Productivity Index - Output

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

Feb. 23, 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:

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

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.

```r
temp<-read.csv(file = paste0(dir.data, "Tornqvist Index-Calculations_OutputEx.csv"))
rownames(temp)<-temp$year
temp$year<-NULL

temp.q<-temp[,grepl(pattern = "Q", x = names(temp))]
temp.q$QE0_0Total<-rowSums(temp.q, na.rm = T)
temp.q$QE1_0Finfish<-rowSums(temp.q[,grepl(x = names(temp.q), pattern = "Q1") ], na.rm = T)
temp.q$QE2_0Shellfish<-rowSums(temp.q[,grepl(x = names(temp.q), pattern = "Q2") ], na.rm = T)

temp.v<-temp[,grepl(pattern = "V", x = names(temp))]
temp.v$V0_0Total<-rowSums(temp.v, na.rm = T)
temp.v$V1_0Finfish<-rowSums(temp.v[,grepl(x = names(temp.v), pattern = "V1") ], na.rm = T)
temp.v$V2_0Shellfish<-rowSums(temp.v[,grepl(x = names(temp.v), pattern = "V2") ], na.rm = T)

temp<-orgional.data<-cbind.data.frame(temp.q, temp.v)
```

	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	
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

```r
ii<-1 #The category index value
baseyr<-2010
pctmiss<-0.50 #If data are missing by the below percentage, remove data
```

In most of the following examples, we will just focus on the finfish (i=1) side of the equation. Here *baseyr* is set to 2010 and the *pctmiss* (The percent of data in a column that we will allow to be missing for analysis; more on that later) is set to 0.5%.

Here I am just going to do some housekeeping:

```r
place <-  "Test"

warnings.list<-list() #save issues we encounter in the code

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 = T)[1]))

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 *pctmiss*

```r
VColumns0<-VColumns
QColumns0<-QColumns<-which(names(temp) %in%
                             paste0("Q", substr(x = names(temp)[VColumns],
                                                start = 2,
                                                stop = nchar(names(temp)[VColumns]))))
for (i in 1:length(VColumns)) {

  #if the percent missing is less in V or Q columns for a species than the percentmissingtrheshold, we
  if (sum(is.na(temp[VColumns[i]]))/nrow(temp) > pctmiss | #V
      sum(is.na(temp[QColumns[i]]))/nrow(temp) > pctmiss ) {#Q

    names(temp)[VColumns[i]]<-paste0("REMOVED_", names(temp)[VColumns[i]])
    VColumns0<-VColumns0[!(VColumns0 %in% VColumns[i])]
    names(temp)[QColumns[i]]<-paste0("REMOVED_", names(temp)[QColumns[i]])
    QColumns0<-QColumns0[!(QColumns0 %in% QColumns[i])]
  }
}

VColumns<-names(temp)[VColumns0]
QColumns<-names(temp)[QColumns0]
```

	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 Category 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.

```r
names(temp)[grep(pattern = NameBasecategory, x = names(temp))]<-
  paste0("REMOVED_",
         names(temp)[grep(pattern = NameBasecategory, x = names(temp))])

  # Q
  temp.q<-temp[,grepl(pattern = paste0("Q", ii), x = substr(names(temp), start = 1, stop = 2)) ]
  temp.q<-data.frame(temp.q)
  if (ncol(temp.q)>1) {
    temp.q<-rowSums(temp.q, na.rm = T)
  }
  temp[ncol(temp)+1]<-temp.q
```

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

# V
temp.v<-temp[,grepl(pattern = paste0("V", ii), x = substr(names(temp), start = 1, stop = 2)) ]
temp.v<-data.frame(temp.v)
if (ncol(temp.v)>1) {
  temp.v<-rowSums(temp.v, na.rm = T)
}
temp[ncol(temp)+1]<-temp.v
names(temp)[ncol(temp)]<-paste0("V",NameBasecategory)
```

	REMOVED_QE1_0Finfish	REMOVED_V1_0Finfish	QE1_0Finfish	V1_0Finfish
2007	3000	3800	3000	2800
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 ($P_{s,t,i}$; 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) =

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

where:

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

```r
    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 $V_{s,t,i}$, don't calculate that species $P_{s,t,i}$

```r
#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) > pctmiss) {
    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
2013	16.36364	1.400000
2014	15.45455	1.391304
2015	20.00000	1.458333
2016	12.00000	1.454546

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

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

We use this *ReplaceFirst* 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: 0x000000001ad5c780>
```

```
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

	P1__1Salmon	P1__2Cod
2015	20.00000	1.458333
2016	12.00000	1.454546

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

We use this *ReplaceMid* function:

```
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: 0x000000001a8c46c8>
```

```
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 Impute values of $V_{t,i,s}$ 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 $V_{t,i,s}$.

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

2.0.7.1 1. If the first value of $V_{t,i,s}$ is 0/NA in a timeseries, we let the next available non-zero value of $V_{t,i,s}$ 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
2014	170	3200
2015	200	3500
2016	180	3200

2.0.7.2 2. If there is a value in the middle of $V_{t,i,s}$'s timeseries that is 0/NA, we let the most recent past available non-zero of $V_{t,i,s}$ 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 $VV_{t,i}$ where P was able to be calculated

$R_{t,i}$, 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 $V_{t,i}$ available (called $VV_{t,i}$) for the category using only values for species that were used to calculate $P_{t,i}$ (called $V_{s,t,i,available}$).

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

where:

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

```r
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
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 ($R_{s,t,i}$; e.g., Salmon and Flounder)

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

where:

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

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

```r
tempR<-data.frame(data = rep_len(x = NA, length.out = nrow(temp)))
for (c in 1:length(PColumns)) {

  #for renaming the columns
    NameBase<-substr(start = 2,
                     stop = nchar(PColumns[c]),
                     x = PColumns[c])

  V<-(temp[,names(temp) %in% paste0("VV", NameBasecategory)])  # sum of V where P was calculated
  V0<-temp[,names(temp) %in% paste0("V", NameBase)] #V of species; to make sure its the same column
  tempR[,c]<-V0/V
  names(tempR)[c]<-paste0("R", NameBase ) #name the column
}

tempR<-data.frame(tempR)
temp<-cbind.data.frame(temp, tempR)
```

	R1_1Salmon	R1_2Cod
1	0.0344828	0.9655172
2	0.0357143	0.9642857
3	0.0333333	0.9666667
4	0.0322581	0.9677419

	R1_1Salmon	R1_2Cod
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

2.0.9.1 Analysis Warnings Checks

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

Is there a warning?

No warning.

2.0.10 Revenue Share-Weighted Price Changes for each species ($PCW_{s,t,i}$; e.g., Salmon and Flounder)

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

Where:

- $PCW_{t,i,s}$ = Revenue share-weighted price change for a species (s)

Such that:

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

We use this *PriceChange* function:

```
print(PriceChange)
```

```
## function(R0, P0) {
##   PCW<-rep_len(x = 0, length.out = length(P0))
##   for (t in 2:length(P0)) {
##     AverageR<-((R0[t]+R0[t-1])/2) #Average Revenue Share
##     PC<-ln(P0[t]/P0[t-1]) #Price Change
##     PCW[t]<-AverageR*PC #Revenue Share-Weighted Price Chage
##   }
##   return(PCW)
## }
## <bytecode: 0x000000000c6f8750>
```

```
#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("PCW", NameBase ) #name the column
}

temp<-cbind.data.frame(temp, tempPC)
```

2.0.11 Price Changes for the category ($PC_{t,i}$; e.g., Finfish)

$$PC_{t,i} = ln(\frac{P_{t,i}}{P_{t-1,i}}) = \sum_{s=1}^{n}(PCW_{t,i,s})$$

Where:

- $PC_{t,i}$ = Price change for a category (i)

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

	PCW1_1Salmon	PCW1_2Cod	PC1_0Finfish
1	0.0000000	0.0000000	0.0000000
2	0.0000000	0.0144018	0.0144018
3	0.0000000	0.0194695	0.0194695
4	0.0000000	-0.1830357	-0.1830357
5	0.0220102	0.0712743	0.0932846
6	0.0089738	-0.0231613	-0.0141875
7	0.0147572	0.0989356	0.1136927
8	-0.0031679	-0.0058852	-0.0090532
9	0.0134715	0.0445941	0.0580655
10	-0.0274080	-0.0024612	-0.0298692

2.0.12 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 $PI_{t=firstyear} = 1$

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

Where

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

```
#Note that the first row of this column is = 1
tempPI_yr1<-data.frame(c(1, rep_len(x = NA, length.out = nrow(temp)-1)))
rownames(tempPI_yr1)<-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:nrow(tempPI_yr1)){  #Since the first row is defined, we need to start at the second row
    tempPI_yr1[t,1]<-tempPI_yr1[t-1,1]*exp(PC0[t])
}
```

Then, to change the price index into base year dollars, we use the following equation:

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

In this example, our base year is 2010. Notice that the $PI_{t,i=baseyr} = 1$

```
tempPI_yrb<-tempPI_yr1[,1]/tempPI_yr1[rownames(tempPI_yr1) %in% baseyr,1]
```

In the future, we will use this *PriceIndex* function to do this step:

```
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
##   tempPI_yr1<-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(tempPI_yr1)){  #Since the first row is defined, we need to start at the second r
##     tempPI_yr1[t]<-tempPI_yr1[t-1]*exp(PC0[t])
##   }
##
##   tempPI_yr1<-data.frame(tempPI_yr1)
##   rownames(tempPI_yr1)<-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
##
##   tempPI_yrb<-tempPI_yr1/tempPI_yr1[rownames(tempPI_yr1) %in% baseyr,]
##
##   tempPI<-data.frame(tempPI_yrb)
##   names(tempPI)<-paste0("PI", BaseColName)
##
##   return(tempPI)
## }
## <bytecode: 0x000000000c4f3208>
```

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)
```

	tempPI_yr1	tempPI_yrb	PI1_0Finfish
2007	1.0000000	1.160864	1.160864
2008	1.0145060	1.177703	1.177703
2009	1.0344514	1.200857	1.200857
2010	0.8614275	1.000000	1.000000
2011	0.9456527	1.097774	1.097774
2012	0.9323310	1.082309	1.082309
2013	1.0445909	1.212628	1.212628
2014	1.0351767	1.201699	1.201699
2015	1.0970642	1.273542	1.273542
2016	1.0647803	1.236065	1.236065

2.0.13 Implicit Quantity/Output for each category ($Q_{t,i}$; 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.

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

```
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)
```

	x
1	2411.997
2	2394.491
3	2506.543
4	3190.000
5	2987.864
6	2910.443
7	2539.938
8	2804.362
9	2905.283
10	2734.484

2.0.14 Analysis Warnings Checks

2.0.14.1 1. When back calculated, V_t should equal $PI_t * Q_t$

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

```
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")
```

	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

Is there a warning?

No warning.

2.0.14.2 2. When back calculated, $Q_{t,i}$ should equal $V_{t,i}/PI_{t,i}$

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

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

	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

Is there a warning?

No warning.

2.1 Redo Analysis for Shellfish

Now lets redo that whole analysis up to this point (via function) for the two species of the shellfish group, as we will need them for the next steps of this analysis.

We use this *ImplicitQuantityOutput* function to calculate the Implicit Quanity Output at Species and category Level:

```
print(ImplicitQuantityOutput.species.cat)
```

```
## function(temp, ii, baseyr, maxyr, minyr, pctmiss, warnings.list) {
##
##   ########Housekeeping
##   # Here I am just going to collect some housekeeping items
##   temp<-data.frame(temp)
##
##   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 = names(temp)[grep(x = names(temp), pattern = paste0(NumberOfSpecies, "Tota
##                         start = 2,
##                         stop = nchar(names(temp)[grep(x = names(temp),
##                                                       pattern = paste0(NumberOfSpecies, "Total"))][2
##
##   NameBasecategory<-names(temp)[grepl(pattern = NumberOfSpecies,
##                                       x = names(temp)) &
##                                   grepl(pattern = paste0("V", ii), x = names(temp))]
##
##   NameBasecategory<-substr(start = 2,
##                            stop = nchar(NameBasecategory),
##                            x = NameBasecategory)
##
##
##   VColumns<-grep(pattern = paste0("V", ii,"_"),
##                  x = substr(x = names(temp),
##                             start = 1,
##                             stop = (2+nchar(ii))))
##
##   VColumns<-VColumns[!(grepl(pattern = NameBasecategory,
##                              x = names(temp)[VColumns]))]
##
##
##   ###Remove any related V and Q data where V column has less data than the $pctmiss$
##   VColumns0<-VColumns
##   QColumns0<-QColumns<-which(names(temp) %in%
##                               paste0("Q", substr(x = names(temp)[VColumns],
##                                                  start = 2,
##                                                  stop = nchar(names(temp)[VColumns]))))
##
##   for (i in 1:length(VColumns)) {
```

```
##
##     #if the percent missing is less in V or Q columns for a species than the percentmissingtrheshold
##     if (sum(is.na(temp[VColumns[i]]))/nrow(temp) > pctmiss) {
##
##       names(temp)[VColumns[i]]<-paste0("REMOVED_", names(temp)[VColumns[i]])
##       VColumns0<-VColumns0[!(VColumns0 %in% VColumns[i])]
##       names(temp)[QColumns[i]]<-paste0("REMOVED_", names(temp)[QColumns[i]])
##       QColumns0<-QColumns0[!(QColumns0 %in% QColumns[i])]
##     }
##   }
##
##   VColumns<-names(temp)[VColumns0]
##   QColumns<-names(temp)[QColumns0]
##
##
##   ###Caluclate Catagory Sums of $V$ and $Q$
##
##   # Because we removed some columns for not meeting a perecent missing threshold and those columns wi
##   names(temp)[grep(pattern = NameBasecategory, x = names(temp))]<-
##     paste0("REMOVED_",
##            names(temp)[grep(pattern = NameBasecategory, x = names(temp))])
##
##   ####
##   #if there are still columns to assess that haven't been "removed"
##   PColumns<-c()
##   if (length(VColumns) == 0) {
##     warnings.list[length(warnings.list)+1]<-paste0("FYI: ", NameBasecategory, " is no longer being ca
##
##   } else {
##
##     # Q
##   temp.q<-temp[,grepl(pattern = paste0("Q", ii), x = substr(names(temp), start = 1, stop = 2+nchar(ii
##   temp.q<-data.frame(temp.q)
##   if (ncol(temp.q)>1) {
##     temp.q<-rowSums(temp.q, na.rm = T)
##   }
##   temp[ncol(temp)+1]<-temp.q
##   names(temp)[ncol(temp)]<-paste0("QE",NameBasecategory)
##
##   # V
##   temp.v<-temp[,grepl(pattern = paste0("V", ii), x = substr(names(temp), start = 1, stop = 2+nchar(ii
##   temp.v<-data.frame(temp.v)
##   if (ncol(temp.v)>1) {
##     temp.v<-rowSums(temp.v, na.rm = T)
##   }
##   temp[ncol(temp)+1]<-temp.v
##   names(temp)[ncol(temp)]<-paste0("V",NameBasecategory)
##
##
##
##   # #Caulculate the summed quantity
##   # temp[,(ncol(temp)+1)]<-rowSums(temp[, grep(pattern = paste0("Q", strsplit(x = NameBasecategory, s
##   #                                  x = names(temp))], na.rm = T)
##   # names(temp)[ncol(temp)]<-paste0("QE", NameBasecategory)
```

```
##
##
##   # 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))
##
##   # There may be instances where price cannot be calculated because there is no Q or V data for that
##
##   #### 2.3. V and/or Q are completely missing from the timeseries. In this case, we will remove the
##
##   #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 "PColu
##     if (sum(temp[,PColumns[c]] %in% c(0, NA, NaN))/nrow(temp) > pctmiss |
##         # sum(temp[,QColumns[c]] %in% c(0, NA, NaN))/nrow(temp) > pctmiss|
##         sum(temp[,VColumns[c]] %in% c(0, NA, NaN))/nrow(temp) > pctmiss) {
##       cc<-c(cc, c)#Collect offending columns
##     }
##   }
##
##   if (length(cc)>0){
##     PColumns<-PColumns[-cc]
##     # VColumns<-VColumns[-cc]
##   }
##
##   }
##
##   if (length(PColumns) == 0) {
##
##     warnings.list[length(warnings.list)+1]<-paste0("FYI: ", NameBasecategory, " is no longer being ca
##
##   } else {
##
##
##   # 2.1. If the first value of P is 0 in a timeseries, we let the next available non-zero value of P
##
##   for (c in 1:length(PColumns)) {
##
##     #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,PColumns[c]] %in% c(0, NA, NaN)) {
##       findfirstvalue<-temp[which(!(temp[,PColumns[c]]  %in% c(0, NA, NaN))), PColumns[c]][1]
##       temp[1,PColumns[c]]<-findfirstvalue
##     }
##   }
##
##   # 2.2. If there is a value in the middle of P's timeseries that is 0, we let the most recent past a
##   for (c in 1:length(PColumns)) {
##
##     #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[,PColumns[c]] %in% c(0, NA, NaN))>0) {
##       troublenumber<-which(temp[,PColumns[c]] %in% c(0, NA, NaN))
##       for (r in 1:length(troublenumber)){
##         findlastvalue<-temp[troublenumber[r]-1, PColumns[c]][1]
##         temp[troublenumber[r],PColumns[c]]<-findlastvalue
##       }
##     }
##   }
##
##   ###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 bec
##   # $$where \begin{cases} if: V_{i,t=1} = 0, then: V_{i,t=1} = V_{i,t=1+1...} \\ if: V_{i,t\neq1} = (
##
##   #### 1. If the first value of $V_{i,t,s}$ is 0/NA in a timeseries, we let the next available non-ze
##   VVColumns<-paste0("V", substr(x = PColumns, start = 2, stop = nchar(PColumns)))
##   temp<-ReplaceFirst(colnames = VVColumns, temp)
##
##   #### 2. If there is a value in the middle of $V_{i,t,s}$'s timeseries that is 0/NA, we let the most
##   temp<-ReplaceMid(colnames = VVColumns, temp)
##
##   ###Value of species where P was able to be calculated
##   # $R_{i,t}$, defined and discussed in the subsequent step, will need to sum to 1 across all species
##   # $$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 valu
##   #   - $V_{s,i,t, available}$ are the $V_{s,i,t}$ where P were able to be calculated
##
##   VVColumns<-paste0("V", substr(x = PColumns, start = 2, stop = nchar(PColumns)))
##
##   temp[ncol(temp)+1]<-rowSums(data.frame(temp[,names(temp) %in% VVColumns]), na.rm = T)
##   names(temp)[ncol(temp)]<-paste0("VV",NameBasecategory)
##
##   ###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 ($) per individual species (s), category (i), for each year (t)
##
##   tempR<-data.frame(data = rep_len(x = NA, length.out = nrow(temp)))
##   for (c in 1:length(PColumns)) {
##
##     #for renaming the columns
##     NameBase<-substr(start = 2,
##                      stop = nchar(PColumns[c]),
##                      x = PColumns[c])
##
##     VV<-temp[,names(temp) %in% paste0("VV", NameBasecategory)]  # sum of V where P was calculated
##     V0<-temp[,names(temp) %in% paste0("V", NameBase)] #V of species; to make sure its the same column
##     tempR[,c]<-V0/VV
##     names(tempR)[c]<-paste0("R", NameBase) #name the column
##   }
##
##   tempR<-data.frame(tempR)
##   temp<-cbind.data.frame(temp, tempR)
##
##   #Note if there is an error
##   if (sum(rowSums(tempR, na.rm = T)) != nrow(temp)) {
##     warnings.list[length(warnings.list)+1]<-paste0("FYI: Rows of R_{s,i,t} for ",NameBasecategory," d
##   }
##
##   #remove duplicates
##   temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##   temp <- temp[, !duplicated(colnames(temp))]
##   temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##   temp <- temp[, !duplicated(colnames(temp))]
##
##
##   ###Price Changes for each species ($PC_{s,i,t}$ aka $\Delta ln(P_{s,i,t})$; e.g., Salmon and Flound
##
##   # $$PCW_{i,t,s} = \frac{R_{s,i,t} + R_{s,i,t-1}}{2} * ln(\frac{P_{s,i,t}}{P_{s,i,t-1}}) = \frac{R_
##   # Where:
##     # - $PCW_{i,t,s}$ = Revenue share-weighted price change for a species (s)
##   # Such that:
##     # - category's (i) Price Change for each species (s) = $\frac{R_{s,i,t} + R_{s,i,t-1}}{2}$
##     # - category's (i) Revenue Share for each species (s) = $ln(\frac{P_{s,i,t}}{P_{s,i,t-1}} = [ln(
##
##   #Find which columns in this table are price and revenue share columns
##   tempPC<-data.frame(data = rep_len(x = NA, length.out = nrow(temp)))
##   for (c in 1:length(PColumns)){
##     #For nameing columns
##     NameBase<-substr(start = 2,
##                      stop = nchar(PColumns[c]),
##                      x = PColumns[c])
##
##     # Calculate
##     P0<-temp[, names(temp) %in% paste0("P", NameBase)]
##     R0<-temp[, names(temp) %in% paste0("R", NameBase)] #to make sure its the same column
##     tempPC[,c]<-PriceChange(R0, P0)
##     names(tempPC)[c]<-paste0("PCW", NameBase ) #name the column
##   }
##
##   temp<-cbind.data.frame(temp, tempPC)
##
##   ###Price Changes for the catagory ($PC_{i,t}$; e.g., Finfish)
##   # $$PC_{i,t} = ln(\frac{P_{i,t}}{P_{i,t-1}}) = \sum_{s=1}^n(PCW_{i,t,s}) $$
##   #   Where:
##   #   - $PC_{i,t}$ = Revenue share-weighted price change for a catagory (i)
##
##   temp[ncol(temp)+1]<-rowSums(tempPC, na.rm = T)
##   names(temp)[ncol(temp)]<-paste0("PC", NameBasecategory)
##
##   ###4. Price Changes for each species ($PC_{s,i,t}$ aka $\Delta ln(P_{s,i,t})$; e.g., Salmon and Fl
##
##   #Find which columns in this table are price and revenue share columns
##   tempPC<-data.frame(data = rep_len(x = NA, length.out = nrow(temp)))
##   for (c in 1:length(PColumns)){
##     #For nameing columns
##     NameBase<-substr(start = 2,
##                      stop = nchar(PColumns[c]),
##                      x = PColumns[c])
##
##     # Calculate
##     P0<-temp[, names(temp) %in% paste0("P", NameBase)]
##     R0<-temp[, names(temp) %in% paste0("R", NameBase)] #to make sure its the same column
##     tempPC[,c]<-PriceChange(R0, P0)
##     names(tempPC)[c]<-paste0("PC", NameBase ) #name the column
##   }
##
##   temp[ncol(temp)+1]<-rowSums(tempPC, na.rm = T)
##   names(temp)[ncol(temp)]<-paste0("PC", NameBasecategory)
##
##
##
##   #remove duplicates
##   temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##   temp <- temp[, !duplicated(colnames(temp))]
##   temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##   temp <- temp[, !duplicated(colnames(temp))]
##
## ###6. Price Indexes for each category ($PI_{i,t}$; Finfish & others and Shellfish)
## # We calculate the price index first by comparing by multiplying the previous years $PI_{t-1}$ by tha
##   ###Price Index for the each category ($PI_t$)
##   #
##   # We calculate the price index first by comparing by multiplying the previous years $PI_{t-1}$ by t
##   # $$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$$
##   #Note that the first row of this column is = 1
##   #
##   # Then, to change the price (calulated later) into base year dollars, we use the following equation
##     # $$PI_{t} = PI_{t}/PI_{t = baseyear}$$
##     # In this example, we'll decide that the base year is `r baseyr`, for whatever reason. Notice tha
##
##   tempPI<-PriceIndex(temp, BaseColName = NameBasecategory, baseyr)
##   temp[ncol(temp)+1]<-(tempPI)
##   names(temp)[ncol(temp)]<-paste0("PI", NameBasecategory)
##
##
##   #remove duplicates
##   temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##   temp <- temp[, !duplicated(colnames(temp))]
##   temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##   temp <- temp[, !duplicated(colnames(temp))]
##
##   ###7. Implicit Quantity/Output for each category ($Q_{i,t}$; Finfish & others and Shellfish)
##   temp[,ncol(temp)+1]<-temp[,names(temp) %in% paste0("V", NameBasecategory)]/
##     temp[,names(temp) %in% paste0("PI", NameBasecategory)]
##
##   names(temp)[ncol(temp)]<-paste0("Q", NameBasecategory)
##
##
##   #############WARNINGS
##   ####1. When back calculated, $V_t$ did not equal $P_t * Q_{t}$
##   # $$V_i = P_t * Q_i$$
##
##   temp0<-temp[names(temp) %in% c(paste0("Q",NameBasecategory),
##                                  paste0("PI",NameBasecategory),
##                                  paste0("V",NameBasecategory))]
##
##   temp0[,(ncol(temp0)+1)]<-temp0[,paste0("Q",NameBasecategory)]*temp0[,paste0("PI",NameBasecategory)]
##   names(temp0)[ncol(temp0)]<-paste0("V", NameBasecategory, "_Check")
##
##   if ((length(setdiff(as.character(temp0[,paste0("V", NameBasecategory, "_Check")]),
##                       as.character(temp0[,paste0("V", NameBasecategory)]))) != 0)) {
##     warnings.list[length(warnings.list)+1]<-"Warning: When back calculated, V_{i,t} did not equal PI
##   }
##
##
##   ####2. When back calculated, $Q_{t}$ did not equal $V_t / P_{t}$
##   # $$Q_{i,t} = V_t / P_{i,t}$$
##
##   temp0[,(ncol(temp0)+1)]<-temp0[, paste0("V", NameBasecategory)]/temp0[, paste0("PI", NameBasecateg
##   names(temp0)[ncol(temp0)]<-paste0("Q", NameBasecategory, "_Check")
##
##  if (length(setdiff(as.character(temp0[,paste0("Q", NameBasecategory, "_Check")]),
##                                  as.character(temp0[,paste0("Q", NameBasecategory)]))) != 0) {
##     warnings.list[length(warnings.list)+1]<-"Warning: When back calculated, Q_{i,t} did not equal V_
##   }
##
## }
##   return(list(temp, warnings.list))
## }
## <bytecode: 0x000000001ad02ff8>
```

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

tempS<-ImplicitQuantityOutput.species.cat(temp, ii, baseyr, maxyr, minyr,
                          pctmiss = pctmiss,
                          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?

	R2_2Clam	PCW2_1Shrimp	PCW2_2Clam	PC2_0Shellfish	PI2_0Shellfish	Q2_0Shellfish
2007	0.5555556	0.0000000	0.0000000	0.0000000	1.030337	1747.0009
2008	0.5454545	0.0183493	0.0648402	0.0831894	1.119717	1964.7830
2009	0.5000000	-0.0087575	-0.0805788	-0.0893363	1.024023	1757.7726
2010	0.5625000	-0.0237392	0.0000000	-0.0237392	1.000000	700.0000
2011	0.5000000	0.1730144	0.0000000	0.1730144	1.188883	757.0129
2012	0.4736842	-0.0604413	0.0000000	-0.0604413	1.119154	893.5320
2013	0.4545455	0.0977034	0.0488994	0.1466028	1.295862	1697.7119
2014	0.4500000	-0.0998625	0.0614193	-0.0384432	1.246990	1603.8619
2015	0.5000000	0.0553143	-0.0293044	0.0260098	1.279850	1562.6836
2016	0.4782609	0.0393171	-0.0549436	-0.0156266	1.260005	1825.3889

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

$R_{t,i}$, 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 $V_{t,i}$ (called VV_t) for the category using only values for species that were used to calculate $PI_{t,i}$.

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

where:

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

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 each category ($R_{t,i}$)

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

where:

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

```r
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

2.1.2.1 Analysis Warnings Checks

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

	x
2007	1
2008	1
2009	1
2010	1

	x
2011	1
2012	1
2013	1
2014	1
2015	1
2016	1

Is there a warning?

No warning.

2.1.3 Revenue Share-Weighted Price Changes for each category ($PCW_{t,i}$; e.g., Salmon and Flounder)

$$PCW_{t,i} = \frac{R_{t,i} + R_{t-1,i}}{2} * ln(\frac{PI_{t,i}}{PI_{t-1,i}}) = \frac{R_{t,i} + R_{t-1,i}}{2} * [ln(PI_{t,i}) - ln(PI_{t-1,i})]$$

Where:

- $PCW_{t,i}$ = Revenue share-weighted price change for a category (i)

Such that:

- Price Change for each category (i) = $\frac{R_{t,i}+R_{t-1,i}}{2}$
- Revenue Share for each category (i) = $ln(\frac{PI_{t,i}}{PI_{t-1,i}} = [ln(PI_{t,i}) - ln(PI_{t-1,i})]$

```
#Find which columns in this table are price and revenue share columns
tempPCW<-cbind.data.frame("PCW1_0Finfish" =
                            PriceChange(R0 = temp$R1_0Finfish, P0 = temp$PI1_0Finfish),
                          "PCW1_0Shellfish" =
                            PriceChange(R0 = temp$R2_0Shellfish, P0 = temp$PI2_0Shellfish))
```

2.1.4 Price Changes for the entire fishery ($PC_{t,i}$; e.g., Finfish)

$$PC_t = ln(\frac{P_t}{P_{t-1}}) = \sum_{s=1}^{n}(PCW_{t,i})$$

Where:

- PC_t = Price change for the entire fishery

```
temp$PC0_0Total<-rowSums(tempPCW, na.rm=T)
```

	PCW1_0Finfish	PCW1_0Shellfish	PC0_0Total
1	0.0000000	0.0000000	0.0000000
2	0.0084283	0.0345050	0.0429332
3	0.0115603	-0.0362914	-0.0247311
4	-0.1323193	-0.0065778	-0.1388971
5	0.0748488	0.0341928	0.1090416
6	-0.0109508	-0.0137889	-0.0247398
7	0.0763088	0.0482052	0.1245141
8	-0.0054812	-0.0151679	-0.0206491

	PCW1_0Finfish	PCW1_0Shellfish	PC0_0Total
9	0.0370656	0.0094067	0.0464723
10	-0.0185815	-0.0059053	-0.0244869

2.1.5 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 $PI_{t=firstyear} = 1$

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

Where

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

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

2.1.6 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

	x
8	4448.919
9	4507.880
10	4603.418

2.1.7 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
2016	...	0.4049296	-0.0244869	1.233866	4603.418	1.1833979

2.1.8 Sum Total Simple Sum Quantity Output Index

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

Where:

- QE_t is the sum of Q before these calculations; the simple sum
- QEI_t 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

2.1.9 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 should equal $PI_t * Q_t$?

$$V_t = P_t * Q_t$$

```
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")
```

	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

Is there a warning?

```
  if (length(setdiff(as.character(temp0[,paste0("V", NameBaseTotal, "_Check")]),
                     as.character(temp0[,paste0("V", NameBaseTotal)]))) != 0) {
  warnings.list[length(warnings.list)+1] <- "Warning: When back calculated, V_t did not equal PI_t x Q_

    a<-warnings.list[length(warnings.list)][[1]]
} else {
    a<-"No warning."
}
```

No warning.

2.2.0.2 2. When back calculated, Q_t should V_t/PI_t?

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

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

	V0__0Total	PI0__0Total	Q0__0Total	Q0__0Total__Check
2007	4600	1.128281	4077.000	4077.000
2008	5020	1.177776	4262.269	4262.269
2009	4810	1.149006	4186.228	4186.228
2010	3890	1.000000	3890.000	3890.000
2011	4180	1.115209	3748.177	3748.177
2012	4150	1.087957	3814.488	3814.488
2013	5280	1.232218	4284.956	4284.956
2014	5370	1.207035	4448.919	4448.919
2015	5700	1.264452	4507.880	4507.880
2016	5680	1.233866	4603.418	4603.418

Is there a warning?

```
  if (length(setdiff(as.character(temp0[,paste0("Q", NameBaseTotal, "_Check")]),
                     as.character(temp0[,paste0("Q", NameBaseTotal)]))) != 0) {
  warnings.list[length(warnings.list)+1]<-"Warning: When back calculated, Q_t did not equal V_t/PI_t"
    a<-warnings.list[length(warnings.list)][[1]]
} else {
    a<-"No warning."
}
```

No warning.

2.2.0.3 3. When back calculated, growth rate?

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

```
names0<-c(paste0("Q",NameBaseTotal))
for (i in 1:ii) {
  names0<-c(names0,
            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"
```

	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	0.0444654
2009	4186.228	2506.543	0.6257796	1757.7726	0.3742204	0.0180017	-0.0180726
2010	3890.000	3190.000	0.8200514	700.0000	0.1799486	0.0733908	-0.0808110
2011	3748.177	2987.864	0.7846890	757.0129	0.2153110	0.0371395	-0.0370504
2012	3814.488	2910.443	0.7590361	893.5320	0.2409639	-0.0175368	0.0175616
2013	4284.956	2539.938	0.5833333	1697.7119	0.4166667	-0.1163037	0.1196593
2014	4448.919	2804.362	0.6275605	1603.8619	0.3724395	-0.0375509	0.0375242
2015	4507.880	2905.283	0.6491228	1562.6836	0.3508772	-0.0131659	0.0131616
2016	4603.418	2734.484	0.5950704	1825.3889	0.4049296	-0.0209719	0.0210303

Is there a warning?

```
  if (length(setdiff(as.character(temp0[,"part1"]),
                     as.character(temp0[,"part2"]))) != 0) {
  warnings.list[length(warnings.list)+1]<-"Warning: When back calculated, ln(Q_t/Q_{t-1}) = did not equa
    a<-warnings.list[length(warnings.list)][[1]]
} else {
    a<-"No warning."
}
```

Warning: When back calculated, ln(Q_t/Q_{t-1}) = did not equal sum(((R_{i, t} - R_{i, t-1})(2)) ln((Q_{t,i})(Q_{t-1,i}))*

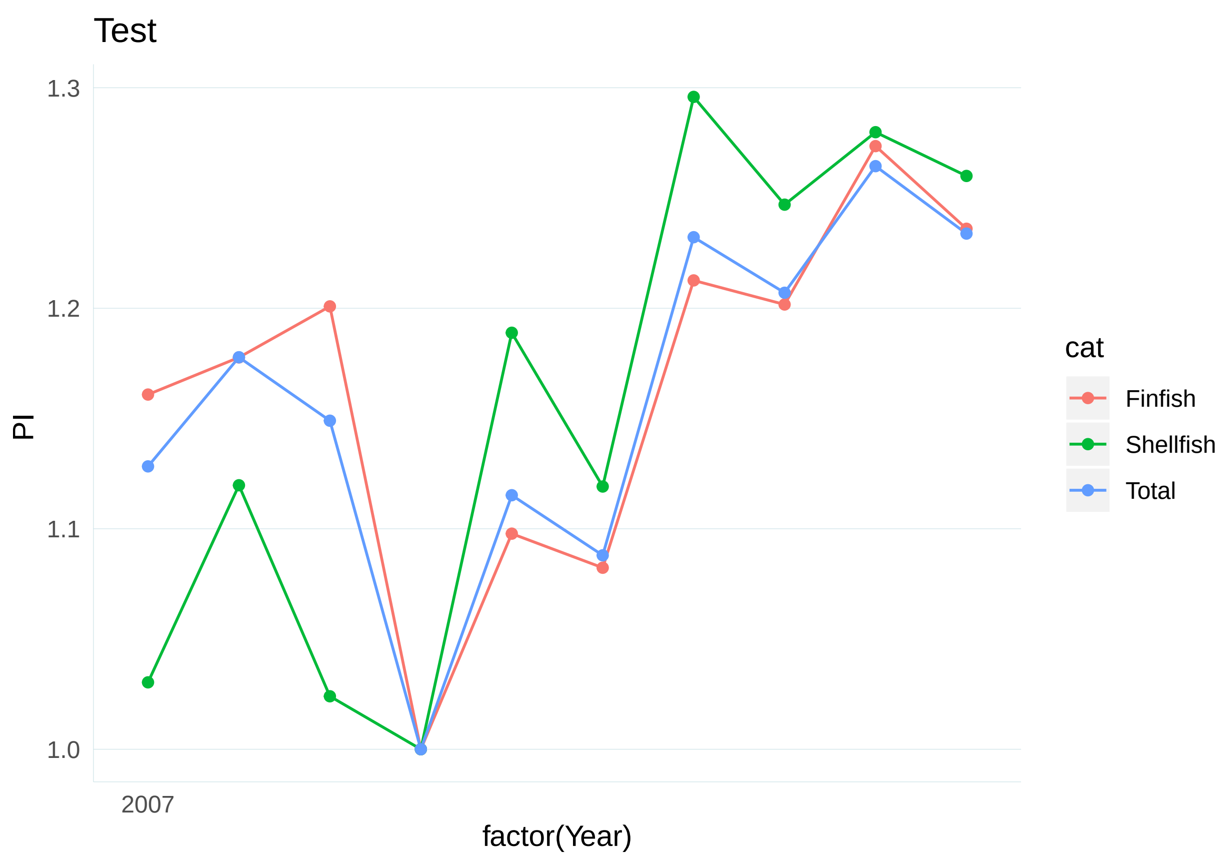
2.2.1 View Total Outputs

	QE0_0Total	REMOVED_V0_0Total	VV0_0Total	V0_0Total	PC0_0Total	PI0_0Total	Q0_0Total	Q
2007	3250	5600	4700	4600	0.0000000	1.128281	4077.000	
2008	3380	6220	5000	5020	0.0429332	1.177776	4262.269	
2009	3150	5710	4800	4810	-0.0247311	1.149006	4186.228	
2010	2610	3890	4700	3890	-0.1388971	1.000000	3890.000	
2011	2490	4180	5000	4180	0.1090416	1.115209	3748.177	
2012	2412	4150	4950	4150	-0.0247398	1.087957	3814.488	
2013	3251	6280	5180	5280	0.1245141	1.232218	4284.956	
2014	3431	6270	5370	5370	-0.0206491	1.207035	4448.919	
2015	3630	6700	5700	5700	0.0464723	1.264452	4507.880	
2016	3575	6780	5680	5680	-0.0244869	1.233866	4603.418	

2.2.2 Graph 1: Price Index

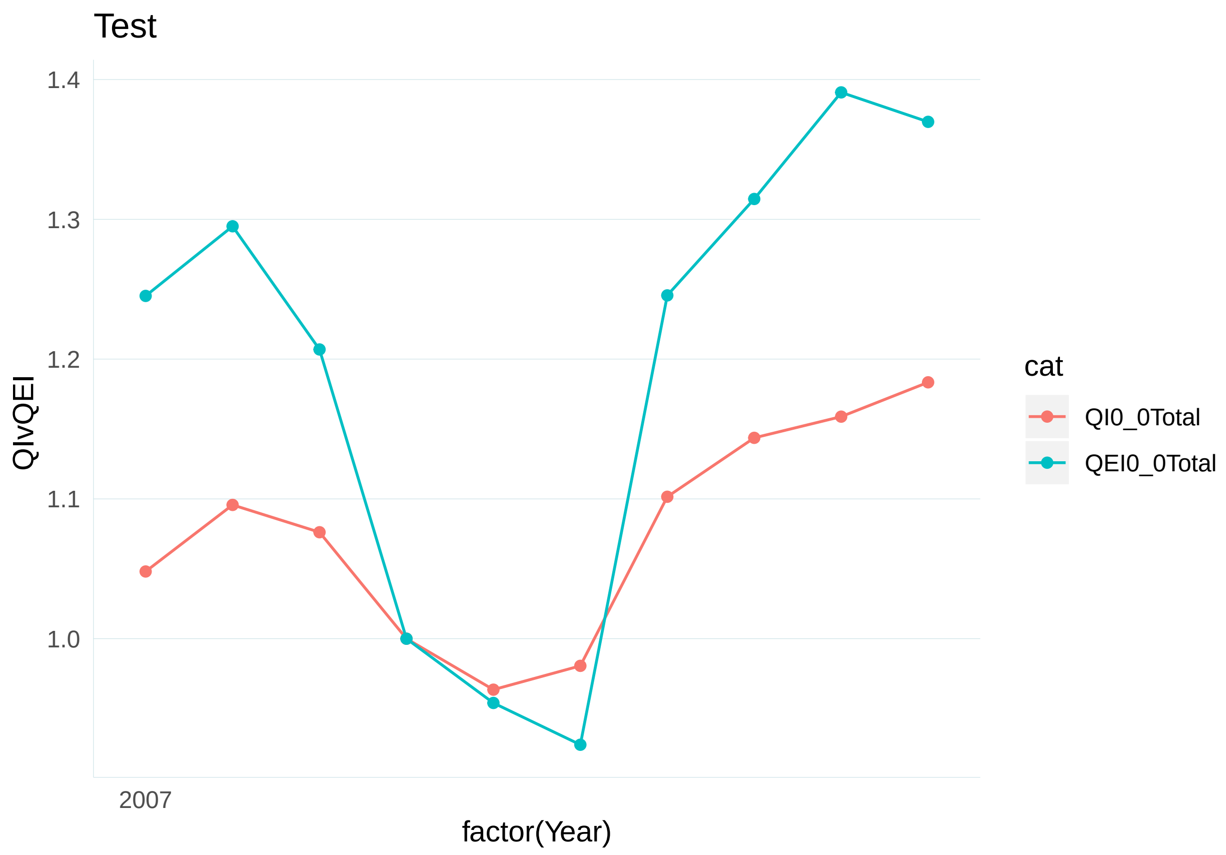
In theory, PI should be negative slope after the baseyear and positive after the base year, but because this data was fabricated without thinking of this, we don't see that here. The index value for the base year is = 1, however.

```
title00<- "_PI-Line"

a0<-data.frame(temp[,grepl(
  pattern = paste0("PI[0-9]+_", NumberOfSpecies),
  x = names(temp))])

a0$Year<-rownames(a0)

a <- gather(a0, Category, val, names(a0)[grepl(pattern = NumberOfSpecies, x = names(a0))], factor_key

a$cat<-as.character(lapply(X = strsplit(x = as.character(a$Category), split = paste0("_", NumberOfSpe

temp0<-a

plotnlines(dat = temp0, title00, place)
```



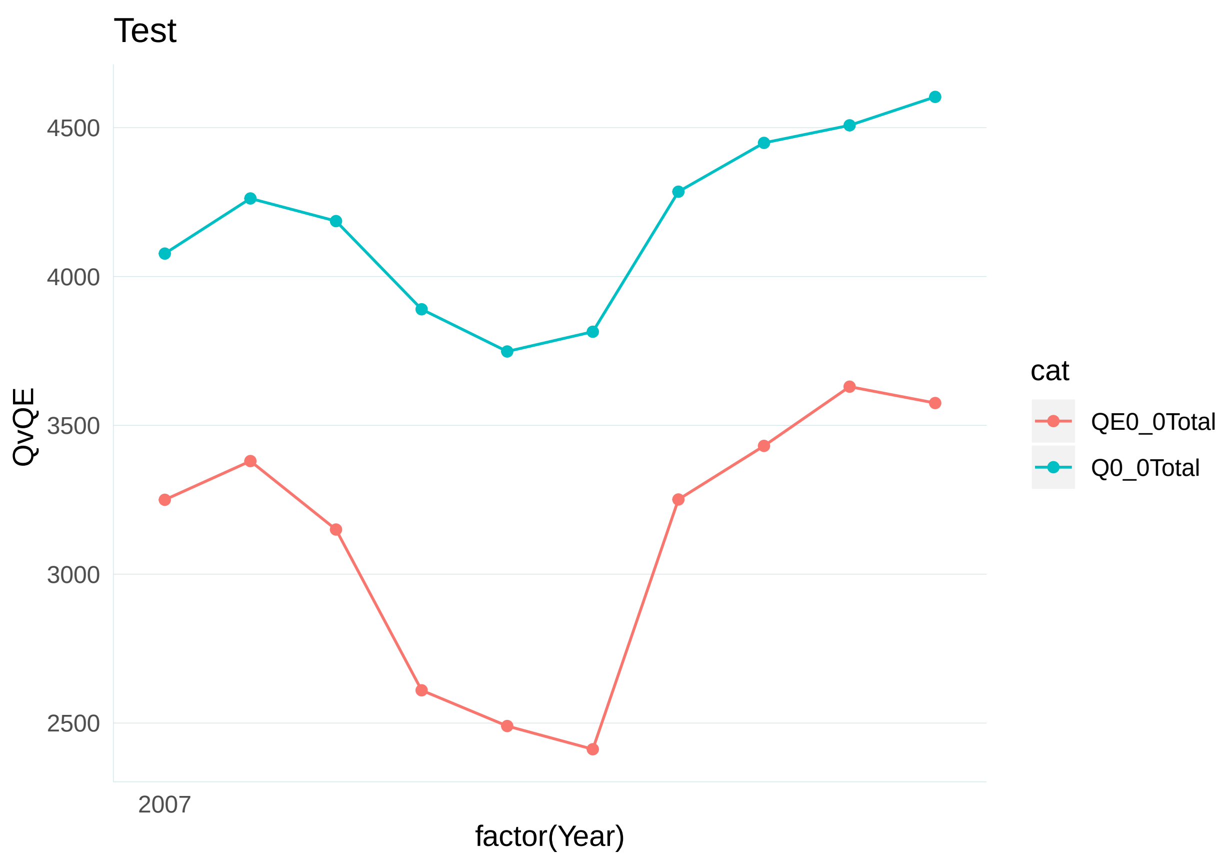
2.2.3 Graph 2: Quantity Index Compare

```
title00<- "_QIvQEI-Line"

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

temp0<-data.frame(temp0[,names(temp0) %in% c("Year",
                                             paste0("QI", NameBaseTotal),
                                             paste0("QEI", NameBaseTotal))])
temp0$Year<-rownames(temp)

temp0<-gather(temp0, cat, val,
              names(temp0)[1]:names(temp0)[length(names(temp0))-1],
              factor_key = T)

plotnlines(dat = temp0,  title00, place)
```



2.2.4 Graph 3: Quantity Compare

```r
title00<- "_QvQE-Line"

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

temp0<-data.frame(temp0[,names(temp0) %in% c("Year",
                                             paste0("Q", NameBaseTotal),
                                             paste0("QE", NameBaseTotal))])
temp0$Year<-rownames(temp)

temp0<-gather(temp0, cat, val,
              names(temp0)[1]:names(temp0)[length(names(temp0))-1],
              factor_key = T)

plotnlines(dat = temp0,  title00, place)
```



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

We use this *ImplicitQuantityOutput* function to calculate the Implicit Quanity Output at Fishery Level:

```
print(ImplicitQuantityOutput)
```

```
## function(temp, baseyr, pctmiss = 1.00,
##                                  title0 = "", place = ""){
##
##   temp.orig<-temp
##
##   warnings.list<-list()
##   figures.list<-list()
##
```

```
##   NumberOfSpecies<-numbers0(x = c(0, strsplit(x =
##                                             strsplit(x = names(temp)[3],
##                                                      split = "_")[[1]][2],
##                                           split = "[a-zA-Z]")[[1]][1]))[1]
##
##   #####HOUSEKEEPING
##   NameBaseTotal<-substr(x = names(temp)[grep(x = names(temp), pattern = paste0(NumberOfSpecies, "Tota
##                         start = 2, stop = nchar(names(temp)[grep(x = names(temp),
##                                                                  pattern = paste0(NumberOfSpecies, "T
##
##   ######START ANALYSIS
##   category<-unique(as.character(lapply(X = strsplit(x = as.character(names(temp)),
##                                                     split = paste0("_")),
##                                        function(x) x[1])))
##   category<-unique(substr(x = category, start = 2, stop = nchar(category)))
##   category<-category[!grepl(pattern = "[a-zA-Z]", x = category)]
##   category<-category[!(category %in% numbers0(c(0, (category)[1]))[1])]
##
##   temp0<-data.frame(rep_len(x = NA, length.out = nrow(temp)))
##   tempPC<-data.frame(rep_len(x = NA, length.out = nrow(temp0)))
##   tempQC<-data.frame(rep_len(x = NA, length.out = nrow(temp0)))
##
##   maxyr<-max(rownames(temp))
##   minyr<-min(rownames(temp))
##
##   category<-category00<-sort(category)
##   category.rm<-c()
##
##   for (ii in 1:length(category)){
##
##     VColumns<-grep(pattern = paste0("V", category[ii],"_"),
##                    x = substr(x = names(temp),
##                               start = 1,
##                               stop = (2+nchar(category[ii]))))
##
##     NameBasecategory<-names(temp)[grepl(pattern = NumberOfSpecies,
##                                         x = names(temp)) &
##                                     grepl(pattern = paste0("V", category[ii]), x = names(temp))]
##
##     NameBasecategory<-substr(start = 2,
##                              stop = nchar(NameBasecategory),
##                              x = NameBasecategory)
##
##     VColumns<-VColumns[!(grepl(pattern = NameBasecategory,
##                                x = names(temp)[VColumns]))]
##
##     #if there are still columns to assess that haven't been "removed"
##     # if (length(VColumns) != 0) {
##     ###Append species and category level calculations
##     temp00<-ImplicitQuantityOutput.species.cat(temp, ii=category[ii],
##                                               baseyr, maxyr, minyr,
##                                               pctmiss, warnings.list)
##
##
##     #If data for a catagory is no longer available after precentmissingthreshold etc, remove it from
##     if (sum(names(temp00[1][[1]]) %in% paste0("PI", NameBasecategory)) == 0) {
##       category.rm<-c(category.rm, ii)
##     } else {
##      temp0<-cbind.data.frame(temp0, temp00[[1]])
##     }
##
##     ###Remove duplicate columns
##     temp0<-temp0[, !(grepl(pattern = "\\.[0-9]+", x = names(temp0)))]
##
##     warnings.list<-ImplicitQuantityOutput.species.cat(temp, ii=category[ii],
##                                      baseyr, maxyr, minyr,
##                                      pctmiss, warnings.list)[[2]]
##
##
##
##   }
##
##   if(!(is.null(category.rm))) {
##     category<-category[-category.rm]
##   }
##
##   temp<-temp0#[,2:ncol(temp0)]
##
##
##     ###8. 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 speci
##     # $$VV_{t} = \sum_{s=1}^{n}(VV_{i,t})$$
##     # where:
##     # - $VV_{t}$ is the new total of $V_{i,t}$ for the entire fishery using only values for species
##
##     temp[,ncol(temp)+1]<-rowSums(temp[,grep(pattern = "VV", x = names(temp))], na.rm = T)
##     names(temp)[ncol(temp)]<-paste0("VV",NameBaseTotal)
##
##     ###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 spec
##
##     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<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##     temp <- temp[, !duplicated(colnames(temp))]
##
##
##     tempR<-data.frame(rep_len(x = NA, length.out = nrow(temp)))
##
##     for (ii in 1:length(category)){
##
##       NameBasecategory<-paste0(category[ii], "_", NumberOfSpecies)
##       NameBasecategory<-paste0(NameBasecategory,
##                                strsplit(x = names(temp)[grep(pattern = NameBasecategory,
##                                                              x = names(temp))][1], split = NumberOfS
##
##       # 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<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##       temp <- temp[, !duplicated(colnames(temp))]
##
##
##     tempR[,ii]<-data.frame(temp[,names(temp) %in% paste0("V", NameBasecategory)]/
##                              temp[,names(temp) %in% paste0("V", NameBaseTotal)])
##     names(tempR)[ii]<-paste0("R", NameBasecategory)
##
##     temp[,ncol(temp)+1]<-tempR[,ii]
##     names(temp)[ncol(temp)]<-paste0("R", NameBasecategory)
##
##     #remove duplicates
##     temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##     temp <- temp[, !duplicated(colnames(temp))]
##     temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##     temp <- temp[, !duplicated(colnames(temp))]
##
##
##     ###8. 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 $P_{i,t}$ e
##
##     # $$PC_{t} = ln(\frac{P_{t}}{P_{t-1}}) =  \sum_{i=1}^n([\frac{R_{i,t} + R_{i,t-1}}{2}] * [ln(P_{
##     tempPC[,ii]<-PriceChange(R0 = temp[, names(temp) %in%
##                                              paste0("R", NameBasecategory) &
##                                             !grepl(pattern = "REMOVED_", x = names(temp))],
##                              P0 = temp[, names(temp) %in%
##                                              paste0("PI", NameBasecategory) &
##                                             !grepl(pattern = "REMOVED_", x = names(temp))])
```

```
##
##     names(tempPC)[ii]<-paste0("PC", NameBasecategory)
##
##     #Calculate Quantity Change
##     tempQC[,ii]<-PriceChange(R0 = temp[, names(temp) %in%
##                                          paste0("R", NameBasecategory) &
##                                         !grepl(pattern = "REMOVED_", x = names(temp))],
##                              P0 = temp[, names(temp) %in%
##                                          paste0("Q", NameBasecategory) &
##                                         !grepl(pattern = "REMOVED_", x = names(temp))])
##
##     names(tempQC)[ii]<-paste0("QC", NameBasecategory)
##
##   }
##
##   temp<-cbind.data.frame(temp, tempPC)
##   temp[,ncol(temp)+1]<-rowSums(tempPC, na.rm = T)
##   names(temp)[ncol(temp)]<-paste0("PC", NameBaseTotal)
##
##   #Note if there is an Error
##   if (sum(rowSums(tempR, na.rm = T)) != nrow(temp)) {
##     warnings.list[length(warnings.list)+1]<-paste0("Warning: Rows of R_{i,t} for ",NameBaseTotal," di
##   }
##
##   ###14.  Solve Output portion of the equation for the Output Changes:
##   # $$QC = \sum_{i=1}^n((\frac{R_{it} + R_{it-1}}{2}) * ln(\frac{Q_{it}}{Q_{it-1}}))$$
##   temp<-cbind.data.frame(temp, tempQC)
##   temp[,ncol(temp)+1]<-rowSums(tempQC, na.rm = T)
##   names(temp)[ncol(temp)]<-paste0("QC", NameBaseTotal)
##
##   ###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
##   #
##   # 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
##   tempPI<-PriceIndex(temp, BaseColName = NameBaseTotal, baseyr)
##   temp[ncol(temp)+1]<-(tempPI)
##   names(temp)[ncol(temp)]<-paste0("PI", NameBaseTotal)
##
##   ### 11. 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 a
##
##   temp[,ncol(temp)+1]<-temp[,names(temp) %in% paste0("V", NameBaseTotal)]/
##     temp[, names(temp) %in% paste0("PI", NameBaseTotal)]
##   names(temp)[ncol(temp)]<-paste0("Q", NameBaseTotal)
##
##   ### 12. Total Implicit Quantity/Output Index
##     temp[,ncol(temp)+1]<-temp[,names(temp) %in% paste0("Q", NameBaseTotal)]/
##       temp[rownames(temp) %in% baseyr, names(temp) %in% paste0("Q", NameBaseTotal)]
##     names(temp)[ncol(temp)]<-paste0("QI", NameBaseTotal)
##
##
##     ### 13. Sum Total Implicit Quantity/Output Index
##     temp[,ncol(temp)+1]<-temp[,names(temp) %in% paste0("QE", NameBaseTotal)]/
##       temp[rownames(temp) %in% baseyr, names(temp) %in% paste0("QE", NameBaseTotal)]
##     names(temp)[ncol(temp)]<-paste0("QEI", NameBaseTotal)
##
##
##   #Remove Duplicate Columns
##   temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##   temp <- temp[, !duplicated(colnames(temp))]
##
##
##   ## 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:
##
##   ####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(as.character(temp0[,paste0("V", NameBaseTotal, "_Check")]),
##                      as.character(temp0[,paste0("V", NameBaseTotal)]))) != 0) {
##   warnings.list[length(warnings.list)+1]<-"Warning: When back calculated, V_t did not equal PI_t * Q
##   }
##
##
##   ####2. When back calculated, $Q_{t}$ did not equal $V_t / PI_{t}$
##   # $$Q_{i,t} = V_t / P_{i,t}$$
##
##   temp0[,(ncol(temp0)+1)]<-temp0[,paste0("V",NameBaseTotal)]/temp0[,paste0("PI",NameBaseTotal)]
##   names(temp0)[ncol(temp0)]<-paste0("Q", NameBaseTotal, "_Check")
##
##   if (length(setdiff(as.character(temp0[,paste0("Q", NameBaseTotal, "_Check")]),
##                      as.character(temp0[,paste0("Q", NameBaseTotal)]))) != 0) {
##   warnings.list[length(warnings.list)+1]<-"Warning: When back calculated, Q_t did not equal V_t/PI_t
##   }
##
##
##   ####3. When back calculated, growth rate ?
##
##   # $$ln(Q_t/Q_{t-1}) = \sum( ( \frac{R_{i, t} + R_{i, t-1}}{2})  * ln(\frac{Q_{i,t}}{Q_{i,t-1}}))$$
##
##   #Remove Duplicate Columns
##   temp<-temp[, !(grepl(pattern = "\\.[0-9]+", x = names(temp)))]
##   temp <- temp[, !duplicated(colnames(temp))]
```

```
##
##   names0<-c(paste0("Q",NameBaseTotal))
##   for (ii in 1:length(category)) {
##     names0<-c(names0,
##               names(temp)[grep(pattern = paste0("Q", category[ii], "_", NumberOfSpecies), names(temp
##               names(temp)[grep(pattern = paste0("R", category[ii], "_", NumberOfSpecies), names(temp
##   }
##
##   temp0<-temp[,unique(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(rep_len(x = NA, length.out = nrow(temp)))
##   for (ii in 1:length(category)) {
##     R0<-temp0[,grep(pattern = paste0("R", category[ii]), x = names(temp0)) ]
##     Q0<-temp0[,grep(pattern = paste0("Q", category[ii]), x = names(temp0)) ]
##
##     for (r in 2:(nrow(temp))){
##       temp00[r,ii]<-(((R0[r] + R0[r-1])/2) * ln(Q0[r] / Q0[r-1]) )
##     }
##   }
##
##   temp0[,(ncol(temp0)+1)]<-c(NA, rowSums(temp00, na.rm = T)[2:length(rowSums(temp00, na.rm = T))])
##   names(temp0)[ncol(temp0)]<-"part2"
##
##   if (length(setdiff(as.character(temp0[,"part1"]),
##                      as.character(temp0[,"part2"]))) != 0) {
##     warnings.list[length(warnings.list)+1]<-"Warning: When back calculated, ln(Q_t/Q_{t-1}) = did no
##   }
##
##
##
##   ####4. Missing Data
##
##   #value
##   a<-temp
##   a<-a[,grep(pattern = "V[1-9]+_[1-9]+", x = names(a))]
##   if(length(grep(pattern = "REMOVED_", x = names(a)) &
##             grep(pattern = "Total", x = names(a), ignore.case = T)) != 0 ){
##     a<-a[,-c(grep(pattern = "REMOVED_", x = names(a)), grep(pattern = "Total", x = names(a), ignore.
##   }
##   ncol0<-ncol(a)
##   aa<-0
##   a<-data.frame(a)
##   if(ncol(a) != 0){
##     for (iii in 1:ncol(a)) {
##     aa<-c(aa, ifelse(sum(a[,iii] %in% c(NA, NaN, 0)) == nrow(a), iii, NA))
##   }
##   vv<-(aa[!(is.na(aa))])
## } else {
##   pp<-0
## }
##   #quantity
##   a<-temp
##   a<-a[,grep(pattern = "Q[1-9]+_[1-9]+", x = names(a))]
##   if(length(grep(pattern = "REMOVED_", x = names(a)) &
##             grep(pattern = "Total", x = names(a), ignore.case = T)) != 0 ){
##     a<-a[,-c(grep(pattern = "REMOVED_", x = names(a)), grep(pattern = "Total", x = names(a), ignore.c
##   }
##   ncol0<-ncol(a)
##   aa<-0
##   a<-data.frame(a)
##   if(ncol(a) != 0){
##     for (iii in 1:ncol(a)) {
##     aa<-c(aa, ifelse(sum(a[,iii] %in% c(NA, NaN, 0)) == nrow(a), iii, NA))
##   }
##   qq<-(aa[!(is.na(aa))])
##   } else {
##     pp<-0
##   }
##   #Price
##   a<-temp
##   a<-a[,grep(pattern = "P[1-9]+_[1-9]+", x = names(a))]
##   if(length(grep(pattern = "REMOVED_", x = names(a)) &
##             grep(pattern = "Total", x = names(a), ignore.case = T)) != 0 ){
##     a<-a[,-c(grep(pattern = "REMOVED_", x = names(a)), grep(pattern = "Total", x = names(a), ignore.c
##   }
##   ncol0<-ncol(a)
##   aa<-0
##
##   a<-data.frame(a)
##   if (ncol(a) != 0) {
##    for (iii in 1:ncol(a)) {
##     aa<-c(aa, ifelse(sum(a[,iii] %in% c(NA, NaN, 0)) == nrow(a), iii, NA))
##    }
##     pp<-(aa[!(is.na(aa))])
##   } else {
##     pp<-0
##   }
##
##
##   warnings.list[length(warnings.list)+1]<-paste0("FYI: ", ifelse(length(vv)==1, 0, length(vv)-1) ,
##                                                  " of species V columns are completely empty, ",
##                                                  ifelse(length(qq)==1, 0, length(qq)-1) ,
##                                                  " of species Q columns are completely empty, and ",
##                                                  ifelse(length(pp)==1, 0, length(pp)-1) ," of ", ncol
##                                                  " species P columns are completely empty. ")
##
##
##   # ####5. Negative Numbers
##   # a0<-landings.data[idx,]
##   # a0[(a0$Dollars<0 & !(is.na(a0$Dollars))),] %>%
##   #   knitr::kable(row.names = F, booktabs = T)
##   #
##   # a0<-landings.data[idx,]
##   # a0[(a0$Pounds<0 & !(is.na(a0$Pounds))),] %>%
##   #   knitr::kable(row.names = F, booktabs = T)
##   #
##   # if (sum(temp0[,"part1"] %in% temp0[,"part2"]) != nrow(temp0))  {
##   #   warnings.list[length(warnings.list)+1]<-"When back calculated, ln(Q_t/Q_{t-1}) = did not equal
##   # }
##
##   ###########################################GRAPHS
##
##
##
##
##   #####Calculated Q by Species
##   title00<- "_QBySpecies"
##
##   a0<-data.frame(temp[,grepl(
##     pattern = paste0("Q[0-9]+_", NumberOfSpecies),
##     x = names(temp))])
##
##   a0$Year<-rownames(a0)
##
##   a <- gather(a0, Category, val, names(a0)[grepl(pattern = NumberOfSpecies, x = names(a0))], factor_
##
##   a$cat<-as.character(lapply(X = strsplit(x = as.character(a$Category), split = paste0("_", NumberOf
##
##   temp0<-a
##
##   g<-plotnlines(dat = temp0, title00, place)
##
##   figures.list[[length(figures.list)+1]]<-g
##   names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##
##   #####Summed Q By Species
##   title00<- "_QE-Species"
##
##   a0<-data.frame(temp[,grepl(
##     pattern = paste0("QE[0-9]+_", NumberOfSpecies),
##     x = names(temp))])
##
##   a0$Year<-rownames(a0)
##
##   a <- gather(a0, Category, val, names(a0)[grepl(pattern = NumberOfSpecies, x = names(a0))], factor_
##
##   a$cat<-as.character(lapply(X = strsplit(x = as.character(a$Category), split = paste0("_", NumberOf
##
##   temp0<-a
##
##   g<-plotnlines(dat = temp0, title00, place)
##
##   figures.list[[length(figures.list)+1]]<-g
##   names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##   #####Price Index
##   title00<- "_PI-Line"
```

```
##
##   a0<-data.frame(temp[,grepl(
##     pattern = paste0("PI[0-9]+_", NumberOfSpecies),
##     x = names(temp))])
##
##   a0$Year<-rownames(a0)
##
##   a <- gather(a0, Category, val, names(a0)[grepl(pattern = NumberOfSpecies, x = names(a0))], factor_
##
##   a$cat<-as.character(lapply(X = strsplit(x = as.character(a$Category), split = paste0("_", NumberOf
##
##   temp0<-a
##
##   g<-plotnlines(dat = temp0, title00, place)
##
##   figures.list[[length(figures.list)+1]]<-g
##   names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##
##   ########VV
##   title00<- "_VV-Line"
##
##   a0<-data.frame(temp[,grepl(
##     pattern = paste0("VV[0-9]+_", NumberOfSpecies),
##     x = names(temp))])
##
##   a0$Year<-rownames(a0)
##
##   a <- gather(a0, Category, val, names(a0)[grepl(pattern = NumberOfSpecies, x = names(a0))], factor_
##
##   a$cat<-as.character(lapply(X = strsplit(x = as.character(a$Category), split = paste0("_", NumberOf
##
##   temp0<-a
##
##   g<-plotnlines(dat = temp0, title00, place)
##
##   figures.list[[length(figures.list)+1]]<-g
##   names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##
##   ########V
##   title00<- "_V-Line"
##
##   a0<-data.frame(temp[,grepl(
##     pattern = paste0("V[0-9]+_", NumberOfSpecies),
##     x = names(temp))])
##   a0<-a0[,-grep(pattern = "REMOVED_", x = names(a0))]
##   a0<-a0[,-grep(pattern = "VV[0-9]+_", x = names(a0))]
##
##   a0$Year<-rownames(a0)
##
##   a <- gather(a0, Category, val, names(a0)[grepl(pattern = NumberOfSpecies, x = names(a0))], factor_
##
##   a$cat<-as.character(lapply(X = strsplit(x = as.character(a$Category), split = paste0("_", NumberOf
##
##   temp0<-a
##
##   g<-plotnlines(dat = temp0, title00, place)
##
##   figures.list[[length(figures.list)+1]]<-g
##   names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##
##
##   ########V and VV
##   title00<- "_VvVV-Line"
##
##   a0<-data.frame(temp[,grepl(
##     pattern = paste0("V[0-9]+_", NumberOfSpecies),
##     x = names(temp))])
##   a0<-a0[,-grep(pattern = "REMOVED_", x = names(a0))]
##   # a0<-a0[,-grep(pattern = "VV[0-9]+_", x = names(a0))]
##
##   a0$Year<-rownames(a0)
##
##   a <- gather(a0, Category, val, names(a0)[grepl(pattern = NumberOfSpecies, x = names(a0))], factor_
##
##   a$cat<-paste0(as.character(lapply(X = strsplit(x = as.character(a$Category), split = paste0("_", Nu
##                      "_",
##                      as.character(lapply(X = strsplit(x = as.character(a$Category), split = paste0("
##
##   temp0<-a
##
##   g<-plotnlines(dat = temp0, title00, place)
##
##   figures.list[[length(figures.list)+1]]<-g
##   names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##
##   ########VE
##   title00<- "_VE-Line"
##
##   a0<-data.frame(temp[,grepl(
##     pattern = paste0("V[0-9]+_", NumberOfSpecies),
##     x = names(temp))])
##
##   a0<-a0[,grep(pattern = "REMOVED_", x = names(a0))]
##   names(a0)<-gsub(pattern = "REMOVED_", replacement = "", x = names(a0))
##   names(a0)<-paste0("VE", substr(x = names(a0), start = 3, stop = nchar(names(a0))))
##
##   a0$Year<-rownames(a0)
##
##   temp0<-a0
##
##   temp0<-gather(temp0, cat, val,
##                 names(temp0)[1]:names(temp0)[length(names(temp0))-1],
##                 factor_key = T)
##
```

```
##   temp0$cat<-paste0(as.character(lapply(X = strsplit(x = as.character(temp0$cat),
##                                                      split = paste0("_", NumberOfSpecies)), function
##                 "_",
##                 as.character(lapply(X = strsplit(x = as.character(temp0$cat),
##                                                  split = paste0("_", NumberOfSpecies)), function(x) 
##
##
##   g<-plotnlines(dat = temp0, title00, place)
##
##   figures.list[[length(figures.list)+1]]<-g
##   names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##   #### Quantity Index Compare
##   # For comparison, let's recreate those graphs to make sure we are getting the same output:
##   title00<-"_QIvQEI-Line"
##
##   temp0<-temp
##   temp0$Year<-rownames(temp0)
##
##   temp0<-data.frame(temp0[,names(temp0) %in% c("Year",
##                                                 paste0("QI", NameBaseTotal),
##                                                 paste0("QEI", NameBaseTotal))])
##   temp0$Year<-rownames(temp)
##
##   temp0<-gather(temp0, cat, val,
##                 names(temp0)[1]:names(temp0)[length(names(temp0))-1],
##                 factor_key = T)
##
##   temp0$cat<-paste0(as.character(lapply(X = strsplit(x = as.character(temp0$cat),
##                                                      split = paste0("_", NumberOfSpecies)), function
##                     "_",
##                     as.character(lapply(X = strsplit(x = as.character(temp0$cat),
##                                                      split = paste0("_", NumberOfSpecies)), function
##
##   g<-plotnlines(dat = temp0, title00, place)
##
##   figures.list[[length(figures.list)+1]]<-g
##   names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##   #### Quantity Compare
##   title00<-"_QvQE-Line"
##   temp0<-temp
##   temp0$Year<-rownames(temp0)
##
##   temp0<-data.frame(temp0[,names(temp0) %in% c("Year",
##                                                 paste0("Q", NameBaseTotal),
##                                                 paste0("QE", NameBaseTotal))])
##   temp0$Year<-rownames(temp)
##
##   temp0<-gather(temp0, cat, val,
##                 names(temp0)[1]:names(temp0)[length(names(temp0))-1],
##                 factor_key = T)
##
##   temp0$cat<-paste0(as.character(lapply(X = strsplit(x = as.character(temp0$cat),
##                                                     split = paste0("_", NumberOfSpecies)), function
##                      "_",
##                      as.character(lapply(X = strsplit(x = as.character(temp0$cat),
##                                                     split = paste0("_", NumberOfSpecies)), function
##
##    g<-plotnlines(dat = temp0, title00, place)
##
##    figures.list[[length(figures.list)+1]]<-g
##    names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##
##    #### Revenue Share
##    title00<-"_R-Line"
##    temp0<-temp
##
##    temp0<-temp0[grep(pattern = paste0("R[0-9]+_", NumberOfSpecies), x = names(temp0))]
##    temp0$Year<-rownames(temp)
##
##    temp0<-gather(temp0, cat, val,
##                  names(temp0)[1]:names(temp0)[length(names(temp0))-1],
##                  factor_key = T)
##
##    plotnlines(dat = temp0, title00, place)
##
##    figures.list[[length(figures.list)+1]]<-g
##    names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##    #### Price Change
##    title00<-"_PC-Line"
##    temp0<-temp
##
##    temp0<-temp0[grep(pattern = paste0("PC[0-9]+_", NumberOfSpecies), x = names(temp0))]
##    temp0$Year<-rownames(temp)
##
##    temp0<-gather(temp0, cat, val,
##                  names(temp0)[1]:names(temp0)[length(names(temp0))-1],
##                  factor_key = T)
##
##    temp0$cat<-paste0(as.character(lapply(X = strsplit(x = as.character(temp0$cat),
##                                                     split = paste0("_", NumberOfSpecies)), function
##                      "_",
##                      as.character(lapply(X = strsplit(x = as.character(temp0$cat),
##                                                     split = paste0("_", NumberOfSpecies)), function
##
##    g<-plotnlines(dat = temp0, title00, place)
##
##    figures.list[[length(figures.list)+1]]<-g
##    names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##
##
##    ################Number Missing V Per Year
##    title00<- "_VNumberMissing-Line"
##
```

```
##    a0<-data.frame(temp.orig[,grepl(
##      pattern = paste0("V[0-9]+_"),
##      x = names(temp.orig)) &
##        !(grepl(
##          pattern = paste0("V[0-9]+_", NumberOfSpecies),
##          x = names(temp.orig)))])
##
##    total.no.v<-ncol(a0)*nrow(a0)
##
##    cat0<-data.frame(names0 = (names(temp.orig)[grepl(pattern = paste0("V[0-9]+_", NumberOfSpecies),
##                                                      x = names(temp.orig))]))
##
##    cat0$no<-as.character(lapply(X = strsplit(x = as.character(cat0$names0), split = "_"),
##                                     function(x) x[1]))
##
##    cat0$numberofspp<-NA
##    for (i in 1:nrow(cat0)) {
##      cat0$numberofspp[i]<-length(grep(pattern = cat0$no[i],
##                                   x = substr(x = names(a0),
##                                              start = 1,
##                                              stop = max(nchar(cat0$no))) ))
##    }
##
##
##    cat0$no<-as.character(gsub(pattern = "[a-zA-Z]", replacement = "", x = cat0$no))
##
##    cat0$catname<-as.character(lapply(X = strsplit(x = as.character(cat0$names0), split = NumberOfSpec
##                                  function(x) x[2]))
##    cat0$label<-paste0(cat0$catname, " (n=",  cat0$numberofspp,")")
##
##    # cat0$catname<-(names(temp00[[2]]))[merge(x = merge(names(temp00[[2]])]
##    # cat0$names0<-NULL
##    a<-temp.orig
##    a$Year<-rownames(a)
##    a<-a[,!grepl(pattern = NumberOfSpecies, x = names(a))]
##    a00<-gather(data = a, spp, val, names(a)[1]:names(a)[length(names(a))-1], factor_key = T)
##    a00<-a00[grep(pattern = "V", x = substr(x = a00$spp, start = 1, stop = 1)),]
##    a00$na<-0
##    a00$na[(is.na(a00$val)) | a00$val %in% 0]<-1
##    a00$no<-gsub(pattern = "[a-zA-Z]", replacement = "", x = as.character(lapply(X = strsplit(x = as.c
##                             function(x) x[1])) ) #catagory number
##    a00$x<-1
##
##    aa<-a00
##
##    a00<-merge(x = aa, y = cat0, by = "no")
##
##    #SUM
##    a<-aggregate(x = a00[,c("na", "x")],
##                 by = list("Year" = a00$Year, "Category" = a00$catname),
##                 FUN = sum, na.rm = T)
##
##    a<-rbind.data.frame(a,
##                        data.frame(Year = aggregate(x = a00$na, by = list("Year" = a00$Year), FUN = su
##                                 Category = "Total",
##                                 na = aggregate(x = a00$na, by = list("Year" = a00$Year), FUN = sum,
##                                 x = nrow(temp.orig)*ncol(temp.orig)))
##
##   a$x.perc<-a$na/a$x*100
##   a$bins<-round_any(a$x.perc, 10)
##
##   xnames<-as.numeric(paste0(a$Year))
##   xnames[!(xnames %in% seq(from = min((as.numeric(xnames))),
##                           to = max(as.numeric(xnames)),
##                           by = 10))]<-""
##
##   g<-ggplot(data = a, aes(x = factor(Year), y = na, color = Category)) +
##     geom_line(aes(group = Category)) +
##     geom_point() +
##     theme(
##       panel.grid.major.y = element_line( color=NOAALightBlue, size = .1 ),
##       panel.grid.minor.y = element_blank(),
##       panel.grid.major.x = element_blank(),
##       panel.grid.minor.x = element_blank(),
##       axis.line = element_line( color=NOAALightBlue, size = .1 ),
##       axis.ticks = element_blank(), # remove ticks
##       panel.background = element_blank()
##     )  +
##     scale_x_discrete(labels= xnames) +
##     guides(fill=FALSE) +
##     ggtitle(paste0(place, " ", title00))
##
##   figures.list[[length(figures.list)+1]]<-g
##   names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##   #################Percent Missing V
##   # title00<- "_PctMissingV_Line"
##   #
##   # g<-ggplot(data = a, aes(x = factor(Year), y = x.perc, color = Category)) +
##   #   geom_line(aes(group = Category)) +
##   #   geom_point() +
##   #   theme(
##   #     panel.grid.major.y = element_line( color=NOAALightBlue, size = .1 ),
##   #     panel.grid.minor.y = element_blank(),
##   #     panel.grid.major.x = element_blank(),
##   #     panel.grid.minor.x = element_blank(),
##   #     axis.line = element_line( color=NOAALightBlue, size = .1 ),
##   #     axis.ticks = element_blank(), # remove ticks
##   #     panel.background = element_blank()
##   #   )  +
##   #   scale_x_discrete(labels= xnames) +
##   #   guides(fill=FALSE) +
##   #   ggtitle(paste0(place, " ", title00))
##   #
##   # figures.list[[length(figures.list)+1]]<-g
##   # names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##
```

```
##   # How many V columns have X percentage data missing
##   title00<- "_VPctMissing-Bar"
##
##   a00<-data.frame(table(a[,names(a) %in% c("bins", "Category")]))
##
##   cat00<-merge(y = cat0[,c("catname", "numberofspp")],
##                x = a,
##                by.y = "catname", by.x = "Category")
##
##   cat00$label<-paste0(cat00$Category, " (n=",  cat00$numberofspp,")")
##
##   cat000<-data.frame(table(cat00[,names(cat00) %in% c("label", "bins")]))
##
##   xnames<-paste0(sort(as.numeric(paste(unique(a$bins)))), "%")
##
##   g<-ggplot(data = cat000, aes(x = factor(bins), y = Freq, fill = label)) +
##     geom_bar(stat="identity", position=position_dodge()) +
##     theme(
##       panel.grid.major.y = element_line( color=NOAALightBlue, size = .1 ),
##       panel.grid.minor.y = element_blank(),
##       panel.grid.major.x = element_blank(),
##       panel.grid.minor.x = element_blank(),
##       axis.line = element_line( color=NOAALightBlue, size = .1 ),
##       axis.ticks = element_blank(), # remove ticks
##       panel.background = element_blank()
##     )  +
##     scale_x_discrete(labels = xnames) +
##     ggtitle(paste0(place, " ", title00))
##
##   figures.list[[length(figures.list)+1]]<-g
##   names(figures.list)[length(figures.list)]<-paste0(title0, title00)
##
##
##   return(list(temp, warnings.list, figures.list))
## }
## <bytecode: 0x000000000cff09e0>
```

2.3.2 A. Import and Edit data

```
temp<-read.csv(file = paste0(dir.data, "Tornqvist Index-Calculations_OutputEx.csv"))
rownames(temp)<-temp$year
temp$year<-NULL

temp.q<-temp[,grepl(pattern = "Q", x = names(temp))]
temp.q$QE0_0Total<-rowSums(temp.q, na.rm = T)
temp.q$QE1_0Finfish<-rowSums(temp.q[,grepl(x = names(temp.q), pattern = "Q1") ], na.rm = T)
temp.q$QE2_0Shellfish<-rowSums(temp.q[,grepl(x = names(temp.q), pattern = "Q2") ], na.rm = T)

temp.v<-temp[,grepl(pattern = "V", x = names(temp))]
temp.v$V0_0Total<-rowSums(temp.v, na.rm = T)
temp.v$V1_0Finfish<-rowSums(temp.v[,grepl(x = names(temp.v), pattern = "V1") ], na.rm = T)
temp.v$V2_0Shellfish<-rowSums(temp.v[,grepl(x = names(temp.v), pattern = "V2") ], na.rm = T)
```

```
temp<-orgional.data<-cbind.data.frame(temp.q, temp.v)
```

2.3.3 B. Enter base year

```
baseyr<-baseyr
```

2.3.4 C. Run the function

```
temp00<-ImplicitQuantityOutput(orgional.data, baseyr, pctmiss)
temp<-temp00[[1]]
warnings.list0<-temp00[[2]]
figures.list0<-temp00[[3]]
```

2.3.5 D. Obtain the implicit quantity estimates

	QE0_0Total	REMOVED_V0_0Total	VV0_0Total	V0_0Total	PC0_0Total	QC0_0Total	PI0_0Total
2007	3250	5600	4700	11200	0.0000000	0.0000000	1.063375
2008	3380	6220	5000	12440	0.0188110	0.0336998	1.083567
2009	3150	5710	4800	11420	-0.0084612	-0.0343301	1.074438
2010	2610	3890	4700	7780	-0.0717975	-0.1216869	1.000000
2011	2490	4180	5000	8360	0.0545208	-0.0185252	1.056034
2012	2412	4150	4950	8300	-0.0123699	0.0087808	1.043052
2013	3251	6280	5180	12560	0.0617111	0.1459460	1.109447
2014	3431	6270	5370	12540	-0.0094442	0.0086389	1.099019
2015	3630	6700	5700	13400	0.0240843	0.0090783	1.125809
2016	3575	6780	5680	13560	-0.0126638	0.0186203	1.111642

Did all of the analyses work as intended?

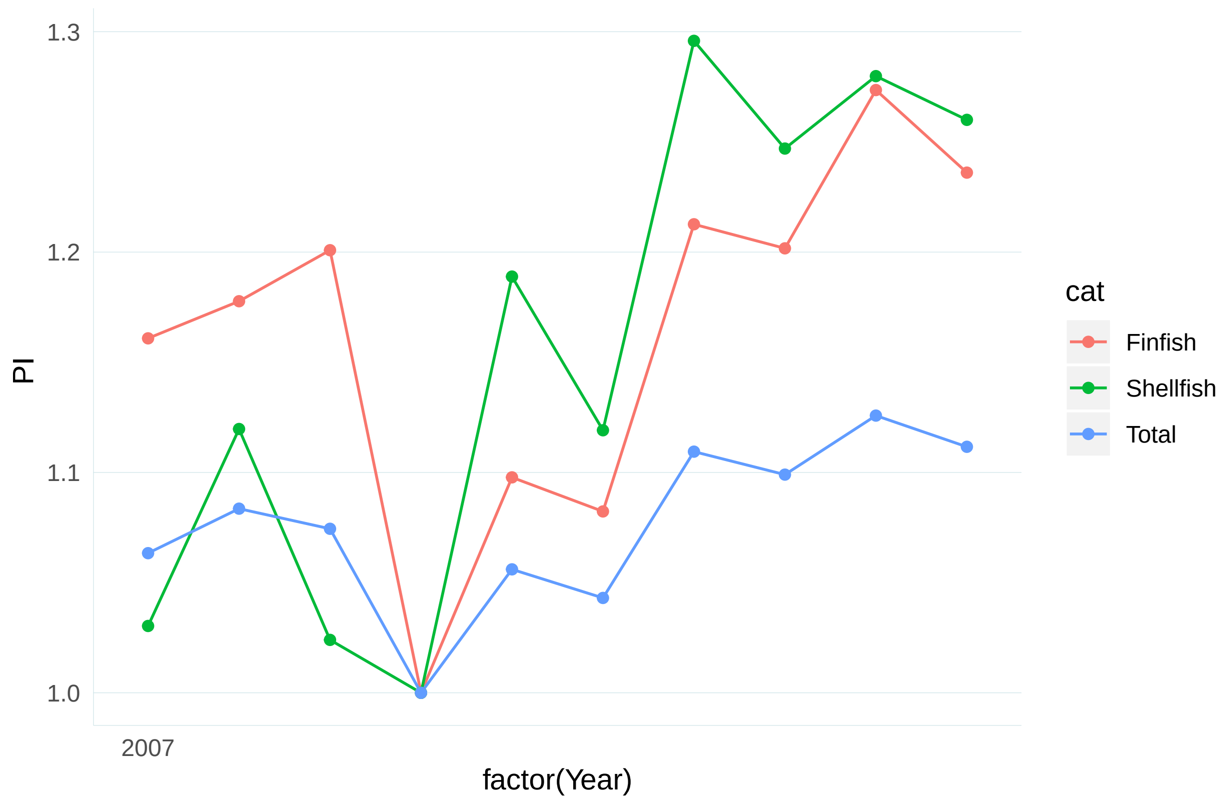
Warning: Rows of R_{i,t} for 0_0Total did not sum to 1, Warning: When back calculated, ln(Q_t/Q_{t-1}) = did not equal sum((R_{i, t} - R_{i, t-1}) / 2) x ln((Q_{i,t}) / (Q_{i,t-1})), FYI: 0 of species V columns are completely empty, 1 of species Q columns are completely empty, and 1 of 6 species P columns are completely empty.

2.3.6 E. Graph

2.3.6.1 Graph 1: Price Index

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

```
figures.list0$`_PI-Line`
```



2.3.6.2 Graph 2: Quantity Index Compare

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

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

```
## NULL
```

2.3.6.3 Graph 3: Quantity Compare

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

```
## NULL
```

2.4 Practice with real data (For National Data)

2.4.1 A. Import and Edit data

Load and subset Data

```
#Load Data (This data has been edited to include category columns)
landings.data<-read.csv(file = paste0(dir.data, "landings_edited.csv"))
landings.data<-landings.data[landings.data$Year < 2018,] #FUS 2018 hasn't been published yet
landings.data<-landings.data[landings.data$State %in% unique(state.codes$NAME),]
region<-"National"
#We'll categorize by this column I already added to the data
category0 = "category.orig"
```

Summary information about the commercial dataset:

Var1	Var2	Freq
	Tsn	Min. : 0
	Tsn	1st Qu.:160845
	Tsn	Median :167674
	Tsn	Mean :164501
	Tsn	3rd Qu.:169611
	Tsn	Max. :775091
	Tsn	NA's :98
	Year	Min. :1950
	Year	1st Qu.:1977
	Year	Median :1995
	Year	Mean :1991
	Year	3rd Qu.:2008
	Year	Max. :2017
	Year	NA
	State	West Florida :10405
	State	East Florida : 8973
	State	New York : 7106
	State	California : 6899
	State	North Carolina: 6436
	State	New Jersey : 5642
	State	(Other) :63642
	AFS.Name	FINFISH ** : 1467
	AFS.Name	OYSTER, EASTERN : 1187
	AFS.Name	SHARKS, UNCLASSIFIED **: 1169
	AFS.Name	BLUEFISH : 1103
	AFS.Name	SHAD, AMERICAN : 1083
	AFS.Name	SQUIDS ** : 1027
	AFS.Name	(Other) :102067
	Pounds	Min. : -321
	Pounds	1st Qu.: 3882
	Pounds	Median : 44779
	Pounds	Mean : 4871705
	Pounds	3rd Qu.: 483428
	Pounds	Max. :3410064761
	Pounds	NA's :11870
	Dollars	Min. : -4494
	Dollars	1st Qu.: 1739
	Dollars	Median : 21213
	Dollars	Mean : 1659774
	Dollars	3rd Qu.: 237607
	Dollars	Max. :540962350
	Dollars	NA's :12125
	category.orig	Finfish :82734
	category.orig	Other : 5683

Var1	Var2	Freq
	category.orig	Shellfish:20686
	category.orig	NA
	category.orig	NA
	category.orig	NA
	category.orig	NA

Edit/Restructure Data

```
temp00<-EditCommData(dat = landings.data, category0)
temp<-temp00[[1]]
```

	Q1_0010ALEWIFE_	Q1_0011ALFONSIN_	Q1_0014AMBERJACK_..	Q1_0015AMBERJACK_GREATE
1950	757043	NA	1955	N
1951	765521	NA	2322	N
1952	743937	NA	5299	N
1953	757242	NA	3954	N
1954	664708	NA	6601	N

2.4.2 B. Enter base year

```
baseyr<-2010
pctmiss = 0.60
```

2.4.3 C. Run the function

```
temp00<-ImplicitQuantityOutput(temp, baseyr, pctmiss)
temp<-temp00[[1]]
warnings.list0<-temp00[[2]]
figures.list0<-temp00[[3]]
```

2.4.4 D. Obtain the implicit quantity estimates

	VV_Total	V_Total	PC_Total	QC_Total	PI_Total	Q_Total	QI_Total
1950	5245879816	9818506620	0.0000000	0.0000000	2.9354973	3344750698	0.2113687
1951	4794342982	8934657980	-0.0663038	0.0190579	2.7471749	3252307684	0.2055268
1952	4793747056	8907895933	0.0210779	-0.0225525	2.8056942	3174934761	0.2006373
1953	4888308503	9081602367	-0.0186317	0.0284058	2.7539032	3297720259	0.2083966
1954	5140369062	9566498328	-0.0236380	0.0496500	2.6895697	3556888113	0.2247745
1955	5204421274	9728097996	0.0164691	-0.0082005	2.7342313	3557891451	0.2248379
1956	5666538056	10629450067	-0.0083105	0.0524076	2.7116026	3919988115	0.2477203
1957	5163140746	9623342204	-0.0045330	-0.0452452	2.6993387	3565073992	0.2252918
1958	5142667236	9627835961	-0.0211851	0.0211926	2.6427545	3643106412	0.2302230
1959	5465497637	10272214624	0.0390424	-0.0067168	2.7479746	3738103873	0.2362263
1960	5343929532	10027971626	0.0348152	-0.0468770	2.8453308	3524360542	0.2227190
1961	5609304998	10562507100	-0.0230355	0.0488581	2.7805363	3798730202	0.2400575
1962	5828252598	10967389930	-0.0117686	0.0308758	2.7480051	3991036967	0.2522102

	VV_Total	V_Total	PC_Total	QC_Total	PI_Total	Q_Total	QI_Total
1963	5286945663	9881838960	-0.0065696	-0.0453894	2.7300111	3619706541	0.2287443
1964	5002164745	9316326424	-0.0324194	0.0028470	2.6429249	3525006093	0.2227598
1965	5224509221	9766170976	-0.0327352	0.0564970	2.5578091	3818178244	0.2412865
1966	4749445680	8813234494	-0.0438320	-0.0074227	2.4481168	3600005741	0.2274993
1967	4476942557	8269157448	0.0474981	-0.0793416	2.5672035	3221075902	0.2035532
1968	4711200579	8748060092	-0.0198467	0.0478187	2.5167553	3475927982	0.2196583
1969	4796727573	8902238380	-0.0667223	0.0755962	2.3543111	3781249855	0.2389529
1970	5284412463	9874827205	-0.0447498	0.0966266	2.2512785	4386319600	0.2771898
1971	5522231846	10348401606	0.0015422	0.0220882	2.2547532	4589593970	0.2900355
1972	5560171110	9972616393	-0.0471327	0.0287390	2.1509460	4636386132	0.2929925
1973	5571865210	9998781792	-0.2261708	0.2274740	1.7155556	5828305431	0.3683148
1974	5708930416	10281482880	0.0115984	0.0022353	1.7355691	5923983717	0.3743611
1975	5647185119	10155431570	0.0457207	-0.0517524	1.8167624	5589851221	0.3532459
1976	6155108975	11171173811	-0.0592071	0.1068707	1.7123195	6524000716	0.4122787
1977	5939356150	10743224081	-0.0603779	0.0408703	1.6119925	6664562060	0.4211614
1978	6496093706	12315518540	-0.0471624	0.1153801	1.5377320	8008885091	0.5061147
1979	6799479390	12937145905	-0.0562544	0.0809134	1.4536158	8899975888	0.5624265
1980	6879957142	13117276772	-0.0013109	0.0082099	1.4517116	9035732013	0.5710055
1981	6403870540	12042459942	-0.0002584	-0.0427504	1.4513365	8297496603	0.5243533
1982	6820926341	12876531593	0.0010039	0.0323995	1.4527942	8863286616	0.5601079
1983	6653589709	12788013760	-0.0103593	0.0067382	1.4378220	8894017216	0.5620499
1984	6652182493	12800500614	-0.0111285	0.0118573	1.4219099	9002329198	0.5688946
1985	6579440056	12654922787	0.0121831	-0.0178067	1.4393391	8792175834	0.5556141
1986	6247599778	12174407959	-0.0464997	0.0271543	1.3739425	8860929666	0.5599590
1987	7127985691	13940568659	-0.0396646	0.1083039	1.3205123	10556939693	0.6671369
1988	7347571420	14689354133	-0.0808559	0.1082087	1.2179436	12060783529	0.7621710
1989	8703831791	17402616195	0.0470876	0.0381256	1.2766654	13631305975	0.8614189
1990	9757305102	19521907692	-0.0168612	0.0743144	1.2553197	15551342995	0.9827540
1991	9594907037	19181078449	-0.0035386	-0.0052351	1.2508855	15334000491	0.9690192
1992	9906052649	19814924667	-0.0712662	0.0874467	1.1648421	17010824876	1.0749847
1993	9926845179	19858598115	0.0960382	-0.0948894	1.2822595	15487191755	0.9787000
1994	10054255709	20104979305	-0.0363681	0.0425049	1.2364640	16260060804	1.0275408
1995	9627786875	19254336401	-0.0589713	0.0373212	1.1656564	16518021103	1.0438424
1996	9339533995	18676723381	0.0423855	-0.0575915	1.2161253	15357565266	0.9705084
1997	9579995371	19161323317	-0.0199672	0.0327680	1.1920835	16073810203	1.0157709
1998	8957006563	17915753877	0.0875758	-0.1211690	1.3011889	13768757497	0.8701050
1999	9065699732	18121655321	0.0041004	0.0016127	1.3065353	13870008659	0.8765035
2000	8835528763	17662325166	-0.0225632	0.0096956	1.2773857	13826931432	0.8737813
2001	9247621688	18499609739	0.0120686	0.0107818	1.2928955	14308666381	0.9042241
2002	9198719970	18408211017	0.0723838	-0.0749802	1.3899504	13243789594	0.8369301
2003	9282057328	18574250924	-0.0036968	0.0082831	1.3848215	13412740330	0.8476068
2004	9232270660	18297032435	-0.1006733	0.0913364	1.2521949	14611968704	0.9233910
2005	9129244809	18121412150	-0.0559401	0.0501519	1.1840702	15304339155	0.9671448
2006	9009775463	18184685135	-0.0454014	0.0396258	1.1315139	16071110607	1.0156003
2007	8790589502	17615535473	-0.0157499	0.0014798	1.1138322	15815250265	0.9994314
2008	7653089299	15579666375	-0.1184738	0.0479589	0.9893894	15746749108	0.9951025
2009	7563486182	15350915977	0.0767031	-0.0834419	1.0682650	14369951242	0.9080969
2010	7763910259	15824248184	-0.0660358	0.0793133	1.0000000	15824248184	1.0000000
2011	9190352844	18578445768	-0.0383585	0.1240723	0.9623678	19304932219	1.2199589
2012	9089997736	18374346162	-0.0125581	0.0072750	0.9503579	19334133974	1.2218043
2013	9211378445	18550684352	-0.0134169	0.0200177	0.9376921	19783342636	1.2501916
2014	8907330541	17982999175	0.0333766	-0.0497364	0.9695172	18548405759	1.1721508

	VV_Total	V_Total	PC_Total	QC_Total	PI_Total	Q_Total	QI_Total
2015	9169488737	18412455107	0.0097933	0.0045024	0.9790587	18806283082	1.1884472
2016	9156939746	18454164979	-0.0235885	0.0228743	0.9562344	19298787908	1.2195706
2017	9407628445	18945836668	0.0505858	-0.0369165	1.0058506	18835636824	1.1903022

Did all of the analyses work as intended?

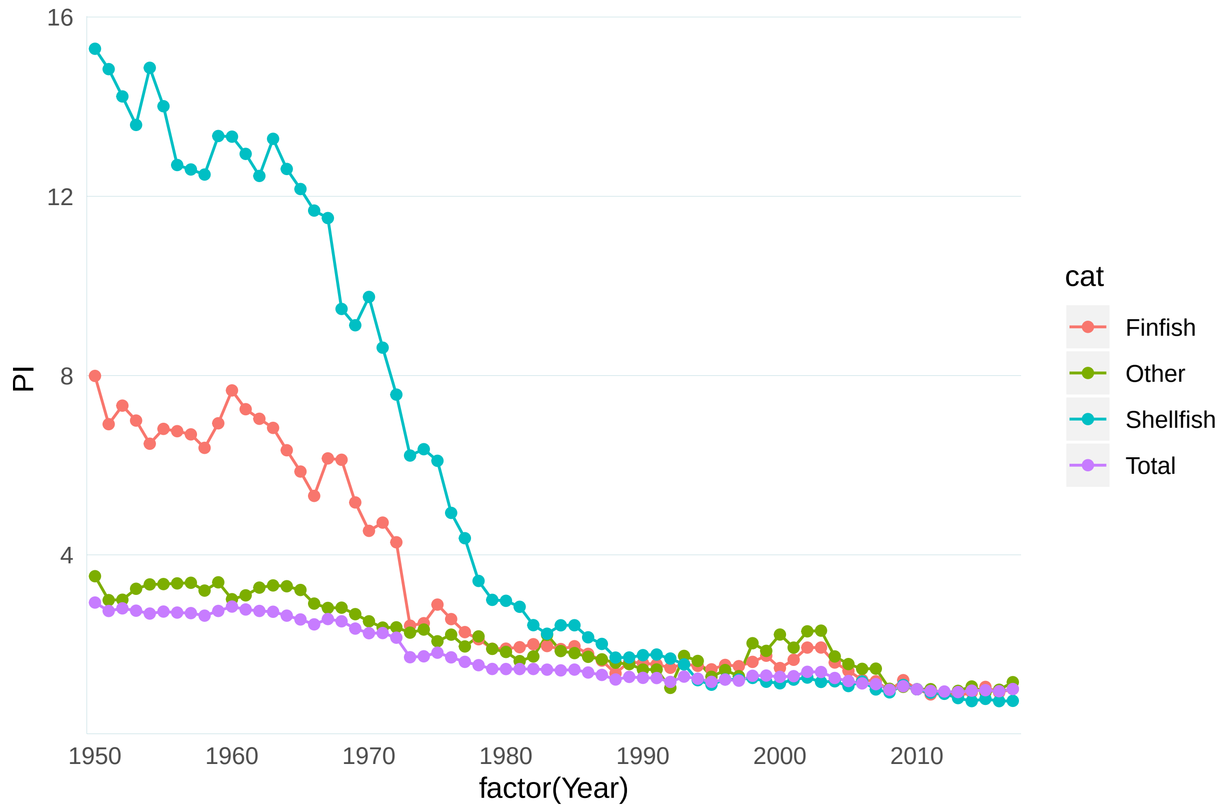
Warning: Rows of R_{i,t} for 0_0000Total did not sum to 1, Warning: When back calculated, ln(Q_t/Q_{t-1}) = did not equal sum((R_{i, t} - R_{i, t-1}) / 2) x ln((Q_{i,t}) / (Q_{i,t-1})), FYI: 32 of species V columns are completely empty, 34 of species Q columns are completely empty, and 0 of 30 species P columns are completely empty.

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`
```



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`
```

```
## NULL
```

2.4.5.3 Graph 3: Quantity Compare

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

```
## NULL
```