group_10

January 26, 2021

1 DOPP 2020W Exercise 3

1.1 Group 10

Question:

How has the use of nuclear energy evolved over time? How well does the use of nuclear energy correlate with changes in carbon emissions? Are there characteristics of a country that correlate with increases or decreases in the use of nuclear energy?

Members:

- Josef Glas 08606876
- Felix Korbelius 01526132
- Frank Ebel 01429282
- Johannes Schabbauer 11776224

Work method:

Each person wrote python scripts for their respective tasks. These scripts were merged in this notebook by Frank and modified if necessary. The repostory is available on GitHub.

1.2 Loading necessary modules

Since we have widgets, this option must be run first:
`jupyter nbextension enable --py widgetsnbextension --sys-prefix`

[ ]:
```
# file manipulation
import requests
import re
import os

# working with data
import math
import numpy as np
import pandas as pd

# modelling
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.preprocessing import PolynomialFeatures
```

```
# country manipulation
import country_converter
import pycountry
import logging

# widgets
import ipywidgets as widgets

# visualization
import matplotlib.pyplot as plt
import seaborn as sns
import pycountry
import plotly.express as px
import mplcursors
```

[ ]:
```
# initialize CountryConverter as cc and disable warnings
country_converter.logging.getLogger().setLevel(logging.CRITICAL)
cc = country_converter.CountryConverter()
```

1.3 Finding appropiate datasets.

We decided that each person should focus on different categories. Each dataset should be sorted by the combination of country and year. To be consistent with country names we decided to use ISO 3166 alpha 3 country codes. They categories were divided the following way:

- Josef: energy consumption and production data
- Felix: ecological data (CO_2-emission, pollution, ...)
- Frank: economical data (GDP, income, growth, ...)
- Johannes: gouvernment and democracy indicators, operating reactors, accidents, nuclear warheads

Used datasets

All used datasets are in the folder `./data`.

- **U.S. Energy Information Administration**
  `USEIA/`
- **Data on CO2 and Greenhouse Gas Emissions by Our World in Data**
  `owid-co2-data.csv`
- **GDP (current USD)**
  `API_NY.GDP.MKTP.CD_DS2_en_csv_v2_1740389/`
- **GDP growth (annual %)**
  `API_NY.GDP.MKTP.CD_DS2_en_csv_v2_1740389/`
- **GDP per capita (current USD)**
  `API_NY.GDP.PCAP.CD_DS2_en_csv_v2_1740213`
- **GDP per capita growth (annual %)**
  `API_NY.GDP.PCAP.KD.ZG_DS2_en_csv_v2_1740284/`

- **Adjusted net national income per capita (current USD)**
  API_NY.ADJ.NNTY.PC.CD_DS2_en_csv_v2_1745486
- **Adjusted net national income per capita (annual % growth)**
  API_NY.ADJ.NNTY.PC.KD.ZG_DS2_en_csv_v2_1745488/
- **Power Reactor Information System**
  reactor_numbers_PRIS_IAEA.csv
- **Nuclear Wareheads per country**
  nuclear_warheads_1945_2016.csv
- **Gross domestic expenditure on R&D (GERD), GERD as a percentage of GDP**
  SCN_DS_16122020083400698.csv
- **Nuclear Power Accidents (Deaths and Costs)**
  C_id_35_NuclearPowerAccidents2016.csv
- **The Global State of Democracy Indices**
  gsodi_pv_4.csv

Dataset loading by Josef

[ ]:
```
def load_useia_data():
    data = pd.DataFrame()

    ### CONSUMPTION
    target = './data/USEIA/USEIA_CONSUMPTION_1980-2018.csv'

    consumption = pd.read_csv(target, sep=",", decimal=".", header=0,␣
↪skiprows=1, na_values='--')

    # rename columns
    consumption = consumption.rename(columns={'Unnamed: 1': 'text'})
    consumption.insert(loc=0, column='country', value=0)
    consumption.insert(loc=0, column='check', value='X')

    consumption['country'] = consumption['API'].str[-10:-7]
    #consumption = map_historic_countries(consumption)
    consumption['check'] = cc.convert(names=consumption['country'].to_list(),␣
↪to='ISO3')

    consumption = consumption.sort_values(by=['country'])

    # data cleaning

    # remove rows where country code check failed
    raw = consumption[consumption['check'] != 'not found']

    raw = raw.reset_index(drop=True)

    consumption = pd.DataFrame(columns=['year', 'country', 'cons_btu',␣
↪'coal_cons_btu', 'gas_cons_btu', \
```

```
                                        'oil_cons_btu', 'nuclear_cons_btu',␣
↪'renewables_cons_btu'])

    countries = pd.DataFrame(raw['country'])
    countries.sort_values(by=['country'])
    countries.drop_duplicates(inplace=True)
    countries = countries.reset_index(drop=True)

    counter = 0
    for idx1, c in countries.iterrows():

        temp = raw[raw['country'] == c.iloc[0]]

        j = 4
        for i in range(1980, 2019):

            v_cons = 0
            v_coal = 0
            v_gas = 0
            v_oil = 0
            v_nuclear = 0
            v_renewables = 0

            new_row = {'year': i, 'country': c.iloc[0]}
            for idx2, row in temp.iterrows():

                text = row.iloc[3]

                if text.find("Consumption") != -1:
                    v_cons = row.iloc[j]
                elif text.find("Coal") != -1:
                    v_coal = row.iloc[j]
                elif text.find("Natural gas") != -1:
                    v_gas = row.iloc[j]
                elif text.find("Petroleum") != -1:
                    v_oil = row.iloc[j]
                elif text.find("Nuclear (") != -1:
                    v_nuclear = row.iloc[j]
                elif text.find("Renewables and other") != -1:
                    v_renewables = row.iloc[j]

            new_row['cons_btu'] = v_cons
            new_row['coal_cons_btu'] = v_coal
            new_row['gas_cons_btu'] = v_gas
            new_row['oil_cons_btu'] = v_oil
            new_row['nuclear_cons_btu'] = v_nuclear
            new_row['renewables_cons_btu'] = v_renewables
```

```
            consumption.loc[counter] = new_row
            counter += 1
            j += 1

    ### PRODUCTION
    target = './data/USEIA/USEIA_PRODUCTION_1980-2018.csv'

    production = pd.read_csv(target, sep=",", decimal=".", header=0,␣
↪skiprows=1, na_values='--')

    # rename columns
    production = production.rename(columns={'Unnamed: 1': 'text'})
    production.insert(loc=0, column='country', value=0)
    production.insert(loc=0, column='check', value='X')

    production['country'] = production['API'].str[-10:-7]
    #production = map_historic_countries(production)
    production['check'] = cc.convert(names=production['country'].to_list(),␣
↪to='ISO3')

    production.sort_values('country')

    # data cleaning

    # remove rows where country code check failed
    raw = production[production['check'] != 'not found']

    raw = raw.reset_index(drop=True)

    production = pd.DataFrame(columns=['year', 'country', 'prod_btu',␣
↪'coal_prod_btu', 'gas_prod_btu', \
                                       'oil_prod_btu', 'nuclear_prod_btu',␣
↪'renewables_prod_btu'])

    countries = pd.DataFrame(raw['country'])
    countries.sort_values(by=['country'])
    countries.drop_duplicates(inplace=True)
    countries = countries.reset_index(drop=True)

    counter = 0
    for idx3, c in countries.iterrows():

        temp = raw[raw['country'] == c.iloc[0]]

        j = 4
        for i in range(1980, 2019):
```

```
            v_prod = 0
            v_coal = 0
            v_gas = 0
            v_oil = 0
            v_nuclear = 0
            v_renewables = 0

            new_row = {'year': i, 'country': c.iloc[0]}
            for idx4, row in temp.iterrows():

                text = row.iloc[3]

                if text.find("Production") != -1:
                    v_prod = row.iloc[j]
                elif text.find("Coal") != -1:
                    v_coal = row.iloc[j]
                elif text.find("Natural gas") != -1:
                    v_gas = row.iloc[j]
                elif text.find("Petroleum") != -1:
                    v_oil = row.iloc[j]
                elif text.find("Nuclear (") != -1:
                    v_nuclear = row.iloc[j]
                elif text.find("Renewables and other") != -1:
                    v_renewables = row.iloc[j]

            new_row['prod_btu'] = v_prod
            new_row['coal_prod_btu'] = v_coal
            new_row['gas_prod_btu'] = v_gas
            new_row['oil_prod_btu'] = v_oil
            new_row['nuclear_prod_btu'] = v_nuclear
            new_row['renewables_prod_btu'] = v_renewables

            production.loc[counter] = new_row
            counter += 1
            j += 1

    data = pd.merge(consumption, production, how="outer", on=['year',␣
↪'country'])

    data['year'] = data['year'].astype('int64')
    data['nuclear_prod_btu'] = data['nuclear_prod_btu'].astype('float64')

    return data
```

Dataset loading by Felix

```
[ ]: def load_emission_data():
        """
        Load all emission data files and combine them into a single Pandas␣
    ↪DataFrame.
        Common data structure: 0-year, 1-country code, 2+-features.
        Check for correct typing.

        return:
        emission_data: data frame containing different emission data per country␣
    ↪per year.
        """

        path = './data/owid-co2-data.csv'
        df_emission_data = pd.read_csv(path, sep=',')

        cols = ['year', 'iso_code']
        # Rearrange columns, so that year and country-code (iso-code) are the first␣
    ↪two columns.
        new_cols = cols + df_emission_data.columns.drop(cols).tolist()
        # Drop country column.
        df_emission_data = df_emission_data[new_cols].drop(['country'], axis=1)
        # Rename iso_code to country and convert to string.
        df_emission_data[['iso_code']] = df_emission_data[['iso_code']].
    ↪astype('string')
        df_emission_data = df_emission_data.rename(columns={'iso_code': 'country'})
        return df_emission_data


    def resize_emission(df):
        """ Index dataframe and eliminate non-country specific data.

        Attention: When handling NaN values look at the values of a specific␣
    ↪column, if there exists a NaN value
        above/below a 0 entry, it is highly possible that NaN are truly missing␣
    ↪values.

        Time-range: 1980-2018

        return:
        trimmed down and somewhat ordered emission_data."""
        data_emission_i = df.copy()
        # Only keep countries (check len(country_code) == 3) - raw data contains␣
    ↪continental data, etc. with a blank
        # country code (i.e. length 0).
        data_emission_i = data_emission_i[data_emission_i['country'].str.len() == 3]
        # Set index on country_code and year (group by country_code).
```

```
    # data_emission_i = data_emission_i.set_index(['country_code', 'year'])
    # Keep most interesting columns:
    data_emission_i = data_emission_i.drop(data_emission_i.iloc[:, -5:-2],␣
↪axis=1)
    data_emission_i = data_emission_i.drop(data_emission_i.iloc[:, 16:26],␣
↪axis=1)  # delete cement,... produc. emission
    data_emission_i = data_emission_i.drop(['gdp', 'trade_co2',␣
↪'trade_co2_share'], axis=1)
    return data_emission_i
```

Dataset loading by Frank

[ ]:
```
def load_economical_data():
    """Load economical data into dataframe and return it.

    Common data structure:
    0: year
    1: country code
    3..: features"""

    def get_df(filepath):
        df = pd.read_csv(filepath, sep=',', skip_blank_lines=True, header=2)
        df.drop(columns_drop, axis=1, inplace=True)
        df.rename(columns={'Country Code': 'country'}, inplace=True)
        return df

    columns_drop = ['Country Name', 'Indicator Name',  'Indicator Code',␣
↪'Unnamed: 65']  # columns to drop
    dfs = []  # List of all dataframes.

    # load dataframe of GDP
    df_GDP = get_df('./data/API_NY.GDP.MKTP.CD_DS2_en_csv_v2_1740389/API_NY.GDP.
↪MKTP.CD_DS2_en_csv_v2_1740389.csv')
    # melt and order to get in right format
    df_GDP = df_GDP.melt(id_vars=['country'], var_name='year', value_name='GDP')
    df_GDP['year'] = df_GDP['year'].astype('int64')
    dfs.append(df_GDP)

    # load dataframe of GDP growth
    df_GDP_growth = get_df('./data/API_NY.GDP.MKTP.KD.ZG_DS2_en_csv_v2_1836177/'
                           'API_NY.GDP.MKTP.KD.ZG_DS2_en_csv_v2_1836177.csv')
    # melt and order to get in right format
    df_GDP_growth = df_GDP_growth.melt(id_vars=['country'], var_name='year',␣
↪value_name='GDP growth')
    df_GDP_growth['year'] = df_GDP_growth['year'].astype('int64')
    dfs.append(df_GDP_growth)
```

```
    # load dataframe of GDP per capita
    df_GDP_per_capita = get_df('./data/API_NY.GDP.PCAP.CD_DS2_en_csv_v2_1740213/
↪'
                               'API_NY.GDP.PCAP.CD_DS2_en_csv_v2_1740213.csv')
    # melt and order to get in right format
    df_GDP_per_capita = df_GDP_per_capita.melt(id_vars=['country'],␣
↪var_name='year', value_name='GDP per capita')
    df_GDP_per_capita['year'] = df_GDP_per_capita['year'].astype('int64')
    dfs.append(df_GDP_per_capita)

    # load dataframe of GDP per capita growth
    df_GDP_per_capita_growth = get_df('./data/API_NY.GDP.PCAP.KD.
↪ZG_DS2_en_csv_v2_1740284/'
                                      'API_NY.GDP.PCAP.KD.
↪ZG_DS2_en_csv_v2_1740284.csv')
    # melt and order to get in right format
    df_GDP_per_capita_growth = df_GDP_per_capita_growth.
↪melt(id_vars=['country'], var_name='year',
                                                          value_name='GDP␣
↪per capita growth')
    df_GDP_per_capita_growth['year'] = df_GDP_per_capita_growth['year'].
↪astype('int64')
    dfs.append(df_GDP_per_capita_growth)

    # load dataframe of income per capita
    df_income_per_capita = get_df('./data/API_NY.ADJ.NNTY.PC.
↪CD_DS2_en_csv_v2_1745486/'
                                  'API_NY.ADJ.NNTY.PC.CD_DS2_en_csv_v2_1745486.
↪csv')
    # melt and order to get in right format
    df_income_per_capita = df_income_per_capita.melt(id_vars=['country'],␣
↪var_name='year',
                                                     value_name='income per␣
↪capita')
    df_income_per_capita['year'] = df_income_per_capita['year'].astype('int64')
    dfs.append(df_income_per_capita)

    # load dataframe of income per capita growth
    df_income_per_capita_growth = get_df('./data/API_NY.ADJ.NNTY.PC.KD.
↪ZG_DS2_en_csv_v2_1745488/'
                                         'API_NY.ADJ.NNTY.PC.KD.
↪ZG_DS2_en_csv_v2_1745488.csv')
    # melt and order to get in right format
    df_income_per_capita_growth = df_income_per_capita_growth.
↪melt(id_vars=['country'], var_name='year',
```

```
↪value_name='income per capita growth')
    df_income_per_capita_growth['year'] = df_income_per_capita_growth['year'].
↪astype('int64')
    dfs.append(df_income_per_capita_growth)

    # merge and sort all dataframes
    result = dfs[0]
    for df in dfs[1:]:
        result = result.merge(df, how='outer', on=['country', 'year'])
    result.sort_values(['country', 'year'], inplace=True)
    result.reset_index(inplace=True, drop=True)

    # Since there are some aggregated values (e. g. WLD for world) remove all␣
↪rows which don't have a valid
    # ISO 3166 Alpha-3 code.
    alpha_3_list = [country.alpha_3 for country in list(pycountry.countries)] ␣
↪# all valid codes
    valid_entry = result['country'].isin(alpha_3_list)  # boolean series if␣
↪each row is valid or not
    result = result.loc[valid_entry]
    # invalid = set(result.loc[~valid_entry]['country'].tolist())
    # print('invalid code\n', invalid)

    return result
```

Dataset loading by Johannes

[ ]:
```
# initialize CountryConverter as cc and disable warnings
country_converter.logging.getLogger().setLevel(logging.CRITICAL)
cc = country_converter.CountryConverter()
# dictionary for country replacements (that cannot be read by country_converter)
# using current ones for outdated names, e.g. 'USSR' --> 'Russia'
_dict_country_repl = {'UK':'United Kingdom', 'USSR':'Russia', 'Soviet Union':
↪'Russia', 'East Germany':'Germany',
                      'Illinois':'US', 'Tawian':'Taiwan', 'Yugoslavia':
↪'Serbia', 'Scotland':'United Kingdom'}

##########################################################################################
def load_political_data():
    # read data from diffenernt datasets in the category 'political'

    # nuclear warheads
    # read file and exclude last (empty) line
    warheads = pd.read_csv('./data/nuclear_warheads_1945_2016.csv',
                           sep=';',thousands='.', decimal=',').iloc[:-1]
    warheads['Year'] = warheads['Year'].astype('int')
```

```
    warheads = warheads.set_index('Year')
    # extent years to 2020 and interpolate
    warheads = warheads.reindex(np.arange(1945,2021,dtype='int'),␣
↪method='ffill')
    # transform datafame from 2D to MultiIndex
    warheads = warheads.stack()
    warheads = warheads.reset_index()
    # set column names and convert country names to ISO3
    warheads.columns = ['year', 'country', 'nuclear_warheads']
    warheads['country'] = cc.convert(warheads['country'].to_list(), to='ISO3')
    warheads = warheads.set_index(['year', 'country'])


    # research expenditure
    research = pd.read_csv('./data/SCN_DS_16122020083400698.csv')
    # choose only lines with relatilve expenditure (for all reaseach categories)
    research = research.loc[research['Indicator']=="GERD as a percentage of␣
↪GDP"]
    # chose relevant columns and rename them
    research = research[['Time','Country', 'Value']]
    research.columns = ['year', 'country', 'research_%GDP']
    # convert to ISO3 and exclude regions (cannot be converted to countrycode)
    research['country'] = research['country'].replace({'Oceania (Australia/New␣
↪Zealand)':'not found'})
    research['country'] =  cc.convert(research['country'].to_list(), to='ISO3',␣
↪not_found='not found')
    research = research[research['country'] != 'not found']
    research = research.set_index(['year', 'country'])


    # accidents of nuclear power plants
    accidents = pd.read_csv('./data/C_id_35_NuclearPowerAccidents2016.csv')
    accidents = accidents[['Date', 'Location', 'Cost (millions 2013US$)',␣
↪'Fatalities']]
    accidents.columns = ['year', 'country', 'accident_cost_MioUSD2013',␣
↪'accident_deaths']
    # use only year from Date column
    accidents['year'] = accidents['year'].str.slice(start=-4).astype('int')
    # use last part of Location (usually the country)
    accidents['country'] = accidents['country'].str.split(',').str[-1].str.
↪lstrip(' ')
    # do some corrections (e.g. old country names or missing ones)
    accidents['country'] = accidents['country'].replace(_dict_country_repl)
    # conversion to ISO3
    accidents['country'] = cc.convert(accidents['country'].to_list(), to='ISO3')
    accidents = accidents.set_index(['year', 'country'])
```

```
    # sum values, if there was more than one accident per year and country
    accidents = accidents.sum(level=['year','country'])

    # democarcy indicators
    democracy = pd.read_csv('./data/gsodi_pv_4.csv', low_memory=False)
    # choose five main categories
    democracy =␣
↪democracy[['ID_year','ID_country_name','C_A1','C_A2','C_A3','C_A4','C_SD51']]
    democracy.columns = ['year', 'country', 'representative_government',␣
↪'fundamental_rights',
                         'checks_on_gouvernment', 'impartial_administration',␣
↪'civil_society_participation']
    # avoid that 'Southern Africa' is converted to 'ZAF' and count 'East␣
↪Germany' as 'Germany'
    democracy['country'] = democracy['country'].replace(
            {'Southern Africa':' ','German Democratic Republic':'Germany'})
    democracy['country'] = cc.convert(democracy['country'].to_list(), to='ISO3')
    # exclude regions (and east germany)
    democracy = democracy[democracy['country'] != 'not found']
    democracy = democracy.set_index(['year', 'country'])
    # use mean value for duplicate values (EAST and WEST GERMANY)
    democracy = democracy.mean(level=['year','country'])

    # get number of reactors from seperate function
    reactors = load_reactor_numbers()

    # merging and fill some of the missing values
    merge = pd.concat(
            [reactors,warheads,accidents,research,democracy],
            axis=1, join='outer')

    merge = merge.sort_index(level=['country'])

    return merge

##########################################################################################
def load_reactor_numbers():
    # loading number of operational nuclear power plants from IAEA-PRIS␣
↪database (public version)

    # if data was already loaded from webpages, read directly from saved csv␣
↪file
    if os.path.isfile('./data/reactor_numbers_PRIS_IAEA.csv'):
        reactors = pd.read_csv('./data/reactor_numbers_PRIS_IAEA.csv',␣
↪index_col=[0,1])
        return reactors
```

```
    # create containers for reactor data per country
    startup_dict=dict()
    shutdown_dict=dict()

    # fetch table for reactors from public webpage
    url = 'https://pris.iaea.org/PRIS/CountryStatistics/ReactorDetails.aspx?
↪current='
    for num in range(1000): # manual maximal id of reactor
        page = requests.get(url+str(num))
        if page.status_code < 400: # exclude non-existing IDs
            # find country (ISO2) in html and load tables from page
            country = re.findall('[\d\D]*color="DarkGray"', str(page.
↪content))[0][-26:-24]
            country = cc.convert(country, src='ISO2', to='ISO3')
            # create dict entries for new countries
            if country not in startup_dict.keys():
                startup_dict[country] = np.empty(shape=0, dtype='int')
                shutdown_dict[country] = np.empty(shape=0, dtype='int')
            page_df = pd.read_html(page.content)
            if len(page_df) < 3: # exclude reactor if never started
                continue
            # get year of startup
            if page_df[0].iloc[6,1]=='Commercial Operation Date':
                # if 'Commercial Operation Date' is not given (NaN), use 'First␣
↪Grid Connection'
                if type(page_df[0].iloc[7,1]) != 'str':
                        startup_dict[country] = np.
↪append(startup_dict[country], int(page_df[0].iloc[7,0][-4:]))
                else:
                    startup_dict[country] = np.append(startup_dict[country],␣
↪int(page_df[0].iloc[7,1][-4:]))
            # get year of reactor shutdown (if given)
            if page_df[0].iloc[8,0]=='Permanent Shutdown Date':
                shutdown_dict[country] = np.append(shutdown_dict[country],␣
↪int(page_df[0].iloc[9,0][-4:]))

    # calculate operating reactors from startup and shutdown dates
    # of each reactor (from dicts) for each country per year
    reactors = pd.DataFrame()
    for ISO in startup_dict.keys():
        if len(startup_dict[ISO])==0:
            continue
        reactors_country = pd.DataFrame()
        reactors_country['year'] = np.arange(startup_dict[ISO].min(),2021)
```

```
        reactors_country['country'] = np.full(shape=reactors_country.shape[0],␣
↪fill_value=ISO)
        reactors_country['built_reactors'] = np.fromiter(
                (startup_dict[ISO][startup_dict[ISO] <= year].size for year in␣
↪reactors_country['year'] )
                ,dtype='int')
        reactors_country['shutdown_reactors'] = np.fromiter(
                (shutdown_dict[ISO][shutdown_dict[ISO] <= year].size for year␣
↪in reactors_country['year'] )
                ,dtype='int')
        reactors_country['operating_reactors'] =␣
↪reactors_country['built_reactors'] - reactors_country['shutdown_reactors']
        reactors = pd.concat([reactors, reactors_country],axis=0)
    reactors = reactors.set_index(['year', 'country'])
    # save DataFrame to csv-file, to fetch data not everytime
    reactors.to_csv('./data/reactor_numbers_PRIS_IAEA.csv')
    return reactors
```

1.3.1 Merging datasets

This was worked on by Frank. The function clean_data_after_merge() was written by Johannes. Since running the cell below takes a lot of time, the merged and cleaned dataframe was written to `./data/data_merged/data.csv`. For exploring the data, loading the csv was much faster than running the code in the next cell each time.

[ ]:
```
def clean_data_after_merge(df):
    """Fill some missing data in merged dataframe."""

    for column in ['built_reactors', 'shutdown_reactors', 'operating_reactors',␣
↪'nuclear_warheads']:
        df[column].fillna(value=0, inplace=True)
    for column in ['accident_cost_MioUSD2013', 'accident_deaths']:
        df[column].fillna(value=0, inplace=True)


df_energy = load_useia_data()
df_emission = resize_emission(load_emission_data())
df_economy = load_economical_data()
df_politics = load_political_data()
df_politics.reset_index(drop=False, inplace=True)

# merge all dataframes:
dataframe = df_energy
for df in [df_emission, df_economy, df_politics]:
    dataframe = dataframe.merge(df, how='left', on=['year', 'country'])

# clean up some values
```

```
clean_data_after_merge(dataframe)

# export to csv
export = False
if export:
    dataframe.to_csv('../data/data_merged/data.csv', index=False)
```

1.4 Observation and Comments about used datasets:

This part was written by all members.

Question 1 was answered by data of primary energy production, while for question 2 electrical energy production was used. As mentioned above, the merged dataframe was saved seperately. A codebook `description.csv` was produced in conjunction which explains what information is contained in each column.

Josef With regard to engery sources we considered two different data sets: - a. the statistical review of world energy from bp.com - b. data set about production and consumption of primary energy sources from U.S. Energy Information Administration www.eia.gov We regarded the data from UAEIA as more suitable, because: - data available on country level (without aggregations for regions), bp used aggregations for smaller countries (like "other south america") - consistent data for time range 1980-2018 - appropriate split of energy sources into 5 categories: coal, natural gas, petroleum and other liquids, nuclear, renowables/other (Remark: bp is using a slightly different break down) We continued with a. and performed no data cleansing with two exceptions: - instead of quadrillion btu (british therminal unit) we converted the figures into exajoule - formation of new nations: On an aggregated (world) level figures are consistent, however on country level we had to keep in mind that some coutries disappeared and new countries have been created (e.g. dissolution of the Soviet Union or Yugoslavia). In case of Soviet Union we manually merged energy data with Russia, but for other countries we ignored this fact. In the end this aspect was not an issue, because we mostly focused on the time frame 1998 to 2018 (past 20 years).

Frank

- Since it was decided beforehand what each person had to search, it was much easier for me to narrow down what to look for. I found data in .csv and .xslx formats. Of these I thought that .csv formats are easier to work in python with.
- Some web results only offer datasets behind a paywall, which could not be used for this exercise.
- I asked other members if we should include datasets like World Happiness Report We decided against it, since these rankings are mostly aggregated values of the given datasets.

Felix To answer question 2 I focused on the correlation between CO2 emissions from electricity/ heat generation and electricity production in units of energy. - CO2 emission data was gathered from https://ourworldindata.org and https://www.climatewatchdata.org - Electricitiy production data was gathered from https://www.eia.gov/. - Since I used data from different sources, they had to be each handled a bit differently. CO2 data from ourworldindata ranged from 1751 to 2018 while data from climatewatchdata ranged from 1990 to 2017. Production data from eia ranges from 1980 to 2019. All the data has been adjusted so that it fits the investigated time span from 1990 to 2017, since this range granted the least amount of missing values. - Missing values (NaNs/ zeroes) were mainly an issue after merging the different data sets (outer join), due to the different sizes

(time range) of raw data. This could be solved by narrowing down the time span, where most of the data was available(since extrapolation would alter the actual behaviour of the data). - Since we planned to use country codes according to ISO 3166-1 alpha-3, some data (production data from eia) had to be transformed to be able to merge it with the other production related data (they already possessed a column with ISO 3166-1 alpha-3 codes).

Johannes - Regarding the datasets with the number of nucelar reactors, the accidents in nuclear power plants and the warheads, the data can be expected to be complete, and all years and countries can be filled with the values 0 without expecting a distortion (e.g. if a country does not have nuclear reactors, it just is not listed in the PRIS database). - One small issue with the data of nuclear warheads was, the the information was only provided until 2016. This was fixed by extending the years until 2018 (forward-fill of values), which will not cause a considerable error, because this affects only two years. - The data for research expenduture (in %GDP) is provided by the UNESCO only from the year 1996, and also some countries are missing. Extrapolationg to all year from 1980 to 2018 would cause a significant distortion. Instead, the correltion with changes in nuclear energy usage should be only calculated in time ranges after 1996 (e.g. from 1998-2018 as in the last section of the notebook.

1.5 First Visualization of data

The code for the cell below was written by Josef. The code for the dropdown widget was added by Frank.

[ ]:
```
def load_data_for_plot():

    desc_file = './data/data_merged/description.csv'
    data_file = './data/data_merged/data.csv'

    desc = pd.read_csv(desc_file, sep=",", header=None)
    desc.set_index(0, inplace=True)
    data = pd.read_csv(data_file, sep=",", decimal=".")



    return data, desc


def show_map(df, desc, feature, scope):
    # print(df)
    # print(desc)

    minimum = df[feature].min()
    maximum = df[feature].max()

    fig = px.choropleth(data_frame=df,
                        locations="country",
                        color=feature,  # value in feature column determines␣
 ↪color
```

```
                        hover_name="country",
                        scope=scope,
                        color_continuous_scale='Reds',  # color scale
                        range_color=(minimum, maximum),
                        animation_frame="year",
                        title='Development of feature ' + feature + ': ' + desc)
    # do not show antarctica in world map
    if scope == 'world':
        fig.layout.geo.lataxis.range = [-55, 90]
    fig.show()


def wrapper(feature):
    description = desc.loc[feature][1]
    # change scope if necessary (world, usa, europe, asia, africa, north␣
 ↪america, ...)
    show_map(df, description, feature, 'world')


df, desc = load_data_for_plot()

options = df.columns.drop(['year', 'country'])
widgets.interact(wrapper, feature=options);
```

The code for the cell below was written by Johannes.

[ ]:
```
def plot_operating_reactors():
    """
    Show how the number of operating reactors has evolved over time worldwide.
    """
    # get data of nuclear reactors
    data = pd.read_csv('./data/reactor_numbers_PRIS_IAEA.csv', index_col=[0,1])

    data_sum = data.sum(axis=0, level='year').sort_index()
    data_sum.plot()
    plt.legend(loc='best')
    plt.title('Total Nuclear Reactors Worldwide')
    plt.xlim(data_sum.index.min(), data_sum.index.max())
    plt.grid()
    plt.show()

plot_operating_reactors()
```

1.6 Question 1: How has the use of nuclear energy evolved over time?

The code to answer this question was generated by Josef.

[ ]:
```
def load_data_q1():
    desc_file = './data/data_merged/description.csv'
    data_file = './data/data_merged/data.csv'

    data = pd.read_csv(data_file, sep=",", decimal=".")

    data = data[['year', 'country',
                 'cons_btu', 'coal_cons_btu', 'gas_cons_btu', 'oil_cons_btu',␣
 ↪'nuclear_cons_btu', 'renewables_cons_btu',
                 'prod_btu', 'coal_prod_btu', 'gas_prod_btu', 'oil_prod_btu',␣
 ↪'nuclear_prod_btu',
                 'renewables_prod_btu', 'accident_deaths',␣
 ↪'operating_reactors']]

    # convert quad btu in EJ (except for first two and last two columns)
    conversion_factor = 1.055
    data.iloc[:,2:-2] = data.iloc[:,2:-2] * conversion_factor
    return data


def show_plot0(df):
    df1 = df[['year', 'oil_prod_btu', 'coal_prod_btu', 'gas_prod_btu',␣
 ↪'nuclear_prod_btu', 'renewables_prod_btu']]

    df1 = df1.groupby(['year']).sum()

    y = [df1["nuclear_prod_btu"], df1["oil_prod_btu"], df1["coal_prod_btu"],␣
 ↪df1["gas_prod_btu"],
         df1["renewables_prod_btu"]]

    colors = ['yellow', 'dimgray', 'black', 'darkcyan', 'green']
    labels = ['nuclear', 'oil', 'coal', 'gas', 'renewables and other']

    plt.stackplot(df1.index, y, labels=labels, colors=colors)

    plt.title('Overall energy production 1980-2018')
    plt.xlabel(xlabel='year')
    plt.ylabel(ylabel='production in EJ')
    plt.legend(loc='upper left')
    plt.xlim(df['year'].min(), df['year'].max())

    plt.show()

    return


def show_plot1(df):
```

```
    df1 = df[['year', 'oil_prod_btu', 'coal_prod_btu', 'gas_prod_btu',␣
↪'nuclear_prod_btu', 'renewables_prod_btu']]

    df1 = df1.groupby(['year']).sum()

    y = [df1["nuclear_prod_btu"], df1["oil_prod_btu"], df1["coal_prod_btu"],␣
↪df1["gas_prod_btu"],
         df1["renewables_prod_btu"]]

    y0 = (y[0] / (y[0] + y[1] + y[2] + y[3] + y[4]) * 100)
    y1 = (y[1] / (y[0] + y[1] + y[2] + y[3] + y[4]) * 100)
    y2 = (y[2] / (y[0] + y[1] + y[2] + y[3] + y[4]) * 100)
    y3 = (y[3] / (y[0] + y[1] + y[2] + y[3] + y[4]) * 100)
    y4 = (y[4] / (y[0] + y[1] + y[2] + y[3] + y[4]) * 100)

    percent = [y0, y1, y2, y3, y4]

    colors = ['yellow', 'dimgray', 'black', 'darkcyan', 'green']
    labels = ['nuclear', 'oil', 'coal', 'gas', 'renewables and other']

    plt.stackplot(df1.index, percent, labels=labels, colors=colors)

    plt.title('Distribution of energy production 1980-2018')
    plt.xlabel(xlabel='year')
    plt.ylabel(ylabel='production in %')
    plt.legend(loc=(0.01, 0.1))
    plt.xlim(df['year'].min(), df['year'].max())
    plt.ylim((0, 100))
    ax2 = plt.twinx()
    ax2.set_ylim((0, 100))

    plt.show()

    return


def show_plot2(df):
    df1 = df[['year', 'oil_prod_btu', 'coal_prod_btu', 'gas_prod_btu',␣
↪'renewables_prod_btu', 'nuclear_prod_btu']]
    df1 = df1.groupby(['year']).sum()
    df1 = df1.rename(columns={'oil_prod_btu': 'oil', 'coal_prod_btu': 'coal',␣
↪'gas_prod_btu': 'gas',
                             'renewables_prod_btu': 'renewables',␣
↪'nuclear_prod_btu': 'nuclear'})
    df1 = df1.reset_index()

    df2 = df[['year', 'accident_deaths']]
```

```
    df2 = df2.groupby(['year']).sum()

    ax = plt.gca()

    df1.plot(kind='line', x='year', y='nuclear', color='yellow', ax=ax,␣
↪linewidth=3)
    df1.plot(kind='line', x='year', y='oil', color='dimgray', ax=ax,␣
↪linewidth=3)
    df1.plot(kind='line', x='year', y='coal', color='black', ax=ax, linewidth=3)
    df1.plot(kind='line', x='year', y='gas', color='darkcyan', ax=ax,␣
↪linewidth=3)
    df1.plot(kind='line', x='year', y='renewables', color='green', ax=ax,␣
↪linewidth=3)

    plt.title('energy prod per energy source incl deaths in nuclear power␣
↪plants')
    plt.xlabel(xlabel='year')
    plt.ylabel(ylabel='production in EJ')
    plt.legend(loc='upper left')
    plt.grid()
    plt.xlim(df['year'].min(), df['year'].max())

    shift = -4
    for x, y in zip(df1['year'], df1['nuclear']):
        label = df2.loc[x]
        if label[0] > 0:
            plt.annotate(label[0].astype('int32'), (x, y + shift),␣
↪fontweight='bold')
            shift = shift * -1

    plt.show()

    return


def show_plot3(df):  # top 10 nuclear energy producers 1980 vs 2018

    df['year'] = df['year'].astype('int32')

    df1 = df.where(df["year"] == 1998)
    df2 = df1.groupby(['country']).sum()
    df2 = df2.sort_values(by=['nuclear_prod_btu'], ascending=False)
    df2 = df2[['nuclear_prod_btu', 'operating_reactors']].head(20)
    df2.rename(columns={'operating_reactors': 'op_reactors_1998',␣
↪'nuclear_prod_btu': 'nuclear_prod_1998'},
               inplace=True)
```

```
    df2.insert(0, 'rank1998', range(1, 21))

    df3 = df.where(df["year"] == 2018)
    df4 = df3.groupby(['country']).sum()
    df4 = df4.sort_values(by=['nuclear_prod_btu'], ascending=False)
    df4 = df4[['nuclear_prod_btu', 'operating_reactors']].head(20)
    df4.rename(columns={'operating_reactors': 'op_reactors_2018',␣
↪'nuclear_prod_btu': 'nuclear_prod_2018'},
               inplace=True)
    df4.insert(0, 'rank2018', range(1, 21))

    df5 = df2.merge(df4, how='outer', on=['country'])
    df5 = df5.fillna(0)
    df5.insert(0, 'movement', (df5['rank1998'] - df5['rank2018']).
↪astype('int32'))

    df5['movement'] = np.where((df5.rank1998 == 0), 0, df5.movement)
    df5['movement'] = np.where((df5.rank2018 == 0), 0, df5.movement)

    df6 = df.groupby(['country']).sum()
    df6 = df6[['accident_deaths']]

    df7 = df5.merge(df6, how='left', on=['country'])

    # print(df7)

    df8 = df7.drop(columns=['accident_deaths', 'op_reactors_1998',␣
↪'op_reactors_2018',
                            'movement', 'rank2018', 'rank1998'])

    # figsize
    fig, ax = plt.subplots(figsize=(16, 9))

    df8.plot(kind="bar", ax=ax)
    plt.title('Nuclear production comparison 1998, 2018')
    plt.xlabel(xlabel='')
    plt.ylabel(ylabel='production in EJ')
    plt.legend(loc='upper right')

    plt.gca().set_xticks([])

    df9 = df7[['rank1998', 'rank2018', 'movement', 'op_reactors_1998',␣
↪'op_reactors_2018', 'accident_deaths']]
    df9.insert(5, 'change2', (df9['op_reactors_2018'] -␣
↪df9['op_reactors_1998']).astype('int32'))
    df9['change2'] = np.where((df9.op_reactors_1998 == 0), 0, df9.change2)
    df9['change2'] = np.where((df9.op_reactors_2018 == 0), 0, df9.change2)
```

```
    df9 = df9.T
    rowlabels = ['rank 1998', 'rank 2018', 'delta ranking', 'reactors 1998',␣
 ↪'reactors 2018', 'delta reactors', 'accident deaths']
    the_table = plt.table(cellText=df9.astype('int').values,
                          rowLabels=rowlabels,
                          colLabels=df9.columns,
                          cellLoc='right', rowLoc='center',
                          loc='bottom')
    the_table.auto_set_font_size(False)
    the_table.set_fontsize(14)
    the_table.scale(1,2)

    plt.subplots_adjust(bottom=0.3)

    plt.show()
    return


def show_plot4(df):
    df0 = df[['year', 'renewables_prod_btu', 'nuclear_prod_btu', 'country',␣
 ↪'accident_deaths']]
    df0.index = df0.year

    df1 = df0[df0['country'] == 'JPN']
    df2 = df0[df0['country'] == 'UKR']

    df1 = df1.rename(columns={'renewables_prod_btu': 'renewables JPN',␣
 ↪'nuclear_prod_btu': 'nuclear JPN'})
    df2 = df2.rename(columns={'renewables_prod_btu': 'renewables UKR',␣
 ↪'nuclear_prod_btu': 'nuclear UKR'})

    ax = plt.gca()

    df1.plot(kind='line', x='year', y='nuclear JPN', color='yellow', ax=ax,␣
 ↪linewidth=3)
    df1.plot(kind='line', x='year', y='renewables JPN', color='green', ax=ax,␣
 ↪linewidth=3)
    df2.plot(kind='line', x='year', y='nuclear UKR', color='yellow',␣
 ↪linestyle='dashed', ax=ax,
             linewidth=3)
    df2.plot(kind='line', x='year', y='renewables UKR', color='green',␣
 ↪linestyle='dashed', ax=ax,
             linewidth=3)


    plt.title('Comparison of energy prod in JPN and UKR')
```

```
    plt.xlabel(xlabel='years')
    plt.ylabel(ylabel='production in EJ')
    plt.legend(loc='upper left')
    plt.grid()

    shift = 0
    for x, y in zip(df1.year, df1['nuclear JPN']):
        label = df1.loc[x].accident_deaths
        # print(label)
        if label > 0:
            plt.annotate(label.astype('int32'), (x, y + shift),␣
↪fontweight='bold')
            shift = shift * -1

    shift = 0
    for x, y in zip(df2.year, df2['nuclear UKR']):
        label = df2.loc[x].accident_deaths
        # print(label)
        if label > 0:
            plt.annotate(label.astype('int32'), (x, y + shift),␣
↪fontweight='bold')
            shift = shift * -1

    plt.show()


df = load_data_q1()
show_plot0(df)
show_plot1(df)
show_plot2(df)
show_plot3(df)
show_plot4(df)
```

Observations

- Overall energy production
  Shows overall development of energy production over time in EJ. In the selected timeframe (1980 to 2018) the overall energy requirements almost doubled. Annual production of all types of energy climbed. oil +45 %, coal +140 %, gas +167 %, nuclear +283 %, renewables + 238 %.
- Distribution of energy production
  The stacked area chart is focusing on the composition (by energy source). In 1980 we see about 90% fossil energy sources, 3% nuclear and 7% renewables. Whereas in 2018 fossil energy sources drop to 84%, nuclear 4.4% and renewables 11.4%
- line plot with annotations
  Shows energy production per energy source including reported number of deaths from accidents in nuclear power plants. Annual increase in average (1980-2018) +3,7% nuclear produc-

tion and +3,2% renewables / others. However since 2000, nuclear energy production started to stagnate. But production of renewables doubled (from 32 to 69 EJ) Remark coal production: The reason of the climb and the shape of the curve is due to the rapid growth in China, which is predominant on the coal market.

- Bar plot with data table attached
  USA and France are leading the ranking in 1998 as well as 2018. China catched up and is now #3 The top 7 countries increased production in selected timeframe. Further observations: EU countries decrease production and a reduce number of operating reactors. Japan reduced production and number of reactors significantly, obviously driven by recent accident of 2011. In the Ukraine the development stagnates. No decline due to Chernobyl disaster in 1986.

- Comparison of energy production in JPN and UKR
  Since Japan and Ukraine had the biggest nuclear catastrophes, we decided to selectively have a closer look at these two countries. The Fukushima Daiichi Accident happened in March 2011. An immediate drop of nuclear power production can be observed since Japan decided to shut down almost all reactors. The Chernobyl disaster occured in April 1986. Since Ukraine was part of the USSR that time, we do not have data about energy production. The data shows that the country does not have major changes in nuclear and renewable energy production since 1992.

1.7 Question 2: How well does the use of nuclear energy correlate with changes in carbon emissions?

The code to answer this question was generated by Felix.

[ ]:
```
def load_df():
    """Load data_merged and data_electrical and merge into a pandas dataframe.
    """
    df_load_electric = pd.read_csv('./data/data_merged/data_electrical.csv')
    df_load_primary = pd.read_csv('./data/data_merged/data.csv')
    df_load = df_load_primary.merge(df_load_electric, how='outer',␣
↪on=['country', 'year'])
    desc_df_emission = pd.read_csv('./data/data_merged/description.csv')
    # Reduce to relevant dataframe:
    df_e = df_load[['year', 'country', 'co2', 'consumption_co2',␣
↪'cumulative_co2', 'population', 'Electricity/Heat',
                    'Transportation', 'Manufacturing/Construction', 'Other',␣
↪'Fugitive Emissions', 'prod_electric',
                    'prod_electric_nuclear', 'prod_electric_fossil',␣
↪'prod_electric_renewable']]
    df_emission = df_e.copy()
    # Convert quad BTU to exajoules:
    convert = ['prod_electric', 'prod_electric_nuclear',␣
↪'prod_electric_fossil', 'prod_electric_renewable']
    df_emission[convert] = df_e[convert].multiply(3.6e-3)
    # df_emission['fossil_production_btu'] = df_emission['coal_prod_btu'] +␣
↪df_emission['oil_prod_btu'] + df_emission[
```

```
    #     'gas_prod_btu']
    return df_emission, desc_df_emission


def corr_matrix(df):
    # Heatmap for correlation visualization.
    year = 1990

    df_heatmap = df.copy()
    df_heatmap = df_heatmap[df_heatmap['year'] >= year]
    fig, ax = plt.subplots(figsize=[10, 6])
    fig.suptitle(
        r'Correlation matrix of worldwide energy-related CO$_2$ emissions' +␣
↪'\n and electricity production from '
        + str(year) + ' to 2018.', fontsize=16)
    sns.heatmap(df_heatmap.drop(['year', 'country'], axis=1).
↪corr(method='pearson'), annot=True, cmap='coolwarm',
                vmin=-1, vmax=1)
    plt.subplots_adjust(left=0.2, bottom=0.31)
    plt.show()
    return


def corr(df, country):
    df_country = df[['year', 'country', 'prod_electric_fossil',␣
↪'prod_electric_nuclear', 'prod_electric_renewable',
                     'Electricity/Heat']]
    df_country_corr = df_country[df_country['country'] == str(country)].
↪drop(['country', 'year'], axis=1).corr()
    return df_country_corr


def rel_growth(df, start, stop):
    # Emission script
    df_e = df[['year', 'country', 'Electricity/Heat', 'prod_electric',␣
↪'prod_electric_nuclear', 'prod_electric_fossil',
               'prod_electric_renewable']]
    df_yearly = df_e.groupby(['year']).sum()  # Sum over all countries for a␣
↪given year.
    # Only take production feature.
    features = [feature for feature in df_yearly.columns]
    df_yearly = df_yearly[features]
    df_yearly.sort_index(inplace=True)

    # compare relative growth in % between first and last year
    growth = (df_yearly.loc[stop] / df_yearly.loc[start] - 1) * 100
```

```
    return growth


def plot_pie(df):
    # Look at CO2 emission in the energy sector

    df1 = df[
        ['year', 'Electricity/Heat', 'Transportation', 'Manufacturing/
↪Construction', 'Other', 'Fugitive Emissions']]
    df1 = df1.groupby(['year']).sum()

    # Generate a sum of the columns respectively for a pie plot
    fig, ax1 = plt.subplots(figsize=[10, 6])
    df1.loc['Total'] = df[
        ['Electricity/Heat', 'Transportation', 'Manufacturing/Construction',␣
↪'Other', 'Fugitive Emissions']].sum()
    df_pie = df1.loc['Total'].T
    df_pie.plot.pie(autopct="%.1f%%", title=r"Distribution of worldwide CO$_2$␣
↪emissions in the energy sector", ylabel='')
    plt.show()

    # Save plot as .pdf and .png
    save = False
    if save:
        fig.savefig('./figures/q2/q2_plot_pie.pdf', bbox_inches='tight')
        fig.savefig('./figures/q2/q2_plot_pie.png', bbox_inches='tight',␣
↪dpi=300)
    return


def plot1_world_abs(df):
    # How well does the use of nuclear energy correlate with changes in carbon␣
↪emissions in heat/electricity production.
    x = 'year'
    y0 = 'prod_electric_renewable'
    y1 = 'prod_electric_fossil'
    y2 = 'prod_electric_nuclear'
    y3 = 'Electricity/Heat'

    # Aggregate for lineplots
    df_line = df.groupby(['year']).sum()

    # Aggregate for barplots
    df_bar = df[['year', 'Electricity/Heat']]
    df_bar = df_bar.groupby(['year']).sum()  # otherwise, barplot shows␣
↪different color for every country
```

```
    # Instantiate figure
    fig, ax1 = plt.subplots(figsize=[10, 6])
    ax1.set_xlim(1990, 2017)
    sns.set_style('whitegrid')
    ax1.set_xlabel('year')
    ax1.set_ylabel(r'emissions in Mt CO$_2$')
    plt.bar(x=df_bar.index, height=df_bar[y3], width=0.75, alpha=0.4,␣
↪align='center',
            label=r'CO$_2$ emissions from electricity and heat generation')
    ax2 = ax1.twinx()  # instantiate a second axes that shares the same x-axis
    ax2.set_ylabel('production in EJ')
    sns.lineplot(data=df_line, x=x, y=y0, ax=ax2, color='green', ci=None,␣
↪label='renewables electricity production',
                 alpha=0.5, legend=False)
    sns.lineplot(data=df_line, x=x, y=y1, ax=ax2, color='brown', ci=None,␣
↪label='fossil fuel electricity production',
                 legend=False)
    sns.lineplot(data=df_line, x=x, y=y2, ax=ax2, color='yellow', ci=None,␣
↪label='nuclear electricity production',
                 legend=False)

    fig.suptitle(r'Electricity production compared to CO$_2$ emissions -␣
↪World', fontsize=16)
    plt.annotate('Fukushima', xy=(2011, 52), xytext=(2011, 58), ha="center",␣
↪va="center",
                 bbox=dict(facecolor='none', edgecolor='black',␣
↪boxstyle='round'),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8))
    plt.annotate('Financial \n crisis', xy=(2008, 47.3), xytext=(2008, 53.),␣
↪ha="center", va="center",
                 bbox=dict(facecolor='none', edgecolor='black',␣
↪boxstyle='round'),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8))

    fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)
    fig.tight_layout()
    ax1.set_ylim(0, 17000)
    ax2.set_ylim(bottom=0)
    plt.show()

    # Save plot as .pdf and .png
    save = False
```

```
    if save:
        fig.savefig('./figures/q2/q2_plot_world_abs.pdf', bbox_inches='tight')
        fig.savefig('./figures/q2/q2_plot_world_abs.png', bbox_inches='tight',␣
↪dpi=300)
    return


def plot2_world_rel(df):
    df_country = df.copy()
    df_world = df_country.groupby('year').sum().reset_index()

    year_ref = 2010
    for col in ['prod_electric_fossil', 'prod_electric_nuclear',␣
↪'prod_electric_renewable']:
        ref_val = df_world[df_world['year'] == year_ref][col].values[0]  # ref␣
↪value of year_ref
        df_world[col + '_rel_val'] = df_world[col] / ref_val  # create new␣
↪column with normalized value to ref_val

    df_bar = df[['year', 'Electricity/Heat']]
    df_bar = df_bar.groupby(['year']).sum()  # otherwise, barplot shows␣
↪different color for every country

    fig, ax1 = plt.subplots(figsize=[10, 6])
    ax1.set_xlim(1990, 2017)
    sns.set_style('whitegrid')
    ax1.set_xlabel('year')
    ax1.set_ylabel(r'emissions in Mt CO$_2$')

    plt.bar(x=df_bar.index, height=df_bar['Electricity/Heat'], width=0.75,␣
↪alpha=0.4, align='center',
            label=r'CO$_2$ emissions from electricity and heat generation')

    ax2 = ax1.twinx()  # instantiate a second axes that shares the same x-axis
    ax2.set_ylabel('electricity production relative to %i' % year_ref)
    sns.lineplot(data=df_world, x='year', y='prod_electric_nuclear_rel_val',␣
↪ax=ax2, color='yellow',
                 label='nuclear electricity production', legend=False)
    sns.lineplot(data=df_world, x='year', y='prod_electric_fossil_rel_val',␣
↪ax=ax2, color='brown',
                 ci=None, label='fossil fuel electricity production',␣
↪legend=False)

    fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)
```

```
    fig.suptitle(r'Electricity production compared to CO$_2$ emissions relative␣
↪to ' + str(year_ref) + ' - World',
                 fontsize=16)
    plt.annotate('Fukushima', xy=(2011, 0.96),␣
↪arrowprops=dict(facecolor='black', headwidth=8, width=3, headlength=8),
                 xytext=(2008, 0.8))
    plt.annotate('Financial crisis', xy=(2008, 0.99),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8), xytext=(2006, 1.1))
    fig.tight_layout()
    ax1.set_ylim(bottom=0)
    ax2.set_ylim(bottom=0)
    plt.show()
    return


def plot3_jpn_abs(df):
    df_jpn = df.copy(deep=True)
    df_jpn = df_jpn[df_jpn['country'] == 'JPN']

    x = 'year'
    y0 = 'prod_electric_renewable'
    y1 = 'prod_electric_fossil'
    y2 = 'prod_electric_nuclear'
    y3 = 'Electricity/Heat'

    df_bar = df_jpn[['year', 'Electricity/Heat']]
    df_bar = df_bar.groupby(['year']).sum()  # otherwise, barplot shows␣
↪different color for every country

    fig, ax1 = plt.subplots(figsize=[10, 6])
    ax1.set_xlim(1990, 2017)
    sns.set_style('whitegrid')
    ax1.set_xlabel('year')
    ax1.set_ylabel(r'emissions in Mt CO$_2$')
    plt.bar(x=df_bar.index, height=df_bar[y3], width=0.75, alpha=0.4,␣
↪align='center',
            label=r'CO$_2$ emissions from electricity and heat generation')

    ax2 = ax1.twinx()  # instantiate a second axes that shares the same x-axis
    ax2.set_ylabel('production in EJ')
    sns.lineplot(data=df_jpn, x=x, y=y0, ax=ax2, color='green', ci=None,␣
↪label='renewables electricity production',
                 alpha=0.4, legend=False)
    sns.lineplot(data=df_jpn, x=x, y=y1, ax=ax2, color='brown', ci=None,␣
↪label='fossil fuel electricity production',
```

```
                 legend=False)
    sns.lineplot(data=df_jpn, x=x, y=y2, ax=ax2, color='yellow', ci=None,␣
↪label='nuclear electricity production',
                 legend=False)

    fig.suptitle(r'Electricity production compared to CO$_2$ emissions for␣
↪Japan',
                 fontsize=16)
    plt.annotate('Fukushima', xy=(2011, 0.56),␣
↪arrowprops=dict(facecolor='black', headwidth=8, width=3, headlength=8),
                 xytext=(2012, 0.9))
    plt.annotate('Financial crisis', xy=(2008, 0.87),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8), xytext=(2006, 1.3))
    fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)

    fig.tight_layout()
    ax1.set_ylim(bottom=0)  # has to be here - after the fig was plotted
    ax2.set_ylim(bottom=0)
    plt.show()

    # Save plot as .pdf and .png
    save = False
    if save:
        fig.savefig('./figures/q2/q2_jpn_abs.pdf', bbox_inches='tight')
        fig.savefig('./figures/q2/q2_jpn_abs.png', bbox_inches='tight', dpi=300)
    return


def plot3_jpn_rel(df):
    # Normalize energy production of nuclear energy and fossil fuels --> year␣
↪ref 2013
    df_jpn = df.copy(deep=True)
    df_jpn = df_jpn[df_jpn['country'] == 'JPN']

    # loop for normalized values for a specific year
    year_ref = 2010
    rel_list = ['prod_electric_fossil', 'prod_electric_nuclear',␣
↪'prod_electric_renewable']
    for col in rel_list:
        ref_val = df_jpn[df_jpn['year'] == year_ref][col].values[0]  # ref␣
↪value of year_ref
        df_jpn[col + '_rel_val'] = df_jpn[col] / ref_val  # create new column␣
↪with normalized value to ref_val
```

```
    # fix so that barplot does not show different color for every country
    df_bar = df[df['country'] == 'JPN']
    df_bar = df_bar.groupby(['year']).sum()

    fig, ax1 = plt.subplots(figsize=[10, 6])
    ax1.set_xlim(1990, 2017)
    sns.set_style('whitegrid')
    ax1.set_xlabel('year')
    ax1.set_ylabel(r'emissions in Mt CO$_2$')

    plt.bar(x=df_bar.index, height=df_bar['Electricity/Heat'], width=0.75,␣
↪alpha=0.4, align='center',
            label=r'CO$_2$ emissions from electricity and heat generation')

    ax2 = ax1.twinx()  # instantiate a second axes that shares the same x-axis
    ax2.set_ylabel('electricity production relative to %i' % year_ref)
    sns.lineplot(data=df_jpn, x='year', y='prod_electric_nuclear_rel_val',␣
↪ax=ax2, color='yellow',
                 label='nuclear electricity production', legend=False)
    sns.lineplot(data=df_jpn, x='year', y='prod_electric_fossil_rel_val',␣
↪ax=ax2, color='brown',
                 ci=None, label='fossil fuel electricity production',␣
↪legend=False)

    fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)
    fig.suptitle(r'Electricity production compared to CO$_2$ emissions for␣
↪Japan relative '
                 'to ' + str(year_ref), fontsize=16)
    plt.annotate('Fukushima', xy=(2011, 0.55),␣
↪arrowprops=dict(facecolor='black', headwidth=8, width=3, headlength=8),
                 xytext=(2008, 0.3))
    plt.annotate('Financial crisis', xy=(2008, 0.86),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8),
                 xytext=(2006, 0.6))
    fig.tight_layout()
    ax1.set_ylim(bottom=0)  # has to be here - after the fig was plotted
    ax2.set_ylim(bottom=0, top=1.4)
    plt.show()
    return


def plot4_fra_abs(df):
    df_fra = df.copy(deep=True)
    df_fra = df_fra[df_fra['country'] == 'FRA']
```

```
   x = 'year'
   y0 = 'prod_electric_renewable'
   y1 = 'prod_electric_fossil'
   y2 = 'prod_electric_nuclear'
   y3 = 'Electricity/Heat'

   df_bar = df_fra[['year', 'Electricity/Heat']]
   df_bar = df_bar.groupby(['year']).sum()  # otherwise, barplot shows␣
↪different color for every country

   fig, ax1 = plt.subplots(figsize=[10, 6])
   ax1.set_xlim(1990, 2017)
   sns.set_style('whitegrid')
   ax1.set_xlabel('year')
   ax1.set_ylabel(r'emissions in Mt CO$_2$')
   plt.bar(x=df_bar.index, height=df_bar[y3], width=0.75, alpha=0.4,␣
↪align='center',
           label=r'CO$_2$ emissions from electricity and heat generation')

   ax2 = ax1.twinx()  # instantiate a second axes that shares the same x-axis
   ax2.set_ylabel('production in EJ')
   sns.lineplot(data=df_fra, x=x, y=y0, ax=ax2, color='green', ci=None,␣
↪label='renewables electricity production',
                alpha=0.4, legend=False)
   sns.lineplot(data=df_fra, x=x, y=y1, ax=ax2, color='brown', ci=None,␣
↪label='fossil fuel electricity production',
                legend=False)
   sns.lineplot(data=df_fra, x=x, y=y2, ax=ax2, color='yellow', ci=None,␣
↪label='nuclear electricity production',
                legend=False)

   fig.suptitle(r'Electricity production compared to CO$_2$ emissions for␣
↪France',
                fontsize=16)
   plt.annotate('Fukushima', xy=(2011, 1.52),␣
↪arrowprops=dict(facecolor='black', headwidth=8, width=3, headlength=8),
                xytext=(2012, 1.3))
   plt.annotate('Financial crisis', xy=(2008, 1.51),
                arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8), xytext=(2006, 1.3))
   fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)

   fig.tight_layout()
```

```
    ax1.set_ylim(bottom=0, top=110)  # has to be here - after the fig was␣
↪plotted
    ax2.set_ylim(bottom=0, top=2)
    plt.show()

    # Save plot as .pdf and .png
    save = False
    if save:
        fig.savefig('./figures/q2/q2_fra_abs.pdf', bbox_inches='tight')
        fig.savefig('./figures/q2/q2_fra_abs.png', bbox_inches='tight', dpi=300)
    return


def plot4_fra_rel(df):
    # Normalize energy production of nuclear energy and fossil fuels --> year␣
↪ref 2013
    df_fra = df.copy(deep=True)
    df_fra = df_fra[df_fra['country'] == 'FRA']

    # loop for normalized values for a specific year
    year_ref = 2011
    rel_list = ['prod_electric_fossil', 'prod_electric_nuclear',␣
↪'prod_electric_renewable']
    for col in rel_list:
        ref_val = df_fra[df_fra['year'] == year_ref][col].values[0]  # ref␣
↪value of year_ref
        df_fra[col + '_rel_val'] = df_fra[col] / ref_val  # create new column␣
↪with normalized value to ref_val

    # fix so that barplot does not show different color for every country
    df_bar = df[df['country'] == 'FRA']
    df_bar = df_bar.groupby(['year']).sum()

    fig, ax1 = plt.subplots(figsize=[10, 6])
    ax1.set_xlim(1990, 2017)
    sns.set_style('whitegrid')
    ax1.set_xlabel('year')
    ax1.set_ylabel(r'emissions in Mt CO$_2$')

    plt.bar(x=df_bar.index, height=df_bar['Electricity/Heat'], width=0.75,␣
↪alpha=0.4, align='center',
            label=r'CO$_2$ emissions from electricity and heat generation')

    ax2 = ax1.twinx()  # instantiate a second axes that shares the same x-axis
    ax2.set_ylabel('electricity production relative to %i' % year_ref)
```

```
    sns.lineplot(data=df_fra, x='year', y='prod_electric_nuclear_rel_val',␣
↪ax=ax2, color='yellow',
                 label='nuclear electricity production', legend=False)
    sns.lineplot(data=df_fra, x='year', y='prod_electric_fossil_rel_val',␣
↪ax=ax2, color='brown',
                 ci=None, label='fossil fuel electricity production',␣
↪legend=False)

    fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)
    fig.suptitle(r'Electricity production compared to CO$_2$ emissions for␣
↪France relative '
                 'to ' + str(year_ref), fontsize=16)
    plt.annotate('Fukushima', xy=(2011, 1), arrowprops=dict(facecolor='black',␣
↪headwidth=8, width=3, headlength=8),
                 xytext=(2012, 1.3))
    plt.annotate('Financial crisis', xy=(2008, 0.99),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8),
                 xytext=(2006, 1.3))
    fig.tight_layout()
    ax1.set_ylim(bottom=0, top=110)  # has to be here - after the fig was␣
↪plotted
    ax2.set_ylim(bottom=0)
    plt.show()
    return


def plot5_usa_abs(df):
    # How well does the use of nuclear energy correlate with changes in carbon␣
↪emissions in heat/electricity production
    # in the USA.

    df_usa = df.copy(deep=True)
    df_usa = df_usa[df_usa['country'] == 'USA']

    x = 'year'
    y0 = 'prod_electric_renewable'
    y1 = 'prod_electric_fossil'
    y2 = 'prod_electric_nuclear'
    y3 = 'Electricity/Heat'

    df_bar = df_usa[['year', 'Electricity/Heat']]
    df_bar = df_bar.groupby(['year']).sum()  # otherwise, barplot shows␣
↪different color for every country
```

```
    fig, ax1 = plt.subplots(figsize=[10, 6])
    ax1.set_xlim(1990, 2017)
    sns.set_style('whitegrid')
    ax1.set_xlabel('year')
    ax1.set_ylabel(r'emissions in Mt CO$_2$')
    plt.bar(x=df_bar.index, height=df_bar[y3], width=0.75, alpha=0.4,␣
↪align='center',
            label=r'CO$_2$ emissions from electricity and heat generation')

    ax2 = ax1.twinx()  # instantiate a second axes that shares the same x-axis
    ax2.set_ylabel('production in EJ')
    sns.lineplot(data=df_usa, x=x, y=y0, ax=ax2, color='green', ci=None,␣
↪label='renewables electricity production',
                 alpha=0.4, legend=False)
    sns.lineplot(data=df_usa, x=x, y=y1, ax=ax2, color='brown', ci=None,␣
↪label='fossil fuel electricity production',
                 legend=False)
    sns.lineplot(data=df_usa, x=x, y=y2, ax=ax2, color='yellow', ci=None,␣
↪label='nuclear electricity production',
                 legend=False)

    fig.suptitle(r'Electricity production compared to CO$_2$ emissions for USA',
                 fontsize=16)
    plt.annotate('Fukushima', xy=(2011, 2.84),␣
↪arrowprops=dict(facecolor='black', headwidth=8, width=3, headlength=8),
                 xytext=(2012, 4))
    plt.annotate('Financial crisis', xy=(2008, 2.89),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8), xytext=(2006, 4))
    fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)

    fig.tight_layout()
    ax1.set_ylim(bottom=0, top=3600)  # has to be here - after the fig was␣
↪plotted
    ax2.set_ylim(bottom=0, top=13)
    plt.show()

    # Save plot as .pdf and .png
    save = False
    if save:
        fig.savefig('./figures/q2/q2_usa_abs.pdf', bbox_inches='tight')
        fig.savefig('./figures/q2/q2_usa_abs.png', bbox_inches='tight', dpi=300)
    return
```

```
def plot5_usa_rel(df):
    # Normalize energy production of nuclear energy and fossil fuels --> year␣
↪ref 2013
    df_usa = df.copy(deep=True)
    df_usa = df_usa[df_usa['country'] == 'USA']

    # loop for normalized values for a specific year
    year_ref = 2011
    rel_list = ['prod_electric_fossil', 'prod_electric_nuclear',␣
↪'prod_electric_renewable']
    for col in rel_list:
        ref_val = df_usa[df_usa['year'] == year_ref][col].values[0]  # ref␣
↪value of year_ref
        df_usa[col + '_rel_val'] = df_usa[col] / ref_val  # create new column␣
↪with normalized value to ref_val

    # fix so that barplot does not show different color for every country
    df_bar = df[df['country'] == 'USA']
    df_bar = df_bar.groupby(['year']).sum()

    fig, ax1 = plt.subplots(figsize=[10, 6])
    ax1.set_xlim(1990, 2017)
    sns.set_style('whitegrid')
    ax1.set_xlabel('year')
    ax1.set_ylabel(r'emissions in Mt CO$_2$')

    plt.bar(x=df_bar.index, height=df_bar['Electricity/Heat'], width=0.75,␣
↪alpha=0.4, align='center',
            label=r'CO$_2$ emissions from electricity and heat generation')

    ax2 = ax1.twinx()  # instantiate a second axes that shares the same x-axis
    ax2.set_ylabel('electricity production relative to %i' % year_ref)
    sns.lineplot(data=df_usa, x='year', y='prod_electric_nuclear_rel_val',␣
↪ax=ax2, color='yellow',
                 label='nuclear electricity production', legend=False)
    sns.lineplot(data=df_usa, x='year', y='prod_electric_fossil_rel_val',␣
↪ax=ax2, color='brown',
                 ci=None, label='fossil fuel electricity production',␣
↪legend=False)

    fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)
    fig.suptitle(r'Electricity production compared to CO$_2$ emissions for the␣
↪USA relative '
                 'to ' + str(year_ref), fontsize=16)
```

```
    plt.annotate('Fukushima', xy=(2011, 1), arrowprops=dict(facecolor='black',␣
↪headwidth=8, width=3, headlength=8),
                 xytext=(2012, 0.85))
    plt.annotate('Financial crisis', xy=(2008, 1.02),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8),
                 xytext=(2006, 0.85))
    fig.tight_layout()
    ax1.set_ylim(bottom=0, top=3500)
    ax2.set_ylim(bottom=0, top=1.4)
    plt.show()
    return


def plot6_chn_abs(df):
    df_chn = df.copy(deep=True)
    df_chn = df_chn[df_chn['country'] == 'CHN']

    x = 'year'
    y0 = 'prod_electric_renewable'
    y1 = 'prod_electric_fossil'
    y2 = 'prod_electric_nuclear'
    y3 = 'Electricity/Heat'

    df_bar = df_chn[['year', 'Electricity/Heat']]
    df_bar = df_bar.groupby(['year']).sum()  # otherwise, barplot shows␣
↪different color for every country

    fig, ax1 = plt.subplots(figsize=[10, 6])
    ax1.set_xlim(1990, 2017)
    sns.set_style('whitegrid')
    ax1.set_xlabel('year')
    ax1.set_ylabel(r'emissions in Mt CO$_2$')
    plt.bar(x=df_bar.index, height=df_bar[y3], width=0.75, alpha=0.4,␣
↪align='center',
            label=r'CO$_2$ emissions from electricity and heat generation')

    ax2 = ax1.twinx()  # instantiate a second axes that shares the same x-axis
    ax2.set_ylabel('production in EJ')
    sns.lineplot(data=df_chn, x=x, y=y0, ax=ax2, color='green', ci=None,␣
↪label='renewables electricity production',
                 alpha=0.4, legend=False)
    sns.lineplot(data=df_chn, x=x, y=y1, ax=ax2, color='brown', ci=None,␣
↪label='fossil fuel electricity production',
                 legend=False)
```

```
    sns.lineplot(data=df_chn, x=x, y=y2, ax=ax2, color='yellow', ci=None,␣
↪label='nuclear electricity production',
                 legend=False)

    fig.suptitle(r'Electricity production compared to CO$_2$ emissions for␣
↪China',
                 fontsize=16)
    plt.annotate('Fukushima', xy=(2011, 0.29),␣
↪arrowprops=dict(facecolor='black', headwidth=8, width=3, headlength=8),
                 xytext=(2012, 2))
    plt.annotate('Financial crisis', xy=(2008, 0.24),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8), xytext=(2006, 1.7))
    fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)

    fig.tight_layout()
    ax1.set_ylim(bottom=0)  # has to be here - after the fig was plotted
    ax2.set_ylim(bottom=0)
    plt.show()

    # Save plot as .pdf and .png
    save = False
    if save:
        fig.savefig('./figures/q2/q2_chn_abs.pdf', bbox_inches='tight')
        fig.savefig('./figures/q2/q2_chn_abs.png', bbox_inches='tight', dpi=300)
    return


def plot6_chn_rel(df):
    # Normalize energy production of nuclear energy and fossil fuels --> year␣
↪ref 2013
    df_chn = df.copy(deep=True)
    df_chn = df_chn[df_chn['country'] == 'CHN']

    # loop for normalized values for a specific year
    year_ref = 2011
    rel_list = ['prod_electric_fossil', 'prod_electric_nuclear',␣
↪'prod_electric_renewable']
    for col in rel_list:
        ref_val = df_chn[df_chn['year'] == year_ref][col].values[0]  # ref␣
↪value of year_ref
        df_chn[col + '_rel_val'] = df_chn[col] / ref_val  # create new column␣
↪with normalized value to ref_val

    # fix so that barplot does not show different color for every country
```

```
    df_bar = df[df['country'] == 'CHN']
    df_bar = df_bar.groupby(['year']).sum()

    fig, ax1 = plt.subplots(figsize=[10, 6])
    ax1.set_xlim(1990, 2017)
    sns.set_style('whitegrid')
    ax1.set_xlabel('year')
    ax1.set_ylabel(r'emissions in Mt CO$_2$')

    plt.bar(x=df_bar.index, height=df_bar['Electricity/Heat'], width=0.75,␣
↪alpha=0.4, align='center',
            label=r'CO$_2$ emissions from electricity and heat generation')

    ax2 = ax1.twinx()
    ax2.set_ylabel('electricity production relative to %i' % year_ref)
    sns.lineplot(data=df_chn, x='year', y='prod_electric_nuclear_rel_val',␣
↪ax=ax2, color='yellow',
                 label='nuclear electricity production', legend=False)
    sns.lineplot(data=df_chn, x='year', y='prod_electric_fossil_rel_val',␣
↪ax=ax2, color='brown',
                 ci=None, label='fossil fuel electricity production',␣
↪legend=False)

    fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)
    fig.suptitle(r'Electricity production compared to CO$_2$ emissions for␣
↪China relative '
                 'to ' + str(year_ref), fontsize=16)
    plt.annotate('Fukushima', xy=(2011, 1), arrowprops=dict(facecolor='black',␣
↪headwidth=8, width=3, headlength=8),
                 xytext=(2012, 0.7))
    plt.annotate('Financial crisis', xy=(2008, 0.79),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8),
                 xytext=(2006, 1.2))
    fig.tight_layout()
    ax1.set_ylim(bottom=0)
    ax2.set_ylim(bottom=0)
    plt.show()
    return


def poly_reg_nuclear(df):
    # Polynomial fit of nuclear energy for the world
    y2 = 'prod_electric_nuclear'
```

```
   # Aggregate
   df_polyreg = df.drop('country', axis=1).groupby(['year']).sum().
↪reset_index()

   # Regression
   poly = PolynomialFeatures(degree=2, include_bias=True)
   poly.fit_transform(df_polyreg[['year']])

   poly_model = LinearRegression(fit_intercept=True)
   poly_model.fit(poly.fit_transform(df_polyreg[['year']]), df_polyreg[y2])

   x = np.linspace(df_polyreg['year'].min(), df_polyreg['year'].max(), 1000)
   fx = poly_model.predict(poly.fit_transform(pd.DataFrame(x)))

   # R2 score
   fxp = poly_model.predict(poly.fit_transform(df_polyreg[['year']]))
   print("R^2 score for nuclear fit:", r2_score(y_pred=fxp,␣
↪y_true=df_polyreg[y2]))

   # Instantiate figure
   fig, ax1 = plt.subplots(figsize=[11, 6])
   sns.set_style('whitegrid')
   ax1.set_xlabel('year')
   ax1.set_ylabel('production in EJ')

   sns.scatterplot(data=df_polyreg, x='year', y=y2, color='blue', ci=None,
                   label='nuclear energy electricity production', legend=False)
   sns.lineplot(x=x, y=fx, color='red', label='prediction', legend=False)

   fig.suptitle('Polynomial fit for electricity production - World',␣
↪fontsize=16)
   plt.annotate('Fukushima', xy=(2011, 9), xytext=(2011, 7.7), ha="center",␣
↪va="center",
                bbox=dict(facecolor='none', edgecolor='black',␣
↪boxstyle='round'),
                arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8))
   plt.annotate('Financial \n crisis', xy=(2008, 9.3), xytext=(2008, 8),␣
↪ha="center", va="center",
                bbox=dict(facecolor='none', edgecolor='black',␣
↪boxstyle='round'),
                arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8))
   fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)
   fig.tight_layout()
```

```
    ax1.set_xlim(1979.5, 2018.5)
    ax1.set_ylim(bottom=0)
    plt.show()
    save = False
    if save:
        fig.savefig('./figures/q2/q2_poly_reg_nuclear.pdf', bbox_inches='tight')
        fig.savefig('./figures/q2/q2_poly_reg_nuclear.png',␣
↪bbox_inches='tight', dpi=300)
    return


def poly_reg_emission(df):
    # How well does the use of nuclear energy correlate with changes in carbon␣
↪emissions in heat/electricity production.
    y0 = 'prod_electric_renewable'
    y1 = 'prod_electric_fossil'
    y2 = 'prod_electric_nuclear'
    y3 = 'Electricity/Heat'

    # Aggregate for lineplots
    df_polyreg = df.drop('country', axis=1).groupby(['year']).sum().
↪reset_index()
    df_polyreg = df_polyreg[(df_polyreg['year'] >= 1990) & (df_polyreg['year']␣
↪< 2018)]

    # Regression
    poly = PolynomialFeatures(degree=2, include_bias=True)
    poly.fit_transform(df_polyreg[['year']])

    poly_model = LinearRegression(fit_intercept=True)
    poly_model.fit(poly.fit_transform(df_polyreg[['year']]), df_polyreg[y3])

    x = np.linspace(df_polyreg['year'].min(), df_polyreg['year'].max(), 1000)
    fx = poly_model.predict(poly.fit_transform(pd.DataFrame(x)))

    # R2 score
    fxp = poly_model.predict(poly.fit_transform(df_polyreg[['year']]))
    print("R^2 score for emission fit:", r2_score(y_pred=fxp,␣
↪y_true=df_polyreg[y3]))

    # Instantiate figure
    fig, ax1 = plt.subplots(figsize=[11, 6])
    sns.set_style('whitegrid')
    ax1.set_xlabel('year')
    ax1.set_ylabel(r'emissions in Mt CO$_2$')

    sns.scatterplot(data=df_polyreg, x='year', y=y3, color='blue', ci=None,
```

```
                    label=r'CO$_2$ emissions from electricity and heat␣
↪generation', legend=False)
    sns.lineplot(x=x, y=fx, color='red', label='prediction', legend=False)

    fig.suptitle(r'Polynomial fit of CO$_2$ emissions - World', fontsize=16)
    plt.annotate('Fukushima', xy=(2011, 14650), xytext=(2011, 15600),␣
↪ha="center", va="center",
                 bbox=dict(facecolor='none', edgecolor='black',␣
↪boxstyle='round'),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8))
    plt.annotate('Financial \n crisis', xy=(2008, 13400), xytext=(2008, 15300),␣
↪ha="center", va="center",
                 bbox=dict(facecolor='none', edgecolor='black',␣
↪boxstyle='round'),
                 arrowprops=dict(facecolor='black', headwidth=8, width=3,␣
↪headlength=8))
    fig.legend(loc="upper left", bbox_to_anchor=(0, 1), bbox_transform=ax1.
↪transAxes)
    fig.tight_layout()
    ax1.set_xlim(1989.5, 2017.5)
    ax1.set_ylim(bottom=0)
    plt.show()
    save = False
    if save:
        fig.savefig('./figures/q2/q2_poly_reg_emission.pdf',␣
↪bbox_inches='tight')
        fig.savefig('./figures/q2/q2_poly_reg_emission.png',␣
↪bbox_inches='tight', dpi=300)
    return


# Load df
df, desc = load_df()
# Define parameters for relative growth
start = 1990
stop = 2017
growth = rel_growth(df, start, stop)

# Visualizations:

# Polynomial regression - fit for trends:
poly_reg_nuclear(df)
poly_reg_emission(df)

# Misc.
```

```
corr_matrix(df)
plot_pie(df)  # CO2 emission in the energy sector
print('Relative growth in percent from ' + str(start) + ' to ' + str(stop) + ':
 ↪', growth, sep='\n')
plot1_world_abs(df)  # World - absolute
plot2_world_rel(df)  # World - relative

# Japan
print('Correlation matrix of Japan: ', corr(df, 'JPN'), sep='\n')
plot3_jpn_abs(df)
plot3_jpn_rel(df)

# France
print('Correlation matrix of France: ', corr(df, 'FRA'), sep='\n')
plot4_fra_abs(df)
plot4_fra_rel(df)

# USA
print('Correlation matrix of the USA: ', corr(df, 'USA'), sep='\n')
plot5_usa_abs(df)
plot5_usa_rel(df)

# China
print('Correlation matrix of China: ', corr(df, 'CHN'), sep='\n')
plot6_chn_abs(df)
plot6_chn_rel(df)
```

- corr_matrix: Heat map
  Visualization of correlations between data from energy production/ consumption and CO_2 emission in the energy sector. The focus here lies on correlation between nuclear electricity production and CO_2 emission due to electricity/ heat generation in a time span from 1990 to 2018. The correlation matrix acts as an indicator to possible interesting correlation for further investigation.
- plot_pie: Pie chart
  Gives an overview of the individual CO_2 emission contributors when talking about the energy sector - also shows that electricity/ heat generation contributes to around half (47%) of total CO_2 emissions in this sector with energy production by fossil fuel combustion being the main contributor.
- plot1_world_abs: Absolute values - World
  Absolute values of the 3 main categories of electricity production compared to electricity and heat generation-related CO_2 emissions per year. The plot shows that electricity production by fossil fuel combustion (mainly coal, gas, oil) correlates very well with CO_2 emission in the electricity/ heat sector, since energy production by renewables or through nuclear fission generate only a very small indirect amount of CO_2 emissions compared to fossil fuel energy production. The nuclear reactor incident and world financial crisis of 2008 have been annotated and are further investigated.

- plot2_world_rel: Relative to year 2010 - World
  Annual electricity/ heat generation global CO_2 emissions are plotted in comparison to electricity production by fossil fuels and nuclear fission relative to the year 2010. It is apparent that the nuclear disaster in Fukushima had a palpable impact on future worldwide nuclear energy production, however energy-related CO_2 emissions do not seem to have been impacted in the same way. This can be explained that energy production by fossil fuel combustion still grew over the years following the catastrophic event in Japan, which is the main reason for CO_2 emissions in this sector. It can be seen that the financial crisis in 2008 had a significant impact on both CO_2 emissions as well as energy production.

- plot3_jpn_abs: Absolute values - Japan
  Japan poses as an interesting country, especially since it is one of the largest nuclear energy producers, as well as the historic event in Fukushima in 2011 (reactor meltdown). Annual electricity/ heat generation CO_2 emissions are plotted in comparison to electricity production by fossil fuels, renewables and nuclear energy. After the nuclear reactor accident in 2011 nuclear energy production reduced greatly (2014 even producing zero electricity by nuclear energy). Due to this immense drop and continuous demand for electricity CO_2 emissions peaked at around 2013, since fossil fuel electricity generation increased steeply, with it being the main contributor to CO_2 emissions. CO_2 emissions correlate (positively and) very strongly with fossil fuel electricity production (coeff = 0.994), while it correlates negatively with production by nuclear energy (coeff = -0.807).

- plot3_jpn_rel: Relative values - Japan
  Relative view of electricity generation trends relative to the year 2010 to further emphasize the development of electricity production by nuclear energy and fossil fuels.

The following few countries have been investigated, as they are among the top 3 of nuclear energy production contributors in the world. They can be taken as rough representatives of electricity production through nuclear energy and since they all lie in different continents, they can help in providing a rough overview of the current situation worldwide.

- plot4_fra_abs: Absolute values - France
  France produces most of its electricity through nuclear energy as it holds the largest share of electricity from nuclear power in the world. When taking a look a the correlation coefficient it becomes apparent that CO_2 emissions still correlate more with electricity production by fossil fuels (coeff = 0.559) than by nuclear energy (coeff = -0.081). This again contributes to the fact that CO_2 emissions in this sector are primarily caused by combustion of fossil fuels. Annual electricity/ heat generation CO2 emissions are plotted in comparison to electricity production by fossil fuels, renewables and nuclear energy. Impactful historic events have been annotated.

- plot4_fra_rel: Relative values - France
  Additional plot to visualize the change of electricity production by fossil fuels and nuclear energy relative to 2011.

- plot5_usa_abs: Absolute values - USA
  USA is one of the top CO_2 emission contributors in the electricity/heat production sector, since the majority of energy is provided by means of fossil fuel combustion. It also comes first in nuclear energy production when compared to other countries globally. When looking at the plot, nuclear energy did not undergo a lot of changes the last couple of years, while CO_2 emissions still fluctuated due to fossil energy electricity production. The correlation

coefficient of fossil fuels to CO_2 emissions is corr = 0.539, while the coefficient for nuclear energy is corr = 0.234.

- plot5_usa_rel: Relative values - USA
  Additional plot to further visualize the change of electricity production by fossil fuels and nuclear energy relative to 2011.

- plot6_chn_abs: Absolute values - CHN
  China is currently the country with the highest CO_2 emissions per country when it comes to emissions in the energy sector. Additionally it is one of the countries with the steepest rise in nuclear energy production. Annual electricity/ heat generation CO_2 emissions are plotted in comparison to electricity production by fossil fuels, renewables and nuclear energy. Again, CO_2 emissions correlate positively with electricity production by fossil fuels. Correlation of CO_2 emissions with electricity production by nuclear energy (coeff = 0.897) and fossil fuels (coeff = 0.997) are both very high, whith emphasis on the fact that the majority of emissions result from production via fossil fuels.

- plot6_chn_rel: Relative values - CHN
  Emission and production values relative to the year of 2011. Even though this is marks the year of the Fukushima nuclear reactor accident, electricity production by nuclear energy rises as the line trend of electricity production by fossil fuels takes a small bend. The steep rise of nuclear energy (as well as the increase of production by renewables) might be a good indicator to why CO_2 emissions began to decrease 2011 til around 2016.

- poly_reg_nuclear: polynomial regression with degree 2 - R^2 score: 0.9820348464528007
  Polynomial regression for the trend of nuclear energy electricity production worldwide. Due to the high R2 score the prediction acts as a good fit for the actual data. It shows that usage of nuclear energy tends to decrease over time having its peak at around 2007. The financial crisis as well as the disaster event in Fukushima seem to be two of the main reasons for this result. It should be noted that when looking at the data from 2012 onwards nuclear energy tends to rise linearly, while the prediction correlates negatively with the actual values from the dataset. Therefore, while the prediction indicates a general decline in nuclear energy in electricity production, we can expect more usage and development in nuclear energy usage for electricity generation in the future.

- poly_reg_emission: polynomial regression with degree 2 - R^2 score: 0.9719937093947953
  Polynomial regression for the trend of CO_2 emission from the electricity/heat generation worldwide. Even setting the polynomial degree to 2, the regression yields a nearly perfect linear prediction. It can be seen that the event in 2008 seems to have reduced the increase of CO_2 emissions worldwide (there is an indication for increase in emissions up until 2007 when looking at the plot and the dataset).

Conclusion: While electricity production by nuclear energy has developed over the past couple of years and contribute to a more 'CO_2-emission-free' future, CO_2 emissions by electricity/ heat production are mainly caused by means of fossil fuel combustion (coal, oil, gas). CO_2 emissions caused by the electricity/ heat sector increased by around 75%, electricity production from fossil fuels increased by around 187% and from nuclear energy by around 62% in a time span from 1990 to 2017. The prediction model for nuclear energy indicates a decrease for the future, while it is important to focus on the time span between 2012 and today, where there is an indication in a new (linear) increase of nuclear energy usage in the electricity sector.

1.8 Question 3: Are there characteristics of a country that correlate with increases or decreases in the use of nuclear energy?

The code to answer this question was generated by Johannes.

How many humans live in countries that use nuclear energy? Was there a big change in the last 4 decades? Can we see a tendency of a correlation between the size of a nation (population) and if it uses nuclear energy?

[ ]:
```
def plot_population_countries():
    reactors = load_reactor_numbers()
    population = resize_emission(load_emission_data()).
↪set_index(['year','country'])['population']

    data = reactors.join(population, how='right').query('year >= 1960')
    data.iloc[:,:3] = data.iloc[:,:3].fillna(0)
    data = data.sort_index()

    data['population_in_billion'] = data['population']/1e9  # Convert to␣
↪billion people
    data['BOOL'] = (data['operating_reactors']>0)
    df = pd.DataFrame()
    df['is nuclear'] = data['population_in_billion'][data['BOOL']].
↪sum(level='year')
    df['not nuclear'] = data['population_in_billion'][~data['BOOL']].
↪sum(level='year')
    df = df.sort_index()

    %matplotlib inline
    fig, ax = plt.subplots(1,2, figsize=[15,5])

    df.plot(kind='area', ax=ax[0])
    ax[0].set_ylabel('population in billion')
    ax[0].set_xlim(df.index.min(),df.index.max())
    ax[0].set_title('Cumulative Population', fontsize=14)

    for ISO in data[data['BOOL']].index.get_level_values('country').
↪drop_duplicates():
        start_year = data[data['BOOL']].xs(ISO, level='country').index.min()
        if start_year > data.index.get_level_values('year').min() and␣
↪population.loc[start_year,ISO]>7.4e7:
            ycoord = 0.5 * (df.loc[start_year-1,'is nuclear'] + df.
↪loc[start_year,'is nuclear'])
            ax[0].annotate(text = f"{cc.convert(ISO, src='ISO3', to='short')}",
                   xy=(start_year-0.5,ycoord),
                   xytext=(start_year-0.5+2,ycoord-0.5),
                   ha='left', arrowprops=dict(arrowstyle='->'))
```

```
    df = pd.DataFrame()
    df['is nuclear'] = data['BOOL'].sum(level='year')
    #df['not nuclear'] = (~data['BOOL']).sum(level='year')

    df.plot(kind='area', ax=ax[1])
    ax[1].grid()
    ax[1].set_ylabel('Number of Countries')
    ax[1].set_xlim(df.index.min(),df.index.max())
    ax[1].set_title('Number of Nuclear Countries\n', fontsize=14)
    ax[1].annotate(text="(total number of countries in the merged dataset:␣
↪218)", xy=(0.5,1.02),
                   xycoords='axes fraction', fontsize=12, ha="center")

    ycoord = 0.5 * (df.loc[2008,'is nuclear'] + df.loc[2009,'is nuclear'])
    ax[1].annotate(text = "Shutdown of Lithuania's \nnuclear reactor(s)",
                   xy=(2008.5,ycoord),
                   xytext=(2008.5-15,ycoord-5),
                   ha='left', arrowprops=dict(arrowstyle='->'))
    #plt.savefig('./figures/q3_population.pdf', bbox_inches='tight')
    plt.show()

    # print countries' years of first startup or last shutdown
    df = data[data['BOOL']]
    print("Countries with startup of first or shutdown of last nuclear reactor␣
↪since 1960:")
    for year in range(df.index.get_level_values('year').min()+1,2018):
        prev = df.loc[year-1].index
        next = df.loc[year+1].index
        start_countries = df.loc[year].query("country not in @prev").index.
↪to_list()
        end_countries = df.loc[year].query("country not in @next").index.
↪to_list()
        print(year)
        if len(start_countries)!=0:
            print(f'  start: {start_countries}')
        if len(end_countries)!=0:
            print(f'  end:   {end_countries}')
    return None

plot_population_countries()
```

- In the plots generated below one can see, that mostly big countries have operating nuclear power plants.
- Since the startup of the first reactor in China in 1991, more than half of the world's population live in countries that use nuclear energy.
- The population of both nation groups grew in total in the last 30 years (since 1991).

- However, only a small fraction of all countries produce nuclear power (right plot), because there are many small nations without operating reactors.

Remark: There is no jump in the *total number of nations* after the breakup of the USSR in 1991, because here the population is given in today's nations already before 1991.

1.8.1 Are there correlations between the change of nuclear energy production and the change of other properties of a nation?

To make the properties of countries compareable one has two consider two points:

- The data has to be transfomed to values that are independent of the countries' size, e.g. compare two years of the same country (which can be done by calculating the relative change) of only use percentual values (like those normalized per capita)
- The *average* country that uses nuclear energy is very different from the *average* country that does not, which can be seen in the previos plot, where the nuclear nations are much lager in size and also the distribution is very non-uniformly distributed across the continents (see world maps). Both arguments make it quite hard to compare properties of the *nuclear nations* to the *non-nuclear* ones, and can be easily biased (e.g. the average nuclear country has a much higher GDP per capita than the non-nuclear average, because mainly industialized countries use nuclear energy).

Compare relative change of two years (e.g. 1998 and 2018): In the code below the relative change of the countries' nuclear production are compared to the following properties: - population - GDP and GDP per capita - income per capita - research expenditue per year in %GDP - Democracy indicators: representative government, fundamental rights, checks on gouvernment, impartial administration, civil society participation - nuclear warheads

A detailed expanation of the democarcy indicators can be found in the **Codebook**

[ ]:
```
def correlation_q3(features, start, end, nuclear_countries_only=True):
    data = pd.read_csv('./data/data_merged/data.csv').
↪set_index(['year','country'])
    # exclude countries, that do not use nuclear energy (in both years)
    if nuclear_countries_only:
        nuclear_countries = data.loc[[start,end],'operating_reactors'].
↪sum(axis=0, level='country').replace(0,np.nan).dropna().index
        data = data.query("country in @nuclear_countries")
    df = data[['nuclear_prod_btu']+features]
    data_start = df.xs(start, level='year')
    data_end   = df.xs(end, level='year')
    data_quot  = (data_end.divide(data_start)-1).sort_index() # relative change

    # fill missing values with 0 and drop infinities
    data_quot = data_quot.fillna(0)
    data_quot = data_quot.replace(np.inf,np.nan).dropna()

    # make plot
    %matplotlib inline
```

```
    plt.figure(figsize=[10,10])
    sns.heatmap(data_quot.corr(), vmin=-1, vmax=1, annot=True, cmap='vlag')
    plt.title(f'Correlations of relative change between {start} and {end}' +
              f'\n Number of Countries used: {data_quot.index.size}')
    plt.show()

    return data_quot
```

[ ]:
```
def compare_years_q3(features, start, end, nuclear_countries_only=True):
    data = pd.read_csv('./data/data_merged/data.csv').
↪set_index(['year','country']).sort_index()
    # exclude countries, that do not use nuclear energy (in both years)
    if nuclear_countries_only:
        nuclear_countries = data.loc[[start,end],'operating_reactors'].
↪sum(axis=0, level='country').replace(0,np.nan).dropna().index
        data = data.query("country in @nuclear_countries")
    if type(features)!=list: features = [features]
    df = data[features+['nuclear_prod_btu']]
    df = df.fillna(0)

    data_start = df.xs(start, level='year')
    data_end   = df.xs(end, level='year')
    data_quot  = (data_end.divide(data_start)-1).sort_index() # relative change

    data_quot  = data_quot.fillna(-np.inf)

    # get continent information (for colorcode of scatterplots)
    data_quot['Continent'] = cc.convert(data_quot.index.
↪get_level_values('country').to_list(), src='ISO3', to='continent')

    # scale down countries with large change in nuclear production
    cutoff = 1.5
    max_quot_nuc_prod =␣
↪data_quot[data_quot['nuclear_prod_btu']>cutoff]['nuclear_prod_btu'].to_dict()
    data_quot.loc[max_quot_nuc_prod.keys(),'nuclear_prod_btu'] = cutoff

    # make interactive plot
    %matplotlib notebook
    %matplotlib notebook
    for feature in features:
        fig, ax = plt.subplots(figsize=[10,7])
        sns.scatterplot(data=data_quot,
                x='nuclear_prod_btu', y=feature,
                hue='Continent', legend='full', label='', ax = ax,
                palette={'Asia':'C0','Europe':'C1','Africa':'C2','America':
↪'C3','Oceania':'C4','Antarctica':'C5'}
        )
```

```
        ax.set_title(feature.upper() + f', relative change from {start} to␣
↪{end}')
        ax.set_xlabel('Nuclear Production, Relative Change')
        ax.set_ylabel('')

        # Show ISO code of country when clicking
        mplcursors.cursor(multiple = True).connect(
            "add", lambda sel: sel.annotation.set_text(
                  data_quot.index[sel.target.index]
        ))

        # Add arrows for countries with large change in nuclear production␣
↪(that were scaled down)
        for ISO3 in max_quot_nuc_prod.keys():
            ax.annotate(text=f'{max_quot_nuc_prod[ISO3]:.1f}',
                        xy=(cutoff,data_quot.loc[ISO3,feature]),
                        xytext=(cutoff*1.1,data_quot.loc[ISO3,feature]),
                        ha='left', va='center', arrowprops=dict(arrowstyle='<-',␣
↪color='C0'))

        # Move Axes to centre, passing through (0,0)
        ax.spines['left'].set_position('zero')
        ax.spines['bottom'].set_position('zero')
        ax.spines['right'].set_color('none')
        ax.spines['top'].set_color('none')
        ax.xaxis.set_label_coords(0.5, -0.025, transform=ax.xaxis.
↪get_ticklabels()[0].get_transform())

        ax.legend(loc='best')
    plt.show()

    return data_quot
```

Usage of Plots: By default, only the countries with nuclear reactors are shown (`nuclear_countries_only=True`). All countries are plotted, if this value is set to `False`.

To show the ISO3 code of the countires, click on a point (works as long as it is in 'interactive' mode). To hide ISO3 again, right-click on the text. The 'interactive' mode can be use to zoom in, too

Cell below may be executed more than once, until interactive scatterplots appear! (switching of plotting mode may not work on the first try)

[ ]:
```
features = ['population','GDP','GDP per capita','income per␣
↪capita','research_%GDP',
            ␣
↪'representative_government','fundamental_rights','checks_on_gouvernment',
```

```
␣
↪'impartial_administration','civil_society_participation','nuclear_warheads']

start = 1998
end   = 2018
nuclear_countries_only=True

correlation_q3(features, start, end, nuclear_countries_only);
compare_years_q3(features, start, end, nuclear_countries_only);
```

[ ]:
```
# change plotting mode back to inline (normal)
%matplotlib inline
```

Observations In the plots above, the **relative change** of the countires' properties are compared for the 20 years from **1998 to 2018**, and **only countries that produce nuclear energy** are considered (altough only 33 nations are analized, by using all countries, there is a high risk of distortion of the total correltion, as explained before).

1. Correlation:

If we just investigate the correlation of the nuclear production (relative change from 1998 to 2018) with the other features, we can make the following observations:

- Countries that had a growth of their population, also increased the production of nuclear energy. This seems very plausible, as also the total demand on electricity grows.
- There is also a positive correlation with the economic values (GDP, GDP per capita, income per capita), which can be also likely due to the increasing total energy demand.
- There is a low positive correlation between change of nuclear energy production and reasearch expenditure (which could both be caused by economic growth).
- No clear correlation can be seen with the domocracy indicators.
- An increase of nulcear energy production correlates with in incease of the number of nuclear warheads in the country.

2. Scatter Plots:

By looking at the scatterplots, the plausibility of the described correlations can be reviewed:

- For the three economical indicators, i.e. GDP (per capita) and income per capita, there can indeed be seen a positive correlation with no drastic anomalities.
- For all other indicators, the countries are more widely distributed in a cicle around the origin. The correaltions are caused by the average of this cloud, or by single extreme countries.
- Especially the positive correlation between nuclear energy and nuclear warheads is only caused by two countries (India and Pakistan) that both had a drastic increase in these two numbers.

For further investigation, the scatterplot above can be explored for single countries (e.g. that Russia and Hungary had the lagest decrease of *checks on gouvernment*.

Another possibility is to change the range of year for calcualting the relative change. By chosing the range of the *10 years between 2008 and 2018* (just change the variable `start`), one can see that there was a *negative correlation* of the nuclear production with the indicator *checks on gouvernment*

as well as *civil sociaty participation*. By looking at the scatterplots it indeed looks like there was a general trend in these years, instead of these two correlations just being an effect of *overfitting*.

ANNUAL Correlation for Change of Nuclear-Energy-Use with Country Properties

[ ]:
```
def correlation_q3_single_year(start=1980, end=2018,␣
↪nuclear_countries_only=True):
    data = pd.read_csv('./data/data_merged/data.csv').
↪set_index(['year','country'])
    # exclude countries, that do not use nuclear energy (in both years)
    if nuclear_countries_only:
        nuclear_countries = data['operating_reactors'].sum(axis=0,␣
↪level='country').replace(0,np.nan).dropna().index
        data = data.query("country in @nuclear_countries")

    # select columns
    data_amount = data[['nuclear_prod_btu', 'population','GDP','GDP per␣
↪capita','income per capita','research_%GDP',
                        ␣
↪'representative_government','fundamental_rights','checks_on_gouvernment',
                        ␣
↪'impartial_administration','civil_society_participation','nuclear_warheads']]

    # calculate ANNUAL growth (relative change of two succeeding years)
    data_growth = pd.DataFrame()
    for column in data_amount.columns:
        data_growth[column] = data_amount[column].unstack().pct_change().stack()

    # restrict time range
    data_growth = data_growth.query("@start <= year <= @end")

    # fill missing values with 0 and drop infinities
    data_growth = data_growth.fillna(0)
    data_growth = data_growth.replace(np.inf,np.nan).dropna()

    # make plot
    %matplotlib inline
    plt.figure(figsize=[10,10])
    sns.heatmap(data_growth.corr(), vmin=-1, vmax=1, annot=True, cmap='vlag')
    plt.title(f'Correlations of Annual Growth' +
              f'\n Number of Countries used: {data_growth.index.
↪get_level_values("country").drop_duplicates().size}')
    plt.show()

    return data_growth
```

```
df = correlation_q3_single_year(start=1998, end=2018,␣
↪nuclear_countries_only=True)
```

In contrast to the previous cells, in this plot the **annual change** (relative change of succeeding countries) of nuclear energy production is compared to country properties (again only nuclear nations were chosen). Although the range of year was chosen the same as for the *total relative change* before (year from 1998 to 2008), there can be seen no clear correlation of the change of nuclear energy production with the selected country properties. This implies that **changes of nuclear energy usage only happen on a timescale longer than one year** (except for big accidents like Fukushima).

[ ]: