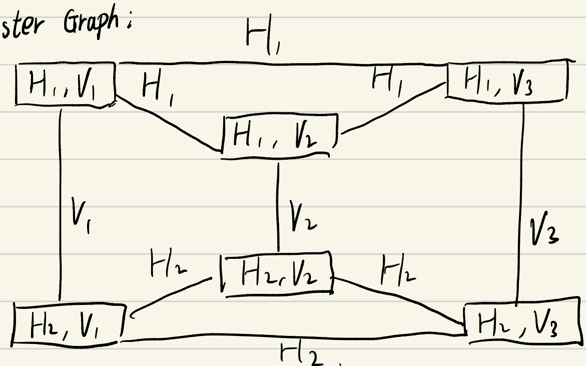
第6次作业

自硕21 崔晏菲 2021210976

1. 解:

(1) Gibbs distribution $= \frac{1}{Z} \exp\{-(\sum_{i=1}^{2} \alpha_i h_i + \sum_{i=1}^{3} \beta_i V_i + \sum_{i=1}^{2}\sum_{j=1}^{3} W_{ij} h_i V_j)\}$

(2) Cluster Graph:



Belief propagation 代码为:

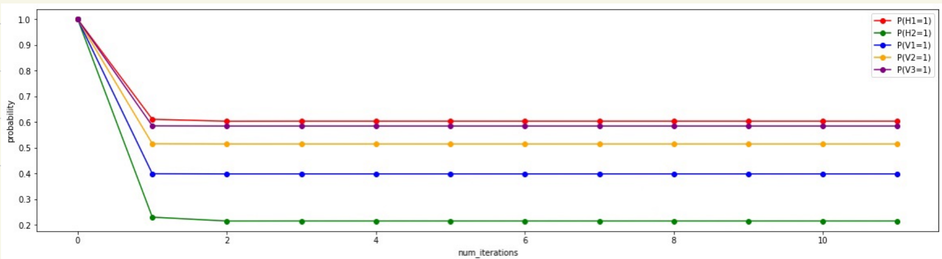
```
def Belief_propagation(self):
    delta = 1e-20
    potentials = self.markov_network + self.markov_network.T
    for i in range(potentials.shape[0]):
        potentials[i,i] /= 2
    messages = np.ones((potentials.shape[0], potentials.shape[1], 2))
    beliefs = np.ones((potentials.shape[0], 2))
    before = None
    belief_iterations = beliefs[:,1].reshape(-1,1).copy()

    while(True):
        for t in range(potentials.shape[0]):
            for s in range(potentials.shape[1]):
                s_neighbors = set(list(np.where(potentials[s]!=0)[0]))
                s_neighbors.discard(s)
                s_neighbors.discard(t)
                s_neighbors = list(s_neighbors)
                messages[s,t, 0] = np.exp(-(potentials[s,s]*0+potentials[s,t]*0*0))*np.prod(messages[s_neighbors, s, 0]) + np.exp(-(potentials[s,s]*1+potentials[s,t]*1*0))*np.prod(messages[s_neighbors,
                messages[s,t, 1] = np.exp(-(potentials[s,s]*0+potentials[s,t]*0*1))*np.prod(messages[s_neighbors, s, 0]) + np.exp(-(potentials[s,s]*1+potentials[s,t]*1*1))*np.prod(messages[s_neighbors,
                messages[s,t] = 8*messages[s,t]/np.sum(messages[s,t])
        for s in range(beliefs.shape[0]):
            s_neighbors = set(list(np.where(potentials[s]!=0)[0]))
            s_neighbors.discard(s)
            s_neighbors = list(s_neighbors)
            beliefs[s, 0] = np.exp(-potentials[s,s]*0)*np.prod(messages[s_neighbors, s, 0])
            beliefs[s, 1] = np.exp(-potentials[s,s]*1)*np.prod(messages[s_neighbors, s, 1])
        beliefs = beliefs/(np.sum(beliefs, axis=1, keepdims=True)+1e-12)

        if(before is None):
            before = beliefs.copy()
            belief_iterations = np.concatenate([belief_iterations, before[:,1].reshape(-1,1)], axis=1)
        elif(np.sum(np.abs(beliefs-before))>=delta):
            before = beliefs.copy()
            belief_iterations = np.concatenate([belief_iterations, before[:,1].reshape(-1,1)], axis=1)
        else:
            belief_iterations = np.concatenate([belief_iterations, beliefs[:,1].reshape(-1,1)], axis=1)
            break

    return beliefs, belief_iterations
```

迭代了12次就收敛了.



得到的边缘分布如下，可以看到和真实值十分接近.

Belief propagation得到的每个节点的边缘分布为：

	节点	取值	BP得到的边缘分布	真实边缘分布
0	H1	0	0.396029	0.396086
1	H1	1	0.603971	0.603914
2	H2	0	0.784402	0.784246
3	H2	1	0.215598	0.215754
4	V1	0	0.601171	0.601142
5	V1	1	0.398829	0.398858
6	V2	0	0.484688	0.484701
7	V2	1	0.515312	0.515299
8	V3	0	0.414948	0.414986
9	V3	1	0.585052	0.585014

(3) 设 $Q(x)$ 是5个伯努利分布互相独立的.

则 $q_j(x_j) \propto \exp\{E_{-q_j}[\ln \tilde{p}(x)]\}$

而 $E_{-q_j}[\ln \tilde{p}(x)] = E_{-q_j}\left(\sum_{i,j} x_i x_j\right) + C$

$$= x_j \cdot \sum_{i \in Neighbr(j)} [-W_{ij}\theta_i] - W_{jj} x_j + C$$

记 $m_j = \sum_{i \in Neighbr(j)} [W_{ij}\theta_i]$

代码为：

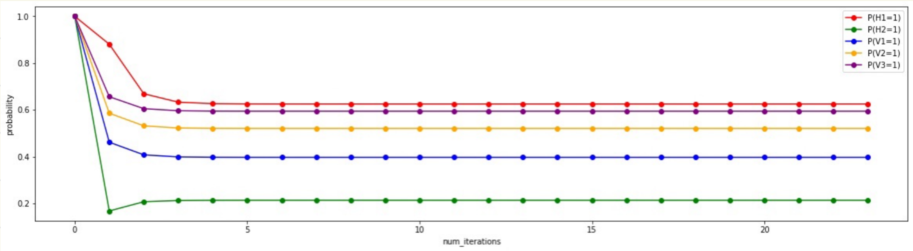
```python
def Mean_field_inference(self):
    delta = 1e-20
    potentials = self.markov_network + self.markov_network.T
    Q = np.ones((potentials.shape[0], 2))
    for i in range(potentials.shape[0]):
        potentials[i,i] /= 2
    before = None
    Q_iterations = Q[:,1].reshape(-1,1).copy()

    while(True):
        for j in range(potentials.shape[0]):
            j_neighbors = set(list(np.where(potentials[j]!=0)[0]))
            j_neighbors.discard(j)
            j_neighbors = list(j_neighbors)
            index = np.where(potentials[j]!=0, 1, 0)
            mj0 = 0*np.sum(-potentials[j, j_neighbors]*Q[j_neighbors,0]) - potentials[j,j]*0
            mj1 = 1*np.sum(-potentials[j, j_neighbors]*Q[j_neighbors,1]) - potentials[j,j]*1
            Q[j] = np.exp(np.array([mj0, mj1]))
            Q[j] = Q[j]/np.sum(Q[j])

        if(before is None):
            before = Q.copy()
            Q_iterations = np.concatenate([Q_iterations, before[:,1].reshape(-1,1)], axis=1)
        elif(np.sum(np.abs(Q-before))>=delta):
            before = Q.copy()
            Q_iterations = np.concatenate([Q_iterations, before[:,1].reshape(-1,1)], axis=1)
        else:
            Q_iterations = np.concatenate([Q_iterations, Q[:,1].reshape(-1,1)], axis=1)
            break

    return Q, Q_iterations
```

迭代了24次就收敛了.



得到的边缘分布为:

Mean field variational inference得到的每个节点的边缘分布为:

	节点	取值	MFI得到的边缘分布	真实边缘分布
0	H1	0	0.374662	0.396086
1	H1	1	0.625338	0.603914
2	H2	0	0.786248	0.784246
3	H2	1	0.213752	0.215754
4	V1	0	0.602864	0.601142
5	V1	1	0.397136	0.398858
6	V2	0	0.479365	0.484701
7	V2	1	0.520635	0.515299
8	V3	0	0.405503	0.414986
9	V3	1	0.594497	0.585014

可以看到和真实值较为接近，但有一定偏差.

(4) 经过比较我们可以发现，平均场推断在很短迭代内就会收敛，最终结果有偏差。偏差是无法避免的，因为Q和P本身就有偏差。而Gibbs sampling和MH sampling经过足够长的迭代次数，总会收敛到真实值.