Tutorial Consolidation Week

1. Consider a small town with five coffee shop spots. Each café was evaluated by analysts in terms of its popularity (based on customer ratings and footfall) and accessibility (based on its proximity to transportation links or central areas in the town). The numerical score for each category varied from 1 to 10.

Coffee Shop	Popularity	Accessibility
A	7	8
B	4	6
C	9	6
D	5	7
E	8	5

Suppose we wish to categorise these coffee shops into two groups to assist new residents or visitors in town in quickly identifying the type of cafés they might be interested in. Utilise the K-means algorithm to group the coffee shops. Start with a "qualified guess" by supposing the coffee shops will be grouped between those well-evaluated ($\mu_1^0 = \begin{bmatrix}8 & 8\end{bmatrix}$) and those poorly evaluated for both features ($\mu_2^0 = \begin{bmatrix}3 & 3\end{bmatrix}$).

Solution: Considering the initial guess for all centroids, μ_k^0 for $k = 1, 2$, we have

$$\mu^0 = \begin{bmatrix}8 & 8\\ 3 & 3\end{bmatrix}.$$

The updated configuration for each café will be based on the closest distance for each centroid. For the first café,

$$\begin{aligned}\|x_A - \mu_1\|^2 &= (7-8)^2 + (8-8)^2 = 1\\ \|x_A - \mu_2\|^2 &= (7-3)^2 + (8-3)^2 = 41\end{aligned}$$

Therefore, the coffee shop A is closer to the first centroid, μ_1^0, and therefore, $X_1^1 = [X_{1,1}^1, X_{1,2}^1] = [1, 0]$. Performing the same calculations for each café, we have that:

$$k^1 = \underset{k'=1,2}{\arg\min} \|x_i - \mu_{k'}^0\|^2 = \underset{k'=1,2}{\arg\min} \begin{bmatrix}1 & 41\\ 20 & 10\\ 5 & 45\\ 10 & 20\\ 9 & 29\end{bmatrix} = \begin{bmatrix}1\\ 2\\ 1\\ 1\\ 1\end{bmatrix}$$

which will provide the following configuration for the first iteration

$$X^1 = \begin{bmatrix}1 & 0\\ 0 & 1\\ 1 & 0\\ 1 & 0\\ 1 & 0\end{bmatrix}$$

To update the centroids, we have to compute the following:

$$\mu^1 = \begin{bmatrix}\frac{1}{N_1}\sum_{i=1}^5 X_{i,1}^1 x_i\\ \frac{1}{N_2}\sum_{i=1}^5 X_{i,2}^1 x_i\end{bmatrix} = \begin{bmatrix}7.25 & 6.5\\ 4 & 6\end{bmatrix}$$

Since the configuration $X^1 \neq X^0$, we find a new configuration based on the updated centroids:

$$k^2 = \arg\min_{k'=1,2} \|x_i - \mu^1_{k'}\|^2 = \arg\min_{k'=1,2} \begin{bmatrix} 2.3125 & 13 \\ 10.8125 & 0 \\ 3.3125 & 25 \\ 5.3125 & 2 \\ 2.8125 & 17 \end{bmatrix}$$

which results in the configuration:

$$X^2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

with the updated centroids

$$\mu^2 = \begin{bmatrix} 8 & 6.3 \\ 4.5 & 6.5 \end{bmatrix}$$

Since X^2 is different from X^1, we can use the updated centroid to find a new configuration based on

$$k^3 = \arg\min_{k'=1,2} \|x_i - \mu^2_{k'}\|^2 = \arg\min_{k'=1,2} \begin{bmatrix} 3.78 & 8.5 \\ 16.11 & 0.5 \\ 1.11 & 20.5 \\ 9.44 & 0.5 \\ 1.78 & 14.5 \end{bmatrix}$$

which results in $X^3 = X^2$, meaning that we reached convergence to the problem. The optimal solution groups the five cafés into 2 groups, assigning coffee shops A, C, and E to one group, with evaluations centred around 8 and 6.3 for popularity and accessibility, respectively, while coffee shops B and D were part of the second group that has average popularity of 4.5 and accessibility of 6.5.

2. Consider the task of sorting a list of unique letters, such as $[C, A, R, D]$.
   (a) How can this combinatorial optimization problem be mathematically formalised?
   (b) List the configuration set and problem size of this task.
   (c) Construct the computational graph representing the fully exhaustive SDP for solving this task. Identify the optimal solution in your graph.
   (d) Use the greedy SDP to solve this task, and construct the computational graph representing this solution. For this specific problem, would the greedy solution be exact? Explain.
   (e) Compare the complexity class of the greedy solution with the exhaustive solution for solving this task.

Solution: This is a simple combinatorial search problem that reviews the basics of a combinatorial optimization problem. You can refer to the lecture notes (Sections 3 and 4) to remind yourself about the terminology and notation.

(a) We can express the objective function for this problem as

$$F(X) = \begin{cases} 0, & \text{if } X \text{ is in alphabetical order} \\ 1, & \text{otherwise} \end{cases}$$

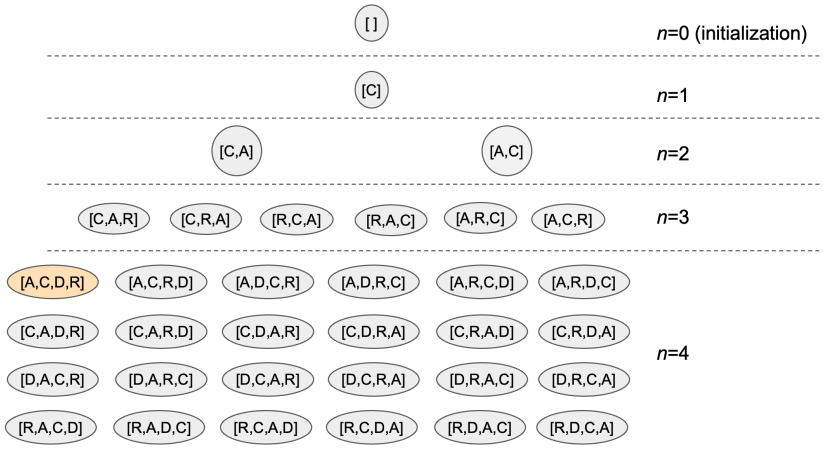
The optimal solution is given by:

$$X^* = \arg\min_{X \in \mathcal{X}} F(X).$$

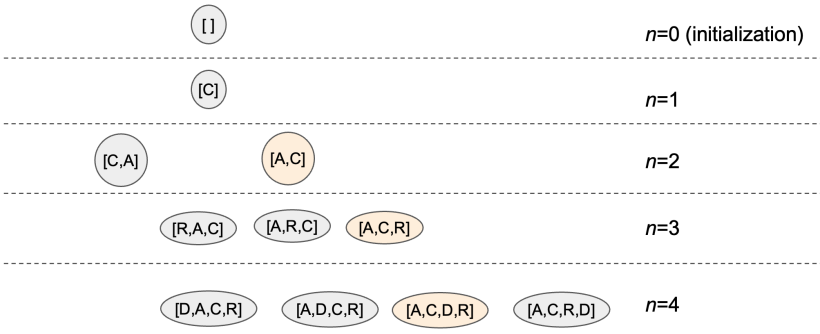
(b) Since there are four unique letters, the number of distinct arrangements (permutations) is given by $4! = 4 \times 3 \times 2 \times 1 = 24$. The configuration set for the problem size $N = 4$ is:

$$\begin{aligned} \mathcal{X} = \{&[C,A,R,D],[C,A,D,R],[C,R,A,D],[C,R,D,A],[C,D,A,R],[C,D,R,A],\\ &[A,C,R,D],[A,C,D,R],[A,R,C,D],[A,R,D,C],[A,D,C,R],[A,D,R,C],\\ &[R,C,A,D],[R,C,D,A],[R,A,C,D],[R,A,D,C],[R,D,A,C],[R,D,C,A],\\ &[D,C,A,R],[D,C,R,A],[D,R,A,C],[D,R,C,A],[D,A,R,C],[D,A,C,R]\} \end{aligned}$$

(c) The computational graph for a fully exhaustive SDP would involve generating all possible permutations and then checking each one to see if it is sorted. At each step, we add the element of the list to generate a set of partial configurations until all elements are added. The configuration set is built from the partial configurations of earlier iterations, as shown below:



(d) We can follow the steps presented in Algorithm 4.1 of the notes to find the optimal solution to this problem. In the greedy approach, the SDP will remove all but the **current best** candidate at each stage.
Initialization: start with an empty set, $S = []$
Iteration for each letter in the list:
Extension: At $n = 1$, we consider 'C' (first item of the list), so that the new set of candidate configurations will be $S = \{[C]\}$.
At $n = 2$, we will have $S = \{[C,A],[A,C]\}$; the reduction step will remove $[C,A]$ since the current best solution with this partial configuration is $[A,C]$;
at $n = 3$, we will have $S = \{[A,C,R],[A,R,C],[R,A,C]\}$, which will result in keeping the configuration $[A,C,R]$ after the reduction step;
at $n = 4$, we will have $S = \{[A,C,R,D],[A,C,D,R],[A,D,C,R],[D,A,C,R]\}$, which leads to the optimal configuration $X^* = [A,C,D,R]$.
The computational graph for the greedy SDP is shown below:



Note that, for this specific task, restricting the solution at each iteration always ensures the possibility of solving the problem exactly, since a non-sorted permutation at the previous stage cannot be extended to a sorted permutation at the next. So, restricting to one sorted permutation at each stage always ensures the possibility of solving the problem exactly (i.e., finding a sorted permutation of the complete set of input data). This makes it both exact and greedy.

(e) By checking the solution to part (b), we can see that the complexity class for the exhaustive problem is $O(N!)$ as it involves generating all permutations of the set at each iteration.
The greedy solution from part (d) will take 1 time unit to compute the first letter (since there's only one configuration possible), 2 time units for the second letter (there are 2 possible configurations), 3 for the third letter, and so on. For the n-th letter, it will take N time units. Summing these up, the total number of computations will take $1+2+...+N$, which is equivalent to $N(N+1)/2$ and leads to $O(N^2)$ (polynomial) complexity. This is significantly lower than the factorial complexity of the exhaustive approach, especially for larger lists.

3. Consider a UK census data analysis scenario with the following facts:

   1. Every household has completed the census questionnaire;
   2. The census data has been accurately processed;
   3. The demographic statistics are up-to-date,

   along with the rule "If every household has completed the census questionnaire and the census data has been accurately processed, then the demographic statistics are considered up-to-date."

   (a) Convert the rule above to clausal form (i.e., translate it into symbols).

   (b) Describe the knowledge base for this case.

   (c) Does the knowledge base entail that if the demographic statistics are up-to-date, then every household has completed the census questionnaire and the census data has been accurately processed?

Solution: This problem reviews the basic aspects of logic and logical inference (Section 8 of the lecture notes).

(a) In order to convert the rule to clausal form, let's first define the following propositions:
H = "Every household has completed the census questionnaire"
P = "The census data has been accurately processed"
S = "The demographic statistics are up-to-date."
Therefore, the rule can be represented as $R = (H \wedge P) \implies S$.

(b) The knowledge base (KB) contains the propositions H, P, and S defined above along with the rule $R = (H \wedge P) \implies S$.

(c) We want to know whether $KB \models Y$, where $Y = S \implies (H \wedge P)$ is the causal representation of the proposition to be tested. Since there are 3 propositions, we have $2^3 = 8$ possible models to be tested, which are given in the first 3 columns of the table below.

H	P	S	$(H \wedge P)$	$R = (H \wedge P) \implies S$	$Y = S \implies (H \wedge P)$
F	F	F	F	T	T
F	F	T	F	T	F
F	T	F	F	T	T
F	T	T	F	T	F
T	F	F	F	T	T
T	F	T	F	T	F
T	T	F	T	F	T
T	T	T	T	T	T

From the table above, we can see that the set of models where KB holds true is explicitly given by

$$M(KB) = \{[F, F, F], [F, F, T], [F, T, F], [F, T, T], [T, F, F], [T, F, T], [T, T, T]\}$$

In parallel, the set of models where Y holds (last column of the table) is given by

$$M(Y) = \{[F, F, F], [F, T, F], [T, F, F], [T, T, F], [T, T, T]\}$$

By comparing the sets, we can easily see that $M(KB) \nsubseteq M(Y)$ because there are elements in $M(KB)$ that are not in $M(Y)$ (i.e., R is true in cases where Y is not). So, $KB \models Y$ is false, i.e., the KB does not entail that if the demographic statistics are up-to-date, then every household has completed the census questionnaire and the census data has been accurately processed. (In fact, the proposition Y is called the logical *converse* of R. Although we tend to intuitively believe that if two propositions, p and q, are so that $p \implies q$, then $q \implies p$ is true, these two statements say different things and therefore do not have the same logical meaning.)

p => q
doesn't mean
q => p

but if p<=>q
then q =>q