The University of Queensland
Australia
Create Change

ECON3350 - Applied Econometrics for Macroeconomics and Finance

Tutorial 7: Dynamic Relationships

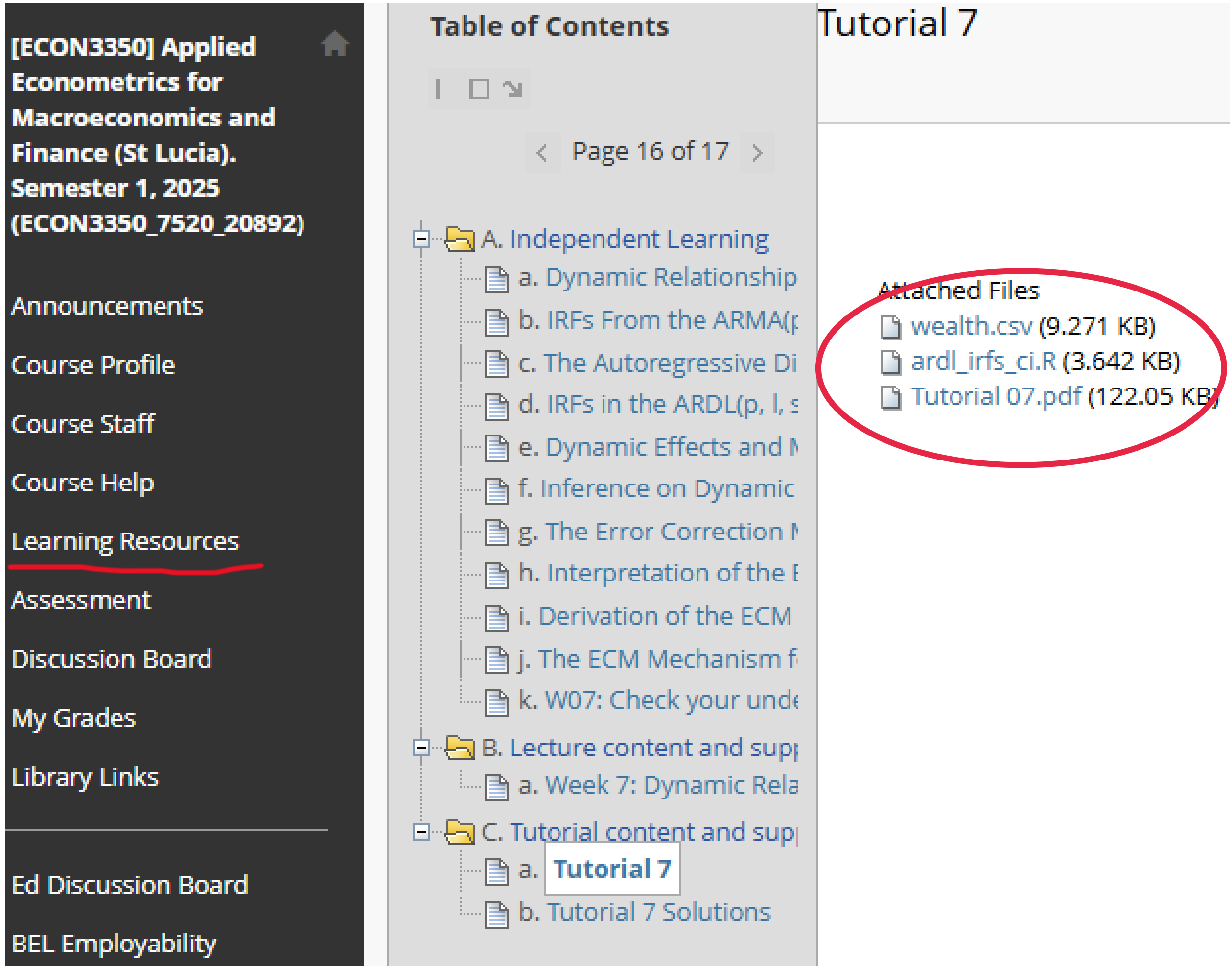
Tutor: Francisco Tavares Garcia

Tutorial 7: Dynamic Relationships

At the end of this tutorial you should be able to:

- derive the ECM representation of an ARDL(p, l, s) model;
- create a function in R;
- estimate IRFs to permanent and one-off shocks as well as LRMs;
- construct confidence intervals for IRFs and LRMs;
- Select and adequate set of ARDL models and draw inference on dynamic relationships.

Let's download the tutorial and the dataset.



Now, let's download the script for the tutorial.

- Copy the code from Github,
  - https://github.com/tavaresgarcia/teaching
- Save the scripts in the same folder as the data.

1. Derive the ECM representation of the following ARDL(1, 1, 2) model:

$$y_t = a_0 + a_1 y_{t-1} + b_0 x_t + b_1 x_{t-1} + c_0 w_t + c_1 w_{t-1} + c_2 w_{t-2} + \epsilon_{y,t}.$$

Which parameter(s) in the resulting ECM are long-run multiplier(s) and adjustment parameter(s)?

Two large names:

Autoregressive Distributed Lag Model (ARDL or ADL)

Error Correction Model (ECM)

One Large equation:

$$y_t = a_0 + a_1 y_{t-1} + b_0 x_t + b_1 x_{t-1} + c_0 w_t + c_1 w_{t-1} + c_2 w_{t-2} + \epsilon_{y,t}$$

THE UNIVERSITY OF QUEENSLAND AUSTRALIA

Let's understand the ARDL first:

Solution Remember the regression equation?

$$y_t = a_0 + b_0 x_t + \epsilon_{y,t},$$

the simplest Distributed Lag equation adds one lag for x:

$$y_t = a_0 + b_0 x_t + b_1 x_{t-1} + \epsilon_{y,t}.$$

Now, remember our autoregressive function? It has one lag of the dependent variable in the equation too:

$$y_t = a_0 + a_1 y_{t-1} + \epsilon_{y,t}.$$

So an Distributed Lag model which is also autoregressive should look like an ARDL(1,1):

$$y_t = a_0 + a_1 y_{t-1} + b_0 x_t + b_1 x_{t-1} + \epsilon_{y,t}.$$

Finally, let's add another independent variable with 2 of its lags to our regression. We have an ARDL(1,1,2):

$$y_t = a_0 + a_1 y_{t-1} + b_0 x_t + b_1 x_{t-1} + c_0 w_t + c_1 w_{t-1} + c_2 w_{t-2} + \epsilon_{y,t}.$$

THE UNIVERSITY OF QUEENSLAND AUSTRALIA

Now, let's understand the ECM:

ECMs are useful for estimating both short-term and long-term effects of one time series on another.

The error correction representation of an ARDL(1,1,2) is a reorganization of the equation where we isolate some important coefficients:

$$\Delta y_t = \gamma + \underbrace{\alpha}_{\text{Speed}}(y_{t-1} - \underbrace{\mu - \beta_x x_{t-1} - \beta_w w_{t-1}}_{\text{Long-run}}) + \underbrace{b_0 \Delta x_t + c_0 \Delta w_t - c_2 \Delta w_{t-1}}_{\text{Short-run}} + \epsilon_t,$$

where

$$\alpha = -(1 - a_1), \qquad \text{Speed of adjustment parameter,}$$

$$\beta_x = \frac{b_0 + b_1}{1 - a_1}, \qquad \text{Long-run multiplier of } x,$$

$$\beta_w = \frac{c_0 + c_1 + c_2}{1 - a_1}, \qquad \text{Long-run multiplier of } w,$$

$$\mu = \frac{a_0 - \gamma}{1 - a_1} \qquad \text{Long-run constant.}$$

To convert this to the ECM representation, we add and subtract five terms: γ, y_{t-1}, b_0x_{t-1}, c_0w_{t-1} and c_2w_{t-1}. Specifically,

$$\begin{aligned} y_t = {} & a_0 + \gamma - \gamma + y_{t-1} - y_{t-1} + a_1y_{t-1} + b_0x_t + b_0x_{t-1} - b_0x_{t-1} + b_1x_{t-1} \\ & + c_0w_t + c_0w_{t-1} - c_0w_{t-1} + c_1w_{t-1} + c_2w_{t-1} - c_2w_{t-1} + c_2w_{t-2} + \epsilon_t, \end{aligned}$$

$$\begin{aligned} y_t - y_{t-1} = {} & \gamma - y_{t-1} + a_1y_{t-1} + (a_0 - \gamma) + b_0x_{t-1} + b_1x_{t-1} + c_0w_{t-1} + c_1w_{t-1} + c_2w_{t-1} \\ & + b_0x_t - b_0x_{t-1} + c_0w_t - c_0w_{t-1} - c_2w_{t-1} + c_2w_{t-2} + \epsilon_t, \end{aligned}$$

$$\begin{aligned} y_t - y_{t-1} = {} & \gamma - (1 - a_1)y_{t-1} + (a_0 - \gamma) + (b_0 + b_1)x_{t-1} + (c_0 + c_1 + c_2)w_{t-1} \\ & + b_0(x_t - x_{t-1}) + c_0(w_t - w_{t-1}) - c_2(w_{t-1} - w_{t-2}) + \epsilon_t, \end{aligned}$$

$$\begin{aligned} \Delta y_t = {} & \gamma - (1 - a_1)\left(y_{t-1} + \left(\frac{a_0 - \gamma}{1 - a_1}\right) + \left(\frac{b_0 + b_1}{1 - a_1}\right)x_{t-1} + \left(\frac{c_0 + c_1 + c_2}{1 - a_1}\right)w_{t-1}\right) \\ & + b_0\Delta x_t + c_0\Delta w_t - c_2\Delta w_{t-1} + \epsilon_t, \end{aligned}$$

$$\Delta y_t = \gamma + \alpha(y_{t-1} - \mu - \beta_x x_{t-1} - \beta_w w_{t-1}) + b_0\Delta x_t + c_0\Delta w_t - c_2\Delta w_{t-1} + \epsilon_t.$$

Speed Long-run Short-run

2. Create a function in R to compute coefficients $\theta_0, \ldots, \theta_h$ in

$$\theta(L) = b(L)/a(L) = \theta_0 + \theta_1 L + \cdots + \theta_h L^h + \cdots ,$$

where $a(L) = a_0 + a_1 L + \cdots + a_p L^p$ and $(L) = b_0 + b_1 L + \cdots + b_q L^q$.

IRFs From the ARMA(p, q)

Suppose y_t is an ARMA(p, q) process: $a(L)y_t = a_0 + b(L)\varepsilon_t$.

Using polynomial expansion and division, $\theta(L) = b(L)/a(L)$ can be decomposed as:

$$\theta(L) = \frac{b(L)}{a(L)} = 1 + \theta_1 L + \cdots + \theta_h L^h + \beta(L)L^{h+1} + \alpha(L)\frac{b(L)}{a(L)}L^{h+1},$$

where

- h is an integer greater than or equal to 0;
- $\theta_1, \ldots, \theta_h$ are obtained recursively using the method of undetermined coefficients;
- $\beta(L)$ is a polynomial of degree $q + h$;
- $\alpha(L)$ is a polynomial of degree $(p - 1)(h + 1)$;
- dividing $(1 - \alpha(L))/a(L)$ produces a polynomial of degree ph.

2. Create a function in R to compute coefficients $\theta_0, \ldots, \theta_h$ in

$$\theta(L) = b(L)/a(L) = \theta_0 + \theta_1 L + \cdots + \theta_h L^h + \cdots ,$$

where $a(L) = a_0 + a_1 L + \cdots + a_p L^p$ and $(L) = b_0 + b_1 L + \cdots + b_q L^q$.

Solution An example of such a functions as follows.

```r
arma2ma <- function(a, b, h)
{
  # we always start here
  theta0 <- b[1] / a[1]

  if (h == 0)
  {
    # if the horizon is zero, then just return theta0
    # note: good practice would be to check that h is an integer
    return(theta = theta0)
  }

  # get the orders of a(L) and b(L); in fact, the ARMA orders
  # are p - 1 and q - 1 because we also have a0 and b0 to take
  # into account
  p <- length(a)
  q <- length(b)
```

```r
  # augment the AR and MA coefficients vectors to match the
  # number of thetas we are going to compute -- this makes
  # things easier later
  if (h > p)
  {
    a = c(a, numeric(1 + h - p))
  }

  if (h > q)
  {
    b = c(b, numeric(1 + h - q))
  }

  # allocate space for 1 + h thetas and set theta0 = b0 / a0
  theta <- c(theta0, numeric(h))
  for (j in 1:h)
  {
    theta[1 + j] <- (b[1 + j] - sum(a[2:(1 + j)]
                                * theta[j:1])) / a[1]
  }

  return(theta)
}
```

3. Create a function in R to compute IRFs (to both one-off and permanent shocks) up to horizon h as well as the LRMs for the ARDL(p, l, s):

$$a(L)y_t = a_0 + b(L)x_t + c(L)w_t + \epsilon_{y,t}.$$

Solution An example of such a function is as follows.

```
ardl_irfs <- function(ardl_est, h = 40, cumirf = T)
{
  # extract the lag orders and coefficient estimates from the
  # estimated ARDL
  order <- ardl_est$order
  coefficients <- ardl_est$coefficients

  # extract the autoregressive coefficients and construct a(L)
  j <- 1 + order[1]
  a <- c(1, -coefficients[2:j])

  # get the number of exogenous variables in the ARDL: we want to
  # get IRFs to each one of these separately
  k <- length(order) - 1

  # allocate space for all the IRFs
  irfs <- matrix(nrow = 1 + h, ncol = k)
  colnames(irfs) <- rep("", k)

  # allocate space for LRMs
  lrm <- numeric(k)
  names(lrm) <- rep("", k)

  # now, cycle through each exogenous variable and compute
  # IRFs/LRMs
  for (i in 1:k)
  {
    # advance the index to where the estimated coefficients are
    # stored in the variable "coefficients", then extract them
    # and construct b(L) for the ith exogenous variable in the
    # ARDL
    j0 <- 1 + j
    j <- j0 + order[1 + i]
    b <- coefficients[j0:j]
    colnames(irfs)[i] <- names(coefficients[j0])
    names(lrm)[i] <- names(coefficients[j0])

    if (cumirf)
    {
      # compute the first "h" terms of theta(L) = b(L)/a(L) if
      # cumulative IRFs are requested, do a cumulative sum of
      # theta coefficients
      irfs[, i] <- cumsum(arma2ma(a, b, h))
    }
    else
    {
      # compute the first "h" terms of theta(L) = b(L)/a(L)
      # and save them
      irfs[, i] <- arma2ma(a, b, h)
    }
    lrm[i] <- sum(b) / sum(a)
  }

  return(list(irfs = irfs, lrm = lrm))
}
```

4. The file `wealth.csv` contains observations on:

- c_t: the log of total real per capita expenditures on durables, nondurables and services;
- a_t: the log of a measure of real per capita household net worth (including all financial and household wealth); and
- y_t: the log of after-tax labour income.

The sample period from 1952Q2 through 2006Q2 (see Koop, G., S. Potter and R. W. Strachan (2008) "Re-examining the consumption-wealth relationship: The role of uncertainty" *Journal of Money, Credit and Banking*, Vol. 40, No. 2.3, 341-367).

	A	B	C	D
1	obs	AT	CT	YT
2	1952Q2	3.73943	2.036684	1.931234
3	1952Q3	3.753401	2.036757	1.944872
4	1952Q4	3.749103	2.066671	1.97353
5	1953Q1	3.744504	2.074874	1.987001
6	1953Q2	3.741456	2.077342	1.999325
7	1953Q3	3.73228	2.070411	1.99097
8	1953Q4	3.732246	2.059009	1.978606
9	1954Q1	3.736681	2.058456	1.965747
10	1954Q2	3.743897	2.066826	1.964638
11	1954Q3	3.758827	2.075413	1.967731
12	1954Q4	3.788056	2.09108	1.98796
13	1955Q1	3.797435	2.108962	1.993736
14	1955Q2	3.804548	2.12369	2.016339
15	1955Q3	3.826077	2.131416	2.032845
16	1955Q4	3.845266	2.138801	2.043807
17	1956Q1	3.851497	2.136107	2.049101
18	1956Q2	3.873904	2.135369	2.057179
19	1956Q3	3.865797	2.132954	2.057052
20	1956Q4	3.859665	2.141506	2.068752
21	1957Q1	3.870679	2.144099	2.065481
22	1957Q2	3.860149	2.141664	2.066298
23	1957Q3	3.870434	2.145319	2.065232
24	1957Q4	3.850632	2.141337	2.057061

(a) Estimate an ARDL$(1, 2, 2)$ specified for c_t and use the functions created in Questions 2 and 3 to obtain the estimated IRFs to permanent shocks in a_t and y_t as well the LRMs. Hint: to estimate the ARDL parameters, try the `ardl` function that is provided by the `ARDL` package.

Solution We will need to install and load the `ARDL` library in R.

```
library(ARDL)
```

Next, load the data and use the `ardl` function to estimate the parameters of the an ARDL$(1, 1, 2)$.

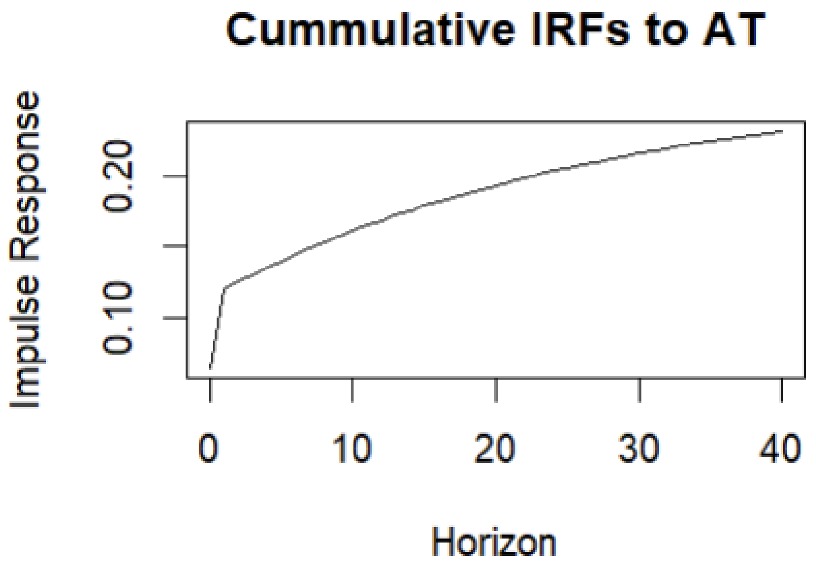
```
mydata <- read.delim("wealth.csv", header = TRUE,  sep = ",")
ardl_est <- ardl(CT ~ AT + YT, mydata, order = c(1, 2, 2))
```

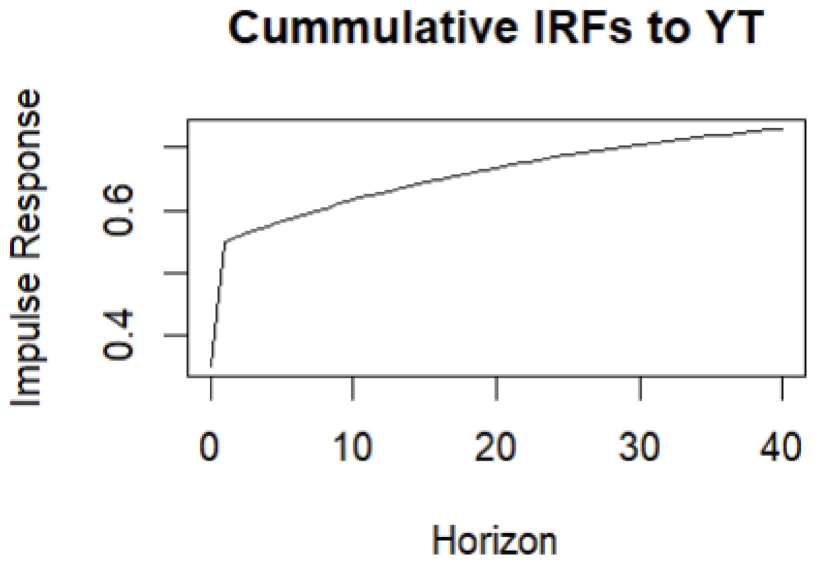
Finally, use our `ardl_irfs` function to compute the cumulative IRFs and LRMs.

```
irfs_lrm <- ardl_irfs(ardl_est)
for (i in 1:ncol(irfs_lrm$irfs))
{
  plot(0:40, irfs_lrm$irfs[, i], type = "l",
       ylab = "Impulse Response", xlab = "Horizon",
       main = paste("Cummulative IRFs to",
                    colnames(irfs_lrm$irfs)[i]))
}
```

```
print(irfs_lrm$lrm)
```

```
> print(irfs_lrm$lrm)
       AT        YT
0.2687276 0.7873063
```





(b) Estimate the ECM representation of the ARDL(1, 2, 2) and report the results. How do the LRMs in the estimated ECM compare to those computed in part (a)? Hint: use the `recm` and `multipliers` functions to convert the output produced by `ardl`.

Solution Given output from the `ardl` function, obtaining the ECM form entails two steps. First, use the `recm` function with the `case = 2` option. This option corresponds to "no linear trend" and an intercept within the equlibrium relation ($\gamma = 0$ and μ unrestricted in our notation).[1]

The function `recm` will produce estimates of the "short-run" coefficients in the ECM, but not the LRMs (`recm` only produces the aggregate *error correction term*, which it calls `ect`). To obtain estimates of the LRMs (i.e., details of the `ect`), use the `multipliers` function.

```
ecm_sr <- recm(ardl_est, case = 2)
ecm_lrm <- multipliers(ardl_est)
print(summary(ecm_sr))
print(ecm_lrm)
```

```
> print(summary(ecm_sr))

Time series regression with "zooreg" data:
Start = 3, End = 217

Call:
dynlm::dynlm(formula = full_formula, data = data, start = start,
    end = end)

Residuals:
       Min         1Q     Median         3Q        Max 
-0.0230749 -0.0039518  0.0003092  0.0034368  0.0156863 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
d(AT)       0.063151   0.017998   3.509 0.000551 ***
d(L(AT, 1))  0.049423   0.017916   2.759 0.006318 ** 
d(YT)       0.351155   0.041389   8.484 3.90e-15 ***
d(L(YT, 1))  0.183718   0.043249   4.248 3.24e-05 ***
ect        -0.035626   0.007755  -4.594 7.50e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.005683 on 210 degrees of freedom
  (0 observations deleted due to missingness)
Multiple R-squared:  0.6333,	Adjusted R-squared:  0.6246 
F-statistic: 72.54 on 5 and 210 DF,  p-value: < 2.2e-16

> print(ecm_lrm)
         Term   Estimate Std. Error   t value     Pr(>|t|)
1 (Intercept) -0.4683080  0.2067787 -2.264778 2.456199e-02
2          AT  0.2687276  0.1213092  2.215228 2.783439e-02
3          YT  0.7873063  0.1407531  5.593525 6.989283e-08
```

(c) Use the function `ardl_irfs_ci` that is provided in the file `ardl_irfs_ci.R` to construct 68% confidence intervals for the IRFs obtained in part (a).

The function `ardl_irfs_ci` takes the following inputs:

- `ardl_est`: this is the output of `ardl`;
- `h`: the maximum IRF horizon (default is 40);
- `cumirf`: whether to compute IRFs to a permanent shock (default is `TRUE`);
- `conf`: the confidence level of the intervals (default is 0.95).

It returns the following outputs:

- `lb`: an $h \times k$ matrix of lower-bounds for confidence intervals;
- `md`: an $h \times k$ matrix of mid-points for confidence intervals;
- `ub`: an $h \times k$ matrix of upper-bounds for confidence intervals.

Note that k is the number of independent variables in the ARDL, so that column j of `lb`, `md` and `ub` is related the confidence intervals for IRFs to a shock in the jth independent variable.

(c) Use the function `ardl_irfs_ci` that is provided in the file `ardl_irfs_ci.R` to construct 68% confidence intervals for the IRFs obtained in part (a).

Solution Include the functions provided in the file `ardl_irfs_ci.R`.

```
source("ardl_irfs_ci.R")
```

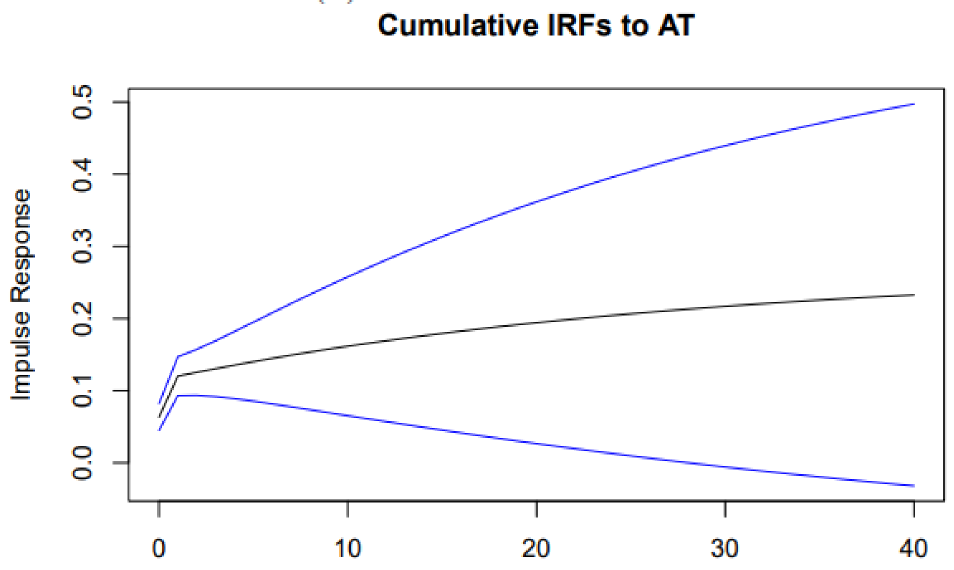
Use the function `ardl_irfs_ci` to estimate confidence intervals for IRFs to permanent shocks in a_t and y_t.

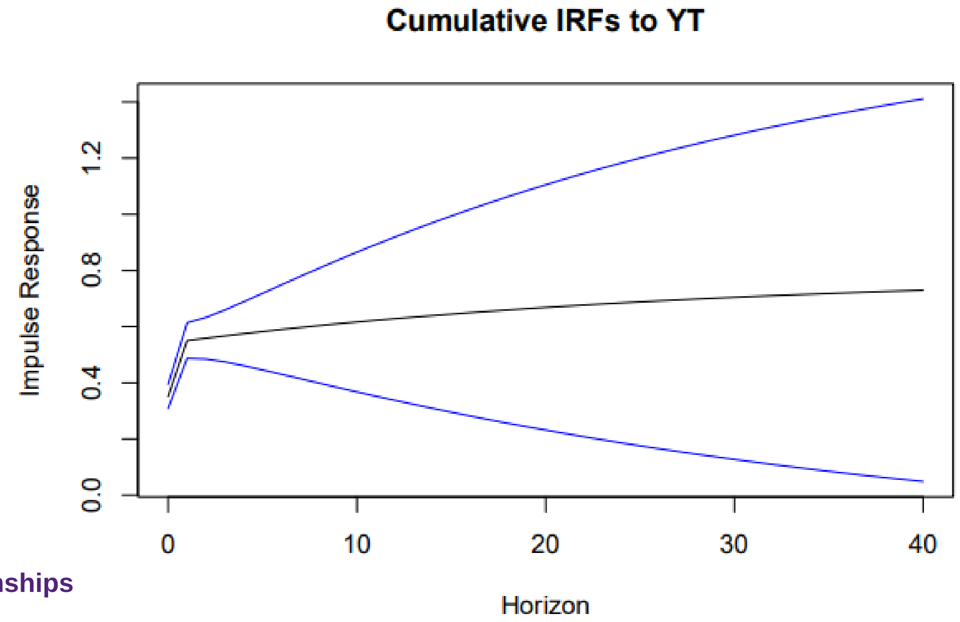
```
irfs_ci <- ardl_irfs_ci(ardl_est, conf = 0.68)
```

Here, we choose 68% as the confidence level. This corresponds to an interval that is approximately two standard deviations in width (in fact, it is exactly that if the sampling distribution of the estimator is normal). You should try other confidence levels to see how the width of the interval changes. In doing so, think about how "confidence level'' relates to *risk* in decisions that would be made based on inference from these confidence intervals.

The best way to examine estimated IRFs is to plot them.

```
for (i in 1:ncol(irfs_ci$md))
{
  plot(0:40, irfs_ci$md[, i], type = "l",
       ylab = "Impulse Response", xlab = "Horizon",
       main = paste("Cumulative IRFs to",
                              colnames(irfs_ci$md)[i]),
       ylim = c(min(irfs_ci$lb[, i]), max(irfs_ci$ub[, i])))
  lines(0:40, irfs_ci$ub[, i], type = "l", col = "blue")
  lines(0:40, irfs_ci$lb[, i], type = "l", col = "blue")
}
```





(c) Use the function `ardl_irfs_ci` that is provided in the file `ardl_irfs_ci.R` to construct 68% confidence intervals for the IRFs obtained in part (a).

We first observe that the IRF confidence intervals increase in width as the horizon h increases. This is equally true for shocks in both a_t and y_t. This is not surprising—because we are analysing "cumulative" responses, we expect that responses at more distant horizons are more difficult to estimate accurately with the same sample size.

In this particular case, we also observe that our estimate of a_1 is close to 1, which means the estimated $a(L)$ has a root close to unity, and the estimated ARDL is close to unstable. Hence, we are drawing inference on realised ARDLs that are "close" to the instability region.

Recall that an unstable ARDL does not have finite LRMs. When realised ARDLs are "close" to this, but still stable, they will result in finite LRMs. However, the sampling variance of these LRMs will be very large, thereby resulting in very wide confidence intervals. Our cumulative IRFs tend towards such LRMs as the horizon increases, and so, it is not surprising that the confidence intervals become wider at more distant horizons.

The bottom line is that when an ARDL is estimated close to instability, cumulative IRFs at longer horizons are not reliable. What this means in practical terms is that certain dynamic effects are very difficult to infer in the long run. This is the case here: the effect of permanent increases in either income or wealth on expenditures are difficult to infer for horizons beyond 20 quarters or so (i.e., five years).

Any attempt at long-run inference would be accompanied by a high degree of uncertainty—i.e., be rather inaccurate. Is it still useful? That of course depends on the objective. For example, we would conclude that with 68% confidence, a \$1 increase in wealth leads to anywhere between \$0 and \$0.50 increase in expenditures after 10 years.

Whether or not this is useful depends on the question. If the question is: does more wealth lead to a permanent increase in expenditures? Then, our long-run inference is not very useful because our confidence intervals include negative effects for all horizons beyond 10 years. On the other hand, if the question is: do people end up spending all wealth increases in the long run? In this case, our data tells us that with 68% confidence they *do not*—less than half of the wealth increase expected to be spent after 10 years.

Finally, even if long-run effects are difficult to infer, short-term dynamics are typically still accurately estimated and are often themselves of interest. For example, in this case we can infer that with 68% confidence, a \$1 increase in wealth leads to an immediate increase in expenditures between \$0.05 and \$0.15. On the other hand, a \$1 increase in income leads to an immediate increase in expenditures between \$0.45 and \$0.65. So conclusion we can confidently claim is that with 68% confidence, an increase in income leads to more expenditures relative to an equivalent increase in wealth, in the short-run.

There are many other interesting inferential claims we can make from these plots! Try and see if you can draw more conclusions.

(d) Compare the values in `md` to the IRFs estimates obtained in part (a).

Solution The midpoint of the confidence interval is exactly the estimate so the values in `md` should match exactly to the estimated IRFs. However, this may not be the case due to "rounding'' errors in practice. The best way to compare two vectors or matrices is to look at the *norm* of their difference. Even when rounding errors result in a norm that is not zero (when in theory it should be), the norm should yield an obviously small number.

```
print(norm(irfs_lrm$irfs - irfs_ci$md))
```

```
## [1] 0
```

(e) Use the LRM estimates and standard deviations obtained in part (b) to construct 68% confidence intervals for the LRMs, assuming the sampling distributions of the LRM estimators are approximately normal. How do they compare to the IRF confidence intervals obtained in part (c)?

Solution To construct the confidence intervals, we first need to obtain the appropriate percentile z from the normal distribution. To obtain a 68% confidence interval, this requires setting z to be the 84th percentile. We know that for the normal distribution $\Pr(z \leq 1) \approx 0.84$, so $z \approx 1$, but we can also use the R function `qnorm` to compute it.

```
z <- qnorm(.84)
```

Now, construct the confidence intervals and compare.

```
lrm_hat <- t(as.matrix(ecm_lrm$estimate[2:3]))
lrm_se <- t(as.matrix(ecm_lrm$std.error[2:3]))
ones <- matrix(1, nrow = 3, ncol = 1)
zz <- as.matrix(c(-z, 0, z))
lrm_ci <-  ones %*% lrm_hat + zz %*% lrm_se
rownames(lrm_ci) <- c("lb", "md", "ub")
colnames(lrm_ci) <- c("AT", "CT")
print(lrm_ci)
```

```
##           AT        CT
## lb 0.1480907 0.6473332
## md 0.2687276 0.7873063
## ub 0.3893645 0.9272794
```

```
irfs_41_ci <- rbind(irfs_ci$lb[41,], irfs_ci$md[41,],
                    irfs_ci$ub[41,])
rownames(irfs_41_ci) <- c("lb", "md", "ub")
print(irfs_41_ci)
```

```
##             AT         YT
## lb -0.03213304 0.04864858
## md  0.23256522 0.72974583
## ub  0.49726349 1.41084308
```

The key observation in these results is that the confidence intervals for cumulative impulse response at $h = 40$ are in fact wider than the confidence intervals for the LRMs. You should find this quite unsettling!

In fact, both confidence intervals are *asymptotically* correct (with few exceptions), meaning that in an infinite sample we should observe confidence intervals for LRMs that are wider than those for the 40th horizon IR. But we do not have an infinite sample, and this is the main reason for the discrepancy.

We have already discussed in part (d) how inference on cumulative impulse responses in a finite sample is expected to be less accurate at more distant horizons. Indeed, the same is true for estimates of the standard errors that we use to form the confidence intervals!

At more distant horizons, and culminating with the LRMs, standard errors estimates are subject to (often substantial) finite sample bias. The exercise here underscores two points: (i) it is not uncommon to encounter discrepancies, and (ii) inference on dynamic effects at longer horizons requires extra care. The underlying lesson is simply that one should never take output from statistical software for granted.

(f) Construct an adequate set of ARDL(p, l, s) models for c_t.

Solution We follow a procedure very similar to the one used in Tutorial 3 to construct an adequate set of ARMA models. In particular, we estimate all specifications obtained from $p = 1, \ldots, 5$, $l = 0, \ldots, 4$ and $s = 0, \ldots, 4$ and record the AICs and BICs in an easy-to-analyse table. Unlike in Tutorial 3, we will now also store the estimation results for each specification so as to avoid re-estimating the models later.

```
ardl_est <- list()
ic <- matrix(nrow = 125, ncol = 5)
colnames(ic) <- c("p", "l", "s", "aic", "bic")

i <- 0;
for (p in 1:5)
{
  for (l in 0:4)
  {
    for (s in 0:4)
    {
      i <- i + 1
      ardl_est[[i]] <- ardl(CT ~ AT + YT, mydata,
                                          order = c(p, l, s))
      ic[i,] <- c(p, l, s, AIC(ardl_est[[i]]),
                            BIC(ardl_est[[i]]))
    }
  }
}

ic_aic <- ic[order(ic[,4], decreasing = FALSE),][1:10,]
ic_bic <- ic[order(ic[,5], decreasing = FALSE),][1:10,]
```

```
> ic_aic
      p l s       aic       bic
 [1,] 1 2 2 -1600.168 -1569.832
 [2,] 3 2 2 -1598.965 -1561.939
 [3,] 1 3 2 -1598.863 -1565.203
 [4,] 2 2 2 -1598.473 -1564.766
 [5,] 3 2 3 -1598.280 -1557.888
 [6,] 1 2 3 -1598.139 -1564.479
 [7,] 3 3 2 -1597.185 -1556.793
 [8,] 1 3 3 -1597.015 -1559.989
 [9,] 2 3 2 -1596.953 -1559.927
[10,] 3 3 3 -1596.453 -1552.695
> ic_bic
      p l s       aic       bic
 [1,] 1 2 2 -1600.168 -1569.832
 [2,] 1 1 2 -1594.746 -1567.781
 [3,] 1 3 2 -1598.863 -1565.203
 [4,] 1 0 2 -1588.508 -1564.914
 [5,] 2 2 2 -1598.473 -1564.766
 [6,] 1 2 3 -1598.139 -1564.479
 [7,] 1 1 3 -1592.887 -1562.593
 [8,] 2 1 2 -1592.795 -1562.460
 [9,] 3 2 2 -1598.965 -1561.939
[10,] 3 1 2 -1594.738 -1561.078
```

(f) Construct an adequate set of ARDL(p, l, s) models for c_t.

The first six models preferred by the BIC also have sufficient support in terms of the AIC, so we take this to be the adequate set.

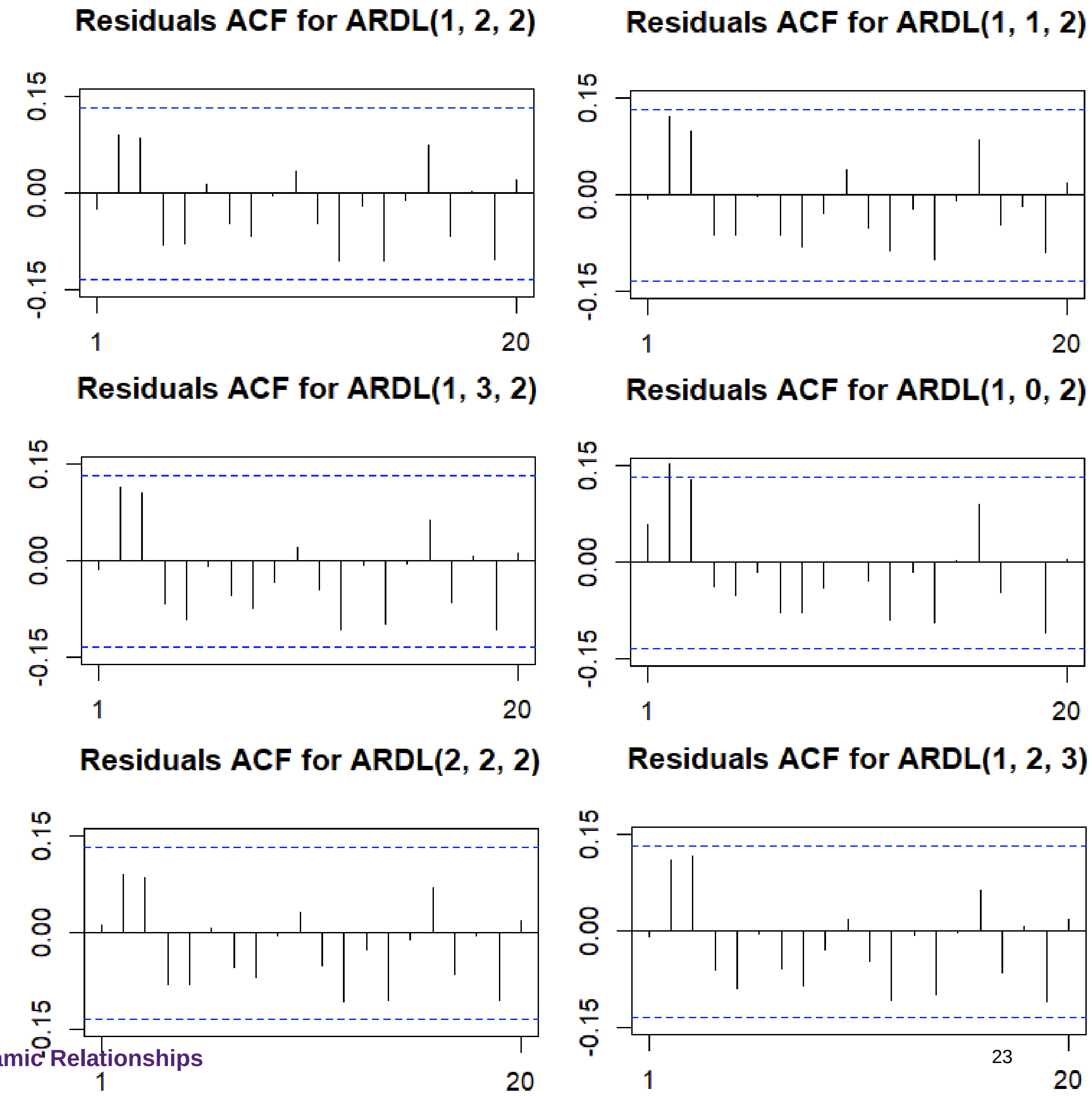
```
adq_set <- ic_bic[1:6,]
```

We also need to match the orders in the adequate set to the set of all models that we have estimated.

```
adq_idx <- match(data.frame(t(adq_set[, 1:3])),
                 data.frame(t(ic[, 1:3])))
```

W finally do a quick scan of the residuals for obvious anomalies.

```
for (i in 1:length(adq_idx))
{
  order <- adq_set[i,1:3]
  acf(ardl_est[[adq_idx[i]]]$residuals,
      xlim = c(1, 20), xaxp = c(1, 20, 1),
      ylim = c(-0.15, 0.15), yaxp = c(-0.15, 0.15, 2),
      main = paste("Residuals ACF for ARDL(", order[1], ", ",
                                              order[2], ", ",
                                              order[3], ")",
                                              sep = ""))
}
```

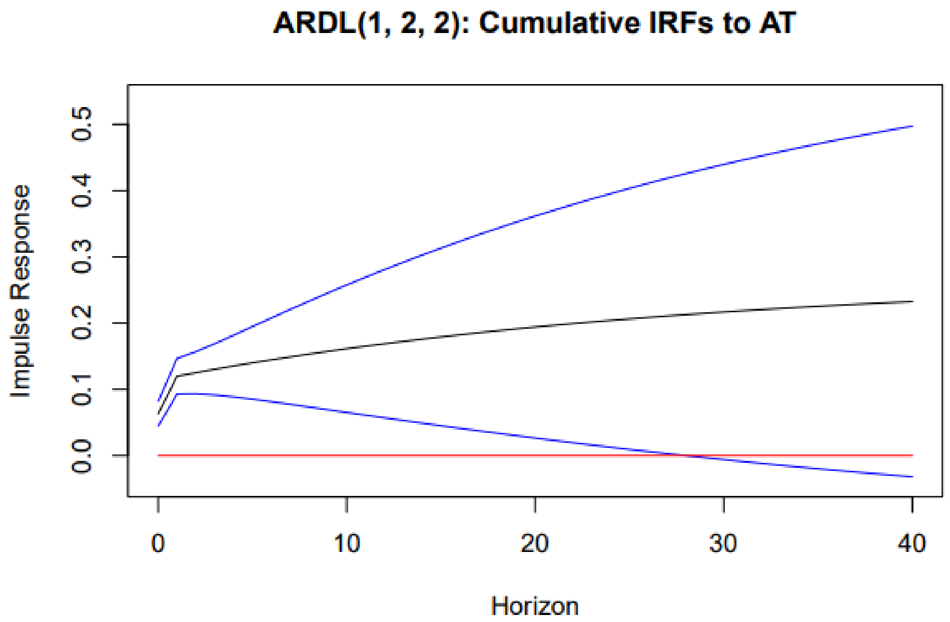


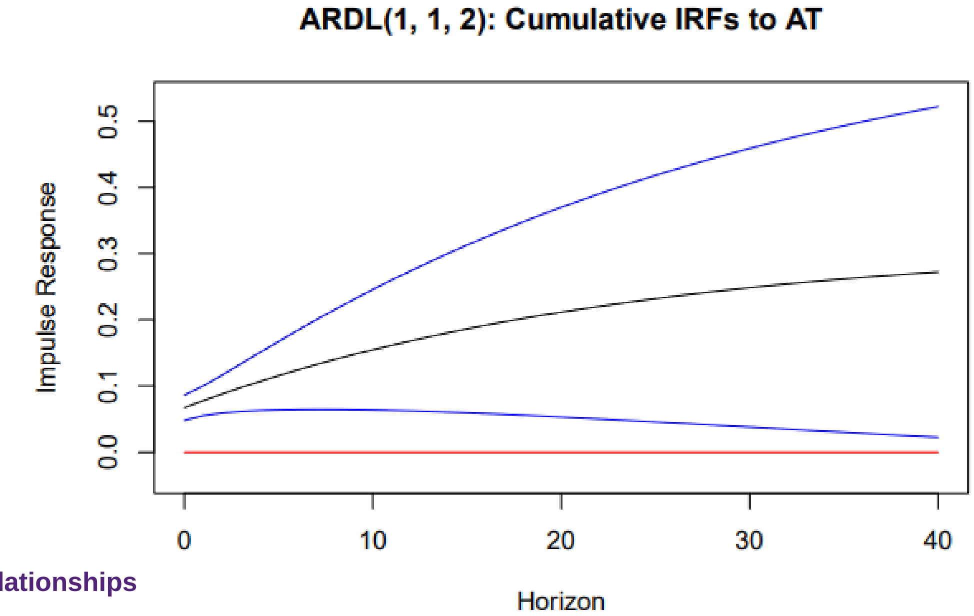
(g) Draw inference about the dynamic relationship between expenditures and wealth.

Solution Generate the plots of IRFs with confidence intervals for all ARDL specifications in the adequate set.

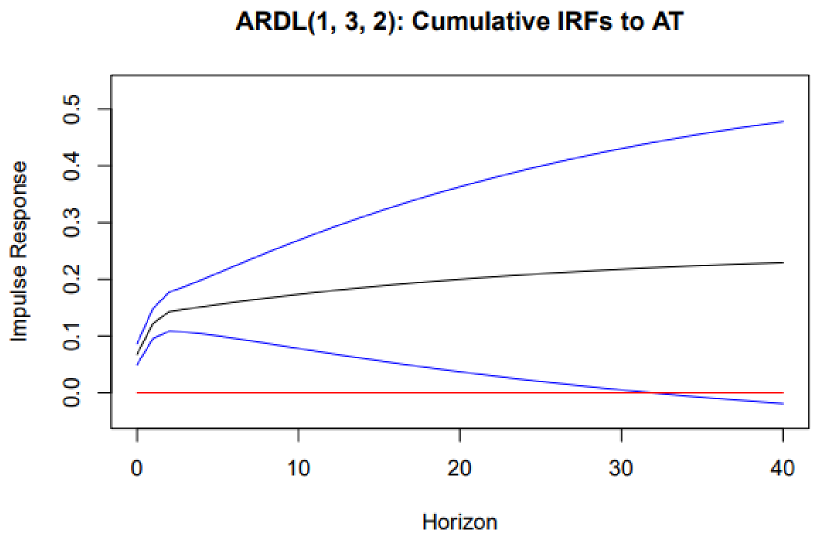
```
j <- 1 # select responses to AT
shock_name <- "AT"
y_min <- Inf
y_max <- -Inf
irfs_ci <- list()
for (i in 1:length(adq_idx))
{
  irfs_ci_i <- ardl_irfs_ci(ardl_est[[adq_idx[i]]], conf = 0.68)
  irfs_ci[[i]] <- cbind(irfs_ci_i$lb[, j], irfs_ci_i$md[, j],
                        irfs_ci_i$ub[, j])
  y_min <- min(y_min, irfs_ci_i$lb[, j])
  y_max <- max(y_max, irfs_ci_i$ub[, j])
}
```

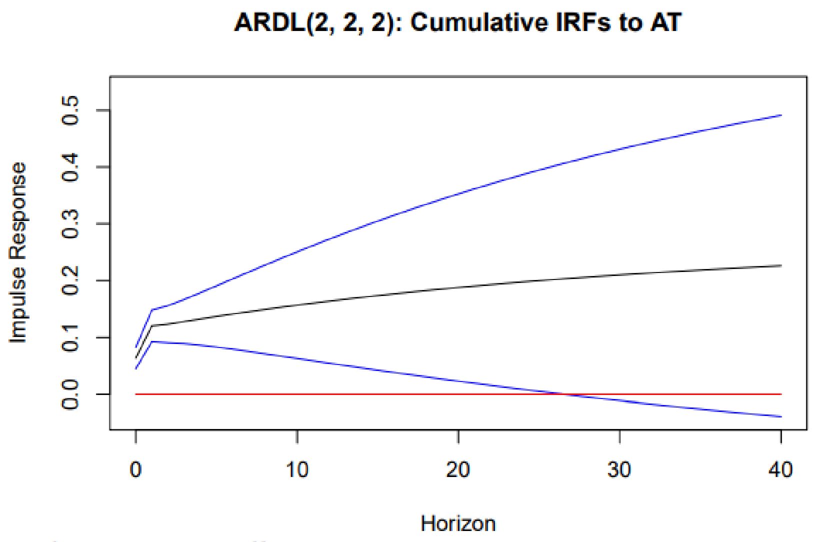
```
for (i in 1:length(adq_idx))
{
  order <- adq_set[i,1:3]
  plot(0:40, irfs_ci[[i]][, 2], type = "l", ylim = c(y_min, y_max),
    ylab = "Impulse Response", xlab = "Horizon",
    main = paste("ARDL(", order[1], ", ", order[2], ", ", order[3],
              "): Cumulative IRFs to ", shock_name, sep = ""))
  lines(0:40, irfs_ci[[i]][, 1], type = "l", col = "blue")
  lines(0:40, irfs_ci[[i]][, 3], type = "l", col = "blue")
  lines(0:40, numeric(41), type = "l", col = "red")
}
```

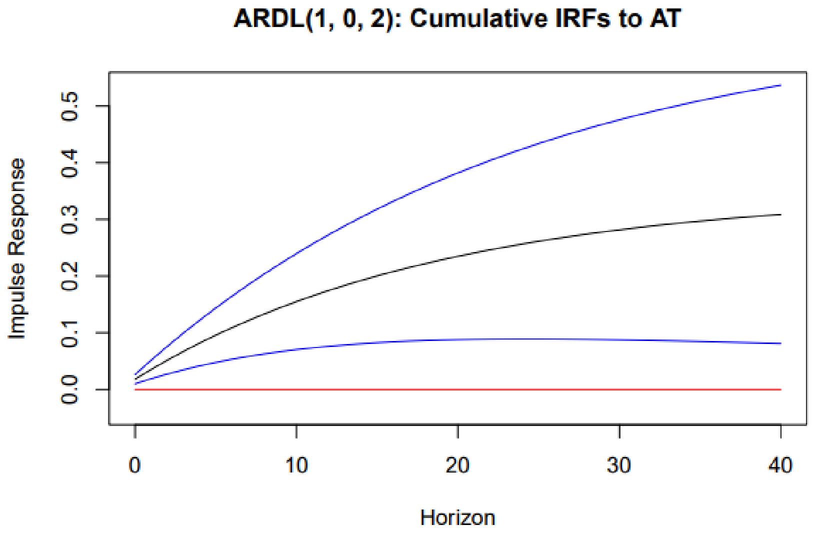




(g) Draw inference about the dynamic relationship between expenditures and wealth.



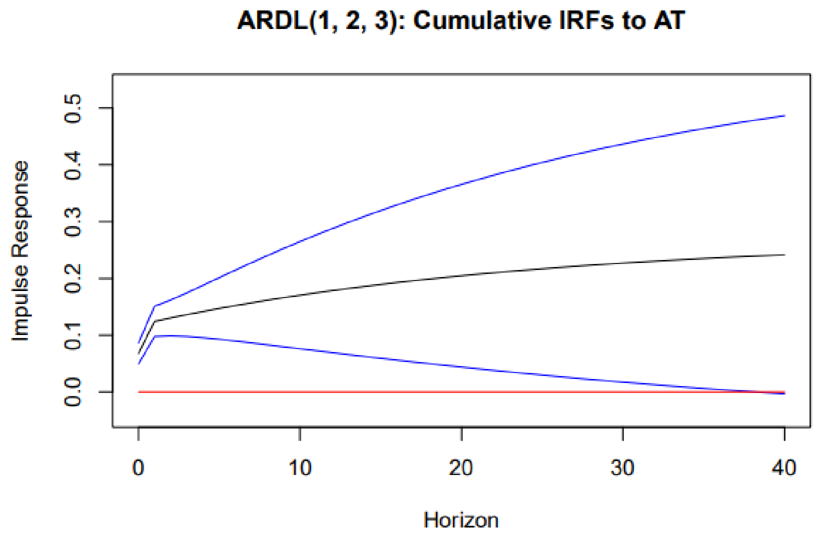




We divide the inference into short-run and long-run effects. By long-run we will mean around 10 years (equivalently, $h \approx 40$) following a shock to wealth. By short-run we will mean within the first year (equivalent, $h \leq 4$).

Starting with the long-run effects, we conclude with reasonable confidence that, *cetiris paribus*, an increase in wealth will be *at most* partially spent after 10 years. In particular, it is very unlikely that more than half of a wealth increase is spent, but there is a small possibility that overall expenditures contract, in the ten years following a wealth increase.

In the short-run, we conclude with reasonable confidence that an increase in wealth leads to an increase in expenditures. Two scenarios are possible. In the first, expenditures jump quickly (within one quarter) and then level off. In the second scenario, expenditures rise gradually. Under both scenarios, expenditures are unlikely exceed 20% of the wealth increase within the first year. However, the first scenario results in a greater short-term expenditure increase relative to the second scenario.



What inference can you obtain about the dynamic relationship between expenditures and income?

Tutorial 7: Dynamic Relationships

At the end of this tutorial you should be able to:

- derive the ECM representation of an ARDL(p, l, s) model;
- create a function in R;
- estimate IRFs to permanent and one-off shocks as well as LRMs;
- construct confidence intervals for IRFs and LRMs;
- Select and adequate set of ARDL models and draw inference on dynamic relationships.

Thank you

Francisco Tavares Garcia
Academic Tutor | School of Economics
tavaresgarcia.github.io

Reference

Tsay, R. (2010). Analysis of Financial Time Series, 3rd Edition, John Wiley & Sons.