1. The class NP contains all decision problems where ‘yes’ answers can be verified (proved) in polynomial time. Show that MAP inference is NP-hard. A problem is NP-Hard if given an O(1) oracle to solve it, every problem in NP can be solved in polynomial time. i.e. An NP-hard problem is at least as hard as any problem in class NP.

The class #P is the class of problems that count the number of accepting paths in a Turing machine that is nondeterministic and runs in polynomial time. i.e. #P is the problem of finding the number of solutions to a problem in NP.

3-SAT asks about the satisfiability of a logical formula defined on $n$ literals (binary variables or their negations), $Q_1, Q_2, ..., Q_n$. e.g.

$$(\neg Q_1 \land Q_2 \lor Q_3) \lor (\neg Q_4 \land Q_2 \lor Q_5) \land ...$$

where the disjunction terms is called a clause, e.g.

$C(Q_1, Q_2, Q_3) = \neg Q_1 \land Q_2 \lor Q_3$

In 3-SAT, each clause contains at most 3 literals.

Now show that the MAP inference in Bayesian networks is NP-hard.

Consider a 3-SAT formula, with $n$ literals $Q_1, Q_2, ..., Q_n$, and $m$ clauses $C_1, C_2, ..., C_m$, and $X$ as the output.

$$p(q, c, X = 1) = 0$$

for any $q$ that does not satisfy all clauses.

$$p(Q = q, C = 1, X = 1) = \frac{1}{2^n}$$

for any satisfying assignment.

Therefore, we can find a satisfying assignment by finding the MAP assignment

$$\text{argmax}_{q, c} p(Q = q, C = c | X = 1)$$

2. Recap of Gibbs Sampling and MH algorithm

**MH algorithm summary**

- Draws a sample $x'$ from $Q(x'|x)$, where $x$ is the previous sample.
The new sample $x'$ is accepted or rejected with some probability $A(x'|x) = \min(1, \frac{P(x')Q(x|x')}{P(x)Q(x'|x)})$.

In case that $Q$ is symmetric, i.e. $Q(x|x') = Q(x'|x)$ (Gaussian, etc.), the acceptance probability simplifies to $\min(1, \frac{P(x')}{P(x)})$.

**pseudo-code for M-H algorithm**

1. Initialize starting state $x^{(0)}$, set $t = 0$.

2. Burn-in: while samples have “not converged”:
   - $x = x^{(t)}$
   - $t = t+1$
   - sample $x* \sim Q(x*|x)$ (draw proposal)
   - sample $u \sim Uniform(0,1)$ (draw acceptance threshold)
   - if $u < A(x*|x)$: $x^{(t)} = x*$ (accept, make state transition)
   - else: $x^{(t)} = x$ (reject, stay in current state)

3. Takes samples from $P(x)$: after observing convergence, do the same as 2 to sample from the distribution.

Example code of M-H sampling from $\mathcal{N}(3,4)$:

```python
import random
import math
from scipy.stats import norm
import matplotlib.pyplot as plt

def norm_dist_prob(theta):
    y = norm.pdf(theta, loc=3, scale=2)
    return y

T = 5000
pi = [0 for i in range(T)]
sigma = 1
t = 0
while t < T-1:
    t = t + 1
    pi_star = norm.rvs(loc=pi[t - 1], scale=sigma, size=1, random_state=None)
    alpha = min(1, (norm_dist_prob(pi_star[0]) / norm_dist_prob(pi[t - 1])))
    u = random.uniform(0, 1)
    if u < alpha:
        pi[t] = pi_star[0]
    else:
        pi[t] = pi[t - 1]
```

**Gibbs sampling**
• Let $x^{(t)}$ be the initial assignment to variables.
• Set $t = 1$
• while true:
  – for $i = 1...J$:
    * sample $x_i^{(t+1)} \sim p(x_i | \{x_j^{(t)} (j \neq i)\})$
    * set $x_i^{(t+1)}$ to $x_i^{(t)}$
    * $t = t+1$

3. Consider $X_1, ..., X_n$ being i.i.d. Poisson($\lambda$). Show that a $\text{Gamma}(\alpha, \beta)$ prior on $\lambda$ is a conjugate prior, and find the posterior distribution.

Likelihood:
\[
L(\lambda | x) = \prod_{i=1}^{n} \frac{\exp(-\lambda)\lambda^{x_i}}{x_i!} = \frac{\exp(-n\lambda)\lambda^{\sum_i x_i}}{\prod_i x_i!}
\]

Prior:
\[
p(\lambda) \sim \text{Gamma}(\alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} \exp(-\beta\lambda)
\]

Posterior:
\[
p(\lambda) \propto L(\lambda | x)p(\lambda) \propto \lambda^{\sum_i x_i + \alpha - 1} \exp(-(\beta + n)\lambda)
\]

So $p(\lambda)$ is $\text{Gamma}(\sum_i x_i + \alpha, n + \beta)$

4. Gibbs sampling can proceed either rotationally (sweeping through indices $i$) or randomly (by sampling $i$). For the purposes of this problem consider the version where $i$ is sampled randomly with probability $\pi_i$. **Show that Gibbs sampling satisfies detailed balance.**

Detailed balance means that for each pair of states $x$ and $x'$, (1) arriving at $x$ then $x'$ and (2) arriving at $x'$ then $x$ are equiprobable. That is,
\[
S(x' \leftarrow x)p(x) = S(x \leftarrow x')p(x')
\]

First, let’s consider the transition probability $S(x' \leftarrow x)$. Since Gibbs sampling samples from the full conditionals, this probability is given by:
\[
S(x' \leftarrow x) = \pi_i p(x'_i | x_{\setminus i})
\]

Next, let’s compute the left hand side and right hand sides of the detailed balance equation separately.

LHS:
\[
S(x' \leftarrow x)p(x) = \pi_i p(x'_i | x_{\setminus i})p(x) = \pi_i p(x'_i | x_{\setminus i})p(x_i | x_{\setminus i})p(x_{\setminus i})
\]
RHS:

\[
S(x \leftarrow x') p(x') = \pi_i p(x_i|x'_i) p(x') \\
= \pi_i p(x_i|x'_i) p(x'_i|x_i) p(x'_i) \\
= \pi_i p(x_i|x_i) p(x'_i|x_i) p(x_i') \\
= S(x' \leftarrow x) p(x)
\]

where the second to last step follows from the observation that \(x'_i = x_i\) because Gibbs sampling holds the other variables constant when updating the \(i\)th variable. Thus, detailed balance holds.

Note: to prove detailed balance for the version of Gibbs sampling where we sweep through indices \(i\), we would consider the update after a full sweep.