7611  247
```

In [20]:
```
print('=== Student Presence Rates ===')
print(sp.groupby(['tracking', 'bottomhalf']).agg(
    presence_rate=('pres', 'mean'),
    n_obs=('pres', 'count')
).round(4))
print()
print('Student presence by gender x tracking:')
print(sp.groupby(['tracking', 'girl']).agg(
    presence=('pres', 'mean'),
    n=('pres', 'count')
).round(4))
```

```
=== Student Presence Rates ===
                     presence_rate  n_obs
tracking bottomhalf
0.0      0.0                0.8775  24098
         1.0                0.8589  18688
1.0      0.0                0.8744  13904
         1.0                0.8611  11663

Student presence by gender x tracking:
               presence      n
tracking girl
0.0      0.0     0.8655  26243
         1.0     0.8605  25123
1.0      0.0     0.8667  12951
         1.0     0.8705  12513
```

In [21]:

```python
fig, axes = plt.subplots(1, 2, figsize=(14, 5))

t_plot = tp.groupby(['tracking', 'etpteacher'])['pres'].mean().unstack()
t_plot.index = ['Non-Tracking', 'Tracking']
t_plot.columns = ['Civil Servant', 'ETP']
t_plot.plot(kind='bar', ax=axes[0], color=['#607D8B', '#FF5722'], edgecol
axes[0].set_title('Teacher Presence Rate')
axes[0].set_ylabel('Presence Rate')
axes[0].set_ylim(0, 1)
axes[0].tick_params(axis='x', rotation=0)

s_plot = sp.groupby(['tracking', 'bottomhalf'])['pres'].mean().unstack()
s_plot.index = ['Non-Tracking', 'Tracking']
s_plot.columns = ['Top Half', 'Bottom Half']
s_plot.plot(kind='bar', ax=axes[1], color=['#2196F3', '#EF5350'], edgecol
axes[1].set_title('Student Presence Rate')
axes[1].set_ylabel('Presence Rate')
axes[1].set_ylim(0, 1)
axes[1].tick_params(axis='x', rotation=0)

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

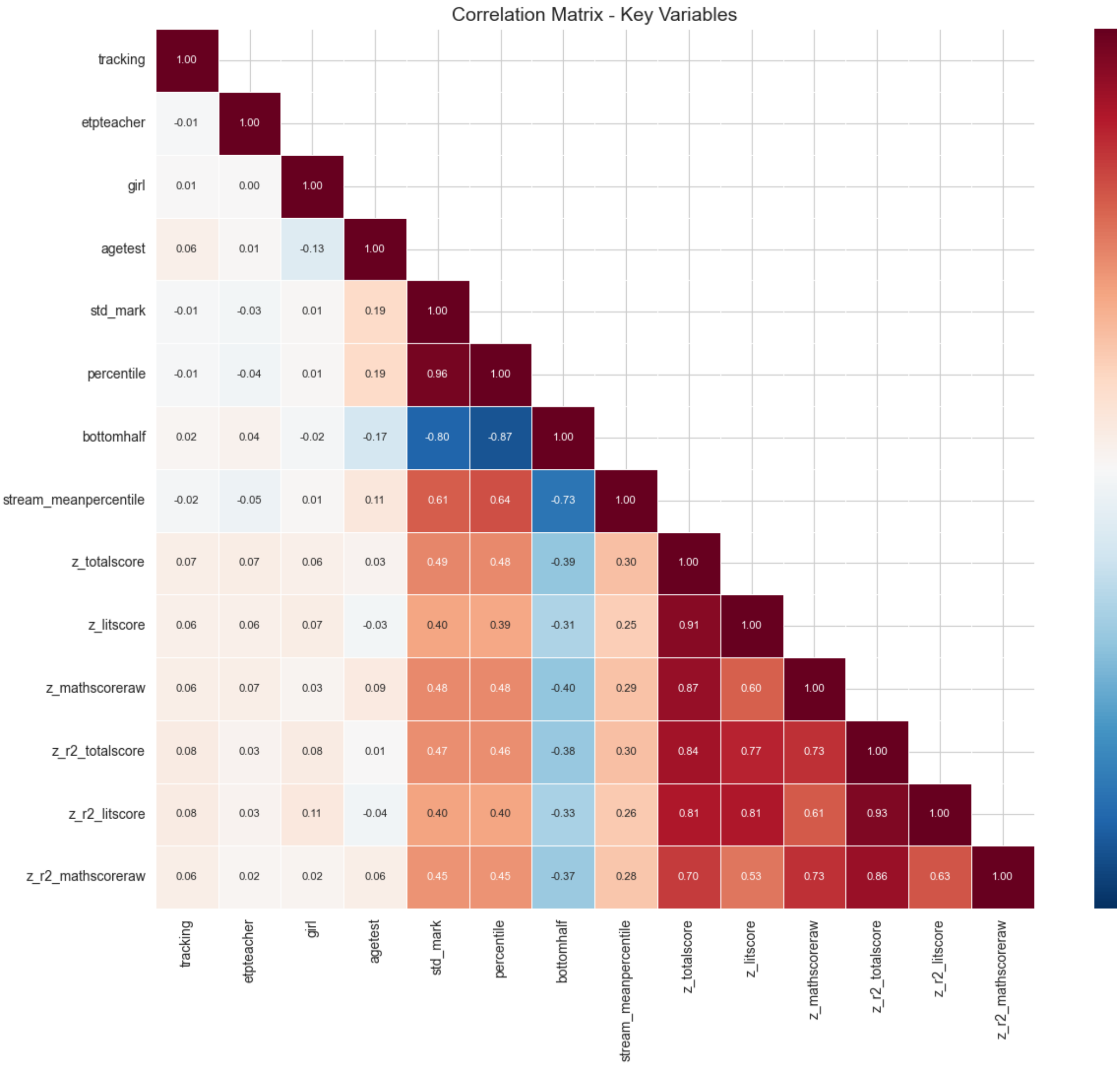


. Correlation Heatmap — Key Variables

In [22]:

```python
corr_vars = ['tracking', 'etpteacher', 'girl', 'agetest', 'std_mark', 'pe
             'bottomhalf', 'stream_meanpercentile',
             'z_totalscore', 'z_litscore', 'z_mathscoreraw',
```

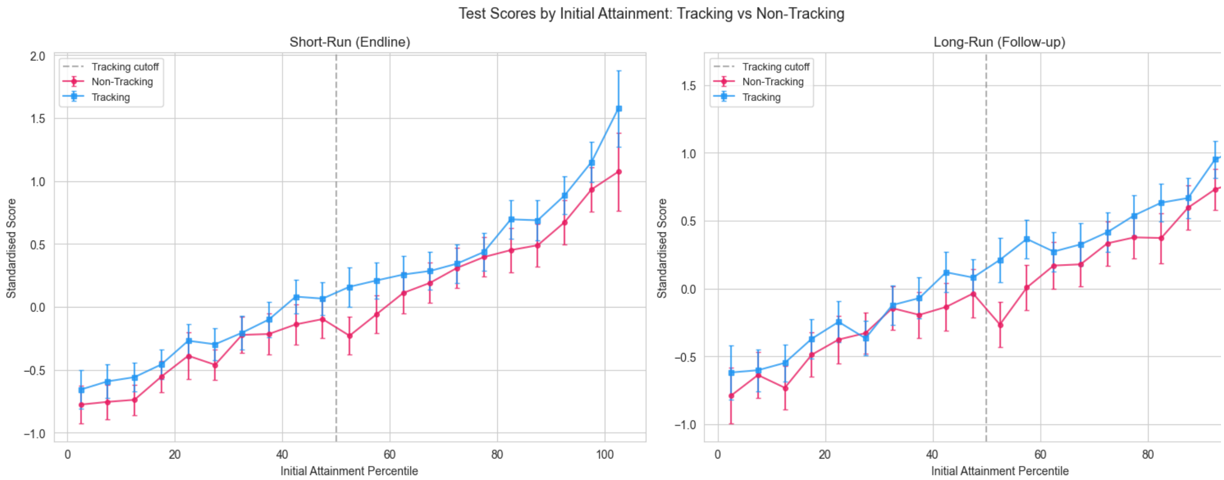
```
                'z_r2_totalscore', 'z_r2_litscore', 'z_r2_mathscoreraw']

corr_matrix = st[corr_vars].corr()

fig, ax = plt.subplots(figsize=(14, 11))
mask = np.triu(np.ones_like(corr_matrix, dtype=bool), k=1)
sns.heatmap(corr_matrix, mask=mask, annot=True, fmt='.2f', cmap='RdBu_r',
            center=0, vmin=-1, vmax=1, square=True, linewidths=0.5, ax=ax
            annot_kws={'size': 8})
ax.set_title('Correlation Matrix - Key Variables', fontsize=14)
plt.tight_layout()
plt.show()
```



. RDD-style Plot: Effect by Baseline Percentile

In [23]:
```
fig, axes = plt.subplots(1, 2, figsize=(16, 6))

for ax, (col, title) in zip(axes, [('z_totalscore', 'Short-Run (Endline)'
                                   ('z_r2_totalscore', 'Long-Run (Follow
    for track_val, label, color, marker in [(0, 'Non-Tracking', '#E91E63'
                                             (1, 'Tracking', '#2196F3', '
        sub = st[st['tracking'] == track_val][['realpercentile', col]].dr
        if len(sub) < 10:
```

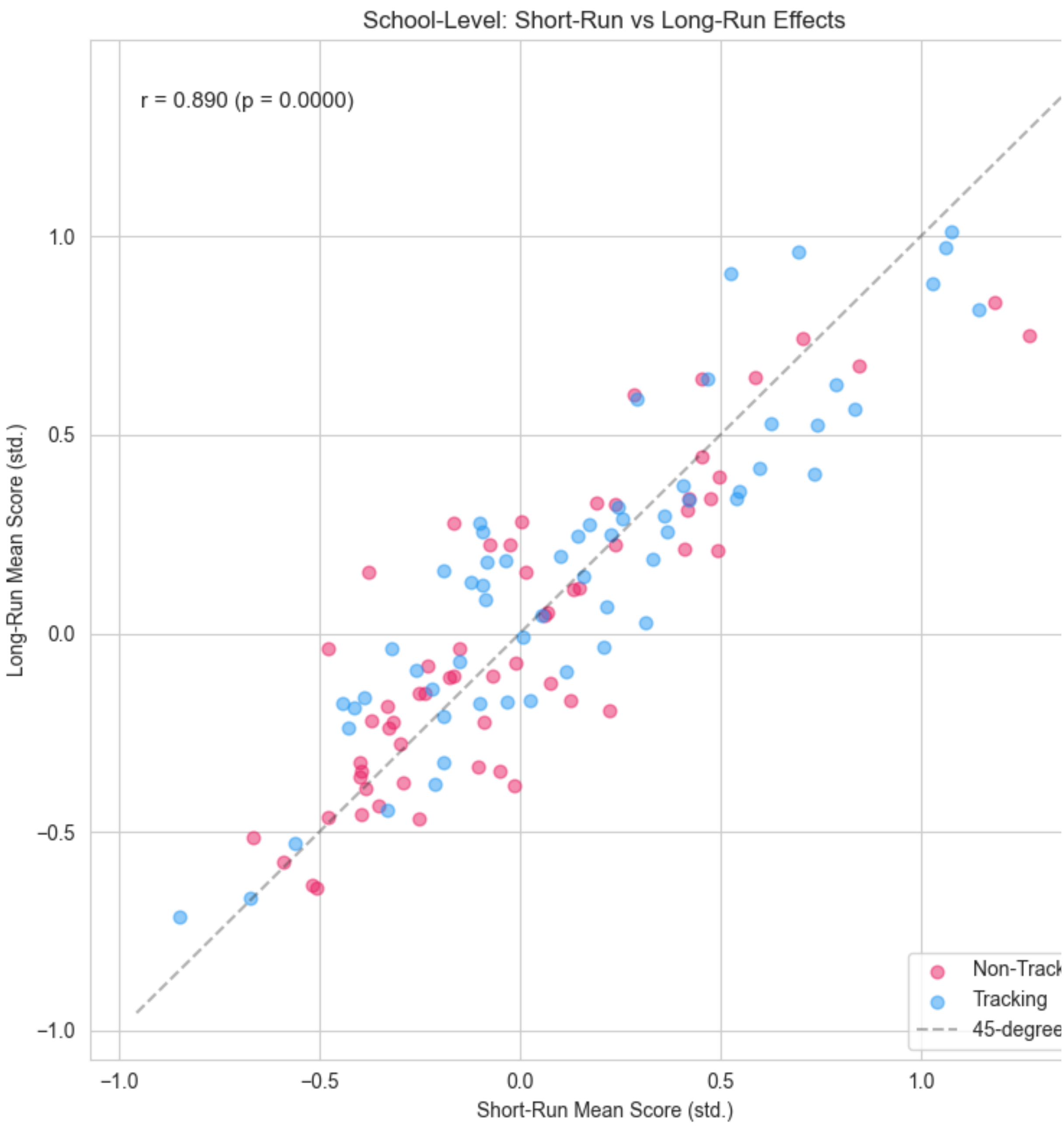
```
            continue
        sub['pctile_bin'] = (sub['realpercentile'] // 5) * 5 + 2.5
        binned = sub.groupby('pctile_bin')[col].agg(['mean', 'sem']).drop
        ax.errorbar(binned.index, binned['mean'], yerr=1.96*binned['sem']
                    fmt=f'{marker}-', color=color, label=label, markersiz
                    capsize=2, alpha=0.8, linewidth=1.5)

    ax.axvline(50, color='gray', linestyle='--', alpha=0.6, label='Tracki
    ax.set_xlabel('Initial Attainment Percentile')
    ax.set_ylabel('Standardised Score')
    ax.set_title(title)
    ax.legend(fontsize=9)

fig.suptitle('Test Scores by Initial Attainment: Tracking vs Non-Tracking
plt.tight_layout()
plt.show()
```



. Short-Run vs Long-Run: Persistence of Effects

In [24]:
```
school = st.groupby(['schoolid', 'tracking']).agg(
    sr_total=('z_totalscore', 'mean'),
    lr_total=('z_r2_totalscore', 'mean'),
    n_students=('pupilid', 'count'),
    pct_girl=('girl', 'mean'),
    mean_age=('agetest', 'mean'),
    pct_etp=('etpteacher', 'mean'),
    mean_baseline=('std_mark', 'mean')
).reset_index()

fig, ax = plt.subplots(figsize=(8, 8))
for track_val, label, color in [(0, 'Non-Tracking', '#E91E63'), (1, 'Trac
    sub = school[school['tracking'] == track_val].dropna(subset=['sr_tota
    ax.scatter(sub['sr_total'], sub['lr_total'], alpha=0.5, s=40, color=c

lim = [min(ax.get_xlim()[0], ax.get_ylim()[0]), max(ax.get_xlim()[1], ax.
ax.plot(lim, lim, 'k--', alpha=0.3, label='45-degree line')
ax.set_xlabel('Short-Run Mean Score (std.)')
ax.set_ylabel('Long-Run Mean Score (std.)')
ax.set_title('School-Level: Short-Run vs Long-Run Effects')
ax.legend()

both = school.dropna(subset=['sr_total', 'lr_total'])
r, p = stats.pearsonr(both['sr_total'], both['lr_total'])
```

```
ax.text(0.05, 0.95, f'r = {r:.3f} (p = {p:.4f})', transform=ax.transAxes,
plt.tight_layout()
plt.show()
```



. Math vs Literacy: Differential Subject Effects

In [25]:
```
subject_het = []
for half_label, half_val in [('Bottom half', 1.0), ('Top half', 0.0), ('A
    for subj_label, subj_col in [('Literacy (SR)', 'z_litscore'), ('Math
                                 ('Literacy (LR)', 'z_r2_litscore'), ('M
        if half_val is not None:
            sub = st[st['bottomhalf'] == half_val]
        else:
            sub = st
        c = sub.loc[sub['tracking'] == 0, subj_col].dropna()
        t = sub.loc[sub['tracking'] == 1, subj_col].dropna()
        if len(c) > 1 and len(t) > 1:
            diff = t.mean() - c.mean()
            se = np.sqrt(c.var()/len(c) + t.var()/len(t))
```

```
                _, pval = stats.ttest_ind(c, t)
                subject_het.append({'group': half_label, 'subject': subj_labe
                                    'ATE': diff, 'SE': se, 'p_value': pval})

subj_df = pd.DataFrame(subject_het)
print('=== Math vs Literacy Effects ===')
print(subj_df.round(4).to_string(index=False))
```

```
=== Math vs Literacy Effects ===
      group        subject    ATE     SE  p_value
Bottom half Literacy (SR) 0.0872 0.0307   0.0045
Bottom half     Math (SR) 0.1758 0.0363   0.0000
Bottom half Literacy (LR) 0.1117 0.0361   0.0020
Bottom half     Math (LR) 0.1202 0.0392   0.0022
   Top half Literacy (SR) 0.1826 0.0425   0.0000
   Top half     Math (SR) 0.1346 0.0344   0.0001
   Top half Literacy (LR) 0.2130 0.0401   0.0000
   Top half     Math (LR) 0.1348 0.0342   0.0001
        All Literacy (SR) 0.1239 0.0266   0.0000
        All     Math (SR) 0.1251 0.0262   0.0000
        All Literacy (LR) 0.1613 0.0272   0.0000
        All     Math (LR) 0.1283 0.0267   0.0000
```

. Age Heterogeneity

In [26]:
```
st['age_group'] = pd.cut(st['agetest'], bins=[0, 8, 9, 10, 20], labels=['

age_het = []
for ag in ['<=8', '9', '10', '>=11']:
    sub = st[st['age_group'] == ag]
    for out_label, out_col in [('Total (SR)', 'z_totalscore'), ('Total (L
        c = sub.loc[sub['tracking'] == 0, out_col].dropna()
        t = sub.loc[sub['tracking'] == 1, out_col].dropna()
        if len(c) > 5 and len(t) > 5:
            diff = t.mean() - c.mean()
            se = np.sqrt(c.var()/len(c) + t.var()/len(t))
            _, pval = stats.ttest_ind(c, t)
            age_het.append({'age_group': ag, 'outcome': out_label,
                            'ATE': diff, 'SE': se, 'p_value': pval, 'N':

age_df = pd.DataFrame(age_het)
print('=== Tracking Effect by Age Group ===')
print(age_df.round(4).to_string(index=False))
```

```
=== Tracking Effect by Age Group ===
age_group    outcome    ATE     SE  p_value    N
      <=8 Total (SR) 0.1770 0.0502   0.0004 1748
      <=8 Total (LR) 0.1878 0.0492   0.0001 1670
        9 Total (SR) 0.1709 0.0511   0.0009 1522
        9 Total (LR) 0.1797 0.0526   0.0007 1388
       10 Total (SR) 0.1262 0.0511   0.0137 1468
       10 Total (LR) 0.1493 0.0552   0.0069 1346
     >=11 Total (SR) 0.0245 0.0624   0.6948 1042
     >=11 Total (LR) 0.1351 0.0670   0.0447  930
```

. District / Geographic Heterogeneity

In [27]:
```
dist_het = []
for dist_val, dist_label in [(1.0, 'Bungoma'), (0.0, 'Non-Bungoma')]:
    sub = st[st['bungoma'] == dist_val]
    for out_label, out_col in [('Total (SR)', 'z_totalscore'), ('Total (L
        c = sub.loc[sub['tracking'] == 0, out_col].dropna()
        t = sub.loc[sub['tracking'] == 1, out_col].dropna()
        if len(c) > 1 and len(t) > 1:
            diff = t.mean() - c.mean()
            se = np.sqrt(c.var()/len(c) + t.var()/len(t))
            _, pval = stats.ttest_ind(c, t)
            dist_het.append({'district': dist_label, 'outcome': out_label
                             'ATE': diff, 'SE': se, 'p_value': pval})

print('=== Tracking Effect by District ===')
print(pd.DataFrame(dist_het).round(4).to_string(index=False))
```

```
=== Tracking Effect by District ===
   district     outcome    ATE     SE  p_value
    Bungoma Total (SR) 0.0933 0.0487   0.0553
    Bungoma Total (LR) 0.1009 0.0541   0.0626
Non-Bungoma Total (SR) 0.1689 0.0313   0.0000
Non-Bungoma Total (LR) 0.1840 0.0313   0.0000
```

. Regression-Based Estimates (with controls)

In [28]:
```
import statsmodels.formula.api as smf

# Drop rows missing key variables so groups align with model sample
reg_vars = ['z_totalscore', 'tracking', 'girl', 'percentile', 'agetest',
reg = st.dropna(subset=reg_vars).copy()
reg['tracking_bottomhalf'] = reg['tracking'] * reg['bottomhalf']

print('=== Model 1: Simple ITT ===')
m1 = smf.ols('z_totalscore ~ tracking', data=reg).fit(cov_type='cluster',
print(f'  tracking coef = {m1.params["tracking"]:.4f}, SE = {m1.bse["trac

print('\n=== Model 2: ITT with controls ===')
m2 = smf.ols('z_totalscore ~ tracking + girl + percentile + agetest + etp
             data=reg).fit(cov_type='cluster', cov_kwds={'groups': reg['s
print(m2.summary2().tables[1].to_string())

print('\n=== Model 3: Heterogeneous by achievement half ===')
m3 = smf.ols('z_totalscore ~ tracking + bottomhalf + tracking_bottomhalf
             data=reg).fit(cov_type='cluster', cov_kwds={'groups': reg['s
print(m3.summary2().tables[1].to_string())
print(f'\n  Effect on top half:    tracking = {m3.params["tracking"]:.4f}
print(f'  Effect on bottom half: tracking + interaction = {m3.params["tra
```

```
=== Model 1: Simple ITT ===
  tracking coef = 0.1477, SE = 0.0773, p = 0.0560

=== Model 2: ITT with controls ===
               Coef.  Std.Err.          z         P>|z|    [0.025    0.975]
Intercept  -0.665404  0.131190  -5.072059   3.935338e-07 -0.922532 -0.408276
tracking    0.174448  0.077037   2.264472   2.354512e-02  0.023458  0.325438
girl        0.082115  0.028819   2.849361   4.380709e-03  0.025631  0.138599
percentile  0.017512  0.000728  24.042751   9.940494e-128  0.016084  0.018939
agetest    -0.041261  0.013461  -3.065223   2.175080e-03 -0.067644 -0.014878
etpteacher  0.181895  0.037897   4.799703   1.589012e-06  0.107618  0.256172

=== Model 3: Heterogeneous by achievement half ===
                        Coef.  Std.Err.          z         P>|z|    [0.025
0.975]
Intercept           -0.920462  0.140075  -6.571198  4.991201e-11 -1.195004
-0.645920
tracking             0.193168  0.093018   2.076676  3.783151e-02  0.010856
0.375480
bottomhalf           0.208327  0.053365   3.903790  9.469809e-05  0.103733
0.312921
tracking_bottomhalf -0.041130  0.070371  -0.584483  5.588953e-01 -0.179054
0.096793
girl                 0.084572  0.028673   2.949585  3.182010e-03  0.028375
0.140770
percentile           0.020324  0.001032  19.692413  2.504696e-86  0.018301
0.022347
agetest             -0.040237  0.013487  -2.983375  2.850888e-03 -0.066672
-0.013803
etpteacher           0.181014  0.037852   4.782161  1.734211e-06  0.106826
0.255202

  Effect on top half:    tracking = 0.1932 (p=0.0378)
  Effect on bottom half: tracking + interaction = 0.1520
```

In [29]:

```python
# Long-run regressions — use separate clean sample
reg_lr_vars = ['z_r2_totalscore', 'tracking', 'girl', 'percentile', 'aget
reg_lr = st.dropna(subset=reg_lr_vars).copy()
reg_lr['tracking_bottomhalf'] = reg_lr['tracking'] * reg_lr['bottomhalf']

print('=== Long-Run: Model 2 with controls ===')
m2lr = smf.ols('z_r2_totalscore ~ tracking + girl + percentile + agetest
               data=reg_lr).fit(cov_type='cluster', cov_kwds={'groups': r
print(m2lr.summary2().tables[1].to_string())

print('\n=== Long-Run: Model 3 heterogeneous ===')
m3lr = smf.ols('z_r2_totalscore ~ tracking + bottomhalf + tracking_bottom
               data=reg_lr).fit(cov_type='cluster', cov_kwds={'groups': r
print(m3lr.summary2().tables[1].to_string())
print(f'\n  LR Effect on top half:    {m3lr.params["tracking"]:.4f}')
print(f'  LR Effect on bottom half: {m3lr.params["tracking"] + m3lr.param
```

```
=== Long-Run: Model 2 with controls ===
                Coef.  Std.Err.          z           P>|z|    [0.025    0.975]
Intercept  -0.433323  0.132404  -3.272746   1.065082e-03 -0.692830 -0.173817
tracking    0.176476  0.073361   2.405575   1.614705e-02  0.032690  0.320261
girl        0.126548  0.031435   4.025767   5.678993e-05  0.064938  0.188159
percentile  0.016883  0.000658  25.660712  3.210786e-145  0.015593  0.018172
agetest    -0.057292  0.013596  -4.214018   2.508669e-05 -0.083939 -0.030645
etpteacher  0.095202  0.033306   2.858387   4.258009e-03  0.029923  0.160480

=== Long-Run: Model 3 heterogeneous ===
                         Coef.  Std.Err.          z         P>|z|    [0.025
0.975]
Intercept           -0.666862  0.142671  -4.674113  2.952266e-06 -0.946492
-0.387231
tracking             0.216226  0.079896   2.706345  6.802832e-03  0.059633
0.372820
bottomhalf           0.205714  0.056664   3.630432  2.829469e-04  0.094655
0.316773
tracking_bottomhalf -0.084763  0.065230  -1.299443  1.937919e-01 -0.212612
0.043086
girl                 0.129670  0.031233   4.151686  3.300343e-05  0.068454
0.190886
percentile           0.019283  0.000999  19.292968  6.153679e-83  0.017324
0.021242
agetest             -0.056228  0.013631  -4.124978  3.707710e-05 -0.082944
-0.029511
etpteacher           0.095713  0.033209   2.882117  3.950135e-03  0.030624
0.160802

  LR Effect on top half:    0.2162
  LR Effect on bottom half: 0.1315
```

. Summary of Findings

Key patterns to investigate further:

Finding	Strength	Promising for
Tracking raises scores overall	Moderate positive ITT	Baseline replica
Both halves benefit (bottom & top)	Key result - not zero-sum	Core finding
Effects differ by gender	Interaction effects	Gender x tracki
Contract teacher interactions	ETP x tracking x stream	Mechanism sto
Peer mean correlates with outcomes	Reduced-form in non-tracking schools	IV strategy
Teacher attendance differs by treatment	Mechanism	Behavioural cha
Short-run vs long-run persistence	Effects may fade	Dynamic questi
Math vs literacy differential	Subject-specific effects	Possible extens
Geographic heterogeneity	Bungoma vs other	Context depend

Top research question candidates:

1. **Peer effects via IV -** Use random stream assignment in non-tracking schools to instrument quality
2. **Gender x tracking interaction -** Do girls benefit more/less? Does the gender gap narrow o
3. **Contract teacher as mechanism -** Does the tracking effect operate through changed teach incentives?
4. **Persistence -** Why do some effects persist to the long run and others don't?
5. **RDD at the tracking cutoff -** Discontinuity in outcomes at the median within tracking schoo