In [3]:

```
# =============================================================================
# food transfers July 2022
# =============================================================================

import numpy as np
import pandas as pd
import os
import warnings
warnings.filterwarnings("ignore")

os.chdir('C:/Users/rodri/Dropbox/Malawi/SIEG2021 (1)/2022 July/Data/Clean data/Phase
percentiles = [0.05, 0.1, .25, .5, .75, 0.8, 0.9, 0.95, 0.99]
#July 14th 2022 MWK vs US dollar
dollar_MWK = 1030.36
pd.options.display.float_format = '{:,.2f}'.format

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


#  I followed Pau's code in 2019 to generate a dataset containing all the food trans
data = pd.read_stata("transfers.dta")


# Now, let's use this data to
# (1) convert the transfers to kgs and monetary units.
# (2) Provide summary stats of the transfers.
# (3) Create food transfers datasets
# (4) Create measures of total transfers_in transfers_out at household level to appe
#inreturn_sum = pd.value_counts(data['in.return'])



# (1) convert the transfers to kgs and monetary units. -----------------------------
data.replace('cassava', 'cassavatubers', inplace=True)
data.replace('potatocrips', 'potatocrisps', inplace=True)

list_items = list(data['good'].unique())
list_units = list(data['units'].unique())

## Other units
# hands/hand/handful conversion rate
data.loc[data['unit.other'].isin(['Hand','Hands']), 'transf_kg'] = 0.113 ##0.5 cup/
# Dove
data.loc[data['unit.other'].isin(['Dove']), 'transf_kg'] = 0.3  ## average weight of
# Pot
data.loc[data['unit.other'].isin(['Pot']), 'transf_kg'] = 0.5  ##from previous year
data_other = data.loc[data['units']=='other']


# Leandro-Raul kgs through price method

### NOTE ON CONVERSIONS =================================
conversionkg_pivot = pd.read_csv('C:/Users/rodri/Dropbox/Malawi/Chied_Field_June_19/

#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)

## import median price of goods
prices = pd.read_csv('C:/Users/rodri/Dropbox/Malawi/SIEG2021 (1)/2022 July/Data/Clea
# there was not consumption of samosas in the village (not possible to compute price
# so I'll use the price for mandazidou
conversionkg_pivot['samosa'] = conversionkg_pivot['mandazidou']
prices.loc[40] =['samosa',1651.98]
prices.loc[41] =['chips',       1174.4]
# chips/crisps
conversionkg_pivot['chips'] = conversionkg_pivot['potatocrisps']

data['units'] = data['units'].replace('other',100)
data['units'] = pd.to_numeric(data['units'])
data['transf_kg'] = np.nan
data[['units']] = data[['units']].replace([np.nan, 23, 25, 0], 99)


## Convert to kgs
for i in range(0,len(data)):
    good = data['good'][i]
    if good != 'fert':
        data['transf_kg'][i] = data['quant'][i]*conversionkg_pivot.loc[int(data['uni


data.loc[data['good']=='fert','transf_kg'] = data.loc[data['good']=='fert','kg']

## Convert to MWK

# get a price from fertilizer for those that had to pay
print('for the food items I use the median consumption prices in the village')

print('for fertilizer I used the mean price reported in the agricultural production

p_fert =  14620.3665/50   # mean price 50kg fertilizer bag 14620.366563891745

new_row = {'good': 'fert', 'p_c': p_fert}
prices = prices.append(new_row, ignore_index=True)

data = data.merge(prices, on='good', how='left')
data['transf_MWK'] = data['transf_kg']*data['p_c']
data['transf_dollar'] = data['transf_MWK']/dollar_MWK


# for the moment dont remove outliers. Now we have fertilizer so outliers not so cle
'''
def fun(x):
    q_99 = x.quantile(0.95)
    q_1 = x.quantile(0.00)
    return (x>q_99) | (x<q_1)

print('Summary transfers in the village')
print(data['transf_dollar'].describe(percentiles=[0.25, .5, .75, 0.95, 0.99]))

data['outlier'] = 1*(data['transf_dollar']>data['transf_dollar'].quantile(0.995))
data = data.loc[data['outlier']==0]

data = data.drop(columns='outlier')
'''
print('===============================================================')
print('Summary transfers in the village')
print('===============================================================')
print(data['transf_dollar'].describe(percentiles=[0.1,.5,0.9]))

print('===============================================================')
print('Summary only FOOD transfers')
print('===============================================================')
print(data.loc[data['good']!='fert', ['transf_kg','transf_dollar']].describe(percent
print(data['good'].describe())

print('=================================================================')
print('Summary only fertilizer')
print('=================================================================')
print(data.loc[data['good']=='fert', ['transf_kg','transf_dollar','recipro.cashback'

N_fert = len(data.loc[data['good']=='fert','good'])
N_payfert = sum(data['recipro.cashback']>0)
print('Proportion fert transfers household had to pay back: ',round(N_payfert/N_fert

print('Comparison value given median price vs payback for those hhs that reported to
print(data.loc[(data['good']=='fert') & (data['recipro.cashback']>0), ['transf_kg','
print('There are some large discrepancies but avg is similar')

del data['kg']
# .
data.to_csv('transfers_in_kg_MWK_22.csv')

print('food+fertilizer transfers data in long format with values to kgs and MWK save

#%% Create household level transfers dataset ========================================

all_count_bydirection = data[['direction','quant']].groupby(by='direction').count()
all_avg_bydirection = data[['direction','transf_kg','transf_MWK']].groupby(by='direc
print('=================================================================')
print('transfers in vs out: number and average value')
print(all_count_bydirection)
print(all_avg_bydirection)
# in: 612, out:712
# Eliminate barter transfers: transfers were household was supposed to receive or gi
# data['in.return'] = data['in.return'].astype(str)
# data = data[data['in.return']=='nan']   #there were 8 cases that reported return


# Only transfers that we could match in the village
data_match = data.loc[data['id']!=0]
count_bydirection = data_match[['direction','quant']].groupby(by='direction').count(
avg_bydirection = data_match[['direction','quant']].groupby(by='direction').mean()
print('=================================================================')
print('transfers in vs out MATCHED in the village:')
print(count_bydirection )

print('Almost all transfers were matched! 97% of in-transfers matched, 97.6% of out-

data_match_out = data_match.loc[data_match['direction']=='out']
data_match_in = data_match.loc[data_match['direction']=='in']
data_match_in.rename(columns={'hhid':'id','id':'hhid'},inplace=True)



### Only transfers that we can cross-validate directions
data_directions_match = data_match_in.merge(data_match_out, on=['hhid','id'], how='i

### Only transfers that we can match cross-validate directions and the item
data_item_match = data_match_in.merge(data_match_out, on=['hhid','id','good'], how='

print('=================================================================')
print('Number of transfers cross-validated based on direction: coinciding hhid givin
print('num transf:',len(data_directions_match))
### Only transfers that we can match cross-validate directions and the item
data_item_match = data_match_in.merge(data_match_out, on=['hhid','id','good'], how='

print('=================================================================')
print('Number of transfers cross-validated based on direction and food item')
print('num transf:',len(data_item_match))
print('the low number of matched transfers is reasonable in the sense that we are as
print('The number is sligthly higher than in 2019 (119)')



### Import consumption dataset to create base ids
c22 = pd.read_csv('C:/Users/rodri/Dropbox/Malawi/SIEG2021 (1)/2022 July/Data/Clean d
data_final = c22[['hhid']]


# household level net transfers variables ========================

# ----------- variable 1: 'tranfers1_net' -------------- (no exploit network)
#Take only what households reoort to give and receive (not info about what other hou

print('=================================================================')
print('household level net transfers variables')
print('=================================================================')
print('net transfers only includes food transfers. Reasons: (1) most fertlizer trans

data = data.loc[data['good']!='fert']
data_given = data.loc[data['direction']=='out',['hhid','transf_MWK']].groupby(by='hh
data_given.columns = ['transfers1_out']

data_received = data.loc[data['direction']=='in',['hhid','transf_MWK']].groupby(by='
data_received.columns = ['transfers1_in']

transfers1 = data_received.merge(data_given,on='hhid', how='outer')

transfers1['transfers1_net'] = transfers1['transfers1_in'].fillna(0) - transfers1['t
data_final = data_final.merge(transfers1, on='hhid', how='left')

# -----------variable 2: 'tranfers2_net' -------------- (restrictive)
# only those transfers that we could cross-validate---X reports giving food item to
# _x variables denote from direction in. _y variables denote from direction out.

# given the problem of the big span of time across surveys, I'd not use this measure

data_item_match[['id','hhid','good','transf_kg_x','transf_MWK_x','transf_kg_y','tran

data_item_match['transf_MWK_avg'] = data_item_match[['transf_MWK_x','transf_MWK_y']]

data_given = data_item_match[['id','transf_MWK_avg']].groupby(by='id').sum()
data_given.reset_index(inplace=True)
data_given.columns = ['hhid','transfers2_in']

data_received = data_item_match[['hhid','transf_MWK_avg']].groupby(by='hhid').sum()
data_received.reset_index(inplace=True)
data_received.columns = ['hhid','transfers2_out']

transfers2 = data_given.merge(data_received,on='hhid', how='outer')
transfers2['transfers2_net'] = transfers2['transfers2_in'].fillna(0) - transfers2['t


data_final = data_final.merge(transfers2, on='hhid', how='left')


# ----------- variable 3: 'tranfers3_net' -------------- (extensive)
# what household X reports to receive plus what rest of households report to give to
# First, to avoid double-counting, eliminate the transfers that we could cross-valid
transfers3 = data.merge(data_item_match[['hhid','id','good','transf_MWK_avg']],on=['
transfers3 = transfers3[transfers3['transf_MWK_avg'].isnull()]
```

```
# we lose 130 observations. Exactly the number of transfers we chould cross-validate

data_given1 = transfers3.loc[transfers3['direction']=='out',['hhid','transf_MWK']]
data_given2 = transfers3.loc[transfers3['direction']=='in',['id','transf_MWK']]
data_given2.columns = ['hhid','transf_MWK']

data_given= data_given1.append(data_given2)
data_given= data_given.groupby(by='hhid').sum()
data_given.columns = ['transfers3_out']

data_received1 = transfers3.loc[transfers3['direction']=='in',['hhid','transf_MWK']]
data_received2 = transfers3.loc[transfers3['direction']=='out',['id','transf_MWK']]
data_received2.columns = ['hhid','transf_MWK']

data_received= data_received1.append(data_received2)
data_received= data_received.groupby(by='hhid').sum()
data_received.columns = ['transfers3_in']

transfers3 = data_received.merge(data_given,on='hhid', how='outer')
transfers3['transfers3_net'] = transfers3['transfers3_in'].fillna(0) - transfers3['t



data_final = data_final.merge(transfers3, on='hhid', how='left')
data_final.set_index('hhid', inplace=True)
data_final = data_final[['transfers1_net','transfers2_net','transfers3_net']]
data_final_year = data_final*4*12


data_final.to_csv('hhtransfers_week_22.csv')
data_final_year.to_csv('hhtransfers_year_22.csv')


print('================================================================')
print('saved datasets net food transfers hh level: hhtransfers_week_22.csv, hhtransf

data_final_dollars = data_final/dollar_MWK
print('================================================================')
print(data_final_dollars.describe(percentiles=[0.05,0.25,0.5,0.75,0.9,0.95]))


## transfers as gifts or as an exchange?
print('================================================================')
print('Comment: transfers as gifts or as an exchange? ')
print('================================================================')
print('93.29% of the food transfers reported nothing was given/received back for the
print('a common concern or criticism we have received is that, perhaps, these transf

print('Note: the number was computed with the data Albert first clean on transfers.
```

```
for the food items I use the median consumption prices in the village
for fertilizer I used the mean price reported in the agricultural production section
================================================================
Summary transfers in the village
================================================================
count   2,068.00
mean        1.34
std        13.53
min         0.00
10%         0.02
50%         0.20
90%         1.42
max       356.38
Name: transf_dollar, dtype: float64
================================================================
Summary only FOOD transfers
================================================================
       transf_kg  transf_dollar
count   1,972.00       1,972.00
mean        1.97           1.12
std        26.32          13.76
min         0.00           0.00
10%         0.02           0.02
50%         0.43           0.19
90%         2.18           0.83
max     1,080.00         356.38
count       2087
unique        43
top       thobwa
freq         259
Name: good, dtype: object
================================================================
Summary only fertilizer
================================================================
       transf_kg  transf_dollar  recipro.cashback
count      96.00          96.00             59.00
mean       20.81           5.91          8,055.93
std        18.75           5.32          6,875.59
min         1.00           0.28          1,500.00
10%         5.00           1.42          2,500.00
50%        15.00           4.26          7,000.00
90%        50.00          14.19         15,000.00
max       110.00          31.22         46,000.00
Proportion fert transfers household had to pay back:  0.61
Comparison value given median price vs payback for those hhs that reported to pay ba
ck for fertilizer transfer
       transf_kg  transf_MWK  recipro.cashback
count      59.00       59.00             59.00
mean       24.68    7,216.02          8,055.93
std        20.30    5,935.02          6,875.59
min         5.00    1,462.04          1,500.00
10%         5.00    1,462.04          2,500.00
50%        20.00    5,848.15          7,000.00
90%        50.00   14,620.37         15,000.00
max       110.00   32,164.81         46,000.00
There are some large discrepancies but avg is similar
food+fertilizer transfers data in long format with values to kgs and MWK saved: Phas
e 3/transfers/transfers_in_kg_MWK.csv
================================================================
transfers in vs out: number and average value
           quant
direction
in           720
out         1271
           transf_kg  transf_MWK
direction
in              4.14    1,546.60
out             2.06    1,289.00
================================================================
transfers in vs out MATCHED in the village:
           quant
direction
in           705
out         1244
Almost all transfers were matched! 97% of in-transfers matched, 97.6% of out-transfe
rs matched. booklet work very well. Enumerators did great job
================================================================
Number of transfers cross-validated based on direction: coinciding hhid giving with
hhid from who you received---and equvilently for receiving.
num transf: 1269
================================================================
Number of transfers cross-validated based on direction and food item
num transf: 132
the low number of matched transfers is reasonable in the sense that we are asking fo
```

```
r last 7 days while the span of the surveys was 2 months
The number is sligthly higher than in 2019 (119)
===============================================================
household level net transfers variables
===============================================================
net transfers only includes food transfers. Reasons: (1) most fertlizer transfers hh
actually payed (2) consistent with previous data (3) timing is different
===============================================================
saved datasets net food transfers hh level: hhtransfers_week_22.csv, hhtransfers_yea
r_22.csv
===============================================================
       transfers1_net  transfers2_net  transfers3_net
count          263.00          118.00          271.00
mean            -3.59            0.00           -1.62
std             31.05            3.46           47.95
min           -336.87          -14.19         -358.56
5%              -9.42           -4.86          -11.46
25%             -1.31           -0.34           -1.35
50%             -0.06            0.04            0.36
75%              0.88            0.41            2.69
90%              4.32            1.70            6.30
95%              6.60            3.62           10.47
max             20.31           14.11          341.05
===============================================================
Comment: transfers as gifts or as an exchange?
===============================================================
93.29% of the food transfers reported nothing was given/received back for the transf
er. Neither give/receive back the food, or another food item,  ganyu labor or other
potential exchanges.
a common concern or criticism we have received is that, perhaps, these transfers hav
e nothing to do with insurance or other motives, but they are just exchanges of good
s in an economy where there is little use of cash. Our results reject this hypothesi
s. Whatever is going on with the transfers in the village, it is not direct exchange
s as in a barter economy.
Note: the number was computed with the data Albert first clean on transfers. check p
df Data/Summaries/old_foodtransfers_22_summary
```

```
In [ ]:
```