Macroeconomic Wavelet Decomposition

Introduction

Any given macroeconomic time series is usually composed of both long-term trends and short-term fluctuations. Understanding how these components sum up to the whole is important not only for policymakers, who are typically attempting to stabilize such fluctuations, but also to corporate firms, who might be trying to respond to these changes as part of their strategy.

Here we will explore this theme through an econometrics lens, using wavelets. Wavelets allow us to take a single time series (in this case we will use Unemployment as a macroeconomic indicator) and decompose that single line of data into its component wave parts. These decomposed parts will allow us to see, for a given level of Unemployment at a given point in time, how much that level is affected by short-, medium-, and long-term changes.

I'll start by importing some R packages that I'll need along the way. Note that most of these packages are imported purely for convenience - dplyr, reshape2, and tidyverse, for example, only provide tools that make R objects like dataframes easier to work with. They are not strictly required for this analysis. The key workhorse package here is the first one required below, which is the wavelets package:

```
require( wavelets )
require( dplyr )
require( reshape2 )
require( tidyverse )
require( lubridate )
require( forecast )
require( tidyquant )
require( data.table )
require( knitr )
require( scales )
```

For plotting later, let's also define a custom color palette

```
# define a custom color palette
custom_col_blue =  rgb( 33/255, 48/255, 67/255 , alpha = 1 )
custom_col_green =  rgb( 61/255, 159/255, 168/255 , alpha = 1 )
custom_col_yellow =  rgb( 252/255, 210/255, 82/255 , alpha = 1 )
custom_col_red = rgb( 255/255, 61/255, 75/255 , alpha = 1 )
```

Data

Next I'm going to do a manual data definition. Specifically, I'd like to define the official US recession dates, taken from the National Bureau of Economic Research (NBER):

```
business_cycles =
  # manually define the dates as given on the website above
  # note that each recession is defined by a peak (start of the recession),
  # and a trough (end of a recession):
  list(
    "Peaks" = c("1948-11-01","1953-07-01","1957-08-01",
```

```
                "1960-04-01","1969-12-01","1973-11-01",
                "1980-01-01","1981-07-01","1990-07-01",
                "2001-03-01","2007-12-01"),
    "Troughs" = c("1949-10-01","1954-05-01","1958-04-01",
                  "1961-02-01","1970-11-01","1975-03-01",
                  "1980-07-01","1982-11-01","1991-03-01",
                  "2001-11-01","2009-06-01")
  ) %>%
  as.data.frame() %>%
  # turn the dates into year-fractions, which will make plotting easier later:
  mutate(
    Peak_frac = year(Peaks) + (month(Peaks)-1)/12  ,
    Trough_frac = year(Troughs) + (month(Troughs)-1)/12   )
```

Now we want to pull some of our real data. One really nice feature of working with macroeconomic data is that the Federal Reserve has a great API, and so we'll leverage that in this work here. Specifically, all we have to do is provide the symbols of the data we want (UNRATE, for example, is Unemployment Rate). And a great feature of working with APIs is that this becomes a very automated analysis (consider the alternative, which would be something like downloading the Federal Reserve data to an Excel and importing that Excel into R - we would need to manually update the Excel each time):

```
# define the variables we want to pull from the Federal Reserve
# although for this example we'll only analyze Unemployment Rate (UNRATE),
# we'll still pull a few different time series,
# in order to illustrate the ease with which we can
# get lots of Federal Reserve data
tickers_monthly = c("UNRATE","FEDFUNDS",
                    "GS10","HSN1F",
                    "MSACSR","HOUST",
                    "HOUST1F","HOUST5F",
                    "TTLHHM156N")

# this line pulls all the data from the Federal Reserve,
# and puts it into one object automatically:
data_pull_monthly = tidyquant::tq_get(tickers_monthly,
                                      get="economic.data",
                                      from="1954-01-01")

# take a look at the data:
head( data_pull_monthly )
```

```
## # A tibble: 6 x 3
##   symbol date       price
##   <chr>  <date>     <dbl>
## 1 UNRATE 1954-01-01   4.9
## 2 UNRATE 1954-02-01   5.2
## 3 UNRATE 1954-03-01   5.7
## 4 UNRATE 1954-04-01   5.9
## 5 UNRATE 1954-05-01   5.9
## 6 UNRATE 1954-06-01   5.6
```

All of the data we've decided to pull here comes in monthly form, but as a data manipulation exercise, let's convert everything to quarterly data, by taking the average value for the three months in each quarter:

```
# convert monthly data to quarter data
data_pull_quarterly_wide <-
  data_pull_monthly %>%
  mutate(
    Year = year( date ),
    Quarter = quarter( date )
  ) %>%
  dplyr::group_by(
    symbol,
    Year,
    Quarter   ) %>%
  dplyr::summarise(
    qtrly_mean = mean(price),
    qtrly_date = first(date)   ) %>%
  dplyr::ungroup() %>%
  # convert formats from long to wide
  dcast( qtrly_date ~ symbol , value.var = 'qtrly_mean' ) %>%
  # compute year fractions, like we did for the recession dates above
  mutate( Year_Frac = year(qtrly_date) + (month(qtrly_date)-1)/12    )
```

Now we have a dataframe where each row is a quarter in time, and each column is a different variable that we pulled from the Federal Reserve.

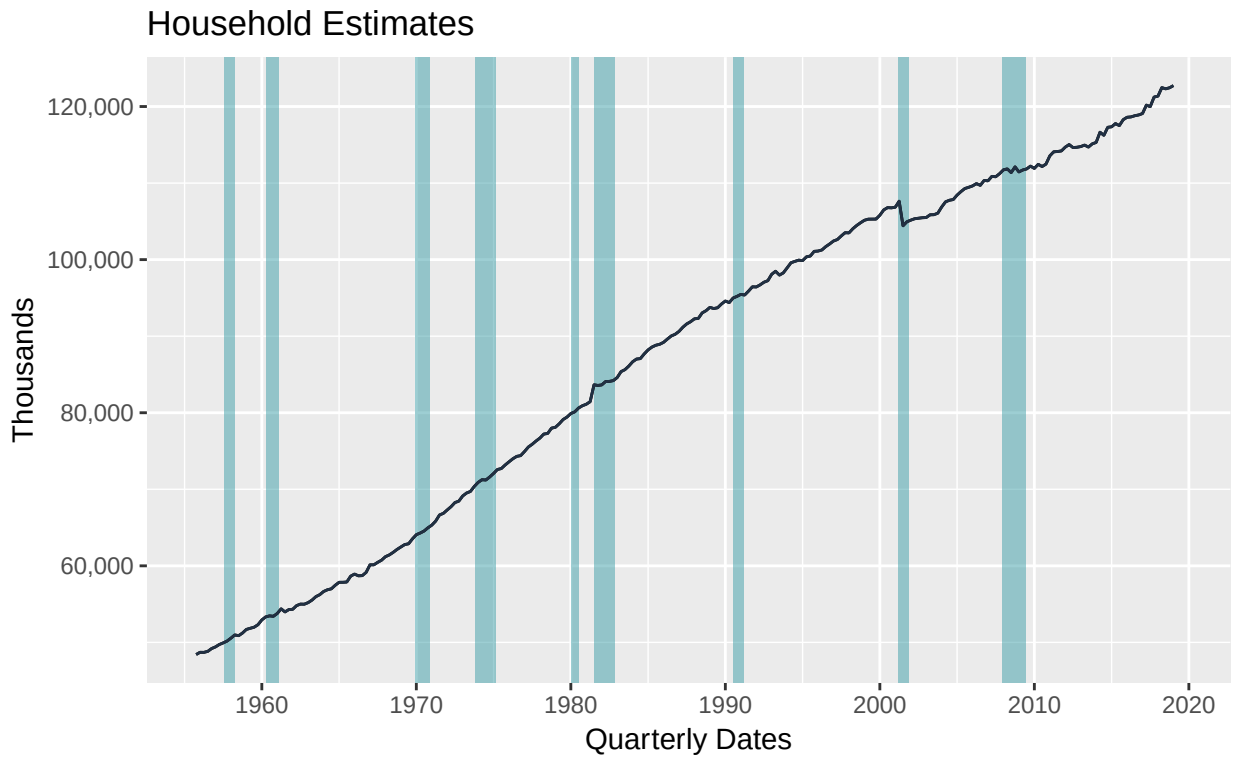
Since it's something we might do repeatedly, let's define a function that will convert these data columns into an R time series object:

```
# create a custom function that will turn a column of our dataframe into
# a time series (ts) object
TS_CONVERTER <- function(
  yearfrac_series_x = data_pull_quarterly_wide$Year_Frac ,
  time_series_y = data_pull_quarterly_wide$UNRATE
) {
  df_tmp =
    cbind( yearfrac_series_x, time_series_y ) %>%
    na.omit %>%
    as.data.frame
  start_date = min( df_tmp$yearfrac_series_x )
  end_date = max( df_tmp$yearfrac_series_x )
  ts_object = ts( df_tmp$time_series_y ,
                  start = start_date ,
                  frequency = 4   )
  return( list(
    ts_object = ts_object ,
    date_start = start_date ,
    date_end = end_date
  )  )
}
```

As a test of our new custom function, let's convert and plot the Household Estimates time series (TTLHHM156N):

```
# call the function we just defined
ts_summary = TS_CONVERTER( yearfrac_series_x = data_pull_quarterly_wide$Year_Frac,
```

```
                          time_series_y = data_pull_quarterly_wide$TTLHHM156N)

figure_1_TTLHHM156N =
  autoplot(  ts_summary$ts_object ) +
  scale_x_continuous(
    limits = c(  ts_summary$date_start , ts_summary$date_end ) ,
    breaks =  seq(1900, 3000, by = 10)
  ) +
  labs( x = "Quarterly Dates",
        y = "Thousands",
        title = "Household Estimates",
        caption = paste0("Note: U.S. Census Bureau, Household Estimates [TTLHHM156N],",
                         "\nretrieved from FRED, Federal Reserve Bank of St.",
                         "Louis;\nhttps://fred.stlouisfed.org/series/TTLHHM156N, ",
                         # note in this very line how we've automated the
                         # date in the caption of the figure
                         format(today(), "%B %d, %Y") ,  ".") ) +
  theme(plot.caption=element_text(hjust=0)) +
  scale_y_continuous(label=comma ) +
  geom_line( color = custom_col_blue )

# add some shading to indicate recessions
figure_1_recession_shading =
  geom_rect(
    data = business_cycles ,
    inherit.aes = FALSE ,
    aes( xmin = Peak_frac , xmax = Trough_frac ,
         ymin = -Inf , ymax = +Inf ),
    fill = custom_col_green,
    alpha = 0.5
  )

figure_1_TTLHHM156N$layers <- c(figure_1_recession_shading,
                                figure_1_TTLHHM156N$layers)

figure_1_TTLHHM156N
```



Note: U.S. Census Bureau, Household Estimates [TTLHHM156N], retrieved from FRED, Federal Reserve Bank of St.Louis; https://fred.stlouisfed.org/series/TTLHHM156N, December 27, 2019.

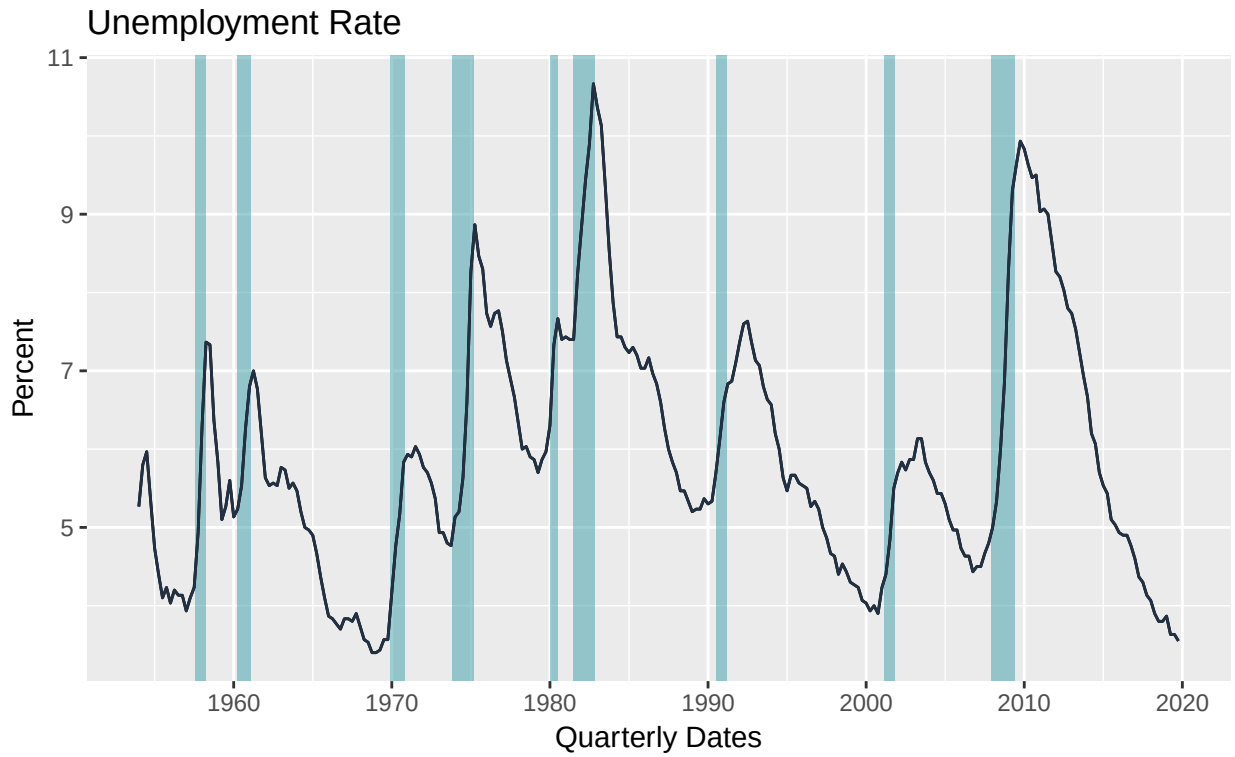
Note that this time series shows quite an increase through time. A wavelet decomposition like we are proposing to use here tends to look better with oscillating or stationary data. So, in order to make this series stationary, we take log differences:

```
data_pull_quarterly_wide =
  data_pull_quarterly_wide %>%
  mutate(
    TTLHHM156N_Lag = dplyr::lag( TTLHHM156N , 1 ) ,
    TTLHHM156N_Log_Diffs = log( TTLHHM156N ) - log( TTLHHM156N_Lag )
  )
```

Now let's also plot Unemployment Rate, since that will be the main focus of the rest of our analysis:

```
# call the function we just defined
ts_summary = TS_CONVERTER( yearfrac_series_x = data_pull_quarterly_wide$Year_Frac,
                           time_series_y = data_pull_quarterly_wide$UNRATE)

figure_2_UNRATE =
  autoplot(  ts_summary$ts_object ) +
  scale_x_continuous(
    limits = c(  ts_summary$date_start , ts_summary$date_end ) ,
    breaks =  seq(1900, 3000, by = 10)
  ) +
  labs( x = "Quarterly Dates",
        y = "Percent",
        title = "Unemployment Rate",
```

```
        caption = paste0("U.S. Bureau of Labor Statistics, Unemployment Rate [UNRATE], \n",
                         "retrieved from FRED, Federal Reserve Bank of St. Louis; \n",
                         format(today(), "%B %d, %Y") ,  ".") ) +
  theme(plot.caption=element_text(hjust=0)) +
  scale_y_continuous(label=comma ) +
  geom_line( color = custom_col_blue )


# add some shading to indicate recessions
figure_2_recession_shading =
  geom_rect(
    data = business_cycles ,
    inherit.aes = FALSE ,
    aes( xmin = Peak_frac , xmax = Trough_frac ,
         ymin = -Inf , ymax = +Inf ),
    fill = custom_col_green,
    alpha = 0.5
  )

figure_2_UNRATE$layers <- c(figure_2_recession_shading,
                            figure_2_UNRATE$layers)

figure_2_UNRATE
```



U.S. Bureau of Labor Statistics, Unemployment Rate [UNRATE], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/UNRATE, December 27, 2019.

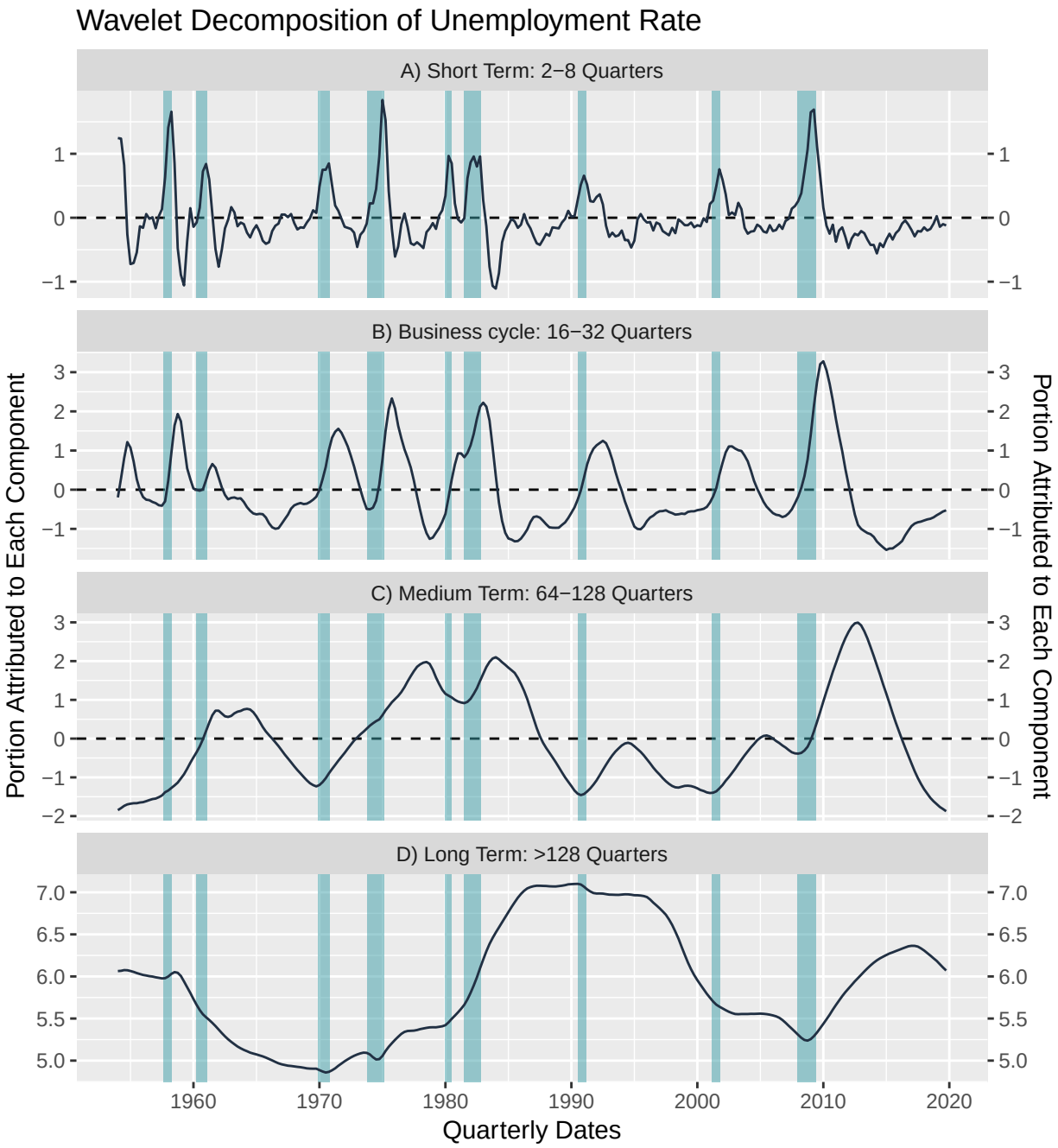
Analysis

Now we can get started with actually performing the wavelet decomposition. We'll use the R package to make it easier: we can simply provide the time series, and ask for the function to decompose it into 6 levels:

```
num_wavelet_levels = 6

wavelet_decomposition = modwt(  ts_summary$ts_object,
                                filter="haar",
                                n.levels=num_wavelet_levels,
                                boundary="periodic",
                                fast=TRUE  )

# extract the W object and immediately turn it into a dataframe
# each column in this dataframe corresponds to one of the wavelet levels
# for a total of 6 levels in this case,
# and each row corresponds to a point in time
Wdf = data.frame(wavelet_decomposition@W)
Wdf_time_series <- ts(Wdf, start=ts_summary$date_start, frequency=4)
component_short <- ts(rowSums(Wdf[,1:2]), start=ts_summary$date_start, frequency=4)
component_cycle <- ts(rowSums(Wdf[,3:4]), start=ts_summary$date_start, frequency=4)
component_medium <- ts(rowSums(Wdf[,5:6]), start=ts_summary$date_start, frequency=4)

# this is the portion not explained, or "left over from" the wavelet decomposition:
component_long <- ts_summary$ts_object-rowSums(Wdf)

wavelet_time_series = ts.union(component_short,
                               component_cycle,
                               component_medium,
                               component_long  )
```

```
plot_ts <- window( wavelet_time_series,
                   start=ts_summary$date_start )

plot_df <-
  data.frame(dates=time(plot_ts), plot_ts)  %>%
  melt( id.vars = c( "dates" ) ) %>%
  # labels for series
  mutate(
    series_id =
      case_when(variable=="component_short" ~ "A) Short Term: 2-8 Quarters",
                variable=="component_cycle" ~ "B) Business cycle: 16-32 Quarters",
                variable=="component_medium" ~ "C) Medium Term: 64-128 Quarters",
                T ~ "D) Long Term: >128 Quarters"  )
  )



# create the plot object
ggplot(data=plot_df, aes(x=dates, y=value))+
  # wrap the plots to create sub-windows:
  facet_wrap(~series_id, ncol=1, scales="free_y")+
  # for all wavelets except the last one, mark the y axis at 0:
```

```
geom_hline(aes(yintercept=ifelse(variable=="component_long",NA,0)), linetype=2)+
# add the recession data just like before:
geom_rect(
  data = business_cycles ,
  inherit.aes = FALSE ,
  aes( xmin = Peak_frac , xmax = Trough_frac ,
       ymin = -Inf , ymax = +Inf ),
  # fill = 'darkgray' ,
  fill = custom_col_green,
  alpha = 0.5
) +
# add the decomposed wavelet lines:
geom_line(size=0.5, color = custom_col_blue)  +
scale_y_continuous(sec.axis=dup_axis())  +
scale_x_continuous(
  limits = c(  min( plot_df$dates) , max(plot_df$dates) ) ,
  breaks =  seq(1900, 3000, by = 10)
) +
labs(
  # subtitle=plot_subtitle,
  title="Wavelet Decomposition of Unemployment Rate",
  x="Quarterly Dates",
  y="Portion Attributed to Each Component"
)  +
theme(plot.caption=element_text(hjust=0))
```



Discussion

Now we're able to actually look at the results and interpret them.

Panel D of the decomposition shows the very low frequency movements in Unemployment Rate (>128 quarters, or 32 years). From the 90's to the 2000's, this component of Unemployment was gradually decreasing. But then rose sharply as a result of the Financial Crisis in 2008, and does not appear to have recovered.

The medium term (8-32 years) shows a lot more movement, and here we some of the post-recession recovery, but it is still interesting to note that the series only became negative (which is a good thing, considering we

want unemployment to be minimized) around 2016.

Now as we look at the business cycle component (Panel B: 4-8 years) we would expect to see quicker responses to actual business conditions. And as expected, we see Unemployment increase drastically during the shaded Financial Crisis recessionary period, and then peak and start to recover immediately following.

The short term component is very active and can largely be considered to be "noise".

The key takeaway here is the consideration of the business cycle component. In other words, if the short term component is just noise, and the medium- and long-term components encompass larger trends that policymakers and business leaders have less control over, then the business cycle component is of the most relevance to those making strategy decisions.

With that framework set, it's very interesting to note that, while Unemployment Rate data continues to appear strong and decreasing (see the original time series plot above), the business cycle wavelet component in Panel B appears to have already "turned" and is starting to increase. This could potentially be a leading indicator of macroeconomic weakness, which typically leads to decreased consumer spending.