In [10]:

```
# =============================================================================
# CONSUMPTION 22 MALAWI VILLAGE
# =============================================================================

root_path = 'C:/Users/rodri/Dropbox/Malawi/SIEG2021 (1)/Data Collection July 2022'
path_19 = 'C:/Users/rodri/Dropbox/Malawi/Chied_Field_June_19/Data/'

import numpy as np
import pandas as pd
import os
os.chdir(root_path+'/Code/Phase 3/Auxiliary files/')
from data_functions_albert import remove_outliers, gini

# Set the working directory
os.chdir(root_path+'/Data/Phase 3/Consumption')

save=True

## Display  set-up
pd.options.display.float_format = '{:,.2f}'.format
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)

os.environ['PYTHONWARNINGS']='ignore::FutureWarning'
import warnings
warnings.filterwarnings("ignore")

percentiles = [0.05, .25, .5, .75, 0.95, 0.99]

#July 14th 2022 MWK vs US dollar
dollar_MWK = 1030.36

# Import village 19 data
data19 = pd.read_csv(path_19+'/Finished dataframes/data19_w18.csv')


# =============================================================================
# Import data: Data from the field and conversion rates (ISA-LSMS price conversions)
# =============================================================================

data = pd.read_stata(root_path+"/Data/Raw/2022-SIEG-Phase 3-Final Data.dta", convert

# take out from the data hhid
# had very extreme values in maize consumption, clothes consumption, other non-durab
# household was problematic.

data.drop(data.index[data['hhid'] ==1330], inplace = True)

##### Create conversion kg matrix(unitxitems) with the exact same names and units La

#item labels data
list_items = ['maizemgaiwa', 'maizerefined', 'maizemadeya', 'maizegrain', 'greenmaiz
, 'ipotatoes', 'potatocrisps', 'bbean', 'pigeonpea', 'groundnut', 'groundnutf', 'oni
'driedfish', 'fleshfish', 'goat', 'chicken', 'otherpoultry', 'smockedfish', 'mango',
'wildfruits',  'sugar', 'sugarcane', 'cookingoil', 'softdrinks',
'thobwa', 'locallybrewed', 'salt', 'fingermillet', 'mandazidou']


'''
noncon_items = ['potatocrisps','otherpoultry','mango','guava', 'locallybrewed','fing

for element in list_items:
```

```
    if element in noncon_items:
        list_items.remove(element)
'''


### NOTE ON CONVERSIONS ================================
# Using ISA-LSMS 17 I didnt have crop-units conversions for several units. What I ma
# 1. Check if missing units are the ones from upper numbers (above 25)
# 2. Use conversion units from the production side for the crop-units possible: pail
# 3. Use conversion units from the consumption side of an older ISA-LSMS (15): bale,
conversionkg_pivot = pd.read_csv('conversions/conversionkg_final_matrix.csv',  index

#4.  All units have at least one crop conversion. To fill the whole matrix I use the
conversionkg_pivot = conversionkg_pivot.apply(lambda x: x.fillna(x.median()),axis=1)

conversion_median =conversionkg_pivot.median(axis=1).to_frame()
conversion_median.columns =['conversionkg']

#if save==True:
#    conversion_median.to_csv('conversions/median_conversions_kg.csv')


# ============================================================================
# Generate empty variables
# ============================================================================

#Obtain the names of the variables per each question of item. Question c is monetary
a_var = []
b_var = []
c_var = []
d_var = []

#Generate variable lables in a list
for item in list_items :
    a = item+'_a'
    b= item+'_b'
    c = item+'_c'  ## expenditure
    d = item+'_d'

    a_var.append(a)
    b_var.append(b)
    c_var.append(c)
    d_var.append(d)

list_questions = ['a','b','d']


# check question on whether did something in return and what


# convert all empty observations to 0. I do that to convert empty units to 99. If no
# Note that empty doesn't necessary mean 0, so we careful at looking the data
#data = data.stack().apply(pd.to_numeric, errors='ignore').fillna(0).unstack()


# Drop nan observations. Also drop unit 25 (number not in our choices). Also drop 24


#there is an issue with unit 3 for some items. Diddnt have this problem in 2019 or i
# unit3 refers to consumption coming from own-production. thus, it is natural that t

data['ipotatoes_unit3'] = np.nan
data['potatocrisps_unit3'] = np.nan
data['cabbage_unit3'] = np.nan
```

```
data['driedfish_unit3'] = np.nan
data['fleshfish_unit3'] = np.nan
data['goat_unit3'] = np.nan
data['otherpoultry_unit2'] = np.nan
data['smockedfish_unit3'] = np.nan
data['sugar_unit3'] = np.nan
data['cookingoil_unit3'] = np.nan
data['softdrinks_unit3'] = np.nan
data['locallybrewed_unit3'] = np.nan
data['salt_unit3'] = np.nan
data['mandazidou_unit3'] = np.nan


#Find the households-questions that reported other units.
df_other_units = pd.DataFrame(columns=['hhid','question', 'other_unit'])
for var in list_items:
    for i in range(1,4):   #Loop over unit questions.
        # Find who said other units
        other_units_guy = data.loc[data[var+'_unit'+str(i)]=='other', ['hhid', var+'
        if other_units_guy.empty:
            continue
        else:
            d = {'hhid': other_units_guy.iloc[:,0], 'question': other_units_guy.colu
            row = pd.DataFrame(data=d)
            df_other_units = df_other_units.append(row)

df_other_units['kg'] = np.nan
df_other_units.to_csv('other units/other_units_consumption.csv')

print('===============================================================')
print('All households-item-question combinations that reported "other" units')
print('===============================================================')
print(df_other_units)

# Create other units dataset when we have more info

#df_other_units2 = pd.read_excel('other units/other_units_consumption_conversion.xls

### add Leandro conversions:

#df_other_units = df_other_units[['hhid',,'other_unit','kg']]



for var in list_items:
    for i in range(1,4):   #Loop over unit questions.
        data[[var+'_unit'+str(i)]] = data[[var+'_unit'+str(i)]].replace('other', 100
        data[[var+'_unit'+str(i)]] = data[[var+'_unit'+str(i)]].replace(np.nan, 99)
        data[[var+'_unit'+str(i)]] = data[[var+'_unit'+str(i)]].replace(25, 99)
        data[[var+'_unit'+str(i)]] = data[[var+'_unit'+str(i)]].replace(23, 99)
        data[[var+'_unit'+str(i)]] = data[[var+'_unit'+str(i)]].replace(0, 99)
        data[[var+'_unit'+str(i)]] = data[[var+'_unit'+str(i)]].replace('', 99)
```

```
===============================================================
All households-item-question combinations that reported "other" units
===============================================================
     hhid             question                       other_unit  quantity  kg
177  1349   cassavatubers_unit1                             Pot      1.00 nan
177  1349   cassavatubers_unit3                             Pot      1.00 nan
177  1349       groundnut_unit1                            Hand      1.00 nan
```

```
119  1232        chicken_unit1  Full Chicken  and 4 pieces      1.40 nan
208  1435         thobwa_unit3                         Pot      1.00 nan
```

In [13]:

```
#%% ===========================================================================
#   Convert to kgs:
# =============================================================================

# Generate kg variables empty
for item in list_items:
    # items not yet in the data:

    for q in list_questions:
        data[item+'_'+q+'kg']= np.nan  # from total reported quantity   (rice_akg: t
    data[item+'_kg2']= np.nan  # from summing bought+own-produced (rice_kg2: bought+
    data[item+'_difftotal_kg']= np.nan # difference total reported - bought+own-prod

print('================================================================')
print('a: Total Consumption (in kg)')
print('================================================================')
for var in a_var:
    item = var[:-2]

    for i in range(len(data)):
        unit_code = int(data.iloc[i,  data.columns.get_loc(item+'_unit1')])
        data.iloc[i,data.columns.get_loc(var+'kg')] = data.iloc[i,data.columns.get_l
    #print(data[[var+'kg']].describe())


data.loc[data['hhid']==1232, 'chicken_akg'] = 2*(1.25) # answer other units: 1 chick
data.loc[data['hhid']==1349, 'cassavatubers_akg'] = 0.5
data.loc[data['hhid']==1349, 'groundnut_akg'] = 0.5





### Check households with an extreme value of a food kg consumption from previous de
# First check if it is an issue of conversion units.
# IF not, we might have to reinterview them or use the consumption summing bought, o



# Replace extreme values for median (let's be careful with this)
# WE MIGHT WANT TO CHANGE THESE CORRECTIONS OF EXTREME VALUES
data.loc[data['sugar_akg']>5,['sugar_akg']] = data['sugar_akg'].median()
data.loc[data['mandazidou_akg']>10,['mandazidou_akg']] = data['mandazidou_akg'].medi
data.loc[data['thobwa_akg']>30,['thobwa_akg']] = data['thobwa_akg'].median()
data.loc[data['tomato_akg']>10,['tomato_akg']] = data['tomato_akg'].median()
data.loc[data['tanaposi_akg']>10,['tanaposi_akg']] = data['tomato_akg'].median()
data.loc[data['maizemgaiwa_akg']>60,['maizemgaiwa_akg']] = data['maizemgaiwa_akg'].m
data.loc[data['maizerefined_akg']>60,['maizerefined_akg']] = data['maizerefined_akg'

print('================================================================')
print('a: Total Consumption (in kg) after corrections (cleaned)')
print('================================================================')

for var in a_var:
    item = var[:-2]
    print(data[[var+'kg']].describe())



# there might be some other outliers. Like 15kg of sweet potatoes, 22kg maizemadeya,
# for the moment to be careful, I keep them as they are.
```

```
#print('b: Bought')
for var in b_var:
    item = var[:-2]
    for i in range(len(data)):
        data.iloc[i,data.columns.get_loc(var+'kg')] = data.iloc[i,data.columns.get_l

    #print(data[[var+'kg']].describe())


data.loc[data['sugar_bkg']>5,['sugar_bkg']] = data['sugar_bkg'].median()
data.loc[data['tanaposi_bkg']>10,['tanaposi_bkg']] = data['tomato_bkg'].median()
data.loc[data['maizemgaiwa_bkg']>60,['maizemgaiwa_bkg']] = data['maizemgaiwa_bkg'].m
data.loc[data['maizerefined_bkg']>60,['maizerefined_bkg']] = data['maizerefined_bkg'



#print('d: Own-produced')
for var in d_var:
    item = var[:-2]
    for i in range(len(data)):
        data.iloc[i,data.columns.get_loc(var+'kg')] = data.iloc[i,data.columns.get_l
    #print(data[[var+'kg']].describe())



data.loc[data['thobwa_dkg']>30,['thobwa_dkg']] = data['thobwa_dkg'].median()
data.loc[data['tanaposi_dkg']>10,['tanaposi_dkg']] = data['tomato_dkg'].median()




### compute total quantity kg 2 (bought+own produced)
for item in list_items:
    data[item+'_kg2']= data[item+'_bkg'].fillna(0) + data[item+'_dkg'].fillna(0)
    data[item+'_difftotal_kg']= data[item+'_akg'].fillna(0) - data[item+'_kg2'].fill
```

```
=================================================================
a: Total Consumption (in kg)
=================================================================
=================================================================
a: Total Consumption (in kg) after corrections (cleaned)
=================================================================
       maizemgaiwa_akg
count           225.00
mean             13.09
std               9.80
min               0.88
25%               5.00
50%              10.00
75%              19.20
max              46.67
       maizerefined_akg
count            121.00
mean               9.99
std               10.05
min                0.48
25%                3.00
50%                5.00
75%               12.50
max               50.00
       maizemadeya_akg
count           116.00
mean              3.68
```

```
std            3.27
min            0.35
25%            1.43
50%            2.86
75%            4.50
max           22.50
       maizegrain_akg
count        102.00
mean           0.51
std            0.36
min            0.04
25%            0.35
50%            0.35
75%            0.71
max            2.25
       greenmaize_akg
count         20.00
mean           1.11
std            1.28
min            0.34
25%            0.34
50%            0.67
75%            1.01
max            5.05
       rice_akg
count     90.00
mean       0.92
std        1.12
min        0.09
25%        0.42
50%        0.71
75%        1.00
max       10.00
       cassavatubers_akg
count           147.00
mean              1.29
std               1.45
min               0.16
25%               0.50
50%               0.95
75%               1.58
max               9.45
       wsweetpotatoes_akg
count              88.00
mean                1.62
std                 1.73
min                 0.25
25%                 0.76
50%                 1.26
75%                 2.53
max                15.00
       osweetpotatoes_akg
count              42.00
mean                1.68
std                 0.98
min                 0.23
25%                 0.93
50%                 1.28
75%                 2.33
max                 3.49
       ipotatoes_akg
count         18.00
mean           0.96
std            0.59
min            0.43
25%            0.50
50%            0.71
75%            1.31
max            2.50
```

```
       potatocrisps_akg
count             11.00
mean               0.39
std                0.11
min                0.05
25%                0.43
50%                0.43
75%                0.43
max                0.43
       bbean_akg
count      60.00
mean        0.46
std         0.21
min         0.06
25%         0.35
50%         0.43
75%         0.45
max         1.05
       pigeonpea_akg
count         215.00
mean            0.97
std             0.70
min             0.20
25%             0.50
50%             0.75
75%             1.00
max             5.00
       groundnut_akg
count          69.00
mean            0.48
std             0.67
min             0.01
25%             0.25
50%             0.38
75%             0.50
max             5.00
       groundnutf_akg
count          114.00
mean             0.18
std              0.13
min              0.01
25%              0.10
50%              0.15
75%              0.29
max              0.86
       onion_akg
count     177.00
mean        0.27
std         0.22
min         0.08
25%         0.08
50%         0.24
75%         0.32
max         1.54
       cabbage_akg
count        47.00
mean          1.27
std           1.46
min           0.12
25%           0.43
50%           1.22
75%           1.61
max           8.00
       tanaposi_akg
count        122.00
mean           0.77
std            0.49
min            0.23
25%            0.46
```

```
50%           0.70
75%           0.93
max           2.78
       leafyvegetables_akg
count               163.00
mean                  0.47
std                   0.31
min                   0.14
25%                   0.29
50%                   0.43
75%                   0.50
max                   2.13
       tomato_akg
count      266.00
mean         1.85
std          1.50
min          0.24
25%          0.50
50%          1.32
75%          2.64
max          7.03
       eggs_akg
count     66.00
mean       0.30
std        0.19
min        0.06
25%        0.18
50%        0.24
75%        0.36
max        1.14
       driedfish_akg
count         225.00
mean            1.09
std             1.14
min             0.09
25%             0.43
50%             0.70
75%             1.39
max             9.74
       fleshfish_akg
count         122.00
mean            0.93
std             0.83
min             0.21
25%             0.43
50%             0.70
75%             1.04
max             5.29
       goat_akg
count     38.00
mean       0.66
std        0.49
min        0.06
25%        0.43
50%        0.50
75%        1.00
max        2.00
       chicken_akg
count       102.00
mean          3.47
std           3.26
min           0.43
25%           2.00
50%           2.00
75%           3.28
max          17.47
       otherpoultry_akg
count             10.00
mean               3.56
```

```
std              1.83
min              2.00
25%              2.00
50%              3.78
75%              4.00
max              8.00
       smockedfish_akg
count           139.00
mean              0.77
std               0.77
min               0.06
25%               0.33
50%               0.67
75%               1.00
max               6.66
       mango_akg
count     195.00
mean        1.67
std         1.44
min         0.20
25%         0.61
50%         1.22
75%         2.03
max        10.00
       banana_akg
count        91.00
mean          2.11
std           1.93
min           0.17
25%           1.01
50%           1.14
75%           2.53
max          12.67
       guava_akg
count      23.00
mean        0.30
std         0.32
min         0.09
25%         0.17
50%         0.26
75%         0.26
max         1.71
       wildfruits_akg
count           29.00
mean             1.01
std              1.09
min              0.25
25%              0.50
50%              0.50
75%              1.25
max              5.00
       sugar_akg
count     149.00
mean        0.49
std         0.43
min         0.02
25%         0.18
50%         0.26
75%         1.00
max         2.00
       sugarcane_akg
count          24.00
mean            3.23
std             2.44
min             0.70
25%             1.79
50%             2.31
75%             4.62
max            11.54
```

```
       cookingoil_akg
count          216.00
mean             0.31
std              0.23
min              0.01
25%              0.13
50%              0.27
75%              0.41
max              2.17
       softdrinks_akg
count           18.00
mean             0.49
std              0.43
min              0.07
25%              0.25
50%              0.35
75%              0.50
max              2.00
       thobwa_akg
count      200.00
mean         5.10
std          6.42
min          0.25
25%          0.71
50%          2.00
75%          5.62
max         30.00
       locallybrewed_akg
count               6.00
mean                0.35
std                 0.26
min                 0.00
25%                 0.25
50%                 0.28
75%                 0.47
max                 0.75
       salt_akg
count    272.00
mean       0.29
std        0.25
min        0.01
25%        0.11
50%        0.22
75%        0.33
max        1.10
       fingermillet_akg
count              9.00
mean               0.97
std                1.64
min                0.11
25%                0.11
50%                0.11
75%                1.00
max                5.00
       mandazidou_akg
count           70.00
mean             0.32
std              0.57
min              0.06
25%              0.12
50%              0.18
75%              0.24
max              4.00
```

In [15]:

```python
#Check

#check total consumption in kgs
data['total_foodkg'] = 0
```

```
data['total_foodkg2'] = 0
data['purchased_kg'] = 0
data['ownproduced_kg'] = 0

for item in list_items:
    data['total_foodkg'] += data[item+'_akg'].replace(np.nan, 0)
    data['purchased_kg'] += data[item+'_bkg'].replace(np.nan, 0)
    data['ownproduced_kg'] += data[item+'_dkg'].replace(np.nan, 0)

data['total_foodkg2'] =  data['purchased_kg'] + data['ownproduced_kg']
sumtotalfoodkg = data[['total_foodkg', 'total_foodkg2', 'purchased_kg','ownproduced_
print('================================================================')

print('==== Summary Food Consumption last 7 days in kgs aggregated across items ====
print('================================================================')
print(sumtotalfoodkg)

print('Foodkg is reported total food consumption. foodkg2 is the sum or purchases an

print('a potential thing I can do is to compute purchases+own_produced+transfers.')

print('')
print('================================================================')
print('Check: Total kg vs Bought+own-produced kg.(All food items together)')
print('================================================================')

buy_larger_total = data.loc[(data['purchased_kg']>data['total_foodkg']+2),['hhid','t
prod_larger_total = data.loc[(data['ownproduced_kg']>data['total_foodkg']+2),['hhid'

print(buy_larger_total)
print('Reported c-buying more kg than total kg consumption. Difficult to argue which
print(prod_larger_total)
print('Reported c-ownproducing kg more kg than total kg consumption')

### check hhs with very low and very high total consumption.
### Check if error comes from a particular crop.
### check if error comes from unit conversion issue.
### If not to above, reinterview.
```

```
================================================================
==== Summary Food Consumption last 7 days in kgs aggregated across items =======
================================================================
       total_foodkg  total_foodkg2  purchased_kg  ownproduced_kg
count        272.00         272.00        272.00          272.00
mean          33.22          30.71         19.65           11.06
std           17.84          18.04         14.35           13.03
min            6.98           0.75          0.75            0.00
25%           19.43          16.12          9.37            1.78
50%           30.90          28.64         15.55            5.40
75%           43.20          41.27         28.22           16.64
max          108.46         107.46         85.88           73.41
Foodkg is reported total food consumption. foodkg2 is the sum or purchases and own p
roduced. Interestingly the tewo distributions look quite similar. Though notice that
(1) should includes transfers while (2) not. There are also more outliers in (2)
a potential thing I can do is to compute purchases+own_produced+transfers.

================================================================
Check: Total kg vs Bought+own-produced kg.(All food items together)
================================================================
Reported c-buying more kg than total kg consumption
     hhid  total_foodkg  purchased_kg
2    1003         56.15         58.23
166  1337         50.18         85.88
194  1419         32.11         47.38
Reported c-ownproducing kg more kg than total kg consumption
Empty DataFrame
```

```
Columns: [hhid, total_foodkg, ownproduced_kg]
Index: []
===============================================================
Hhs that reported kg consumption from buying+own production larger than total (per c
rop)
===============================================================
['maizemgaiwa', 2      1003
166    1337
194    1419
Name: hhid, dtype: int16, 'cassavatubers', 77    1133
Name: hhid, dtype: int16, 'wsweetpotatoes', 207    1434
Name: hhid, dtype: int16, 'osweetpotatoes', 130    1244
Name: hhid, dtype: int16, 'pigeonpea', 75    1131
Name: hhid, dtype: int16, 'mango', 109    1220
191    1416
Name: hhid, dtype: int16, 'thobwa', 1    1002
Name: hhid, dtype: int16]
===============================================================
Hhs that reported total kg consumption MUCH larger than from buying+own production
(per crop)
===============================================================
['maizemgaiwa', 158    1329
260    1535
Name: hhid, dtype: int16, 'chicken', 269    1547
Name: hhid, dtype: int16, 'thobwa', 256    1531
269    1547
Name: hhid, dtype: int16]
these could be potential outliers
```

In [18]:

```
#%%  CONVERT TO MONETARY VALUE

### CHECK PRICES FROM ISA-LSMS

data['otherpoultry_c'] = np.nan

p_isalsms = pd.read_stata('prices/price_consumption_kg.dta')
# Can we get a more recent year?
p_isalsms = p_isalsms.loc[(p_isalsms['region']=='Southern')& (p_isalsms['year']==201
p_isalsms = p_isalsms.groupby(by=['crop_code']).median()

p_isalsms_sell = pd.read_stata('prices/price_prioduction_kg.dta')
p_isalsms_sell = p_isalsms_sell.groupby(by=['crop_code']).median()

# Compute village prices:

# Generate price variables
for item in list_items:
        data[item+'_price']= np.nan

# price per household
for item in list_items:
    data[item+'_price'] = data[item+'_c'] / data[item+'_bkg'].replace(0,np.nan)

price_data = pd.DataFrame(list_items, columns=['good'])
price_data['p_c'] = np.nan

for item in list_items:
    #print('Median Price 1 kg of '+item)
    data['med_price_'+item] = data[item+'_price'].median()
    #print(data['med_price_'+item].mean())
    price_data.loc[price_data['good']==item,'p_c'] = data['med_price_'+item].mean()

 ### For nan values use price of similar food items
price_data.loc[price_data['good']=='otherpoultry','p_c'] = float(price_data.loc[pric
price_data.loc[price_data['good']=='mango','p_c'] = float(price_data.loc[price_data[
```

```
if save==True:
    price_data.to_csv('prices/village_c_prices_22.csv', index=False)


# For the check let's use the prices from the village in 2019
p_19 = pd.read_csv(path_19+'/Consumption/prices/village_c_prices.csv')
p_19.columns = ['good', 'p_c_19']
p_19 = p_19.merge(price_data, on='good', how='outer')
p_19.columns = ['good', 'p_c_19', 'p_c_22']

print(' Comparison median consumption prices (per kg) in the villlage: 2019 vs 2022
print(p_19)

for item in list_items:
    for q in list_questions:
        data[item+'_'+q+'MWK']= np.nan

# Total consumption
for item in list_items:
    #print(item)
    data[item+'_aMWK'] = data[item+'_akg']*float(price_data.loc[price_data['good']==
    '''
    if data[item+'_aMWK'].count()>0:
        print('Food Consumption in MWK during last 7 days item: '+item)
        print(data[item+'_aMWK'].describe(percentiles=percentiles))
    '''


# Bought
for item in list_items:
    #print(item)
    data[item+'_bMWK'] = data[item+'_bkg']*float(price_data.loc[price_data['good']==

# own-produced
for item in list_items:
    #print(item)
    data[item+'_dMWK'] = data[item+'_dkg']*float(price_data.loc[price_data['good']==


#check total consumption
data['c_food'] = 0
data['c_food_purch'] = 0
data['c_food_ownprod'] = 0

for item in list_items:
    data['c_food'] += data[item+'_aMWK'].replace(np.nan, 0)
    data['c_food_purch'] += data[item+'_bMWK'].replace(np.nan, 0)
    data['c_food_ownprod'] += data[item+'_dMWK'].replace(np.nan, 0)

data[['c_food', 'c_food_purch' , 'c_food_ownprod']] = data[['c_food', 'c_food_purch'


pd.options.display.float_format = '{:,.2f}'.format
sumcfood= ((data[['c_food', 'c_food_purch' , 'c_food_ownprod']]/dollar_MWK).replace(
print('===============================================================')
print('==== Summary Food Consumption 7 days in $ =======')
print('===============================================================')
print(sumcfood)
```

```
 Comparison median consumption prices (per kg) in the villlage: 2019 vs 2022 =======
========
               good    p_c_19    p_c_22
0       maizemgaiwa    200.00    340.00
```

```
1      maizerefined   193.33   200.00
2       maizemadeya   160.00   104.76
3        maizegrain   319.70   480.81
4        greenmaize   173.20   296.91
5              rice   560.00 1,440.00
6     cassavatubers   158.65   158.65
7    wsweetpotatoes    79.13   237.40
8    osweetpotatoes    85.99   257.98
9         ipotatoes   262.50   450.00
10     potatocrisps   862.49 1,174.47
11            bbean   718.75 1,428.57
12        pigeonpea   240.00   500.00
13        groundnut   400.00 1,000.00
14       groundnutf   700.28 1,500.00
15            onion   616.25   616.25
16          cabbage   163.89   204.87
17         tanaposi   215.62   215.62
18  leafyvegetables   346.62   646.62
19           tomato   113.82   227.64
20             eggs 1,671.83 2,507.74
21        driedfish   574.99   862.49
22        fleshfish   530.03   790.62
23             goat 2,000.00 2,800.00
24          chicken   950.00 1,500.00
25     otherpoultry   168.45 1,500.00
26      smockedfish   600.60   900.90
27           banana    87.39   118.43
28            guava   117.12   390.41
29       wildfruits   200.13    33.36
30            sugar 1,000.00 1,142.86
31        sugarcane    43.32    86.65
32       cookingoil   800.00 1,500.00
33       softdrinks   833.33 1,272.73
34           thobwa   200.00   282.83
35    locallybrewed 1,250.00 2,000.00
36             salt   454.55   909.09
37       mandazidou   825.99 1,651.98
38            mango      nan   390.41
39     fingermillet      nan 1,000.00
===============================================================
==== Summary Food Consumption 7 days in $ =======
===============================================================
       c_food  c_food_purch  c_food_ownprod
count  272.00        272.00          258.00
mean    13.68          8.10            4.35
std      8.05          5.60            5.31
min      2.05          0.44            0.04
5%       4.21          1.60            0.24
25%      7.54          4.22            0.77
50%     12.48          6.71            2.44
75%     17.45         10.53            5.93
95%     30.07         17.81           14.71
99%     36.89         25.60           23.75
max     51.25         35.05           34.16
```

In [20]:

```
## non-food consumption  (month) ========================

data['c_housing'] = data['nonf_cons_a_1']*3
data['c_clothes'] = data['nonf_cons_b_1']
data['c_education'] = data['nonf_cons_c_1']
data['c_health'] = data['nonf_cons_d_1']
data['c_funeralout'] = data['nonf_cons_e_1']
data['c_funeralin'] = data['nonf_cons_f_1']
data['c_weddingout'] = data['nonf_cons_g_1']
data['c_weddingin'] = data['nonf_cons_h_1']

print('outliers checked by Augustine. One of the extreme values verified. Some hh mo
```

```
data['c_nonfood'] = data[['c_housing', 'c_clothes', 'c_education', 'c_health', 'c_fu

sum_cnonfood = ((data[['c_nonfood','c_housing', 'c_clothes', 'c_education', 'c_healt
print('=================================================================')
print(' SUMMARY  NON-FOOD CONSUMPTION (3 MONTH LEVEL, in $)')
print('=================================================================')
print(sum_cnonfood)
print('looks all good. Outliers do not seem crazy')

outl_food = data.loc[data['c_food']>150*dollar_MWK,['hhid','c_food','maizemgaiwa_aMW

otl_housing = data.loc[data['c_housing']>1000*dollar_MWK,['hhid','c_housing','c_food
otl2 = data.loc[data['c_clothes']>100*dollar_MWK,['hhid','c_clothes']]
otl3 = data.loc[data['c_education']>100*dollar_MWK,['hhid','c_education']]
otl4 = data.loc[data['c_funeralin']>100*dollar_MWK,['hhid', 'c_weddingout']]
otl5 = data.loc[data['c_funeralin']>100*dollar_MWK,['hhid', 'c_weddingout']]



#data[['c_food_purch', 'c_housing']] = remove_outliers(data[['c_food_purch', 'c_hous
```

```
outliers checked by Augustine. One of the extreme values verified. Some hh move out
of village while hhid 1330 we removed it from the survey.
=================================================================
 SUMMARY  NON-FOOD CONSUMPTION (3 MONTH LEVEL, in $)
=================================================================
       c_nonfood  c_housing  c_clothes  c_education  c_health  c_funeralout  \
count     272.00     270.00     141.00       149.00    202.00         58.00
mean       53.98      35.99      15.38         8.42      5.83          3.88
std        67.84      43.21      29.38        20.93     10.60         10.67
min         0.19       0.73       0.19         0.10      0.05          0.05
25%        14.12       8.77       1.94         1.94      0.49          0.49
50%        30.14      19.51       4.85         3.40      1.99          0.97
75%        60.27      43.67      16.50         7.28      6.79          1.94
max       510.50     232.93     217.69       213.52     97.05         74.25

       c_funeralin  c_weddingout  c_weddingin
count         6.00         41.00         2.00
mean          8.88          2.01         0.87
std          19.43          2.24         0.82
min           0.19          0.19         0.29
25%           0.39          0.49         0.58
50%           1.21          0.97         0.87
75%           1.82          2.43         1.16
max          48.53          9.71         1.46
looks all good. Outliers do not seem crazy
```

In [22]:

```
#data[['c_food_purch','c_food_ownprod']] = remove_outliers(data[['c_food_purch','c_f

## short dataset
datacon_short = data[['hhid','c_food','c_food_purch','c_food_ownprod', 'c_nonfood','

## Food at 3 months level
datacon_short[['c_food','c_food_purch','c_food_ownprod']] = datacon_short[['c_food',
datacon_short['ctotal'] = datacon_short[['c_nonfood', 'c_food']].sum(axis = 1, skipn


if save==True:
    datacon_short.to_csv('cons_22_3months.csv', index=False)


## Consumption at year level
datacon_short[['ctotal','c_food','c_food_purch','c_food_ownprod', 'c_nonfood','c_hou
c_summary =  ((datacon_short[['ctotal','c_food','c_food_purch','c_food_ownprod', 'c_
```

```python
print('=================================================================')
print(' Total Consumption Summary (year level, in $)')
print('=================================================================')


print(c_summary)
if save==True:
    datacon_short.to_csv('cons_22_year.csv', index=False)
##Long dataset (at rainy season)
```

```
=================================================================
 Total Consumption Summary (year level, in $)
=================================================================
        ctotal   c_food  c_food_purch  c_food_ownprod  c_nonfood  c_housing  \
count   272.00   272.00        272.00          258.00     272.00     270.00   
mean    872.61   656.71        389.03          208.61     215.91     143.97   
std     541.11   386.61        268.96          254.75     271.36     172.86   
min     109.45    98.38         21.01            1.73       0.78       2.91   
5%      227.40   202.11         76.94           11.38      15.07      12.49   
25%     478.29   361.72        202.65           37.18      56.49      35.08   
50%     764.34   598.87        322.04          117.07     120.54      78.03   
75%   1,165.61   837.76        505.33          284.41     241.08     174.70   
95%   1,903.42 1,443.37        854.93          706.16     828.25     579.18   
99%   2,361.44 1,770.67      1,228.98        1,140.17   1,180.95     775.07   
max   3,531.70 2,460.22      1,682.23        1,639.92   2,042.00     931.71   

       c_clothes  c_education  c_health  
count     141.00       149.00    202.00  
mean       61.51        33.68     23.33  
std       117.53        83.72     42.41  
min         0.78         0.39      0.19  
5%          2.91         2.10      0.78  
25%         7.76         7.76      1.94  
50%        19.41        13.59      7.96  
75%        66.00        29.12     27.17  
95%       225.16       139.76     81.33  
99%       542.72       327.81    193.45  
max       870.76       854.07    388.21  
```

In [ ]: