Assignment #4
Solving 0-1 Knapsack Problem using Gibbs Sampling

In this assignment, you are going to use Gibbs Sampling to find an approximate solution to a modified version of the 0-1 Knapsack Problem. The original 0-1 Knapsack Problem is defined as follows. We have \( n \) items, 1 through \( n \). Each item \( i \) has a value (or profit) \( v_i \) and a weight (or cost) \( w_i \). We assume that all values and weights are nonnegative. We have a bag, i.e., the knapsack, and the maximum total weight that we can carry in the bag is \( W \). In the 0-1 Knapsack Problem our goal is to decide which of the \( n \) items we will put in the bag, so that we will maximize our profit and will the total weight of the items put in the bag will not exceed \( W \). Formally, the problem is:

\[
\text{maximize } \sum_{i=1}^{n} v_i x_i \\
\text{subject to } \sum_{i=1}^{n} w_i x_i \leq W, \quad x_i \in \{0, 1\}
\]

where \( x_i \) is the indicator variable which shows whether item \( i \) is put inside the bag (\( x_i = 1 \)) or not (\( x_i = 0 \)).

This problem is a combinatorial optimization problem. In this assignment you are going to solve a modified version of the 0-1 Knapsack Problem defined above. The modification adds another constraint to the optimization problem and requires the number of taken items to be exactly \( k \). Formally, the following constraint is added:

\[
\sum_{i=1}^{n} x_i = k
\]

Informally, the modification requires you to search for subsets of size exactly \( k \) in the set of \( n \) items. In the original problem, the number of items that can be put in the bag is not restricted and all subsets of different sizes should be searched. The original 0-1 Knapsack Problem is known to be an NP-complete problem. Therefore, there is no polynomial time, i.e., efficient, algorithm to find the optimal solution. In the modified version if the number \( k \) is small, e.g., 2, efficient searching can be performed, e.g., \( O(n^2) \) for \( k=2 \). In this assignment, \( n \) is 1000 and \( k \) is 20, so finding the optimal solution will not be possible in a reasonable amount of time, i.e. about \( 1000^{20} \) subsets should be tested. Therefore, you are going to search this huge search space using Gibbs sampling.
For Gibbs sampling formulation of the problem, you will have \( k=20 \) parameters to search for. In other words, \( \theta = (\theta_1, \theta_2, \ldots, \theta_{20}) \). Each \( \theta_i \) is a random variable that indicates the identifier of an item that is put inside the bag. As expected, \( (\theta_1 \neq \theta_2 \neq \theta_3 \ldots \neq \theta_{20}) \). You are going to start with an initial configuration \( \theta^0 \), i.e., an initial set of items, which fit in the bag. At each iteration you will take out one of the 20 items and given the 19 items that are currently in the bag, you will randomly sample from the rest of the \( n-19 \) items based on the following conditional distribution:

\[
P(\theta_i = z \mid \theta_1, \theta_2, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_{20}) = \begin{cases} 
0 & \text{if } w_z > W - \sum_{\forall \theta_j, j \neq i} w_{\theta_j} \\
\frac{v_z}{\sum_{\forall m \text{ s.t. } w_m \leq W - \sum_{\forall \theta_j, j \neq i} w_{\theta_j} \text{ and } \theta_j \neq \theta_i} v_m} & \text{if } w_z \leq W - \sum_{\forall \theta_j, j \neq i} w_{\theta_j}
\end{cases}
\]

Informally, the probability of an item \( z \) is 0 if it does not fit in the bag as the 20\(^{th} \) item given the 19 items already in the bag and the probability is greater than 0 if it fits in the bag. The probability is computed as the ratio of the value of item \( z \) over summation of the values of all the items that are outside the bag and can fit in the bag, including \( z \). You are going to sample from this distribution randomly at each step of the Gibbs sampling. In other words, your goal is not to select the item with the highest value that can fit in the bag; with a certain probability an item with a lesser value can be selected as \( \theta_i \).

In this assignment, you will implement a Gibbs sampler in R for this problem as described above. You will run it on the dataset provided at [http://www.ceng.metu.edu.tr/~tcan/bin504_20101/hw4_input.txt](http://www.ceng.metu.edu.tr/~tcan/bin504_20101/hw4_input.txt). The parameters are \( n=1000, k=20 \), \( W=700 \). In the input file, each row contains tab separated number and describes and item. The first number is the item id, the second number is the item’s value (profit) and the last number is the item’s weight (cost). Your Gibbs sampler may iterate until convergence or until a predefined number of maximum iterations, e.g., 100,000 iterations.

As for the initial distribution \( \theta^0 \), you will try two different choices. In the first choice, you will initialize \( \theta^0 \) by selecting the 20 items with minimum weights among the 1000 items. In case of ties you will choose the items with higher values. In the second choice, you will select a random set of 20 items which fit in the bag, i.e., their total weight \( \leq W \). Which choice converges faster? Can you suggest a better initialization strategy?

**Deliverables:**

A short report with your Gibbs sampling solutions for the two different initial values of \( \theta^0 \) and answers to the questions above.

Your R code.