Penentuan Yield Rates dengan Metode Companion Matrix

Permasalahan utama yang sering kali dihadapi saat menentuka yield rates adalah cara penyelesaian persamaan polinomial yang biasanya memiliki derajat tinggi sehingga sulit untuk diselesaikan secara manual. Pada tugas ini, saya menggunakan metode *companion matrix* untuk mencoba mencari akar-akar dari fungsi polinomial saaat penentuan *yield rates*.

Metode *companion matrix* melibatkan konversi dari bentuk polinomial ke dalam bentuk matriks (*companion matrix*). Nilai-nilai eigen dari matrix ini berkoresponden dengan akar-akar dari persamaan polinomial. Misalkan, kita memiliki sebuah polinomial sebagai berikut:

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

Berdasarkan polinomial diatas, dapat mengkonstruksi sebuah *companion matrix* sebagai berikut:

$$C = \begin{pmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \\ -\frac{a_0}{a_n} & -\frac{a_1}{a_n} & -\frac{a_2}{a_n} & -\frac{a_3}{a_n} & \cdots & -\frac{a_{n-1}}{a_n} \end{pmatrix}$$

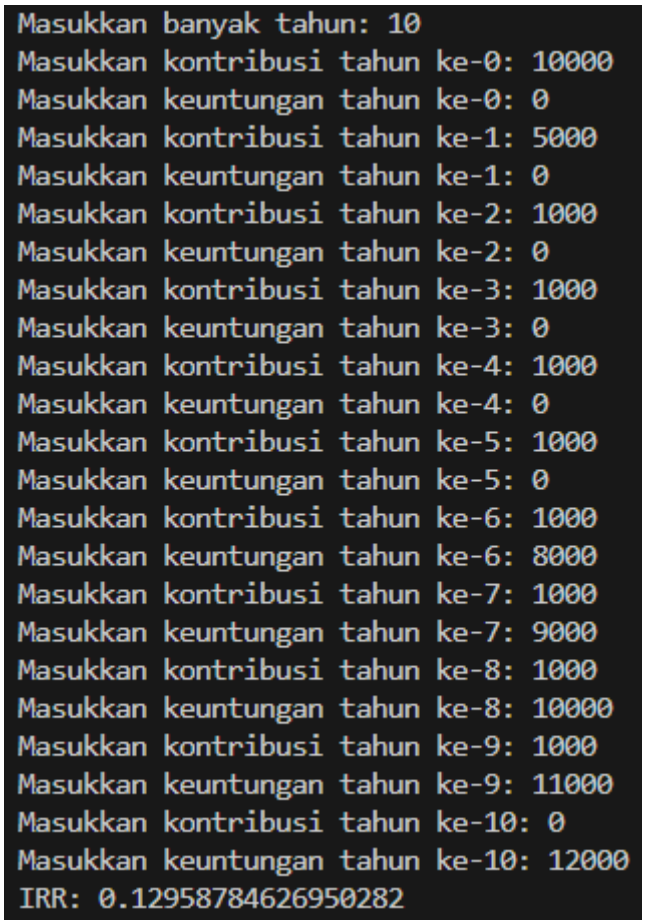
yang memiliki ukuran n × n dengan n adalah derajat dari polinomial $P(x)$. Akar-akar dari polinomial tersebut adalah nilai-nilai eigen dari *matrix companion* C.

Berdasarkan penjelasan singkat sebelumnya, berikut adalah kode untuk mencari akar-akar dari fungsi polinomial dengan bahasa python.

```python
import numpy as np

def companion_matrix(coeffs):
  """
  Membuat companion matrix untuk sebuah polinomial yang memiliki koefisien
  berdasarkan input yang diberikan. List dari koefisien polinomial memiliki urutan
  [a_n, a_{n-1}, ..., a_1, a_0]
  """
  n = len(coeffs) - 1
  a_n = coeffs[0]

  coeffs = [c / a_n for c in coeffs]

  # inisialisasi companion matrix dengan ukuran n x n
  C = np.zeros((n, n))
```

```
  # set sub-diagonal agar bernilai 1
  for i in range(1, n):
      C[i, i-1] = 1

  # set baris terakhir berdasrkan koefisien yang sudah dinormalized
  C[0, :] = -np.array(coeffs[1:])

  return C

def polynomial_roots(coeffs):
  """
  Mencari akar-akar dari suatu polinomial dengan koefisien
berdasarkan masukan dengan
  metode matrix companion dengan urutan [a_n, a_{n-1}, ..., a_1,
a_0]
  """
  C = companion_matrix(coeffs)
  # hitung nilai eigen dari matrix companion
  roots = np.linalg.eigvals(C)
  return roots

def determine_yield_rate():
  n = int(input('Masukkan banyak tahun: '))
  arr = []
  for i in range(0, n + 1):
    kontribusi = float(input(f'Masukkan kontribusi tahun ke-{i}:
'))
    keuntungan = float(input(f'Masukkan keuntungan tahun ke-{i}:
'))
    curr = keuntungan - kontribusi
    arr.append(curr)
  arr.reverse()
  roots = polynomial_roots(arr)
  for root in roots:
    real = root.real
    if (np.abs(root.imag) < 1e-10 and real > 0):
      break
  print(f"IRR: {1/real - 1}")


determine_yield_rate()
```

Berikut adalah pengujian berdasarkan contoh pada slide perkuliahan.



Hal ini sesuai dengan perhitungan pada contoh di slide.

Jawab:

Tahun	Konstribusi	Keuntungan	Net Cash Flow
0	10,000	0	-10,000
1	5,000	0	-5,000
2	1,000	0	-1,000
3	1,000	0	-1,000
4	1,000	0	-1,000
5	1,000	0	-1,000
6	1,000	8,000	7,000
7	1,000	9,000	8,000
8	1,000	10,000	9,000
9	1,000	11,000	10,000
10	0	12,000	12,000

$$1{,}000(-10 - 5v - v^2 - v^3 - v^4 - v^5 + 7v^6 + 8v^7 + 9v^8 + 10v^9 + 12v^{10} = 0$$
$$IRR = 12.96\%$$