Analyzing Credit Ratings with Markov Chains

Raphael Alexander Lesmana (2540128114)

July 2024

Abstract

A Markov chain model is constructed based on the transition matrix of European companies published in the yearly transition study by Standard & Poor. Its steady state distribution vector is calculated in order to understand its long-term behavior by solving a system of linear equations. The chain is also used to model an investor's choice when making an investment.

Introduction

A credit rating is an assessment of a debtor's ability to pay back their debt. Credit ratings are given to individuals, companies, and countries by credit rating agencies (Langohr and Langohr 2012). A good rating indicates that the debtor is trustworthy and has a lower risk of failing to pay their obligations, while a bad rating indicates that the debtor has a higher risk of failing to pay their obligations. A debtor who fails to pay their obligations in time is said to have defaulted. As such, they are used by investors to determine the potential risk associated with investing in a company.

One of the world's largest credit rating agencies is Standard & Poor Global Ratings (S&P). Along with Fitch and Moody's, it is considered one of the Big Three credit-rating agencies (Cash 2021). S&P issues ratings to companies and governments across the world. The company came to be as the result of the merger of H.V. and H.W. Poor Co.—a company that published financial and operational guidebooks about the railway companies of the United States—with the Standard Statistics Bureau—a company that published financial information on non-railroad industries S&P.

Theory

A *Markov process* is a stochastic process describing the transitions of *states* from one to another as determined by probability with respect to time. Transitions are only dependent on the current state, without any regard for the states in the past. Markov processes that are discrete-time are commonly referred to as *Markov chains*. In a Markov chain, the probability of transitioning from one state to another can be represented as a *transition matrix* P, where each element p_{ij} of P is the probably of transitioning from the ith state to the jth state (Blanco-Castañeda and Arunachalam 2023):

$$p_{ij} = P(X_1 = j | X_0 = i), p_{ij} \geq 0$$

Given a Markov chain, the probability distribution among the states at any given point in time can be represented as a row vector $\boldsymbol{\tau}$:

$$\boldsymbol{\tau} = [t_1 \quad t_2 \quad \dots \quad t_n]$$

Where each entry t_i is a real number in the interval $[0,1]$ and the sum of the vector's elements must be equal to 1:

$$\sum_{i=1}^{n} t_i = 1$$

The state distribution vector of the next timestep $\boldsymbol{\tau}_{i+1}$ is obtained by the matrix multiplication of the current timestep's state distribution vector $\boldsymbol{\tau}_i$ with the transition matrix.

$$\boldsymbol{\tau}_i P = \boldsymbol{\tau}_{i+1}$$

Certain Markov chains have a distribution vector $\mathbf{x}$, such that the state distribution after a transition is the same as it was before the transition:

$$\mathbf{x}P = \mathbf{x}$$

Following the convention of writing multiplications of matrices with vectors with the matrix preceding the vector when solving linear equations, the equation above can be rewritten as:

$$P^{\mathrm{T}}\mathbf{x}^{\mathrm{T}} = \mathbf{x}^{\mathrm{T}} \tag{1}$$

The vector $\mathbf{x}$ is known as the *steady state distribution vector* or the *equilibrium vector* of the Markov chain. It exists if and only if the Markov chain is *ergodic*, that is, it is possible to visit every other state in the Markov chain, no matter what the starting state is. It follows that a Markov chain with at least one *absorbing state*, that is, a state that does not lead to any other state but itself, cannot have a steady state distribution vector, as it is not possible to visit any of the other states by starting on an absorbing state.

A *regular Markov chain* is a subset of ergodic Markov chains whose transition matrix are *primitive*, that is, the transition matrix P satisfies:

$$\exists k \in \mathbb{N}(P^k > 0)$$

Where $P^k > 0$ signifies all entries of P are greater than 0 (Pinsky, Karlin and Taylor 2011). Let $\boldsymbol{\tau}$ be any arbitrary distribution vector of a regular Markov chain, then the steady state distribution of the Markov chain may be interpreted as the following limit:

$$\mathbf{x} = \boldsymbol{\tau} \lim_{k\to\infty} P^k \tag{2}$$

It might be tempting to think that to prove that a matrix is primitive, then one must check a large set of integer exponents. However, Wielandt's theorem shows a square matrix P of order n is primitive, if and only if $P^{(n-1)^2+1} > 0$ (Horn and Johnson 2012). The theorem shows that it is only necessary to check for values of k up to $(n-1)^2 + 1$ when checking $P^k > 0$.

There are several ways to calculate the steady state distribution vector of an ergodic Markov chain. The first one is to compute the limit in equation (2). This will only work if the Markov chain is regular. Another one is to consider the equation $P^{\mathrm{T}}\mathbf{x}^{\mathrm{T}} = \mathbf{x}^{\mathrm{T}}$, which should suggest the definition of an eigenvector:

$$A\mathbf{v} = \lambda\mathbf{v}$$

In this case, the steady state distribution vector is the normalized eigenvector of P^{T} with the associated eigenvalue $\lambda = 1$ (Anton 2010).

Yet another way to calculate the steady state distribution vector is to solve equation (1) augmented with the following equation:

$$\sum_{i=1}^{n} x_i = 1$$

Which, when written in matrix multiplication form becomes:

$$\mathbf{1}^{\mathrm{T}}\mathbf{x}^{\mathrm{T}} = 1 \tag{3}$$

Where **1** represents the column vector whose entries are all 1.

Equations (1) and (3) may be combined to give a system of linear equations written in matrix form:

$$\begin{bmatrix} P^{\mathrm{T}} \\ \mathbf{1}^{\mathrm{T}} \end{bmatrix} \mathbf{x}^{\mathrm{T}} = \begin{bmatrix} \mathbf{x}^{\mathrm{T}} \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} p_{11} & p_{21} & \cdots & p_{n1} \\ p_{12} & p_{22} & \cdots & p_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ p_{1n} & p_{2n} & \cdots & p_{nn} \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} p_{11}-1 & p_{21} & \cdots & p_{n1} \\ p_{12} & p_{22}-1 & \cdots & p_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ p_{1n} & p_{2n} & \cdots & p_{nn}-1 \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

Which may be abbreviated as:

$$A\mathbf{x}^{\mathrm{T}} = \mathbf{b} \tag{4}$$

Where:

$$A = \begin{bmatrix} p_{11}-1 & p_{21} & \cdots & p_{n1} \\ p_{12} & p_{22}-1 & \cdots & p_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ p_{1n} & p_{2n} & \cdots & p_{nn}-1 \\ 1 & 1 & \cdots & 1 \end{bmatrix}$$

$$\mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

The number of equations needed to solve a system with n variables is n equations. This system has $n + 1$ equations, which indicates that one of the rows of A is linearly dependent. Techniques such as Gaussian elimination may be used to determine the linearly dependent rows, which can then be removed.

This system of equations may be solved for any ergodic Markov chain. It does not require P to be a primitive matrix. However, it is convenient to have P as a primitive matrix in order to numerically compare the results from the equation and from the limit.

Methods and Data

Every year, S&P publishes a study about the transition between corporate ratings which includes transition matrixes for companies from different regions and different time spans. These studies can be freely accessed from S&P's website.

S&P uses the following ratings in their transition studies:

- **AAA**: Extremely strong capacity to meet financial commitments.
- **AA**: Very strong capacity to meet financial commitments.
- **A**: Strong capacity to meet financial commitments, but somewhat susceptible to economic conditions and changes in circumstances.
- **BBB**: Adequate capacity to meet financial commitments, but more subject to adverse economic conditions.
- **BB**: Adequate capacity to meet financial commitments, but more subject to adverse economic conditions.
- **B**: More vulnerable to adverse business, financial and economic conditions but currently has the capacity to meet financial commitments.
- **CCC**: Currently vulnerable and dependent on favorable business, financial and economic conditions to meet financial commitments.
- **CC**: Highly vulnerable; default has not yet occurred, but is expected to be a virtual certainty.
- **C**: Currently highly vulnerable to non-payment, and ultimate recovery is expected to be lower than that of higher rated obligations.
- **D**: Payment default on a financial commitment or breach of an imputed promise; also used when a bankruptcy petition has been filed.
- **NR**: Not rated.

Quoted from https://www.spglobal.com/ratings/en/about/intro-to-credit-ratings.

The study merges the ratings from **CCC** to **C** into one state labelled CCC/C.

The following transition matrix describes the average one-year transition probabilities of European companies from 1983 to 2023.

	AAA	AA	A	BBB	BB	B	CCC/C	D	NR	$\sum$
AAA	0.8720	0.0903	0.0046	0.0000	0.0000	0.0000	0.0011	0.0000	0.0320	1.0000
AA	0.0025	0.8661	0.0935	0.0050	0.0000	0.0000	0.0000	0.0000	0.0329	1.0000
A	0.0001	0.0167	0.8859	0.0492	0.0013	0.0003	0.0000	0.0003	0.0462	1.0000
BBB	0.0000	0.0006	0.0380	0.8634	0.0306	0.0026	0.0009	0.0005	0.0635	1.0001
BB	0.0000	0.0000	0.0008	0.0521	0.7567	0.0638	0.0035	0.0035	0.1195	0.9999
B	0.0000	0.0000	0.0002	0.0018	0.0460	0.7415	0.0501	0.0179	0.1425	1.0000
CCC/C	0.0000	0.0000	0.0000	0.0016	0.0000	0.1351	0.4472	0.2500	0.1661	1.0000
D	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	1.0000
NR	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	1.0000

Adjustments need to be made as rows BBB and BB do not sum up to exactly 1. An entry from each row is selected and modified slightly in order to correct the row sums.

	AAA	AA	A	BBB	BB	B	CCC/C	D	NR	$\sum$
AAA	0.8720	0.0903	0.0046	0.0000	0.0000	0.0000	0.0011	0.0000	0.0320	1.0000
AA	0.0025	0.8661	0.0935	0.0050	0.0000	0.0000	0.0000	0.0000	0.0329	1.0000
A	0.0001	0.0167	0.8859	0.0492	0.0013	0.0003	0.0000	0.0003	0.0462	1.0000
BBB	0.0000	0.0006	0.0380	0.8634	0.0306	0.0026	0.0009	0.0005	0.0634	1.0000
BB	0.0000	0.0000	0.0008	0.0522	0.7567	0.0638	0.0035	0.0035	0.1195	1.0000
B	0.0000	0.0000	0.0002	0.0018	0.0460	0.7415	0.0501	0.0179	0.1425	1.0000
CCC/C	0.0000	0.0000	0.0000	0.0016	0.0000	0.1351	0.4472	0.2500	0.1661	1.0000
D	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	1.0000
NR	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	1.0000

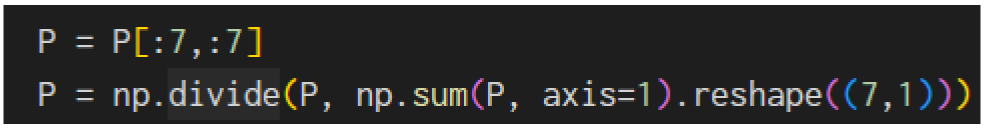
This matrix is manually inputted into a Jupyter notebook running Python 3 and NumPy, which will be used to perform most of the calculations below.

D and *NR* are absorbing states. For the purposes of computing the steady state distribution, these states must be removed, and the entire transition matrix be recomputed by normalizing the remaining elements of the rows using the following formula:

$$p'_{ij} = \frac{p_{ij}}{\sum_{k\in S\setminus B} p_{ik}} \quad i,j \in S \setminus B$$

Where $S \setminus B$ is the set of all indices of P excluding the absorbing states' indices.

The Python code used to calculate this new matrix is as follows:

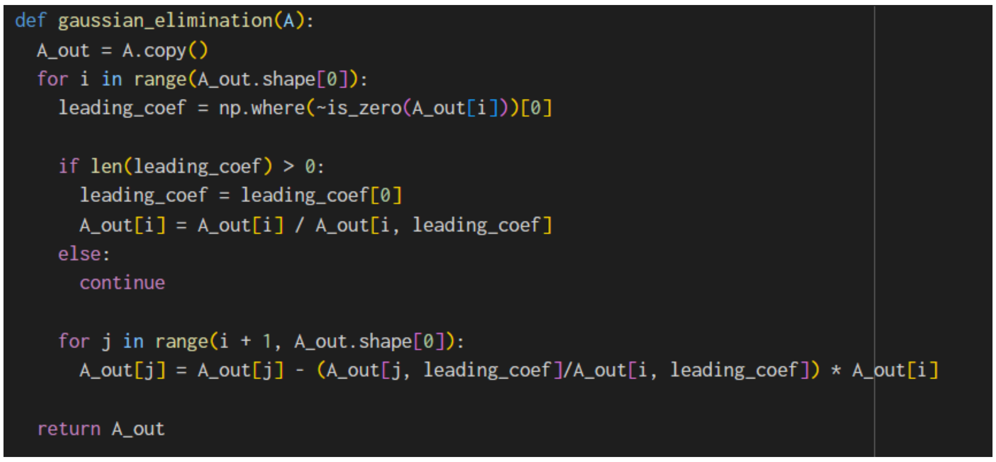


```
P = P[:7,:7]
P = np.divide(P, np.sum(P, axis=1).reshape((7,1)))
```

This new matrix shall be referred as P', where $P' = \{p'_{ij}\}_{i,j\in S\setminus B}$. To verify that P' is a primitive matrix, it is squared repeatedly until its power is greater than the upper bound

$(n-1)^2+1$. Compared to testing every integer from 1 to $(n-1)^2+1$, which takes $O(n^2)$ multiplications to perform, this method only uses $O(\log n)$ multiplications.

After verifying that P' is primitive, the equation matrix A is constructed. The linearly dependent rows are found using an implementation of the Gaussian elimination algorithm in Python. This implementation does not make any optimizations and is a straightforward implementation of the algorithm,

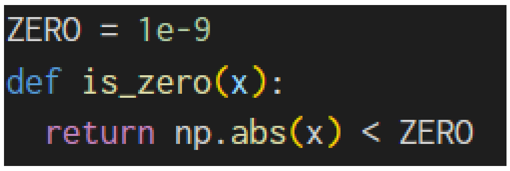


```
def gaussian_elimination(A):
  A_out = A.copy()
  for i in range(A_out.shape[0]):
    leading_coef = np.where(~is_zero(A_out[i]))[0]

    if len(leading_coef) > 0:
      leading_coef = leading_coef[0]
      A_out[i] = A_out[i] / A_out[i, leading_coef]
    else:
      continue

    for j in range(i + 1, A_out.shape[0]):
      A_out[j] = A_out[j] - (A_out[j, leading_coef]/A_out[i, leading_coef]) * A_out[i]

  return A_out
```

A crucial part of the algorithm's operation is determining if a row has a leading coefficient, that is, the first entry in a row that is nonzero. This is done by checking each element whether or not it is zero or not. It is important to note that due to floating-point errors, simple equality comparisons are unreliable. Instead, equalities of the form $a=b$ are determined by looking at the following inequality:

$$|a-b|<\epsilon, \epsilon \in \mathbb{R}^+$$

The value of ϵ is chosen arbitrarily small. In this case, it is chosen to be $\epsilon = 10^{-9}$. This is implemented in Python as follows:



```
ZERO = 1e-9
def is_zero(x):
  return np.abs(x) < ZERO
```

Rather than being a function that directly compares two numbers, the function only takes one input, x, and compares it against ϵ. To compare two numbers a and b, x must be defined as $x = a - b$.

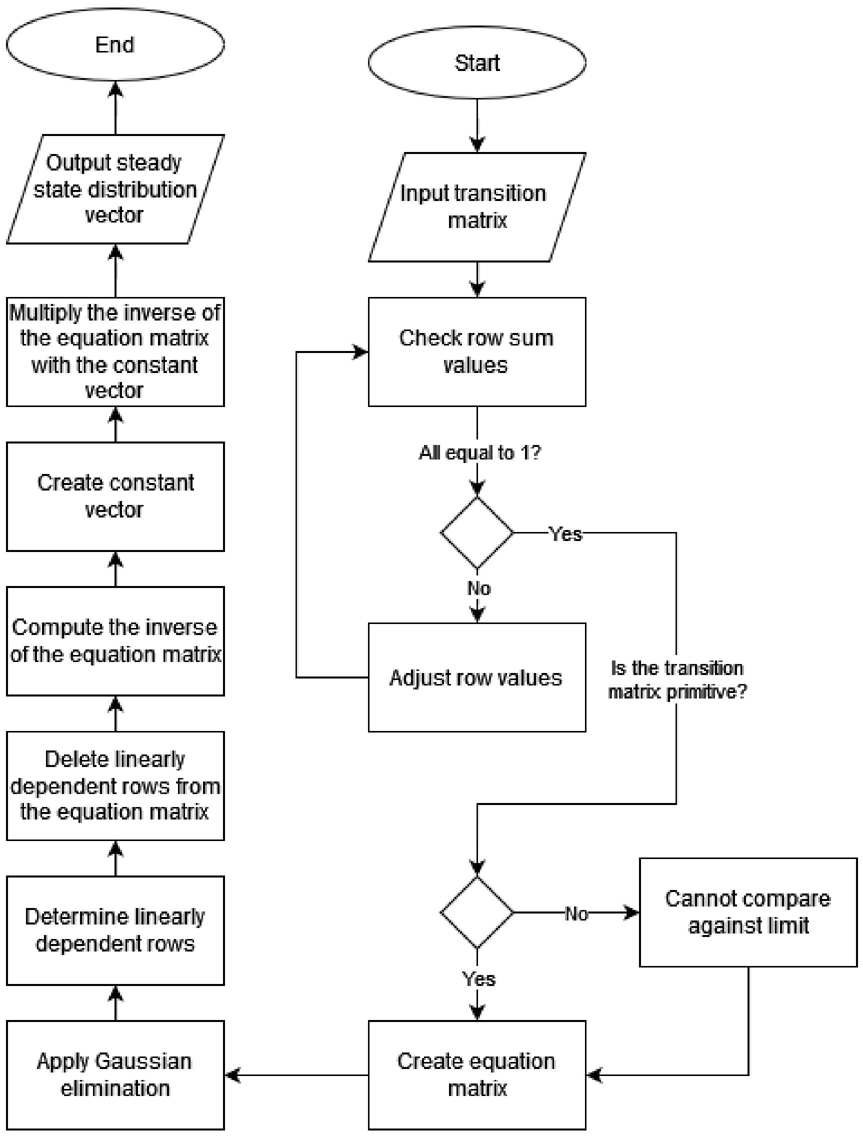
After removing the linearly dependent row, the inverse of the equation matrix is computed, and the steady state distribution vector is computed by multiplying the inverse of the equation matrix with the constant vector $\mathbf{b}$:

$$\mathbf{x}^{\mathrm{T}} = A^{-1}\mathbf{b}$$

This result will be compared graphically with the limit in equation (2) by using the matplotlib library. The algorithm to calculate this limit is simple: let $\boldsymbol{\tau}$ be any arbitrary state distribution vector of the Markov chain and Y be a sufficiently large integer. Calculate $\boldsymbol{\tau} {P'}^k$ for values of k from 1 to Y and plot the results against the results obtained by solving the system of equations. The Python implementation is as follows:

```
limit_vectors = np.zeros((Y, m))
for i in range(1, Y + 1):
  limit_vectors[i - 1] = tau @ np.linalg.matrix_power(P, i)
fig, ax = plt.subplots(figsize=(10,5))
ax.plot(np.linspace(1, Y, num=Y), limit_vectors)
ax.hlines(steady_state, 1, Y, color="black")
ax.legend(RATINGS)
```

The entire process of computing the steady state distribution vector can be summarized in the following flowchart:


End
Output steady state distribution vector
Multiply the inverse of the equation matrix with the constant vector
Create constant vector
Compute the inverse of the equation matrix
Delete linearly dependent rows from the equation matrix
Determine linearly dependent rows
Apply Gaussian elimination
Start
Input transition matrix
Check row sum values
All equal to 1?
Yes
No
Adjust row values
Is the transition matrix primitive?
No
Yes
Cannot compare against limit
Create equation matrix

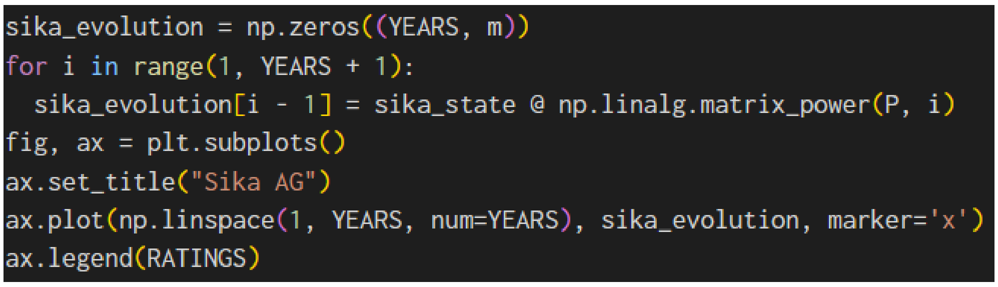
For the scenario of this study, consider an investor who is looking to invest in one of the following companies:

- Sika AG, a Swiss-based chemical company that specializes in producing construction chemicals. S&P rated this company A.

- Akzo Nobel N.V. (AkzoNobel), a Dutch-based chemical company that specializes in manufacturing paint and coating. S&P rated this company BBB.

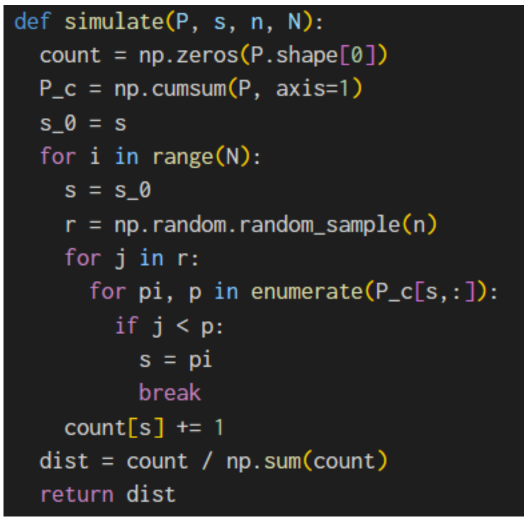
The investor has an interesting requirement when choosing which company he will invest in: if the company's chance of having its rating be upgraded within 5 years is greater than 5%, and if the company's chance of having its rating be downgraded by two levels within 5 years is less than 10%, then the investor will invest in that company. The investor would also like to see the plot of the distribution states' 20-year evolution.

This scenario requires the computation of $\boldsymbol{\tau}_0 P^k$ for $k = 1 \ldots 20$ where $\boldsymbol{\tau}_0$ is the starting state distribution of any of the two companies. This is implemented in Python as follows:



```python
sika_evolution = np.zeros((YEARS, m))
for i in range(1, YEARS + 1):
  sika_evolution[i - 1] = sika_state @ np.linalg.matrix_power(P, i)
fig, ax = plt.subplots()
ax.set_title("Sika AG")
ax.plot(np.linspace(1, YEARS, num=YEARS), sika_evolution, marker='x')
ax.legend(RATINGS)
```

The state distribution vector of the fifth year can be extracted by selecting the fourth index in the array. This result is compared against a Markov chain simulation performed over a large number of iterations. The code for this simulation is implemented as follows:



```python
def simulate(P, s, n, N):
  count = np.zeros(P.shape[0])
  P_c = np.cumsum(P, axis=1)
  s_0 = s
  for i in range(N):
    s = s_0
    r = np.random.random_sample(n)
    for j in r:
      for pi, p in enumerate(P_c[s,:]):
        if j < p:
          s = pi
          break
    count[s] += 1
  dist = count / np.sum(count)
  return dist
```

Results and Analysis

The first step is to recalculate the matrix so that the absorbing states are not included in the computation. Each row is summed without the absorbing states:

State	$\sum_{j \in S \setminus B} p_{ij}$
AAA	0.9680
AA	0.9671
A	0.9535
BBB	0.9361
BB	0.8770
B	0.8396
CCC/C	0.5839

From a glance, the state probability that will be most affected by this adjustment is CCC/C, as the probabilities nearly double ($1/0.5839 \approx 2$).

The elements of each row, excluding the absorbing states, are then divided by the sum of the corresponding row to obtain the modified matrix P':

	AAA	AA	A	BBB	BB	B	CCC/C	$\sum$
AAA	0.9008	0.0933	0.0048	0.0000	0.0000	0.0000	0.0011	1.0000
AA	0.0026	0.8955	0.0967	0.0052	0.0000	0.0000	0.0000	1.0000
A	0.0001	0.0175	0.9291	0.0516	0.0014	0.0003	0.0000	1.0000
BBB	0.0000	0.0006	0.0406	0.9223	0.0327	0.0028	0.0010	1.0000
BB	0.0000	0.0000	0.0009	0.0595	0.8629	0.0727	0.0040	1.0000
B	0.0000	0.0000	0.0002	0.0021	0.0548	0.8832	0.0597	1.0000
CCC/C	0.0000	0.0000	0.0000	0.0027	0.0000	0.2314	0.7659	1.0000

The matrix is checked to see if it is primitive. The squares of the matrix are computed repeatedly:

${P'}^2$ has one non-positive entry in row CCC/C, column AAA:

	AAA	AA	A	BBB	BB	B	CCC/C
AAA	0.81172993	0.16765974	0.01771482	0.00073061	0.00000648	0.00026442	0.00189399
AA	0.00465389	0.80397275	0.17663246	0.01437839	0.00030082	0.00004478	0.00000790
A	0.00023719	0.03200085	0.86702270	0.09570555	0.00414707	0.00081270	0.00007394
BBB	0.00000591	0.00187618	0.07524974	0.85475840	0.05856245	0.00762800	0.00191931
BB	0.00000009	0.00005412	0.00406813	0.10646895	0.75040449	0.12810603	0.01089818
B	0.00000002	0.00000555	0.00056870	0.00730761	0.09572950	0.79776740	0.09762122
CCC/C	0.00000000	0.00000176	0.00016635	0.00512210	0.01276616	0.37155537	0.60038826

${P'}^4$ contains all positive entries:

	AAA	AA	A	BBB	BB	B	CCC/C
AAA	0.65968996	0.27145656	0.05940843	0.00533747	0.00022630	0.00117655	0.00270472
AA	0.00756129	0.65283185	0.29631837	0.04085320	0.00204707	0.00036780	0.00006829
A	0.00054768	0.05369283	0.76460411	0.16569244	0.01240071	0.00263398	0.00041824
BBB	0.00003644	0.00552433	0.1301380	0.74414139	0.09506963	0.02090124	0.00418897
BB	0.00000200	0.00041482	0.01467589	0.17228210	0.58176152	0.20300386	0.02755981
B	0.00000025	0.00004615	0.00190349	0.02282788	0.14989505	0.68638200	0.13894516

CCC/C	0.00000008	0.00002021	0.00089878	0.01161681	0.05407119	0.53514843	0.39824449

This confirms that the matrix P' is primitive. This allows the solution of the linear equation to be compared against the solution obtained from the limit.

The equation matrix is constructed. To the transition matrix is appended $\mathbf{1}^{\mathrm{T}}$, which enforces the sum of the state distribution vector to be equal to 1:

	AAA	AA	A	BBB	BB	B	CCC/C
AAA	0.9008	0.0933	0.0048	0.0000	0.0000	0.0000	0.0011
AA	0.0026	0.8955	0.0967	0.0052	0.0000	0.0000	0.0000
A	0.0001	0.0175	0.9291	0.0516	0.0014	0.0003	0.0000
BBB	0.0000	0.0006	0.0406	0.9223	0.0327	0.0028	0.0010
BB	0.0000	0.0000	0.0009	0.0595	0.8629	0.0727	0.0040
B	0.0000	0.0000	0.0002	0.0021	0.0548	0.8832	0.0597
CCC/C	0.0000	0.0000	0.0000	0.0027	0.0000	0.2314	0.7659
$\mathbf{1}^{\mathrm{T}}$	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

Gaussian elimination is then applied to find linearly dependent rows. The result of this step is a matrix in row-echelon form:

	AAA	AA	A	BBB	BB	B	CCC/C
AAA	1.0000	-2.6066	-0.0016	0.0000	0.0000	0.0000	0.0000
AA	0.0000	1.0000	-0.1727	-0.0063	0.0000	0.0000	0.0000
A	0.0000	0.0000	1.0000	-0.7605	-0.0168	0.0003	0.0000
BBB	0.0000	0.0000	0.0000	1.0000	-1.6019	-0.0630	-0.0727
BB	0.0000	0.0000	0.0000	0.0000	1.0000	-0.6847	-0.0295
B	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	-3.6817
CCC/C	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
$\mathbf{1}^{\mathrm{T}}$	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000

Row CCC/C is linearly dependent, it is thus removed from the equation matrix to produce:

	AAA	AA	A	BBB	BB	B	CCC/C
AAA	0.9008	0.0933	0.0048	0.0000	0.0000	0.0000	0.0011
AA	0.0026	0.8955	0.0967	0.0052	0.0000	0.0000	0.0000
A	0.0001	0.0175	0.9291	0.0516	0.0014	0.0003	0.0000
BBB	0.0000	0.0006	0.0406	0.9223	0.0327	0.0028	0.0010
BB	0.0000	0.0000	0.0009	0.0595	0.8629	0.0727	0.0040
B	0.0000	0.0000	0.0002	0.0021	0.0548	0.8832	0.0597
$\mathbf{1}^{\mathrm{T}}$	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

The constant matrix **b** is constructed:

AAA	0.0000
AA	0.0000
A	0.0000
BBB	0.0000
BB	0.0000
B	0.0000
CCC/C	0.0000
$\mathbf{1}^{\mathrm{T}}$	1.0000

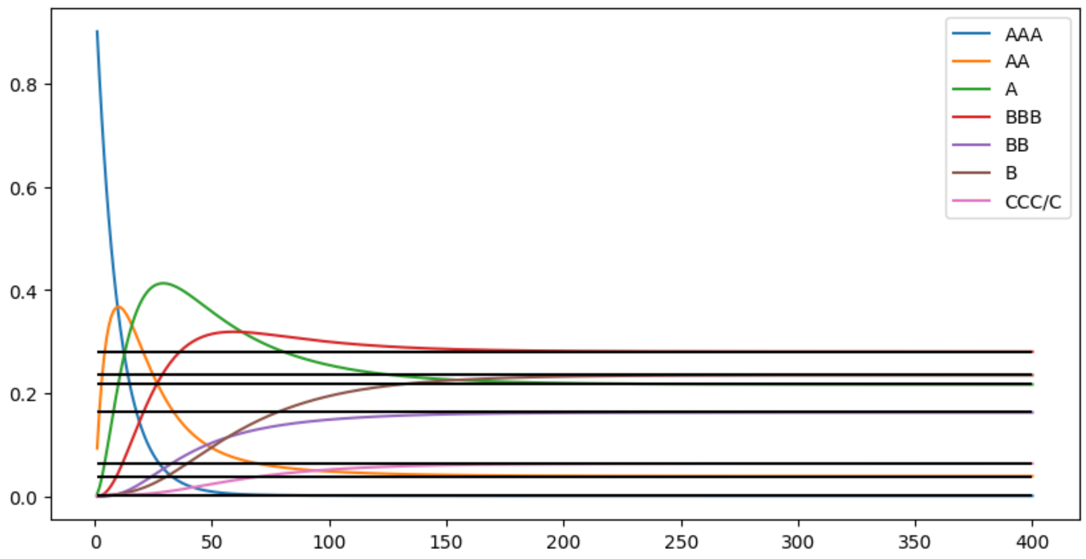
The equation matrix is inverted:

	AAA	AA	A	BBB	BB	B	CCC/C
AAA	-10.4468	-0.3920	-0.1529	-0.0792	-0.0286	-0.0045	0.0013
AA	-13.0528	-14.0548	-4.7629	-2.4830	-0.8974	-0.1402	0.0392
A	-21.9549	-24.2812	-27.1567	-13.7162	-4.9612	-7.7568	0.2170
BBB	-6.4364	-9.2506	-0.1174	-0.1824	-6.4779	-0.9960	0.2804
BB	10.1450	8.6720	7.2023	3.9045	-4.7582	-0.7504	0.1629
B	29.7866	27.8356	25.7162	21.0016	10.3275	-1.2652	0.2352
$\mathbf{1}^T$	11.9592	11.4740	10..8997	9.6155	6.7958	3.9320	0.0639

The inverted equation matrix is multiplied to obtain the steady state distribution vector:

State	Probability
AAA	0.0013
AA	0.0392
A	0.2170
BBB	0.2804
BB	0.1629
B	0.2352
CCC/C	0.0639
$\sum$	0.9999

The solution of this system of equations is compared against the limit in equation (2). An arbitrary starting vector is chosen and then multiplied with P' for 400 iterations. The results are then plotted using matplotlib to give a visual comparison with the solution obtained before:



The graph visually confirms that the steady state distribution vector obtained by solving the linear equation is correct. As the time increases, the probability of each state converges to their steady state values.

From the steady state distribution values, it can be seen that it is very unlikely, over time, for a company to end up with the best rating of AAA. The ratings tend more toward the range of BBB to CCC/C. It is also interesting to note the peak of A's probability sometime between 25 and 50 years after the initial state.

To solve the scenario described in the previous section, the initial state of each company is described. The initial state vector of Sika AG can be written as:

State	Probability
AAA	0.0000
AA	0.0000
A	1.0000
BBB	0.0000
BB	0.0000
B	0.0000
CCC/C	0.0000

The vector for AkzoNobel's initial state can be written as:

State	Probability
AAA	0.0000
AA	0.0000
A	0.0000
BBB	1.0000
BB	0.0000
B	0.0000
CCC/C	0.0000

After manually inputting these vectors into the Jupyter notebook, the distribution vectors for each year, $\boldsymbol{\tau}P^k, k = 1 \ldots 20$, is calculated. The probability of a company's rating increasing is given as $P(X > i)$, while the probability of a company's rating decreasing by more than two levels is given as $P(X < i - 2)$.

The following is the state distribution vector for Sika AG after five years:

State	Probability
AAA	0.0007
AA	0.0616
A	0.7223
BBB	0.1933
BB	0.0173
B	0.0040
CCC/C	0.0007

Sika AG is currently sitting with a rating of A. The probability of having its rating go up is 6.22%. The probability of having its rating go down below two ratings is 2.20%.

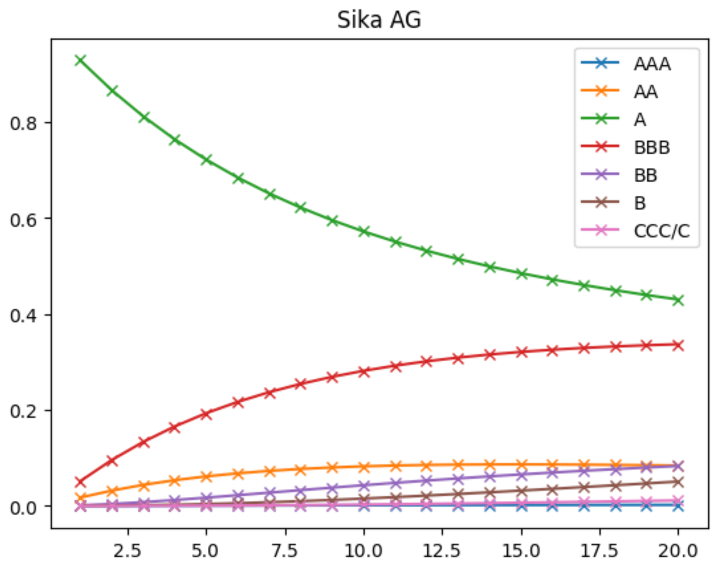
The following is the state distribution vector for AkzoNobel after five years:

State	Probability
AAA	0.0001
AA	0.0077

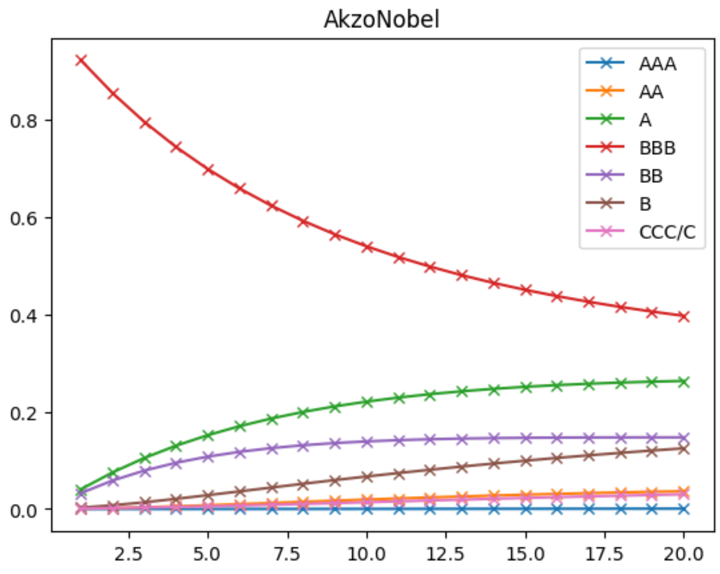
A	0.1517
BBB	0.6988
BB	0.1077
B	0.0284
CCC/C	0.0056

AkzoNobel is currently sitting with a rating of BBB. The probability of having its rating go up is 15.95%. The probability of having its rating go down below two ratings is 3.40%.

The following is a graph of the evolution of Sika AG's state distribution vector over 20 years.



The following is a graph of the evolution of AkzoNobel's state distribution vector over 20 years.



The graph shows that the probability of Sika AG being downgraded to BBB is higher than the probability of AkzoNobel being downgraded to BB. The cumulative probability also shows that the probability of Sika AG being downgraded is higher than the probability of AkzoNobel being downgraded, even though AkzoNobel has a lower rating than Sika AG.

The results are compared with the results obtained by simulating the Markov chain 9,000,000 times:

Sika AG:

State	Simulation Probability	Δ
AAA	0.0007	0.0000
AA	0.0616	0.0000
A	0.7226	0.0003
BBB	0.1931	0.0002
BB	0.0172	0.0001
B	0.0040	0.0000
CCC/C	0.0007	0.0000

AkzoNobel:

State	Simulation Probability	Δ
AAA	0.0000	0.0001
AA	0.0077	0.0000
A	0.1519	0.0002
BBB	0.6986	0.0002
BB	0.1077	0.0000
B	0.0285	0.0001
CCC/C	0.0055	0.0001

The average deviation of Sika AG's results from the simulation are less than 0.0001, while the average deviation of AkzoNobel's results is exactly 0.0001.

It is important to note that these calculations cannot reflect actual real-life scenarios, as the Markov chain has been altered in such a way that the absorbing states are not included in the computation. In actual scenarios, companies are able to end up defaulting on their loans.

The Jupyter notebook used in this paper can be accessed here: https://colab.research.google.com/drive/1sOTOqH9IuIJc9IIbQ8qeoPvmXCGQObXg?usp=sharing. An explanation of the code can be found in the notebook.

Conclusion

The transition matrix shows that over time, companies are less likely to end up with an AAA rating, and are more likely to end up with ratings in the range BBB to CCC/C. This applies to any company with any starting rating.

With the information given to him, the investor now knows that both companies satisfy his criteria. However, given that Sika AG's probability to go down in ratings is greater than AkzoNobel, he decided to invest in AkzoNobel instead.

References

Anton, Howard. 2010. *Elementary Linear Algebra.* 10th ed. Hoboken, NJ: Wiley.

Blanco-Castañeda, Liliana, and Viswanathan Arunachalam. 2023. *Applied Stochastic Modeling.* Cham: Springer Nature Switzerland. Accessed June 26, 2024. doi:10.1007/978-3-031-31282-3.

Cash, Daniel. 2021. *Sustainability Rating Agencies vs Credit Rating Agencies: The Battle to Serve the Mainstream Investor.* Cham: Springer International Publishing. Accessed June 26, 2024. doi:10.1007/978-3-030-71693-6.

Horn, Roger A., and Charles R. Johnson. 2012. *Matrix Analysis:.* 2. Cambridge University Press. Accessed June 24, 2024. doi:10.1017/CBO9781139020411.

Langohr, Herwig M., and Patricia T. Langohr, . 2012. *The Rating Agencies and their Credit Ratings: What They Are, How They Work and Why They Are Relevant.* 1. Wiley. Accessed June 26, 2024. doi:10.1002/9781119208785.

Pinsky, Mark A., Samuel Karlin, and Howard M. Taylor. 2011. *An introduction to stochastic modeling.* Fourth edition. Burlington, MA: Academic Press/Elsevier.