Programming for Economists **week 43 2025**

Dataproject - A small guide

Casper Moes

Class 5

1.1 General best practice when coding

1.1.1 Pseudocode (writing your code in human language)

Whenever you are faced with a programming problem (a task you want the computer to automate or do) then i would strongly recommend that you write out the steps you want to code to comlete - in plain english.

Example: Solving the first part of basics 1.1. in the dataproject assignment: The problem is as follows: *Using table PRIS113 from Denmark Statistics produce a figure with (1.) The consumer price index (CPI), P_t, indexed to 100 on average in 2020*

An example of some (very thorough) pseudocode could be:

```
# download the pris113 data with all the periods
# create a time variable, so that we can plot the data easialy and use build in pandas method[1]
# make sure that price data is of datatype float (and not str)
# tidy the data up by removing unwanted columns
# retrieve all datapoints in 2022 and calculate the mean of these values.
# scale all the price datapoints by the average we just calculated to create the index
# plot the results.
```

1.1.2 Understanding functions

When programing in pytthon you will be using a lot of functions build by other people - this is the strength of open source sofware like python and R!

[1] remember methods is just functions inside a class instance. A class is just a structured object you or someone else has defined (you made your own in datacamp!)

Whenever you are learning a new function, you should get in the habit of looking at the documentation. These are found by simply googling the module and function. For example you could google "numpy array documentation" and this webpage appears: https://numpy.org/devdocs//reference/generated

Or use chatbots Alternatively you can use chatbots and get them to layout the inputs and outputs of a given function / method.

1.1.3 Downloading modules

We mosly use the program `pip` to install packages. You can download official python modules by writing `pip install <module_name>`. For example if you want to install numpy `pip install numpy`.

1.2 DST API module

Download the module write `pip install git+https://github.com/alemartinello/dstapi` in the terminal to tell `pip` to install the given github repository at the url on your computer.

In a notebook you should prefix this with a %-sign to tell jupyter that this is not an ordinary python command.

1.2.1 How to use the dstapi module

The basic syntax follows this example in lecture 05_06:

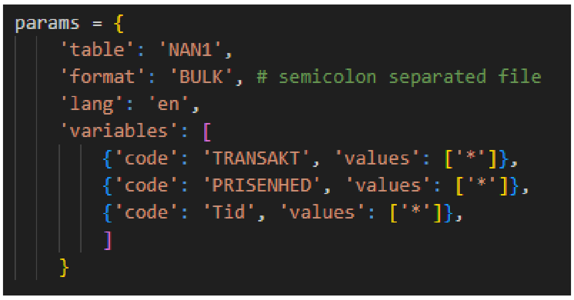
Initiate the API call: `NAN1 = DstApi('NAN1')`

Learn about variables contained in the dataset `NAN1.tablesummary(language='en')`

View which values a specific variable take on (here PRISENHED):
`NAN1.variable_levels('PRISENHED',language='en')`

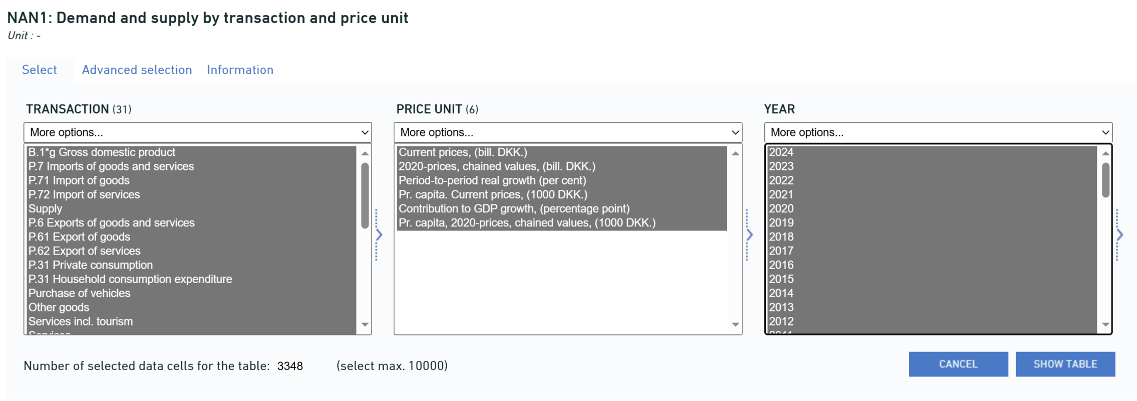
Understand paramter-structure for the specific table `NAN1._define_base_params()`

You will get an output similar to the this (where * symbolizes that you want every variable value)



```
params = {
    'table': 'NAN1',
    'format': 'BULK', # semicolon separated file
    'lang': 'en',
    'variables': [
        {'code': 'TRANSAKT', 'values': ['*']},
        {'code': 'PRISENHED', 'values': ['*']},
        {'code': 'Tid', 'values': ['*']},
        ]
    }
```

Which corresponds to this in the "statistikbank" (Directly translates to: statistics-bank) you can change the variables to specify and limit which series are outputtet. To do this use the `variable_levels()` function, which i showed the syntax of above.



NAN1: Demand and supply by transaction and price unit

Unit : -

TRANSACTION (31)

More options...

B.1*g Gross domestic product
P.7 Imports of goods and services
P.71 Import of goods
P.72 Import of services
Supply
P.6 Exports of goods and services
P.61 Export of goods
P.62 Export of services
P.31 Private consumption
P.31 Household consumption expenditure
Purchase of vehicles
Other goods
Services incl. tourism

PRICE UNIT (6)

More options...

Current prices, (bill. DKK.)
2020-prices, chained values, (bill. DKK.)
Period-to-period real growth (per cent)
Pr. capita. Current prices, (1000 DKK.)
Contribution to GDP growth, (percentage point)
Pr. capita, 2020-prices, chained values, (1000 DKK.)

YEAR

More options...

2024
2023
2022
2021
2020
2019
2018
2017
2016
2015
2014
2013
2012

Number of selected data cells for the table: 3348 (select max. 10000)

1.3 Instantanious inflation-rate

This section attempts to explain the instantaious inflation rate in a thorough manner. (for those who are interested).

Annual CPI-inflation is given by the relative change in the pricelevel (of a "representative" consumption-basket).

$$\pi_t^{12} \equiv \frac{p_t - p_{t-12}}{p_{t-12}} \tag{1.1}$$

But how does this relate to the montly price changes which are in the PRIS113 dataset? Let us rewrite (1.1) to see this:

$$\frac{p_t}{p_{t-12}} - 1 = \frac{p_t}{p_{t-1}} \frac{p_{t-1}}{p_{t-2}} \cdots \frac{p_{t-11}}{p_{t-12}} - 1 = (1+\pi_t^m)(1+\pi_{t-1}^m)\cdots(1+\pi_{t-11}^m) - 1 = \Pi_{k=0}^{11}(1+\pi_{t-k}^m) - 1 \tag{1.2}$$

where $\pi_t^m \equiv p_t/p_{t-1} - 1$ is the montly inflation (which PRIS113 contains an index over - therefore the calculation is exactely the same! as with "level" data - meaning the price-level instead of index).

This is the inflation rate most central banks and think tanks work with. But is this really usefull in the times of rapidly changing inflation rates? For example as we experienced during 2022? In short; I do not think so, because this measuement (or metric) has a lot of inertia, and will by biased downwards in times of rising inflation and biased downwards in times of falling inflation.

Therefore it would be nice with a metric for inflation that weighs resent price changes more than price changes long age. This is the basic motivation for including some sort of weighting function κ. Then we have the intentanious inflation defined as

$$\pi_t^{12:instant} = \Pi_{k=0}^{11}(1 + \pi_{t-k}^m)^{\kappa(k)} - 1 \tag{1.3}$$

where the k in $\kappa(k)$ denotes how many periods in the past the weight should apply to.

We use a polynomial kernel which is parametrices by α:

$$\kappa(k, \alpha) = \frac{(T-k)^\alpha}{\sum_{k=0}^{T-1}(T-k)^\alpha} \cdot T \tag{1.4}$$

Properties of this kernel (our weighting function) The sum of the weights are always T (just like our original neutral weighting)

$$\sum_{k=0}^{T-1} \kappa(k,\alpha) = \sum_{k=0}^{T-1} \left[\frac{(T-k)^{\alpha}}{\sum_{k=0}^{T-1}(T-k)^{\alpha}} \cdot T \right] = \frac{T}{\sum_{k=0}^{T-1}(T-k)^{\alpha}} \sum_{k=0}^{T-1}(T-k)^{\alpha} = T \tag{1.5}$$

This is eaxtely the same as our neutral weight where each monyl inflation rate is weighted by 1 in equation (1.2). ($\sum_{k=0}^{T-1} 1 = T$).

Therefore we are in fact just weighting the periods differently.[2]

[2] just like when you calculate your weighted grade point average (because you recieved great mat A grades;) where the weights should sum to 1.