NHH

HOME EXAMINATION BAN400

Fall, 2022

Start: December 19. 2022 09:00

End: December 21. 2022 14:00

THE HOME EXAMINATION SHOULD BE SUBMITTED IN WISEFLOW

You can find information on how to submit your paper here:

`https://www.nhh.no/en/for-students/examinations/home-exams-and-assignments/`

Your candidate number will be announced on StudentWeb. The candidate number should be noted on all pages (not your name or student number). In case of group examinations, the candidate numbers of all group members should be noted.

SUPPLEMENTARY REGULATIONS FOR HOME EXAMINATIONS

You can find supplementary regulations under the headline “Regulations”:

`https://www.nhh.no/en/for-students/regulations/`

Find more information under chapter 4.0 in the Supplementary provisions to the regulations for fulltime study programmes

Number of pages, including front page: 11

Number of attachments: 3 (BAN400-H22-template.Rmd, Stefanski-pandemic-data.xlsx, WPKAE-2020-057.pdf)

About the home exam

In order to complete BAN400, you must pass this individual home exam. Please take note of the following items:

- You should submit a single `.zip` file on WiseFlow, and the `.zip`-file should contain:
  - One `.rmd`-file with your answer to the questions.
  - One `.html`-file, compiled from the `.rmd`-file that you submit.
- The `.rmd`-file should be based on the template-file `BAN400-H22-template.Rmd`.
- The examiner should be able to reproduce the `.html`-file by compiling the `.rmd`-file without any modifications. State clearly if you make use of packages that must be installed before compilation.

- Note that this is an *individual* exam, and any cooperation is *not* permitted.
- The deadline for submission in WiseFlow is a hard constraint. Make sure you submit in time.
- The grader is mostly interested in your code. However, we recommend also writing a few sentences to each question, explaining your results and answers to the questions. This helps the examiner in assessing your submission.
- If you do not manage to complete some assignments, you may score partial points by submitting a partial solution, with a comment on how your solution fails or is incomplete.
- You will likely need to use functions or packages *not* explicitly covered in BAN400/BAN420. Understanding documentation and independently resolving programming issues are part of the learning outcomes of the course.

Stefanski pandemic data

In a working paper, Stefanski (2020) (the attached working paper, file named `WPKAE-2020-057.pdf`, hereafter abbreviated as S2020) analyses the effect of pandemics, arguing that pandemics have a *positive* effect on GDP per capita. The working paper uses a panel data set going back to the 13th century.

A panel data contains data on units over time. In this case, the units are countries. We have data on $N = 33$ different countries. We observe each country many times, where each row in the data set is data on one country in one year. An observation $y_{i,t}$ denotes an observation of variable the y on country i in year t.

The main variable of interest, $y_{i,t}$, is the logarithm of GDP per capita. We want to estimate what the effects of pandemics $P_{i,t}$ are on $y_{i,t}$. S2020 uses different measures of pandemics. In this home exam, we will focus on `pandemic_01`, which is a binary variable taking the value 1 if a country suffered a pandemic in a given year. S2020 also uses other control variables such as war, $W_{i,t}$, as well as indicators for specific time periods.

A way of estimating the effect of shocks (in our case, the shock is a pandemic) on the outcome variable y, is to estimate an *impulse response function* $IRF(h)$. Letting X denote "other" variables, the impulse response function is defined as:

$$IRF(h) = E[y_{t+h}|P_t = 1, X] - E[y_{t+h}|P_t = 0, X],$$

i.e., the $IRF(h)$ is the expected difference in y h periods in the future resulting from a pandemic at time t. Note that there are no "country"-subscripts in the equation above. This is because the impulse response function will be the same for all countries in the models used in this home exam. If we have estimates of $IRF(0), IRF(1), ..., IRF(H)$, we can plot the estimated expected effect of a pandemic H years into the future.

Impulse response functions can be estimated using linear regression. See the explanation in S2020, Section 3.1, where the example is a simple regression like

$$y_{t+h} = \alpha^h + \beta^h D_t + \varepsilon_t,$$

where the superscript h on β_t^h is an *index*, not an exponent.

After running the regression $H+1$ times with h equal to $0, 1, 2, ..., H$, the estimates of β^h will then be the estimated impulse response function.

However, with panel data, we need to add more explanatory variables to the model. Most importantly, we need to add dummy variables for each country, as well as to include the lagged dependent variable as a covariate. This data set was also used in the exam in the course MET4 at NHH in the fall semester of 2021 in a much simpler setting. The following specification was used there:

$$\begin{aligned} Y_{i,t+h} = \alpha_i^h + \beta_1^h P_{i,t} + \beta_2^h P_{i,t-1} + \beta_3^h W_{i,t} + \beta_4^h W_{i,t-1} + \beta_5^h Vienna_{t-1} + \beta_6^h Y_{t-1} + \beta_7^h Oil_t \\ + \beta_8^h WW2_t + \beta_9^h t + \beta_{10}^h t * Vienna_t + \beta_{11}^h t * Oil_t + \beta_{12}^h t * WW2_t + \varepsilon_{i,t}, \end{aligned} \tag{1}$$

where $War_{i,t}$ is a variable indicating whether the country i is at war at time t, and $Vienna_t, Oil_t, WW2_t$ are dummy variables becoming 1 in the year 1815, 1945 and 1973, respectively. In the equation above, β_1^h is the parameter of interest, giving the impulse response at an h-period horizon.

Assignment 1

The code below was used in the solution proposal of the exam in MET4 in the fall of 2021 (note that a few bugs have been removed from the original code). Run the code and compare the figure you get with S2020 Figure 9. Comment on whether you get the similar results as S2020, even though the model used here is simpler.

```
library(tidyverse)
library(magrittr)
library(readxl)

# Load data, select relevant variables:
df <-
  read_excel("Stefanski_pandemic_data.xlsx") %>%
  select(-tree_ring, -pandemic_pc, -gdp)
```

```r
# Define function for estimating regression for a given
# point of the impulse response function, and return a
# data frame of coefficients with summary statistics:
runreg <-
  function(h, df) {
    df %>%
      select(country, year, ln_gdp, pandemic_01, war) %>%
      mutate(country = factor(country)) %>%
      group_by(country) %>%
      mutate(
        lead_ln_gdp = lead(ln_gdp, h, order_by = year),
        lag_1_pandemic_01 = lag(pandemic_01, 1, order_by = year),
        lag_1_war = lag(war, 1, order_by = year),
        lag_ln_gdp = lag(ln_gdp, 1, order_by = year),
        vienna = case_when(year >= 1815 ~ 1, TRUE ~ 0),
        ww2 = case_when(year >= 1945 ~ 1, TRUE ~ 0),
        oil = case_when(year >= 1973 ~ 1, TRUE ~ 0),
        vienna_t = vienna * year,
        ww2_t = ww2 * year,
        oil_t = oil * year
      ) %>%
      ungroup() %>%
      lm(
        lead_ln_gdp ~
          country + lag_ln_gdp + pandemic_01 +
          lag_1_pandemic_01 + year + war +
          lag_1_war + vienna + oil + year +
          ww2 + vienna_t + oil_t + ww2_t,
        data = .
      ) %>%
      summary() %>%
      .$coefficients %>%
      as.data.frame() %>%
      rownames_to_column() %>%
      as_tibble() %>%
      mutate(h = h)
  }

# Iterate over h=0,...,40, and estimate the different
# IRF(h) coefficients. Thereafter, plot the results
lapply(0:40, FUN = runreg, df = df) %>%
  bind_rows() %>%
  filter(rowname == "pandemic_01") %>%
  ggplot(aes(x = h)) +
  geom_line(aes(y = Estimate * 100)) +
  geom_ribbon(aes(
```

```
    ymin = (Estimate - 1.96 * `Std. Error`) * 100,
    ymax = (Estimate + 1.96 * `Std. Error`) * 100
  ), alpha = .1) +
  geom_hline(yintercept = 0) +
  xlab("Fcast horizon") +
  ylab("Impulse response, percent") +
  ggtitle("Estimated impulse response of pandemics on GDP pr capita") +
  theme_classic()
```

Assignment 2

A *dynamic panel data model* is a model that looks like $y_{i,t} = \alpha_i + \beta y_{i,t-1} + ... + \varepsilon_{i,t}$ – i.e. that the lagged value of the dependent variable (i.e. $y_{i,t-1}$) enters the regression equation.

There are multiple methodological issues related to estimating dynamic panel data models, as discussed by S2020 pages 8 and 10. Further, with a complicated regression setup chances are that we might have done something *wrong* in the implementation of the code. In order to check if the code and methodological setup works we can *simulate* data and verify that the estimated impulse response function do indeed resemble the *true* values.

Create a function `sim_one_country`. The function should return a data frame. The arguments of the function should be:

- `country`: The "name" or "value" of the country being simulated
- `T`: How many years of data to simulate
- `p_pand`: The probability of a pandemic in a given year
- `p_war`: The probability of a war in a given year
- `coeff_lag_ln_gdp`: Coefficient on $y_{i,t-1}$ (i.e. the coefficient β_6^h in Equation (1) in the introduction, or the coefficient β_6 in Equation (2) below, which is the specification that we will use in this assignment).
- `sigma`: Standard error of $\varepsilon_{i,t}$.
- `impulse_response`: A *vector* with the different values of β_1^h. For example, `impulse_response=c(1,.5)` implies IRF(0) = 1, IRF(1) = 0.5, and IRF(2) = IRF(3) = $\cdots$ = 0. The function should be able to handle any length of this vector larger than 1.
- `sigma_initial_value`: Standard error of the *first* observation in the data set (i.e. $y_{i,1}$).
- `sigma_alpha`: Standard error of the fixed effect (i.e. α_i).

The function should *return* a data frame with variables `country, year, ln_gdp, pandemic_01, war`.

The function should simulate data according to the following procedure:

- `country` should be equal to the value of `country` supplied as an argument.
- The variable `pandemic` should be a binary random variable, taking the value `1` with probability `p_pand`.
- The variable `war` should be a binary random variable, taking the value `1` with probability `p_war`.
- The variable *year* should take all values from `1` to `T`.

Let α_i be given by `rnorm(1, sd=sigma_alpha)`. The first value of `ln_gdp`, i.e. $y_{i,1}$, should be a initialized as a random normal variable `alpha + rnorm(1, sd=sigma_initial_value)` (where, of course, `alpha` is α_i that we generated above). Further, let $\zeta_{i,t}$ be a normally distributed variable with expectation 0 and standard deviation equal to `sigma`. Then `ln_gdp` ($y_{i,t}$) should be given by

$$y_{i,t} = \alpha_i + \beta_6 y_{i,t-1} + \sum_{h=0}^{M-1} \beta^h P_{t-h} + \zeta_{i,t}, \tag{2}$$

where M is the length of the vector `impulse_response` that is supplied as an argument.

Finally, create a function `simdat`. This function should accept arguments `N`, `T`, `p_pand`, `p_war`, `sigma`, `coeff_lag_ln_gdp`, `impulse_response`, `sigma_alpha`, `sigma_initial_value` – i.e. the same arguments as `sim_one_country`, except that we have the argument N instead of `country`. `N` ≥ 1 should be an integer. The function `simdat` should call the function `sim_one_country` `N` times, and return a single data frame with simulated data for N different countries. The different simulated countries should have different values of `country` (for example `country1`, `country2`,... – or just numbers `1,..,N`).

Your functions should pass the following tests, which are included in the template `.Rmd`-file. You can also inspect the tests below for additional pointers on the expected behavior of the functions. You may need to apply your own judgement to be able to solve this assignment.

```
library(assertthat)

assert_that(
  sim_one_country(
    country = 1,
    T = 5,
    p_pand = 1,
    p_war = 0,
    sigma = 0,
    coeff_lag_ln_gdp = 0,
    impulse_response = c(1),
    sigma_initial_value = 0,
    sigma_alpha = 0
  ) %>%
    filter(year > 1) %>%
    filter(ln_gdp == 1) %>%
    nrow() == 4,
  msg = "Impulse responses are not correct"
)


assert_that(
  all(
    sim_one_country(
      country = 1,
```

```
      T = 5,
      p_pand = 1,
      p_war = 0,
      sigma = 0,
      coeff_lag_ln_gdp = 0,
      impulse_response = c(1, 2, 3, 4),
      sigma_initial_value = 0,
      sigma_alpha = 0
    )$ln_gdp == c(0, 3, 6, 10, 10)
  ),
  msg = "Impulse response is not implemented correctly"
)

assert_that(
  all(
    sim_one_country(
      country = 1,
      T = 5,
      p_pand = 1,
      p_war = 0,
      sigma = 0,
      coeff_lag_ln_gdp = 0,
      impulse_response = rep(1, 10),
      sigma_initial_value = 0,
      sigma_alpha = 0
    )$ln_gdp == c(0, 2, 3, 4, 5)
  ),
  msg = "Function fails if impulse response is longer than 1:T"
)

assert_that(abs(sd(
  replicate(
    1000,
    sim_one_country(
      country = 1,
      T = 1,
      p_pand = 1,
      p_war = 0,
      sigma = 0,
      coeff_lag_ln_gdp = 0,
      impulse_response = c(0),
      sigma_initial_value = 1,
      sigma_alpha = 0
    )$ln_gdp
  )
) - 1) < .1,
```

```
msg = "Random number generation does not work"
)


assert_that(abs(mean(
  simdat(
    N = 1,
    T = 10000,
    p_pand = .1,
    p_war = 0,
    sigma = 0,
    coeff_lag_ln_gdp = 0,
    impulse_response = rep(1, 10),
    sigma_initial_value = 0,
    sigma_alpha = 0
  )$pandemic_01
) - .1) < .01,
msg = "Pandemic variable simulation does not work"
)


assert_that(
  simdat(
    N = 10,
    T = 5,
    p_pand = .1,
    p_war = 0,
    sigma = 1,
    coeff_lag_ln_gdp = 0,
    impulse_response = rep(1, 10),
    sigma_initial_value = 0,
    sigma_alpha = 0
  ) %>% nrow() == 50,
  msg = "The simdat function return wrong number of rows"
)

assert_that(
  length(unique(simdat(
    N = 100,
    T = 5,
    p_pand = .1,
    p_war = 0,
    sigma = 1,
    coeff_lag_ln_gdp = 0,
    impulse_response = rep(1, 10),
    sigma_initial_value = 0,
```

```
    sigma_alpha = 0
  )$country)) == 100,
  msg = "The simdat function return wrong number of countries"
)
```

Assignment 3

We can now use simulated data to check if our IRF-estimator works as intendend.

Use the following parameter values:

- `N=50`
- `T=1000`
- `p_pand=.1`
- `p_war=0`
- `sigma=10`
- `impulse_response=c(1,-1,2,-2,0,0,1)`
- `sigma_initial_value = 1`
- `sigma_alpha = 5`

Simulate data and estimate the IRF twice using the `runreg`-function from Assignment 1:

1. One with `coeff_lag_ln_gdp=0`. This means that the data isn't "dynamic", as the lagged dependent variable doesn't influence future gdp directly.
2. One with `coeff_lag_ln_gdp=.96`. This is a similar coefficient as the one used on the real data set, and means that the data *is* dynamic (in fact, estimated to being almost non-stationary).

Create figures similar to that of Assignment 1, but add the real impulse response function (i.e. `c(1,-1,2,-2,0,0,1)`) to the plot.

S2020, page 10, writes: *The set of regressions is estimated with the fixed effects estimator. Even though this estimator is biased in a dynamic panel setting (Nickell 1981), the downward bias on the lagged dependent variable coefficient is diminishing with increasing time dimension of the panel, and thus can be safely ignored in the sample which covers 765 years.*

If one coefficient estimate is biased, this may raise concerns about the other coefficient estimates being biased as well. Based on your figures, comment briefly on the paragraph above.

Assignment 4

S2020 has many lags of many of the variables in his regressions. Focusing on the variable `ln_gdp` and `pandemic_01`, create a function `runreg_baseline`. The function should return a data frame with regression coefficients, similarly to the `runreg`-function, but the `runreg_baseline`-function should estimate a regression on the form

$$
\begin{aligned}
Y_{i,t+h} =& \alpha_i^h + \sum_{j=0}^{K} \beta_j^h P_{i,t-j} + \sum_{m=1}^{K} \theta_j^h y_{i,t-m} + \\
& \delta_1^h W_{i,t} + \delta_2^h W_{i,t-1} + \delta_3^h Vienna_{t-1} + \delta_4^h Oil_t + \\
& \delta_5^h WW2_t + \delta_6^h t + \delta_7^h t * Vienna_t + \delta_8^h t * Oil_t + \delta_9^h t * WW2_t + \varepsilon_{i,t},
\end{aligned} \tag{3}
$$

where, for each h, the impulse response function $IRF(h)$ is given by β_0^h. The arguments of the function should be:

- `h`: How many leads should be used for the dependent variable (similar to assignment 1-3)
- `df`: The data set used in the regression
- `K`: How many lags should be used in the regression for variables $y_{i,t}$ and $P_{i,t}$.

S2020 generally used `K=30`, but we want to be able to try other values. Note that one of the sums in Equation (3) starts with the index $j = 0$, and the other starts at $m = 1$.

Make sure that the names of the lagged variables are named as `ln_gdp_lag_1,ln_gdp_lag_2,..` and `pandemic_01_lag_1,pandemic_01_lag_2,...`, where `ln_gdp_lag_1`= $y_{i,t-1}$ et cetera.

Create a figure similarly to assignment 1, but with `K=30`. Further, your function should pass the tests below.

```
assert_that(
  runreg_baseline(0, df, 40) %>%
    filter(substr(rowname, 1, 11) == "pandemic_01") %>%
    nrow(.) == 41,
  msg = "Function does not have the right number of lags for pandemics"
)

assert_that(
  runreg_baseline(0, df, 70) %>%
    filter(substr(rowname, 1, 6) == "ln_gdp") %>%
    nrow(.) == 70,
  msg = "Function does not have the right number of lags for ln_gdp"
)
```

Assignment 5

The data set has many interesting possibilities for illustrating pandemics over time. Create a visualization of the pandemic data on a map, showcasing where and when pandemics have occurred. Consider, for instance, making the illustration a gif. You may also use the `pandemic`-variable in the original data set, which measures the percent of the population that died due to the pandemic in a given year.

Assessment of exam submissions

The learning outcomes and general competencies defines the targets for the assessment of the exam. Submissions will be ranked by the point system below. The cutoffs between grades will be determined at grading.

- **Read and understand documentation of packages and functions.**
  - 1 point: The submission uses functions/packages not covered explicitly in BAN400
- **Use basic data structures (lists, arrays, matrices, vectors and data frames) as appropriate. Combine, merge and reshape data sets in R.**
  - 4 points: The submission passes all assert-tests in assignment 2. Partial credit is given with 0.5 points for each passed test, and 0.5 points as "bonus" for successfully completing all tests.
- **Independently resolve warnings, errors, and other basic programming issues.**
  - 1 point: The examiner successfully reproduces the results from the submission, without needing to editing the code.
- **Use functions, loops, assignments, subsetting and conditionals in an R-script.**
  - 1 point: The submission makes use of map-functions as applicable for iterations
  - 2 points: The submission passes both assert-tests in assignment 4.
- **Use vectorization, iterations and parallelisation as needed computationally demanding tasks.**
  - 1 points: The submissions allows for use of multiple CPU-cores to speed up the time for compilation of rmd-file.
- **Write documented and standardized, formatted code as part of code development.**
  - 1 point: Applies functions to reduce code repetition and improve readability of code.
  - 1 point: Naming of functions and function arguments improves readability of the code.
- **Use R to program and apply selected prediction and machine learning methods and correctly interpret the output in the relevant context.**
  - 2 points: The estimated and true impulse-response functions in assignment 3 in the case without dynamics are close to overlapping.
  - 1 points: The estimated and true impulse-response functions in assignment 3 in the case with dynamics are similar (but do not have to be close to overlapping).
- **Create and export convincing tables and figures for use in reports and presentations.**
  - 1 point: The figures are visually pleasing, appropriately labeled, and informative of the case questions.
  - 1 point: Submission creates as requested figures in assignments 1-4.
  - 3 points: The submission successfully creates a figure as requested in assignment 5.
- **Apply R to empirical business and economics problems.**
  - 1 point: All questions are answered, with code that balances readability and compactness.