Productivity Index - Output

Emily Markowitz (Emily.Markowitz@noaa.gov) and Sun-Ling Wang

Feb. 05, 2020

Contents

1 Math Theory: General Total Factor Productivity (TFP) Equation

The general form of the TFP can be measured as aggregate output (Y) divided by real total inputs (X). Rates of TFP growth are constructed using the Törnqvist index approach. The TFP growth over two time periods is defined as:

$$ln(TFP_t/TFP_{t-1}) = \sum_{i=1}^{n}((\frac{R_{i,t}+R_{i,t-1}}{2}) * ln(\frac{Y_{i,t}}{Y_{i,t-1}}))) - \sum_{j=1}^{m}((\frac{W_{j,t}+W_{j,t-1}}{2}) * ln(\frac{X_{j,t}}{X_{j,t-1}})))$$

Such that:

- Output $= \sum_{i=1}^{n}((\frac{R_{it}+R_{it-1}}{2}) * ln(\frac{Y_{it}}{Y_{it-1}}))$
- Input $= \sum_{j=1}^{n}((\frac{W_{jt}+W_{jt-1}}{2}) * ln(\frac{X_{jt}}{X_{jt-1}}))$

where:

- Y_i are individual outputs. This will later be refered to as Q_i in the following equations.
- X_j are individual inputs
- R_i are output revenue shares
- W_j are input cost shares
- t and $t-1$ are time subscripts, where 1 is the minimum year in the dataset
- i is category, e.g., Finfish (=1), Shellfish (=2)
- s is species, e.g., Salmon, Alewife, Surf Clams

---

2 Output Method: From Price to Quantity Measures

2.0.1 Variable Summary

Variables

- Q are individual quantity outputs in pounds (lbs).
- V are individual value outputs in dollars ($)
- R are output revenue shares
- P are prices
- PC are price changes
- PI are price indicies, often defined by a price from a base year *baseyr*
- *baseyr* is the year to base all indicides from

Inidicies

- t and $t-1$ are time subscripts, where 1 is the minimum year in the dataset
- i is category, e.g., Finfish (=1), Shellfish (=2)
- s is species, e.g., Salmon, Alewife, Surf Clams

2.0.2 Data requirements

We need time series data for the value of all species (V_t; e.g., Total), value of all species in a category (i) ($V_{i=1}$; e.g., Finfish), value of each species in a category (i) ($V_{i=1,s=n}$; e.g., Salmon and Summer Flounder), quanity of all species in a category (i) (in lbs, $Q_{i=1}$; e.g., Finfish and others), and the quantity of each species in a category (i) ($Q_{i=1,s=n}$; e.g., Salmon and Flounder):

2.0.2.1 Edit Data

Here we summate the category and total V because there may be instances where these values may not be the sum of their parts (though they are here). The caluclation Price Index aims to deal with this potiental issue.

	Q1_1Salmon	Q1_2Cod	Q2_1Shrimp	Q2_2Clam	Q1_3Flounder	Q1_4SeaBass	QE0_0Total	QE1_0Fi
2007	NA	2000	100	150	NA	1000	3250	
2008	NA	1900	120	160	NA	1200	3380	
2009	NA	2000	110	140	NA	900	3150	

	Q1__1Salmon	Q1__2Cod	Q2__1Shrimp	Q2__2Clam	Q1__3Flounder	Q1__4SeaBass	QE0__0Total	QE1__0Fi
2010	20	2500	90	NA	NA	NA	2610	
2011	10	2400	80	NA	NA	NA	2490	
2012	12	2300	100	NA	NA	NA	2412	
2013	11	2000	100	140	NA	1000	3251	
2014	11	2300	110	110	NA	900	3431	
2015	10	2400	90	130	NA	1000	3630	
2016	15	2200	100	160	NA	1100	3575	

2.0.2.2 The nameing conventions of the column names.

For example, in "V1__0Finfish":

- "V"… refers to the variable represented in the column (here V = "Value")
- …"1"… refers to the category index (here, = Finfish)
- …"__"… is simply a seperator in the title
- Since this is the total, …"0".. refers to the index of the species, which is not relevant since this is the sum of the category, hense = 0
- …"Finfish" is purely descriptive (here the name of the category), so you can follow along with what is happening!

Similarly for "Q2__2Clam":

- "Q"… refers to the variable represented in the column (here Q = "Quantity")
- …"2"… refers to the category index (here, = Shellfish)
- …"__"… is simply a seperator in the title
- …"2".. refers to the index of the species, such that this organism happens to be the second species of this category.
- …"Clams" is purely descriptive (here the name of the species), so you can follow along with what is happening!

We can do the structuring work in a function

This funciton standardizes the length of the category or species numbers e.g.,(numbers of 33, 440, and 1 are converted to 033, 440, and 001)

2.0.3 Lets get started

In most of the following examples, we will just focus on the finfish (i=1) side of the equation.

```
ii<-1 #The category index value
warnings.list<-list() #save issues
baseyr<-2010
```

Here I am just going to do some housekeeping:

```
    #If data are missing by the below percentage, remove data
    PercentMissingThreshold<-0.50



  NumberOfSpecies<-numbers0(x = c(0, strsplit(x =
                                              strsplit(x = names(temp)[1],
```

```
                                                      split = "_")[[1]][2],
                                             split = "[a-zA-Z]")[[1]][1]))[1]

  NameBaseTotal<-substr(x = sort(names(temp)[grep(x = names(temp),
                                                  pattern = "0Total")], decreasing = T)[1],
                        start = 2, stop = nchar(sort(names(temp)[grep(x = names(temp),
                                                                pattern = "0Total")], decreasing =

  VColumns<-grep(pattern = paste0("V", ii,"_"),
                 x = substr(x = names(temp),
                            start = 1,
                            stop = (2+nchar(ii))))

NameBasecategory<-substr(start = 2,
                         stop = nchar(names(temp)[VColumns[(grepl(
                           pattern = paste0("V", ii,"_",
                                            numbers0(x = c(0, length(VColumns)-1))[1]),
                           x = names(temp)[VColumns]))]]),
                         x = names(temp)[VColumns[(grepl(
                           pattern = paste0("V", ii,"_",
                                            numbers0(x = c(0, length(VColumns)-1))[1]),
                           x = names(temp)[VColumns]))]])

VColumns<-VColumns[!(grepl(pattern = paste0("V", ii,"_",
                                            numbers0(x = c(0, length(VColumns)-1))[1]),
                           x = names(temp)[VColumns]))]
```

2.0.4 Remove any V and Q data where V column has less data than the specifed *percentmissingthreshold*

	Q1_1Salmon	Q1_2Cod	Q2_1Shrimp	Q2_2Clam	REMOVED_Q1_3Flounder	Q1_4SeaBass
2007	NA	2000	100	150	NA	1000
2008	NA	1900	120	160	NA	1200
2009	NA	2000	110	140	NA	900
2010	20	2500	90	NA	NA	NA
2011	10	2400	80	NA	NA	NA
2012	12	2300	100	NA	NA	NA
2013	11	2000	100	140	NA	1000
2014	11	2300	110	110	NA	900
2015	10	2400	90	130	NA	1000
2016	15	2200	100	160	NA	1100

2.0.5 Caluclate Catagory Sums of V and Q

Because we removed some columns for not meeting a perecent missing threshold of 0.5% and those columns will not be used at all in any part of the further analysis, we need to re-calculate the totals of V and Q for the catagories and the fishery as a whole.

	REMOVED_QE1_0Finfish	REMOVED_V1_0Finfish	QE1_0Finfish	V1_0Finfish
2007	3000	3800	3000	2800

	REMOVED_QE1_0Finfish	REMOVED_V1_0Finfish	QE1_0Finfish	V1_0Finfish
2008	3100	4020	3100	2820
2009	2900	3910	2900	3010
2010	2520	3190	2520	3190
2011	2410	3280	2410	3280
2012	2312	3150	2312	3150
2013	3011	4080	3011	3080
2014	3211	4270	3211	3370
2015	3410	4700	3410	3700
2016	3315	4480	3315	3380

2.0.6 Price for each species ($P_{s,i,t}$; e.g., Salmon and Flounder)

We first measure output price for each species in each of the categories (e.g., Finfish & Others and Shellfish) using detailed landings time series data on value ($) and pounds (lbs).

Price for a species (s) of category (i) in year (t) =

$$P_{s,i,t} = V_{s,i,t}/Q_{s,i,t}$$

where:

- $P_{s,i,t}$ is the price per individual species (s), category (i), for each year (t)
- $Q_{s,i,t}$ is the quantity (lb) per individual species (s), category (i), for each year (t)
- $V_{i,t}$ is the value ($) per category (i), for each year (t)

Here we calculate the price for each species

```
    # Find which columns in this table are price Columns - we will need this for later
    PColumns<-paste0("P", substr(x = VColumns,
                                start = 2,
                                stop = nchar(VColumns)))

#####Price for each species#####
tempP<-data.frame(data = rep_len(x = NA, length.out = nrow(temp)))
for (c in 1:length(VColumns)) {

    NameBase<-substr(start = 2,
                    stop = nchar(VColumns[c]),
                    x = VColumns[c])

    Q0<-temp[,names(temp) %in% paste0("Q", NameBase)]
    V0<-temp[,names(temp) %in% paste0("V", NameBase)] #to make sure its the same column
  tempP[,c]<-V0/Q0
  names(tempP)[c]<-paste0("P", NameBase ) #name the column
}

  tempP<-as.matrix(tempP)
  tempP[tempP %in% Inf]<-NA
  tempP<-data.frame(tempP)
  temp<-cbind.data.frame(temp, data.frame(tempP))
```

	P1_1Salmon	P1_2Cod	P1_4SeaBass
1	NA	1.400000	NA
2	NA	1.421053	0.1000000
3	NA	1.450000	0.1222222
4	5.00000	1.200000	NA
5	10.00000	1.291667	NA
6	12.50000	1.260870	NA
7	16.36364	1.400000	0.1000000
8	15.45455	1.391304	NA
9	20.00000	1.458333	NA
10	12.00000	1.454546	NA

There may be instances where price cannot (or should not) be calculated because there is no or too few Q or V data for that species in a year or ever. The next goal will be to calculate the price change, so we need to have a value in there that won't show change. If we left a 0 in the spot, then the price change from 0 to the next year would be huge and misrepresented on the index. To avoid this, we have to deal with four senarios:

2.0.6.1 1. If there are instances for a species where there are too few pairs of V and/or Q are completely missing from the timeseries or where a percent of V is missing from the timeseries, we will remove the offending price columns entierly, so they don't influence the downstream price change or price index calculations.

Let's say here that if 50% of the data is missing in a given $V_{s,i,t}$, don't calculate that species $P_{s,i,t}$

```
#Find which columns in this table are price Columns

cc<-c() #Empty
for (c in 1:length(VColumns)) {

  #If price could never be caluclated at any point in the timeseries (is 0/NaN/NA) for a column (c)
  #Remove the column from the analysis.
  #We will not be removing the column from the data, but simply remove it from the varaible "PColumns"
  if (#sum(temp[,PColumns[c]] %in% c(0, NA, NaN)) %in% nrow(temp) |
      sum(temp[,PColumns[c]] %in% c(0, NA, NaN))/nrow(temp) > PercentMissingThreshold) {
    cc<-c(cc, c)#Collect offending columns
  }

}

if (length(cc)>0){
  PColumns<-PColumns[-cc]
  # VColumns<-VColumns[-cc]
  # QColumns<-QColumns[-cc]
}
```

	P1_1Salmon	P1_2Cod
2007	NA	1.400000
2008	NA	1.421053
2009	NA	1.450000
2010	5.00000	1.200000
2011	10.00000	1.291667
2012	12.50000	1.260870

	P1_1Salmon	P1_2Cod
2013	16.36364	1.400000
2014	15.45455	1.391304
2015	20.00000	1.458333
2016	12.00000	1.454546

$$where \begin{cases} if : P_{i,t=1} = 0, then : P_{i,t=1} = P_{i,t=1+1...} \\ if : P_{i,t\neq 1} = 0, then : P_{i,t} = P_{i,t-1} \end{cases}$$

2.0.6.2 2. If the first value of $P_{i,t,s}$ is 0/NA in a timeseries, we let the next available non-zero/non-NA value of P in the timeseries inform the past.

We use this function:

```
print(ReplaceFirst)
```

```
## function(colnames, temp) {
##   for (c in 1:length(colnames)) {
##
##     #If the first value of the timeseries of this column (c) is 0/NaN/NA
##     #Change the first value (and subsequent 0/NaN/NA values) to the first available non-0/NaN/NA valu
##     if (temp[1,colnames[c]] %in% c(0, NA, NaN)) {
##       findfirstvalue<-temp[which(!(temp[,colnames[c]]  %in% c(0, NA, NaN))),
##                            colnames[c]][1]
##       temp[1,colnames[c]]<-findfirstvalue
##     }
##   }
##   return(temp)
## }
## <bytecode: 0x00000000037978a0>
```

```
temp<-ReplaceFirst(colnames = PColumns, temp)
```

	P1_1Salmon	P1_2Cod
2007	5.00000	1.400000
2008	NA	1.421053
2009	NA	1.450000
2010	5.00000	1.200000
2011	10.00000	1.291667
2012	12.50000	1.260870
2013	16.36364	1.400000
2014	15.45455	1.391304
2015	20.00000	1.458333
2016	12.00000	1.454546

2.0.6.3 3. If there is a value in the middle of $P_{i,t,s}$'s timeseries that is 0/NA, we let the most recent past available non-zero/non-NA of $P_{i,t,s}$ in the timeseries inform the future.

```
print(ReplaceMid)
```

```
## function(colnames, temp) {
```

```
##   for (c in 1:length(colnames)) {
##     #If a middle value of the timeseries of this column (c) is 0/NaN/NA
##     #Change the currently 0/NaN/NA value to the previous available non-0/NaN/NA value
##     if (sum(temp[,colnames[c]] %in% c(0, NA, NaN))>0) {
##       troublenumber<-which(temp[,colnames[c]] %in% c(0, NA, NaN))
##       for (r in 1:length(troublenumber)){
##         findlastvalue<-temp[troublenumber[r]-1, colnames[c]][1]
##         temp[troublenumber[r],colnames[c]]<-findlastvalue
##       }
##     }
##   }
##   return(temp)
## }
## <bytecode: 0x00000000693d6c00>
```

```
temp<-ReplaceMid(colnames = PColumns, temp)
```

	P1_1Salmon	P1_2Cod
2007	5.00000	1.400000
2008	5.00000	1.421053
2009	5.00000	1.450000
2010	5.00000	1.200000
2011	10.00000	1.291667
2012	12.50000	1.260870
2013	16.36364	1.400000
2014	15.45455	1.391304
2015	20.00000	1.458333
2016	12.00000	1.454546

2.0.7 Fill in values of $V_{i,t,s}$ where P was able to be calculated

To ensure that the price index does not rise or fall to quickly with changes (that are really because of NA values) we fill in the missing instances of $V_{i,t,s}$.

$$where \begin{cases} if : V_{i,t=1} = 0, then : V_{i,t=1} = V_{i,t=1+1...} \\ if : V_{i,t\neq 1} = 0, then : V_{i,t} = V_{i,t-1} \end{cases}$$

2.0.7.1 1. If the first value of $V_{i,t,s}$ is 0/NA in a timeseries, we let the next available non-zero value of $V_{i,t,s}$ in the timeseries inform the past.

```
VVColumns<-paste0("V", substr(x = PColumns, start = 2, stop = nchar(PColumns)))
temp<-ReplaceFirst(colnames = VVColumns, temp)
```

	V1_1Salmon	V1_2Cod
2007	100	2800
2008	NA	2700
2009	NA	2900
2010	100	3000
2011	100	3100
2012	150	2900
2013	180	2800

	V1__1Salmon	V1__2Cod
2014	170	3200
2015	200	3500
2016	180	3200

2.0.7.2 2. If there is a value in the middle of $V_{i,t,s}$'s timeseries that is 0/NA, we let the most recent past available non-zero of $V_{i,t,s}$ in the timeseries inform the future.

```
temp<-ReplaceMid(colnames = VVColumns, temp)
```

	V1__1Salmon	V1__2Cod
2007	100	2800
2008	100	2700
2009	100	2900
2010	100	3000
2011	100	3100
2012	150	2900
2013	180	2800
2014	170	3200
2015	200	3500
2016	180	3200

2.0.8 Value of species $VV_{i,t}$ where P was able to be calculated

$R_{i,t}$, as defined and discussed in the subsequent step, will need to sum to 1 across all species in a category. Therefore, you will need to sum a new total of $V_{i,t}$ available (called $VV_{i,t}$) for the category using only values for species that were used to calculate $P_{i,t}$ (called $V_{s,i,t,available}$).

$$VV_{i,t} = \sum_{s=1}^{n}(V_{s,i,t,available})$$

where:

- $VV_{i,t}$ is the new total of $V_{i,t}$ (called $VV_{i,t}$) for the category using only values for species that were used to calculate $P_{i,t}$
- $V_{s,i,t,available}$ are the $V_{s,i,t}$ where P were able to be calculated

```
temp0<-data.frame(temp[,names(temp) %in% VVColumns],
                  rowSums(temp[,names(temp) %in% VVColumns], na.rm = T))
names(temp0)[ncol(temp0)]<-paste0("VV",NameBasecategory)
temp0<-data.frame(temp0)
temp[ncol(temp)+1]<-temp0[ncol(temp0)]

temp0 %>%
    knitr::kable(row.names = T, booktabs = T)
```

	V1__1Salmon	V1__2Cod	VV1__0Finfish
2007	100	2800	2900
2008	100	2700	2800
2009	100	2900	3000

	V1_1Salmon	V1_2Cod	VV1_0Finfish
2010	100	3000	3100
2011	100	3100	3200
2012	150	2900	3050
2013	180	2800	2980
2014	170	3200	3370
2015	200	3500	3700
2016	180	3200	3380

2.0.9 Revenue Share for each species ($R_{s,i,t}$; e.g., Salmon and Flounder)

$$R_{s,i,t} = V_{s,i,t}/VV_{i,t}$$

where:

- $R_{s,i,t}$ is the revenue share per individual species (s), category (i), for each year (t)
- $V_{s,i,t}$ is the value ($) per individual species (s), category (i), for each year (t)

Here we divide $V_{s,i,t}$ by $VV_{i,t}$ because $VV_{i,t}$ only includes species used to calculate $V_{s,i,t}$ as per the above price calculations.

	R1_1Salmon	R1_2Cod
1	0.0344828	0.9655172
2	0.0357143	0.9642857
3	0.0333333	0.9666667
4	0.0322581	0.9677419
5	0.0312500	0.9687500
6	0.0491803	0.9508197
7	0.0604027	0.9395973
8	0.0504451	0.9495549
9	0.0540541	0.9459459
10	0.0532544	0.9467456

As an additional check, let's make sure that each row sums to 1.

	x
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1

2.0.10 Price Changes for each species ($PC_{s,i,t}$ aka $\Delta ln(P_{s,i,t})$; e.g., Salmon and Flounder)

$$PC_{i,t} = ln(\frac{P_{i,t}}{P_{i,t-1}}) = \sum_{s=1}^{n}([\frac{R_{s,i,t}+R_{s,i,t-1}}{2}]*[ln(\frac{P_{s,i,t}}{P_{s,i,t-1}}]) = \sum_{s=1}^{n}([\frac{R_{s,i,t}+R_{s,i,t-1}}{2}]*[ln(P_{s,i,t})-ln(P_{s,i,t-1})])$$

Such that:

category's (i) Price Change $= ln(\frac{P_{i,t}}{P_{i,t-1}})$

category's (i) Price Change for each species (s) $= \frac{R_{s,i,t}+R_{s,i,t-1}}{2}$

category's (i) Revenue Share for each species (s) $= ln(\frac{P_{s,i,t}}{P_{s,i,t-1}}$

Which can be adapted to this function/macro:

```
#A function to caluclate the price change
# print(PriceChange)
```

Now put it into practice for the total dataset:

```
#Find which columns in this table are price and revenue share columns
tempPC<-data.frame(data = rep_len(x = NA, length.out = nrow(temp)))
for (c in 1:length(PColumns)){
  #For nameing columns
    NameBase<-substr(start = 2,
                     stop = nchar(PColumns[c]),
                     x = PColumns[c])

  # Calculate
  P0<-temp[, names(temp) %in% paste0("P", NameBase)]
  R0<-temp[, names(temp) %in% paste0("R", NameBase)] #to make sure its the same column
  tempPC[,c]<-PriceChange(R0, P0)
  names(tempPC)[c]<-paste0("PC", NameBase ) #name the column
}

temp<-cbind.data.frame(temp, tempPC)
temp[ncol(temp)+1]<-rowSums(tempPC, na.rm = T)
names(temp)[ncol(temp)]<-paste0("PC", NameBasecategory)
```

For reference, here are the Price Changes for each species ($PC_{s,i,t}$):

	PC1_1Salmon	PC1_2Cod
2007	0.0000000	0.0000000
2008	0.0000000	0.0144018
2009	0.0000000	0.0194695
2010	0.0000000	-0.1830357
2011	0.0220102	0.0712743
2012	0.0089738	-0.0231613
2013	0.0147572	0.0989356
2014	-0.0031679	-0.0058852
2015	0.0134715	0.0445941
2016	-0.0274080	-0.0024612

And here is the summed ($\sum$) Price Change for the category:

	Other…	R1_1Salmon	R1_2Cod	PC1_1Salmon	PC1_2Cod	PC1_0Finfish
2007	…	0.0344828	0.9655172	0.0000000	0.0000000	0.0000000
2008	…	0.0357143	0.9642857	0.0000000	0.0144018	0.0144018
2009	…	0.0333333	0.9666667	0.0000000	0.0194695	0.0194695
2010	…	0.0322581	0.9677419	0.0000000	-0.1830357	-0.1830357
2011	…	0.0312500	0.9687500	0.0220102	0.0712743	0.0932846
2012	…	0.0491803	0.9508197	0.0089738	-0.0231613	-0.0141875
2013	…	0.0604027	0.9395973	0.0147572	0.0989356	0.1136927
2014	…	0.0504451	0.9495549	-0.0031679	-0.0058852	-0.0090532
2015	…	0.0540541	0.9459459	0.0134715	0.0445941	0.0580655
2016	…	0.0532544	0.9467456	-0.0274080	-0.0024612	-0.0298692

2.0.11 Price Index for the each category (PI_t)

We calculate the price index first by comparing by multiplying the previous years PI_{t-1} by that year's price change PC_t, where the PI of the first year $PI_{t=firstyear} = 1$

$$PI_t = PI_{t-1} * exp(ln(\frac{P_{i,t}}{P_{i,t-1}})) = PI_{t-1} * exp(PC_t)$$

Where

$$PI_{i,t_{firstyear}} = 1$$

```
#Note that the first row of this column is = 1
tempPI1<-data.frame(c(1, rep_len(x = NA, length.out = nrow(temp)-1)))
rownames(tempPI1)<-rownames(temp)

PC0<-temp[,names(temp) %in% paste0("PC", NameBasecategory)] #this is equal to ln(P_it/P_it-1)

  # Calculate
for (t in 2:length(tempPI1)){  #Since the first row is defined, we need to start at the second row
    tempPI1[t]<-tempPI1[t-1]*exp(PC0[t])
}
```

Then, to change the price (calulated later) into base year dollars, we use the following equation:

$$PI_t = PI_t/PI_{t=baseyear}$$

In this example, we'll decide that the base year is 2010, for whatever reason. Notice that the $PI_{i,t=baseyr} = 1$

```
tempPI2<-tempPI1/tempPI1[rownames(tempPI1) %in% baseyr,]
```

	tempPI1	tempPI2
2007	1.014506	NA
2008	NA	NA
2009	NA	NA
2010	NA	NA
2011	NA	NA
2012	NA	NA
2013	NA	NA

	tempPI1	tempPI2
2014	NA	NA
2015	NA	NA
2016	NA	NA

Which can be summarized in this function:

```
print(PriceIndex)
```

```
## function(temp, BaseColName, baseyr) {
##   ###Price Index for the entire commercial fishery ($PI_t$)
##
##   # We calculate the price index first by comparing by multiplying the previous years $PI_{t-1}$ by t
##   # $$PI_t = PI_{t-1}*exp(ln(\frac{P_{i,t}}{P_{i,t-1}})) = PI_{t-1}*exp(PC_{t})$$
##   # Where
##   # $$PI_{i, t_{first year}} = 1$$
##
##   #Note that the first row of this column is = 1
##   tempPI1<-c(1, rep_len(x = NA, length.out = nrow(temp)-1))
##
##   PC0<-temp[,names(temp) %in% paste0("PC", BaseColName)] #this is equal to ln(P_it/P_it-1)
##
##   # Calculate
##   for (t in 2:length(tempPI1)){  #Since the first row is defined, we need to start at the second row
##     tempPI1[t]<-tempPI1[t-1]*exp(PC0[t])
##   }
##
##   tempPI1<-data.frame(tempPI1)
##   rownames(tempPI1)<-rownames(temp)
##
##   # Then, to change the price (calulated later) into base year dollars, we use the following equation
##   # $$PI_{t} = PI_{t}/PI_{t = baseyear}$$
##   # In this example, we'll decide that the base year is `r baseyr`, for whatever reason. Notice that
##
##   tempPI2<-tempPI1/tempPI1[rownames(tempPI1) %in% baseyr,]
##
##   tempPI<-data.frame(tempPI2)
##   names(tempPI)<-paste0("PI", BaseColName)
##
##   return(tempPI)
## }
## <bytecode: 0x00000000599200e0>
```

And we add the PI to the data

```
tempPI<-PriceIndex(temp, BaseColName = NameBasecategory, baseyr)
temp[ncol(temp)+1]<-(tempPI)
names(temp)[ncol(temp)]<-paste0("PI", NameBasecategory)
```

	PI1_0Finfish
2007	1.160864
2008	1.177703
2009	1.200857
2010	1.000000

	PI1_0Finfish
2011	1.097774
2012	1.082309
2013	1.212628
2014	1.201699
2015	1.273542
2016	1.236065

2.0.12 Implicit Quantity/Output for each category ($Q_{i,t}$; Finfish & others and Shellfish)

Note here that all columns of V are being used, despite having been removed earlier in the analysis when PI could not be calculated and PI columns have functionally been removed from the analysis.

$$Q_{i,t} = V_{i,t}/PI_{i,t}$$

```
temp[,ncol(temp)+1]<-temp[,names(temp) %in% paste0("V", NameBasecategory)]/
  temp[,names(temp) %in% paste0("PI", NameBasecategory)]

names(temp)[ncol(temp)]<-paste0("Q", NameBasecategory)
```

	Other…	PC1_1Salmon	PC1_2Cod	PC1_0Finfish	PI1_0Finfish	Q1_0Finfish
2007	…	0.0000000	0.0000000	0.0000000	1.160864	2411.997
2008	…	0.0000000	0.0144018	0.0144018	1.177703	2394.491
2009	…	0.0000000	0.0194695	0.0194695	1.200857	2506.543
2010	…	0.0000000	-0.1830357	-0.1830357	1.000000	3190.000
2011	…	0.0220102	0.0712743	0.0932846	1.097774	2987.864
2012	…	0.0089738	-0.0231613	-0.0141875	1.082309	2910.443
2013	…	0.0147572	0.0989356	0.1136927	1.212628	2539.938
2014	…	-0.0031679	-0.0058852	-0.0090532	1.201699	2804.362
2015	…	0.0134715	0.0445941	0.0580655	1.273542	2905.283
2016	…	-0.0274080	-0.0024612	-0.0298692	1.236065	2734.484

2.0.13 Analysis Warnings Checks

2.0.13.1 1. When back calculated, V_t did not equal $P_t * Q_t$

$$V_i = PI_t * Q_i$$

```
  temp0<-temp[names(temp) %in% c(paste0("Q",NameBasecategory),
                                 paste0("PI",NameBasecategory),
                                 paste0("V",NameBasecategory))]

  temp0[,(ncol(temp0)+1)]<-temp0[,paste0("Q",NameBasecategory)]*temp0[,paste0("PI",NameBasecategory)]
  names(temp0)[ncol(temp0)]<-paste0("V", NameBasecategory, "_Check")

  if (length(setdiff(temp0[,paste0("V", NameBasecategory, "_Check")],
                     temp0[,paste0("V", NameBasecategory)])) == 0) {
    warnings.list[length(warnings.list)+1]<-"When back calculated, V_{i,t} did not equal PI_{i,t} * Q_{i
```

```
    print("When back calculated, V_{i,t} did not equal PI_{i,t} * Q_{i,t}")
  }
```

```
## [1] "When back calculated, V_{i,t} did not equal PI_{i,t} * Q_{i,t}"
```

	V1_0Finfish	PI1_0Finfish	Q1_0Finfish	V1_0Finfish_Check
2007	2800	1.160864	2411.997	2800
2008	2820	1.177703	2394.491	2820
2009	3010	1.200857	2506.543	3010
2010	3190	1.000000	3190.000	3190
2011	3280	1.097774	2987.864	3280
2012	3150	1.082309	2910.443	3150
2013	3080	1.212628	2539.938	3080
2014	3370	1.201699	2804.362	3370
2015	3700	1.273542	2905.283	3700
2016	3380	1.236065	2734.484	3380

2.0.13.2 2. When back calculated, Q_t did not equal V_t/PI_t

$$Q_{i,t} = V_t/PI_{i,t}$$

```
  temp0[,(ncol(temp0)+1)]<-temp0[,paste0("V",NameBasecategory)]/temp0[,paste0("PI",NameBasecategory)]
  names(temp0)[ncol(temp0)]<-paste0("Q", NameBasecategory, "_Check")

  if (length(setdiff(temp0[,paste0("Q", NameBasecategory, "_Check")],
                     temp0[,paste0("Q", NameBasecategory)])) == 0) {
    warnings.list[length(warnings.list)+1]<-"When back calculated, Q_{i,t} did not equal V_{i,t}/PI_{i,t
    print("When back calculated, Q_{i,t} did not equal V_{i,t}/PI_{i,t}")
  }
```

```
## [1] "When back calculated, Q_{i,t} did not equal V_{i,t}/PI_{i,t}"
```

	V1_0Finfish	PI1_0Finfish	Q1_0Finfish	V1_0Finfish_Check	Q1_0Finfish_Check
2007	2800	1.160864	2411.997	2800	2411.997
2008	2820	1.177703	2394.491	2820	2394.491
2009	3010	1.200857	2506.543	3010	2506.543
2010	3190	1.000000	3190.000	3190	3190.000
2011	3280	1.097774	2987.864	3280	2987.864
2012	3150	1.082309	2910.443	3150	2910.443
2013	3080	1.212628	2539.938	3080	2539.938
2014	3370	1.201699	2804.362	3370	2804.362
2015	3700	1.273542	2905.283	3700	2905.283
2016	3380	1.236065	2734.484	3380	2734.484

2.1 Redo Analysis for Shellfish

Pretending that we also did all of that work we just did for FinFish and Others for Shellfish and two species of shellfish, I'll use a function called "species.cat.level" that I will print out for you after this example:

```r
ii<-2 #The category index value

tempS<-species.cat.level(temp, ii, baseyr, maxyr, minyr,
                          PercentMissingThreshold = PercentMissingThreshold,
                          warnings.list = warnings.list)
temp<-cbind.data.frame(temp, tempS[[1]])
warnings.list<-tempS[[2]]
###Remove duplicate columns
temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
```

What does the Shellfish data look like?

	VV2__0Shellfish	R2__1Shrimp	R2__2Clam	PC2__0Shellfish	PI2__0Shellfish	Q2__0Shellfish
2007	1800	0.4444444	0.5555556	0.0000000	1.030337	1747.0009
2008	2200	0.4545455	0.5454545	0.0831894	1.119717	1964.7830
2009	1800	0.5000000	0.5000000	-0.0893363	1.024023	1757.7726
2010	1600	0.4375000	0.5625000	-0.0237392	1.000000	700.0000
2011	1800	0.5000000	0.5000000	0.1730144	1.188883	757.0129
2012	1900	0.5263158	0.4736842	-0.0604413	1.119154	893.5320
2013	2200	0.5454545	0.4545455	0.1466028	1.295862	1697.7119
2014	2000	0.5500000	0.4500000	-0.0384432	1.246990	1603.8619
2015	2000	0.5000000	0.5000000	0.0260098	1.279850	1562.6836
2016	2300	0.5217391	0.4782609	-0.0156266	1.260005	1825.3889

2.1.1 Value for all fisheries for species where P was able to be calculated

$R_{i,t}$, defined and discussed in the subsequent step, will need to sum to 1 across all species in a category. Therefore, you will need to sum a new total of $V_{i,t}$ (called VV_t) for the category using only values for species that were used to calculate $PI_{i,t}$.

$$VV_t = \sum_{s=1}^{n}(VV_{i,t})$$

where:

- VV_t is the new total of $V_{i,t}$ for the entire fishery using only values for species that were used to calculate $P_{i,t}$

VV1	_0Finfish VV2	_0Shellfish VV0	_0Total
2007	2900	1800	4700
2008	2800	2200	5000
2009	3000	1800	4800
2010	3100	1600	4700
2011	3200	1800	5000
2012	3050	1900	4950
2013	2980	2200	5180
2014	3370	2000	5370
2015	3700	2000	5700
2016	3380	2300	5680

2.1.2 Revenue Share for the entire commercial fishery (R_t)

$$R_{i,t} = V_{i,t}/V_t$$

where:

- $R_{i,t}$ is the revenue share per individual species (s), category (i), for each year (t)
- $V_{i,t}$ is the value ($) per individual species (s), category (i), for each year (t)

Here, we don't use VV_t beacause we want to expand the proportion to include all of the species caught, regardless if they were used in the price calculations.

```
names(temp)[names(temp) %in% paste0("V", NameBaseTotal)]<-paste0("REMOVED_V", NameBaseTotal)

temp0<-temp[grep(x = names(temp),
                              pattern = paste0("V[1-9]+_", NumberOfSpecies))]
temp0<-temp0[,-(grep(x = names(temp0), pattern = c("VV")))]
temp0<-temp0[,-(grep(x = names(temp0), pattern = c("REMOVED_")))]

temp[ncol(temp)+1]<-rowSums(temp0, na.rm = T)
names(temp)[ncol(temp)]<-paste0("V", NameBaseTotal)

#remove duplicates
  temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
  temp <- temp[, !duplicated(colnames(temp))]

# temp$R1_0Finfish<-temp$VV1_0Finfish/temp$VV0_0Total
# temp$R2_0Shellfish<-temp$VV2_0Shellfish/temp$VV0_0Total

temp$R1_0Finfish<-temp$V1_0Finfish/temp$V0_0Total
temp$R2_0Shellfish<-temp$V2_0Shellfish/temp$V0_0Total
```

	R1_0Finfish	R2_0Shellfish	V1_0Finfish	V2_0Shellfish	V0_0Total
2007	0.6086957	0.3913043	2800	1800	4600
2008	0.5617530	0.4382470	2820	2200	5020
2009	0.6257796	0.3742204	3010	1800	4810
2010	0.8200514	0.1799486	3190	700	3890
2011	0.7846890	0.2153110	3280	900	4180
2012	0.7590361	0.2409639	3150	1000	4150
2013	0.5833333	0.4166667	3080	2200	5280
2014	0.6275605	0.3724395	3370	2000	5370
2015	0.6491228	0.3508772	3700	2000	5700
2016	0.5950704	0.4049296	3380	2300	5680

As an additional check, let's make sure that each row sums to 1.

	x
2007	1
2008	1
2009	1
2010	1
2011	1
2012	1

	x
2013	1
2014	1
2015	1
2016	1

2.1.3 Price Changes for the entire commercial fishery (PC_t)

Measure output price changes (PC_t) for total output (Q_t) using $R_{i,t}$ and $P_{i,t}$ estimates.

$$PC_t = ln(\frac{P_t}{P_{t-1}}) = \sum_{i=1}^{n}([\frac{R_{i,t} + R_{i,t-1}}{2}] * [ln(P_{i,t}) - ln(P_{i,t-1})])$$

```
temp$PC0_0Total<-rowSums(cbind(PriceChange(R0 = temp$R1_0Finfish, P0 = temp$PI1_0Finfish),
                                PriceChange(R0 = temp$R2_0Shellfish, P0 = temp$PI2_0Shellfish)),
                          na.rm = T)
```

	Other…	VV0_0Total	V0_0Total	R1_0Finfish	R2_0Shellfish	PC0_0Total
2007	…	4700	4600	0.6086957	0.3913043	0.0000000
2008	…	5000	5020	0.5617530	0.4382470	0.0429332
2009	…	4800	4810	0.6257796	0.3742204	-0.0247311
2010	…	4700	3890	0.8200514	0.1799486	-0.1388971
2011	…	5000	4180	0.7846890	0.2153110	0.1090416
2012	…	4950	4150	0.7590361	0.2409639	-0.0247398
2013	…	5180	5280	0.5833333	0.4166667	0.1245141
2014	…	5370	5370	0.6275605	0.3724395	-0.0206491
2015	…	5700	5700	0.6491228	0.3508772	0.0464723
2016	…	5680	5680	0.5950704	0.4049296	-0.0244869

2.1.4 Price Index for the entire commercial fishery (PI_t)

We calculate the price index first by comparing by multiplying the previous years PI_{t-1} by that year's price change PC_t, where the PI of the first year $PI_{t=firstyear} = 1$

$$PI_t = PI_{t-1} * exp(ln(\frac{P_{i,t}}{P_{i,t-1}})) = PI_{t-1} * exp(PC_t)$$

Where

$$PI_{i,t_{firstyear}} = 1$$

```
tempPI<-PriceIndex(temp, BaseColName = NameBaseTotal, baseyr)
temp[ncol(temp)+1]<-(tempPI)
names(temp)[ncol(temp)]<-paste0("PI", NameBaseTotal)
```

	PI0_0Total
2007	1.128281
2008	1.177776
2009	1.149006

	PI0_0Total
2010	1.000000
2011	1.115209
2012	1.087957
2013	1.232218
2014	1.207035
2015	1.264452
2016	1.233866

2.1.5 Total Implicit Quantity/Output for the entire commercial fishery ($Q_t = Y_t$)

To get quantity estimates for total output using total value of landings divided by price index as follow: $Y = Q = V/I$

$$Q_t = V_t/PI_t$$

```
temp$Q0_0Total<-temp$V0_0Total/temp$PI0_0Total
```

	x
1	4077.000
2	4262.269
3	4186.228
4	3890.000
5	3748.177
6	3814.488
7	4284.956
8	4448.919
9	4507.880
10	4603.418

2.1.6 Total Implicit Quantity/Output Index

$$QI_t = Q_t/Q_{t=baseyr}$$

Where:

- QI is the sum of Q after these equations

```
temp$QI0_0Total<-temp$Q0_0Total/temp$Q0_0Total[rownames(temp) %in% baseyr]
```

	Other…	R2_0Shellfish	PC0_0Total	PI0_0Total	Q0_0Total	QI0_0Total
2007	…	0.3913043	0.0000000	1.128281	4077.000	1.0480719
2008	…	0.4382470	0.0429332	1.177776	4262.269	1.0956991
2009	…	0.3742204	-0.0247311	1.149006	4186.228	1.0761510
2010	…	0.1799486	-0.1388971	1.000000	3890.000	1.0000000
2011	…	0.2153110	0.1090416	1.115209	3748.177	0.9635417
2012	…	0.2409639	-0.0247398	1.087957	3814.488	0.9805883
2013	…	0.4166667	0.1245141	1.232218	4284.956	1.1015311
2014	…	0.3724395	-0.0206491	1.207035	4448.919	1.1436810
2015	…	0.3508772	0.0464723	1.264452	4507.880	1.1588382

	Other...	R2_0Shellfish	PC0_0Total	PI0_0Total	Q0_0Total	QI0_0Total
2016	...	0.4049296	-0.0244869	1.233866	4603.418	1.1833979

2.1.7 Sum Total Implicit Quantity Output Index (Optional)

$$QEI_t = QE_t/QE_{t=baseyr}$$

Where:

- QE is the sum of Q before these equations
- QEI is the index of the sum of Q before these equations

```
temp$QEI0_0Total<-temp$QE0_0Total/temp$QE0_0Total[rownames(temp) %in% baseyr]
```

	Other...	PC0_0Total	PI0_0Total	Q0_0Total	QI0_0Total	QEI0_0Total
2007	...	0.0000000	1.128281	4077.000	1.0480719	1.2452107
2008	...	0.0429332	1.177776	4262.269	1.0956991	1.2950192
2009	...	-0.0247311	1.149006	4186.228	1.0761510	1.2068966
2010	...	-0.1388971	1.000000	3890.000	1.0000000	1.0000000
2011	...	0.1090416	1.115209	3748.177	0.9635417	0.9540230
2012	...	-0.0247398	1.087957	3814.488	0.9805883	0.9241379
2013	...	0.1245141	1.232218	4284.956	1.1015311	1.2455939
2014	...	-0.0206491	1.207035	4448.919	1.1436810	1.3145594
2015	...	0.0464723	1.264452	4507.880	1.1588382	1.3908046
2016	...	-0.0244869	1.233866	4603.418	1.1833979	1.3697318

2.1.8 Solve Output portion of the equation for the Output Changes:

$$QC_t = \sum_{i=1}^{n}((\frac{R_{it} + R_{it-1}}{2}) * ln(\frac{Q_{it}}{Q_{it-1}}))$$

```
temp$QC0_0Total<-rowSums(cbind(PriceChange(R0 = temp$R1_0Finfish, P0 = temp$Q1_0Finfish),
                               PriceChange(R0 = temp$R2_0Shellfish, P0 = temp$Q2_0Shellfish)),
                         na.rm = T)
```

	Q0_0Total	QI0_0Total	QC0_0Total
2007	4077.000	1.0480719	0.0000000
2008	4262.269	1.0956991	0.0444654
2009	4186.228	1.0761510	-0.0180726
2010	3890.000	1.0000000	-0.0808110
2011	3748.177	0.9635417	-0.0370504
2012	3814.488	0.9805883	0.0175616
2013	4284.956	1.1015311	0.1196593
2014	4448.919	1.1436810	0.0375242
2015	4507.880	1.1588382	0.0131616
2016	4603.418	1.1833979	0.0210303

2.2 Other Analysis Warnings Checks

To make sure our analyses worked as inteded, let's see if we can back calculate our numbers.

We want the calcuated V to equal this check:

2.2.0.1 1. When back calculated, V_t did not equal $PI_t * Q_t$?

$$V_i = P_t * Q_i$$

```
temp0<-temp[names(temp) %in% c(paste0("Q",NameBaseTotal),
                               paste0("PI",NameBaseTotal),
                                      paste0("V",NameBaseTotal))]

temp0[,(ncol(temp0)+1)]<-temp0[,paste0("Q",NameBaseTotal)]*temp0[,paste0("PI",NameBaseTotal)]
names(temp0)[ncol(temp0)]<-paste0("V", NameBaseTotal, "_Check")

if (length(setdiff(temp0[,paste0("V", NameBaseTotal, "_Check")],
                   temp0[,paste0("V", NameBaseTotal)])) == 0) {
warnings.list[length(warnings.list)+1]<-"When back calculated, V_t did not equal PI_t * Q_t"
print("When back calculated, V_t did not equal PI_t * Q_t")
}
```

	V0_0Total	PI0_0Total	Q0_0Total	V0_0Total_Check
2007	4600	1.128281	4077.000	4600
2008	5020	1.177776	4262.269	5020
2009	4810	1.149006	4186.228	4810
2010	3890	1.000000	3890.000	3890
2011	4180	1.115209	3748.177	4180
2012	4150	1.087957	3814.488	4150
2013	5280	1.232218	4284.956	5280
2014	5370	1.207035	4448.919	5370
2015	5700	1.264452	4507.880	5700
2016	5680	1.233866	4603.418	5680

2.2.0.2 2. When back calculated, Q_t did not equal V_t/P_t?

$$Q_{i,t} = V_t/P_{i,t}$$

```
temp0[,(ncol(temp0)+1)]<-temp0[,paste0("V",NameBaseTotal)]/temp0[,paste0("PI",NameBaseTotal)]
names(temp0)[ncol(temp0)]<-paste0("Q", NameBaseTotal, "_Check")

if (length(setdiff(temp0[,paste0("Q", NameBaseTotal, "_Check")],
                   temp0[,paste0("Q", NameBaseTotal)])) == 0) {
warnings.list[length(warnings.list)+1]<-"When back calculated, Q_t did not equal V_t/PI_t"
print("When back calculated, Q_t did not equal V_t/PI_t")
}
```

```
## [1] "When back calculated, Q_t did not equal V_t/PI_t"
```

	V0_0Total	PI0_0Total	Q0_0Total	V0_0Total_Check	Q0_0Total_Check
2007	4600	1.128281	4077.000	4600	4077.000
2008	5020	1.177776	4262.269	5020	4262.269
2009	4810	1.149006	4186.228	4810	4186.228
2010	3890	1.000000	3890.000	3890	3890.000
2011	4180	1.115209	3748.177	4180	3748.177
2012	4150	1.087957	3814.488	4150	3814.488
2013	5280	1.232218	4284.956	5280	4284.956
2014	5370	1.207035	4448.919	5370	4448.919
2015	5700	1.264452	4507.880	5700	4507.880
2016	5680	1.233866	4603.418	5680	4603.418

2.2.0.3 3. When back calculated, growth rate?

$$ln(Q_t/Q_{t-1}) = \sum((\frac{R_{i,t} - R_{i,t-1}}{2}) * ln(\frac{Q_{i,t}}{Q_{i,t-1}}))$$

```
names0<-c(paste0("Q",NameBaseTotal))
for (i in 1:ii) {
  names0<-c(names0,
            # names(temp)[grep(pattern = paste0("QE", i, "_", NumberOfSpecies), names(temp))
            #   [!(grep(pattern = paste0("QE", i, "_", NumberOfSpecies), names(temp))) %in%
            #               grep(pattern = paste0("REMOVED_"), names(temp))] ],
            names(temp)[grep(pattern = paste0("Q", i, "_", NumberOfSpecies), names(temp))],
            names(temp)[grep(pattern = paste0("R", i, "_", NumberOfSpecies), names(temp))])
}

temp0<-temp[,names0]

temp0[,(ncol(temp0)+1)]<-c(NA, ln(temp0[-nrow(temp0),paste0("Q",NameBaseTotal)]/
                              temp0[-1,paste0("Q",NameBaseTotal)]))
names(temp0)[ncol(temp0)]<-"part1"

temp00<-data.frame()
for (i in 1:(ii)) {
  R0<-temp0[,grep(pattern = paste0("R", i), x = names(temp0))]
  Q0<-temp0[,grep(pattern = paste0("Q", i), x = names(temp0))]

  for (r in 2:(nrow(temp))){
    temp00[r,i]<-(((R0[r]-R0[r-1])/2) * ln(Q0[r] / Q0[r-1]) )
  }
}

temp0[,(ncol(temp0)+1)]<-rowSums(temp00)
names(temp0)[ncol(temp0)]<-"part2"

  if (length(setdiff(temp0[,"part1"],
                     temp0[,"part2"])) == 0) {
  warnings.list[length(warnings.list)+1]<-"When back calculated, ln(Q_t/Q_{t-1}) = did not equal sum( (
  print("When back calculated, ln(Q_t/Q_{t-1}) = did not equal sum( ( \frac{R_{i, t} - R_{i, t-1}}{2})
  }
```

	Q0_0Total	Q1_0Finfish	R1_0Finfish	Q2_0Shellfish	R2_0Shellfish	part1	part2
2007	4077.000	2411.997	0.6086957	1747.0009	0.3913043	NA	NA
2008	4262.269	2394.491	0.5617530	1964.7830	0.4382470	-0.0444404	0.0029284
2009	4186.228	2506.543	0.6257796	1757.7726	0.3742204	0.0180017	0.0050283
2010	3890.000	3190.000	0.8200514	700.0000	0.1799486	0.0733908	0.1128563
2011	3748.177	2987.864	0.7846890	757.0129	0.2153110	0.0371395	0.0025419
2012	3814.488	2910.443	0.7590361	893.5320	0.2409639	-0.0175368	0.0024634
2013	4284.956	2539.938	0.5833333	1697.7119	0.4166667	-0.1163037	0.0683502
2014	4448.919	2804.362	0.6275605	1603.8619	0.3724395	-0.0375509	0.0034476
2015	4507.880	2905.283	0.6491228	1562.6836	0.3508772	-0.0131659	0.0006616
2016	4603.418	2734.484	0.5950704	1825.3889	0.4049296	-0.0209719	0.0058370

2.2.0.4 4. Missing Data

```
## [1] "Out of 5 columns, 0 of species V columns are completely empty, 1 of species Q columns are compl
```

2.2.0.5 5. Removed Data

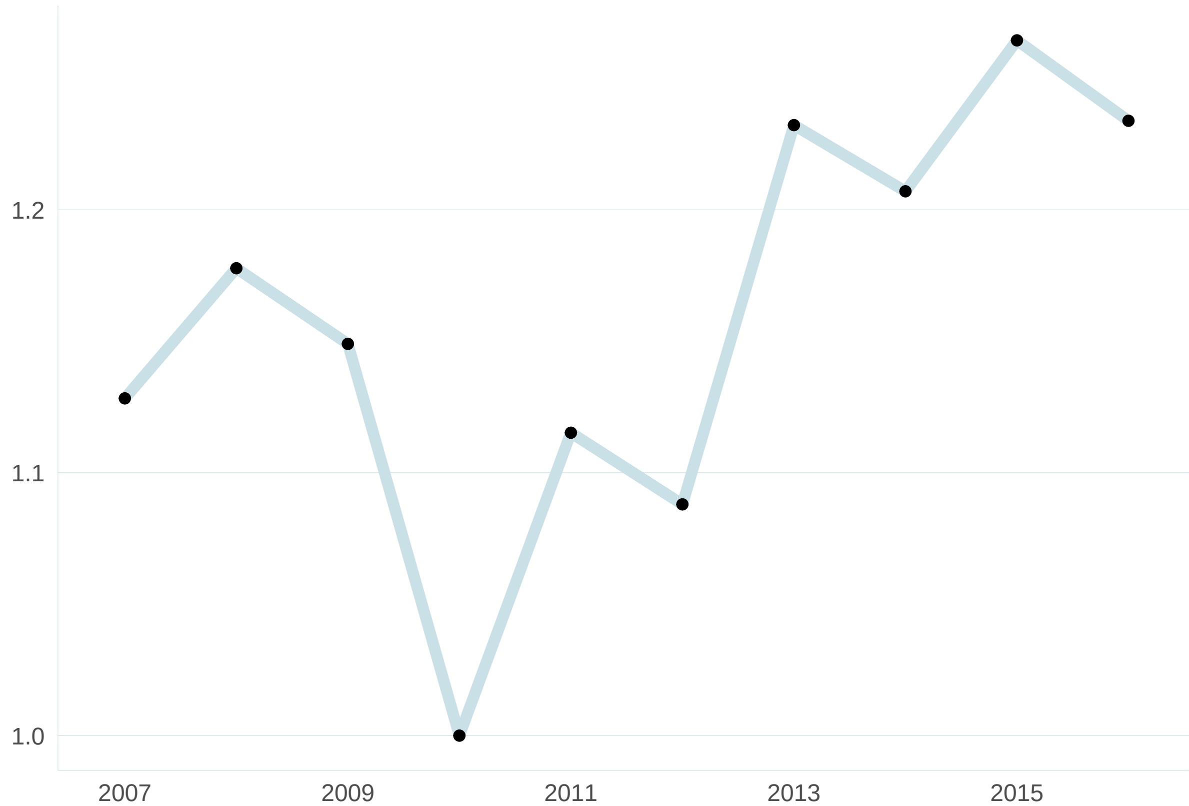
```
## [1] "Out of 6 columns, 1 of species V columns were removed, 1 of species Q columns were removed, and
```

	Q0_0Total	Q1_0Finfish	R1_0Finfish	Q2_0Shellfish	R2_0Shellfish	part1	part2
2007	4077.000	2411.997	0.6086957	1747.0009	0.3913043	NA	NA
2008	4262.269	2394.491	0.5617530	1964.7830	0.4382470	-0.0444404	0.0029284
2009	4186.228	2506.543	0.6257796	1757.7726	0.3742204	0.0180017	0.0050283
2010	3890.000	3190.000	0.8200514	700.0000	0.1799486	0.0733908	0.1128563
2011	3748.177	2987.864	0.7846890	757.0129	0.2153110	0.0371395	0.0025419
2012	3814.488	2910.443	0.7590361	893.5320	0.2409639	-0.0175368	0.0024634
2013	4284.956	2539.938	0.5833333	1697.7119	0.4166667	-0.1163037	0.0683502
2014	4448.919	2804.362	0.6275605	1603.8619	0.3724395	-0.0375509	0.0034476
2015	4507.880	2905.283	0.6491228	1562.6836	0.3508772	-0.0131659	0.0006616
2016	4603.418	2734.484	0.5950704	1825.3889	0.4049296	-0.0209719	0.0058370

2.2.1 Graph 1: Price Index

In theory, PI should be negative slope after the baseyear and positive after the base year, but because this data was fabricated without thinking of this, we don't see that here. The index value for the base year is =1, however.

```
# A function I made to plot this pretty in ggplot2
temp <- temp[, !duplicated(colnames(temp))]
plot1line(temp, PI = temp$PI0_0Total,
          NOAALightBlue, NOAADarkBlue, NOAADarkGrey)
```



2.2.2 Graph 2: Quantity Index Compare

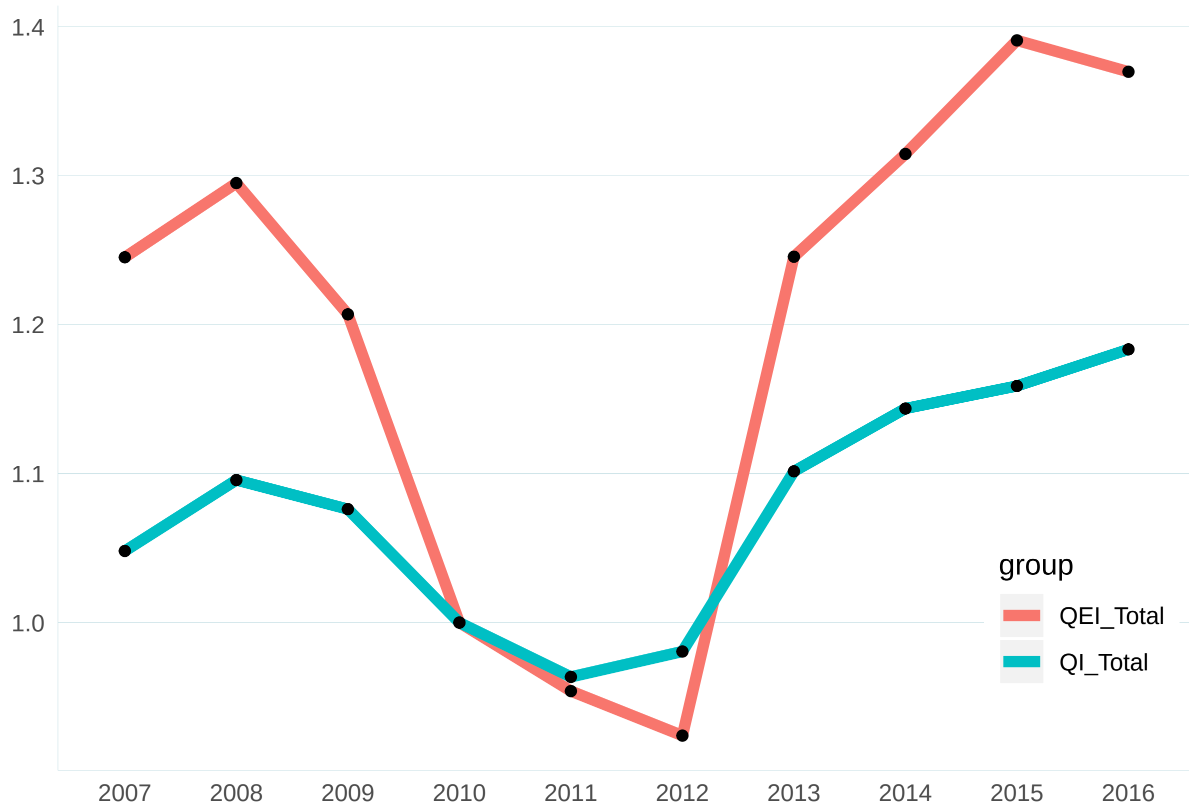
```
temp0<-temp
temp0$Year<-rownames(temp0)

tempA<-data.frame(temp0[,names(temp0) %in% c("Year", "QI0_0Total")])
names(tempA)<-c("Index", "Year")
tempA$group<-"QI_Total"
tempA$col<-NOAALightBlue

tempB<-data.frame(temp0[,names(temp0) %in% c("Year", "QEI0_0Total")])
names(tempB)<-c("Index", "Year")
tempB$group<-"QEI_Total"
tempB$col<-NOAADarkBlue

temp0<-rbind.data.frame(tempA, tempB)
rownames(temp0)<-NULL
temp0$col<-as.factor(temp0$col)

#A function I made to plot this pretty in ggplot2
plot2line(temp0, Year = temp0$Year, Index=temp0$Index, col = temp0$col, group = temp0$group,
                    NOAALightBlue, NOAADarkBlue, NOAADarkGrey)
```



2.2.3 Graph 3: Quantity Compare

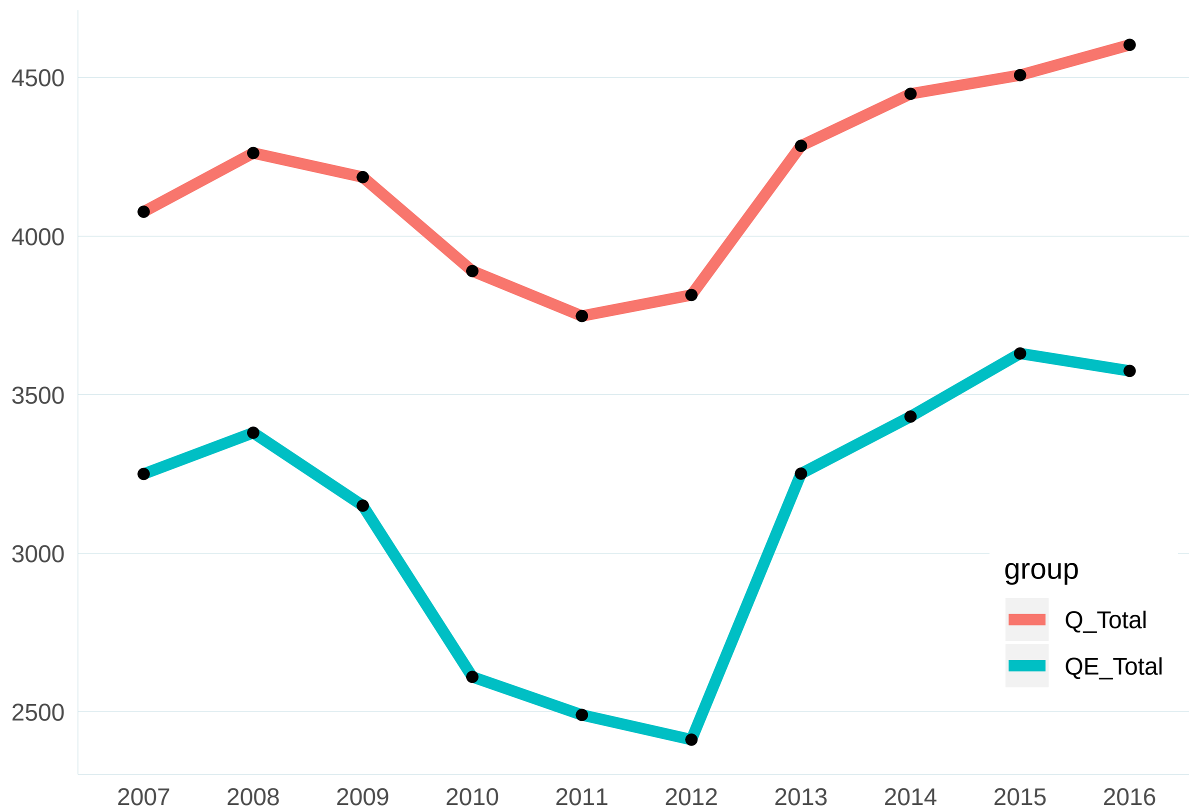
```
temp0<-temp
temp0$Year<-rownames(temp0)

tempA<-data.frame(temp0[,names(temp0) %in% c("Year", "Q0_0Total")])
names(tempA)<-c("Quantity", "Year")
tempA$group<-"Q_Total"
tempA$col<-NOAALightBlue

tempB<-data.frame(temp0[,names(temp0) %in% c("Year", "QE0_0Total")])
names(tempB)<-c("Quantity", "Year")
tempB$group<-"QE_Total"
tempB$col<-NOAADarkBlue

temp0<-rbind.data.frame(tempA, tempB)
rownames(temp0)<-NULL
temp0$col<-as.factor(temp0$col)

#A function I made to plot this pretty in ggplot2
plot2line(temp0, Year = temp0$Year, Index=temp0$Quantity, col = temp0$col, group = temp0$group,
                    NOAALightBlue, NOAADarkBlue, NOAADarkGrey)
```



2.3 Do same analysis via a function!

Now that we know the method, we can simplify most of it into a function and do this whole analysis in 4 easy steps:

- A. Import and Edit data
- B. Enter base year
- C. Run the function
- D. Obtain the implicit quantity estimates

2.3.1 Function to calculate the Implicit Quanity Output at Species and category Level

```
# print(species.cat.level)
```

2.3.2 Function to calculate the Implicit Quanity Output at Fishery Level

2.3.3 A. Import and Edit data

2.3.4 B. Enter base year

2.3.5 C. Run the function

2.3.6 D. Obtain the implicit quantity estimates

Did all of the analyses work as intended?

```
print(warnings.list0)
```

```
## [[1]]
## [1] "When back calculated, V_{i,t} did not equal PI_{i,t} * Q_{i,t}"
##
## [[2]]
## [1] "When back calculated, Q_{i,t} did not equal V_{i,t}/PI_{i,t}"
##
## [[3]]
## [1] "When back calculated, Q_{i,t} did not equal V_{i,t}/PI_{i,t}"
##
## [[4]]
## [1] "When back calculated, Q_t did not equal V_t/PI_t"
##
## [[5]]
## [1] "Out of 6 columns, 0 of species V columns are completely empty, 1 of species Q columns are compl
```

2.3.7 E. Graph

2.3.7.1 Graph 1: Price Index

For comparison, let's recreate those graphs to make sure we are getting the same output:

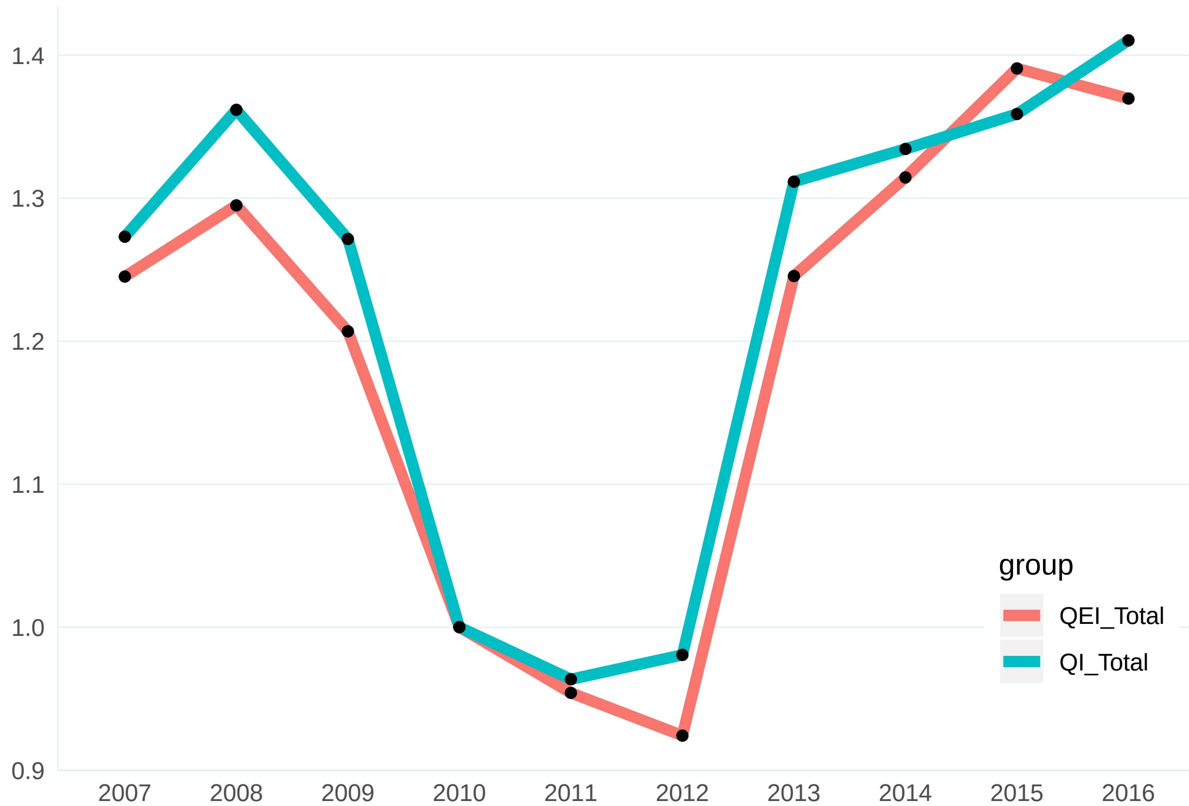
```
figures.list0$`_PI_Line`
```

```
## NULL
```

2.3.7.2 Graph 2: Quantity Index Compare

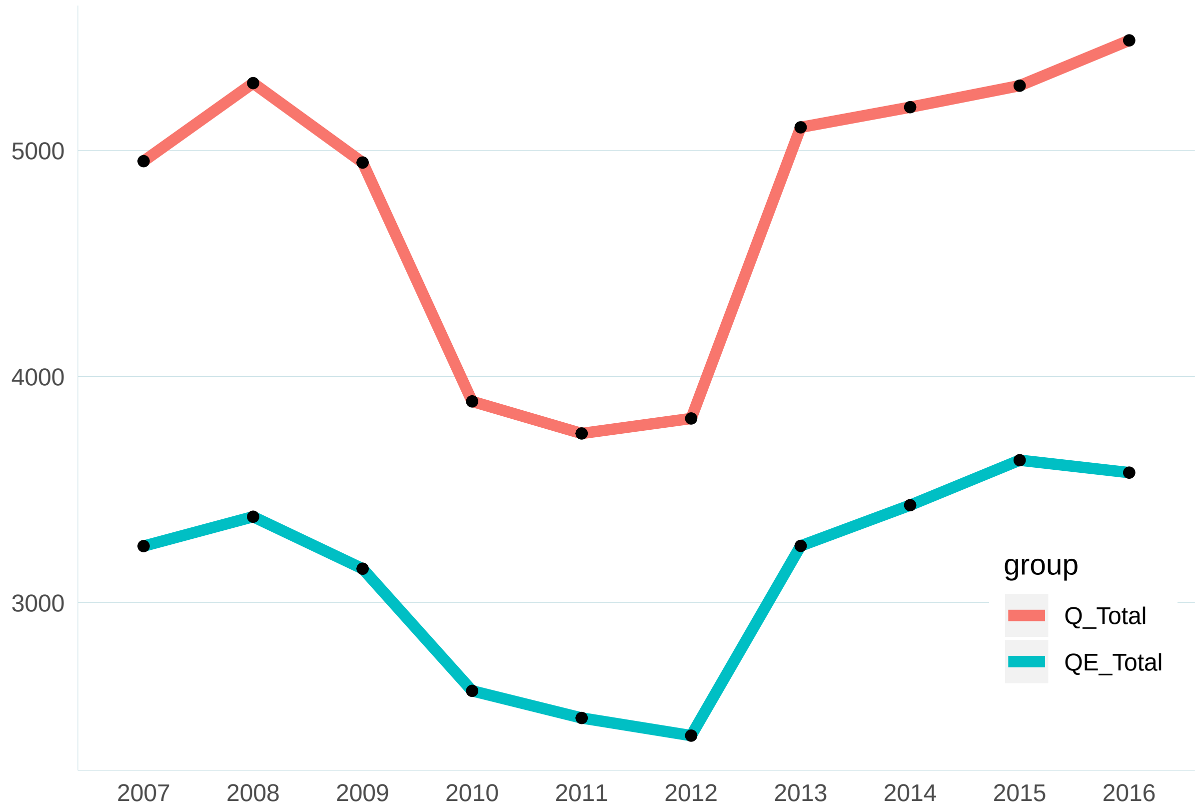
For comparison, let's recreate those graphs to make sure we are getting the same output:

```
figures.list0$`_QuantityIndexCompare`
```



2.3.7.3 Graph 3: Quantity Compare

```
figures.list0$`_QuantityCompare`
```



2.4 Practice with real data (For National Data)

2.4.1 A. Import and Edit data

Load and subset Data

Summary information about the commercial dataset:

```
summary(landings.data[,c("Tsn", "Year", "State", "AFS.Name", "Pounds", "Dollars", "category.orig")])
```

```
##       Tsn              Year                 State                               AFS.Name          Pounds
##  Min.   :     0   Min.   :1950   West Florida   :10405   FINFISH **                 :  1467   Min.   :
##  1st Qu.:160845   1st Qu.:1977   East Florida   : 8973   OYSTER, EASTERN            :  1187   1st Qu.:
##  Median :167674   Median :1995   New York       : 7106   SHARKS, UNCLASSIFIED **:  1169   Median :
##  Mean   :164501   Mean   :1991   California     : 6899   BLUEFISH                   :  1103   Mean   :
##  3rd Qu.:169611   3rd Qu.:2008   North Carolina: 6436   SHAD, AMERICAN             :  1083   3rd Qu.:
##  Max.   :775091   Max.   :2017   New Jersey     : 5642   SQUIDS **                  :  1027   Max.   :3410
##  NA's   :98                      (Other)        :63642   (Other)                    :102067   NA's   :118
##     Dollars           category.orig
##  Min.   :     -4494   Finfish  :82734
##  1st Qu.:      1739   Other    : 5683
##  Median :     21213   Shellfish:20686
##  Mean   :   1659774
##  3rd Qu.:    237607
```

```
## Max.   :540962350
## NA's   :12125
```

Edit/Restructure Data

2.4.2 B. Enter base year

2.4.3 C. Run the function

2.4.4 D. Obtain the implicit quantity estimates

	REMOVED_V_Total	VV_Total	V_Total	PC_Total	QC_Total	PI_Total	Q_Total	QI_To
1950	4910008722	5245879816	4908497898	0.0000000	0.0000000	8.6296251	568796190	0.0725
1951	4468280396	4794342982	4466377584	-0.1326319	0.0381229	7.5577170	590969151	0.0754
1952	4454820075	4793747056	4453075858	0.0421644	-0.0451142	7.8831976	564881931	0.0720
1953	4541451462	4888308503	4540150905	-0.0372698	0.0568212	7.5947999	597797302	0.0762
1954	4783727020	5140369062	4782771308	-0.0472818	0.0993119	7.2440611	660233434	0.0842
1955	4864502898	5204421274	4863595098	0.0329414	-0.0164026	7.4866646	649634429	0.0828
1956	5315091181	5666538056	5314358886	-0.0166223	0.1048237	7.3632475	721741177	0.0920
1957	4812006928	5163140746	4811335276	-0.0090667	-0.0904966	7.2967891	659377050	0.0841
1958	4814125255	5142667236	4813710706	-0.0423725	0.0423875	6.9940647	688256533	0.0878
1959	5136276317	5465497637	5135938307	0.0780877	-0.0134340	7.5621053	679167785	0.0866
1960	5014090964	5343929532	5013880662	0.0696323	-0.0937565	8.1074378	618429739	0.0789
1961	5281445600	5609304998	5281061500	-0.0460724	0.0977190	7.7423830	682097685	0.0870
1962	5483719015	5828252598	5483670915	-0.0235377	0.0617528	7.5622734	725135233	0.0925
1963	4940953280	5286945663	4940885680	-0.0131392	-0.0907794	7.4635612	662001096	0.0844
1964	4658198362	5002164745	4658145962	-0.0648392	0.0056940	6.9949849	665926526	0.0849
1965	4883101538	5224509221	4883084238	-0.0654705	0.1129942	6.5516897	745316773	0.0951
1966	4406698497	4749445680	4406594597	-0.0876642	-0.0148455	6.0017961	734212650	0.0936
1967	4134834274	4476942557	4134767374	0.0949944	-0.1586802	6.5998911	626490241	0.0799
1968	4374304496	4711200579	4374214096	-0.0396917	0.0956332	6.3430612	689606167	0.0879
1969	4451508990	4796727573	4451412090	-0.1334375	0.1511842	5.5506998	801955115	0.1023
1970	4937890925	5284412463	4937749680	-0.0894937	0.1932402	5.0755267	972854633	0.1241
1971	5174335253	5522231846	5174217353	0.0030843	0.0441749	5.0912054	1016304978	0.1296
1972	4986532651	5560171110	4986310542	-0.0942653	0.0574779	4.6332070	1076211473	0.1373
1973	4999752596	5571865210	4999369896	-0.4523424	0.4549488	2.9473512	1696224718	0.2164
1974	5142256144	5708930416	5139376036	0.0231999	0.0044713	3.0165287	1703738458	0.2173
1975	5077858104	5647185119	5077574366	0.0914546	-0.1035198	3.3054128	1536139259	0.1960
1976	5585649671	6155108975	5585524140	-0.1184165	0.2137455	2.9362840	1902242506	0.2427
1977	5371655443	5939356150	5371569138	-0.1207570	0.0817414	2.6022796	2064178285	0.2633
1978	6158425762	6496093706	6157094183	-0.0943302	0.2307737	2.3680283	2600093207	0.3317
1979	6468717520	6799479390	6468442787	-0.1125161	0.1618371	2.1160297	3056877134	0.3900
1980	6558742947	6879957142	6558543970	-0.0026218	0.0164200	2.1104892	3107594242	0.3965
1981	6021841354	6403870540	6021797283	-0.0005168	-0.0854977	2.1093987	2854745835	0.3642
1982	6438791754	6820926341	6438787721	0.0020076	0.0647933	2.1136378	3046306063	0.3887
1983	6394874948	6653589709	6394772190	-0.0207164	0.0134750	2.0703012	3088812440	0.3941
1984	6400733217	6652182493	6400720110	-0.0222549	0.0237124	2.0247357	3161262041	0.4033
1985	6327817404	6579440056	6327760091	0.0243648	-0.0356111	2.0746739	3050002296	0.3891
1986	6087762826	6247599778	6087575242	-0.0929944	0.0543056	1.8904399	3220189759	0.4108
1987	6970940970	7127985691	6970738599	-0.0793242	0.2165942	1.7462758	3991774248	0.5093
1988	7345259874	7347571420	7345092864	-0.1617019	0.2164044	1.4855476	4944367223	0.6309
1989	8703012869	8703831791	8700805819	0.0941748	0.0762532	1.6322482	5330565528	0.6801
1990	9763270615	9757305102	9759236532	-0.0337264	0.1486467	1.5781162	6184104978	0.7890

	REMOVED_V_Total	VV_Total	V_Total	PC_Total	QC_Total	PI_Total	Q_Total	QI_To
1991	9591954950	9594907037	9589284690	-0.0070783	-0.0104719	1.5669853	6119575589	0.7808
1992	9910321012	9906052649	9904847441	-0.1425606	0.1749281	1.3587878	7289473430	0.9301
1993	9931205231	9926845179	9927604931	0.1921181	-0.1898201	1.6465976	6029162722	0.7693
1994	10054778439	10054255709	10050347153	-0.0727500	0.0850261	1.5310612	6564301437	0.8376
1995	9630008417	9627786875	9626118805	-0.1179617	0.0746546	1.3607001	7074386712	0.9026
1996	9340724555	9339533995	9336940076	0.0847820	-0.1151980	1.4810946	6304081040	0.8044
1997	9583115360	9579995371	9579311407	-0.0399403	0.0655456	1.4231050	6731275380	0.8589
1998	8960217365	8957006563	8955981563	0.1751826	-0.2423808	1.6955780	5281963744	0.6739
1999	9062894698	9065699732	9059086055	0.0082025	0.0032261	1.7095432	5299126737	0.6761
2000	8833974470	8835528763	8829301770	-0.0451354	0.0193951	1.6340977	5403166400	0.6894
2001	9253069890	9247621688	9249195256	0.0241410	0.0215658	1.6740264	5525118952	0.7050
2002	9208948179	9198719970	9201271724	0.1447947	-0.1499882	1.9348436	4755563486	0.6068
2003	9291777375	9282057328	9284304076	-0.0073960	0.0165712	1.9205864	4834098625	0.6168
2004	9182300008	9232270660	9139955573	-0.2014713	0.1827859	1.5701314	5821140622	0.7427
2005	9109639858	9129244809	9043527465	-0.1120388	0.1004462	1.4037125	6442578111	0.8220
2006	9138912762	9009775463	9068378806	-0.0910082	0.0794313	1.2816038	7075804955	0.9028
2007	8885395234	8790589502	8793721529	-0.0315668	0.0029636	1.2417795	7081548169	0.9036
2008	7882138532	7653089299	7752010705	-0.2377079	0.0962281	0.9790599	7917810377	1.0103
2009	7788358546	7563486182	7617582061	0.1543682	-0.1679189	1.1424851	6667555121	0.8507
2010	8046768286	7763910259	7836994486	-0.1332058	0.1599890	1.0000000	7836994486	1.0000
2011	9426480100	9190352844	9206821949	-0.0774289	0.2504433	0.9254929	9948020601	1.2693
2012	9316801850	9089997736	9101820510	-0.0253465	0.0146835	0.9023296	10087024065	1.2871
2013	9387237694	9211378445	9221764369	-0.0270387	0.0403398	0.8782587	10500054763	1.3398
2014	9101596982	8907330541	8928117767	0.0671839	-0.1001128	0.9392908	9505168892	1.2128
2015	9284445654	9169488737	9163838776	0.0196999	0.0090559	0.9579782	9565810923	1.2205
2016	9311345804	9156939746	9189639557	-0.0473826	0.0459479	0.9136453	10058213112	1.2834
2017	9565233796	9407628445	9430258606	0.1016064	-0.0741503	1.0113576	9324356220	1.1897

Did all of the analyses work as intended?

```
print(warnings.list0)
```

```
## [[1]]
## [1] "When back calculated, Q_{i,t} did not equal V_{i,t}/PI_{i,t}"
##
## [[2]]
## [1] "When back calculated, Q_{i,t} did not equal V_{i,t}/PI_{i,t}"
##
## [[3]]
## [1] "When back calculated, Q_{i,t} did not equal V_{i,t}/PI_{i,t}"
##
## [[4]]
## [1] "When back calculated, V_t did not equal PI_t * Q_t"
##
## [[5]]
## [1] "When back calculated, Q_t did not equal V_t/PI_t"
##
## [[6]]
## [1] "Out of 30 columns, 32 of species V columns are completely empty, 34 of species Q columns are co
```

2.4.5 E. Graph

2.4.5.1 Graph 1: Price Index

For comparison, let's recreate those graphs to make sure we are getting the same output:

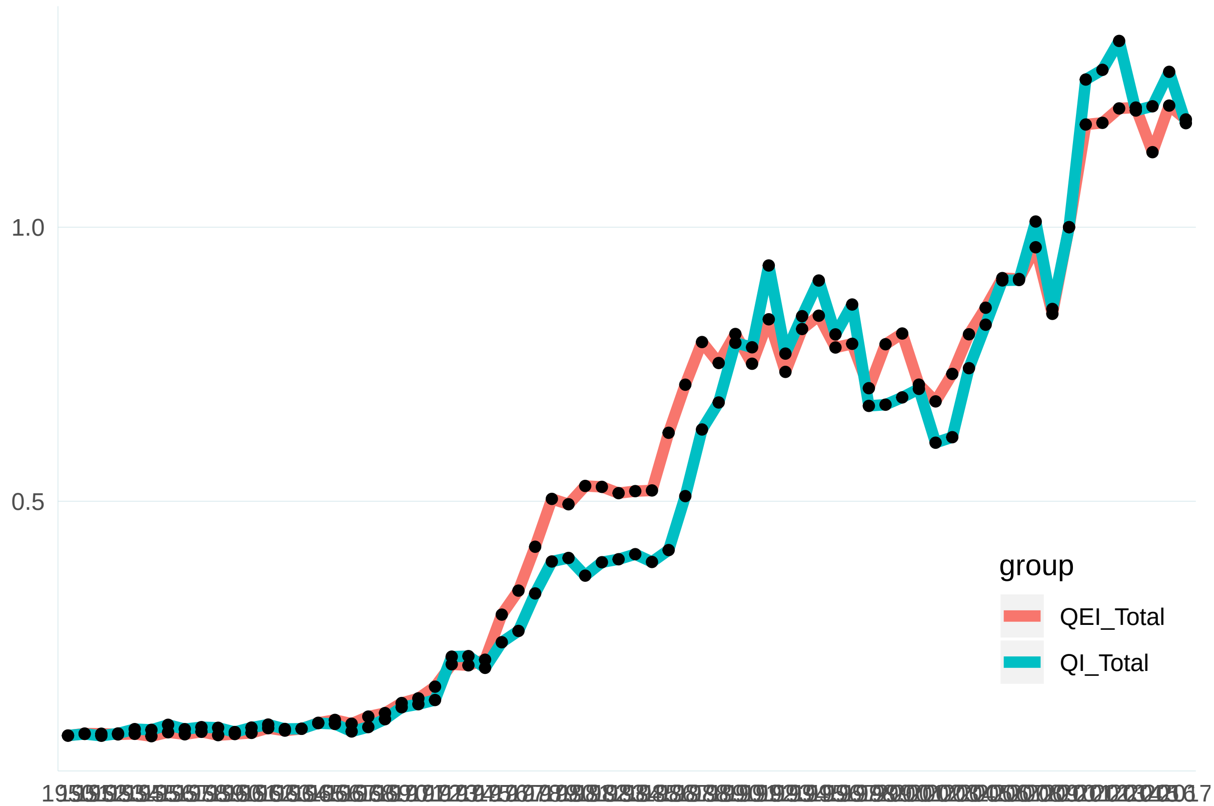
```
figures.list0$`_PI_Line`
```

```
## NULL
```

2.4.5.2 Graph 2: Quantity Index Compare

For comparison, let's recreate those graphs to make sure we are getting the same output:

```
figures.list0$`_QuantityIndexCompare`
```



2.4.5.3 Graph 3: Quantity Compare

```
figures.list0$`_QuantityCompare`
```