# 1 Stochastic Processes

### Random Variables

A random variable $X(t)$ is a variable that takes on values at random, i.e. it has (potentially) different values for given $t$'s.
If $t \in \mathbb{N}$, $X$ is a discrete time random variable and can also be denoted as $X_t$. The set of all $X(t)$ is *countable*. An example of discrete time random variables are stock market closing prices, because their values change in discrete intervals (once per trading day).
With $t \in \mathbb{R}$, $X$ is a continous time random variable, i.e. its value can change from one point in time to the next. The common example for continous time random variables is the queue length in a bank, because the queue length may change at any given time.
A random variable $X_t$ is *memoryless*, if its future behaviour only depends on its current state and not on any state before that. The property of memorylessness can also be written as:

$$P(X_{n+1} = X|X_n, X_{n-1}, ..., X_1, X_0) = P(X_{n+1} = X|X_n) \tag{1}$$

### State Space

The state space of a random variable $X(t)$ is its domain, i.e. the set of all possible values $X(t)$ can take on.

# 2 DTMCs - Discrete Time Markov Chains

A discrete time Markov chain or DTMC can be used to describe a system with a *finite* or *infinite countable* number of states. The system may only be in one state at any given time and may only change its state at discrete points in time. For a system to be representable by a DTMC the system has to fulfill the *Markov Property* and therefore must be memoryless. This means that the selection of the next state of the system *must only* depend on the previous state if we look on first order Markov chains. Making this assumption is a simplification of real systems most of the time and often leads to invalid or inaccurate results.
The Markov property is almost true for (among others):

- roulette results
- playing the lottery
- coin tossing

## 2.1 Representations

### 2.1.1 Graphical Representation

A DTMC can be represented as a *directed graph* where the nodes represent states of the Markov chain and edges represent transitions between those states. The edge weights denote the probabilities of the respective transitions.
An invariant of all DTMCs is that the sum of all edge weights of edges leaving one node has to be exactly 1.

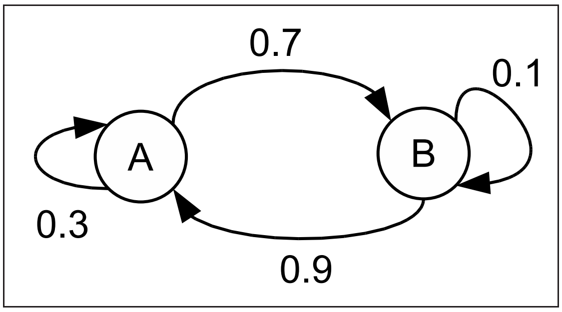


Figure 1: Example of a DTMC

### 2.1.2 Mathematical Representation

#### Stochastic Matrices

A square matrix $P_{n\times n}$ is said to be a *stochastic matrix*, if the sum of the values of every single row is one:

$$\sum_j P_{i,j} = 1 \tag{2}$$

So the matrix $A = \begin{pmatrix} 0.3 & 0.7 \\ 0.9 & 0.1 \end{pmatrix}$ is a stochastic matrix (and in addition also a transition probability matrix) and the matrix $B = \begin{pmatrix} 0.9 & -0.4 \\ 0.2 & 1.3 \end{pmatrix}$ is not.

#### Transition Probability Matrix

If a matrix fullfils the property (2) and additionally the property

$$0 \leq P_{i,j} \leq 1 \ \forall i, j$$

the matrix is said to be a *transition probability matrix*. A transition probability matrix has the form:

$$P = \begin{pmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,n} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ P_{n,1} & P_{n,2} & \cdots & P_{n,n} \end{pmatrix}$$

Here, the matrix element $P_{i,j}$ is the probability for the system to go from state $i$ to state $j$ in one step and $P_{i,i}$ is the probability to stay in state $i$.
Similarly, ${P_{i,j}}^{(n)}$ is the probability for the system to go from state $i$ to state $j$ in exactly $n$ steps.

#### Probability Vector

A vector containing the probabilities of the system to be in each state at a given point in time is called *probability vector*. It is of the form

$$\Pi = \begin{pmatrix} P(S_1) \\ P(S_2) \\ \vdots \\ P(S_n) \end{pmatrix}$$

where $P(S_j)$ denotes the probability of the system to be in state $S_j$ at the given point in time and will be abbreviated as $P_j$ in the following.

This requires the vector $\Pi^T(P_1, P_2, ..., P_n)$ to satisfy the constraint

$$(\sum_{i=1}^{n} P_i) = 1$$

An *initial state vector* $\Pi_0$ is a probability vector describing the system state at time 0 (initial time).
The transition from one DTMC state to the next one can be computed with the *balance equation* (here $\Pi_i$ denotes the $i$th element of $\Pi$):

$$\Pi_i = \sum_j \Pi_j P_{j,i}$$

This formula can be rewritten to a vector equation and then be used to calculate a state vector $\Pi_{n+1}$ from a known $\Pi_n$:

$$\Pi_{n+1}^T = \Pi_n^T P \tag{3}$$

**Steady State**

A steady state is a system state (represented by a state vector) that does not change over time. If a DTMC has a steady state, there exists a steady state vector $\Pi_n$ with:

$$\Pi_n = \Pi_{n+1}$$

By using equation (3) this can also be written as:

$$(\Pi_n)^T = {\Pi_n}^T P$$

However not all DTMCs do have a steady state. Let's take for the example the DTMC desribed by $P = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. Solving the DTMC with the initial probabiltiy vector $(\pi_0)^T = (1, 0)$ here will never lead to a steady state solution but to a infinite altering between the values of (1,0) and (0,1). This behavior is called periodic. With the theorem of Person-Froebenius it can be determined if a transient solution will lead to a periodic solution or not.

### 2.1.3 Introduction of some property definitions

- time homogeneous:
  $P(X_2 = x_2 | X_1 = x_1) = P(X_{n+1} = x_2 | X_n = x_1)$
  $P_{ij}(n) = p_{ij} = P(X_{n+1} = x_i | X_n = x_i)$
- reachable states:
  $p_{ij}^{(n)} > 0$
- communicating states:
  $p_{ij}^{(n)} > 0 \; and \; p_{ji}^{(n)} > 0$
- irreducible states
  $\forall i, j : i \leftrightarrow j$, in words: all i,j are communicating
- periodic states:
- recurrent states:
  $P_{ii}^{(n)} = 1$, the probability of coming back to this state is one
- transient states:
  $\exists p_{ii}^{(n)} > 0$, the probability to come back to this state is not zero

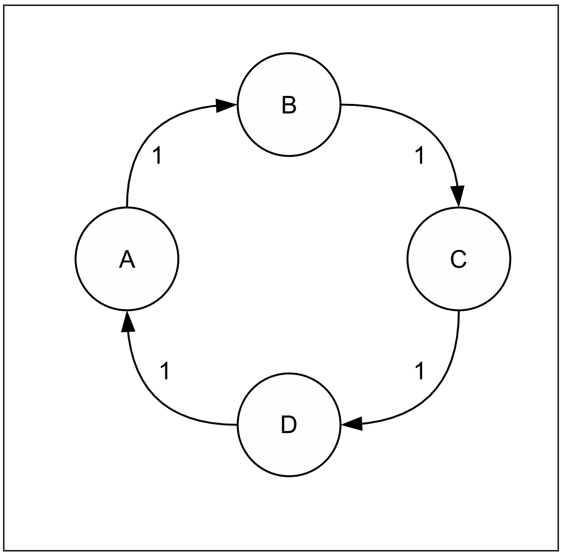


Figure 2: e.g. A is a periodic with d = 4 (d= depth)

- absorbing states:
  $p_{ii} = 1$, the probability to stay in this state is one (there is no possibility to leave this state)
- stationary distribution:
  $\exists \ \ if \ \ \exists z : z \cdot P = z$
- limiting distribution:
  $\exists \pi : \pi = \lim_{n\to\infty} \Pi_n$
- ergodic:
  irreducable, aperiodic and finite ???

If a marcov chain is irreducable and aperiodic then there always exists a $\pi$ independently from the choosen initial state vector $\pi_0$.

#### 2.1.4 Nearly Decomposable DTMCs

If we have given a DTMC as shown in the figure below and we assume that $\epsilon$ is very small the computation of a steady state would be expensive. Especially when we use ${\Pi_0}^T = (1, 0, 0, 0)$ as the inistial state vector. To handle this situation we divide the DTMC into two parts and compute several iterations. After that we reconstruct the old DTMC and use the computed values as our new initial state vector. This proceeding makes the computation of the steady state faster but it doesnt solve the problem. How good or bad this method works depends on the first initial state vector, if this one is a bad choise this method doesn't help computing the steady state of the DTMC much faster than before.
It is very important to know how to handle such DTMCs because this problem emerges quite often in practice.

### 2.2 Solving DTMCs

**System of Linear Equations**

The naive approach to solve a DTMC - i.e. to calculate its steady state vector - is to solve the system of linear equations given by the steady state equation:

$$\Pi^T = \Pi^T P$$

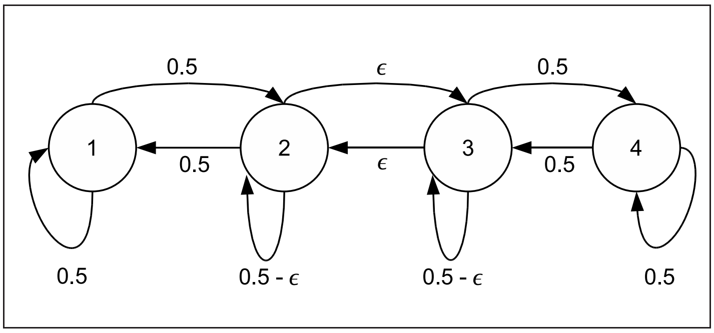


Figure 3: Example of an nearly decomposable DTMC

which can be reorganized to

$$0 = \Pi^T(P - I)$$

where I is the *identity matrix*. (All main diagonal elements are one and all the others zero) However, multiplying this equation by $(P - I)^{-1}$ would only yield

$$\Pi = 0$$

which is a correct solution for the system of linear equations, but is useless for solving the DTMC as it does not satisfy the constraint for probability vectors ($\sum \Pi_i = 1$). This is because we solved a homogenous system of linear equations, which always has either an infinite number of solutions or only the solution $(0, 0, ..., 0)^T$, but $(0, 0, ..., 0)^T$ is always part of the set of solutions.
To avoid this problem, one can either solve a system of linear equations that incorporates the constraint given above. But that would yield a non-square matrix which cannot be inverted easily. Another approach is to start with a solution that satisfies the constraint given above (e.g. the vector $(1, 0, ..., 0)^T$) and use an iterative algorithm to get better results that are closer to the steady state solution in each step:

### 2.2.1 Power Method

The simplest iterative algorithm is to compute

$$\Pi_{n+1}^T = \Pi_n^T P$$

over and over again until the result converges to the steady-state solution.
We also can get $\Pi_{n+1}^T$ by doing the following computation

$$\Pi_{n+1}^T = \Pi_0^T P^{n+1}$$

where only $\Pi_0$ and $P$ are needed. This is called the **power method**. It can easily be implemented, but is computationally expensive.

### 2.2.2 Method of Jacobi

Another iterative approach is to start with the final equation of the system of linear equations

$$0 = \Pi^T \cdot (P - I)$$

or by merging $(P - I)$ to $A$

$$0 = \Pi^T \cdot A$$

Now, instead of solving this equation, one decomposes A to an *upper triangle* ($U$), a *lower triangle* ($L$) and a *diagonal matrix* ($D$), so that

$$A = D - (L + U)$$

This yields

$$0 = \Pi^T(D - (L + U))$$

rearranged

$$\Pi^T(L + U) = \Pi^T D$$

which in turn can be used as a iterative algorithm:

$$\Pi_k^T(L + U) = \Pi_{k+1}^T D$$

as soon as it is rearranged to

$$\Pi_{k+1}^T = \Pi_k^T \cdot D^{-1} \cdot (L + U)$$

### 2.2.3 Gauss-Seidel Method

Here nearly the same approach as for the Jacobi method is used. We start again with $0 = \Pi^T \cdot A$ But use annother composition for A:

$$A = U + L + D$$

This yields

$$0 = \Pi^T(U + L + D)$$

rearranged

$$\Pi^T(D + L) = -\Pi^T U$$

which in turn can be used as a iterative algorithm:

$$\Pi_{k+1}^T(D + L) = -\Pi_k^T U$$

as soon as it is rearranged to

$$\Pi_{k+1}^T = -\Pi_k^T \cdot U \cdot (D + L)^{-1}$$

It has been proven that this algorithm converges, always calculates the correct solution with respect to the constraint for the steady-state solution and is in most cases faster than the power method (in pracise often twice as fast as the power method) and never slower. The proof for correctness itself however would go far beyond the scope of this script and is therefore omitted.

## 2.3 Example(s)

**Which meal for lunch?**

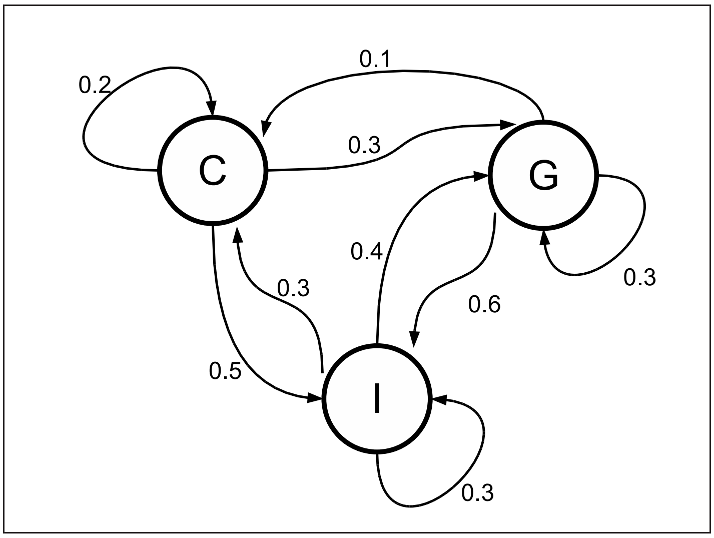


Figure 4: Chinese, Greek or Italian food for lunch?

- task definition:
  - Every noon Mr. M. Infidel goes for lunch to a {Chinese, Greek or Italian} restaurant. A private investigator hired by his wife wants know which restaurant he should observate in the next days and got the following data from Mrs. Bond:
  - probability vector (also initial state vector) $\pi_0 = \begin{pmatrix} \pi_C \\ \pi_G \\ \pi_I \end{pmatrix}_{3x1} = \begin{pmatrix} 1.0 \\ 0.0 \\ 0.0 \end{pmatrix}$
  - transition probability matrix (selected from given picture) P = $\begin{pmatrix} 0.2 & 0.3 & 0.5 \\ 0.1 & 0.3 & 0.3 \\ 0.3 & 0.4 & 0.3 \end{pmatrix}_{3x3}$
- solution:
  $\pi_{k+1} = \pi_k \cdot P$

  auxiliary calculation: $\pi_1 = \pi_0 \cdot P$ $\qquad \downarrow \begin{pmatrix} 0.2 & 0.3 & 0.5 \\ 0.1 & 0.3 & 0.3 \\ 0.3 & 0.4 & 0.3 \end{pmatrix}_{\underline{3}x3}$

  $\rightarrow$

  $\pi_0{}^T = (1.0\ 0.0\ 0.0)_{1x\underline{3}}$ $\qquad \pi_1{}^T = (0.2\ 0.3\ 0.5)$

  $\pi_1{}^T = (0.2000\ 0.3000\ 0.5000)$
  $\pi_2{}^T = (0.2200\ 0.3500\ 0.4300)$
  $\pi_3{}^T = (0.2080\ 0.3430\ 0.4490)$
  $\pi_4{}^T = (0.2106\ 0.3449\ 0.4445)$
  $\pi_5{}^T = (0.2100\ 0.3445\ 0.4456)$
  $\pi_6{}^T = (0.2101\ 0.3446\ 0.4453)$
  $\pi_7{}^T = (0.2101\ 0.3445\ 0.4454)$
  $\pi_8{}^T = (0.2101\ 0.3445\ 0.4454)$