HW 12-13

Imports

In [1]:
```
import warnings

import numpy as np
import pandas as pd

warnings.filterwarnings("ignore")
```

HW 13

In [2]:
```
# Setup
data: np.ndarray = pd.read_csv("uspopulation.txt", delimiter=" ").values.ast

years = data[:,0]
pops = data[:,1]
year_indices = (years-1900)/10
year_indices
```

Out[2]:
```
array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10., 11.])
```

In [3]:
```
# A

# a design matrix is created as follows:
def design_mat(k: int, vec) -> np.ndarray:
    n = vec.shape[0]
    m = k
    A: np.ndarray = np.zeros((n, m))

    for i in range(n):
        for j in range(m):
            A[i, j] = (vec[i])**j ## type: ignore

    return A

# for example, the design matrix given our data and a monomial basis up to k
des_mat = design_mat(11, year_indices)
pd.DataFrame(des_mat)
```

Out[3]:

	0	1	2	3	4	5	6	7	8
0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
2	1.0	2.0	4.0	8.0	16.0	32.0	64.0	128.0	256.0
3	1.0	3.0	9.0	27.0	81.0	243.0	729.0	2187.0	6561.0
4	1.0	4.0	16.0	64.0	256.0	1024.0	4096.0	16384.0	65536.0
5	1.0	5.0	25.0	125.0	625.0	3125.0	15625.0	78125.0	390625.0
6	1.0	6.0	36.0	216.0	1296.0	7776.0	46656.0	279936.0	1679616.0
7	1.0	7.0	49.0	343.0	2401.0	16807.0	117649.0	823543.0	5764801.0
8	1.0	8.0	64.0	512.0	4096.0	32768.0	262144.0	2097152.0	16777216.0
9	1.0	9.0	81.0	729.0	6561.0	59049.0	531441.0	4782969.0	43046721.0
10	1.0	10.0	100.0	1000.0	10000.0	100000.0	1000000.0	10000000.0	100000000.0
11	1.0	11.0	121.0	1331.0	14641.0	161051.0	1771561.0	19487171.0	214358881.0

In [4]:

```
# B

# Build a dataframe to hold our solutions
df = pd.DataFrame(
     {
          "K": [i+1 for i in range(11)],
          "sum of squared error": [0 for i in range(11)],
          "predicted population in 2020": [0 for i in range(11)],
          "condition number": [0 for i in range(11)]
      }
)

# Let's loop to solve
for i, row in df.iterrows():
     k = row["K"]
     A = design_mat(k, year_indices)
     theta_hat, sse, rank, s = np.linalg.lstsq(A, pops)
     df.loc[i, "sum of squared error"] = sse
     df.loc[i, "predicted population in 2020"] = design_mat(k, np.array([12]
     df.loc[i, "condition number"] = s[0]/s[-1]


# Here are our results
df

## What do you notice?
# It looks like the projected population aligns with my intuition (310-360M,
# After that, it looks like the model overfits, eventually predicting extinc
```

Out[4]:

	K	sum of squared error	predicted population in 2020	condition number
0	1	64833.089851	177.340667	1.000000e+00
1	2	1211.729360	314.443621	1.242415e+01
2	3	106.477333	342.047136	1.579169e+02
3	4	105.948478	341.124798	2.090786e+03
4	5	86.107178	332.065503	3.108043e+04
5	6	65.260770	348.063864	5.218438e+05
6	7	64.757088	352.746326	9.669491e+06
7	8	27.127825	267.999303	2.000555e+08
8	9	20.248611	181.742209	4.745540e+09
9	10	7.605490	508.821952	1.337054e+11
10	11	0.504059	-345.881336	4.749833e+12

Acknowledgment

Work in this repository and with associated assignments and projects may be adapted or copied from similar files used in my prior academic and industry work (e.g., using a LaTeX file or Dockerfile as a starting point). Those files and any other work in this repository may have been developed with the help of LLM's like ChatGPT. For example, to provide context, answer questions, refine writing, understand function call syntax, and assist with repetitive tasks. In these cases, deliverables and associated work reflect my best efforts to optimize my learning and demonstrate my capacity, while using available resources and LLM's to facilitate the process.

ChatGPT Conversation