results_analysis_visualisation

May 14, 2021

[1]:
```
import os
import pandas as pd
from matplotlib import pyplot as plt
```

[2]:
```
RESULTS_DIR = '..\\..\\results\\'
AGENTS = [
    'Random-vs-Random',
    'SL-vs-Random',
    'SL-scaled-vs-Random',
    'RL-vs-Random',
    'SL-vs-SL',
    'RL-vs-2SL',
    'SL-vs-2RL'
]
```

[3]:
```
scores = {}
for agent in AGENTS:
    if agent == 'Random-vs-Random':
        scores[agent] = []
    else:
        scores[agent] = {
            'East': [[], [], []],
            'South': [[], [], []],
            'West': [[], [], []]
        }

scores
```

[3]:
```
{'Random-vs-Random': [],
 'SL-vs-Random': {'East': [[], [], []],
  'South': [[], [], []],
  'West': [[], [], []]},
 'SL-scaled-vs-Random': {'East': [[], [], []],
  'South': [[], [], []],
  'West': [[], [], []]},
 'RL-vs-Random': {'East': [[], [], []],
  'South': [[], [], []],
  'West': [[], [], []]},
```

```
 'SL-vs-SL': {'East': [[], [], []],
  'South': [[], [], []],
  'West': [[], [], []]},
 'RL-vs-2SL': {'East': [[], [], []],
  'South': [[], [], []],
  'West': [[], [], []]},
 'SL-vs-2RL': {'East': [[], [], []],
  'South': [[], [], []],
  'West': [[], [], []]}}
```

[4]:
```
illegal_counts = {}
for agent in AGENTS:
    illegal_counts[agent] = 0

illegal_counts
```

[4]:
```
{'Random-vs-Random': 0,
 'SL-vs-Random': 0,
 'SL-scaled-vs-Random': 0,
 'RL-vs-Random': 0,
 'SL-vs-SL': 0,
 'RL-vs-2SL': 0,
 'SL-vs-2RL': 0}
```

[5]:
```
with open(os.path.join(RESULTS_DIR, 'Random-vs-Random.txt'), 'r') as fread:
    scores['Random-vs-Random'] = []
    for i in range(5000):
        line = fread.readline().strip().split()
        seed, score1, score2, score3 = map(int, line)
        if score1 != 0:
            scores['Random-vs-Random'].append(score1)
        if score2 != 0:
            scores['Random-vs-Random'].append(score2)
        if score3 != 0:
            scores['Random-vs-Random'].append(score3)
```

[6]:
```
with open(os.path.join(RESULTS_DIR, 'SL-vs-SL.txt'), 'r') as fread:
    scores['SL-vs-SL'] = {
        'East': [[], [], []],
        'South': [[], [], []],
        'West': [[], [], []]
    }
    for i in range(5000):
        line = fread.readline().strip().split()
        seed, score_east, score_south, score_west = map(int, line)
        if not score_east == score_south == score_west == 0:
            if score_east >= score_south >= score_west:
```

```
                scores['SL-vs-SL']['East'][0].append(score_east)
                scores['SL-vs-SL']['South'][1].append(score_south)
                scores['SL-vs-SL']['West'][2].append(score_west)
            elif score_east >= score_west >= score_south:
                scores['SL-vs-SL']['East'][0].append(score_east)
                scores['SL-vs-SL']['South'][2].append(score_south)
                scores['SL-vs-SL']['West'][1].append(score_west)
            elif score_south >= score_east >= score_west:
                scores['SL-vs-SL']['East'][1].append(score_east)
                scores['SL-vs-SL']['South'][0].append(score_south)
                scores['SL-vs-SL']['West'][2].append(score_west)
            elif score_south >= score_west >= score_east:
                scores['SL-vs-SL']['East'][2].append(score_east)
                scores['SL-vs-SL']['South'][0].append(score_south)
                scores['SL-vs-SL']['West'][1].append(score_west)
            elif score_west >= score_east >= score_south:
                scores['SL-vs-SL']['East'][1].append(score_east)
                scores['SL-vs-SL']['South'][2].append(score_south)
                scores['SL-vs-SL']['West'][0].append(score_west)
            elif score_west >= score_south >= score_east:
                scores['SL-vs-SL']['East'][2].append(score_east)
                scores['SL-vs-SL']['South'][1].append(score_south)
                scores['SL-vs-SL']['West'][0].append(score_west)
            else:
                assert False
```

[7]:
```
for agent in AGENTS:
    if agent != 'Random-vs-Random' and agent != 'SL-vs-SL':
        scores[agent] = {
            'East': [[], [], []],
            'South': [[], [], []],
            'West': [[], [], []]
        }
        for (wind_index, wind) in (0, 'East'), (1, 'South'), (2, 'West'):
            filename = agent + '\\' + agent + '-' + wind + '.txt'
            with open(os.path.join(RESULTS_DIR, filename), 'r') as fread:
                for i in range(5000):
                    line = fread.readline().strip().split()
                    if line[1] == 'ILLEGAL':
                        illegal_counts[agent] += 1
                    else:
                        score = [0, 0, 0]
                        seed, score[0], score[1], score[2] = map(int, line)
                        if not score[0] == score[1] == score[2] == 0:
                            if score[wind_index] >= score[(wind_index + 1) % 3]␣
↪\
```

```
                                    and score[wind_index] >= score[(wind_index␣
↪+ 2) % 3]:
                                scores[agent][wind][0].append(score[wind_index])
                            elif score[wind_index] < score[(wind_index + 1) %␣
↪3] \
                                    and score[wind_index] < score[(wind_index +␣
↪2) % 3]:
                                scores[agent][wind][2].append(score[wind_index])
                            else:
                                scores[agent][wind][1].append(score[wind_index])
```

[8]:
```
summary = {}
for agent in AGENTS:
    if agent == 'Random-vs-Random':
        summary[agent] = {}
        summary[agent]['1st_place_count'] = len(scores[agent]) // 2
        summary[agent]['2nd_place_count'] = len(scores[agent]) // 2
        summary[agent]['3rd_place_count'] = len(scores[agent]) // 2
        summary[agent]['draw_count'] = (5000 - illegal_counts[agent]) * 3 -␣
↪len(scores[agent]) // 2 * 3
    else:
        summary[agent] = {
            'East': {},
            'South': {},
            'West': {},
            'Total': {}
        }
        for wind in 'East', 'South', 'West':
            summary[agent][wind]['1st_place_count'] =␣
↪len(scores[agent][wind][0])
            summary[agent][wind]['2nd_place_count'] =␣
↪len(scores[agent][wind][1])
            summary[agent][wind]['3rd_place_count'] =␣
↪len(scores[agent][wind][2])
            summary[agent][wind]['draw_count'] = 5000 - illegal_counts[agent] -␣
↪len(scores[agent][wind][0]) - len(scores[agent][wind][1]) -␣
↪len(scores[agent][wind][2])
        summary[agent]['Total']['1st_place_count'] =␣
↪len(scores[agent]['East'][0]) + len(scores[agent]['South'][0]) +␣
↪len(scores[agent]['West'][0])
        summary[agent]['Total']['2nd_place_count'] =␣
↪len(scores[agent]['East'][1]) + len(scores[agent]['South'][1]) +␣
↪len(scores[agent]['West'][1])
        summary[agent]['Total']['3rd_place_count'] =␣
↪len(scores[agent]['East'][2]) + len(scores[agent]['South'][2]) +␣
↪len(scores[agent]['West'][2])
```

```
        summary[agent]['Total']['draw_count'] = (5000 - illegal_counts[agent])␣
↪* 3 - summary[agent]['Total']['1st_place_count'] -␣
↪summary[agent]['Total']['2nd_place_count'] -␣
↪summary[agent]['Total']['3rd_place_count']

summary
```

[8]:

```
{'Random-vs-Random': {'1st_place_count': 3,
  '2nd_place_count': 3,
  '3rd_place_count': 3,
  'draw_count': 14991},
 'SL-vs-Random': {'East': {'1st_place_count': 1100,
   '2nd_place_count': 3,
   '3rd_place_count': 4,
   'draw_count': 3893},
  'South': {'1st_place_count': 1134,
   '2nd_place_count': 3,
   '3rd_place_count': 1,
   'draw_count': 3862},
  'West': {'1st_place_count': 1036,
   '2nd_place_count': 8,
   '3rd_place_count': 2,
   'draw_count': 3954},
  'Total': {'1st_place_count': 3270,
   '2nd_place_count': 14,
   '3rd_place_count': 7,
   'draw_count': 11709}},
 'SL-scaled-vs-Random': {'East': {'1st_place_count': 1070,
   '2nd_place_count': 2,
   '3rd_place_count': 7,
   'draw_count': 3921},
  'South': {'1st_place_count': 1136,
   '2nd_place_count': 4,
   '3rd_place_count': 0,
   'draw_count': 3860},
  'West': {'1st_place_count': 1001,
   '2nd_place_count': 10,
   '3rd_place_count': 3,
   'draw_count': 3986},
  'Total': {'1st_place_count': 3207,
   '2nd_place_count': 16,
   '3rd_place_count': 10,
   'draw_count': 11767}},
 'RL-vs-Random': {'East': {'1st_place_count': 3642,
   '2nd_place_count': 1,
   '3rd_place_count': 162,
   'draw_count': 1144},
```

```
 'South': {'1st_place_count': 3560,
  '2nd_place_count': 4,
  '3rd_place_count': 171,
  'draw_count': 1214},
 'West': {'1st_place_count': 3544,
  '2nd_place_count': 3,
  '3rd_place_count': 141,
  'draw_count': 1261},
 'Total': {'1st_place_count': 10746,
  '2nd_place_count': 8,
  '3rd_place_count': 474,
  'draw_count': 3619}},
'SL-vs-SL': {'East': {'1st_place_count': 935,
  '2nd_place_count': 595,
  '3rd_place_count': 1242,
  'draw_count': 2228},
 'South': {'1st_place_count': 987,
  '2nd_place_count': 1216,
  '3rd_place_count': 569,
  'draw_count': 2228},
 'West': {'1st_place_count': 850,
  '2nd_place_count': 961,
  '3rd_place_count': 961,
  'draw_count': 2228},
 'Total': {'1st_place_count': 2772,
  '2nd_place_count': 2772,
  '3rd_place_count': 2772,
  'draw_count': 6684}},
'RL-vs-2SL': {'East': {'1st_place_count': 2869,
  '2nd_place_count': 387,
  '3rd_place_count': 695,
  'draw_count': 1042},
 'South': {'1st_place_count': 2891,
  '2nd_place_count': 816,
  '3rd_place_count': 246,
  'draw_count': 1040},
 'West': {'1st_place_count': 2780,
  '2nd_place_count': 908,
  '3rd_place_count': 262,
  'draw_count': 1043},
 'Total': {'1st_place_count': 8540,
  '2nd_place_count': 2111,
  '3rd_place_count': 1203,
  'draw_count': 3125}},
'SL-vs-2RL': {'East': {'1st_place_count': 288,
  '2nd_place_count': 354,
  '3rd_place_count': 4073,
```

```
 'draw_count': 281},
'South': {'1st_place_count': 305,
 '2nd_place_count': 1903,
 '3rd_place_count': 2466,
 'draw_count': 322},
'West': {'1st_place_count': 245,
 '2nd_place_count': 1883,
 '3rd_place_count': 2582,
 'draw_count': 286},
'Total': {'1st_place_count': 838,
 '2nd_place_count': 4140,
 '3rd_place_count': 9121,
 'draw_count': 889}}}
```

[9]:
```
win_rates = {}
for agent in AGENTS:
    if agent == 'Random-vs-Random':
        win_rates[agent] = {}
        win_rates[agent]['1st_place_rate'] = len(scores[agent]) / 2 / ((5000 -␣
↪illegal_counts[agent]) * 3)
        win_rates[agent]['2nd_place_rate'] = len(scores[agent]) / 2 / ((5000 -␣
↪illegal_counts[agent]) * 3)
        win_rates[agent]['3rd_place_rate'] = len(scores[agent]) / 2 / ((5000 -␣
↪illegal_counts[agent]) * 3)
        win_rates[agent]['draw_rate'] = ((5000 - illegal_counts[agent]) * 3 -␣
↪len(scores[agent]) // 2 * 3) / ((5000 - illegal_counts[agent]) * 3)
    else:
        win_rates[agent] = {
            'East': {},
            'South': {},
            'West': {},
            'Total': {}
        }
        for wind in 'East', 'South', 'West':
            win_rates[agent][wind]['1st_place_rate'] =␣
↪len(scores[agent][wind][0]) / (5000 - illegal_counts[agent])
            win_rates[agent][wind]['2nd_place_rate'] =␣
↪len(scores[agent][wind][1]) / (5000 - illegal_counts[agent])
            win_rates[agent][wind]['3rd_place_rate'] =␣
↪len(scores[agent][wind][2]) / (5000 - illegal_counts[agent])
            win_rates[agent][wind]['draw_rate'] = (5000 - illegal_counts[agent]␣
↪- len(scores[agent][wind][0]) - len(scores[agent][wind][1]) -␣
↪len(scores[agent][wind][2])) / (5000 - illegal_counts[agent])
        win_rates[agent]['Total']['1st_place_rate'] =␣
↪(len(scores[agent]['East'][0]) + len(scores[agent]['South'][0]) +␣
↪len(scores[agent]['West'][0])) / ((5000 - illegal_counts[agent]) * 3)
```

```
        win_rates[agent]['Total']['2nd_place_rate'] =␣
↪(len(scores[agent]['East'][1]) + len(scores[agent]['South'][1]) +␣
↪len(scores[agent]['West'][1])) / ((5000 - illegal_counts[agent]) * 3)
        win_rates[agent]['Total']['3rd_place_rate'] =␣
↪(len(scores[agent]['East'][2]) + len(scores[agent]['South'][2]) +␣
↪len(scores[agent]['West'][2])) / ((5000 - illegal_counts[agent]) * 3)
        win_rates[agent]['Total']['draw_rate'] = 1 -␣
↪win_rates[agent]['Total']['1st_place_rate'] -␣
↪win_rates[agent]['Total']['2nd_place_rate'] -␣
↪win_rates[agent]['Total']['3rd_place_rate']

win_rates
```

[9]:
```
{'Random-vs-Random': {'1st_place_rate': 0.0002,
  '2nd_place_rate': 0.0002,
  '3rd_place_rate': 0.0002,
  'draw_rate': 0.9994},
 'SL-vs-Random': {'East': {'1st_place_rate': 0.22,
   '2nd_place_rate': 0.0006,
   '3rd_place_rate': 0.0008,
   'draw_rate': 0.7786},
  'South': {'1st_place_rate': 0.2268,
   '2nd_place_rate': 0.0006,
   '3rd_place_rate': 0.0002,
   'draw_rate': 0.7724},
  'West': {'1st_place_rate': 0.2072,
   '2nd_place_rate': 0.0016,
   '3rd_place_rate': 0.0004,
   'draw_rate': 0.7908},
  'Total': {'1st_place_rate': 0.218,
   '2nd_place_rate': 0.0009333333333333333,
   '3rd_place_rate': 0.0004666666666666666,
   'draw_rate': 0.7806000000000001}},
 'SL-scaled-vs-Random': {'East': {'1st_place_rate': 0.214,
   '2nd_place_rate': 0.0004,
   '3rd_place_rate': 0.0014,
   'draw_rate': 0.7842},
  'South': {'1st_place_rate': 0.2272,
   '2nd_place_rate': 0.0008,
   '3rd_place_rate': 0.0,
   'draw_rate': 0.772},
  'West': {'1st_place_rate': 0.2002,
   '2nd_place_rate': 0.002,
   '3rd_place_rate': 0.0006,
   'draw_rate': 0.7972},
  'Total': {'1st_place_rate': 0.2138,
   '2nd_place_rate': 0.0010666666666666667,
```

```
  '3rd_place_rate': 0.0006666666666666666,
  'draw_rate': 0.7844666666666666}},
'RL-vs-Random': {'East': {'1st_place_rate': 0.7359062436855931,
  '2nd_place_rate': 0.00020206102242877348,
  '3rd_place_rate': 0.03273388563346131,
  'draw_rate': 0.23115780965851687},
 'South': {'1st_place_rate': 0.7193372398464336,
  '2nd_place_rate': 0.0008082440897150939,
  '3rd_place_rate': 0.034552434835320264,
  'draw_rate': 0.245302081228531},
 'West': {'1st_place_rate': 0.7161042634875733,
  '2nd_place_rate': 0.0006061830672863204,
  '3rd_place_rate': 0.02849060416245706,
  'draw_rate': 0.2547989492826834},
 'Total': {'1st_place_rate': 0.7237825823398666,
  '2nd_place_rate': 0.000538829393143396,
  '3rd_place_rate': 0.03192564154374621,
  'draw_rate': 0.24375294672324382}},
'SL-vs-SL': {'East': {'1st_place_rate': 0.187,
  '2nd_place_rate': 0.119,
  '3rd_place_rate': 0.2484,
  'draw_rate': 0.4456},
 'South': {'1st_place_rate': 0.1974,
  '2nd_place_rate': 0.2432,
  '3rd_place_rate': 0.1138,
  'draw_rate': 0.4456},
 'West': {'1st_place_rate': 0.17,
  '2nd_place_rate': 0.1922,
  '3rd_place_rate': 0.1922,
  'draw_rate': 0.4456},
 'Total': {'1st_place_rate': 0.1848,
  '2nd_place_rate': 0.1848,
  '3rd_place_rate': 0.1848,
  'draw_rate': 0.4456000000000001}},
'RL-vs-2SL': {'East': {'1st_place_rate': 0.5746044462247146,
  '2nd_place_rate': 0.07750851191668336,
  '3rd_place_rate': 0.13919487282195073,
  'draw_rate': 0.2086921690366513},
 'South': {'1st_place_rate': 0.5790106148608051,
  '2nd_place_rate': 0.16342880032044863,
  '3rd_place_rate': 0.04926897656719407,
  'draw_rate': 0.2082916082515521 8},
 'West': {'1st_place_rate': 0.5567794912878029,
  '2nd_place_rate': 0.181854596435009,
  '3rd_place_rate': 0.05247346284798718,
  'draw_rate': 0.20889244942920088},
 'Total': {'1st_place_rate': 0.5701315174577742,
```

```
    '2nd_place_rate': 0.140930636224047,
    '3rd_place_rate': 0.08031243741237733,
    'draw_rate': 0.20862540890580145}},
  'SL-vs-2RL': {'East': {'1st_place_rate': 0.057646116893514815,
    '2nd_place_rate': 0.07085668534827862,
    '3rd_place_rate': 0.8152522017614091,
    'draw_rate': 0.056244995996797435},
   'South': {'1st_place_rate': 0.06104883907125701,
    '2nd_place_rate': 0.38090472377902324,
    '3rd_place_rate': 0.4935948759007206,
    'draw_rate': 0.0644515612489992},
   'West': {'1st_place_rate': 0.04903923138510809,
    '2nd_place_rate': 0.37690152121697357,
    '3rd_place_rate': 0.5168134507606085,
    'draw_rate': 0.057245796637309845},
   'Total': {'1st_place_rate': 0.0559113957832933,
    '2nd_place_rate': 0.27622097678142515,
    '3rd_place_rate': 0.608553509474246,
    'draw_rate': 0.05931411796103547}}}
```

[10]:
```
def get_rank(score, wind_index):
    # 3: 1st place, 2: draw, 1: 2nd place, 0: 3rd place
    if score[0] == score[1] == score[2] == 0:
        return 2
    elif score[wind_index] >= score[(wind_index + 1) % 3] and score[wind_index]␣
↪>= score[(wind_index + 2) % 3]:
        return 3
    elif score[wind_index] < score[(wind_index + 1) % 3] and score[wind_index]␣
↪< score[(wind_index + 2) % 3]:
        return 0
    else:
        return 1
```

[11]:
```
plus = {
    'East': 0,
    'South': 0,
    'West': 0,
    'Total': 0
}
minus = {
    'East': 0,
    'South': 0,
    'West': 0,
    'Total': 0
}
tie = {
    'East': 0,
```

```
    'South': 0,
    'West': 0,
    'Total': 0
}

for (wind_index, wind) in (0, 'East'), (1, 'South'), (2, 'West'):
    with open(os.path.join(RESULTS_DIR, 'SL-vs-Random\\SL-vs-Random-' + wind +␣
↪'.txt'), 'r') as fread_sl:
        with open(os.path.join(RESULTS_DIR,␣
↪'SL-scaled-vs-Random\\SL-scaled-vs-Random-' + wind + '.txt'), 'r') as␣
↪fread_sl_scaled:
            for i in range(5000):
                line_sl = fread_sl.readline().strip().split()
                score_sl = [0, 0, 0]
                seed_sl, score_sl[0], score_sl[1], score_sl[2] = map(int,␣
↪line_sl)

                line_sl_scaled = fread_sl_scaled.readline().strip().split()
                score_sl_scaled = [0, 0, 0]
                seed_sl_scaled, score_sl_scaled[0], score_sl_scaled[1],␣
↪score_sl_scaled[2] = map(int, line_sl_scaled)

                assert seed_sl == seed_sl_scaled == i

                if get_rank(score_sl_scaled, wind_index) > get_rank(score_sl,␣
↪wind_index):
                    plus[wind] += 1
                    plus['Total'] += 1
                elif get_rank(score_sl_scaled, wind_index) < get_rank(score_sl,␣
↪wind_index):
                    minus[wind] += 1
                    minus['Total'] += 1
                else:
                    tie[wind] += 1
                    tie['Total'] += 1

{
    'Plus': plus,
    'Minus': minus,
    'Null': tie
}
```

[11]:
```
{'Plus': {'East': 513, 'South': 529, 'West': 488, 'Total': 1530},
 'Minus': {'East': 545, 'South': 528, 'West': 523, 'Total': 1596},
 'Null': {'East': 3942, 'South': 3943, 'West': 3989, 'Total': 11874}}
```

[12]:
```
from scipy.stats import binom

for wind in 'East', 'South', 'West', 'Total':
    n = 2 * ((tie[wind] + 1) // 2) + plus[wind] + minus[wind]
    k = (tie[wind] + 1) // 2 + min(plus[wind], minus[wind])
    print(wind + ': n = ' + str(n) + ', k = ' + str(k) + ', prob = ' +␣
 ↪str(binom.cdf(k, n, 0.5)))
```

```
East: n = 5000, k = 2484, prob = 0.3305480645857834
South: n = 5001, k = 2500, prob = 0.5000000000030298
West: n = 5001, k = 2483, prob = 0.31533701123345803
Total: n = 15000, k = 7467, prob = 0.29780618378120405
```

[13]:
```
from scipy.stats import multinomial

p_1 = win_rates['Random-vs-Random']['1st_place_rate']
p_2 = win_rates['Random-vs-Random']['2nd_place_rate']
p_3 = win_rates['Random-vs-Random']['3rd_place_rate']
p_draw = win_rates['Random-vs-Random']['draw_rate']

for action in 'SL-vs-Random', 'SL-scaled-vs-Random', 'RL-vs-Random':
    print(action + ':')
    for wind in 'East', 'South', 'West', 'Total':
        x_1 = summary[action][wind]['1st_place_count']
        x_2 = summary[action][wind]['2nd_place_count']
        x_3 = summary[action][wind]['3rd_place_count']
        x_draw = summary[action][wind]['draw_count']

        n = x_1 + x_2 + x_3 + x_draw
        multinom = multinomial(n, [p_1, p_2, p_3, p_draw])

        cdf = 0
        for i in range(x_1 + 1, n + 1):
            for j in range(0, min(n - i + 1, x_3)):
                for k in range(0, n - i - j + 1):
                    cdf += multinom.pmf([i, k, j, n - i - j - k])
        for k in range(0, x_2):
            cdf += multinom.pmf([x_1, k, x_3, n - i - j - k])
        cdf += multinom.pmf([x_1, x_2, x_3, x_draw])

        print(wind + ': n = ' + str(n)
              + ', x_1 = ' + str(x_1)
              + ', x_2 = ' + str(x_2)
              + ', x_3 = ' + str(x_3)
              + ', x_draw = ' + str(x_draw)
              + ', prob = ' + str(cdf))
```

```
SL-vs-Random:
```

```
East: n = 5000, x_1 = 1100, x_2 = 3, x_3 = 4, x_draw = 3893, prob = 0.0
South: n = 5000, x_1 = 1134, x_2 = 3, x_3 = 1, x_draw = 3862, prob = 0.0
West: n = 5000, x_1 = 1036, x_2 = 8, x_3 = 2, x_draw = 3954, prob = 0.0
Total: n = 15000, x_1 = 3270, x_2 = 14, x_3 = 7, x_draw = 11709, prob = 0.0
SL-scaled-vs-Random:
East: n = 5000, x_1 = 1070, x_2 = 2, x_3 = 7, x_draw = 3921, prob = 0.0
South: n = 5000, x_1 = 1136, x_2 = 4, x_3 = 0, x_draw = 3860, prob = 0.0
West: n = 5000, x_1 = 1001, x_2 = 10, x_3 = 3, x_draw = 3986, prob = 0.0
Total: n = 15000, x_1 = 3207, x_2 = 16, x_3 = 10, x_draw = 11767, prob = 0.0
RL-vs-Random:
East: n = 4949, x_1 = 3642, x_2 = 1, x_3 = 162, x_draw = 1144, prob = 0.0
South: n = 4949, x_1 = 3560, x_2 = 4, x_3 = 171, x_draw = 1214, prob = 0.0
West: n = 4949, x_1 = 3544, x_2 = 3, x_3 = 141, x_draw = 1261, prob = 0.0
Total: n = 14847, x_1 = 10746, x_2 = 8, x_3 = 474, x_draw = 3619, prob = 0.0
```

[14]:
```python
for action in 'SL-vs-2RL', 'RL-vs-2SL':
    print(action + ':')
    for wind in 'East', 'South', 'West', 'Total':
        p_1 = win_rates['SL-vs-SL'][wind]['1st_place_rate']
        p_2 = win_rates['SL-vs-SL'][wind]['2nd_place_rate']
        p_3 = win_rates['SL-vs-SL'][wind]['3rd_place_rate']
        p_draw = win_rates['SL-vs-SL'][wind]['draw_rate']

        x_1 = summary[action][wind]['1st_place_count']
        x_2 = summary[action][wind]['2nd_place_count']
        x_3 = summary[action][wind]['3rd_place_count']
        x_draw = summary[action][wind]['draw_count']

        n = x_1 + x_2 + x_3 + x_draw
        multinom = multinomial(n, [p_1, p_2, p_3, p_draw])

        cdf = 0
        for i in range(x_1 + 1, n + 1):
            for j in range(0, min(n - i + 1, x_3)):
                for k in range(0, n - i - j + 1):
                    cdf += multinom.pmf([i, k, j, n - i - j - k])
        for k in range(0, x_2):
            cdf += multinom.pmf([x_1, k, x_3, n - i - j - k])
        cdf += multinom.pmf([x_1, x_2, x_3, x_draw])

        print(wind + ': n = ' + str(n)
              + ', x_1 = ' + str(x_1)
              + ', x_2 = ' + str(x_2)
              + ', x_3 = ' + str(x_3)
              + ', x_draw = ' + str(x_draw)
              + ', prob = ' + str(cdf))
```

```
SL-vs-2RL:
```

```
East: n = 4996, x_1 = 288, x_2 = 354, x_3 = 4073, x_draw = 281, prob = 1.0
South: n = 4996, x_1 = 305, x_2 = 1903, x_3 = 2466, x_draw = 322, prob = 1.0
West: n = 4996, x_1 = 245, x_2 = 1883, x_3 = 2582, x_draw = 286, prob = 1.0
Total: n = 14988, x_1 = 838, x_2 = 4140, x_3 = 9121, x_draw = 889, prob = 1.0
RL-vs-2SL:
East: n = 4993, x_1 = 2869, x_2 = 387, x_3 = 695, x_draw = 1042, prob = 0.0
South: n = 4993, x_1 = 2891, x_2 = 816, x_3 = 246, x_draw = 1040, prob = 0.0
West: n = 4993, x_1 = 2780, x_2 = 908, x_3 = 262, x_draw = 1043, prob = 0.0
Total: n = 14979, x_1 = 8540, x_2 = 2111, x_3 = 1203, x_draw = 3125, prob = 0.0
```

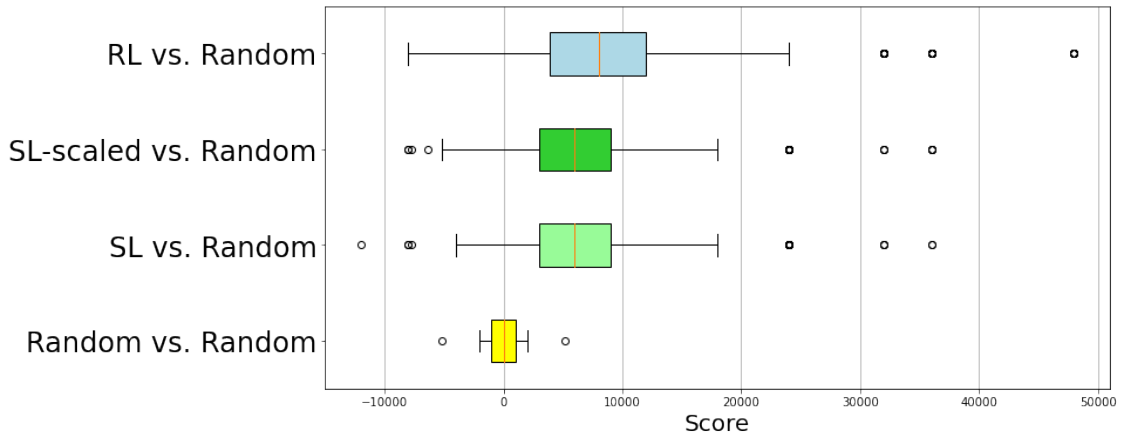
[15]:
```
plt.figure(figsize=(12, 6))

labels = ['Random vs. Random', 'SL vs. Random', 'SL-scaled vs. Random', 'RL vs.␣
↪Random']
boxplot_data = []
for label in labels:
    agent = label.replace('.', '').replace(' ', '-')
    if agent == 'Random-vs-Random':
        boxplot_data.append([scores[agent] + [0, 0, 0]])
    else:
        boxplot_data.append([
            *scores[agent]['East'][0], *scores[agent]['East'][1],␣
↪*scores[agent]['East'][2],
            *scores[agent]['South'][0], *scores[agent]['South'][1],␣
↪*scores[agent]['South'][2],
            *scores[agent]['West'][0], *scores[agent]['West'][1],␣
↪*scores[agent]['West'][2]])

plt.xlabel('Score', fontsize=20)
plt.yticks(size=24)
plt.grid(axis='x')
boxplot = plt.boxplot(boxplot_data, labels=labels, patch_artist=True,␣
↪vert=False)

colours = ['yellow', 'palegreen', 'limegreen', 'lightblue']
for box, colour in zip(boxplot['boxes'], colours):
    box.set_facecolor(colour)

plt.savefig("../../../../Dissertation/figs/boxplot-agents-vs-random.png",␣
↪bbox_inches='tight')
plt.show()
```



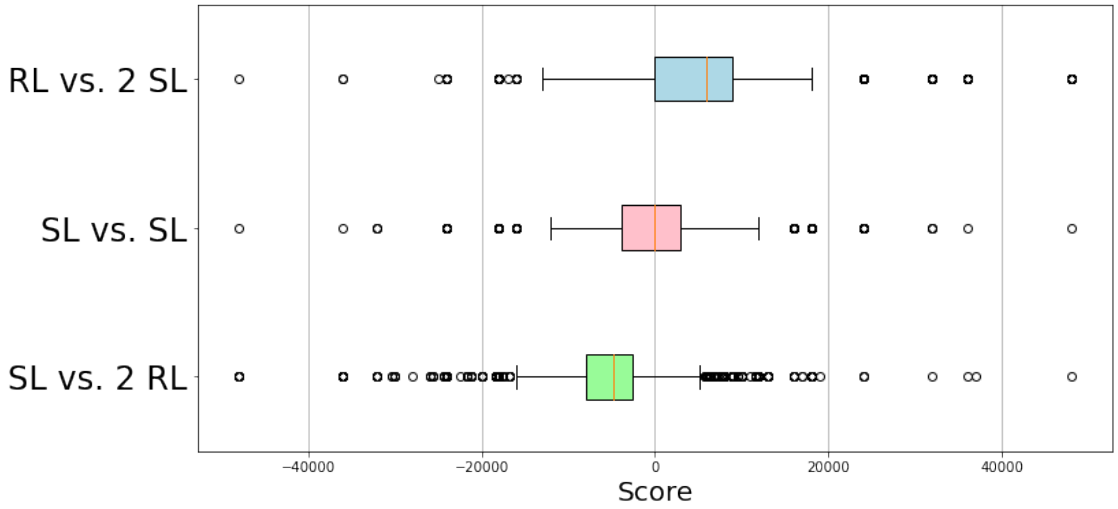
[16]:
```
plt.figure(figsize=(12, 6))

labels = ['SL vs. 2 RL', 'SL vs. SL', 'RL vs. 2 SL']
agents = ['SL-vs-2RL', 'SL-vs-SL', 'RL-vs-2SL']
boxplot_data = []
for agent in agents:
    if agent == 'Random-vs-Random':
        boxplot_data.append([scores[agent]])
    else:
        boxplot_data.append([
            *scores[agent]['East'][0], *scores[agent]['East'][1],␣
 ↪*scores[agent]['East'][2],
            *scores[agent]['South'][0], *scores[agent]['South'][1],␣
 ↪*scores[agent]['South'][2],
            *scores[agent]['West'][0], *scores[agent]['West'][1],␣
 ↪*scores[agent]['West'][2]])

plt.xlabel('Score', fontsize=20)
plt.yticks(size=24)
plt.grid(axis='x')
boxplot = plt.boxplot(boxplot_data, labels=labels, patch_artist=True,␣
 ↪vert=False)

colours = ['palegreen', 'pink', 'lightblue']
for box, colour in zip(boxplot['boxes'], colours):
    box.set_facecolor(colour)

plt.savefig("../../../../Dissertation/figs/boxplot-rl-vs-sl.png",␣
 ↪bbox_inches='tight')
plt.show()
```


RL vs. 2 SL
SL vs. SL
SL vs. 2 RL
−40000
−20000
0
20000
40000
Score