Jade's Part

April 27, 2022

[1]:
```
!pip install gspread --upgrade
!pip install -r requirements.txt
```

```
Requirement already up-to-date: gspread in /opt/conda/lib/python3.9/site-
packages (5.3.2)
Requirement already satisfied, skipping upgrade: google-auth-oauthlib>=0.4.1 in
/opt/conda/lib/python3.9/site-packages (from gspread) (0.4.5)
Requirement already satisfied, skipping upgrade: google-auth>=1.12.0 in
/opt/conda/lib/python3.9/site-packages (from gspread) (2.6.2)
Requirement already satisfied, skipping upgrade: requests-oauthlib>=0.7.0 in
/opt/conda/lib/python3.9/site-packages (from google-auth-
oauthlib>=0.4.1->gspread) (1.3.1)
Requirement already satisfied, skipping upgrade: pyasn1-modules>=0.2.1 in
/opt/conda/lib/python3.9/site-packages (from google-auth>=1.12.0->gspread)
(0.2.8)
Requirement already satisfied, skipping upgrade: rsa<5,>=3.1.4; python_version
>= "3.6" in /opt/conda/lib/python3.9/site-packages (from google-
auth>=1.12.0->gspread) (4.8)
Requirement already satisfied, skipping upgrade: six>=1.9.0 in
/opt/conda/lib/python3.9/site-packages (from google-auth>=1.12.0->gspread)
(1.16.0)
Requirement already satisfied, skipping upgrade: cachetools<6.0,>=2.0.0 in
/opt/conda/lib/python3.9/site-packages (from google-auth>=1.12.0->gspread)
(5.0.0)
Requirement already satisfied, skipping upgrade: oauthlib>=3.0.0 in
/opt/conda/lib/python3.9/site-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib>=0.4.1->gspread) (3.2.0)
Requirement already satisfied, skipping upgrade: requests>=2.0.0 in
/opt/conda/lib/python3.9/site-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib>=0.4.1->gspread) (2.26.0)
Requirement already satisfied, skipping upgrade: pyasn1<0.5.0,>=0.4.6 in
/opt/conda/lib/python3.9/site-packages (from pyasn1-modules>=0.2.1->google-
auth>=1.12.0->gspread) (0.4.8)
Requirement already satisfied, skipping upgrade: certifi>=2017.4.17 in
/opt/conda/lib/python3.9/site-packages (from requests>=2.0.0->requests-
oauthlib>=0.7.0->google-auth-oauthlib>=0.4.1->gspread) (2019.11.28)
Requirement already satisfied, skipping upgrade: charset-normalizer~=2.0.0;
python_version >= "3" in /opt/conda/lib/python3.9/site-packages (from
```

```
requests>=2.0.0->requests-oauthlib>=0.7.0->google-auth-oauthlib>=0.4.1->gspread)
(2.0.0)
Requirement already satisfied, skipping upgrade: urllib3<1.27,>=1.21.1 in
/opt/conda/lib/python3.9/site-packages (from requests>=2.0.0->requests-
oauthlib>=0.7.0->google-auth-oauthlib>=0.4.1->gspread) (1.25.7)
Requirement already satisfied, skipping upgrade: idna<4,>=2.5; python_version >=
"3" in /opt/conda/lib/python3.9/site-packages (from requests>=2.0.0->requests-
oauthlib>=0.7.0->google-auth-oauthlib>=0.4.1->gspread) (2.8)
Requirement already satisfied: CFEDemands>=0.4.1 in
/opt/conda/lib/python3.9/site-packages (from -r requirements.txt (line 5))
(0.4.1)
Requirement already satisfied: gspread>=4.0.1 in /opt/conda/lib/python3.9/site-
packages (from -r requirements.txt (line 8)) (5.3.2)
Requirement already satisfied: matplotlib>=3.3.4 in
/opt/conda/lib/python3.9/site-packages (from -r requirements.txt (line 11))
(3.4.3)
Collecting numpy>=1.22.2
  Using cached
numpy-1.22.3-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (16.8 MB)
Requirement already satisfied: oauth2client>=4.1.3 in
/opt/conda/lib/python3.9/site-packages (from -r requirements.txt (line 18))
(4.1.3)
Collecting pandas>=1.4.1
  Using cached
pandas-1.4.2-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (11.7 MB)
Collecting plotly>=5.5.0
  Using cached plotly-5.7.0-py2.py3-none-any.whl (28.8 MB)
Requirement already satisfied: eep153_tools>=0.11 in
/opt/conda/lib/python3.9/site-packages (from -r requirements.txt (line 28))
(0.11)
Requirement already satisfied: gnupg in /opt/conda/lib/python3.9/site-packages
(from -r requirements.txt (line 29)) (2.3.1)
Requirement already satisfied: ConsumerDemands in /opt/conda/lib/python3.9/site-
packages (from -r requirements.txt (line 31)) (0.3.dev0)
Requirement already satisfied: google-auth>=1.12.0 in
/opt/conda/lib/python3.9/site-packages (from gspread>=4.0.1->-r requirements.txt
(line 8)) (2.6.2)
Requirement already satisfied: google-auth-oauthlib>=0.4.1 in
/opt/conda/lib/python3.9/site-packages (from gspread>=4.0.1->-r requirements.txt
(line 8)) (0.4.5)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.9/site-
packages (from matplotlib>=3.3.4->-r requirements.txt (line 11)) (0.11.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
/opt/conda/lib/python3.9/site-packages (from matplotlib>=3.3.4->-r
requirements.txt (line 11)) (1.4.2)
Requirement already satisfied: pyparsing>=2.2.1 in
/opt/conda/lib/python3.9/site-packages (from matplotlib>=3.3.4->-r
requirements.txt (line 11)) (3.0.7)
```

```
Requirement already satisfied: python-dateutil>=2.7 in
/opt/conda/lib/python3.9/site-packages (from matplotlib>=3.3.4->-r
requirements.txt (line 11)) (2.8.0)
Requirement already satisfied: pillow>=6.2.0 in /opt/conda/lib/python3.9/site-
packages (from matplotlib>=3.3.4->-r requirements.txt (line 11)) (8.3.2)
Requirement already satisfied: pyasn1-modules>=0.0.5 in
/opt/conda/lib/python3.9/site-packages (from oauth2client>=4.1.3->-r
requirements.txt (line 18)) (0.2.8)
Requirement already satisfied: six>=1.6.1 in /opt/conda/lib/python3.9/site-
packages (from oauth2client>=4.1.3->-r requirements.txt (line 18)) (1.16.0)
Requirement already satisfied: rsa>=3.1.4 in /opt/conda/lib/python3.9/site-
packages (from oauth2client>=4.1.3->-r requirements.txt (line 18)) (4.8)
Requirement already satisfied: httplib2>=0.9.1 in /opt/conda/lib/python3.9/site-
packages (from oauth2client>=4.1.3->-r requirements.txt (line 18)) (0.20.4)
Requirement already satisfied: pyasn1>=0.1.7 in /opt/conda/lib/python3.9/site-
packages (from oauth2client>=4.1.3->-r requirements.txt (line 18)) (0.4.8)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.9/site-
packages (from pandas>=1.4.1->-r requirements.txt (line 23)) (2021.1)
Requirement already satisfied: tenacity>=6.2.0 in /opt/conda/lib/python3.9/site-
packages (from plotly>=5.5.0->-r requirements.txt (line 26)) (8.0.1)
Requirement already satisfied: psutil>=1.2.1 in /opt/conda/lib/python3.9/site-
packages (from gnupg->-r requirements.txt (line 29)) (5.9.0)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in
/opt/conda/lib/python3.9/site-packages (from google-
auth>=1.12.0->gspread>=4.0.1->-r requirements.txt (line 8)) (5.0.0)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
/opt/conda/lib/python3.9/site-packages (from google-auth-
oauthlib>=0.4.1->gspread>=4.0.1->-r requirements.txt (line 8)) (1.3.1)
Requirement already satisfied: requests>=2.0.0 in /opt/conda/lib/python3.9/site-
packages (from requests-oauthlib>=0.7.0->google-auth-
oauthlib>=0.4.1->gspread>=4.0.1->-r requirements.txt (line 8)) (2.26.0)
Requirement already satisfied: oauthlib>=3.0.0 in /opt/conda/lib/python3.9/site-
packages (from requests-oauthlib>=0.7.0->google-auth-
oauthlib>=0.4.1->gspread>=4.0.1->-r requirements.txt (line 8)) (3.2.0)
Requirement already satisfied: charset-normalizer~=2.0.0; python_version >= "3"
in /opt/conda/lib/python3.9/site-packages (from requests>=2.0.0->requests-
oauthlib>=0.7.0->google-auth-oauthlib>=0.4.1->gspread>=4.0.1->-r
requirements.txt (line 8)) (2.0.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/opt/conda/lib/python3.9/site-packages (from requests>=2.0.0->requests-
oauthlib>=0.7.0->google-auth-oauthlib>=0.4.1->gspread>=4.0.1->-r
requirements.txt (line 8)) (1.25.7)
Requirement already satisfied: idna<4,>=2.5; python_version >= "3" in
/opt/conda/lib/python3.9/site-packages (from requests>=2.0.0->requests-
oauthlib>=0.7.0->google-auth-oauthlib>=0.4.1->gspread>=4.0.1->-r
requirements.txt (line 8)) (2.8)
Requirement already satisfied: certifi>=2017.4.17 in
/opt/conda/lib/python3.9/site-packages (from requests>=2.0.0->requests-
```

```
oauthlib>=0.7.0->google-auth-oauthlib>=0.4.1->gspread>=4.0.1->-r
requirements.txt (line 8)) (2019.11.28)
Installing collected packages: numpy, pandas, plotly
  Attempting uninstall: numpy
    Found existing installation: numpy 1.21.5
    Uninstalling numpy-1.21.5:
      Successfully uninstalled numpy-1.21.5
  Attempting uninstall: pandas
    Found existing installation: pandas 1.3.5
    Uninstalling pandas-1.3.5:
      Successfully uninstalled pandas-1.3.5
  Attempting uninstall: plotly
    Found existing installation: plotly 5.2.1
    Uninstalling plotly-5.2.1:
      Successfully uninstalled plotly-5.2.1
```

```
ERROR: After October 2020 you may experience errors when installing or
updating packages. This is because pip will change the way that it resolves
dependency conflicts.
We recommend you use --use-feature=2020-resolver to test your packages with the
new resolver before it becomes the default.
tensorflow 2.6.3 requires h5py~=3.1.0, but you'll have h5py 3.3.0 which is
incompatible.
tensorflow 2.6.3 requires numpy~=1.19.2, but you'll have numpy 1.22.3 which is
incompatible.
tensorflow 2.6.3 requires six~=1.15.0, but you'll have six 1.16.0 which is
incompatible.
tensorboard 2.6.0 requires google-auth<2,>=1.6.3, but you'll have google-auth
2.6.2 which is incompatible.
pysal 2.5.0 requires urllib3>=1.26, but you'll have urllib3 1.25.7 which is
incompatible.
pynwb 1.5.1 requires h5py<3,>=2.9, but you'll have h5py 3.3.0 which is
incompatible.
pynwb 1.5.1 requires hdmf<3,>=2.5.6, but you'll have hdmf 2.4.0 which is
incompatible.
pynwb 1.5.1 requires numpy<1.21,>=1.16, but you'll have numpy 1.22.3 which is
incompatible.
pandas 1.4.2 requires python-dateutil>=2.8.1, but you'll have python-dateutil
2.8.0 which is incompatible.
numba 0.55.1 requires numpy<1.22,>=1.18, but you'll have numpy 1.22.3 which is
incompatible.
hdmf 2.4.0 requires h5py<3,>=2.9, but you'll have h5py 3.3.0 which is
incompatible.
hdmf 2.4.0 requires jsonschema<4,>=2.6.0, but you'll have jsonschema 4.4.0 which
is incompatible.
hdmf 2.4.0 requires numpy<1.19.4,>=1.16, but you'll have numpy 1.22.3 which is
incompatible.
fenics-dolfin 2019.1.0 requires pybind11==2.2.4, but you'll have pybind11 2.8.1
which is incompatible.
fancyimpute 0.6.0 requires keras==2.4.3, but you'll have keras 2.6.0 which is
incompatible.
fancyimpute 0.6.0 requires numpy==1.19.5, but you'll have numpy 1.22.3 which is
```

```
Successfully installed numpy-1.22.3 pandas-1.4.2 plotly-5.7.0
```

0.0.1 From Sheet to DataFrame

We begin by defining a dictionary that contains the spreadsheet key.

[2]:
```
nigeria_data = '17L5cDhXRLNAckP3JvBLTLSYIguFqP2ebMvQLH96c0n4'
nigeria_production = '1kG_fVBmj9EEF9LOwxN30HBxkQENOoWeQjVPYzMJe3b4-8DA'
nigeria_consumption = '1kG_fVBmj9EEF9LOwxN30HBxkQENOoWeQjVPYzMJe3b4'
```

With the spreadsheet defined, grab it and define a couple of dataframes.

[3]:
```
import pandas as pd
import numpy as np
import sys
from eep153_tools.sheets import read_sheets

expend = read_sheets(nigeria_data,sheet='Expenditures')
expend.columns.name = 'i'

# Change 'ICRISAT' to key of your own sheet in Sheets, above
hh_char = read_sheets(nigeria_data,sheet="HH Characteristics")
hh_char.columns.name = 'k'

# Assume a single market: (Comment this out to make each village its own market)
hh_char['m'] = 1
expend['m'] = 1

# x may have duplicate columns
expend = expend.groupby('i',axis=1).sum()
expend = expend.apply(lambda x: pd.to_numeric(x,errors='coerce'))
expend = expend.replace(0,np.nan)

# Take logs of expenditures; call this y
log_expend = np.log(expend.set_index(['j','t','m']))

hh_char.set_index(['j','t','m'],inplace=True)
```

```
Key available for students@eep153.iam.gserviceaccount.com.
Key available for students@eep153.iam.gserviceaccount.com.
```

Sort the new Data Frame in order to group by year.

[4]:
```
expend = expend.set_index(['t','j','m']).sort_index()
expend = expend.replace(0.0,np.nan) # Replace zeroes with np.nan
expend
```

[4]:
```
i              (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
t   j      m
```

```
2010 10001  1                     NaN              280.0         NaN     NaN
     10002  1                     NaN              280.0         NaN     NaN
     10003  1                     NaN              180.0         NaN     NaN
     10004  1                     NaN              180.0         NaN     NaN
     10006  1                     NaN                NaN         NaN     NaN
…                                 …                …          …     …
2018 379146 1                     NaN             1100.0         NaN     NaN
     379148 1                   100.0                NaN         NaN     NaN
     379151 1                     NaN              900.0         NaN     NaN
     379154 1                   200.0             1200.0         NaN     NaN
     379155 1                   100.0              950.0         NaN     NaN

i              Avocado pear  Baby milk powder  Bananas    Beef  \
t    j      m
2010 10001  1           NaN               NaN    200.0   500.0
     10002  1           NaN               NaN    180.0  1200.0
     10003  1           NaN               NaN    100.0   500.0
     10004  1           NaN               NaN    100.0   500.0
     10006  1           NaN               NaN    300.0   300.0
…                       …                 …      …      …
2018 379146 1           NaN               NaN      NaN     NaN
     379148 1           NaN               NaN      NaN   700.0
     379151 1           NaN               NaN    500.0     NaN
     379154 1           NaN               NaN      NaN  1300.0
     379155 1           NaN               NaN      NaN  1400.0

i              Beer (local and imported)  Biscuits  …  Sweet Potatoes  \
t    j      m                                       …
2010 10001  1                      540.0       NaN  …           150.0
     10002  1                     2000.0       NaN  …           200.0
     10003  1                        NaN       NaN  …           200.0
     10004  1                        NaN       NaN  …             NaN
     10006  1                        NaN       NaN  …             NaN
…                                 …         …  …            …
2018 379146 1                        NaN       NaN  …             NaN
     379148 1                        NaN       NaN  …             NaN
     379151 1                        NaN       NaN  …             NaN
     379154 1                        NaN       NaN  …             NaN
     379155 1                        NaN       NaN  …             NaN

i                Tea  Tomato puree(canned)  Tomatoes  Watermelon  Wheat flour  \
t    j      m
2010 10001  1    NaN                 150.0     150.0         NaN          NaN
     10002  1  140.0                 240.0     120.0         NaN          NaN
     10003  1   60.0                  90.0     100.0         NaN          NaN
     10004  1   30.0                  60.0     100.0         NaN          NaN
     10006  1  650.0                   NaN     400.0         NaN          NaN
```

```
…                 …                       …       …         …           …
2018 379146 1     NaN                     NaN       NaN       500.0          NaN
     379148 1     NaN                    60.0     200.0       150.0          NaN
     379151 1     NaN                   150.0     600.0       600.0        750.0
     379154 1     NaN                     NaN     100.0       200.0          NaN
     379155 1     NaN                     NaN     300.0       200.0          NaN

i                White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10001  1          600.0             NaN        NaN     1500.0
     10002  1          400.0             NaN        NaN     1200.0
     10003  1          100.0             NaN        NaN      400.0
     10004  1          100.0             NaN        NaN      400.0
     10006  1            NaN             NaN        NaN      400.0
…                        …             …       …       …
2018 379146 1            NaN             NaN        NaN     1800.0
     379148 1            NaN             NaN        NaN     1600.0
     379151 1         1600.0             NaN        NaN     3500.0
     379154 1            NaN             NaN        NaN      650.0
     379155 1            NaN             NaN        NaN     2500.0

[19141 rows x 124 columns]
```

1 People per Household, Total Expenditures, and Expenditures per Capita

Use the household data to calculate the number of people per household.

[5]:
```
people = hh_char.sum(axis=1)
num_people = pd.DataFrame(people)
num_people = num_people.rename(columns={0:'People per HH'})
num_people = num_people.reset_index().set_index(['t','j','m']).sort_index()
num_people
```

[5]:
```
                 People per HH
t    j      m
2010 10001  1                7
     10002  1                7
     10003  1                6
     10004  1                3
     10006  1                3
…                     …
2018 379146 1                4
     379148 1                1
     379151 1                5
     379154 1                2
```

```
      379155 1              4

[19249 rows x 1 columns]
```

Aggregate the expenditure data to find the total expenditures for each household.

```
[6]: total_expend = expend.iloc[:, 0:124].sum(axis=1)
     total = pd.DataFrame(total_expend)
     total = total.rename(columns={0:'Total Expenditures'})
     total
```

```
[6]:                 Total Expenditures
     t    j      m
     2010 10001  1             20225.0
          10002  1             15365.0
          10003  1              4675.0
          10004  1              4465.0
          10006  1              7565.0
     …                              …
     2018 379146 1             31100.0
          379148 1              6410.0
          379151 1             20540.0
          379154 1             22650.0
          379155 1              7550.0

     [19141 rows x 1 columns]
```

Add the total expenditures and people per household information to the dataframe. Then, use these columns to add an expenditures per capita column as well.

```
[7]: expend['Total Expenditures'] = total['Total Expenditures']
     expend['People per HH'] = num_people['People per HH']
     expend['Expenditures per capita'] = expend['Total Expenditures'] /
      ↪expend['People per HH']
     expend
```

```
[7]: i               (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
     t    j      m
     2010 10001  1                       NaN              280.0         NaN     NaN
          10002  1                       NaN              280.0         NaN     NaN
          10003  1                       NaN              180.0         NaN     NaN
          10004  1                       NaN              180.0         NaN     NaN
          10006  1                       NaN                NaN         NaN     NaN
     …                                 …                  …         …       …
     2018 379146 1                       NaN             1100.0         NaN     NaN
          379148 1                     100.0                NaN         NaN     NaN
          379151 1                       NaN              900.0         NaN     NaN
          379154 1                     200.0             1200.0         NaN     NaN
```

```
     379155 1                  100.0              950.0          NaN     NaN

i              Avocado pear  Baby milk powder  Bananas    Beef  \
t    j      m
2010 10001  1           NaN               NaN    200.0   500.0
     10002  1           NaN               NaN    180.0  1200.0
     10003  1           NaN               NaN    100.0   500.0
     10004  1           NaN               NaN    100.0   500.0
     10006  1           NaN               NaN    300.0   300.0
…                     …                …      …      …
2018 379146 1           NaN               NaN      NaN     NaN
     379148 1           NaN               NaN      NaN   700.0
     379151 1           NaN               NaN    500.0     NaN
     379154 1           NaN               NaN      NaN  1300.0
     379155 1           NaN               NaN      NaN  1400.0

i              Beer (local and imported)  Biscuits  …  Tomatoes  Watermelon  \
t    j      m                                       …
2010 10001  1                      540.0       NaN  …     150.0         NaN
     10002  1                     2000.0       NaN  …     120.0         NaN
     10003  1                        NaN       NaN  …     100.0         NaN
     10004  1                        NaN       NaN  …     100.0         NaN
     10006  1                        NaN       NaN  …     400.0         NaN
…                                  …       …  …       …         …
2018 379146 1                        NaN       NaN  …       NaN       500.0
     379148 1                        NaN       NaN  …     200.0       150.0
     379151 1                        NaN       NaN  …     600.0       600.0
     379154 1                        NaN       NaN  …     100.0       200.0
     379155 1                        NaN       NaN  …     300.0       200.0

i              Wheat flour  White beans  Wild game meat  Yam flour  Yam-roots  \
t    j      m
2010 10001  1          NaN        600.0             NaN        NaN     1500.0
     10002  1          NaN        400.0             NaN        NaN     1200.0
     10003  1          NaN        100.0             NaN        NaN      400.0
     10004  1          NaN        100.0             NaN        NaN      400.0
     10006  1          NaN          NaN             NaN        NaN      400.0
…                    …          …             …        …         …
2018 379146 1          NaN          NaN             NaN        NaN     1800.0
     379148 1          NaN          NaN             NaN        NaN     1600.0
     379151 1        750.0       1600.0             NaN        NaN     3500.0
     379154 1          NaN          NaN             NaN        NaN      650.0
     379155 1          NaN          NaN             NaN        NaN     2500.0

i              Total Expenditures  People per HH  Expenditures per capita
t    j      m
2010 10001  1             20225.0              7              2889.285714
```

```
      10002  1             15365.0              7             2195.000000
      10003  1              4675.0              6              779.166667
      10004  1              4465.0              3             1488.333333
      10006  1              7565.0              3             2521.666667
…                            …            …                       …
2018 379146 1              31100.0              4             7775.000000
     379148 1               6410.0              1             6410.000000
     379151 1              20540.0              5             4108.000000
     379154 1              22650.0              2            11325.000000
     379155 1               7550.0              4             1887.500000

[19141 rows x 127 columns]
```

2 Putting into Quartiles

[8]:

```python
def one_year(df, year):
    new_df = df.loc[[year]]
    return new_df

def quartiles_by_te(df, year, quartile):
    # Selecting out one year, sorting by total expenditures, then filtering out␣
 ↪the households that spent nothing
    one_year_df = one_year(df, year)
    one_year_df = one_year_df.reset_index().sort_values('Total Expenditures',␣
 ↪axis=0).replace(0,np.nan)
    one_year_df = one_year_df.dropna(axis=0, how='any', subset=['Total␣
 ↪Expenditures'])

    # Number of rows for each quartile
    total_rows = len(one_year_df)
    rows_per_qtr = round(total_rows / 4)

    # Selecting the necessary rows for each quartile
    if quartile == 1:
        return one_year_df.iloc[0:rows_per_qtr-1]
    else:
        first_row = (quartile-1) * rows_per_qtr
        last_row = (quartile * rows_per_qtr) - 1
        return one_year_df.iloc[first_row:last_row]

def quartiles_by_epc(df, year, quartile):
    # Selecting out one year, sorting by expenditures per capita, then␣
 ↪filtering out the households that spent nothing
    one_year_df = one_year(df, year)
    one_year_df = one_year_df.reset_index().sort_values('Expenditures per␣
 ↪capita', axis=0).replace(0,np.nan)
```

```
    one_year_df = one_year_df.dropna(axis=0, how='any', subset=['Expenditures␣
↪per capita'])

    # Number of rows for each quartile
    total_rows = len(one_year_df)
    rows_per_qtr = round(total_rows / 4)

    # Selecting the necessary rows for each quartile
    if quartile == 1:
        return one_year_df.iloc[0:rows_per_qtr-1]
    else:
        first_row = (quartile-1) * rows_per_qtr
        last_row = (quartile * rows_per_qtr) - 1
        return one_year_df.iloc[first_row:last_row]
```

[9]:
```
one_year(expend, 2018)
```

[9]:
```
i               (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
t    j      m
2018 10001  1                       NaN              300.0         NaN     NaN
     10002  1                       NaN                NaN         NaN     NaN
     10003  1                       NaN              300.0         NaN   400.0
     10004  1                       NaN              300.0         NaN     NaN
     10005  1                       NaN                NaN         NaN     NaN
…                                 …                 …         …       …
     379146 1                       NaN             1100.0         NaN     NaN
     379148 1                     100.0                NaN         NaN     NaN
     379151 1                       NaN              900.0         NaN     NaN
     379154 1                     200.0             1200.0         NaN     NaN
     379155 1                     100.0              950.0         NaN     NaN

i               Avocado pear  Baby milk powder  Bananas    Beef  \
t    j      m
2018 10001  1            NaN               NaN    300.0  1200.0
     10002  1            NaN               NaN      NaN  1200.0
     10003  1            NaN               NaN    300.0  2200.0
     10004  1            NaN               NaN    100.0  1000.0
     10005  1            NaN               NaN      NaN  1000.0
…                      …                …      …      …
     379146 1            NaN               NaN      NaN     NaN
     379148 1            NaN               NaN      NaN   700.0
     379151 1            NaN               NaN    500.0     NaN
     379154 1            NaN               NaN      NaN  1300.0
     379155 1            NaN               NaN      NaN  1400.0

i               Beer (local and imported)  Biscuits  …  Tomatoes  Watermelon  \
t    j      m                                        …
```

```
2018 10001  1                        NaN     150.0  …     400.0        300.0
     10002  1                        NaN     150.0  …     400.0        500.0
     10003  1                        NaN       NaN  …     400.0        300.0
     10004  1                        NaN     100.0  …     200.0          NaN
     10005  1                        NaN       NaN  …     100.0          NaN
…                                  …       …  …       …          …
     379146 1                        NaN       NaN  …       NaN        500.0
     379148 1                        NaN       NaN  …     200.0        150.0
     379151 1                        NaN       NaN  …     600.0        600.0
     379154 1                        NaN       NaN  …     100.0        200.0
     379155 1                        NaN       NaN  …     300.0        200.0

i              Wheat flour  White beans  Wild game meat  Yam flour  Yam-roots  \
t    j      m
2018 10001  1          NaN       1000.0             NaN        NaN      700.0
     10002  1          NaN          NaN             NaN        NaN        NaN
     10003  1        900.0       1000.0             NaN        NaN     1750.0
     10004  1          NaN          NaN             NaN        NaN      600.0
     10005  1          NaN          NaN             NaN        NaN        NaN
…                    …           …               …          …          …
     379146 1          NaN          NaN             NaN        NaN     1800.0
     379148 1          NaN          NaN             NaN        NaN     1600.0
     379151 1        750.0       1600.0             NaN        NaN     3500.0
     379154 1          NaN          NaN             NaN        NaN      650.0
     379155 1          NaN          NaN             NaN        NaN     2500.0

i              Total Expenditures  People per HH  Expenditures per capita
t    j      m
2018 10001  1             13200.0              6              2200.000000
     10002  1             20260.0              5              4052.000000
     10003  1             36950.0              6              6158.333333
     10004  1             18890.0              4              4722.500000
     10005  1              1600.0              6               266.666667
…                             …            …                        …
     379146 1             31100.0              4              7775.000000
     379148 1              6410.0              1              6410.000000
     379151 1             20540.0              5              4108.000000
     379154 1             22650.0              2             11325.000000
     379155 1              7550.0              4              1887.500000

[4976 rows x 127 columns]
```

Using the above functions, we were able to find the upper (fourth) and lower (first) quartiles in 2010 by total expenditures.

[10]:
```
Q1_2010_TE = quartiles_by_te(expend, 2010, 1)
Q1_2010_TE
```

```
#TE is total expenditure
```

[10]:
```
i         t       j  m  (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  \
2659   2010  200065  1                      NaN                NaN         NaN
899    2010   70086  1                      NaN                NaN         NaN
4633   2010  350063  1                      NaN                NaN         NaN
3394   2010  260068  1                      NaN                NaN         NaN
2944   2010  220071  1                      NaN                NaN         NaN
…     …       … ..                     …                 …           …
1815   2010  140071  1                      NaN                NaN         NaN
1631   2010  120055  1                      NaN                NaN         NaN
602    2010   50023  1                      NaN                NaN         NaN
4145   2010  310043  1                     50.0                NaN         NaN
213    2010   20107  1                      NaN                NaN         NaN

i      Apples  Avocado pear  Baby milk powder  Bananas  …  Tomatoes  \
2659      NaN           NaN               NaN      NaN  …       NaN
899       NaN           NaN               NaN      NaN  …       NaN
4633      NaN           NaN               NaN      NaN  …       NaN
3394      NaN           NaN               NaN      NaN  …       NaN
2944      NaN           NaN               NaN      NaN  …       NaN
…         …             …                …        …  …        …
1815      NaN           NaN               NaN      NaN  …       NaN
1631      NaN           NaN               NaN      NaN  …     100.0
602       NaN           NaN               NaN      NaN  …     150.0
4145      NaN           NaN               NaN      NaN  …     120.0
213       NaN           NaN               NaN      NaN  …     100.0

i      Watermelon  Wheat flour  White beans  Wild game meat  Yam flour  \
2659          NaN          NaN          NaN             NaN        NaN
899           NaN          NaN          NaN             NaN        NaN
4633          NaN          NaN          NaN             NaN        NaN
3394          NaN          NaN          NaN             NaN        NaN
2944          NaN          NaN          NaN             NaN        NaN
…            …           …           …              …          …
1815          NaN          NaN        140.0             NaN        NaN
1631          NaN          NaN        180.0             NaN        NaN
602           NaN          NaN          NaN             NaN        NaN
4145          NaN          NaN          NaN             NaN        NaN
213           NaN          NaN          NaN             NaN        NaN

i      Yam-roots  Total Expenditures  People per HH  Expenditures per capita
2659         NaN                70.0              6                11.666667
899          NaN               100.0              1               100.000000
4633         NaN               100.0              4                25.000000
3394         NaN               100.0              4                25.000000
2944         NaN               100.0              1               100.000000
```

```
…         …                  …              …                        …
1815      600.0              1970.0              1            1970.000000
1631        NaN              1970.0              4             492.500000
602         NaN              1970.0              5             394.000000
4145        NaN              1980.0             12             165.000000
213         NaN              1980.0              5             396.000000

[1201 rows x 130 columns]
```

[11]:
```
Q4_2010 = quartiles_by_te(expend, 2010, 4)
Q4_2010
#TE is total expenditure
```

[11]:
```
i         t       j  m  (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  \
2408   2010  190012  1                      NaN                NaN         NaN
2410   2010  190015  1                      NaN                NaN         NaN
2025   2010  160062  1                      NaN                NaN         NaN
4753   2010  360056  1                      NaN                NaN         NaN
4024   2010  300148  1                      NaN                NaN         NaN
…      …      … ..                        …                  …           …
2403   2010  190007  1                     80.0                NaN         NaN
2299   2010  180019  1                      NaN              100.0         NaN
3282   2010  250025  1                      NaN                NaN         NaN
3277   2010  250020  1                      NaN                NaN         NaN
4005   2010  300125  1                      NaN             2100.0         NaN

i      Apples  Avocado pear  Baby milk powder  Bananas  …  Tomatoes  \
2408      NaN           NaN               NaN      NaN  …     300.0
2410      NaN           NaN               NaN      NaN  …     150.0
2025      NaN           NaN               NaN    200.0  …      50.0
4753      NaN           NaN               NaN      NaN  …     120.0
4024      NaN           NaN               NaN      NaN  …     100.0
…       …            …                 …        …   …       …
2403      NaN           NaN            1100.0      NaN  …     250.0
2299      NaN           NaN               NaN      NaN  …     500.0
3282      NaN           NaN               NaN      NaN  …       NaN
3277      NaN           NaN               NaN      NaN  …     100.0
4005      NaN           NaN               NaN    700.0  …     200.0

i      Watermelon  Wheat flour  White beans  Wild game meat  Yam flour  \
2408          NaN          NaN        300.0             NaN        NaN
2410          NaN          NaN        250.0             NaN        NaN
2025          NaN          NaN        300.0             NaN        NaN
4753          NaN          NaN        200.0             NaN        NaN
4024          NaN          NaN          NaN             NaN        NaN
…           …            …            …               …          …
2403          NaN          NaN       1500.0             NaN        NaN
```

```
2299           NaN           NaN         560.0              NaN        NaN
3282           NaN           NaN           NaN              NaN        NaN
3277           NaN           NaN        1200.0              NaN        NaN
4005           NaN        1400.0           NaN              NaN     4000.0

i      Yam-roots  Total Expenditures  People per HH  Expenditures per capita
2408         NaN              5405.0             10               540.500000
2410         NaN              5410.0              2              2705.000000
2025       500.0              5410.0              3              1803.333333
4753         NaN              5410.0             11               491.818182
4024         NaN              5410.0              8               676.250000
…          …                  …          …                  …
2403         NaN             35120.0             11              3192.727273
2299      2000.0             35190.0              6              5865.000000
3282     24000.0             37530.0             10              3753.000000
3277     32000.0             44630.0              6              7438.333333
4005      1500.0             45240.0              4             11310.000000

[1201 rows x 130 columns]
```

Expenditures per capita (EPC) is more representative of the household spending as it takes into account the amount of people in the home. Therefore, we will be using EPC for our analysis. Below we have found the upper and lower quartiles for all of the years.

[12]:
```python
Q1_10 = quartiles_by_epc(expend, 2010, 1)
Q1_12 = quartiles_by_epc(expend, 2012, 1)
Q1_15 = quartiles_by_epc(expend, 2015, 1)
Q1_18 = quartiles_by_epc(expend, 2018, 1)
Q1 = pd.concat([Q1_10, Q1_12, Q1_15, Q1_18]).reset_index().
 ↪drop(columns=['index']).set_index(['t', 'j', 'm']).sort_values(['t','j'])
Q1 = Q1.drop(columns=['Total Expenditures', 'People per HH', 'Expenditures per␣
 ↪capita'])
Q1
#epc is expenditure per capita
```

[12]:
```
i               (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
t    j      m
2010 10013  1                       NaN                NaN         NaN     NaN
     10022  1                       NaN                NaN         NaN     NaN
     10063  1                       NaN                NaN         NaN     NaN
     10066  1                       NaN                NaN         NaN     NaN
     10069  1                       NaN                NaN         NaN     NaN
…                                  …                  …          …       …
2018 379090 1                       NaN                NaN         NaN     NaN
     379092 1                       NaN                NaN         NaN     NaN
     379094 1                       NaN                NaN         NaN     NaN
     379096 1                       NaN                NaN         NaN     NaN
```

```
     379127 1                      NaN                NaN         NaN     NaN

i              Avocado pear  Baby milk powder  Bananas   Beef  \
t    j      m
2010 10013  1           NaN               NaN    100.0  400.0
     10022  1           NaN               NaN    150.0    NaN
     10063  1           NaN               NaN      NaN    NaN
     10066  1           NaN               NaN      NaN  300.0
     10069  1           NaN               NaN      NaN    NaN
…                     …                …      …    …
2018 379090 1           NaN               NaN      NaN  500.0
     379092 1           NaN               NaN      NaN    NaN
     379094 1           NaN               NaN      NaN    NaN
     379096 1           NaN               NaN      NaN    NaN
     379127 1           NaN               NaN    150.0    NaN

i              Beer (local and imported)  Biscuits  …  Sweet Potatoes   Tea  \
t    j      m                                       …
2010 10013  1                        NaN       NaN  …             NaN  30.0
     10022  1                        NaN       NaN  …             NaN   NaN
     10063  1                        NaN       NaN  …             NaN   NaN
     10066  1                        NaN       NaN  …             NaN   NaN
     10069  1                        NaN       NaN  …             NaN   NaN
…                                  …       …  …              …   …
2018 379090 1                        NaN       NaN  …             NaN   NaN
     379092 1                        NaN       NaN  …             NaN  50.0
     379094 1                        NaN     120.0  …             NaN   NaN
     379096 1                        NaN       NaN  …             NaN   NaN
     379127 1                        NaN       NaN  …             NaN   NaN

i              Tomato puree(canned)  Tomatoes  Watermelon  Wheat flour  \
t    j      m
2010 10013  1                  60.0     100.0         NaN          NaN
     10022  1                  30.0      50.0         NaN          NaN
     10063  1                  35.0     100.0         NaN          NaN
     10066  1                  35.0     100.0         NaN          NaN
     10069  1                   NaN     100.0         NaN          NaN
…                             …       …          …           …
2018 379090 1                   NaN     200.0         NaN          NaN
     379092 1                   NaN       NaN         NaN          NaN
     379094 1                   NaN     250.0         NaN          NaN
     379096 1                   NaN     300.0         NaN          NaN
     379127 1                 350.0       NaN       200.0          NaN

i              White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10013  1        100.0             NaN        NaN      200.0
```

```
     10022  1         200.0             NaN        NaN        NaN
     10063  1           NaN             NaN        NaN        NaN
     10066  1         100.0             NaN        NaN        NaN
     10069  1           NaN             NaN        NaN      300.0
…                       …             …        …        …
2018 379090 1           NaN             NaN        NaN        NaN
     379092 1           NaN             NaN        NaN        NaN
     379094 1           NaN             NaN        NaN        NaN
     379096 1           NaN             NaN        NaN        NaN
     379127 1           NaN             NaN        NaN        NaN

[4752 rows x 124 columns]
```

[13]:

```python
Q2_10 = quartiles_by_epc(expend, 2010, 2)
Q2_12 = quartiles_by_epc(expend, 2012, 2)
Q2_15 = quartiles_by_epc(expend, 2015, 2)
Q2_18 = quartiles_by_epc(expend, 2018, 2)
Q2 = pd.concat([Q2_10, Q2_12, Q2_15, Q2_18]).reset_index().
 ↪drop(columns=['index']).set_index(['t', 'j', 'm']).sort_values(['t','j'])
Q2 = Q2.drop(columns=['Total Expenditures', 'People per HH', 'Expenditures per␣
 ↪capita'])
Q2
#epc is expenditure per capita
```

[13]:

```
i              (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
t    j      m
2010 10019  1                      NaN                NaN         NaN     NaN
     10020  1                      NaN                NaN         NaN     NaN
     10021  1                      NaN                NaN         NaN     NaN
     10025  1                      NaN                NaN         NaN     NaN
     10027  1                    100.0                NaN         NaN     NaN
…                                …                 …          …       …
2018 379079 1                      NaN                NaN         NaN     NaN
     379082 1                      NaN                NaN         NaN     NaN
     379091 1                      NaN                NaN         NaN     NaN
     379093 1                     50.0                NaN         NaN     NaN
     379105 1                      NaN              600.0         NaN     NaN

i              Avocado pear  Baby milk powder  Bananas   Beef  \
t    j      m
2010 10019  1           NaN               NaN      NaN    NaN
     10020  1           NaN               NaN      NaN  200.0
     10021  1           NaN               NaN    150.0  600.0
     10025  1           NaN               NaN    100.0  250.0
     10027  1           NaN               NaN      NaN    NaN
…                     …                …       …      …
2018 379079 1           NaN               NaN      NaN    NaN
```

```
     379082 1           NaN               NaN      NaN    NaN
     379091 1           NaN               NaN      NaN    NaN
     379093 1           NaN               NaN      NaN  500.0
     379105 1           NaN               NaN      NaN    NaN

i              Beer (local and imported)  Biscuits  …  Sweet Potatoes   Tea  \
t    j      m                                       …
2010 10019  1                        NaN       NaN  …             NaN   NaN
     10020  1                        NaN       NaN  …             NaN  30.0
     10021  1                        NaN       NaN  …             NaN   NaN
     10025  1                        NaN       NaN  …             NaN   NaN
     10027  1                        NaN       NaN  …             NaN   NaN
…                                  …       …  …             …   …
2018 379079 1                        NaN       NaN  …             NaN   NaN
     379082 1                        NaN       NaN  …             NaN   NaN
     379091 1                        NaN       NaN  …             NaN   NaN
     379093 1                        NaN       NaN  …             NaN   NaN
     379105 1                        NaN       NaN  …             NaN   NaN

i              Tomato puree(canned)  Tomatoes  Watermelon  Wheat flour  \
t    j      m
2010 10019  1                 210.0     100.0         NaN          NaN
     10020  1                  50.0     100.0         NaN          NaN
     10021  1                  30.0     100.0         NaN          NaN
     10025  1                  30.0      50.0         NaN          NaN
     10027  1                  30.0     100.0         NaN          NaN
…                             …       …         …          …
2018 379079 1                   NaN     200.0         NaN          NaN
     379082 1                   NaN     150.0         NaN          NaN
     379091 1                   NaN       NaN         NaN          NaN
     379093 1                   NaN     200.0         NaN          NaN
     379105 1                   NaN     100.0       300.0          NaN

i              White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10019  1        480.0             NaN        NaN        NaN
     10020  1        100.0             NaN        NaN      200.0
     10021  1        280.0             NaN        NaN        NaN
     10025  1        100.0             NaN        NaN        NaN
     10027  1          NaN             NaN        NaN      200.0
…                    …             …        …        …
2018 379079 1          NaN             NaN        NaN        NaN
     379082 1          NaN             NaN        NaN        NaN
     379091 1          NaN             NaN        NaN        NaN
     379093 1       1250.0             NaN        NaN        NaN
     379105 1          NaN             NaN        NaN        NaN
```

```
[4752 rows x 124 columns]
```

[14]:
```
Q3_10 = quartiles_by_epc(expend, 2010, 3)
Q3_12 = quartiles_by_epc(expend, 2012, 3)
Q3_15 = quartiles_by_epc(expend, 2015, 3)
Q3_18 = quartiles_by_epc(expend, 2018, 3)
Q3 = pd.concat([Q3_10, Q3_12, Q3_15, Q3_18]).reset_index().
 ↪drop(columns=['index']).set_index(['t', 'j', 'm']).sort_values(['t','j'])
Q3 = Q3.drop(columns=['Total Expenditures', 'People per HH', 'Expenditures per␣
 ↪capita'])
Q3
#epc is expenditure per capita
```

[14]:
```
i                (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
t    j      m
2010 10003  1                        NaN              180.0         NaN     NaN
     10008  1                        NaN              360.0         NaN     NaN
     10011  1                        NaN                NaN         NaN     NaN
     10012  1                        NaN                NaN         NaN     NaN
     10015  1                        NaN                NaN         NaN     NaN
…                                      …                  …           …       …
2018 379103 1                        NaN             1000.0         NaN     NaN
     379121 1                        NaN                NaN         NaN     NaN
     379123 1                        NaN               80.0         NaN     NaN
     379143 1                      150.0              200.0         NaN     NaN
     379155 1                      100.0              950.0         NaN     NaN

i                Avocado pear  Baby milk powder  Bananas    Beef  \
t    j      m
2010 10003  1             NaN               NaN    100.0   500.0
     10008  1            90.0               NaN    300.0     NaN
     10011  1             NaN               NaN      NaN   500.0
     10012  1             NaN            1200.0      NaN   500.0
     10015  1             NaN               NaN      NaN     NaN
…                           …                 …        …       …
2018 379103 1             NaN               NaN      NaN     NaN
     379121 1             NaN               NaN    250.0     NaN
     379123 1             NaN               NaN      NaN  1300.0
     379143 1             NaN               NaN      NaN     NaN
     379155 1             NaN               NaN      NaN  1400.0

i                Beer (local and imported)  Biscuits  …  Sweet Potatoes   Tea  \
t    j      m                                         …
2010 10003  1                          NaN       NaN  …           200.0  60.0
     10008  1                          NaN       NaN  …             NaN   NaN
     10011  1                          NaN       NaN  …             NaN   NaN
     10012  1                          NaN       NaN  …             NaN   NaN
```

```
      10015   1                      NaN        NaN  …              NaN    NaN
…                                  …       …  …              …   …
2018 379103 1                      NaN        NaN  …            300.0    NaN
     379121 1                      NaN        NaN  …              NaN   50.0
     379123 1                      NaN        NaN  …            100.0    NaN
     379143 1                      NaN        NaN  …              NaN    NaN
     379155 1                      NaN        NaN  …              NaN    NaN

i              Tomato puree(canned)  Tomatoes  Watermelon  Wheat flour  \
t    j      m
2010 10003  1                  90.0     100.0         NaN          NaN
     10008  1                 350.0     100.0         NaN          NaN
     10011  1                  60.0     100.0         NaN          NaN
     10012  1                 120.0     150.0         NaN          NaN
     10015  1                  30.0      80.0         NaN          NaN
…                             …       …         …          …
2018 379103 1                 600.0       NaN         NaN          NaN
     379121 1                   NaN     500.0         NaN          NaN
     379123 1                   NaN       NaN         NaN          NaN
     379143 1                   NaN       NaN       320.0          NaN
     379155 1                   NaN     300.0       200.0          NaN

i              White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10003  1        100.0             NaN        NaN      400.0
     10008  1        400.0             NaN        NaN      400.0
     10011  1        200.0             NaN        NaN      400.0
     10012  1        300.0             NaN        NaN      600.0
     10015  1        100.0             NaN        NaN      400.0
…                     …              …        …        …
2018 379103 1          NaN             NaN        NaN     1400.0
     379121 1        450.0             NaN        NaN     2500.0
     379123 1        400.0             NaN        NaN        NaN
     379143 1        800.0             NaN        NaN     1100.0
     379155 1          NaN             NaN        NaN     2500.0

[4752 rows x 124 columns]
```

[15]:
```
Q4_10 = quartiles_by_epc(expend, 2010, 4)
Q4_12 = quartiles_by_epc(expend, 2012, 4)
Q4_15 = quartiles_by_epc(expend, 2015, 4)
Q4_18 = quartiles_by_epc(expend, 2018, 4)
Q4 = pd.concat([Q4_10, Q4_12, Q4_15, Q4_18]).reset_index().
 ↪drop(columns=['index']).set_index(['t', 'j', 'm']).sort_values(['t','j'])
Q4 = Q4.drop(columns=['Total Expenditures', 'People per HH', 'Expenditures per␣
 ↪capita'])
Q4
```

```
[15]: i               (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
t    j      m
2010 10001  1                     NaN              280.0         NaN     NaN
     10002  1                     NaN              280.0         NaN     NaN
     10004  1                     NaN              180.0         NaN     NaN
     10006  1                     NaN                NaN         NaN     NaN
     10009  1                     NaN                NaN         NaN     NaN
…                                   …                  …           …       …
2018 379144 1                     NaN                NaN         NaN   900.0
     379146 1                     NaN             1100.0         NaN     NaN
     379148 1                   100.0                NaN         NaN     NaN
     379151 1                     NaN              900.0         NaN     NaN
     379154 1                   200.0             1200.0         NaN     NaN

i               Avocado pear  Baby milk powder  Bananas    Beef  \
t    j      m
2010 10001  1            NaN               NaN    200.0   500.0
     10002  1            NaN               NaN    180.0  1200.0
     10004  1            NaN               NaN    100.0   500.0
     10006  1            NaN               NaN    300.0   300.0
     10009  1            NaN             600.0    100.0   300.0
…                          …                 …        …       …
2018 379144 1            NaN               NaN    600.0     NaN
     379146 1            NaN               NaN      NaN     NaN
     379148 1            NaN               NaN      NaN   700.0
     379151 1            NaN               NaN    500.0     NaN
     379154 1            NaN               NaN      NaN  1300.0

i               Beer (local and imported)  Biscuits  …  Sweet Potatoes  \
t    j      m                                        …
2010 10001  1                       540.0       NaN  …           150.0
     10002  1                      2000.0       NaN  …           200.0
     10004  1                         NaN       NaN  …             NaN
     10006  1                         NaN       NaN  …             NaN
     10009  1                         NaN       NaN  …             NaN
…                                       …         …  …               …
2018 379144 1                         NaN    2200.0  …             NaN
     379146 1                         NaN       NaN  …             NaN
     379148 1                         NaN       NaN  …             NaN
     379151 1                         NaN       NaN  …             NaN
     379154 1                         NaN       NaN  …             NaN

i                 Tea  Tomato puree(canned)  Tomatoes  Watermelon  Wheat flour  \
t    j      m
2010 10001  1    NaN                  150.0     150.0         NaN          NaN
     10002  1  140.0                  240.0     120.0         NaN          NaN
     10004  1   30.0                   60.0     100.0         NaN          NaN
```

```
     10006  1  650.0                   NaN     400.0         NaN          NaN
     10009  1   60.0                 120.0     200.0         NaN          NaN
…              …                   …       …         …          …
2018 379144 1    NaN                   NaN     400.0       100.0          NaN
     379146 1    NaN                   NaN       NaN       500.0          NaN
     379148 1    NaN                  60.0     200.0       150.0          NaN
     379151 1    NaN                 150.0     600.0       600.0        750.0
     379154 1    NaN                   NaN     100.0       200.0          NaN

i              White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10001  1        600.0             NaN        NaN     1500.0
     10002  1        400.0             NaN        NaN     1200.0
     10004  1        100.0             NaN        NaN      400.0
     10006  1          NaN             NaN        NaN      400.0
     10009  1        270.0             NaN        NaN      400.0
…                    …               …         …          …
2018 379144 1          NaN             NaN     1100.0     3500.0
     379146 1          NaN             NaN        NaN     1800.0
     379148 1          NaN             NaN        NaN     1600.0
     379151 1       1600.0             NaN        NaN     3500.0
     379154 1          NaN             NaN        NaN      650.0

[4752 rows x 124 columns]
```

2.1 Filter Household Dataframe to create one only including 1st quartile households and another including just 4th quartile households.

[16]:
```
#First Quartile
hh_char = hh_char.reorder_levels(['t','j','m'])
Q1Index = Q1.index.tolist()
Q2Index = Q2.index.tolist()
Q3Index = Q3.index.tolist()
Q4Index = Q4.index.tolist()
hh_charQ1 = hh_char[hh_char.index.isin(Q1Index)]
hh_charQ2 = hh_char[hh_char.index.isin(Q2Index)]
hh_charQ3 = hh_char[hh_char.index.isin(Q3Index)]
hh_charQ4 = hh_char[hh_char.index.isin(Q4Index)]
hh_charQ1
```

[16]:
```
k              M 0-3  M 4-8  M 9-13  M 14-18  M 19-30  M 31-50  M 51+  F 0-3  \
t    j      m
2010 10013  1      0      0       0        0        0        2      0      1
     10022  1      0      1       1        1        0        1      0      0
     10063  1      0      0       0        0        3        0      1      0
     10066  1      0      0       0        1        0        0      1      0
     10069  1      0      0       0        0        1        0      0      0
```

```
…                 …     …      …       …       …       …       …     …
2018 379090 1       1      0       2       0       0       1      0      0
     379092 1       0      0       1       0       1       0      0      2
     379094 1       1      0       0       0       0       1      0      0
     379096 1       0      1       1       1       0       1      0      0
     379127 1       1      0       0       0       0       1      0      0

k                 F 4-8  F 9-13  F 14-18  F 19-30  F 31-50  F 51+
t    j      m
2010 10013  1       0       0        1        2        1      1
     10022  1       0       1        0        0        1      0
     10063  1       0       0        0        0        0      1
     10066  1       0       0        2        1        1      0
     10069  1       0       1        1        3        0      1
…                 …      …       …        …        …      …
2018 379090 1       1       0        2        0        1      1
     379092 1       2       0        0        2        0      0
     379094 1       1       1        0        1        0      0
     379096 1       1       0        1        1        1      0
     379127 1       0       1        0        1        0      0

[4752 rows x 14 columns]
```

[17]:
```
#Fourth Quartile
hh_charQ4
```

[17]:
```
k                 M 0-3  M 4-8  M 9-13  M 14-18  M 19-30  M 31-50  M 51+  F 0-3  \
t    j      m
2010 10001  1       0      0       0       0        1        2       0      1
     10002  1       0      0       1       1        1        1       0      0
     10004  1       0      0       1       0        0        0       1      0
     10006  1       0      0       0       0        1        1       0      0
     10009  1       0      0       0       0        0        1       0      1
…                 …     …      …       …       …       …       …     …
2018 379144 1       0      0       0       0        0        1       0      0
     379146 1       0      0       0       0        1        1       1      0
     379148 1       0      0       0       0        1        0       0      0
     379151 1       0      0       2       0        0        0       1      0
     379154 1       0      0       0       0        0        0       1      0

k                 F 4-8  F 9-13  F 14-18  F 19-30  F 31-50  F 51+
t    j      m
2010 10001  1       0       0        0        1        2      0
     10002  1       0       0        0        2        1      0
     10004  1       0       0        0        0        0      1
     10006  1       0       0        0        1        0      0
     10009  1       0       0        1        1        0      0
```

```
…                …      …        …        …        …     …
2018 379144 1       1        0        0        1        0       0
     379146 1       0        0        0        0        0       1
     379148 1       0        0        0        0        0       0
     379151 1       0        0        1        0        1       0
     379154 1       0        0        0        1        0       0

[4752 rows x 14 columns]
```

[18]:

```python
#Log of Food Expenditure Dataframe (after running np.log on values)

Q1 = Q1.replace(0,np.nan)
Q2 = Q2.replace(0,np.nan)
Q3 = Q3.replace(0,np.nan)
Q4 = Q4.replace(0,np.nan)

log_Q1 = np.log(Q1)
log_Q2 = np.log(Q2)
log_Q3 = np.log(Q3)
log_Q4 = np.log(Q4)
```

[19]:

```python
log_Q1
```

[19]:

```
i              (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
t    j      m
2010 10013  1                      NaN                NaN         NaN     NaN
     10022  1                      NaN                NaN         NaN     NaN
     10063  1                      NaN                NaN         NaN     NaN
     10066  1                      NaN                NaN         NaN     NaN
     10069  1                      NaN                NaN         NaN     NaN
…                                 …                 …          …       …
2018 379090 1                      NaN                NaN         NaN     NaN
     379092 1                      NaN                NaN         NaN     NaN
     379094 1                      NaN                NaN         NaN     NaN
     379096 1                      NaN                NaN         NaN     NaN
     379127 1                      NaN                NaN         NaN     NaN

i              Avocado pear  Baby milk powder   Bananas      Beef  \
t    j      m
2010 10013  1           NaN               NaN  4.605170  5.991465
     10022  1           NaN               NaN  5.010635       NaN
     10063  1           NaN               NaN       NaN       NaN
     10066  1           NaN               NaN       NaN  5.703782
     10069  1           NaN               NaN       NaN       NaN
…                      …                …       …        …
2018 379090 1           NaN               NaN       NaN  6.214608
     379092 1           NaN               NaN       NaN       NaN
```

```
     379094 1           NaN               NaN       NaN       NaN
     379096 1           NaN               NaN       NaN       NaN
     379127 1           NaN               NaN  5.010635       NaN

i              Beer (local and imported)  Biscuits  …  Sweet Potatoes  \
t    j      m                                       …
2010 10013  1                        NaN       NaN  …             NaN
     10022  1                        NaN       NaN  …             NaN
     10063  1                        NaN       NaN  …             NaN
     10066  1                        NaN       NaN  …             NaN
     10069  1                        NaN       NaN  …             NaN
…                                  …        …  …              …
2018 379090 1                        NaN       NaN  …             NaN
     379092 1                        NaN       NaN  …             NaN
     379094 1                        NaN  4.787492  …             NaN
     379096 1                        NaN       NaN  …             NaN
     379127 1                        NaN       NaN  …             NaN

i                   Tea  Tomato puree(canned)  Tomatoes  Watermelon  \
t    j      m
2010 10013  1  3.401197              4.094345  4.605170         NaN
     10022  1       NaN              3.401197  3.912023         NaN
     10063  1       NaN              3.555348  4.605170         NaN
     10066  1       NaN              3.555348  4.605170         NaN
     10069  1       NaN                   NaN  4.605170         NaN
…                  …                   …        …          …
2018 379090 1       NaN                   NaN  5.298317         NaN
     379092 1  3.912023                   NaN       NaN         NaN
     379094 1       NaN                   NaN  5.521461         NaN
     379096 1       NaN                   NaN  5.703782         NaN
     379127 1       NaN              5.857933       NaN    5.298317

i              Wheat flour  White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10013  1          NaN     4.605170             NaN        NaN   5.298317
     10022  1          NaN     5.298317             NaN        NaN        NaN
     10063  1          NaN          NaN             NaN        NaN        NaN
     10066  1          NaN     4.605170             NaN        NaN        NaN
     10069  1          NaN          NaN             NaN        NaN   5.703782
…                     …          …             …         …        …
2018 379090 1          NaN          NaN             NaN        NaN        NaN
     379092 1          NaN          NaN             NaN        NaN        NaN
     379094 1          NaN          NaN             NaN        NaN        NaN
     379096 1          NaN          NaN             NaN        NaN        NaN
     379127 1          NaN          NaN             NaN        NaN        NaN

[4752 rows x 124 columns]
```

[20]:

```
#Log Household Size and add to household dataframe for Q1 and Q4

# set index to j, t, m so that df.sum() ignore index values
hh_charQ1 = hh_charQ1.reset_index()
hh_charQ1.set_index(['j','t','m'], inplace=True)
hh_charQ2 = hh_charQ2.reset_index()
hh_charQ2.set_index(['j','t','m'], inplace=True)
hh_charQ3 = hh_charQ3.reset_index()
hh_charQ3.set_index(['j','t','m'], inplace=True)
hh_charQ4 = hh_charQ4.reset_index()
hh_charQ4.set_index(['j','t','m'], inplace=True)

# create new column of household size
hh_charQ1['Hsize'] = hh_charQ1.sum(axis=1).values
hh_charQ2['Hsize'] = hh_charQ2.sum(axis=1).values
hh_charQ3['Hsize'] = hh_charQ3.sum(axis=1).values
hh_charQ4['Hsize'] = hh_charQ4.sum(axis=1).values

# remove erroneous data with household_size = 0
hh_charQ1 = hh_charQ1[hh_charQ1['Hsize'] > 0]
hh_charQ2 = hh_charQ2[hh_charQ2['Hsize'] > 0]
hh_charQ3 = hh_charQ3[hh_charQ3['Hsize'] > 0]
hh_charQ4 = hh_charQ4[hh_charQ4['Hsize'] > 0]

# create new column 'log Hsize'
hh_charQ1['log Hsize'] = np.log(hh_charQ1['Hsize'])
hh_charQ2['log Hsize'] = np.log(hh_charQ2['Hsize'])
hh_charQ3['log Hsize'] = np.log(hh_charQ3['Hsize'])
hh_charQ4['log Hsize'] = np.log(hh_charQ4['Hsize'])

# remove Hsize column
hh_charQ1 = hh_charQ1.drop(columns=['Hsize'])
hh_charQ2 = hh_charQ2.drop(columns=['Hsize'])
hh_charQ3 = hh_charQ3.drop(columns=['Hsize'])
hh_charQ4 = hh_charQ4.drop(columns=['Hsize'])
```

[21]:

```
#test
hh_charQ1
```

[21]:

```
k                M 0-3  M 4-8  M 9-13  M 14-18  M 19-30  M 31-50  M 51+  F 0-3  \
j     t    m
10013 2010 1         0      0       0        0        0        2      0      1
10022 2010 1         0      1       1        1        0        1      0      0
10063 2010 1         0      0       0        0        3        0      1      0
10066 2010 1         0      0       0        1        0        0      1      0
10069 2010 1         0      0       0        0        1        0      0      0
…                 …     …     …      …       …       …       …     …
```

```
379090 2018 1       1       0       2       0       0       1       0       0
379092 2018 1       0       0       1       0       1       0       0       2
379094 2018 1       1       0       0       0       0       1       0       0
379096 2018 1       0       1       1       1       0       1       0       0
379127 2018 1       1       0       0       0       0       1       0       0

k              F 4-8  F 9-13  F 14-18  F 19-30  F 31-50  F 51+  log Hsize
j      t    m
10013  2010 1      0       0        1        2        1      1   2.079442
10022  2010 1      0       1        0        0        1      0   1.791759
10063  2010 1      0       0        0        0        0      1   1.609438
10066  2010 1      0       0        2        1        1      0   1.791759
10069  2010 1      0       1        1        3        0      1   1.945910
…              …     …      …      …      …      …        …
379090 2018 1      1       0        2        0        1      1   2.197225
379092 2018 1      2       0        0        2        0      0   2.079442
379094 2018 1      1       1        0        1        0      0   1.609438
379096 2018 1      1       0        1        1        1      0   2.079442
379127 2018 1      0       1        0        1        0      0   1.386294

[4752 rows x 15 columns]
```

2.2 Estimation

Below, we estimate the demand system for the upper and lower quartile households in Nigeria.

[22]:
```
log_Q1
```

[22]:
```
i              (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
t    j      m
2010 10013  1                      NaN                NaN         NaN     NaN
     10022  1                      NaN                NaN         NaN     NaN
     10063  1                      NaN                NaN         NaN     NaN
     10066  1                      NaN                NaN         NaN     NaN
     10069  1                      NaN                NaN         NaN     NaN
…                                …                 …           …       …
2018 379090 1                      NaN                NaN         NaN     NaN
     379092 1                      NaN                NaN         NaN     NaN
     379094 1                      NaN                NaN         NaN     NaN
     379096 1                      NaN                NaN         NaN     NaN
     379127 1                      NaN                NaN         NaN     NaN

i              Avocado pear  Baby milk powder   Bananas      Beef  \
t    j      m
2010 10013  1           NaN               NaN  4.605170  5.991465
     10022  1           NaN               NaN  5.010635       NaN
     10063  1           NaN               NaN       NaN       NaN
```

```
     10066  1           NaN               NaN       NaN  5.703782
     10069  1           NaN               NaN       NaN       NaN
…                     …              …       …       …
2018 379090 1           NaN               NaN       NaN  6.214608
     379092 1           NaN               NaN       NaN       NaN
     379094 1           NaN               NaN       NaN       NaN
     379096 1           NaN               NaN       NaN       NaN
     379127 1           NaN               NaN  5.010635       NaN

i              Beer (local and imported)  Biscuits  …  Sweet Potatoes  \
t    j      m                                       …
2010 10013  1                        NaN       NaN  …             NaN
     10022  1                        NaN       NaN  …             NaN
     10063  1                        NaN       NaN  …             NaN
     10066  1                        NaN       NaN  …             NaN
     10069  1                        NaN       NaN  …             NaN
…                                  …       …  …             …
2018 379090 1                        NaN       NaN  …             NaN
     379092 1                        NaN       NaN  …             NaN
     379094 1                        NaN  4.787492  …             NaN
     379096 1                        NaN       NaN  …             NaN
     379127 1                        NaN       NaN  …             NaN

i                   Tea  Tomato puree(canned)  Tomatoes  Watermelon  \
t    j      m
2010 10013  1  3.401197              4.094345  4.605170         NaN
     10022  1       NaN              3.401197  3.912023         NaN
     10063  1       NaN              3.555348  4.605170         NaN
     10066  1       NaN              3.555348  4.605170         NaN
     10069  1       NaN                   NaN  4.605170         NaN
…                 …                    …       …       …
2018 379090 1       NaN                   NaN  5.298317         NaN
     379092 1  3.912023                   NaN       NaN         NaN
     379094 1       NaN                   NaN  5.521461         NaN
     379096 1       NaN                   NaN  5.703782         NaN
     379127 1       NaN              5.857933       NaN    5.298317

i              Wheat flour  White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10013  1          NaN     4.605170             NaN        NaN   5.298317
     10022  1          NaN     5.298317             NaN        NaN        NaN
     10063  1          NaN          NaN             NaN        NaN        NaN
     10066  1          NaN     4.605170             NaN        NaN        NaN
     10069  1          NaN          NaN             NaN        NaN   5.703782
…                    …          …              …         …        …
2018 379090 1          NaN          NaN             NaN        NaN        NaN
     379092 1          NaN          NaN             NaN        NaN        NaN
```

```
379094 1           NaN          NaN             NaN        NaN        NaN
379096 1           NaN          NaN             NaN        NaN        NaN
379127 1           NaN          NaN             NaN        NaN        NaN

[4752 rows x 124 columns]
```

[23]:
```python
log_Q1 = log_Q1.reorder_levels(['j','t','m'])
log_Q2 = log_Q2.reorder_levels(['j','t','m'])
log_Q3 = log_Q3.reorder_levels(['j','t','m'])
log_Q4 = log_Q4.reorder_levels(['j','t','m'])
```

[24]:
```python
import cfe
log_expend = np.log(expend)
log_expend = log_expend.reorder_levels(['j','t','m'])
result = cfe.Result(y=expend,z=hh_char)
```

```
Missing dependencies for OracleDemands.

/opt/conda/lib/python3.9/site-packages/pandas/core/internals/blocks.py:402:
RuntimeWarning: divide by zero encountered in log
  result = func(self.values, **kwargs)
```

[25]:
```python
import cfe

result1 = cfe.Result(y=log_Q1,z=hh_charQ1)
result2 = cfe.Result(y=log_Q2,z=hh_charQ2)
result3 = cfe.Result(y=log_Q3,z=hh_charQ3)
result4 = cfe.Result(y=log_Q4,z=hh_charQ4)
```

[26]:
```python
result1
```

[26]:
```
<xarray.Result>
Dimensions:           (k: 15, j: 3197, t: 4, m: 1, i: 9)
Coordinates:
  * j                 (j) int64 10005 10009 10013 10022 … 379094 379096 379127
  * t                 (t) int64 2010 2012 2015 2018
  * m                 (m) int64 1
  * i                 (i) <U36 'Bread' … 'White beans'
  * k                 (k) <U9 'M 0-3' 'M 4-8' 'M 9-13' … 'F 51+' 'log Hsize'
Data variables: (12/20)
    alpha             object None
    beta              object None
    delta             object None
    prices            object None
    characteristics   (k, j, t, m) float64 nan nan nan 0.0 … nan nan nan 1.386
    loglambdas        object None
    …                …
    se_beta           object None
```

```
    se_alpha          object None
    se_a              object None
    y                 (i, j, t, m) float64 nan nan nan nan … nan nan nan nan
    logp              object None
    z                 (k, j, t, m) float64 nan nan nan 0.0 … nan nan nan 1.386
Attributes:
    firstround:               2010
    min_proportion_items:     0.125
    min_xproducts:            30
    all_tm:                   True
    common_alpha:             True
    useless_expenditures:     False
    stderr_tol:               0.01
    indices:                  Indices(j='j', t='t', m='m', i='i', k='k')
    iterate:                  False
    verbose:                  False
```

This creates a complicated "Result" object, with lots of different attributes. Note from below that attributes y and z are now defined.

[27]:
```
result1.get_predicted_expenditures().sum(['m','i']).mean('j')
```

```
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
```

```
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
```

[27]:
```
<xarray.DataArray (t: 4)>
array([21.72649895, 11.63452879, 36.59153191, 56.15069086])
Coordinates:
  * t        (t) int64 2010 2012 2015 2018
```

[28]:
```
result1.get_reduced_form()
result2.get_reduced_form()
result3.get_reduced_form()
result4.get_reduced_form()
```

```
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
```

```
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:447: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  X=rhs.append(znil.join(timednil))
/opt/conda/lib/python3.9/site-packages/cfe/estimation.py:451: FutureWarning: The
frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  myb,mye=ols(X,lhs.append(ynil),return_se=False,return_v=False,return_e=True)
```

[29]:
```
result2.get_predicted_expenditures().sum(['m','i']).mean('j')
result1.get_predicted_expenditures().sum(['m','i']).mean('j')
```

[29]:
```
<xarray.DataArray (t: 4)>
array([21.72649895, 11.63452879, 36.59153191, 56.15069086])
Coordinates:
  * t        (t) int64 2010 2012 2015 2018
```

2.2.1 Estimate Demand System

[30]:
```
result1.delta.to_dataframe().unstack('k')
```

[30]:
```
                                           delta                                \
k                                          M 0-3     M 4-8    M 9-13   M 14-18
i
Bread                                  -0.003088  0.082963  0.034989  0.169686
Condiments,(salt,spices,pepper, etc)    0.264284  0.231446  0.067266  0.135015
Groundnut oil                           0.102788 -0.038919 -0.048425  0.032208
Onions                                  0.100306  0.056826  0.095169  0.044415
Palm oil                               -0.003929  0.123044  0.129281  0.053631
Rice-local                             -0.119735 -0.004833  0.076124 -0.017761
Sugar                                   0.143217  0.121148  0.123899  0.139555
Tomatoes                               -0.018850 -0.057002  0.040346 -0.004064
White beans                             0.074460  0.176303  0.186693  0.098305

                                                                                \
k                                        M 19-30   M 31-50     M 51+     F 0-3
```

```
i
Bread                                0.015024  0.299578  0.291810 -0.059507
Condiments,(salt,spices,pepper, etc)  0.212801 -0.087472 -0.022021 -0.114583
Groundnut oil                        0.021546  0.247795  0.188258  0.074367
Onions                               0.081890  0.058262  0.133201 -0.024845
Palm oil                             0.127674 -0.041609 -0.154937  0.043462
Rice-local                          -0.048299  0.102536 -0.044278 -0.113868
Sugar                                0.012479  0.170398  0.045830  0.120489
Tomatoes                            -0.073002 -0.134035  0.004232 -0.007084
White beans                          0.205889  0.280750  0.090692 -0.022304

                                                                               \
k                                       F 4-8    F 9-13   F 14-18   F 19-30
i
Bread                                0.065292 -0.032147 -0.039754 -0.007267
Condiments,(salt,spices,pepper, etc)  0.255336  0.073331  0.073238  0.276805
Groundnut oil                       -0.078428  0.047924 -0.063912  0.152385
Onions                              -0.048369  0.010305 -0.052113  0.185825
Palm oil                             0.074346  0.108757  0.087074  0.165168
Rice-local                          -0.025423  0.094143 -0.000832  0.019538
Sugar                               -0.012868  0.003006 -0.069184  0.060883
Tomatoes                            -0.047483 -0.103632 -0.015516 -0.117228
White beans                          0.193684  0.122504  0.058622  0.136236


k                                     F 31-50     F 51+ log Hsize
i
Bread                               -0.010486  0.028787  0.267501
Condiments,(salt,spices,pepper, etc)  0.243430  0.110919 -0.563309
Groundnut oil                        0.050279 -0.080470  0.257256
Onions                               0.064186 -0.064723  0.160344
Palm oil                             0.055128  0.076823 -0.208557
Rice-local                          -0.180575 -0.126230  1.553010
Sugar                                0.075988  0.012633  0.763390
Tomatoes                            -0.174486 -0.069434  1.307806
White beans                          0.133272  0.027455 -0.433032
```

[31]:
```
result2.delta.to_dataframe().unstack('k')
```

[31]:
```
                                        delta                                  \
k                                       M 0-3     M 4-8    M 9-13   M 14-18
i
Agricultural eggs                   -0.149408 -0.029330  0.061333  0.014785
Bananas                              0.006462 -0.044960  0.032047  0.046317
Beef                                 0.028448 -0.008889 -0.004314  0.022559
Bread                               -0.079721  0.022023 -0.040479 -0.010383
Chocolate drinks                     0.133956 -0.122197 -0.010376 -0.002650
```

Condiments,(salt,spices,pepper, etc)	0.000502	0.082354	-0.002154	0.041064
Fish-Dried	0.046038	-0.054245	0.053627	0.069409
Fish-Frozen	-0.036815	-0.032825	-0.103183	0.012847
Fish-Smoked	0.044318	-0.087294	-0.105796	0.001042
Gari-White	0.076384	0.071579	-0.042999	0.104379
Groundnut oil	-0.022746	0.019457	-0.025789	-0.029104
Malt drinks	0.039378	0.115862	0.064490	0.027959
Milk powder	0.035144	-0.060811	0.079790	0.006849
Okra-fresh	0.073012	0.030285	0.086437	0.059413
Onions	0.027240	0.051260	0.013784	0.026897
Orange/tangerine	0.077832	0.126425	0.201823	0.163846
Other vegetables (fresh or canned)	0.047578	-0.034661	-0.099673	0.039893
Palm oil	0.028490	-0.010349	-0.025235	-0.013668
Plantains	0.051501	0.042952	0.111423	0.016898
Rice-local	0.004247	0.064982	0.040946	-0.032713
Sachet water	-0.077003	0.014596	0.025873	0.002419
Soft drinks (Coca cola, spirit etc)	0.062965	0.115520	0.070267	0.046627
Sugar	0.103983	0.079862	0.075806	0.047237
Tomato puree(canned)	0.048917	0.146464	-0.022179	0.080208
Tomatoes	-0.051011	0.047982	-0.046268	0.036451
White beans	-0.013188	0.049317	-0.023283	-0.058784
Yam-roots	-0.085264	-0.012726	0.037266	0.022904

\

k	M 19-30	M 31-50	M 51+	F 0-3
i				
Agricultural eggs	0.157119	-0.092332	-0.098213	0.060626
Bananas	0.038857	-0.056998	-0.100910	0.011760
Beef	0.003155	0.184722	0.187461	-0.023672
Bread	0.033765	-0.006243	0.021375	0.002285
Chocolate drinks	-0.038670	-0.083450	0.086666	0.139527
Condiments,(salt,spices,pepper, etc)	0.080214	0.105215	0.046314	0.159833
Fish-Dried	0.010979	0.071410	0.211244	0.021914
Fish-Frozen	-0.011492	-0.014539	0.055522	-0.020630
Fish-Smoked	0.062337	0.148159	0.107323	0.039837
Gari-White	-0.019880	-0.112050	-0.094600	-0.013716
Groundnut oil	-0.018929	0.009173	0.005768	0.005526
Malt drinks	-0.044970	0.140354	0.101275	0.119221
Milk powder	0.018666	0.146864	0.107368	0.127773
Okra-fresh	0.070622	-0.006358	0.085941	0.169547
Onions	0.030333	-0.018891	-0.008810	0.022632
Orange/tangerine	0.142139	0.201692	0.053430	0.131577
Other vegetables (fresh or canned)	0.105345	-0.200324	-0.071762	0.031058
Palm oil	-0.018297	-0.012219	-0.062770	0.006929
Plantains	-0.041466	-0.081510	0.078351	-0.100267
Rice-local	-0.017309	-0.009752	-0.019157	0.017380
Sachet water	0.079241	0.078367	0.205532	-0.025409

Soft drinks (Coca cola, spirit etc)	0.087477	0.123134	0.130210	0.092915
Sugar	0.062641	0.195590	0.067960	0.113864
Tomato puree(canned)	0.019062	-0.016023	0.013298	-0.005240
Tomatoes	-0.035788	-0.020524	0.065835	0.028852
White beans	-0.009671	-0.073803	-0.017692	-0.082063
Yam-roots	0.031901	-0.003700	-0.036998	0.031024

\

k	F 4-8	F 9-13	F 14-18	F 19-30
i				
Agricultural eggs	-0.004245	0.009784	-0.117230	0.003414
Bananas	-0.176151	-0.038690	-0.090330	-0.016012
Beef	-0.015165	0.011912	-0.002533	0.057462
Bread	0.005712	-0.008587	-0.026800	0.009275
Chocolate drinks	0.011077	-0.103455	0.052081	0.059220
Condiments,(salt,spices,pepper, etc)	0.083865	-0.051950	0.071937	0.014040
Fish-Dried	-0.021538	0.048588	0.101467	-0.005804
Fish-Frozen	-0.044020	-0.049671	-0.033901	-0.046894
Fish-Smoked	0.035355	-0.027455	0.056289	-0.027519
Gari-White	0.000963	-0.052294	0.000935	0.051273
Groundnut oil	-0.006850	-0.022580	0.012036	-0.015446
Malt drinks	0.082079	0.048835	0.146727	0.143269
Milk powder	0.081462	0.105423	0.029949	0.168437
Okra-fresh	0.085566	0.060401	0.060328	0.068516
Onions	0.019222	0.015903	0.044747	0.035912
Orange/tangerine	0.059301	0.130712	0.085661	0.095532
Other vegetables (fresh or canned)	-0.009731	-0.025615	0.012764	0.077753
Palm oil	-0.043598	-0.019202	-0.027863	-0.047211
Plantains	0.166734	0.097934	-0.000572	0.076049
Rice-local	0.039714	0.057240	0.066865	0.007918
Sachet water	-0.019090	0.178656	-0.015702	0.161823
Soft drinks (Coca cola, spirit etc)	0.071199	0.077210	0.132343	0.117131
Sugar	0.115462	0.059339	0.039200	0.062596
Tomato puree(canned)	0.031623	0.087846	0.154802	0.007820
Tomatoes	-0.004027	-0.000578	-0.003167	0.034138
White beans	0.034420	-0.012039	-0.037539	-0.008887
Yam-roots	0.040959	0.027863	0.013791	0.036928

k	F 31-50	F 51+	log Hsize
i			
Agricultural eggs	0.246177	0.014704	0.590517
Bananas	0.027479	-0.129261	0.805642
Beef	0.104752	0.001348	0.576457
Bread	0.046501	0.004192	0.682959
Chocolate drinks	0.203958	-0.041522	0.645734
Condiments,(salt,spices,pepper, etc)	0.112256	0.071462	0.469904

```
Fish-Dried                            0.046177  0.032737  0.413277
Fish-Frozen                          -0.002191 -0.044574  0.612085
Fish-Smoked                          -0.069849 -0.000958  0.612188
Gari-White                            0.157146  0.066903  0.645995
Groundnut oil                         0.038907  0.004623  0.628594
Malt drinks                           0.188959  0.133330 -0.100534
Milk powder                           0.150786 -0.072434  0.361369
Okra-fresh                            0.027598  0.137080  0.088681
Onions                                0.017850 -0.063791  0.493494
Orange/tangerine                     -0.057096 -0.034665 -0.248649
Other vegetables (fresh or canned)    0.064233 -0.118151  0.186942
Palm oil                             -0.004157  0.024547  0.787429
Plantains                            -0.101012 -0.060717  0.350049
Rice-local                           -0.052730  0.022589  1.099508
Sachet water                          0.147095 -0.183437  0.007532
Soft drinks (Coca cola, spirit etc)   0.043424 -0.046401 -0.092703
Sugar                                 0.042698  0.113093  0.759162
Tomato puree(canned)                  0.007902  0.034529  0.136066
Tomatoes                              0.007953  0.037552  0.591284
White beans                           0.015531 -0.009298  0.909166
Yam-roots                             0.032751  0.035769  0.732822
```

[32]:
```
result3.delta.to_dataframe().unstack('k')
```

[32]:
```
                                        delta                                \
k                                       M 0-3     M 4-8    M 9-13   M 14-18
i
(Cocoyam, Spinach, etc)               -0.101613  0.009736  0.014774 -0.021126
Agricultural eggs                     -0.094751 -0.020564 -0.097716 -0.059444
Bananas                               -0.026129 -0.053275 -0.020785 -0.003237
Beef                                  -0.011454  0.032995 -0.014269  0.042526
Bread                                 -0.015976  0.020409 -0.010424 -0.013506
Brown beans                            0.029903 -0.002820  0.005118 -0.016054
Chocolate drinks                       0.038106 -0.030012 -0.076438 -0.035719
Condiments,(salt,spices,pepper, etc) -0.108476 -0.001157 -0.001062  0.027418
Fish-Dried                            -0.057958  0.035462  0.054086 -0.031372
Fish-Fresh                            -0.050738 -0.021556 -0.007719 -0.008926
Fish-Frozen                           -0.064174 -0.015708 -0.003002  0.033685
Fish-Smoked                            0.033675  0.068787 -0.009994  0.021870
Garden eggs/egg plant                  0.157543  0.035721  0.097367  0.015024
Gari -Yellow                          -0.042706  0.027527 -0.028890  0.005838
Gari-White                             0.016917  0.014651 -0.027111 -0.042706
Groundnut oil                          0.040276  0.010718  0.032090  0.016067
Malt drinks                            0.004591 -0.006779 -0.018112  0.070083
Milk powder                            0.080475 -0.039754 -0.019755 -0.040508
Milk tinned (unsweetened)              0.094871  0.035653 -0.049096  0.003795
Okra-fresh                             0.008874  0.013462 -0.032638  0.019656
```

Onions	0.044297	0.040957	0.027109	0.046577
Orange/tangerine	0.039485	0.027183	0.056666	0.029260
Palm oil	0.019831	0.020954	0.033893	-0.000249
Plantains	0.146162	0.032520	0.119187	0.152486
Rice-Imported	0.011850	0.043939	0.001646	0.046391
Rice-local	0.136776	0.091696	0.116553	0.054471
Sachet water	-0.105584	-0.032982	-0.115722	0.038392
Soft drinks (Coca cola, spirit etc)	0.039747	0.024923	-0.028489	0.017013
Sugar	-0.015443	0.102630	0.065437	0.007583
Tea	0.112024	-0.182584	0.039989	-0.002522
Tomato puree(canned)	0.011772	0.000955	0.051615	-0.057446
Tomatoes	0.031670	-0.011566	-0.020328	0.022288
White beans	-0.037582	-0.028275	-0.002046	0.010233
Yam-roots	0.056094	0.000171	0.062682	0.060092

k	M 19-30	M 31-50	M 51+	F 0-3
i				
(Cocoyam, Spinach, etc)	-0.000517	-0.049552	-0.027786	-0.056071
Agricultural eggs	0.010247	-0.006913	-0.075705	0.009077
Bananas	-0.040094	-0.069887	-0.064069	0.009188
Beef	0.042413	0.083946	0.063645	0.000212
Bread	0.001353	0.002645	0.054042	-0.018945
Brown beans	-0.008772	-0.078679	0.005186	0.022939
Chocolate drinks	0.068077	0.010507	-0.031300	0.033843
Condiments,(salt,spices,pepper, etc)	0.061151	0.001223	-0.044594	0.131481
Fish-Dried	0.036942	0.097424	-0.002744	0.021681
Fish-Fresh	0.039136	-0.110149	-0.086005	-0.120026
Fish-Frozen	0.010503	0.019331	-0.024833	-0.030551
Fish-Smoked	-0.009153	0.010835	0.021057	0.020097
Garden eggs/egg plant	0.007025	0.125786	0.139841	0.088489
Gari -Yellow	0.024332	-0.072653	-0.066001	-0.089357
Gari-White	0.028449	0.080609	0.105608	-0.047211
Groundnut oil	0.002997	0.021004	0.050357	0.047455
Malt drinks	0.007418	0.053089	0.052723	0.028829
Milk powder	0.026254	0.038752	0.052267	-0.006918
Milk tinned (unsweetened)	0.065099	0.029779	0.033389	0.052502
Okra-fresh	0.001021	0.010336	0.007086	-0.017186
Onions	0.038711	0.067721	0.055729	0.008528
Orange/tangerine	0.053454	0.011208	0.004243	0.020314
Palm oil	0.004605	-0.015315	0.017361	0.039013
Plantains	0.075064	0.035438	0.106043	0.122559
Rice-Imported	0.047160	0.009581	-0.010254	-0.033201
Rice-local	0.049304	0.089217	0.034852	0.031416
Sachet water	-0.053358	0.041837	-0.093581	-0.086949
Soft drinks (Coca cola, spirit etc)	0.044827	0.005086	-0.017571	0.081171
Sugar	0.100876	0.008425	-0.028308	0.022939

Tea	-0.188362	0.064226	-0.155837	-0.080844
Tomato puree(canned)	-0.054485	-0.044221	-0.004699	-0.005860
Tomatoes	0.020659	0.012966	0.024084	0.003112
White beans	-0.052245	-0.062964	-0.021034	-0.058732
Yam-roots	-0.005656	-0.044430	0.053049	-0.001740

\

k	F 4-8	F 9-13	F 14-18	F 19-30
i				
(Cocoyam, Spinach, etc)	0.004383	-0.035575	0.007843	0.028245
Agricultural eggs	0.033623	-0.053427	-0.107071	-0.042087
Bananas	-0.016333	-0.035867	-0.004621	-0.028252
Beef	0.001525	0.004998	0.029513	0.034945
Bread	-0.021224	0.008890	0.018510	0.007968
Brown beans	-0.013461	-0.007905	0.011269	0.034928
Chocolate drinks	-0.019501	0.023503	-0.096586	0.056719
Condiments,(salt,spices,pepper, etc)	0.015075	-0.020592	0.070371	-0.054939
Fish-Dried	-0.003312	0.026928	0.047333	0.027351
Fish-Fresh	-0.025077	-0.067034	-0.038021	0.031967
Fish-Frozen	-0.002012	-0.030803	0.028372	0.027206
Fish-Smoked	0.010204	-0.029588	0.145062	-0.065502
Garden eggs/egg plant	0.065873	0.144870	0.036684	0.025786
Gari -Yellow	0.004719	-0.017886	-0.095526	-0.016012
Gari-White	-0.033572	-0.002572	0.057006	0.039819
Groundnut oil	0.022777	0.016139	0.041827	-0.006221
Malt drinks	0.004297	-0.003375	0.013592	0.026708
Milk powder	-0.015580	0.050438	-0.043978	0.030322
Milk tinned (unsweetened)	0.022248	0.006637	-0.043171	-0.028621
Okra-fresh	0.007024	-0.027783	-0.007426	0.030295
Onions	0.013981	0.023530	0.029815	0.015616
Orange/tangerine	0.062747	0.035989	0.041444	-0.045708
Palm oil	0.031787	0.021588	0.012405	0.004591
Plantains	0.036008	0.100552	0.057675	0.152550
Rice-Imported	0.051005	0.075407	0.072270	0.035737
Rice-local	0.140484	0.070627	0.011927	0.032740
Sachet water	-0.037186	-0.119118	-0.023541	-0.082914
Soft drinks (Coca cola, spirit etc)	0.023753	0.094971	0.041766	0.000882
Sugar	0.039221	0.113481	0.048585	-0.014638
Tea	-0.196202	0.058074	-0.026707	-0.085671
Tomato puree(canned)	-0.050040	0.001101	0.034821	-0.026613
Tomatoes	0.004883	0.000436	-0.018331	0.022071
White beans	-0.038058	-0.001737	-0.039247	-0.038623
Yam-roots	-0.019156	0.002941	-0.043965	0.000541

k	F 31-50	F 51+	log Hsize
i			

```
(Cocoyam, Spinach, etc)              -0.017427  0.040222  0.566638
Agricultural eggs                     0.053721 -0.089442  0.877207
Bananas                               0.061929 -0.045779  0.655963
Beef                                  0.068021  0.017202  0.538004
Bread                                 0.028410 -0.059618  0.697600
Brown beans                           0.043603  0.039199  0.878659
Chocolate drinks                      0.048133 -0.006635  0.849484
Condiments,(salt,spices,pepper, etc)  0.049714 -0.052162  0.646916
Fish-Dried                            0.048000 -0.025411  0.399808
Fish-Fresh                           -0.016569 -0.151240  0.495920
Fish-Frozen                           0.019353 -0.011803  0.454118
Fish-Smoked                           0.061397  0.096051  0.448971
Garden eggs/egg plant                 0.060789  0.128848 -0.042241
Gari -Yellow                         -0.042824 -0.105659  0.778344
Gari-White                            0.082678 -0.016844  0.606481
Groundnut oil                         0.009749 -0.009158  0.569290
Malt drinks                           0.010340  0.081463  0.387473
Milk powder                           0.044644  0.007753  0.830787
Milk tinned (unsweetened)             0.008090  0.076985  0.322839
Okra-fresh                            0.018894  0.034659  0.460350
Onions                               -0.007032 -0.010399  0.513672
Orange/tangerine                     -0.064142 -0.039780  0.495509
Palm oil                              0.019322  0.011080  0.539990
Plantains                             0.215641  0.113222  0.052553
Rice-Imported                         0.062528  0.013166  0.933506
Rice-local                            0.078112  0.118097  0.716899
Sachet water                         -0.012795 -0.137798  0.870446
Soft drinks (Coca cola, spirit etc)   0.043664  0.021899  0.427704
Sugar                                -0.138079 -0.157281  0.976634
Tea                                   0.024660 -0.043106  0.875037
Tomato puree(canned)                 -0.004714 -0.001105  0.580772
Tomatoes                              0.040905  0.015591  0.554454
White beans                          -0.081140 -0.035455  0.998716
Yam-roots                             0.034202  0.056262  0.696974
```

[33]:
```python
result4.delta.to_dataframe().unstack('k')
```

[33]:
```
                                         delta                               \
k                                        M 0-3     M 4-8    M 9-13   M 14-18
i
(Cocoyam, Spinach, etc)               0.059970 -0.026824 -0.128255 -0.060140
Agricultural eggs                     0.086844  0.059903  0.027056  0.020432
Bananas                               0.112941  0.019600  0.028835  0.030731
Beef                                  0.041234  0.017796  0.033165  0.038998
Bread                                -0.026338  0.063221  0.001727  0.057792
Brown beans                           0.014834  0.091460  0.068840  0.029102
Chicken                              -0.061032  0.033191 -0.050040 -0.038478
```

Chocolate drinks	-0.035630	0.055367	0.068203	0.017425
Condiments,(salt,spices,pepper, etc)	0.148230	0.006426	0.044869	-0.000825
Fish-Dried	0.155122	0.026463	0.001232	0.097397
Fish-Fresh	-0.080494	0.004656	0.041046	-0.004362
Fish-Frozen	0.058999	0.005119	0.006594	0.050949
Fish-Smoked	-0.041197	0.043157	-0.019277	0.053862
Garden eggs/egg plant	0.000739	0.037447	-0.130051	0.129735
Gari -Yellow	-0.049401	-0.014213	0.034010	0.051743
Gari-White	-0.027567	-0.005664	0.127520	0.021062
Goat	0.074985	-0.016815	-0.062995	0.118926
Groundnut oil	0.065055	0.054413	0.067312	0.091114
Malt drinks	0.068250	0.042481	0.056144	0.057165
Milk powder	0.009672	-0.018615	0.002166	0.021498
Milk tinned (unsweetened)	0.081237	0.063547	-0.009673	0.066668
Okra-fresh	0.000071	0.074140	0.041411	0.014963
Onions	0.028344	0.014653	-0.004037	0.051690
Orange/tangerine	0.062839	0.021212	0.037592	0.041828
Other vegetables (fresh or canned)	-0.024739	0.126333	0.053614	-0.010630
Palm oil	-0.008042	0.083789	0.004990	0.085870
Pineapples	0.034971	0.010693	0.042640	-0.050638
Plantains	0.070069	0.154872	0.108133	0.100389
Rice-Imported	0.031562	0.190569	0.048920	0.138436
Rice-local	0.235522	0.142063	0.106548	0.130315
Sachet water	-0.050036	0.011843	-0.071714	0.016340
Soft drinks (Coca cola, spirit etc)	0.019878	0.055400	0.020369	0.071855
Sugar	0.069818	0.215184	0.077732	0.107936
Sweet Potatoes	0.022577	0.039896	0.034979	0.175059
Tea	0.128394	-0.182782	-0.054116	-0.003283
Tomato puree(canned)	0.029677	-0.028415	-0.069827	-0.022324
Tomatoes	-0.011822	0.041274	0.022443	0.019055
White beans	0.101548	0.085012	0.024664	0.099476
Yam-roots	0.013173	0.034806	0.067913	0.059875

\

k	M 19-30	M 31-50	M 51+	F 0-3
i				
(Cocoyam, Spinach, etc)	-0.066222	-0.203679	-0.003154	-0.110133
Agricultural eggs	0.009668	0.162421	0.187265	-0.030647
Bananas	0.057242	0.092288	0.082695	-0.005638
Beef	0.083915	0.071213	0.120941	0.023505
Bread	0.043891	0.078987	0.101278	0.003712
Brown beans	0.034687	0.146752	0.119398	0.011646
Chicken	-0.003820	0.006639	0.103351	-0.066502
Chocolate drinks	-0.019891	0.009374	0.086222	-0.002688
Condiments,(salt,spices,pepper, etc)	0.024444	0.072649	0.062000	0.017190
Fish-Dried	0.084522	0.179060	0.158097	-0.045523
Fish-Fresh	0.039594	0.123120	0.134129	-0.087933

Fish-Frozen	0.073138	0.092317	0.116882	0.081598
Fish-Smoked	-0.000638	-0.004062	0.008987	0.033863
Garden eggs/egg plant	0.051826	0.047545	-0.003457	-0.133323
Gari -Yellow	0.049269	0.077268	0.077347	-0.074477
Gari-White	0.034031	0.170529	0.045180	0.019730
Goat	0.052371	0.015405	0.084998	-0.025937
Groundnut oil	0.070049	0.160545	0.064058	0.024082
Malt drinks	0.088148	0.058488	0.110959	-0.039787
Milk powder	0.030419	0.088972	0.127245	0.118727
Milk tinned (unsweetened)	-0.001520	-0.028025	0.052931	-0.031587
Okra-fresh	0.079703	0.096141	0.093903	-0.036283
Onions	0.043624	0.045234	0.026020	0.026170
Orange/tangerine	0.052159	0.084348	0.158066	0.049671
Other vegetables (fresh or canned)	0.097943	0.068959	0.035589	-0.108061
Palm oil	0.016412	0.026375	0.055513	0.018027
Pineapples	0.039806	0.049608	0.107058	-0.094647
Plantains	0.140681	0.117783	0.182774	0.065346
Rice-Imported	0.090646	0.245140	0.178468	0.108508
Rice-local	0.056572	0.112045	0.117969	0.101608
Sachet water	0.021713	0.046515	0.025247	-0.114484
Soft drinks (Coca cola, spirit etc)	0.079734	0.104544	0.111133	-0.013417
Sugar	0.145846	0.151165	0.040849	0.037970
Sweet Potatoes	-0.015309	-0.048072	-0.074557	0.110313
Tea	-0.034689	-0.008443	-0.022654	-0.116460
Tomato puree(canned)	0.024931	0.083855	0.003042	-0.078219
Tomatoes	0.033369	0.014018	0.016700	0.010866
White beans	0.085855	0.056639	0.113576	0.093698
Yam-roots	0.040960	0.069798	0.050798	-0.063235

\

k	F 4-8	F 9-13	F 14-18
i			
(Cocoyam, Spinach, etc)	-0.029828	-3.905910e-02	-0.065547
Agricultural eggs	0.037358	3.864903e-02	0.069396
Bananas	0.001193	8.091823e-02	0.027608
Beef	0.048921	9.010419e-02	0.094665
Bread	0.050708	5.207216e-02	0.055545
Brown beans	-0.004123	1.634427e-01	0.118931
Chicken	-0.033737	-3.769419e-02	-0.073332
Chocolate drinks	0.054594	-8.011901e-02	0.082074
Condiments,(salt,spices,pepper, etc)	0.115625	1.286864e-01	-0.000847
Fish-Dried	0.088062	2.077112e-02	0.109638
Fish-Fresh	0.044270	-3.795449e-02	0.063810
Fish-Frozen	0.036344	8.648224e-02	0.061256
Fish-Smoked	0.124475	3.642729e-02	0.111974
Garden eggs/egg plant	0.061322	-2.445847e-02	-0.056014
Gari -Yellow	0.085948	6.789672e-02	-0.022039

Gari-White	0.018731	1.112754e-01	0.061417
Goat	-0.056854	4.049850e-02	0.002154
Groundnut oil	0.060424	7.836431e-02	0.123231
Malt drinks	0.073434	1.231474e-02	0.042965
Milk powder	-0.045061	-3.552280e-07	0.014775
Milk tinned (unsweetened)	0.033148	-1.705808e-02	0.100396
Okra-fresh	0.082160	5.184306e-02	0.078115
Onions	0.069938	5.697349e-02	0.066008
Orange/tangerine	0.098695	6.934952e-02	0.061431
Other vegetables (fresh or canned)	0.074483	1.515885e-01	-0.000268
Palm oil	0.046437	7.582694e-02	0.127327
Pineapples	-0.029737	5.125779e-02	-0.007099
Plantains	0.059474	5.915016e-02	0.141509
Rice-Imported	0.097398	1.346403e-01	0.205625
Rice-local	0.155178	2.456903e-02	0.084441
Sachet water	-0.051710	-2.654213e-02	0.072687
Soft drinks (Coca cola, spirit etc)	0.027401	3.549516e-02	0.076675
Sugar	0.150018	9.698721e-02	0.120533
Sweet Potatoes	0.067849	3.063824e-02	0.022007
Tea	-0.017558	-1.496209e-01	-0.016611
Tomato puree(canned)	-0.009420	2.706373e-02	0.052196
Tomatoes	0.045620	1.480582e-02	0.074541
White beans	0.102571	6.466621e-02	0.023207
Yam-roots	0.052264	3.412164e-02	0.094128

k	F 19-30	F 31-50	F 51+	log Hsize
i				
(Cocoyam, Spinach, etc)	-0.039992	0.074851	-0.026010	0.455216
Agricultural eggs	0.056060	0.043409	0.007082	0.364332
Bananas	0.054544	0.030339	0.028244	0.183273
Beef	0.083271	0.064280	0.019953	0.245994
Bread	0.050361	0.044239	0.009327	0.390736
Brown beans	0.031816	0.063036	0.067179	0.218931
Chicken	0.013690	0.148931	0.065574	0.360872
Chocolate drinks	0.076718	-0.059831	-0.013869	0.544797
Condiments,(salt,spices,pepper, etc)	0.011411	0.031594	0.002202	0.341355
Fish-Dried	0.075742	0.061020	0.044019	0.192434
Fish-Fresh	0.146593	0.098645	0.152332	0.094940
Fish-Frozen	0.016630	0.085143	0.126944	0.180569
Fish-Smoked	0.019259	-0.027527	0.113652	0.311213
Garden eggs/egg plant	0.053827	0.064776	-0.005209	0.154766
Gari -Yellow	0.047046	-0.018677	-0.048561	0.444479
Gari-White	0.062529	0.043093	0.019069	0.330706
Goat	0.039837	-0.048295	-0.069985	0.308494
Groundnut oil	0.073113	0.031765	0.037063	0.132225
Malt drinks	0.009565	0.056179	0.054838	0.184947

```
Milk powder                               0.022588 -0.007110 -0.095894  0.601637
Milk tinned (unsweetened)                 0.023336  0.009617 -0.005414  0.177990
Okra-fresh                                0.087881  0.027587  0.023591  0.175968
Onions                                    0.031761  0.047450  0.072197  0.322043
Orange/tangerine                          0.043334  0.014848  0.013041  0.211652
Other vegetables (fresh or canned)        0.079156  0.012786  0.021057  0.229056
Palm oil                                  0.014345  0.049636  0.039322  0.286405
Pineapples                                0.014889  0.098692  0.038717  0.180877
Plantains                                 0.016716  0.100259  0.100861  0.065388
Rice-Imported                             0.134045  0.140734  0.083182  0.504086
Rice-local                                0.023051  0.024474  0.149101  0.649946
Sachet water                             -0.010634 -0.051092 -0.059065  0.427609
Soft drinks (Coca cola, spirit etc)       0.066147  0.000801 -0.068077  0.223496
Sugar                                     0.112642 -0.039974 -0.133297  0.369131
Sweet Potatoes                            0.001933 -0.009569  0.224169  0.229141
Tea                                      -0.085098  0.146087  0.124593  0.486475
Tomato puree(canned)                      0.016105  0.064547  0.034630  0.303025
Tomatoes                                  0.021682  0.062196  0.065249  0.360306
White beans                               0.060743  0.069916  0.100083  0.270242
Yam-roots                                 0.040643  0.024213  0.053291  0.366944
```

Also the good-time constants a_{it} (this captures the effects of prices)

[34]:
```python
result1.a.to_dataframe().unstack('i')
```

[34]:
```
                a                                                             \
i           Bread Condiments,(salt,spices,pepper, etc) Groundnut oil    Onions
t    m
2010 1   4.132231                                 3.761195      4.355794  3.539646
2012 1   4.028556                                 3.931693      3.893221  3.369722
2015 1   4.234435                                 4.438573      4.592232  3.224604
2018 1   4.443681                                 4.508119      4.499117  3.743854


i         Palm oil Rice-local     Sugar  Tomatoes White beans
t    m
2010 1   5.324580   2.455150  2.376955  2.690814    4.727037
2012 1   5.162490   2.555986  1.724645  2.469391    5.317079
2015 1   5.175408   2.948691  2.232912  2.480203    5.461796
2018 1   5.471405   2.971027  2.077796  2.694839    5.639164
```

2.2.2 Second step of Estimation

The second step involves using Singular Value Decomposition to find the rank one matrix that best approximates the residuals e_{it}^j. This can be interpreted as

$$-\beta_i \log \lambda_t^j,$$

where the $\log \lambda_t^j$ is the log of the marginal utility of expenditures (MUE) for household j at time t, and where β_i are the corresponding "Frisch elasticities" that tell us how much demand changes as the MUE falls.

Estimates can also be computed as a one-liner:

[35]:
```
result1.get_beta(as_df=True)
```

[35]:
```
i
Bread                                  0.249771
Condiments,(salt,spices,pepper, etc)   0.150775
Groundnut oil                          0.105369
Onions                                 0.401157
Palm oil                               0.160994
Rice-local                             0.018439
Sugar                                  1.200716
Tomatoes                               0.271085
White beans                            0.103897
Name: beta, dtype: float64
```

[36]:
```
result2.get_beta(as_df=True)
```

[36]:
```
i
Agricultural eggs                      0.088323
Bananas                                0.222667
Beef                                   0.001545
Bread                                  0.154059
Chocolate drinks                       0.143203
Condiments,(salt,spices,pepper, etc)   0.919390
Fish-Dried                            -0.033787
Fish-Frozen                           -0.025459
Fish-Smoked                            0.204248
Gari-White                            -0.114889
Groundnut oil                          0.167468
Malt drinks                            0.142770
Milk powder                            0.157325
Okra-fresh                             0.092925
Onions                                 0.333897
Orange/tangerine                       0.302125
Other vegetables (fresh or canned)    -0.208509
Palm oil                               0.134580
Plantains                              0.066261
Rice-local                             0.075686
Sachet water                           0.048858
Soft drinks (Coca cola, spirit etc)    0.183064
Sugar                                  0.894168
Tomato puree(canned)                   0.253551
Tomatoes                               0.170956
```

```
White beans                              0.042951
Yam-roots                                0.025025
Name: beta, dtype: float64
```

[37]:
```python
result3.get_beta(as_df=True)
```

[37]:
```
i
(Cocoyam, Spinach, etc)                  0.215651
Agricultural eggs                        0.338687
Bananas                                  0.233286
Beef                                     0.186948
Bread                                    0.251313
Brown beans                              0.006532
Chocolate drinks                         0.622164
Condiments,(salt,spices,pepper, etc)     0.347105
Fish-Dried                               0.129598
Fish-Fresh                               0.163382
Fish-Frozen                              0.113992
Fish-Smoked                              0.190246
Garden eggs/egg plant                    0.023959
Gari -Yellow                            -0.014658
Gari-White                              -0.039566
Groundnut oil                            0.115448
Malt drinks                              0.218365
Milk powder                              0.752278
Milk tinned (unsweetened)                0.243653
Okra-fresh                               0.191827
Onions                                   0.227210
Orange/tangerine                         0.227180
Palm oil                                 0.101873
Plantains                                0.014821
Rice-Imported                           -0.080449
Rice-local                               0.032910
Sachet water                             0.519180
Soft drinks (Coca cola, spirit etc)      0.239942
Sugar                                    0.740135
Tea                                      0.346896
Tomato puree(canned)                     0.204246
Tomatoes                                 0.221208
White beans                              0.139314
Yam-roots                                0.005897
Name: beta, dtype: float64
```

[38]:
```python
result4.get_beta(as_df=True)
```

[38]:
```
i
(Cocoyam, Spinach, etc)                  0.214391
```

```
Agricultural eggs                         0.500903
Bananas                                   0.310199
Beef                                      0.261743
Bread                                     0.262224
Brown beans                               0.231783
Chicken                                   0.212681
Chocolate drinks                          0.503696
Condiments,(salt,spices,pepper, etc)      0.419001
Fish-Dried                                0.378614
Fish-Fresh                                0.330833
Fish-Frozen                               0.200550
Fish-Smoked                               0.295947
Garden eggs/egg plant                     0.288953
Gari -Yellow                              0.247140
Gari-White                                0.274865
Goat                                      0.208295
Groundnut oil                             0.325517
Malt drinks                               0.328769
Milk powder                               0.607743
Milk tinned (unsweetened)                 0.285548
Okra-fresh                                0.257580
Onions                                    0.346715
Orange/tangerine                          0.305852
Other vegetables (fresh or canned)        0.254765
Palm oil                                  0.267732
Pineapples                                0.253503
Plantains                                 0.285294
Rice-Imported                             0.322117
Rice-local                                0.352096
Sachet water                              0.280185
Soft drinks (Coca cola, spirit etc)       0.310553
Sugar                                     0.537008
Sweet Potatoes                            0.385539
Tea                                       0.334811
Tomato puree(canned)                      0.335210
Tomatoes                                  0.286938
White beans                               0.313100
Yam-roots                                 0.266059
Name: beta, dtype: float64
```

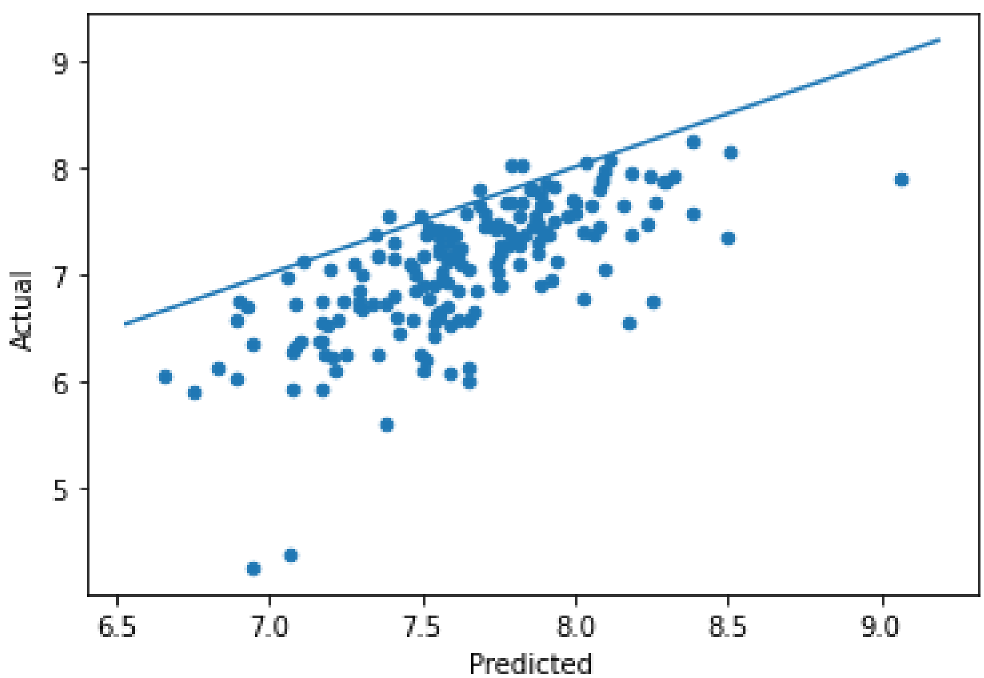
That's all there is to estimation! Note that we didn't estimate demands for all goods—lots of goods didn't have enough observations, and were automatically dropped. (This can be controlled using the `min_proportion_items` and `min_xproducts` attributes when one instantiates the result object.)

2.2.3 Assessment of Fit

Now, let's see how we did, by comparing total expenditures predicted by the model we've estimated with actual total expenditures:

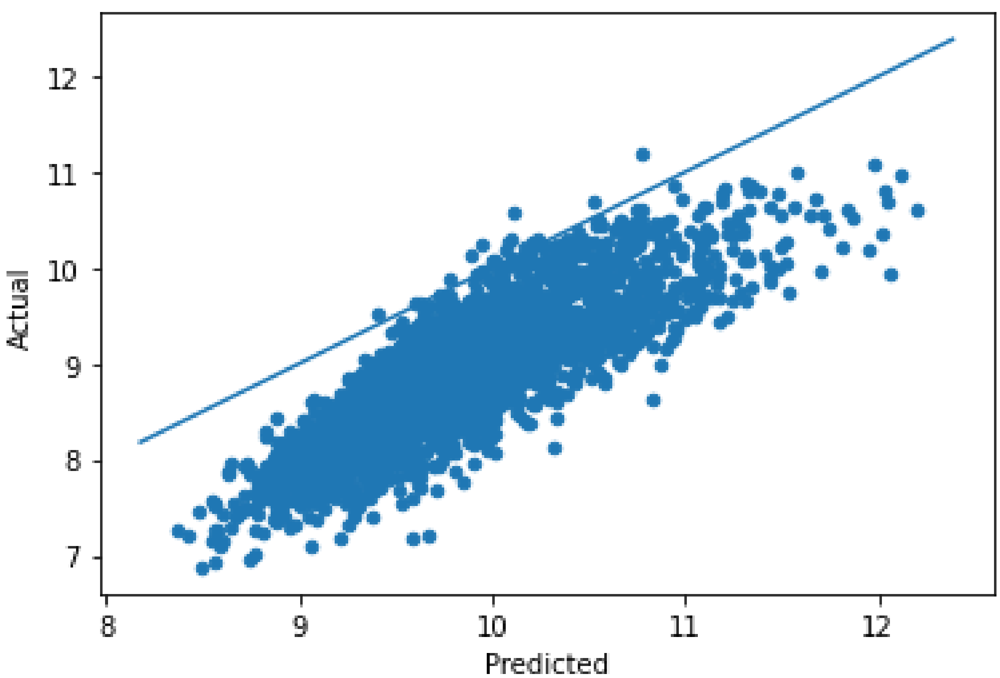
[39]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import matplotlib.cm as cm

xbar = np.exp(result1.y).sum(['m','i']).to_dataframe('xbar').replace(0,np.nan).
 ↪squeeze()
xhat = result1.get_predicted_expenditures().sum(['m','i']).to_dataframe('xhat').
 ↪replace(0,np.nan).squeeze()

# Make dataframe of actual & predicted
df = pd.DataFrame({'Actual':np.log(xbar),'Predicted':np.log(xhat)})

df.plot.scatter(x='Predicted',y='Actual')

# Add 45 degree line
v = plt.axis()
vmin = np.max([v[0],v[2]])
vmax = np.max([v[1],v[3]])
plt.plot([vmin,vmax],[vmin,vmax])
```

[39]:
```
[<matplotlib.lines.Line2D at 0x7f5376858550>]
```



[40]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import matplotlib.cm as cm

xbar = np.exp(result4.y).sum(['m','i']).to_dataframe('xbar').replace(0,np.nan).
↪squeeze()
xhat = result4.get_predicted_expenditures().sum(['m','i']).to_dataframe('xhat').
↪replace(0,np.nan).squeeze()

# Make dataframe of actual & predicted
df = pd.DataFrame({'Actual':np.log(xbar),'Predicted':np.log(xhat)})

df.plot.scatter(x='Predicted',y='Actual')

# Add 45 degree line
v = plt.axis()
vmin = np.max([v[0],v[2]])
vmax = np.max([v[1],v[3]])
plt.plot([vmin,vmax],[vmin,vmax])
```

[40]:
```
[<matplotlib.lines.Line2D at 0x7f5376858e20>]
```



[41]:
```
result1.to_dataset('icrisat.ds')
```

[41]:
```
<xarray.Dataset>
Dimensions:            (j: 3197, i: 9, k: 15, t: 4, m: 1, kp: 15)
Coordinates:
  * j                  (j) int64 10005 10009 10013 10022 … 379094 379096 379127
  * t                  (t) int64 2010 2012 2015 2018
  * m                  (m) int64 1
  * i                  (i) <U36 'Bread' … 'White beans'
  * k                  (k) <U9 'M 0-3' 'M 4-8' 'M 9-13' … 'F 51+' 'log Hsize'
  * kp                 (kp) <U9 'M 0-3' 'M 4-8' 'M 9-13' … 'F 51+' 'log Hsize'
Data variables: (12/20)
    alpha              object None
    beta               (i) float64 0.2498 0.1508 0.1054 … 1.201 0.2711 0.1039
    delta              (k, i) float64 -0.003088 0.2643 0.1028 … 1.308 -0.433
    prices             object None
    characteristics    (k, j, t, m) float64 nan nan nan 0.0 … nan nan nan 1.386
    loglambdas         (j, t, m) float64 nan nan nan nan nan … nan nan nan nan
    …                  …
    se_beta            object None
    se_alpha           object None
    se_a               (i, t, m) float64 0.08262 0.1346 … 0.07711 0.08249
    y                  (i, j, t, m) float64 nan nan nan nan … nan nan nan nan
    logp               object None
    z                  (k, j, t, m) float64 nan nan nan 0.0 … nan nan nan 1.386
```

3 Nutritional Data

Read in the consumption data for Nigerian households.

[42]:
```python
fdc_table = '1ed8FASRCkN9KwTWTvMzKT6UT4jWbSSZQEwZEmXCt8IQ'

fdc_codes = read_sheets(fdc_table,sheet="Sheet1")

consumption = read_sheets(nigeria_consumption,sheet='Consumption')
consumption.insert(loc=2, column='m', value=1)
consumption = consumption.set_index(['t', 'j', 'm'])
consumption = consumption.drop(columns=['Canned'])
consumption
```

```
Key available for students@eep153.iam.gserviceaccount.com.
Key available for students@eep153.iam.gserviceaccount.com.
```

[42]:
```
                              u  (Cocoyam, Spinach, etc)  Agricultural eggs  \
t    j     m
2010 10001 1          Kilograms                      NaN               0.89
           1             Litres                      NaN                NaN
     10002 1          Kilograms                      NaN               0.89
           1             Litres                      NaN                NaN
```

```
     10003  1          Kilograms                      NaN              0.44
…                          …                    …                …
2018 379148 1      2. GRAMS (GR)                      NaN               NaN
     379151 1  1. KILOGRAMS (KG)                      NaN               NaN
            1      3. LITRES (L)                      NaN               NaN
     379154 1  1. KILOGRAMS (KG)                      NaN               NaN
     379155 1  1. KILOGRAMS (KG)                      NaN               NaN

               Animal fat  Apples  Avocado pear  Baby milk powder  Bananas  \
t    j      m
2010 10001  1         NaN     NaN           NaN               NaN     1.30
            1         NaN     NaN           NaN               NaN      NaN
     10002  1         NaN     NaN           NaN               NaN     1.30
            1         NaN     NaN           NaN               NaN      NaN
     10003  1         NaN     NaN           NaN               NaN     0.35
…                    …      …            …                …      …
2018 379148 1         NaN     NaN           NaN               NaN      NaN
     379151 1         NaN     NaN           NaN               NaN      NaN
            1         NaN     NaN           NaN               NaN      NaN
     379154 1         NaN     NaN           NaN               NaN      NaN
     379155 1         NaN     NaN           NaN               NaN      NaN

                Beef  Beer (local and imported)  …  Sweet Potatoes   Tea  \
t    j      m                                    …
2010 10001  1    1.0                        NaN  …             1.5   NaN
            1    NaN                       2.25  …             NaN   NaN
     10002  1    2.0                        NaN  …             1.8  0.12
            1    NaN                       9.00  …             NaN   NaN
     10003  1    0.3                        NaN  …             1.4  0.30
…              …                          …  …               …   …
2018 379148 1  500.0                        NaN  …             NaN   NaN
     379151 1    NaN                        NaN  …             NaN   NaN
            1    NaN                        NaN  …             NaN   NaN
     379154 1    1.0                        NaN  …             NaN   NaN
     379155 1    1.0                        NaN  …             NaN   NaN

               Tomato puree(canned)  Tomatoes  Watermelon  Wheat flour  \
t    j      m
2010 10001  1                  0.42       1.0         NaN          NaN
            1                   NaN       NaN         NaN          NaN
     10002  1                  0.56       1.0         NaN          NaN
            1                   NaN       NaN         NaN          NaN
     10003  1                  0.21       1.0         NaN          NaN
…                                …       …          …           …
2018 379148 1                   NaN       NaN         NaN          NaN
     379151 1                   NaN       NaN         NaN          2.0
            1                   NaN       NaN         NaN          NaN
```

```
     379154 1                  NaN       NaN         NaN          NaN
     379155 1                  NaN       NaN         NaN          NaN

                 White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10001  1            3.0             NaN        NaN       16.0
            1            NaN             NaN        NaN        NaN
     10002  1            2.0             NaN        NaN       13.8
            1            NaN             NaN        NaN        NaN
     10003  1            0.6             NaN        NaN        4.6
…                          …              …         …         …
2018 379148 1            NaN             NaN        NaN        NaN
     379151 1            NaN             NaN        NaN        NaN
            1            NaN             NaN        NaN        NaN
     379154 1            NaN             NaN        NaN        NaN
     379155 1            NaN             NaN        NaN        NaN

[39172 rows x 124 columns]
```

3.0.1 Create a dictionary that will map all of the food units to their equivalent values in hectograms.

Find every unit of measure used in the data in order to convert them to hectograms.

[43]:
```
food_units_df = consumption.reset_index()
unique_food_units = np.unique(pd.DataFrame(food_units_df['u'])).tolist()
unique_food_units
```

[43]:
```
['1. KILOGRAMS (KG)',
 '2. GRAMS (G)',
 '2. GRAMS (GR)',
 '3. LITRES (L)',
 '4. CENTILITRES (CL)',
 'Basin: Big/Large (40 kg)',
 'Basin: Medium (25 kg)',
 'Basin: Small (10 kg)',
 'Basket: Big (50 kg)',
 'Basket: Medium (30 kg)',
 'Basket: Small (15 kg)',
 'Bunch of Plantain/FFB: Small (5 kg)',
 'Bunch of plantain/FFB: Big (15 kg)',
 'Bunch of plantain/FFB: Medium (8 kg)',
 'Grams',
 'Kilograms',
 'Litres',
 'Mililitre',
 'Sack/Bag: Medium (50 kg)',
```

```
 'Sack/Bag: Small (20 kg)',
 'Tuber of Yam: Big/Large (8 kg)',
 'Tuber of Yam: Medium (5 kg)',
 'Tuber of Yam: Small (3 kg)',
 'Wheel Barrow: Small (60 kg)',
 'centilitre (cl)',
 'grams (g)',
 'kilogram (kg)',
 'litre (l)']
```

Create the dictionary itself.

For example: food_unit_map_dict['Kilograms'] = 10 because 1 kilogram = 10 hectograms

[44]:
```
values = [10, 0.01, 0.01, 10, 0.1, 400, 250, 100, 500, 300, 150, 50, 150, 80, 0.
 ↪01, 10, 10, 0.01, 500, 200, 80, 50, 30, 600, 0.1, 0.01, 10, 10]
food_unit_map_dict = dict(zip(unique_food_units, values))
food_unit_map_dict
```

[44]:
```
{'1. KILOGRAMS (KG)': 10,
 '2. GRAMS (G)': 0.01,
 '2. GRAMS (GR)': 0.01,
 '3. LITRES (L)': 10,
 '4. CENTILITRES (CL)': 0.1,
 'Basin: Big/Large (40 kg)': 400,
 'Basin: Medium (25 kg)': 250,
 'Basin: Small (10 kg)': 100,
 'Basket: Big (50 kg)': 500,
 'Basket: Medium (30 kg)': 300,
 'Basket: Small (15 kg)': 150,
 'Bunch of Plantain/FFB: Small (5 kg)': 50,
 'Bunch of plantain/FFB: Big (15 kg)': 150,
 'Bunch of plantain/FFB: Medium (8 kg)': 80,
 'Grams': 0.01,
 'Kilograms': 10,
 'Litres': 10,
 'Mililitre': 0.01,
 'Sack/Bag: Medium (50 kg)': 500,
 'Sack/Bag: Small (20 kg)': 200,
 'Tuber of Yam: Big/Large (8 kg)': 80,
 'Tuber of Yam: Medium (5 kg)': 50,
 'Tuber of Yam: Small (3 kg)': 30,
 'Wheel Barrow: Small (60 kg)': 600,
 'centilitre (cl)': 0.1,
 'grams (g)': 0.01,
 'kilogram (kg)': 10,
 'litre (l)': 10}
```

Convert all of the original units to hectograms in a data frame.

[45]:
```
consumption_in_hect = consumption.set_index('u', append=True)

for index in consumption_in_hect.index:
    unit_used = index[3]
    multiplier = food_unit_map_dict[unit_used]
    consumption_in_hect.loc[index] *= multiplier

consumption_in_hect = consumption_in_hect.reset_index().set_index(['t', 'j',␣
↪'m'])

# Change all the units to hectograms in the data frame
consumption_in_hect['u'] = consumption_in_hect['u'].apply(lambda x:␣
↪'Hectograms')
consumption_in_hect
```

[45]:
```
                        u  (Cocoyam, Spinach, etc)  Agricultural eggs  \
t    j      m
2010 10001  1  Hectograms                      NaN                8.9
            1  Hectograms                      NaN                NaN
     10002  1  Hectograms                      NaN                8.9
            1  Hectograms                      NaN                NaN
     10003  1  Hectograms                      NaN                4.4
…                       …                        …                  …
2018 379148 1  Hectograms                      NaN                NaN
     379151 1  Hectograms                      NaN                NaN
            1  Hectograms                      NaN                NaN
     379154 1  Hectograms                      NaN                NaN
     379155 1  Hectograms                      NaN                NaN

               Animal fat  Apples  Avocado pear  Baby milk powder  Bananas  \
t    j      m
2010 10001  1         NaN     NaN           NaN               NaN     13.0
            1         NaN     NaN           NaN               NaN      NaN
     10002  1         NaN     NaN           NaN               NaN     13.0
            1         NaN     NaN           NaN               NaN      NaN
     10003  1         NaN     NaN           NaN               NaN      3.5
…                       …       …             …                 …        …
2018 379148 1         NaN     NaN           NaN               NaN      NaN
     379151 1         NaN     NaN           NaN               NaN      NaN
            1         NaN     NaN           NaN               NaN      NaN
     379154 1         NaN     NaN           NaN               NaN      NaN
     379155 1         NaN     NaN           NaN               NaN      NaN

               Beef  Beer (local and imported)  …  Sweet Potatoes  Tea  \
t    j      m                                   …
```

```
2010 10001  1  10.0                       NaN  …          15.0  NaN
            1   NaN                      22.5  …           NaN  NaN
     10002  1  20.0                       NaN  …          18.0  1.2
            1   NaN                      90.0  …           NaN  NaN
     10003  1   3.0                       NaN  …          14.0  3.0
…              …                        …  …            …  …
2018 379148 1   5.0                       NaN  …           NaN  NaN
     379151 1   NaN                       NaN  …           NaN  NaN
            1   NaN                       NaN  …           NaN  NaN
     379154 1  10.0                       NaN  …           NaN  NaN
     379155 1  10.0                       NaN  …           NaN  NaN

               Tomato puree(canned)  Tomatoes  Watermelon  Wheat flour  \
t    j      m
2010 10001  1                   4.2      10.0         NaN          NaN
            1                   NaN       NaN         NaN          NaN
     10002  1                   5.6      10.0         NaN          NaN
            1                   NaN       NaN         NaN          NaN
     10003  1                   2.1      10.0         NaN          NaN
…                             …       …           …          …
2018 379148 1                   NaN       NaN         NaN          NaN
     379151 1                   NaN       NaN         NaN         20.0
            1                   NaN       NaN         NaN          NaN
     379154 1                   NaN       NaN         NaN          NaN
     379155 1                   NaN       NaN         NaN          NaN

               White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10001  1         30.0             NaN        NaN      160.0
            1          NaN             NaN        NaN        NaN
     10002  1         20.0             NaN        NaN      138.0
            1          NaN             NaN        NaN        NaN
     10003  1          6.0             NaN        NaN       46.0
…                  …             …        …        …
2018 379148 1          NaN             NaN        NaN        NaN
     379151 1          NaN             NaN        NaN        NaN
            1          NaN             NaN        NaN        NaN
     379154 1          NaN             NaN        NaN        NaN
     379155 1          NaN             NaN        NaN        NaN

[39172 rows x 124 columns]
```

Once all the foods are in the same unit, we can group the rows so that there is only one row per household.

[46]:
```
consumption_in_hect = consumption_in_hect.groupby(level=[0, 1]).sum()
consumption_in_hect.insert(loc=2, column='m', value=1)
```

```
consumption_in_hect = consumption_in_hect.reset_index().set_index(['t', 'j',␣
↪'m'])
consumption_in_hect = consumption_in_hect.replace(0, np.nan)
```

[47]:
```
c_in_h = consumption_in_hect.index.tolist()
food = expend[expend.index.isin(c_in_h)]
food = food.drop(columns=['Total Expenditures', 'People per HH', 'Expenditures␣
↪per capita', 'Canned'])
```

[48]:
```
food
```

[48]:
```
i              (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
t    j      m
2010 10001  1                      NaN              280.0         NaN     NaN
     10002  1                      NaN              280.0         NaN     NaN
     10003  1                      NaN              180.0         NaN     NaN
     10004  1                      NaN              180.0         NaN     NaN
     10006  1                      NaN                NaN         NaN     NaN
…                                    …                  …           …       …
2018 379146 1                      NaN             1100.0         NaN     NaN
     379148 1                    100.0                NaN         NaN     NaN
     379151 1                      NaN              900.0         NaN     NaN
     379154 1                    200.0             1200.0         NaN     NaN
     379155 1                    100.0              950.0         NaN     NaN

i              Avocado pear  Baby milk powder  Bananas    Beef  \
t    j      m
2010 10001  1           NaN               NaN    200.0   500.0
     10002  1           NaN               NaN    180.0  1200.0
     10003  1           NaN               NaN    100.0   500.0
     10004  1           NaN               NaN    100.0   500.0
     10006  1           NaN               NaN    300.0   300.0
…                         …                 …        …       …
2018 379146 1           NaN               NaN      NaN     NaN
     379148 1           NaN               NaN      NaN   700.0
     379151 1           NaN               NaN    500.0     NaN
     379154 1           NaN               NaN      NaN  1300.0
     379155 1           NaN               NaN      NaN  1400.0

i              Beer (local and imported)  Biscuits  …  Sweet Potatoes  \
t    j      m                                       …
2010 10001  1                      540.0       NaN  …           150.0
     10002  1                     2000.0       NaN  …           200.0
     10003  1                        NaN       NaN  …           200.0
     10004  1                        NaN       NaN  …             NaN
     10006  1                        NaN       NaN  …             NaN
…                                      …         …  …               …
```

```
2018 379146 1                    NaN       NaN  …             NaN
     379148 1                    NaN       NaN  …             NaN
     379151 1                    NaN       NaN  …             NaN
     379154 1                    NaN       NaN  …             NaN
     379155 1                    NaN       NaN  …             NaN

i                 Tea  Tomato puree(canned)  Tomatoes  Watermelon  Wheat flour  \
t    j      m
2010 10001  1     NaN                 150.0     150.0         NaN          NaN
     10002  1   140.0                 240.0     120.0         NaN          NaN
     10003  1    60.0                  90.0     100.0         NaN          NaN
     10004  1    30.0                  60.0     100.0         NaN          NaN
     10006  1   650.0                   NaN     400.0         NaN          NaN
…               …                    …       …         …           …
2018 379146 1     NaN                   NaN       NaN       500.0          NaN
     379148 1     NaN                  60.0     200.0       150.0          NaN
     379151 1     NaN                 150.0     600.0       600.0        750.0
     379154 1     NaN                   NaN     100.0       200.0          NaN
     379155 1     NaN                   NaN     300.0       200.0          NaN

i               White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10001  1         600.0             NaN        NaN     1500.0
     10002  1         400.0             NaN        NaN     1200.0
     10003  1         100.0             NaN        NaN      400.0
     10004  1         100.0             NaN        NaN      400.0
     10006  1           NaN             NaN        NaN      400.0
…                     …              …        …          …
2018 379146 1           NaN             NaN        NaN     1800.0
     379148 1           NaN             NaN        NaN     1600.0
     379151 1        1600.0             NaN        NaN     3500.0
     379154 1           NaN             NaN        NaN      650.0
     379155 1           NaN             NaN        NaN     2500.0

[17023 rows x 123 columns]
```

[49]:
```
divided = food.div(consumption_in_hect)
prices = divided.mean(axis=0)
prices
```

[49]:
```
i
(Cocoyam, Spinach, etc)      33.748964
Agricultural eggs           121.522371
Animal fat                   85.000000
Apples                      160.000000
Avocado pear                 14.849578
                                …
```

```
Wheat flour                395.163283
White beans                 21.931440
Wild game meat              96.238442
Yam flour                   27.317893
Yam-roots                   32.820847
Length: 123, dtype: float64
```

[50]:
```python
pricedf = pd.DataFrame(prices)
pricedf.reset_index().set_index('i')
```

[50]:
```
                                  0
i
(Cocoyam, Spinach, etc)   33.748964
Agricultural eggs        121.522371
Animal fat                85.000000
Apples                   160.000000
Avocado pear              14.849578
…                               …
Wheat flour              395.163283
White beans               21.931440
Wild game meat            96.238442
Yam flour                 27.317893
Yam-roots                 32.820847

[123 rows x 1 columns]
```

[51]:
```python
avghh = hh_charQ1.drop(columns=['log Hsize']).mean(axis=0)
avghh
```

[51]:
```
k
M 0-3      0.414983
M 4-8      0.694865
M 9-13     0.623527
M 14-18    0.513889
M 19-30    0.553662
M 31-50    0.506103
M 51+      0.394150
F 0-3      0.385732
F 4-8      0.648359
F 9-13     0.552820
F 14-18    0.378367
F 19-30    0.640572
F 31-50    0.757786
F 51+      0.295244
dtype: float64
```

[52]:
```
dri_mins_sheet = '1XJRHTnxNJwrUXperIhwrwDp1HcVxPEVoQobYDmjg9Qw'
dri_mins = read_sheets(dri_mins_sheet,sheet='diet_minimums')
dri_mins = dri_mins.reset_index(drop=True).set_index('Nutrition').
↪drop('Source', axis=1)
dri_mins['M 0-3'] = dri_mins['C 1-3']
dri_mins['F 0-3'] = dri_mins['C 1-3']
dri_mins = dri_mins.drop(columns=['C 1-3'])
dri_mins
```

```
Key available for students@eep153.iam.gserviceaccount.com.
```

[52]:
```
                                F 4-8   M 4-8  F 9-13  M 9-13  F 14-18  \
Nutrition
Energy                         1200.0  1400.0  1600.0  1800.0   1800.0
Protein                          19.0    19.0    34.0    34.0     46.0
Fiber, total dietary             16.8    19.6    22.4    25.2     25.2
Folate, DFE                     200.0   200.0   300.0   300.0    400.0
Calcium, Ca                    1000.0  1000.0  1300.0  1300.0   1300.0
Carbohydrate, by difference     130.0   130.0   130.0   130.0    130.0
Iron, Fe                         10.0    10.0     8.0     8.0     15.0
Magnesium, Mg                   130.0   130.0   240.0   240.0    360.0
Niacin                            8.0     8.0    12.0    12.0     14.0
Phosphorus, P                   500.0   500.0  1250.0  1250.0   1250.0
Potassium, K                   3800.0  3800.0  4500.0  4500.0   4700.0
Riboflavin                        0.6     0.6     0.9     0.9      1.0
Thiamin                           0.6     0.6     0.9     0.9      1.0
Vitamin A, RAE                  400.0   400.0   600.0   600.0    700.0
Vitamin B-12                      1.2     1.2     1.8     1.8      2.4
Vitamin B-6                       0.6     0.6     1.0     1.0      1.2
Vitamin C, total ascorbic acid   25.0    25.0    45.0    45.0     65.0
Vitamin E (alpha-tocopherol)      7.0     7.0    11.0    11.0     15.0
Vitamin K (phylloquinone)        55.0    55.0    60.0    60.0     75.0
Zinc, Zn                          5.0     5.0     8.0     8.0      9.0

                                M 14-18  F 19-30  M 19-30  F 31-50  M 31-50  \
Nutrition
Energy                           2200.0   2000.0   2400.0   1800.0   2200.0
Protein                            52.0     46.0     56.0     46.0     56.0
Fiber, total dietary               30.8     28.0     33.6     25.2     30.8
Folate, DFE                       400.0    400.0    400.0    400.0    400.0
Calcium, Ca                      1300.0   1000.0   1000.0   1000.0   1000.0
Carbohydrate, by difference       130.0    130.0    130.0    130.0    130.0
Iron, Fe                           11.0     18.0      8.0     18.0      8.0
Magnesium, Mg                     410.0    310.0    400.0    320.0    420.0
Niacin                             16.0     14.0     16.0     14.0     16.0
Phosphorus, P                    1250.0    700.0    700.0    700.0    700.0
Potassium, K                     4700.0   4700.0   4700.0   4700.0   4700.0
```

```
Riboflavin                            1.3      1.1      1.3      1.1      1.3
Thiamin                               1.2      1.1      1.2      1.1      1.2
Vitamin A, RAE                      900.0    700.0    900.0    700.0    900.0
Vitamin B-12                          2.4      2.4      2.4      2.4      2.4
Vitamin B-6                           1.3      1.3      1.3      1.3      1.3
Vitamin C, total ascorbic acid       75.0     75.0     90.0     75.0     90.0
Vitamin E (alpha-tocopherol)         15.0     15.0     15.0     15.0     15.0
Vitamin K (phylloquinone)            75.0     90.0    120.0     90.0    120.0
Zinc, Zn                             11.0      8.0     11.0      8.0     11.0

                                 F 51+   M 51+   M 0-3   F 0-3
Nutrition
Energy                          1600.0  2000.0  1000.0  1000.0
Protein                           46.0    56.0    13.0    13.0
Fiber, total dietary              22.4    28.0    14.0    14.0
Folate, DFE                      400.0   400.0   150.0   150.0
Calcium, Ca                     1200.0  1000.0   700.0   700.0
Carbohydrate, by difference      130.0   130.0   130.0   130.0
Iron, Fe                           8.0     8.0     7.0     7.0
Magnesium, Mg                    320.0   420.0    80.0    80.0
Niacin                            14.0    16.0     6.0     6.0
Phosphorus, P                    700.0   700.0   460.0   460.0
Potassium, K                    4700.0  4700.0  3000.0  3000.0
Riboflavin                         1.1     1.3     0.5     0.5
Thiamin                            1.1     1.2     0.5     0.5
Vitamin A, RAE                   700.0   900.0   300.0   300.0
Vitamin B-12                       2.4     2.4     0.9     0.9
Vitamin B-6                        1.5     1.7     0.5     0.5
Vitamin C, total ascorbic acid    75.0    90.0    15.0    15.0
Vitamin E (alpha-tocopherol)      15.0    15.0     6.0     6.0
Vitamin K (phylloquinone)         90.0   120.0    30.0    30.0
Zinc, Zn                           8.0    11.0     3.0     3.0
```

[53]:
```
dri0,avghh0=dri_mins.align(avghh,axis=1)
hh_dri = dri0.replace('',0)@avghh0
l = hh_dri.index.tolist()
```

[54]:
```
l
```

[54]:
```
['Energy',
 'Protein',
 'Fiber, total dietary',
 'Folate, DFE',
 'Calcium, Ca',
 'Carbohydrate, by difference',
 'Iron, Fe',
 'Magnesium, Mg',
```

```
 'Niacin',
 'Phosphorus, P',
 'Potassium, K',
 'Riboflavin',
 'Thiamin',
 'Vitamin A, RAE',
 'Vitamin B-12',
 'Vitamin B-6',
 'Vitamin C, total ascorbic acid',
 'Vitamin E (alpha-tocopherol)',
 'Vitamin K (phylloquinone)',
 'Zinc, Zn']
```

[55]:
```
hh_dri
```

[55]:
```
Nutrition
Energy                               12718.097643
Protein                                279.378367
Fiber, total dietary                   178.053367
Folate, DFE                           2357.565236
Calcium, Ca                           7799.473906
Carbohydrate, by difference            956.807660
Iron, Fe                                78.940025
Magnesium, Mg                         2003.023990
Niacin                                  90.158670
Phosphorus, P                         5828.956229
Potassium, K                         31786.889731
Riboflavin                               7.064478
Thiamin                                  6.867698
Vitamin A, RAE                        4704.713805
Vitamin B-12                            14.145391
Vitamin B-6                              7.813215
Vitamin C, total ascorbic acid         419.534933
Vitamin E (alpha-tocopherol)            87.743266
Vitamin K (phylloquinone)              562.292719
Zinc, Zn                                57.128998
dtype: float64
```

[57]:
```
nutritional_df = pd.read_csv('my_nutrients.csv').reset_index(drop=True)
nutritional_df[''] = nutritional_df['Unnamed: 0']
nutritional_df = nutritional_df.drop(columns=['Unnamed: 0']).set_index('')
nutritional_df
```

[57]:
```
                               (Cocoyam, Spinach, etc)  Agricultural eggs  \

Alanine                                            0.0              0.714
Alcohol, ethyl                                     0.0              0.000
```

```
Amino acids                                0.0              0.000
Arginine                                   0.0              0.691
Ash                                        0.0              0.650
…                                         …               …
Vitamin K (Menaquinone-4)                  0.0              0.000
Vitamin K (phylloquinone)                  0.0              0.000
Vitamins and Other Components              0.0              0.000
Water                                      0.0             86.300
Zinc, Zn                                   0.0              0.000

                               Animal fat  Apples  Avocado pear  \

Alanine                               0.0     0.0          0.00
Alcohol, ethyl                        0.0     0.0          0.00
Amino acids                           0.0     0.0          0.00
Arginine                              0.0     0.0          0.00
Ash                                   0.0     0.0          0.00
…                                   …     …           …
Vitamin K (Menaquinone-4)             0.0     0.0          0.00
Vitamin K (phylloquinone)             0.0     0.0         21.00
Vitamins and Other Components         0.0     0.0          0.00
Water                                 0.0     0.0         73.23
Zinc, Zn                              0.0     0.0          0.64

                               Baby milk powder  Bananas   Beef  \

Alanine                                    0.00     0.00   0.00
Alcohol, ethyl                             0.00     0.00   0.00
Amino acids                                0.00     0.00   0.00
Arginine                                   0.00     0.00   0.00
Ash                                        0.00     0.00   0.00
…                                         …      …    …
Vitamin K (Menaquinone-4)                  0.00     0.00   0.00
Vitamin K (phylloquinone)                  5.80     0.50   1.70
Vitamins and Other Components              0.00     0.00   0.00
Water                                     87.26    74.91  62.58
Zinc, Zn                                   0.66     0.15   4.23

                               Beer (local and imported)  Biscuits  …  Tea  \
                                                                    …
Alanine                                             0.00       0.0  …  0.0
Alcohol, ethyl                                      3.90       0.0  …  0.0
Amino acids                                         0.00       0.0  …  0.0
Arginine                                            0.00       0.0  …  0.0
Ash                                                 0.00       0.0  …  0.0
…                                                  …        …  …  …
Vitamin K (Menaquinone-4)                           0.00       0.0  …  0.0
```

Vitamin K (phylloquinone)	0.00	0.0	…	0.0
Vitamins and Other Components	0.00	0.0	…	0.0
Water	91.96	0.0	…	0.0
Zinc, Zn	0.01	0.0	…	0.0

	Tomato puree(canned)	Tomatoes \
Alanine	0.052	0.00
Alcohol, ethyl	0.000	0.00
Amino acids	0.000	0.00
Arginine	0.032	0.00
Ash	1.280	0.00
…	…	…
Vitamin K (Menaquinone-4)	0.000	0.00
Vitamin K (phylloquinone)	3.400	7.90
Vitamins and Other Components	0.000	0.00
Water	87.880	94.52
Zinc, Zn	0.360	0.17

	Unground Ogbono	Watermelon	Wheat flour \
Alanine	0.00	0.0	0.0
Alcohol, ethyl	0.00	0.0	0.0
Amino acids	0.00	0.0	0.0
Arginine	0.00	0.0	0.0
Ash	0.00	0.0	0.0
…	…	…	…
Vitamin K (Menaquinone-4)	0.00	0.0	0.0
Vitamin K (phylloquinone)	4.20	0.0	0.0
Vitamins and Other Components	0.00	0.0	0.0
Water	83.46	0.0	0.0
Zinc, Zn	0.09	0.0	0.0

	White beans	Wild game meat	Yam flour \
Alanine	0.00	1.273	0.0
Alcohol, ethyl	0.00	0.000	0.0
Amino acids	0.00	0.000	0.0
Arginine	0.00	1.493	0.0
Ash	0.00	0.970	0.0
…	…	…	…
Vitamin K (Menaquinone-4)	0.00	0.000	0.0
Vitamin K (phylloquinone)	0.00	0.000	0.0
Vitamins and Other Components	0.00	0.000	0.0
Water	0.00	72.540	0.0
Zinc, Zn	3.54	0.000	0.0

```
                               Yam-roots

Alanine                            0.063
Alcohol, ethyl                     0.000
Amino acids                        0.000
Arginine                           0.127
Ash                                0.820
…                                    …
Vitamin K (Menaquinone-4)          0.000
Vitamin K (phylloquinone)          2.300
Vitamins and Other Components      0.000
Water                             69.600
Zinc, Zn                           0.240

[173 rows x 132 columns]
```

[58]:

```
n = nutritional_df[nutritional_df.index.isin(l)]
fct = n.T
fct
```

[58]:

```
                         Calcium, Ca  Carbohydrate, by difference  Energy  \
(Cocoyam, Spinach, etc)         94.0                         3.53    24.0
Agricultural eggs                0.0                         2.36   231.0
Animal fat                       0.0                         0.00   867.0
Apples                           8.0                        14.05    54.0
Avocado pear                    12.0                         8.53   160.0
…                                …                           …       …
Wheat flour                      0.0                        70.70   345.0
White beans                    236.0                         0.00     0.0
Wild game meat                  12.0                         0.00   510.0
Yam flour                       20.0                        84.00   267.0
Yam-roots                       17.0                        27.88   494.0

                         Fiber, total dietary  Folate, DFE  Iron, Fe  \
(Cocoyam, Spinach, etc)                   1.2          0.0      2.12
Agricultural eggs                         0.0          0.0      0.00
Animal fat                                0.0          0.0      0.00
Apples                                    2.1          0.0      0.15
Avocado pear                              6.7         81.0      0.55
…                                         …            …        …
Wheat flour                               2.6          0.0      0.00
White beans                               4.3          0.0      4.93
Wild game meat                            0.0          0.0      0.00
Yam flour                                 1.0          0.0      0.72
Yam-roots                                 4.1         23.0      0.54

                         Magnesium, Mg  Niacin  Phosphorus, P  Potassium, K  \
```

(Cocoyam, Spinach, etc)	0.0	0.000	0.0	0.0
Agricultural eggs	0.0	0.000	0.0	0.0
Animal fat	0.0	0.000	0.0	0.0
Apples	0.0	0.000	0.0	107.0
Avocado pear	29.0	1.738	52.0	485.0
…	…	…	…	…
Wheat flour	0.0	0.000	0.0	0.0
White beans	182.0	0.000	533.0	1540.0
Wild game meat	0.0	4.000	120.0	0.0
Yam flour	0.0	0.000	0.0	0.0
Yam-roots	21.0	0.552	55.0	816.0

	Protein	Riboflavin	Thiamin	Vitamin A, RAE \
(Cocoyam, Spinach, etc)	2.35	0.000	0.000	0.0
Agricultural eggs	10.70	0.391	0.000	0.0
Animal fat	0.00	0.000	0.000	0.0
Apples	0.41	0.000	0.000	0.0
Avocado pear	2.00	0.130	0.067	7.0
…	…	…	…	…
Wheat flour	11.80	0.000	0.000	0.0
White beans	24.50	0.000	0.000	0.0
Wild game meat	21.51	0.110	0.390	0.0
Yam flour	2.00	0.000	0.000	0.0
Yam-roots	1.53	0.032	0.112	7.0

	Vitamin B-12	Vitamin B-6 \
(Cocoyam, Spinach, etc)	0.0	0.000
Agricultural eggs	0.0	0.000
Animal fat	0.0	0.000
Apples	0.0	0.000
Avocado pear	0.0	0.257
…	…	…
Wheat flour	0.0	0.000
White beans	0.0	0.000
Wild game meat	0.0	0.000
Yam flour	0.0	0.000
Yam-roots	0.0	0.293

	Vitamin C, total ascorbic acid \
(Cocoyam, Spinach, etc)	21.2
Agricultural eggs	0.0
Animal fat	0.0
Apples	2.0
Avocado pear	10.0
…	…
Wheat flour	0.0
White beans	0.0

```
Wild game meat                                 0.0
Yam flour                                      1.2
Yam-roots                                     17.1

                         Vitamin E (alpha-tocopherol)  \
(Cocoyam, Spinach, etc)                          0.00
Agricultural eggs                                0.00
Animal fat                                       0.00
Apples                                           0.00
Avocado pear                                     2.07
…                                                 …
Wheat flour                                      0.00
White beans                                      0.00
Wild game meat                                   0.00
Yam flour                                        0.00
Yam-roots                                        0.35

                         Vitamin K (phylloquinone)  Zinc, Zn
(Cocoyam, Spinach, etc)                        0.0      0.00
Agricultural eggs                              0.0      0.00
Animal fat                                     0.0      0.00
Apples                                         0.0      0.00
Avocado pear                                  21.0      0.64
…                                               …        …
Wheat flour                                    0.0      0.00
White beans                                    0.0      3.54
Wild game meat                                 0.0      0.00
Yam flour                                      0.0      0.00
Yam-roots                                      2.3      0.24

[132 rows x 20 columns]
```

4 Nutrient System/Adequacy

[59]:

```python
import warnings

def nutrient_demand(x,p):
    with warnings.catch_warnings():
        warnings.simplefilter("ignore")
        c = r.demands(x,p)

    fct0,c0 = fct.align(c,axis=0,join='inner')
    N = fct0.T@c0

    N = N.loc[~N.index.duplicated()]
```

```
    return N

def my_prices(p0,p=prices,i='Bread'):
    p = p.copy()
    p.loc[i] = p0
    return p

def nutrient_adequacy_ratio(x,p):
    return nutrient_demand(x,p)/hh_dri
```

[60]:
```
import numpy as np
import matplotlib.pyplot as plt

def graph_bud_log_nut(reference_x, UseNutrients=1):
    reference_x = r.get_predicted_expenditures().mean('j').sum('i').sel(t=t,m=m)

    X = np.linspace(reference_x/5,reference_x*5,50)

    df = pd.concat({myx:np.log(nutrient_demand(myx,prices))[UseNutrients] for␣
 ↪myx in X},axis=1).T
    ax = df.plot()

    ax.set_xlabel('budget')
    ax.set_ylabel('log nutrient')
    plt.legend(bbox_to_anchor=(1.02, 1), loc='upper left', borderaxespad=0)

def graph_log_p_log_nut(reference_x, USE_GOOD, UseNutrients=1):
    ref_price = r.prices.sel(i=USE_GOOD,t=t,m=m,drop=True)

    P = np.linspace(1,5,20).tolist()

    ndf = pd.DataFrame({p0:np.
 ↪log(nutrient_demand(reference_x,my_prices(p0,i=USE_GOOD)))[UseNutrients] for␣
 ↪p0 in P}).T

    ax = ndf.plot()

    ax.set_xlabel('log price')
    ax.set_ylabel('log nutrient')
    plt.legend(bbox_to_anchor=(1.02, 1), loc='upper left', borderaxespad=0)

def graph_bud_log_nut_adq(reference_x, UseNutrients=1):
    X = np.linspace(reference_x/5,reference_x*5,50)

    ndf = pd.concat({x:np.log(nutrient_adequacy_ratio(x,prices))[UseNutrients]␣
 ↪for x in X},axis=1).T
```

```
    ax = ndf.plot()

    ax.set_xlabel('budget')
    ax.set_ylabel('log nutrient adequacy ratio')
    ax.axhline(0)
    ax.axvline(reference_x)
    plt.legend(bbox_to_anchor=(1.02, 1), loc='upper left', borderaxespad=0)

def graph_p_log_NAR(reference_x, USE_GOOD, UseNutrients=1):
    Pscale = np.linspace(prices[USE_GOOD]/5, prices[USE_GOOD]* 5, 20).tolist()

    log_nar = {s0:np.
  ↪log(nutrient_adequacy_ratio(reference_x,my_prices(s0,prices,i=USE_GOOD)))[UseNutrients]␣
  ↪for s0 in Pscale}

    log_nar = pd.DataFrame(log_nar).T

    ax = log_nar.plot(ylabel='log NAR',xlabel='Price')


    ax.axhline(0)
    ax.axvline(prices[USE_GOOD])
    plt.legend(bbox_to_anchor=(1.02, 1), loc='upper left', borderaxespad=0)
```

4.1 All Households

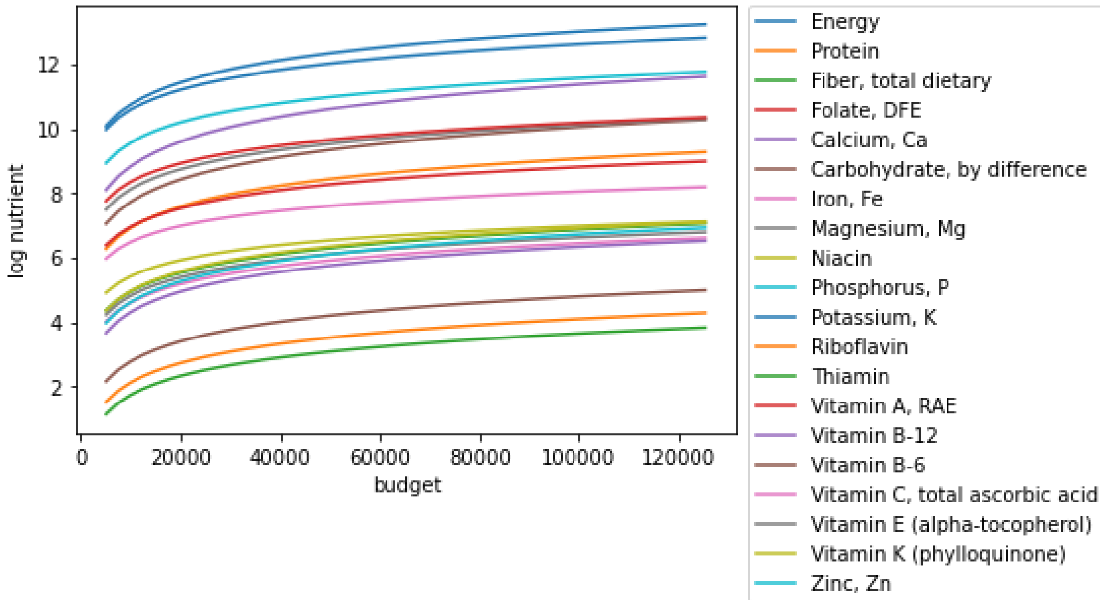
[61]:
```
t = 2018
m = 1

r = cfe.result.from_dataset('Nigeria_small.ds',engine='netcdf4')
reference_x = r.get_predicted_expenditures().mean('j').sum('i').sel(t=t,m=m)
reference_xQ1 = result1.get_predicted_expenditures().mean('j').sum('i').
  ↪sel(t=t,m=m)
reference_xQ2 = result2.get_predicted_expenditures().mean('j').sum('i').
  ↪sel(t=t,m=m)
reference_xQ3 = result3.get_predicted_expenditures().mean('j').sum('i').
  ↪sel(t=t,m=m)
reference_xQ4 = result4.get_predicted_expenditures().mean('j').sum('i').
  ↪sel(t=t,m=m)
```

```
/opt/conda/lib/python3.9/site-packages/xarray/core/nputils.py:152:
RuntimeWarning: Degrees of freedom <= 0 for slice.
  result = getattr(npmodule, name)(values, axis=axis, **kwargs)
```

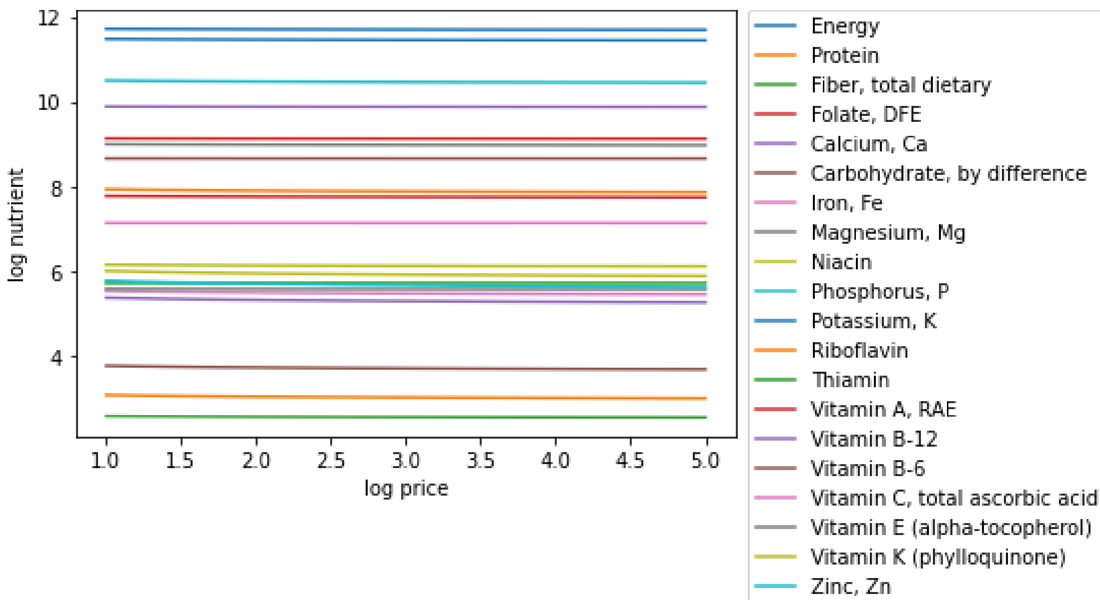
[62]:
```
graph_bud_log_nut(reference_x)
```

```
/opt/conda/lib/python3.9/site-packages/xarray/core/nputils.py:152:
```

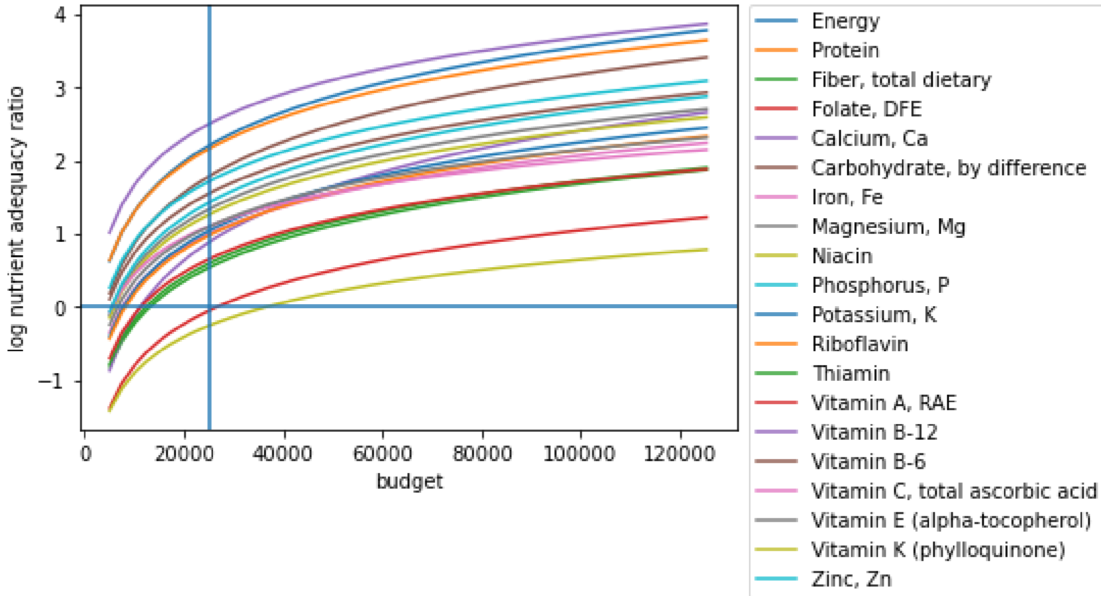
```
RuntimeWarning: Degrees of freedom <= 0 for slice.
  result = getattr(npmodule, name)(values, axis=axis, **kwargs)
```



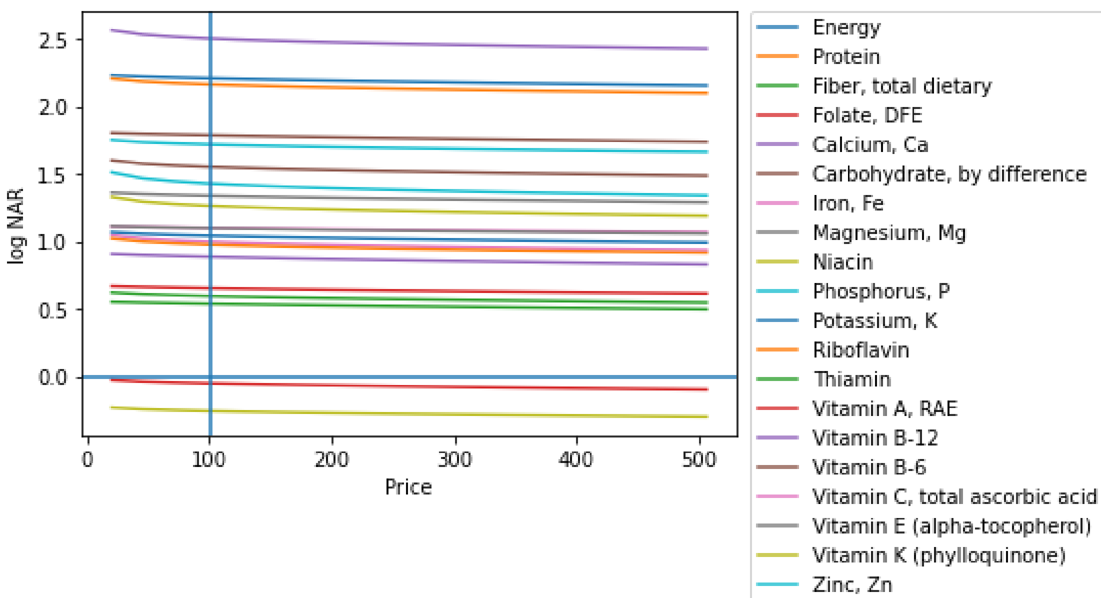
[63]:
```
graph_log_p_log_nut(reference_x, 'Beef')
```



[64]:
```
graph_bud_log_nut_adq(reference_x)
```



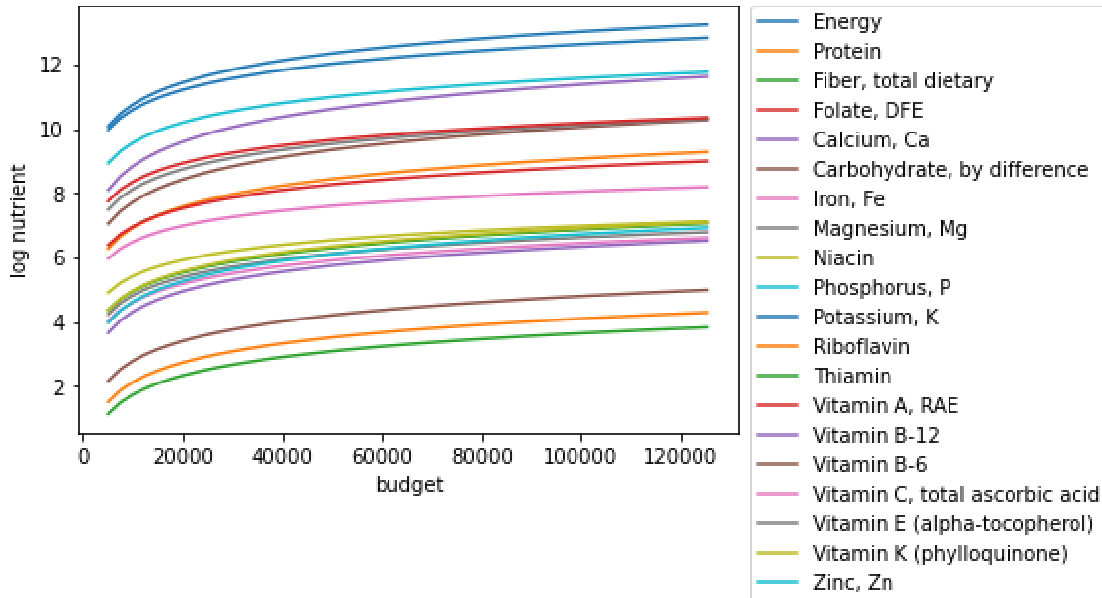
[65]:
```python
graph_p_log_NAR(reference_x, 'Beef')
```



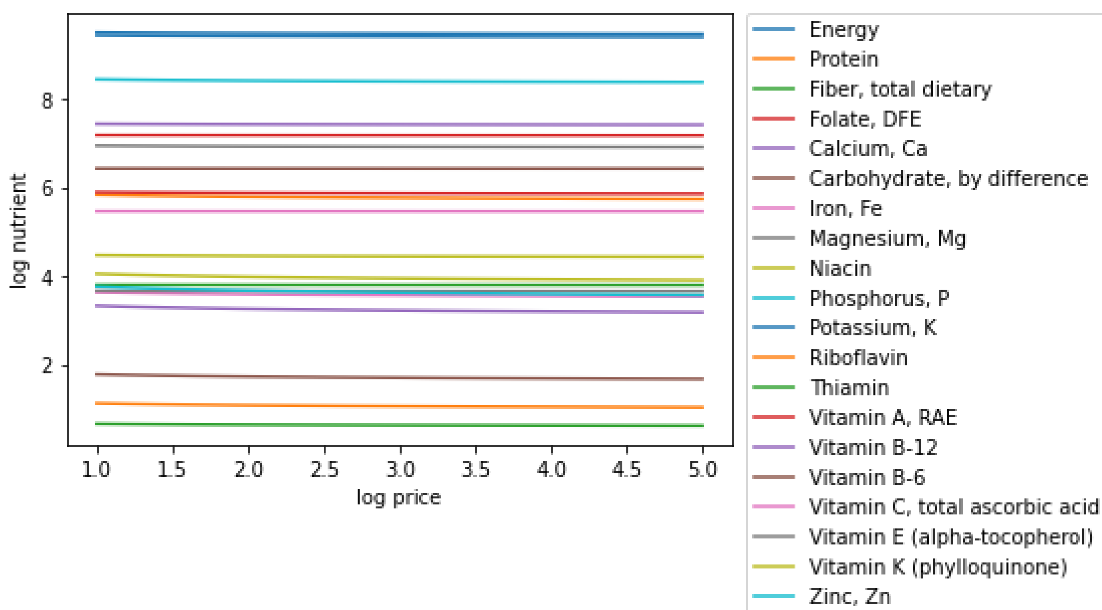
5 Quartile 1

[66]:
```
graph_bud_log_nut(reference_xQ1)
```

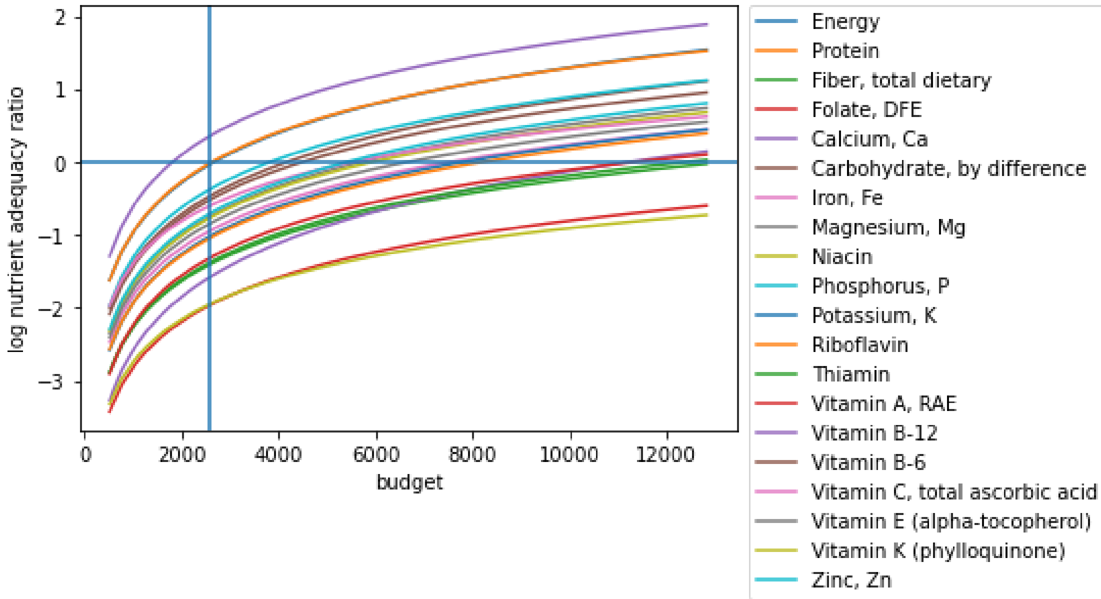
```
/opt/conda/lib/python3.9/site-packages/xarray/core/nputils.py:152:
RuntimeWarning: Degrees of freedom <= 0 for slice.
  result = getattr(npmodule, name)(values, axis=axis, **kwargs)
```



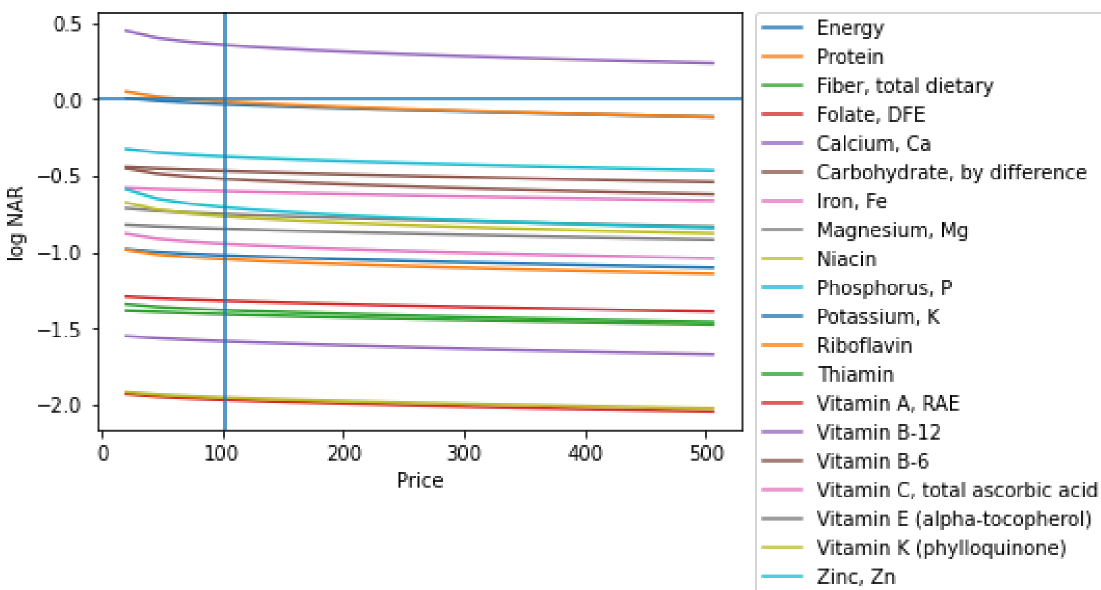
[67]:
```
graph_log_p_log_nut(reference_xQ1, 'Beef')
```



[68]:
```
graph_bud_log_nut_adq(reference_xQ1)
```



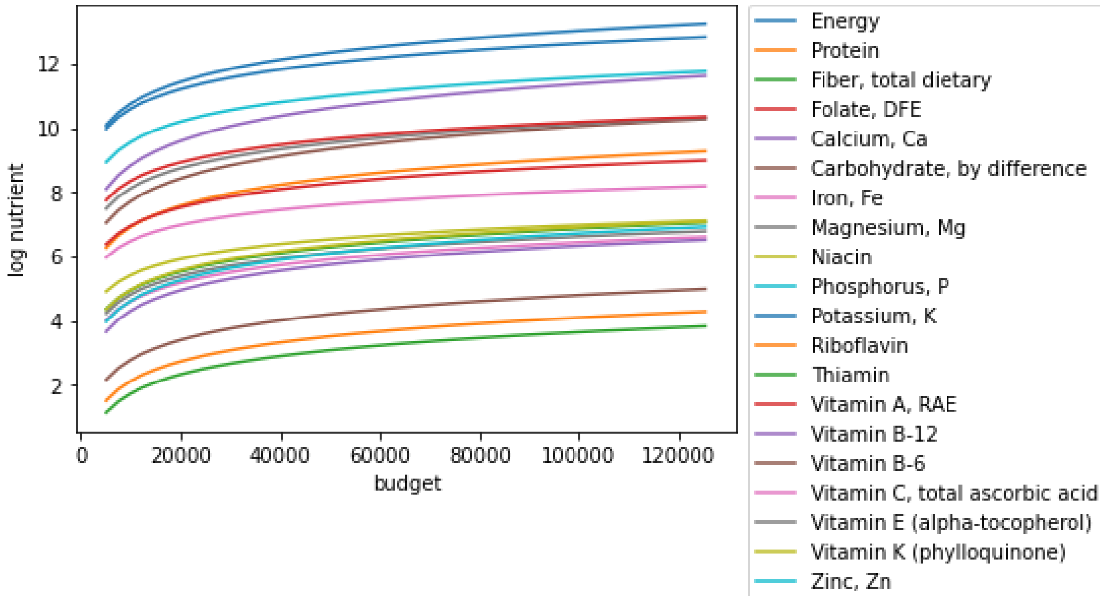
[69]:
```
graph_p_log_NAR(reference_xQ1, 'Beef')
```



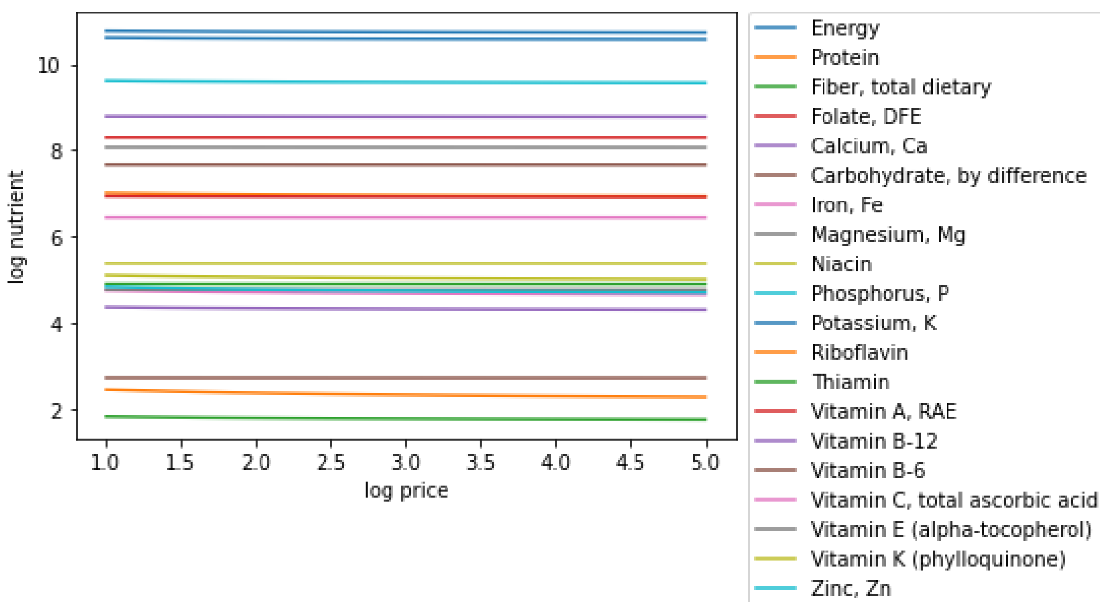
6 Quartile 2

[70]:
```
graph_bud_log_nut(reference_xQ2)
```

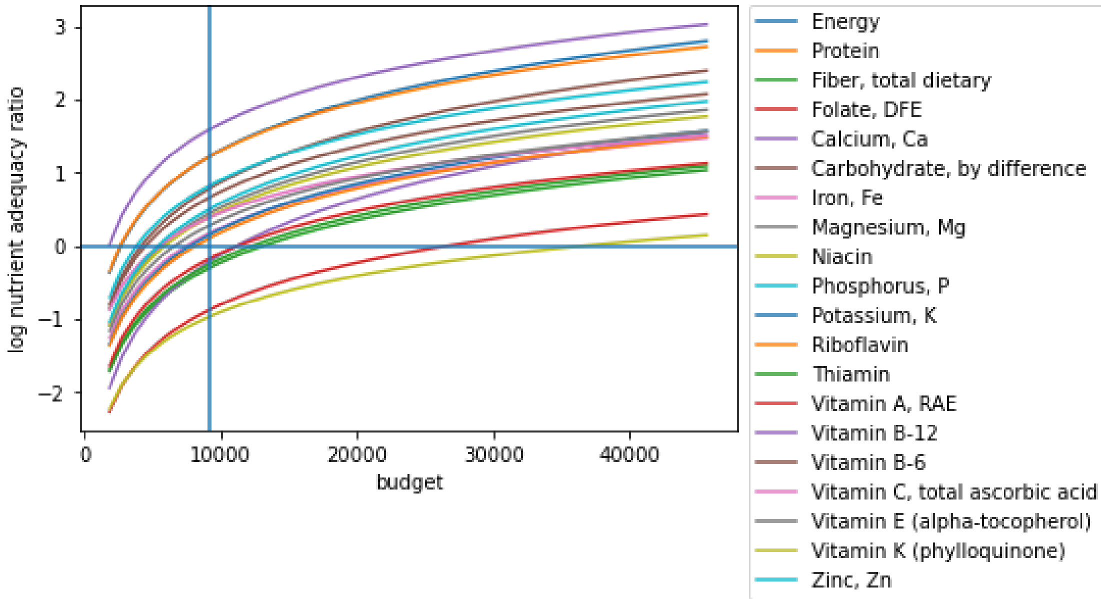
```
/opt/conda/lib/python3.9/site-packages/xarray/core/nputils.py:152:
RuntimeWarning: Degrees of freedom <= 0 for slice.
  result = getattr(npmodule, name)(values, axis=axis, **kwargs)
```



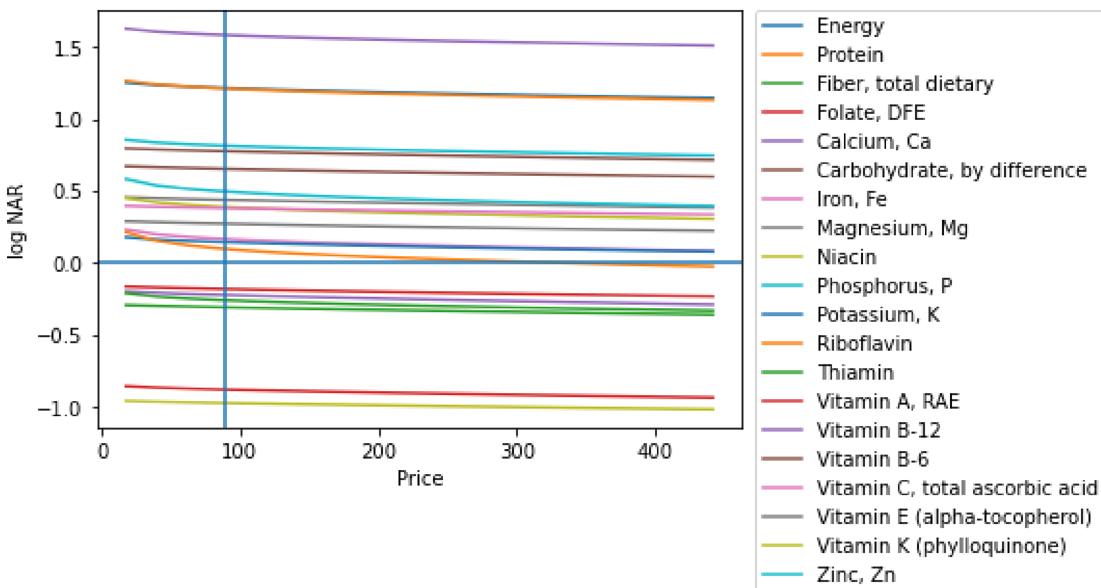
[71]:
```
graph_log_p_log_nut(reference_xQ2, 'Goat')
```



[72]:
```
graph_bud_log_nut_adq(reference_xQ2)
```



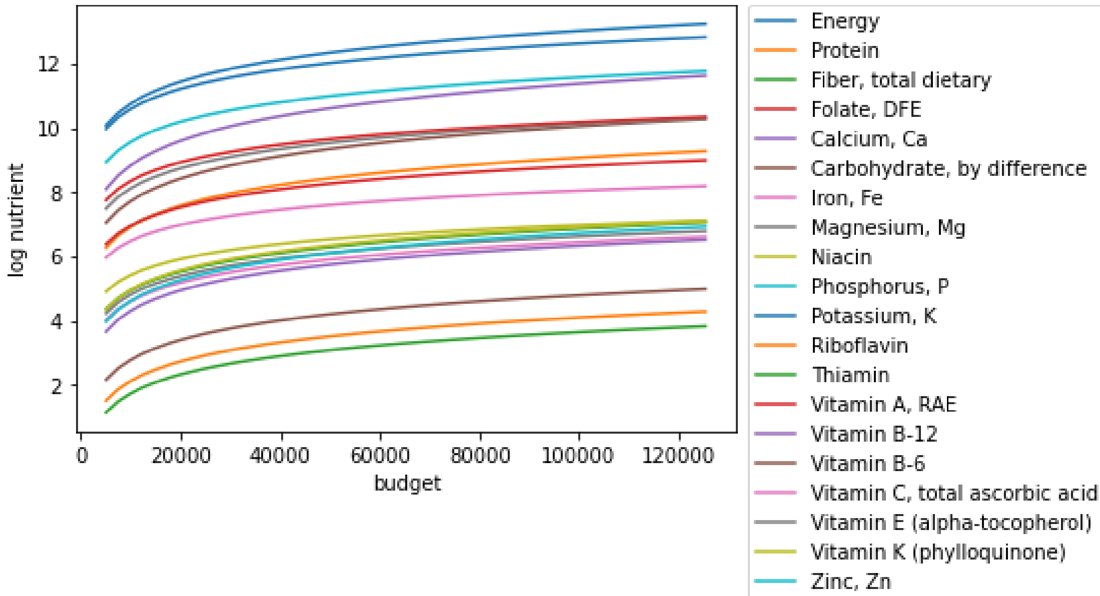
[73]:
```
graph_p_log_NAR(reference_xQ2, 'Goat')
```



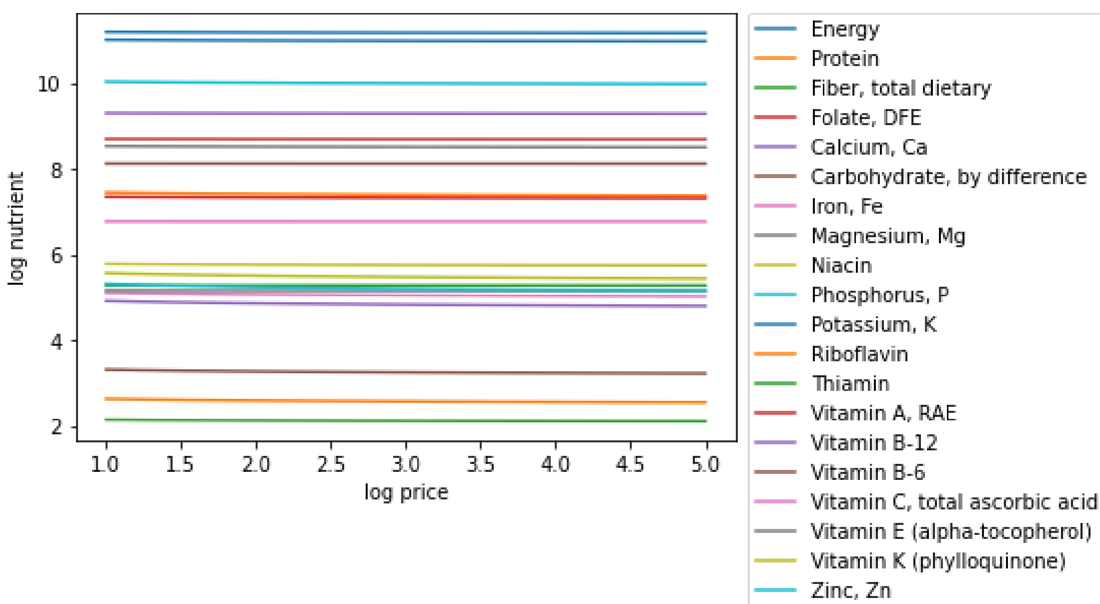
7 Quartile 3

[74]:
```
graph_bud_log_nut(reference_xQ3)
```

```
/opt/conda/lib/python3.9/site-packages/xarray/core/nputils.py:152:
RuntimeWarning: Degrees of freedom <= 0 for slice.
  result = getattr(npmodule, name)(values, axis=axis, **kwargs)
```

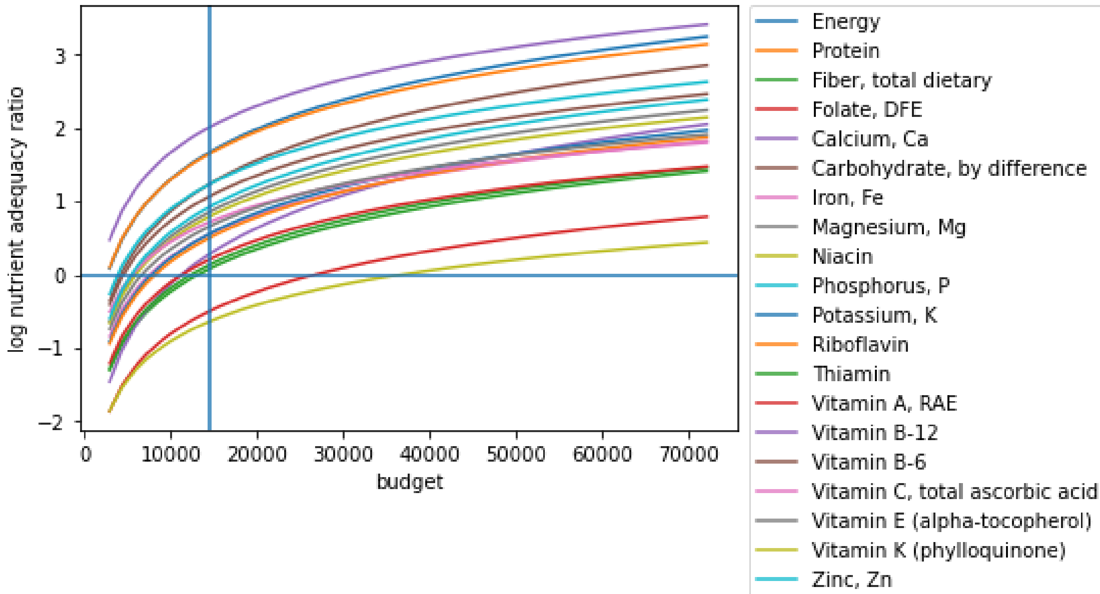


[75]:
```
graph_log_p_log_nut(reference_xQ3, 'Beef')
```



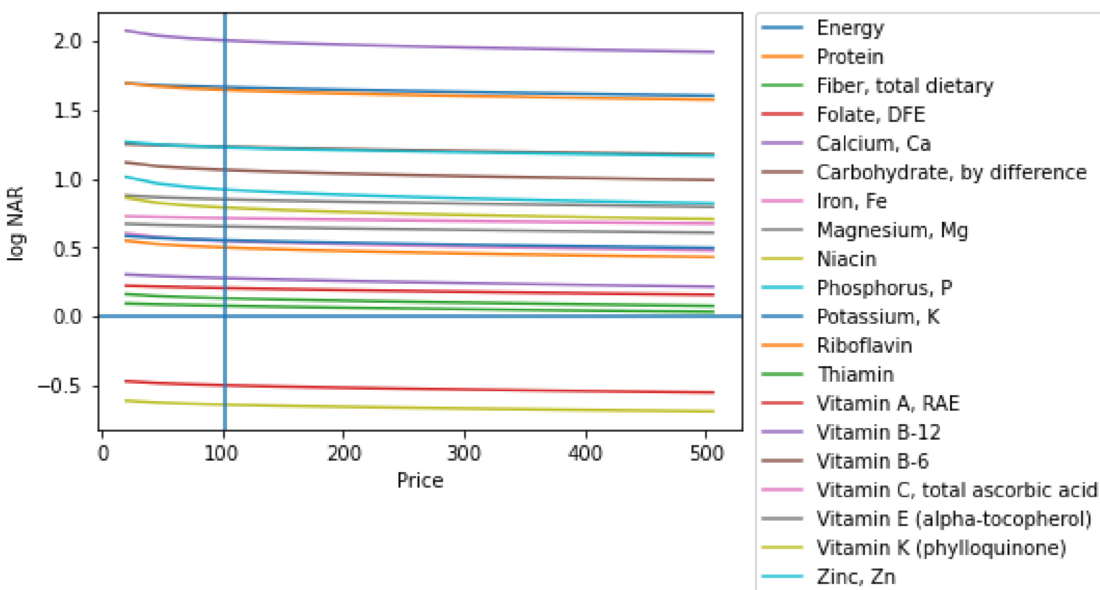
[76]:

```
graph_bud_log_nut_adq(reference_xQ3)
```



[77]:

```
graph_p_log_NAR(reference_xQ3, 'Beef')
```



```
[ ]:
```

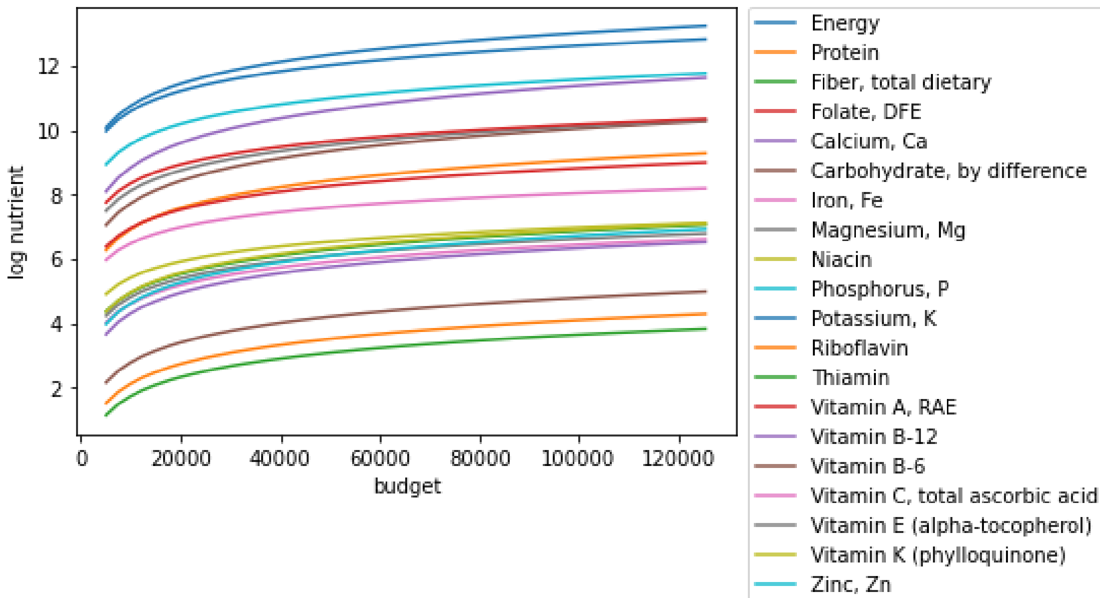
```
[78]: min_budQ1 = result1.get_predicted_expenditures().sum('i').min(['j','t','m'])
      min_budQ1
```

```
[78]: <xarray.DataArray ()>
      array(0.)
```

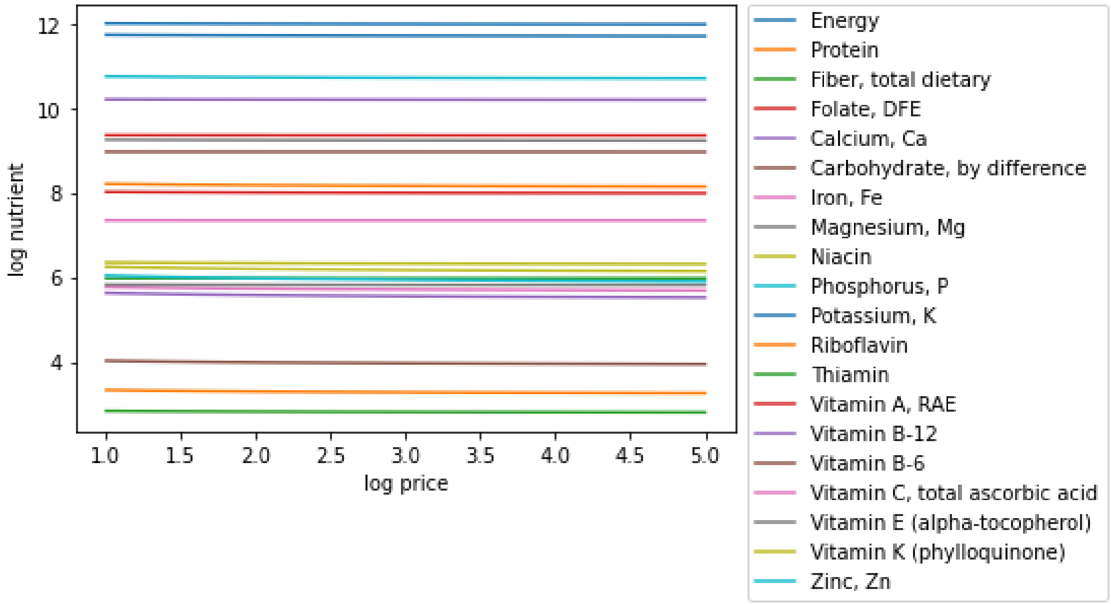
8 Quartile 4

```
[79]: graph_bud_log_nut(reference_xQ4)
```

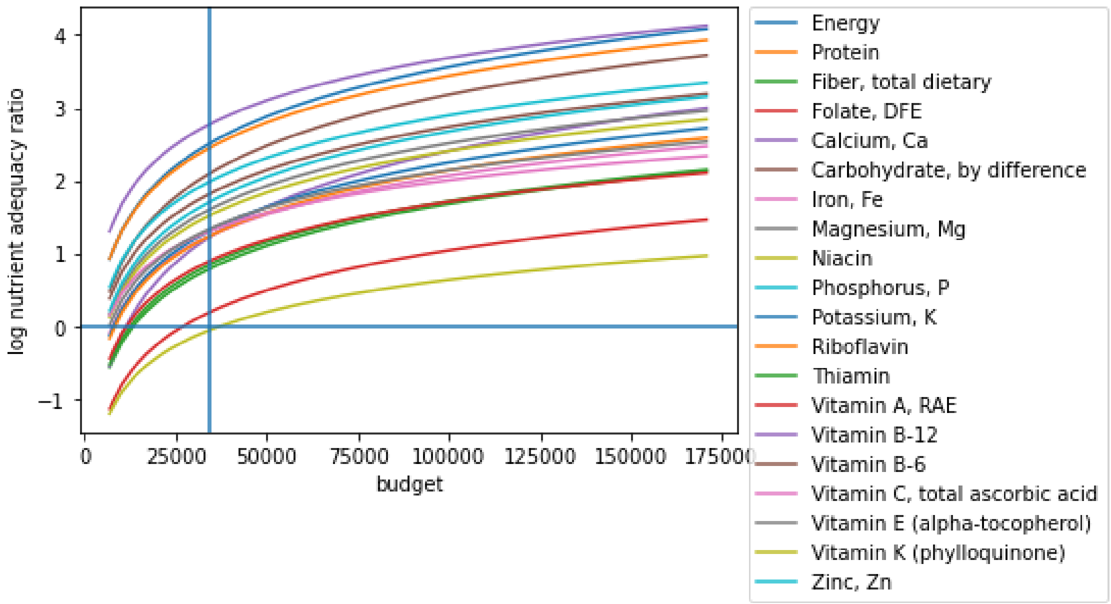
```
/opt/conda/lib/python3.9/site-packages/xarray/core/nputils.py:152:
RuntimeWarning: Degrees of freedom <= 0 for slice.
  result = getattr(npmodule, name)(values, axis=axis, **kwargs)
```



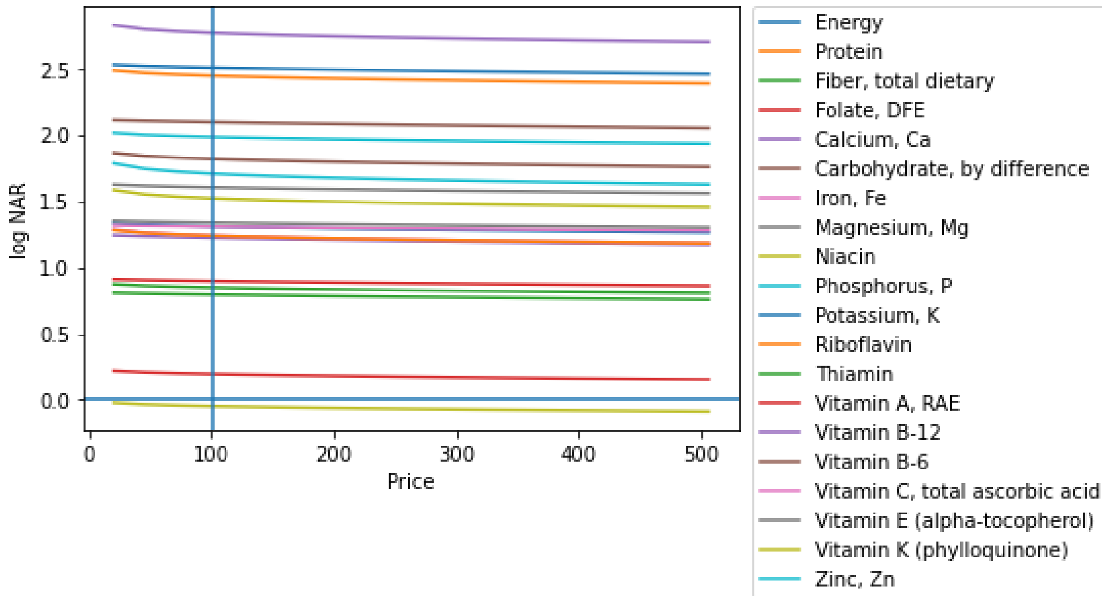

```
[80]: graph_log_p_log_nut(reference_xQ4, 'Beef')
```



[81]:
```
graph_bud_log_nut_adq(reference_xQ4)
```



[82]:
```
graph_p_log_NAR(reference_xQ4, 'Beef')
```



[83]:
```
max_budQ4 = result4.get_predicted_expenditures().sum('i').max(['j','t','m'])
max_budQ4
```

[83]:
```
<xarray.DataArray ()>
array(197373.55583845)
```

[84]:
```
min_budQ4 = result4.get_predicted_expenditures().sum('i').min(['j','t','m'])
min_budQ4
```

[84]:
```
<xarray.DataArray ()>
array(0.)
```

8.0.1 Find the consumption for the quartiles and convert it to daily consumption

Find the consumption for the households in the upper and lower quartiles from the consumption in hectograms data frame. Then, divide the weekly consumption values by 7 in order to get daily consumptions for these households.

[92]:
```
Q3_consumption
```

[92]:
```
                (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
t    j     m
2010 10003 1                        NaN                4.4         NaN     NaN
     10008 1                        NaN                8.9         NaN     NaN
     10011 1                        NaN                NaN         NaN     NaN
     10012 1                        NaN                NaN         NaN     NaN
     10015 1                        NaN                NaN         NaN     NaN
```

```
…                                …                …         …      …
2018 379081 1                      NaN                NaN        NaN     NaN
     379089 1                      NaN                NaN        NaN     NaN
     379123 1                      NaN                NaN        NaN     NaN
     379143 1                      NaN                NaN        NaN     NaN
     379155 1                      NaN                NaN        NaN     NaN

               Avocado pear  Baby milk powder  Bananas  Beef  \
t    j      m
2010 10003  1           NaN               NaN      3.5   3.0
     10008  1           9.0               NaN     15.5   NaN
     10011  1           NaN               NaN      NaN  10.0
     10012  1           NaN               5.0      NaN  10.0
     10015  1           NaN               NaN      NaN   NaN
…                     …               …      …   …
2018 379081 1           NaN               NaN      NaN  10.0
     379089 1           NaN               NaN      NaN  15.0
     379123 1           NaN               NaN      NaN  10.0
     379143 1           NaN               NaN      NaN   NaN
     379155 1           NaN               NaN      NaN  10.0

               Beer (local and imported)  Biscuits  …  Sweet Potatoes  Tea  \
t    j      m                                       …
2010 10003  1                        NaN       NaN  …            14.0  3.0
     10008  1                        NaN       NaN  …             NaN  NaN
     10011  1                        NaN       NaN  …             NaN  NaN
     10012  1                        NaN       NaN  …             NaN  NaN
     10015  1                        NaN       NaN  …             NaN  NaN
…                                  …       …  …             …  …
2018 379081 1                        NaN       NaN  …             NaN  NaN
     379089 1                        NaN       NaN  …             NaN  NaN
     379123 1                        NaN       NaN  …             NaN  NaN
     379143 1                        NaN       NaN  …             NaN  NaN
     379155 1                        NaN       NaN  …             NaN  NaN

               Tomato puree(canned)  Tomatoes  Watermelon  Wheat flour  \
t    j      m
2010 10003  1                   2.1      10.0         NaN          NaN
     10008  1                  35.0      10.0         NaN          NaN
     10011  1                   1.4      10.0         NaN          NaN
     10012  1                   2.8      10.0         NaN          NaN
     10015  1                   0.7       5.0         NaN          NaN
…                             …       …         …           …
2018 379081 1                   NaN       NaN         NaN          NaN
     379089 1                   NaN       NaN         NaN          NaN
     379123 1                   NaN       NaN         NaN          NaN
     379143 1                   NaN       NaN         NaN          NaN
```

```
     379155 1                    NaN       NaN         NaN           NaN

                White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10003  1           6.0             NaN        NaN       46.0
     10008  1          20.0             NaN        NaN       32.0
     10011  1          12.0             NaN        NaN       46.0
     10012  1          18.0             NaN        NaN       69.0
     10015  1           6.0             NaN        NaN       46.0
…                       …             …        …        …
2018 379081 1           NaN             NaN        NaN        NaN
     379089 1           NaN             NaN        NaN        NaN
     379123 1           NaN             NaN        NaN        NaN
     379143 1           NaN             NaN        NaN        NaN
     379155 1           NaN             NaN        NaN        NaN

[4408 rows x 123 columns]
```

[112]:
```
Q1 = Q1.replace(0,np.nan)
Q4 = Q4.replace(0,np.nan)

Q2 = Q2.replace(0,np.nan)
Q3 = Q3.replace(0,np.nan)

Q1_consumption = consumption_in_hect[consumption_in_hect.index.isin(Q1.index)].
 ↪fillna(0)
Q1_consumption_daily = Q1_consumption / 7
Q1_consumption_daily

Q2_consumption = consumption_in_hect[consumption_in_hect.index.isin(Q2.index)].
 ↪fillna(0)
Q2_consumption_daily = Q2_consumption / 7
Q2_consumption_daily.head()

Q3_consumption = consumption_in_hect[consumption_in_hect.index.isin(Q3.index)].
 ↪fillna(0)
Q3_consumption_daily = Q3_consumption / 7
Q3_consumption_daily.head()

Q4_consumption = consumption_in_hect[consumption_in_hect.index.isin(Q4.index)].
 ↪fillna(0)
Q4_consumption_daily = Q4_consumption / 7
Q4_consumption_daily
```

[112]:
```
                (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  Apples  \
t    j      m
2010 10001  1                       0.0           1.271429         0.0     0.0
```

```
     10002  1                    0.0          1.271429         0.0     0.0
     10004  1                    0.0          0.628571         0.0     0.0
     10006  1                    0.0          0.000000         0.0     0.0
     10009  1                    0.0          0.000000         0.0     0.0
…                             …                …         …      …
2018 379144 1                    0.0          0.000000         0.0     0.0
     379146 1                    0.0          0.000000         0.0     0.0
     379148 1                    0.0          0.000000         0.0     0.0
     379151 1                    0.0          0.000000         0.0     0.0
     379154 1                    0.0          0.000000         0.0     0.0

               Avocado pear  Baby milk powder   Bananas      Beef  \
t    j      m
2010 10001  1           0.0          0.000000  1.857143  1.428571
     10002  1           0.0          0.000000  1.857143  2.857143
     10004  1           0.0          0.000000  0.500000  0.428571
     10006  1           0.0          0.000000  1.500000  0.857143
     10009  1           0.0          0.642857  0.500000  0.428571
…                    …                …        …       …
2018 379144 1           0.0          0.000000  0.000000  0.000000
     379146 1           0.0          0.000000  0.000000  0.000000
     379148 1           0.0          0.000000  0.000000  0.714286
     379151 1           0.0          0.000000  0.000000  0.000000
     379154 1           0.0          0.000000  0.000000  1.428571

               Beer (local and imported)  Biscuits  …  Sweet Potatoes  \
t    j      m                                       …
2010 10001  1                   3.214286       0.0  …        2.142857
     10002  1                  12.857143       0.0  …        2.571429
     10004  1                   0.000000       0.0  …        0.000000
     10006  1                   0.000000       0.0  …        0.000000
     10009  1                   0.000000       0.0  …        0.000000
…                                 …        …  …            …
2018 379144 1                   0.000000       0.0  …        0.000000
     379146 1                   0.000000       0.0  …        0.000000
     379148 1                   0.000000       0.0  …        0.000000
     379151 1                   0.000000       0.0  …        0.000000
     379154 1                   0.000000       0.0  …        0.000000

                    Tea  Tomato puree(canned)  Tomatoes  Watermelon  \
t    j      m
2010 10001  1  0.000000                   0.6  1.428571         0.0
     10002  1  0.171429                   0.8  1.428571         0.0
     10004  1  0.214286                   0.2  1.428571         0.0
     10006  1  6.714286                   0.0  5.714286         0.0
     10009  1  0.428571                   0.4  2.857143         0.0
…                 …                    …       …          …
```

```
2018 379144 1   0.000000                     0.0  0.000000         0.0
     379146 1   0.000000                     0.0  0.000000         0.0
     379148 1   0.000000                     0.0  0.000000         0.0
     379151 1   0.000000                     0.0  0.000000         0.0
     379154 1   0.000000                     0.0  0.000000         0.0

                Wheat flour  White beans  Wild game meat  Yam flour  Yam-roots
t    j      m
2010 10001  1      0.000000     4.285714             0.0   0.000000  22.857143
     10002  1      0.000000     2.857143             0.0   0.000000  19.714286
     10004  1      0.000000     0.857143             0.0   0.000000   6.571429
     10006  1      0.000000     0.000000             0.0   0.000000   6.571429
     10009  1      0.000000     2.571429             0.0   0.000000   6.571429
…                     …           …          …         …          …
2018 379144 1      0.000000     0.000000             0.0   1.428571   0.000000
     379146 1      0.000000     0.000000             0.0   0.000000   0.000000
     379148 1      0.000000     0.000000             0.0   0.000000   0.000000
     379151 1      2.857143     0.000000             0.0   0.000000   0.000000
     379154 1      0.000000     0.000000             0.0   0.000000   0.000000

[4502 rows x 123 columns]
```

8.0.2 Consumption Dataframe

The cell below outputs a dataframe describing the different foods that each quartile of households consumed, what % of the households consumed each food, and out of those that consumed the food in each row, how much they consumed on average (in hectograms).

[113]:
```
pd.DataFrame(Q4_consumption_daily.astype(bool).sum(axis=0))
```

[113]:
```
                          0
(Cocoyam, Spinach, etc)  157
Agricultural eggs        235
Animal fat                 1
Apples                     2
Avocado pear              48
…                        …
Wheat flour              177
White beans              633
Wild game meat            39
Yam flour                177
Yam-roots               1335

[123 rows x 1 columns]
```

[114]:
```
summed_foodsQ1 = pd.DataFrame(Q1_consumption_daily.astype(bool).sum(axis=0))
summed_foodsQ1['Q1 (# HH Ate)'] = summed_foodsQ1[0]
```

```
summed_foodsQ1 = summed_foodsQ1.drop(columns=0)
summed_foodsQ1['Q1 (% Ate)'] = summed_foodsQ1['Q1 (# HH Ate)'] /␣
↪len(Q1_consumption_daily) * 100
summed_foodsQ1['Q1 (Average Consumption)'] = Q1_consumption_daily.replace(0, np.
↪nan).mean(axis=0)

summed_foodsQ4 = pd.DataFrame(Q4_consumption_daily.astype(bool).sum(axis=0))
summed_foodsQ4['Q4 (# HH Ate)'] = summed_foodsQ4[0]
summed_foodsQ4 = summed_foodsQ4.drop(columns=0)
summed_foodsQ4['Q4 (% Ate)'] = summed_foodsQ4['Q4 (# HH Ate)'] /␣
↪len(Q4_consumption_daily) * 100
summed_foodsQ4['Q4 (Average Consumption)'] = Q4_consumption_daily.replace(0, np.
↪nan).mean(axis=0)

summed_foodsQ2 = pd.DataFrame(Q2_consumption_daily.astype(bool).sum(axis=0))
summed_foodsQ2['Q2 (# HH Ate)'] = summed_foodsQ2[0]
summed_foodsQ2 = summed_foodsQ2.drop(columns=0)
summed_foodsQ2['Q2 (% Ate)'] = summed_foodsQ2['Q2 (# HH Ate)'] /␣
↪len(Q2_consumption_daily) * 100
summed_foodsQ2['Q2 (Average Consumption)'] = Q2_consumption_daily.replace(0, np.
↪nan).mean(axis=0)

summed_foodsQ3 = pd.DataFrame(Q3_consumption_daily.astype(bool).sum(axis=0))
summed_foodsQ3['Q3 (# HH Ate)'] = summed_foodsQ3[0]
summed_foodsQ3 = summed_foodsQ3.drop(columns=0)
summed_foodsQ3['Q3 (% Ate)'] = summed_foodsQ3['Q3 (# HH Ate)'] /␣
↪len(Q3_consumption_daily) * 100
summed_foodsQ3['Q3 (Average Consumption)'] = Q3_consumption_daily.replace(0, np.
↪nan).mean(axis=0)


all_summed_nutrients = pd.concat([summed_foodsQ1, summed_foodsQ2,␣
↪summed_foodsQ3, summed_foodsQ4], axis=1)
all_summed_nutrients.sort_values(['Q4 (% Ate)'], axis=0, ascending=False)
```

[114]:

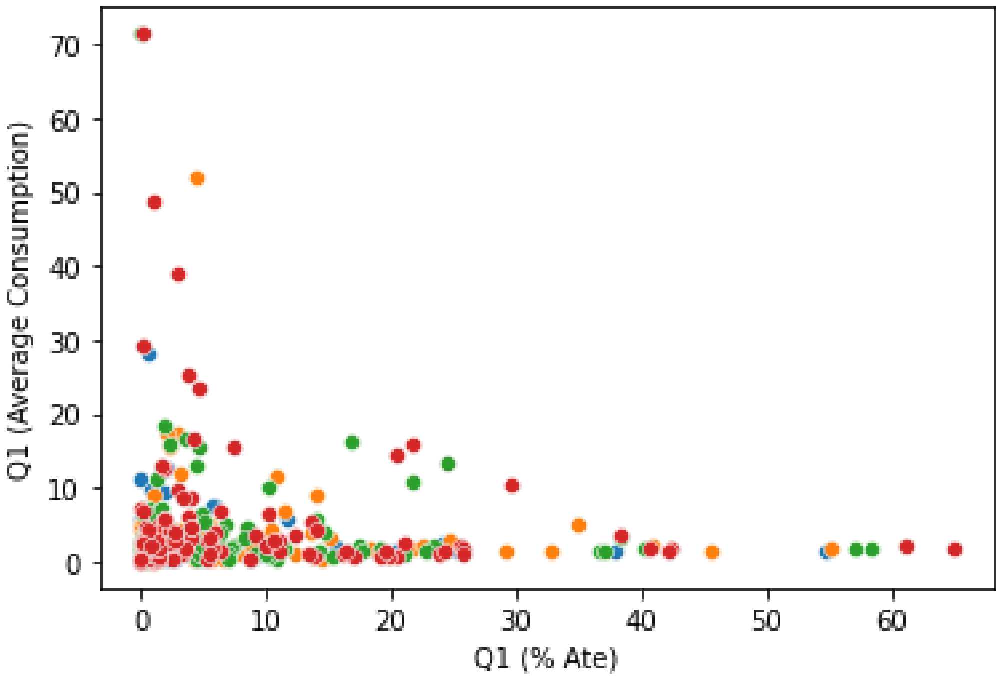
```
                            Q1 (# HH Ate)  Q1 (% Ate)  \
Beef                                  979   25.271038
Palm oil                             2123   54.801239
Bread                                 724   18.688694
Onions                               1466   37.842024
Groundnut oil                         934   24.109448
…                                       …           …
Guava                                   0    0.000000
Maize (shelled/on the cob)              0    0.000000
Maize (shelled/off the cob)             0    0.000000
Maize (on the cob)                      0    0.000000
```

```
Coconut                                0    0.000000

                             Q1 (Average Consumption)  Q2 (# HH Ate)  \
Beef                                         1.300738           1923
Palm oil                                     1.492680           2331
Bread                                        1.589655           1235
Onions                                       1.490585           1729
Groundnut oil                                1.364518           1386
…                                                 …              …
Guava                                             NaN              0
Maize (shelled/on the cob)                        NaN              0
Maize (shelled/off the cob)                       NaN              0
Maize (on the cob)                                NaN              0
Coconut                                           NaN              0

                             Q2 (% Ate)  Q2 (Average Consumption)  \
Beef                          45.579521                  1.559542
Palm oil                      55.250059                  1.816095
Bread                         29.272339                  1.444860
Onions                        40.981275                  2.213572
Groundnut oil                 32.851387                  1.378174
…                                  …                        …
Guava                          0.000000                       NaN
Maize (shelled/on the cob)     0.000000                       NaN
Maize (shelled/off the cob)    0.000000                       NaN
Maize (on the cob)             0.000000                       NaN
Coconut                        0.000000                       NaN

                             Q3 (# HH Ate)  Q3 (% Ate)  \
Beef                                  2578   58.484574
Palm oil                              2515   57.055354
Bread                                 1617   36.683303
Onions                                1777   40.313067
Groundnut oil                         1633   37.046279
…                                      …         …
Guava                                    0    0.000000
Maize (shelled/on the cob)               0    0.000000
Maize (shelled/off the cob)              0    0.000000
Maize (on the cob)                       0    0.000000
Coconut                                  0    0.000000

                             Q3 (Average Consumption)  Q4 (# HH Ate)  \
Beef                                         1.834679           2922
Palm oil                                     1.787097           2758
Bread                                        1.580538           1910
Onions                                       1.781074           1896
Groundnut oil                                1.460912           1835
```

```
…                                                  …            …
Guava                                             NaN          0
Maize (shelled/on the cob)                        NaN          0
Maize (shelled/off the cob)                       NaN          0
Maize (on the cob)                                NaN          0
Coconut                                           NaN          0

                             Q4 (% Ate)  Q4 (Average Consumption)
Beef                          64.904487                  1.924269
Palm oil                      61.261661                  2.231074
Bread                         42.425589                  1.806526
Onions                        42.114616                  1.483361
Groundnut oil                 40.759662                  1.704181
…                                   …                       …
Guava                          0.000000                       NaN
Maize (shelled/on the cob)     0.000000                       NaN
Maize (shelled/off the cob)    0.000000                       NaN
Maize (on the cob)             0.000000                       NaN
Coconut                        0.000000                       NaN

[123 rows x 12 columns]
```

[115]:

```python
import seaborn as sns
scatter1 = sns.scatterplot(data=all_summed_nutrients, x="Q1 (% Ate)", y="Q1␣
↪(Average Consumption)")
scatter2 = sns.scatterplot(data=all_summed_nutrients, x="Q2 (% Ate)", y="Q2␣
↪(Average Consumption)")
scatter3 = sns.scatterplot(data=all_summed_nutrients, x="Q3 (% Ate)", y="Q3␣
↪(Average Consumption)")
scatter4 = sns.scatterplot(data=all_summed_nutrients, x="Q4 (% Ate)", y="Q4␣
↪(Average Consumption)")
```



[116]:
```
all_summed_nutrients.to_csv(r'QuartileConsumption.csv')
```

Sort the individual dataframes in descending order to see which of the nutrient minimums are satisfied most often in the upper quartile versus the lower quartile.

[117]:
```
all_summed_nutrients = all_summed_nutrients.reset_index()
```

[118]:
```
asn_zinc = all_summed_nutrients[all_summed_nutrients['index'].str.
 ↪contains("Goat|Coffee|Beef|Brown beans|Canned beef|White␣
 ↪beans|Seafood|Mutton|Groundnuts|Other domestic poultry")]
asn_fiber = all_summed_nutrients[all_summed_nutrients['index'].str.
 ↪contains("Pepper|Fresh pepper|Dry pepper|Okra-dried|Pawpaw|Groundnuts|Kola␣
 ↪nut|Maize flour|Guinea Corn")]
asn_iron = all_summed_nutrients[all_summed_nutrients['index'].str.
 ↪contains("Pepper|Fresh pepper|Dry pepper|White beans|Brown beans|Other␣
 ↪eggs|Other nuts|Kola nut|Snails|Guinea Corn|Bread")]
asn_ribo = all_summed_nutrients[all_summed_nutrients['index'].str.
 ↪contains("Goat|tinned|domestic poultry|Bread|Fish-Dried|Chicken|Other␣
 ↪eggs|Agricultural|Maize flour|Duck|Mutton")]
```

[119]:
```
asn_fiber
```

[119]:
```
                         index  Q1 (# HH Ate)  Q1 (% Ate)  \
29                  Dry pepper            192    4.956118
36                Fresh pepper            351    9.060403
```

45	Groundnuts	76	1.961797
46	Groundnuts (shelled)	48	1.239029
47	Groundnuts (unshelled)	6	0.154879
49	Guinea Corn/Sorghum	78	2.013423
52	Kola nut	0	0.000000
60	Maize flour	32	0.826020
74	Okra-dried	343	8.853898
96	Pawpaw	0	0.000000
97	Pepper	618	15.952504

	Q1 (Average Consumption)	Q2 (# HH Ate)	Q2 (% Ate)	\
29	0.291979	177	4.195307	
36	0.647843	588	13.936952	
45	1.917810	134	3.176108	
46	0.576190	41	0.971794	
47	0.578571	13	0.308130	
49	12.535256	125	2.962787	
52	NaN	1	0.023702	
60	3.557143	59	1.398436	
74	2.268884	373	8.840958	
96	NaN	0	0.000000	
97	1.373853	771	18.274473	

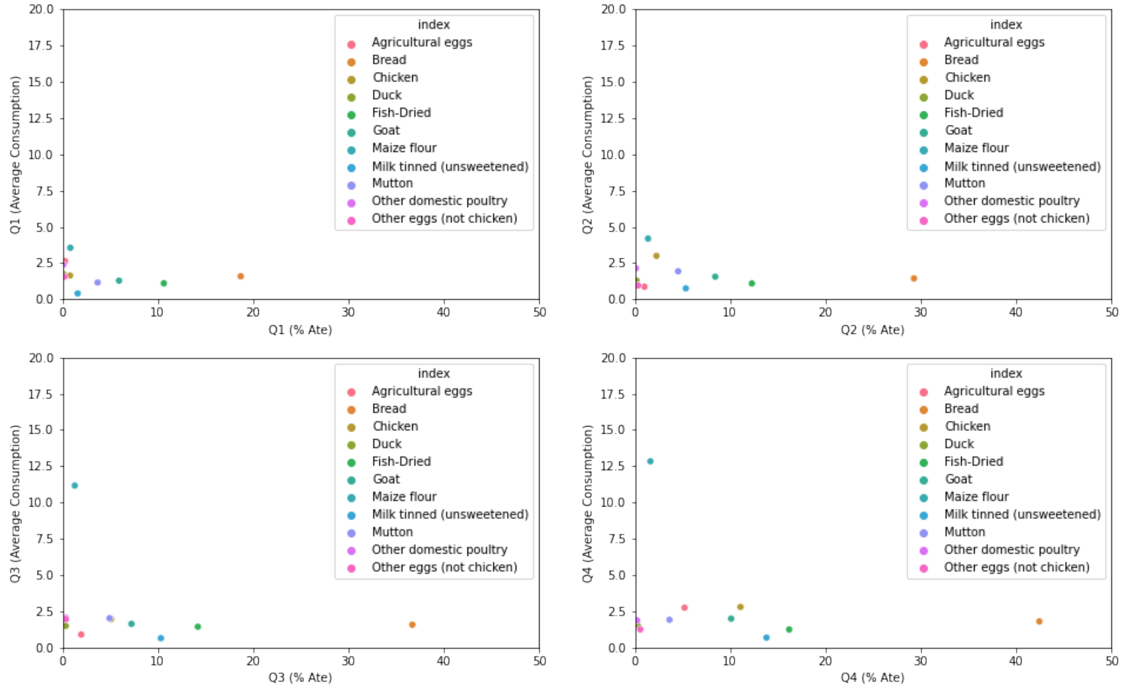
	Q2 (Average Consumption)	Q3 (# HH Ate)	Q3 (% Ate)	\
29	0.388630	199	4.514519	
36	0.829220	673	15.267695	
45	2.689925	119	2.699637	
46	0.330575	63	1.429220	
47	0.474725	22	0.499093	
49	17.368800	203	4.605263	
52	0.071429	0	0.000000	
60	4.187094	57	1.293103	
74	2.174694	308	6.987296	
96	NaN	0	0.000000	
97	1.655299	845	19.169691	

	Q3 (Average Consumption)	Q4 (# HH Ate)	Q4 (% Ate)	\
29	0.312818	244	5.419813	
36	0.897306	766	17.014660	
45	2.123950	131	2.909818	
46	0.341701	114	2.532208	
47	0.737662	22	0.488672	
49	15.401689	169	3.753887	
52	NaN	1	0.022212	
60	11.167168	74	1.643714	
74	1.079443	246	5.464238	
96	NaN	5	0.111062	

```
97                  1.659142           880   19.546868

    Q4 (Average Consumption)
29                  0.331070
36                  0.745864
45                  2.440731
46                  0.383559
47                  0.490260
49                 25.358157
52                  0.285714
60                 12.844054
74                  1.153084
96                  2.200000
97                  1.322935
```

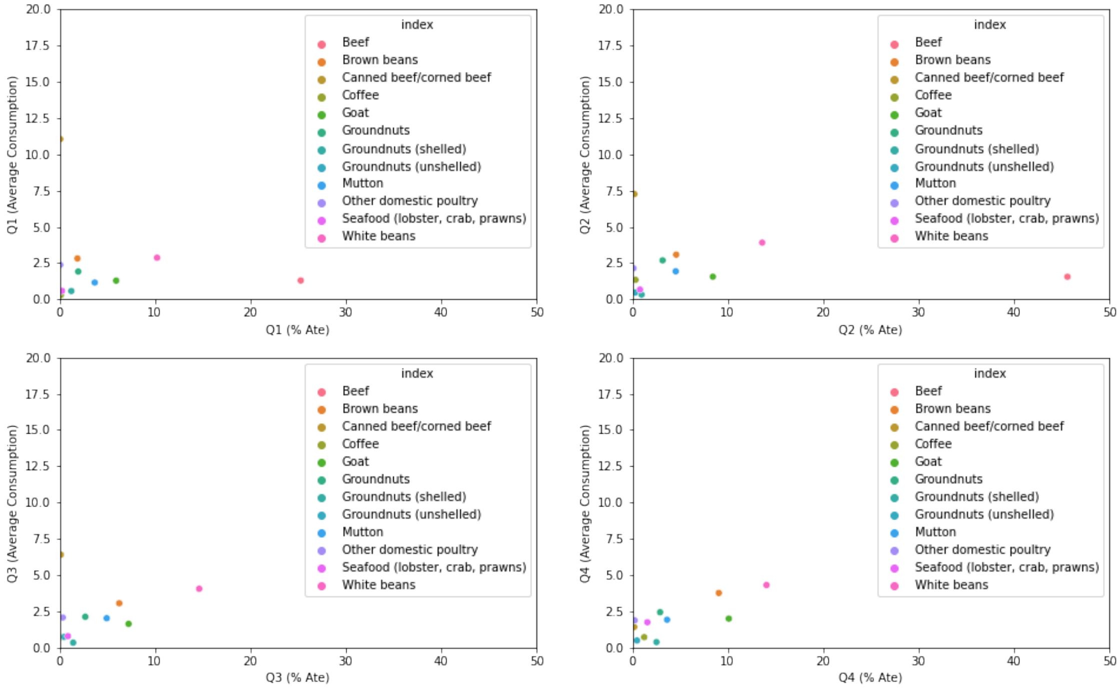
[120]:
```
#Riboflavin Top 11
fig, ([ax1,ax2], [ax3, ax4]) = plt.subplots(2,2, figsize=(16,10))
scatter1 = sns.scatterplot(ax = ax1, data=asn_ribo, x="Q1 (% Ate)", y="Q1␣
↪(Average Consumption)", hue = 'index')
ax1.set_xlim(0,50)
ax1.set_ylim(0,20)
ax2.set_xlim(0,50)
ax2.set_ylim(0,20)
ax3.set_xlim(0,50)
ax3.set_ylim(0,20)
scatter2 = sns.scatterplot(ax = ax2, data=asn_ribo, x="Q2 (% Ate)", y="Q2␣
↪(Average Consumption)", hue = 'index')
scatter3 = sns.scatterplot(ax = ax3, data=asn_ribo, x="Q3 (% Ate)", y="Q3␣
↪(Average Consumption)", hue = 'index')
scatter4 = sns.scatterplot(ax = ax4, data=asn_ribo, x="Q4 (% Ate)", y="Q4␣
↪(Average Consumption)", hue = 'index')
ax4.set_xlim(0,50)
ax4.set_ylim(0,20)
```

[120]:
```
(0.0, 20.0)
```



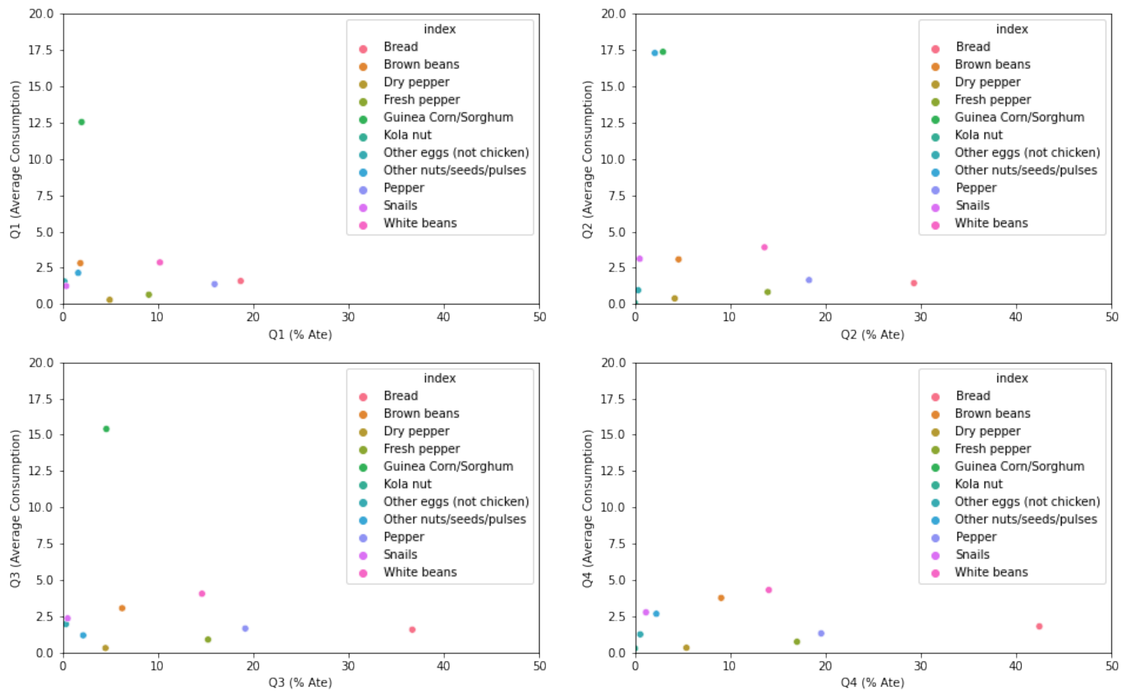
[121]:
```
#Zinc Top 12
fig, ([ax1,ax2], [ax3, ax4]) = plt.subplots(2,2, figsize=(16,10))
scatter1 = sns.scatterplot(ax = ax1, data=asn_zinc, x="Q1 (% Ate)", y="Q1 (Average Consumption)", hue = 'index')
ax1.set_xlim(0,50)
ax1.set_ylim(0,20)
ax2.set_xlim(0,50)
ax2.set_ylim(0,20)
ax3.set_xlim(0,50)
ax3.set_ylim(0,20)
scatter2 = sns.scatterplot(ax = ax2, data=asn_zinc, x="Q2 (% Ate)", y="Q2 (Average Consumption)", hue = 'index')
scatter3 = sns.scatterplot(ax = ax3, data=asn_zinc, x="Q3 (% Ate)", y="Q3 (Average Consumption)", hue = 'index')
scatter4 = sns.scatterplot(ax = ax4, data=asn_zinc, x="Q4 (% Ate)", y="Q4 (Average Consumption)", hue = 'index')
ax4.set_xlim(0,50)
ax4.set_ylim(0,20)
```

[121]: (0.0, 20.0)



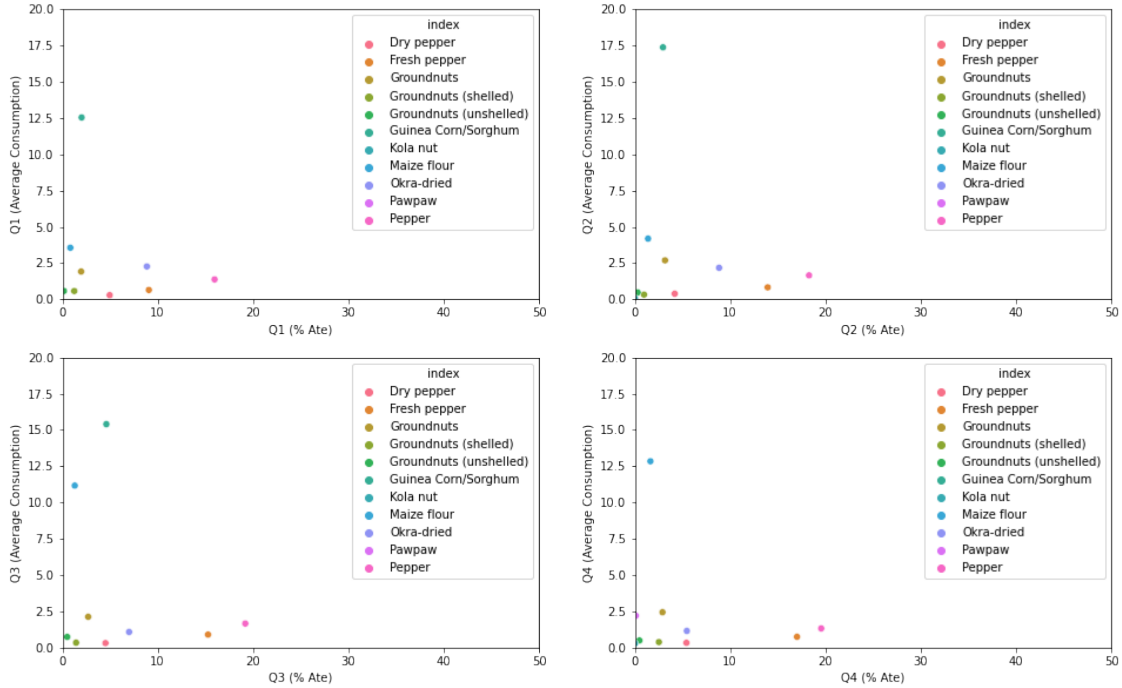
[122]:
```python
#Iron Top 11
fig, ([ax1,ax2], [ax3, ax4]) = plt.subplots(2,2, figsize=(16,10))
scatter1 = sns.scatterplot(ax = ax1, data=asn_iron, x="Q1 (% Ate)", y="Q1␣
↪(Average Consumption)", hue = 'index')
ax1.set_xlim(0,50)
ax1.set_ylim(0,20)
ax2.set_xlim(0,50)
ax2.set_ylim(0,20)
ax3.set_xlim(0,50)
ax3.set_ylim(0,20)
scatter2 = sns.scatterplot(ax = ax2, data=asn_iron, x="Q2 (% Ate)", y="Q2␣
↪(Average Consumption)", hue = 'index')
scatter3 = sns.scatterplot(ax = ax3, data=asn_iron, x="Q3 (% Ate)", y="Q3␣
↪(Average Consumption)", hue = 'index')
scatter4 = sns.scatterplot(ax = ax4, data=asn_iron, x="Q4 (% Ate)", y="Q4␣
↪(Average Consumption)", hue = 'index')
ax4.set_xlim(0,50)
ax4.set_ylim(0,20)
```

[122]: (0.0, 20.0)



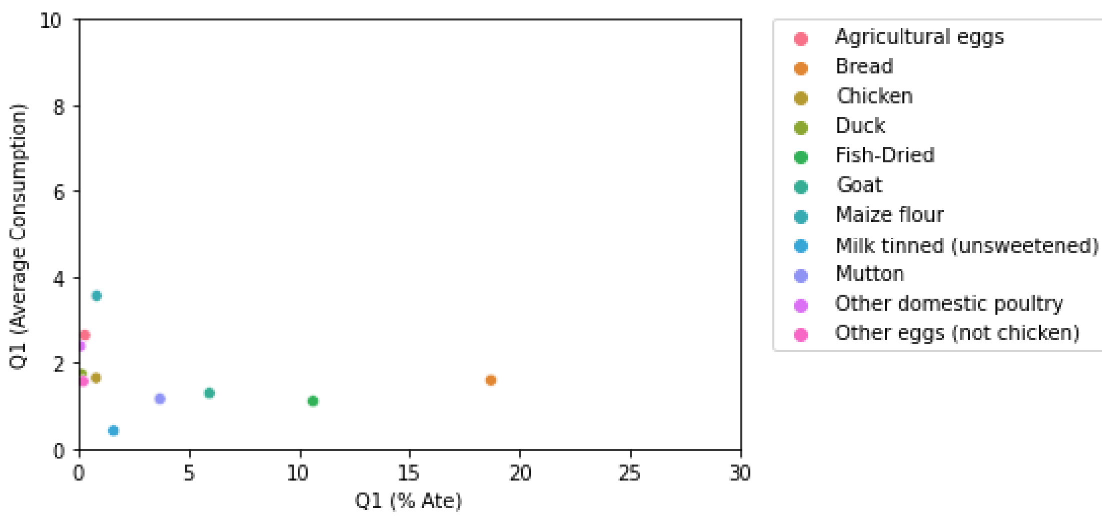
[123]:
```python
#Fiber Top 11
fig, ([ax1,ax2], [ax3, ax4]) = plt.subplots(2,2, figsize=(16,10))
scatter1 = sns.scatterplot(ax = ax1, data=asn_fiber, x="Q1 (% Ate)", y="Q1 (Average Consumption)", hue = 'index')
ax1.set_xlim(0,50)
ax1.set_ylim(0,20)
ax2.set_xlim(0,50)
ax2.set_ylim(0,20)
ax3.set_xlim(0,50)
ax3.set_ylim(0,20)
scatter2 = sns.scatterplot(ax = ax2, data=asn_fiber, x="Q2 (% Ate)", y="Q2 (Average Consumption)", hue = 'index')
scatter3 = sns.scatterplot(ax = ax3, data=asn_fiber, x="Q3 (% Ate)", y="Q3 (Average Consumption)", hue = 'index')
scatter4 = sns.scatterplot(ax = ax4, data=asn_fiber, x="Q4 (% Ate)", y="Q4 (Average Consumption)", hue = 'index')
ax4.set_xlim(0,50)
ax4.set_ylim(0,20)
```

[123]: (0.0, 20.0)



[124]:
```python
scatter1 = sns.scatterplot(data=asn_ribo, x="Q1 (% Ate)", y="Q1 (Average␣
↪Consumption)", hue = 'index')
plt.ylim(0, 10)
plt.xlim(0, 30)
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
scatter1
```

[124]: <AxesSubplot:xlabel='Q1 (% Ate)', ylabel='Q1 (Average Consumption)'>



[125]:
```
scatter2 = sns.scatterplot(data=all_summed_nutrients, x="Q2 (% Ate)", y="Q2␣
↪(Average Consumption)", hue = 'index')
plt.ylim(0, 60)
plt.xlim(0, 60)
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
scatter2
```

[125]:
```
<AxesSubplot:xlabel='Q2 (% Ate)', ylabel='Q2 (Average Consumption)'>
```


Q2 (Average Consumption)
Q2 (% Ate)
0
10
20
30
40
50
60
(Cocoyam, Spinach, etc)
Agricultural eggs
Animal fat
Apples
Avocado pear
Baby milk powder
Bananas
Beef
Beer (local and imported)
Biscuits
Bottled water
Bread
Brown beans
Buns/PofPof/Donuts
Butter/Margarine
Cake
Canned beef/corned beef
Canned fish/seafood
Cashew nut
Cassava flour
Cassava-Roots
Cheese (wara)
Chicken
Chocolate drinks
Coconut
Coconut oil
Cocoyam
Coffee
Condiments,(salt,spices,pepper, etc)
Dry pepper
Duck
Fish-Dried
Fish-Fresh
Fish-Frozen
Fish-Smoked
Fresh milk
Fresh pepper
Fruit juice canned
Garden eggs/egg plant
Gari -Yellow
Gari-White
Gin
Goat
Grinded pepper
Groundnut oil
Groundnuts
Groundnuts (shelled)
Groundnuts (unshelled)
Guava
Guinea Corn/Sorghum
Honey
Jams
Kola nut
Local eggs
Maize
Maize (off the cob/grains)
Maize (on the cob)
Maize (shelled/off the cob)
Maize (shelled/on the cob)
Maize (unshelled/on the cob)
Maize flour
Maize shelled
Malt drinks
Mangoes
Meat pie/Sausage roll
Melon (ground)
Melon (shelled)
Melon (unshelled)
Milk powder
Milk tinned (unsweetened)
Millet
Mutton
Ogbonno grinded
Ogbonno ungrinded
Okra-dried
Okra-fresh
Onions
Orange/tangerine
Other Non-acoholic drinks
Other Oil and Fat
Other alcoholic beverages
Other domestic poultry
Other eggs (not chicken)
Other fish or seafood
Other fruites
Other grains and flour
Other meat (excl. poultry)
Other milk products
Other non-acoholic drinks
Other nuts/seeds/pulses
Other oils and fats
Other roots and tuber
Other sweets Confectionary
Other vegetables (fresh or canned)
Palm oil
Palm wine
Pawpaw
Pepper
Pineapples
Pito
Plantains
Pork
Potatoes
Rice-Imported
Rice-local
Sachet water
Salt
Seafood (lobster, crab, prawns)
Sheabutter
Snails
Soft drinks (Coca cola, spirit etc)
Soya beans
Sugar
Sweet Potatoes
Tea
Tomato puree(canned)
Tomatoes
Watermelon
Wheat flour
White beans
Wild game meat
Yam flour
Yam-roots

[126]:
```
scatter3 = sns.scatterplot(data=all_summed_nutrients, x="Q3 (% Ate)", y="Q3␣
↪(Average Consumption)", hue = 'index')
plt.ylim(0, 80)
plt.xlim(0, 70)
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
scatter3
```

[126]:
```
<AxesSubplot:xlabel='Q3 (% Ate)', ylabel='Q3 (Average Consumption)'>
```

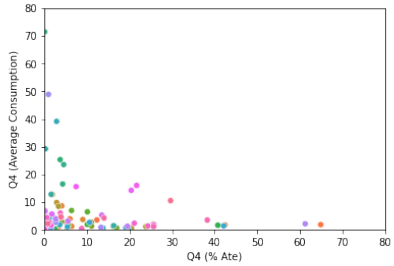
Q3 (Average Consumption)
Q3 (% Ate)
(Cocoyam, Spinach, etc)
Agricultural eggs
Animal fat
Apples
Avocado pear
Baby milk powder
Bananas
Beef
Beer (local and imported)
Biscuits
Bottled water
Bread
Brown beans
Buns/PofPof/Donuts
Butter/Margarine
Cake
Canned beef/corned beef
Canned fish/seafood
Cashew nut
Cassava flour
Cassava-Roots
Cheese (wara)
Chicken
Chocolate drinks
Coconut
Coconut oil
Cocoyam
Coffee
Condiments,(salt,spices,pepper, etc)
Dry pepper
Duck
Fish-Dried
Fish-Fresh
Fish-Frozen
Fish-Smoked
Fresh milk
Fresh pepper
Fruit juice canned
Garden eggs/egg plant
Gari -Yellow
Gari-White
Gin
Goat
Grinded pepper
Groundnut oil
Groundnuts
Groundnuts (shelled)
Groundnuts (unshelled)
Guava
Guinea Corn/Sorghum
Honey
Jams
Kola nut
Local eggs
Maize
Maize (off the cob/grains)
Maize (on the cob)
Maize (shelled/off the cob)
Maize (shelled/on the cob)
Maize (unshelled/on the cob)
Maize flour
Maize shelled
Malt drinks
Mangoes
Meat pie/Sausage roll
Melon (ground)
Melon (shelled)
Melon (unshelled)
Milk powder
Milk tinned (unsweetened)
Millet
Mutton
Ogbonno grinded
Ogbonno ungrinded
Okra-dried
Okra-fresh
Onions
Orange/tangerine
Other Non-acoholic drinks
Other Oil and Fat
Other alcoholic beverages
Other domestic poultry
Other eggs (not chicken)
Other fish or seafood
Other fruites
Other grains and flour
Other meat (excl. poultry)
Other milk products
Other non-acoholic drinks
Other nuts/seeds/pulses
Other oils and fats
Other roots and tuber
Other sweets Confectionary
Other vegetables (fresh or canned)
Palm oil
Palm wine
Pawpaw
Pepper
Pineapples
Pito
Plantains
Pork
Potatoes
Rice-Imported
Rice-local
Sachet water
Salt
Seafood (lobster, crab, prawns)
Sheabutter
Snails
Soft drinks (Coca cola, spirit etc)
Soya beans
Sugar
Sweet Potatoes
Tea
Tomato puree(canned)
Tomatoes
Watermelon
Wheat flour
White beans
Wild game meat
Yam flour
Yam-roots

[127]:

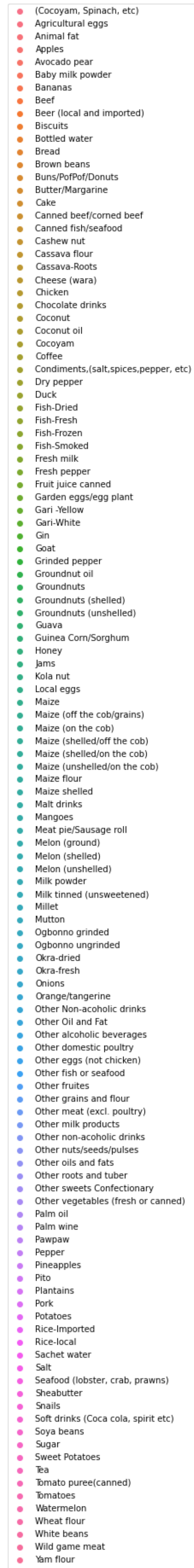
```
scatter4 = sns.scatterplot(data=all_summed_nutrients, x="Q4 (% Ate)", y="Q4␣
↪(Average Consumption)", legend = 'auto', hue = 'index')
plt.ylim(0, 80)
plt.xlim(0, 80)
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
scatter4
```

[127]:

```
<AxesSubplot:xlabel='Q4 (% Ate)', ylabel='Q4 (Average Consumption)'>
```



Q4 (Average Consumption)
Q4 (% Ate)
0
10
20
30
40
50
60
70
80



(Cocoyam, Spinach, etc)
Agricultural eggs
Animal fat
Apples
Avocado pear
Baby milk powder
Bananas
Beef
Beer (local and imported)
Biscuits
Bottled water
Bread
Brown beans
Buns/PofPof/Donuts
Butter/Margarine
Cake
Canned beef/corned beef
Canned fish/seafood
Cashew nut
Cassava flour
Cassava-Roots
Cheese (wara)
Chicken
Chocolate drinks
Coconut
Coconut oil
Cocoyam
Coffee
Condiments,(salt,spices,pepper, etc)
Dry pepper
Duck
Fish-Dried
Fish-Fresh
Fish-Frozen
Fish-Smoked
Fresh milk
Fresh pepper
Fruit juice canned
Garden eggs/egg plant
Gari -Yellow
Gari-White
Gin
Goat
Grinded pepper
Groundnut oil
Groundnuts
Groundnuts (shelled)
Groundnuts (unshelled)
Guava
Guinea Corn/Sorghum
Honey
Jams
Kola nut
Local eggs
Maize
Maize (off the cob/grains)
Maize (on the cob)
Maize (shelled/off the cob)
Maize (shelled/on the cob)
Maize (unshelled/on the cob)
Maize flour
Maize shelled
Malt drinks
Mangoes
Meat pie/Sausage roll
Melon (ground)
Melon (shelled)
Melon (unshelled)
Milk powder
Milk tinned (unsweetened)
Millet
Mutton
Ogbonno grinded
Ogbonno ungrinded
Okra-dried
Okra-fresh
Onions
Orange/tangerine
Other Non-acoholic drinks
Other Oil and Fat
Other alcoholic beverages
Other domestic poultry
Other eggs (not chicken)
Other fish or seafood
Other fruites
Other grains and flour
Other meat (excl. poultry)
Other milk products
Other non-acoholic drinks
Other nuts/seeds/pulses
Other oils and fats
Other roots and tuber
Other sweets Confectionary
Other vegetables (fresh or canned)
Palm oil
Palm wine
Pawpaw
Pepper
Pineapples
Pito
Plantains
Pork
Potatoes
Rice-Imported
Rice-local
Sachet water
Salt
Seafood (lobster, crab, prawns)
Sheabutter
Snails
Soft drinks (Coca cola, spirit etc)
Soya beans
Sugar
Sweet Potatoes
Tea
Tomato puree(canned)
Tomatoes
Watermelon
Wheat flour
White beans
Wild game meat
Yam flour
Yam-roots

[128]:
```
# nutritional df filtered by low nutrients: riboflavin, iron, zinc, and fiber
lowdf = nutritional_df.copy()
lowdf = lowdf.loc[['Zinc, Zn', 'Fiber, insoluble', 'Fiber, soluble', 'Fiber,␣
 ↪total dietary', 'Iron, Fe', 'Riboflavin'], :]
lowdf
```

[128]:
```
                     (Cocoyam, Spinach, etc)  Agricultural eggs  Animal fat  \

Zinc, Zn                                0.00              0.000         0.0
Fiber, insoluble                        0.00              0.000         0.0
Fiber, soluble                          0.00              0.000         0.0
Fiber, total dietary                    1.20              0.000         0.0
Iron, Fe                                2.12              0.000         0.0
Riboflavin                              0.00              0.391         0.0

                      Apples  Avocado pear  Baby milk powder  Bananas   Beef  \

Zinc, Zn                0.00          0.64             0.660    0.150  4.230
Fiber, insoluble        0.00          0.00             0.000    0.000  0.000
Fiber, soluble          0.00          0.00             0.000    0.000  0.000
Fiber, total dietary    2.10          6.70             0.000    2.600  0.000
Iron, Fe                0.15          0.55             1.180    0.260  1.970
Riboflavin              0.00          0.13             0.092    0.073  0.151

                      Beer (local and imported)  Biscuits  …  Tea  \
                                                           …
Zinc, Zn                                  0.010       0.0  …  0.0
Fiber, insoluble                          0.000       0.0  …  0.0
Fiber, soluble                            0.000       0.0  …  0.0
Fiber, total dietary                      0.000       1.3  …  0.0
Iron, Fe                                  0.020       2.4  …  0.0
Riboflavin                                0.025       0.0  …  0.0

                      Tomato puree(canned)  Tomatoes  Unground Ogbono  \

Zinc, Zn                              0.36     0.170            0.090
Fiber, insoluble                      0.00     0.000            0.000
Fiber, soluble                        0.00     0.000            0.000
Fiber, total dietary                  1.90     1.200            1.600
Iron, Fe                              1.78     0.270            0.160
Riboflavin                            0.08     0.019            0.038

                      Watermelon  Wheat flour  White beans  Wild game meat  \

Zinc, Zn                     0.0          0.0         3.54            0.00
```

```
Fiber, insoluble          0.0        0.0       0.00          0.00
Fiber, soluble            0.0        0.0       0.00          0.00
Fiber, total dietary      0.4        2.6       4.30          0.00
Iron, Fe                  0.0        0.0       4.93          0.00
Riboflavin                0.0        0.0       0.00          0.11

                      Yam flour  Yam-roots

Zinc, Zn                   0.00      0.240
Fiber, insoluble           0.00      0.000
Fiber, soluble             0.00      0.000
Fiber, total dietary       1.00      4.100
Iron, Fe                   0.72      0.540
Riboflavin                 0.00      0.032

[6 rows x 132 columns]
```

[129]:
```
# function to create df for food items Nigerians eat with specified nutrient,␣
 ↪is specific to lowdf
def get_low_df(nutrient, df):
    nutr_lst = [self for self in (lowdf.loc[nutrient]) if self>0]
    new = df.T
    nut_series = new[new[nutrient].isin(nutr_lst)].loc[:, nutrient]
    res = nut_series.reset_index().rename(columns = {'index':'Food Item'})
    return res

fiberdf = get_low_df('Fiber, total dietary', lowdf)
irondf = get_low_df('Iron, Fe', lowdf)
b12df = get_low_df('Riboflavin', lowdf)
zincdf = get_low_df('Zinc, Zn', lowdf)
```

[130]:
```
# show df and graphs categorizing the foods Nigerians eat with specified␣
 ↪nutrient using plotly express for highlighting feature
import plotly.express as px

# Fiber
fiberfig = px.scatter(fiberdf, x="Food Item", y="Fiber, total dietary",␣
 ↪color="Food Item")
fiberfig.show()
fiberdf = fiberdf.sort_values(by='Fiber, total dietary',ascending=False)
fiberdf
```

[130]:

	Food Item	Fiber, total dietary
16	Fresh pepper	25.3
57	Pepper	25.3
15	Dry pepper	23.3
47	Okra-dried	20.0

```
13         Coconut                  13.3
..               …                    …
40  Melon (ground)                    0.8
61   Rice-Imported                    0.4
62      Rice-local                    0.4
67      Watermelon                    0.4
27           Honey                    0.2

[72 rows x 2 columns]
```

[131]:
```python
# Iron(Fe)
ironfig = px.scatter(irondf, x="Food Item", y="Iron, Fe", color="Food Item")
ironfig.show()
irondf = irondf.sort_values(by='Iron, Fe',ascending=False)
irondf
```

[131]:
```
                              Food Item  Iron, Fe
81                               Pepper      9.71
29                         Fresh pepper      9.71
22                           Dry pepper      9.60
98                          White beans      4.93
9                           Brown beans      4.70
..                                  …         …
92  Soft drinks (Coca cola, spirit etc)      0.02
83                                 Pito      0.02
6             Beer (local and imported)      0.02
70            Other alcoholic beverages      0.01
79                             Palm oil      0.01

[101 rows x 2 columns]
```

[132]:
```python
# B-12 (Riboflavin)
b12fig = px.scatter(b12df, x="Food Item", y="Riboflavin", color="Food Item")
b12fig.show()
# b12df.sortby(columns='Riboflavin')
b12df = b12df.sort_values(by='Riboflavin',ascending=False)
b12df
```

[132]:
```
                          Food Item  Riboflavin
25                             Goat       0.490
50          Other eggs (not chicken)      0.404
0                 Agricultural eggs       0.391
49            Other domestic poultry      0.323
40        Milk tinned (unsweetened)       0.309
..                              …           …
63  Seafood (lobster, crab, prawns)       0.014
62                       Rice-local       0.013
```

```
61                 Rice-Imported      0.013
48        Other alcoholic beverages      0.007
24                              Gin      0.004

[72 rows x 2 columns]
```

[133]:

```
zincfig = px.scatter(zincdf, x="Food Item", y="Zinc, Zn", color="Food Item")
zincfig.show()
zincdf = zincdf.sort_values(by='Zinc, Zn',ascending=False)
zincdf
```

[133]:

```
                     Food Item  Zinc, Zn
15                      Coffee     15.00
3                         Beef      4.23
27                        Goat      4.00
6                  Brown beans      3.71
7       Canned beef/corned beef      3.57
..                         ...       ...
59                        Pito      0.01
29               Groundnut oil      0.01
70                       Sugar      0.01
4   Beer (local and imported)      0.01
37                 Malt drinks      0.01

[77 rows x 2 columns]
```