Supplementary materials

This materials contains two main parts:

1. The National Carbon Dioxide Quota (NCQ) estimation code which referred to the 'Assessing Negative Carbon Dioxide Emissions from the Perspective of a National "Fair Share" of the Remaining Global Carbon Budget: Supplementary Material | Zenodo'. https://zenodo.org/record/3257409

2. The code of annual Remaining Carbon Dioxide Quota curve and the annual Carbon Dioxide Emissions Rate curve under three scenarios

Notes: These two codes were initially run on Jupyter Notebook, if needed, it could run on other Software but 'numpy' and 'matplotlib.pyplot' libraries should be installed first. And some of these codes may need to be modified before use because of the blank or space distance issues.

Code 1:

Part 1: # Definition of basic functions

```
import math
def rate_from_quota(x_0, Q):
    R = -(x_0/Q)
```

```
    return R
def quota_from_rate(x_0, R):
    Q = -(x_0/R)
    return Q

def fixed_term_quota_from_rate(x_0, R, start_year, end_year):
    n = end_year - start_year + 1 # n is inclusive of both start and end year
    if (R == 0.0):
        Q_n = x_0 * n # Flatline
    else:
        Q_n = x_0 * (((R + 1.0)**n - 1.0)/R)
    return Q_n

def rate_from_two_points(year_0, x_0, year_n, x_n):
    n = year_n - year_0 + 1 # n is inclusive of both year_0 and year_n
    R = math.exp((1.0/n)*math.log(x_n/x_0)) - 1.0
    return R

# Test

# print("Test 1: rate_from_quota()")
# x_0 = 80.0
# Q = 1000.0
# print("In: x_0 = %5.3f" % x_0)
# print("In: Q = %5.2f" % Q)
# R = rate_from_quota(x_0, Q)
# print("Out: R = %4.3f%%" % (R*100.0))
x_0 = 80.0
R = -0.1213
print("Test 2: quota_from_rate()")
print("In: x_0 = %5.3f" % x_0)
print("In: R = %4.3f%%" % (R*100.0))
Q = quota_from_rate(x_0, R)
print("Out: Q = %5.2f" % Q)

print("Test 3: fixed_term_quota_from_rate()")
print("In: x_0 = %5.3f" % x_0)
print("In: R = %4.3f%%" % (R*100.0))
start_year = 2015
end_year = 2030
print("In: start_year = %4d" % start_year)
print("In: end_year = %4d" % end_year)
Q_n = fixed_term_quota_from_rate(x_0, R, start_year, end_year)
print("Out: Q_n = %5.2f" % Q_n)
```

```
print("Test 4: rate_from_two_points()")
year_0 = 2015
year_n = 2030
x_n = 11.5 # -80% over the pathway
print("In: year_0 = %4d" % year_0)
print("In: x_0 = %5.3f" % x_0)
print("In: year_n = %4d" % year_n)
print("In: x_n = %5.3f" % x_n)
R = rate_from_two_points(year_0, x_0, year_n, x_n)
print("Out: R = %4.3f%%" % (R*100.0))
```

```
Test 2: quota_from_rate()
In: x_0 = 80.000
In: R = -12.130%
Out: Q = 659.52
Test 3: fixed_term_quota_from_rate()
In: x_0 = 80.000
In: R = -12.130%
In: start_year = 2015
In: end_year = 2030
Out: Q_n = 576.22
Test 4: rate_from_two_points()
In: year_0 = 2015
In: x_0 = 80.000
In: year_n = 2030
In: x_n = 11.500
Out: R = -11.417%
```

Part 2: # GCB definition and emissions definition

```
GCB_from_2018_to_NettZero = {}
GCB_from_2018_to_NettZero['mid'] = 1.170E+06
  # SR15, Table 2.2 (p. 108), <+2C @66%, central estimate, MtCO2
  # Cumulative from 2018 to time of nett zero emission rate
GCB_range_SR15 = 0.5
  # SR15 (p. 107) aggregated GCB fractional uncertainty (±)
GCB_from_2018_to_NettZero['low'] = (GCB_from_2018_to_NettZero['mid'] * (1.0 -
GCB_range_SR15))
GCB_from_2018_to_NettZero['high'] = (GCB_from_2018_to_NettZero['mid'] * (1.0
+ GCB_range_SR15))

GCB_NettZero_to_2100 = -100.0E+03
  # SR15, p. 107, additional order-of-magnitude cumulative removals (negati
  # from time of nett zero emission rate to 2100 (to stabilize temperature
  # continuing Earth system feedbacks)
```

```
GCP_emissions_global_FFI_Gt = {'2015': 9.68,'2016': 9.74, '2017': 9.87 }
  # Global Carbon Project, GtC/yr
GCP_emissions_global_LU_Gt = {'2015': 1.62,'2016': 1.30, '2017': 1.39 }
  # Global Carbon Project, GtC/yr
GtC_MtCO2_multiplier = 3.664e3

GCP_emissions_global = {} # MtCO2/yr
for key in ['2015', '2016', '2017'] :
        GCP_emissions_global[key] = (GCP_emissions_global_FFI_Gt[key] +
        GCP_emissions_global_LU_Gt[key]) * GtC_MtCO2_multiplier

GCP_emissions_global_2015_to_2017 = sum(GCP_emissions_global.values())

GCB = {}
  # Global Carbon Budget, 2015-2100 (MtCO2, nett FFI+LU)
  # Combine components for 2015-2017 (historical) + 2018 to time of nett ze
  # emissions + time of nett zero emissions to 2100. Given relatively high
  # uncertainty, we round to 1e4 (MtCO2)
for key in ['low', 'mid', 'high'] :
        GCB_raw = (GCB_from_2018_to_NettZero[key] +
        GCP_emissions_global_2015_to_2017 + GCB_NettZero_to_2100)
        GCB[key] = round(GCB_raw / 1e4) * 1e4

emissions_Israel = {
        '1990': 40.000,
        '2015': 80.000,
        '2030_BAU': 105.512,
        '2030_INT': 81.65} # All MtCO2/yr, nett FFI+LU

emissions_Israel_share = {'2015':
emissions_Israel['2015']/GCP_emissions_global['2015']}

pop_global = {'2015': 7.38E+09} # UNEP
pop_Israel_2015 = 7.98E+06
print(pop_Israel_2015)
  # Linear interpolation

pop_Israel_share = {'2015': pop_Israel_2015/pop_global['2015']}

print(GCB)
```

```
7980000.0
{'low': 610000.0, 'mid': 1190000.0, 'high': 1780000.0}
```

Part 3: # define the visual method

```
from IPython.display import (display, display_html, display_png, display_svg)

class Scenario:
    def __init__(self):
        self.dict = {}

    def __repr__(self):
        slist = []
        #slist.append('[repr] ')
        dict = self.dict
        keys = dict.keys()
        if 'name' in keys:
            slist.append(dict['name'])
        if 'Q_2030' in keys:
            slist.append(', Q_2030: %.f' % dict['Q_2030'])
        if 'Q' in keys:
            slist.append(', Q: %.f' % dict['Q'])
        if 'Q_per_capita' in keys:
            slist.append(', Q_per_capita: %.f' % dict['Q_per_capita'])
        if 'R' in keys:
            slist.append(', R: %+3.2f%%' % (dict['R']*100.0))
        return ''.join(slist)

    def _repr_html_(self):
        slist = ['<tr>']
        #slist.append('<p><strong>[repr_html]</strong> ')
        dict = self.dict
        keys = dict.keys()
        value = ''
        if 'name' in keys:
            value = dict['name']
        slist.append('<td>%s</td>' % value)
        value = ''
        if 'Q_2030' in keys:
            value = '%.f' % dict['Q_2030']
        slist.append('<td>%s</td>' % value)
        value = ''
        if 'Q' in keys:
            value = '%.f' % dict['Q']
```

```
            slist.append('<td>%s</td>' % value)
            value = ''
            if 'Q_per_capita' in keys:
                value = '%.f' % dict['Q_per_capita']
            slist.append('<td>%s</td>' % value)
            value = ''
            if 'R' in keys:
                value = '%+3.2f%%' % (dict['R']*100.0)
            slist.append('<td>%s</td>' % value)
            slist.append('</tr>')
            return ''.join(slist)

class ScenarioSet:
    def __init__(self):
        self.list = []

    def __repr__(self):
        slist = []
        for s in self.list:
            slist.append(s.__repr__())
        #return ''.join(slist)
        return 'hello world'

    def _repr_html_(self):
        slist = []
        slist.append('<table>')
        slist.append('<tr>')
        slist.append('<th>Scenario</th>')
        slist.append('<th>Quota [2015,2030]</th>')
        slist.append('<th>Quota</th>')
        slist.append('<th>Quota per capita</th>')
        slist.append('<th>R</th>')
        slist.append('</tr>')
        for s in self.list:
            slist.append(s._repr_html_())
        slist.append('</table>')
        return ''.join(slist)

all_scenarios = ScenarioSet() # Global container

def clear_all_scenarios():
    all_scenarios.list = []
```

Part 4: # calculate the Israel's national quota under(low, mid, high) GCB and (pop, blend, inertia) scenario

```
# M1 Raupach

# National CO₂   Quota derived from Global Carbon Budget (GCB)

def raupach_quotas_from_GCB(GCB_value):
    quotas = {}
    quotas['pop'] = pop_Israel_share['2015'] * GCB_value
    quotas['inertia'] = emissions_Israel_share['2015'] * GCB_value
    blend_w = 0.5
    quotas['blend'] = (quotas['pop'] * blend_w) + (quotas['inertia'] * (1-blend_w))
    return quotas

def raupach_scenarios():
    scenarios = ScenarioSet()
    for GCB_name in ('low', 'mid', 'high'):
        GCB_value = GCB[GCB_name]
        quotas = raupach_quotas_from_GCB(GCB_value)
        for sharing in ('pop', 'blend', 'inertia'):
            s=Scenario()
            d=s.dict
            d['method'] = 'raupach'
            d['GCB_name'] = GCB_name
            d['GCB_value'] = GCB_value
            d['sharing'] = sharing
            d['name'] = d['method'] + '-' + GCB_name + 'GCB-' + sharing
            d['Q'] = quotas[sharing]
 # Convert from MtCO₂ to tCO₂
            d['Q_per_capita'] = (d['Q']/pop_Israel_2015) * 1e6
            d['R'] = rate_from_quota(emissions_Israel['2015'], d['Q'])
            d['Q_2030'] = fixed_term_quota_from_rate(
                emissions_Israel['2015'], d['R'], 2015, 2030)
            scenarios.list.append(s)
    return scenarios
clear_all_scenarios()
r_scenarios = raupach_scenarios()
display_html(r_scenarios)
all_scenarios.list.extend(r_scenarios.list)
#display_html(all_scenarios)
```

Scenario	Quota [2015,2030]	Quota	Quota per capita	R
raupach-lowGCB-pop	576	660	83	-12.13%
raupach-lowGCB-blend	705	919	115	-8.70%
raupach-lowGCB-inertia	796	1179	148	-6.79%
raupach-midGCB-pop	826	1287	161	-6.22%
raupach-midGCB-blend	929	1793	225	-4.46%
raupach-midGCB-inertia	995	2299	288	-3.48%
raupach-highGCB-pop	949	1925	241	-4.16%
raupach-highGCB-blend	1030	2682	336	-2.98%
raupach-highGCB-inertia	1079	3439	431	-2.33%

Part 5: # Method 2: "Projections" (BAU,INT)

```
def projections_scenarios():
    scenarios = ScenarioSet()
    year_0 = 2015
    x_0 = emissions_Israel['2015']
    year_n = 2030
    x_projected = {'BAU': emissions_Israel['2030_BAU'],
    'INT':emissions_Israel['2030_INT']}
    for ambition_name in ('BAU', 'INT'):
        s=Scenario()
        d=s.dict
        d['method'] = 'projections'
        d['ambition_name'] = ambition_name
        d['name'] = d['method'] + '-' + ambition_name
        d['emissions_2030'] = x_projected[ambition_name]
        x_n = d['emissions_2030']
        d['R'] = rate_from_two_points(year_0, x_0, year_n, x_n)
        d['Q_2030'] = fixed_term_quota_from_rate(
            emissions_Israel['2015'], d['R'], 2015, 2030)
        d['Q_per_capita'] = (d['Q_2030']/pop_Israel_2015) * 1e6 # Convert from
MtCO2 to tCO2
        scenarios.list.append(s)
    return scenarios
#clear_all_scenarios()
prj_scenarios = projections_scenarios()
display_html(prj_scenarios)
all_scenarios.list.extend(prj_scenarios.list)
#display_html(all_scenarios)
```

Scenario	Quota [2015,2030]	Quota	Quota per capita	R
projections-BAU	1462		183	+1.75%
projections-INT	1292		162	+0.13%

display_html(all_scenarios)

Scenario	Quota [2015,2030]	Quota	Quota per capita	R
raupach-lowGCB-pop	576	660	83	-12.13%
raupach-lowGCB-blend	705	919	115	-8.70%
raupach-lowGCB-inertia	796	1179	148	-6.79%
raupach-midGCB-pop	826	1287	161	-6.22%
raupach-midGCB-blend	929	1793	225	-4.46%
raupach-midGCB-inertia	995	2299	288	-3.48%
raupach-highGCB-pop	949	1925	241	-4.16%
raupach-highGCB-blend	1030	2682	336	-2.98%
raupach-highGCB-inertia	1079	3439	431	-2.33%
projections-BAU	1462		183	+1.75%
projections-INT	1292		162	+0.13%

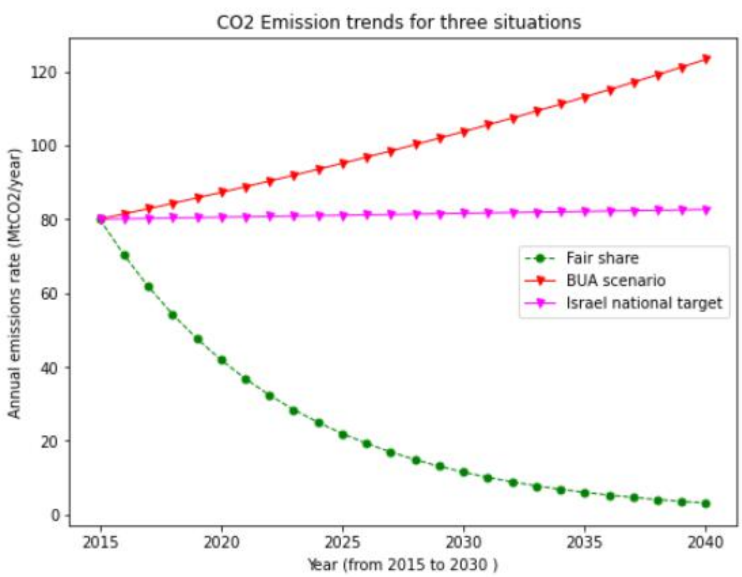
Code 2:

Part 1: # plot the Annual emissions rate curve under three scenarios

```
# fair share: 2015: 80MtCO2-----------R = -12.13%-----------------2030: 11.50MtCO2
# BUA: 2015: 80MtCO2---------------R = 1.744%    -----------------2030: 105.5MtCO2
# INT: 2015: 80MtCO2---------------R = 0.128%    -----------------2030: 81.65MtCO2

import numpy as np
import matplotlib.pyplot as plt

# define a period of time from 2015 to 2040
# define the initial annual emission for 2015 y_0 = 80
n = np.linspace(2015, 2040, 26)
y_0 = 80.0

# fair-share scenario
R = -0.1213
r = R + 1
```

```
y_n = y_0*(r**(n-2015))

# BAU scenario
R_1 = 0.01744
r_1 = R_1 + 1
y_1_n = y_0*(r_1**(n-2015))

# INT scenario
R_2 = 0.00128
r_2 = R_2 + 1
y_2_n = y_0*(r_2**(n-2015))

# test the annual emission for 2030 under BAU scenario
y_2030 = y_0*(r_1**16)
print(y_2030)

# draw the curve
plt.figure(figsize=(8,6))
plt.plot(n,y_n,color='green', marker='o', linestyle='dashed', linewidth=1,
markersize=5)
plt.plot(n,y_1_n,color='red', marker='v', linestyle='solid', linewidth=1, markersize=5)
plt.plot(n,y_2_n,color='magenta', marker='v', linestyle='solid', linewidth=1,
markersize=5)

plt.xlabel("Year (from 2015 to 2030 )")
plt.ylabel("Annual emissions rate (MtCO2/year)")
plt.legend(('Fair share', 'BUA scenario', 'Israel national target'), loc='center right')
plt.title("CO2 Emission trends for three situations ")
plt.show()
```

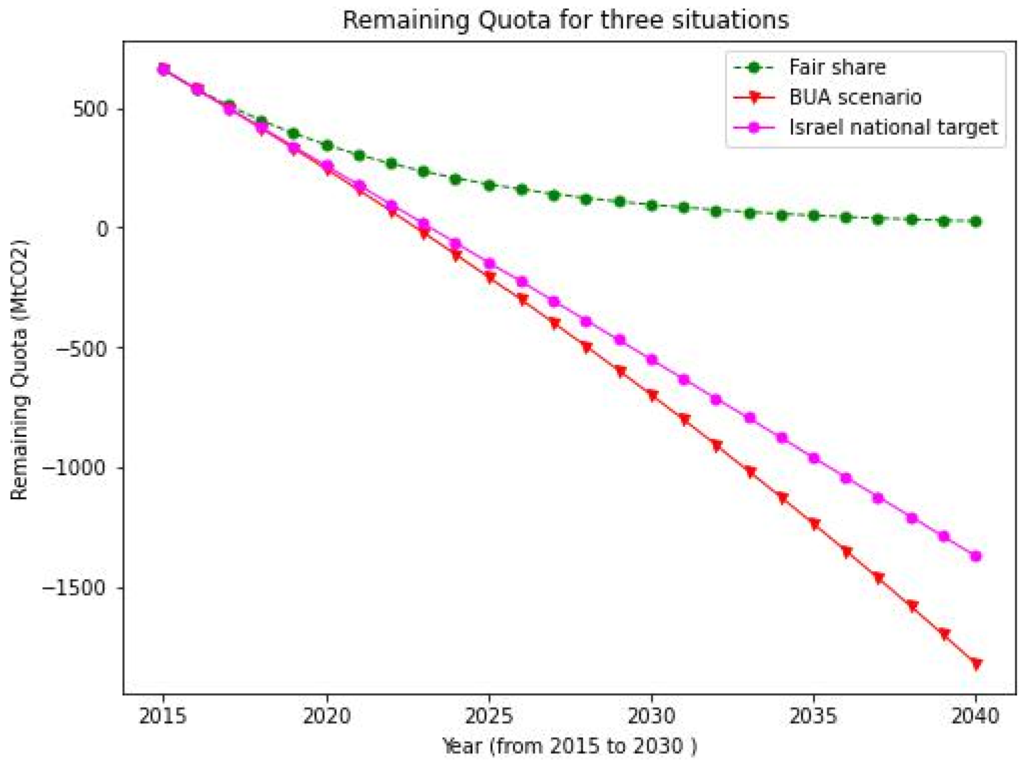
```
105.49476504571766
```



Part 2: # plot the Annual remaining quotas curve under three scenarios

```
# fair share: 2015: 80MtCO2------------R = -12.13%-----------------2030: 11.50MtCO2
# BUA: 2015: 80MtCO2---------------R = 1.744%    -----------------2030: 105.5MtCO2
# INT: 2015: 80MtCO2---------------R = 0.128%    -----------------2030: 81.65MtCO2
# Q_2015 = 660MtCO2

import numpy as np
import matplotlib.pyplot as plt

# define the intial quota for the year 2015 Q_2015
# define the initial annual emission for the year 2015 y_0
# defien the period of time from 2015 to 2040
Q_2015 = 660
y_0 = 80
n = np.linspace(2015, 2040, 26)

# fair-share
R_0 = -0.1213
Q_n_0 = Q_2015 - y_0 * (((R_0 + 1.0)**(n-2015) - 1.0)/R_0)

# BAU
R_1 = 0.01744
Q_n_1 = Q_2015 - y_0 * (((R_1 + 1.0)**(n-2015) - 1.0)/R_1)

# INT
R_2 = 0.00128
Q_n_2 = Q_2015 - y_0 * (((R_2 + 1.0)**(n-2015) - 1.0)/R_2)

# draw the curve
plt.figure(figsize=(8,6))
plt.plot(n,Q_n_0,color='green', marker='o', linestyle='dashed', linewidth=1,
markersize=5)
plt.plot(n,Q_n_1,color='red', marker='v', linestyle='solid', linewidth=1, markersize=5)
plt.plot(n,Q_n_2,color='magenta', marker='o', linestyle='solid', linewidth=1,
markersize=5)

plt.xlabel("Year (from 2015 to 2030 )")
plt.ylabel("Remaining Quota (MtCO2)")
plt.legend(('Fair share', 'BUA scenario', 'Israel national target'), loc='upper right')
plt.title("Remaining Quota for three situations ")
plt.show()
```

# test the remaining quotas for the year 2030 under the fair-share, BAU, INT scenario
print("Quota in 2030 for fair share: %5.3f" % (Q_2015 - y_0 * (((R_0 + 1.0)**(2030-2015) - 1.0)/R_0)))
print("Quota in 2030 for BUA scenario: %5.3f" % (Q_2015 - y_0 * (((R_1 + 1.0)**(2030-2015) - 1.0)/R_1)))
print("Quota in 2030 for Israel national target: %5.3f" % (Q_2015 - y_0 * (((R_2 + 1.0)**(2030-2015) - 1.0)/R_2)))




```
Quota in 2030 for fair share: 95.285
Quota in 2030 for BUA scenario: -698.169
Quota in 2030 for Israel national target: -550.812
```