CHAPTER 30

Solutions

1.1

$$p(x,y|z) = \frac{p(x,y,z)}{p(z)} = \frac{p(y|x,z)p(x,z)}{p(z)} = p(y|x,z)p(x|z) \tag{30.0.1}$$

$$p(x|y,z) = \frac{p(x,y,z)}{p(y,z)} = \frac{p(y|x,z)p(x,z)}{p(y,z)} = \frac{p(y|x,z)p(x|z)}{p(y|z)} \tag{30.0.2}$$

1.2

$$p(a \cup b) = p(a) + p(b) - p(a,b) \leq 1 \rightarrow p(a,b) \geq p(a) + p(b) - 1 \tag{30.0.3}$$

1.3

$$p(box = 1|redball) = \frac{p(box = 1, redball)}{p(redball)} \tag{30.0.4}$$

$$= \frac{p(redball|box = 1)p(box = 1)}{p(redball|box = 1)p(box = 1) + p(redball|box = 2)p(box = 2)} \tag{30.0.5}$$

$$= \frac{3/8}{3/8 + 2/7} = \frac{21}{37} \tag{30.0.6}$$

1.4 We want

$$p(2 \text{ red in box}|\ 3 \text{ red drawn}) = \frac{p(3 \text{ red drawn}|2 \text{ red in box})p(2 \text{ red in bag})}{p(3 \text{ red drawn})}$$

A priori, there are 3 possibilities for what balls can be in the box

$$a: \text{ 2 white balls in box,} \qquad p(a) = 1/4 \tag{30.0.7}$$

$$b: \text{ 2 red balls in box,} \qquad p(b) = 1/4 \tag{30.0.8}$$

$$c: \text{ 1 red 1 white ball in box,} \qquad p(c) = 1/2 \tag{30.0.9}$$

In the use of Bayes' rule above we have

$$p(3 \text{ red drawn}|2 \text{ red in box})p(2 \text{ red in box}) = 1 \times 1/4 \tag{30.0.10}$$

Also, using the shorthand for the above events:

$$p(3 \text{ red drawn}) = p(3 \text{ red drawn}|a)p(a) + p(3 \text{ red drawn}|b)p(b) + p(3 \text{ red drawn}|c)p(c) \tag{30.0.11}$$

$$= 0 + 1 \times 1/4 + 0 + 1/8 \times 1/2 \tag{30.0.12}$$

Hence the result is 4/5.

1.5

$$p(terr = \mathsf{tr}|scan = \mathsf{tr}, inf) = \frac{p(scan = \mathsf{tr}|terr = \mathsf{tr}, inf)p(terr = \mathsf{tr}|inf)}{p(scan = \mathsf{tr}|terr = \mathsf{tr}, inf)p(terr = \mathsf{tr}|inf) + p(scan = \mathsf{tr}|terr = \mathsf{fa}, inf)p(terr = \mathsf{fa}|inf)} \tag{30.0.13}$$

$$= \frac{0.95 \times 0.01}{0.95 \times 0.01 + 0.05 \times 0.99} = 0.161 \tag{30.0.14}$$

1.6 $2 + 2 + 1 = 5$. This is compared to the 7 parameters in a general three binary-variable distribution.

1.7 See `demoClouseau2.m`.

1.8

$$a \cap (b \cup c) = (a \cap b) \cup (a \cap c) \tag{30.0.15}$$

so

$$p(a \cap (b \cup c)) = p(a \cap b) + p(a \cap c) - p(a \cap b \cap a \cap c) = p(a, b) + p(a, c) - p(a, b, c) \tag{30.0.16}$$

1.9

$$p(x|z) = \sum_y p(x, y|z) = \sum_y \frac{p(x, y, z)}{p(z)} = \sum_y \frac{p(x|y, z)p(y, z)}{p(z)} = \sum_y p(x|y, z)p(y|z) \tag{30.0.17}$$

Similar argument for second assertion, beginning with:

$$p(x|z) = \sum_{y,w} p(x, y, w|z) \tag{30.0.18}$$

1.10 The issue here is one of interpretation. Essentially, what the confidence argument is saying is that if we were able to repeat this experiment of visiting Berlin at some point during its wall's lifetime, and use the procedure above to predict the end time, then 95% of the time our predictions will be correct. However, this is an odd thing to want. Intuitively, we are really interested in

$$p(t_{end}|t_{now}) = \frac{p(t_{now}|t_{end})p(t_{end})}{p(t_{now})} \tag{30.0.19}$$

To manipulate this quantity, we must use a prior $p(t_{end})$. Perhaps a less passionate example is that you are given a uniformly generated random positive value from the interval $[1, M]$ with unknown maximum M. Your task is to estimate M based on this single observed value. For example, you observe the value 80. You then say, OK with 95% confidence I think M is between 82 and 3200. I then tell you the true answer. We then repeat this experiment, giving each time a possible range for M. In the limit of many repetitions, the true M each time will indeed lie in the predicted interval in 95% of the cases. However, this is really more a statement about the procedure than an interesting statement about M for any particular experiment. We really want to know something about the Berlin wall, not about the delta-t method. To answer that, we must use a prior.
See `www.cs.ucl.ac.uk/staff/D.Barber/publications/tipping-barber-future.pdf` for a more detailed discussion.

1.11 See `softxor.m`.

1.12 The point here is that if we wish to define a distribution $p(ham|KJ)$ for use with BRMLtoolbox, we are not explicitly given $p(ham = \mathsf{tr}|KJ = \mathsf{fa}) = \gamma$. However, we can figure out this probability using

$$\underbrace{p(ham = \mathsf{tr})}_{\alpha} = \underbrace{p(ham = \mathsf{tr}|KJ = \mathsf{tr})}_{0.9}\underbrace{p(KJ = \mathsf{tr})}_{\beta} + \underbrace{p(ham = \mathsf{tr}|KJ = \mathsf{fa})}_{\gamma}\underbrace{p(KJ = \mathsf{fa})}_{1-\beta} \tag{30.0.20}$$

where $\beta = 1/100000$, $\alpha = 1/2$. Solving this gives

$$\gamma = \frac{\alpha - 0.9\beta}{1 - \beta} \tag{30.0.21}$$

which can then be used to define the table entries required for BRMLtoolbox. See `hamburger.m`.

1.13 See `twoDiceExample.m`.

1.14 There will be $1000, 000/100 = 10000$ people each week that choose $3, 5, 7, 9$. If they win, they will each take home £$1000, 0000/10000 =$ £100. The probability that $3, 5, 7, 9$ (or any four numbers) arises each week is $4!/(9 \times 8 \times 7 \times 6)$. The expected winnings for someone choosing $3, 5, 7, 9$ on any given week is therefore £0.7937. Given that it costs £1 to enter each week, they will loose around 21 pence a week. On the other hand for the least popular number $1, 2, 3, 4$, if this number comes up, each player will win £10, 000, so that the average profit per week is £$79.37 - 1 =$ £78.37. The lottery is purely random, but there is 'skill' involved in maximising the amount of money one wins (contrary to some government definitions of a lottery for which skill should play no part in the 'success').

1.15 Probability of getting $0, 1, 2, 3, 4, 5$ correct matches is $44/120, 45/120, 20/120, 10/120, 0, 1/120$. The expected number of correct matches is $(45 + 2 \times 20 + 3 \times 10 + 5)/120 = 1$. Probability of at least 1 correct is 1 minus the probability of no correct matches $1 - 44/120 = 76/120$. The clever way to do this is using 'rencontres numbers'. The brute force enumeration approach is in `psychometry.m`.

1.17
1. A nice way to show this is to consider using 'dividers' that separate the 7 friends into their pizza choices. For example $oo|ooo|o|o$ would say that two have chosen pizza A, three pizza B, one pizza C and one pizza D. We are interested in the number of such partitions. There are then $7 + 3 = 10$ positions, each position containing either a friend o or a divider |. We can place the dividers in $10 \times 9 \times 8 = 720$ positions. Since the dividers have themselves $3 \times 2 = 6$ arrangements, the number of partitions is $720/6 = 120$.

2. Note that this is not 1/120 – this would be the probability of the chef uniformly choosing one of the 120 partitions. However, he chooses a pizza at random, so that some partitions are more probable than others. For example, it is less likely that all friends chose pizza A, than two friends choosing pizza A, two friends pizza B, two friends pizza C and one friend pizza D. One needs therefore to compute these probabilities, $p_i, i = 1, \ldots, 120$. Since the chef and friends chose independently, the probability of agreement is given by $\sum_i p_i^2 \approx 0.0184$. See `pizza.m`.

1.18 Assuming that $p(\mathsf{Alice}) = p(\mathsf{Bella}) = 0.5$, $\mathrm{dom}(f) = \{\mathsf{Alice}, \mathsf{Bella}\}$, and $p(sushi, car|f) = p(sushi|f)p(car|f)$, then

$$p(\mathsf{Alice}|\mathsf{dislike\ sushi, white\ car}) = \frac{p(\mathsf{dislike\ sushi, white\ car}|\mathsf{Alice})p(\mathsf{Alice})}{p(\mathsf{dislike\ sushi, white\ car}|\mathsf{Alice})p(\mathsf{Alice}) + p(\mathsf{dislike\ sushi, white\ car}|\mathsf{Bella})p(\mathsf{Bella})} \tag{30.0.22}$$

$$= \frac{0.5 \times 0.9 \times 0.5}{0.5 \times 0.9 \times 0.5 + 0.1 \times 0.5 \times 0.5} = 0.9 \tag{30.0.23}$$

1.19

1. The transition matrix $p(w_{t+1}|w_t)$ for $w \in \{\mathsf{rain}, \mathsf{run}\}$ is

$$\begin{pmatrix} p(\mathsf{rain}|\mathsf{rain}) & p(\mathsf{rain}|\mathsf{sun}) \\ p(\mathsf{sun}|\mathsf{rain}) & p(\mathsf{sun}|\mathsf{sun}) \end{pmatrix} = \begin{pmatrix} 0.7 & 0.6 \\ 0.3 & 0.4 \end{pmatrix}$$

We just use then Bayes' rule to compute the probability.

$$p(w_t = \mathsf{rain}|w_{t+1} = \mathsf{sun}) = \frac{p(w_{t+1} = \mathsf{sun}|w_t = \mathsf{rain})p(w_t = \mathsf{rain})}{p(w_{t+1} = \mathsf{sun}|w_t = \mathsf{rain})p(w_t = \mathsf{rain}) + p(w_{t+1} = \mathsf{sun}|w_t = \mathsf{sun})p(w_t = \mathsf{sun})} = 3/7$$

where we used $p(\mathsf{rain}) = p(\mathsf{sun}) = 0.5$

2. In this case, the stationary distribution can be found either by explicitly solving for the stationary distribution, or just multiplying the transition matrix. The stationary distribution satisfies

$$\begin{pmatrix} p(\mathsf{rain}|\mathsf{rain}) & p(\mathsf{rain}|\mathsf{sun}) \\ p(\mathsf{sun}|\mathsf{rain}) & p(\mathsf{sun}|\mathsf{sun}) \end{pmatrix} \begin{pmatrix} p(\mathsf{rain}) \\ p(\mathsf{sun}) \end{pmatrix} = \begin{pmatrix} p(\mathsf{rain}) \\ p(\mathsf{sun}) \end{pmatrix}$$

Using $p(\mathsf{sun}) = 1 - p(\mathsf{rain})$ gives the equation

$$0.7p(\mathsf{rain}) + 0.6(1 - p(\mathsf{rain})) = p(\mathsf{rain})$$

This gives $p(\mathsf{rain}) = 2/3, p(\mathsf{sun}) = 1/3$.

3. Using $p(\mathsf{rain}) = 2/3$, $p(\mathsf{sun}) = 1/3$ in the previous calculation, we arrive at $p(w_t = \mathsf{rain}|w_{t+1} = \mathsf{sun}) = 3/5$.

2.1 Let n^k_{ij} be the number of paths from j to i in k steps. Then $n^k_{ij} = \sum_l A_{il} n^{k-1}_{lj}$ since the number of paths must be the number of paths that get one to an intermediate node l in $k-1$ steps, provided one can get from l to i in an additional step. In matrix notation this is $N^{(k)} = AN^{(k-1)} = A^2N^{(k-2)} = A^k$.

2.2 From the previous exercise, we know that $\left[A^k\right]_{ij}$ is the number of paths from j to i in k steps. If a subset of variables is connected, there must be a path of some length, less than or equal to n (the number of nodes in the graph) between them. Defining $B_{ij} = 1$ if $\sum_k \left[A^k\right]_{ij} > 0$, and zero otherwise, B_{ij} is 1 if i and j are connected by some path. We then look at column i (or equivalently a row) of B, which describes the nodes that can be reached from node i, and put these in a connected component. Since connected components are disjoint, we can then remove these variables from consideration, and continue to another column to find another set of connected variables.

2.3 $1 - 2 - 3 - 1$ and $4 - 5$ is multiply connected and satisfies $E = V - 1$.

2.4 If the graph is connected and the number of edges is less than the number of nodes, it must be a tree. However, to deal with the general case in which it is unknown if the graph is connected we check using elimination. A tree/singly-connected graph must admit a recursive simplical node elimination. That is at any stage in the elimination there must be a node with either zero or 1 neighbours in the remaining graph. See `istree.m`.

2.5
```
done=false;
a=parents(A,x); % start with the parents of x
while ~done
    aold=a;
    a=union(a,parents(A,a)); % include the parents of the current ancestors
    done=isempty(setdiff(a,aold)); % done if this doesn't introduce any more nodes
end
a=setdiff(a,x);
```
See `ancestors.m`.

2.6 If we interpret i 'knowing' j as the same as j knowing i, we need to first symmetrize the matrix $\mathbf{A}$. Then if $\left[\mathbf{A}^k\right]_{ij}$ is non-zero then there is a length k path between i and j. However, in order to find the separation, we need to make sure that we only count the lowest length path between i and j. This means that the separation is k, provided that no shorter length path already exists. Doing so, we find that the number of unique pairs with separations from 1 to 7 are

$$1898, 19988, 55258, 46679, 13709, 1781, 96$$

with no longer separations. See `makeWikiGraph.m`.

Remarks

1. There is some ambiguity in the question and the answer will depend on whether you interpret that i 'knows' j, implies that j 'knows' i. (That is you make the matrix A symmetric by using $A = A + A'$)

2. A common error is to simply look for a length k path between two users i and j. This is not what is intended by 'separation'. What we want is the length of the *shortest* path that connects i and j.

2.7 There are many different ways to do this based on finding which cliques are wholly contained in another. The solution is

$$119, 447, 463, 474, 487, 703, 751, 755, 765, 831, 863, 886, 893, 954, 983, 1006, 1013$$

The main routing for finding the unique cliques is

```
uniqueflag=true(1,length(cl));
for i=1:length(cl)
    for j=find(uniqueflag)
        if i~=j
            if isempty(setdiff(cl{i},intersect(cl{i},cl{j})));
                uniqueflag(i)=false;
                break;
            end
        end
    end
end
```

This first assumes all cliques are unique and then looks at clique i. The code checks all other remaining unique cliques to see if clique i is a subset of one of them. If so, clique i is removed from the list of unique cliques, and the search continues. See `cliquesproblem.m`.

Remarks
Note that there are some cliques that are exactly identical. A common error is to remove both two cliques when they are identical, leaving only 15 cliques, rather than the 17 unique cliques.

2.8 We construct a bipartite graph, placing half of the nodes in the upper part and half in the lower. We then connect the both parts together.

2.9 In this case every maximal clique sizes must be $N/3$ since if we were to include another node, this means that there must be one of the $N/3$ partitions that contains two nodes in the proposed clique – however this cannot be the case since the lack of connections within a partition prohibits such a clique formation. Each clique can then be formed by choosing one of the 3 nodes from each of the $N/3$ partitions, giving $3^{N/3}$ maximal cliques.

Remarks
This result is perhaps at first sight surprising and helps explain why finding maximal cliques or enumerating all the cliques in a graph is in general a hard problem.

3.1 The answer is 0.609660. See `partyanimal.m`.

3.2 (i) a and b are independent. (ii) a and b are dependent given c.

3.3 $tuberculosis \perp\!\!\!\perp smoking|\, shortness\ of\ breath = \mathsf{fa}$; $lung\ cancer \perp\!\!\!\perp bronchitis|\, smoking = \mathsf{tr}$; $visit\ to\ Asia \perp\!\!\!\perp smoking|\, lung\ cancer = \mathsf{tr}$; $visit\ to\ Asia \perp\!\!\!\perp smoking \mid lung\ cancer, shortness\ of\ breath = \mathsf{fa}$. One can verify this using `demoChestClinic; A=dag(pot); [condindep(A,xray,smoker,dys) condindep(A,lcancer,bronch,smoker) condindep(A,asia,smoker,lcancer) condindep(A,asia,smoker,[lcancer dys])]`

1. $tuberculosis \perp smoking \mid shortness\,of\,breath$. Paths connecting t to s are:
(i) $t \to \underset{\substack{\downarrow\\ \mathbf{d}}}{e} \leftarrow l \leftarrow s$. Path is not blocked since node e is a collider and its decendant, node d, is in the conditioned set. The non collider nodes on this path are not being conditioned on.
(ii) $t \to e \to \mathbf{d} \leftarrow b \leftarrow s$. Path is not blocked for the same reasons.
So $tuberculosis \perp smoking \mid shortness\,of\,breath$ is false.

2. $lung\,cancer \perp bronchitis \mid smoking$. Paths connecting l to b are:
(i) $l \leftarrow \mathbf{s} \to b$. Path is blocked. Node s is not a collider and is in the conditioning set.
(ii) $l \to e \to d \leftarrow b$. Path is blocked. Node d is a collider and is not in the conditioning set.
Every path from l to b is blocked given s so $lung\,cancer \perp bronchitis \mid smoking$ is true.

3. $visit\,to\,Asia \perp smoking \mid lung\,cancer$. Paths connecting a to s are:
(i) $a \to t \to e \leftarrow \mathbf{l} \leftarrow s$. Path is blocked by l - node is not a collider and is in conditioning set.
(ii)$a \to t \to e \to d \leftarrow b \leftarrow s$. Path is blocked by node d - collider and not in conditioning set.
$visit\,to\,Asia \perp smoking \mid lung\,cancer$ true since all paths are blocked.

4. $visit\,to\,Asia \perp smoking \mid lung\,cancer$, $shortness\,of\,breath$. Paths connecting a to s are:
(i) $a \to t \to e \leftarrow \mathbf{l} \leftarrow s$. Again, path is blocked by l - node is not a collider and is in conditioning set.
(ii)$a \to t \to e \to \mathbf{d} \leftarrow b \leftarrow s$. Path is not blocked by d since it is collider and is in the conditioning set - all other nodes on this path are not colliders and are not in conditioning set.
$visit\,to\,Asia \perp smoking \mid lung\,cancer$, $shortness\,of\,breath$ is false - since there exists unblocked paths from node a to node s.

3.4 The marginal $p(d)$ can be calculated as

$$\begin{aligned} p(d) &= \sum_{a,s,t,l,b,e,x} p(a,s,t,l,b,e,x,d) \\ &= \sum_{a,s,t,l,b,e,x} p(a)p(s)p(t|a)p(l|s)p(b|s)p(e|t,l)p(x|e)p(d|b,e) \\ &= \sum_{a,s,t,l,b,e} \left(p(a)p(s)p(t|a)p(l|s)p(b|s)p(e|t,l)p(d|b,e) \sum_x p(x|e) \right) \\ &= \sum_{s,t,l,b,e} \left(p(s)p(l|s)p(b|s)p(e|t,l)p(d|b,e) \sum_a p(t|a)p(a) \right), \end{aligned}$$

where we have noted that $\sum_x p(x|e) = 1$. The final term on the RHS is just the marginal $p(t)$, which is

$$\begin{aligned} p(t = \mathsf{tr}) &= p(t = \mathsf{tr}|a = \mathsf{tr}) \times p(a = \mathsf{tr}) + p(t = \mathsf{tr}|a = \mathsf{fa}) \times p(a = \mathsf{fa}) \\ &= 0.05 \times 0.01 + 0.01 \times 0.99 = 0.01 \times 1.04 = 0.0104. \end{aligned}$$

Armed with this, we can further simplify the expression for $p(d)$ to

$$p(d) = \sum_{s,l,b,e} \left(p(s)p(l|s)p(b|s)p(d|b,e) \sum_t p(e|t,l)p(t) \right).$$

The last term on the RHS is now $p(e|l)$, which is

$$\begin{aligned} p(e = \mathsf{tr}|l = \mathsf{tr}) =& p(e = \mathsf{tr}|l = \mathsf{tr}, t = \mathsf{tr}) \times p(t = \mathsf{tr}) + p(e = \mathsf{tr}|l = \mathsf{tr}, t = \mathsf{fa}) \times p(t = \mathsf{fa}) \\ =& 1 \times 0.0104 + 1 \times 0.9896 = 1, \\ p(e = \mathsf{tr}|l = \mathsf{fa}) =& p(e = \mathsf{tr}|l = \mathsf{fa}, t = \mathsf{tr}) \times p(t = \mathsf{tr}) + p(e = \mathsf{tr}|l = \mathsf{fa}, t = \mathsf{fa}) \times p(t = \mathsf{fa}) \\ =& 1 \times 0.0104 + 0 \times 0.9896 = 0.0104. \end{aligned}$$

The marginal $p(d)$ is now

$$p(d) = \sum_{s,b,e} p(s)p(b|s)p(d|b,e) \sum_l p(e|l)p(l|s),$$

which we can further simplify by calculating $p(e|s)$, which is

$$\begin{aligned} p(e = \mathsf{tr}|s = \mathsf{tr}) =& p(e = \mathsf{tr}|l = \mathsf{tr}) \times p(l = \mathsf{tr}|s = \mathsf{tr}) + p(e = \mathsf{tr}|l = \mathsf{fa}) \times p(l = \mathsf{fa}|s = \mathsf{tr}) \\ =& 1 \times 0.1 + 0.0104 \times 0.9 = 0.10936, \\ p(e = \mathsf{tr}|s = \mathsf{fa}) =& p(e = \mathsf{tr}|l = \mathsf{tr}) \times p(l = \mathsf{tr}|s = \mathsf{fa}) + p(e = \mathsf{tr}|l = \mathsf{fa}) \times p(l = \mathsf{fa}|s = \mathsf{fa}) \\ =& 1 \times 0.01 + 0.0104 \times 0.99 = 0.020296. \end{aligned}$$

This now gives us

$$p(d) = \sum_{s,b} p(b|s)p(s) \sum_e p(d|b,e)p(e|s),$$

leading us to calculate $p(d|b,s)$, which is

$$\begin{aligned} p(d = \mathsf{tr}|b = \mathsf{tr}, s = \mathsf{tr}) =& p(d = \mathsf{tr}|b = \mathsf{tr}, e = \mathsf{tr}) \times p(e = \mathsf{tr}|s = \mathsf{tr}) + p(d = \mathsf{tr}|b = \mathsf{tr}, e = \mathsf{fa}) \times p(e = \mathsf{fa}|s = \mathsf{tr}) \\ &= 0.9 \times 0.10936 + 0.8 \times 0.89064 = 0.8109360, \\ p(d = \mathsf{tr}|b = \mathsf{tr}, s = \mathsf{fa}) =& p(d = \mathsf{tr}|b = \mathsf{tr}, e = \mathsf{tr}) \times p(e = \mathsf{tr}|s = \mathsf{fa}) + p(d = \mathsf{tr}|b = \mathsf{tr}, e = \mathsf{fa}) \times p(e = \mathsf{fa}|s = \mathsf{fa}) \\ &= 0.9 \times 0.020296 + 0.8 \times 0.979704 = 0.8020296, \\ p(d = \mathsf{tr}|b = \mathsf{fa}, s = \mathsf{tr}) =& p(d = \mathsf{tr}|b = \mathsf{fa}, e = \mathsf{tr}) \times p(e = \mathsf{tr}|s = \mathsf{tr}) + p(d = \mathsf{tr}|b = \mathsf{fa}, e = \mathsf{fa}) \times p(e = \mathsf{fa}|s = \mathsf{tr}) \\ &= 0.7 \times 0.10936 + 0.1 \times 0.89064 = 0.1656160, \\ p(d = \mathsf{tr}|b = \mathsf{fa}, s = \mathsf{fa}) =& p(d = \mathsf{tr}|b = \mathsf{fa}, e = \mathsf{tr}) \times p(e = \mathsf{tr}|s = \mathsf{fa}) + p(d = \mathsf{tr}|b = \mathsf{fa}, e = \mathsf{fa}) \times p(e = \mathsf{fa}|s = \mathsf{fa}) \\ &= 0.7 \times 0.020296 + 0.1 \times 0.979704 = 0.1121776 \end{aligned}$$

We now have

$$p(d) = \sum_s p(s) \sum_b p(d|b,s)p(b|s),$$

so we calculate $p(d|s)$, which is

$$\begin{aligned} p(d = \mathsf{tr}|s = \mathsf{tr}) =& p(d = \mathsf{tr}|b = \mathsf{tr}, s = \mathsf{tr}) \times p(b = \mathsf{tr}|s = \mathsf{tr}) + p(d = \mathsf{tr}|b = \mathsf{fa}, s = \mathsf{tr}) \times p(b = \mathsf{fa}|s = \mathsf{tr}) \\ &= 0.8109360 \times 0.6 + 0.1656160 \times 0.4 = 0.5528080, \\ p(d = \mathsf{tr}|s = \mathsf{fa}) =& p(d = \mathsf{tr}|b = \mathsf{tr}, s = \mathsf{fa}) \times p(b = \mathsf{tr}|s = \mathsf{fa}) + p(d = \mathsf{tr}|b = \mathsf{fa}, s = \mathsf{fa}) \times p(b = \mathsf{fa}|s = \mathsf{fa}) \\ &= 0.8020296 \times 0.3 + 0.1121776 \times 0.7 = 0.3191332. \end{aligned}$$

Now, we can calculate $p(d) = \sum_s p(d|s)p(s)$, which is

$$\begin{aligned} p(d = \mathsf{tr}) =& p(d = \mathsf{tr}|s = \mathsf{tr}) \times p(s = \mathsf{tr}) + p(d = \mathsf{tr}|s = \mathsf{fa}) \times p(s = \mathsf{fa}) \\ & 0.5528080 \times 0.5 + 0.3191332 \times 0.5 = 0.4359706 \end{aligned}$$

Thus we have $p(d = \mathsf{tr}) = 0.4359706$, $p(d = \mathsf{tr}|s = \mathsf{tr}) = 0.5528080$ and $p(d = \mathsf{tr}|s = \mathsf{fa}) = 0.3191332$.

3.5 $p(d = true||b = false)$ could be inferred by removing the b node from the graph and setting $b = false$ in node b's children (*i.e.* node d). This result would differ from $p(d = true|b = false)$ since this inference calculation would include the factor $p(b = false|s)$ whereas the causal calculation would not.

3.6 $p(fuel = empty|start = false) \approx 0.4537$. See `carstart.m`.

3.7
1. A and S are independent : $p(A, S) = p(A)p(S)$.
2. Include extra variable B : $p(C|A, S)p(A|B)p(S|B)P(B)$

3.8 All results are in `demoHolmes.m`. The first part is straightforward.

$$\begin{aligned}
p(B = tr|W = tr) &= \frac{\sum_{A,G} p(B = tr, A, G, W = tr)}{\sum_{A,B,G} p(B, A, G, W = tr)} && (30.0.24)\\
&= \frac{p(B = tr)\sum_A p(A|B = tr)p(W = tr|A)\sum_G p(G|A)}{p(B = tr)\sum_A p(A|B = tr)p(W = tr|A) + p(B = fa)\sum_A p(A|B = fa)p(W = tr|A)} && (30.0.25)\\
&= \frac{0.01 \times (0.99 \times 0.9 + 0.01 \times 0.5)}{0.01 \times (0.99 \times 0.9 + 0.01 \times 0.5) + 0.99 \times (0.05 \times 0.9 + 0.95 \times 0.5)} && (30.0.26)\\
&= 0.0171 && (30.0.27)
\end{aligned}$$

$$\begin{aligned}
p(B = tr|W = tr, G = fa) &= \frac{\sum_A p(B = tr, A, G = fa, W = tr)}{\sum_{A,B} p(B, A, G = fa, W = tr)}\\
&= \frac{p(B = tr)\sum_A p(A|B = tr)p(W = tr|A)p(G = fa|A)}{p(B = tr)\sum_A p(A|B = Tr)p(W = tr|A)p(G = fa|A) + p(B = fa)\sum_A p(A|B = fa)p(W = tr|A)p(G = fa|A)}\\
&= \frac{0.01 \times (0.99 \times 0.9 \times 0.3 + 0.01 \times 0.5 \times 0.3)}{0.01 \times (0.99 \times 0.9 \times 0.3 + 0.01 \times 0.5 \times 0.3) + 0.99 \times (0.95 \times 0.9 \times 0.3 + 0.95 \times 0.5 \times 0.8)}\\
&= 0.0069
\end{aligned}$$

The second part can be solved by computing $p(B|W)$:,

$$p(B = tr|\tilde{W}) = \sum_W p(B = tr|W)p(W|\tilde{W}) \tag{30.0.28}$$

So we need to calculate $p(B = tr|W = fa)$

$$\begin{aligned}
p(B = tr|W = fa) &= \frac{p(B = tr)\sum_A p(A|B = tr)p(W = fa|A)}{p(B = tr)\sum_A p(A|B = tr)p(W = fa|A) + p(B = fa)\sum_A p(A|B = fa)p(W = fa|A)}\\
&= \frac{0.01 \times (0.99 \times 0.1 + 0.01 \times 0.5)}{0.01 \times (0.99 \times 0.1 + 0.01 \times 0.5) + 0.99 \times (0.05 \times 0.1 + 0.95 \times 0.5)}\\
&= 0.0022
\end{aligned}$$

So,

$$\begin{aligned}
p(B = tr|\tilde{W}) &= 0.0171 \times 0.3 + 0.0022 \times 0.7 && (30.0.29)\\
&= 0.0068 && (30.0.30)
\end{aligned}$$

Assuming $\tilde{W}$ and $\tilde{G}$ are independent then:

$$p(B = tr|\tilde{W}, \tilde{G}) = \sum_{W,G} p(B = tr|W, G)p(W|\tilde{W})p(G|\tilde{G})$$

$$\begin{aligned}
p(B = tr|W = tr, G = tr) &= \frac{p(B = tr, W = tr, G = tr)}{p(W = tr, G = tr)} = \frac{\sum_A p(A, B = tr, W = tr, G = tr)}{\sum_{A,B} p(A, B, W = tr, G = tr)} = 0.0472\\
p(B = tr|W = fa, G = fa) &= \frac{p(B = tr, W = fa, G = fa)}{p(W = fa, G = fa)} = \frac{\sum_A p(A, B = tr, W = fa, G = fa)}{\sum_{A,B} p(A, B, W = fa, G = fa)} = 0.00089\\
p(B = tr|W = fa, G = tr) &= \frac{p(B = tr, W = fa, G = tr)}{p(W = fa, G = tr)} = \frac{\sum_A p(A, B = tr, W = fa, G = tr)}{\sum_{A,B} p(A, B, W = fa, G = tr)} = 0.0072
\end{aligned}$$

Hence

$$p(B = tr|\tilde{W}, \tilde{G}) = 0.0069 \times 0.3 \times 0.9 + 0.0472 \times 0.1 \times 0.1 + 0.0089 \times 0.7 \times 0.9 + 0.0072 \times 0.7 \times 0.1 = 0.0043$$

3.9 (1) $a \to (r, d); g \to (d, r); d \to r$. (2) This can be computed using standard inference. (3). In this case we need to remove the term $p(d|a, g)$ from the distribution and clamp d to drug and g to state young.

$$p(A, G, D, R) = p(R|A, D, G)p(D|A, G)p(A)p(G) \tag{30.0.31}$$

$$p(R = tr|D = tr) = \frac{\sum_{A,G} p(A, G, D = tr, R = tr)}{\sum_{A,G,R} p(A, G, D = tr, R)} \tag{30.0.32}$$

To calculate $p(R = tr|do(drug), A = young)$ we need to remove the D node from the graph and set its children's potentials to $Drug = true$ and then carry out standard inference. So,

$$p(R = tr|do(drug), A = young) = \frac{\sum_G p(R|A = young, D = tr, G)p(A = young)p(G)}{\sum_{G,R} p(R|A = young, D = tr, G)p(A = young)p(G)} \tag{30.0.33}$$

$$= \sum_G p(R|A = young, D = tr, G)p(G) \tag{30.0.34}$$

3.10 See wetgrass.m.

3.11 Need to include a link from B to E and replace $p(B)$ with the factor $p(B|E)$ in the distribution. $p(B|E)$ is suitably defined so that $p(B = \mathsf{tr}|E = \mathsf{tr})$ is higher than $p(B = \mathsf{tr})$.

3.12 See MarkovEquiv.m and immoralities.m. Finding the skeleton is easy since we just need to remove the arrows by symmetrising the adjacency matrix. To find the immoralities, we can make a new adjacency matrix using

```
function IM=immoralities(A)
%IMMORALITIES find the immoralities of a DAG
% IM=immoralities(A)
% immoralities are returned in the matrix IM
% If there is a child C with unconnected parents A,B, the IM(A,B)=IM(B,A)=1
% and IM(A,C)=IM(B,C)=1;
% That is we link the unmarried parents and connect them to the child
N=size(A,1); IM=zeros(N,N);
A=A>0; % convert to logical (faster)
B=~(A|A'); % B is the matrix of non-connected nodes (either direction)
B=triu(B,1); % ignore lack of self-connections and only need to non-connectivity in one direction
for i=1:N
    pa=brml.parents(A,i); % get the parents of node i
    [ind1 ind2]=find(B(pa,pa)); % find those parents that are not connected
    for j=1:length(ind1)
        IM(pa(ind1(j)),pa(ind2(j)))=1; % connect parents
        IM(pa(ind2(j)),pa(ind1(j)))=1; % .. both directions
        IM(pa(ind1(j)),i)=1;  % connect a parent to the child
        IM(pa(ind2(j)),i)=1;  % and the other parent to child
    end
end
```

3.13 The two graphs are Markov equivalent. See MarkovEquivExample.m. The skeletons are clearly the same and both have the same parent-child-parent immoralities, namely $1-3-2$, $1-3-5$, $4-9-7$.

3.14 The only possibilities consistent with $\mathcal{C}$ are

C_{13}	C_{23}	C_{21}	C_{12}	C_{32}	probability
0	0	0	1	0	$p(1-p)^4$
0	0	0	1	1	$p^2(1-p)^3$
0	0	1	1	0	$p^2(1-p)^3$
0	0	1	1	1	$p^3(1-p)^2$
1	0	0	1	0	$p^2(1-p)^3$
1	0	0	1	1	$p^3(1-p)^2$
1	0	0	0	1	$p^2(1-p)^3$

where $p = p(C_{ij} = 1)$, $i \neq j$. Then

$$p(C_{12} = 1|\mathcal{C}) = \frac{p(C_{12} = 1, \mathcal{C})}{p(\mathcal{C})} = \frac{p(1-p)^4 + 3p^2(1-p)^3 + 2p^3(1-p)^2}{p(1-p)^4 + 4p^2(1-p)^3 + 2p^3(1-p)^2} = \frac{110}{119} \approx 0.9244$$

$$p(C_{13} = 1|\mathcal{C}) = \frac{p(C_{13} = 1, \mathcal{C})}{p(\mathcal{C})} = \frac{19}{119} \approx 0.1597$$

$$p(C_{23} = 1|\mathcal{C}) = \frac{p(C_{23} = 1, \mathcal{C})}{p(\mathcal{C})} = 0$$

$$p(C_{32} = 1|\mathcal{C}) = \frac{p(C_{32} = 1, \mathcal{C})}{p(\mathcal{C})} = \frac{20}{119} \approx 0.1681$$

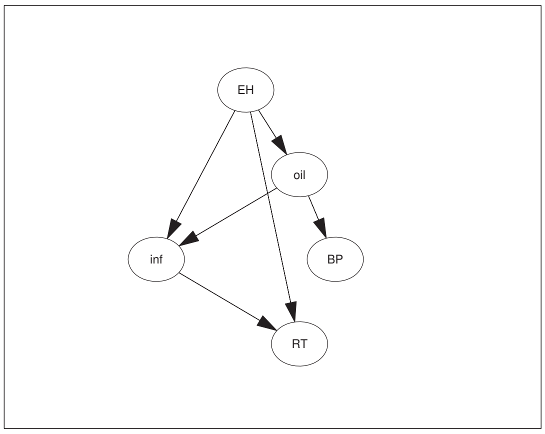
$$p(C_{21} = 1|\mathcal{C}) = \frac{p(C_{21} = 1, \mathcal{C})}{p(\mathcal{C})} = \frac{10}{119} \approx 0.0840$$

$$p(C_{31} = 1|\mathcal{C}) = \frac{p(C_{31} = 1, \mathcal{C})}{p(\mathcal{C})} = p(C_{31} = 1) = 0.1$$

See also `ConnectedComputerProblem.m` which suggests how this may be extended to more general situations numerically.

3.15 See `EconHealth.m`.

```
inf  =high 0.984746
inf  =low  0.015254
```



3.16 See `uniquepots.m`. The basic idea is to form a tree of the form $A_{ji} = 1$ if the variables of potential i are a subset of the variables of potential j. The tree is formed such that each potential has at most a single parent. Given this tree we can identify the connected components which will form the new unique potentials. For each connected component we first instantiate each non-maximal clique with its potential. We then find the parent of each node (starting from the leaves) and multiply the parent potential with the node potential. We retain then only the potentials of the root leaves of the connected components.

```
function [newpot A] = uniquepots(inpot,varargin)
%UNIQUEPOTS Eliminate redundant potentials (those contained wholly within another) by multiplying redundant potentials
% [newpot A]= uniquepots(pot,<tables>)
% if tables=0 then just merge the variables
% The matrix has A(j,i)=1 if pot(i) is contained within pot(j)
import brml.*
inpot=str2cell(inpot); % conver to cell array if needed
tables=1; if nargin==2; tables=varargin{1}; end
% Find a tree of cliques by identifying a single parent clique j that contains clique i.
C=length(inpot); r=zeros(1,C);
for i=1:C; r(i)=isempty(inpot{i}.variables); end
pot=inpot(~r);C=length(pot);
A=sparse(C,C);
for i=1:C
    j=1;
    while j<=C
        A(j,i)=ischildpot(pot,i,j);
        if A(j,i);break;end
        j=j+1;
    end
end
newpot=pot;
% Now merge from bottom up:
[tree elimset sched]=istree(A); % returns an elimination schedule for all components
remove=zeros(1,C);
if tables
    for s=1:size(sched,1)
        ch=sched(s,1); pa=sched(s,2);
        if ch~=pa
            newpot{pa}=multpots(newpot([pa ch]));
            remove(ch)=1;
        end
    end
else
    for s=1:size(sched,1)
        ch=sched(s,2); pa=sched(s,1);
        if ch~=pa
            newpot{pa}.variables=union(newpot{ch}.variables,newpot{pa}.variables);
            remove(ch)=1;
        end
    end
end
newpot=newpot(remove==0);

function m = ischildpot(pot,i,j)
if i==j; m=0; return; end
m=all(ismember(pot{i}.variables,pot{j}.variables));
```

3.17 See `ABCproblem.m`.

1. We need to show that $p(a,c) = p(a)p(c)$:

```
[a b c]=assign([1 2 3]);
pA=array(a,[3/5 2/5]);
pBgA=array([b a],[1/4 15/40; 1/12 1/8; 2/3 1/2]);
pCgB=array([c b],[1/3 1/2 15/40; 2/3 1/2 5/8]);
pABC=pA*pBgA*pCgB;

pAC=sum(pABC,b);
pApC = sum(pAC,a)*sum(pAC,c);
```

This gives

```
p(a,c):

ans =

    0.2250    0.3750
    0.1500    0.2500

p(a)p(c):

ans =

    0.2250    0.3750
    0.1500    0.2500
```

showing that the distributions are the same and hence that $a \perp\!\!\!\perp c$.

2.
$$p(a=i,c=k) = \sum_j p(a=i,b=j,c=k) = \frac{1}{Z}\sum_j \phi(a=i,b=j)\psi(b=j,c=k) = \frac{1}{Z}\sum_j M_{ij}N_{kj} = \frac{1}{Z}\left[\mathbf{M}\mathbf{N}^{\mathsf{T}}\right]_{ik}$$

3. If $\mathbf{M}\mathbf{N}^{\mathsf{T}} = \mathbf{m}_0\mathbf{m}_0^{\mathsf{T}}$ then
$$p(a=i,c=k) = \frac{1}{Z}\left[\mathbf{M}\mathbf{N}^{\mathsf{T}}\right]_{ik} = \frac{1}{Z}\left[\mathbf{m}_0\mathbf{m}_0^{\mathsf{T}}\right]_{ik} = \frac{1}{Z}m_{0,i}m_{0,k}$$
Hence $p(a,c)$ factorises into a function of a times a function of c, implying therefore that a and c are independent.

4.
$$\begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \end{pmatrix}\begin{pmatrix} n_{11} & n_{21} \\ n_{12} & n_{22} \\ n_{13} & n_{23} \end{pmatrix} = \begin{pmatrix} m_{11}n_{11}+m_{12}n_{12}+m_{13}n_{13} & m_{11}n_{21}+m_{12}n_{22}+m_{13}n_{23} \\ m_{21}n_{11}+m_{22}n_{12}+m_{23}n_{13} & m_{21}n_{21}+m_{22}n_{22}+m_{23}n_{23} \end{pmatrix}$$
Since
$$\mathbf{m}_1\mathbf{n}_1^{\mathsf{T}} = \begin{pmatrix} m_{11}n_{11} & m_{11}n_{21} \\ m_{21}n_{11} & m_{21}n_{21} \end{pmatrix}$$
and similarly for the other cases $\mathbf{m}_2\mathbf{n}_2^{\mathsf{T}}$, $\mathbf{m}_3\mathbf{n}_3^{\mathsf{T}}$, the result follows.

5.
$$\begin{aligned}\mathbf{M}\mathbf{N}^{\mathsf{T}} &= \mathbf{m}_1\mathbf{n}_1^{\mathsf{T}} + \lambda\mathbf{m}_1\mathbf{n}_2^{\mathsf{T}} + \gamma\mathbf{m}_3\left(\mathbf{n}_1+\lambda\mathbf{n}_2\right)^{\mathsf{T}} \\ &= \left(\mathbf{m}_1+\gamma\mathbf{m}_3\right)\mathbf{n}_1^{\mathsf{T}} + \lambda\left(\mathbf{m}_3+\gamma\mathbf{m}_1\right)\mathbf{n}_2^{\mathsf{T}} = \left(\mathbf{m}_1+\gamma\mathbf{m}_3\right)\left(\mathbf{n}_1+\lambda\mathbf{n}_2\right)^{\mathsf{T}}\end{aligned}$$

6. We now know how to construct the joint distribution $p(a,b,c)$ up to some unknown normalisation constant Z. We simply normalise the table to fill in this missing normalisation constant. The routine `condpot.m` also returns a correctly normalised distribution.

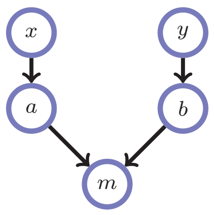
```
M(:,1)=rand(2,1);
lambda=rand;
M(:,2)=lambda*M(:,1);
M(:,3)=rand(2,1);
N(:,1)=rand(2,1);
N(:,2)=rand(2,1);

gamma=rand;
N(:,3)=gamma*(N(:,1)+lambda*N(:,2));
[a b c]=assign([1 2 3]);
```

```
potAB=array([a b],M);
potBC=array([b c],N');
potABC=condpot(potAB*potBC); % normalise

disp('p(a):'); table(condpot(sum(potABC,[b c])))
disp('p(b|a):'); table(condpot(potABC,b,a))'
disp('p(c|b):'); table(condpot(potABC,c,b))'

disp('check if a and c are independent:')
potAC=sumpot(potABC,b);
potApotC = sum(potAC,a)*sum(potAC,c);
disp('p(a,c):');table(potAC)
disp('p(a)p(c):');table(potApotC)
```

This produces for example

```
p(a):

ans =

    0.6876
    0.3124

p(b|a):

ans =

    0.6193    0.7498
    0.0933    0.1130
    0.2873    0.1372

p(c|b):

ans =

    0.1392    0.4014    0.1736
    0.8608    0.5986    0.8264

check if a and c are independent:
p(a,c):

ans =

    0.1194    0.5683
    0.0542    0.2582

p(a)p(c):

ans =

    0.1194    0.5683
    0.0542    0.2582
```

3.18 $A \leftarrow T \rightarrow B$. A and B are dependent since when we sum over T, we introduce a link between A and B.

3.19 

Conditioned on m, x and y are dependent.

3.20 See `demoCars.m`. Intuitively, it's clear in this case that h and w are dependent since, if we know that the husband has a cheap car, the wife must also have a cheap car.

4.1 (1)

$$p(x_1|x_2,x_4) = \frac{\phi(x_1,x_2)\phi(x_4,x_1)}{\sum_{x_1}\phi(x_1,x_2)\phi(x_4,x_1)} \tag{30.0.35}$$

$$p(x_2|x_1,x_3) = \frac{\phi(x_1,x_2)\phi(x_2,x_3)}{\sum_{x_2}\phi(x_1,x_2)\phi(x_2,x_3)} \tag{30.0.36}$$

$$p(x_3|x_2,x_4) = \frac{\phi(x_2,x_3)\phi(x_3,x_4)}{\sum_{x_3}\phi(x_2,x_3)\phi(x_3,x_4)} \tag{30.0.37}$$

$$p(x_4|x_1,x_3) = \frac{\phi(x_3,x_4)\phi(x_4,x_1)}{\sum_{x_4}\phi(x_3,x_4)\phi(x_4,x_1)} \tag{30.0.38}$$

(2) It is not always possible to do this. For example, one can define joint distribution $p_i(x_2, x_2, x_3, x_4)$, $i = 1, \ldots, 4$ which are all different. The conditionals, for example, $p_1(x_1|x_2, x_4)$ then do not come from the same consistent joint distribution.

4.2 Marginal is

$$\phi(a, b = 1)\phi(b = 1, c) + \phi(a, b = 2)\phi(b = 2, c)$$

If $\phi(b = 1, c) = 0$, then a and c are independent.

4.3 The only place $\mathbf{W}$ enters is through the expression $\phi \equiv \mathbf{x}^\mathsf{T}\mathbf{W}\mathbf{x}$. Since this is a scalar, we can take the transpose, so that $\phi \equiv \mathbf{x}^\mathsf{T}\mathbf{W}^\mathsf{T}\mathbf{x}$, and $\phi = \mathbf{x}^\mathsf{T}\frac{1}{2}\left(\mathbf{W} + \mathbf{W}^\mathsf{T}\right)\mathbf{x}$. Hence any non-symmetric $\mathbf{W}$ is essentially converted to a symmetric form, so that we may therefore assume this symmetry with loss of generality.

4.4 (1)

$$p(\mathbf{h}|\mathbf{v}) \propto e^{\mathbf{h}^\mathsf{T}\left(\mathbf{b}+\mathbf{W}^\mathsf{T}\mathbf{v}\right)} = \prod_i e^{h_i\left(b_i + \sum_j W_{ji}v_j\right)} \tag{30.0.39}$$

Using the fact that $h_i \in \{0, 1\}$, we obtain the required results. (2)

$$p(\mathbf{v}|\mathbf{h}) = \prod_i p(v_i|\mathbf{h}), \qquad p(v_i|\mathbf{h}) = \sigma\left(a_i + \sum_j W_{ij}h_j\right) \tag{30.0.40}$$

(3) $p(\mathbf{h})$ is not factorised since summing over $\mathbf{v}$ introduces dependencies. (4) For a general $\mathbf{W}$, there is no known efficient way to compute Z. One way to argue this is that, after summing over $\mathbf{v}$, all the $\mathbf{h}$ components become coupled into one clique. This clique has no graphical structure that can be exploited to easily reduce the computation. Hence the computation will scale with $O\left(V2^H\right)$ (or $O\left(H2^V\right)$ if $H > V$).

4.5 From the first constraint we have

$$p(x, y|z, u) = p(x|zu)p(y|zu) \tag{30.0.41}$$

we can then write the joint as

$$p(x, y, z, u) = p(x|zu)p(y|zu)p(z, u) \tag{30.0.42}$$

The second statement says $p(z, u) = p(z)p(u)$. Hence the most general form of the distribution is

$$p(x, y, z, u) = p(x|zu)p(y|zu)p(z)p(u) \tag{30.0.43}$$

4.6 Using the obvious shorthand, we have

$$p_{12345} = \phi_{12}\phi_{23}\phi_{34}\phi_{45}\phi_{51} \tag{30.0.44}$$

By summing over 3, 4, we obtain

$$p_{125} = \phi_{12}\phi_{51}\sum_{34}\phi_{23}\phi_{34}\phi_{45} \tag{30.0.45}$$

Similarly, we obtain

$$p_{245} = \phi_{45}\sum_{13}\phi_{12}\phi_{23}\phi_{34}\phi_{51}, \qquad p_{234} = \phi_{23}\phi_{34}\sum_{15}\phi_{12}\phi_{45}\phi_{51} \tag{30.0.46}$$

Multiplying these together we obtain

$$p_{125}p_{245}p_{234} = \phi_{12}\phi_{51}\phi_{45}\phi_{23}\phi_{34}\left(\sum_{34}\phi_{23}\phi_{34}\phi_{45}\right)\left(\sum_{13}\phi_{12}\phi_{23}\phi_{34}\phi_{51}\right)\left(\sum_{15}\phi_{12}\phi_{45}\phi_{51}\right) \tag{30.0.47}$$

Also

$$p_{25} = \sum_{134}\phi_{12}\phi_{23}\phi_{34}\phi_{45}\phi_{51}, \qquad p_{24} = \sum_{135}\phi_{12}\phi_{23}\phi_{34}\phi_{45}\phi_{51} \tag{30.0.48}$$

Hence

$$\begin{aligned}
\frac{p_{125}p_{245}p_{234}}{p_{25}p_{24}} &= p\frac{\left(\sum_{34}\phi_{23}\phi_{34}\phi_{45}\right)\left(\sum_{13}\phi_{12}\phi_{23}\phi_{34}\phi_{51}\right)\left(\sum_{15}\phi_{12}\phi_{45}\phi_{51}\right)}{\left(\sum_{134}\phi_{12}\phi_{23}\phi_{34}\phi_{45}\phi_{51}\right)\left(\sum_{135}\phi_{12}\phi_{23}\phi_{34}\phi_{45}\phi_{51}\right)} && (30.0.49)\\
&= p\frac{\left(\sum_{34}\phi_{23}\phi_{34}\phi_{45}\right)\left(\sum_{13}\phi_{12}\phi_{23}\phi_{34}\phi_{51}\right)\left(\sum_{15}\phi_{12}\phi_{45}\phi_{51}\right)}{\left(\sum_{1}\phi_{12}\phi_{51}\sum_{34}\phi_{23}\phi_{34}\phi_{45}\right)\left(\sum_{3}\phi_{23}\phi_{34}\sum_{15}\phi_{12}\phi_{45}\phi_{51}\right)} && (30.0.50)\\
&= p\frac{\left(\sum_{34}\phi_{23}\phi_{34}\phi_{45}\right)\left(\sum_{1}\phi_{12}\phi_{51}\right)\left(\sum_{3}\phi_{23}\phi_{34}\right)\left(\sum_{15}\phi_{12}\phi_{45}\phi_{51}\right)}{\left(\sum_{1}\phi_{12}\phi_{51}\sum_{34}\phi_{23}\phi_{34}\phi_{45}\right)\left(\sum_{3}\phi_{23}\phi_{34}\sum_{15}\phi_{12}\phi_{45}\phi_{51}\right)} && (30.0.51)\\
&= p && (30.0.52)
\end{aligned}$$

The reason this works is more clearly seen when considering the junction tree representation. By triangulation, we can write the distribution using cliques on (1, 2, 5), (2, 4, 5) and (2, 3, 4), with (2, 4) and (2, 5) being their separators.

4.7 (1) The MN is obtained by summing over h_1, h_2, which fully couples all x_1, x_2, x_3. (2) This is not a perfect map since in $p(x_1, x_2, x_3)$ we have $x_1 \perp\!\!\!\perp x_3 | \emptyset$, which is violated by the MN representation.

4.8 (1) Not possible in general since if we are given $p(x|y)$ we only need to specify $p(y)$. In specifying $p(y|x)$ we have too many degrees of freedom. However, if the data x, y that both labs used to compute their estimates was the same and generated from the same joint distribution $p(x, y)$, one would hope that these conditional marginals are reasonably consistent. (2) Writing

$$p(x) = \sum_y p_A(x|y)p(y) = \sum_y p_A(x|y) \sum_x p_B(y|x)p(x) = \sum_x \underbrace{\sum_y p_A(x|y)p_B(y|x)}_{C} p(x) \tag{30.0.53}$$

we have an eigen-equation. The matrix C is by construction stochastic so is guaranteed to have an probability distribution with eigenvalue 1. The same is true for $p(y)$ with a stochastic matrix D. This defines $p(x)$ and $p(y)$.

4.9 (1) The fully factored distribution is consistent with all independence statements, so there is at least one trivial solution. (2) This is a difficult problem. One approach would be to assume a graphical model class, for example a Markov Network, and then search over the exponentially many graph structures, checking that each is consistent with all statements in both lists.

4.10 (1)

$$p(x|z) = \frac{\sum_{w,y} p(x, y, w, z)}{\sum_{x,w,y} p(x, y, w, z)} = \frac{\sum_{w,y} p(z|w)p(w|x, y)p(x)p(y)}{\sum_{x,w,y} p(z|w)p(w|x, y)p(x)p(y)} \tag{30.0.54}$$

(2)

$$p(y|z) = \frac{\sum_{w,x} p(x, y, w, z)}{\sum_{x,w,y} p(x, y, w, z)} = \frac{\sum_{w,x} p(z|w)p(w|x, y)p(x)p(y)}{\sum_{x,w,y} p(z|w)p(w|x, y)p(x)p(y)} \tag{30.0.55}$$

(3)

$$p(x, y|z) = \frac{\sum_{w} p(x, y, w, z)}{\sum_{x,w,y} p(x, y, w, z)} = \frac{\sum_{w} p(z|w)p(w|x, y)p(x)p(y)}{\sum_{x,w,y} p(z|w)p(w|x, y)p(x)p(y)} \tag{30.0.56}$$

The numerator does not split into a product of a function of x times a function of y and hence in general this is not product of $p(x|z)p(y|z)$ and x and y are dependent conditioned on z.

4.11 (2) No, since summing over h couples all t_1, t_2, y_1, y_2 together. (3) One can show the independence either by explicit summation and then using the definition of independence. Alternatively a simple observation that there is a collider on the path is sufficient.

4.12 (1) An undirected 4 cycle. (2) Distribution cannot be written as a belief network since a BN must have a collider (two arrows pointing in). We can then marginalise over this collider node and the remaining three variables will have a missing edge. Viewed as a Markov network, the graph of these three variables has a missing edge. If we take the original Markov network over 4 variables, and marginalise over one variable, the remaining three become fully connected, with no missing edges. (3) a and c are dependent since they are connected by b.

4.13 Simply choose any node on the undirected graph as the root of the DAG and then orient edges consistently away from this root.

4.14 We first note that $\mathbb{I}[s_i = 0, s_j = 0] = (1-s_i)(1-s_j)$,$\mathbb{I}[s_i = 0, s_j = 1] = (1-s_i)s_{)}$,$\mathbb{I}[s_i = 1, s_j = 0] = s_i(1-s_j)$, $\mathbb{I}[s_i = 1, s_j = 1] = s_i s_j$. Consider a term

$$\begin{aligned}\phi_{ij}(s_i, s_j) &= e^{\log \phi_{ij}(s_i,s_j)} \\ &= e^{\mathbb{I}[s_i=0,s_j=0] \log \phi_{ij}(0,0) + \mathbb{I}[s_i=0,s_j=1] \log \phi_{ij}(0,1) + \mathbb{I}[s_i=1,s_j=0] \log \phi_{ij}(1,0) + \mathbb{I}[s_i=1,s_j=1] \log \phi_{ij}(1,1)}\end{aligned}$$

Since each $\mathbb{I}[\cdot]$ term is a quadratic form in s, the above can be written as a Boltzmann machine by collecting all the quadratic terms $s_i s_j$ and linear terms s_i with suitable coefficients.

4.15 The file `marginalconsistency.m` contains a routine for this based on a simple linear solver. One could also call the `linprog.m` routine, but the routine is potentially more instructive.

1.

$$\underbrace{\begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & -1 & 0 & -1 \end{pmatrix}}_{\mathbf{M}} \underbrace{\begin{pmatrix} q_{12}(0,0) \\ q_{12}(0,1) \\ q_{12}(1,0) \\ q_{12}(1,1) \\ q_{13}(0,0) \\ q_{13}(0,1) \\ q_{13}(1,0) \\ q_{13}(1,1) \\ q_{23}(0,0) \\ q_{23}(0,1) \\ q_{23}(1,0) \\ q_{23}(1,1) \end{pmatrix}}_{\mathbf{y}} = \underbrace{\begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}}_{\mathbf{c}}$$

2.

$$\underbrace{\begin{pmatrix} 1&1&0&0&0&0&0&0\\ 0&0&1&1&0&0&0&0\\ 0&0&0&0&1&1&0&0\\ 0&0&0&0&0&0&1&1\\ 1&0&1&0&0&0&0&0\\ 0&1&0&1&0&0&0&0\\ 0&0&0&0&1&0&1&0\\ 0&0&0&0&0&1&0&1\\ 1&0&0&0&1&0&0&0\\ 0&1&0&0&0&1&0&0\\ 0&0&1&0&0&0&1&0\\ 0&0&0&1&0&0&0&1 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} p_{123}(0,0,0)\\ p_{123}(0,0,1)\\ p_{123}(0,1,0)\\ p_{123}(0,1,1)\\ p_{123}(1,0,0)\\ p_{123}(1,0,1)\\ p_{123}(1,1,0)\\ p_{123}(1,1,1) \end{pmatrix}}_{\mathbf{z}} = \mathbf{y}$$

5.1 By definition, a singly connected network must have an edge node that is connected to at most one neighbour. Let's call this x_e with neighbour x_n. Then there is a factor in the MN $\phi(x_e, x_n)$. We can then write

$$Z = \sum_{\mathcal{X}} \prod_{i\sim j} \phi(x_i, x_j) = \sum_{\mathcal{X}\backslash\mathcal{E}} \phi(rest) \underbrace{\sum_{x_e} \phi(x_e, x_n)}_{\psi(x_n)}$$

where $\phi(rest)$ represents the product of the potentials on the remaining graph, excluding $\phi(x_e, x_n)$. Since $\psi(x_n)$ is a function only of it x_n, we can merge $\psi(x_n)$ with another existing potential to form a new singly connected graph on $N-1$ variables (assuming there are N variables in the distribution). That is, we identify a variable m that is a neighbour of n and replace $\phi(x_n, x_m) \to \phi(x_n, x_m)\psi(x_n)$. The remaining graph has $N-1$ variables and is singly connected. We then recursively apply this rule, ending up with a pair of nodes at the end whose states we can simply sum over. This results in a time complexity that scales with N.

5.2 Naively, this would appear difficult since it is not singly connected so that when we sum over one of the variables, we introduce links to between the remaining variables. However, if we condition on one of the variables, say x_1, the graph is singly connected. This means that we can carry out two efficient maximisations, one for each state of x_1, and then choose that with the highest value. This is a form of loop-cut conditioning. More formally, we can write

$$\max x_1, \ldots, x_{100} p(x) = \max x_1 \underbrace{\max x_2, \ldots, x_{100} \phi(x_1, x_{100}) \prod_{i=1}^{99} \phi(x_i, x_{i+1})}_{f(x_1)}$$

For each state of x_1, $f(x_1)$ can be computed efficiently using message passing since the graph (conditioned on a state of x_1) is singly connected. By enumerating over the states of x_1, one may then find x_1^*. Given this one then sets x_1 to this optimal state, breaking the loop, and may carry out the maximisation in the resulting singly connected structure. Alternatively, if the two sets of messages (for binary variables) have been retained, one backtracks, knowing that x_1 is optimally in the state x_1^*.

5.3 1. 14, 33, 74, 77, 91, 97, 98, 99, 100. One may achieve this by taking $(\mathbf{M}+\mathbf{I})^{10}$, and listing which elements of this matrix have zero in their second column.

2. Yes. This can be verified by taking powers $(\mathbf{M}+\mathbf{I})^n$, noticing there is a zero component in this matrix until $n = 13$.

3. We first need to compute a transition matrix with elements

$$T_{ij} = \frac{M_{ij}}{\sum_i M_{ij}}$$

This is the stationary distribution $\mathbf{T}^\infty \mathbf{s}$ where $\mathbf{s}$ is the zero vector except for $s_1 = 1$. One can compute $\mathbf{M}^\infty$ using an eigendecomposition (or approximately by repeated multiplication):

$$\mathbf{T}^n = \mathbf{E}\boldsymbol{\lambda}^n \mathbf{E}^\mathsf{T}$$

For $n \to \infty$, only those eigenvalues $\lambda_{i,i}$ that are 1 will survive. For the rest λ_{jj}^n tends to 0 for $\lambda_{jj} < 1$. Then, taking the first element of the vector $\mathbf{T}^\infty \mathbf{s}$ gives the stationary probability for being in room 1, given that we started in room 1. This gives value 0.0080808080808. Note that there are in fact three eigenvectors with eigenvalue 1 in this case - that is there is no unique stationary distribution (there is no 'equilibrium' distribution. The stationary distribution we end up with depends on which room we begin in.

4.

$$\begin{aligned} p(\text{at least one in room 1}) &= 1 - p(\text{neither in room 1}) = 1 - p(\text{player 1 not in room 1})p(\text{player 2 not in room 1}) \\ &= 1 - (1 - p(\text{player 1 in room 1}))(1 - p(\text{player 2 in room 1})) = 1 - (1 - 0.0080808080808)^2 = 0.01609 \end{aligned}$$

5.4 1. This is given in fig(23.4).

2. See fig(23.5a).

3. Since $p(h_{1:T}|v_{1:T}) \propto p(h_{1:T}, v_{1:T})$, we may compute the 'marginals' $p(h_t, v_{1:T})$ in which all the observations are set. Once we've set $v_{1:T}$, then as a factor graph, we can consider

$$\psi_t(h_t) \equiv p(v_t|h_t), \qquad \phi_t(h_t, h_{t-1}) \equiv p(h_t|h_{t-1})$$

and

$$p(h_{1:T}, v_{1:T}) = \prod_t \psi_t(h_t)\phi(h_t, h_{t-1})$$

We can pass a factor-to-variable message:

$$\mu_{\psi_t \to h_t}(h_t) = \psi(h_t), \qquad \mu_{\phi_t \to h_t}(h_t) = \sum_{h_{t-1}} \phi(h_t, h_{t-1})\mu_{h_{t-1}\to\phi_t}(h_{t-1})$$

4. We can pass variable-to-factor messages:

$$\mu_{h_{t-1}\to\phi_t}(h_{t-1}) = \mu_{\psi_{t-1}\to h_{t-1}}(h_{t-1})\,\mu_{\phi_{t-1}\to h_{t-1}}(h_{t-1})$$

We can pass messages by starting from time 1, computing the ψ_1 to h_1 message, and then the h_1 to ϕ_1 factor message, then the ϕ_1 to h_2 message. We continue in this way until time T. One we are at the end, we then pass the reverse messages:

We can pass a factor-to-variable message:

$$\mu_{\phi_t \to h_{t-1}}(h_{t-1}) = \sum_{h_t} \phi(h_t, h_{t-1})\mu_{h_t \to \phi_t}(h_t)$$

We can pass variable-to-factor messages:

$$\mu_{h_t\to\phi_t}(h_t) = \mu_{\psi_t\to h_t}(h_t)\,\mu_{\phi_{t+1}\to h_t}(h_t)$$

Once we've passed these messages back from time T to 1, then the marginal is

$$p(h_t|v_{1:T}) \propto \mu_{\phi_t\to h_t}(h_t)\,\mu_{\psi_t\to h_t}(h_t)\,\mu_{\phi_{t+1}\to h_t}(h_t)$$

5. To compute $p(h_t, h_{t+1}, v_1, \ldots, v_T)$ we need to sum over all the variables in the distribution before h_t and after h_{t+1}. This will leave us with

$$p(h_t, h_{t+1}|v_1, \ldots, v_T) \propto \mu_{\phi_t\to h_t}(h_t)\,\mu_{\psi_t\to h_t}(h_t)\,\phi_{t+1}(h_t, h_{t+1})\mu_{\phi_{t+2}\to h_{t+1}}(h_{t+1})\,\mu_{\psi_{t+1}\to h_{t+!}}(h_{t+!})$$

5.5 Consider a simple linear chain $x_1, \ldots, x_n$. In this case the summation over the variables $x_{m+1}, \ldots, x_n$ is straightforward and will not affect the structure of the chain. This intuition then suggests that as long as the connection between the removed variables and the remaining variables is limited in a similar manner, the marginal distribution still retains a tree structure.

5.6 This is essentially answered in the book. The basic idea is that the unit eigenvector of $\mathbf{M}$ corresponds to the stationary distribution, and the i^{th} entry of this eigenvector is the corresponding probability of visiting page i by random surfing. There are some subtleties, however, since it might be that there is more than one unit eigenvalue and therefore no unique stationary eigenvector. Intuitively, this can happen when there are essentially isolated parts of the internet that cannot be reached from other islands – in this case it matters in which island one begins in as to what the stationary distribution will be. In order to prevent this, one can make an 'ergodic' Markov chain by adding a small amount ϵ to each entry of $\mathbf{M}$ and renormalising. In this way, we will be able to visit any node from any other node, and avoid having isolated islands. In this case the stationary distribution will be independent of where we start. The relevance of page i for a search engine can then be taken to be the i^{th} entry of the corresponding unique unit-eigenvector.

5.7 1. The distribution

$$\begin{aligned}
p(h_1, \ldots, h_T, x_1, \ldots, x_T) &= \sum_{y_1,\ldots,y_T} p(x_1, \ldots, x_T, y_1, \ldots, y_T, h_1, \ldots, h_T) \\
&= \sum_{y_1,\ldots,t_T} p(x_1|h_1)p(y_1|h_1)p(h_1)\prod_{t=2}^{T} p(h_t|h_{t-1})p(x_t|h_t)p(y_t|h_t) \\
&= \left(\prod_{t=1}^{T} \underbrace{\sum_{y_t} p(y_t|h_t)}_{1}\right) p(x_1|h_1)p(h_1)\prod_{t=2}^{T} p(h_t|h_{t-1})p(x_t|h_t) \\
&= p(x_1|h_1)p(h_1)\prod_{t=2}^{T} p(h_t|h_{t-1})p(x_t|h_t)
\end{aligned}$$

Hence $p(h_1, \ldots, h_T, x_1, \ldots, x_T)$ is a simple HMM. We can then compute the optimal state $h_1^*, \ldots, h_T^*$ by the max-product algorithm on the corresponding factor graph. Then

$$p(y_1, \ldots, y_T, h_1, \ldots, h_T) = \sum_{x_1,\ldots,x_T} p(h_1, \ldots, h_T, x_1, \ldots, x_T)$$

By symmetry with the above argument, we obtain

$$p(y_1, \ldots, y_T, h_1, \ldots, h_T) = p(y_1|h_1)p(h_1)\prod_{t=2}^{T} p(h_t|h_{t-1})p(y_t|h_t)$$

which is again a HMM. When $h_1, \ldots, h_T$ is clamped into a known state $h_1^*, \ldots, h_T^*$, then the above, as a function of $y_1, \ldots, y_T$ is simply a chain (when represented as a factor graph), for which the most likely assignment $y_1^*, \ldots, y_T^*$ can be computed using the max-product algorithm.

2. See `demoBanana.m`

```
The most likely y-sequence, based on the most likely h:
CTTGACTTGACTGACTGACTGACTGACCTGATTTTTTGACTGAGACTGACTGTTTTTTTTCTGACTGACTGACTGACTGACTGACTGACTGACTGA
```

```
function demoBanana
import brml.*
load banana

T=100; H=5;
ht=1:T;
xt=T+1:2*T;
yt=2*T+1:3*T;

xx(find(x=='A'))=1;
xx(find(x=='C'))=2;
xx(find(x=='G'))=3;
xx(find(x=='T'))=4;
x=xx;

for t=1:T
    varinf(xt(t)).name=''; varinf(xt(t)).domain={'a','c','g','t'};
    varinf(yt(t)).name=''; varinf(yt(t)).domain={'a','c','g','t'};
    varinf(ht(t)).name=''; varinf(ht(t)).domain=1:H;
end

for t=1:T
    pot{xt(t)}=array([xt(t) ht(t)],pxgh);
    pot{yt(t)}=array([yt(t) ht(t)],pygh);
end
pot{ht(1)}=array(ht(1),ph1);

for t=2:T
    pot{ht(t)}=array([ht(t) ht(t-1)],phtghtm);
end

for t=1:T
    newpot{t}=multpots([setpot(pot{xt(t)},xt(t),x(t)) pot{ht(t)}]);
end

% This computes the most likely joint h-state first, and then the most
% likely y's from that h-state.
A = FactorGraph(newpot); % variable nodes are first in A
[hmax maxval mess]=maxprodFG(newpot,A);
for t=1:T
    y(t)=argmax(pygh(:,hmax(t)));
end
yy(find(y==1))='A';
yy(find(y==2))='C';
yy(find(y==3))='G';
yy(find(y==4))='T';

fprintf(1,'most likely y-sequence, based on the most likely h:\n%s\n\n',yy)
```

3. This is not generally tractable since this is a form of mixed inference. Intuitively, when we sum over $h_1, \ldots, h_T$ we couple together all the $y_1, \ldots, y_T$ into one large clique, for which there is no structure left for a message passing approach.

4. By treating $y_1, \ldots, y_T$ as parameters, effectively we wish to find the most likely parameters. The EM algorithm would then use

$$q(h_1, \ldots, h_T) \propto p(x_1, \ldots, x_T, y_1^{old}, \ldots, y_T^{old}, h_1, \ldots, h_T)$$

which is just a HMM. The M-step is then

$$y_1^{new}, \ldots, y_T^{new} = \underset{y_1,\ldots,y_T}{\text{argmax}} \ \langle \log p(x_1, \ldots, x_T, y_1, \ldots, y_T, h_1, \ldots, h_T) \rangle_{q(h_1,\ldots,h_T)}$$

Since $p(x_1, \ldots, x_T, y_1, \ldots, y_T, h_1, \ldots, h_T)$ is a Markov chain, taking the logarithm means that we will have a sum of transitions such as

$$\langle \log p(x_1, \ldots, x_T, y_1, \ldots, y_T, h_1, \ldots, h_T) \rangle_{q(h_1,\ldots,h_T)} = \sum_t \underbrace{\langle \log p(x_t|h_t)p(y_t|h_t)p(h_t|h_{t-1}) \rangle_{q(h_t,h_{t-1})}}_{\psi(y_t)}$$

This is simply a sum of terms(given $x_1, \ldots, x_T$)

$$f(y_1, \ldots, y_T) \equiv \psi(y_1) + \psi(y_2) + \ldots \psi(y_T)$$

The maximum of this function is easy to compute:

$$y_t^* = \underset{y_t}{\text{argmax}}\ \psi(y_t)$$

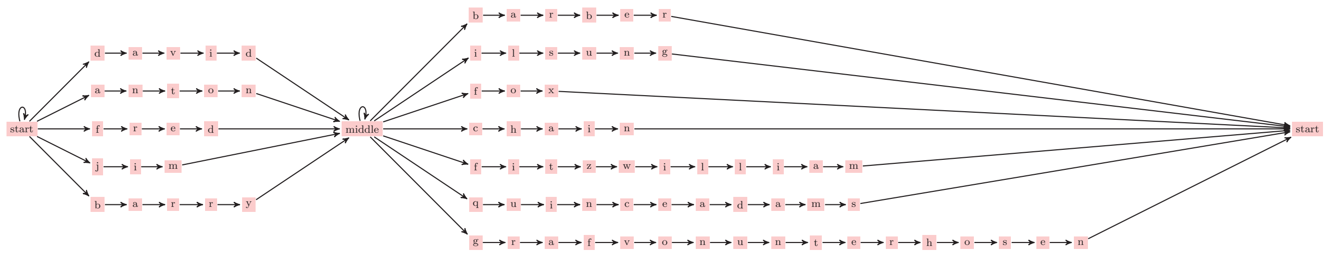
The EM algorithm will then converge to a (possibly) local optimum. One needs to run this several times to be reasonably confident that the most likely state is found - that with the corresponding highest likelihood under several different EM runs. See `demoBanana.m`.

```
EM for approximating the most likely state of p(y|x):
EM loop=1, log likelihood=-185.973942
most likely sequence:
ATTGACTTGACTGACTGACTGACTGACCTGATTTTTTGACTGAGACTGACTGTTTTTTTTATGACTGACTGACTGACTGACTGACTGACTGACTGA
EM loop=2, log likelihood=-184.913254
most likely sequence:
ATTGACTTGACTGACTGACTGACTGACCTGATTTTTTGACTGAGACTGACTGTTTTTTTTTTGACTGACTGACTGACTGACTGACTGACTGACTGA
EM loop=3, log likelihood=-184.479771
most likely sequence:
ATTGACTTGACTGACTGACTGACTGACCTGATTTTTTGACTGAGACTGACTGTTTTTTTTTTGACTGACTGACTGACTGACTGACTGACTGACTGA
EM loop=4, log likelihood=-184.479771
most likely sequence:
ATTGACTTGACTGACTGACTGACTGACCTGATTTTTTGACTGAGACTGACTGTTTTTTTTTTGACTGACTGACTGACTGACTGACTGACTGACTGA
EM loop=5, log likelihood=-184.479771
most likely sequence:
ATTGACTTGACTGACTGACTGACTGACCTGATTTTTTGACTGAGACTGACTGTTTTTTTTTTGACTGACTGACTGACTGACTGACTGACTGACTGA
```

See `demoBanana.m`:

```
% The following approximates the the most likely joint y given x (after
% summing over all h) using an EM approximation approach
disp('EM for approximating the most likely state of p(y|x):')
for t=1:T
    potx{t} = setpot(pot{xt(t)},xt(t),x(t));
end
maxstateEM=y;

st=[];
for emloop=1:5
    for t=1:T
        newpot{t}=multpots([potx{t} setpot(pot{yt(t)},yt(t),maxstateEM(t)) pot{ht(t)}]);
    end
    [marg mess normconstpot]=sumprodFG(newpot,A);
    for t=1:T
        tmppot  = multpots([logpot(pot{yt(t)}) marg{t}]);
        [dum maxstateEM(t)]=maxpot(sumpot(tmppot,ht(t)),yt(t));
    end

    yEM(find(maxstateEM==1))='A';
    yEM(find(maxstateEM==2))='C';
    yEM(find(maxstateEM==3))='G';
    yEM(find(maxstateEM==4))='T';

    fprintf(1,'EM loop=%d, log likelihood=%f\n',emloop,log(normconstpot.table));
    fprintf(1,'most likely sequence:\n%s\n',yEM)
end
```

5.8 This can be addressed by using a HMM in which we define a state for each character in all the firstnames and surnames, with an additional two states, giving the state-transition diagram as depicted.



The `start` and `middle` states self-transition with probability 0.8. The transitions into the the names are uniform and the transitions within and out of the names are deterministic. This defines a sparse 84×84 transition matrix A for the HMM. The 26×84 dimensional emission matrix has an entry 0.3 for emitting the correct letter, with a uniform probability on the other letters. The `start` and `middle` states emit a letter with uniform probability. This then defines the HMM, for which running Viterbi decoding on the $T = 10,000$ length sequence and computing the matrix m gives:

$$\begin{pmatrix} 6 & 3 & 7 & 12 & 19 & 10 & 11 \\ 8 & 7 & 7 & 6 & 16 & 10 & 13 \\ 8 & 7 & 8 & 13 & 11 & 7 & 19 \\ 14 & 12 & 12 & 16 & 21 & 12 & 18 \\ 11 & 6 & 10 & 12 & 5 & 10 & 13 \end{pmatrix}$$

See `solutionFirstnameSurname.m`.

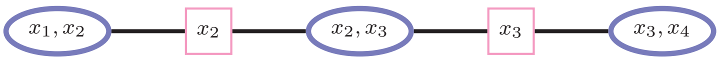
5.10 The expected price gain is 1.432298 with standard deviation 8.540987. See `solutionBearBull.m`. This can be achieved by filtering. If $\alpha(h_t)$ represents $p(h_t|v_{1:t})$, then

$$\alpha(h_t) \propto \sum_{h_{t-1}} p(h_t|h_{t-1})p(v_t|v_{t-1}, h_t)\alpha(h_{t-1}) \tag{30.0.57}$$

with initialisation $\alpha(h_1) = 0.5$. Prediction is then given by

$$p(v_{T+1}|v_{1:T}) \propto \sum_{h_{T+1}} p(h_{T+1}|h_T)\alpha(h_T)p(v_{T+1}|h_{T+1}) \tag{30.0.58}$$

```
function solutionBearBull
close all
import brml.*

[bear bull]=assign(1:2); % bear goes up, bull goes down

Tran(bear,bear)=0.8;
Tran(bull,bear)=0.2;
Tran(bull,bull)=0.7;
Tran(bear,bull)=0.3;

P=100; % number of price values
price=1:P; % price values
load BearBullproblem
T=length(p); % length of timeseries
subplot(2,1,1); plot(1:T,p,'-o'); title('price')

% dynamic programming for filtering:
[htm ht vtm vt]=assign(1:4);
tranpot=array([ht htm],Tran);
for h=1:2
    for ptm=price
        for pt=price
            if h==bull
                tmp(pt,ptm,h)=pbull(pt,ptm);
            else
                tmp(pt,ptm,h)=pbear(pt,ptm);
            end
        end
    end
end
empot=array([vt vtm ht],tmp);

f(:,1)=[0.5 0.5]';
filt=array(1,[0.5 0.5]);
for t=2:T
    filt=condpot(setpot(empot,[vtm vt],[p(t-1) p(t)])*tranpot*filt,ht); % filtered update
    f(:,t)=filt.table;
    filt.variables=1; % need to reset the filtered distribution variable number for the next timestep
end
subplot(2,1,2); plot(f(2,:),'-o'); title('p bull')

% prediction:
predh=condpot(filt*tranpot,ht); % latent state prediction
predv=condpot(setpot(empot,vtm,p(T))*predh,vt); % output state prediction

U=0; V=0;
for i=price
    U=U+predv.table(i)*(i-p(T)); % expected gain
    V=V+predv.table(i)*(i-p(T)).^2; % expected squared gain
end
fprintf(1,'Expected price gain = %f\n', U); % total expected gain
fprintf(1,'Expected price gain standard deviation = %f\n', sqrt(V-U^2));
```

6.2 1. 

2.

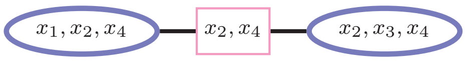
$$p(x_1, x_2, x_3, x_4) = \frac{p(x_1, x_2)p(x_2, x_3)p(x_3, x_4)}{p(x_2)p(x_3)}$$

For interest:

$$\begin{aligned}
\phi^*(x_2) &= \sum_{x_1}\phi(x_1,x_2)\\
\phi^*(x_2,x_3) &= \phi(x_2,x_3)\phi^*(x_2) = \phi(x_2,x_3)\sum_{x_1}\phi(x_1,x_2)\\
\phi^*(x_3) &= \sum_{x_2}\phi^*(x_2,x_3) = \sum_{x_2}\phi(x_2,x_3)\sum_{x_1}\phi(x_1,x_2)\\
\phi^*(x_3,x_4) &= \phi(x_3,x_4)\phi^*(x_3) = \phi(x_3,x_4)\sum_{x_2}\phi(x_2,x_3)\sum_{x_1}\phi(x_1,x_2) = \sum_{x_1,x_2} p(x_1,x_2,x_3,x_4) = p(x_3,x_4)
\end{aligned}$$

Going back from right to left we have

$$\begin{aligned}
\phi^{**}(x_3) &= \sum_{x_4}\phi^*(x_3,x_4) = p(x_3)\\
\phi^{**}(x_2,x_3) &= \phi^*(x_2,x_3)\frac{\phi^{**}(x_3)}{\phi^*(x_3)} = \frac{\phi(x_2,x_3)\sum_{x_1}\phi(x_1,x_2)\sum_{x_4}\phi(x_3,x_4)\sum_{x_2}\phi(x_2,x_3)\sum_{x_1}\phi(x_1,x_2)}{\sum_{x_2}\phi(x_2,x_3)\sum_{x_1}\phi(x_1,x_2)}\\
&= \sum_{x_1,x_4}\phi(x_1,x_2)\phi(x_2,x_3)\phi(x_3,x_4) = \sum_{x_1,x_4} p(x_1,x_2,x_3,x_4) = p(x_2,x_3)\\
\phi^{**}(x_2) &= \sum_{x_3}\phi^{**}(x_2,x_3) = p(x_2)\\
\phi^{**}(x_1,x_2) &= \frac{\phi(x_1,x_2)\phi^{**}(x_2)}{\phi^*(x_2)} = \frac{\phi(x_1,x_2)\sum_{x_3}\sum_{x_1,x_4}\phi(x_1,x_2)\phi(x_2,x_3)\phi(x_3,x_4)}{\sum_{x_1}\phi(x_1,x_2)}\\
&= \sum_{x_3,x_4}\phi(x_1,x_2)\phi(x_2,x_3)\phi(x_3,x_4) = \sum_{x_3,x_4} p(x_1,x_2,x_3,x_4) = p(x_1,x_2)
\end{aligned}$$

6.3 1. 

2. Assign $\phi(x_1,x_2,x_4) = \phi(x_1,x_2)\phi(x_1,x_4)$, $\phi(x_2,x_3,x_4) = \phi(x_2,x_3)\phi(x_3,x_4)$. We'll go from left to right with initial separators set to unity:

$$\begin{aligned}
\phi^*(x_2,x_4) &= \sum_{x_1}\phi(x_1,x_2,x_4) = \sum_{x_1}\phi(x_1,x_2)\phi(x_1,x_4)\\
\phi^*(x_2,x_3,x_4) &= \phi(x_2,x_3,x_4)\phi^*(x_2,x_4) = \phi(x_2,x_3)\phi(x_3,x_4)\sum_{x_1}\phi(x_1,x_2)\phi(x_1,x_4)\\
&= \sum_{x_1} p(x_1,x_2,x_3,x_4) = p(x_2,x_3,x_4)
\end{aligned}$$

Going back from right to left we have

$$\begin{aligned}
\phi^{**}(x_2,x_4) &= \sum_{x_3}\phi^*(x_2,x_3,x_4) = \sum_{x_3} p(x_2,x_3,x_4) = p(x_2,x_4)\\
\phi^*(x_1,x_2,x_4) &= \frac{\phi(x_1,x_2,x_4)\phi^{**}(x_2,x_4)}{\phi^*(x_2,x_4)}\\
&= \frac{\phi(x_1,x_2,x_4)\sum_{x_3}\phi(x_2,x_3)\phi(x_3,x_4)\sum_{x_1}\phi(x_1,x_2)\phi(x_1,x_4)}{\sum_{x_1}\phi(x_1,x_2)\phi(x_1,x_4)}\\
&= \sum_{x_3}\phi(x_1,x_2)\phi(x_2,x_3)\phi(x_3,x_4)\phi(x_1,x_4)\\
&= \sum_{x_3} p(x_1,x_2,x_3,x_4) = p(x_1,x_2,x_3,x_4) = p(x_1,x_2,x_4)
\end{aligned}$$

We can then find $p(x_1) = \sum_{x_2,x_4}\phi^{**}(x_1,x_2,x_4) = \sum_{x_2,x_4} p(x_1,x_2,x_4) = p(x_1)$.

6.4 1. To be done.

2. To be done.

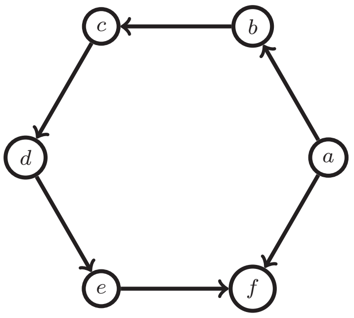
3. No unique solution. A triangulated graph contains cliques heb, hba, $hacg$, adg, $hcfg$, fgi. A clique of size 4 is the minimal largest clique available in this case.

4. To be done.

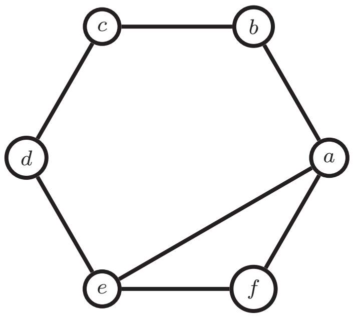
5. To be done.

6. To be done.

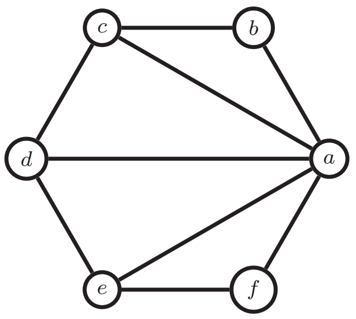
6.5 1. A directed hexagon with incoming links to f, with a as the root.

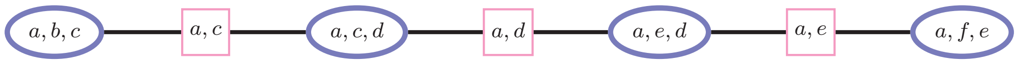


2. Extra link $a - e$.



3. Several equivalent answers. One is to radiate all links from a.



4. 

5. Several possible answers, but just distribute probability factors in one of the cliques in the junction tree, *e.g.*

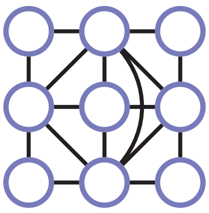
$$\phi(a,b,c) = p(a)p(b|a)p(c|a)$$
$$\phi(a,c,d) = p(d|c)$$
$$\phi(a,e,d) = p(e|d)$$
$$\phi(a,f,e) = p(f|a,e)$$

Set the separator potentials to 1.

6. A valid schedule is $abc \to acd \to aed \to afe \to aed \to acd \to abc$.

7. This is clear from the fact that this is a junction tree for the original distribution. One can also show this directly by repeated use of Bayes rule and the conditional independencies encoded in the distribution:

$$\begin{aligned} p(a|f)p(b|a,c)p(c|a,d)p(d|a,e)p(e|a,f)p(f) &= \frac{p(a,f)p(b,a,c)p(c,a,d)p(d,a,e)p(e,a,f)p(f)}{p(f)p(a,c)p(a,d)p(a,e)p(a,f)} \\ &= \frac{p(b,a,c)p(c,a,d)p(d,a,e)p(f|a,e)}{p(a,c)p(a,d)} \\ &= p(b,a,c)p(d|a,c)p(e|d,a)p(f|a,e) \\ &= p(a,b)p(c|b,a)p(d|c)p(e|d)p(f|a,e) \\ &= p(a)p(b|a)p(c|b)p(d|c)p(e|d)p(f|a,e) \end{aligned}$$

6.6 Different possible answers. One is 

6.7 Complexity is $O\left(n2^{2n}\right)$. For $n = 10$, $\log Z = 186.7916$. See `IsingZ.m`:

```
function IsingZ
import brml.*
N=10;

% form the potentials for each pair of neighbouring variables:
for x=1:2
    for y=1:2
        isingtable(x,y)=exp(1*(x==y));
    end
end
S = reshape(1:N*N,N,N);
c=0;
for s1=1:N*N
    [i1 j1]=find(S==s1);
    for s2=s1+1:N*N
        [i2 j2]=find(S==s2);
        if (j1==j2)&(abs(i1-i2)==1) | (i1==i2)&(abs(j1-j2)==1)
            c=c+1;
            phi{c}=array([s1 s2],isingtable);
            tab(i1,i2,j1,j2)=c;
            tab(i2,i1,j2,j1)=c;
        end
    end
end

% form column potentials:
colphi{1}=const(1);
for i=1:N-1
    colphi{i}=const(1);
    for j=1:N-1
        colphi{i}=multpots([colphi{i} phi{tab(j,j,i,i+1)}]);
        colphi{i}=multpots([colphi{i} phi{tab(j,j+1,i,i)}]);
    end
    colphi{i}=multpots([colphi{i} phi{tab(N,N,i,i+1)}]);
end
for j=1:N-1
    colphi{i}=multpots([colphi{i} phi{tab(j,j+1,i+1,i+1)}]);
end

% do message passing in this new chain with the column potentials:
mmess=[];
sumover=setdiff(colphi{1}.variables,intersect(colphi{1}.variables,colphi{2}.variables));
mess=sumpot(colphi{1},sumover);
for i=2:N-2
    sumover=setdiff(colphi{i}.variables,intersect(colphi{i}.variables,colphi{i+1}.variables));
    mess=sumpot(multpots([mess colphi{i}]),sumover);
    mmess=[mmess max(mess.table(:))];
    mess.table=mess.table./mmess(end); % tables may overflow so remove constant factor
end
logZ = log(table(sumpot(multpots([mess colphi{N-1}]),[],0)));
logZ = logZ+ sum(log(mmess)) % add back any removed constant factor
```

6.8 This is given by the reabsorption procedure, as described in the text.

6.9
```
function diseaseNet
import brml.*
D=20;S=40; dv=1:D; sv=D+1:D+S;
load diseaseNet
pot=str2cell(setpotclass(pot,'array'));
drawNet(dag(pot),variable);

tstart=tic;
[jtpot jtsep infostruct]=jtree(pot); % setup the Junction Tree
jtpot=absorption(jtpot,jtsep,infostruct); % do full round of absorption
t=toc(tstart);
disp(['Junction Tree takes ',num2str(t),' seconds to form and perform absoprtion'])

for i=1:length(jtpot); label{i}=horzcat(variable(jtpot{i}.variables).name); sz(i)=length(jtpot{i}.variables); end
figure; draw_layout(infostruct.cliquetree,label);
drawnow
disp(['Maximum JT clique size is ',num2str(max(sz))])

for s=1:S
    jtpotnum = whichpot(jtpot,sv(s),1); % find a single JT potential that contains the sympton
    margpot=sumpot(jtpot{jtpotnum},sv(s),0);
    ps_jt(s)=margpot.table(1);
end

% More efficent summation based on needing only the parents of each sympton:
```

```
tstart=tic;
A = dag(pot);
for s=1:S
    margpot2 = sumpot(multpots([pot(sv(s)) pot(parents(A,sv(s)))]),sv(s),0);
    ps_eff(s)=margpot2.table(1);
end
t=toc(tstart);
disp(['Efficent inference takes ',num2str(t),' seconds'])

disp('JT marginals and Efficient Inference marginals are the same:')
[ps_jt' ps_eff']

[jtpot jtsep]=jtassignpot(setpot(pot,[sv(1:5) sv(6:10)],[1 1 1 1 1 2 2 2 2 2]),infostruct);
jtpot=absorption(jtpot,jtsep,infostruct); % do full round of absorption
for d=1:D
    jtpotnum = whichpot(jtpot,d,1); % find a single JT potential that contains the disease
    margpot=condpot(sumpot(jtpot(jtpotnum),d,0));
    pd_jt(d)=margpot.table(1);
end
disp('The marginals p(disease=1|evidence):')
```

1. See `diseaseNet.m`.

```
p(s(1)=1): Junction tree 0.441834 : efficient inference: 0.441834
p(s(2)=1): Junction tree 0.456675 : efficient inference: 0.456675
p(s(3)=1): Junction tree 0.441405 : efficient inference: 0.441405
p(s(4)=1): Junction tree 0.491275 : efficient inference: 0.491275
p(s(5)=1): Junction tree 0.493892 : efficient inference: 0.493892
p(s(6)=1): Junction tree 0.657483 : efficient inference: 0.657483
p(s(7)=1): Junction tree 0.504563 : efficient inference: 0.504563
p(s(8)=1): Junction tree 0.268693 : efficient inference: 0.268693
p(s(9)=1): Junction tree 0.649077 : efficient inference: 0.649077
p(s(10)=1): Junction tree 0.49074 : efficient inference: 0.49074
p(s(11)=1): Junction tree 0.422552 : efficient inference: 0.422552
p(s(12)=1): Junction tree 0.429096 : efficient inference: 0.429096
p(s(13)=1): Junction tree 0.545021 : efficient inference: 0.545021
p(s(14)=1): Junction tree 0.632959 : efficient inference: 0.632959
p(s(15)=1): Junction tree 0.42954 : efficient inference: 0.42954
p(s(16)=1): Junction tree 0.458794 : efficient inference: 0.458794
p(s(17)=1): Junction tree 0.427559 : efficient inference: 0.427559
p(s(18)=1): Junction tree 0.404255 : efficient inference: 0.404255
p(s(19)=1): Junction tree 0.582093 : efficient inference: 0.582093
p(s(20)=1): Junction tree 0.589591 : efficient inference: 0.589591
p(s(21)=1): Junction tree 0.76127 : efficient inference: 0.76127
p(s(22)=1): Junction tree 0.695588 : efficient inference: 0.695588
p(s(23)=1): Junction tree 0.508702 : efficient inference: 0.508702
p(s(24)=1): Junction tree 0.419962 : efficient inference: 0.419962
p(s(25)=1): Junction tree 0.351942 : efficient inference: 0.351942
p(s(26)=1): Junction tree 0.389611 : efficient inference: 0.389611
p(s(27)=1): Junction tree 0.325973 : efficient inference: 0.325973
p(s(28)=1): Junction tree 0.469624 : efficient inference: 0.469624
p(s(29)=1): Junction tree 0.522868 : efficient inference: 0.522868
p(s(30)=1): Junction tree 0.717312 : efficient inference: 0.717312
p(s(31)=1): Junction tree 0.524199 : efficient inference: 0.524199
p(s(32)=1): Junction tree 0.353704 : efficient inference: 0.353704
p(s(33)=1): Junction tree 0.512679 : efficient inference: 0.512679
p(s(34)=1): Junction tree 0.529404 : efficient inference: 0.529404
p(s(35)=1): Junction tree 0.385751 : efficient inference: 0.385751
p(s(36)=1): Junction tree 0.489095 : efficient inference: 0.489095
p(s(37)=1): Junction tree 0.633595 : efficient inference: 0.633595
p(s(38)=1): Junction tree 0.589604 : efficient inference: 0.589604
p(s(39)=1): Junction tree 0.423164 : efficient inference: 0.423164
p(s(40)=1): Junction tree 0.528234 : efficient inference: 0.528234
```

2. The point is that in computing the marginals $p(s_i)$, we can do this efficiently using

$$p(s_i) = \sum_{\mathrm{pa}(s_i)} p(s_i|\mathrm{pa}(s_i))p(\mathrm{pa}(s_i))$$

where the parent diseases $\mathrm{pa}(s_i)$ are independent. Since there are only 3 parental diseases, we can compute these marginals by $2^3 = 8$ summations. In contrast, the Junction Tree has cliques of size 12, meaning that the complexity of carrying out absorption for the JT is at least 2^{12}, which is much more expensive. The point therefore is that the JT doesn't know that there is structure that can be exploited in the tables, and is therefore an upper bound on the complexity of inference.

3.
```
The marginals p(disease=1|evidence):
p(d(1)=1|s(1:10))=0.0297756
p(d(2)=1|s(1:10))=0.38176
p(d(3)=1|s(1:10))=0.954235
```

```
p(d(4)=1|s(1:10))=0.396644
p(d(5)=1|s(1:10))=0.496467
p(d(6)=1|s(1:10))=0.435154
p(d(7)=1|s(1:10))=0.187487
p(d(8)=1|s(1:10))=0.701183
p(d(9)=1|s(1:10))=0.0431266
p(d(10)=1|s(1:10))=0.610313
p(d(11)=1|s(1:10))=0.287322
p(d(12)=1|s(1:10))=0.489833
p(d(13)=1|s(1:10))=0.8996
p(d(14)=1|s(1:10))=0.619565
p(d(15)=1|s(1:10))=0.920476
p(d(16)=1|s(1:10))=0.706096
p(d(17)=1|s(1:10))=0.201247
p(d(18)=1|s(1:10))=0.908494
p(d(19)=1|s(1:10))=0.864967
p(d(20)=1|s(1:10))=0.883929
```

6.10 1. The JT consists of a single clique on all the variables (due to marrying all the $x_1, \dots, x_T$). The complexity of inference from the JT would suggest that this is $O\left(2^T\right)$.

2. If we sum first over y, the remaining distribution on $x_1, \dots, x_T$ is simply a linear chain, for which inference of $p(x_T)$ is $O(T)$ time.

6.11 See `ShaferShenoy.m` and `demoJTreeShaferShenoy.m`.

```
function [jtmargpot jtmess]=ShaferShenoy(jtpot,infostruct)
%SHAFERSHENOY Perform full round of Shafer Shenoy messages on a Junction Tree
% [jtmargpot jtmess]=ShaferShenoy(jtpot,infostruct)
%
% Perform (sum) messages for a JT specified with potentials jtpot. The distribution marginals are contained in
% the returned jtmargpot.
%
% infostruct is a structure with information about the Junction Tree:
% infostruct.cliquetree : connectivity structure of Junction Tree cliques
% infostruct.EliminationSchedule is a list of clique to clique
% eliminations. The root clique is contained in the last row.
% infostruct.ForwardOnly is optional. If this is set to 1 only the schedule contained
% in infostruct.EliminationSchedule is carried out. Otherwise both a
% forward and backward schedule is carried out.
Atree=infostruct.cliquetree; % connectivity structure of JT cliques
import brml.*
schedule=infostruct.EliminationSchedule;
ForwardOnly=0;
if isfield(infostruct,'ForwardOnly')
    ForwardOnly=infostruct.ForwardOnly;
end
if ~ForwardOnly
    reverseschedule=flipud(fliplr(schedule));
    schedule=vertcat(schedule,reverseschedule); % full round over all separators in both directions
end
for count=1:length(schedule)
    [elim neighb]=assign(schedule(count,:));
    if elim~=neighb
        incoming=setdiff(neighb,neigh(Atree,elim));
        if isempty(incoming)
            jtmess{elim,neighb} = sumpot(jtpot{neighb}, jtpot{neighb}.variables,0);
        else
            jtmess{elim,neighb} = sumpot(multpots([jtmess(incoming) jtpot{neighb}]), jtpot{neighb}.variables,0);
        end
    end
end
for c=1:length(jtpot)
    jtmargpot{c}=multpots(jtmess(neigh(infostruct.cliquetree,c),c));
end
```

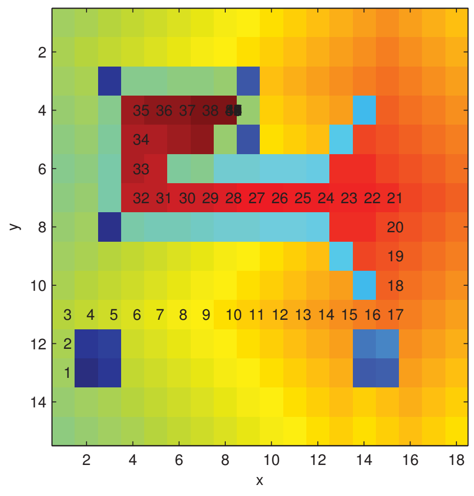
7.1 The expected gain is

$$pS - (1-p)S = (2p-1)S$$

7.2 The probability that a salary s is larger than a random salary is given by the cumulative sum function since : $\int \mathbb{I}[s > s']\, p(s')ds' = \int_{-\infty}^{s} p(s')ds'$. This gives rise to a 'sigmoidal' (s-shaped) curve. Repeating the experiment, provided we have a high salary, we would have an expected utility much lower than our salary and would therefore be not advised to take the bet. One can compute the salary at which one would be advised to take the bet and this will depend heavily on the shape of the salary profile. However, in general, the salary at which one would be advised to take the bet would be rather low typically. Intuitively, since there are only a few people with very high salaries, and human status intuition is typically about comparing ourselves to each others, we would only be advised to take the bet if we would expect to be better off than a randomly chosen person. If our existing salary is high, we are already better off than nearly everyone, so we wouldn't therefore be advised to take the bet. See also `demoMoneySalary.m`.

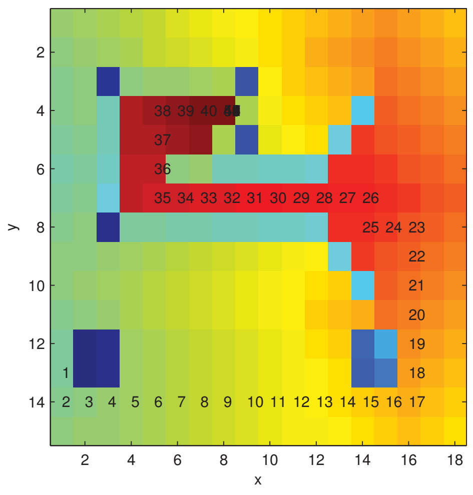
7.3 $test < \{\} < drill < \{Seismic, Oil\}$

7.4



```
Optimal (x,y) position is:

     1    13
     1    12
     1    11
     2    11
     3    11
     4    11
     5    11
     6    11
     7    11
     8    11
     9    11
    10    11
    11    11
    12    11
    13    11
    14    11
    15    11
    15    10
    15     9
    15     8
    15     7
    14     7
    13     7
    12     7
    11     7
    10     7
     9     7
     8     7
     7     7
     6     7
     5     7
     4     7
     4     6
     4     5
     4     4
     5     4
     6     4
     7     4
```

See `demoMDPairplane.m`



```
Optimal (x,y) position is:
     1    13
     1    14
     2    14
     3    14
     4    14
     5    14
     6    14
     7    14
     8    14
     9    14
    10    14
    11    14
    12    14
    13    14
    14    14
    15    14
    16    14
    16    13
    16    12
    16    11
    16    10
    16     9
    16     8
    15     8
    14     8
    14     7
    13     7
    12     7
    11     7
    10     7
     9     7
     8     7
     7     7
     6     7
     5     7
     5     6
     5     5
     5     4
     6     4
     7     4
     8     4
```

7.5 $U(d_1 = \textsf{no play}) = 0$. Utility of playing is the probability that we win p_1 times the amount we would win W_1. There is a cost of C_1 to play.

7.6

1. Replicate the same structure from the first stage to the second stage. Then include a link from x_1 to x_2 and from x_1 to d_2.

2. We need to compute the expected utility of the first decision not knowing whether or not we will win either of the first or both stages. If we win the first stage we assume that we will take the optimal decision d_2 to maximise the expected return. This gives (following the course notes):

$$U(d_1) = \sum_{w_1} \mathbb{I}\left[w_1 = \textsf{win}\right] p(w_1|d_1) \max_{d_2} \left(\sum_{w_2} p(w_2|w_1, d_2) u(w_2) - c(d_2) \right) + \sum_{w_1} p(w_1|d_1) u_1(w_1) - c(d_1) \tag{30.0.59}$$

The term $\sum_{w_1} p(w_1|d_1)u_1(w_1) - c(d_1)$ is just the first stage which for $d_1 = \textsf{play}$ contributes $p_1 W_1 - C_1$. The second stage contribution is, assuming we win the first stage:

$$p_1 \max_{d_2} \left(\sum_{w_2} p(w_2|w_1, d_2) u(w_2) - c(d_2) \right) \tag{30.0.60}$$

If $d_2 = \textsf{play}$ this is

$$p_1 \left(p_2 W_2 - C_2\right) \tag{30.0.61}$$

else if $d_2 = \textsf{no play}$ this is 0. Hence provided $p_2 W_2 - C_2 \geq 0$ we have $p_1 \left(p_2 W_2 - C_2\right)$, otherwise 0. Using these two conditions and adding on the first stage expected utility, we arrive at the answer.

7.7 1.

$$U(\mathsf{bet}) = p_w W + (1-p_w)L, \qquad U(\mathsf{no\ bet}) = B \tag{30.0.62}$$

2.

$$U_{gain}(bet) = p_w(W-B) - (1-p_w)(B-L) = p_w W - (B - L + p_w L) = p_w W - B + L - p_w L \tag{30.0.63}$$

$$U_{gain}(nobet) = 0 \tag{30.0.64}$$

3.

$$U_{gain}(bet) - U_{gain}(nobet) = p_w W - B + L - p_w L \tag{30.0.65}$$

$$U(\mathsf{bet}) - U(\mathsf{no\ bet}) = p_w W + (1-p_w)L - B \tag{30.0.66}$$

which gives the required result.

7.8 EM fails since the energy in the M-step is negative infinity until $\theta_{new} = \theta_{old}$ in which case the energy is zero.

7.10 See `IDjensen.m`.

7.11 In a POMDP, we need to sum first over all the unobserved $h_{1:T}$. This creates a single large clique that contains all the other variables. From this perspective, the complexity of exact inference in the POMDP is exponential in T.

7.12 $(abc)- > (bcde)- > (bdei) < -(degi) < -(dgh) < -(fdh)$

7.13 See `exerciseInvest.m` and `optdec.m`. See also `demoInvest.m`.

8.1 The Professor is wrong (in principle). Consider a highly non-symmetric distribution with many (slightly) above average values and a few extremely low values. A simple demonstration is to assume that the average IQ is 100, and the minimum 0. If there are only two possible scores, the above average score, and the below average score, then one can show that 90 percent of people can indeed have an above average IQ if the above average IQ score is less than 111.111.

8.2 The distribution is symmetric around $(0,0)$, so the mean is therefore 0. Furthermore, x and y are uncorrelated since the distribution is spherically symmetric (isotropic), so that $\langle xy \rangle = 0$. They are nevertheless dependent since

$$p(x|y) \propto (x^2+y^2)^2 e^{-x^2}$$

so that the distribution of x depends on y^2 (and vice versa).

8.3 If we write the states as the real values $0, 1$, then the question is equivalent to the distribution of the new variable

$$y = \sum_{i=1}^{n} x_i \tag{30.0.67}$$

Note that we can write $p(x_i) = \theta^{x_i}(1-\theta)^{1-x_i}$ Formally,

$$\begin{aligned} p(y=k) &= \sum_x p(y=k|x)p(x) = \sum_x \delta\left(k - \sum_{i=1}^n x_i\right) \prod_i p(x_i) \\ &= \sum_x \delta\left(k - \sum_{i=1}^n x_j\right) \prod_i \theta^{x_i}(1-\theta)^{1-x_i} \\ &= \sum_x \delta\left(k - \sum_{j=1}^n x_j\right) \theta^{\sum_i x_i}(1-\theta)^{n-\sum_i x_i} \\ &= \binom{n}{k}\theta^k (1-\theta)^{n-k} \end{aligned}$$

The factor $\binom{n}{k}$ arises since we are summing over all the configurations in which k of the n variables are in state 1.

8.4

$$I^2 = \int_{-\infty}^{\infty} e^{-\frac{1}{2}x^2+y^2} dxdy \tag{30.0.68}$$

Using Polar coordinates, $r^2 = x^2+y^2, \theta = \arccos(x/r)$, the Jacobian is $rdrd\theta$. Hence

$$I^2 = \int_{\theta=0}^{2\pi}\int_{r=0}^{\infty} e^{-\frac{1}{2}r^2} rdrd\theta = -2\pi e^{-\frac{1}{2}r^2}|_0^\infty = 2\pi \tag{30.0.69}$$

For part (ii), just use the change of variables $z = \frac{x-\mu}{\sigma}$.

8.5 1.

$$
\begin{aligned}
\langle x\rangle_{\mathcal{N}(x|\mu,\sigma^2)} &= \frac{1}{\sqrt{2\pi\sigma^2}}\int x e^{-\frac{1}{2\sigma^2}(x-\mu)^2}dx \\
&= \frac{1}{\sqrt{2\pi\sigma^2}}\int (x-\mu+\mu) e^{-\frac{1}{2\sigma^2}(x-\mu)^2}dx \\
&= \frac{1}{\sqrt{2\pi\sigma^2}}\int (x-\mu) e^{-\frac{1}{2\sigma^2}(x-\mu)^2}dx + \mu\frac{1}{\sqrt{2\pi\sigma^2}}\int e^{-\frac{1}{2\sigma^2}(x-\mu)^2}dx \\
&= \mu\frac{1}{\sqrt{2\pi\sigma^2}}\int e^{-\frac{1}{2\sigma^2}(x-\mu)^2}dx = \mu
\end{aligned}
$$

2.

$$
\left\langle (x-\mu)^2\right\rangle_{\mathcal{N}(x|\mu,\sigma^2)} = \frac{1}{\sqrt{2\pi\sigma^2}}\int (x-\mu)^2 e^{-\frac{1}{2\sigma^2}(x-\mu)^2}dx
$$

Changing variables, to $z \equiv (x-\mu)/\sigma$, we have

$$
\begin{aligned}
\left\langle (x-\mu)^2\right\rangle_{\mathcal{N}(x|\mu,\sigma^2)} &= \frac{\sigma^2}{\sqrt{2\pi\sigma^2}}\int z^2 e^{-\frac{1}{2}z^2}\sigma dz \\
&= \sigma^2\frac{1}{\sqrt{2\pi}}\int z^2 e^{-\frac{1}{2}z^2}dz
\end{aligned}
$$

The latter integral can be readily shown to be 1 by integration by parts.

8.6 Follows from

$$
\mathbf{\Sigma} = \left\langle \mathbf{x}\mathbf{x}^{\mathsf{T}}\right\rangle - \boldsymbol{\mu}\boldsymbol{\mu}^{\mathsf{T}}
$$

8.7

$$
\begin{aligned}
\int_{\theta_j}\text{Dirichlet}(\theta|\mathbf{u}) &= \frac{1}{Z(\mathbf{u})}\int_{\theta_j}\prod_i \theta_i^{u_i-1} \\
&= \frac{1}{Z(\mathbf{u})}\left[\prod_{i\neq j}\theta_i^{u_i-1}\right]\int_{\theta_j}\theta_j^{u_j-1} \\
&= \frac{1}{Z(\mathbf{u})}Z(u_j)\prod_{i\neq j}\theta_i^{u_i-1} \\
&= \text{Dirichlet}\left(\theta_{\setminus j}|\mathbf{u}_{\setminus j}\right)
\end{aligned}
$$

8.8

$$
\left\langle x^k\right\rangle = \frac{1}{B(\alpha,\beta)}\int x^k x^{\alpha-1}(1-x)^{\beta-1} = \frac{B(\alpha+k,\beta)}{B(\alpha,\beta)}
$$

Since

$$
B(\alpha+k,\beta) = \frac{\Gamma(\alpha+k)\Gamma(\beta)}{\Gamma(\alpha+k+\beta)} = \frac{(\alpha+k)\Gamma(\alpha+k-1)\Gamma(\beta)}{(\alpha+\beta+k)\Gamma(\alpha+k-1+\beta)} = \frac{(\alpha+k)(\alpha+k-1)\ldots(\alpha)\Gamma(\alpha)\Gamma(\beta)}{(\alpha+\beta+k)(\alpha+\beta+k-1)\ldots(\alpha+\beta)\Gamma(\alpha+\beta)}
$$

$$
\left\langle x^k\right\rangle = \frac{(\alpha+k)(\alpha+k-1)\ldots(\alpha)}{(\alpha+\beta+k)(\alpha+\beta+k-1)\ldots(\alpha+\beta)}
$$

8.9

$$
\frac{d}{dt}g(t) = \int p(x)\frac{de^{tx}}{dt} = \int p(x)xe^{tx}
$$

Similarly,

$$
\frac{d^k}{dt^k}g(t) = \int p(x)x^k e^{tx}
$$

Taking then the limit $t \to 0$, we obtain the required result.

8.10 Using Bayes' rule as

$$
\begin{aligned}
p(y) &= \int p(y|x)p(x)dx \\
&= \int \delta\left(y-f(x)\right)p(x)dx \\
&= \int \delta\left(y-f(x)\right)p(x)\frac{dx}{df}df \\
&= \int \delta\left(y-f(x)\right)p(x)\frac{dx}{df}df \\
&= p(x)\left(\frac{df}{dx}\right)^{-1}, \quad x = f^{-1}(y)
\end{aligned}
$$

where $f^{-1}(f(x)) = x$.

8.11 The Jacobian is $\det\left(\mathbf{\Sigma}^{-\frac{1}{2}}\right)$. Hence

$$I = \det\left(\mathbf{\Sigma}^{\frac{1}{2}}\right)\int_{-\infty}^{\infty} e^{-\frac{1}{2}(z)^{\mathsf{T}}z}dz \tag{30.0.70}$$

Since $z^{\mathsf{T}}z = \sum_i w_i^2$, the integral splits into a product of one-dimensional integrals

$$I = \det\left(\mathbf{\Sigma}^{\frac{1}{2}}\right)\prod_i \int_{-\infty}^{\infty} e^{-\frac{1}{2}z_i^2}dz_i = \det\left((2\pi)^{\frac{1}{2}}\mathbf{\Sigma}^{\frac{1}{2}}\right) = \sqrt{\det\left(2\pi\mathbf{\Sigma}\right)}. \tag{30.0.71}$$

8.13 The skewness is clearly zero since the Gaussian is symmetric. For the kurtosis, w.l.o.g we can consider a zero mean Gaussian, for which

$$\langle x^4\rangle = \frac{1}{\sqrt{2\pi\sigma^2}}\int x^4 e^{-\frac{1}{2\sigma^2}x^2}dx = \sigma^4\frac{1}{\sqrt{2\pi}}\int z^4 e^{-\frac{1}{2\sigma^2}z^2}dz$$

Using integration by parts, with parts $xz3$ and $ze^{-\frac{1}{2}z^2}dz$, this becomes

$$\langle x^4\rangle = \frac{\sigma^4}{\sqrt{2\pi}}\left(-z^2e^{-\frac{1}{2}z^2}|_{-\infty}^{\infty} + 3\int z^2e^{-\frac{1}{2}z^2}\right) = 3\sigma^4$$

Using the definition of kurtosis, we obtain the required result.

8.14 Let's first find the probability there is no event in the interval, which is $1-\gamma$, where $\gamma = \theta\delta t$. Then the probability there is no event in all intervals is

$$(1-\gamma)^{t/\delta t} = (1-\theta\delta t)^{t/\delta t} \tag{30.0.72}$$

In the limit $\delta t \to 0$, this becomes

$$e^{-\theta t} \tag{30.0.73}$$

Hence the probability that there is at least one event is $1 - e^{-\theta t}$.

8.15

1. The joint distribution factorises since $e^{\sum_i \theta_i\phi_i(x_i)} = \prod_i e^{\theta_i\phi_i(x_i)}$, and therefore all x_i are independent.
2. Defining $Z_i = \int e^{\theta_i\phi_i(x_i)}dx_i$, then $Z = \prod_i Z_i$.
3. Using the Jacobian, the distribution of $\mathbf{y}$ is given by

   $$p(\mathbf{y}) = \det\left(\mathbf{M}\right)\prod_i e^{\theta_i\phi_i(\sum_j M_{ij}y_j)}$$

   This is clearly non-factorised for a general $\mathbf{M}$. However, this is a tractable distribution since the correlations in $\mathbf{y}$ simply appear because of a linear coordinate transformation.

8.16

$$m = \frac{\alpha}{\alpha+\beta} \to \alpha = \beta\gamma, \gamma = m/(1-m) \tag{30.0.74}$$

Substitute this in the equation for the variance and solve for β.

8.23 For $\gamma \to \infty$, we have $\tilde{a} = N, \tilde{b} = \sum_n x^n, \tilde{c} = \sum_n (x^n)^2$. As $\beta \to \infty$, $1/\tilde{\beta} = N\sigma^2/2$. As a function of λ the posterior is

$$-\lambda\left(\frac{1}{\tilde{a}}\left(\mu - \tilde{b}/\tilde{a}\right)^2 + 1/\tilde{\beta}\right) + (\alpha + N/2 - 1/2)\log\lambda \tag{30.0.75}$$

Differentiating with respect to λ and equating to zero, we have that optimally $1\lambda = N\sigma^2/\left(2\alpha + N - 1\right)$. Setting $\alpha = 1/2$ gives then the standard result for the variance. The maximum likelihood mean can be obtained in a similar way by optimising with respect to μ, which gives $\mu = \tilde{b}/\tilde{a}$ which is $\sum_n x^n/N$ in the given limits.

8.24 In the expressed limits, $\tilde{b} = \sum_n x^n$, $\tilde{a} = N$, and $\tilde{c} = \sum_n (x^n)^2$. In this case

$$\tilde{c} - \frac{\tilde{b}^2}{\tilde{a}} = \sum_n (x^n)^2 - \frac{\left(\sum_n x^n\right)^2}{N} = N\sigma^2_{ML} \tag{30.0.76}$$

where $\sigma^2_{ML} = \frac{1}{N}\sum_n (x^n - \bar{x})^2$, $\bar{x} = \frac{1}{N}\sum_n x^n$.

The optimal setting for μ is clearly given at $\mu_* = \tilde{b}/\tilde{a} = \bar{x}$. Using this setting for μ, the log posterior distribution, ignoring constants, is

$$\frac{1}{2}\log\lambda + (\alpha + N/2 - 1)\log\lambda - \lambda/\tilde{\beta} \tag{30.0.77}$$

Differentiating and equating to zero we have the optimal solution given by

$$\frac{1}{\lambda_*} = \frac{1}{\tilde{\beta}} + \frac{1}{\alpha - 1 + \frac{N+1}{2}} \tag{30.0.78}$$

In the limits given, $1/\tilde{\beta} = \frac{1}{2}N\sigma^2_{ML}$, hence

$$\sigma^2_* = \frac{N}{2\alpha - 2 + N + 1}\sigma^2_{ML} \tag{30.0.79}$$

which for $\alpha = 1$ gives the 'unbiased' estimator of the variance[1].

8.25

$$p(\mu|\mathcal{X}) = \frac{1}{\Gamma(\alpha)\beta^\alpha}\sqrt{\frac{N'}{2\pi}}\int_\lambda \lambda^{\frac{1}{2}} e^{-\frac{\lambda N'}{2}(\mu-\hat{m})^2}\lambda^{\alpha-1}e^{-\lambda/\beta} \tag{30.0.80}$$

where

$$\hat{m} \equiv \tilde{a}/\tilde{b}, \qquad N' \equiv (1+\tilde{a}) \tag{30.0.81}$$

Collecting the polynomial terms in λ together, and collecting the exponential terms in λ together, this becomes proportional to a Gamma distribution. Using the fact that the normalisation constant of a Gamma distribution is $\Gamma(\alpha)\beta^\alpha$ we have

$$p(\mu|\mathcal{X}) = \frac{\Gamma(\alpha+\frac{1}{2})}{\Gamma(\alpha)\beta^\alpha}\sqrt{\frac{N'}{2\pi}}\left[\frac{1}{\beta} + \frac{N'}{2}(\mu-\hat{m})^2\right]^{-\alpha-\frac{1}{2}} \tag{30.0.82}$$

which is a Student's t-distribution for suitably chosen parameters. This is easily seen by taking out a factor of $1/\beta$ to give

$$p(\mu|\mathcal{X}) = \frac{\Gamma(\alpha+\frac{1}{2})}{\Gamma(\alpha)}\sqrt{\frac{\beta N'}{2\pi}}\left[1 + \frac{\beta N'}{2}(\mu-\hat{m})^2\right]^{-\alpha-\frac{1}{2}} \tag{30.0.83}$$

which is a Student's t-distribution, definition(8.26) with mean $\hat{m}$, degrees of freedom 2α and 'scale' $N'\beta\alpha$.

8.29

1.
$$p(\tau|\mathcal{X}) \propto p(\mathcal{X}|\tau)p(\tau) = p(\tau)\prod_n p(x^n|\tau) \propto \tau^{a-1}e^{-b\tau}\prod_n\left[\tau^{1/2}e^{-\frac{\tau}{2}(x^n-\mu)^2}\right]$$

   Gathering terms in the exponent establishes the result.

2.
$$\int_\tau e^{-\frac{\tau}{2}(x-\mu)^2}\tau^{a'-1}e^{-b'\tau}$$

3. We can appeal directly to equation (8.3.28) to establish the result.

8.35 Ignoring the 1/2 factors, the first two terms give

$$\left(\mathbf{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \mathbf{\Sigma}_2^{-1}\boldsymbol{\mu}_2\right)^\mathsf{T}\left(\mathbf{\Sigma}_1(\mathbf{\Sigma}_1+\mathbf{\Sigma}_2)^{-1}\mathbf{\Sigma}_2\right)\left(\mathbf{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \mathbf{\Sigma}_2^{-1}\boldsymbol{\mu}_2\right) - \left(\boldsymbol{\mu}_1^\mathsf{T}\mathbf{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^\mathsf{T}\mathbf{\Sigma}_2^{-1}\boldsymbol{\mu}_2\right)$$

which is

$$\left(\mathbf{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \mathbf{\Sigma}_2^{-1}\boldsymbol{\mu}_2\right)^\mathsf{T}\left(\mathbf{\Sigma}_1(\mathbf{\Sigma}_1+\mathbf{\Sigma}_2)^{-1}\mathbf{\Sigma}_2\right)\left(\mathbf{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \mathbf{\Sigma}_2^{-1}\boldsymbol{\mu}_2\right) - \left(\boldsymbol{\mu}_1^\mathsf{T}\mathbf{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^\mathsf{T}\mathbf{\Sigma}_2^{-1}\boldsymbol{\mu}_2\right)$$

Defining $\mathbf{S} = (\mathbf{\Sigma}_1 + \mathbf{\Sigma}_2)$,

$$\left(\mathbf{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \mathbf{\Sigma}_2^{-1}\boldsymbol{\mu}_2\right)^\mathsf{T}\left(\mathbf{\Sigma}_2\mathbf{S}^{-1}\boldsymbol{\mu}_1 + \mathbf{\Sigma}_1\mathbf{S}^{-1}\boldsymbol{\mu}_2\right) - \left(\boldsymbol{\mu}_1^\mathsf{T}\mathbf{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^\mathsf{T}\mathbf{\Sigma}_2^{-1}\boldsymbol{\mu}_2\right)$$

$$\boldsymbol{\mu}_1^\mathsf{T}\left(\mathbf{\Sigma}_1^{-1}\mathbf{\Sigma}_2\mathbf{S}^{-1}\boldsymbol{\mu}_1 + \mathbf{S}^{-1}\boldsymbol{\mu}_2\right) + \boldsymbol{\mu}_2^\mathsf{T}\left(\mathbf{\Sigma}_2^{-1}\mathbf{\Sigma}_1\mathbf{S}^{-1}\boldsymbol{\mu}_2 + \mathbf{S}^{-1}\boldsymbol{\mu}_1\right) - \left(\boldsymbol{\mu}_1^\mathsf{T}\mathbf{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^\mathsf{T}\mathbf{\Sigma}_2^{-1}\boldsymbol{\mu}_2\right)$$

$$= \boldsymbol{\mu}_1^\mathsf{T}\left(\mathbf{\Sigma}_1^{-1}\mathbf{\Sigma}_2\mathbf{S}^{-1} - \mathbf{\Sigma}_1^{-1}\right)\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^\mathsf{T}\left(\mathbf{\Sigma}_2^{-1}\mathbf{\Sigma}_1\mathbf{S}^{-1} - \mathbf{\Sigma}_2^{-1}\right)\boldsymbol{\mu}_2 + 2\boldsymbol{\mu}_1^\mathsf{T}\mathbf{S}^{-1}\boldsymbol{\mu}_2 \tag{30.0.84}$$

$$2\boldsymbol{\mu}_1^\mathsf{T}\mathbf{S}^{-1}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1^\mathsf{T}\mathbf{S}^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2^\mathsf{T}\mathbf{S}^{-1}\boldsymbol{\mu}_2$$

[1]For readers familiar with David MacKay's Bayesian setting of the mean and variance of a Gaussian, here for conjugacy we constrain the prior mean $\sigma_0^2 = \gamma\sigma^2$, which leads to a slightly different treatment than presented in [197].

Note that the form equation (3) only holds provided that all quantities are well defined. This is not the case for example if we want to find the product of two Gaussians on different spaces

$$\mathcal{N}\left(x_1 | \mu_1, \sigma_1^2\right) \mathcal{N}\left(x_2 | \mu_2, \sigma_2^2\right)$$

Defining $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, we can then write the exponent for this as proportional to

$$(\mathbf{x} - \tilde{\boldsymbol{\mu}}_1)^\mathsf{T} \underbrace{\begin{pmatrix} \sigma_1^{-2} & 0 \\ 0 & 0 \end{pmatrix}}_{\tilde{\boldsymbol{\Sigma}}_1^{-1}} (\mathbf{x} - \tilde{\boldsymbol{\mu}}_1) + (\mathbf{x} - \tilde{\boldsymbol{\mu}}_2)^\mathsf{T} \underbrace{\begin{pmatrix} 0 & 0 \\ 0 & \sigma_2^{-2} \end{pmatrix}}_{\tilde{\boldsymbol{\Sigma}}_2^{-1}} (\mathbf{x} - \tilde{\boldsymbol{\mu}}_2)$$

where

$$\tilde{\boldsymbol{\mu}}_1 = \begin{pmatrix} \mu_1 \\ 0 \end{pmatrix}, \quad \tilde{\boldsymbol{\mu}}_2 = \begin{pmatrix} 0 \\ \mu_2 \end{pmatrix}$$

However in this case the covariance in $\mathbf{x}$ is not well defined. A general expression that works in this case is therefore to define the matrices

$$\tilde{\boldsymbol{\Sigma}}_1^{-1}, \tilde{\boldsymbol{\Sigma}}_2^{-1}$$

which may contain zeros on their diagonals, and have the $\boldsymbol{\Sigma}_i^{-1}$ embedded within them, as above. Then defining $\mathbf{A} = \tilde{\boldsymbol{\Sigma}}_1^{-1} + \tilde{\boldsymbol{\Sigma}}_2^{-1}$ and $\mathbf{b} = \tilde{\boldsymbol{\Sigma}}_1^{-1}\tilde{\boldsymbol{\mu}}_1 + \tilde{\boldsymbol{\Sigma}}_2^{-1}\tilde{\boldsymbol{\mu}}_2$, the product is a Gaussian with mean $\mathbf{A}^{-1}\mathbf{b}$ and covariance $\mathbf{A}^{-1}$ with log prefactor

$$\frac{1}{2}\mathbf{b}^\mathsf{T}\mathbf{A}^{-1}\mathbf{b} - \frac{1}{2}\left(\boldsymbol{\mu}_1^\mathsf{T}\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^\mathsf{T}\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\mu}_2\right) - \frac{1}{2}\log\det\left(2\pi\boldsymbol{\Sigma}_1\right)\det\left(2\pi\boldsymbol{\Sigma}_2\right) - \frac{1}{2}\log\det\left(\mathbf{A}/(2\pi)\right)$$

8.41 Defining $\Delta x^n \equiv x^n - \langle x \rangle$, $\Delta y^n \equiv y^n - \langle y \rangle$, one can first show that

$$\rho_{x,z} = \frac{\sqrt{N}}{\sqrt{\sum_{n=1}^N \left(\Delta x^n + \Delta y^n\right)^2}} \left(\sigma_x + \rho_{x,y}\sigma_y\right) \tag{30.0.85}$$

Then use the triangle inequality:

$$\frac{1}{\sqrt{N}}\sqrt{\sum_{n=1}^N \left(\Delta x^n + \Delta y^n\right)^2} \leq \sigma_x + \sigma_y \tag{30.0.86}$$

which gives

$$\rho_{x,z} \geq \frac{\sigma_x + \sigma_y \rho_{x,y}}{\sigma_x + \sigma_y} \tag{30.0.87}$$

Defining $0 \leq p \equiv \sigma_x/(\sigma_x + \sigma_y) \leq 1$ then

$$\rho_{x,z} \geq \rho_{x,y} + p(1 - \rho_{x,y}) \tag{30.0.88}$$

Since $-1 \leq \rho_{x,y} \leq 1$, and $p > 0$ we must have $p(1 - \rho_{x,y}) \geq 0$, giving the desired result. The result is 'geometrically obvious' since if we take two vectors $\mathbf{x}$ and $\mathbf{y}$ then $\mathbf{z} = \mathbf{x} + \mathbf{y}$ must be closer in angle to $\mathbf{x}$ than $\mathbf{y}$ is to $\mathbf{x}$.

9.1 See `demoMLprinter.m`. For the final part, the symptom variables for which no evidence is known simply disappear under marginalisation (since they are nodes on the graph with no children), leaving a graph with the same structure but on a reduced set of variables. One can then set the evidential states of this new distribution and carry out max-absorption.

```
Part 1:

p(fuse):
fuse  =0  4/5
fuse  =1  1/5

--------------------
p(drum):
drum  =0  11/15
drum  =1  4/15

--------------------
p(toner):
toner  =0  2/3
toner  =1  1/3

--------------------
p(poorpaper):
poorpaper  =0  7/15
poorpaper  =1  8/15
```

```
--------------------
p(roller):
roller  =0  4/5
roller  =1  1/5

--------------------
p(burning| fuse):
burning  =0  fuse  =0  1
burning  =1  fuse  =0  0
burning  =0  fuse  =1  1/3
burning  =1  fuse  =1  2/3

--------------------
p(poorprint| drum toner poorpaper):
poorprint  =0  drum  =0  toner  =0  poorpaper  =0  1
poorprint  =1  drum  =0  toner  =0  poorpaper  =0  0
poorprint  =0  drum  =1  toner  =0  poorpaper  =0  0
poorprint  =1  drum  =1  toner  =0  poorpaper  =0  1
poorprint  =0  drum  =0  toner  =1  poorpaper  =0  0
poorprint  =1  drum  =0  toner  =1  poorpaper  =0  1
poorprint  =0  drum  =1  toner  =1  poorpaper  =0  0
poorprint  =1  drum  =1  toner  =1  poorpaper  =0  1
poorprint  =0  drum  =0  toner  =0  poorpaper  =1  4/5
poorprint  =1  drum  =0  toner  =0  poorpaper  =1  1/5
poorprint  =0  drum  =1  toner  =0  poorpaper  =1  0
poorprint  =1  drum  =1  toner  =0  poorpaper  =1  1
poorprint  =0  drum  =0  toner  =1  poorpaper  =1  0
poorprint  =1  drum  =0  toner  =1  poorpaper  =1  1
poorprint  =0  drum  =1  toner  =1  poorpaper  =1  0
poorprint  =1  drum  =1  toner  =1  poorpaper  =1  1

--------------------
p(wrinkled| fuse poorpaper):
wrinkled  =0  fuse  =0  poorpaper  =0  4/5
wrinkled  =1  fuse  =0  poorpaper  =0  1/5
wrinkled  =0  fuse  =1  poorpaper  =0  1/2
wrinkled  =1  fuse  =1  poorpaper  =0  1/2
wrinkled  =0  fuse  =0  poorpaper  =1  5/7
wrinkled  =1  fuse  =0  poorpaper  =1  2/7
wrinkled  =0  fuse  =1  poorpaper  =1  0
wrinkled  =1  fuse  =1  poorpaper  =1  1

--------------------
p(multiple| poorpaper roller):
multiple  =0  poorpaper  =0  roller  =0  1
multiple  =1  poorpaper  =0  roller  =0  0
multiple  =0  poorpaper  =1  roller  =0  5/7
multiple  =1  poorpaper  =1  roller  =0  2/7
multiple  =0  poorpaper  =0  roller  =1  1/2
multiple  =1  poorpaper  =0  roller  =1  1/2
multiple  =0  poorpaper  =1  roller  =1  0
multiple  =1  poorpaper  =1  roller  =1  1

--------------------
p(jam| fuse roller):
jam  =0  fuse  =0  roller  =0  3/5
jam  =1  fuse  =0  roller  =0  2/5
jam  =0  fuse  =1  roller  =0  1/2
jam  =1  fuse  =1  roller  =0  1/2
jam  =0  fuse  =0  roller  =1  0
jam  =1  fuse  =0  roller  =1  1
jam  =0  fuse  =1  roller  =1  1
jam  =1  fuse  =1  roller  =1  0

--------------------

--------------------
Part 2:

probability fuse assembly malfunctioned is:
fuse  =0
fuse  =1  1.000000e+000

--------------------
Part 3 (Bayesian tables):

p(fuse):
fuse  =0  13/17
fuse  =1  4/17
```

```
--------------------
p(drum):
drum  =0  12/17
drum  =1  5/17

--------------------
p(toner):
toner  =0  11/17
toner  =1  6/17

--------------------
p(poorpaper):
poorpaper  =0  8/17
poorpaper  =1  9/17

--------------------
p(roller):
roller  =0  13/17
roller  =1  4/17

--------------------
p(burning| fuse):
burning  =0  fuse  =0  13/14
burning  =1  fuse  =0  1/14
burning  =0  fuse  =1  2/5
burning  =1  fuse  =1  3/5

--------------------
p(poorprint| drum toner poorpaper):
poorprint  =0  drum  =0  toner  =0  poorpaper  =0  4/5
poorprint  =1  drum  =0  toner  =0  poorpaper  =0  1/5
poorprint  =0  drum  =1  toner  =0  poorpaper  =0  1/3
poorprint  =1  drum  =1  toner  =0  poorpaper  =0  2/3
poorprint  =0  drum  =0  toner  =1  poorpaper  =0  1/4
poorprint  =1  drum  =0  toner  =1  poorpaper  =0  3/4
poorprint  =0  drum  =1  toner  =1  poorpaper  =0  1/3
poorprint  =1  drum  =1  toner  =1  poorpaper  =0  2/3
poorprint  =0  drum  =0  toner  =0  poorpaper  =1  5/7
poorprint  =1  drum  =0  toner  =0  poorpaper  =1  2/7
poorprint  =0  drum  =1  toner  =0  poorpaper  =1  1/3
poorprint  =1  drum  =1  toner  =0  poorpaper  =1  2/3
poorprint  =0  drum  =0  toner  =1  poorpaper  =1  1/3
poorprint  =1  drum  =0  toner  =1  poorpaper  =1  2/3
poorprint  =0  drum  =1  toner  =1  poorpaper  =1  1/3
poorprint  =1  drum  =1  toner  =1  poorpaper  =1  2/3

--------------------
p(wrinkled| fuse poorpaper):
wrinkled  =0  fuse  =0  poorpaper  =0  5/7
wrinkled  =1  fuse  =0  poorpaper  =0  2/7
wrinkled  =0  fuse  =1  poorpaper  =0  1/2
wrinkled  =1  fuse  =1  poorpaper  =0  1/2
wrinkled  =0  fuse  =0  poorpaper  =1  2/3
wrinkled  =1  fuse  =0  poorpaper  =1  1/3
wrinkled  =0  fuse  =1  poorpaper  =1  1/3
wrinkled  =1  fuse  =1  poorpaper  =1  2/3

--------------------
p(multiple| poorpaper roller):
multiple  =0  poorpaper  =0  roller  =0  6/7
multiple  =1  poorpaper  =0  roller  =0  1/7
multiple  =0  poorpaper  =1  roller  =0  2/3
multiple  =1  poorpaper  =1  roller  =0  1/3
multiple  =0  poorpaper  =0  roller  =1  1/2
multiple  =1  poorpaper  =0  roller  =1  1/2
multiple  =0  poorpaper  =1  roller  =1  1/3
multiple  =1  poorpaper  =1  roller  =1  2/3

--------------------
p(jam| fuse roller):
jam  =0  fuse  =0  roller  =0  7/12
jam  =1  fuse  =0  roller  =0  5/12
jam  =0  fuse  =1  roller  =0  1/2
jam  =1  fuse  =1  roller  =0  1/2
jam  =0  fuse  =0  roller  =1  1/4
jam  =1  fuse  =0  roller  =1  3/4
jam  =0  fuse  =1  roller  =1  2/3
jam  =1  fuse  =1  roller  =1  1/3
```

```
--------------------
probability fuse assembly malfunctioned is:
fuse  =0  3.813848e-001
fuse  =1  6.186152e-001

--------------------
Part 4 (using Bayesian tables):

optimal joint state of faults given evidence:
optimal states in 1/2 coding is:
fuse=2
drum=1
toner=1
poorpaper=1
roller=1

--------------------
Part 5 (using Bayesian tables):

optimal joint state of faults given only burning smell and paper jammed:
optimal states in 1/2 coding is:
fuse=2
drum=1
toner=1
poorpaper=2
roller=1
```

9.4 For a discrete variable, the normalisation constraint is

$$1 = \sum_s p(x_i = s|\text{pa}(x_i) = \mathbf{t}^i) = \sum_s \theta_s^i(\mathbf{t}^i) \tag{30.0.89}$$

The log likelihood as a function of the table parameters is then

$$\sum_{i=1}^K \sum_{n=1}^N \mathbb{I}[x_i^n = s] \mathbb{I}\left[\text{pa}(x_i^n) = \mathbf{t}^i\right] \log \theta_s^i(\mathbf{t}) + \sum_{i=1}^K \sum_{\mathbf{t}^i} \lambda^i(\mathbf{t}^i) \left(1 - \sum_s \theta_s^i(\mathbf{t}^i)\right) \tag{30.0.90}$$

where $\lambda^i(\mathbf{t}^i)$ is a set of Lagrange multipliers. Differentiating the above with respect to $\theta_s^j(\mathbf{t}^j)$ and equating to zero, we have

$$\sum_{n=1}^N \mathbb{I}\left[x_j^n = s\right] \mathbb{I}\left[\text{pa}\left(x_j^n\right) = \mathbf{t}^j\right] \frac{1}{\theta_s^j(\mathbf{t}^j)} - \lambda^j(\mathbf{t}^j) = 0 \tag{30.0.91}$$

Rearranging gives

$$\theta_s^j(\mathbf{t}^j) = \frac{1}{\lambda^j(\mathbf{t}^j)} \sum_{n=1}^N \mathbb{I}\left[x_j^n = s\right] \mathbb{I}\left[\text{pa}\left(x_j^n\right) = \mathbf{t}^j\right] \tag{30.0.92}$$

From normalisation, we have

$$\theta_s^j(\mathbf{t}^j) = \frac{\sum_{n=1}^N \mathbb{I}\left[x_j^n = s\right] \mathbb{I}\left[\text{pa}\left(x_j^n\right) = \mathbf{t}^j\right]}{\sum_{n=1}^N \sum_s \mathbb{I}\left[x_j^n = s\right] \mathbb{I}\left[\text{pa}\left(x_j^n\right) = \mathbf{t}^j\right]} \tag{30.0.93}$$

9.5 Following the standard maximum likelihood example, one can write

$$CL(\theta) \overset{N\to\infty}{=} \langle \log p(y|x,\theta) \rangle_{p(x,y|\theta^0)} \tag{30.0.94}$$

We assume for concreteness that the variables are continuous (the same holds for discrete variables on replacing integration with summation) and rewrite the above as

$$CL(\theta) = \int p(x|\theta^0) \int p(y|x,\theta^0) \log p(y|x,\theta) \tag{30.0.95}$$

We can write this as

$$CL(\theta) = -\int p(x|\theta^0) \text{KL}\big(p(y|x,\theta^0)|p(y|x,\theta)\big) + \text{const}. \tag{30.0.96}$$

When $\theta = \theta^0$ the KL term is zero, and $CL(\theta)$ is maximal. Hence $\theta = \theta^0$ corresponds to an optimum of the conditional likelihood. However, this does not mean that one can necessarily learn all parameters. For example if a distribution is parameterised as

$$p(x,y|\theta) = p(y|x,\theta_1)p(x|\theta_2) \tag{30.0.97}$$

then the conditional likelihood of $p(y|x,\theta)$ is independent of θ_2.

9.6 The mean and variance of a distribution are given by

$$m = \frac{\alpha}{\alpha+\beta}, \qquad s = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$$

Simply plugging in the definitions and rearranging gives the result.

9.7 Follows from the definition and the Beta distribution and the digamma function. We can then use any gradient based technique.

9.9 Using $Z(u) = \frac{\prod_i \Gamma(u_i)}{\Gamma(\sum_i u_i)}$ and substituting in we obtain the following expression for the likelihood

$$\prod_{k,j} \frac{\prod_i \Gamma\left(u'_i(v_k;j)\right)}{\Gamma\left(\sum_i u'_i(v_k;j)\right)} \frac{\Gamma\left(\sum_i u_i(v_k;j)\right)}{\prod_i \Gamma\left(u_i(v_k;j)\right)} = \prod_k \prod_j \frac{\Gamma\left(\sum_i u_i(v_k;j)\right)}{\Gamma\left(\sum_i u'_i(v_k;j)\right)} \prod_i \left[\frac{\Gamma\left(u'_i(v_k;j)\right)}{\Gamma\left(u_i(v_k;j)\right)}\right] \tag{30.0.98}$$

9.10

1. $(1+7+7*6/2)*(1+6+6*5/2)*(1+5+5*4/2)*(1+4+4*3/2)*(1+3+3*2/2)*4*2 = 6288128$
2. $(1+7+7*6/2)+(1+6+6*5/2)+(1+5+5*4/2)+(1+4+4*3/2)+(1+3+3*2/2)+4+2 = 91$ seconds since we can use the decomposability to optimise each variable separately.
3. There are $2^7*2^6*2^5*2^4*2^3*2^2*2 = 268435456$ graphs. Using the decomposability, though this gives $\left(2^7+2^6+2^5+2^4+2^3+2^2+2\right)*$ 8! seconds or 118.5 days.

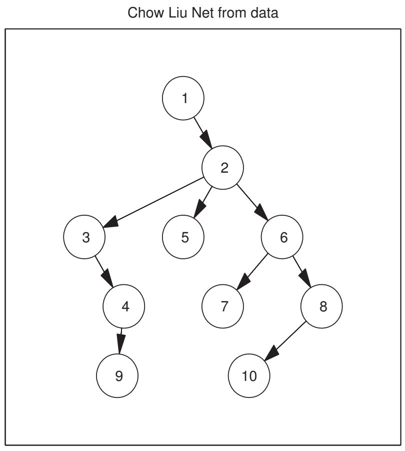
9.12 Sketch: One continues as in the standard IS development, reaching equation (9.6.43). At this point we may use, for arbitrarily non-negative p_c, $\sum_c p_c = 1$,

$$Z = \sum_{\mathcal{X}} e^{\sum_c \theta_c f_c(\mathcal{X}_c)} = \sum_{\mathcal{X}} e^{\langle \theta_c f_c(\mathcal{X}_c)\rangle_p}$$

Now $e^{\langle x\rangle} \leq \langle e^x\rangle$. Hence

$$Z \leq \sum_{\mathcal{X}} \sum_c p_c e^{\theta_c f_c(\mathcal{X}_c)/p_c} = \sum_c \gamma_c p_c \sum_{\mathcal{X}_c} e^{\theta_c f_c(\mathcal{X}_c)/p_c}$$

here γ_c takes care of enumerating over the states that are outside of $\mathcal{X}_c$. This then results in a parameter decoupling. However, we now have additional parameters to set, namely p_c. These can be optimised by including a Lagrange constraint $\sum_c p_c = 1$.



9.13 See `ChowLiu.m` and run `drawNet(ChowLiu(X))`. See also `demoChowLiu.m` for a demo.

```
function A=ChowLiu(data)
%CHOWLIU Chow Liu algorithm
%A=ChowLiu(data)
%data is a data matrix. See demoChowLiu
import brml.*
data=squeezestates(data);
nstates=maxarray(data,2); nvars=size(data,1);
c=0; % compute all pairwise Mutual Informations
for i=1:nvars
    for j=i+1:nvars
        w(i,j)=MIemp(data(i,:),data(j,:),nstates(i),nstates(j));
        c=c+1;
        link(c,:)=[i j]; val(c)=w(i,j);
    end
end
[dum b]=sort(val,'descend'); edgelist=link(b(1:c),:); % sort the edges
A=spantree(edgelist); % find a spanning
A=triu(A); % orient away from node 1 to produce a DAG
```

9.14 Lets consider an ordering of the nodes $1, 2, 3, \ldots, N$ with the last node N having parents from lower ordered nodes. For node N, we can either include or not an edge from each of lower ordered nodes. This gives 2^{N-1} possible parental graphs. We then move on to node $N-1$. Since this is a DAG we assume it can only have connections from nodes numbered from 1 to $N-2$, for which there are 2^{N-2} possible graphs, *etc.* This gives the lower bound result. The upper bound is obtained from the fact that there are $N!$ possible orderings of the nodes, each of which can have many parental graphs. This is an upper bound since we will overcount some graphs this way. For example, trivially, the graphs with no parental connections at all are clearly the same graph, whatever the ordering of the nodes.

10.1 Looking at the data, the estimates using maximum likelihood are

$$p(C=1|Young) = 0.5, p(F=1|Young) = 1, p(SP=1|Young) = 1, p(B=1|Young) = 0 \tag{30.0.99}$$

and

$$p(C=1|Ol\mathcal{D}) = 0.75, p(F=1|Ol\mathcal{D}) = 0.25, p(Sp=1|Ol\mathcal{D}) = 0.25, p(B=1|Ol\mathcal{D}) = 0.75 \tag{30.0.100}$$

and $p(Young) = 2/6$ and $p(Ol\mathcal{D}) = 4/6$. Plugging this into Bayes formula, we get

$$p(Young|C=0, F=1, SP=1, B=0) \propto 0.5 * 1 * 1 * 1/6 \tag{30.0.101}$$

$$p(Old|C=0, F=1, SP=1, B=0) \propto 0.25 * 0.25 * 0.25 * 0.25 * 4/6 \tag{30.0.102}$$

Using the fact that these probabilities sum to 1, this gives $p(Young|C=0, F=1, SP=1, B=0) = 64/65$

10.3 See `exercisePoliticsSport.m`.

```
probability x is about politics = 0.830599

P=[1 0 1 1 1 0 1 1;  % Politics
    0 0 0 1 0 0 1 1;
    1 0 0 1 1 0 1 0;
    0 1 0 0 1 1 0 1;
    0 0 0 1 1 0 1 1;
    0 0 0 1 1 0 0 1]

xS=[1 1 0 0 0 0 0 0; % Sport
    0 0 1 0 0 0 0 0;
    1 1 0 1 0 0 0 0;
    1 1 0 1 0 0 0 1;
    1 1 0 1 1 0 0 0;
    0 0 0 1 0 1 0 0;
    1 1 1 1 1 0 1 0]

pP = size(xP,1)/(size(xP,1) + size(xS,1)); pS =1-pP; % ML class priors pE = p(c=E), pS=p(c=S)

mP = mean(xP); % ML estimates of p(x=1|c=E)
mS = mean(xS); % ML estimates of p(x=1|c=S)

xtest=[1 0 0 1 1 1 1 0]; % test point

npP = pP*prod(mP.^xtest.*(1-mP).^(1-xtest)); % p(x,c=P)
npS = pS*prod(mS.^xtest.*(1-mS).^(1-xtest)); % p(x,c=S)

pxP = npP/(npP+npS);
fprintf(1,'probability x is about politics = %g\n',pxP);
```

10.4 The decision boundary is given by

$$p_1 \prod_i p(x_i|1) = p_0 \prod_i p(x_i|0)$$

$$p_1 \prod_i {\theta_i^1}^{x_i} (1-\theta_i^1)^{1-x_i} = p_0 \prod_i {\theta_i^0}^{x_i} (1-\theta_i^0)^{1-x_i}$$

Taking log, and defining $\alpha_i = \log \theta_i$, $\beta_i = \log(1-\theta_i)$, we have

$$\log p_1 + \sum_i x_i \alpha_i^1 + (1-x_i)\beta_i^1 = \log p_0 + \sum_i x_i \alpha_i^0 + (1-x_i)\beta_i^0$$

$$\sum_i x_i \underbrace{\left(\alpha_i^1 - \beta_i^1 - \alpha_i^0 + \beta_i^0\right)}_{w_i} + \underbrace{\log p_1 - \log p_0 + \sum_i \left(\beta_i^1 - \beta_i^0\right)}_{b} = 0$$

10.5 1. Standard Naive Bayes training as in the main text.

2.

$$p(c|x) = \frac{p(c)\prod_i p(x_i|c)}{\sum_c p(c)\prod_i p(x_i|c)}$$

3. The classification is certain that it is not spam. This is because the probability is zero, which annihilates all terms in the product in Naive Bayes. We can use a Bayesian method on the tables to help with this. Placing for example a uniform Dirichlet prior on the tables has the effect of simply adding 1 to all the data counts. This means that there would no longer be any zero counts in the data and the problem is thereby diminished. A spammer could attempt to fool a simple Naive Bayes classifier by for example including the spam text at the beginning of the email, and then appending a large amount of normal text at the end of the email. This would have the effect of biasing the statistics of the email towards a normal email, meaning that the Naive Bayes classifier would class this as a normal 'ham' email.

10.6 This just follows from defining

$$p(x, c) = \frac{1}{N}\sum_n \delta(x - x^n)\delta(c - c^n)$$

Then

$$\mathrm{KL}(p|q) = \mathsf{const.} - \langle \log q(x, c)\rangle_{p(x,c)} = \mathsf{const.} - \frac{1}{N}\sum_n \log q(x^n, c^n)$$

Hence minimising $\mathrm{KL}(p|q)$ corresponds to maximising $\sum_n \log q(x^n, c^n)$, which is the data log likelihood assuming the data is i.i.d..

10.7 Plugging in the structured form of $q(x, c)$ it is immediately clear that the optimal form is

$$q(x_i|x_{pa(i)}, c) = p(x_i|x_{pa(i)}, c) \tag{30.0.103}$$

Plugging this back in to the Kullback-Leibler divergence, we obtain that minimising the Kullback-Leibler divergence is

$$\begin{aligned}
&= -\sum_i \left\langle \log p(x_i|x_{pa(i)}, c)\right\rangle_{p(x_i,x_{pa(i)},c)}\\
&- \sum_i \left[\left\langle \log p(x_i, x_{pa(i)}, c)\right\rangle_{p(x_i,x_{pa(i)},c)} - \left\langle \log p(x_{pa(i)}, c)\right\rangle_{p(x_{pa(i)},c)}\right]\\
&- \sum_i \left[\left\langle \log p(x_i, x_{pa(i)}, c)\right\rangle_{p(x_i,x_{pa(i)},c)} - \left\langle \log p(x_{pa(i)}, c)\right\rangle_{p(x_{pa(i)},c)} - \langle \log p(x_i|c)\rangle_{p(x_i,c)} + \langle \log p(x_i|c)\rangle_{p(x_i,c)}\right]\\
&- \sum_i \left\langle \log \frac{p(x_i, x_{pa(i)}, c)}{p(x_{pa(i)}, c)p(x_i|c)}\right\rangle_{p(x_i,x_{pa(i)},c)} + \mathsf{const.}
\end{aligned}$$

where in the last line we used the fact that the final term $\langle \log p(x_i|c)\rangle_{p(x_{i,c})}$ is independent of the order $pa(i)$, with which we wish to maximise the expression L over. Dividing the numerator and denominator by $p(c)$, we arrive at the result that the maximum likelihood optimal setting is given by computing the weights associated with conditional mutual information. To learn the optimal tree we then find a maximum weighted spanning tree using these weights, and then orient the edges outwards from a chosen root. Finally we add links from c to each variable x_i.

11.1 See `demoEMprinter.m`. Based on 50 EM iterations, the probability of a drum unit problem, given the evidence and the learned table, is 0.618 to the decimal places.

11.2

11.3 This does not correspond to maximum likelihood. The reason is the available information is not fully used. When we observe y, even if we don't observe x, this gives some information about the state of x – indeed, this is what is used in the EM approach, since we will make use of the distribution $p(x|y^n)$. The suggested approach is not unreasonable, and essentially corresponds to throwing away the whole datapoint if there are missing entries. In general this would be very wasteful.

11.4 1. The energy is (for $q^n(h) \equiv p^{old}(h|\mathbf{v}^n)$):

$$\sum_n \langle \log p(\mathbf{v}^n|h)p(h)\rangle_{q^n(h)} = \sum_n \sum_{t=1}^{T-1} \left\langle \log p(v^n_{t+1}|v^n_t, h)\right\rangle_{q^n(h)} + \sum_n \langle \log p(v^n_1|h)\rangle_{q^n(h)} + T\sum_n \langle \log p(h)\rangle_{q^n(h)}$$

We need to optimise this *w.r.t.* the transitions $p(v_{t+1} = i|v_t = j, h) \equiv \theta^h_{ij}$, initial distribution $p(v_1|h)$ and prior $p(h)$. Using a Lagrange term, the contribution of the transition to the energy is

$$\mathcal{L} = \sum_n \sum_{t=1}^{T-1} \sum_{i,j} \sum_h q^n(h)\mathbb{I}\left[v^n_t = j, v^n_{t+1} = i\right] \log \theta^h_{ij} + \sum_h \sum_j \lambda_j \sum_i (1 - \theta^h_{ij})$$

Differentiating *w.r.t.* θ^h_{ij} and setting to zero, we obtain

$$\sum_n \sum_{t=1}^{T-1} q^n(h)\mathbb{I}\left[v^n_t = j, v^n_{t+1} = i\right] \frac{1}{\theta^h_{ij}} - \lambda_j = 0$$

Rearranging, this gives

$$\theta^h_{ij} = \frac{\sum_n q^n(h) \sum_{t=1}^{T-1} \mathbb{I}\left[v_t^n = j, v_{t+1}^n = i\right]}{\lambda_j}$$

Due to normalisation, $\sum_i \theta^h_{ij} = 1$, therefore

$$\theta^h_{ij} = \frac{\sum_n q^n(h) \sum_{t=1}^{T-1} \mathbb{I}\left[v_t^n = j, v_{t+1}^n = i\right]}{\sum_i \sum_n q^n(h) \sum_{t=1}^{T-1} \mathbb{I}\left[v_t^n = j, v_{t+1}^n = i\right]}$$

This has the interpretation of the weighted number of j to i transitions across the sequences. Similarly, the optimum of the energy *w.r.t.* the initial distribution $p(v_1 = i)|h) \equiv \theta_i$ is

$$\theta_i = \frac{\sum_n q^n(h)\mathbb{I}\left[v_1^n = i\right]}{\sum_i \sum_n q^n(h)\mathbb{I}\left[v_1^n = i\right]}$$

The optimal prior $p(h)$ is

$$p(h) = \frac{\sum_n q^n(h)}{\sum_h \sum_n q^n(h)}$$

One then iterates these equations to convergence.

2. See `mixMarkov.m`.

3. See `demoMixMarkov.m`. Running this several times, we end up with the highest log likelihood of -483.635. For this the sequences assignments are:

Group 1	Group 2
CATAGGCATTCTATGTGCTG	TGGAACCTTAAAAAAAAAAA
CCAGTTACGGACGCCGAAAG	GTCTCCTGCCCTCTCTGAAC
CGGCCGCGCCTCCGGGAACG	GTGCCTGGACCTGAAAAGCC
ACATGAACTACATAGTATAA	AAAGTGCTCTGAAAACTCAC
GTTGGTCAGCACACGGACTG	CCTCCCCTCCCCTTTCCTGC
CACTACGGCTACCTGGGCAA	TAAGTGTCCTCTGCTCCTAA
CGGTCCGTCCGAGGCACTCG	AAAGAACTCCCCTCCCTGCC
CACCATCACCCTTGCTAAGG	AAAAAAACGAAAAACCTAAG
CAAATGCCTCACGCGTCTCA	GCGTAAAAAAAGTCCTGGGT
GCCAAGCAGGGTCTCAACTT	
CATGGACTGCTCCACAAAGG	

11.5 To be done.

11.6

1. In the three layered general case, this is in principle more powerful than the two-layered case since when we marginalise over $\mathbf{h}_3$ we end up with a non-factorised contribution in $\mathbf{h}_2$, which is not in the exponential family. Only if the form of $\phi(\mathbf{h}_1, \mathbf{h}_2)$ were completely unrestricted (not of a BM form), does the extra layer play no role.

2. For the restricted BM part of the question, the same holds as well. In this case it is even more clear since the additional layer has the effect of inducing links between the nodes in the $\mathbf{h}_2$ layer.

11.7

1. A belief network in which the top row represents $p(\mathbf{x})$, and then each layer below $p(\mathbf{x}^{l-1}|\mathbf{x}^l)$. The variables $\mathbf{x}^0$ are in the bottom layer.

2. The graph is multiply-connected. However, one can collect all variables in each layer and use message passing from layer to layer. The number of entries of each table $p(\mathbf{x}^{l-1}|\mathbf{x}^l)$ is $2^w \times 2^w = 2^{2w}$. Since we have a linear-chain factor graph on the cluster variables $\mathbf{x}$, then we can compute the marginal in $O\left(L2^{2w}\right)$ time using message passing from layer to layer.

3. The energy term decomposes as

$$\begin{aligned}\left\langle \log p(\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^L)\right\rangle_{q(\mathbf{x})} &= \sum_l \sum_i \left\langle \log p(x_i^{l-1}|\mathbf{x}^l)\right\rangle_{q(x_i^{l-1}, \mathbf{x}^l)} \\ &= \sum_l \sum_i \left\langle \log \sigma\left((2x_i^{l-1} - 1)\mathbf{w}_{i,l}^{\mathsf{T}}\mathbf{x}^l\right)\right\rangle_{q(x_i^{l-1}, \mathbf{x}^l)}\end{aligned}$$

To compute each average, we therefore need to know the distribution of $(2x_i^{l-1} - 1)\mathbf{w}_{i,l}^{\mathsf{T}}\mathbf{x}^l$. If we first assume $x_i^{l-1} = 1$, we would therefore need to compute the distribution of the projection $\mathbf{w}_{i,l}^{\mathsf{T}}\mathbf{x}^l$. Whilst for certain distributions, such as Gaussian $\mathbf{x}^l$, this is straightforward (the distribution of a linear projection of a Gaussian is another Gaussian), for binary $\mathbf{x}^l$, there is no simple way to compute the distribution of this projection. In general, it would require that we compute $O\left(2^w\right)$ summations to find this exactly. For moderate w, this may still be feasible. The complexity of computing the variational EM bound is therefore $O\left(LW2^w\right)$, compared to $O\left(L2^{2w}\right)$ in the exact EM case.

11.8

$$L(\theta_b, \theta_g, \theta_p) = \log\theta_b + \log\theta_g + \lambda\left(1 - \theta_b - \theta_g - \theta_p\right)$$

Differentiating *w.r.t.* θ_b, and equating to zero, we have

$$\frac{1}{\theta_b} - \lambda = 0, \Rightarrow \theta_b = \frac{1}{\lambda}$$

Similarly, optimising *w.r.t.* θ_g gives $\theta_g = 1/\lambda = \theta_b$. Since $\theta_b \geq 0$ then $\lambda \geq 0$. Hence, for θ_p, the optimal setting is given when $\theta_p = 0$ (since then we are subtracting the minimal amount from L). The normalisation criterion $\theta_b + \theta_g + \theta_q = 1$ then gives that $\theta_b = \theta_g = 1/2$.

11.9

$$p(x_1 = i) = \sum_{x_2} p(x_1 = i, x_2) = \sum_{j=1}^{2} \theta_{1,j}$$

This means that θ only contributes to the likelihood via $\sum_{j=1}^{2} \theta_{1,j}$. Any θ which has this same 'projection' is therefore maximum likelihood equivalent. In terms of the matrix θ, this means that two different θ's which have the same row sum will have the same likelihood. This is the case in the example since in both cases, the row sum of the two matrices is

$$\begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix}$$

12.1 To avoid numerical underflow we can use the `logsumexp.m function. See demoCoin.m.`

12.2 See MATLAB code `demoBayesModelHedge.m`.

12.4

$$p(\mathbf{o}_a, \mathbf{o}_b, \mathbf{o}_c) p(H_{\mathsf{indep}} | \mathbf{o}_a, \mathbf{o}_b, \mathbf{o}_c) = p(H_{\mathsf{indep}}) \frac{Z(\mathbf{u} + \sharp^a)}{Z(\mathbf{u})} \frac{Z(\mathbf{u} + \sharp^b)}{Z(\mathbf{u})} \frac{Z(\mathbf{u} + \sharp^c)}{Z(\mathbf{u})} \tag{30.0.104}$$

where $Z(\mathbf{u})$ is given by equation (12.6.10).

$$\begin{aligned} p(\mathbf{o}_a, \mathbf{o}_b, \mathbf{o}_c) p(H_{\mathsf{same}} | \mathbf{o}_a, \mathbf{o}_b, \mathbf{o}_c) &= p(H_{\mathsf{same}}) \int p(\mathbf{o}_a | \boldsymbol{\alpha}, H_{\mathsf{same}}) p(\mathbf{o}_b | \boldsymbol{\alpha}, H_{\mathsf{same}}) p(\mathbf{o}_c | \boldsymbol{\alpha}, H_{\mathsf{same}}) p(\boldsymbol{\alpha} | H_{\mathsf{same}}) d\boldsymbol{\alpha} \\ &= p(H_{\mathsf{same}}) \frac{Z(\mathbf{u} + \sharp^a + \sharp^b + \sharp^c)}{Z(\mathbf{u})} \end{aligned}$$

If we assume no prior preference for either hypothesis, $p(H_{\mathsf{indep}}) = p(H_{\mathsf{same}})$, then

$$\frac{p(H_{\mathsf{indep}} | \mathbf{o}_a, \mathbf{o}_b, \mathbf{o}_c)}{p(H_{\mathsf{same}} | \mathbf{o}_a, \mathbf{o}_b, \mathbf{o}_c)} = \frac{Z(\mathbf{u} + \sharp^a) Z(\mathbf{u} + \sharp^b) Z(\mathbf{u} + \sharp^c)}{Z^2(\mathbf{u}) Z(\mathbf{u} + \sharp^a + \sharp^b + \sharp^c)} = \frac{Z([14, 4, 5]) Z([5, 10, 8]) Z([9, 9, 5])}{Z^2([1, 1, 1]) Z([26, 21, 16])} = 2.7585863899362$$

This is computed using `logZdirichlet.m`. Hence it is around 2.7 times more probable that the people are classifying the images using the same distribution, as opposed to each using their own distribution.

12.5

1. $$p(\mathcal{D} | \mathcal{H}_{\mathsf{random}}) = \prod_{n=1}^{R+W} 0.5 = 0.5^{R+W}$$

2. $$p(\mathcal{D} | \mathcal{H}_{\mathsf{non\ random}}) = \int_\theta \theta^R (1-\theta)^W \theta^{a-1} (1-\theta)^{b-1} / B(a, b) = B(R + a, W + b) / B(a, b)$$

3. $$p(\mathcal{H}_{\mathsf{random}} | \mathcal{D}) = \frac{p(\mathcal{D} | \mathcal{H}_{\mathsf{random}}) p(\mathcal{H}_{\mathsf{random}})}{p(\mathcal{D} | \mathcal{H}_{\mathsf{non\ random}}) p(\mathcal{H}_{\mathsf{non\ random}}) + p(\mathcal{D} | \mathcal{H}_{\mathsf{random}}) p(\mathcal{H}_{\mathsf{random}})}$$

   Using $p(\mathcal{H}_{\mathsf{non\ random}}) = p(\mathcal{H}_{\mathsf{random}})$ and the definitions above, gives the required result.

4. $$0.5^2 2 / (0.5^2 2 + beta(11, 13)/beta(1, 1)) = 0.780024735721027$$

   $$0.5^2 20 / (0.5^2 20 + beta(101, 121)/beta(1, 1)) = 0.827528384491262$$

   This is slightly higher than in the first instance.

5. Let $N = R + W$. For $x_n \in \{0, 1\}$:

   $$\left\langle \sum_{n=1}^{N} x_n \right\rangle = 0.5N$$

   $$\left\langle \left(\sum_{n=1}^{N} x_n \right)^2 \right\rangle = \sum_{n,n'} \langle x_n x'_n \rangle = 0.5N + 0.5^2 N(N-1)$$

   Hence the variance is

   $$0.5N + 0.5^2 N(N-1) - 0.5^2 N^2 = 0.5N - 0.25N = 0.25N$$

   and standard deviation is $\sqrt{0.25N} = 0.5\sqrt{R + W}$. In the above computations, we have $N = 22$ and $N = 220$. In the first case, for a random classifier, we would therefore expect to see deviations from 11 (the mean) of around 1.65. In the second case, we would expect deviations from the mean of 110 of around 5.24.

13.1 The decision boundary is given by

$$p(\textsf{class } 1|x) = p(\textsf{class } 2|x)$$

Ignoring the common denominator and taking logs, this is equivalent to

$$\log p(x|\textsf{class } 1) + \log p_1 = \log p(x|\textsf{class } 2) + \log p_2$$

which is

$$-\frac{1}{2\sigma_1^2}(x-\mu_1)^2 - \frac{1}{2}\log\left(2\pi\sigma_1^2\right) + \log p_1 = -\frac{1}{2\sigma_2^2}(x-\mu_2)^2 - \frac{1}{2}\log\left(2\pi\sigma_2^2\right) + \log p_2$$

$$-\frac{1}{\sigma_1^2}(x-\mu_1)^2 - \log\left(2\pi\sigma_1^2\right) + 2\log p_1 = -\frac{1}{\sigma_2^2}(x-\mu_2)^2 - \log\left(2\pi\sigma_2^2\right) + 2\log p_2$$

This can be written in the form

$$ax^2 + bx + c = 0$$

where

$$a = \frac{1}{\sigma_2^2} - \frac{1}{\sigma_1^2}$$
$$b = 2\left(\frac{\mu_1}{\sigma_1^2} - \frac{\mu_2}{\sigma_2^2}\right)$$
$$c = \frac{\mu_2^2}{\sigma_2^2} - \frac{\mu_1^2}{\sigma_1^2} - \log\sigma_1^2 + 2\log p_1 + \log\sigma_2^2 - 2\log p_2$$

Since we have a quadratic equation, there are two solutions,

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Using this one may create a solution in which the two Gaussians intersect where they both have appreciable mass, but another solution in which the Gaussians intersect when both are in the tails of the distribution. See `plotGaussianDecBoundary.m`. This is a natural, but potentially unintended, consequence of the Gaussian distribution. It's unlikely we will have confidence in estimating the tails of a distribution well, but the decision boundary can change there too.

13.2 Define the shorthand $p_j = p(\textsf{class } j|\mathbf{x})$, $q_j = q(\textsf{class } j|\mathbf{x})$. The classifier makes an error if the state sampled from q does not match the state sampled from p. Since we sample q and p independently, the expected error is given by

$$E = \sum_{i,j} \mathbb{I}\left[i \neq j\right] p(\textsf{class } i|\mathbf{x}) q(\textsf{class } j|\mathbf{x})$$

Since $\mathbb{I}\left[i \neq j\right] = 1 - \mathbb{I}\left[i = j\right]$, we can equivalently consider the q that maximises the accuracy

$$A = \sum_{i,j} \mathbb{I}\left[i = j\right] p(\textsf{class } i|\mathbf{x}) q(\textsf{class } j|\mathbf{x}) = \sum_j p(\textsf{class } j|\mathbf{x}) q(\textsf{class } j|\mathbf{x}) = \sum_j p_j q_j$$

Our task is then to find the distribution q that maximises $\sum_j p_j q_j$. Since $\sum_j p_j q_j$ is the average of p with respect to q, the optimal setting of q is to place all the mass in the highest valued component of p. That is, find that component i^* which corresponds to the highest value of $p_1, \ldots, p_K$, and then set

$$q_i = \begin{cases} 1 & \text{if } i = i^* \\ 0 & \text{otherwise} \end{cases}$$

This is Bayes' decision rule.

13.3

$$U(c^{pred}) = \sum_{c^{true}} U(c^{true}, c^{pred}) p(c^{true}) = \begin{cases} 3.2 & \text{class 1} \\ 2.9 & \text{class 2} \\ 1.3 & \text{class 3} \end{cases}$$

So the best decision is to choose class 1. See `ExerciseUtilpred.m`.

13.4

$$p(c=1|x) = \frac{p(x|c=1)p(c=1)}{p(x|c=1)p(c=1) + p(x|c=2)p(c=2)} = \frac{p(x|c=1)}{p(x|c=1) + p(x|c=2)} = \frac{1}{1 + \frac{p(x|c=1)}{p(x|c=2)}}$$

Using the Gaussians, we have

$$\frac{p(x|c=1)}{p(x|c=2)} = e^{-\frac{1}{2\sigma^2}(x-m_1)^2 + \frac{1}{2\sigma^2}(x-m_2)^2} = e^{\frac{1}{2\sigma^2}\left(2x(m_1-m_2) + m_2^2 - m_1^2\right)}$$

so

$$a = -\frac{1}{\sigma^2}(m_1 - m_2), \qquad b = -\frac{m_2^2 - m_1^2}{2\sigma^2}$$

13.5 What's not clear is how 'lucky' WowCo is in finding the solution. In principle, it could attempt a great number of solutions until one just hits on the solution that has minimal test error. The point is that if one is allowed to search repeatedly for an algorithm that gets minimal test error, this is effectively finding an algorithm that is trained to minimise the test error itself. There is no reason to have any confidence that such an algorithm would perform well on any novel problem.

13.6

$$\mathrm{KL}(q|\hat{p}) = \langle \log q \rangle_q - \langle \log \hat{p} \rangle = \langle \log q \rangle_q - \langle \log p \rangle_q - \langle \log \tilde{p} \rangle - \log A \geq 0$$

This gives

$$\log A \geq \langle \log \tilde{p}(y|x) \rangle_{q(x,y)} - \mathrm{KL}(q(x,y)|p(x,y))$$

Then for the particular empirical,

$$q(x,y) = \frac{1}{N}\sum_{n=1}^{N} \delta(x, x^n)\,\delta(y, y^n)$$

the first term reduces to enumerating the integrand at the training points:

$$\log A \geq \frac{1}{N}\sum_{n=1}^{N} \log \tilde{p}(y^n|x^n) - \mathrm{KL}(q(x,y)|p(x,y))$$

We can write

$$p(x,y) = p(y|x)p(x)$$

so that

$$\mathrm{KL}(q(x,y)|p(x,y)) = \langle \log q(x,y) \rangle_{q(x,y)} - \langle \log p(y|x) \rangle_{q(x,y)} - \langle \log p(x) \rangle_{q(x,y)}$$

Similarly, we can write

$$q(x,y) = q(y|x)q(x)$$

so that

$$\mathrm{KL}(q(x,y)|p(x,y)) = \langle \log q(x) \rangle_{q(x)} + \langle \log q(y|x) \rangle_{q(x,y)} - \frac{1}{N}\sum_n \log p(y^n|x^n) - \langle \log p(x) \rangle_{q(x)}$$

Assuming that there is a unique output y for each training input x, then $q(y|x)$ has probability 1 and that $p(y^n|x^n) = 1$, then

$$\log A \geq \frac{1}{N}\sum_{n=1}^{N} \log \tilde{p}(y^n|x^n) - \mathrm{KL}(q(x)|p(x))$$

In the case that the predictor classifies each training point with certainty, the first term is zero. Exponentiating, we obtain

$$A \geq e^{-\mathrm{KL}(q(x)|p(x))}$$

13.7 $p(c|x,y) \propto p(y|c)p(c|x)$.

14.1 See `exerciseNearNeigh.m`.

14.2 To be written.

14.3
1. This can be achieved using validation in which a portion of the training data is used to assess the performance of the classifier for a given number of neighbours.
2. Those elements i corresponding to small w_i will have little impact on the BQ. If we assume that all the pixels are in greyscale (so x_i is positive). Therefore one might be able to explain BQ based on highlighting the image $\mathbf{x}$ with each pixel x_i shaded green for $w_i > 0$ and red for $w_i < 0$. A drawback of this approach is that it is evaluating each pixel individually in terms of its contribution to the BQ. In reality we would like a more global feature (such as say the symmetry of the face) to assess the BQ.

15.2 Because of the symmetry, the covariance matrix is the identity. This means that the principal component is not not well defined – one can take any direction. A suitable one dimensional representation in this case is given by the non-linear mapping $x_i = \cos\theta, y_i = \sin\theta$.

15.3 Let $\mathbf{y}^a = \mathbf{c} + \sum_{i=1}^{M} a_i \mathbf{e}^i$, $\mathbf{y}^b = \mathbf{c} + \sum_{i=1}^{M} b_i \mathbf{e}^i$

$$\left(\mathbf{y}^a - \mathbf{y}^b\right)^2 = \left(\sum_{i=1}^{M}(a_i - b_i)\mathbf{e}^i\right)^2 = \sum_{ij}(a_i - b_i)(a_j - b_j){\mathbf{e}^i}^{\mathsf{T}}\mathbf{e}^j = \sum_i (a_i - b_i)^2$$

15.4

$$\mathbf{S}_{xy}\mathbf{S}_{yy}^{-1}\mathbf{S}_{yx}\mathbf{a} = \lambda^2\mathbf{S}_{xx}\mathbf{a}$$

We can write this as

$$\mathbf{S}_{xy}\mathbf{S}_{yy}^{-\frac{1}{2}}\mathbf{S}_{yy}^{-\frac{1}{2}}\mathbf{S}_{yx}\mathbf{S}_{xx}^{-\frac{1}{2}}\mathbf{S}_{xx}^{\frac{1}{2}}\mathbf{a} = \lambda^2\mathbf{S}_{xx}^{\frac{1}{2}}\mathbf{S}_{xx}^{\frac{1}{2}}\mathbf{a}$$

Multiplying by $\mathbf{S}_{xx}^{-\frac{1}{2}}$ we obtain

$$\underbrace{\mathbf{S}_{xx}^{-\frac{1}{2}}\mathbf{S}_{xy}\mathbf{S}_{yy}^{-\frac{1}{2}}}_{\mathbf{UDV}^\mathsf{T}}\underbrace{\mathbf{S}_{yy}^{-\frac{1}{2}}\mathbf{S}_{yx}\mathbf{S}_{xx}^{-\frac{1}{2}}}_{\mathbf{VDU}^\mathsf{T}}\mathbf{S}_{xx}^{\frac{1}{2}}\mathbf{a} = \lambda^2\mathbf{S}_{xx}^{\frac{1}{2}}\mathbf{a}$$

Defining $\tilde{\mathbf{a}} = \mathbf{S}_{xx}^{\frac{1}{2}}\mathbf{a}$ we have then

$$\mathbf{U}\mathbf{D}^2\mathbf{U}^\mathsf{T}\tilde{\mathbf{a}} = \lambda^2\tilde{\mathbf{a}}$$

This is an eigen-equation. The optimal solution for $\tilde{\mathbf{a}}$ is given by the principal eigen vector $\mathbf{u}_1$, with $\mathbf{D}$ ordered to have $D_{ii} \geq D_{jj}$ for $i < j$. Hence

$$\mathbf{a} = \mathbf{S}_{xx}^{-\frac{1}{2}}\mathbf{u}_1$$

By symmetry, the solution for $\mathbf{b}$ is given by interchanging x and y throughout. This is equivalent to the above on interchanging $\mathbf{V}$ and $\mathbf{U}$, giving the desired result.

15.5 Using the approximations, we have

$$(\mathbf{x}^a - \mathbf{x}^b)^\mathsf{T}\mathbf{S}^{-1}(\mathbf{x}^a - \mathbf{x}^b) \approx (\sum_i a_i\mathbf{e}^i - \sum_i b_i\mathbf{e}^i)^\mathsf{T}\mathbf{S}^{-1}(\sum_j a_j\mathbf{e}^j - \sum_i b_j\mathbf{e}^j) \tag{30.0.105}$$

Due to the orthonormality of the eigenvectors, this is $\sum_i a_i^2/\lambda_i - 2a_ib_i/\lambda_i + b_i^2/\lambda_i = (\mathbf{a}-\mathbf{b})^\mathsf{T}\mathbf{D}^{-1}(\mathbf{a}-\mathbf{b})$ where $\mathbf{D}$ is a diagonal matrix containing the eigenvalues.

15.6 To optimise equation (15.11.6), it is straightforward to show that we should first transform the data to be zero mean : $\sum_n \mathbf{x}^n = \mathbf{0}$ and $\sum_n v_k^n = 0$, $k = 1, \ldots, K$. We may assume, without loss of generality, that the $\mathbf{b}^j$ are orthonormal (since we could rescale the y_j^n if not). However, we cannot assume that the $\mathbf{c}^k$, $k = 1, \ldots, K$ are orthonormal, since we cannot rescale the v. Similarly, we assume nothing, a priori, regarding the relationship between the vectors $\mathbf{b}^j$ and $\mathbf{c}^k$. Differentiating equation (15.11.6) with respect to $\mathbf{w}^n$ gives (using the orthonormality constraint on the $\mathbf{b}^i$)

$$\mathbf{w}^n = \sum_i(\mathbf{b}^i)^\mathsf{T}\left(\mathbf{x}^n - \sum_l v_l^n\mathbf{c}^l\right) \tag{30.0.106}$$

The residual vector (difference between $\mathbf{x}^n$ and the linear reconstruction) is then

$$\mathbf{r}^n = \mathbf{x}^n - \sum_i(\mathbf{b}^i)^\mathsf{T}\left(\mathbf{x}^n - \sum_l v_l^n\mathbf{c}^l\right)\mathbf{b}^i - \sum_j v_j^n\mathbf{c}^j \tag{30.0.107}$$

By defining $\tilde{B} \equiv I - \sum_i \mathbf{b}^i(\mathbf{b}^i)^\mathsf{T} \equiv \mathbf{I} - \mathbf{U}\mathbf{U}^\mathsf{T}$, (using the notation of the previous section), the residual is

$$\mathbf{r}^n = \tilde{B}\left(\mathbf{x}^n - \sum_j v_j^n\mathbf{c}^j\right) \tag{30.0.108}$$

Differentiating $E = \sum_n(\mathbf{r}^n)^\mathsf{T}\mathbf{r}^n$ with respect to $\mathbf{c}^i$, we get

$$\sum_n v_i^n\tilde{B}\mathbf{x}^n = \sum_j\sum_n v_j^n v_i^n\tilde{B}\mathbf{c}^j \tag{30.0.109}$$

Define

$$[\tilde{V}]_{ij} = \sum_n v_i^n v_j^n, \qquad [\tilde{\mathbf{X}}]_{ij} = \sum_n v_i^n\mathbf{x}_j^n, \qquad \mathbf{C} = [\mathbf{c}^1, \ldots, \mathbf{c}^K] \tag{30.0.110}$$

Then the above has solution

$$\mathbf{C} = \tilde{\mathbf{X}}\tilde{V}^{-1} \tag{30.0.111}$$

Hence, the objective involves the matrix

$$\tilde{S} = \sum_n(\mathbf{x}^n - \mathbf{d}^n)(\mathbf{x}^n - \mathbf{d}^n)^\mathsf{T} \tag{30.0.112}$$

where

$$\mathbf{d}^n = \sum_j v_j^n\mathbf{c}^j \tag{30.0.113}$$

Hence, the optimal solution is given by taking the principal eigenvectors of $\tilde{S}$, with $\mathbf{C}$ set as above.

15.7 See `exercisePCA2D.m`.

15.8

$$\tilde{\mathbf{B}} = \mathbf{U}\mathbf{D}\mathbf{V}^\mathsf{T}\mathbf{V}\mathbf{D}^{-1} = \mathbf{U}$$

Since $\mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{I}$, the result follows.

15.9 Let $\tilde{p}(x, y, z) = \tilde{p}(x|z)\tilde{p}(z|y)$,

$$\tilde{p}(x|y) = \frac{\sum_z \tilde{p}(x|z)\tilde{p}(z|y)}{\sum_{xz} \tilde{p}(x|z)\tilde{p}(z|y)}$$

and consider

$$\sum_y \mathrm{KL}(p(x|y)|\tilde{p}(x|y)) = \sum_y \left(\langle \log p\rangle_p - \langle \log \tilde{p}\rangle_p\right)$$

Since p is fixed, minimising the Kullback-Leibler divergence with respect to the approximation $\tilde{p}$ is equivalent to maximising the 'likelihood' term $\langle \log \tilde{p}\rangle_p$. This is

$$\sum_{x,y} p(x|y) \log \tilde{p}(x|y)$$

Consider

$$\mathrm{KL}(q(z|x, y)|\tilde{p}(z|x, y)) = \sum_z q(z|x, y) \log q(z|x, y) - \sum_z q(z|x, y) \log \tilde{p}(z|x, y) \geq 0$$

where $\sum_z$ implies summation over all states of the variable z. Using

$$\tilde{p}(z|x, y) = \frac{\tilde{p}(x, y, z)}{\tilde{p}(x, y)}$$

and rearranging, this gives the bound,

$$\log \tilde{p}(x|y) \geq -\sum_z q(z|x, y) \log q(z|x, y) + \sum_z q(z|x, y) \log \tilde{p}(z, x, y)$$

Plugging this into the 'likelihood' term above, we have the bound

$$\sum_{x,y} p(x|y) \log \tilde{p}(x|y) \geq -\sum_{x,y} p(x|y) \sum_z q(z|x, y) \log q(z|x, y) + \sum_{x,y} p(x|y) \sum_z q(z|x, y) \left[\log \tilde{p}(x|z) + \log \tilde{p}(z|y)\right]$$

M-step

Optimally,

$$\tilde{p}(x|z) \propto \sum_y p(x|y)q(z|x, y)$$

and similarly,

$$\tilde{p}(z|y) \propto \sum_x p(x|y)q(z|x, y)$$

E-step

The optimal setting for the q distribution at each iteration is

$$q(z|x, y) = \tilde{p}(z|x, y)$$

which is fixed throughout the M-step.

16.1 In this case, the matrix $\mathbf{B}$ is not formally invertible. A simple solution is to constrain $\mathbf{w}$ to lie in the space spanned by the data. This can be achieved by finding a basis for the data, as in section(16.3.1). The derivation is analogous, and one obtains

$$\mathbf{w} \propto \mathbf{Q}\left(\mathbf{Q}^\mathsf{T}\mathbf{B}\mathbf{Q}\right)^{-1}\mathbf{Q}^\mathsf{T}(\mathbf{m}_1 - \mathbf{m}_2) \tag{30.0.114}$$

16.2 See `exerciseCanonVarDigits.m`.

16.3 Zero derivative conditions are

$$\sum_{n_1}\left(y_1 - b - \mathbf{w}^\mathsf{T}\mathbf{x}_{n_1}\right) + \sum_{n_2}\left(y_2 - b - \mathbf{w}^\mathsf{T}\mathbf{x}_{n_2}\right) = 0 \tag{30.0.115}$$

and

$$\sum_{n_1}\left(y_1 - b - \mathbf{w}^\mathsf{T}\mathbf{x}_{n_1}\right)\mathbf{x}_{n_1}^\mathsf{T} + \sum_{n_2}\left(y_2 - b - \mathbf{w}^\mathsf{T}\mathbf{x}_{n_2}\right)\mathbf{x}_{n_2}^\mathsf{T} = 0 \tag{30.0.116}$$

Let $N = N_1 + N_2$. Then

$$b = \frac{N_1 y_1 + N_2 y_2}{N_1 + N_2} - \mathbf{w}^\mathsf{T}\mathbf{m} \tag{30.0.117}$$

where $\mathbf{m}$ is the mean of all the data. Under the setting of y_1 and y_2, this gives

$$b = -\mathbf{w}^\mathsf{T}\mathbf{m} \tag{30.0.118}$$

Plugging this into the zero derivative for $\mathbf{w}$ we have

$$N_1 y_1 \mathbf{m}_1 + N_2 y_2 \mathbf{m}_2 = \mathbf{w}^\mathsf{T}\left(N\mathbf{B} + N_1\mathbf{m}_1\mathbf{m}_1^\mathsf{T} + +N_2\mathbf{m}_2\mathbf{m}_2^\mathsf{T} - N_1\mathbf{m}_1\mathbf{m}^\mathsf{T} - N_2\mathbf{m}_2\mathbf{m}^\mathsf{T}\right) \tag{30.0.119}$$

Now using $\mathbf{m} - \mathbf{m}_1 = (\mathbf{m}_2 - \mathbf{m}_1)N_2/N$ and the setting for y_1, y_2 we have

$$N\left(\mathbf{m}_1 - \mathbf{m}_2\right) = \left(N\mathbf{B} + \frac{N_1 N_2}{N}\left(\mathbf{m}_1 - \mathbf{m}_2\right)\left(\mathbf{m}_1 - \mathbf{m}_2\right)^\mathsf{T}\right)\mathbf{w} \tag{30.0.120}$$

which can be written as

$$f(\mathbf{w})\left(\mathbf{m}_1 - \mathbf{m}_2\right) = \mathbf{B}\mathbf{w} \tag{30.0.121}$$

for some scalar function $f(\mathbf{w})$. This means that

$$\mathbf{w} \propto \mathbf{B}^{-1}\left(\mathbf{m}_1 - \mathbf{m}_2\right) \tag{30.0.122}$$

which is Fisher's solution.

16.4 See `exerciseCanonVarDigits57.m`. Interestingly, the classification performance of the CCA projected data is slightly worse than for PCA. For projecting down to a very low number of dimensions, say 2 or 3, then CCA performs better. A possible explanation is that whilst CCA attempts to find well separated data, it is not designed to find a projection that is explicitly tailored for KNN. In this sense, there exists potentially more suitable projections.

16.5 Since $B(w)$ is positive, we have

$$\frac{A(w)}{B(w)} > \lambda^{old} = \frac{A(w^{old})}{B(w^{old})} \tag{30.0.123}$$

Hence, this procedure cannot decrease the quotient $F(w)$. This suggests a simple algorithm:

- Initialise w^1 randomly.
- (1) Set $\lambda^i \equiv \frac{A(w^i)}{B(w^i)}$
- (2) find $w^{i+1} = \arg\max_w A(w) - \lambda^i B(w)$
- Repeat (1) and (2) to convergence.

17.1

1. In two dimensions, $(0,0), (1,0)$ in class 1 and $(2,0)$ in class 2.
2. Data that is linearly independent is linearly separable, as discussed in section(17.4.1).

17.2 From equation (17.1.5) we see that if the y data has zero mean, and x data has zero mean, then $a = 0$. Hence, if we were to shift the data so that it has zero mean, then the ordinary least squares fit must go through zero – in other words it goes through the mean in the non-shifted data. We showed in chapter(15) that PCA goes through the mean of the data.

17.3

1. When $C = 2$ we have

$$p(c=1|\mathbf{x}) = \frac{e^{\mathbf{w}_1^\mathsf{T}\mathbf{x}}}{e^{\mathbf{w}_1^\mathsf{T}\mathbf{x}} + e^{\mathbf{w}_2^\mathsf{T}\mathbf{x}}} = \frac{1}{1 + e^{(\mathbf{w}_2 - \mathbf{w}_1)^\mathsf{T}\mathbf{x}}} = \sigma\left(\mathbf{w}_2 - \mathbf{w}_1{}^\mathsf{T}\mathbf{x}\right)$$

For $C = 2$, this is therefore logistic regression parameterised by the difference $\mathbf{w} \equiv \mathbf{w}_2 - \mathbf{w}_1$. Note that there is therefore redundancy in the softmax parameterisation since only this difference plays a role in computing the probability. The same hold for $C > 2$ – by multiplying and dividing by $\exp(-\mathbf{w}_d^\mathsf{T}\mathbf{x})$, for some chosen $d \in \{1, \ldots, C\}$ only the differences $\mathbf{w}'_c \equiv \mathbf{w}_c - \mathbf{w}_d$ are required to specify the probability. This is potentially important to realise for optimisation purposes since there will therefore be equivalent solutions in the redundant parameterisation.

2.

$$L(\mathcal{D}) = \sum_n \left[\mathbf{w}_{c^n}^\mathsf{T} \mathbf{x}^n - \log \sum_{d=1}^{C} e^{\mathbf{w}_d^\mathsf{T} \mathbf{x}^n} \right]$$

3.

$$\frac{\partial}{\partial \mathbf{w}_c} L = \sum_n \left[\mathbb{I}\left[c^n = c\right] \mathbf{x}^n - \frac{\mathbf{x}^n e^{\mathbf{w}_c^\mathsf{T}\mathbf{x}^n}}{\sum_{d=1}^{C} e^{\mathbf{w}_d^\mathsf{T}\mathbf{x}^n}} \right]$$

$$\frac{\partial^2}{\partial \mathbf{w}_c \partial \mathbf{w}_{c'}^\mathsf{T}} L = -\sum_n \mathbf{x}^n \left(\mathbf{x}^n\right)^\mathsf{T} \frac{Z_n \delta(c,c') e^{\mathbf{w}_c^\mathsf{T}\mathbf{x}^n} - e^{\mathbf{w}_c^\mathsf{T}\mathbf{x}^n} e^{\mathbf{w}_{c'}^\mathsf{T}\mathbf{x}^n}}{Z_n^2}$$

where $Z_n \equiv \sum_{d=1}^{C} e^{\mathbf{w}_d^\mathsf{T}\mathbf{x}^n}$. Defining $\pi_c^n = p(c^n|\mathbf{x}^n)$, we have

$$\frac{\partial^2}{\partial \mathbf{w}_c \partial \mathbf{w}_{c'}^\mathsf{T}} L = -\sum_n \mathbf{x}^n \left(\mathbf{x}^n\right)^\mathsf{T} \left(\delta(c,c')\pi_c^n - \pi_c^n \pi_{c'}^n\right)$$

Then for a vector

$$\mathbf{z} = \begin{pmatrix} \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_C \end{pmatrix}$$

$$\begin{aligned}
\mathbf{z}^\mathsf{T}\mathbf{H}\mathbf{z} &= \sum_c \mathbf{z}_c \mathbf{H}_{cc} \mathbf{z}_c + \sum_{c \neq c'} \mathbf{z}_c \mathbf{H}_{cc'} \mathbf{z}_{c'} \\
&= -\sum_n \left[\sum_c \left(\mathbf{z}_c\right)^\mathsf{T} \mathbf{x}^n \left(\mathbf{x}^n\right)^\mathsf{T} \mathbf{z}_c \left(\pi_c^n - (\pi_c^n)^2\right) - \sum_{c\neq c'} \left(\mathbf{z}_c\right)^\mathsf{T} \mathbf{x}^n \left(\mathbf{x}^n\right)^\mathsf{T} \left(\mathbf{z}_{c'}\right) \pi_c^n \pi_{c'}^n \right]
\end{aligned}$$

Let $y_c^n \equiv \left(\mathbf{x}^n\right)^\mathsf{T} \mathbf{z}_c$. Then

$$\begin{aligned}
\mathbf{z}^\mathsf{T}\mathbf{H}\mathbf{z} &= -\sum_n \left[\sum_c \left(y_c^n\right)^2 \left(\pi_c^n - (\pi_c^n)^2\right) - \sum_{c\neq c'} y_c^n y_{c'}^n \pi_c^n \pi_{c'}^n \right] \\
&= \sum_n \left[-\sum_c \left(y_c^n\right)^2 \pi_c^n + \sum_{c,c'} y_c^n y_{c'}^n \pi_c^n \pi_{c'}^n \right]
\end{aligned}$$

Since π_c^n is a distribution in c, we have

$$\mathbf{z}^\mathsf{T}\mathbf{H}\mathbf{z} = -\sum_n \left(\left\langle (y^n)^2 \right\rangle - \left\langle y^n \right\rangle^2 \right) \leq 0$$

The last step follows since each variance is ≥ 0.

17.4 Using

$$C\xi^n = \alpha^n$$

Then

$$\xi^n = \alpha^n / C \to \sum_n \left(\xi^n\right)^2 = \sum_n (\alpha^n)^2 / C^2$$

and

$$\sum_n \alpha^n \xi^n = \frac{1}{C} \sum_n (\alpha^n)^2$$

Also

$$\sum_n \alpha y^n \mathbf{w}^\mathsf{T} \mathbf{x}^n = \mathbf{w}^\mathsf{T}\mathbf{w}$$

Hence the objective is to optimise

$$(\frac{1}{2} - 1)\mathbf{w}^\mathsf{T}\mathbf{w} + \frac{1}{C}(\frac{1}{2} - 1) \sum_n (\alpha^n)^2 + \sum_n \alpha^n$$

Using $\mathbf{w} = \sum_n \alpha^n y^n \mathbf{x}^n$,

$$\mathbf{w}^\mathsf{T}\mathbf{w} = \sum_n \alpha^n y^n \mathbf{x}^n \sum_m \alpha^m y^m \mathbf{x}^m = \sum_{n,m} y^n y^m \alpha^n \alpha^m,$$

we obtain the desired form. Note that provided $C \geq 0$, the constraint $\xi^n \geq 0$ is automatically satisfied since $\xi^n = \alpha^n / C \geq 0$. Also $L(\alpha)$ is negative semidefinite wrt α so that the Lagrange stationary requirement means that we need to find a minimum *w.r.t.* α.

17.5

$$\mathbf{y} = \mathbf{M}\mathbf{x} \tag{30.0.124}$$

The argument of the sigmoid $\sigma(h)$ is

$$h = \mathbf{w}^\mathsf{T}\acute{\mathbf{x}} + b = \mathbf{w}^\mathsf{T}\mathbf{M}\acute{\mathbf{x}} + b = (\mathbf{M}^\mathsf{T}\mathbf{w})^\mathsf{T}\mathbf{x} + b = \tilde{\mathbf{w}}^\mathsf{T}\mathbf{x} + b \tag{30.0.125}$$

PCA is a linear projection of the data and, since the argument of the logistic function is also a linear function of the data, overall using PCA first results still in a linear decision boundary. However, the problem is constrained since we have forced linear regression to work in the subspace spanned by the PCA vectors. Consider 100 training vectors randomly positioned in a 1000 dimensional space each with a random class 0 or 1. With very high probability, these 100 vectors will be linearly separable. Now project these vectors onto a 10 dimensional space: with very high probability, 100 vectors plotted in a 10 dimensional space will *not* be linearly separable. Hence, attention should should be paid to the fact that using PCA first could potentially transform a linearly separable problem into a non-linearly separable problem.

17.6

$$\sigma^{-1}(x) = \log\left(\frac{\sigma(x)}{\sigma(x)(1-\sigma(x))}\right)$$

17.7

$$L = \sum_n c^n \log\sigma(\mathbf{w}^\mathsf{T}\mathbf{x}^n + b) + (1-c^n)\log(1-\sigma(\mathbf{w}^\mathsf{T}\mathbf{x}^n + b))$$

Using the derivative result $\sigma' = \sigma(1-\sigma)$, and $\sigma_n \equiv \sigma(\mathbf{w}^\mathsf{T}\mathbf{x}^n + b)$ we have

$$\nabla_{\mathbf{w}} L = \sum_{n=1} c^n(1-\sigma_n)\mathbf{x}^n - (1-c^n)\sigma_n\mathbf{x}^n = \sum_{n=1}(c^n - \sigma_n)\mathbf{x}^n$$

17.8

1. Since the data is linearly separable, for class +1 data and class −1 data

   $$\mathbf{w}^\mathsf{T}\mathbf{x}_+ + b \geq 0$$

   and

   $$\mathbf{w}^\mathsf{T}\mathbf{x}_- + b \leq 0$$

   Multiplying by a positive scalar preserves the inequalities:

   $$\lambda\mathbf{w}^\mathsf{T}\mathbf{x}_+ + \lambda b \geq 0$$

   and

   $$\lambda\mathbf{w}^\mathsf{T}\mathbf{x}_- + \lambda b \leq 0$$

   which demonstrates the required result.

2. In terms of maximum likelihood this means that we can make $\mathbf{w}$ and b as large as we like, once we have a separable solution. For logistic regression, the most probable settings are when the σ functions saturate to 1 (or 0 in the case of negative datapoints). In order for this to happen the argument $\mathbf{w}^\mathsf{T}\mathbf{x} + b$ needs to go to $\pm\infty$, which means that $\mathbf{w}$ and b need to become infinite. This means that for any incremental training procedure for maximum likelihood, this will never terminate since the weights will continue to increase without bound. We therefore need to introduce a termination criterion.

17.9 In general, we may assume that the data is linearly separable since there are N points in an N dimensional space. For a given set of $\epsilon^n > 0$, we can therefore set $b = 0$ and use $\mathbf{w} = \mathbf{X}^{-\mathsf{T}}\boldsymbol{\epsilon}$ where $\mathbf{X} = \left[\mathbf{x}^1, \ldots, \mathbf{x}^N\right]$. This will result in a finite length $\mathbf{w}$. This cannot be equivalent to the maximum likelihood solution for logistic regression, which would in principal have infinite length.

17.10

1. $$L = \sum_n c^n \log\sigma(f(\mathbf{x}^n)) + (1-c^n)\log(1-\sigma(f(\mathbf{x}^n)))$$

   where

   $$f(\mathbf{x}) \equiv v_0 + v_1 g(\mathbf{w}_1^\mathsf{T}\mathbf{x} + b_1) + v_2 g(\mathbf{w}_2^\mathsf{T}\mathbf{x} + b_2)$$

2. $$\frac{\partial}{\partial\theta}L = \sum_n \left(c^n(1-\sigma^n) - (1-c^n)\sigma^n\right)\frac{\partial}{\partial\theta}f(\mathbf{x}^n) = \sum_n \left(c^n - \sigma^n\right)\frac{\partial}{\partial\theta}f(\mathbf{x}^n)$$

   $$\frac{\partial}{\partial\mathbf{w}_i}f(\mathbf{x}^n) = -v_i(\mathbf{w}_i^\mathsf{T}\mathbf{x}^n + b_i)e^{-(\mathbf{w}_i^\mathsf{T}\mathbf{x}^n + b_i)^2/2}\mathbf{x}^n$$

   For

   $$\frac{\partial}{\partial b_i}f(\mathbf{x}^n) = -v_i(\mathbf{w}_i^\mathsf{T}\mathbf{x}^n + b_i)e^{-(\mathbf{w}_i^\mathsf{T}\mathbf{x}^n + b_i)^2/2}$$

   For $i = 1, 2$

   $$\frac{\partial}{\partial v_i}f(\mathbf{x}^n) = g(\mathbf{w}_i^\mathsf{T}\mathbf{x}^n + b_i)$$

   $$\frac{\partial}{\partial v_0}f(\mathbf{x}^n) = 1$$

3. This is a non-linear form of logistic regression.

4. Given a trained set of parameters, the decision boundary is given by those $\mathbf{x}$ that satisfy:

$$0 = v_0 + v_1 g(\mathbf{w}_1^\mathsf{T}\mathbf{x} + b_1) + v_2 g(\mathbf{w}_2^\mathsf{T}\mathbf{x} + b_2)$$

Given any solution $\mathbf{x}$, then $\mathbf{x}+\mathbf{w}_1^\perp+\mathbf{w}_2^\perp$, where $\mathbf{w}_1^\perp$ is orthogonal to $\mathbf{w}_1$ and $\mathbf{w}_2^\perp$ is orthogonal to $\mathbf{w}_2$ will also satisfy this equation. This means that the decision boundary will be an $N-2$ dimensional non-linear manifold.

18.1 1.

$$p(f|\mathbf{x}) = \int\int \delta(f - \mathbf{w}^\mathsf{T}\mathbf{x})p(\mathbf{w})d\mathbf{w}$$

From our general results, in chapter(8), we know that f is Gaussian distributed since it is linear in $\mathbf{w}$ and $\mathbf{w}$ itself if Gaussian. The mean and variance are given by

$$\langle f\rangle = \mathbf{x}^\mathsf{T}\langle \mathbf{w}\rangle = 0, \quad \langle f^2\rangle = \mathbf{x}^\mathsf{T}\left\langle \mathbf{w}\mathbf{w}^\mathsf{T}\right\rangle\mathbf{x} = \mathbf{x}^\mathsf{T}\mathbf{\Sigma}\mathbf{x}$$

2. This is a Gaussian distribution, which we can find from conditioning:

$$\langle t\rangle = \langle f\rangle = 0, \quad \langle t^2\rangle = \langle f^2\rangle + \sigma^2, \langle tf\rangle = \langle f^2\rangle$$

Using Gaussian conditioning (the partitioned Gaussian results),

$$p(f|t,\mathbf{x}) = \mathcal{N}\left(f\middle|(t\langle f^2\rangle/(\langle f^2\rangle+\sigma^2), \langle f^2\rangle - \langle f^2\rangle\langle f^2\rangle/(\langle f^2\rangle+\sigma^2)\right)$$

18.3

$$p(\mathbf{y}_{val}|\mathcal{X}_{train},\mathcal{Y}_{train},\mathcal{X}_{val}) = \int p(\mathbf{y}_{val}|\mathcal{X}_{val},\mathbf{w})p(\mathbf{w}|\mathcal{X}_{train},\mathcal{Y}_{train})$$

Using

$$\mathbf{\Phi}_{val}^\mathsf{T} = \left[\boldsymbol{\phi}(\mathbf{x}_{val}^1),\ldots,\boldsymbol{\phi}(\mathbf{x}_{val}^M)\right]$$

we can write

$$\mathbf{y}_{val} = \mathbf{\Phi}_{val}\mathbf{w} + \boldsymbol{\eta}_{val}$$

Since this is linearly related to $\mathbf{w}$ which itself is Gaussian, then $\mathbf{y}_{val}$ is Gaussian. It's mean is

$$\langle \mathbf{y}_{val}\rangle = \mathbf{\Phi}_{val}\underbrace{\langle\mathbf{w}\rangle}_{\mathbf{m}} + \underbrace{\langle\boldsymbol{\eta}_{val}\rangle}_{\mathbf{0}}$$

The covariance is obtained from

$$\left\langle [\mathbf{y}_{val} - \langle\mathbf{y}_{val}\rangle][\mathbf{y}_{val} - \langle\mathbf{y}_{val}\rangle]^\mathsf{T}\right\rangle = \mathbf{\Phi}_{val}\left\langle(\mathbf{w}-\mathbf{m})(\mathbf{w}-\mathbf{m})^\mathsf{T}\right\rangle\mathbf{\Phi}_{val}^\mathsf{T} + \left\langle\boldsymbol{\eta}_{val}\boldsymbol{\eta}_{val}^\mathsf{T}\right\rangle$$

The cross terms involving $\langle\boldsymbol{\eta}_{val}\mathbf{w}\rangle$ are zero since $\boldsymbol{\eta}_{val}$ and $\mathbf{w}$ are uncorrelated and $\boldsymbol{\eta}_{val}$ has zero mean. Thus

$$\mathbf{C}_{val} = \mathbf{\Phi}_{val}\mathbf{S}\mathbf{\Phi}_{val}^\mathsf{T} + \sigma^2\mathbf{I}_M$$

Hence

$$p(\mathbf{y}_{val}|\mathcal{X}_{train},\mathcal{Y}_{train},\mathcal{X}_{val}) = \mathcal{N}(\mathbf{y}_{val}|\mathbf{\Phi}_{val}\mathbf{m},\mathbf{C}_{val})$$

Taking logs gives the required result.

18.4 1.

$$\frac{\partial}{\partial w_i}E = \alpha w_i - \sum_n \frac{1}{\sigma_n}\sigma_n(1-\sigma_n)h_i^n = \alpha w_i - \sum_n(1-\sigma_n)h_i^n$$

$$\frac{\partial^2}{\partial w_i\partial w_j}E = \alpha\delta_{ij} + \sum_n \sigma_n(1-\sigma_n)h_i^n h_j^n = \left[\alpha\mathbf{I} + \sum_{n=1}^N \sigma^n(1-\sigma^n)\mathbf{h}^n(\mathbf{h}^n)^\mathsf{T}\right]_{ij}$$

Since $\mathbf{h}^n = (2c^n-1)\boldsymbol{\phi}^n$

$$\mathbf{h}^n(\mathbf{h}^n)^\mathsf{T} = \underbrace{(2c^n-1)^2}_{1}\boldsymbol{\phi}^n(\boldsymbol{\phi}^n)^\mathsf{T}$$

This gives the desired result.

2. Since $\sigma(1-\sigma) \geq 0$ all data terms in the Hessian are positive sums of outer products of vectors and hence positive semidefinite. The additional prior term $\alpha\mathbf{I}$ is positive definite, so that $\mathbf{H}$ is overall positive definite.

18.5

$$\int f(\mathbf{x}^\mathsf{T}\mathbf{w})p(\mathbf{w})d\mathbf{w} = \int f(\mathbf{x}^\mathsf{T}vw)p(h,\mathbf{w})d\mathbf{w}dh = \int f(\mathbf{x}^\mathsf{T}\mathbf{w})p(h|\mathbf{w})p(\mathbf{w})d\mathbf{w}dh \tag{30.0.126}$$

$$= \int f(\mathbf{x}^\mathsf{T}\mathbf{w})\delta(h-\mathbf{x}^\mathsf{T}\mathbf{w})p(\mathbf{w})d\mathbf{w}dh = \int f(h)\left\{\delta(h-\mathbf{x}^\mathsf{T}\mathbf{w})p(\mathbf{w})d\mathbf{w}\right\}dh = \int f(h)p(h)dh \tag{30.0.127}$$

18.6

$$\frac{\partial}{\partial\alpha}L = -\frac{1}{2}(\mathbf{w}^*)^\mathsf{T}\mathbf{w}^* - \frac{1}{2}\text{trace}\left((\alpha\mathbf{I}+\mathbf{J})^{-1}\right) + \frac{B}{2\alpha}$$

To find the optimum of $L(\alpha)$ we set the derivative to zero and solve for α. This gives that the optimum α satisfies:

$$\alpha = \frac{N}{(\mathbf{w}^*)^\mathsf{T}\mathbf{w}^*} + \text{trace}\left((\alpha\mathbf{I}+\mathbf{J})^{-1}\right)$$

We can make a fixed point equation from this, as described in the text.

19.1 1.

$$\sum_{ij} y_i K^+_{ij} y_j = \sum_{ij} y_i\left(K^1_{ij}+K^2_{ij}\right)y_j = \underbrace{\sum_{ij} y_i K^1_{ij} y_j}_{\geq 0} + \underbrace{\sum_{ij} y_i K^2_{ij} y_j}_{\geq 0} \geq 0$$

2.

$$\sum_{ij} y_i K^*_{ij} y_j = \sum_{lm}\sum_{ij} y_i u_{il} u_{jl} v_{im} v_{jm} y_j = \sum_{lm} \underbrace{\left(\sum_i y_i u_{il} v_{im}\right)}_{z_{lm}} \underbrace{\left(\sum_j u_{jl} v_{jm} y_j\right)}_{z_{lm}} \geq 0$$

19.2

$$\sum_{ij} y_i S_{ij} y_j = \sum_{ij}\left(\frac{1}{N}\sum_n x^n_i y_i x^n_j y_j - \frac{1}{N^2}\sum_n x^n_i y_i \sum_m x^m_j y_j\right)$$

Let $z_n \equiv \sum_n y_i x^n_i$. Then

$$\sum_{ij} y_i S_{ij} y_j = \frac{1}{N}\sum_n z_n^2 - \frac{1}{N^2}\sum_n z_n \sum_m z_m = \frac{1}{N}\sum_n\left(z_n - \frac{1}{N}\sum_m z_m\right)^2 \geq 0$$

19.3 Consider the vector

$$\mathbf{u}(x) = \lambda\begin{pmatrix}\cos\omega x\\ \sin\omega x\end{pmatrix}$$

Then a valid kernel is given by

$$k(x,x') = e^{-|\mathbf{u}(x)-\mathbf{u}(x')|}$$

We can simplify the exponent as follows:

$$\begin{aligned}|\mathbf{u}(x)-\mathbf{u}(x')|^2 &= \mathbf{u}(x)^\mathsf{T}\mathbf{u}(x) + \mathbf{u}(x')^\mathsf{T}\mathbf{u}(x') - 2\mathbf{u}(x)^\mathsf{T}\mathbf{u}(x')\\ &= 2\lambda^2\left(1-\mathbf{u}(x)^\mathsf{T}\mathbf{u}(x')\right) = 2\lambda^2\left(1-\cos(\omega(x-x'))\right)\\ &= 4\lambda^2\sin^2\left(\frac{\omega(x-x')}{2}\right)\end{aligned}$$

Hence, setting $\lambda = 1/2$ and $\omega = 2$, we obtain the covariance function

$$k(x-x') = e^{-|\sin(x-x')|}$$

19.4 [Renato Vicente]

$$e^{x_i x_j} = \sum_{k=0}^\infty \frac{1}{k!}(x_i x_j)^k = \sum_{k=0}^\infty \left(\frac{x_i}{\sqrt{k!}}^{1/k}\right)^k \left(\frac{x_j}{\sqrt{k!}}^{1/k}\right)^k \tag{30.0.128}$$

Note that this expansion has infinite radius of convergence. Using this expansion, we have

$$f(x_i,x_j) = \sum_{k=0}^\infty e^{-\frac{1}{2}x_i^2}\left(\frac{x_i}{\sqrt{k!}^{1/k}}\right)^k \underbrace{\left(\frac{x_j}{\sqrt{k!}^{1/k}}\right)^k e^{-\frac{1}{2}x_j^2}}_{g_k(x_j)} \tag{30.0.129}$$

Thus

$$\sum_{ij} z_i f(x_i,x_j) z_j = \sum_{k=0}^\infty\left(\sum_i z_i g_k(x_i)\right)\left(\sum_j z_j g_k(x_j)\right) = \sum_{k=0}^\infty\left(\sum_i z_i g_k(x_i)\right)^2 \geq 0 \tag{30.0.130}$$

This construction also gives an explicit infinite dimensional feature representation, corresponding to a non-orthogonal basis.

19.5 Since the product of two covariance functions is a covariance function, by repeatedly applying this rule, we have that the integer power of a covariance function, $k^n(x,x')$ is a covariance function. Also, since the sum of two covariance functions is a covariance function and $\lambda k(x,x')$ is a covariance function for $\lambda \geq 0$, then the power series

$$\sum_n \lambda_n k^n(x,x')$$

is a covariance function for $\lambda_n \geq 0$. Since

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

Then $e^{k(x,x')}$ is expressible as an infinite sum of positive coefficient powers of the covariance function $k(x,x')$ and hence itself a covariance function. The same holds for $\tan k_1(x,x')$ since

$$\tan x = x + \frac{x^3}{3} + \frac{2x^5}{15} + \dots$$

19.6 By defining $\mathbf{y} = \mathbf{A}^{1/2}\mathbf{x}$, $\mathbf{y}' = \mathbf{A}^{1/2}\mathbf{x}'$,

$$k_2(\mathbf{x},\mathbf{x}') = f\left((\mathbf{y}-\mathbf{y}')^{\mathsf{T}}(\mathbf{y}-\mathbf{y}')\right) = k_1(\mathbf{y},\mathbf{y}')$$

Since k_1 is a covariance function, so is k_2.

19.8
1. First we define a set of substrings of interest, for example 'goal', 'score', 'policy', 'fiscal', *etc.* , and a set of positive weights $\mathbf{w}$ for these substrings. Then for two documents x and x', we can compute the covariance function $k(x,x')$ using the string kernel. This can then be used as part of the standard GP classifier approach, when faced with a novel example x^*.
2. The Laplace method approximates the log likelihood of the training data, as in equation (19.5.30). The $\mathbf{w}$ can then be treated as hyperparameters, as in section(19.5.3). The derivatives are obtained using the formula in exercise(19.7), where

$$\frac{\partial}{\partial w_s}\mathbf{K}_{\mathbf{x},\mathbf{x}} = \phi_s(x_i)\phi_s(x_j)$$

19.9

$$p(\mathbf{y}^*|\mathbf{x}^*,\mathcal{X},\mathcal{Y}) \propto p(\mathbf{y}^*,\mathcal{Y}|\mathbf{x}^*,\mathcal{X})$$

This requires us to specify a Gaussian covariance over the joint outputs for two difference $\mathbf{y}$, $\mathbf{y}'$. Since the dimensions are assumed independent, then

$$p(\mathbf{y},\mathbf{y}'|\mathbf{x},\mathbf{x}') = \prod_i p(y_i,y_i'|\mathbf{x},\mathbf{x}')$$

Hence we can form a GP predictor based on $p(y_i,y_i'|\mathbf{x},\mathbf{x}')$ for each input dimension alone.

19.10
1. By definition, $\mathbf{x}_1$ is Gaussian. Then $p(\mathbf{x}_2,\mathbf{x}_1) = p(\mathbf{x}_2|\mathbf{x}_1)p(\mathbf{x}_1)$ is also jointly Gaussian since $\mathbf{x}_2$ depends linearly on $\mathbf{x}_1$, $\mathbf{x}_2 = \mathbf{A}\mathbf{x}_1 + \boldsymbol{\eta}_2$. Since the sum of two Gaussian distributed variables is Gaussian distributed, then $\mathbf{x}_2$ is Gaussian distributed. Hence $p(\mathbf{x}_2,\mathbf{x}_1)$ is Gaussian. By repeating this, one sees that $\mathbf{x}_1,\dots,\mathbf{x}_t$ is jointly Gaussian.
2. By iterating, we have

$$\mathbf{x}_t = \mathbf{A}\left(\mathbf{A}\mathbf{x}_{t-2} + \boldsymbol{\eta}_{t-1}\right) + \boldsymbol{\eta}_t = \mathbf{A}^2\mathbf{x}_{t-2} + \mathbf{A}\boldsymbol{\eta}_{t-1} + \boldsymbol{\eta}_t = \mathbf{A}^{t-1}\mathbf{x}_1 + \sum_{\tau=2}^{t}\mathbf{A}^{t-\tau}\boldsymbol{\eta}_\tau$$

Then

$$\langle \mathbf{x}_{t'}\mathbf{x}_t\rangle = \left\langle \left[\mathbf{A}^{t'-1}\mathbf{x}_1 + \sum_{\tau=2}^{t'}\mathbf{A}^{t'-\tau}\boldsymbol{\eta}_\tau\right]\left[\mathbf{A}^{t-1}\mathbf{x}_1 + \sum_{\rho=2}^{t}\mathbf{A}^{t-\rho}\boldsymbol{\eta}_\rho\right]^{\mathsf{T}}\right\rangle$$

Since $\langle \mathbf{x}\boldsymbol{\eta}_\tau^{\mathsf{T}}\rangle = \mathbf{0}$ and $\langle \boldsymbol{\eta}_\tau\boldsymbol{\eta}_\rho^{\mathsf{T}}\rangle = \sigma^2\mathbf{I}\delta_{\tau,\rho}$, this reduces to

$$\langle \mathbf{x}_{t'}\mathbf{x}_t\rangle = \mathbf{A}^{t'-1}\boldsymbol{\Sigma}\left(\mathbf{A}^{t-1}\right)^{\mathsf{T}} + \sigma^2\sum_{\tau=2}^{t'}\sum_{\rho=2}^{t}\delta_{\tau,\rho}\mathbf{A}^{t'-\tau}\left(\mathbf{A}^{t-\rho}\right)^{\mathsf{T}} = \mathbf{A}^{t'-1}\boldsymbol{\Sigma}\left(\mathbf{A}^{t-1}\right)^{\mathsf{T}} + \sigma^2\sum_{\tau=2}^{\min(t,t')}\mathbf{A}^{t'-\tau}\left(\mathbf{A}^{t-\tau}\right)^{\mathsf{T}}$$

This therefore specifies the covariance matrix explicitly in terms of $\mathbf{A}$ and $\boldsymbol{\Sigma}$. This is a Gaussian Process since the joint distribution of $\mathbf{x}_1,\dots,\mathbf{x}_t$ is Gaussian, with covariance function given by equation (19.8.18).
3. Since the $\mathbf{x}$ are jointly Gaussian, and $\mathbf{y}_1,\dots,\mathbf{y}_t$ is linearly dependent on $\mathbf{x}_1,\dots,\mathbf{x}_t$, then $\mathbf{y}_1,\dots,\mathbf{y}_t$ is Gaussian. This indeed a zero mean Gaussian Process with covariance function

$$\left\langle \mathbf{y}_{t'}\mathbf{y}_t^{\mathsf{T}}\right\rangle = \left\langle \left(\mathbf{B}\mathbf{x}_{t'} + \boldsymbol{\epsilon}_{t'}\right)\left(\mathbf{B}\mathbf{x}_t + \boldsymbol{\epsilon}_t\right)^{\mathsf{T}}\right\rangle = \mathbf{B}\left\langle \mathbf{x}_{t'}\mathbf{x}_t^{\mathsf{T}}\right\rangle\mathbf{B}^{\mathsf{T}}$$

20.1 Since the data is assumed i.i.d., we can concentrate on the observations n. In this case

$$p(v_1^n, \ldots, v_{i-1}^n, v_{i+1}^n, \ldots, v^N) = \int_{v_i^n} p(v) = \sum_h p(h) \underbrace{\left[\int_{v_i^n} p(v_i|h)\right]}_{1} \left[\prod_{j\neq i} p(v_j^n|h)\right]$$

Hence

$$p(v_1^n, \ldots, v_{i-1}^n, v_{i+1}^n, \ldots, v^N) = \sum_h p(h) \prod_{j\neq i} p(v_j^n|h)$$

which is equivalent to 'ignoring' the i^{th} component of datapoint n.

20.2 In this case one can readily adjust equation (20.3.12) to give

$$\text{trace}\left(\mathbf{S}_i^{-1} \sum_{n=1}^{N} p^{old}(i|\mathbf{x}^n)\Delta_i^n \left(\Delta_i^n\right)^{\mathsf{T}}\right) - \log\det\left(\mathbf{S}_i^{-1}\right) \sum_{n=1}^{N} p^{old}(i|\mathbf{x}^n) \tag{30.0.131}$$

where $\mathbf{S}_i = \text{diag}\left(d_1^i, \ldots, d_D^i\right)$. The determinant is the product of the diagonal entries for a diagonal matrix. The contribution from d_j^i is therefore

$$\sum_{n=1}^{N} p^{old}(i|\mathbf{x}^n) \left(\Delta_i^n\right)^{\mathsf{T}} \Delta_i^n \frac{1}{d_j^i} + \log d_j^i \sum_{n=1}^{N} p^{old}(i|\mathbf{x}^n) \tag{30.0.132}$$

Differentiating with respect to d_j^i and equating to zero gives

$$d_j^i = \sum_{n=1}^{N} p^{old}(i|\mathbf{x}^n) \left(\Delta_i^n\right)^{\mathsf{T}} \Delta_i^n \tag{30.0.133}$$

20.3

$$p(n|i) = \frac{p(i|\mathbf{x}^n)}{\sum_n p(i|\mathbf{x}^n)} = \frac{p(\mathbf{x}^n|i)\cancel{p(i)}}{\sum_n p(\mathbf{x}^n|i)\cancel{p(i)}}$$

Using the GMM form this is

$$p(n|i) = \frac{e^{-\frac{1}{2\sigma^2}(\mathbf{x}^n - \mathbf{m}_i)^2}}{\sum_n e^{-\frac{1}{2\sigma^2}(\mathbf{x}^n - \mathbf{m}_i)^2}}$$

As σ^2 becomes very small, the numerator term becomes exponentially small. However, given i, there will be an $\mathbf{x}^n$ that is closest to $\mathbf{m}_i$. Whilst the numerator is small for this n, it is exponentially larger than for any other $n' \neq n$. By normalisation, it is therefore the case that $p(n|i)$ tends to 1 for that n such that $\mathbf{x}^n$ is closest to $\mathbf{m}_i$, with all other elements of $p(n|i)$ tending to zero.

20.4

$$\frac{\partial}{\partial p(h)} L = \sum_n \frac{\partial}{\partial p(h)} \sum_{h'} p^{old}(h'|v^n) \log p(h') - \lambda = \sum_n p^{old}(h|v^n) \frac{1}{p(h)} - \lambda$$

Equating to zero for the optimum, we have

$$0 = \frac{1}{p(h)} \sum_n p^{old}(h|v^n) - \lambda$$

hence

$$p(h) = \frac{1}{\lambda} \sum_n p^{old}(h|v^n)$$

Since $\sum_h p(h) = 1$, we have the desired result.

20.5 The maximum likelihood solution for the covariance is

$$\mathbf{\Sigma} = \frac{1}{N} \sum_n (\mathbf{x}^n - \boldsymbol{\mu})(\mathbf{x}^n - \boldsymbol{\mu})^{\mathsf{T}}$$

where $\boldsymbol{\mu}$ is given the data mean. Hence $\mathbf{\Sigma}$ always remains finite (for bounded data).

20.7 Firstly, any matrix formed from an outerproduct $\mathbf{Z}_*\mathbf{Z}_*^\top$ is positive semidefinite. Since $\mathbf{Z}$ is a clique matrix for $\mathbf{A}$, we must have, for $S_{ij} = 0$

$$\sum_k Z^*_{ik} Z^*_{kj} = \sum_k Z_{ik} Z_{kj} \theta_{ik} \theta_{kj}$$

Since for $S_{ij} = 0$, $\sum_k Z_{ik} Z_{kj} = 0$ and $Z_{ik} \in \{0, 1\}$, then for all k, $Z_{ik} Z_{kj} = 0$. This means that for $S_{ij} = 0$

$$[\mathbf{S}_*]_{ij} = \sum_k Z^*_{ik} Z^*_{kj} = 0$$

Hence $\mathbf{S}_*$ has the same zeros structure as $\mathbf{S}$ and is a valid semidefinite covariance matrix for any θ.

21.1 For a factor analysis model

$$\mathbf{x} = \mathbf{M}\mathbf{h} + \boldsymbol{\eta}$$

where $\boldsymbol{\eta}$ is noise from a diagonal matrix. Then

$$\mathbf{C}\mathbf{x} = \mathbf{C}\mathbf{M}\mathbf{h} + \mathbf{C}\boldsymbol{\eta}$$

or

$$\mathbf{y} = \mathbf{M}'\mathbf{h} + \boldsymbol{\eta}'$$

If $\boldsymbol{\eta}$ has diagonal covariance $\mathbf{D}$ then $\boldsymbol{\eta}'$ has diagonal covariance $\text{diag}\left(C_{ii}^2 D_{ii}\right)$.

21.2 By writing

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_A \\ \mathbf{x}_B \end{pmatrix}$$

The contribution of $\mathbf{A}$ and $\mathbf{B}$ to the energy term is proportional to

$$\sum_n \left\langle (\mathbf{x}_A - \mathbf{A}\mathbf{h})^\top \boldsymbol{\Psi}_A (\mathbf{x}_A - \mathbf{A}\mathbf{h}) \right\rangle_n + \sum_n \left\langle (\mathbf{x}_B - \mathbf{B}\mathbf{h})^\top \boldsymbol{\Psi}_B (\mathbf{x}_B - \mathbf{B}\mathbf{h}) \right\rangle_n$$

Since the contributions of $\mathbf{A}$ and $\mathbf{B}$ are separate, and each is exactly of the standard FA form, we can use the standard M-step to arrive at

$$\mathbf{A} = \mathbf{M}_A \mathbf{H}^{-1}$$

where

$$\mathbf{M}_A \equiv \frac{1}{N} \sum_n (\mathbf{x}_A^n - \bar{\mathbf{x}}_A) \langle \mathbf{h} \rangle^\top_{q(\mathbf{h}|\mathbf{x}^n)}, \quad \mathbf{H} = \frac{1}{N} \sum_n \left\langle \mathbf{h}\mathbf{h}^\top \right\rangle_{q(\mathbf{h}|\mathbf{x}^n)}$$

An analogous update holds for $\mathbf{B}$.

21.3 Under this prior the marginal distribution of the visible variables is

$$\mathbf{v} \sim \mathcal{N}\left(\mathbf{v} \middle| \mathbf{c}, \mathbf{F}\boldsymbol{\Sigma}_H \mathbf{F}^\top + \boldsymbol{\Psi}\right) \tag{30.0.134}$$

By defining a new covariance matrix

$$\tilde{\mathbf{F}} \equiv \mathbf{F}\boldsymbol{\Sigma}_H^{\frac{1}{2}} \tag{30.0.135}$$

where we define a square root (Cholesky factor for example) such that $\boldsymbol{\Sigma}_H^{\frac{1}{2}} {\boldsymbol{\Sigma}_H^{\frac{1}{2}}}^\top = \boldsymbol{\Sigma}_H$, then

$$\mathbf{v} \sim \mathcal{N}\left(\mathbf{v} \middle| \mathbf{c}, \tilde{\mathbf{F}}\tilde{\mathbf{F}}^\top + \boldsymbol{\Psi}\right) \tag{30.0.136}$$

This is of the same for as the FA for an uncorrelated prior, simply with a renaming of $\mathbf{F}$. There is therefore nothing gained from using a correlated Gaussian prior $p(\mathbf{h})$.

21.4

$$\begin{aligned}
\boldsymbol{\Sigma}_D^{-1}\mathbf{F} &= \left(\mathbf{F}\mathbf{F}^\top + \boldsymbol{\Psi}\right)^{-1}\mathbf{F} \\
&= \left[\boldsymbol{\Psi}^{-1} - \boldsymbol{\Psi}^{-1}\mathbf{F}\left(\mathbf{I} + \mathbf{F}^\top\boldsymbol{\Psi}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\boldsymbol{\Psi}^{-1}\right]\mathbf{F} \\
&= \boldsymbol{\Psi}^{-1}\mathbf{F} - \boldsymbol{\Psi}^{-1}\mathbf{F}\left(\mathbf{I} + \mathbf{F}^\top\boldsymbol{\Psi}^{-1}\mathbf{F}\right)^{-1}\left(\mathbf{F}^\top\boldsymbol{\Psi}^{-1}\mathbf{F} + \mathbf{I} - \mathbf{I}\right) \\
&= \cancel{\boldsymbol{\Psi}^{-1}\mathbf{F} - \boldsymbol{\Psi}^{-1}\mathbf{F}} + \boldsymbol{\Psi}^{-1}\mathbf{F}\left(\mathbf{I} + \mathbf{F}^\top\boldsymbol{\Psi}^{-1}\mathbf{F}\right)^{-1}
\end{aligned}$$

21.5 Differentiating *w.r.t.* σ^2 and equating to zero immediately gives the result.

21.6

1. The dependence of the energy on $\mathbf{W}$ is proportional to

$$\left\langle (\mathbf{y}^n - \mathbf{W}\mathbf{x}^n)^2 \right\rangle_{q(\mathbf{x}|\mathbf{y}^n,\mathbf{W}^{old})}$$

Since this is quadratic in $\mathbf{x}$ we only need the statistics $\langle \mathbf{x} \rangle$, $\left\langle \mathbf{x}\mathbf{x}^\mathsf{T} \right\rangle$.

2.

$$p(\mathbf{x}|\mathbf{y},\mathbf{W}) \propto p((y)|\mathbf{x},\mathbf{W})p(\mathbf{x}) = e^{-\frac{1}{2\sigma^2}(\mathbf{y}-\mathbf{W}\mathbf{x})^2} \prod_i p(x_i)$$

This is non-Gaussian in $\mathbf{x}$ since the priors $p(x_i)$ are non-Gaussian. In general, this distribution is not tractable since there is no coordinate rotation that will render this as a product of one dimensional integrals.

3. In the limit $\sigma^2 \to 0$, the term

$$e^{-\frac{1}{2\sigma^2}(\mathbf{y}-\mathbf{W}\mathbf{x})^2}$$

dominates and

$$p(\mathbf{x}|\mathbf{y},\mathbf{W}) = \delta(\mathbf{x} - \mathbf{x}_\mathbf{y})$$

where $\mathbf{x}_\mathbf{y}$ is that vector $\mathbf{x}$ for which $(\mathbf{y} - \mathbf{W}\mathbf{x})^2$ is minimal.

Let's look at the M-step for $\mathbf{w}_j$. This is given by the minimum of

$$\sum_n \left\langle (\mathbf{y}^n - \mathbf{W}\mathbf{x}^n)^2 \right\rangle_{p(\mathbf{x}|\mathbf{y}^n,\mathbf{W}^{old})}$$

However, this is minimal for $\mathbf{W} = \mathbf{W}^{old}$, meaning that no update occurs and EM fails.

22.1

$$20, 1, 11, 13, 2, 16, 6, 8, 5, 7$$

See `exerciseRaschBronco.m`.

22.3

1. See `exerciseLaReine.m`
2.

$$59, 8, 77, 44, 39, 48, 78, 24, 98, 42$$

22.4

1. TBD
2. TBD
3. The highest positive weights indicate the most valuable player.
4. This can be done by looking at the probability that team 1 beats team 2 as a function of the binary factors of team 2. One can maximise this probability by searching over the space of all the binary factors.

23.1

$$\sum_{ij} M_{ij} e_j = \sum_j e_j = \lambda \sum_i e_i$$

Hence, provided $\sum_i e_i \neq 0$, any stochastic matrix $\mathbf{M}$ has an eigenvector with unit eigenvalue.

23.2 The effect of this transition is simply to deterministically the state at each time step. Hence

$$p(x_t = 1) = \mathbb{I}[x_1 = 1] \int t \text{is odd} + \mathbb{I}[x_1 = 2] \int t \text{is even}$$

Hence, no matter how large t is, there remains a dependence on the initial condition. This means that there is no equilibrium distribution. For a stationary distribution we require

$$\begin{pmatrix} p_1 \\ p_2 \end{pmatrix} = \mathbf{M} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \end{pmatrix}$$

The distribution $p(x = 1) = p(x_2) = 1/2$ satisfies this and is therefore a stationary distribution.

23.3 This problem is small enough to be computed 'by hand'. It is of course easier to do this numerically, which can be achieved using the standard forward and backward routines. An inefficient, but perhaps instructive way to do this is to use BRMLtoolbox, see `exerciseHMMsimple.m`. This gives:

1. $p(v_{1:3}) = 0.153823$
2. $p(h_1|v_{1:3}) = \begin{pmatrix} 0.930933605507629 \\ 0.069066394492372 \\ 0 \end{pmatrix}$
3. $\arg\max_{h_{1:3}} p(h_{1:3}|v_{1:3}) = (1, 2, 3)$

23.4 The 659th most likely joint sequence is the first that is a correct sentence 'the monkey is on the branch', with $\log p(h_{1:27}|v_{1:27}) = -94.5250$. See `demoHMMbigramMonkey.m`.

23.5 Under these initial settings, the visible variables have no influence on the posterior $p^{old}(h_{1:T}|v_{1:T})$, which is uniform. Similarly, all the marginals such as $p^{old}(h_t, h_{t+1}|v_{1:T})$ are also uniform. From equation (23.3.5) we see that setting $p^{old}(i, i') = 1/H$ for a constant k gives $A^{new}_{i',i} = 1/H$. Similarly, $p^{new}(v_t|h_t)$ does not depend on h_t, $v_t \perp\!\!\!\perp h_t$ meaning that the visible variables still play no role in the updates. Hence the M-step retains the pure symmetry of the initialisation and no sensible update is made.

23.6

1. We need to carry out the computation:

$$\max_{v_{1:T}} \sum_{h_{1:T}} \prod_{t=1}^{T} p(v_t|h_t)p(h_t|h_{t-1})$$

In general, we are not able to interchange the order of the sum and the max, so that we cannot distribute all the required computations to make a recursive algorithm. Carrying out the summation first means that the resulting $p(v_{1:T})$ does not have any structure, being a single clique on $v_{1:T}$. If each observation as V states, the complexity of computing the most likely observation sequence is therefore $O\left(V^T\right)$.

2. The idea is to treat $v_{1:T}$ as 'parameters' in an EM algorithm:

$$\underset{v_{1:T}}{\text{argmax}} \ \log p(v_{1:T}) = \underset{v_{1:T}}{\text{argmax}} \ \log \sum_{h_{1:T}} p(v_{1:T}|h_{1:T})p(h_{1:T})$$

From the EM algorithm, we need to consider the M-step:

$$v^{new}_{1:T} = \underset{v_{1:T}}{\text{argmax}} \sum_t \left\langle \log p(v_t|h_t)p(h_t|h_{t-1}) \right\rangle_{p(h_{1:T}|v^{old}_{1:T})} = \underset{v_{1:T}}{\text{argmax}} \sum_t \left\langle \log p(v_t|h_t) \right\rangle_{p(h_t|v^{old}_{1:T})}$$

Hence the M-step is given by

$$v^{new}_t = \underset{v_t}{\text{argmax}} \ \left\langle \log p(v_t|h_t) \right\rangle_{p(h_t|v^{old}_{1:T})}$$

In order to compute this update, we require the smoothed marginal $p(h_t|v^{old}_{1:T})$ which can be computed efficiently using the forward-backward algorithm. BY virtue of this being an EM algorithm, updating in this way guarantees to find a more likely $v_{1:T}$ at each stage until we reach a local optimum.

23.7 In general, one may add a Lagrange term to the energy to enforce the constraint. In the case of an upper triangular structure, we may simply optimise the energy with respect to the upper triangular parts alone, setting the lower triangular parts to zero. This is equivalent to the standard M-step but setting the lower triangular part to zero and renormalising.

23.8 See `exerciseACGTHMM.m`

23.9 Write

$$\max_{h_1,\ldots,h_T} \prod_{t=2}^{T} \phi_t(h_t, h_{t-1}) = \max_{h_1,\ldots,h_{T-1}} \prod_{t=2}^{T-1} \phi_t(h_t, h_{t-1}) \underbrace{\max_{h_T} \phi_T(h_T, h_{T-1})}_{\gamma_{T-1\leftarrow T}(h_{T-1})} \tag{30.0.137}$$

Similarly

$$\max_{h_1,\ldots,h_T} \prod_{t=2}^{T} \phi_t(h_t, h_{t-1}) = \max_{h_1,\ldots,h_{T-2}} \prod_{t=2}^{T-2} \underbrace{\max_{h_{T-1}} \phi_T(h_{T-1}, h_{T-2})\gamma_{T-1\leftarrow T}(h_{T-1})}_{\gamma_{T-2\leftarrow T-1}(h_{T-2})} \tag{30.0.138}$$

which describes the recursion. Once we have repeated the recursion, we will have

$$\max_{h_1,\ldots,h_T} \prod_{t=2}^{T} \phi_t(h_t, h_{t-1}) = \max_{h_1} \gamma_{1\leftarrow 2}(h_1) \tag{30.0.139}$$

We can compute the most probable state h_1^* then explicitly. For h_2, knowing h_1^* we have

$$h_2^* = \underset{h_2}{\text{argmax}} \ \phi(h_2, h_1^*)\gamma_{2\leftarrow 3}(h_2) \tag{30.0.140}$$

we continue to backtrack in this manner.

23.11 One can arrive at the solution by following the same procedure as for the first order HMM. One writes down a suitable potential $\phi_t = p(v_t|h_t)p(h_t|h_{t-1}, h_{t-2})$. One now performs a maximisation over h_t, which defines a message

$$\gamma_{t-1\leftarrow t}(h_{t-1}, h_{t-2}) = \max_{h_t} \phi(h_t, h_{t-1}, h_{t-2})\gamma_{t\leftarrow t+1}(h_t, h_{t-1}) \tag{30.0.141}$$

Everything is the same as before except that a vector message is replaced with a matrix message. Once the recursion down to h_1, h_2 is completed, the optimal h_1, h_2 can be computed explicitly. Backtracking then continues.

23.12 Just use the general formula for the gradient give in section(11.6).

23.13

$$p(\mathcal{H}|\mathcal{V}) \propto \prod_t \underbrace{p(h_t|h_{t-1})p(v_t|h_t)}\phi(h_t, h_{t-1})$$

This is therefore an undirected distribution that is a chain. If we start at the end of the chain we have the term

$$\phi(h_T, h_{T-1})$$

which is the only place that h_T appears. We can therefore write

$$\tilde{p}(h_T|h_{T-1}) = \frac{\phi(h_T, h_{T-1})}{\sum_{h_T} \phi(h_T, h_{T-1})}$$

Let $z(h_{T-1}) = \sum_{h_T} \phi(h_T, h_{T-1})$

$$p(\mathcal{H}|\mathcal{V}) \propto \left[\prod_{t=1}^{T-1} \phi(h_t, h_{t-1})\right] z(h_{T-1})\tilde{p}(h_T|h_{T-1}) = \left[\prod_{t=1}^{T-2} \phi(h_t, h_{t-1})\right] \underbrace{\phi(h_{T-1}, h_{t-2})z(h_{T-1})}_{\tilde{p}(h_{T-1}|h_{T-2})z(h_{T-2})}\tilde{p}(h_T|h_{T-1})$$

In general,

$$z(h_{t-1}) = \sum_{h_t} \phi(h_t, h_{t-1})z(h_t), \quad \tilde{p}(h_t|h_{t-1}) = \frac{\phi(h_t, h_{t-1})z(h_t)}{z(h_{t-1})}$$

23.15 All parts are straightforward. The last part can be achieved using a β recursion.

23.16
1. A HMM would normally take $O\left(TL^2\right)$ operations. This can be reduced here since the transition is deterministic for all states h_t that are not 0. This means that no sum is required over the $h_t \neq 0$ states and the computation is reduced to linear scaling in L.
2. See `teststringsearch.m`.

24.1
1. The eigenvalues are given by $\cos\theta \pm i \sin\theta$, which are in general imaginary. Intuitively, this has to be the case since the definition of an eigenvalue-eigenvector is that the matrix operating on the eigenvector simply rescales the eigenvector. For a rotation matrix, clearly this cannot simply rescale the eigenvector – therefore there can be no real solution (unless the rotation corresponds to flipping the direction, $\theta = \pi$).
2. This is explained in the text: we can take use a two dimensional latent $\mathbf{h}_t$ with transition matrix given by $\mathbf{R}_\theta$. Then for the emission matrix, we can take the projection so that $v_t = [10]\mathbf{h}_t$. This takes then only the first component of $\mathbf{h}_t$, which will trace out a sinusoid.
3. Using $a = \cos\theta$, $b = -\sin\theta$, $c = \sin\theta$, $d = \cos\theta$,

   $$x_t = ax_{t-1} + by_{t-1}, y_t = cx_{t-1} + dy_{t-1} \tag{30.0.142}$$

   The first equation also means

   $$x_{t+1} = ax_t + by_t = ax_t + b\left(cx_{t-1} + dy_{t-1}\right) = ax_t + b\left(cx_{t-1} + \frac{d}{b}\left[x_t - ax_{t-1}\right]\right) \tag{30.0.143}$$
4. This shows that one may express a sinusoidal curve exactly as a second order difference equation.
5. The solution of this continuous time harmonic oscillator is a sinusoid. This is interesting since it shows that there exists in an exact time discretisation for a second order harmonic oscillator. That is, the discrete time system exactly matches the continuous time system at the discrete time points.

24.2

$$\mathbf{A}^\mathsf{T}\mathbf{A} = e^{\mathbf{M}}e^{\mathbf{M}^\mathsf{T}} = e^{\mathbf{M}+\mathbf{M}^\mathsf{T}} = e^{\mathbf{M}-\mathbf{M}} = e^{\mathbf{0}} = \mathbf{I}$$

Note that $\mathbf{A}^\mathsf{T} = e^{\mathbf{M}^\mathsf{T}}$ since $e^{\mathbf{M}}$ is obtained from the Taylor series expansion of exp. This means that it is a function of powers of $\mathbf{M}$, and since ${\mathbf{M}^k}^\mathsf{T} = {\mathbf{M}^\mathsf{T}}^k$, the result follows. We can have some control over the orthogonal matrix by adjusting $\mathbf{M}$. If the elements of $\mathbf{M}$ are very small, then $\mathbf{A}$ will be orthogonal but close to the identity. The larger $\mathbf{M}$ is the larger the rotation enacted by $\mathbf{A}$ will be.

24.5 TBD

25.1 See `exerciseSLDSprice.m`.

25.2 The solution is in `exerciseSLDSmeanrev.m` : 0.8298 and 0.9142

26.1 Since

$$p(v_i(t+1)|\mathbf{v}(t)) = \sigma\left(v_i(t+1)\left(b_i + \sum_j w_{ij}v_j(t)\right)\right)$$

as we saw in equation (26.3.19),

$$\frac{dL}{dw_{ij}} = \beta\sum_{t=1}^{T-1}\gamma_i(t)v_i(t+1)v_j(t)$$

which means that if we start gradient ascent from $w_{ij}=0$, we must end up with w of the form

$$w_{ij} = \sum_{t=1}^{T-1} u_i(t)v_i(t+1)v_j(t)$$

for some suitable 'dual' parameters $u_i(t)$. In this case we can then learn the dual parameters rather than the weights. Note that there are VT dual parameters. Since we assume $V \ll T$, then this requires much less storage than w. One can proceed to learn the u by maximum likelihood. Then

$$p(v_i(t+1)|\mathbf{v}(t)) = \sigma\left(v_i(t+1)\left(b_i + \sum_{j,\tau} u_j(\tau)v_i(\tau+1)v_j(\tau)v_j(t)\right)\right)$$

26.2 There are N neurons. If we assume all images are independent, and no restriction on w, we can store maximally N patterns. Hence we can store

$$\frac{10^6}{10\times 60\times 60} \approx 27.8$$

hours of binary video.

26.3

$$\langle v_i(t+1)\rangle = \sigma(a_i(t)) - (1-\sigma(a_i(t)))$$

Hence

$$\sigma(a_i(t)) = \frac{1}{2})(1+\langle v_i(t+1)\rangle)$$

Also

$$\gamma_i(t) = 1 - \frac{v_i(t+1)+1}{2}\sigma(a_i(t)) + \frac{v_i(t+1)-1}{2}(1-\sigma(a_i(t)))$$

Combining these gives the required result.

26.4

$$\begin{aligned}
\sum_{i,j,k,l} x_{ij}x_{kl}\frac{d^2L}{dw_{ij}dw_{kl}} &= -\beta^2\sum_{i,j,k,l}x_{ij}x_{kl}\sum_{t=1}^{T-1}\left(v_i(t+1)v_j(t)\right)\gamma_i(t)(1-\gamma_i(t))v_k(t+1)v_l(t)\delta_{ik}\\
&= -\beta^2\sum_{i,j,l}x_{ij}x_{il}\sum_{t=1}^{T-1}v_j(t)\gamma_i(t)(1-\gamma_i(t))v_l(t)\\
&= -\beta^2\sum_{t,i}\gamma_i(t)(1-\gamma_i(t))\underbrace{\sum_j x_{ij}v_j(t)}_{z_i(t)}\underbrace{\sum_l v_l(t)x_{il}}_{z_i(t)}\\
&= -\beta^2\sum_{t,i}\gamma_i(t)(1-\gamma_i(t))z_i^2(t) \le 0
\end{aligned}$$

Which follows since $0 \le \gamma_i(t) \le 1$.

26.5 TBD

26.6 Based on maximum likelihood training, I estimate that one can store around 20 patterns robustly. See `exerciseHopfieldSelfTran.m`.

27.1 Writing $\mathbf{y} = (y_1, y_2)$ and $\mathbf{x} = (x_1, x_2)$, we have

$$p(\mathbf{y}) = \int \delta(\mathbf{y} - f(\mathbf{x}))d\mathbf{x} = \int \delta(\mathbf{x} - f^{-1}(\mathbf{y})) \left| \frac{d}{d\mathbf{y}}\mathbf{x} \right| d\mathbf{y}$$

From exercise(8.10), we need to find the absolute value of the determinant of the Jacobian:

$$\begin{vmatrix} \frac{\partial}{\partial x_1} y_1 & \frac{\partial}{\partial x_2} y_1 \\ \frac{\partial}{\partial x_1} y_2 & \frac{\partial}{\partial x_2} y_2 \end{vmatrix} = \begin{vmatrix} -(-2\log x_1)^{-\frac{1}{2}} \frac{\cos 2\pi x_2}{x_1} & -2\pi(-2\log x_1)^{\frac{1}{2}} \sin 2\pi x_2 \\ -(-2\log x_1)^{-\frac{1}{2}} \frac{\sin 2\pi x_2}{x_1} & 2\pi(-2\log x_1)^{\frac{1}{2}} \cos 2\pi x_2 \end{vmatrix} = -\frac{2\pi}{x_1}\left(\cos^2 2\pi x_2 + \sin^2 2\pi x_2\right) = -\frac{2\pi}{x_1}$$

From the transformation, we also have

$$x_1 = e^{-\frac{1}{2}(y_1^2 + y_2^2)}$$

Hence, since $p(x_1) = p(x_2) = 1$,

$$p(\mathbf{y}) = \frac{1}{2\pi} e^{-\frac{1}{2}(y_1^2 + y_2^2)} = \mathcal{N}(y_1|0,1)\,\mathcal{N}(y_2|0,1)$$

27.2

$$\frac{p^*(x)}{q(x)} = e^{\sin(x) + \frac{x^2}{2\sigma^2}} \sqrt{2\pi\sigma^2}$$

Since $-\pi \leq x \leq \pi$ this quantity certainly cannot be greater than

$$e^{1 + \frac{\pi^2}{2\sigma^2}} \sqrt{2\pi\sigma^2} = M$$

See `demoRejSamp.m`.

27.3

$$p'(x_1, x_2, x_3, x_4, x_6) = \frac{p(x_1)p(x_2)p(x_3|x_1, x_2)p(x_4|x_3)p(\mathsf{x}_5|x_3)p(x_6|x_4, \mathsf{x}_5)}{\sum_{x_1,x_2,x_3,x_4,x_6} p(x_1)p(x_2)p(x_3|x_1, x_2)p(x_4|x_3)p(\mathsf{x}_5|x_3)p(x_6|x_4, \mathsf{x}_5)}$$

Since

$$\sum_{x_1,x_2,x_3,x_4,x_6} p(x_1)p(x_2)p(x_3|x_1, x_2)p(x_4|x_3)p(\mathsf{x}_5|x_3)p(x_6|x_4, \mathsf{x}_5) = \sum_{x_1,x_2,x_3} p(x_1)p(x_2)p(x_3|x_1, x_2)p(\mathsf{x}_5|x_3)$$

Hence

$$p'(x_1, x_2, x_3, x_4, x_6) = p'(x_1, x_2, x_3)p(x_4|x_3)p(x_6|x_4, \mathsf{x}_5)$$

where

$$p'(x_1, x_2, x_3) = \frac{p(x_1)p(x_2)p(x_3|x_1, x_2)p(\mathsf{x}_5|x_3)}{\sum_{x_1,x_2,x_3} p(x_1)p(x_2)p(x_3|x_1, x_2)p(\mathsf{x}_5|x_3)}$$

To perform ancestral sampling, we can therefore sample a joint state x_1, x_2, x_3 from $p'(x_1, x_2, x_3)$ and then a state x_4 from $p(x_4|x_3)$ and a state x_6 from $p(x_6|x_4, \mathsf{x}_5)$.

27.4 Consider a 4×4 example:

$$\begin{pmatrix} w_1 & b_1 & w_2 & b_2 \\ b_3 & w_3 & b_4 & w_4 \\ w_5 & b_5 & w_6 & b_6 \\ b_7 & w_7 & b_8 & w_8 \end{pmatrix}$$

Since all neighbours of any white variable all all black, and similarly, all neighbours of any black variable are all white, when we condition on all the black variables, the white variables become independent, and vice versa. This means that in the Gibbs step to sample from

$$p(b_1, b_2, \ldots, |w_1, w_2, \ldots) = p(b_1|w_1, w_2, \ldots)p(b_2|w_1, w_2, \ldots)\ldots$$

we can draw a sample from the joint distribution by independently drawing a sample from each of the black variables conditioned on its neighbouring white variables. The converse is true when we sample from

$$p(w_1, e_2, \ldots, |b_1, b_2, \ldots) = p(w_1|b_1, b_2, \ldots)p(w_2|b_1, b_2, \ldots)\ldots$$

This is advantageous since we are able to truly draw independent samples from all black variables conditioned on all white variables, and vice versa.

27.5

$$\left\langle \log \frac{p(\mathbf{x}')}{p(\mathbf{x})} \right\rangle_{\tilde{q}(\mathbf{x}'|\mathbf{x})} = \frac{1}{2\sigma_p^2}\left(\mathbf{x}^2 - \left\langle (\mathbf{x}')^2 \right\rangle_{\tilde{q}(\mathbf{x}'|\mathbf{x})}\right)$$

Since

$$\left\langle (\mathbf{x}')^2 \right\rangle_{\tilde{q}(\mathbf{x}'|\mathbf{x})} = \mathbf{x}^2 + N\sigma_q^2$$

the result follows. The ratio

$$\frac{p(\mathbf{x}')}{p(\mathbf{x})}$$

is the Metropolis acceptance probability for a proposed $\mathbf{x}'$ generated by adding random Gaussian vector onto the current sample $\mathbf{x}$. What the result shows is that the rough scaling of this acceptance probability is that it will be exponentially small in the dimension N. This suggests that the acceptance probability for this proposal distribution will be impractically small in high dimensions. To counter this effect one would need to make σ_q^2 extremely small.

27.6 See `exerciseSampleHMM.m`. As λ increases, the variables in the posterior become more dependent. This means that this becomes far from the condition under which single variable Gibbs updating is valid. As such, the accuracy tends to go down as we increase the dependence in the posterior.

27.8 For the last part we need

$$\frac{1}{\sigma^4}\left\langle \left[\left(\mathbf{x}^i - \mathbf{x}\right)^2 - \left(\mathbf{x}^j - \mathbf{x}\right)^2 \right]^2 \right\rangle \tag{30.0.144}$$

The average is straightforward based on moments of Gaussians. This can be made a little easier by defining

$$\mathbf{y}_i = \mathbf{x} - \mathbf{x}^i$$

which is Gaussian with zero mean and variance $2\nu^2\mathbf{I}$, and $\langle y_{ik} y_{jl} \rangle = \nu^2$ for $i \neq j$. Then the required computation is

$$2\left(\left\langle \left(\mathbf{y}_i^2\right)^2 \right\rangle - \left\langle (\mathbf{y}_i)^2 (\mathbf{y}_j)^2 \right\rangle\right) = 2D\left(\langle y^4 \rangle - \langle y_i^2 y_j^2 \rangle\right) = 2D\left(3(2\nu^2)^2 - 6\nu^4\right)$$

28.1 0.0160. See `exerciseKLfit.m`.

28.2 `mean error BP = 0.0402797 mean error MF = 0.0501128` See `exerciseMFBP.M`. Both work quite well for this small problem, with BP working slightly better.

28.3 TBD

28.5

1.

$$\begin{aligned} \int e^x &\geq \int 0 \\ e^x - e^a &\geq 0 \qquad\qquad (x \geq a) \\ \int \left(e^x - e^a\right) &\geq \int 0 \\ e^x - e^a - (x-a)e^a &\geq 0 \end{aligned}$$

Rearranging, we obtain

$$e^x \geq e^a(1 + x - a)$$

2.

$$e^{\mathbf{s}^\mathsf{T}\mathbf{W}\mathbf{s}} \geq e^{\mathbf{h}^\mathsf{T}\mathbf{s}}(1 + \mathbf{s}^\mathsf{T}\mathbf{W}\mathbf{s} - \mathbf{h}^\mathsf{T}\mathbf{s})$$

$$Z = \sum_{\mathbf{s}} e^{\mathbf{s}^\mathsf{T}\mathbf{W}\mathbf{s}} \geq \sum_{\mathbf{s}} e^{\mathbf{h}^\mathsf{T}\mathbf{s}+\theta}(1 + \mathbf{s}^\mathsf{T}\mathbf{W}\mathbf{s} - \mathbf{h}^\mathsf{T}\mathbf{s} - \theta)$$

3. Differentiating *w.r.t.* θ, and equating to zero we have

$$\sum_{\mathbf{s}} e^{\mathbf{h}^\mathsf{T}\mathbf{s}+\theta}(-1 + 1 + \mathbf{s}^\mathsf{T}\mathbf{W}\mathbf{s} - \mathbf{h}^\mathsf{T}\mathbf{s} - \theta) = 0$$

Hence

$$\sum_{\mathbf{s}} e^{\mathbf{h}^\mathsf{T}\mathbf{s}+\theta}(\mathbf{s}^\mathsf{T}\mathbf{W}\mathbf{s} - \mathbf{h}^\mathsf{T}\mathbf{s} - \theta) = 0$$

and

$$\theta = \frac{1}{\sum_{\mathbf{s}} e^{\mathbf{h}^\mathsf{T}\mathbf{s}}} \sum_{\mathbf{s}} e^{\mathbf{h}^\mathsf{T}\mathbf{s}} (\mathbf{s}^\mathsf{T}\mathbf{W}\mathbf{s} - \mathbf{h}^\mathsf{T}\mathbf{s})$$

Plugging these into the bound and taking the logarithm, we have

$$\log Z \geq= \log Z_q + \left\langle \mathbf{s}^\mathsf{T}\mathbf{W}\mathbf{s} - \mathbf{h}^\mathsf{T}\mathbf{s} \right\rangle = H_q + + \left\langle \mathbf{s}^\mathsf{T}\mathbf{W}\mathbf{s} \right\rangle$$

where we define a factorised MF distribution

$$q(\mathbf{s}) = \frac{e^{\mathbf{h}^\mathsf{T}\mathbf{s}}}{\sum_{\mathbf{s}} e^{\mathbf{h}^\mathsf{T}\mathbf{s}}}$$

and $H_q = -\langle \log q(\mathbf{s}) \rangle_q$. This is then in the form of a standard naive mean field bound.

4. If we perform another double integration, on the bound

$$e^x \geq e^a(1 + x - a)$$

we obtain a cubic lower bound on e^x:

$$e^x - e^b \geq e^a(1-a)(x-b) + \frac{1}{2}e^a(x^2 - b^2) \qquad x \geq b$$

$$e^x - e^b - (x-b)e^b \geq -b(x-b)e^a(1-a) + \frac{1}{2}e^a(1-a)(x^2-b^2) - \frac{1}{2}b^2 e^a(x-b) + \frac{1}{3}e^a(x^3 - b^3)$$

If we use $a = b$, then we can rearrange to get

$$e^x \geq e^a f(x)$$

where $f(x)$ is a cubic polynomial in x. By using the same procedure as before, replacing $a \to \mathbf{h}^\mathsf{T}\mathbf{s} + \theta$, we can optimise the bound wrt to a factorised approximation.

28.6

$$Z = \sum_{\mathbf{x}} e^{\mathbf{b}^\mathsf{T}\mathbf{x}} e^{\mathbf{x}^\mathsf{T}\langle W\rangle\mathbf{x}} \leq \sum_i p_i \sum_{\mathbf{x}} e^{\mathbf{x}^\mathsf{T} W_i \mathbf{x} + \mathbf{b}^\mathsf{T}\mathbf{x}}$$

Each term $\sum_{\mathbf{x}} e^{\mathbf{x}^\mathsf{T} W_i \mathbf{x} + \mathbf{b}^\mathsf{T}\mathbf{x}}$ is the normalisation of a tree distribution, which is therefore tractable. Naively, one can therefore optimise the bound then with respect to the parameters p_i.

28.7 A special case of the IM framework is to use a linear Gaussian decoder

$$\begin{aligned}\langle \log q(\mathbf{x}|\mathbf{y}) \rangle_{p(\mathbf{x},\mathbf{y})} &= -\frac{1}{2}\left\langle (\mathbf{x} - \mathbf{U}\mathbf{y})^\mathsf{T} \mathbf{\Sigma}^{-1} (\mathbf{x} - \mathbf{U}\mathbf{y}) \right\rangle_{p(\mathbf{x},\mathbf{y})} - \frac{1}{2}\log\det(2\pi\mathbf{\Sigma}) \\ &= -\frac{1}{2}\text{trace}\left(\mathbf{\Sigma}^{-1}\left\langle (\mathbf{x} - \mathbf{U}\mathbf{y})(\mathbf{x} - \mathbf{U}\mathbf{y})^\mathsf{T} \right\rangle_{p(\mathbf{x},\mathbf{y})}\right) - \frac{1}{2}\log\det(2\pi\mathbf{\Sigma})\end{aligned}$$

Writing, more compactly $\langle \cdot \rangle$ for $\langle \cdot \rangle_{p(\mathbf{x},\mathbf{y})}$, and optimising *w.r.t.* $\mathbf{\Sigma}$ and $\mathbf{U}$ we obtain:

$$\mathbf{\Sigma} = \left\langle (\mathbf{x} - \mathbf{U}\mathbf{y})(\mathbf{x} - \mathbf{U}\mathbf{y})^\mathsf{T} \right\rangle, \qquad \mathbf{U} = \left\langle \mathbf{x}\mathbf{y}^\mathsf{T} \right\rangle \left\langle \mathbf{y}\mathbf{y}^\mathsf{T} \right\rangle^{-1}$$

Plugging these into the bound we obtain

$$I(X,Y) \geq H(X) - \frac{1}{2}\text{trace}(\mathbf{I}) - \frac{1}{2}\log\det(2\pi\mathbf{\Sigma})$$

We can write this directly in terms of the statistics of $p(\mathbf{x}, \mathbf{y})$ since

$$\mathbf{\Sigma} = \left\langle (\mathbf{x} - \mathbf{U}\mathbf{y})(\mathbf{x} - \mathbf{U}\mathbf{y})^\mathsf{T} \right\rangle = \left\langle \mathbf{x}\mathbf{x}^\mathsf{T} - \mathbf{x}\mathbf{y}^\mathsf{T}\mathbf{U}^\mathsf{T} - \mathbf{y}^\mathsf{T}\mathbf{U}^\mathsf{T}\mathbf{x} + \mathbf{U}\mathbf{y}\mathbf{y}^\mathsf{T}\mathbf{U}^\mathsf{T} \right\rangle = \left\langle \mathbf{x}\mathbf{x}^\mathsf{T} \right\rangle - \left\langle \mathbf{x}\mathbf{y}^\mathsf{T} \right\rangle \left\langle \mathbf{y}\mathbf{y}^\mathsf{T} \right\rangle^{-1} \left\langle \mathbf{y}\mathbf{x}^\mathsf{T} \right\rangle$$

28.8 1.

$$r(x) = \frac{p(x)f(x)}{Z}$$

where, by construction, therefore, $J = \log Z$. Taking $\text{KL}(r|q)$ gives the bound

$$\log Z = J \geq -\langle \log q \rangle_q + \langle \log p(x) \rangle_q + \langle \log f(x) \rangle_q = -\text{KL}(q|p) + \langle \log f(x) \rangle_q$$

2. The result follows simply from adding and subtracting the entropy $H(q)$. The result is particularly interpretable for a function $f(x)$ that is positive and integrates to 1, although the result holds more generally, even when f is not a distribution (on using the standard expression for the KL divergence). Essentially the three requirements translate into q being equal, optimally, to pf.

28.9

$$p(x_i) = \sum_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_D} p(\mathbf{x}) = \frac{1}{Z} \sum_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_D} e^{\mathbf{x}^{\mathsf{T}}\mathbf{W}\mathbf{x}}$$

If we wish to lower bound $p(x_i)$ we need a lower bound on $Z_{\setminus i}$ and a lower bound on $1/Z$ since the product of two lower bounds is a lower bound. In other words we need an upper bound on Z. To upper bound $p(x_i)$, we need the converse.

28.10 TBD

28.11 For any $y \in \mathcal{T}$ let's look at the flow $\gamma(y)$ through this node. This is the amount coming in minus the amount going out.

$$\begin{aligned}\gamma(y) &= \sum_x f(x,y) - \sum x f(y,x)\\ &= \sum_{x\in\mathcal{S}} f(x,y) + \sum_{x\in\mathcal{T}} f(x,y) - \sum_{x\in\mathcal{S}} f(y,x) - \sum_{x\in\mathcal{T}} f(y,x)\end{aligned}$$

Now sum this over $y \in \mathcal{T}$:

$$\sum_{y\in\mathcal{T}} \gamma(y) = \sum_{x\in\mathcal{S},y\in\mathcal{T}} f(x,y) + \cancel{\sum_{x\in\mathcal{T},y\in\mathcal{T}} f(x,y)} - \sum_{x\in\mathcal{S},y\in\mathcal{T}} f(y,x) - \cancel{\sum_{x\in\mathcal{T},y\in\mathcal{T}} f(y,x)} \tag{30.0.145}$$

Since from flow conversation $\gamma(y) = 0$ except for $y = t$, when $\gamma(t) = val(f)$, namely the flow across the network. Hence

$$val(f) = \sum_{x\in\mathcal{S},y\in\mathcal{T}} f(x,y) - \sum_{x\in\mathcal{S},y\in\mathcal{T}} f(y,x) \tag{30.0.146}$$

The second part follows since the term $\sum_{x\in\mathcal{S},y\in\mathcal{T}} f(y,x) \geq 0$, showing that the maximum flow cannot be greater than the minimum cut. The max-flow min s-t-cut theorem goes further and states that the maximum flow is actually equal to the minimum cut. See for example [40] for a concise proof.

28.12

$$\mathbb{I}\left[x_i = x_j\right] = \mathbb{I}\left[s_i\alpha + (1-s_i)x_i^{old} = s_j\alpha + (1-s_j)x_j^{old}\right] \tag{30.0.147}$$

Since $s_i \in \{0,1\}$, this can be written as

$$\begin{aligned}\mathbb{I}\left[x_i = x_j\right] &= (1-s_i)(1-s_j)\mathbb{I}\left[x_i^{old} = x_j^{old}\right] + (1-s_i)s_j\mathbb{I}\left[x_i^{old} = \alpha\right] + s_i(1-s_j)\mathbb{I}\left[x_j^{old} = \alpha\right] + s_i s_j\\ &= s_i s_j u_{ij} + a_i s_i + b_j s_j + \mathsf{const}.\end{aligned}$$

with

$$u_{ij} \equiv 1 - \mathbb{I}\left[x_i^{old} = \alpha\right] - \mathbb{I}\left[x_j^{old} = \alpha\right] + \mathbb{I}\left[x_i^{old} = x_j^{old}\right] \tag{30.0.148}$$

and similarly defined a_i, b_i. By enumeration it is straightforward to show that u_{ij} is either 0, 1 or 2. Using the mathematical identity

$$s_i s_j = \frac{1}{2}\left(\mathbb{I}\left[s_i = s_j\right] + s_i + s_j - 1\right) \tag{30.0.149}$$

we can write, for $x_i = s_i\alpha + (1-s_i)x_i^{old}$,

$$\mathbb{I}\left[x_i = x_j\right] = \frac{u_{ij}}{2}\left(\mathbb{I}\left[s_i = s_j\right] + s_i + s_j\right) + a_i s_i + b_j s_j + \mathsf{const}. \tag{30.0.150}$$

Hence terms $w_{ij}\mathbb{I}\left[x_i = x_j\right]$ translate to positive interaction terms $w_{ij}u_{ij}/2\mathbb{I}\left[s_i = s_j\right]$. All the unary terms are easily exactly mapped into corresponding unary terms $c_i' s_i$ for c_i' defined as the sum of all unary terms in s_i. This shows that the positive interaction w_{ij} in terms of the original variables x maps to a positive interaction in the new variables s. Hence we can find the maximal state of s using a graph cut algorithm.

28.13

1. $$\mathrm{KL}(p|q) = \langle \log p \rangle_p - \langle \log \rangle_p = -\boldsymbol{\phi}^{\mathsf{T}} \langle x \rangle_p + \log Z(\boldsymbol{\phi})\mathsf{const}.$$

   Differentiating *w.r.t.* ϕ_i and equating to zero, the optimal $\boldsymbol{\phi}$ satisfies

   $$\langle g_i(x) \rangle_{p(x)} - \frac{1}{Z(\boldsymbol{\phi})}\int_x g_i(x) e^{\boldsymbol{\phi}^{\mathsf{T}}\mathbf{g}(x)} = 0$$

   which is the desired result.

2. This is clearly true since a Gaussian is quadratic in the exponential.

   $$-\frac{1}{2\sigma^2}(x-\mu)^2 = -\frac{1}{2\sigma^2}x^2 + \frac{\mu}{\sigma^2}x + \mathsf{const}.$$

3. Optimally

$$\langle g_1(x) \rangle_p = \langle g_1(x) \rangle_q$$

$$\langle g_2(x) \rangle_p = \langle g_2(x) \rangle_q$$

and the result follows.

28.14 These results just follow directly. Consider

$$\frac{d\mathcal{B}}{db_i} = \frac{\partial \mathcal{B}}{\partial b_i} + \sum_j \frac{d\mathcal{B}}{d\alpha_j}\frac{\partial \alpha_j}{\partial b_i}$$

At the optimum of the bound,

$$\frac{d\mathcal{B}}{d\alpha_j} = 0$$

Hence at the optimum the mean is given by

$$\frac{d\mathcal{B}}{db_i} = \frac{\partial \mathcal{B}}{\partial b_i} = \tanh(\alpha_i) = \langle x_i \rangle_q$$

Similarly, for $i \neq j$, the covariance term is

$$\frac{d^2}{db_i db_j} \log Z(w, b) = \frac{d \tanh(\alpha_i)}{db_j} = 0$$

which is consistent with using a factorised (independent) approximation q.

28.15 Since $x_i \in \{0, 1\}$

$$\langle x_i x_j \rangle = 0 \times p(x_i = 0, x_j = 0) + 0 \times p(x_i = 0, x_j = 1) + 0 \times p(x_i = 1, x_j = 0) + 1 \times p(x_i = 1, x_j = 1) = p(x_i = 1, x_j = 1) = p(x_i = 1 | x_j = 1) p(x_j = 1$$

where the last step follows from Bayes' rule. We can use this 'algebraic perturbation' to obtain an improved approximation to $\langle x_i x_j \rangle_p$ by running

$$\langle x_i x_j \rangle_p \approx \langle x_i | x_j = 1 \rangle_q \langle x_j \rangle_q$$

For each j we now clamp x_j into state 1 and then run a factorised approximation on the remaining variables, giving $\langle x_i | x_j = 1 \rangle$. One repeats for all j. This method gives an improved approximation to $\langle x_i x_j \rangle$, but is more expensive since a separate mean field optimisation is required for each i.

28.16 Defining

$$E(\tilde{\phi}_{3\to2}(x_2)) = \log \tilde{Z} - \left\langle \log \tilde{\phi}_{3\to2}(x_2) \tilde{\phi}_{2\to3}(x_3) \right\rangle_{\tilde{p}_*(x_2, x_3)} = \log \tilde{Z} - \left\langle \log \tilde{\phi}_{3\to2}(x_2) \right\rangle_{\tilde{p}_*(x_2)} + \mathsf{const}.$$

and using

$$\tilde{Z} \propto \sum_{x_2} \tilde{\phi}_{1\to2}(x_2) \tilde{\phi}_{3\to2}(x_2) \sum_{x_3} \tilde{\phi}_{2\to3}(x_3) \tilde{\phi}_{4\to3}(x_3)$$

Then differentiating and equating to zero, we have at the optimum

$$\frac{\partial E}{\partial \tilde{\phi}_{3\to2}(x_2)} = \frac{1}{\tilde{Z}} \tilde{\phi}_{1\to2}(x_2) - \frac{\tilde{p}_*(x_2)}{\tilde{\phi}_{3\to2}(x_2)} = 0$$

rearranging

$$\tilde{p}_*(x_2) = \frac{1}{\tilde{Z}} \tilde{\phi}_{1\to2}(x_2) \tilde{\phi}_{3\to2}(x_2)$$

Since we are free to normalise the messages as we wish, we may define a marginal distribution

$$\tilde{p}_*(x_2) = \frac{\tilde{\phi}_{1\to2}(x_2) \tilde{\phi}_{3\to2}(x_2)}{\sum_{x_2} \tilde{\phi}_{1\to2}(x_2) \tilde{\phi}_{3\to2}(x_2)}$$

A similar derivation holds for $\tilde{p}_*(x_3)$.